Guide d'utilisation des programmes SAS pour la mesure des indicateurs du parcours BPCO à partir des données du SNDS (DCIR/PMSI)

Table des matières

1. Structures des bases de données

Pendant la phase de développement des indicateurs comprenant la programmation SAS, les libellés des indicateurs ont évolué. Voici la correspondance entre les anciens noms et les nouveaux noms des 7 indicateurs mesurés à partir du SNDS (DCIR/PMSI) ainsi que leurs acronymes :

- Diagnostic de BPCO recherché (DG) → Spirométrie ou EFR à visée diagnostique chez les patients à risque de BPCO (DG)
- Vaccin contre la grippe (VACG) → Vaccin contre la grippe chez les patients atteints de BPCO (VACG)
- Réalisation d'EFR ou d'une spirométrie annuelle (EFR_SPIRO) → Réalisation d'EFR ou d'une spirométrie annuelle chez les patients atteints de BPCO (EFR_SPIRO)
- Suivi médical dans les 7 jours après hospitalisation pour EABPCO (SMED_P) → Suivi médical dans les 7 jours après une hospitalisation pour exacerbation de BPCO des patients sortis à domicile (SMED_P)
- Suivi médical à distance après hospitalisation pour exacerbation (SMED_D) → Suivi par le pneumologue dans les 60 jours après une hospitalisation pour exacerbation de BPCO des patients sortis à domicile (SMED_D)
- Traitement après hospitalisation pour exacerbation (BDLA) → Traitement remboursé de BDLA délivré dans les 90 jours après une hospitalisation pour exacerbation de BPCO des patients sortis à domicile (BDLA)
- Réadaptation respiratoire après exacerbation (RR) → Soins de rééducation dans les 90 jours après une hospitalisation pour exacerbation de BPCO des patients sortis à domicile (RR)

La base de données ci-dessous recensera l'ensemble des individus appartenant à au moins une des populations cible d'un indicateur.

Indicateur concerné	Nom de la variable	Libellé
Informations générales sur les patients **T_INDI_BPCO_17**	**Ben_Idt_Ano**	Numéro d'individu
	BPCO_DG_CIBLE	Patient appartenant à la population cible BPCO_DG *(1:Oui | 0:Non)*
	BPCO_VACG_PROB_CIBLE	Patient appartenant à la population cible BPCO_VACG_PROB *(1:Oui | 0:Non)*
	BPCO_VACG_DIAG_CIBLE	Patient appartenant à la population cible BPCO_VACG_DIAG *(1:Oui | 0:Non)*
	BPCO_VACG_CIBLE	Patient appartenant à la population cible BPCO_VACG *(1:Oui | 0:Non)*
	BPCO_RR_EA_CIBLE	Patient appartenant à la population cible BPCO_RR_EA *(1:Oui | 0:Non)*
	BPCO_BDLA_EA_CIBLE	Patient appartenant à la population cible BPCO_BDLA_EA *(1:Oui | 0:Non)*
	BPCO_SMED_P_EA_CIBLE	Patient appartenant à la population cible BPCO_SMED_P_EA *(1:Oui | 0:Non)1*
	BPCO_SMED_D_EA_CIBLE	Patient appartenant à la population cible BPCO_SMED_D_EA *(1:Oui | 0:Non)*
	BPCO_EFR_SPIRO_CIBLE	Patient appartenant à la population cible BPCO_EFR_SPIRO *(1:Oui | 0:Non)*
	Age	Age du patient au 1er janvier 2017 (dernière situation connue au 1er janvier 2017 disponible dans référentiel bénéficiaire)

Indicateur concerné	Nom de la variable	Libellé
	Ben_Sex_Cod	Sexe du patient
	Ben_Res_Dpt	Département de résidence du patient
	Ben_Res_Com	Commune de résidence
	Rgm_Grg_Cod	Grand régime d'affiliation
	Ben_Dcd_Dte	Si patient décédé, date de décès (format Date9.). Si plusieurs dates de décès pour individu, prise en compte de la date de décès la plus récente
	ALD	Au moins une ALD active en 2017 *(1:Oui | 0:Non)*
	ALD_Chapitre_i	Bénéficiaire en ALD pour le chapitre i active en 2017 (22 classes) *(1:Oui | 0:Non)*
	ALD_BPCO_17	Patient ayant une ALD BPCO active en 2017 *(1:Oui | 0:Non)*
	Nb_Broncho_LDA	Nombre de délivrances remboursées de bronchodilatateur anticholinergique ou β2 longue durée d'action en 2017
	Charlson	Indice de Charlson en 2017
	Charlson_IDM	Composante de l'indice de Charlson – Infarctus du myocarde
	Charlson_Insuf_Cardiaque	Composante de l'indice de Charlson – Insuffisance cardiaque congestive
	Charlson_Malad_Vasculaire	Composante de l'indice de Charlson – Maladie vasculaire périphérique
	Charlson_Malad_CerebroVasc	Composante de l'indice de Charlson – Maladies cérébrovasculaires
	Charlson_Demence	Composante de l'indice de Charlson – Démence
	Charlson_Malad_Pulmonaires	Composante de l'indice de Charlson – Maladies pulmonaires chroniques
	Charlson_Malad_Rhumatisme	Composante de l'indice de Charlson – Maladies rhumatismales
	Charlson_Ulcere	Composante de l'indice de Charlson – Ulcère de l'estomac
	Charlson_Hepatite	Composante de l'indice de Charlson – Hépatite
	Charlson_Diabete_Compl	Composante de l'indice de Charlson – Diabète avec complications chroniques
	Charlson_Diabete_SansCompl	Composante de l'indice de Charlson – Diabète sans complications chroniques
	Charlson_Hemiplegie_Para	Composante de l'indice de Charlson – Hémiplégies ou paraplégies
	Charlson_Malad_Renale	Composante de l'indice de Charlson – Maladies rénales
	Charlson_Cancer	Composante de l'indice de Charlson – Cancers
	Charlson_Patho_Foie	Composante de l'indice de Charlson – Pathologies du foie de sévère à modérer
	Charlson_Tumeur	Composante de l'indice de Charlson – Tumeurs malignes métastasique
	Charlson_Malad_VIH	Composante de l'indice de Charlson – Maladies dues au VIH
	FDEP13	Indice de défavorisation sociale

Indicateur concerné	Nom de la variable	Libellé
	Quintile_POP	Quintiles de population générale en fonction du niveau de désavantage social de la commune
	Ben_Cmu_Top	Patient bénéficiant de la CMUc en 2017 *(1:Oui | 0:Non)*
	Nb_Sej_Exacerbation	Nombre de séjours pour exacerbation BPCO en 2017

Pour chaque indicateur défini, sera fourni :

- Une base de données contenant autant de lignes que de patients ou séjours correspondants aux critères d'inclusions et d'exclusion de la population cible. L'ensemble des variables à créer sont listées ci-dessous pour chacun des indicateurs.

Indicateur concerné	Nom de la variable	Libellé
Spirométrie ou EFR à visée diagnostique chez les patients à risque de BPCO *une observation par patient de la population cible* **T_INDI_BPCO_DG_17**	**Ben_Idt_Ano**	Numéro d'individu
	Date_index_broncho	Date de la première délivrance de bronchodilatateur anticholinergique longue durée d'action ou β2 longue durée d'action en 2017 (format Date9.)
	Date_index_antibio_memejour	Date de la première délivrance remboursée d'antibiothérapie en 2017 associée le même jour d'une délivrance remboursée de bronchodilatateur anticholinergique courte durée d'action ou un β2 courte durée d'action. Cette délivrance (antibio + broncho) doit être précédée dans les 365 jours d'au moins une délivrance remboursée d'antibiothérapie pour infections respiratoires (format Date9.)
	Date_index_antibio_365jour	Date de la première délivrance remboursée d'antibiothérapie en 2017 précédée dans les 365 jours d'au moins une délivrance remboursée d'antibiothérapie pour infections respiratoires associée le même jour d'une délivrance remboursée de bronchodilatateur anticholinergique courte durée d'action ou un β2 courte durée d'action (format Date9.)
	Date_index_substitut	Date de la première délivrance de substituts nicotiniques (remboursable ou forfait) ou d'un traitement d'arrêt du tabac en 2017 (format Date9.)
	BPCO_DG_Date_index	Date index retenue (format Date9.)
	ATB_Inf_Resp_3	Patient ayant eu au moins une délivrance remboursée d'antibiothérapie pour infections respiratoires en 2017 précédée d'au moins 2 délivrances remboursées d'antibiothérapie pour infections respiratoires dans les 365 jours *(1:Oui | 0:Non)*
	ATB_Inf_Resp_2	Patient ayant eu au moins une délivrance remboursée d'antibiothérapie pour infections respiratoires en 2017 précédée d'au moins une délivrance remboursée d'antibiothérapie pour infections respiratoires dans les 365 jours *(1:Oui | 0:Non)*

Indicateur concerné	Nom de la variable	Libellé
	ATB_Inf_Meme_Jour	Patients ayant eu au moins une délivrance remboursée d'antibiothérapie pour infections respiratoires en 2017 associée le même jour d'une délivrance remboursée de bronchodilatateur anticholinergique courte durée d'action ou un β2 courte durée d'action. Cette délivrance (antibiothérapie + (broncho ou β2 courte durée)) doit être précédée dans les 365 jours d'au moins une délivrance remboursée d'antibiothérapie pour infections respiratoires (1:Oui | 0:Non)
	ATB_Inf_365_Jour	Patients ayant eu au moins une délivrance remboursée d'antibiothérapie pour infections respiratoires en 2017 Cette délivrance doit être précédée dans les 365 jours d'au moins une délivrance remboursée d'antibiothérapie pour infections respiratoires qui est associée le même jour à une délivrance remboursée de bronchodilatateur anticholinergique courte durée d'action ou un β2 courte durée d'action (1:Oui | 0:Non)
	Broncho_CDA	Patient ayant eu au moins une délivrance remboursée en 2017 d'un bronchodilatateur anticholinergique ou β2 courte durée d'action délivrée le même jour que la cure d'antibiothérapie *(1:Oui | 0:Non)*
	Nb_ATB_Inf_Resp_AV365J	Nombre de délivrances remboursées d'antibiothérapie pour infections respiratoires dans les 365 jours précédant la date index
	Nb_Broncho_CDA_AV365j	Nombre de délivrances remboursées de bronchodilatateur anticholinergique ou β2 courte durée d'action dans les 365 jours précédant la date index
	Broncho_LDA	Patient ayant eu au moins une délivrance remboursée en 2017 d'un bronchodilatateur anticholinergique longue durée d'action ou β2 longue durée d'action *(1:Oui | 0:Non)*
	Subs_Nico	Patient ayant eu au moins une délivrance remboursée de substituts nicotiniques en 2017 *(1:Oui | 0:Non)*

Indicateur concerné	Nom de la variable	Libellé
	BPCO_DG_CIBLE	Patient correspondant aux critères d'inclusion et d'exclusion de la population cible *(1:Oui | 0:Non)* – *Toujours égal à 1*
	Nb_ATB_Inf_Resp	Nombre de délivrances remboursées d'antibiothérapie pour infections respiratoires en 2017
	Nb_Broncho_CDA	Nombre de délivrances remboursées de bronchodilatateur anticholinergique ou β2 courte durée d'action en 2017
	Nb_Subs_Nico	Nombre de délivrances remboursées de substituts nicotiniques en 2017
	BPCO_DG_OBS	Patient de la population cible pour lequel une spirométrie ou des EFR ont été réalisés dans les 365 jours précédant la date index ou les 365 jours suivant la date index *(1:Oui | 0:Non)*
	Spiro	Spirométrie réalisée dans les 365 jours avant ou suivant la date index
	Nb_Spiro	Nombre de spirométries réalisées dans les 365 jours avant ou suivant la date index
	Date_Spiro	Date de la réalisation de la première spirométrie dans les 365 jours avant ou suivant la date index (format Date9.)
	Lieu_Spiro	Lieu de réalisation de la première spirométrie avant ou suivant la date index *(1 :Ville | 2:Etablissement de santé)*
	EFR	EFR réalisée dans les 365 jours avant ou suivant la date index
	Nb_EFR	Nombre d'EFR réalisées dans les 365 jours avant ou suivant la date index
	Date_EFR	Date de la réalisation de la première EFR dans les 365 jours avant ou suivant la date index (format Date9.)
	Lieu_EFR	Lieu de réalisation de la première EFR dans les 365 jours avant ou suivant la date index *(1 :Ville | 2:Etablissement de santé)*
	Spiro_ Av_365	Spirométrie réalisée dans les 365 jours avant la date index (incluse)
	Nb_Spiro_ Av_365	Nombre de spirométries réalisées dans les 365 jours avant la date index (incluse)
	Date_Spiro_ Av_365	Date de la réalisation de la première spirométrie dans les 365 jours avant la date index (incluse) (format Date9.)
	Lieu_Spiro_ Av_365	Lieu de réalisation de la première spirométrie avant la date index (incluse) *(1 :Ville | 2:Etablissement de santé)*
	EFR_ Av_365	EFR réalisée dans les 365 jours avant la date index (incluse)

Indicateur concerné	Nom de la variable	Libellé
	Nb_EFR_ Av_365	Nombre d'EFR réalisées dans les 365 jours avant la date index (incluse)
	Date_EFR_ Av_365	Date de la réalisation de la première EFR dans les 365 jours avant la date index (incluse) (format Date9.)
	Lieu_EFR_ Av_365	Lieu de réalisation de la première EFR dans les 365 jours avant la date index (incluse) *(1 :Ville | 2:Etablissement de santé)*
	Spiro_AP_365	Spirométrie réalisée dans les 365 jours suivant la date index (exclue)
	Nb_Spiro_AP_365	Nombre de spirométries réalisées dans les 365 jours suivant la date index (exclue)
	Date_Spiro_AP_365	Date de la réalisation de la première spirométrie dans les 365 jours suivant la date index (exclue) (format Date9.)
	Lieu_Spiro_AP_365	Lieu de réalisation de la première spirométrie suivant la date index (exclue) *(1 :Ville | 2:Etablissement de santé)*
	EFR_AP_365	EFR réalisée dans les 365 jours suivant la date index (exclue)
	Nb_EFR_AP_365	Nombre d'EFR réalisées dans les 365 jours suivant la date index (exclue)
	Date_EFR_AP_365	Date de la réalisation de la première EFR dans les 365 jours suivant la date index (exclue) (format Date9.)
	Lieu_EFR_AP_365	Lieu de réalisation de la première EFR dans les 365 jours suivant la date index (exclue) *(1 :Ville | 2:Etablissement de santé)*
	Pneumothorax	Patients ayant eu un pneumothorax dans les 365 jours suivant la date index ou dans les 365 jours précédant la date index (1 :Oui / 0 :Non)
	Pneumothorax_AV_365	Patients ayant eu un pneumothorax dans les 365 jours avant la date index (incluse) (1 :Oui / 0 :Non)
	Pneumothorax_AP_365	Patients ayant eu un pneumothorax dans les 365 jours suivant la date index (exclue) (1 :Oui / 0 :Non)
	Infarctus	Patients ayant eu un diagnostic d'infarctus dans les 365 jours suivant la date index ou dans les 365 jours précédant la date index (1 :Oui / 0 :Non)
	Infarctus_AV_365	Patients ayant eu un diagnostic d'infarctus dans les 365 jours avant la date index (incluse) (1 :Oui / 0 :Non)

	Infarctus_AP_365	Patients ayant eu un diagnostic d'infarctus dans les 365 jours suivant la date index (exclue) (1 :Oui / 0 :Non)

Indicateur concerné	Nom de la variable	Libellé
Vaccin contre la grippe chez les patients atteints de BPCO *une observation par patient de la population cible* **T_INDI_BPCO_VACG_17**	**Ben_Idt_Ano**	Numéro d'individu
	BPCO_VACG_PROB_CIBLE	Patient correspondant aux critères d'inclusion et d'exclusion de la population cible probable [1]– BPCO probable *(1:Oui | 0:Non)*
	Diag_BPCO	Patient ayant eu au moins un diagnostic de BPCO codé lors d'un séjour hospitalier en MCO terminé entre le 1er janvier 2016 et le 31 aout 2017 ou lors d'un séjour hospitalier en SSR, HAD entre le 1er janvier 2016 et le 31 aout 2017 *(1:Oui | 0:Non)*
	ALD_BPCO_avSep	Patient en ALD BPCO active le 1er septembre 2017 *(1:Oui | 0:Non)*
	BPCO_diagnostique	Patient diagnostiqué BPCO *(1:Oui | 0:Non)*
	ALD_BPCO_Date	Date de début de mise en ALD BPCO (format Date9.)
	BPCO_VACG_DIAG_CIBLE	Patient correspondant aux critères d'inclusion et d'exclusion de la population cible diagnostiquée – BPCO diagnostiqués *(1:Oui | 0:Non)*
	BPCO_VACG_CIBLE	Patient correspondant aux critères d'inclusion et d'exclusion de la population cible *(1:Oui | 0:Non)*
	Nb_Sej_Diag_BPCO_MCO	Nombre de séjours avec un diagnostic BPCO terminés entre le 1er janvier 2016 et le 31 août 2017 en MCO
	Nb_Sej_Diag_BPCO_SSR	Nombre de séjours avec un diagnostic BPCO entre le 1er janvier 2016 et le 31 août 2017 en SSR
	Nb_Sej_Diag_BPCO_HAD	Nombre de séjours avec un diagnostic BPCO entre le 1er janvier 2016 et le 31 août 2017 en HAD
	BPCO_VACG_PROB_OBS	Patient de la population cible probable ayant eu une délivrance remboursée du vaccin contre la grippe entre le 1er septembre 2017 et le 31 mai 2018 *(1:Oui | 0:Non)*
	BPCO_VACG_DIAG_OBS	Patient de la population cible diagnostiquée ayant eu une délivrance remboursée du vaccin contre la grippe entre le 1er septembre 2017 et le 31 mai 2018 *(1:Oui | 0:Non)*
	BPCO_VACG_OBS	Patient de la population cible ayant eu une délivrance remboursée du vaccin contre la grippe entre le 1er septembre 2017 et le 31 mai 2018 *(1:Oui | 0:Non)*
	Date_VACG	Date de la délivrance remboursée du vaccin contre la grippe entre le 1er septembre 2017 et le 31 mai 2018 (format Date9.)
Indicateur concerné	**Nom de la variable**	**Libellé**

[1] La population probable correspond aux patients avec un traitement de référence de la BPCO

Indicateur concerné	Nom de la variable	Libellé
Réalisation d'EFR ou d'une spirométrie annuelle chez les patients atteints de BPCO *une observation par patient de la population cible* **T_INDI_BPCO_EFR_SPIRO_17**	**Ben_Idt_Ano**	Numéro d'individu
	BPCO_Probable	Patient ayant une BPCO probable *(1:Oui | 0:Non)*
	Diag_BPCO	Patient ayant eu au moins un diagnostic de BPCO codé lors d'un séjour hospitalier en MCO terminé en 2017 ou lors d'un séjour hospitalier en, SSR, HAD en 2017 *(1:Oui | 0:Non)*
	BPCO_Diagnostique	Patient diagnostiqué BPCO *(1:Oui | 0:Non)*
	BPCO_EFR_SPIRO_CIBLE	Patient correspondant aux critères d'inclusion et d'exclusion de la population cible *(1:Oui | 0:Non) - Toujours égal à 1*
	ALD_BPCO_Date	Date de début de mise en ALD BPCO (format Date9.)
	Nb_Sej_Diag_BPCO_MCO	Nombre de séjours avec un diagnostic BPCO terminés en 2017 en MCO
	Nb_Sej_Diag_BPCO_SSR	Nombre de séjours avec un diagnostic BPCO en 2017 en SSR
	Nb_Sej_Diag_BPCO_HAD	Nombre de séjours avec un diagnostic BPCO en 2017 en HAD
	BPCO_EFR_SPIRO_OBS	Patient de la population cible ayant bénéficiés d'une exploration fonctionnelle respiratoire ou d'une spirométrie entre le 1er janvier 2017 et le 28 février 2018 *(1:Oui | 0:Non)*
	Spiro	Spirométrie réalisée entre le 1er janvier 2017 et le 28 février 2018
	Nb_Spiro	Nombre de spirométries réalisées entre le 1er janvier 2017 et le 28 février 2018
	Date_Spiro	Date de la réalisation de la première spirométrie entre le 1er janvier 2017 et le 28 février 2018 (format Date9.)
	Lieu_Spiro	Lieu de réalisation de la première spirométrie *(1 :Ville | 2:Etablissement de santé)*
	EFR	EFR réalisé entre le 1er janvier 2017 et le 28 février 2018
	Nb_EFR	Nombre d'EFR réalisés entre le 1er janvier 2017 et le 28 février 2018
	Date_EFR	Date de la première réalisation d'EFR entre le 1er janvier 2017 et le 28 février 2018 (format Date9.)
	Lieu_EFR	Lieu de réalisation du première EFR entre le 1er janvier 2017 et le 28 février 2018 *(1 :Ville | 2:Etablissement de santé)*
	Pneumothorax	Patients ayant eu un pneumothorax entre le 1er janvier 2017 et le 28 février 2018 (1:Oui / 0:Non)

Indicateur concerné	Nom de la variable	Libellé
	Infarctus	Patients ayant un diagnostic d'infarctus entre le 1er janvier 2017 et le 28 février 2018
	Oxygenotherapie	Patients ayant eu une délivrance remboursée d'un traitement par oxygénothérapie entre le 1er janvier 2017 et le 28 février 2018
	VNI	Patients ayant eu une délivrance remboursée d'un traitement par VNI entre le 1er janvier 2017 et le 28 février 2018

Indicateur concerné	Nom de la variable	Libellé
Suivi médical dans les 7 jours après une hospitalisation pour exacerbation de BPCO des patients sortis à domicile *une observation par séjour de la population cible* **T_INDI_BPCO_SMED_P_EA_17**	**Ben_Idt_Ano**	Numéro d'individu *(un même individu peut avoir plusieurs séjours cibles)*
	BPCO_SMED_P_EA_Date_index	Date index (format Date9.)
	BPCO_SMED_P_EA_Sej_Index	Identifiant du séjour index (ETA_NUM|RSA_NUM|Annee_PMSI)
	Finess_PMSI	Finess PMSI ayant réalisé le séjour index
	Finess_GEO	Finess géographique du premier RUM
	Dp	Diagnostic principal du séjour index
	Dp_classe	Diagnostic principal du séjour index *(1:BPCO | 2:pneumopathie | 3:insuffisance respiratoire aigüe|4:grippe | 5:embolie pulmonaire | 6:insuffisance cardiaque aigue | 7:pneumothorax)*
	DA_BPCO	BPCO codé en diagnostic secondaire du séjour index *(1:Oui | 0:Non)*
	GHM	GHM du séjour index
	BPCO_SMED_P_EA_CIBLE	Séjour correspondant aux critères d'inclusion et d'exclusion de la population cible *(1:Oui | 0:Non)* - *Toujours égal à 1*
	Nb_Sej_EA_BPCO_DP_MCO	Nombre de séjours MCO d'exacerbation de BPCO codé en DP terminés en 2017
	Nb_Sej_EA_BPCO_DAS_MCO	Nombre de séjours MCO d'exacerbation de BPCO codé en DAS terminés en 2017
	BPCO_SMED_P_EA_OBS	Séjour de la population cible pour lequel le patient a bénéficié d'un suivi dans les 7 jours après la sortie de l'hôpital *(1:Oui | 0:Non)*
	Contact_Med_7j	Séjour de la population cible pour lequel le patient a eu un contact médecin traitant ou MG dans les 7 jours suivant la date index *(1:Oui | 0:Non)*
	Nb_Contact_Med_7j	Nombre de contacts médecin traitant ou MG dans les 7 jours suivant la date index
	Date_Contact_Med_180j	Date du premier contact médecin traitant ou MG dans les 180 jours suivant la date index (format Date9.)
	Contact_Pneumo_7j	Séjour de la population cible pour lequel le patient a eu un contact pneumologue dans les 7 jours suivant la date index *(1:Oui | 0:Non)*
	Nb_Contact_Pneumo_7j	Nombre de contacts pneumologue dans les 7 jours suivant la date index
	Date_Contact_Pneumo_180j	Date du premier contact pneumologue dans les 180 jours suivant la date index (format Date9.)

Indicateur concerné	Nom de la variable	Libellé
Suivi par le pneumologue dans les 60 jours après une hospitalisation pour exacerbation de BPCO des patients sortis à domicile *une observation par séjour de la population cible* **T_INDI_BPCO_SMED_D_EA_17**	**Ben_Idt_Ano**	Numéro d'individu *(un même individu peut avoir plusieurs séjours cibles)*
	BPCO_SMED_D_EA_Date_index	Date index (format Date9.)
	BPCO_SMED_D_EA_Sej_Index	Identifiant du séjour index (ETA_NUM|RSA_NUM|Annee_PMSI)
	Finess_PMSI	Finess PMSI ayant réalisé le séjour index
	Finess_GEO	Finess géographique du premier RUM
	Dp	Diagnostic principal du séjour index
	Dp_classe	Diagnostic principal du séjour index *(1:BPCO | 2:pneumopathie | 3:insuffisance respiratoire aigüe|4:grippe | 5:embolie pulmonaire | 6:insuffisance cardiaque aigue | 7:pneumothorax)*
	DA_BPCO	BPCO codé en diagnostic secondaire du séjour index *(1:Oui | 0:Non)*
	GHM	GHM du séjour index
	BPCO_SMED_D_EA_CIBLE	Séjour correspondant aux critères d'inclusion et d'exclusion de la population cible *(1:Oui | 0:Non)* - *Toujours égal à 1*
	Nb_Sej_EA_BPCO_DP_MCO	Nombre de séjours MCO d'exacerbation de BPCO codé en DP terminés en 2017
	Nb_Sej_EA_BPCO_DAS_MCO	Nombre de séjours MCO d'exacerbation de BPCO codé en DAS terminés en 2017
	BPCO_SMED_D_EA_OBS	Séjour de la population cible pour lequel le patient a bénéficié d'un suivi à distance après la sortie de l'hôpital *(1:Oui | 0:Non)*
	Nb_Contact_Pneumo_60j	Nombre de contacts pneumologue dans les 60 jours suivant la date index
	Date_Contact_Pneumo_180j	Date du premier contact pneumologue dans les 180 jours suivant la date index (format Date9.)

Indicateur concerné	Nom de la variable	Libellé
Traitement remboursé de BDLA délivré dans les 90 jours après une hospitalisation pour exacerbation de BPCO des patients sortis à domicile *une observation par séjour de la population cible* **T_INDI_BPCO_BDLA_EA_17**	**Ben_Idt_Ano**	Numéro d'individu
	BPCO_BDLA_EA_Date_index	Date index (format Date9.)
	BPCO_BDLA_EA_Sej_Index	Identifiant du séjour index (ETA_NUM|RSA_NUM|Annee_PMSI)
	Finess_PMSI	Finess PMSI ayant réalisé le séjour index
	Finess_GEO	Finess géographique du premier RUM
	Dp	Diagnostic principal du séjour index
	Dp_classe	Diagnostic principal du séjour index *(1:BPCO | 2:pneumopathie | 3:insuffisance respiratoire aigüe|4:grippe | 5:embolie pulmonaire | 6:insuffisance cardiaque aigue | 7:pneumothorax)*
	DA_BPCO	BPCO codé en diagnostic secondaire du séjour index *(1:Oui | 0:Non)*
	GHM	GHM du séjour index
	BPCO_BDLA_EA_CIBLE	Séjour correspondant aux critères d'inclusion et d'exclusion de la population cible *(1:Oui | 0:Non) - Toujours égal à 1*
	BPCO_BDLA_EA_OBS	Séjours de la population cible pour lesquels le patient a eu une délivrance remboursée de traitement bronchodilatateur anticholinergique ou bronchodilatateur β2 longue durée d'action dans les 90 jours suivant la date index *(1:Oui | 0:Non)*
	Nb_Sej_EA_BPCO_DP_MCO	Nombre de séjours MCO d'exacerbation de BPCO codé en DP terminés en 2017
	Nb_Sej_EA_BPCO_DAS_MCO	Nombre de séjours MCO d'exacerbation de BPCO codé en DAS terminés en 2017
	Nb_Broncho_LDA_90j	Nombre de délivrances remboursées de bronchodilatateur anticholinergique ou β2 longue durée d'action dans les 90 jours suivant la date index
	Date_Broncho_LDA_180j	Date de la première délivrance remboursée de bronchodilatateur anticholinergique ou β2 longue durée d'action dans les 180 jours suivant la date index (format Date9.)
	Broncho_LDA_90javant	Patient ayant eu au moins une délivrance remboursée d'un bronchodilatateur anticholinergique ou β2 longue durée d'action dans les 90 jours précédant la date index *(1:Oui | 0:Non)*

Indicateur concerné	Nom de la variable	Libellé
Soins de rééducation dans les 90 jours après une hospitalisation pour exacerbation de BPCO des patients sortis à domicile*une observation par patient de la population cible* **T_INDI_BPCO_RR_EA_17**	**Ben_Idt_Ano**	Numéro d'individu
	BPCO_RR_EA_Date_index	Date index (format Date9.)
	BPCO_RR_EA_Sej_Index	Identifiant du séjour index (ETA_NUM|RSA_NUM|Annee_PMSI)
	Finess_PMSI	Finess PMSI ayant réalisé le séjour index
	Finess_GEO	Finess géographique du premier RUM
	Dp	Diagnostic principal du séjour index (exacerbation de BPCO, insuffisance respiratoire aigüe ou pneumopathie lobaire)
	Dp_classe	Diagnostic principal du séjour *index (1:BPCO | 2:pneumopathie | 3:insuffisance respiratoire aigüe|4:grippe | 5:embolie pulmonaire | 6:insuffisance cardiaque aigue | 7:pneumothorax)*
	DA_BPCO	BPCO codé en diagnostic secondaire du séjour index *(1:Oui | 0:Non)*
	GHM	GHM du séjour index
	BPCO_RR_EA_CIBLE	Patient correspondant aux critères d'inclusion et d'exclusion de la population cible *(1:Oui | 0:Non) - Toujours égal à 1*
	Sej_SSR	Patient de la population cible ayant eu un séjour en SSR dans les 90 jours suivant la date index (*1:Oui | 0:Non)*
	Sej_CM04	Patient ayant eu un séjour en SSR dans la CM 04 dans les 90 jours suivant la date index (*1:Oui | 0:Non)*
	Sej_UM54	Patient de la population cible ayant eu un séjour en SSR et pris en charge dans une unité médicale spécialisée « Affections respiratoires » dans les 90 jours suivant la date index (*1:Oui | 0:Non)*
	SEJ_CM04_UM54	Patient ayant eu un séjour en SSR dans la CM 04 et pris en charge dans une unité médicale spécialisée « Affections respiratoires » dans les 90 jours suivant la date index (*1:Oui | 0:Non)*
	Sej_UM50	Patient de la population cible ayant eu un séjour en SSR et pris en charge dans une unité médicale spécialisée « « Soins de suite et de réadaptation indifférenciés ou polyvalents » dans les 90 jours suivant la date index (*1:Oui | 0:Non)*
	SEJ_CM04_UM50	Patient ayant eu un séjour en SSR dans la CM 04 et pris en charge dans une unité médicale spécialisée « Soins de suite et de réadaptation indifférenciés ou polyvalents » dans les 90 jours suivant la date index (*1:Oui | 0:Non)*

	Sej_CSAR_CCAM	Patient de la population cible ayant eu un séjour avec au moins un acte CSARR ou CCAM de la liste de réadaptation respiratoire dans les 90 jours suivant la date index (*1:Oui | 0:Non)*
	Sej_CM04_CSAR_CCAM	Patient de la population cible ayant eu un séjour en SSR dans la CM04 avec au moins un acte CSARR ou CCAM de la liste de réadaptation respiratoire dans les 90 jours suivant la date index (*1:Oui | 0:Non)*
	SEJ_UM50_CSAR_CCAM	Patient ayant eu un séjour en SSR dans une unité médicale « Soins de suite et de réadaptation indifférenciés ou polyvalents » avec au moins un acte CSARR ou CCAM de la liste de réadaptation respiratoire dans les 90 jours suivant la date index (*1:Oui | 0:Non)*
	SEJ_CM04_UM50_CSAR_CCAM	Patient ayant eu un séjour en SSR dans la CM 04 et dans une unité médicale « Soins de suite et de réadaptation indifférenciés ou polyvalents » avec au moins un acte CSARR ou CCAM de la liste de réadaptation respiratoire dans les 90 jours suivant la date index (*1:Oui | 0:Non)*
	KINE_AMK_AMC_28	Réadaptation respiratoire kinésithérapique respiratoire pour les patients atteints de handicap respiratoire dans les 90 jours suivant la date index (*1:Oui | 0:Non)*
	KINE_AMK_AMC_20	Réadaptation respiratoire kinésithérapique respiratoire pour les patients atteints de handicap respiratoire chronique en prise en charge de groupe de 2 à 4 personnes avec rééducation respiratoire en individuel dans les 90 jours suivant la date index (*1:Oui | 0:Non)*
	KINE_AMK_AMC_AMS_9_5_8	Réadaptation respiratoire codé en coefficient 9.5 avec des coefficients 8 (délai maximum entre les 2 actes <= 30 jours avant ou après le code 9,5 mais le date d'exécution ne doit pas être antérieur à la date index et les actes doivent être réalisés tous les 2 soit en ville, soit en MCO, soit en SSR) dans les 90 jours suivant la date index (*1:Oui | 0:Non)*
	KINE_AMK_AMC_AMS_9_5_4	Réadaptation respiratoire codé en coefficient 9.5 avec des coefficients 8/2 (réalisés le même jour et les actes doivent être réalisé tous les 2 soit en ville, soit en MCO, soit en SSR) dans les

		90 jours suivant la date index (*1:Oui | 0:Non)*
	KINE_AMK_AMC_AMS_8	Réadaptation respiratoire kinésithérapique respiratoire codé en coefficient 8 dans les 90 jours suivant la date index (*1:Oui | 0:Non)*
	KINE_AMS_AMC_AMK_13_5	Rééducation respiratoire et motrice codé avec le coefficient 13.5 dans les 90 jours suivant la date index (1:Oui | 0:Non)
	KINE_BPC	Rééducation respiratoire et motrice codé avec l'acte BPC dans les 90 jours suivant la date index (1:Oui | 0:Non)
	BPCO_RR_EA_OBS	Patient de la population cible ayant bénéficié d'une réadaptation respiratoire dans les 90 jours suivant la date index (1:Oui | 0:Non)
	Nb_Sej_EA_BPCO_DP_MCO	Nombre de séjours MCO d'exacerbation de BPCO codé en DP terminés en 2017
	Nb_Sej_EA_BPCO_DAS_MCO	Nombre de séjours MCO d'exacerbation de BPCO codé en DAS terminés en 2017
	Date_RR	Date de réalisation de la réadaptation respiratoire dans les 90 jours suivant la sortie du séjour index (format Date9.) *Remarque :* *Si un séjour SSR a débuté dans cette période, Date_RR est égale à la date de début du 1er séjour SSR dans cette période* *Si un séjour SSR n'a pas débuté dans cette période, Date_RR est égale à la date de réalisation du 1er acte de kiné dans cette période*
	Lieu_RR	Lieu de réalisation de la réadaptation respiratoire dans les 90 jours suivant la sortie du séjour index *(1:Ville | 2 :SSR | 3 :MCO)* *Si un séjour SSR a débuté dans cette période, Lieu_RR est égal à SSR* *Si un séjour SSR n'a pas débuté dans cette période, Lieu_RR est égal à Ville/MCO/SSR, en fonction de la source d'où est extrait le 1er acte de kiné réalisé dans cette période*
	Date_RR_prec	Date de la dernière RR dans les 365 jours précédant la date d'entrée du séjour index (format Date9.) *Si un séjour SSR a débuté dans cette période, Date_RR_prec est égale à la date de début du dernier séjour SSR dans cette période* *Si un séjour SSR n'a pas eu débuté dans cette période, Date_RR_prec est égale*

		à la date de réalisation du dernier acte de kiné dans cette période
	Polyarthrite_Rhuma	Patient pris en charge pour polyarthrite rhumatoïdes et maladies apparentées *(1:Oui | 0:Non)* [2]
	Spondylarthirte_Anky	Patient pris en charge pour spondylarthrite ankylosante et maladies apparentées *(1:Oui | 0:Non)*[2]
	Prothese_Hanche	Patient ayant eu au moins un acte CCAM de prise en charge d'une prothèse de hanche *(1:Oui | 0:Non)* [2]
	Prothese_Genou	Patient ayant eu au moins un acte CCAM de prise en charge d'une prothèse de genou *(1:Oui | 0:Non)* [2]
	Chir_Rachis	Patient ayant eu au moins un acte CCAM de prise en charge de la chirurgie du rachis *(1:Oui | 0:Non)* [2]
	Pontage_Coro	Patient ayant eu au moins un diagnostic de pontage coronarien *(1:Oui | 0:Non)* [2]
	Rempla_Valve_card	Patient ayant eu au moins un diagnostic de remplacement de valve cardiaque *(1:Oui | 0:Non)* [2]
	Stent	Patient ayant eu au moins un diagnostic de prothèse vasculaire (stents) *(1:Oui | 0:Non)* [2]
	Radiotherapie	Patient ayant eu au moins une séance de radiothérapie *(1:Oui | 0:Non)* [2]
	Chimiotherapie	Patients ayant eu au moins une séance de chimiothérapie *(1:Oui | 0:Non)* [2]

[2] *Se référer aux spécifications techniques en annexe*

2. Description des programmes

2.1 Définition des librairies, formats et des macros-programmes

Les scripts de cette première partie doivent être exécutés à chaque ouverture de session lorsque nous souhaitons exécuter les programmes SASDéfinition des librairies, formats et des macros-programmes (*Définition des librairies, formats et des macros-programmes*).

Les scripts de cette première partie permettent de :

- Initialiser les options de la sessions SAS ouverte
- Initialiser les 6 macro-variables 'Année :
    - *annee_N : Année du résultat de l'indicateur (Exemple : 2017),*
    - *an_N : Année du résultat de l'indicateur sans le siècle (Exemple : 17),*
    - *annee_4N : Année du résultat de l'indicateur sans le siècle moins 4 ans (Exemple : 13 si les résultats de l'indicateur sont mesurés sur l'année 2017),,*
    - *annee_2N : Année du résultat de l'indicateur sans le siècle moins 2 ans (Exemple : 15 si les résultats de l'indicateur sont mesurés sur l'année 2017),*
    - *annee_1N : Année du résultat de l'indicateur sans le siècle moins 1 an (Exemple : 16 si les résultats de l'indicateur sont mesurés sur l'année 2017,*
    - *annee_N1 : Année du résultat de l'indicateur sans le siècle plus 1 an (Exemple : 18 si les résultats de l'indicateur sont mesurés sur l'année 2017.*

La création de ces 6 macro-variables permettent de mesurer les indicateurs BPCO quel que soit l'année sans que nous ayons à modifier aucun programme.
Le choix de l'année pour laquelle la mesure des indicateurs est réalisée est effectuée en modifiant la valeur de la macro-variable *annee_N.*

- Initialiser la macro-variable version_carto. Actuellement, plusieurs versions de cartographies des pathologies existent. Chaque année, une nouvelle version est mise à disposition. Le choix de la version de cartographie que nous souhaitons utiliser est fait à partir de cette macro-variable.
- Initialiser les deux macro-variables cle_inc1 et cle_inc2. Les valeurs dans ces deux macro-variables décrivent des identifiants dans les tables du PMSI avec des clés de chainage incorrecte. Les valeurs de ces deux macro-variables peuvent être modifiées lorsqu'un nouveau process du changement de la clé de cryptage du NIR sera réalisé (tous les deux ans).
- Initialiser les librairies suivantes : les chemins de chacune des librairies sont à modifier en fonction de votre espace travail sur le portail SNDS (DCIR/PMSI)
    - TRAVAIL : Les tables intermédiaires permettant de mesurer les indicateurs et les informations complémentaires sont stockées dans cette librairie.
    - POP : La table indicateurs_ *an_N* est stockée dans cette librairie et permet d'avoir tous les identifiants disponibles dans IR_BEN_R des individus pré-sélectionnés pour un des indicateurs du projet (Exemple : Pour l'indicateur Diagnostic de BPCO, seuls les patients avec au moins 3 BDLA remboursées sont sélectionnées). La création de cette table permet de réduire la taille des bases de travail créées par la suite.
    - RES : Cette librairie contient les tables suivantes :
        - La table T_Indi_BPCO_ *an_N* avec les informations générales sur les patients ; 1 observation par individu appartenant à la population cible d'au moins un

indicateur. Dans cette table, les critères d'exclusion ont été appliqués au contraire de la table indicateurs_*an_N,* ce qui explique qu'il y ait moins de patients la table T_Indi_BPCO_ an_N.

- Sept tables décrivant chacune un indicateur :
  - La table T_Indi_BPCO_DG_*an_N :* 1 observation par individu de la population cible de l'indicateur 'Diagnostic de BPCO recherché'
  - La table T_Indi_BPCO_VACG_*an_N :* 1 observation par individu de la population cible de l'indicateur 'Vaccin contre la grippe'
  - La table T_Indi_BPCO_RR_EA_*an_N :* 1 observation par individu de la population cible de l'indicateur 'Réadaptation respiratoire après hospitalisation pour exacerbation'
  - La table T_Indi_BPCO_BDLA_EA_*an_N :* 1 observation par séjour de la population cible de l'indicateur 'Traitement après hospitalisation pour exacerbation'
  - La table T_Indi_BPCO_SMED_P_EA_*an_N :* 1 observation par séjour de la population cible de l'indicateur 'Suivi médicale précoce après hospitalisation pour exacerbation'
  - La table T_Indi_BPCO_SMED_D_EA_*an_N :* 1 observation par séjour de la population cible de l'indicateur 'Suivi médicale à distance après hospitalisation pour exacerbation'
  - La table T_Indi_BPCO_EFR_SPIRO_*an_N :* 1 observation par individu de la population cible de l'indicateur 'Réalisation d'une EFR ou d'une spirométrie annuelle'

- FLOWCH : Cette librairie contient sept tables décrivant chacune le flowchart d'un indicateur :
  - La table Flow_chart_BPCO_DG_*an_N :* le flow-chart de l'indicateur 'Diagnostic de BPCO recherché' : Effectifs de la population d'étude/cible, des critères d'exclusion, le taux de l'indicateur
  - La table Flow_chart_BPCO_VACG_*an_N :* le flow-chart de l'indicateur 'Vaccin contre la grippe ' : Effectifs de la population d'étude/cible, des critères d'exclusion, le taux de l'indicateur
  - La table Flow_chart_BPCO_RR_EA_*an_N :* le flow-chart de l'indicateur 'Réadaptation respiratoire après hospitalisation pour exacerbation' : Effectifs de la population d'étude/cible, des critères d'exclusion, le taux de l'indicateur
  - La table Flow_chart_BPCO_BDLA_EA_*an_N :* le flow-chart de l'indicateur 'Traitement après hospitalisation pour exacerbation' : Effectifs de la population d'étude/cible, des critères d'exclusion, le taux de l'indicateur
  - La table Flow_chart_BPCO_SMED_P_EA_*an_N :* le flow-chart l'indicateur 'Suivi médicale précoce après hospitalisation pour exacerbation' : Effectifs de la population d'étude/cible, des critères d'exclusion, le taux de l'indicateur
  - La table Flow_chart_BPCO_SMED_D_EA_*an_N :* le flow-chart de l'indicateur 'Suivi médicale à distance après hospitalisation pour exacerbation' : Effectifs de la population d'étude/cible, des critères d'exclusion, le taux de l'indicateur
  - La table Flow_chart_BPCO_EFR_SPIRO_*an_N :* le flow-chart de l'indicateur Réalisation d'une EFR ou d'une spirométrie annuelle' : Effectifs de la population d'étude/cible, des critères d'exclusion, le taux de l'indicateur
- FORMATS : Cette librairie contient le catalogue des formats

- VERIF : Cette librairie contient les tables de vérification des tables intermédiaires permettant le calcul des indicateurs

- Définir les différents macros-programmes utilisés

Nom du programme (Avec le chemin)	Objectif du programme (en quelques phrases)
Programmes/_autoexec/	**Options, paramètres, formats, librairies et macros à initialiser à chaque ouverture du projet (exécution automatique dans un flux nommé "autoexec"**
Programmes/_autoexec/Autoexec_Definition_des_formats.sas	Programme permettant de définir l'ensemble des formats utilisés dans l'ensemble du projet
Programmes/_autoexec/Autoexec_Options_de_la_log_et_parametres.sas	Programme permetttant de définir les options de la log, les paramètres (années) et les librairies utilisées dans l'ensemble du projet
Programmes/_autoexec/Macro_Code_pratique_Arreter_un_script_si_erreur.sas	Macro permettant d'arrêter un script dès qu'une erreur ou un warning est repéré
Programmes/_autoexec/Macro_Code_pratique_Proc_Freq.sas	Macro pour effectuer des vérifications (proc freq) au cours du projet
Programmes/_autoexec/Macro_Code_pratique_Supprimer_une_table.sas	Macro permettant de supprimer une table si elle existe
Programmes/_autoexec/Macro_Code_pratique_Verifier_l_existence_d_une_variable.sas	Macro permettant de vérifier l'existance d'une variable
Programmes/_autoexec/Macro_Reperage_des_Soins_CCAM_DCIR.sas	Macro permettant de repérer les soins de ville à partir d'une liste de codes CCAM Suppression des doublons, NIR fictifs, lignes pour information et application du programme de régulations Jointure avec une table de population
Programmes/_autoexec/Macro_Reperage_des_soins_CCAM_PMSI.sas	Macro permettant de repérer les hospitalisations et/ou ACE à partir d'une liste de codes CCAM (HAD, MCO, RIP et/ou SSR) Filtres recommandés par l'ATIH (doublons APHP, APHM, HCL, séjours en erreur, …) Jointure avec une table de population
Programmes/_autoexec/Macro_Reperage_des_soins_CIM10_ALD.sas	Macro permettant de repérer les périodes sous ALD à partir d'une liste de codes CIM10 Sélection des lignes pour ALD uniquement Jointure avec une table de population
Programmes/_autoexec/Macro_Reperage_des_soins_CIM10_PMSI.sas	Macro permettant de repérer les hospitalisations à partir d'une liste de codes CIM10 (HAD, MCO, RIP et/ou SSR) Filtres recommandés par l'ATIH (doublons APHP, APHM, HCL, séjours en erreur, …) Jointure avec une table de population
Programmes/_autoexec/Macro_Reperage_des_soins_CIP_DCIR.sas	Macro permettant de repérer les soins de ville à partir d'une liste de codes CIP (Médicaments) Suppression des doublons, NIR fictifs, lignes pour information et application du programme de régulations Jointure avec une table de population
Programmes/_autoexec/Macro_Reperage_des_soins_Codes_actes_PMSI.sas	Macro permettant de repérer les hospitalisations et/ou ACE à partir d'une liste de codes actes (HAD, MCO, RIP et/ou SSR) Filtres recommandés par l'ATIH (doublons APHP, APHM, HCL, séjours en erreur, …) Jointure avec une table de population
Programmes/_autoexec/Macro_Reperage_des_soins_Codes_GME_PMSI_SSR.sas	Macro permettant de repérer les hospitalisations SSR à partir d'une liste de codes GME Filtres recommandés par l'ATIH (séjours en erreur, …) Jointure avec une table de population
Programmes/_autoexec/Macro_Reperage_des_soins_CSARR_PMSI_SSR.sas	Macro permettant de repérer les hospitalisations SSR à partir d'une liste de codes CSARR Filtres recommandés par l'ATIH (doublons APHP, APHM, HCL, séjours en erreur, …) Jointure avec une table de population
Programmes/_autoexec/Macro_Reperage_des_soins_GHM_PMSI_MCO.sas	Macro permettant de repérer les hospitalisations MCO à partir d'une liste de GHM Filtres recommandés par l'ATIH (doublons APHP, APHM, HCL, séjours en erreur, …) Jointure avec une table de population
Programmes/_autoexec/Macro_Reperage_des_soins_LPP_DCIR.sas	Macro pour repérer les remboursements des codes LPP souhaités
Programmes/_autoexec/Macro_Reperage_des_soins_Medecin_traitant.sas	Macro permettant de repérer les consultations chez un MG ou chez un médecin traitant Suppression des doublons, NIR fictifs, lignes pour information et application du programme de régulations Jointure avec une table de population
Programmes/_autoexec/Macro_Reperage_des_soins_NABM_DCIR.sas	Macro permettant de repérer les soins de ville à partir d'une liste de codes NABM (Biologie) Suppression des doublons, NIR fictifs, lignes pour information et application du programme de régulations Jointure avec une table de population
Programmes/_autoexec/Macro_Reperage_des_soins_NABM_PMSI.sas	Macro permettant de repérer les hospitalisations et/ou ACE à partir d'une liste de codes NABM (HAD, MCO, RIP et/ou SSR) Filtres recommandés par l'ATIH (doublons APHP, APHM, HCL, séjours en erreur, …) Jointure avec une table de population
Programmes/_autoexec/Macro_Reperage_des_soins_Pneumologue_PMSI_SSR.sas	Macro permettant de repérer les consultations chez un pneumologue lors d'une hospitalisation en SSR Filtres recommandés par l'ATIH (séjours en erreur, …) Jointure avec une table de population
Programmes/_autoexec/Macro_Reperage_des_soins_PRS_NAT_REF_DCIR.sas	Macro permettant de repérer les soins de ville à partir d'une liste de natures de prestation Suppression des doublons, NIR fictifs, lignes pour information et application du programme de régulations Jointure avec une table de population
Programmes/_autoexec/Macro_Reperage_des_soins_Tous_sejours_PMSI.sas	Macro permettant de repérer l'ensemble des séjours du PMSI (HAD, MCO, RIP et/ou SSR) Nécessaire pour exclure les séjours index suivis d'hospitalisations dans les XX jours suivant la date index
Programmes/_autoexec/Macro_Reperage_des_soins_UCD_DCIR.sas	Macro permettant de repérer les soins de ville à partir d'une liste de codes UCD (Médicaments) Suppression des doublons, NIR fictifs, lignes pour information et application du programme de régulations Jointure avec une table de population
Programmes/_autoexec/Macro_Reperage_des_soins_UCD_PMSI.sas	Macro permettant de repérer les hospitalisations et/ou ACE à partir d'une liste de codes UCD (HAD, MCO, RIP et/ou SSR) Filtres recommandés par l'ATIH (doublons APHP, APHM, HCL, séjours en erreur, …) Jointure avec une table de population

[3]

[3] RIP dans tous les tableaux correspond au RIM-P (psychiatrie)

2.2 Construction des tables de valeurs et référentiels

Les scripts de cette deuxième partie stockés dans le répertoire '01_Tables_de_valeurs_et_referentiels' doivent être exécutés une seule fois et permettent de créer toutes les tables de valeurs permettant de repérer les soins utilisés au cours du projet BPCO.

Les tables de valeurs et référentiels créées et stockées dans la librairie **ORAUSER** sont les suivantes **:**

Nom du programme (Avec le chemin)	Objectif du programme (en quelques phrases)
Programmes/01_Tables_de_valeurs_et_referentiels/	**Création de toutes les tables de valeurs permettant de repérer les soins utilisés au cours du projet**
Programmes/01_Tables_de_valeurs_et_referentiels/Tables_de_valeurs_et_referentiels_Codes_actes_PMSI_BPCO.sas	Programme permettant de créer une table de valeur dans orauser contenant les codes actes nécessaires à l'ensemble des indicateurs du projet (pour un repérage dans le PMSI) **Table créée : orauser.codes_actes_BPCO**
Programmes/01_Tables_de_valeurs_et_referentiels/Tables_de_valeurs_et_referentiels_Codes_CCAM_BPCO.sas	Programme permettant de créer une table de valeur dans orauser contenant les codes CCAM nécessaires à l'ensemble des indicateurs du projet **Table créée : orauser.codes_CCAM_BPCO**
Programmes/01_Tables_de_valeurs_et_referentiels/Tables_de_valeurs_et_referentiels_Codes_CCAM_Charlson.sas	Programme permettant de créer une table de valeur dans orauser contenant les codes CCAM nécessaires au calcul du score de Charlson **Table créée : orauser.codes_CCAM_Charlson**
Programmes/01_Tables_de_valeurs_et_referentiels/Tables_de_valeurs_et_referentiels_Codes_CIM10_ALD_BPCO.sas	Programme permettant de créer une table de valeur dans orauser contenant les codes CIM10 nécessaires à l'ensemble des indicateurs du projet (pour un repérage via les ALD) **Table créée : orauser.ALD_codes_CIM_BPCO**
Programmes/01_Tables_de_valeurs_et_referentiels/Tables_de_valeurs_et_referentiels_Codes_CIM10_Charlson.sas	Programme permettant de créer une table de valeur dans orauser contenant les codes CIM10 nécessaires au calcul du score de Charlson **Table créée : codes_CIM_Charlson**
Programmes/01_Tables_de_valeurs_et_referentiels/Tables_de_valeurs_et_referentiels_Codes_CIM10_Hospit_BPCO.sas	Programme permettant de créer une table de valeur dans orauser contenant les codes CIM10 nécessaires à l'ensemble des indicateurs du projet (pour un repérage lors d'hospitalisations) **Table créée : orauser.diag_codes_CIM_BPCO**
Programmes/01_Tables_de_valeurs_et_referentiels/Tables_de_valeurs_et_referentiels_Codes_CIP_UCD_BPCO.sas	Programme permettant de créer une table de valeur dans orauser contenant les codes CIP et/ou UCD nécessaires à l'ensemble des indicateurs du projet **Table créée : orauser.codes_ATC_BPCO**
Programmes/01_Tables_de_valeurs_et_referentiels/Tables_de_valeurs_et_referentiels_Codes_CIP_UCD_Charlson.sas	Programme permettant de créer une table de valeur dans orauser contenant les codes CIP et/ou UCD nécessaires au calcul du score de Charlson **Table créée : orauser.codes_ATC_Charlson**
Programmes/01_Tables_de_valeurs_et_referentiels/Tables_de_valeurs_et_referentiels_Codes_CSARR_BPCO.sas	Programme permettant de créer une table de valeur dans orauser contenant les codes CSARR nécessaires à l'ensemble des indicateurs du projet **Table créée : orauser.codes_CSARR_BPCO**
Programmes/01_Tables_de_valeurs_et_referentiels/Tables_de_valeurs_et_referentiels_Codes_GHM_BPCO.sas	Programme permettant de créer une table de valeur dans orauser contenant les codes GHM nécessaires à l'ensemble des indicateurs du projet **Table créée : orauser.codes_GHM_BPCO**
Programmes/01_Tables_de_valeurs_et_referentiels/Tables_de_valeurs_et_referentiels_Codes_GHM_Charlson.sas	Programme permettant de créer une table de valeur dans orauser contenant les codes GHM nécessaires au calcul du score de Charlson **Table créée : orauser.codes_GHM_Charlson**
Programmes/01_Tables_de_valeurs_et_referentiels/Tables_de_valeurs_et_referentiels_Codes_GME_BPCO.sas	Programme permettant de créer une table de valeur dans orauser contenant les codes GME nécessaires à l'ensemble des indicateurs du projet **Table créée : orauser.codes_GME_BPCO**
Programmes/01_Tables_de_valeurs_et_referentiels/Tables_de_valeurs_et_referentiels_Codes_LPP_BPCO.sas	Programme permettant de créer une table de valeur dans orauser contenant les codes LPP nécessaires à l'ensemble des indicateurs du projet **Table créée : orauser.codes_LPP_BPCO**
Programmes/01_Tables_de_valeurs_et_referentiels/Tables_de_valeurs_et_referentiels_Codes_NABM_BPCO.sas	Programme permettant de créer une table de valeur dans orauser contenant les codes NABM nécessaires à l'ensemble des indicateurs du projet **Table créée : orauser.codes_NABM_BPCO**
Programmes/01_Tables_de_valeurs_et_referentiels/Tables_de_valeurs_et_referentiels_Natures_de_prestation_BPCO.sas	Programme permettant de créer une table de valeur dans orauser contenant les natures de prestation nécessaires à l'ensemble des indicateurs du projet **Table créée : orauser.codes_presta_BPCO**

2.3 Sélection des patients

Les scripts de cette troisième partie stockés dans le répertoire '02_Selection_des_patients' doivent être exécutés une seule fois et permettent de sélectionner les patients correspondant aux critères d'inclusion de chaque indicateur.

Les tables créées et stockées dans les librairies **POP, RES, TRAVAIL et ORAUSER** sont les suivantes **:**

Nom du programme (Avec le chemin)	Objectif du programme (en quelques phrases)
Programmes/02_Selection_des_patients/	**Programmes permettant de sélectionner les patients correspondant aux critères de chaque indicateur**
Programmes/02_Selection_des_patients/01_Chainage.sas	Programme permettant de sélectionner l'ensemble des patients et les dernières infos disponibles dans la table IR_BEN_R dans une table pop.T_INDI_BPCO_&an_N. Flag des patients avec un NIR fictif **Table créée : pop.T_INDI_BPCO_&an_N.** **Périmètre de population** : Tous les identifiants disponibles dans IR_BEN_R
Programmes/02_Selection_des_patients/02_Exclusion_des_moins_de_40_ans.sas	Programme permettant de calculer l'âge des patients l'année N Flag des patients de moins de 40 ans **Table mise à jour : pop.T_INDI_BPCO_&an_N.**
Programmes/02_Selection_des_patients/03_Exclusion_des_jumeaux.sas	Programme permettant de repérer les id patient de jumeaux Flag des patients avec des id patients de jumeaux **Table mise à jour : pop.T_INDI_BPCO_&an_N.** **Table créée : orauser.corresp_id_patient** **Périmètre de population :** Tous les identifiants disponibles dans IR_BEN_R
Programmes/02_Selection_des_patients/04_Exclusion_des_patients_sans_remboursement.sas	Programme qui scanne l'ensemble des données de PRS afin de repérer les patients sans remboursement de soins l'année N Flag des patients sans remboursement de soins l'année N **Table mise à jour : pop.T_INDI_BPCO_&an_N.**
Programmes/02_Selection_des_patients/05_Exclusion_des_patients_decedes.sas	Programme permettant de récupérer les dates de décès des patients Dans le PMSI (HAD, MCO, RIP ou SSR) : date de fin de séjour avec mode de sortie = 9 Dans le DCIR : variable BEN_DCD_DTE lorsqu'elle est renseignée On récupère la date de décès minimale s'il en existe plusieurs. Flag des patients décédé avant l'année N On remplace la table de correspondance ans orauser contenant l'ensemble des patients qui ne sont pas exclus précédemment (NIR fictifs, moins de 40 ans, jumeaux, sans remboursement l'année N) **Table mise à jour : pop.T_INDI_BPCO_&an_N.** **Table créée : travail.histo_deces** **Périmètre de population :** Tous les identifiants disponibles dans IR_BEN_R, avec au moins une date de décès dans une des sources **Table mise à jour : orauser.corresp_id_patient** **Périmètre de population :** Tous les identifiants disponibles dans IR_BEN_R, non exclus (supprime et remplace la précédente)
Programmes/02_Selection_des_patients/06_Hospit_BPCO.sas	Programme permettant de repérer les séjours pour BPCO en MCO, SSR ou HAD de l'année N - 4 à l'année N + 1 Table mise à jour : pop.T_INDI_BPCO_&an_N. Table créée : travail.reperage_hospit_BPCO_&annee_4N_&annee_N1. Périmètre de population : Tous les identifiants disponibles dans orauser.corresp_id_patient (création dans le programme Programmes/02_Selection_des_patients/03_Exclusion_des_jumeaux.sas)

Programmes/02_Selection_des_patients/07_Hospit_exacerbation_de_BPCO.sas	Programme permettant de repérer les séjours pour exacerbation de BPCO en MCO terminés dans l'année N **Table mise à jour : pop.T_INDI_BPCO_&an_N.**
Programmes/02_Selection_des_patients/08_ALD_BPCO.sas	Programme permettant de repérer les patients avec une ALD BPCO active dans l'année N **Table mise à jour : pop.T_INDI_BPCO_&an_N.**
Programmes/02_Selection_des_patients/09_Delivrances_de_traitement_BPCO.sas	Programme permettant de compter le nombre de délivrances remboursées de traitement BPCO : BDLA de l'année N - 2 à l'année N + 1, ATB l'année N et arrêt du tabac l'année N - 2 à l'année N + 1 Ajout de l'information sur le nombre de délivrances remboursées de BDLA, ATB ou arrêt du tabac dans la table de population **Table mise à jour : pop.T_INDI_BPCO_&an_N.** **Périmètre de population :** Tous les identifiants disponibles dans IR_BEN_R
Programmes/02_Selection_des_patients/10_Table_finale.sas	Ajout d'indicateurs pour pré-sélectionner les populations de chaque indicateur Création d'une table **res.T_INDI_BPCO_&an_N.** réduite au patients pré-repérés pour au moins un indicateur **Table mise à jour : pop.T_INDI_BPCO_&an_N.** **Table mise à jour : res.T_INDI_BPCO_&an_N.** **Périmètre de population :** Tous les identifiants disponibles dans IR_BEN_R, pré-sélectionnés pour un des indicateurs du projet **Table créée : pop.indicateurs_&an_N.** **Périmètre de population :** Tous les identifiants disponibles dans IR_BEN_R, pré-sélectionnés pour un des indicateurs du projet **Table créée : travail.ref_flowchart** (on stocke le nombre de patients dans pop.T_INDI_BPCO_&an_N. qui ont de 40 ans et plus ayant bénéficié de soins remboursés au moins une fois l'année N) **Table créée : orauser.corresp_id_patient** Périmètre de population : Tous les identifiants disponibles dans res.T_INDI_BPCO_&an_N.

2.4 Informations générales sur les patients

Les scripts de cette quatrième partie stockés dans le répertoire '03_Information_generales_sur_les_patients' doivent être exécutés une seule fois et permettent d'ajouter des informations supplémentaires sur les patients.

Les tables créées et stockées dans les librairies **RES et TRAVAIL** sont les suivantes **:**

Nom du programme (Avec le chemin)	Objectif du programme (en quelques phrases)
Programmes/03_Information_generales_sur_les_patients/	**Ajout d'informations supplémentaires sur les patients - Traitements longs, exécutés sur moins de patients**
Programmes/03_Information_generales_sur_les_patients/01_CMUc.sas	Programme permettant de récupérer l'information de la CMU-C pour chaque patient **Table mise à jour : res.T_INDI_BPCO_&an_N.**
Programmes/03_Information_generales_sur_les_patients/02_IDS.sas	Programme permettant de récupérer l'information de l'IDS pour chaque patient **Table mise à jour : res.T_INDI_BPCO_&an_N.**
Programmes/03_Information_generales_sur_les_patients/03_Infos_ALD.sas	Programme permettant de récupérer l'information de l'ALD par chapitre pour chaque patient **Table mise à jour : res.T_INDI_BPCO_&an_N.**
Programmes/03_Information_generales_sur_les_patients/04_Score_de_Charlson.sas	Programme permettant de calculer le score de Charlson pour chaque patient **Table mise à jour : res.T_INDI_BPCO_&an_N.**
Programmes/03_Information_generales_sur_les_patients/05_Table_T_INDI_BPCO.sas	Ajout de labels et de formats sur l'ensemble des variables de la table res.T_INDI_BPCO_&an_N. **Table mise à jour : res.T_INDI_BPCO_&an_N.** **Table créée : travail.corresp_id_patient** **Périmètre de population :** Tous les identifiants disponibles dans res.T_INDI_BPCO_&an_N.

2.5 Tables communes

Les scripts de cette cinquième partie stockés dans le répertoire '04_Calcul_des_indicateurs/_Tables_communes' doivent être exécutés une seule fois et permettent de sélectionner des soins nécessaires aux calculs des indicateurs.

Les tables créées et stockées dans la librairie **TRAVAIL** sont les suivantes **:**

Nom du programme (Avec le chemin)	Objectif du programme (en quelques phrases)
Programmes/04_Calcul_des_indicateurs/_Tables_communes/	**Sélection de soins, stockés dans la librairie "TRAVAIL", nécessaires aux calculs des indicateurs**
Programmes/04_Calcul_des_indicateurs/_Tables_communes/01_Sejours_pour_exacerbation_de_BPCO.sas	Programme permettant de récupérer les séjours pour exacerbation de BPCO (MCO) Création d'une copie de résultat en excluant les séjours qui se chevauchent (= transferts) qui permettront de repérer les séjours cibles pour les indicateurs **Tables créées : travail.sejours_exacerbation_&annee_1N., travail.sejours_exacerbation_&annee_N., travail.sejours_exacerbation_&annee_N1., travail.sejours_cible_exacerbation_&annee_1N., travail.sejours_cible_exacerbation_&annee_N., travail.sejours_cible_exacerbation_&annee_N1.** **Périmètre de population :** Tous les identifiants disponibles dans res.T_INDI_BPCO_&an_N.
Programmes/04_Calcul_des_indicateurs/_Tables_communes/02_Delivrances_remboursees_de_BDLA.sas	Programme permettant de récupérer l'historique des délivrances remboursées de BDLA pour les années N - 2 à N + 1 **Table créée : travail.reperage_BDLA_&Annee_2N._&Annee_N1.** **Périmètre de population :** Tous les identifiants disponibles dans res.T_INDI_BPCO_&an_N.
Programmes/04_Calcul_des_indicateurs/_Tables_communes/03_Contacts_medecin_generaliste_ou_traitant.sas	Programme permettant de récupérer l'historique des contacts avec un médecin généraliste ou un médecin traitant, pour les années N à N + 1 **Table créée : travail.contact_MG_PMSI_&Annee_N._&Annee_N1. et travail.contact_MT_&Annee_N._&Annee_N1.** **Périmètre de population :** Tous les identifiants pré-sélectionnés pour les indicateurs 4, 5 et 6
Programmes/04_Calcul_des_indicateurs/_Tables_communes/04_Contacs_pneumologues.sas	Programme permettant de récupérer les contacs avec un pneumologue, pour les années N à N + 1 **Table créée : travail.contact_pneumo_&Annee_N._&Annee_N1.** **Périmètre de population :** Tous les identifiants disponibles dans res.T_INDI_BPCO_&an_N.
Programmes/04_Calcul_des_indicateurs/_Tables_communes/05_Rehabilitation_respiratoire.sas	Programme permettant de récupérer les actes de réhabilitation respiratoire, pour les années N - 1 à N + 1, en conservant la source de la RR et le lieu de réalisation On conserve l'information des patients avec un séjour dans une UM 50 ou 54 **Tables créées : travail.reperage_RR_CCAM_CSARR, travail.reperage_CM04_&Annee_1N._&Annee_N1., travail.reperage_RR_&Annee_1N._&Annee_N1., travail.RR_UM54_&Annee_1N._&Annee_N1. et travail.RR_UM50_&Annee_1N._&Annee_N1.** **Périmètre de population :** Tous les identifiants disponibles dans res.T_INDI_BPCO_&an_N.
Programmes/04_Calcul_des_indicateurs/_Tables_communes/06_Oxygenotherapie.sas	Programme permettant de récupérer les actes d'oxygénothérapie, pour les années N - 1 à N + 1 **Table créée : travail.reperage_Oxygeno_&annee_1N._&annee_N1.** **Périmètre de population :** Tous les identifiants disponibles dans res.T_INDI_BPCO_&an_N.
Programmes/04_Calcul_des_indicateurs/_Tables_communes/07_Delivrances_remboursees_d_antibiotherapie.sas	Programme permettant de récupérer les délivrances remboursées d'antibiothérapie respiratoire, pour les années N - 1 à N + 1 **Table créée : travail.reperage_Oxygeno_&annee_1N._&annee_N1.** **Périmètre de population :** Tous les identifiants disponibles dans res.T_INDI_BPCO_&an_N.
Programmes/04_Calcul_des_indicateurs/_Tables_communes/08_Delivrances_remboursees_de_vaccin_anti_grippal.sas	Programme permettant de récupérer les délivrances remboursées de vaccin anti-grippal, pour les années N - 1 à N + 1 **Table créée : travail.reperage_vacc_grippe_&Annee_N._&Annee_N1.** **Périmètre de population :** Tous les identifiants disponibles dans res.T_INDI_BPCO_&an_N.
Programmes/04_Calcul_des_indicateurs/_Tables_communes/09_Sejours_avec_sortie_pour_fuite.sas	Programme permettant de récupérer les séjours avec un code diagnostic de sortie contre avis médical ou fuite, pour l'année N **Table créée : travail.fuite_&annee_N.** **Périmètre de population :** Tous les identifiants disponibles dans res.T_INDI_BPCO_&an_N.
Programmes/04_Calcul_des_indicateurs/_Tables_communes/10_Sejours_soins_palliatifs.sas	Programme permettant de récupérer les séjours avec un code diagnostic de sortie contre avis médical ou fuite, pour l'année N **Table créée : travail.fuite_&annee_N.** **Périmètre de population :** Tous les identifiants disponibles dans res.T_INDI_BPCO_&an_N.
Programmes/04_Calcul_des_indicateurs/_Tables_communes/11_Sejours_PMSI.sas	Programme permettant de récupérer tous les séjours du PMSI (HAD, MCO, RIP, SSR), pour les années N et N+1 **Table créée : travail.sejours_&Annee_N._&Annee_N1.** **Périmètre de population :** Tous les identifiants disponibles dans res.T_INDI_BPCO_&an_N.

Programmes/04_Calcul_des_indicateurs/_Tables_communes/12_Historique_des_ALD.sas	Programme permettant de récupérer tous l'historique des ALD **Table créée : travail.Histo_ALD** **Périmètre de population :** Tous les identifiants disponibles dans res.T_INDI_BPCO_&an_N.
Programmes/04_Calcul_des_indicateurs/_Tables_communes/13_Traitements_asthmatiques.sas	Programme permettant de récupérer toutes les délivrances remboursées de traitements asthamtiques, pour les années N - 2 à N **Table créée : travail.reperage_med_asthme_&Annee_2N._&Annee_N.** **Périmètre de population :** Tous les identifiants disponibles dans res.T_INDI_BPCO_&an_N.
Programmes/04_Calcul_des_indicateurs/_Tables_communes/14_Sejours_pour_asthme.sas	Programme permettant de récupérer tous les séjours avec un code diagnostic pour asthme et état de mal asthamtique, pour les années N - 1 à N + 1 **Table créée : travail.diag_asthme_&annee_1N._&annee_N1.** **Périmètre de population :** Tous les identifiants disponibles dans res.T_INDI_BPCO_&an_N.
Programmes/04_Calcul_des_indicateurs/_Tables_communes/15_Thermoplastie_bronchique.sas	Programme permettant de récupérer toutes les thermoplastie bronchique en MCO, pour les années N - 1 à N + 1 **Table créée : travail.reperage_thermo_bronch_&annee_1N._&annee_N1.** **Périmètre de population :** Tous les identifiants disponibles dans res.T_INDI_BPCO_&an_N.
Programmes/04_Calcul_des_indicateurs/_Tables_communes/16_Polyarthrite_rhumoatoide.sas	Programme permettant de récupérer tous les séjours en avec un code diagnostic pour polyarthrite rhumatoïde, pour les années N - 4 à N + 1 **Table créée : travail.polyarthrite_rhum_&annee_4N._&annee_N1.** **Périmètre de population :** Tous les identifiants disponibles dans res.T_INDI_BPCO_&an_N.
Programmes/04_Calcul_des_indicateurs/_Tables_communes/17_Spondylarthrite_ankylosante.sas	Programme permettant de récupérer tous les séjours avec un code diagnostic pour spondylarthrite ankylosante, pour les années N - 4 à N + 1 **Table créée : travail.Spondylarthrite_&annee_4N._&annee_N1.** **Périmètre de population :** Tous les identifiants disponibles dans res.T_INDI_BPCO_&an_N.
Programmes/04_Calcul_des_indicateurs/_Tables_communes/18_Prothese_hanche_genou_chirurgie_rachis.sas	Programme permettant de récupérer tous les séjours en MCO avec un acte CCAM de prise en charge d'une prothèse de hanche, de genou ou chirurgie du rachis, pour les années N - 1 à N + 1 **Table créée : travail.prothese_chir_rachis_&annee_1N._&annee_N1.** **Périmètre de population :** Tous les identifiants disponibles dans res.T_INDI_BPCO_&an_N.
Programmes/04_Calcul_des_indicateurs/_Tables_communes/19_Insuffisance_coronarienne_aortique.sas	Programme permettant de récupérer tous les séjours en MCO pour insuffisance coronarienne et/ou aortique, pour les années N à N + 1 **Table créée : travail.insuf_coro_aortique_&annee_N._&annee_N1.** **Périmètre de population :** Tous les identifiants disponibles dans res.T_INDI_BPCO_&an_N.
Programmes/04_Calcul_des_indicateurs/_Tables_communes/20_Radiotherapie_chimiotherapie.sas	Programme permettant de récupérer tous les séjours en MCO pour radiothérapie et chimiothérapie, pour les années N - 1 à N + 1 **Table créée : travail.radio_chimio_&annee_1N._&annee_N1.** **Périmètre de population :** Tous les identifiants disponibles dans res.T_INDI_BPCO_&an_N.
Programmes/04_Calcul_des_indicateurs/_Tables_communes/21_EFR_spirometrie.sas	Programme permettant de récupérer tous les actes d'EFR ou de spirométrie, pour les années N - 1 à N + 1 **Table créée : travail.reperage_EFR_spiro_&annee_1N._&annee_N1.** **Périmètre de population :** Tous les identifiants disponibles dans res.T_INDI_BPCO_&an_N.
Programmes/04_Calcul_des_indicateurs/_Tables_communes/22_Pneumothorax_infarctus.sas	Programme permettant de récupérer tous les séjours en MCO pour pneumothorax ou infarctus, pour les années N à N + 1 **Table créée : travail.reperage_VNI_&Annee_N._&annee_N1.** **Périmètre de population** : Tous les identifiants disponibles dans res.T_INDI_BPCO_&an_N.
Programmes/04_Calcul_des_indicateurs/_Tables_communes/23_Ventilation_non_invasive.sas	Programme permettant de récupérer tous les actes CCAM pour ventilation non invasive, pour les années N à N + 1 **Table créée : travail.pneumothorax_infarctus_&Annee_N._&annee_N1.** **Périmètre de population** : Tous les identifiants disponibles dans res.T_INDI_BPCO_&an_N.
Programmes/04_Calcul_des_indicateurs/_Tables_communes/24_Delivrance_remboursee_de_BDCA_et_de_substitut_nicotinique.sas	Programme permettant de récupérer toutes les délivrances remboursées de BDCA et de substitut nicotinique, pour les années N - 1 à N + 1 **Table créée : travail.reperage_BDCA_tabac_&Annee_1N._&Annee_N1.** **Périmètre de population :** Tous les identifiants disponibles dans res.T_INDI_BPCO_&an_N.

2.6 Calcul des indicateurs

2.6.1 Spirométrie ou EFR à visée diagnostique chez les patients à risque de BPCO

Les scripts de cette septième partie stockés dans le répertoire '04_Calcul_des_indicateur/ Indicateur_01_Diagnostic_de_BPCO_recherche' doivent être exécutés une seule fois et permettent de calculer l'indicateur 01_Diagnostic de BPCO recherché.

Les tables créées et stockées dans les librairies **RES et FLOWCH** sont les suivantes **:**

Nom du programme (Avec le chemin)	Objectif du programme (en quelques phrases)
Programmes/04_Calcul_des_indicateurs/Indicateur_01_Diagnostic_de_BPCO_recherche/	**Programmes permettant de calculer l'indicateur 01 "Diagnostic de BPCO recherché (V1)"**
Programmes/04_Calcul_des_indicateurs/Indicateur_01_Diagnostic_de_BPCO_recherche/00_Initialisation_de_la_table_de_resultats.sas	Programme permettant d'initialiser la table de résultat pour l'indicateur 01 avec la sélection des patients à inclure **Table créée : res.T_INDI_BPCO_DG_&an_N.** **Table créée : flowch.Flow_Chart_BPCO_DG_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_01_Diagnostic_de_BPCO_recherche/01_Inclusions_et_exclusions.sas	Programme permettant d'exclure les patients + comptabilisation au fur et à mesure pour le flowchart **Table mise à jour : res.T_INDI_BPCO_DG_&an_N.** **Table mise à jour : flowch.Flow_Chart_BPCO_DG_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_01_Diagnostic_de_BPCO_recherche/02_Traitements_et_population_cible.sas	Programme permettant de repérer les délivrances de traitement : - Délivrances remboursées d'antibiothérapie pour infections respiratoires, précédées de délivrances remboursées d'antibiothérapire pour infection respiratoire dans les 365 jours + Nombre de délivrances remboursées l'année N + Nombre de délivrances remboursées dans les 365 jours précédant la date index - Délivrance remboursée de bronchodilatateur anticholinergique ou Bêta-2 courte durée d'action délivrée le même jour que la cure d'antibiothérapie + Nombre de délivrances remboursées l'année N + Nombre de délivrances remboursées dans les 365 jours précédant la date index - Délivrance remboursée de substituts nicotiniques + Nombre de délivrances remboursées l'année N - Définition de la population cible **Table mise à jour : res.T_INDI_BPCO_DG_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_01_Diagnostic_de_BPCO_recherche/03_Realisation_EFR_ou_spiro.sas	Programme permettant de repérer les patients avec une EFR ou une spirométrie dans es 365 jours suivant la date index **Table mise à jour : res.T_INDI_BPCO_DG_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_01_Diagnostic_de_BPCO_recherche/04_Pneumothorax_infartcus.sas	Programme permettant de repérer les patients avec un pneumothorax ou infarctus (entre -365 et + 365 jours par rapport à la date index) **Table mise à jour : res.T_INDI_BPCO_DG_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_01_Diagnostic_de_BPCO_recherche/05_Table_finale.sas	Ajout de labels et de formats sur l'ensemble des variables de la table **res.T_INDI_BPCO_DG_&an_N.** **Table mise à jour : res.T_INDI_BPCO_DG_&an_N.** **Table mise à jour : flowch.Flow_Chart_BPCO_DG_&an_N.** **Table mise à jour : res.T_INDI_BPCO_&an_N.**

2.6.2 Vaccin contre la grippe chez les patients atteints de BPCO

Les scripts de cette septième partie stockés dans le répertoire '04_Calcul_des_indicateur/ Indicateur_02_Vaccin_contre_la_grippe' doivent être exécutés une seule fois et permettent de de calculer l'indicateur 02_Vaccin contre la grippe (probable et diagnostiqué).

Les tables créées et stockées dans les librairies **RES et FLOWCH** sont les suivantes **:**

Nom du programme (Avec le chemin)	Objectif du programme (en quelques phrases)
Programmes/04_Calcul_des_indicateurs/Indicateur_02_Vaccin_contre_la_grippe/	**Programmes permettant de calculer l'indicateur 02 "Vaccin contre la gippe"**
Programmes/04_Calcul_des_indicateurs/Indicateur_02_Vaccin_contre_la_grippe/00_Initialisation_des_Flowcharts.sas	Programme permettant d'initialiser les tables de résultat temporaires pour les indicateurs 02a et 02b avec la sélection des patients à inclure **Table créée : T_INDI_BPCO_VACGA_&an_N.** **Table créée : T_INDI_BPCO_VACGB_&an_N.** **Table créée : flowch.Flow_Chart_BPCO_VACG_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_02_Vaccin_contre_la_grippe/01_Exclusions_BPCO_probable.sas	Programme permettant d'exclure les patients BPCO probable + comptabilisation au fur et à mesure pour le flowchart **Table mise à jour : flowch.Flow_Chart_BPCO_VACG_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_02_Vaccin_contre_la_grippe/02_Populations_etude_cible.sas	Programme permettant d'exclure les patients + comptabilisation au fur et à mesure pour le flowchart **Table créée : T_INDI_BPCO_VACG_&an_N.** **Table mise à jour : flowch.Flow_Chart_BPCO_D_VACG_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_02_Vaccin_contre_la_grippe/03_Initialisation_de_la_table_de_resultats.sas	Programme permettant d'initialiser la table de résultat pour l'indicateur 02 avec la sélection des patients à inclure : - Définition des populations cible - Repérage des patients ayant eu au moins un diagnostic de BPCO - Repérage des patients en ALD BPCO active le 1er septembre **Table créée : res.T_INDI_BPCO_VACG_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_02_Vaccin_contre_la_grippe/04_Sejours_avec_un_diag_de_BPCO.sas	Programme permettant de compter le nombre de séjours BPCO **Table mise à jour : res.T_INDI_BPCO_VACG_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_02_Vaccin_contre_la_grippe/05_Vaccin_contre_la_grippe.sas	Programme permettant de repérer la vaccination contre la grippe **Table mise à jour : res.T_INDI_BPCO_VACG_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_02_Vaccin_contre_la_grippe/06_Table_finale.sas	Ajout de labels et de formats sur l'ensemble des variables de la table res.T_INDI_BPCO_&an_N. **Table mise à jour : res.T_INDI_BPCO_VACG_&an_N.** **Table mise à jour : flowch.Flow_Chart_BPCO_VACG_&an_N.** **Table mise à jour : res.T_INDI_BPCO_&an_N.**

2.6.3 Réalisation d'EFR ou d'une spirométrie annuelle chez les patients atteints de BPCO

Les scripts de cette septième partie stockés dans le répertoire '04_Calcul_des_indicateur/ Indicateur_03_Realisation_d_EFR_ou_d_une_spirometrie_annuelle' doivent être exécutés une seule fois et permettent de calculer l'indicateur 03_Réalisation d'une EFR ou d'une spirométrie annuelle.

Les tables créées et stockées dans les librairies **RES et FLOWCH** sont les suivantes **:**

Nom du programme (Avec le chemin)	Objectif du programme (en quelques phrases)
Programmes/04_Calcul_des_indicateurs/Indicateur_03_Realisation_d_EFR_ou_d_une_spirometrie_annuelle/	**Programmes permettant de calculer l'indicateur 03 "Réalisation d'EFR ou d'une spirométrie annuelle"**
Programmes/04_Calcul_des_indicateurs/Indicateur_03_Realisation_d_EFR_ou_d_une_spirometrie_annuelle/00_Initialisation_de_la_table_de_resultats.sas	Programme permettant d'initialiser la table de résultat temporaire pour l'indicateur 03 avec la sélection des patients à inclure **Table créée : T_INDI_BPCO_EFR_SPIRO_&an_N.** **Table créée : flowch.Flow_Chart_BPCO_EFR_SPIRO_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_03_Realisation_d_EFR_ou_d_une_spirometrie_annuelle/01_Inclusions_et_exclusions.sas	Programme permettant : - d'initialiser la table de résultat pour l'indicateur 03 avec la sélection des patients à inclure - d'exclure les patients + comptabilisation au fur et à mesure pour le flowchart **Table créée : res.T_INDI_BPCO_EFR_SPIRO_&an_N.** **Table mise à jour : flowch.Flow_Chart_BPCO_EFR_SPIRO_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_03_Realisation_d_EFR_ou_d_une_spirometrie_annuelle/02_Sejours_avec_un_diag_de_BPCO.sas	Programme permettant de compter le nombre de séjours BPCO **Table mise à jour : res.T_INDI_BPCO_EFR_SPIRO_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_03_Realisation_d_EFR_ou_d_une_spirometrie_annuelle/03_Realisation_EFR_ou_spiro.sas	Programme permettant de repérer les réalisations d'EFR et de spirométrie **Table mise à jour : res.T_INDI_BPCO_EFR_SPIRO_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_03_Realisation_d_EFR_ou_d_une_spirometrie_annuelle/04_Pneumo_Infarctus_Oxygeno_VNI.sas	Programme permettant de repérer les réalisations de pneumothorax, infarctus, oxygénothérapie et VNI **Table mise à jour : res.T_INDI_BPCO_EFR_SPIRO_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_03_Realisation_d_EFR_ou_d_une_spirometrie_annuelle/05_Table_finale.sas	Ajout de labels et de formats sur l'ensemble des variables de la table res.T_INDI_BPCO_EFR_SPIRO_&an_N. **Table mise à jour : res.T_INDI_BPCO_EFR_SPIRO_&an_N.** **Table mise à jour : flowch.Flow_Chart_BPCO_EFR_SPIRO_&an_N.** **Table mise à jour : res.T_INDI_BPCO_&an_N.**

2.6.4 Suivi médical dans les 7 jours après une hospitalisation pour exacerbation de BPCO des patients sortis à domicile

Les scripts de cette septième partie stockés dans le répertoire '04_Calcul_des_indicateur/ Indicateur_04_Suivi_medical_precoce_apres_hospitalisation_pour_exacerbation' doivent être exécutés une seule fois et permettent de calculer l'indicateur 04_Suivi médical précoce après hospitalisation pour exacerbation

Les tables créées et stockées dans les librairies **RES et FLOWCH** sont les suivantes **:**

Nom du programme (Avec le chemin)	Objectif du programme (en quelques phrases)
Programmes/04_Calcul_des_indicateurs/Indicateur_Suivi_medical_precoce_apres_hospitalisation_pour_exacerbation/	**Programmes permettant de calculer l'indicateur 04 "Suivi médical précoce après hospitalisation pour exacerbation"**
Programmes/04_Calcul_des_indicateurs/Indicateur_04_Suivi_medical_precoce_apres_hospitalisation_pour_exacerbation/00_Initialisation_de_la_table_de_resultats.sas	Programme permettant d'initialiser la table de résultat temporaire pour l'indicateur 04 avec la sélection des patients à inclure **Table créée : T_INDI_BPCO_SMED_P_EA_&an_N.** **Table créée : flowch.Flow_Chart_BPCO_SMED_P_EA_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_04_Suivi_medical_precoce_apres_hospitalisation_pour_exacerbation/01_Inclusions_et_exclusions.sas	Programme permettant : - d'initialiser la table de résultat pour l'indicateur 04 avec le détail du séjour d'inclusion - d'exclure les patients + comptabilisation au fur et à mesure pour le flowchart **Table créée : res.T_INDI_BPCO_SMED_P_EA_&an_N.** **Table mise à jour : flowch.Flow_Chart_BPCO_SMED_P_EA_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_04_Suivi_medical_precoce_apres_hospitalisation_pour_exacerbation/02_Contact_MG_MT.sas	Programme permettant de repérer les contacts avec un médecin généraliste ou médecin traitant + Nombre de contacts **Table mise à jour : res.T_INDI_BPCO_SMED_P_EA_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_04_Suivi_medical_precoce_apres_hospitalisation_pour_exacerbation/03_Contact_pneumologue.sas	Programme permettant de repérer les contacts avec un pneumologue + Nombre de contacts **Table mise à jour : res.T_INDI_BPCO_SMED_P_EA_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_04_Suivi_medical_precoce_apres_hospitalisation_pour_exacerbation/04_Sejours_exacerbation_de_BPCO_en_MCO.sas	Programme permettant de compter le nombre de séjours pour exacerbation de BPCO **Table mise à jour : res.T_INDI_BPCO_SMED_P_EA_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_04_Suivi_medical_precoce_apres_hospitalisation_pour_exacerbation/05_Table_finale.sas	Ajout de labels et de formats sur l'ensemble des variables de L(-3)Cla table res.T_INDI_BPCO_SMED_P_EA_&an_N. **Table mise à jour : res.T_INDI_BPCO_SMED_P_EA_&an_N.** **Table mise à jour : flowch.Flow_Chart_BPCO_SMED_P_EA_&an_N.** **Table mise à jour : res.T_INDI_BPCO_&an_N.**

2.6.5 Suivi par le pneumologue dans les 60 jours après hospitalisation pour exacerbation de BPCO des patients sortis à domicile

Les scripts de cette septième partie stockés dans le répertoire '04_Calcul_des_indicateur/ Indicateur_05_Suivi_medical_a_distance_apres_hospitalisation_pour_exacerbation' doivent être exécutés une seule fois et permettent de de calculer l'indicateur 05_Suivi médical à distance après hospitalisation pour exacerbation

Les tables créées et stockées dans les librairies **RES et FLOWCH** sont les suivantes **:**

Nom du programme (Avec le chemin)	Objectif du programme (en quelques phrases)
Programmes/04_Calcul_des_indicateurs/Indicateur_05_Suivi_medical_a_distance_apres_hospitalisation_pour_exacerbation/	**Programmes permettant de calculer l'indicateur 05 "Suivi médical à distance après hospitalisation pour exacerbation"**
Programmes/04_Calcul_des_indicateurs/Indicateur_05_Suivi_medical_a_distance_apres_hospitalisation_pour_exacerbation/00_Initialisation_de_la_table_de_resultats.sas	Programme permettant d'initialiser la table de résultat temporaire pour l'indicateur 05 avec la sélection des patients à inclure **Table créée : T_INDI_BPCO_SMED_D_EA_&an_N.** **Table créée : flowch.Flow_Chart_BPCO_SMED_D_EA_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_05_Suivi_medical_a_distance_apres_hospitalisation_pour_exacerbation/01_Inclusions_et_exclusions.sas	Programme permettant : - d'initialiser la table de résultat pour l'indicateur 05 avec le détail du séjour d'inclusion - d'exclure les patients + comptabilisation au fur et à mesure pour le flowchart **Table créée : res.T_INDI_BPCO_SMED_D_EA_&an_N.** **Table mise à jour : flowch.Flow_Chart_BPCO_SMED_D_EA_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_05_Suivi_medical_a_distance_apres_hospitalisation_pour_exacerbation/02_Contact_pneumologue.sas	Programme permettant de repérer les contacts avec un pneumologue + Nombre de contacts **Table mise à jour : res.T_INDI_BPCO_SMED_D_EA_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_05_Suivi_medical_a_distance_apres_hospitalisation_pour_exacerbation/03_Sejour_exacerbation_de_BPCO_en_MCO.sas	Programme permettant de compter le nombre de séjours pour exacerbation de BPCO **Table mise à jour : res.T_INDI_BPCO_SMED_D_EA_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_05_Suivi_medical_a_distance_apres_hospitalisation_pour_exacerbation/04_Table_finale.sas	Ajout de labels et de formats sur l'ensemble des variables de la table res.T_INDI_BPCO_SMED_D_EA_&an_N. **Table mise à jour : res.T_INDI_BPCO_SMED_D_EA_&an_N.** **Table mise à jour : flowch.Flow_Chart_BPCO_SMED_D_EA_&an_N.** **Table mise à jour :** res.T_INDI_BPCO_&an_N.

2.6.6 Traitement remboursé de BDLA délivré dans les 90 jours après une hospitalisation pour exacerbation de BPCO des patients sortis à domicile

Les scripts de cette septième partie stockés dans le répertoire '04_Calcul_des_indicateur/ Indicateur_06_Traitement_apres_hospitalisation_pour_exacerbation' doivent être exécutés une seule fois et permettent de calculer l'indicateur 06_Traitement après hospitalisation pour exacerbation

Les tables créées et stockées dans les librairies **RES et FLOWCH** sont les suivantes **:**

Nom du programme (Avec le chemin)	Objectif du programme (en quelques phrases)
Programmes/04_Calcul_des_indicateurs/Indicateur_06_Traitement_apres_hospitalisation_pour_exacerbation/	**Programmes permettant de calculer l'indicateur 06 "Traitement après hospitalisation pour exacerbation"**
Programmes/04_Calcul_des_indicateurs/Indicateur_06_Traitement_apres_hospitalisation_pour_exacerbation/00_Initialisation_de_la_table_de_resultats.sas	Programme permettant d'initialiser la table de résultat temporaire pour l'indicateur 06 avec la sélection des patients à inclure **Table créée : T_INDI_BPCO_BDLA_EA_&an_N.** **Table créée : flowch.Flow_Chart_BPCO_BDLA_EA_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_06_Traitement_apres_hospitalisation_pour_exacerbation/01_Inclusions_et_exclusions.sas	Programme permettant : - d'initialiser la table de résultat pour l'indicateur 06 avec le détail du séjour d'inclusion - d'exclure les patients + comptabilisation au fur et à mesure pour le flowchart **Table créée : res.T_INDI_BPCO_BDLA_EA_&an_N.** **Table mise à jour : flowch.Flow_Chart_BPCO_BDLA_EA_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_06_Traitement_apres_hospitalisation_pour_exacerbation/02_Traitement_apres_hospitalisation.sas	Programme permettant de repérer les traitements avant et après hospitalisation pour exacerbation + Nombre de délivrances dans les 90 jours + 1ère date de délivrance dans les 180 jours **Table mise à jour : res.T_INDI_BPCO_BDLA_EA_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_06_Traitement_apres_hospitalisation_pour_exacerbation/03_Sejours_d_exacerbation_deBPCO_en_MCO.sas	Programme permettant de compter le nombre de séjours pour exacerbation de BPCO **Table mise à jour : res.T_INDI_BPCO_BDLA_EA_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_06_Traitement_apres_hospitalisation_pour_exacerbation/04_Table_finale.sas	Ajout de labels et de formats sur l'ensemble des variables de la table res.T_INDI_BPCO_BDLA_EA_&an_N. **Table mise à jour : res.T_INDI_BPCO_BDLA_EA_&an_N.** **Table mise à jour : flowch.Flow_Chart_BPCO_BDLA_EA_&an_N.** **Table mise à jour : res.T_INDI_BPCO_&an_N.**

2.6.7 Soins de rééducation dans les 90 jours après une hospitalisation pour exacerbation de BPCO des patients sortis à domicile

Les scripts de cette septième partie stockés dans le répertoire '04_Calcul_des_indicateur/ Indicateur_Indicateur_07_Readaptation_respiratoire_apres_exacerbation ' doivent être exécutés une seule fois et permettent de calculer l'indicateur 07_Réadaptation respiratoire après hospitalisation pour exacerbation

Les tables créées et stockées dans les librairies **RES et FLOWCH** sont les suivantes **:**

Nom du programme (Avec le chemin)	Objectif du programme (en quelques phrases)
Programmes/04_Calcul_des_indicateurs/Indicateur_07_Readaptation_respiratoire_apres_exacerbation/	**Programmes permettant de calculer l'indicateur 07 "Réadaptation respiratoire après exacerbation"**
Programmes/04_Calcul_des_indicateurs/Indicateur_07_Readaptation_respiratoire_apres_exacerbation/00_Initialisation_de_la_table_de_resultats.sas	Programme permettant d'initialiser la table de résultat temporaire pour l'indicateur 07 avec la sélection des patients à inclure **Table créée : T_INDI_BPCO_RR_EA_&an_N.** **Table créée : flowch.Flow_Chart_BPCO_RR_EA_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_07_Readaptation_respiratoire_apres_exacerbation/01_Inclusions_et_exclusions.sas	Programme permettant : - d'initialiser la table de résultat pour l'indicateur 07 avec le détail du séjour d'inclusion - d'exclure les patients + comptabilisation au fur et à mesure pour le flowchart **Table créée : res.T_INDI_BPCO_RR_EA_&an_N.** **Table mise à jour : flowch.Flow_Chart_BPCO_RR_EA_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_07_Readaptation_respiratoire_apres_exacerbation/02_Rehabilitation_respiratoire.sas	Programme permettant de repérer la réhabilitation respiratoire + le lieu de réalisation **Table mise à jour : res.T_INDI_BPCO_RR_EA_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_07_Readaptation_respiratoire_apres_exacerbation/03_Sejours_d_exacerbation_de_BPCO_en_MCO.sas	Programme permettant de compter le nombre de séjours pour exacerbation de BPCO **Table mise à jour : res.T_INDI_BPCO_RR_EA_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_07_Readaptation_respiratoire_apres_exacerbation/04_Polyarthrite_rhumatoides.sas	Programme permettant de repérer les patients pris en charge pour polyarthrite rhumatoïde et maladies apparentées **Table mise à jour : res.T_INDI_BPCO_RR_EA_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_07_Readaptation_respiratoire_apres_exacerbation/05_Spondylarthrite_ankylosante.sas	Programme permettant de repérer les patients pris en charge pour spondylarthrite ankylosante et maladies apparentées **Table mise à jour : res.T_INDI_BPCO_RR_EA_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_07_Readaptation_respiratoire_apres_exacerbation/06_Protheses_et_chirurgie_du_rachis.sas	Programme permettant de repérer les patients pris en charge pour : - une prothèse de hanche - une prothèse de genou - une chirurgie du rachis **Table mise à jour : res.T_INDI_BPCO_RR_EA_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_07_Readaptation_respiratoire_apres_exacerbation/07_Pontage_valve_cardiaque_et_stents.sas	Programme permettant de repérer les patients pris en charge pour : - un pontage coronarien - un remplacement de valve cardiaque - une prothèse vasculaire (stents) **Table mise à jour : res.T_INDI_BPCO_RR_EA_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_07_Readaptation_respiratoire_apres_exacerbation/08_Radiotherapie_chimiotherapie.sas	Programme perm+L(-12)Cettant de repérer les patients ayant eu : - au moins une séance de radiothérapie - au moins une séance de chimiothérapie **Table mise à jour : res.T_INDI_BPCO_RR_EA_&an_N.**
Programmes/04_Calcul_des_indicateurs/Indicateur_07_Readaptation_respiratoire_apres_exacerbation/09_Table_finale.sas	Ajout de labels et de formats sur l'ensemble des variables de la table res.T_INDI_BPCO_RR_EA_&an_N. **Table mise à jour : res.T_INDI_BPCO_RR_EA_&an_N.** **Table mise à jour : flowch.Flow_Chart_BPCO_RR_EA_&an_N.** **Table mise à jour : res.T_INDI_BPCO_&an_N.**

2.6.8 Suppression des décédés dans les tables finales

Les scripts de cette septième partie stockés dans le répertoire '04_Calcul_des_indicateur/ Suppressions_finales doivent être exécutés une seule fois et permettent

- d'exclure les patients avec une date de décès antérieure à l'une des dates décrites dans l'une des 7 tables des indicateurs
- de mettre les flowcharts des 7 indicateurs avec cette nouvelle exclusion
- de supprimer les variables et observations inutiles dans la table T_INDI_BPCO_&an_N

Les tables modifiées dans les librairies **RES et FLOWCH** sont les suivantes **:**

Nom du programme (Avec le chemin)	**Objectif du programme (en quelques phrases)**
Programmes/04_Calcul_des_indicateurs/Suppressions_finales/	**Programmes permettant de** **- Exclure les patients décédés avant l'une des dates des indicateurs** **- Supprimer les patients non repérés par un indicateur dans la table res.T_INDI_BPCO_&an_N.** **- Supprimer les variables non demandées dans la table res.T_INDI_BPCO_&an_N.**
Programmes/04_Calcul_des_indicateurs/Suppressions_finales/01_Patients_decedes_Reperage.sas	Programme permettant de sélectionner les patients décédés avant une des dates décrites dans les tables finales des indicateurs **Table mise à jour : res.T_Indi_BPCO_DG_&an_N.** **Table mise à jour : res.T_Indi_BPCO_VacG_&an_N.** **Table mise à jour : res.T_Indi_BPCO_EFR_SPIRO_&an_N.** **Table mise à jour : res.T_Indi_BPCO_Smed_P_EA_&an_N.** **Table mise à jour : res.T_Indi_BPCO_Smed_D_EA_&an_N.** **Table mise à jour : res. T_Indi_BPCO_BDLA_EA_&an_N.** **Table mise à jour : res.T_Indi_BPCO_RR_EA_&an_N.**
Programmes/04_Calcul_des_indicateurs/Suppressions_finales/02_Patients_decedes_maj_Flowchart.sas	Programme permettant de mettre à jour les flowchart suite à la sélection des patients dans le programme précédant **Table mise à jour : flowch.flow_chart_bpco_DG_&an_N** **Table mise à jour : flowch.flow_chart_bpco_VACG_&an_N** **Table mise à jour : flowch.flow_chart_bpco_EFR_SPIRO_&an_N** **Table mise à jour : flowch.flow_chart_bpco_Smed_EA_P_&an_N** **Table mise à jour : flowch.flow_chart_Smed_D_EA_&an_N** **Table mise à jour : flowch.flow_chart_bpco_BDLEA_EA_&an_N** **Table mise à jour : flowch.flow_chart_bpco_RR_EA_&an_N**
Programmes/04_Calcul_des_indicateurs/Suppressions_finales/03_Patients_decedes_Suppression_decedes.sas	Programme permettant d'exclure les patients décédés avant une des dates décrites dans les tables finales des indicateurs **Table mise à jour : res.T_Indi_BPCO_DG_&an_N.** **Table mise à jour : res.T_Indi_BPCO_VacG_&an_N.** **Table mise à jour : res.T_Indi_BPCO_EFR_SPIRO_&an_N.** **Table mise à jour : res.T_Indi_BPCO_Smed_P_EA_&an_N.** **Table mise à jour : res.T_Indi_BPCO_Smed_D_EA_&an_N.** **Table mise à jour : res. T_Indi_BPCO_BDLA_EA_&an_N.** **Table mise à jour : res.T_Indi_BPCO_RR_EA_&an_N.** **Table mise à jour : res.T_INDI_BPCO_&an_N.**
Programmes/04_Calcul_des_indicateurs/Suppressions_finales/04_Suppression_Infos_temporaires.sas	Programme permettant de supprimer de toutes les tables variables et observations inutiles **Table mise à jour : res.T_INDI_BPCO_&an_N.**

3. Programmes détaillés

3.1 Définition des librairies, formats et des macros-programmes

3.1.1 Autoexec_Definition_des_formats.sas

```
/*
	*****************************************************************************************************
**********************************	*/
/*

*/
/*		Définition des formats

*/
/*

*/
/*
	*****************************************************************************************************
**********************************	*/

* Format f_reperage : 1ère partie : Nom de l onglet du fichier BPCO_LISTE_CODES_202004.xlsx - 2ème partie : Mesure recherchée;
proc format library = formats.formats;
	value f_reperage
		1 = "Actes - EFR"
		2 = "Actes - Spirométrie"
		3 = "Actes - Mesure des gaz du sang"
		4 = "Actes - Thermoplastie bronchique"
		5 = "Actes - Oxygénothérapie BPCO"
		6 = "Actes - Ventilation non invasive BPCO"
		7 = "Actes - Réhabilitation respiratoire"
		8 = "Actes - Prothèse de hanche"
		9 = "Actes - Prothèse du genou"
		10 = "Actes - Chirurgie du rachis"
		11 = "Diagnostics - Pneumothorax"
		12 = "Diagnostics - Infarctus"
		13 = "Diagnostics - Soins palliatif"
		14 = "Diagnostics - Asthme"
		15 = "Diagnostics - Etat de mal asthmatique"
		16 = "Diagnostics - Diagnostic de BPCO"
		17 = "Diagnostics - Insuffisance respiratoire aigue"
		18 = "Diagnostics - Pneumopathie lobaire"
		19 = "Diagnostics - Grippe"
		20 = "Diagnostics - Embolie pulmonaire"
		21 = "Diagnostics - Insuffisance cardiaque aigue"
		22 = "Diagnostics - Insuffisance cardiaque - Œdème aigu pulmonaire"
		23 = "Diagnostics - Infections des voies aériennes"
		24 = "Diagnostics - Pneumonies"
		25 = "Diagnostics - Bronchopneumopathies"
		26 = "Diagnostics - Polyarthrite rhumatoïde et maladies apparentées"
		27 = "Diagnostics - Spondylarthrite ankylosante ou maladies apparentées"
		28 = "Diagnostics - Pontages coronariens"
		29 = "Diagnostics - Remplacement valve cardiaque"
		30 = "Diagnostics - Prothèse vasculaires - Stents"
		31 = "Biologie - Gaz du sang"
		32 = "Prestation - Contact médecin pneumologue"
		33 = "Prestation - Contact médecin généraliste"
		34 = "Pneumo_Sejour_SSR - Contact avec un pneumologue lors d'une hospitalisation en SSR"
		35 = "Médicaments - Médicaments utilisés dans la dépendance tabagique"
		36 = "Médicaments - Antibiotiques"
		37 = "Médicaments - Traitement anti IgE"
		38 = "Médicaments - Ac monoclonaux Traitement anti IL-5"
		39 = "Médicaments - Antileucotriène"
		40 = "Médicaments - Vaccin anti grippal"
```

```
                    41 = "Médicaments - Cortico stéroïdes inhalés (CSI)"
                    42 = "Bronchodilatateurs - Bêta-2 agoniste de courte durée d'action "
                    43 = "Bronchodilatateurs - Anticholinergique de courte durée d'action"
                    44 = "Bronchodilatateurs - Bêta-2 + Anticholinergique de courte durée d'action"
                    45 = "Bronchodilatateurs - Bêta-2 agonistes de longue durée d'action (LABA)"
                    46 = "Bronchodilatateurs - Anticholinergique de longue durée d'action (LAMA)"
                    47 = "Bronchodilatateurs - Bêta-2 agoniste de longue durée d'action et Anticholinergique de longue durée
d'action"
                    48 = "Bronchodilatateurs - Bêta 2 agoniste de longue durée d'action et corticostéroïde inhalé"
                    49 = "Bronchodilatateurs - Bêta 2 agoniste de longue durée d'action + anticholinergique de longue durée d'action
+ corticostéroïde inhalé"
                    50 = "ALD - Maladie d'alzheimer"
                    51 = "ALD - Autres démences"
                    52 = "ALD - BPCO"
                    53 = "ALD - Asthme"
                    54 = "ALD - Mucoviscidose"
                    55 = "ALD - Insuffisance cardiaque"
                    56 = "ALD - Polyarthrite rhumatoïde et maladies apparentées"
                    57 = "ALD - Spondylarthrite ankylosante ou maladies apparentées"
                    58 = "Séances - Chimiothérapie"
                    59 = "Séances - Radiothérapie"
                    ;
          value f_Charlson
                    1 = "Infarctus du myocarde"
                    2 = "Insuffisance cardiaque"
                    3 = "Pathologie vasculaire périphérique"
                    4 = "Pathologie cérébrovasculaire"
                    5 = "Démence"
                    6 = "Pathologie pulmonaire chronique"
                    7 = "Connectivite"
                    8 = "Pathologie ulcéreuse"
                    9 = "Pathologie hépatique légère"
                    10 = "Diabète sans complication"
                    11 = "Hémiplégie"
                    12 = "Pathologie rénale modérée ou sévère"
                    13 = "Diabète avec complication"
                    14 = "Cancer (comprenant les lymphomes et les leucémies et excluant les pathologies néoplasiques de la peau)"
                    15 = "Pathologie hépatique modérée ou sévère"
                    16 = "Pathologie métastatique"
                    17 = "VIH-SIDA"
                    ;
          value f_oui_non
                    0 = "Non"
                    1 = "Oui"
                    ;
          value f_ind_01_flowchart
                    1 = "Patients de 40 ans et plus ayant eu au moins une délivrance remboursée de BDLA"
                    2 = "Patients de 40 ans et plus ayant eu au moins 1 ATB + 1 BDCA (le même jour en 2017) et 1 ATB (365 jours
avant)"
                    2.5 = "Patients de 40 ans et plus ayant eu au moins 1 ATB (en 2017) et 1 ATB + 1 BDCA (le même jour et 365 jours
avant)"
                    3 = "Patients de 40 ans et plus ayant eu au moins 1 TTT arrêt tabac"
                    4 = "Patients de 40 ans et plus ayant eu un traitement spécifique de la BPCO"
                    5 = "Patients asthmatiques - ALD asthme"
                    6 = "Patients asthmatiques - ALD mucoviscidose"
                    7 = "Patients asthmatiques - Antécédents d'au moins 3 CSI seuls"
                    8 = "Patients asthmatiques - Total"
                    9 = "Patients suspects de BPCO hors asthmatiques"
                    10 = "Patients déjà diagnostiqués - ALD BPCO"
                    11 = "Patients déjà diagnostiqués - Antécédents de BDLA"
                    12 = "Patients déjà diagnostiqués - Diagnostic de BPCO codé lors d'une hospitalisation"
                    13 = "Patients déjà diagnostiqués - Spirométrie ou EFR entre 730 jours et 366 jours précédant la date index"
                    14 = "Patients déjà diagnostiqués - Total"
                    15 = "Population d'étude : patients incidents"
                    16 = "Patients pour lesquels la spirométrie ou les EFR ne sont pas réalisables - ALD alzheimer et autres démences"
                    17 = "Patients pour lesquels la spirométrie ou les EFR ne sont pas réalisables - Hospitalisés en soins palliatifs"
                    18 = "Patients pour lesquels la spirométrie ou les EFR ne sont pas réalisables - Total"
                    19 = "Patients avec un problème sur la date de décès"
                    20 = "Population cible"
                    21 = "Indicateur : Spirométrie ou EFR de diagnostic"
                    ;
```

```
value f_ind_02_flowchart
        1 = "Patients de 40 ans et plus avec une ALD BPCO active au 1er septembre &Annee_N."
        2 = "Patients de 40 ans et plus hospitalisés en MCO, SSR HAD avec un diagnostic BPCO codé"
        3 = "Population d'étude 1 : BPCO diagnostiquée"
        4 = "Patients de 40 ans et plus ayant eu un traitement spécifique de la BPCO (3 BDLA)"
        5 = "Patients asthmatiques - ALD asthme"
        6 = "Patients asthmatiques - ALD mucoviscidose"
        7 = "Patients asthmatiques - Au moins 3 CSI seuls"
        8 = "Patients asthmatiques - Au moins 1 Anti-IgE"
        9 = "Patients asthmatiques - Au moins 1 Anti-IL-5"
        10 = "Patients asthmatiques - Au moins 1 antileucotriène"
        11 = "Patients asthmatiques - Acte de thermoplastie bronchique codé lors d'une hospitalisation en MCO"
        12 = "Patients asthmatiques - Diagnostic d'asthme codé lors d'une hospitalisation"
        13 = "Patients asthmatiques - Diagnostic d'état de mal asthmatique codé lors d'une hospitalisation"
        14 = "Patients asthmatiques - Total"
        15 = "Patients diagnostiqués BPCO - Patients avec une ALD active en &Annee_N."
        16 = "Patients diagnostiqués BPCO - Patients hospitalisés en MCO, SSR HAD avec un diagnostic BPCO codé"
        17 = "Patients diagnostiqués BPCO (chez les probables) - Total"
        18 = "Population d'étude 2 : BPCO probables"
        19 = "Population d'étude : Patients avec une BPCO probables ou diagnostiquée"
        20 = "Patients pour lesquels la recherche de vaccination contre la grippe n'est pas possible - Décédés avant la
campagne"
        21 = "Patients pour lesquels la recherche de vaccination contre la grippe n'est pas possible - Hospitalisés en soins
palliatifs"
        22 = "Patients pour lesquels la recherche de vaccination contre la grippe n'est pas possible - Hospitalisés plus de 3
mois en MCO, SSR, psychiatrie"
        23 = "Patients pour lesquels la recherche de vaccination contre la grippe n'est pas possible - Total"
        24 = "Patients avec un problème sur la date de décès"
        25 = "Population cible"
        26 = "Indicateur : Vaccination contre la grippe"
        ;
value f_ind_03_flowchart
        1 = "Patients BPCO probable - Au moins 3 BDLA"
        2 = "Patients BPCO diagnostiquée - ALD BPCO active au 31 décembre &Annee_N."
        3 = "Patients BPCO diagnostiquée - Hospitalisés en MCO, SSR, HAD avec un diagnostic BPCO codé"
        4 = "Patients  de 40 ans et plus ayant une BPCO probable ou diagnostiquée"
        5 = "Patients asthmatiques - ALD asthme"
        6 = "Patients asthmatiques - ALD mucoviscidose"
        7 = "Patients asthmatiques - Au moins 3 CSI seuls"
        8 = "Patients asthmatiques - Au moins 1 Anti-IgE"
        9 = "Patients asthmatiques - Au moins 1 Anti-IL-5"
        10 = "Patients asthmatiques - Au moins 1 antileucotriène"
        11 = "Patients asthmatiques - Acte de thermoplastie bronchique codé lors d'une hospitalisation en MCO"
        12 = "Patients asthmatiques - Diagnostic d'asthme codé lors d'une hospitalisation"
        13 = "Patients asthmatiques - Diagnostic d'état de mal asthmatique codé lors d'une hospitalisation"
        14 = "Patients asthmatiques - Total"
        15 = "Population d'étude : Patients BPCO probable ou diagnostiquée hors asthmatiques"
        16 = "Patients pour lesquels la spirométrie ou les EFR ne sont pas réalisables - ALD alzheimer et autres démences"
        17 = "Patients pour lesquels la spirométrie ou les EFR ne sont pas réalisables - Hospitalisés en soins palliatifs"
        18 = "Patients pour lesquels la spirométrie ou les EFR ne sont pas réalisables - Hospitalisés plus de 90 jours en SSR
ou psychiatrie"
        19 = "Patients pour lesquels la spirométrie ou les EFR ne sont pas réalisables - Total"
        20 = "Patients avec un problème sur la date de décès"
        21 = "Population cible"
        22 = "Indicateur : Réalisation d'EFR ou de spirométrie annuelle "
        ;
value f_ind_04_flowchart
        1 = "Séjours en MCO avec un diagnostic de BPCO en DP"
        2 = "Séjours en MCO avec un diagnostic de BPCO en DAS et pneumopathie lobaire en DP"
        3 = "Séjours en MCO avec un diagnostic de BPCO en DAS et insuffisance respiratoire  aigue en DP"
        4 = "Séjours en MCO avec un diagnostic de BPCO en DAS et grippe en DP"
        5 = "Séjours en MCO avec un diagnostic de BPCO en DAS et infection des voies aériennes en DP"
        6 = "Séjours en MCO avec un diagnostic de BPCO en DAS et bronchopneumopathie en DP"
        7 = "Séjours en MCO avec un diagnostic de BPCO en DAS et pneumonie en DP"
        8 = "Séjours en MCO avec un diagnostic de BPCO en DAS et embolie pulmonaire en DP"
        9 = "Séjours en MCO avec un diagnostic de BPCO en DAS et insuffisance cardiaque aigue en DP"
        10 = "Séjours en MCO avec un diagnostic de BPCO en DAS et œdème aigue du poumon en DP"
        11 = "Séjours en MCO avec un diagnostic de BPCO en DAS et pneumothorax en DP"
        12 = "Population d'étude : Séjours pour exacerbation de BPCO de patients de 40 ans et plus"
        13 = "Séjours de patients pour lesquels le suivi médical précoce n'est pas possible - Séjours dont le mode de sortie
est le décès"
```

```
                14 = "Séjours de patients pour lesquels le suivi médical précoce n'est pas possible - Séjours suivis dans les 7 jours
d'une hospitalisation en MCO, SSR, HAD ou psychiatrie"
                15 = "Séjours de patients pour lesquels le suivi médical précoce n'est pas possible - Total"
                16 = "Séjours de patients avec un problème sur la date de décès"
                17 = "Population cible"
                18 = "Suivi pneumologue dans les 7 jours après hospitalisation"
                19 = "Suivi médecin généraliste dans les 7 jours après hospitalisation"
                20 = "Indicateur : Suivi médical précoce"
                ;
        value f_ind_05_flowchart
                1 = "Séjours en MCO avec un diagnostic de BPCO en DP"
                2 = "Séjours en MCO avec un diagnostic de BPCO en DAS et pneumopathie lobaire en DP"
                3 = "Séjours en MCO avec un diagnostic de BPCO en DAS et insuffisance respiratoire  aigue en DP"
                4 = "Séjours en MCO avec un diagnostic de BPCO en DAS et grippe en DP"
                5 = "Séjours en MCO avec un diagnostic de BPCO en DAS et infection des voies aériennes en DP"
                6 = "Séjours en MCO avec un diagnostic de BPCO en DAS et bronchopneumopathie en DP"
                7 = "Séjours en MCO avec un diagnostic de BPCO en DAS et pneumonie en DP"
                8 = "Séjours en MCO avec un diagnostic de BPCO en DAS et embolie pulmonaire en DP"
                9 = "Séjours en MCO avec un diagnostic de BPCO en DAS et insuffisance cardiaque aigue en DP"
                10 = "Séjours en MCO avec un diagnostic de BPCO en DAS et œdème aigue du poumon en DP"
                11 = "Séjours en MCO avec un diagnostic de BPCO en DAS et pneumothorax en DP"
                12 = "Population d'étude : Séjours pour exacerbation de BPCO de patients de 40 ans et plus"
                13 = "Séjours de patients pour lesquels le suivi médical à distance n'est pas possible - Séjours dont le mode de
sortie est le décès"
                14 = "Séjours de patients pour lesquels le suivi médical à distance n'est pas possible - Séjours suivis dans les 60
jours d'une hospitalisation en MCO, SSR, HAD ou psychiatrie"
                15 = "Séjours de patients pour lesquels le suivi médical à distance n'est pas possible - Total"
                16 = "Séjours de patients avec un problème sur la date de décès"
                17 = "Population cible"
                18 = "Indicateur : Suivi médical à distance"
                ;
        value f_ind_06_flowchart
                1 = "Séjours en MCO avec un diagnostic de BPCO en DP"
                2 = "Séjours en MCO avec un diagnostic de BPCO en DAS et pneumopathie lobaire en DP"
                3 = "Séjours en MCO avec un diagnostic de BPCO en DAS et insuffisance respiratoire  aigue en DP"
                4 = "Séjours en MCO avec un diagnostic de BPCO en DAS et grippe en DP"
                5 = "Séjours en MCO avec un diagnostic de BPCO en DAS et infection des voies aériennes en DP"
                6 = "Séjours en MCO avec un diagnostic de BPCO en DAS et bronchopneumopathie en DP"
                7 = "Séjours en MCO avec un diagnostic de BPCO en DAS et pneumonie en DP"
                8 = "Séjours en MCO avec un diagnostic de BPCO en DAS et embolie pulmonaire en DP"
                9 = "Séjours en MCO avec un diagnostic de BPCO en DAS et insuffisance cardiaque aigue en DP"
                10 = "Séjours en MCO avec un diagnostic de BPCO en DAS et œdème aigue du poumon en DP"
                11 = "Séjours en MCO avec un diagnostic de BPCO en DAS et pneumothorax en DP"
                12 = "Population d'étude : Séjours pour exacerbation de BPCO de patients de 40 ans et plus"
                13 = "Séjours de patients pour lesquels la recherche d'un traitement post hospitalisation n'est pas possible -
Séjours dont le mode de sortie est le décès"
                14 = "Séjours de patients pour lesquels la recherche d'un traitement post hospitalisation n'est pas possible -
Séjours suivis dans les 90 jours d'une hospitalisation en MCO, SSR, HAD ou psychiatrie"
                15 = "Séjours de patients pour lesquels la recherche d'un traitement post hospitalisation n'est pas possible -
Total"
                16 = "Séjours de patients avec un problème sur la date de décès"
                17 = "Population cible"
                18 = "Indicateur : Traitement de BDLA dans les 3 mois après hospitalisation"
                ;
        value f_ind_07_flowchart
                1 = "Patients avec un diagnostic de BPCO en DP lors d'un séjour en MCO"
                2 = "Patients avec un diagnostic de BPCO en DAS et pneumopathie lobaire en DP lors d'un séjour en MCO"
                3 = "Patients avec un diagnostic de BPCO en DAS et insuffisance respiratoire  aigue en DP lors d'un séjour en
MCO"
                4 = "Patients avec un diagnostic de BPCO en DAS et grippe en DP"
                5 = "Patients avec un diagnostic de BPCO en DAS et infection des voies aériennes en DP lors d'un séjour en MCO"
                6 = "Patients avec un diagnostic de BPCO en DAS et bronchopneumopathie en DP lors d'un séjour en MCO"
                7 = "Patients avec un diagnostic de BPCO en DAS et pneumonie en DP lors d'un séjour en MCO"
                8 = "Patients avec un diagnostic de BPCO en DAS et embolie pulmonaire en DP lors d'un séjour en MCO"
                9 = "Patients avec un diagnostic de BPCO en DAS et insuffisance cardiaque aigue en DP lors d'un séjour en MCO"
                10 = "Patients avec un diagnostic de BPCO en DAS et œdème aigue du poumon en DP lors d'un séjour en MCO"
                11 = "Patients avec un diagnostic de BPCO en DAS et pneumothorax en DP lors d'un séjour en MCO"
                12 = "Population d'étude : Patients de 40 ans et plus hospitalisés pour exacerbation de BPCO"
                13 = "Patients pour lesquels la réadaptation respiration n'est pas réalisable - Séjours index dont le mode de sortie
est le décès"
```

```
                    14 = "Patients pour lesquels la réadaptation respiration n'est pas réalisable - Séjours index suivis dans les 90 jours
d'une hospitalisation en MCO, HAD ou psychiatrie"
                    15 = "Patients pour lesquels la réadaptation respiration n'est pas réalisable - ALD Alzheimer et autres démences
active"
                    16 = "Patients pour lesquels la réadaptation respiration n'est pas réalisable - Total"
                    17 = "Patients avec un problème sur la date de décès"
                    18 = "Population cible"
                    19 = "Indicateur : Réalisation de réadaptation respiratoire dans les 3 mois après hospitalisation "
                    ;
          value f_ind_09_flowchart
                    1 = "Séjours en MCO avec un diagnostic de BPCO en DP"
                    2 = "Séjours en MCO avec un diagnostic de BPCO en DAS et pneumopathie lobaire en DP"
                    3 = "Séjours en MCO avec un diagnostic de BPCO en DAS et insuffisance respiratoire  aigue en DP"
                    4 = "Séjours en MCO avec un diagnostic de BPCO en DAS et grippe en DP"
                    5 = "Séjours en MCO avec un diagnostic de BPCO en DAS et infection des voies aériennes en DP"
                    6 = "Séjours en MCO avec un diagnostic de BPCO en DAS et bronchopneumopathie en DP"
                    7 = "Séjours en MCO avec un diagnostic de BPCO en DAS et pneumonie en DP"
                    8 = "Séjours en MCO avec un diagnostic de BPCO en DAS et embolie pulmonaire en DP"
                    9 = "Séjours en MCO avec un diagnostic de BPCO en DAS et insuffisance cardiaque aigue en DP"
                    10 = "Séjours en MCO avec un diagnostic de BPCO en DAS et œdème aigue du poumon en DP"
                    11 = "Séjours en MCO avec un diagnostic de BPCO en DAS et pneumothorax en DP"
                    12 = "Population d'étude : Séjours pour exacerbation de BPCO de patients de 40 ans et plus"
                    13 = "Séjours exclus - Séjours dont le mode de sortie est le décès"
                    14 = "Séjours exclus - Séjours de patients décédés dans les 180 jours et sans réhospitalisation"
                    15 = "Séjours exclus - Séjours précédés dans les 180 jours d'un séjour pour exacerbation"
                    16 = "Séjours exclus - Séjours avec un code diagnostic de sortie contre avis médical"
                    17 = "Séjours exclus - Séjours de patients hospitalisés dans les 180 jours en soins palliatifs"
                    18 = "Séjours exclus - Total"
                    18.5 = "Séjours de patients avec un problème sur la date de décès"
                    19 = "Population cible"
                    20 = "Indicateur : Réhospitalisation à 6 mois"
                    ;
          value f_DP_classe
                    1 = "BPCO"
                    2 = "Pneumopathie"
                    3 = "Insuffisance respiratoire aigüe"
                    4 = "Grippe"
                    5 = "Embolie pulmonaire"
                    6 = "Insuffisance cardiaque aigüe"
                    7 = "Pneumothorax"
                    ;
          value f_lieu_soin
                    1 = "Ville"
                    2 = "MCO"
                    3 = "SSR"
                    ;
          value f_indic_deces
                    1 = "Indicateur 01"
                    2 = "Indicateur 02"
                    3 = "Indicateur 03"
                    4 = "Indicateur 04"
                    5 = "Indicateur 05"
                    6 = "Indicateur 06"
                    7 = "Indicateur 07"
                    9 = "Indicateur 09"
                    99 = "Total"
                    ;
          value f_age_40_
                    0 = "Moins de 40 ans"
                    1 = "40 ans et plus"
                    ;
          picture commafr_0ch other = "000 000 000";
run;
```

3.1.2 Autoexec_Options_de_la_log_et_parametres.sas

```
/*
	*************************************************************************************************************
**********************************	*/
/*

*/
/*		Options de la log et paramètres du projet (Autoexec_Options_de_la_log_et_parametres.sas)

*/
/*

*/
/*
	*************************************************************************************************************
**********************************	*/

*
	*************************************************************************************************************
**********************************	;
*	Options de la log

;
*
	*************************************************************************************************************
**********************************	;

option nodate nonumber notes nosymbolgen ;
ods graphics on ;

option fmtsearch = (work library formats) nofmterr ;

*
	*************************************************************************************************************
**********************************	;
*	Définition des paramètres

;
*
	*************************************************************************************************************
**********************************	;

* Année pour laquelle l indicateur est calculé;
%let annee_N = 2017;
%let an_N = %sysevalf(&annee_N. - 2000);

* Définitions des années N-4, N-2, N-1 et N+1 pour les recerches dans le DCIR et le PMSI;
%let annee_4N = %sysevalf(&annee_N. - 4);
%let annee_2N = %sysevalf(&annee_N. - 2);
%let annee_1N = %sysevalf(&annee_N. - 1);
%let annee_N1 = %sysevalf(&annee_N. + 1);

* Dernière version de la carto disponible;
%let version_carto = G6;

* Exclusion des clés de chainage incorrects;
* Peuvent être modifiées à chaque nouvelle pseudonymisation;
%let cle_inc1 = xxxxxxxxxxxxxxxx;
%let cle_inc2 = BXXXXXXXXXXXXXXX;

*
	*************************************************************************************************************
**********************************	;
*	Définition des librairies

;
```

```
*
          ***************************************************************************************************
**********************************          ;

* Librairie contenant les tables de travail;
libname travail "/sasdata/prd/repbpcoh/data/Bases/travail";

* Librairie contenant les tables de populations;
libname pop "/sasdata/prd/repbpcoh/data/Bases/population";

* Librairie contenant les tables de résultats;
libname res "/sasdata/prd/repbpcoh/data/Bases/resultats";

* Librairie contenant les tables pour réaliser le flowchart;
libname flowch "/sasdata/prd/repbpcoh/data/Bases/flowchart";

* Librairie contenant les formats;
libname formats "/sasdata/prd/repbpcoh/data/Bases/formats";

* Librairie contenant les tables de vérifications;
libname verif "/sasdata/prd/repbpcoh/data/Bases/verifications";
```

3.1.3 Macro_Code_pratique_Arreter_un_script_si_erreur.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*

*/
/*		Macro pour arrêter un script si erreur : arret_erreur
*/
/*

*/
/*
	*****************************************************************************************************************
**********************************	*/

%macro arret_erreur;

	%if &syserr. > 0 %then
		%do;
			%abort;
		%end;

%mend arret_erreur;
```

3.1.4 Macro_Code_pratique_Proc_Freq.sas

```
/*
	*****************************************************************************************************
**********************************	*/
/*
                                                                                                        */
/*			Macro pour effectuer des vérifications (proc freq) au cours du projet

	*/
/*
                                                                                                        */
/*
	*****************************************************************************************************
**********************************	*/
/*
                                                                                                        */
/*			Arguments en entrée :

                                                                                                */
/*
                                                                                                        */
/*				- in_tbl = le nom de la table en entrée (peut contenir une librairie)

	*/
/*
                                                                                                        */
/*				- out_tbl = le nom de la table en sortie (stockée dans la librairie "verif", commençant par _ + numéro
de la vérif	*/
/*
                                                                                                        */
/*				- list_var_in = la ou les variable(s) à analyser (si plusieurs variables, les séparer par des *)
                                                                        */
/*
                                                                                                        */
/*				- list_var_out = la liste de variables à conserver en sortie (inutile sur une seule variable dans list_var_in)
                                                */
/*
                                                                                                        */
/*
	*****************************************************************************************************
**********************************	*/

%global tmp_num_tab;

%let tmp_num_tab = 1;

%macro proc_freq(in_tbl=, out_tbl=, list_var_in=, list_var_out=);

	%if &tmp_num_tab. < 10 %then
		%let num_tab = 00&tmp_num_tab.;

	%else %if &tmp_num_tab. < 100 %then
		%let num_tab = 0&tmp_num_tab.;

	%else
		%let num_tab = &tmp_num_tab.;

	%if %sysfunc(index(&list_var_in., *)) >= 1 %then
		%do;
```

```
                                ods exclude all;
                                ods output Crosstabfreqs = verif._&num_tab.&out_tbl.;
                                proc freq data = &in_tbl.;
                                  table &list_var_in. / missing;
                                run;

                                data verif._&num_tab.&out_tbl. (keep = &list_var_out.);
                                        set verif._&num_tab.&out_tbl. (where = (index(_TYPE_, "0") = 0 and Frequency > 0));
                                run;

                                proc sql noprint; SELECT SUM(Frequency) INTO: nb_tot FROM verif._&num_tab.&out_tbl.; quit;

                                data verif._&num_tab.&out_tbl.;
                                        set verif._&num_tab.&out_tbl.;
                                        length percent 4.;
                                        format percent nlpct7.1 Frequency commafr_0ch.;
                                        percent = Frequency/&nb_tot.;
                                run;
                                ods exclude none;

                        %end;

                %else
                        %do;

                                ods exclude all;
                                ods output Onewayfreqs = verif._&num_tab.&out_tbl.;
                                proc freq data = &in_tbl.;
                                  table &list_var_in.;
                                run;

                                proc sql noprint; SELECT SUM(Frequency) INTO: nb_tot FROM verif._&num_tab.&out_tbl.; quit;

                                data verif._&num_tab.&out_tbl. (keep = F_&list_var_in. Frequency);
                                        set verif._&num_tab.&out_tbl. (where = (Frequency > 0));
                                run;

                                data verif._&num_tab.&out_tbl.;
                                        set verif._&num_tab.&out_tbl.;
                                        length percent 4.;
                                        format percent nlpct7.1 Frequency commafr_0ch.;
                                        percent = Frequency/&nb_tot.;
                                run;
                                ods exclude none;

                        %end;

                %let tmp_num_tab = %sysevalf(&tmp_num_tab. + 1);

%mend proc_freq;
```

3.1.5 Macro_Code_pratique_Supprimer_une_table.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*		Macro pour supprimer une table si elle existe : suppr_table

			*/
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*		Arguments en entrée :

                                                                                                        */
/*
                                                                                                                    */
/*			- lib = le nom de la librairie dans laquelle est stockée la table

	*/
/*			  Par défaut : work

                                                                                                        */
/*
                                                                                                                    */
/*			- table = le nom de la table à supprimer

                                                */
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/

%macro suppr_table(lib = work, table=);

	%if %sysfunc(exist(&lib..&table.)) = 1 %then
		%do;

			proc delete data = &lib..&table.;
			run;

		%end;

	%else
		%put La table &lib..&table. n existe pas !;

%mend suppr_table;

/* Exemple d appel */
/*
%suppr_table(
	lib = work,
	table = test
	);
*/
```

3.1.6 Macro_Code_pratique_Verifier_l_existence_d_une_variable.sas

```
/*
	*************************************************************************************************************
**********************************	*/
/*
                                                                                                                */
/*		Macro pour supprimer une table si elle existe : suppr_table
                                                                                                                */
/*
                                                                                                                */
/*
	*************************************************************************************************************
**********************************	*/
/*
                                                                                                                */
/*		Arguments en entrée :
                                                                                                                */
/*
                                                                                                                */
/*			- lib = le nom de la librairie dans laquelle est stockée la table
	*/
/*			  Par défaut : work
                                                                                                                */
/*
                                                                                                                */
/*			- table = le nom de la table à supprimer
                                                                                                                */
/*
                                                                                                                */
/*
	*************************************************************************************************************
**********************************	*/

%macro suppr_table(lib = work, table=);

	%if %sysfunc(exist(&lib..&table.)) = 1 %then
		%do;

			proc delete data = &lib..&table.;
			run;

		%end;

	%else
		%put La table &lib..&table. n existe pas !;

%mend suppr_table;

/* Exemple d appel */
/*
%suppr_table(
	lib = work,
	table = test
	);
*/
```

3.1.7 Macro_Reperage_des_Soins_CCAM_DCIR.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*			Macro pour repérer les remboursement des codes CCAM souhaités : extract_CCAM_DCIR
                                                                                                                    */
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*			Arguments en entrée :
                                                                                                                    */
/*
                                                                                                                    */
/*				- annee_deb = la première année de soins (ie. filtre sur EXE_SOI_DTD) pour laquelle on souhaite
repérer les codes                                                */
/*
                                                                                                                    */
/*				- annee_fin = la dernière année de soins (ie. filtre sur EXE_SOI_DTD) pour laquelle on souhaite repérer
les codes                                                */
/*
                                                                                                                    */
/*				- tbl_out = le nom de la table en sortie (peut contenir un nom de librairie)
                                                                                                                    */
/*
                                                                                                                    */
/*				- tbl_codes = la table contenant les codes CCAM à repérer (au format Oracle)
                                                                                                                    */
/*
                                                                                                                    */
/*				- tbl_patients = le nom de la table contenant la correspondance BEN_IDT_ANO <-> BEN_NIR_PSA (au
format Oracle)                                                */
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*			Tables en sortie : &tbl_out. (renseignée en paramètres de la macro)

                                                                                                                    */
/*			avec sélection des codes disponibles dans le référentiel, filtrée sur les soins de ville uniquement, les NIR
"normaux" + suppression	*/
/*			des lignes pour information
                                                                                                                    */
/*
                                                                                                                    */
```

```
/*
	*****************************************************************************************************
**********************************	*/

%macro extract_CCAM_DCIR(annee_deb=, annee_fin=, tbl_out=, tbl_codes=, tbl_patients=);

	%let annee_fin_flx = %sysevalf(&annee_fin. + 1);

	* On boucle sur les années de repérage (en flux);
	%do Annee = &annee_deb. %to &annee_fin_flx.;

		* On boucle sur les 12 mois de l année;
		%do i = 1 %to 12;

			%if &i. < 10 %then %let Mois = 0&i.;
			%else %let Mois = &i.;

			%put Données de &Mois./&Annee.;

			proc sql;

				%connectora;

					CREATE TABLE tmp_extract_CCAM_DCIR_&Annee._&Mois. AS
						SELECT * FROM CONNECTION TO ORACLE (
							SELECT
								pop.BEN_IDT_ANO,
								prs.EXE_SOI_DTD,
								prs.EXE_SOI_DTF,
								prs.PRS_ACT_QTE,
								prs.BSE_REM_MNT,
								cam.CAM_PRS_IDE,
								ete.ETB_EXE_FIN,
								ete.ETE_MCO_DDP,
								cod.*
							FROM &tbl_patients. pop
								INNER JOIN ER_PRS_F prs
									ON	pop.BEN_NIR_PSA =
prs.BEN_NIR_PSA
								INNER JOIN ER_CAM_F cam
									ON	cam.FLX_DIS_DTD =
prs.FLX_DIS_DTD
									AND	cam.FLX_TRT_DTD =
prs.FLX_TRT_DTD
									AND	cam.FLX_EMT_TYP =
prs.FLX_EMT_TYP
									AND	cam.FLX_EMT_NUM =
prs.FLX_EMT_NUM
									AND	cam.FLX_EMT_ORD =
prs.FLX_EMT_ORD
									AND	cam.ORG_CLE_NUM =
prs.ORG_CLE_NUM
									AND	cam.DCT_ORD_NUM =
prs.DCT_ORD_NUM
									AND	cam.PRS_ORD_NUM =
prs.PRS_ORD_NUM
									AND	cam.REM_TYP_AFF =
prs.REM_TYP_AFF
								INNER JOIN &tbl_codes. cod
									ON TRIM(cam.CAM_PRS_IDE) =
TRIM(cod.code_CCAM)
								LEFT JOIN ER_ETE_F ete
									ON	ete.FLX_DIS_DTD =
prs.FLX_DIS_DTD
									AND	ete.FLX_TRT_DTD =
prs.FLX_TRT_DTD
									AND	ete.FLX_EMT_TYP =
prs.FLX_EMT_TYP
									AND	ete.FLX_EMT_NUM =
prs.FLX_EMT_NUM
```

```
                                                                                AND         ete.FLX_EMT_ORD =
prs.FLX_EMT_ORD
                                                                                AND         ete.ORG_CLE_NUM =
prs.ORG_CLE_NUM
                                                                                AND         ete.DCT_ORD_NUM =
prs.DCT_ORD_NUM
                                                                                AND         ete.PRS_ORD_NUM =
prs.PRS_ORD_NUM
                                                                                AND         ete.REM_TYP_AFF =
prs.REM_TYP_AFF
                                                                        WHERE CAM_ACT_COD = '1' and CAM_TRT_PHA IN (0, 1)
and prs.BEN_CDI_NIR = '00' AND prs.CPL_MAJ_TOP IN (0, 1)
                                                                                AND (ete.ETE_IND_TAA NOT IN (1, 2) OR
ete.ETE_IND_TAA IS NULL)
                                                                                AND prs.DPN_QLF NOT IN (71, 72) AND
NOT(prs.DPN_QLF = 0 AND prs.PRS_DPN_QLP = 71)
                                                                                AND prs.FLX_DIS_DTD =
TO_DATE(%str(%'&Annee.&Mois.01%'), 'YYYYMMDD')
                                                                                AND prs.EXE_SOI_DTD BETWEEN
TO_DATE(%str(%'&Annee_deb.0101%'), 'YYYYMMDD') AND TO_DATE(%str(%'&Annee_fin.1231%'), 'YYYYMMDD')
                                                                        );

                                        disconnect from oracle;

                                quit;

                                %arret_erreur;

                                * On empile toutes les tables mensuelles;
                                %if %sysfunc(exist(&tbl_out.)) = 1 %then
                                        %do;

                                                proc append base = &tbl_out. data = tmp_extract_CCAM_DCIR_&annee._&mois.;
                                                run;

                                                %arret_erreur;

                                                proc delete data = tmp_extract_CCAM_DCIR_&annee._&mois.;
                                                run; quit;

                                        %end;

                                %if %sysfunc(exist(&tbl_out.)) = 0 %then
                                        %do;

                                                data &tbl_out.;
                                                        set tmp_extract_CCAM_DCIR_&annee._&mois.;
                                                run;

                                                %arret_erreur;

                                                proc delete data = tmp_extract_CCAM_DCIR_&annee._&mois.;
                                                run; quit;

                                        %end;

                        %end;
                        * Fin de la boucle sur les 12 mois;

                %end;
                * Fin de la boucle par année;

                * On applique le programme de régularisations pour céer la table finale;
                proc sql undo_policy = none;

                        CREATE TABLE &tbl_out. AS
                                SELECT
                                        BEN_IDT_ANO,
                                        datepart(EXE_SOI_DTD) AS date_debut length = 4 format ddmmyy10.,
                                        CASE    WHEN datepart(EXE_SOI_DTF) IS NULL THEN datepart(EXE_SOI_DTD)
                                                        ELSE datepart(EXE_SOI_DTF)
```

```
                                                    END AS date_fin length = 4 format ddmmyy10.,
                                    "DCIR" AS type,
                                    CAM_PRS_IDE,
                                    reperage,
                                    ETB_EXE_FIN,
                                    ETE_MCO_DDP,
                                    SUM(BSE_REM_MNT) AS montant,
                                    SUM(PRS_ACT_QTE) AS quantite
                            FROM &tbl_out.
                            GROUP BY 1, 2, 3, 4, 5, 6, 7, 8;

                    DELETE FROM &tbl_out. WHERE montant <= 0 OR quantite < 0;

            quit;

            %arret_erreur;

%mend extract_CCAM_DCIR;
```

3.1.8 Macro_Reperage_des_soins_CCAM_PMSI.sas

```
/*
        ********************************************************************************************************************
**********************************        */
/*

                                                                                                                        */
/*                      Macro pour repérer les remboursement des codes CCAM souhaités, dans le PMSI : extract_CCAM_PMSI
                                                                                                                        */
/*

                                                                                                                        */
/*
        ********************************************************************************************************************
**********************************        */
/*

                                                                                                                        */
/*                      Arguments en entrée :

                                                                                                        */
/*

                                                                                                                        */
/*                              - annee_deb = la première année de soins (ie. année des tables PMSI) pour laquelle on souhaite
repérer les codes                                                       */
/*

                                                                                                                        */
/*                              - annee_fin = la dernière année de soins (ie. année des tables PMSI) pour laquelle on souhaite repérer
les codes                                                       */
/*

                                                                                                                        */
/*                              - tbl_out = le nom de la table en sortie (peut contenir un nom de librairie)
                                                                                                                */
/*

                                                                                                                        */
/*                              - tbl_codes = la table contenant les codes CCAM à repérer (au format Oracle)
                                                                                                                        */
/*

                                                                                                                        */
/*                              - tbl_patients = le nom de la table contenant la correspondance BEN_IDT_ANO <-> BEN_NR_PSA (au
format Oracle)                                                  */
/*

                                                                                                                        */
/*                              - HAD = flag (0/1) : si 1, on recherche les codes CCAM dans le PMSI-HAD (Par défaut : 1)
                                                                                                                */
/*

                                                                                                                        */
/*                              - MCO = flag (0/1) : si 1, on recherche les codes CCAM dans le PMSI-MCO (sejours et ACE) (Par défaut :
1)                                                              */
/*

                                                                                                                        */
/*                              - RIP = flag (0/1) : si 1, on recherche les codes CCAM dans le PMSI-RIP (Par défaut : 1)
                                                                                                        */
/*

                                                                                                                        */
/*                              - SSR = flag (0/1) : si 1, on recherche les codes CCAM dans le PMSI-SSR (sejours et ACE) (Par défaut : 1)
                                                                                                        */
```

```
/*
                                                                                                  */
/*	*****************************************************************************************************
*************************************	*/
/*
                                                                                                  */
/*	Tables en sortie : &tbl_out. (renseignée en paramètres de la macro)
	*/
/*	avec sélection des codes disponibles dans le référentiel &tbl_codes., hors séjours en erreur
	*/
/*
                                                                                                  */
/*	*****************************************************************************************************
*************************************	*/

*	*****************************************************************************************************
*************************************	;
*	Macro pour repérer les codes CCAM dans le PMSI - séjours
	;
*	*****************************************************************************************************
*************************************	;

%macro CCAM_sejours(an_deb_pmsi=, an_fin_pmsi=, an_crea_t=, an_fin_t=, tbl_codes=, tbl_patients=, table=, domaine=, var_CCAM=,
var_qte=, tbl_out=);

	%do i = %sysfunc(max(&an_deb_pmsi., &an_crea_t. )) %to %sysfunc(min(&an_fin_pmsi., &an_fin_t.));

		%if &i. < 10 %then %let an = 0&i.;
		%else %let an = &i.;

		* On définit les variables de jointure pour chaque domaine du PMSI;
		%if &domaine. = HAD %then
			%do;
				%let var_join1 = ETA_NUM_EPMSI;
				%let var_join2 = RHAD_NUM;
				%if &table. = A %then
					%do;
						%let filtre_activite = ACT_COD = '1' AND PHA_COD IN ('0', '1') AND ;
					%end;
			%end;
		%if &domaine. = MCO %then
			%do;
				%let var_join1 = ETA_NUM;
				%let var_join2 = RSA_NUM;
				%let filtre_activite = ACV_ACT = '1' AND PHA_ACT IN ('0', '1') AND ;
			%end;
		%if &domaine. = RIP %then
			%do;
				%let var_join1 = ETA_NUM_EPMSI;
				%let var_join2 = RIP_NUM;
				%let filtre_activite = ACV_ACT = '1' AND PHA_ACT IN ('0', '1') AND ;
			%end;
		%if &domaine. = SSR %then
			%do;
				%let var_join1 = ETA_NUM;
				%let var_join2 = RHA_NUM;
				%let filtre_activite = CCAM_COD_ACT = '1' AND CCAM_PHA_ACT IN ('0', '1') AND ;
			%end;

		* On joint la table CCAM avec la table des séjours en filtrant sur les codes CCAM sélectionnés;
		%put Récupération des séjours du PMSI-&domaine. (table &table.) contenant des actes CCAM pré-définis - Année
&an. ...;
```

```
proc sql;
%connectora;
CREATE TABLE ext_CCAM_PMSI_&domaine._&table._&an. AS SELECT * FROM CONNECTION
TO ORACLE (
SELECT DISTINCT
c.&var_join1.,
c.&var_join2.,
e.ETB_EXE_FIN,
pop.BEN_IDT_ANO,
c.EXE_SOI_DTD,
c.EXE_SOI_DTF,
20&an. AS annee,
&var_qte. AS nb_deliv,
%if &domaine. = MCO %then
%do;
b.GRG_GHM,
um.AUT_TYP1_UM AS type_UM,
um.AUT_TYP2_UM AS type_UM2,
b.SEJ_NBJ,
%end;
%if &domaine. = SSR %then
%do;
SUBSTR(b.HOS_TYP_UM, 1, 2) AS
type_UM,
%end;
cod.*
FROM &tbl_patients. pop
INNER JOIN T_&domaine.&an.C c
ON pop.BEN_NIR_PSA =
c.NIR_ANO_17
%if &domaine. = MCO or &domaine. = HAD or &domaine. =
SSR %then
%do;
INNER JOIN T_&domaine.&an.B b
ON c.&var_join1. =
b.&var_join1.
AND c.&var_join2. =
b.&var_join2.
%end;
INNER JOIN T_&domaine.&an.E e
ON c.&var_join1. = e.ETA_NUM
INNER JOIN T_&domaine.&an.&table. ccam
ON c.&var_join1. = ccam.&var_join1.
AND c.&var_join2. = ccam.&var_join2.
%if &domaine. = MCO %then
%do;
INNER JOIN T_&domaine.&an.UM um
ON c.&var_join1. =
um.&var_join1.
AND c.&var_join2. =
um.&var_join2.
%end;
INNER JOIN &tbl_codes. cod
ON TRIM(ccam.&var_CCAM.) =
TRIM(cod.code_CCAM)
WHERE &filtre_activite.
c.NIR_ANO_17 NOT IN ('&cle_inc1.', '&cle_inc2.') AND
&var_qte. > 0
%if &domaine. = HAD %then
%do;
AND NIR_RET = '0' AND NAI_RET = '0'
AND SEX_RET = '0' AND SEJ_RET = '0' AND FHO_RET = '0'
AND PMS_RET = '0' AND DAT_RET =
'0' %if &an. > 12 %then %do; AND COH_NAI_RET = '0'
AND COH_SEX_RET = '0' %end;
%end;
%if &domaine. = MCO %then
%do;
```

```
                                                                                AND NIR_RET = '0' AND NAI_RET = '0'
AND SEX_RET = '0' AND SEJ_RET = '0' AND FHO_RET = '0'
                                                                                AND PMS_RET = '0' AND DAT_RET =
'0' %if &an. > 12 %then %do; AND COH_NAI_RET = '0'
                                                                                AND COH_SEX_RET = '0' %end;
                                                                                AND c.ETA_NUM NOT IN
('130780521', '130783236', '130783293', '130784234', '130804297', '600100101',
                                                                                        '750041543', '750100018',
'750100042', '750100075', '750100083', '750100091', '750100109',
                                                                                        '750100125', '750100166',
'750100208', '750100216', '750100232', '750100273', '750100299',
                                                                                        '750801441', '750803447',
'750803454', '910100015', '910100023', '920100013', '920100021',
                                                                                        '920100039', '920100047',
'920100054', '920100062', '930100011', '930100037', '930100045',
                                                                                        '940100027', '940100035',
'940100043', '940100050', '940100068', '950100016', '690783154',
                                                                                        '690784137', '690784152',
'690784178', '690787478', '830100558')
                                                                                AND GRG_GHM NOT IN ('90Z00Z')
AND GRG_RET NOT IN ('024') AND GRG_GHM NOT IN ('90H01Z', '90Z00Z',
                                                                                        '90Z01Z', '90Z02Z',
'90Z03Z')
                                                                                AND ((SEJ_TYP = 'A' OR SEJ_TYP IS
NULL) OR (SEJ_TYP = 'B' AND GRG_GHM NOT IN ('28Z14Z','28Z15Z','28Z16Z')))
                                                                        %end;
                                                                %if &domaine. = RIP %then
                                                                        %do;
                                                                                AND NIR_RET = '0' AND NAI_RET = '0'
AND SEX_RET = '0' AND SEJ_RET = '0' AND FHO_RET = '0'
                                                                                AND PMS_RET = '0' AND DAT_RET =
'0' %if &an. > 12 %then %do; AND COH_NAI_RET = '0'
                                                                                AND COH_SEX_RET = '0' %end;
                                                                        %end;
                                                                %if &domaine. = SSR %then
                                                                        %do;
                                                                                AND NIR_RET = '0' AND NAI_RET = '0'
AND SEX_RET = '0' AND SEJ_RET = '0' AND FHO_RET = '0'
                                                                                AND PMS_RET = '0' AND DAT_RET =
'0' %if &an. > 12 %then %do; AND COH_NAI_RET = '0'
                                                                                AND COH_SEX_RET = '0' %end;
                                                                        %end;
                                                        );

                        DISCONNECT FROM ORACLE;

                quit;

                data ext_CCAM_PMSI_&domaine._&table._&an.;
                        set ext_CCAM_PMSI_&domaine._&table._&an.;
                        length date_debut date_fin 4.;
                        domaine = "&domaine.";
                        table = "&table.";
                        type = "PMSI séjours";
                        date_debut = datepart(EXE_SOI_DTD);
                        date_fin = datepart(EXE_SOI_DTF);
                        drop EXE_SOI_DTD EXE_SOI_DTF;
                        format date_debut date_fin ddmmyy10.;
                run;

                %arret_erreur;

        %end;

%mend CCAM_sejours;

*
        ************************************************************************************************************
**********************************          ;
```

```
*           Macro pour repérer les codes CCAM dans le PMSI - ACE

                                        ;
*
          ***************************************************************************************************
************************************          ;

%macro CCAM_ACE(an_deb_pmsi=, an_fin_pmsi=, an_crea_t=, an_fin_t=, tbl_codes=, table=, domaine=, var_CCAM=, var_qte=,
tbl_out=, tbl_patients=);

          %do i = %sysfunc(max(&an_deb_pmsi., &an_crea_t.)) %to %sysfunc(min(&an_fin_pmsi., &an_fin_t.));

                    %if &i. < 10 %then %let an = 0&i.;
                    %else %let an = &i.;

                    * On définit les variables de jointure;
                    %let var_join1 = ETA_NUM;
                    %let var_join2 = SEQ_NUM;

                    * On définit le filtre sur le code activité;
                    %let filtre_activite = ACV_ACT = '1' AND PHA_ACT IN ('0', '1') AND ;


                    * On joint la table CCAM avec la table des ACE en filtrant sur les codes CCAM sélectionnés;
                    %put Récupération des séjours du PMSI-&domaine. (table &table.) contenant des actes CCAM pré-définis - Année
&an. ...;

                    proc sql;

                              %connectora;

                                        CREATE TABLE ext_CCAM_PMSI_&domaine._&table._&an. AS SELECT * FROM CONNECTION
TO ORACLE (
                                                  SELECT DISTINCT
                                                            c.&var_join1.,
                                                            c.&var_join2.,
                                                            pop.BEN_IDT_ANO,
                                                            c.EXE_SOI_DTD,
                                                            c.EXE_SOI_DTF,
                                                            20&an. AS annee,
                                                            &var_qte. AS nb_deliv,
                                                            cod.*
                                                  FROM &tbl_patients. pop
                                                            INNER JOIN T_&domaine.&an.CSTC c
                                                                      ON          pop.BEN_NIR_PSA =
c.NIR_ANO_17
                                                            INNER JOIN T_&domaine.&an.&table. ccam
                                                                      ON          c.&var_join1. = ccam.&var_join1.
                                                                      AND         c.&var_join2. = ccam.&var_join2.
                                                            INNER JOIN &tbl_codes. cod
                                                                      ON TRIM(ccam.&var_CCAM.) =
TRIM(cod.code_CCAM)
                                                  WHERE &filtre_activite.
                                                            c.NIR_ANO_17 NOT IN ('&cle_inc1.', '&cle_inc2.') AND
&var_qte. > 0
                                                            AND c.EXE_SOI_DTD IS NOT NULL AND c.EXE_SOI_DTF IS
NOT NULL
                                                            %if &domaine. = MCO %then
                                                                      %do;
                                                                                AND NIR_RET = '0' AND NAI_RET = '0'
AND SEX_RET = '0' AND IAS_RET = '0' AND ENT_DAT_RET = '0'
                                                                                AND c.ETA_NUM NOT IN
('130780521', '130783236', '130783293', '130784234', '130804297', '600100101',
                                                                                          '750041543', '750100018',
'750100042', '750100075', '750100083', '750100091', '750100109',
                                                                                          '750100125', '750100166',
'750100208', '750100216', '750100232', '750100273', '750100299',
                                                                                          '750801441', '750803447',
'750803454', '910100015', '910100023', '920100013', '920100021',
                                                                                          '920100039', '920100047',
'920100054', '920100062', '930100011', '930100037', '930100045',
```

```
'940100027', '940100035',
'940100043', '940100050', '940100068', '950100016', '690783154',
'690784137', '690784152',
'690784178', '690787478', '830100558')
                                        %end;
                                );

                        DISCONNECT FROM ORACLE;

                quit;

                data ext_CCAM_PMSI_&domaine._&table._&an.;
                        set ext_CCAM_PMSI_&domaine._&table._&an.;
                        length date_debut date_fin 4.;
                        domaine = "&domaine.";
                        table = "&table.";
                        type = "PMSI ACE";
                        date_debut = datepart(EXE_SOI_DTD);
                        date_fin = datepart(EXE_SOI_DTF);
                        format date_debut date_fin ddmmyy10.;
                        drop EXE_SOI_DTD EXE_SOI_DTF;
                        if (year(date_fin) = 20&an. AND 0 <= yrdif(date_debut, date_fin, 'act/act') <= 1) then output;
                run;

                %arret_erreur;

        %end;

%mend CCAM_ACE;

*
        ****************************************************************************************************
************************************            ;
*       Macro finale pour appeler les précédentes, pour chaque table du PMSI

        ;
*
        ****************************************************************************************************
************************************            ;

%macro extract_CCAM_PMSI(annee_deb=, annee_fin=, HAD = 1, MCO = 1, RIP = 1, SSR = 1, tbl_out=, tbl_codes=, tbl_patients=);

        %let an_deb_pmsi = %sysevalf(&annee_deb. - 2000);
        %let an_fin_pmsi = %sysevalf(&annee_fin. - 2000);

        * Pour le PMSI-HAD;
        %if &HAD. = 1 %then
                %do;

                        %let domaine = HAD;

                        * Dans les tables A;
                        %CCAM_sejours(
                                an_deb_pmsi = &an_deb_pmsi.,
                                an_fin_pmsi = &an_fin_pmsi.,
                                an_crea_t = 10,
                                an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY()))) - 2000),
                                tbl_codes = &tbl_codes.,
                                table = A,
                                domaine = &domaine.,
                                var_CCAM = CCAM_COD,
                                var_qte = REAL_NBR,
                                tbl_out = &tbl_out.,
                                tbl_patients = &tbl_patients.
                                );

                %end;
                * Fin du PMSI-HAD;

        * Pour le PMSI-MCO;
        %if &MCO. = 1 %then
```

```
		%do;

			%let domaine = MCO;

			* Dans les tables A;
			%CCAM_sejours(
				an_deb_pmsi = &an_deb_pmsi.,
				an_fin_pmsi = &an_fin_pmsi.,
				an_crea_t = 06,
				an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY()))) - 2000),
				tbl_codes = &tbl_codes.,
				table = A,
				domaine = &domaine.,
				var_CCAM = CDC_ACT,
				var_qte = NBR_EXE_ACT,
				tbl_out = &tbl_out.,
				tbl_patients = &tbl_patients.
				);

			* Dans les tables FMSTC;
			%CCAM_ACE(
				an_deb_pmsi = &an_deb_pmsi.,
				an_fin_pmsi = &an_fin_pmsi.,
				an_crea_t = 09,
				an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY()))) - 2000),
				tbl_codes = &tbl_codes.,
				table = FMSTC,
				domaine = &domaine.,
				var_CCAM = CCAM_COD,
				var_qte = 1,
				tbl_out = &tbl_out.,
				tbl_patients = &tbl_patients.
				);

		%end;
		* Fin du PMSI-MCO;

* Pour le PMSI-RIP;
%if &RIP. = 1 %then
		%do;

			%let domaine = RIP;

			* Dans les tables CCAM;
			%CCAM_sejours(
				an_deb_pmsi = &an_deb_pmsi.,
				an_fin_pmsi = &an_fin_pmsi.,
				an_crea_t = 17,
				an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY()))) - 2000),
				tbl_codes = &tbl_codes.,
				table = CCAM,
				domaine = &domaine.,
				var_CCAM = CDC_ACT,
				var_qte = 1,
				tbl_out = &tbl_out.,
				tbl_patients = &tbl_patients.
				);

		%end;
		* Fin du PMSI-RIP;

* Pour le PMSI-SSR;
%if &SSR. = 1 %then
		%do;

			%let domaine = SSR;

			* Dans les tables CCAM;
			%CCAM_sejours(
				an_deb_pmsi = &an_deb_pmsi.,
				an_fin_pmsi = &an_fin_pmsi.,
```

```
                                        an_crea_t = 09,
                                        an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
                                        tbl_codes = &tbl_codes.,
                                        table = CCAM,
                                        domaine = &domaine.,
                                        var_CCAM = CCAM_ACT,
                                        var_qte = CCAM_NBR_REA,
                                        tbl_out = &tbl_out.,
                                        tbl_patients = &tbl_patients.
                                        );

                                * Dans les tables FMSTC;
                                %CCAM_ACE(
                                        an_deb_pmsi = &an_deb_pmsi.,
                                        an_fin_pmsi = &an_fin_pmsi.,
                                        an_crea_t = 13,
                                        an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
                                        tbl_codes = &tbl_codes.,
                                        table = FMSTC,
                                        domaine = &domaine.,
                                        var_CCAM = CCAM_COD,
                                        var_qte = 1,
                                        tbl_out = &tbl_out.,
                                        tbl_patients = &tbl_patients.
                                        );

                        %end;
                        * Fin du PMSI-SSR;

        * Empilage de la table temporaire à la table Résultat;
        data &tbl_out.;
                        format table $10.;
                        set ext_CCAM_PMSI_:;
        run;

        %arret_erreur;

        * Suppression de la table temporaire;
        proc datasets library = work memtype = data nolist;
                        delete ext_CCAM_PMSI_:;
        run;

%mend extract_CCAM_PMSI;
```

3.1.9 Macro_Reperage_des_soins_CIM10_ALD.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*		Macro pour repérer les périodes d exonération d ALD contenant les codes CIM souhaités : extract_ALD_CIM
                                                                                                                    */
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*		Arguments en entrée :
                                                                                                                    */
/*
                                                                                                                    */
/*			- tbl_out = le nom de la table en sortie (peut contenir un nom de librairie)
                                                                                                                    */
/*
                                                                                                                    */
/*			- tbl_codes = la table contenant les codes CIM à repérer (au format Oracle)
                                                                                                                    */
/*
                                                                                                                    */
/*			- tbl_patients = le nom de la table contenant la correspondance BEN_IDT_ANO <-> BEN_NR_PSA (au
format Oracle)                                                                                                      */
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*		Tables en sortie : &out. (renseignée en paramètres de la macro)
                                                                                                                    */
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/

%macro extract_ALD_CIM(tbl_out=, tbl_codes=, tbl_patients=);

	proc sql;

		%connectora;

			CREATE TABLE &tbl_out. AS SELECT * FROM CONNECTION TO ORACLE (
				SELECT
					c.BEN_IDT_ANO,
					a.IMB_ETM_NAT,
					a.IMB_ALD_DTD,
					a.IMB_ALD_DTF,
					a.INS_DTE,
```

```
                                                    a.UPD_DTE,
                                                    a.IMB_ALD_NUM,
                                                    a.MED_MTF_COD,
                                                    b.*
                                                FROM &tbl_patients. c, IR_IMB_R a, &tbl_codes. b
                                                WHERE c.BEN_NIR_PSA = a.BEN_NIR_PSA AND SUBSTR(TRIM(a.MED_MTF_COD),
1, taille) = TRIM(b.code_CIM)
                                                    AND IMB_ETM_NAT IN (41, 43, 45) AND a.INS_DTE <
TO_DATE(%str(%'&Annee_N1.0501%'), 'YYYYMMDD')
                                                );

                DISCONNECT FROM ORACLE;

        quit;

        proc sort data = &tbl_out.;
                by BEN_IDT_ANO IMB_ETM_NAT IMB_ALD_NUM MED_MTF_COD INS_DTE UPD_DTE IMB_ALD_DTF descending
IMB_ALD_DTD;
        run;

        data &tbl_out.;
                set &tbl_out.;
                by BEN_IDT_ANO IMB_ETM_NAT IMB_ALD_NUM MED_MTF_COD INS_DTE UPD_DTE IMB_ALD_DTF descending
IMB_ALD_DTD;
                if last.MED_MTF_COD then
                        output;
        run;

        data &tbl_out.;
                set &tbl_out.;
                length date_debut date_fin 4.;
                type = "ALD";
                date_debut = datepart(IMB_ALD_DTD);
                date_fin = datepart(IMB_ALD_DTF);
                format date_debut date_fin ddmmyy10.;
                drop IMB_ALD_DTD IMB_ALD_DTF;
        run;

        %arret_erreur;

%mend extract_ALD_CIM;
```

3.1.10 Macro_Reperage_des_soins_CIM10_PMSI.sas

```
/*
	*****************************************************************************************************
*************************************		*/
/*
									*/
/*			Macro pour repérer les remboursement des codes diagnostics souhaités, dans le PMSI : extract_CIM10_PMSI
							*/
/*
									*/
/*
	*****************************************************************************************************
*************************************		*/
/*
									*/
/*			Arguments en entrée :
							*/
/*
									*/
/*				- annee_deb = la première année de soins (ie. année des tables PMSI) pour laquelle on souhaite
repérer les codes				*/
/*
									*/
/*				- annee_fin = la dernière année de soins (ie. année des tables PMSI) pour laquelle on souhaite repérer
les codes				*/
/*
									*/
/*				- tbl_out = le nom de la table en sortie (peut contenir un nom de librairie)
								*/
/*
									*/
/*				- tbl_codes = la table contenant les codes CIM10 à repérer (au format Oracle)
									*/
/*
									*/
/*				- tbl_patients = le nom de la table contenant la correspondance BEN_IDT_ANO <-> BEN_NR_PSA (au
format Oracle)					*/
/*
									*/
/*				- HAD_DP = flag (0/1) : si 1, on recherche les codes CIM10 dans les DP des séjours du PMSI-HAD (Par
défaut : 1)				*/
/*
									*/
/*				- HAD_DAS = flag (0/1) : si 1, on recherche les codes CIM10 dans les DSA des séjours du PMSI-HAD (Par
défaut : 1)				*/
/*
									*/
/*				- HAD_MPP = flag (0/1) : si 1, on recherche les codes CIM10 dans les MPP des sous-séquences du PMSI-
HAD (Par défaut : 1)			*/
/*
									*/
/*				- HAD_MPA = flag (0/1) : si 1, on recherche les codes CIM10 dans les MPA des sous-séquences du
PMSI-HAD (Par défaut : 1)			*/
```

```
/*                                                                                                                    */
/*                          - MCO_DP = flag (0/1) : si 1, on recherche les codes CIM10 dans les DP des séjours du PMSI-MCO (Par
défaut : 1)                                                                                                           */
/*                                                                                                                    */
/*                          - MCO_DR = flag (0/1) : si 1, on recherche les codes CIM10 dans les DR des séjours du PMSI-MCO (Par
défaut : 1)                                                                                                           */
/*                                                                                                                    */
/*                          - MCO_DAS = flag (0/1) : si 1, on recherche les codes CIM10 dans les DSA des séjours du PMSI-MCO
(Par défaut : 1)                                                                                                      */
/*                                                                                                                    */
/*                          - MCO_DP_UM = flag (0/1) : si 1, on recherche les codes CIM10 dans les DP des UM du PMSI-MCO (Par
défaut : 1)                                                                                                           */
/*                                                                                                                    */
/*                          - MCO_DR_UM = flag (0/1) : si 1, on recherche les codes CIM10 dans les DR des UM du PMSI-MCO (Par
défaut : 1)                                                                                                           */
/*                                                                                                                    */
/*                          - SSR_FP = flag (0/1) : si 1, on recherche les codes CIM10 dans les FP des RHS du PMSI-SSR (Par défaut :
1)                                                                                                                    */
/*                                                                                                                    */
/*                          - SSR_MPP = flag (0/1) : si 1, on recherche les codes CIM10 dans les MPP des RHS du PMSI-SSR (Par
défaut : 1)                                                                                                           */
/*                                                                                                                    */
/*                          - SSR_AE = flag (0/1) : si 1, on recherche les codes CIM10 dans les AE des RHS du PMSI-SSR (Par défaut :
1)                                                                                                                    */
/*                                                                                                                    */
/*                          - SSR_DAS = flag (0/1) : si 1, on recherche les codes CIM10 dans les DAS du PMSI-SSR (Par défaut : 1)
                                                                                                                      */
/*                                                                                                                    */
/*
        ************************************************************************************************************
************************************          */
/*                                                                                                                    */
/*                  Tables en sortie : &tbl_out. (renseignée en paramètres de la macro)
                                                                                                                      */
/*                  avec sélection des codes disponibles dans le référentiel &tbl_codes., hors séjours en erreur
                                                                                                                      */
/*                                                                                                                    */
/*
        ************************************************************************************************************
************************************          */

*
        ************************************************************************************************************
************************************          ;
```

```
*           Macro pour repérer les codes CIM10 dans le PMSI - séjours

                        ;
*
        ************************************************************************************************************
************************************        ;

%macro CIM10_sejours(an_deb_pmsi=, an_fin_pmsi=, an_crea_t=, an_fin_t=, tbl_codes=, table=, domaine=, var_CIM10=, tbl_out=,
tbl_patients=);

        %do i = %sysfunc(max(&an_deb_pmsi., &an_crea_t. )) %to %sysfunc(min(&an_fin_pmsi., &an_fin_t.));

                %if &i. < 10 %then %let an = 0&i.;
                %else %let an = &i.;

                * On définit les variables de jointure pour chaque domaine du PMSI;
                %if &domaine. = HAD %then
                        %do;
                                %let var_join1 = ETA_NUM_EPMSI;
                                %let var_join2 = RHAD_NUM;
                        %end;
                %if &domaine. = MCO %then
                        %do;
                                %let var_join1 = ETA_NUM;
                                %let var_join2 = RSA_NUM;
                        %end;
                %if &domaine. = RIP %then
                        %do;
                                %let var_join1 = ETA_NUM_EPMSI;
                                %let var_join2 = RIP_NUM;
                        %end;
                %if &domaine. = SSR %then
                        %do;
                                %let var_join1 = ETA_NUM;
                                %let var_join2 = RHA_NUM;
                        %end;

                * On joint la table CIM10 avec la table des séjours en filtrant sur les codes CIM10 sélectionnés;
                %put Récupération des séjours du PMSI-&domaine. (table &table.) contenant des actes CIM10 pré-définis - Année
&an. ...;

                proc sql;

                        %connectora;

                                CREATE TABLE CIM10_&domaine._&table._&var_CIM10._&an. AS
                                        SELECT *
                                        FROM CONNECTION TO ORACLE (
                                                SELECT DISTINCT
                                                        c.&var_join1.,
                                                        c.&var_join2.,
                                                        pop.BEN_IDT_ANO,
                                                        c.EXE_SOI_DTD,
                                                        c.EXE_SOI_DTF,
                                                        %if &domaine. = MCO %then
                                                                %do;
                                                                        %if &an. < 12 %then %do; '' AS
ETA_NUM_GEO, %end;
                                                                        %if &an. = 12 %then %do;
um.ETA_NUM_GEO1 AS ETA_NUM_GEO, %end;
                                                                        %if &an.> 12 %then %do;
um.ETA_NUM_GEO, %end;
                                                                        um.UM_ORD_NUM,
                                                                        b.SOR_MOD,
                                                                        b.DGN_PAL,
                                                                        b.DGN_REL,
                                                                        b.GRG_GHM,
                                                                        b.SEJ_NBJ,
                                                                %end;
                                                        %if &domaine. = SSR %then
                                                                %do;
```

```
                                                                b.HOS_TYP_UM,
                                                                s.SEJ_NBJ,
                                                        %end;
                                                %if &domaine. = RIP %then
                                                        %do;
                                                                rsa.FOR_ACT,
                                                                b.SEJ_NBJ,
                                                        %end;
                                                %if &domaine. = HAD %then
                                                        %do;
                                                                b.SEJ_NBJ,
                                                        %end;
                                                20&an. AS annee,
                                                cod.*
                                        FROM &tbl_patients. pop
                                                INNER JOIN T_&domaine.&an.C c
                                                        ON      pop.BEN_NIR_PSA = c.NIR_ANO_17
                                                %if &domaine. = MCO or &domaine. = HAD or &domaine. =
SSR %then
                                                        %do;
                                                                INNER JOIN T_&domaine.&an.B b
                                                                        ON      c.&var_join1. =
b.&var_join1.
                                                                        AND     c.&var_join2. =
b.&var_join2.
                                                        %end;
                                                %if &domaine. = MCO %then
                                                        %do;
                                                                INNER JOIN T_&domaine.&an.UM um
                                                                        ON      c.&var_join1. =
um.&var_join1.
                                                                        AND     c.&var_join2. =
um.&var_join2.
                                                        %end;
                                                %if &domaine. = SSR %then
                                                        %do;
                                                                INNER JOIN T_&domaine.&an.S s
                                                                        ON      c.&var_join1. =
s.&var_join1.
                                                                        AND     c.&var_join2. =
s.&var_join2.
                                                        %end;
                                                %if &domaine. = RIP %then
                                                        %do;
                                                                INNER JOIN T_&domaine.&an.RSA rsa
                                                                        ON      c.&var_join1. =
rsa.&var_join1.
                                                                        AND     c.&var_join2. =
rsa.&var_join2.
                                                        %end;
                                                %if &table. = D or &table. = DMPA or &table. = DMPP
%then
                                                        %do;
                                                                INNER JOIN T_&domaine.&an.&table.
cim10
                                                                        ON      c.&var_join1. =
cim10.&var_join1.
                                                                        AND     c.&var_join2. =
cim10.&var_join2.
                                                        %end;
                                                , &tbl_codes. cod

                                        WHERE
                                                %if &table. = D or &table. = DMPA or &table. =
DMPP %then
                                                        %do;

        SUBSTR(TRIM(cim10.&var_CIM10.), 1, taille) = TRIM(cod.code_CIM)
                                                        %end;
                                                %if &table. = UM %then
                                                        %do;
```

```
            SUBSTR(TRIM(um.&var_CIM10.), 1, taille) = TRIM(cod.code_CIM)
                                                                        %end;
                                                                %if &table. = B %then
                                                                        %do;

            SUBSTR(TRIM(b.&var_CIM10.), 1, taille) = TRIM(cod.code_CIM)
                                                                        %end;
                                                        AND c.NIR_ANO_17 NOT IN ('&cle_inc1.', '&cle_inc2.')
                                                        %if &domaine. = HAD %then
                                                                %do;
                                                                        AND NIR_RET = '0' AND NAI_RET = '0'
AND SEX_RET = '0' AND SEJ_RET = '0' AND FHO_RET = '0'
                                                                        AND PMS_RET = '0' AND DAT_RET =
'0' %if &an. > 12 %then %do; AND COH_NAI_RET = '0'
                                                                        AND COH_SEX_RET = '0' %end;
                                                                %end;
                                                        %if &domaine. = MCO %then
                                                                %do;
                                                                        AND NIR_RET = '0' AND NAI_RET = '0'
AND SEX_RET = '0' AND SEJ_RET = '0' AND FHO_RET = '0'
                                                                        AND PMS_RET = '0' AND DAT_RET =
'0' %if &an. > 12 %then %do; AND COH_NAI_RET = '0'
                                                                        AND COH_SEX_RET = '0' %end;
                                                                        AND c.ETA_NUM NOT IN
('130780521', '130783236', '130783293', '130784234', '130804297', '600100101',
                                                                                '750041543', '750100018',
'750100042', '750100075', '750100083', '750100091', '750100109',
                                                                                '750100125', '750100166',
'750100208', '750100216', '750100232', '750100273', '750100299',
                                                                                '750801441', '750803447',
'750803454', '910100015', '910100023', '920100013', '920100021',
                                                                                '920100039', '920100047',
'920100054', '920100062', '930100011', '930100037', '930100045',
                                                                                '940100027', '940100035',
'940100043', '940100050', '940100068', '950100016', '690783154',
                                                                                '690784137', '690784152',
'690784178', '690787478', '830100558')
                                                                        AND GRG_GHM NOT IN ('90Z00Z')
AND GRG_RET NOT IN ('024') AND GRG_GHM NOT IN ('90H01Z', '90Z00Z', '90Z01Z',
                                                                                '90Z02Z', '90Z03Z')
                                                                        AND ((SEJ_TYP = 'A' OR SEJ_TYP IS
NULL) OR (SEJ_TYP = 'B' AND GRG_GHM NOT IN ('28Z14Z','28Z15Z','28Z16Z')))
                                                                %end;
                                                        %if &domaine. = RIP %then
                                                                %do;
                                                                        AND NIR_RET = '0' AND NAI_RET = '0'
AND SEX_RET = '0' AND SEJ_RET = '0' AND FHO_RET = '0'
                                                                        AND PMS_RET = '0' AND DAT_RET =
'0' %if &an. > 12 %then %do; AND COH_NAI_RET = '0'
                                                                        AND COH_SEX_RET = '0' %end;
                                                                %end;
                                                        %if &domaine. = SSR %then
                                                                %do;
                                                                        AND NIR_RET = '0' AND NAI_RET = '0'
AND SEX_RET = '0' AND SEJ_RET = '0' AND FHO_RET = '0'
                                                                        AND PMS_RET = '0' AND DAT_RET =
'0' %if &an. > 12 %then %do; AND COH_NAI_RET = '0'
                                                                        AND COH_SEX_RET = '0' %end;
                                                                %end;
                                                );

                        DISCONNECT FROM ORACLE;

                quit;

                data CIM10_&domaine._&table._&var_CIM10._&an. (rename = (ETA_NUM_GEO2 = ETA_NUM_GEO));
                        set CIM10_&domaine._&table._&var_CIM10._&an.;
                        length table $5. date_debut date_fin 4. ETA_NUM_GEO2 $9. variable $11.;
                        table = "&table.";
                        domaine = "&domaine.";
```

```
                                variable = "&var_CIM10.";
                                type = "PMSI séjours";
                                date_debut = datepart(EXE_SOI_DTD);
                                date_fin = datepart(EXE_SOI_DTF);
                                ETA_NUM_GEO2 = ETA_NUM_GEO;
                                format date_debut date_fin ddmmyy10.;
                                drop EXE_SOI_DTD EXE_SOI_DTF ETA_NUM_GEO;
                        run;

                        %arret_erreur;

                %end;

%mend CIM10_sejours;

*
                ************************************************************************************************************
**********************************          ;
*               Macro finale pour appeler les précédentes, pour chaque table du PMSI

                ;
*
                ************************************************************************************************************
**********************************          ;

%macro extract_CIM10_PMSI(annee_deb=, annee_fin=, HAD_DP = 1, HAD_DAS = 1, HAD_MPP = 1, HAD_MPA = 1, MCO_DP = 1,
MCO_DR = 1, MCO_DAS = 1,
                MCO_DP_UM = 1, MCO_DR_UM = 1, SSR_FP = 1, SSR_MPP = 1, SSR_AE = 1, SSR_DAS = 1, tbl_out=, tbl_codes=,
tbl_patients=);

                %let an_deb_pmsi = %sysevalf(&annee_deb. - 2000);
                %let an_fin_pmsi = %sysevalf(&annee_fin. - 2000);

                * Pour le PMSI-HAD;
                %if &HAD_MPP. = 1 or &HAD_MPA. = 1 %then
                        %do;

                                %let domaine = HAD;

                                %if &HAD_DP. = 1 %then
                                        %do;

                                                * Dans les tables B;
                                                %CIM10_sejours(
                                                        an_deb_pmsi = &an_deb_pmsi.,
                                                        an_fin_pmsi = &an_fin_pmsi.,
                                                        an_crea_t = 06,
                                                        an_fin_t = 11,
                                                        tbl_codes = &tbl_codes.,
                                                        table = B,
                                                        domaine = &domaine.,
                                                        var_CIM10 = DGN_PAL,
                                                        tbl_out = &tbl_out.,
                                                        tbl_patients = &tbl_patients.
                                                        );

                                                %CIM10_sejours(
                                                        an_deb_pmsi = &an_deb_pmsi.,
                                                        an_fin_pmsi = &an_fin_pmsi.,
                                                        an_crea_t = 14,
                                                        an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY()))) - 2000),
                                                        tbl_codes = &tbl_codes.,
                                                        table = B,
                                                        domaine = &domaine.,
                                                        var_CIM10 = DGN_PAL,
                                                        tbl_out = &tbl_out.,
                                                        tbl_patients = &tbl_patients.
                                                        );

                                        %end;
```

```
                        %if &HAD_DAS. = 1 %then
                                %do;

                                        * Dans les tables D;
                                        %CIM10_sejours(
                                                an_deb_pmsi = &an_deb_pmsi.,
                                                an_fin_pmsi = &an_fin_pmsi.,
                                                an_crea_t = 10,
                                                an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
                                                tbl_codes = &tbl_codes.,
                                                table = D,
                                                domaine = &domaine.,
                                                var_CIM10 = DGN_ASS,
                                                tbl_out = &tbl_out.,
                                                tbl_patients = &tbl_patients.
                                                );

                                %end;

                        %if &HAD_MPA. = 1 %then
                                %do;

                                        * Dans les tables DMPA;
                                        %CIM10_sejours(
                                                an_deb_pmsi = &an_deb_pmsi.,
                                                an_fin_pmsi = &an_fin_pmsi.,
                                                an_crea_t = 12,
                                                an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
                                                tbl_codes = &tbl_codes.,
                                                table = DMPA,
                                                domaine = &domaine.,
                                                var_CIM10 = DGN_ASS_MPA,
                                                tbl_out = &tbl_out.,
                                                tbl_patients = &tbl_patients.
                                                );

                                %end;

                        %if &HAD_MPP. = 1 %then
                                %do;

                                        * Dans les tables DMPP;
                                        %CIM10_sejours(
                                                an_deb_pmsi = &an_deb_pmsi.,
                                                an_fin_pmsi = &an_fin_pmsi.,
                                                an_crea_t = 12,
                                                an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
                                                tbl_codes = &tbl_codes.,
                                                table = DMPP,
                                                domaine = &domaine.,
                                                var_CIM10 = DGN_ASS_MPP,
                                                tbl_out = &tbl_out.,
                                                tbl_patients = &tbl_patients.
                                                );

                                %end;

                %end;
                * Fin du PMSI-HAD;

        * Pour le PMSI-MCO;
        %if &MCO_DP. = 1 or &MCO_DR. = 1 or &MCO_DAS. = 1 or &MCO_DP_UM. = 1 or &MCO_DR_UM. = 1 %then
                %do;

                        %let domaine = MCO;

                        %if &MCO_DP. = 1 %then
                                %do;

                                        * Dans les tables B;
                                        %CIM10_sejours(
```

```
                                        an_deb_pmsi = &an_deb_pmsi.,
                                        an_fin_pmsi = &an_fin_pmsi.,
                                        an_crea_t = 06,
                                        an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
                                        tbl_codes = &tbl_codes.,
                                        table = B,
                                        domaine = &domaine.,
                                        var_CIM10 = DGN_PAL,
                                        tbl_out = &tbl_out.,
                                        tbl_patients = &tbl_patients.
                                        );

                                %end;

        %if &MCO_DR. = 1 %then
                        %do;

                                %* Dans les tables B;
                                %CIM10_sejours(
                                        an_deb_pmsi = &an_deb_pmsi.,
                                        an_fin_pmsi = &an_fin_pmsi.,
                                        an_crea_t = 06,
                                        an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
                                        tbl_codes = &tbl_codes.,
                                        table = B,
                                        domaine = &domaine.,
                                        var_CIM10 = DGN_REL,
                                        tbl_out = &tbl_out.,
                                        tbl_patients = &tbl_patients.
                                        );

                                %end;

        %if &MCO_DAS. = 1 %then
                        %do;

                                %* Dans les tables D;
                                %CIM10_sejours(
                                        an_deb_pmsi = &an_deb_pmsi.,
                                        an_fin_pmsi = &an_fin_pmsi.,
                                        an_crea_t = 06,
                                        an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
                                        tbl_codes = &tbl_codes.,
                                        table = D,
                                        domaine = &domaine.,
                                        var_CIM10 = ASS_DGN,
                                        tbl_out = &tbl_out.,
                                        tbl_patients = &tbl_patients.
                                        );

                                %end;

        %if &MCO_DP_UM. = 1 %then
                        %do;

                                %* Dans les tables UM;
                                %CIM10_sejours(
                                        an_deb_pmsi = &an_deb_pmsi.,
                                        an_fin_pmsi = &an_fin_pmsi.,
                                        an_crea_t = 08,
                                        an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
                                        tbl_codes = &tbl_codes.,
                                        table = UM,
                                        domaine = &domaine.,
                                        var_CIM10 = DGN_PAL,
                                        tbl_out = &tbl_out.,
                                        tbl_patients = &tbl_patients.
                                        );

                                %end;
```

```
                        %if &MCO_DR_UM. = 1 %then
                                %do;

                                        * Dans les tables UM;
                                        %CIM10_sejours(
                                                an_deb_pmsi = &an_deb_pmsi.,
                                                an_fin_pmsi = &an_fin_pmsi.,
                                                an_crea_t = 08,
                                                an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
                                                tbl_codes = &tbl_codes.,
                                                table = UM,
                                                domaine = &domaine.,
                                                var_CIM10 = DGN_REL,
                                                tbl_out = &tbl_out.,
                                                tbl_patients = &tbl_patients.
                                                );

                                        %end;

                        %end;
                        * Fin du PMSI-MCO;

* Pour le PMSI-SSR;
%if &SSR_FP. = 1 or &SSR_MPP. = 1 or &SSR_AE. = 1 or &SSR_DAS. = 1 %then
                        %do;

                                %let domaine = SSR;

                                %if &SSR_FP. = 1 %then
                                        %do;

                                                * Dans les tables B;
                                                %CIM10_sejours(
                                                        an_deb_pmsi = &an_deb_pmsi.,
                                                        an_fin_pmsi = &an_fin_pmsi.,
                                                        an_crea_t = 06,
                                                        an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
                                                        tbl_codes = &tbl_codes.,
                                                        table = B,
                                                        domaine = &domaine.,
                                                        var_CIM10 = FP_PEC,
                                                        tbl_out = &tbl_out.,
                                                        tbl_patients = &tbl_patients.
                                                        );

                                        %end;

                                %if &SSR_MPP. = 1 %then
                                        %do;

                                                * Dans les tables B;
                                                %CIM10_sejours(
                                                        an_deb_pmsi = &an_deb_pmsi.,
                                                        an_fin_pmsi = &an_fin_pmsi.,
                                                        an_crea_t = 06,
                                                        an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
                                                        tbl_codes = &tbl_codes.,
                                                        table = B,
                                                        domaine = &domaine.,
                                                        var_CIM10 = MOR_PRP,
                                                        tbl_out = &tbl_out.,
                                                        tbl_patients = &tbl_patients.
                                                        );

                                        %end;

                                %if &SSR_AE. = 1 %then
                                        %do;

                                                * Dans les tables B;
                                                %CIM10_sejours(
```

```
                                                an_deb_pmsi = &an_deb_pmsi.,
                                                an_fin_pmsi = &an_fin_pmsi.,
                                                an_crea_t = 06,
                                                an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY()))) - 2000),
                                                tbl_codes = &tbl_codes.,
                                                table = B,
                                                domaine = &domaine.,
                                                var_CIM10 = ETL_AFF,
                                                tbl_out = &tbl_out.,
                                                tbl_patients = &tbl_patients.
                                                );

                                        %end;

                                %if &SSR_DAS. = 1 %then
                                        %do;

                                                * Dans les tables D;
                                                %CIM10_sejours(
                                                        an_deb_pmsi = &an_deb_pmsi.,
                                                        an_fin_pmsi = &an_fin_pmsi.,
                                                        an_crea_t = 09,
                                                        an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY()))) - 2000),
                                                        tbl_codes = &tbl_codes.,
                                                        table = D,
                                                        domaine = &domaine.,
                                                        var_CIM10 = DGN_COD,
                                                        tbl_out = &tbl_out.,
                                                        tbl_patients = &tbl_patients.
                                                        );

                                                %end;

                        %end;
                        * Fin du PMSI-SSR;

                * Empilage de la table temporaire à la table Résultat;
                data &tbl_out.;
                        set CIM10_:;
                run;

                %arret_erreur;

                * Suppression de la table temporaire;
                proc datasets library = work memtype = data nolist;
                        delete CIM10_:;
                run;

%mend extract_CIM10_PMSI;
```

3.1.11 Macro_Reperage_des_soins_CIP_DCIR.sas

```
/*
	*****************************************************************************************************
************************************	*/
/*
                                                                                                        */
/*		Macro pour repérer les remboursement des codes CIP souhaités : extract_CIP

	*/
/*
                                                                                                        */
/*
	*****************************************************************************************************
************************************	*/
/*
                                                                                                        */
/*		Arguments en entrée :

                                                                                                */
/*
                                                                                                        */
/*			- annee_deb = la première année de soins (ie. filtre sur EXE_SOI_DTD) pour laquelle on souhaite
repérer les codes                                                */
/*
                                                                                                        */
/*			- annee_fin = la dernière année de soins (ie. filtre sur EXE_SOI_DTD) pour laquelle on souhaite repérer
les codes                                        */
/*
                                                                                                        */
/*			- tbl_out = le nom de la table en sortie (peut contenir un nom de librairie)
                                                                                                */
/*
                                                                                                        */
/*			- tbl_codes = la table contenant les classes ATC à repérer (au format Oracle)
                                                                                                */
/*
                                                                                                        */
/*			- tbl_patients = le nom de la table contenant la correspondance BEN_IDT_ANO <-> BEN_NR_PSA (au
format Oracle)                                                */
/*
                                                                                                        */
/*
	*****************************************************************************************************
************************************	*/
/*
                                                                                                        */
/*		Tables en sortie : &tbl_out. (renseignée en paramètres de la macro)

                                                                                                */
/*		avec sélection des codes disponibles dans le référentiel &tbl_codes., filtrée sur les soins de ville uniquement, les
NIR  normaux +	*/
/*		suppression des lignes pour information

                                                        */
```

```
/*
                                                                                                    */
/*      ****************************************************************************************************
***********************************              */

%macro extract_CIP(annee_deb=, annee_fin=, tbl_out=, tbl_codes=, tbl_patients=);

	%let annee_fin_flx = %sysevalf(&annee_fin. + 1);

	* On boucle sur les années de repérage (en flux);
	%do Annee = &annee_deb. %to &annee_fin_flx.;

		* On boucle sur les 12 mois de l année;
		%do i = 1 %to 12;

			%if &i. < 10 %then %let Mois = 0&i.;
			%else %let Mois = &i.;

			%put Données de &Mois./&Annee.;

			proc sql;

				%connectora;

					CREATE TABLE tmp_extract_CIP_DCIR_&annee._&mois. AS SELECT * FROM
CONNECTION TO ORACLE (
							SELECT
								pop.BEN_IDT_ANO,
								prs.EXE_SOI_DTD,
								prs.EXE_SOI_DTF,
								prs.PRS_NAT_REF,
								prs.BSE_REM_MNT,
								ref.PHA_ATC_C07,
								pha.PHA_PRS_IDE AS code_CIP7,
								pha.PHA_PRS_C13 AS code_CIP13,
								pha.PHA_ACT_QSN,
								pha.PHA_DEC_TOP,
								ref.PHA_CND_TOP,
								ete.ETB_EXE_FIN,
								ete.ETE_MCO_DDP,
								cod.*
							FROM &tbl_patients. pop
								INNER JOIN ER_PRS_F prs
									ON	pop.BEN_NIR_PSA =
prs.BEN_NIR_PSA
								INNER JOIN ER_PHA_F pha
									ON	pha.FLX_DIS_DTD =
prs.FLX_DIS_DTD
									AND	pha.FLX_TRT_DTD =
prs.FLX_TRT_DTD
									AND	pha.FLX_EMT_TYP =
prs.FLX_EMT_TYP
									AND	pha.FLX_EMT_NUM =
prs.FLX_EMT_NUM
									AND	pha.FLX_EMT_ORD =
prs.FLX_EMT_ORD
									AND	pha.ORG_CLE_NUM =
prs.ORG_CLE_NUM
									AND	pha.DCT_ORD_NUM =
prs.DCT_ORD_NUM
									AND	pha.PRS_ORD_NUM =
prs.PRS_ORD_NUM
									AND	pha.REM_TYP_AFF =
prs.REM_TYP_AFF
								INNER JOIN IR_PHA_R ref
									ON	ref.PHA_CIP_C13 =
pha.PHA_PRS_C13
								INNER JOIN &tbl_codes. cod
```

```
                                                                        ON ref.PHA_ATC_C07 =
cod.classe_ATC
                                                                LEFT JOIN ER_ETE_F ete
                                                                        ON      ete.FLX_DIS_DTD =
prs.FLX_DIS_DTD
                                                                        AND     ete.FLX_TRT_DTD =
prs.FLX_TRT_DTD
                                                                        AND     ete.FLX_EMT_TYP =
prs.FLX_EMT_TYP
                                                                        AND     ete.FLX_EMT_NUM =
prs.FLX_EMT_NUM
                                                                        AND     ete.FLX_EMT_ORD =
prs.FLX_EMT_ORD
                                                                        AND     ete.ORG_CLE_NUM =
prs.ORG_CLE_NUM
                                                                        AND     ete.DCT_ORD_NUM =
prs.DCT_ORD_NUM
                                                                        AND     ete.PRS_ORD_NUM =
prs.PRS_ORD_NUM
                                                                        AND     ete.REM_TYP_AFF =
prs.REM_TYP_AFF
                                                                WHERE  prs.BEN_CDI_NIR = '00' AND prs.CPL_MAJ_TOP IN
(0, 1)
                                                                        AND (ete.ETE_IND_TAA NOT IN (1, 2) OR
ete.ETE_IND_TAA IS NULL)
                                                                        AND prs.DPN_QLF NOT IN (71, 72) AND
NOT(prs.DPN_QLF = 0 AND prs.PRS_DPN_QLP = 71)
                                                                        AND prs.FLX_DIS_DTD =
TO_DATE(%str(%'&Annee.&Mois.01%'), 'YYYYMMDD')
                                                                        AND prs.EXE_SOI_DTD BETWEEN
TO_DATE(%str(%'&Annee_deb.0101%'), 'YYYYMMDD') AND TO_DATE(%str(%'&Annee_fin.1231%'), 'YYYYMMDD')
                                                                );

                                        disconnect from oracle;

                                quit;

                                %arret_erreur;

                                * On empile toutes les tables mensuelles;
                                %if %sysfunc(exist(&tbl_out.)) = 1 %then
                                        %do;

                                                proc append base = &tbl_out. data = tmp_extract_CIP_DCIR_&annee._&mois.;
                                                run;

                                                %arret_erreur;

                                                proc delete data = tmp_extract_CIP_DCIR_&annee._&mois.;
                                                run; quit;

                                        %end;

                                %if %sysfunc(exist(&tbl_out.)) = 0 %then
                                        %do;

                                                data &tbl_out.;
                                                        set tmp_extract_CIP_DCIR_&annee._&mois.;
                                                run;

                                                %arret_erreur;

                                                proc delete data = tmp_extract_CIP_DCIR_&annee._&mois.;
                                                run; quit;

                                        %end;

                        %end;
                        * Fin de la boucle sur les 12 mois;

                %end;
```

```
	* Fin de la boucle par année;


	* On applique le programme de régularisations pour céer la table finale;
	proc sql undo_policy = none;

		CREATE TABLE &tbl_out. AS
			SELECT
				BEN_IDT_ANO,
				datepart(EXE_SOI_DTD) AS date_debut length = 4 format ddmmyy10.,
				CASE	WHEN datepart(EXE_SOI_DTF) IS NULL THEN datepart(EXE_SOI_DTD)
						ELSE datepart(EXE_SOI_DTF)
						END AS date_fin length = 4 format ddmmyy10.,
				"DCIR" AS type,
				PHA_ATC_C07,
				code_CIP7,
				code_CIP13,
				PHA_DEC_TOP,
				PHA_CND_TOP,
				reperage,
				ETB_EXE_FIN,
				ETE_MCO_DDP,
				SUM(BSE_REM_MNT) AS montant,
				SUM(PHA_ACT_QSN) AS quantite
			FROM &tbl_out.
			GROUP BY 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12;

		DELETE FROM &tbl_out. WHERE montant <= 0 OR quantite < 0;

	quit;

	%arret_erreur;

%mend extract_CIP;
```

3.1.12 Macro_Reperage_des_soins_Codes_actes_PMSI.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*		Macro pour repérer les remboursement des codes diagnostics souhaités, dans le PMSI : extract_ACT_COD_PMSI
                                                                                                                    */
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*		Arguments en entrée :
                                                                                                                    */
/*
                                                                                                                    */
/*			- annee_deb = la première année de soins (ie. année des tables PMSI) pour laquelle on souhaite
repérer les codes                                                                                                   */
/*
                                                                                                                    */
/*			- annee_fin = la dernière année de soins (ie. année des tables PMSI) pour laquelle on souhaite repérer
les codes                                                                                                           */
/*
                                                                                                                    */
/*			- tbl_out = le nom de la table en sortie (peut contenir un nom de librairie)
                                                                                                                    */
/*
                                                                                                                    */
/*			- tbl_codes = la table contenant les codes actes à repérer (au format Oracle)
                                                                                                                    */
/*
                                                                                                                    */
/*			- tbl_patients = le nom de la table contenant la correspondance BEN_IDT_ANO <-> BEN_NR_PSA (au
format Oracle)                                                                                                      */
/*
                                                                                                                    */
/*			- MCO = flag (0/1) : si 1, on recherche les codes acte dans le PMSI-MCO (Par défaut : 1)
                                                                                                                    */
/*
                                                                                                                    */
/*			- SSR = flag (0/1) : si 1, on recherche les codes acte dans le PMSI-SSR (Par défaut : 1)
                                                                                                                    */
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*		Tables en sortie : &tbl_out. (renseignée en paramètres de la macro)
                                                                                                                    */
```

```
/*                          avec sélection des codes disponibles dans le référentiel &tbl_codes., hors séjours en erreur
                                                                                                    */
/*
                                                                                                                            */
/*
          ****************************************************************************************************
***********************************                    */

*
          ****************************************************************************************************
***********************************                    ;
*         Macro pour repérer les codes actes dans le PMSI - séjours

                              ;
*
          ****************************************************************************************************
***********************************                    ;

%macro ACT_COD_sejours(an_deb_pmsi=, an_fin_pmsi=, an_crea_t=, an_fin_t=, tbl_codes=, table=, domaine=, tbl_out=,
tbl_patients=);

          %do i = %sysfunc(max(&an_deb_pmsi., &an_crea_t. )) %to %sysfunc(min(&an_fin_pmsi., &an_fin_t.));

                    %if &i. < 10 %then %let an = 0&i.;
                    %else %let an = &i.;

                    * On définit les variables de jointure pour chaque domaine du PMSI;
                    %if &domaine. = MCO %then
                              %do;
                                        %let var_join1 = ETA_NUM;
                                        %let var_join2 = RSA_NUM;
                              %end;
                    %if &domaine. = SSR %then
                              %do;
                                        %let var_join1 = ETA_NUM;
                                        %let var_join2 = RHA_NUM;
                              %end;

                    * On joint la table de factures avec la table des séjours en filtrant sur les codes actes sélectionnés;
                    %put Récupération des séjours du PMSI-&domaine. (table &table.) contenant des codes actes pré-définis - Année
&an. ...;

                    proc sql;

                              %connectora;

                                        CREATE TABLE ACT_COD_&domaine._&table._&an. AS
                                                  SELECT *
                                                  FROM CONNECTION TO ORACLE (
                                                            SELECT DISTINCT
                                                                      c.&var_join1.,
                                                                      c.&var_join2.,
                                                                      pop.BEN_IDT_ANO,
                                                                      c.EXE_SOI_DTD,
                                                                      c.EXE_SOI_DTF,
                                                                      20&an. AS annee,
                                                                      act.ACT_COD,
                                                                      act.ACT_COE,
                                                                      %if &table. = FC and &an. >= 17 %then
                                                                                %do;
                                                                                          EXE_SPE,
                                                                                %end;
                                                                      cod.*
                                                            FROM &tbl_patients. pop
                                                                      INNER JOIN T_&domaine.&an.C c
                                                                                ON        pop.BEN_NIR_PSA = c.NIR_ANO_17
                                                                      INNER JOIN T_&domaine.&an.B b
                                                                                ON        c.&var_join1. = b.&var_join1.
                                                                                AND       c.&var_join2. = b.&var_join2.
                                                                      INNER JOIN T_&domaine.&an.E e
```

```
                                                                ON          c.&var_join1. = e.ETA_NUM
                                                        INNER JOIN T_&domaine.&an.&table. act
                                                                ON          c.&var_join1. = act.&var_join1.
                                                                AND         c.&var_join2. = act.&var_join2.
                                                        , &tbl_codes. cod
                                                WHERE SUBSTR(act.ACT_COD, 1, taille) = TRIM(cod.code_acte)
                                                        AND c.NIR_ANO_17 NOT IN ('&cle_inc1.', '&cle_inc2.')
                                                        %if &domaine. = MCO %then
                                                                %do;
                                                                        AND NIR_RET = '0' AND NAI_RET = '0'
AND SEX_RET = '0' AND SEJ_RET = '0' AND FHO_RET = '0'
                                                                        AND PMS_RET = '0' AND DAT_RET =
'0' %if &an. > 12 %then %do; AND COH_NAI_RET = '0'
                                                                        AND COH_SEX_RET = '0' %end;
                                                                        AND c.ETA_NUM NOT IN
('130780521', '130783236', '130783293', '130784234', '130804297', '600100101',
                                                                                '750041543', '750100018',
'750100042', '750100075', '750100083', '750100091', '750100109',
                                                                                '750100125', '750100166',
'750100208', '750100216', '750100232', '750100273', '750100299',
                                                                                '750801441', '750803447',
'750803454', '910100015', '910100023', '920100013', '920100021',
                                                                                '920100039', '920100047',
'920100054', '920100062', '930100011', '930100037', '930100045',
                                                                                '940100027', '940100035',
'940100043', '940100050', '940100068', '950100016', '690783154',
                                                                                '690784137', '690784152',
'690784178', '690787478', '830100558')
                                                                        AND GRG_GHM NOT IN ('90Z00Z')
AND GRG_RET NOT IN ('024') AND GRG_GHM NOT IN ('90H01Z', '90Z00Z',
                                                                                '90Z01Z', '90Z02Z',
'90Z03Z')
                                                                        AND ((SEJ_TYP = 'A' OR SEJ_TYP IS
NULL) OR (SEJ_TYP = 'B' AND GRG_GHM NOT IN ('28Z14Z','28Z15Z','28Z16Z')))
                                                                %end;
                                                        %if &domaine. = SSR %then
                                                                %do;
                                                                        AND NIR_RET = '0' AND NAI_RET = '0'
AND SEX_RET = '0' AND SEJ_RET = '0' AND FHO_RET = '0'
                                                                        AND PMS_RET = '0' AND DAT_RET =
'0' %if &an. > 12 %then %do; AND COH_NAI_RET = '0'
                                                                        AND COH_SEX_RET = '0' %end;
                                                                %end;
                                                        );

                                DISCONNECT FROM ORACLE;

                        quit;

                        data ACT_COD_&domaine._&table._&an.;
                                set ACT_COD_&domaine._&table._&an.;
                                length date_debut date_fin 4.;
                                table = "&table.";
                                domaine = "&domaine.";
                                type = "PMSI séjours";
                                date_debut = datepart(EXE_SOI_DTD);
                                date_fin = datepart(EXE_SOI_DTF);
                                drop EXE_SOI_DTD EXE_SOI_DTF;
                                format date_debut date_fin ddmmyy10.;
                        run;

                        %arret_erreur;

                %end;

%mend ACT_COD_sejours;

*
                ********************************************************************************************************
**********************************            ;
```

```
*           Macro pour repérer les codes actes dans le PMSI - ACE

                                        ;
*
          ***************************************************************************************************
************************************          ;

%macro ACT_COD_ACE(an_deb_pmsi=, an_fin_pmsi=, an_crea_t=, an_fin_t=, tbl_codes=, table=, domaine=, tbl_out=, tbl_patients=);

          %do i = %sysfunc(max(&an_deb_pmsi., &an_crea_t. )) %to %sysfunc(min(&an_fin_pmsi., &an_fin_t.));

                    %if &i. < 10 %then %let an = 0&i.;
                    %else %let an = &i.;

                    * On définit les variables de jointure pour chaque domaine du PMSI;
                    %let var_join1 = ETA_NUM;
                    %let var_join2 = SEQ_NUM;

                    * On joint la table de factures avec la table des ACE en filtrant sur les codes actes sélectionnés;
                    %put Récupération des ACE du PMSI-&domaine. (table &table.) contenant des codes actes pré-définis - Année
&an. ...;

                    proc sql;

                              %connectora;

                                        CREATE TABLE ACT_COD_&domaine._&table._&an. AS
                                                  SELECT *
                                                  FROM CONNECTION TO ORACLE (
                                                            SELECT DISTINCT
                                                                      c.&var_join1.,
                                                                      c.&var_join2.,
                                                                      pop.BEN_IDT_ANO,
                                                                      c.EXE_SOI_DTD,
                                                                      c.EXE_SOI_DTF,
                                                                      20&an. AS annee,
                                                                      act.ACT_COD,
                                                                      act.ACT_COE,
                                                                      EXE_SPE,
                                                                      cod.*
                                                            FROM &tbl_patients. pop
                                                                      INNER JOIN T_&domaine.&an.CSTC c
                                                                                ON          pop.BEN_NIR_PSA =
c.NIR_ANO_17
                                                                      INNER JOIN T_&domaine.&an.&table. act
                                                                                ON          c.&var_join1. = act.&var_join1.
                                                                                AND         c.&var_join2. = act.&var_join2.
                                                                      , &tbl_codes. cod
                                                            WHERE SUBSTR(act.ACT_COD, 1, taille) = TRIM(cod.code_acte)
                                                                      AND c.NIR_ANO_17 NOT IN ('&cle_inc1.', '&cle_inc2.')
                                                                      AND c.EXE_SOI_DTD IS NOT NULL AND c.EXE_SOI_DTF IS
NOT NULL  AND PSH_MDT IN ('00', '07')
                                                                      %if &domaine. = MCO %then
                                                                                %do;
                                                                                          AND NIR_RET = '0' AND NAI_RET = '0'
AND SEX_RET = '0' AND IAS_RET = '0' AND ENT_DAT_RET = '0'
                                                                                          AND c.ETA_NUM NOT IN
('130780521', '130783236', '130783293', '130784234', '130804297', '600100101',
                                                                                                    '750041543', '750100018',
'750100042', '750100075', '750100083', '750100091', '750100109',
                                                                                                    '750100125', '750100166',
'750100208', '750100216', '750100232', '750100273', '750100299',
                                                                                                    '750801441', '750803447',
'750803454', '910100015', '910100023', '920100013', '920100021',
                                                                                                    '920100039', '920100047',
'920100054', '920100062', '930100011', '930100037', '930100045',
                                                                                                    '940100027', '940100035',
'940100043', '940100050', '940100068', '950100016', '690783154',
                                                                                                    '690784137', '690784152',
'690784178', '690787478', '830100558')
                                                                                %end;
```

```
                                                                %if &tbl_codes. = actes_MG or &tbl_codes. =
actes_pneumo %then
                                                                        %do;
                                                                                AND act.PSH_MDT IN ('00', '07')
                                                                        %end;
                                                        );

                        DISCONNECT FROM ORACLE;

                quit;

                data ACT_COD_&domaine._&table._&an.;
                        set ACT_COD_&domaine._&table._&an.;
                        length date_debut date_fin 4.;
                        table = "&table.";
                        domaine = "&domaine.";
                        type = "PMSI ACE";
                        date_debut = datepart(EXE_SOI_DTD);
                        date_fin = datepart(EXE_SOI_DTF);
                        format date_debut date_fin ddmmyy10.;
                        drop EXE_SOI_DTD EXE_SOI_DTF;
                        if (year(date_fin) = 20&an. AND 0 <= yrdif(date_debut, date_fin, 'act/act') <= 1) then output;
                run;

                %arret_erreur;

        %end;

%mend ACT_COD_ACE;

*
        ************************************************************************************************************
**********************************          ;
*       Macro finale pour appeler les précédentes, pour chaque table du PMSI

        ;
*
        ************************************************************************************************************
**********************************          ;

%macro extract_ACT_COD_PMSI(annee_deb=, annee_fin=, MCO = 1, SSR = 1, tbl_out=, tbl_codes=, tbl_patients=);

        %let an_deb_pmsi = %sysevalf(&annee_deb. - 2000);
        %let an_fin_pmsi = %sysevalf(&annee_fin. - 2000);

        * Pour le PMSI-MCO;
        %if &MCO. = 1 %then
                %do;

                        %let domaine = MCO;

                        * Dans les tables FB;
                        %ACT_COD_sejours(
                                an_deb_pmsi = &an_deb_pmsi.,
                                an_fin_pmsi = &an_fin_pmsi.,
                                an_crea_t = 06,
                                an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY()))) - 2000),
                                tbl_codes = &tbl_codes.,
                                table = FB,
                                domaine = &domaine.,
                                tbl_out = &tbl_out.,
                                tbl_patients = &tbl_patients.
                                );

                        * Dans les tables FC;
                        %ACT_COD_sejours(
                                an_deb_pmsi = &an_deb_pmsi.,
                                an_fin_pmsi = &an_fin_pmsi.,
                                an_crea_t = 06,
                                an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY()))) - 2000),
                                tbl_codes = &tbl_codes.,
```

```
                                table = FC,
                                domaine = &domaine.,
                                tbl_out = &tbl_out.,
                                tbl_patients = &tbl_patients.
                                );

                        * Dans les tables FBSTC;
                        %ACT_COD_ACE(
                                an_deb_pmsi = &an_deb_pmsi.,
                                an_fin_pmsi = &an_fin_pmsi.,
                                an_crea_t = 08,
                                an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
                                tbl_codes = &tbl_codes.,
                                table = FBSTC,
                                domaine = &domaine.,
                                tbl_out = &tbl_out.,
                                tbl_patients = &tbl_patients.
                                );

                        * Dans les tables FCSTC;
                        %ACT_COD_ACE(
                                an_deb_pmsi = &an_deb_pmsi.,
                                an_fin_pmsi = &an_fin_pmsi.,
                                an_crea_t = 08,
                                an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
                                tbl_codes = &tbl_codes.,
                                table = FCSTC,
                                domaine = &domaine.,
                                tbl_out = &tbl_out.,
                                tbl_patients = &tbl_patients.
                                );


        %end;
        * Fin du PMSI-MCO;

* Pour le PMSI-SSR;
%if &SSR. = 1 %then
        %do;

                        %let domaine = SSR;

                        * Dans les tables FB;
                        %ACT_COD_sejours(
                                an_deb_pmsi = &an_deb_pmsi.,
                                an_fin_pmsi = &an_fin_pmsi.,
                                an_crea_t = 06,
                                an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
                                tbl_codes = &tbl_codes.,
                                table = FB,
                                domaine = &domaine.,
                                tbl_out = &tbl_out.,
                                tbl_patients = &tbl_patients.
                                );

                        * Dans les tables FC;
                        %ACT_COD_sejours(
                                an_deb_pmsi = &an_deb_pmsi.,
                                an_fin_pmsi = &an_fin_pmsi.,
                                an_crea_t = 06,
                                an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
                                tbl_codes = &tbl_codes.,
                                table = FC,
                                domaine = &domaine.,
                                tbl_out = &tbl_out.,
                                tbl_patients = &tbl_patients.
                                );

                        * Dans les tables FBSTC;
                        %ACT_COD_ACE(
                                an_deb_pmsi = &an_deb_pmsi.,
```

```
                                        an_fin_pmsi = &an_fin_pmsi.,
                                        an_crea_t = 13,
                                        an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
                                        tbl_codes = &tbl_codes.,
                                        table = FBSTC,
                                        domaine = &domaine.,
                                        tbl_out = &tbl_out.,
                                        tbl_patients = &tbl_patients.
                                        );

                                * Dans les tables FCSTC;
                                %ACT_COD_ACE(
                                        an_deb_pmsi = &an_deb_pmsi.,
                                        an_fin_pmsi = &an_fin_pmsi.,
                                        an_crea_t = 13,
                                        an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
                                        tbl_codes = &tbl_codes.,
                                        table = FCSTC,
                                        domaine = &domaine.,
                                        tbl_out = &tbl_out.,
                                        tbl_patients = &tbl_patients.
                                        );

                        %end;
                        * Fin du PMSI-SSR;

        * Empilage de la table temporaire à la table Résultat;
        data &tbl_out.;
                        length type $12. table $15.;
                        set ACT_COD_:;
        run;

        %arret_erreur;

        * Suppression de la table temporaire;
        proc datasets library = work memtype = data nolist;
                        delete ACT_COD_:;
        run;

%mend extract_ACT_COD_PMSI;
```

3.1.13 Macro_Reperage_des_soins_Codes_GME_PMSI_SSR.sas

```
/*
	*****************************************************************************************************
************************************	*/
/*
																	*/
/*		Macro pour repérer les remboursement des codes GME souhaités, dans le PMSI : extract_GME_PMSI
									*/
/*
																	*/
/*
	*****************************************************************************************************
************************************	*/
/*
																	*/
/*		Arguments en entrée :
									*/
/*
																	*/
/*			- annee_deb = la première année de soins (ie. année des tables PMSI) pour laquelle on souhaite
repérer les codes			*/
/*
																	*/
/*			- annee_fin = la dernière année de soins (ie. année des tables PMSI) pour laquelle on souhaite repérer
les codes			*/
/*
																	*/
/*			- tbl_out = le nom de la table en sortie (peut contenir un nom de librairie)
									*/
/*
																	*/
/*			- tbl_codes = la table contenant les codes GME à repérer (au format Oracle)
																	*/
/*
																	*/
/*			- tbl_patients = le nom de la table contenant la correspondance BEN_IDT_ANO <-> BEN_NR_PSA (au
format Oracle)			*/
/*
																	*/
/*
	*****************************************************************************************************
************************************	*/
/*
																	*/
/*		Tables en sortie : &tbl_out. (renseignée en paramètres de la macro)

		*/
/*		avec sélection des codes disponibles dans le référentiel &tbl_codes., hors séjours en erreur
									*/
/*
																	*/
/*
	*****************************************************************************************************
************************************	*/
```

```
*
	*************************************************************************************************************
**********************************	;
*	Macro pour repérer les codes GME dans le PMSI - séjours

	;
*
	*************************************************************************************************************
**********************************	;

%macro extract_GME_PMSI(annee_deb=, annee_fin=, an_crea_t = 14 , an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))) - 2000),
tbl_codes=,
	tbl_out=, tbl_patients=);

	%let an_deb_pmsi = %sysevalf(&annee_deb. - 2000);
	%let an_fin_pmsi = %sysevalf(&annee_fin. - 2000);

	%do i = %sysfunc(max(&an_deb_pmsi., &an_crea_t. )) %to %sysfunc(min(&an_fin_pmsi., &an_fin_t.));

		%if &i. < 10 %then %let an = 0&i.;
		%else %let an = &i.;

		* On définit les variables de jointure pour chaque domaine du PMSI;
		%let var_join1 = ETA_NUM;
		%let var_join2 = RHA_NUM;

		* On joint la table GME avec la table des séjours en filtrant sur les codes GME sélectionnés;
		%put Récupération des séjours du PMSI-SSR (table GME) contenant des actes GME pré-définis - Année &an. ...;

		proc sql;

			%connectora;

				CREATE TABLE ext_GME_PMSI_&an. AS
					SELECT *
					FROM CONNECTION TO ORACLE (
						SELECT DISTINCT
							c.&var_join1.,
							c.&var_join2.,
							pop.BEN_IDT_ANO,
							c.EXE_SOI_DTD,
							c.EXE_SOI_DTF,
							b.GRG_GME,
							b.HOS_TYP_UM,
							20&an. AS annee,
							'SSR' AS domaine,
							cod.*
						FROM &tbl_patients. pop
							INNER JOIN T_SSR&an.C c
								ON	pop.BEN_NIR_PSA = c.NIR_ANO_17
							INNER JOIN T_SSR&an.B b
								ON	c.&var_join1. = b.&var_join1.
								AND	c.&var_join2. = b.&var_join2.
							INNER JOIN T_SSR&an.E e
								ON	c.&var_join1. = e.ETA_NUM
							INNER JOIN &tbl_codes. cod
								ON SUBSTR(TRIM(b.GRG_GME), 1, 2) =
TRIM(cod.code_GME)
						WHERE c.NIR_ANO_17 NOT IN ('&cle_inc1.', '&cle_inc2.')
							AND c.NIR_RET = '0' AND c.NAI_RET = '0' AND c.SEX_RET =
'0' AND c.SEJ_RET = '0' AND c.FHO_RET = '0'
							AND c.PMS_RET = '0' AND c.DAT_RET = '0' AND
c.COH_NAI_RET = '0' AND c.COH_SEX_RET = '0'
					);

			DISCONNECT FROM ORACLE;

		quit;

		%arret_erreur;
```

```
	%end;

	* Empilage de la table temporaire à la table Résultat;
	data &tbl_out.;
		set ext_GME_PMSI_:;
		length date_debut date_fin 4.;
		type = "PMSI séjours";
		date_debut = datepart(EXE_SOI_DTD);
		date_fin = datepart(EXE_SOI_DTF);
		drop EXE_SOI_DTD EXE_SOI_DTF;
		format date_debut date_fin ddmmyy10.;
	run;

	%arret_erreur;

	* Suppression de la table temporaire;
	proc datasets library = work memtype = data nolist;
		delete ext_GME_PMSI_:;
	run;

%mend extract_GME_PMSI;
```

3.1.14 Macro_Reperage_des_soins_CSARR_PMSI_SSR.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*

*/
/*			Macro pour repérer les remboursement des codes CSARR souhaités, dans le PMSI : extract_CSARR_PMSI
*/
/*

*/
/*
	*****************************************************************************************************************
**********************************	*/
/*

*/
/*			Arguments en entrée :

*/
/*

*/
/*				- annee_deb = la première année de soins (ie. année des tables PMSI) pour laquelle on souhaite
repérer les codes	*/
/*

*/
/*				- annee_fin = la dernière année de soins (ie. année des tables PMSI) pour laquelle on souhaite repérer
les codes	*/
/*

*/
/*				- tbl_out = le nom de la table en sortie (peut contenir un nom de librairie)
*/
/*

*/
/*				- tbl_codes = la table contenant les codes CSARR à repérer (au format Oracle)
*/
/*

*/
/*				- tbl_patients = le nom de la table contenant la correspondance BEN_IDT_ANO <-> BEN_NR_PSA (au
format Oracle)	*/
/*

*/
/*
	*****************************************************************************************************************
**********************************	*/
/*

*/
/*			Tables en sortie : &tbl_out. (renseignée en paramètres de la macro)

*/
/*			avec sélection des codes disponibles dans le référentiel &tbl_codes., hors séjours en erreur
*/
/*

*/
/*
	*****************************************************************************************************************
**********************************	*/
```

```
*
	*************************************************************************************************************
**********************************	;
*	Macro pour repérer les codes CSARR dans le PMSI - séjours

				;
*
	*************************************************************************************************************
**********************************	;

%macro extract_CSARR_PMSI(annee_deb=, annee_fin=, an_crea_t = 14 , an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))) - 2000),
tbl_codes=,
	tbl_out=, tbl_patients=);

	%let an_deb_pmsi = %sysevalf(&annee_deb. - 2000);
	%let an_fin_pmsi = %sysevalf(&annee_fin. - 2000);

	%do i = %sysfunc(max(&an_deb_pmsi., &an_crea_t. )) %to %sysfunc(min(&an_fin_pmsi., &an_fin_t.));

		%if &i. < 10 %then %let an = 0&i.;
		%else %let an = &i.;

		* On définit les variables de jointure pour chaque domaine du PMSI;
		%let var_join1 = ETA_NUM;
		%let var_join2 = RHA_NUM;

		* On joint la table CSARR avec la table des séjours en filtrant sur les codes CSARR sélectionnés;
		%put Récupération des séjours du PMSI-SSR (table CSARR) contenant des actes CSARR pré-définis - Année &an. ...;

		proc sql;

			%connectora;

				CREATE TABLE ext_CSARR_PMSI_&an. AS
					SELECT *
					FROM CONNECTION TO ORACLE (
						SELECT DISTINCT
							c.&var_join1.,
							c.&var_join2.,
							pop.BEN_IDT_ANO,
							c.EXE_SOI_DTD,
							c.EXE_SOI_DTF,
							20&an. AS annee,
							'SSR' AS domaine,
							'CSARR' AS nom_table,
							SUBSTR(b.AUT_TYP_UM, 1, 2) AS type_UM,
							csarr.NBR_CSARR AS nb_deliv,
							cod.*
						FROM &tbl_patients. pop
							INNER JOIN T_SSR&an.C c
								ON	pop.BEN_NIR_PSA = c.NIR_ANO_17
							INNER JOIN T_SSR&an.B b
								ON	c.&var_join1. = b.&var_join1.
								AND	c.&var_join2. = b.&var_join2.
							INNER JOIN T_SSR&an.E e
								ON	c.&var_join1. = e.ETA_NUM
							INNER JOIN T_SSR&an.CSARR csarr
								ON	c.&var_join1. = csarr.&var_join1.
								AND	c.&var_join2. = csarr.&var_join2.
							INNER JOIN &tbl_codes. cod
								ON TRIM(csarr.CSARR_COD) =
TRIM(cod.code_CSARR)
						WHERE c.NIR_ANO_17 NOT IN ('&cle_inc1.', '&cle_inc2.') AND
csarr.NBR_CSARR > 0
							AND c.NIR_RET = '0' AND c.NAI_RET = '0' AND c.SEX_RET =
'0' AND c.SEJ_RET = '0' AND c.FHO_RET = '0'
							AND c.PMS_RET = '0' AND c.DAT_RET = '0' %if &an. > 12
%then %do; AND c.COH_NAI_RET = '0'
							AND c.COH_SEX_RET = '0' %end;
						);
```

```
                                DISCONNECT FROM ORACLE;

                        quit;

                        %arret_erreur;

                %end;

                * Empilage de la table temporaire à la table Résultat;
                data &tbl_out.;
                        set ext_CSARR_PMSI_:;
                        length date_debut date_fin 4.;
                        type = "PMSI séjours";
                        date_debut = datepart(EXE_SOI_DTD);
                        date_fin = datepart(EXE_SOI_DTF);
                        format date_debut date_fin ddmmyy10.;
                        drop EXE_SOI_DTD EXE_SOI_DTF;
                run;

                %arret_erreur;

                * Suppression de la table temporaire;
                proc datasets library = work memtype = data nolist;
                        delete ext_CSARR_PMSI_:;
                run;

%mend extract_CSARR_PMSI;
```

3.1.15 Macro_Reperage_des_soins_GHM_PMSI_MCO.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*

*/
/*		Macro pour repérer les remboursement via des codes GHM souhaités, dans le PMSI : extract_GHM_PMSI
*/
/*

*/
/*
	*****************************************************************************************************************
**********************************	*/
/*

*/
/*		Arguments en entrée :

*/
/*

*/
/*			- annee_deb = la première année de soins (ie. année des tables PMSI) pour laquelle on souhaite
repérer les codes	*/
/*

*/
/*			- annee_fin = la dernière année de soins (ie. année des tables PMSI) pour laquelle on souhaite repérer
les codes	*/
/*

*/
/*			- tbl_out = le nom de la table en sortie (peut contenir un nom de librairie)
*/
/*

*/
/*			- tbl_codes = la table contenant les codes GHM à repérer (au format Oracle)
*/
/*

*/
/*			- tbl_patients = le nom de la table contenant la correspondance BEN_IDT_ANO <-> BEN_NR_PSA (au
format Oracle)	*/
/*

*/
/*
	*****************************************************************************************************************
**********************************	*/
/*

*/
/*		Tables en sortie : &tbl_out. (renseignée en paramètres de la macro)

*/
/*		avec sélection des codes disponibles dans le référentiel &tbl_codes., hors séjours en erreur
*/
/*

*/
/*
	*****************************************************************************************************************
**********************************	*/
```

```
*
	*****************************************************************************************************************************************
**********************************	;
*	Macro pour repérer les codes GHM dans le PMSI - séjours

	;
*
	*****************************************************************************************************************************************
**********************************	;

%macro extract_GHM_PMSI(annee_deb=, annee_fin=, an_crea_t = 06, an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
tbl_codes=,
	tbl_out=, tbl_patients=);

	%let an_deb_pmsi = %sysevalf(&annee_deb. - 2000);
	%let an_fin_pmsi = %sysevalf(&annee_fin. - 2000);

	%do i = %sysfunc(max(&an_deb_pmsi., &an_crea_t. )) %to %sysfunc(min(&an_fin_pmsi., &an_fin_t.));

		%if &i. < 10 %then %let an = 0&i.;
		%else %let an = &i.;

		* On définit les variables de jointure pour chaque domaine du PMSI;
		%let var_join1 = ETA_NUM;
		%let var_join2 = RSA_NUM;

		* On récupère les séjours en filtrant sur les codes GHM sélectionnés;
		%put Récupération des séjours du PMSI-MCO contenant des GHM pré-définis - Année &an. ...;

		proc sql;

			%connectora;

				CREATE TABLE ext_GHM_PMSI_&an. AS
					SELECT *
					FROM CONNECTION TO ORACLE (
						SELECT DISTINCT
							c.&var_join1.,
							c.&var_join2.,
							pop.BEN_IDT_ANO,
							c.EXE_SOI_DTD,
							c.EXE_SOI_DTF,
							20&an. AS annee,
							b.GRG_GHM,
							b.SEJ_NBJ,
							cod.*
						FROM &tbl_patients. pop
							INNER JOIN T_MCO&an.C c
								ON	pop.BEN_NIR_PSA = c.NIR_ANO_17
							INNER JOIN T_MCO&an.B b
								ON	c.&var_join1. = b.&var_join1.
								AND	c.&var_join2. = b.&var_join2.
							, &tbl_codes. cod
						WHERE SUBSTR(TRIM(b.GRG_GHM), 1, taille) = TRIM(cod.code_GHM)
							AND c.NIR_ANO_17 NOT IN ('&cle_inc1.', '&cle_inc2.')
							AND NIR_RET = '0' AND NAI_RET = '0' AND SEX_RET = '0'
AND SEJ_RET = '0' AND FHO_RET = '0'
							AND PMS_RET = '0' AND DAT_RET = '0' %if &an. > 12 %then
%do; AND COH_NAI_RET = '0' AND COH_SEX_RET = '0' %end;
							AND c.ETA_NUM NOT IN ('130780521', '130783236',
'130783293', '130784234', '130804297', '600100101', '750041543',
								'750100018', '750100042', '750100075',
'750100083', '750100091', '750100109', '750100125', '750100166',
								'750100208', '750100216', '750100232',
'750100273', '750100299', '750801441', '750803447', '750803454',
								'910100015', '910100023', '920100013',
'920100021', '920100039', '920100047', '920100054', '920100062',
								'930100011', '930100037', '930100045',
'940100027', '940100035', '940100043', '940100050', '940100068',
								'950100016', '690783154', '690784137',
'690784152', '690784178', '690787478', '830100558')
```

```
                                                                AND GRG_GHM NOT IN ('90Z00Z') AND GRG_RET NOT IN
('024') AND GRG_GHM NOT IN ('90H01Z', '90Z00Z', '90Z01Z',
                                                                        '90Z02Z', '90Z03Z')
                                                                AND ((SEJ_TYP = 'A' OR SEJ_TYP IS NULL) OR (SEJ_TYP = 'B'
AND GRG_GHM NOT IN ('28Z14Z','28Z15Z','28Z16Z')))
                                                        );

                                DISCONNECT FROM ORACLE;

                        quit;

                        %arret_erreur;

        %end;

        * Empilage de la table temporaire à la table Résultat;
        data &tbl_out.;
                        set ext_GHM_PMSI_:;
                        length date_debut date_fin 4.;
                        type = "PMSI séjours";
                        date_debut = datepart(EXE_SOI_DTD);
                        date_fin = datepart(EXE_SOI_DTF);
                        format date_debut date_fin ddmmyy10.;
                        drop EXE_SOI_DTD EXE_SOI_DTF;
        run;

        %arret_erreur;

        * Suppression de la table temporaire;
        proc datasets library = work memtype = data nolist;
                        delete ext_GHM_PMSI_:;
        run;

%mend extract_GHM_PMSI;
```

3.1.16 Macro_Reperage_des_Soins_LPP_DCIR.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*			Macro pour repérer les remboursements des codes LPP souhaités : extract_LPP_DCIR
                                                                                                                    */
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*			Arguments en entrée :
                                                                                                                    */
/*
                                                                                                                    */
/*				- annee_deb = la première année de soins (ie. filtre sur EXE_SOI_DTD) pour laquelle on souhaite
repérer les codes                                                */
/*
                                                                                                                    */
/*				- annee_fin = la dernière année de soins (ie. filtre sur EXE_SOI_DTD) pour laquelle on souhaite repérer
les codes                                                */
/*
                                                                                                                    */
/*				- tbl_out = le nom de la table en sortie (peut contenir un nom de librairie)
                                                                                                                    */
/*
                                                                                                                    */
/*				- tbl_codes = la table contenant les codes LPP à repérer (au format Oracle)
                                                                                                                    */
/*
                                                                                                                    */
/*				- tbl_patients = le nom de la table contenant la correspondance BEN_IDT_ANO <-> BEN_NR_PSA (au
format Oracle)                                                */
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*			Tables en sortie : &tbl_out. (renseignée en paramètres de la macro)

                                                                                                                    */
/*			avec sélection des codes disponibles dans le référentiel, filtrée sur les soins de ville uniquement, les NIR
"normaux" + suppression	*/
/*			des lignes pour information
                                                                                                                    */
/*
                                                                                                                    */
```

```
/*
	*****************************************************************************************************
**********************************	*/

%macro extract_LPP_DCIR(annee_deb=, annee_fin=, tbl_out=, tbl_codes=, tbl_patients=);

	%let annee_fin_flx = %sysevalf(&annee_fin. + 1);

	* On boucle sur les années de repérage (en flux);
	%do Annee = &annee_deb. %to &annee_fin_flx.;

		* On boucle sur les 12 mois de l année;
		%do i = 1 %to 12;

			%if &i. < 10 %then %let Mois = 0&i.;
			%else %let Mois = &i.;

			%put Données de &Mois./&Annee.;

			proc sql;

				%connectora;

					CREATE TABLE tmp_extract_LPP_DCIR_&Annee._&Mois. AS
						SELECT * FROM CONNECTION TO ORACLE (
							SELECT
								pop.BEN_IDT_ANO,
								prs.EXE_SOI_DTD,
								prs.EXE_SOI_DTF,
								prs.BSE_REM_MNT,
								tip.TIP_ACT_QSN,
								tip.TIP_PRS_IDE,
								ete.ETB_EXE_FIN,
								ete.ETE_MCO_DDP,
								cod.*
							FROM &tbl_patients. pop
								INNER JOIN ER_PRS_F prs
									ON	pop.BEN_NIR_PSA =
prs.BEN_NIR_PSA
								INNER JOIN ER_TIP_F tip
									ON	tip.FLX_DIS_DTD =
prs.FLX_DIS_DTD
									AND	tip.FLX_TRT_DTD =
prs.FLX_TRT_DTD
									AND	tip.FLX_EMT_TYP =
prs.FLX_EMT_TYP
									AND	tip.FLX_EMT_NUM =
prs.FLX_EMT_NUM
									AND	tip.FLX_EMT_ORD =
prs.FLX_EMT_ORD
									AND	tip.ORG_CLE_NUM =
prs.ORG_CLE_NUM
									AND	tip.DCT_ORD_NUM =
prs.DCT_ORD_NUM
									AND	tip.PRS_ORD_NUM =
prs.PRS_ORD_NUM
									AND	tip.REM_TYP_AFF =
prs.REM_TYP_AFF
								INNER JOIN &tbl_codes. cod
									ON tip.TIP_PRS_IDE =
TRIM(cod.code_LPP)
								LEFT JOIN ER_ETE_F ete
									ON	ete.FLX_DIS_DTD =
prs.FLX_DIS_DTD
									AND	ete.FLX_TRT_DTD =
prs.FLX_TRT_DTD
									AND	ete.FLX_EMT_TYP =
prs.FLX_EMT_TYP
									AND	ete.FLX_EMT_NUM =
prs.FLX_EMT_NUM
```

```
                                                                                AND         ete.FLX_EMT_ORD =
prs.FLX_EMT_ORD
                                                                                AND         ete.ORG_CLE_NUM =
prs.ORG_CLE_NUM
                                                                                AND         ete.DCT_ORD_NUM =
prs.DCT_ORD_NUM
                                                                                AND         ete.PRS_ORD_NUM =
prs.PRS_ORD_NUM
                                                                                AND         ete.REM_TYP_AFF =
prs.REM_TYP_AFF
                                                                        WHERE  prs.BEN_CDI_NIR = '00' AND prs.CPL_MAJ_TOP IN
(0, 1)
                                                                                AND (ete.ETE_IND_TAA NOT IN (1, 2) OR
ete.ETE_IND_TAA IS NULL)
                                                                                AND prs.DPN_QLF NOT IN (71, 72) AND
NOT(prs.DPN_QLF = 0 AND prs.PRS_DPN_QLP = 71)
                                                                                AND prs.FLX_DIS_DTD =
TO_DATE(%str(%'&Annee.&Mois.01%'), 'YYYYMMDD')
                                                                                AND prs.EXE_SOI_DTD BETWEEN
TO_DATE(%str(%'&Annee_deb.0101%'), 'YYYYMMDD') AND TO_DATE(%str(%'&Annee_fin.1231%'), 'YYYYMMDD')
                                                                        );

                                        disconnect from oracle;

                                quit;

                                %arret_erreur;

                                * On empile toutes les tables mensuelles;
                                %if %sysfunc(exist(&tbl_out.)) = 1 %then
                                        %do;

                                                proc append base = &tbl_out. data = tmp_extract_LPP_DCIR_&annee._&mois.;
                                                run;

                                                %arret_erreur;

                                                proc delete data = tmp_extract_LPP_DCIR_&annee._&mois.;
                                                run; quit;

                                        %end;

                                %if %sysfunc(exist(&tbl_out.)) = 0 %then
                                        %do;

                                                data &tbl_out.;
                                                        set tmp_extract_LPP_DCIR_&annee._&mois.;
                                                run;

                                                %arret_erreur;

                                                proc delete data = tmp_extract_LPP_DCIR_&annee._&mois.;
                                                run; quit;

                                        %end;

                        %end;
                        * Fin de la boucle sur les 12 mois;

                %end;
                * Fin de la boucle par année;

                * On applique le programme de régularisations pour céer la table finale;
                proc sql undo_policy = none;

                        CREATE TABLE &tbl_out. AS
                                SELECT
                                        BEN_IDT_ANO,
                                        datepart(EXE_SOI_DTD) AS date_debut length = 4 format ddmmyy10.,
                                        CASE    WHEN datepart(EXE_SOI_DTF) IS NULL THEN datepart(EXE_SOI_DTD)
                                                        ELSE datepart(EXE_SOI_DTF)
```

```
                                                    END AS date_fin length = 4 format ddmmyy10.,
                                    "DCIR" AS type,
                                    TIP_PRS_IDE,
                                    reperage,
                                    ETB_EXE_FIN,
                                    ETE_MCO_DDP,
                                    SUM(BSE_REM_MNT) AS montant,
                                    SUM(TIP_ACT_QSN) AS quantite
                            FROM &tbl_out.
                            GROUP BY 1, 2, 3, 4, 5, 6, 7, 8;

                    DELETE FROM &tbl_out. WHERE montant <= 0 OR quantite < 0;

            quit;

            %arret_erreur;

%mend extract_LPP_DCIR;
```

3.1.17 Macro_Reperage_des_soins_Medecin_traitant.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*		Macro pour repérer les remboursement liés à une consultation chez le médecin traitant : extract_MT
                                                                                                                    */
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*		Arguments en entrée :
                                                                                                                    */
/*
                                                                                                                    */
/*			- annee_deb = la première année de soins (ie. filtre sur EXE_SOI_DTD) pour laquelle on souhaite
repérer les codes                                                */
/*
                                                                                                                    */
/*			- annee_fin = la dernière année de soins (ie. filtre sur EXE_SOI_DTD) pour laquelle on souhaite repérer
les codes                                                */
/*
                                                                                                                    */
/*			- tbl_out = le nom de la table en sortie (peut contenir un nom de librairie)
                                                                                                                    */
/*
                                                                                                                    */
/*			- tbl_patients = le nom de la table contenant la correspondance BEN_IDT_ANO <-> BEN_NR_PSA (au
format Oracle)                                                */
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*		Tables en sortie : &tbl_out. (renseignée en paramètres de la macro)

                                                                                                                    */
/*		filtrée sur les soins de ville uniquement, les NIR  normaux  + suppression des lignes pour information
                                                                                                                    */
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/

%macro extract_MT(annee_deb=, annee_fin=, tbl_out=, tbl_patients=);

	%let annee_fin_flx = %sysevalf(&annee_fin. + 1);

	* On boucle sur les années de repérage (en flux);
	%do Annee = &annee_deb. %to &annee_fin_flx.;
```

```
* On boucle sur les 12 mois de l année;
%do i = 1 %to 12;

	%if &i. < 10 %then %let Mois = 0&i.;
	%else %let Mois = &i.;

	%put Données de &Mois./&Annee.;

	proc sql;

		%connectora;

			CREATE TABLE tmp_extract_MT_&annee._&mois. AS SELECT * FROM
CONNECTION TO ORACLE (
				SELECT
					pop.BEN_IDT_ANO,
					prs.EXE_SOI_DTD,
					prs.EXE_SOI_DTF,
					prs.BSE_PRS_NAT,
					prs.PRS_ACT_QTE,
					prs.PSE_SPE_COD,
					prs.BSE_REM_MNT,
					prs.PFS_EXE_NUM,
					prs.PRS_MTT_NUM,
					ete.ETB_EXE_FIN,
					ete.ETE_MCO_DDP,
					ete.MDT_COD,
					ete.ETE_IND_TAA,
					CASE	WHEN prs.PFS_EXE_NUM =
prs.PRS_MTT_NUM THEN 1
						ELSE 0
						END AS contact_MT
				FROM &tbl_patients. pop
					INNER JOIN ER_PRS_F prs
						ON	pop.BEN_NIR_PSA =
prs.BEN_NIR_PSA
					LEFT JOIN ER_ETE_F ete
						ON	ete.FLX_DIS_DTD =
prs.FLX_DIS_DTD
						AND	ete.FLX_TRT_DTD =
prs.FLX_TRT_DTD
						AND	ete.FLX_EMT_TYP =
prs.FLX_EMT_TYP
						AND	ete.FLX_EMT_NUM =
prs.FLX_EMT_NUM
						AND	ete.FLX_EMT_ORD =
prs.FLX_EMT_ORD
						AND	ete.ORG_CLE_NUM =
prs.ORG_CLE_NUM
						AND	ete.DCT_ORD_NUM =
prs.DCT_ORD_NUM
						AND	ete.PRS_ORD_NUM =
prs.PRS_ORD_NUM
						AND	ete.REM_TYP_AFF =
prs.REM_TYP_AFF
				WHERE  prs.BEN_CDI_NIR = '00' AND prs.CPL_MAJ_TOP IN
(0, 1)
					AND (ete.ETE_IND_TAA NOT IN (1, 2) OR
ete.ETE_IND_TAA IS NULL)
					AND prs.DPN_QLF NOT IN (71, 72) AND
NOT(prs.DPN_QLF = 0 AND prs.PRS_DPN_QLP = 71)
					AND prs.FLX_DIS_DTD =
TO_DATE(%str(%'&Annee.&Mois.01%'), 'YYYYMMDD')
					AND prs.EXE_SOI_DTD BETWEEN
TO_DATE(%str(%'&Annee_deb.0101%'), 'YYYYMMDD') AND TO_DATE(%str(%'&Annee_fin.1231%'), 'YYYYMMDD')
					AND ((prs.PFS_EXE_NUM = prs.PRS_MTT_NUM
AND PRS_MTT_NUM NOT IN ('A01234567', 'A', ' ', '00000000', '99999999')
					OR (PSE_SPE_COD IN (1, 22, 23))) AND
prs.PRS_NAT_REF IN (SELECT code_presta FROM codes_presta_BPCO WHERE
					reperage = 33))
```

```
                                                                    );

                                        disconnect from oracle;

                                quit;

                                %arret_erreur;

                                * On empile toutes les tables mensuelles;
                                %if %sysfunc(exist(&tbl_out.)) = 1 %then
                                        %do;

                                                proc append base = &tbl_out. data = tmp_extract_MT_&annee._&mois.;
                                                run;

                                                %arret_erreur;

                                                proc delete data = tmp_extract_MT_&annee._&mois.;
                                                run; quit;

                                        %end;

                                %if %sysfunc(exist(&tbl_out.)) = 0 %then
                                        %do;

                                                data &tbl_out.;
                                                        set tmp_extract_MT_&annee._&mois.;
                                                run;

                                                %arret_erreur;

                                                proc delete data = tmp_extract_MT_&annee._&mois.;
                                                run; quit;

                                        %end;

                        %end;
                        * Fin de la boucle sur les 12 mois;

        %end;
        * Fin de la boucle par année;

        data &tbl_out.;
                set &tbl_out. (where = (ETE_IND_TAA not in (1, 2) and MDT_COD in (., 0, 7)));
        run;

        * On applique le programme de régularisations pour céer la table finale;
        proc sql undo_policy = none;

                CREATE TABLE &tbl_out. AS
                        SELECT
                                BEN_IDT_ANO,
                                datepart(EXE_SOI_DTD) AS date_debut length = 4 format ddmmyy10.,
                                CASE    WHEN datepart(EXE_SOI_DTF) IS NULL THEN datepart(EXE_SOI_DTD)
                                                ELSE datepart(EXE_SOI_DTF)
                                                END AS date_fin length = 4 format ddmmyy10.,
                                "DCIR" AS type,
                                BSE_PRS_NAT,
                                PSE_SPE_COD,
                                ETB_EXE_FIN,
                                ETE_MCO_DDP,
                                contact_MT,
                                SUM(BSE_REM_MNT) AS montant,
                                SUM(PRS_ACT_QTE) AS quantite
                        FROM &tbl_out.
                        GROUP BY 1, 2, 3, 4, 5, 6, 7, 8, 9;

                        DELETE FROM &tbl_out. WHERE montant <= 0 OR quantite < 0;

        quit;
```

```
        %arret_erreur;

%mend extract_MT;
```

3.1.18 Macro_Reperage_des_soins_NABM_DCIR.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*			Macro pour repérer les remboursement des codes NABM souhaités : extract_NABM_DCIR
                                                                                                                    */
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*			Arguments en entrée :
                                                                                                                    */
/*
                                                                                                                    */
/*				- annee_deb = la première année de soins (ie. filtre sur EXE_SOI_DTD) pour laquelle on souhaite
repérer les codes                                                */
/*
                                                                                                                    */
/*				- annee_fin = la dernière année de soins (ie. filtre sur EXE_SOI_DTD) pour laquelle on souhaite repérer
les codes                                                */
/*
                                                                                                                    */
/*				- tbl_out = le nom de la table en sortie (peut contenir un nom de librairie)
                                                                                                                    */
/*
                                                                                                                    */
/*				- tbl_codes = la table contenant les codes NABM à repérer (au format Oracle)
                                                                                                                    */
/*
                                                                                                                    */
/*				- tbl_patients = le nom de la table contenant la correspondance BEN_IDT_ANO <-> BEN_NR_PSA (au
format Oracle)                                                */
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*			Tables en sortie : &tbl_out. (renseignée en paramètres de la macro)

                                                                                                                    */
/*			avec sélection des codes disponibles dans le référentiel, filtrée sur les soins de ville uniquement, les NIR
"normaux" + suppression	*/
/*			des lignes pour information
                                                                                                                    */
/*
                                                                                                                    */
```

```
/*
	*****************************************************************************************************
**********************************	*/

%macro extract_NABM_DCIR(annee_deb=, annee_fin=, tbl_out=, tbl_codes=, tbl_patients=);

	%let annee_fin_flx = %sysevalf(&annee_fin. + 1);

	* On boucle sur les années de repérage (en flux);
	%do Annee = &annee_deb. %to &annee_fin_flx.;

		* On boucle sur les 12 mois de l année;
		%do i = 1 %to 12;

			%if &i. < 10 %then %let Mois = 0&i.;
			%else %let Mois = &i.;

			%put Données de &Mois./&Annee.;

			proc sql;

				%connectora;

					CREATE TABLE tmp_extract_NABM_DCIR_&Annee._&Mois. AS
						SELECT * FROM CONNECTION TO ORACLE (
							SELECT
								pop.BEN_IDT_ANO,
								prs.EXE_SOI_DTD,
								prs.EXE_SOI_DTF,
								prs.BSE_REM_MNT,
								bio.BIO_ACT_QSN,
								bio.BIO_PRS_IDE,
								ete.ETB_EXE_FIN,
								ete.ETE_MCO_DDP,
								cod.*
							FROM &tbl_patients. pop
								INNER JOIN ER_PRS_F prs
									ON	pop.BEN_NIR_PSA =
prs.BEN_NIR_PSA
								INNER JOIN ER_BIO_F bio
									ON	bio.FLX_DIS_DTD =
prs.FLX_DIS_DTD
									AND	bio.FLX_TRT_DTD =
prs.FLX_TRT_DTD
									AND	bio.FLX_EMT_TYP =
prs.FLX_EMT_TYP
									AND	bio.FLX_EMT_NUM =
prs.FLX_EMT_NUM
									AND	bio.FLX_EMT_ORD =
prs.FLX_EMT_ORD
									AND	bio.ORG_CLE_NUM =
prs.ORG_CLE_NUM
									AND	bio.DCT_ORD_NUM =
prs.DCT_ORD_NUM
									AND	bio.PRS_ORD_NUM =
prs.PRS_ORD_NUM
									AND	bio.REM_TYP_AFF =
prs.REM_TYP_AFF
								INNER JOIN &tbl_codes. cod
									ON bio.BIO_PRS_IDE =
TRIM(cod.code_NABM)
								LEFT JOIN ER_ETE_F ete
									ON	ete.FLX_DIS_DTD =
prs.FLX_DIS_DTD
									AND	ete.FLX_TRT_DTD =
prs.FLX_TRT_DTD
									AND	ete.FLX_EMT_TYP =
prs.FLX_EMT_TYP
									AND	ete.FLX_EMT_NUM =
prs.FLX_EMT_NUM
```

```
                                                                        AND     ete.FLX_EMT_ORD =
prs.FLX_EMT_ORD
                                                                        AND     ete.ORG_CLE_NUM =
prs.ORG_CLE_NUM
                                                                        AND     ete.DCT_ORD_NUM =
prs.DCT_ORD_NUM
                                                                        AND     ete.PRS_ORD_NUM =
prs.PRS_ORD_NUM
                                                                        AND     ete.REM_TYP_AFF =
prs.REM_TYP_AFF
                                                                WHERE  prs.BEN_CDI_NIR = '00' AND prs.CPL_MAJ_TOP IN
(0, 1)
                                                                        AND (ete.ETE_IND_TAA NOT IN (1, 2) OR
ete.ETE_IND_TAA IS NULL)
                                                                        AND prs.DPN_QLF NOT IN (71, 72) AND
NOT(prs.DPN_QLF = 0 AND prs.PRS_DPN_QLP = 71)
                                                                        AND prs.FLX_DIS_DTD =
TO_DATE(%str(%'&Annee.&Mois.01%'), 'YYYYMMDD')
                                                                        AND prs.EXE_SOI_DTD BETWEEN
TO_DATE(%str(%'&Annee_deb.0101%'), 'YYYYMMDD') AND TO_DATE(%str(%'&Annee_fin.1231%'), 'YYYYMMDD')
                                                                );

                                        disconnect from oracle;

                                quit;

                                %arret_erreur;

                                * On empile toutes les tables mensuelles;
                                %if %sysfunc(exist(&tbl_out.)) = 1 %then
                                        %do;

                                                proc append base = &tbl_out. data =
tmp_extract_NABM_DCIR_&annee._&mois.;
                                                run;

                                                %arret_erreur;

                                                proc delete data = tmp_extract_NABM_DCIR_&annee._&mois.;
                                                run; quit;

                                        %end;

                                %if %sysfunc(exist(&tbl_out.)) = 0 %then
                                        %do;

                                                data &tbl_out.;
                                                        set tmp_extract_NABM_DCIR_&annee._&mois.;
                                                run;

                                                %arret_erreur;

                                                proc delete data = tmp_extract_NABM_DCIR_&annee._&mois.;
                                                run; quit;

                                        %end;

                        %end;
                        * Fin de la boucle sur les 12 mois;

                %end;
                * Fin de la boucle par année;

                * On applique le programme de régularisations pour céer la table finale;
                proc sql undo_policy = none;

                        CREATE TABLE &tbl_out. AS
                                SELECT
                                        BEN_IDT_ANO,
                                        datepart(EXE_SOI_DTD) AS date_debut length = 4 format ddmmyy10.,
                                        CASE    WHEN datepart(EXE_SOI_DTF) IS NULL THEN datepart(EXE_SOI_DTD)
```

```
                                                    ELSE datepart(EXE_SOI_DTF)
                                                    END AS date_fin length = 4 format ddmmyy10.,
                                    "DCIR" AS type,
                                    BIO_PRS_IDE,
                                    reperage,
                                    ETB_EXE_FIN,
                                    ETE_MCO_DDP,
                                    SUM(BSE_REM_MNT) AS montant,
                                    SUM(BIO_ACT_QSN) AS quantite
                            FROM &tbl_out.
                            GROUP BY 1, 2, 3, 4, 5, 6, 7, 8;

                    DELETE FROM &tbl_out. WHERE montant <= 0 OR quantite < 0;

            quit;

            %arret_erreur;

%mend extract_NABM_DCIR;
```

3.1.19 Macro_Reperage_des_soins_NABM_PMSI.sas

```
/*
        ************************************************************************************************************
**********************************        */
/*

                                                                                                                    */
/*                      Macro pour repérer les remboursement des codes NABM souhaités, dans le PMSI : extract_NABM_PMSI
                                                                                                                    */
/*

                                                                                                                    */
/*
        ************************************************************************************************************
**********************************        */
/*

                                                                                                                    */
/*                      Arguments en entrée :

                                                                                                                    */
/*

                                                                                                                    */
/*                              - annee_deb = la première année de soins (ie. année des tables PMSI) pour laquelle on souhaite
repérer les codes                                                                                                   */
/*

                                                                                                                    */
/*                              - annee_fin = la dernière année de soins (ie. année des tables PMSI) pour laquelle on souhaite repérer
les codes                                                                                                           */
/*

                                                                                                                    */
/*                              - tbl_out = le nom de la table en sortie (peut contenir un nom de librairie)
                                                                                                                    */
/*

                                                                                                                    */
/*                              - tbl_codes = la table contenant les codes NABM à repérer (au format Oracle)
                                                                                                                    */
/*

                                                                                                                    */
/*                              - tbl_patients = le nom de la table contenant la correspondance BEN_IDT_ANO <-> BEN_NR_PSA (au
format Oracle)                                                                                                      */
/*

                                                                                                                    */
/*                              - HAD = flag (0/1) : si 1, on recherche les codes NABM dans le PMSI-HAD (Par défaut : 1)
                                                                                                                    */
/*

                                                                                                                    */
/*                              - MCO = flag (0/1) : si 1, on recherche les codes NABM dans le PMSI-MCO (sejours et ACE) (Par défaut :
1)                                                                                                                  */
/*

                                                                                                                    */
/*                              - RIP = flag (0/1) : si 1, on recherche les codes NABM dans le PMSI-RIP (Par défaut : 1)
                                                                                                                    */
/*

                                                                                                                    */
/*                              - SSR = flag (0/1) : si 1, on recherche les codes NABM dans le PMSI-SSR (sejours et ACE) (Par défaut : 1)
                                                                                                                    */
```

```
/*
                                                                                                                  */
/*      ****************************************************************************************************
***********************************          */
/*
                                                                                                                  */
/*                  Tables en sortie : &tbl_out. (renseignée en paramètres de la macro)
                    */
/*                  avec sélection des codes disponibles dans le référentiel &tbl_codes., hors séjours en erreur
                                                                                                */
/*
                                                                                                                  */
/*      ****************************************************************************************************
***********************************          */

*       ****************************************************************************************************
***********************************          ;
*       Macro pour repérer les codes NABM dans le PMSI - séjours
                              ;
*       ****************************************************************************************************
***********************************          ;

%macro NABM_sejours(an_deb_pmsi=, an_fin_pmsi=, an_crea_t=, an_fin_t=, tbl_codes=, table=, domaine=, var_NABM=, var_qte=,
tbl_out=, tbl_patients=);

        %do i = %sysfunc(max(&an_deb_pmsi., &an_crea_t. )) %to %sysfunc(min(&an_fin_pmsi., &an_fin_t.));

                %if &i. < 10 %then %let an = 0&i.;
                %else %let an = &i.;

                * On définit les variables de jointure pour chaque domaine du PMSI;
                %if &domaine. = HAD %then
                        %do;
                                %let var_join1 = ETA_NUM_EPMSI;
                                %let var_join2 = RHAD_NUM;
                        %end;
                %if &domaine. = MCO %then
                        %do;
                                %let var_join1 = ETA_NUM;
                                %let var_join2 = RSA_NUM;
                        %end;
                %if &domaine. = RIP %then
                        %do;
                                %let var_join1 = ETA_NUM_EPMSI;
                                %let var_join2 = RIP_NUM;
                        %end;
                %if &domaine. = SSR %then
                        %do;
                                %let var_join1 = ETA_NUM;
                                %let var_join2 = RHA_NUM;
                        %end;

                * On joint la table NABM avec la table des séjours en filtrant sur les codes NABM sélectionnés;
                %put Récupération des séjours du PMSI-&domaine. (table &table.) contenant des actes NABM pré-définis - Année
&an. ...;

                proc sql;

                        %connectora;

                                CREATE TABLE ext_NABM_PMSI_&domaine._&table._&an. AS SELECT * FROM
CONNECTION TO ORACLE (
```

```
SELECT DISTINCT
        c.&var_join1.,
        c.&var_join2.,
        pop.BEN_IDT_ANO,
        c.EXE_SOI_DTD,
        c.EXE_SOI_DTF,
        20&an. AS annee,
        &var_qte. AS nb_deliv,
        cod.*
FROM &tbl_patients. pop
        INNER JOIN T_&domaine.&an.C c
                ON      pop.BEN_NIR_PSA = c.NIR_ANO_17
        %if &domaine. = MCO or &domaine. = HAD or &domaine. = SSR %then
                %do;
                        INNER JOIN T_&domaine.&an.B b
                                ON      c.&var_join1. = b.&var_join1.
                                AND     c.&var_join2. = b.&var_join2.
                %end;
        INNER JOIN T_&domaine.&an.E e
                ON      c.&var_join1. = e.ETA_NUM
        INNER JOIN T_&domaine.&an.&table. nabm
                ON      c.&var_join1. = nabm.&var_join1.
                AND     c.&var_join2. = nabm.&var_join2.
        INNER JOIN &tbl_codes. cod
                ON TRIM(nabm.&var_NABM.) = TRIM(cod.code_NABM)
WHERE c.NIR_ANO_17 NOT IN ('&cle_inc1.', '&cle_inc2.') AND &var_qte. > 0
        %if &domaine. = HAD %then
                %do;
                        AND NIR_RET = '0' AND NAI_RET = '0' AND SEX_RET = '0' AND SEJ_RET = '0' AND FHO_RET = '0'
                        AND PMS_RET = '0' AND DAT_RET = '0' %if &an. > 12 %then %do; AND COH_NAI_RET = '0'
                        AND COH_SEX_RET = '0' %end;
                %end;
        %if &domaine. = MCO %then
                %do;
                        AND NIR_RET = '0' AND NAI_RET = '0' AND SEX_RET = '0' AND SEJ_RET = '0' AND FHO_RET = '0'
                        AND PMS_RET = '0' AND DAT_RET = '0' %if &an. > 12 %then %do; AND COH_NAI_RET = '0'
                        AND COH_SEX_RET = '0' %end;
                        AND c.ETA_NUM NOT IN ('130780521', '130783236', '130783293', '130784234', '130804297', '600100101',
                                '750041543', '750100018', '750100042', '750100075', '750100083', '750100091', '750100109',
                                '750100125', '750100166', '750100208', '750100216', '750100232', '750100273', '750100299',
                                '750801441', '750803447', '750803454', '910100015', '910100023', '920100013', '920100021',
                                '920100039', '920100047', '920100054', '920100062', '930100011', '930100037', '930100045',
                                '940100027', '940100035', '940100043', '940100050', '940100068', '950100016', '690783154',
                                '690784137', '690784152', '690784178', '690787478', '830100558')
                        AND GRG_GHM NOT IN ('90Z00Z') AND GRG_RET NOT IN ('024') AND GRG_GHM NOT IN ('90H01Z', '90Z00Z',
                                '90Z01Z', '90Z02Z', '90Z03Z')
                        AND ((SEJ_TYP = 'A' OR SEJ_TYP IS NULL) OR (SEJ_TYP = 'B' AND GRG_GHM NOT IN ('28Z14Z','28Z15Z','28Z16Z')))
                %end;
        %if &domaine. = RIP %then
                %do;
                        AND NIR_RET = '0' AND NAI_RET = '0' AND SEX_RET = '0' AND SEJ_RET = '0' AND FHO_RET = '0'
```

```
                                                                                AND PMS_RET = '0' AND DAT_RET =
'0' %if &an. > 12 %then %do; AND COH_NAI_RET = '0'
                                                                                AND COH_SEX_RET = '0' %end;
                                                                        %end;
                                                                %if &domaine. = SSR %then
                                                                        %do;
                                                                                AND NIR_RET = '0' AND NAI_RET = '0'
AND SEX_RET = '0' AND SEJ_RET = '0' AND FHO_RET = '0'
                                                                                AND PMS_RET = '0' AND DAT_RET =
'0' %if &an. > 12 %then %do; AND COH_NAI_RET = '0'
                                                                                AND COH_SEX_RET = '0' %end;
                                                                        %end;
                                                        );

                                DISCONNECT FROM ORACLE;

                        quit;

                        data ext_NABM_PMSI_&domaine._&table._&an.;
                                set ext_NABM_PMSI_&domaine._&table._&an.;
                                length date_debut date_fin 4.;
                                domaine = "&domaine.";
                                table = "&table.";
                                type = "PMSI séjours";
                                date_debut = datepart(EXE_SOI_DTD);
                                date_fin = datepart(EXE_SOI_DTF);
                                format date_debut date_fin ddmmyy10.;
                        run;

                        %arret_erreur;

                %end;

%mend NABM_sejours;

*
                ************************************************************************************************************
**********************************             ;
*               Macro pour repérer les codes NABM dans le PMSI - ACE

                                                ;
*
                ************************************************************************************************************
**********************************             ;

%macro NABM_ACE(an_deb_pmsi=, an_fin_pmsi=, an_crea_t=, an_fin_t=, tbl_codes=, table=, domaine=, var_NABM=, var_qte=,
tbl_out=, tbl_patients=);

                %do i = %sysfunc(max(&an_deb_pmsi., &an_crea_t.)) %to %sysfunc(min(&an_fin_pmsi., &an_fin_t.));

                        %if &i. < 10 %then %let an = 0&i.;
                        %else %let an = &i.;

                        * On définit les variables de jointure;
                        %let var_join1 = ETA_NUM;
                        %let var_join2 = SEQ_NUM;



                        * On joint la table NABM avec la table des ACE en filtrant sur les codes NABM sélectionnés;
                        %put Récupération des séjours du PMSI-&domaine. (table &table.) contenant des actes NABM pré-définis - Année
&an. ...;

                        proc sql;

                                %connectora;

                                        CREATE TABLE ext_NABM_PMSI_&domaine._&table._&an. AS SELECT * FROM
CONNECTION TO ORACLE (
                                                                SELECT DISTINCT
                                                                        c.&var_join1.,
```

```
                                                                c.&var_join2.,
                                                                pop_BEN_IDT_ANO,
                                                                c.EXE_SOI_DTD,
                                                                c.EXE_SOI_DTF,
                                                                20&an. AS annee,
                                                                &var_qte. AS nb_deliv,
                                                                cod.*
                                                        FROM &tbl_patients. pop
                                                                INNER JOIN T_&domaine.&an.CSTC c
                                                                        ON      pop.BEN_NIR_PSA = c.NIR_ANO_17
                                                                INNER JOIN T_&domaine.&an.&table. nabm
                                                                        ON      c.&var_join1. = nabm.&var_join1.
                                                                        AND     c.&var_join2. = nabm.&var_join2.
                                                                INNER JOIN &tbl_codes. cod
                                                                        ON TRIM(nabm.&var_NABM.) =
TRIM(cod.code_NABM)
                                                        WHERE c.NIR_ANO_17 NOT IN ('&cle_inc1.', '&cle_inc2.') AND
&var_qte. > 0
                                                                AND c.EXE_SOI_DTD IS NOT NULL AND c.EXE_SOI_DTF IS
NOT NULL
                                                                %if &domaine. = MCO %then
                                                                        %do;
                                                                                AND NIR_RET = '0' AND NAI_RET = '0'
AND SEX_RET = '0' AND IAS_RET = '0' AND ENT_DAT_RET = '0'
                                                                                AND c.ETA_NUM NOT IN
('130780521', '130783236', '130783293', '130784234', '130804297', '600100101',
                                                                                        '750041543', '750100018',
'750100042', '750100075', '750100083', '750100091', '750100109',
                                                                                        '750100125', '750100166',
'750100208', '750100216', '750100232', '750100273', '750100299',
                                                                                        '750801441', '750803447',
'750803454', '910100015', '910100023', '920100013', '920100021',
                                                                                        '920100039', '920100047',
'920100054', '920100062', '930100011', '930100037', '930100045',
                                                                                        '940100027', '940100035',
'940100043', '940100050', '940100068', '950100016', '690783154',
                                                                                        '690784137', '690784152',
'690784178', '690787478', '830100558')
                                                                        %end;
                                                        );

                                DISCONNECT FROM ORACLE;

                        quit;

                        data ext_NABM_PMSI_&domaine._&table._&an.;
                                set ext_NABM_PMSI_&domaine._&table._&an.;
                                length date_debut date_fin 4.;
                                domaine = "&domaine.";
                                table = "&table.";
                                type = "PMSI ACE";
                                date_debut = datepart(EXE_SOI_DTD);
                                date_fin = datepart(EXE_SOI_DTF);
                                format date_debut date_fin ddmmyy10.;
                                drop EXE_SOI_DTD EXE_SOI_DTF;
                                if (year(date_fin) = 20&an. AND 0 <= yrdif(date_debut, date_fin, 'act/act') <= 1) then output;
                        run;

                        %arret_erreur;

                %end;

%mend NABM_ACE;

*
                ************************************************************************************************
**********************************          ;
*               Macro finale pour appeler les précédentes, pour chaque table du PMSI

                ;
```

```
*
	*****************************************************************************************************
**********************************	;

%macro extract_NABM_PMSI(annee_deb=, annee_fin=, HAD = 1, MCO = 1, RIP = 1, SSR = 1, tbl_out=, tbl_codes=, tbl_patients=);

	%let an_deb_pmsi = %sysevalf(&annee_deb. - 2000);
	%let an_fin_pmsi = %sysevalf(&annee_fin. - 2000);

	* Pour le PMSI-HAD;
	%if &HAD. = 1 %then
		%do;

			%let domaine = HAD;

			* Dans les tables FL;
			%NABM_sejours(
				an_deb_pmsi = &an_deb_pmsi.,
				an_fin_pmsi = &an_fin_pmsi.,
				an_crea_t = 12,
				an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
				tbl_codes = &tbl_codes.,
				table = FL,
				domaine = &domaine.,
				var_NABM = NABM_COD,
				var_qte = COALESCE(ACT_NBR, 0),
				tbl_out = &tbl_out.,
				tbl_patients = &tbl_patients.
				);

		%end;
		* Fin du PMSI-HAD;

	* Pour le PMSI-MCO;
	%if &MCO. = 1 %then
		%do;

			%let domaine = MCO;

			* Dans les tables FL;
			%NABM_sejours(
				an_deb_pmsi = &an_deb_pmsi.,
				an_fin_pmsi = &an_fin_pmsi.,
				an_crea_t = 12,
				an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
				tbl_codes = &tbl_codes.,
				table = FL,
				domaine = &domaine.,
				var_NABM = NABM_COD,
				var_qte = COALESCE(ACT_NBR, 0),
				tbl_out = &tbl_out.,
				tbl_patients = &tbl_patients.
				);

			* Dans les tables FLSTC;
			%NABM_ACE(
				an_deb_pmsi = &an_deb_pmsi.,
				an_fin_pmsi = &an_fin_pmsi.,
				an_crea_t = 12,
				an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
				tbl_codes = &tbl_codes.,
				table = FLSTC,
				domaine = &domaine.,
				var_NABM = NABM_COD,
				var_qte = COALESCE(ACT_NBR, 0),
				tbl_out = &tbl_out.,
				tbl_patients = &tbl_patients.
				);

		%end;
		* Fin du PMSI-MCO;
```

```
* Pour le PMSI-RIP;
%if &RIP. = 1 %then
	%do;

		%let domaine = RIP;

		* Dans les tables FL;
		%NABM_sejours(
			an_deb_pmsi = &an_deb_pmsi.,
			an_fin_pmsi = &an_fin_pmsi.,
			an_crea_t = 12,
			an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
			tbl_codes = &tbl_codes.,
			table = FL,
			domaine = &domaine.,
			var_NABM = NABM_COD,
			var_qte = COALESCE(ACT_NBR, 0),
			tbl_out = &tbl_out.,
			tbl_patients = &tbl_patients.
			);

	%end;
	* Fin du PMSI-RIP;

* Pour le PMSI-SSR;
%if &SSR. = 1 %then
	%do;

		%let domaine = SSR;

		* Dans les tables FL;
		%NABM_sejours(
			an_deb_pmsi = &an_deb_pmsi.,
			an_fin_pmsi = &an_fin_pmsi.,
			an_crea_t = 12,
			an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
			tbl_codes = &tbl_codes.,
			table = FL,
			domaine = &domaine.,
			var_NABM = NABM_COD,
			var_qte = COALESCE(ACT_NBR, 0),
			tbl_out = &tbl_out.,
			tbl_patients = &tbl_patients.
			);

		* Dans les tables FLSTC;
		%NABM_ACE(
			an_deb_pmsi = &an_deb_pmsi.,
			an_fin_pmsi = &an_fin_pmsi.,
			an_crea_t = 13,
			an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
			tbl_codes = &tbl_codes.,
			table = FLSTC,
			domaine = &domaine.,
			var_NABM = NABM_COD,
			var_qte = COALESCE(ACT_NBR, 0),
			tbl_out = &tbl_out.,
			tbl_patients = &tbl_patients.
			);

	%end;
	* Fin du PMSI-SSR;

* Empilage de la table temporaire à la table Résultat;
data &tbl_out.;
	format table $10.;
	set ext_NABM_PMSI_:;
run;

%arret_erreur;
```

```
	* Suppression de la table temporaire;
	proc datasets library = work memtype = data nolist;
		delete ext_NABM_PMSI_:;
	run;

%mend extract_NABM_PMSI;
```

3.1.20 Macro_Reperage_des_soins_Pneumologue_PMSI_SSR.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*		Macro pour repérer les contacts avec un pneumologue lors d'une hospitalisation en SSR : extract_SSR_pneumo
                                                                                                                    */
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*		Arguments en entrée :
                                                                                                                    */
/*
                                                                                                                    */
/*			- annee_deb = la première année de soins (ie. année des tables PMSI) pour laquelle on souhaite
repérer les codes                                                                                                   */
/*
                                                                                                                    */
/*			- annee_fin = la dernière année de soins (ie. année des tables PMSI) pour laquelle on souhaite repérer
les codes                                                                                                           */
/*
                                                                                                                    */
/*			- tbl_out = le nom de la table en sortie (peut contenir un nom de librairie)
                                                                                                                    */
/*
                                                                                                                    */
/*			- tbl_patients = le nom de la table contenant la correspondance BEN_IDT_ANO <-> BEN_NR_PSA (au
format Oracle)                                                                                                      */
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*		Tables en sortie : &tbl_out. (renseignée en paramètres de la macro)

                                                                                                                    */
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/

*
	*****************************************************************************************************************
**********************************	;
*		Macro pour repérer les contacts avec un pneumologue lors d'une hospitalisation en SSR
                                                                                                                    ;
*
	*****************************************************************************************************************
**********************************	;
```

```
%macro extract_SSR_pneumo(annee_deb=, annee_fin=, an_crea_t = 14 , an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
tbl_out=,
	tbl_patients=);

	%let an_deb_pmsi = %sysevalf(&annee_deb. - 2000);
	%let an_fin_pmsi = %sysevalf(&annee_fin. - 2000);

	%do i = %sysfunc(max(&an_deb_pmsi., &an_crea_t. )) %to %sysfunc(min(&an_fin_pmsi., &an_fin_t.));

		%if &i. < 10 %then %let an = 0&i.;
		%else %let an = &i.;

		* On définit les variables de jointure pour chaque domaine du PMSI;
		%let var_join1 = ETA_NUM;
		%let var_join2 = RHA_NUM;


		proc sql;

			%connectora;

				CREATE TABLE ext_pneumo_SSR_&an. AS
					SELECT *
					FROM CONNECTION TO ORACLE (
						SELECT DISTINCT
							c.&var_join1.,
							c.&var_join2.,
							pop.BEN_IDT_ANO,
							c.EXE_SOI_DTD,
							c.EXE_SOI_DTF,
							SUBSTR(b.AUT_TYP_UM, 1, 2) AS type_UM,
							20&an. AS annee,
							'SSR' AS domaine
						FROM &tbl_patients. pop
							INNER JOIN T_SSR&an.C c
								ON	pop.BEN_NIR_PSA = c.NIR_ANO_17
							INNER JOIN T_SSR&an.B b
								ON	c.&var_join1. = b.&var_join1.
								AND	c.&var_join2. = b.&var_join2.
							INNER JOIN T_SSR&an.E e
								ON	c.&var_join1. = e.ETA_NUM
						WHERE c.NIR_ANO_17 NOT IN ('&cle_inc1.', '&cle_inc2.') AND
SUBSTR(b.AUT_TYP_UM, 1, 2) = '54'
							AND SUBSTR(TRIM(b.GRG_GME), 1, 2) = '04'
							AND c.NIR_RET = '0' AND c.NAI_RET = '0' AND c.SEX_RET =
'0' AND c.SEJ_RET = '0' AND c.FHO_RET = '0'
							AND c.PMS_RET = '0' AND c.DAT_RET = '0' AND
c.COH_NAI_RET = '0' AND c.COH_SEX_RET = '0'
						);

			DISCONNECT FROM ORACLE;

		quit;

		%arret_erreur;

	%end;

	* Empilage de la table temporaire à la table Résultat;
	data &tbl_out.;
		set ext_pneumo_SSR_:;
		length date_debut date_fin 4.;
		type = "PMSI séjours";
		date_debut = datepart(EXE_SOI_DTD);
		date_fin = datepart(EXE_SOI_DTF);
		format date_debut date_fin ddmmyy10.;
		drop EXE_SOI_DTD EXE_SOI_DTF;
	run;

	%arret_erreur;
```

```
	* Suppression de la table temporaire;
	proc datasets library = work memtype = data nolist;
		delete ext_pneumo_SSR_:;
	run;

%mend extract_SSR_pneumo;
```

3.1.21 Macro_Reperage_des_soins_PRS_NAT_REF_DCIR.sas

```
/*
	*************************************************************************************************************
************************************	*/
/*
                                                                                                                */
/*		Macro pour repérer les remboursement des natures de prestation souhaitées : extract_PRS_NAT_REF
                                                                                                                */
/*
                                                                                                                */
/*
	*************************************************************************************************************
************************************	*/
/*
                                                                                                                */
/*		Arguments en entrée :
                                                                                                                */
/*
                                                                                                                */
/*			- annee_deb = la première année de soins (ie. filtre sur EXE_SOI_DTD) pour laquelle on souhaite
repérer les codes                                                */
/*
                                                                                                                */
/*			- annee_fin = la dernière année de soins (ie. filtre sur EXE_SOI_DTD) pour laquelle on souhaite repérer
les codes                                                */
/*
                                                                                                                */
/*			- sel_spe = le filtre supplémentaire sur les spécialistes
                                                                                                                */
/*
                                                                                                                */
/*			- tbl_out = le nom de la table en sortie (peut contenir un nom de librairie)
                                                                                                                */
/*
                                                                                                                */
/*			- tbl_codes = la table contenant les natures de prestation à repérer (au format Oracle)
                                                                                                                */
/*
                                                                                                                */
/*			- tbl_patients = le nom de la table contenant la correspondance BEN_IDT_ANO <-> BEN_NR_PSA (au
format Oracle)                                                */
/*
                                                                                                                */
/*			- regul (flag 0/1) = si 1, on applique le programme de régularisations
                                                                                                                */
/*
                                                                                                                */
/*
	*************************************************************************************************************
************************************	*/
/*
                                                                                                                */
```

```
/*                  Tables en sortie : &tbl_out. (renseignée en paramètres de la macro)

                    */
/*                  avec sélection des codes disponibles dans le référentiel &tbl_codes., filtrée sur les soins de ville uniquement, les
NIR  normaux +      */
/*                  suppression des lignes pour information

                                                                    */
/*

                                                                                                                                    */
/*
        ********************************************************************************************************************************
**********************************          */

%macro extract_PRS_NAT_REF(annee_deb=, annee_fin=, sel_spe=, tbl_out=, tbl_codes=, tbl_patients=, regul = 1);

        %let annee_fin_flx = %sysevalf(&annee_fin. + 1);

        * On boucle sur les années de repérage (en flux);
        %do Annee = &annee_deb. %to &annee_fin_flx.;

                * On boucle sur les 12 mois de l année;
                %do i = 1 %to 12;

                        %if &i. < 10 %then %let Mois = 0&i.;
                        %else %let Mois = &i.;

                        %put Données de &Mois./&Annee.;

                        proc sql;

                                %connectora;

                                        CREATE TABLE tmp_extract_PRS_NAT_REF_&annee._&mois. AS SELECT * FROM
CONNECTION TO ORACLE (
                                                SELECT
                                                        pop.BEN_IDT_ANO,
                                                        prs.EXE_SOI_DTD,
                                                        prs.EXE_SOI_DTF,
                                                        prs.BSE_PRS_NAT,
                                                        prs.PRS_ACT_QTE,
                                                        prs.PSE_SPE_COD,
                                                        prs.PSE_ACT_NAT,
                                                        prs.BSE_REM_MNT,
                                                        prs.PRS_ACT_CFT,
                                                        ete.ETB_EXE_FIN,
                                                        ete.ETE_MCO_DDP,
                                                        ete.MDT_COD,
                                                        ete.ETE_IND_TAA,
                                                        cod.*
                                                FROM &tbl_patients. pop
                                                        INNER JOIN ER_PRS_F prs
                                                                ON      pop.BEN_NIR_PSA =
prs.BEN_NIR_PSA
                                                        INNER JOIN &tbl_codes. cod
                                                                ON prs.BSE_PRS_NAT =
cod.code_presta
                                                        LEFT JOIN ER_ETE_F ete
                                                                ON      ete.FLX_DIS_DTD =
prs.FLX_DIS_DTD
                                                                AND     ete.FLX_TRT_DTD =
prs.FLX_TRT_DTD
                                                                AND     ete.FLX_EMT_TYP =
prs.FLX_EMT_TYP
                                                                AND     ete.FLX_EMT_NUM =
prs.FLX_EMT_NUM
                                                                AND     ete.FLX_EMT_ORD =
prs.FLX_EMT_ORD
                                                                AND     ete.ORG_CLE_NUM =
prs.ORG_CLE_NUM
```

```
                                                                                AND         ete.DCT_ORD_NUM =
prs.DCT_ORD_NUM
                                                                                AND         ete.PRS_ORD_NUM =
prs.PRS_ORD_NUM
                                                                                AND         ete.REM_TYP_AFF =
prs.REM_TYP_AFF
                                                                        WHERE  prs.BEN_CDI_NIR = '00'  AND prs.CPL_MAJ_TOP IN
(0, 1)
                                                                                AND (ete.ETE_IND_TAA NOT IN (1, 2) OR
ete.ETE_IND_TAA IS NULL)
                                                                                AND prs.DPN_QLF NOT IN (71, 72) AND
NOT(prs.DPN_QLF = 0 AND prs.PRS_DPN_QLP = 71)
                                                                                AND prs.FLX_DIS_DTD =
TO_DATE(%str(%'&Annee.&Mois.01%'), 'YYYYMMDD')
                                                                                AND prs.EXE_SOI_DTD BETWEEN
TO_DATE(%str(%'&Annee_deb.0101%'), 'YYYYMMDD') AND TO_DATE(%str(%'&Annee_fin.1231%'), 'YYYYMMDD')
                                                                                &sel_spe.
                                                                        );

                                        disconnect from oracle;

                                quit;

                                %arret_erreur;

                                * On empile toutes les tables mensuelles;
                                %if %sysfunc(exist(&tbl_out.)) = 1 %then
                                        %do;

                                                proc append base = &tbl_out. data =
tmp_extract_PRS_NAT_REF_&annee._&mois.;
                                                run;

                                                %arret_erreur;

                                                proc delete data = tmp_extract_PRS_NAT_REF_&annee._&mois.;
                                                run; quit;

                                        %end;

                                %if %sysfunc(exist(&tbl_out.)) = 0 %then
                                        %do;

                                                data &tbl_out.;
                                                        set tmp_extract_PRS_NAT_REF_&annee._&mois.;
                                                run;

                                                %arret_erreur;

                                                proc delete data = tmp_extract_PRS_NAT_REF_&annee._&mois.;
                                                run; quit;

                                        %end;

                        %end;
                        * Fin de la boucle sur les 12 mois;

                %end;
                * Fin de la boucle par année;

                %if &tbl_codes. = presta_pneumo or &tbl_codes. = codes_presta_RR %then
                        %do;

                                data &tbl_out.;
                                        set &tbl_out. (where = (ETE_IND_TAA not in (1, 2) and MDT_COD in (., 0, 7)));
                                run;

                %end;

                %else
                        %do;
```

```
                        data &tbl_out.;
                                set &tbl_out. (where = (ETE_IND_TAA not in (1, 2)));
                        run;

        %end;

        * On applique le programme de régularisations pour céer la table finale;
        proc sql undo_policy = none;

                CREATE TABLE &tbl_out. AS
                        SELECT
                                BEN_IDT_ANO,
                                datepart(EXE_SOI_DTD) AS date_debut length = 4 format ddmmyy10.,
                                CASE    WHEN datepart(EXE_SOI_DTF) IS NULL THEN datepart(EXE_SOI_DTD)
                                                ELSE datepart(EXE_SOI_DTF)
                                                END AS date_fin length = 4 format ddmmyy10.,
                                "DCIR" AS type,
                                BSE_PRS_NAT,
                                PSE_SPE_COD,
                                PSE_ACT_NAT,
                                PRS_ACT_CFT,
                                reperage,
                                ETB_EXE_FIN,
                                ETE_MCO_DDP,
                                SUM(BSE_REM_MNT) AS montant,
                                SUM(PRS_ACT_QTE) AS quantite
                        FROM &tbl_out.
                        GROUP BY 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11;

        quit;

        %if &regul. = 1 %then
                %do;

                        proc sql;

                                DELETE FROM &tbl_out. WHERE montant <= 0 OR quantite < 0;

                        quit;

                %end;

        %arret_erreur;

%mend extract_PRS_NAT_REF;
```

3.1.22 Macro_Reperage_des_soins_Tous_sejours_PMSI.sas

```
/*
        ************************************************************************************************************
**********************************        */
/*
                                                                                                                    */
/*                      Macro pour repérer les séjours autour d'une date, dans le PMSI : extract_sej_PMSI
                                                                                                                    */
/*
                                                                                                                    */
/*
        ************************************************************************************************************
**********************************        */
/*
                                                                                                                    */
/*                      Arguments en entrée :
                                                                                                                    */
/*
                                                                                                                    */
/*                              - annee_deb = la première année de soins (ie. année des tables PMSI) pour laquelle on souhaite
repérer les codes                                                       */
/*
                                                                                                                    */
/*                              - annee_fin = la dernière année de soins (ie. année des tables PMSI) pour laquelle on souhaite repérer
les codes                                                       */
/*
                                                                                                                    */
/*                              - tbl_patients = le nom de la table en entrée, qui doit contenir la correspondance BEN_IDT_ANO <->
BEN_NIR_PSA                                                             */
/*
                                                                                                                    */
/*                              - tbl_out = le nom de la table en sortie (peut contenir un nom de librairie)
                                                                                                                    */
/*
                                                                                                                    */
/*                              - HAD = flag (0/1) : si 1, on recherche les séjours du PMSI-HAD (Par défaut : 1)
                                                                                                                    */
/*
                                                                                                                    */
/*                              - MCO = flag (0/1) : si 1, on recherche les séjours du PMSI-MCO (Par défaut : 1)
                                                                                                                    */
/*
                                                                                                                    */
/*                              - RIP = flag (0/1) : si 1, on recherche les séjours du PMSI-RIP (Par défaut : 1)
                                                                                                                */
/*
                                                                                                                    */
/*                              - SSR = flag (0/1) : si 1, on recherche les séjours du PMSI-SSR (Par défaut : 1)
                                                                                                                */
/*
                                                                                                                    */
/*
        ************************************************************************************************************
**********************************        */
```

```
/*
                                                                                                                                                    */
/*                      Tables en sortie : &tbl_out. (renseignée en paramètres de la macro)

                        */
/*                      hors séjours en erreur

                                                                                                                                    */
/*
                                                                                                                                                    */
/*
        ***************************************************************************************************************************
***********************************          */

*
        ***************************************************************************************************************************
***********************************          ;
*       Macro pour repérer tous les séjours du PMSI

                                                                ;
*
        ***************************************************************************************************************************
***********************************          ;

%macro sejours_PMSI(an_deb_pmsi=, an_fin_pmsi=, an_crea_t=, an_fin_t=, domaine=, tbl_out=, tbl_patients=);

        %do i = %sysfunc(max(&an_deb_pmsi., &an_crea_t. )) %to %sysfunc(min(&an_fin_pmsi., &an_fin_t.));

                %if &i. < 10 %then %let an = 0&i.;
                %else %let an = &i.;

                * On définit les variables de jointure pour chaque domaine du PMSI;
                %if &domaine. = HAD %then
                        %do;
                                %let var_join1 = ETA_NUM_EPMSI;
                                %let var_join2 = RHAD_NUM;
                        %end;
                %if &domaine. = MCO %then
                        %do;
                                %let var_join1 = ETA_NUM;
                                %let var_join2 = RSA_NUM;
                        %end;
                %if &domaine. = RIP %then
                        %do;
                                %let var_join1 = ETA_NUM_EPMSI;
                                %let var_join2 = RIP_NUM;
                        %end;
                %if &domaine. = SSR %then
                        %do;
                                %let var_join1 = ETA_NUM;
                                %let var_join2 = RHA_NUM;
                        %end;


                proc sql;

                        %connectora;

                                CREATE TABLE sejours_&domaine._&an. AS
                                        SELECT *
                                        FROM CONNECTION TO ORACLE (
                                                SELECT DISTINCT
                                                        pop.BEN_IDT_ANO,
                                                        c.&var_join1.,
                                                        c.&var_join2.,
                                                        c.EXE_SOI_DTD,
                                                        c.EXE_SOI_DTF,
                                                        %if &domaine. = MCO %then %do;
                                                                b.GRG_GHM,
```

```
                                                %end;
                                                %if &domaine. = MCO or &domaine. = HAD %then %do;
                                                        b.SEJ_NBJ AS duree_sejour,
                                                %end;
                                                %if &domaine. = RIP %then %do;
                                                        s.SEJ_DUREE AS duree_sejour,
                                                %end;
                                                %if &domaine. = SSR %then %do;
                                                        s.SEJ_NBJ AS duree_sejour,
                                                        SUBSTR(b.AUT_TYP_UM, 1, 2) AS type_UM,
                                                %end;
                                                20&an. AS annee
                                        FROM &tbl_patients. pop
                                                INNER JOIN T_&domaine.&an.C c
                                                        ON      pop.BEN_NIR_PSA = c.NIR_ANO_17
                                                %if &domaine. = MCO or &domaine. = HAD or &domaine. =
SSR %then
                                                        %do;
                                                                INNER JOIN T_&domaine.&an.B b
                                                                        ON      c.&var_join1. =
b.&var_join1.
                                                                        AND     c.&var_join2. =
b.&var_join2.
                                                        %end;
                                                %if &domaine. = RIP or &domaine. = SSR %then
                                                        %do;
                                                                LEFT JOIN T_&domaine.&an.S s
                                                                        ON      c.&var_join1. =
s.&var_join1.
                                                                        AND     c.&var_join2. =
s.&var_join2.
                                                        %end;

                                        WHERE c.NIR_ANO_17 NOT IN ('&cle_inc1.', '&cle_inc2.')
                                                %if &domaine. = HAD %then
                                                        %do;
                                                                AND NIR_RET = '0' AND NAI_RET = '0'
AND SEX_RET = '0' AND SEJ_RET = '0' AND FHO_RET = '0'
                                                                AND PMS_RET = '0' AND DAT_RET =
'0' %if &an. > 12 %then %do; AND COH_NAI_RET = '0'
                                                                AND COH_SEX_RET = '0' %end;
                                                        %end;
                                                %if &domaine. = MCO %then
                                                        %do;
                                                                AND NIR_RET = '0' AND NAI_RET = '0'
AND SEX_RET = '0' AND SEJ_RET = '0' AND FHO_RET = '0'
                                                                AND PMS_RET = '0' AND DAT_RET =
'0' %if &an. > 12 %then %do; AND COH_NAI_RET = '0'
                                                                AND COH_SEX_RET = '0' %end;
                                                                AND c.ETA_NUM NOT IN
('130780521', '130783236', '130783293', '130784234', '130804297', '600100101',
                                                                        '750041543', '750100018',
'750100042', '750100075', '750100083', '750100091', '750100109',
                                                                        '750100125', '750100166',
'750100208', '750100216', '750100232', '750100273', '750100299',
                                                                        '750801441', '750803447',
'750803454', '910100015', '910100023', '920100013', '920100021',
                                                                        '920100039', '920100047',
'920100054', '920100062', '930100011', '930100037', '930100045',
                                                                        '940100027', '940100035',
'940100043', '940100050', '940100068', '950100016', '690783154',
                                                                        '690784137', '690784152',
'690784178', '690787478', '830100558')
                                                                AND GRG_GHM NOT IN ('90Z00Z')
AND GRG_RET NOT IN ('024') AND GRG_GHM NOT IN ('90H01Z', '90Z00Z', '90Z01Z',
                                                                        '90Z02Z', '90Z03Z')
                                                                AND ((SEJ_TYP = 'A' OR SEJ_TYP IS
NULL) OR (SEJ_TYP = 'B' AND GRG_GHM NOT IN ('28Z14Z','28Z15Z','28Z16Z')))
                                                        %end;
                                                %if &domaine. = RIP %then
                                                        %do;
```

```
                                                                                AND NIR_RET = '0' AND NAI_RET = '0'
AND SEX_RET = '0' AND SEJ_RET = '0' AND FHO_RET = '0'
                                                                                AND PMS_RET = '0' AND DAT_RET =
'0' %if &an. > 12 %then %do; AND COH_NAI_RET = '0'
                                                                                AND COH_SEX_RET = '0' %end;
                                                                        %end;
                                                                %if &domaine. = SSR %then
                                                                        %do;
                                                                                AND NIR_RET = '0' AND NAI_RET = '0'
AND SEX_RET = '0' AND SEJ_RET = '0' AND FHO_RET = '0'
                                                                                AND PMS_RET = '0' AND DAT_RET =
'0' %if &an. > 12 %then %do; AND COH_NAI_RET = '0'
                                                                                AND COH_SEX_RET = '0' %end;
                                                                        %end;
                                                        );

                                DISCONNECT FROM ORACLE;

                        quit;

                        data sejours_&domaine._&an.;
                                set sejours_&domaine._&an.;
                                length date_debut date_fin 4.;
                                domaine = "&domaine.";
                                type = "PMSI séjours";
                                date_debut = datepart(EXE_SOI_DTD);
                                date_fin = datepart(EXE_SOI_DTF);
                                format date_debut date_fin ddmmyy10.;
                                drop EXE_SOI_DTD EXE_SOI_DTF;
                                id_sejour = &var_join1.||"_"||&var_join2.||"_"||put(annee, 4.);
                        run;

                        %arret_erreur;

                %end;

%mend sejours_PMSI;

*
                ************************************************************************************************************
************************************  ;
*               Macro finale pour appeler les précédentes, pour chaque table du PMSI

                ;
*
                ************************************************************************************************************
************************************  ;

%macro extract_sej_PMSI(annee_deb=, annee_fin=, HAD = 1, MCO = 1, SSR = 1, RIP = 1, tbl_out=, tbl_patients=);

                %let an_deb_pmsi = %sysevalf(&annee_deb. - 2000);
                %let an_fin_pmsi = %sysevalf(&annee_fin. - 2000);

                * Pour le PMSI-HAD;
                %if &HAD. = 1 %then
                        %do;

                                %let domaine = HAD;

                                %sejours_PMSI(
                                        an_deb_pmsi = &an_deb_pmsi.,
                                        an_fin_pmsi = &an_fin_pmsi.,
                                        an_crea_t = 06,
                                        an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
                                        domaine = &domaine.,
                                        tbl_out = &tbl_out.,
                                        tbl_patients = &tbl_patients.
                                        );

                        %end;
                        * Fin du PMSI-HAD;
```

```
* Pour le PMSI-MCO;
%if &MCO. = 1 %then
	%do;

		%let domaine = MCO;

		%sejours_PMSI(
			an_deb_pmsi = &an_deb_pmsi.,
			an_fin_pmsi = &an_fin_pmsi.,
			an_crea_t = 06,
			an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
			domaine = &domaine.,
			tbl_out = &tbl_out.,
			tbl_patients = &tbl_patients.
			);

		%end;
	* Fin du PMSI-MCO;

* Pour le PMSI-RIP;
%if &RIP. = 1 %then
	%do;

		%let domaine = RIP;

		%sejours_PMSI(
			an_deb_pmsi = &an_deb_pmsi.,
			an_fin_pmsi = &an_fin_pmsi.,
			an_crea_t = 06,
			an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
			domaine = &domaine.,
			tbl_out = &tbl_out.,
			tbl_patients = &tbl_patients.
			);

		%end;
	* Fin du PMSI-MCO;

* Pour le PMSI-SSR;
%if &SSR. = 1 %then
	%do;

		%let domaine = SSR;

		%sejours_PMSI(
			an_deb_pmsi = &an_deb_pmsi.,
			an_fin_pmsi = &an_fin_pmsi.,
			an_crea_t = 06,
			an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
			domaine = &domaine.,
			tbl_out = &tbl_out.,
			tbl_patients = &tbl_patients.
			);

	%end;
	* Fin du PMSI-SSR;

* Empilage de la table temporaire à la table Résultat;
data &tbl_out.;
	length id_sejour $50.;
	set sejours_:;
run;

%arret_erreur;

* Suppression de la table temporaire;
proc datasets library = work memtype = data nolist;
	delete sejours_:;
run;
```

```
%mend extract_sej_PMSI;
```

3.1.23 Macro_Reperage_des_soins_UCD_DCIR.sas

```
/*
        ************************************************************************************************************
************************************  */
/*

                                                                                                                    */
/*                  Macro pour repérer les remboursement des codes UCD souhaités, dans le DCIR : extract_UCD_DCIR
                                                                                                                    */
/*

                                                                                                                    */
/*
        ************************************************************************************************************
************************************  */
/*

                                                                                                                    */
/*                  Arguments en entrée :

                                                                                                                    */
/*

                                                                                                                    */
/*                          - annee_deb = la première année de soins (ie. filtre sur EXE_SOI_DTD) pour laquelle on souhaite
repérer les codes                                                   */
/*

                                                                                                                    */
/*                          - annee_fin = la dernière année de soins (ie. filtre sur EXE_SOI_DTD) pour laquelle on souhaite repérer
les codes                                                   */
/*

                                                                                                                    */
/*                          - tbl_out = le nom de la table en sortie (peut contenir un nom de librairie)
                                                                                                                    */
/*

                                                                                                                    */
/*                          - tbl_codes = la table contenant les classes ATC à repérer (au format Oracle)
                                                                                                                    */
/*

                                                                                                                    */
/*                          - tbl_patients = le nom de la table contenant la correspondance BEN_IDT_ANO <-> BEN_NR_PSA (au
format Oracle)                                                      */
/*

                                                                                                                    */
/*
        ************************************************************************************************************
************************************  */
/*

                                                                                                                    */
/*                  Tables en sortie : &tbl_out. (renseignée en paramètres de la macro)

                    */
/*                  avec sélection des codes disponibles dans le référentiel &tbl_codes., filtrée sur les soins de ville uniquement, les
NIR  normaux ,      */
/*                  les actes de base, les médicaments rétrocédés, + suppression des lignes pour information
                                                                                                                    */
/*

                                                                                                                    */
/*
        ************************************************************************************************************
************************************  */
```

```
%macro extract_UCD_DCIR(annee_deb=, annee_fin=, tbl_out=, tbl_codes=, tbl_patients=);

	%let annee_fin_flx = %sysevalf(&annee_fin. + 1);

	* On boucle sur les années de repérage (en flux);
	%do Annee = &annee_deb. %to &annee_fin_flx.;

		* On boucle sur les 12 mois de l année;
		%do i = 1 %to 12;

			%if &i. < 10 %then %let Mois = 0&i.;
			%else %let Mois = &i.;

			%put Données de &Mois./&Annee.;

			proc sql;

				%connectora;

					CREATE TABLE tmp_extract_UCD_DCIR_&annee._&mois. AS
						SELECT * FROM CONNECTION TO ORACLE (
							SELECT
								pop.BEN_IDT_ANO,
								prs.EXE_SOI_DTD,
								prs.EXE_SOI_DTF,
								prs.PRS_NAT_REF,
								prs.BSE_REM_MNT,
								ref.PHA_ATC_C07,
								ref.PHA_CND_TOP,
								ucd.UCD_UCD_COD,
								ucd.UCD_DLV_NBR,
								ete.ETB_EXE_FIN,
								ete.ETE_MCO_DDP,
								cod.*
							FROM &tbl_patients. pop
								INNER JOIN ER_PRS_F prs
									ON pop.BEN_NIR_PSA =
prs.BEN_NIR_PSA
								INNER JOIN ER_UCD_F ucd
									ON ucd.FLX_DIS_DTD =
prs.FLX_DIS_DTD
									AND ucd.FLX_TRT_DTD =
prs.FLX_TRT_DTD
									AND ucd.FLX_EMT_TYP =
prs.FLX_EMT_TYP
									AND ucd.FLX_EMT_NUM =
prs.FLX_EMT_NUM
									AND ucd.FLX_EMT_ORD =
prs.FLX_EMT_ORD
									AND ucd.ORG_CLE_NUM =
prs.ORG_CLE_NUM
									AND ucd.DCT_ORD_NUM =
prs.DCT_ORD_NUM
									AND ucd.PRS_ORD_NUM =
prs.PRS_ORD_NUM
									AND ucd.REM_TYP_AFF =
prs.REM_TYP_AFF
								INNER JOIN IR_PHA_R ref
									ON ref.PHA_CIP_UCD =
SUBSTR(ucd.UCD_UCD_COD, 7, 7)
								INNER JOIN &tbl_codes. cod
									ON ref.PHA_ATC_C07 =
cod.classe_ATC
								LEFT JOIN ER_ETE_F ete
									ON ete.FLX_DIS_DTD =
prs.FLX_DIS_DTD
									AND ete.FLX_TRT_DTD =
prs.FLX_TRT_DTD
									AND ete.FLX_EMT_TYP =
prs.FLX_EMT_TYP
```

```
                                                                                AND         ete.FLX_EMT_NUM =
prs.FLX_EMT_NUM
                                                                                AND         ete.FLX_EMT_ORD =
prs.FLX_EMT_ORD
                                                                                AND         ete.ORG_CLE_NUM =
prs.ORG_CLE_NUM
                                                                                AND         ete.DCT_ORD_NUM =
prs.DCT_ORD_NUM
                                                                                AND         ete.PRS_ORD_NUM =
prs.PRS_ORD_NUM
                                                                                AND         ete.REM_TYP_AFF =
prs.REM_TYP_AFF
                                                                        WHERE  ref.PHA_CIP_UCD NOT IN (0) AND
ucd.UCD_TOP_UCD = 0
                                                                                AND (ete.ETE_IND_TAA NOT IN (1, 2) OR
ete.ETE_IND_TAA IS NULL)
                                                                                AND prs.BEN_CDI_NIR = '00' AND
prs.CPL_MAJ_TOP IN (0, 1)
                                                                                AND prs.DPN_QLF NOT IN (71, 72) AND
NOT(prs.DPN_QLF = 0 AND prs.PRS_DPN_QLP = 71)
                                                                                AND prs.FLX_DIS_DTD =
TO_DATE(%str(%'&Annee.&Mois.01%'), 'YYYYMMDD')
                                                                                AND prs.EXE_SOI_DTD BETWEEN
TO_DATE(%str(%'&Annee_deb.0101%'), 'YYYYMMDD') AND TO_DATE(%str(%'&Annee_fin.1231%'), 'YYYYMMDD')
                                                                        );

                                        disconnect from oracle;

                                quit;

                                %arret_erreur;

                                * On empile toutes les tables mensuelles;
                                %if %sysfunc(exist(&tbl_out.)) = 1 %then
                                        %do;

                                                proc append base = &tbl_out. data = tmp_extract_UCD_DCIR_&annee._&mois.;
                                                run;

                                                %arret_erreur;

                                                proc delete data = tmp_extract_UCD_DCIR_&annee._&mois.;
                                                run; quit;

                                        %end;

                                %if %sysfunc(exist(&tbl_out.)) = 0 %then
                                        %do;

                                                data &tbl_out.;
                                                        set tmp_extract_UCD_DCIR_&annee._&mois.;
                                                run;

                                                %arret_erreur;

                                                proc delete data = tmp_extract_UCD_DCIR_&annee._&mois.;
                                                run; quit;

                                        %end;

                        %end;
                        * Fin de la boucle sur les 12 mois;

                %end;
                * Fin de la boucle par année;

                * On applique le programme de régularisations pour céer la table finale;
                proc sql undo_policy = none;

                        CREATE TABLE &tbl_out. AS
                                SELECT
```

```
                                        BEN_IDT_ANO,
                                        datepart(EXE_SOI_DTD) AS date_debut length = 4 format ddmmyy10.,
                                        CASE        WHEN datepart(EXE_SOI_DTF) IS NULL THEN datepart(EXE_SOI_DTD)
                                                            ELSE datepart(EXE_SOI_DTF)
                                                            END AS date_fin length = 4 format ddmmyy10.,
                                        "DCIR" AS type,
                                        PHA_ATC_C07,
                                        UCD_UCD_COD,
                                        PHA_CND_TOP,
                                        reperage,
                                        ETB_EXE_FIN,
                                        ETE_MCO_DDP,
                                        SUM(BSE_REM_MNT) AS montant,
                                        SUM(UCD_DLV_NBR) AS quantite
                                FROM &tbl_out.
                                GROUP BY 1, 2, 3, 4, 5, 6, 7, 8, 9, 10;

                        DELETE FROM &tbl_out. WHERE montant <= 0 OR quantite < 0;

                quit;

                %arret_erreur;

%mend extract_UCD_DCIR;
```

3.1.24 Macro_Reperage_des_soins_UCD_PMSI.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*

*/
/*			Macro pour repérer les remboursement des codes UCD souhaités, dans le PMSI : extract_UCD_PMSI
*/
/*

*/
/*
	*****************************************************************************************************************
**********************************	*/
/*

*/
/*			Arguments en entrée :

*/
/*

*/
/*				- annee_deb = la première année de soins (ie. année des tables PMSI) pour laquelle on souhaite
repérer les codes	*/
/*

*/
/*				- annee_fin = la dernière année de soins (ie. année des tables PMSI) pour laquelle on souhaite repérer
les codes	*/
/*

*/
/*				- tbl_out = le nom de la table en sortie (peut contenir un nom de librairie)
*/
/*

*/
/*				- tbl_codes = la table contenant les classes ATC à repérer (au format Oracle)
*/
/*

*/
/*				- tbl_patients = le nom de la table contenant la correspondance BEN_IDT_ANO <-> BEN_NR_PSA (au
format Oracle)	*/
/*

*/
/*				- HAD = flag (0/1) : si 1, on recherche les codes UCD dans le PMSI-HAD (Par défaut : 1)
*/
/*

*/
/*				- MCO = flag (0/1) : si 1, on recherche les codes UCD dans le PMSI-MCO (sejours et ACE) (Par défaut : 1)
*/
/*

*/
/*				- RIP = flag (0/1) : si 1, on recherche les codes UCD dans le PMSI-RIP (Par défaut : 1)
*/
/*

*/
/*				- SSR = flag (0/1) : si 1, on recherche les codes UCD dans le PMSI-SSR (sejours et ACE) (Par défaut : 1)
*/
```

```
/*
                                                                                                */
/*
        ****************************************************************************************************
*************************************              */
/*
                                                                                                */
/*                  Tables en sortie : &tbl_out. (renseignée en paramètres de la macro)

                    */
/*                  avec sélection des codes disponibles dans le référentiel &tbl_codes., hors séjours en erreur
                                                                                   */
/*
                                                                                                */
/*
        ****************************************************************************************************
*************************************              */

*
        ****************************************************************************************************
*************************************              ;
*          Macro pour repérer les codes UCD dans le PMSI - séjours

                                             ;
*
        ****************************************************************************************************
*************************************              ;

%macro UCD_sejours(an_deb_pmsi=, an_fin_pmsi=, an_crea_t=, an_fin_t=, tbl_codes=, code_UCD7=, code_UCD13=, table=, domaine=,
var_UCD=, var_qte=,
        tbl_out=, tbl_patients=, var_delai=);

        %do i = %sysfunc(max(&an_deb_pmsi., &an_crea_t. )) %to %sysfunc(min(&an_fin_pmsi., &an_fin_t.));

                %if &i. < 10 %then %let an = 0&i.;
                %else %let an = &i.;

                * On définit les variables de jointure pour chaque domaine du PMSI;
                %if &domaine. = HAD %then
                        %do;
                                %let var_join1 = ETA_NUM_EPMSI;
                                %let var_join2 = RHAD_NUM;
                        %end;
                %if &domaine. = MCO %then
                        %do;
                                %let var_join1 = ETA_NUM;
                                %let var_join2 = RSA_NUM;
                        %end;
                %if &domaine. = RIP %then
                        %do;
                                %let var_join1 = ETA_NUM_EPMSI;
                                %let var_join2 = RIP_NUM;
                        %end;
                %if &domaine. = SSR %then
                        %do;
                                %let var_join1 = ETA_NUM;
                                %let var_join2 = RHA_NUM;
                        %end;

                * On joint la table des médicaments avec la table des séjours en filtrant sur les codes UCD sélectionnés;
                %put Récupération des séjours du PMSI-&domaine. (table &table.) contenant des codes UCD pré-définis - Année
&an. ...;

                proc sql;

                        %connectora;

                                CREATE TABLE temp_T_&domaine.&an.&table. AS
```

```
                                        SELECT *
                                        FROM CONNECTION TO ORACLE (
                                                SELECT
                                                        *
                                                FROM T_&domaine.&an.&table.
                                                WHERE LENGTH(&var_uCD.) > 1 AND &var_uCD. NOT LIKE '%PH%'
                                                );

                        DISCONNECT FROM ORACLE;

                quit;

                data orauser.temp_T_&domaine.&an.&table.;
                        set temp_T_&domaine.&an.&table.;
                        %if &code_UCD7. = 1 %then
                                %do;
                                        join_UCD7 = 1*SUBSTR(&var_UCD., 7, 7);
                                %end;
                        %if &code_UCD13. = 1 %then
                                %do;
                                        join_UCD13 = 1*SUBSTR(&var_UCD., 6, 7);
                                %end;
                run;

                %arret_erreur;

                proc sql;

                        %connectora;

                                CREATE TABLE ext_UCD_PMSI_&domaine._&table._&an. AS
                                        SELECT *
                                        FROM CONNECTION TO ORACLE (
                                                SELECT DISTINCT
                                                        c.&var_join1.,
                                                        c.&var_join2.,
                                                        pop.BEN_IDT_ANO,
                                                        c.EXE_SOI_DTD,
                                                        c.EXE_SOI_DTF,
                                                        ref.PHA_CND_TOP,
                                                        20&an. AS annee,
                                                        &var_qte. AS nb_deliv,
                                                        &var_delai. AS delai,
                                                        cod.*
                                                FROM &tbl_patients. pop
                                                        INNER JOIN T_&domaine.&an.C c
                                                                ON      pop.BEN_NIR_PSA = c.NIR_ANO_17
                                                        %if &domaine. = MCO or &domaine. = HAD or &domaine. =
SSR %then
                                                                %do;
                                                                        INNER JOIN T_&domaine.&an.B b
                                                                                ON      c.&var_join1. =
b.&var_join1.
                                                                                AND     c.&var_join2. =
b.&var_join2.
                                                                %end;
                                                        INNER JOIN T_&domaine.&an.E e
                                                                ON      c.&var_join1. = e.ETA_NUM
                                                        INNER JOIN temp_T_&domaine.&an.&table. ucd
                                                                ON      c.&var_join1. = ucd.&var_join1.
                                                                AND     c.&var_join2. = ucd.&var_join2.
                                                        INNER JOIN IR_PHA_R ref
                                                                %if &code_UCD7. = 1 %then
                                                                        %do;
                                                                                ON ucd.join_UCD7 =
ref.PHA_CIP_UCD
                                                                        %end;
                                                                %if &code_UCD13. = 1 %then
                                                                        %do;
                                                                                ON ucd.join_UCD13 =
ref.PHA_CIP_UCD
```

```
                                                                %end;
                                                        INNER JOIN &tbl_codes. cod
                                                                ON ref.PHA_ATC_C07 =
cod.classe_ATC
                                                WHERE c.NIR_ANO_17 NOT IN ('&cle_inc1.', '&cle_inc2.') AND
&var_qte. > 0
                                        %if &domaine. = HAD %then
                                                %do;
                                                        AND NIR_RET = '0' AND NAI_RET = '0'
AND SEX_RET = '0' AND SEJ_RET = '0' AND FHO_RET = '0'
                                                        AND PMS_RET = '0' AND DAT_RET =
'0' %if &an. > 12 %then %do; AND COH_NAI_RET = '0'
                                                        AND COH_SEX_RET = '0' %end;
                                                %end;
                                        %if &domaine. = MCO %then
                                                %do;
                                                        AND NIR_RET = '0' AND NAI_RET = '0'
AND SEX_RET = '0' AND SEJ_RET = '0' AND FHO_RET = '0'
                                                        AND PMS_RET = '0' AND DAT_RET =
'0' %if &an. > 12 %then %do; AND COH_NAI_RET = '0'
                                                        AND COH_SEX_RET = '0' %end;
                                                        AND c.ETA_NUM NOT IN
('130780521', '130783236', '130783293', '130784234', '130804297', '600100101',
                                                                '750041543', '750100018',
'750100042', '750100075', '750100083', '750100091', '750100109',
                                                                '750100125', '750100166',
'750100208', '750100216', '750100232', '750100273', '750100299',
                                                                '750801441', '750803447',
'750803454', '910100015', '910100023', '920100013', '920100021',
                                                                '920100039', '920100047',
'920100054', '920100062', '930100011', '930100037', '930100045',
                                                                '940100027', '940100035',
'940100043', '940100050', '940100068', '950100016', '690783154',
                                                                '690784137', '690784152',
'690784178', '690787478', '830100558')
                                                        AND GRG_GHM NOT IN ('90Z00Z')
AND GRG_RET NOT IN ('024') AND GRG_GHM NOT IN ('90H01Z', '90Z00Z',
                                                                '90Z01Z', '90Z02Z',
'90Z03Z')
                                                        AND ((SEJ_TYP = 'A' OR SEJ_TYP IS
NULL) OR (SEJ_TYP = 'B' AND GRG_GHM NOT IN ('28Z14Z','28Z15Z','28Z16Z')))
                                                %end;
                                        %if &domaine. = RIP %then
                                                %do;
                                                        AND NIR_RET = '0' AND NAI_RET = '0'
AND SEX_RET = '0' AND SEJ_RET = '0' AND FHO_RET = '0'
                                                        AND PMS_RET = '0' AND DAT_RET =
'0' %if &an. > 12 %then %do; AND COH_NAI_RET = '0'
                                                        AND COH_SEX_RET = '0' %end;
                                                %end;
                                        %if &domaine. = SSR %then
                                                %do;
                                                        AND NIR_RET = '0' AND NAI_RET = '0'
AND SEX_RET = '0' AND SEJ_RET = '0' AND FHO_RET = '0'
                                                        AND PMS_RET = '0' AND DAT_RET =
'0' %if &an. > 12 %then %do; AND COH_NAI_RET = '0'
                                                        AND COH_SEX_RET = '0' %end;
                                                %end;
                                        );

                        DISCONNECT FROM ORACLE;

                quit;

                data ext_UCD_PMSI_&domaine._&table._&an.;
                        set ext_UCD_PMSI_&domaine._&table._&an.;
                        length date_debut date_fin 4.;
                        table = "&table.";
                        domaine = "&domaine.";
                        type = "PMSI séjours";
                        date_debut = datepart(EXE_SOI_DTD);
```

```
                                date_fin = datepart(EXE_SOI_DTF);
                                format date_debut date_fin ddmmyy10.;
                                if 0 <= delai <= (date_fin - date_debut + 1) then output;
                        run;

                        %arret_erreur;

                        proc delete data = orauser.temp_T_&domaine.&an.&table.;
                        run;

                        proc delete data = temp_T_&domaine.&an.&table.;
                        run;

                %end;

%mend UCD_sejours;

*
                ****************************************************************************************************
************************************            ;
*               Macro pour repérer les codes UCD dans le PMSI - ACE

                                                ;
*
                ****************************************************************************************************
************************************            ;

%macro UCD_ACE(an_deb_pmsi=, an_fin_pmsi=, an_crea_t=, an_fin_t=, tbl_codes=, code_UCD7=, code_UCD13=, table=, domaine=,
var_UCD=, var_qte=,
                tbl_out=, tbl_patients=, var_delai=);

                %do i = %sysfunc(max(&an_deb_pmsi., &an_crea_t. )) %to %sysfunc(min(&an_fin_pmsi., &an_fin_t.));

                        %if &i. < 10 %then %let an = 0&i.;
                        %else %let an = &i.;

                        * On définit les variables de jointure;
                        %let var_join1 = ETA_NUM;
                        %let var_join2 = SEQ_NUM;


                        * On joint la table des médicaments avec la table des ACE en filtrant sur les codes UCD sélectionnés;
                        %put Récupération des ACE du PMSI-&domaine. (table &table.) contenant des codes UCD pré-définis - Année
&an. ...;

                        proc sql;

                                %connectora;

                                        CREATE TABLE temp_T_&domaine.&an.&table. AS
                                                SELECT *
                                                FROM CONNECTION TO ORACLE (
                                                        SELECT
                                                                *
                                                        FROM T_&domaine.&an.&table.
                                                        WHERE LENGTH(&var_uCD.) > 1 AND &var_uCD. NOT LIKE '%PH%'
                                                        );

                                DISCONNECT FROM ORACLE;

                        quit;

                        data orauser.temp_T_&domaine.&an.&table.;
                                set temp_T_&domaine.&an.&table.;
                                %if &code_UCD7. = 1 %then
                                        %do;
                                                join_UCD7 = 1*SUBSTR(&var_UCD., 7, 7);
                                        %end;
                                %if &code_UCD13. = 1 %then
                                        %do;
                                                join_UCD13 = 1*SUBSTR(&var_UCD., 6, 7);
```

```
                %end;
        run;

        %arret_erreur;

        proc sql;

                %connectora;

                        CREATE TABLE ext_UCD_PMSI_&domaine._&table._&an. AS SELECT * FROM CONNECTION
TO ORACLE (
                                SELECT DISTINCT
                                        c.&var_join1.,
                                        c.&var_join2.,
                                        pop.BEN_IDT_ANO,
                                        c.EXE_SOI_DTD,
                                        c.EXE_SOI_DTF,
                                        ref.PHA_CND_TOP,
                                        20&an. AS annee,
                                        &var_qte. AS nb_deliv,
                                        &var_delai. AS delai,
                                        cod.*
                                FROM &tbl_patients. pop
                                        INNER JOIN T_&domaine.&an.CSTC c
                                                ON      pop.BEN_NIR_PSA =
c.NIR_ANO_17
                                        INNER JOIN temp_T_&domaine.&an.&table. ucd
                                                ON      c.&var_join1. = ucd.&var_join1.
                                                AND     c.&var_join2. = ucd.&var_join2.
                                        INNER JOIN IR_PHA_R ref
                                                %if &code_UCD7. = 1 %then
                                                        %do;
                                                                ON ucd.join_UCD7 =
ref.PHA_CIP_UCD
                                                        %end;
                                                %if &code_UCD13. = 1 %then
                                                        %do;
                                                                ON ucd.join_UCD13 =
ref.PHA_CIP_UCD
                                                        %end;
                                                INNER JOIN &tbl_codes. cod
                                                        ON ref.PHA_ATC_C07 =
cod.classe_ATC
                                WHERE c.NIR_ANO_17 NOT IN ('&cle_inc1.', '&cle_inc2.') AND
&var_qte. > 0
                                        AND EXE_SOI_DTD IS NOT NULL AND EXE_SOI_DTF IS NOT
NULL
                                        %if &domaine. = MCO %then
                                                %do;
                                                        AND NIR_RET = '0' AND NAI_RET = '0'
AND SEX_RET = '0' AND IAS_RET = '0' AND ENT_DAT_RET = '0'
                                                        AND c.ETA_NUM NOT IN
('130780521', '130783236', '130783293', '130784234', '130804297', '600100101',
                                                                '750041543', '750100018',
'750100042', '750100075', '750100083', '750100091', '750100109',
                                                                '750100125', '750100166',
'750100208', '750100216', '750100232', '750100273', '750100299',
                                                                '750801441', '750803447',
'750803454', '910100015', '910100023', '920100013', '920100021',
                                                                '920100039', '920100047',
'920100054', '920100062', '930100011', '930100037', '930100045',
                                                                '940100027', '940100035',
'940100043', '940100050', '940100068', '950100016', '690783154',
                                                                '690784137', '690784152',
'690784178', '690787478', '830100558')
                                                %end;
                                );

                        DISCONNECT FROM ORACLE;

        quit;
```

```
		data ext_UCD_PMSI_&domaine._&table._&an.;
			set ext_UCD_PMSI_&domaine._&table._&an.;
			length date_debut date_fin 4.;
			table = "&table.";
			domaine = "&domaine.";
			type = "PMSI ACE";
			date_debut = datepart(EXE_SOI_DTD);
			date_fin = datepart(EXE_SOI_DTF);
			format date_debut date_fin ddmmyy10.;
			drop EXE_SOI_DTD EXE_SOI_DTF;
			if 0 <= delai <= (date_fin - date_debut + 1) and
				(year(date_fin) = 20&an. AND 0 <= yrdif(date_debut, date_fin, 'act/act') <= 1) then output;
		run;

		%arret_erreur;

		proc delete data = orauser.temp_T_&domaine.&an.&table.;
		run;

		proc delete data = temp_T_&domaine.&an.&table.;
		run;

	%end;

%mend UCD_ACE;

*
	************************************************************************************************************
**********************************	;
*	Macro finale pour appeler les précédentes, pour chaque table du PMSI

	;
*
	************************************************************************************************************
**********************************	;

%macro extract_UCD_PMSI(annee_deb=, annee_fin=, HAD = 1, MCO = 1, RIP = 1, SSR = 1, tbl_out=, tbl_codes=, tbl_patients=);

	%let an_deb_pmsi = %sysevalf(&annee_deb. - 2000);
	%let an_fin_pmsi = %sysevalf(&annee_fin. - 2000);

	%put Année de début : &an_deb_pmsi.;
	%put Année de fin : &an_fin_pmsi.;

	* Pour le PMSI-HAD;
	%if &HAD. = 1 %then
		%do;

			%let domaine = HAD;

			%put On repère dans &domaine.;

			* Dans les tables MED;
			%UCD_sejours(
				an_deb_pmsi = &an_deb_pmsi.,
				an_fin_pmsi = &an_fin_pmsi.,
				an_crea_t = 09,
				an_fin_t = 14,
				tbl_codes = &tbl_codes.,
				code_UCD7 = 1,
				code_UCD13 = 0,
				table = MED,
				domaine = &domaine.,
				var_UCD = UCD_UCD_COD,
				var_qte = COALESCE(ADM_NBR, 0),
				tbl_out = &tbl_out.,
				var_delai = DAT_DELAI,
				tbl_patients = &tbl_patients.
				);
			%UCD_sejours(
```

```
                an_deb_pmsi = &an_deb_pmsi.,
                an_fin_pmsi = &an_fin_pmsi.,
                an_crea_t = 15,
                an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
                tbl_codes = &tbl_codes.,
                code_UCD7 = 0,
                code_UCD13 = 1,
                table = MED,
                domaine = &domaine.,
                var_UCD = UCD_UCD_COD,
                var_qte = COALESCE(ADM_NBR, 0),
                tbl_out = &tbl_out.,
                var_delai = DAT_DELAI,
                tbl_patients = &tbl_patients.
                );

        * Dans les tables MEDATU;
        %UCD_sejours(
                an_deb_pmsi = &an_deb_pmsi.,
                an_fin_pmsi = &an_fin_pmsi.,
                an_crea_t = 12,
                an_fin_t = 14,
                tbl_codes = &tbl_codes.,
                code_UCD7 = 1,
                code_UCD13 = 0,
                table = MEDATU,
                domaine = &domaine.,
                var_UCD = UCD_UCD_COD,
                var_qte = COALESCE(ADM_NBR, 0),
                tbl_out = &tbl_out.,
                var_delai = DAT_DELAI,
                tbl_patients = &tbl_patients.
                );
        %UCD_sejours(
                an_deb_pmsi = &an_deb_pmsi.,
                an_fin_pmsi = &an_fin_pmsi.,
                an_crea_t = 15,
                an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
                code_UCD7 = 0,
                code_UCD13 = 1,
                tbl_codes = &tbl_codes.,
                table = MEDATU,
                domaine = &domaine.,
                var_UCD = UCD_UCD_COD,
                var_qte = COALESCE(ADM_NBR, 0),
                tbl_out = &tbl_out.,
                var_delai = DAT_DELAI,
                tbl_patients = &tbl_patients.
                );

        * Dans les tables FH;
        %UCD_sejours(
                an_deb_pmsi = &an_deb_pmsi.,
                an_fin_pmsi = &an_fin_pmsi.,
                an_crea_t = 10,
                an_fin_t = 12,
                tbl_codes = &tbl_codes.,
                code_UCD7 = 1,
                code_UCD13 = 0,
                table = FH,
                domaine = &domaine.,
                var_UCD = UCD_UCD_COD,
                var_qte = COALESCE(QUA_COD, 0),
                tbl_out = &tbl_out.,
                var_delai = 0,
                tbl_patients = &tbl_patients.
                );
        %UCD_sejours(
                an_deb_pmsi = &an_deb_pmsi.,
                an_fin_pmsi = &an_fin_pmsi.,
                an_crea_t = 13,
```

```
                                        an_fin_t = 14,
                                        tbl_codes = &tbl_codes.,
                                        code_UCD7 = 1,
                                        code_UCD13 = 0,
                                        table = FH,
                                        domaine = &domaine.,
                                        var_UCD = UCD_UCD_COD,
                                        var_qte = COALESCE(QUA_COD, 0),
                                        tbl_out = &tbl_out.,
                                        var_delai = DEL_DAT_ENT*1,
                                        tbl_patients = &tbl_patients.
                                        );
                                %UCD_sejours(
                                        an_deb_pmsi = &an_deb_pmsi.,
                                        an_fin_pmsi = &an_fin_pmsi.,
                                        an_crea_t = 15,
                                        an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY()))) - 2000),
                                        tbl_codes = &tbl_codes.,
                                        code_UCD7 = 1,
                                        code_UCD13 = 0,
                                        table = FH,
                                        domaine = &domaine.,
                                        var_UCD = UCD_UCD_COD,
                                        var_qte = COALESCE(QUA_COD, 0),
                                        tbl_out = &tbl_out.,
                                        var_delai = DEL_DAT_ENT,
                                        tbl_patients = &tbl_patients.
                                        );

                                * Dans les tables MEDCHL;
                                %UCD_sejours(
                                        an_deb_pmsi = &an_deb_pmsi.,
                                        an_fin_pmsi = &an_fin_pmsi.,
                                        an_crea_t = 16,
                                        an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY()))) - 2000),
                                        tbl_codes = &tbl_codes.,
                                        code_UCD7 = 0,
                                        code_UCD13 = 1,
                                        table = MEDCHL,
                                        domaine = &domaine.,
                                        var_UCD = UCD_UCD_COD,
                                        var_qte = COALESCE(ADM_NBR, 0),
                                        tbl_out = &tbl_out.,
                                        var_delai = DAT_DELAI,
                                        tbl_patients = &tbl_patients.
                                        );

                        %end;
                        * Fin du PMSI-HAD;

        * Pour le PMSI-MCO;
        %if &MCO. = 1 %then
                        %do;

                                %let domaine = MCO;

                                %put On repère dans &domaine.;

                                * Dans les tables MED;
                                %UCD_sejours(
                                        an_deb_pmsi = &an_deb_pmsi.,
                                        an_fin_pmsi = &an_fin_pmsi.,
                                        an_crea_t = 08,
                                        an_fin_t = 14,
                                        tbl_codes = &tbl_codes.,
                                        code_UCD7 = 1,
                                        code_UCD13 = 0,
                                        table = MED,
                                        domaine = &domaine.,
                                        var_UCD = UCD_UCD_COD,
                                        var_qte = COALESCE(ADM_NBR, 0),
```

```
                tbl_out = &tbl_out.,
                var_delai = DELAI,
                tbl_patients = &tbl_patients.
                );
        %UCD_sejours(
                an_deb_pmsi = &an_deb_pmsi.,
                an_fin_pmsi = &an_fin_pmsi.,
                an_crea_t = 15,
                an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
                tbl_codes = &tbl_codes.,
                code_UCD7 = 0,
                code_UCD13 = 1,
                table = MED,
                domaine = &domaine.,
                var_UCD = UCD_UCD_COD,
                var_qte = COALESCE(ADM_NBR, 0),
                tbl_out = &tbl_out.,
                var_delai = DELAI,
                tbl_patients = &tbl_patients.
                );

        * Dans les tables MEDATU;
        %UCD_sejours(
                an_deb_pmsi = &an_deb_pmsi.,
                an_fin_pmsi = &an_fin_pmsi.,
                an_crea_t = 12,
                an_fin_t = 14,
                tbl_codes = &tbl_codes.,
                code_UCD7 = 1,
                code_UCD13 = 0,
                table = MEDATU,
                domaine = &domaine.,
                var_UCD = UCD_UCD_COD,
                var_qte = COALESCE(ADM_NBR, 0),
                tbl_out = &tbl_out.,
                var_delai = DAT_DELAI,
                tbl_patients = &tbl_patients.
                );
        %UCD_sejours(
                an_deb_pmsi = &an_deb_pmsi.,
                an_fin_pmsi = &an_fin_pmsi.,
                an_crea_t = 15,
                an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
                tbl_codes = &tbl_codes.,
                code_UCD7 = 0,
                code_UCD13 = 1,
                table = MEDATU,
                domaine = &domaine.,
                var_UCD = UCD_UCD_COD,
                var_qte = COALESCE(ADM_NBR, 0),
                tbl_out = &tbl_out.,
                var_delai = DAT_DELAI,
                tbl_patients = &tbl_patients.
                );

        * Dans les tables MEDTHROMBO;
        %UCD_sejours(
                an_deb_pmsi = &an_deb_pmsi.,
                an_fin_pmsi = &an_fin_pmsi.,
                an_crea_t = 12,
                an_fin_t = 14,
                tbl_codes = &tbl_codes.,
                code_UCD7 = 1,
                code_UCD13 = 0,
                table = MEDTHROMBO,
                domaine = &domaine.,
                var_UCD = UCD_UCD_COD,
                var_qte = COALESCE(ADM_NBR, 0),
                tbl_out = &tbl_out.,
                var_delai = DAT_DELAI,
                tbl_patients = &tbl_patients.
```

```
		);
%UCD_sejours(
		an_deb_pmsi = &an_deb_pmsi.,
		an_fin_pmsi = &an_fin_pmsi.,
		an_crea_t = 15,
		an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
		tbl_codes = &tbl_codes.,
		code_UCD7 = 0,
		code_UCD13 = 1,
		table = MEDTHROMBO,
		domaine = &domaine.,
		var_UCD = UCD_UCD_COD,
		var_qte = COALESCE(ADM_NBR, 0),
		tbl_out = &tbl_out.,
		var_delai = DAT_DELAI,
		tbl_patients = &tbl_patients.
		);

* Dans les tables FH;
%UCD_sejours(
		an_deb_pmsi = &an_deb_pmsi.,
		an_fin_pmsi = &an_fin_pmsi.,
		an_crea_t = 06,
		an_fin_t = 12,
		tbl_codes = &tbl_codes.,
		code_UCD7 = 1,
		code_UCD13 = 0,
		table = FH,
		domaine = &domaine.,
		var_UCD = UCD_UCD_COD,
		var_qte = COALESCE(QUA_COD, 0),
		tbl_out = &tbl_out.,
		var_delai = 0,
		tbl_patients = &tbl_patients.
		);
%UCD_sejours(
		an_deb_pmsi = &an_deb_pmsi.,
		an_fin_pmsi = &an_fin_pmsi.,
		an_crea_t = 13,
		an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
		tbl_codes = &tbl_codes.,
		code_UCD7 = 1,
		code_UCD13 = 0,
		table = FH,
		domaine = &domaine.,
		var_UCD = UCD_UCD_COD,
		var_qte = COALESCE(QUA_COD, 0),
		tbl_out = &tbl_out.,
		var_delai = DEL_DAT_ENT,
		tbl_patients = &tbl_patients.
		);

* Dans les tables FHSTC;
%UCD_ACE(
		an_deb_pmsi = &an_deb_pmsi.,
		an_fin_pmsi = &an_fin_pmsi.,
		an_crea_t = 09,
		an_fin_t = 12,
		tbl_codes = &tbl_codes.,
		code_UCD7 = 1,
		code_UCD13 = 0,
		table = FHSTC,
		domaine = &domaine.,
		var_UCD = UCD_UCD_COD,
		var_qte = COALESCE(QUA, 0),
		tbl_out = &tbl_out.,
		var_delai = 0,
		tbl_patients = &tbl_patients.
		);
%UCD_ACE(
		an_deb_pmsi = &an_deb_pmsi.,
```

```
                                        an_fin_pmsi = &an_fin_pmsi.,
                                        an_crea_t = 13,
                                        an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
                                        tbl_codes = &tbl_codes.,
                                        code_UCD7 = 1,
                                        code_UCD13 = 0,
                                        table = FHSTC,
                                        domaine = &domaine.,
                                        var_UCD = UCD_UCD_COD,
                                        var_qte = COALESCE(QUA, 0),
                                        tbl_out = &tbl_out.,
                                        var_delai = DEL_DAT_ENT,
                                        tbl_patients = &tbl_patients.
                                        );

                %end;
                * Fin du PMSI-MCO;

        * Pour le PMSI-RIP;
        %if &RIP. = 1 %then
                %do;

                        %let domaine = RIP;

                        %put On repère dans &domaine.;

                        * Dans les tables FH;
                        %UCD_sejours(
                                an_deb_pmsi = &an_deb_pmsi.,
                                an_fin_pmsi = &an_fin_pmsi.,
                                an_crea_t = 10,
                                an_fin_t = 12,
                                tbl_codes = &tbl_codes.,
                                code_UCD7 = 1,
                                code_UCD13 = 0,
                                table = FH,
                                domaine = &domaine.,
                                var_UCD = UCD_UCD_COD,
                                var_qte = COALESCE(QUA_COD, 0),
                                tbl_out = &tbl_out.,
                                var_delai = 0,
                                tbl_patients = &tbl_patients.
                                );
                        %UCD_sejours(
                                an_deb_pmsi = &an_deb_pmsi.,
                                an_fin_pmsi = &an_fin_pmsi.,
                                an_crea_t = 13,
                                an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
                                tbl_codes = &tbl_codes.,
                                code_UCD7 = 1,
                                code_UCD13 = 0,
                                table = FH,
                                domaine = &domaine.,
                                var_UCD = UCD_UCD_COD,
                                var_qte = COALESCE(QUA_COD, 0),
                                tbl_out = &tbl_out.,
                                var_delai = DEL_DAT_ENT,
                                tbl_patients = &tbl_patients.
                                );

                %end;
                * Fin du PMSI-RIP;

        * Pour le PMSI-SSR;
        %if &SSR. = 1 %then
                %do;

                        %let domaine = SSR;

                        %put On repère dans &domaine.;
```

```
* Dans les tables MED;
%UCD_sejours(
        an_deb_pmsi = &an_deb_pmsi.,
        an_fin_pmsi = &an_fin_pmsi.,
        an_crea_t = 10,
        an_fin_t = 14,
        tbl_codes = &tbl_codes.,
        code_UCD7 = 1,
        code_UCD13 = 0,
        table = MED,
        domaine = &domaine.,
        var_UCD = UCD_UCD_COD,
        var_qte = COALESCE(ADM_NBR, 0),
        tbl_out = &tbl_out.,
        var_delai = DAT_DELAI,
        tbl_patients = &tbl_patients.
        );
%UCD_sejours(
        an_deb_pmsi = &an_deb_pmsi.,
        an_fin_pmsi = &an_fin_pmsi.,
        an_crea_t = 15,
        an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY()))) - 2000),
        tbl_codes = &tbl_codes.,
        code_UCD7 = 0,
        code_UCD13 = 1,
        table = MED,
        domaine = &domaine.,
        var_UCD = UCD_UCD_COD,
        var_qte = COALESCE(ADM_NBR, 0),
        tbl_out = &tbl_out.,
        var_delai = DAT_DELAI,
        tbl_patients = &tbl_patients.
        );

* Dans les tables MEDATU;
%UCD_sejours(
        an_deb_pmsi = &an_deb_pmsi.,
        an_fin_pmsi = &an_fin_pmsi.,
        an_crea_t = 12,
        an_fin_t = 14,
        tbl_codes = &tbl_codes.,
        code_UCD7 = 1,
        code_UCD13 = 0,
        table = MEDATU,
        domaine = &domaine.,
        var_UCD = UCD_UCD_COD,
        var_qte = COALESCE(ADM_NBR, 0),
        tbl_out = &tbl_out.,
        var_delai = DAT_DELAI,
        tbl_patients = &tbl_patients.
        );
%UCD_sejours(
        an_deb_pmsi = &an_deb_pmsi.,
        an_fin_pmsi = &an_fin_pmsi.,
        an_crea_t = 15,
        an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY()))) - 2000),
        tbl_codes = &tbl_codes.,
        code_UCD7 = 0,
        code_UCD13 = 1,
        table = MEDATU,
        domaine = &domaine.,
        var_UCD = UCD_UCD_COD,
        var_qte = COALESCE(ADM_NBR, 0),
        tbl_out = &tbl_out.,
        var_delai = DAT_DELAI,
        tbl_patients = &tbl_patients.
        );

* Dans les tables FH;
%UCD_sejours(
        an_deb_pmsi = &an_deb_pmsi.,
```

```
                                        an_fin_pmsi = &an_fin_pmsi.,
                                        an_crea_t = 07,
                                        an_fin_t = 13,
                                        tbl_codes = &tbl_codes.,
                                        code_UCD7 = 1,
                                        code_UCD13 = 0,
                                        table = FH,
                                        domaine = &domaine.,
                                        var_UCD = UCD_UCD_COD,
                                        var_qte = COALESCE(QUA_COD, 0),
                                        tbl_out = &tbl_out.,
                                        var_delai = 0,
                                        tbl_patients = &tbl_patients.
                                        );
                                %UCD_sejours(
                                        an_deb_pmsi = &an_deb_pmsi.,
                                        an_fin_pmsi = &an_fin_pmsi.,
                                        an_crea_t = 14,
                                        an_fin_t = %eval(%sysfunc(YEAR(%sysfunc(TODAY())))  - 2000),
                                        tbl_codes = &tbl_codes.,
                                        code_UCD7 = 1,
                                        code_UCD13 = 0,
                                        table = FH,
                                        domaine = &domaine.,
                                        var_UCD = UCD_UCD_COD,
                                        var_qte = COALESCE(QUA_COD, 0),
                                        tbl_out = &tbl_out.,
                                        var_delai = DEL_DAT_ENT,
                                        tbl_patients = &tbl_patients.
                                        );

                        %end;
                        * Fin du PMSI-SSR;

        * Empilage de la table temporaire à la table Résultat;
        data &tbl_out.;
                        length table $15.;
                        set ext_UCD_PMSI_:;
        run;

        %arret_erreur;

        * Suppression de la table temporaire;
        proc datasets library = work memtype = data nolist;
                        delete ext_UCD_PMSI_:;
        run;

%mend extract_UCD_PMSI;
```

3.2 Construction des tables de valeurs et référentiels

3.2.1 Tables_de_valeurs_et_referentiels_Codes_actes_PMSI_BPCO.sas

```
/*
        ********************************************************************************************************
*********************************          */
/*

                                                                                                                */
/*                      Table de valeurs contenant les codes actes nécessaires à l ensemble du projet (pour le repérage dans le PMSI)
                                                                                */
/*

                                                                                                                */
/*
        ********************************************************************************************************
*********************************          */

* On crée la table;
data codes_actes_BPCO;
infile datalines delimiter = "£" dsd ;
retain code_acte reperage;
length code_acte $ 3 reperage 3.;
input code_acte $ reperage;
datalines;
AMK£7
AMC£7
AMS£7
BPC£7
TTE£32
TDT£32
APU£32
APC£32
CCX£32
GS£32
G£32
C£32
CS£32
CA£32
TCP£32
TEP£32
TLC£32
TLE£32
TC£32
TCG£32
TE1£32
TE2£32
VGS£32
VG£32
V£32
VS£32
VL£32
APV£32
VA£32
VU£32
ADA£32
ADC£32
ADE£32
ATM£32
ADI£32
TTE£33
TDT£33
CCX£33
GS£33
G£33
C£33
```

```
CS£33
CA£33
TCP£33
TEP£33
TLC£33
TLE£33
TC£33
TCG£33
TE1£33
TE2£33
VGS£33
VG£33
V£33
VS£33
VL£33
VA£33
VU£33
ADA£33
ADC£33
ADE£33
ATM£33
ADI£33
;
run;

* On supprime la table si elle existe;
%suppr_table(
		lib = orauser,
		table = codes_actes_BPCO
		);

* On ajoute la longueur de code_CIM;
data orauser.codes_actes_BPCO;
		set codes_actes_BPCO;
		taille = length(compress(code_acte));
run;

%suppr_table(
		lib = work,
		table = codes_actes_BPCO
		);
```

3.2.2 Tables_de_valeurs_et_referentiels_Codes_CCAM_BPCO.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*		Table de valeurs contenant les codes CCAM nécessaires à l'ensemble du projet
                                                                                                                    */
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/

* On supprime la table si elle existe;
%suppr_table(
		lib = orauser,
		table = codes_CCAM_BPCO
		);

* On crée la table;
data orauser.codes_CCAM_BPCO;
infile datalines delimiter = "£" dsd ;
retain code_CCAM reperage;
length code_CCAM $ 7 reperage 3.;
input code_CCAM $ reperage;
datalines;
GLQP003£1
GERD001£1
GERD002£1
GLQD001£1
GLQD003£1
YYYY025£1
GLQP008£1
GLQP002£1
GLQP009£1
GLQP014£1
GLQP011£1
GLQP012£2
GLHF001£3
GLMF001£3
GLQF001£3
GEME121£4
GLLD017£5
GLLD012£6
EQQP002£7
EQQP003£7
GLRP001£7
GLRP002£7
GLRP003£7
GLRP004£7
NEMA018£8
NEKA018£8
NEKA011£8
NELA003£8
NELA003£8
NEKA002£8
NEKA012£8
NEKA014£8
NEKA010£8
NEKA016£8
NEKA017£8
NEKA021£8
NEKA015£8
NEKA013£8
NEKA019£8
```

NEGA004£8
NEGA005£8
NEGA002£8
NEGA003£8
NEGA001£8
NEKA022£8
NEKA004£8
NEKA009£8
NEKA020£8
NEKA007£8
NEKA005£8
NEKA003£8
NEKA008£8
NEKA006£8
NEKA001£8
NELA002£8
NELA001£8
NEMA011£8
NFKA009£9
NFKA006£9
NFKA007£9
NFKA008£9
NFGA002£9
NFGA001£9
NFKA004£9
NFKA003£9
NFKA005£9
NFKA001£9
NFKA002£9
NFLA002£9
NFLA001£9
NFMA006£9
AFCA001£10
AFCA002£10
AFCA003£10
AFCA004£10
AFFA001£10
AFFA002£10
AFFA003£10
AFFA004£10
AFFA005£10
AFFA006£10
AFFA007£10
AFFA008£10
AFFA009£10
AFFA010£10
AFFA011£10
AFGA001£10
AFGA002£10
AFJA001£10
AFJA002£10
AFJA003£10
AFJA004£10
AFJA005£10
AFKB001£10
AFKB002£10
AFLA003£10
AFLB008£10
AFLB016£10
AFLB017£10
AFPA001£10
AFSA001£10
AFSA002£10
AFSA003£10
GGJB002£11
GGJB001£11
DDMA003£28
DDMA004£28
DDMA005£28
DDMA006£28
DDMA007£28

DDMA008£28
DDMA009£28
DDMA011£28
DDMA012£28
DDMA013£28
DDMA015£28
DDMA016£28
DDMA017£28
DDMA018£28
DDMA019£28
DDMA020£28
DDMA021£28
DDMA022£28
DDMA023£28
DDMA024£28
DDMA025£28
DDMA026£28
DDMA027£28
DDMA028£28
DDMA029£28
DDMA030£28
DDMA031£28
DDMA032£28
DDMA033£28
DDMA034£28
DDMA035£28
DDMA036£28
DDMA037£28
DDMA038£28
DBEA001£29
DBKA001£29
DBKA002£29
DBKA003£29
DBKA004£29
DBKA005£29
DBKA006£29
DBKA007£29
DBKA008£29
DBKA009£29
DBKA010£29
DBKA011£29
DBKA012£29
DBMA001£29
DBMA004£29
DBMA005£29
DBMA006£29
DBMA007£29
DBMA009£29
DBMA010£29
DBMA013£29
DBMA015£29
DGKA011£29
DGKA014£29
DGKA015£29
DGKA018£29
DDAF003£30
DDAF004£30
DDAF006£30
DDAF007£30
DDAF008£30
DDAF009£30
DDPF002£30
DFAF001£30
DFAF003£30
DGAF004£30
DGAF005£30
DGAF007£30
DGLF001£30
DGLF002£30
DGLF003£30
DGLF005£30

```
DGLF012£30
DGPF002£30
DHAF001£30
DHAF004£30
DHPF002£30
EAAF002£30
EAAF900£30
EAAF902£30
EBAF001£30
EBAF006£30
EBAF010£30
EBAF011£30
EBAF014£30
ECAF001£30
ECAF004£30
ECLF003£30
ECLF004£30
ECPF001£30
ECPF002£30
ECPF005£30
EDAF001£30
EDAF003£30
EDAF005£30
EDAF006£30
EDAF010£30
EDLF004£30
EDLF005£30
EDLF006£30
EDLF007£30
EDLF008£30
EDLF013£30
EDPF001£30
EDPF004£30
EDPF005£30
EDPF006£30
EDPF009£30
EEAF002£30
EEAF004£30
EEAF006£30
EELF002£30
EEPF001£30
EFAF001£30
EFLF001£30
EFPF001£30
EGAF002£30
EGAF004£30
EGPF001£30
EHAF001£30
EHCF002£30
ENAF001£30
EPPF003£30
EZAF002£30
EZJF001£30
EZNF002£30
EZPF003£30
;
run;
```

3.2.3 Tables_de_valeurs_et_referentiels_Codes_CCAM_Charlson.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*

                                                                                                                    */
/*			Table de valeurs contenant les codes CCAM nécessaires au calcul du score de Charlson
                                                                                                                */
/*

                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/

* On supprime la table si elle existe;
%suppr_table(
		lib = orauser,
		table = codes_CCAM_Charlson
		);

* On crée la table;
data orauser.codes_CCAM_Charlson;
infile datalines delimiter = "£" dsd ;
retain reperage code_CCAM;
length code_CCAM $ 7 reperage 3.;
input reperage code_CCAM $;
datalines;
3£EAAF002
3£EAAF900
3£EBAF001
3£EBAF006
3£EBAF010
3£EBAF011
3£ECAF001
3£ECAF004
3£ECLF003
3£ECLF004
3£ECPF001
3£ECPF002
3£ECPF005
3£EDAF001
3£EDAF003
3£EDAF005
3£EDAF006
3£EDAF010
3£EDLF004
3£EDLF005
3£EDLF006
3£EDLF007
3£EDLF008
3£EDLF013
3£EDPF001
3£EDPF004
3£EDPF005
3£EDPF006
3£EDPF009
3£EEAF002
3£EEAF004
3£EEAF006
3£EELF002
3£EEPF001
3£EFAF001
3£EFLF001
3£EFPF001
3£EGAF002
3£EGAF004
```

```
3£EGPF001
3£ENAF001
3£EPPF003
3£EZAF002
3£EZJF001
3£EZNF002
3£EZPF003
3£DGLF002
3£DGLF001
3£DHAF001
3£DHPF002
3£DHAF004
12£JVJB001
12£JVJF004
12£JVJF008
12£JVRP004
12£JVRP007
12£JVRP008
12£YYYY007
;
run;
```

3.2.4 Tables_de_valeurs_et_referentiels_Codes_CIM10_ALD_BPCO.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*

                                                                                                                */
/*		Table de valeurs contenant les codes CIM10 nécessaires à l'ensemble du projet (pour le repérage des ALD)
                                                                                                */
/*

                                                                                                                */
/*
	*****************************************************************************************************************
**********************************	*/

* On crée la table;
data ALD_codes_CIM_BPCO;
infile datalines delimiter = "£" dsd ;
retain code_CIM reperage;
length code_CIM $ 7 reperage 3.;
input code_CIM $ reperage;
datalines;
F00£50
G30£50
F01£51
F02£51
F03£51
J44£52
J45£53
E84£54
M05£56
M06£56
M070£57
M071£57
M072£57
M073£57
M076£57
M45£57
M46£57
;
run;

* On supprime la table si elle existe;
%suppr_table(
		lib = orauser,
		table = ALD_codes_CIM_BPCO
		);

* On ajoute la longueur de code_CIM;
data orauser.ALD_codes_CIM_BPCO;
		set ALD_codes_CIM_BPCO;
		taille = length(compress(code_CIM));
run;

proc delete data = ALD_codes_CIM_BPCO;
run;
```

3.2.5 Tables_de_valeurs_et_referentiels_Codes_CIM10_Charlson.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*

                                                                                                                    */
/*			Table de valeurs contenant les codes CIM10 nécessaires au calcul du score de Charlson
                                                                                                                */
/*

                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/

* On supprime la table si elle existe;
%suppr_table(
		lib = orauser,
		table = codes_CIM_Charlson
		);

* On crée la table;
data codes_CIM_Charlson;
infile datalines delimiter = "£" dsd ;
retain reperage code_CIM;
length reperage 3. code_CIM $ 7;
input reperage code_CIM $;
datalines;
1£I21
1£I22
2£I50
3£I70
3£I71
4£G45
4£G46
4£I60
4£I61
4£I62
4£I63
4£I64
4£I65
4£I66
4£I67
4£I68
4£I69
5£F00
5£F01
5£F02
5£F03
5£G30
6£J40
6£J41
6£J42
6£J43
6£J44
6£J45
6£J46
6£J47
6£J60
6£J61
6£J62
6£J63
6£J64
6£J65
6£J66
6£J67
7£M05
```

7£M06
7£M32
7£M33
7£M34
8£K25
8£K26
8£K27
8£K28
9£B18
9£K73
9£K74
11£G81
11£G82
12£N18
12£N19
14£C00
14£C01
14£C02
14£C03
14£C04
14£C05
14£C06
14£C07
14£C08
14£C09
14£C10
14£C11
14£C12
14£C13
14£C14
14£C15
14£C16
14£C17
14£C18
14£C19
14£C20
14£C21
14£C22
14£C23
14£C24
14£C25
14£C26
14£C30
14£C31
14£C32
14£C33
14£C34
14£C37
14£C38
14£C39
14£C40
14£C41
14£C43
14£C45
14£C46
14£C47
14£C48
14£C49
14£C50
14£C51
14£C52
14£C53
14£C54
14£C55
14£C56
14£C57
14£C58
14£C60
14£C61
14£C62
14£C63

14£C64
14£C65
14£C66
14£C67
14£C68
14£C69
14£C70
14£C71
14£C72
14£C73
14£C74
14£C75
14£C76
14£C81
14£C82
14£C83
14£C84
14£C85
14£C88
14£C90
14£C91
14£C92
14£C93
14£C94
14£C95
14£C96
14£C97
16£C77
16£C78
16£C79
16£C80
17£B20
17£B21
17£B22
17£B24
17£Z21
1£I252
2£I110
2£I130
2£I132
3£I731
3£I738
3£I739
3£I771
3£I790
3£I792
3£K551
3£K558
3£K559
3£Z958
3£Z959
4£H340
5£F051
5£G311
6£I278
6£I279
6£J684
6£J701
6£J703
7£M315
7£M351
7£M353
7£M360
9£K700
9£K701
9£K702
9£K703
9£K709
9£K713
9£K714
9£K715

9£K717
9£K760
9£K762
9£K763
9£K764
9£K768
9£K769
9£Z944
10£E100
10£E101
10£E106
10£E108
10£E109
10£E110
10£E111
10£E116
10£E118
10£E119
10£E120
10£E121
10£E126
10£E128
10£E129
10£E130
10£E131
10£E136
10£E138
10£E139
10£E140
10£E141
10£E146
10£E148
10£E149
11£G041
11£G114
11£G801
11£G802
11£G830
11£G831
11£G832
11£G833
11£G834
11£G839
12£I120
12£I131
12£N032
12£N033
12£N034
12£N035
12£N036
12£N037
12£N052
12£N053
12£N054
12£N055
12£N056
12£N057
12£N250
12£Z490
12£Z491
12£Z492
12£Z940
12£Z992
13£E102
13£E103
13£E104
13£E105
13£E107
13£E112
13£E113
13£E114

```
13£E115
13£E117
13£E122
13£E123
13£E124
13£E125
13£E127
13£E132
13£E133
13£E134
13£E135
13£E137
13£E142
13£E143
13£E144
13£E145
13£E147
15£I850
15£I859
15£I864
15£I982
15£K704
15£K711
15£K721
15£K729
15£K765
15£K766
15£K767
;
run;

data orauser.codes_CIM_Charlson;
	set codes_CIM_Charlson;
	taille = length(compress(code_CIM));
run;

proc delete data = codes_CIM_Charlson;
run;
```

3.2.6 Tables_de_valeurs_et_referentiels_Codes_CIM10_Hospit_BPCO.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*

                                                                                                                                */
/*			Table de valeurs contenant les codes CIM10 nécessaires à l'ensemble du projet (pour le repérage des
hospitalisations)                                                */
/*

                                                                                                                                */
/*
	*****************************************************************************************************************
**********************************	*/

* On crée la table;
data diag_codes_CIM_BPCO;
infile datalines delimiter = "£" dsd ;
retain code_CIM reperage flag_pneumo;
length code_CIM $ 7 reperage 3. flag_pneumo 3.;
input code_CIM $ reperage flag_pneumo;
datalines;
J93£11£1
J942£11£0
J86£11£0
I20£12£0
I21£12£0
I22£12£0
Z515£13£0
J450£14£0
J451£14£0
J458£14£0
J459£14£0
J46£15£0
J440£16£0
J441£16£0
J448£16£0
J449£16£0
J960£17£0
J181£18£0
J09£19£0
J10£19£0
J11£19£0
I26£20£0
I130£21£0
I132£21£0
I110£21£0
I50£21£0
I501£22£0
J12£23£0
J13£23£0
J14£23£0
J15£23£0
J16£23£0
J17£23£0
J18£23£0
J12£24£0
J13£24£0
J14£24£0
J15£24£0
J16£24£0
J17£24£0
J18£24£0
J12£25£0
J13£25£0
J14£25£0
J15£25£0
```

```
J16£25£0
J17£25£0
J18£25£0
M05£26£0
M06£26£0
M070£27£0
M071£27£0
M072£27£0
M073£27£0
M076£27£0
M45£27£0
M46£27£0
Z511£58£0
Z5101£59£0
;
run;

* On supprime la table si elle existe;
%suppr_table(
		lib = orauser,
		table = diag_codes_CIM_BPCO
		);

* On ajoute la longueur de code_CIM;
data orauser.diag_codes_CIM_BPCO;
		set diag_codes_CIM_BPCO;
		taille = length(compress(code_CIM));
run;

proc delete data = diag_codes_CIM_BPCO;
run;
```

3.2.7 Tables_de_valeurs_et_referentiels_Codes_CIP_UCD_BPCO.sas

```
	/*
	*************************************************************************************************************
**********************************	*/
	/*


	*/
	/*			Table de valeurs contenant les codes ATC nécessaires à l'ensemble du projet

		*/
	/*


	*/
	/*
	*************************************************************************************************************
**********************************	*/

	* On supprime la table si elle existe;
	%suppr_table(
		lib = orauser,
		table = codes_ATC_BPCO
		);

	* On crée la table;
	data orauser.codes_ATC_BPCO;
	infile datalines delimiter = "£" dsd ;
	retain classe_ATC reperage;
	length classe_ATC $ 7 reperage 3.;
	input classe_ATC $ reperage;
	datalines;
	N07BA01£35
	N07BA03£35
	N06AX12£35
	J01CA04£36
	J01CR02£36
	J01FG01£36
	J01FA07£36
	J01FA02£36
	J01FA10£36
	J01DC02£36
	J01DD13£36
	J01DD04£36
	J01MA12£36
	J01DD08£36
	J01FA09£36
	J01MA02£36
	J01FA15£36
	R03DX05£37
	R03DX09£38
	R03DX08£38
	R03DC03£39
	J07BB01£40
	J07BB02£40
	J07BB03£40
	J07BB02£40
	J07BB02£40
	R03BA01£41
	R03BA02£41
	R03BA05£41
	R03AC02£42
	R03AC03£42
	R03BB01£43
	R03AL01£44
	R03AC13£45
	R03AC18£45
	R03AC19£45
```

```
R03AC12£45
R03BB06£46
R03BB04£46
R03BB07£46
R03AL04£47
R03AL06£47
R03BB54£47
R03AL03£47
R03AK08£48
R03AK06£48
R03AK07£48
R03AK10£48
R03AL09£49
R03AL08£49
;
run;
```

3.2.8 Tables_de_valeurs_et_referentiels_Codes_CIP_UCD_Charlson.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*

                                                                                                                    */
/*			Table de valeurs contenant les codes ATC nécessaires au calcul du score de Charlson
                                                                                                                */
/*

                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/

* On supprime la table si elle existe;
%suppr_table(
		lib = orauser,
		table = codes_ATC_Charlson
		);

* On crée la table;
data orauser.codes_ATC_Charlson;
infile datalines delimiter = "£" dsd ;
retain classe_ATC reperage;
length classe_ATC $ 7 reperage 3.;
input classe_ATC $ reperage;
datalines;
N06DX01£5
N06DA01£5
N06DA02£5
N06DA03£5
N06DA04£5
R03AB03£6
R03AC02£6
R03AC03£6
R03AC04£6
R03AC08£6
R03AC12£6
R03AC13£6
R03AC18£6
R03AC19£6
R03AK06£6
R03AK07£6
R03AK08£6
R03AK10£6
R03AK11£6
R03AL01£6
R03AL02£6
R03AL03£6
R03AL04£6
R03AL08£6
R03AL09£6
R03BA01£6
R03BA02£6
R03BA03£6
R03BA05£6
R03BA07£6
R03BA08£6
R03BB01£6
R03BB02£6
R03BB04£6
R03BB06£6
R03BB07£6
R03BB54£6
R03BC01£6
R03BC03£6
```

```
R03CC02£6
R03CC03£6
R03CC12£6
R03DA04£6
R03DA05£6
R03DA08£6
R03DC03£6
R03DX03£6
R03DX05£6
R03DX09£6
R03DX10£6
A10AB01£10
A10AB03£10
A10AB04£10
A10AB05£10
A10AB06£10
A10AC01£10
A10AC03£10
A10AC04£10
A10AD01£10
A10AD03£10
A10AD04£10
A10AD05£10
A10AE01£10
A10AE02£10
A10AE03£10
A10AE04£10
A10AE05£10
A10AE06£10
A10AE30£10
A10AE56£10
A10BA02£10
A10BB01£10
A10BB03£10
A10BB04£10
A10BB06£10
A10BB07£10
A10BB09£10
A10BB12£10
A10BD02£10
A10BD03£10
A10BD05£10
A10BD07£10
A10BD08£10
A10BD10£10
A10BF01£10
A10BF02£10
A10BG02£10
A10BG03£10
A10BH01£10
A10BH02£10
A10BH03£10
A10BJ06£10
A10BX02£10
A10BX04£10
A10BX07£10
A10BX14£10
;
run;
```

3.2.9 Tables_de_valeurs_et_referentiels_Codes_CSARR_BPCO.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                */
/*			Table de valeurs contenant les codes CSARR nécessaires à l'ensemble du projet
                                                                                                                */
/*
                                                                                                                */
/*
	*****************************************************************************************************************
**********************************	*/

* On supprime la table si elle existe;
%suppr_table(
		lib = orauser,
		table = codes_CSARR_BPCO
		);

* On crée la table;
data orauser.codes_CSARR_BPCO;
infile datalines delimiter = "£" dsd ;
retain code_CSARR reperage;
length code_CSARR $ 7 reperage 3.;
input code_CSARR $ reperage;
datalines;
DKQ+008£7
EQQ+206£7
EQR+175£7
EQR+275£7
ANR+036£7
GLR+074£7
GLR+077£7
GLR+093£7
GLR+131£7
GLR+139£7
GLR+167£7
GLR+169£7
GLR+170£7
GLR+186£7
GLR+224£7
GLR+226£7
GLR+236£7
GLR+285£7
PCQ+179£7
PCR+025£7
PEQ+266£7
NKR+059£7
NKR+085£7
PER+118£7
PER+285£7
DKR+013£7
DKR+016£7
DKR+061£7
DKR+118£7
DKR+181£7
DKR+182£7
DKR+194£7
DKR+195£7
DKR+200£7
DKR+247£7
DKR+254£7
DKR+291£7
PCM+064£7
PCM+253£7
```

```
PCM+262£7
PCM+283£7
PCR+004£7
PCR+272£7
ZZC+028£7
ZZC+255£7
ZZQ+027£7
ZZQ+261£7
ZZR+227£7
ZZR+238£7
ZZQ+112£7
ZZQ+192£7
ZZR+020£7
ZZR+293£7
GLQ+043£7
GLQ+175£7
GLR+206£7
;
run;
```

3.2.10 Tables_de_valeurs_et_referentiels_Codes_GHM_BPCO.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*

*/
/*			Table de valeurs contenant les codes GHM nécessaires à l'ensemble du projet

	*/
/*

*/
/*
	*****************************************************************************************************************
**********************************	*/

* On crée la table;
data codes_GHM_BPCO;
infile datalines delimiter = "£" dsd ;
retain code_GHM reperage;
length code_GHM $ 5 reperage 3.;
input code_GHM $ reperage;
datalines;
05C04£28
05C05£28
05C02£29
05C03£29
05K05£30
05K06£30
28Z07£58
28Z10£59
28Z11£59
28Z18£59
28Z23£59
28Z24£59
28Z25£59
;
run;

* On supprime la table si elle existe;
%suppr_table(
		lib = orauser,
		table = codes_GHM_BPCO
		);

* On ajoute la longueur de code_GHM;
data orauser.codes_GHM_BPCO;
		set codes_GHM_BPCO;
		taille = length(compress(code_GHM));
run;

proc delete data = codes_GHM_BPCO;
run;
```

3.2.11 Tables_de_valeurs_et_referentiels_Codes_GHM_Charlson.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*

                                                                                                                    */
/*			Table de valeurs contenant les codes GHM nécessaires au calcul du score de Charlson
                                                                                                                */
/*

                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/

* On crée la table;
data codes_GHM_Charlson;
infile datalines delimiter = "£" dsd ;
retain reperage code_GHM;
length code_GHM $ 7 reperage 3.;
input reperage code_GHM $;
datalines;
12£28Z04Z
;
run;

* On supprime la table si elle existe;
%suppr_table(
		lib = orauser,
		table = codes_GHM_Charlson
		);

* On ajoute la longueur de code_GHM;
data orauser.codes_GHM_Charlson;
		set codes_GHM_Charlson;
		length taille 3.;
		taille = length(compress(code_GHM));
run;

proc delete data = codes_GHM_Charlson;
run;
```

3.2.12 Tables_de_valeurs_et_referentiels_Codes_GME_BPCO.sas

```
/*
	**********************************************************************************************************
**********************************	*/
/*
                                                                                                                */
/*			Table de valeurs contenant les codes GME nécessaires à l'ensemble du projet

	*/
/*
                                                                                                                */
/*
	**********************************************************************************************************
**********************************	*/

* On supprime la table si elle existe;
%suppr_table(
		lib = orauser,
		table = codes_GME_BPCO
		);

* On crée la table;
data orauser.codes_GME_BPCO;
infile datalines delimiter = "£" dsd ;
retain code_GME reperage;
length code_GME $ 2 reperage 3.;
input code_GME $ reperage;
datalines;
04£7
;
run;
```

3.2.13 Tables_de_valeurs_et_referentiels_Codes_LPP_BPCO.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                */
/*			Table de valeurs contenant les codes LPP nécessaires à l'ensemble du projet
                                                                                                                */
/*
                                                                                                                */
/*
	*****************************************************************************************************************
**********************************	*/

* On supprime la table si elle existe;
%suppr_table(
		lib = orauser,
		table = codes_LPP_BPCO
		);

* On crée la table;
data orauser.codes_LPP_BPCO;
infile datalines delimiter = "£" dsd ;
length code_LPP 5. reperage 3.;
input code_LPP reperage;
datalines;
1136581£5
1130220£5
1163030£6
1196270£6
;
run;
```

3.2.14 Tables_de_valeurs_et_referentiels_Codes_NABM_BPCO.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*

                                                                                                                    */
/*			Table de valeurs contenant les codes NABM nécessaires à l'ensemble du projet
                                                                                                                    */
/*

                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/

* On supprime la table si elle existe;
%suppr_table(
		lib = orauser,
		table = codes_NABM_BPCO
		);

* On crée la table;
data orauser.codes_NABM_BPCO;
infile datalines delimiter = "£" dsd ;
retain code_NABM reperage;
length code_NABM $ 4 reperage 3.;
input code_NABM $ reperage;
datalines;
0999£31
1612£31
0571£31
;
run;
```

3.2.15 Tables_de_valeurs_et_referentiels_Natures_de_prestation_BPCO.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*

                                                                                                                */
/*		Table de valeurs contenant les natures de prestations nécessaires à l'ensemble du projet
                                                                                                                */
/*

                                                                                                                */
/*
	*****************************************************************************************************************
**********************************	*/

* On supprime la table si elle existe;
%suppr_table(
	lib = orauser,
	table = codes_presta_BPCO
	);

* On crée la table;
data orauser.codes_presta_BPCO;
infile datalines delimiter = "£" dsd ;
retain code_presta reperage;
length code_presta 3. reperage 3.;
input code_presta reperage;
datalines;
3122£7
3121£7
3125£7
9570£7
1096£32
1097£32
1101£32
1103£32
1105£32
1109£32
1110£32
1111£32
1112£32
1115£32
1117£32
1157£32
1158£32
1164£32
1165£32
1191£32
1192£32
1193£32
1194£32
1209£32
1210£32
1211£32
1212£32
1214£32
1215£32
1221£32
1222£32
1323£32
1321£32
1324£32
1352£32
1351£32
1096£33
1097£33
1105£33
```

```
1109£33
1110£33
1111£33
1112£33
1115£33
1117£33
1157£33
1158£33
1164£33
1165£33
1191£33
1192£33
1193£33
1194£33
1209£33
1210£33
1211£33
1212£33
1214£33
1221£33
1222£33
1323£33
1321£33
1324£33
1352£33
1351£33
;
run;
```

3.3 Sélection des patients

3.3.1 01_Chainage.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*

*/
/*			Sélection des patients

*/
/*

*/
/*
	*****************************************************************************************************************
**********************************	*/

*
	*****************************************************************************************************************
**********************************;
*	On récupère tous les BEN_NIR_PSA de chaque patient;

proc sql undo_policy = none;

	%connectora;

		CREATE TABLE patients_IR_BEN_R AS
			SELECT * FROM CONNECTION TO ORACLE (
				SELECT DISTINCT
					BEN_IDT_ANO,
					BEN_NIR_PSA,
					BEN_RNG_GEM
				FROM IR_BEN_R
				);

	disconnect from oracle;

quit;

%suppr_table(
	lib = orauser,
	table = patients_IR_BEN_R
	);

data orauser.patients_IR_BEN_R;
	set patients_IR_BEN_R;
run;

*
	*****************************************************************************************************************
**********************************;
*	On récupère la ligne avec BEN_DTE_MAJ max pour chaque patient;

proc sql undo_policy = none;

	%connectora;

		CREATE TABLE BEN_DTE_MAJ_max AS
			SELECT * FROM CONNECTION TO ORACLE (
				SELECT
					BEN_IDT_ANO,
					MAX(BEN_DTE_MAJ) AS BEN_DTE_MAJ_max
				FROM IR_BEN_R
				GROUP BY BEN_IDT_ANO
```

```
                                        );

        disconnect from oracle;

quit;

%suppr_table(
        lib = orauser,
        table = BEN_DTE_MAJ_max
        );

data orauser.BEN_DTE_MAJ_max;
        set BEN_DTE_MAJ_max;
run;


*
        ************************************************************************************************************
**********************************;
*       On récupère les informations de ces patients;

proc sql undo_policy = none;

        %connectora;

                CREATE TABLE pop.T_INDI_BPCO_&an_N. AS
                        SELECT * FROM CONNECTION TO ORACLE (
                                SELECT DISTINCT
                                        a.BEN_IDT_ANO,
                                        b.BEN_NIR_PSA,
                                        b.BEN_RNG_GEM,
                                        c.BEN_CDI_NIR,
                                        c.BEN_NAI_ANN,
                                        c.BEN_NAI_MOI,
                                        c.BEN_SEX_COD,
                                        c.BEN_DCD_DTE,
                                        c.BEN_RES_DPT,
                                        c.BEN_RES_COM,
                                        SUBSTR(c.ORG_AFF_BEN, 1, 3) AS RGM_GRG_COD
                                FROM BEN_DTE_MAJ_max a
                                        INNER JOIN patients_IR_BEN_R b
                                                ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
                                        INNER JOIN IR_BEN_R c
                                                ON a.BEN_IDT_ANO = c.BEN_IDT_ANO
                                                AND a.BEN_DTE_MAJ_max = c.BEN_DTE_MAJ
                                );

        disconnect from oracle;

quit;

data pop.T_INDI_BPCO_&an_N. (rename = (BEN_DCD_DTE2 = BEN_DCD_DTE));
        set pop.T_INDI_BPCO_&an_N.;
        length BEN_DCD_DTE2 4.;
        BEN_DCD_DTE2 = datepart(BEN_DCD_DTE);
        if BEN_DCD_DTE2 = "01JAN1600"d then
                BEN_DCD_DTE2 = .;
        format BEN_DCD_DTE2 date9.;
        drop BEN_DCD_DTE;
run;

*
        ************************************************************************************************************
**********************************;
*       Exclusion des NIR fictifs;

data pop.T_INDI_BPCO_&an_N.;
        set pop.T_INDI_BPCO_&an_N.;
        exclus_NIR_fictif = "0";
        if BEN_CDI_NIR ne "00" then
                exclus_NIR_fictif = "1";
```

```
run;

*
	***********************************************************************************************************
***********************************;
*	Suppression des tables temporaires;

proc delete data = patients_IR_BEN_R BEN_DTE_MAJ_max;
run; quit;

proc delete data = orauser.patients_IR_BEN_R orauser.BEN_DTE_MAJ_max;
run; quit;

*
	***********************************************************************************************************
***********************************;
*	Vérifications;

*	Variables BEN_CDI_NIR * EXCLUS_NIR_FICTIF;
%proc_freq(
	in_tbl = pop.T_INDI_BPCO_&an_N.,
	out_tbl = _pop_NIR_FICTIF,
	list_var_in = BEN_CDI_NIR*EXCLUS_NIR_FICTIF,
	list_var_out = BEN_CDI_NIR EXCLUS_NIR_FICTIF Frequency
	);

*	Variable BEN_RES_DPT;
%proc_freq(
	in_tbl = pop.T_INDI_BPCO_&an_N.,
	out_tbl = _pop_BEN_RES_DPT,
	list_var_in = BEN_RES_DPT,
	list_var_out = BEN_RES_DPT Frequency
	);

*	Variable BEN_SEX_COD;
%proc_freq(
	in_tbl = pop.T_INDI_BPCO_&an_N.,
	out_tbl = _pop_BEN_SEX_COD,
	list_var_in = BEN_SEX_COD,
	list_var_out = BEN_SEX_COD Frequency
	);
```

3.3.2 02_Exclusion_des_moins_de_40_ans.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*

*/
/*			Sélection des patients - Exclusion de patients de moins de 40 ans

	*/
/*

*/
/*
	*****************************************************************************************************************
**********************************	*/

*
	*****************************************************************************************************************
**********************************;
*	On ajout l âge l année N :
				si année et mois inconnus, ou année de naissance avant 1900 ou après l année de traitement : âge
manquant
				si mois inconnu, on force au milieu d année : juin
				si année et mois connus, on force le jour au milieu du mois = 15;

data pop.T_INDI_BPCO_&an_N.;
	set pop.T_INDI_BPCO_&an_N.;
	length age 3.;
	* Âge;
	if 1900 <= input(BEN_NAI_ANN, 4.) <= year(today()) then
		do;
			if 1 <= input(BEN_NAI_MOI, 2.) <= 12 then
				Age = int(yrdif(mdy(input(BEN_NAI_MOI, 2.), 15, input(BEN_NAI_ANN, 4.)),
"01JAN&annee_N."d, 'ACTUAL'));
			else
				Age = int(yrdif(mdy(06, 15, input(BEN_NAI_ANN, 4.)), "01JAN&annee_N."d, 'ACTUAL'));
		end;
	else Age = .;
run;

*
	*****************************************************************************************************************
**********************************;
*	On exclut les patients qui n ont pas 40 ans;

proc sql;

	ALTER TABLE pop.T_INDI_BPCO_&an_N. ADD exclus_pas_40ans VARCHAR(1);

	UPDATE pop.T_INDI_BPCO_&an_N. SET
		exclus_pas_40ans = CASE		WHEN Age < 40 AND Age NE . THEN "1"
									ELSE "0"
									END;

quit;

*
	*****************************************************************************************************************
**********************************;
*	Vérifications;

*	Variables AGE * EXCLUS_PAS_40ans ;
%proc_freq(
	in_tbl = pop.T_INDI_BPCO_&an_N.,
	out_tbl = _pop_AGE_EXCLUS,
	list_var_in = AGE*EXCLUS_PAS_40ans,
	list_var_out = AGE EXCLUS_PAS_40ans Frequency
```

```
);
```

3.3.3 03_Exclusion_des_jumeaux.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                */
/*			Sélection des patients - Exclusion des jumeaux

                                                */
/*
                                                                                                                */
/*
	*****************************************************************************************************************
**********************************	*/

*
	*****************************************************************************************************************
**********************************;
* On récupère les jumeaux de même sexes;
proc sql;

	CREATE TABLE jumeaux AS
		SELECT *
		FROM pop.T_INDI_BPCO_&an_N.
		GROUP BY BEN_NIR_PSA
		HAVING COUNT(DISTINCT BEN_IDT_ANO) > 1;

quit;

*
	*****************************************************************************************************************
**********************************;
* On les exclut de notre référentiel;
proc sql;

	CREATE INDEX BEN_IDT_ANO ON pop.T_INDI_BPCO_&an_N. (BEN_IDT_ANO);

	CREATE INDEX BEN_IDT_ANO ON jumeaux (BEN_IDT_ANO);

	ALTER TABLE pop.T_INDI_BPCO_&an_N. ADD exclus_jumeaux VARCHAR(1);

	UPDATE pop.T_INDI_BPCO_&an_N. SET
		exclus_jumeaux = CASE		WHEN BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM jumeaux) THEN "1"
                                                                ELSE "0"
                                                                END;

quit;

*
	*****************************************************************************************************************
**********************************;
* On supprime la table temporaire;

proc delete data = jumeaux;
run; quit;

*
	*****************************************************************************************************************
**********************************;
*	On crée une table de correspondance;

%suppr_table(
	lib = orauser,
	table = corresp_id_patient
	);
```

```
proc sort data = pop.T_INDI_BPCO_&an_N. out = orauser.corresp_id_patient (keep = BEN_IDT_ANO BEN_NIR_PSA) nodupkey;
	by BEN_IDT_ANO BEN_NIR_PSA;
run;

*
	**********************************************************************************************************
**********************************;
*	Vérifications;

*	Variable Exclus_Jumeaux ;
%proc_freq(
	in_tbl = pop.T_INDI_BPCO_&an_N.,
	out_tbl = _pop_Exclus_Jumeaux,
	list_var_in = Exclus_Jumeaux,
	list_var_out = Exclus_Jumeaux Frequency
	);
```

3.3.4 04_Exclusion_des_patients_sans_remboursement.sas

```
/*
	*************************************************************************************************************
**********************************	*/
/*

*/
/*		Sélection des patients - On supprime les patients n'ayant pas bénéficié de soin au cours de l'année (table
ER_PRS_F)	*/
/*

*/
/*
	*************************************************************************************************************
**********************************	*/

*
	*************************************************************************************************************
**********************************;
*	On récupère les patients avec au moins une ligne dans ER_PRS_F au cours de l année;

%macro soins_annee(annee_deb=, annee_fin=);

	%let annee_fin_flx = %sysevalf(&annee_fin. + 1);

	* On boucle sur les années de repérage (en flux);
	%do Annee = &annee_deb. %to &annee_fin_flx.;

		* On boucle sur les 12 mois de l année;
		%do i = 1 %to 12;

			%if &i. < 10 %then %let Mois = 0&i.;
			%else %let Mois = &i.;

			%put Données de &Mois./&Annee.;

			proc sql;

				%connectora;

					CREATE TABLE soins_&annee._&mois. AS SELECT * FROM CONNECTION TO
ORACLE (
						SELECT DISTINCT
							a.BEN_IDT_ANO,
							prs.EXE_SOI_DTD,
							prs.FLX_DIS_DTD
						FROM corresp_id_patient a
							INNER JOIN ER_PRS_F prs
								ON a.BEN_NIR_PSA =
prs.BEN_NIR_PSA
						WHERE prs.FLX_DIS_DTD =
TO_DATE(%str(%'&Annee.&Mois.01%'), 'YYYYMMDD')
							AND prs.EXE_SOI_DTD BETWEEN
TO_DATE(%str(%'&Annee_deb.0101%'), 'YYYYMMDD') AND TO_DATE(%str(%'&Annee_fin.1231%'), 'YYYYMMDD')
						);

				disconnect from oracle;

			quit;

			%arret_erreur;

			* On empile toutes les tables mensuelles;
			%if %sysfunc(exist(patients_soins_&annee_N.)) = 1 %then
				%do;

					proc append base = patients_soins_&annee_N. data = soins_&annee._&mois.;
					run;
```

```
                                        %arret_erreur;

                                        proc delete data = soins_&annee._&mois.;
                                        run; quit;

                                %end;

                        %if %sysfunc(exist(patients_soins_&annee_N.)) = 0 %then
                                %do;

                                        data patients_soins_&annee_N.;
                                                set soins_&annee._&mois.;
                                        run;

                                        %arret_erreur;

                                        proc delete data = soins_&annee._&mois.;
                                        run; quit;

                                %end;

                %end;
                * Fin de la boucle sur les 12 mois;

        %end;
        * Fin de la boucle par année;

        %arret_erreur;

%mend soins_annee;

%soins_annee(
        annee_deb = &annee_N.,
        annee_fin = &annee_N.
        );

*
        ************************************************************************************************************
***********************************;
*       Vérifications;

*       Variable Année de la Date de soins;

data verif_date_soins;
        set patients_soins_&annee_N.;
        annee_soins = year(datepart(EXE_SOI_DTD));
        annee_flux = year(datepart(FLX_DIS_DTD));
run;

%proc_freq(
        in_tbl = verif_date_soins,
        out_tbl = _annee_soins,
        list_var_in = annee_soins,
        list_var_out = annee_soins Frequency
        );

%proc_freq(
        in_tbl = verif_date_soins,
        out_tbl = _annee_flux,
        list_var_in = annee_flux,
        list_var_out = annee_flux Frequency
        );

proc delete data = verif_date_soins;
run; quit;

*
        ************************************************************************************************************
***********************************;
*       On exclut les patients qui n ont pas de soins dans l année;
```

```
* On les exclut de notre référentiel;

proc sort data = patients_soins_&annee_N. nodupkey;
  by BEN_IDT_ANO;
run;

proc sql;

		CREATE INDEX BEN_IDT_ANO ON patients_soins_&annee_N. (BEN_IDT_ANO);

		ALTER TABLE pop.T_INDI_BPCO_&an_N. ADD exclus_pas_remb VARCHAR(1);

		UPDATE pop.T_INDI_BPCO_&an_N. SET
			exclus_pas_remb = CASE		WHEN BEN_IDT_ANO NOT IN (SELECT BEN_IDT_ANO FROM
patients_soins_&annee_N.) THEN "1"
                                                                                ELSE "0"
                                                                                END;

quit;

*
		*************************************************************************************************
**********************************;
*		On supprime les tables temporaires;

proc delete data = patients_soins_&annee_N.;
run; quit;

*
		*************************************************************************************************
**********************************;
*		Vérifications;

*		Variable exclus_pas_remb ;
%proc_freq(
		in_tbl = pop.T_INDI_BPCO_&an_N.,
		out_tbl = _pop_exclus_pas_remb,
		list_var_in = exclus_pas_remb,
		list_var_out = exclus_pas_remb Frequency
		);
```

3.3.5 05_Exclusion_des_patients_decedes.sas

```
/*
        ****************************************************************************************************
***********************************            */
/*

                                                                                                    */
/*                  Sélection des patients - Exclusion de patients non vivant au 1er janvier de l'année
                                                                                            */
/*

                                                                                                    */
/*
        ****************************************************************************************************
***********************************            */

*
        ****************************************************************************************************
***********************************;
*       On récupère l information du décès dans IR_BEN_R;

data deces_IR_BEN_R;
        set pop.T_INDI_BPCO_&an_N. (keep = BEN_IDT_ANO BEN_DCD_DTE where = (BEN_DCD_DTE ne .));
run;

*
        ****************************************************************************************************
***********************************;
*       On récupère l information du décès dans le PMSI;

%macro deces_PMSI(annee_deb=, annee_fin=);

        %let an_deb_pmsi = %sysevalf(&annee_deb. - 2000);
        %let an_fin_pmsi = %sysevalf(&annee_fin. - 2000);

        %do an = &an_deb_pmsi. %to &an_fin_pmsi.;

                * Dans le PMSI - HAD;
                proc sql;

                        %connectora;

                                CREATE TABLE deces_PMSI_HAD_&an. AS SELECT * FROM CONNECTION TO ORACLE (
                                        SELECT DISTINCT
                                                a.BEN_IDT_ANO,
                                                c.EXE_SOI_DTD,
                                                c.EXE_SOI_DTF AS BEN_DCD_DTE
                                        FROM corresp_id_patient a
                                                INNER JOIN T_HAD&an.C c
                                                        ON a.BEN_NIR_PSA = c.NIR_ANO_17
                                                INNER JOIN T_HAD&an.B b
                                                        ON      c.ETA_NUM_EPMSI =
b.ETA_NUM_EPMSI
                                                        AND     c.RHAD_NUM = b.RHAD_NUM
                                        WHERE b.SOR_MOD = '9' AND c.NIR_ANO_17 NOT IN ('&cle_inc1.',
'&cle_inc2.')
                                                AND NIR_RET = '0' AND NAI_RET = '0' AND SEX_RET = '0'
AND SEJ_RET = '0' AND FHO_RET = '0'
                                                AND PMS_RET = '0' AND DAT_RET = '0' %if &an. > 12 %then
%do; AND COH_NAI_RET = '0'
                                                AND COH_SEX_RET = '0' %end;
                                        );

                        DISCONNECT FROM ORACLE;

                quit;

                data deces_PMSI_HAD_&an. (rename = (EXE_SOI_DTD2 = EXE_SOI_DTD BEN_DCD_DTE2 = BEN_DCD_DTE));
```

```
                        set deces_PMSI_HAD_&an.;
                        length EXE_SOI_DTD2 BEN_DCD_DTE2 4.;
                        EXE_SOI_DTD2 = datepart(EXE_SOI_DTD);
                        BEN_DCD_DTE2 = datepart(BEN_DCD_DTE);
                        format EXE_SOI_DTD2 BEN_DCD_DTE2 ddmmyy10.;
                        drop EXE_SOI_DTD BEN_DCD_DTE;
                run;

                * Dans le PMSI - MCO;
                proc sql;

                        %connectora;

                                CREATE TABLE deces_PMSI_MCO_&an. AS SELECT * FROM CONNECTION TO ORACLE (
                                        SELECT DISTINCT
                                                a.BEN_IDT_ANO,
                                                c.EXE_SOI_DTD,
                                                c.EXE_SOI_DTF AS BEN_DCD_DTE
                                        FROM corresp_id_patient a
                                                INNER JOIN T_MCO&an.C c
                                                        ON a.BEN_NIR_PSA = c.NIR_ANO_17
                                                INNER JOIN T_MCO&an.B b
                                                        ON      c.ETA_NUM = b.ETA_NUM
                                                        AND     c.RSA_NUM = b.RSA_NUM
                                        WHERE b.SOR_MOD = '9' AND c.NIR_ANO_17 NOT IN ('&cle_inc1.',
'&cle_inc2.')
                                                AND NIR_RET = '0' AND NAI_RET = '0' AND SEX_RET = '0'
AND SEJ_RET = '0' AND FHO_RET = '0'
                                                AND PMS_RET = '0' AND DAT_RET = '0' %if &an. > 12 %then
%do; AND COH_NAI_RET = '0'
                                                AND COH_SEX_RET = '0' %end;
                                                AND c.ETA_NUM NOT IN ('130780521', '130783236',
'130783293', '130784234', '130804297', '600100101',
                                                        '750041543', '750100018', '750100042',
'750100075', '750100083', '750100091', '750100109',
                                                        '750100125', '750100166', '750100208',
'750100216', '750100232', '750100273', '750100299',
                                                        '750801441', '750803447', '750803454',
'910100015', '910100023', '920100013', '920100021',
                                                        '920100039', '920100047', '920100054',
'920100062', '930100011', '930100037', '930100045',
                                                        '940100027', '940100035', '940100043',
'940100050', '940100068', '950100016', '690783154',
                                                        '690784137', '690784152', '690784178',
'690787478', '830100558')
                                                AND GRG_GHM NOT IN ('90Z00Z') AND GRG_RET NOT IN
('024') AND GRG_GHM NOT IN ('90H01Z', '90Z00Z', '90Z01Z',
                                                        '90Z02Z', '90Z03Z')
                                                AND ((SEJ_TYP = 'A' OR SEJ_TYP IS NULL) OR (SEJ_TYP = 'B'
AND GRG_GHM NOT IN ('28Z14Z','28Z15Z','28Z16Z')))
                                        );

                        DISCONNECT FROM ORACLE;

                quit;

                data deces_PMSI_MCO_&an. (rename = (EXE_SOI_DTD2 = EXE_SOI_DTD BEN_DCD_DTE2 = BEN_DCD_DTE));
                        set deces_PMSI_MCO_&an.;
                        length EXE_SOI_DTD2 BEN_DCD_DTE2 4.;
                        EXE_SOI_DTD2 = datepart(EXE_SOI_DTD);
                        BEN_DCD_DTE2 = datepart(BEN_DCD_DTE);
                        format EXE_SOI_DTD2 BEN_DCD_DTE2 ddmmyy10.;
                        drop EXE_SOI_DTD BEN_DCD_DTE;
                run;

                * Dans le PMSI - RIP;
                proc sql;

                        %connectora;

                                CREATE TABLE deces_PMSI_RIP_&an. AS SELECT * FROM CONNECTION TO ORACLE (
```

```
                                                SELECT DISTINCT
                                                        a.BEN_IDT_ANO,
                                                        c.EXE_SOI_DTD,
                                                        c.EXE_SOI_DTF AS BEN_DCD_DTE
                                                FROM corresp_id_patient a
                                                        INNER JOIN T_RIP&an.C c
                                                                ON a.BEN_NIR_PSA = c.NIR_ANO_17
                                                        INNER JOIN T_RIP&an.RSA rsa
                                                                ON      c.ETA_NUM_EPMSI =
rsa.ETA_NUM_EPMSI
                                                                AND     c.RIP_NUM = rsa.RIP_NUM
                                                WHERE rsa.SOR_MOD = '9' AND c.NIR_ANO_17 NOT IN ('&cle_inc1.',
'&cle_inc2.')
                                                        AND NIR_RET = '0' AND NAI_RET = '0' AND SEX_RET = '0'
AND SEJ_RET = '0' AND FHO_RET = '0'
                                                        AND PMS_RET = '0' AND DAT_RET = '0' %if &an. > 12 %then
%do; AND COH_NAI_RET = '0'
                                                        AND COH_SEX_RET = '0' %end;
                                                );

                        DISCONNECT FROM ORACLE;

        quit;

        data deces_PMSI_RIP_&an. (rename = (EXE_SOI_DTD2 = EXE_SOI_DTD BEN_DCD_DTE2 = BEN_DCD_DTE));
                set deces_PMSI_RIP_&an.;
                length EXE_SOI_DTD2 BEN_DCD_DTE2 4.;
                EXE_SOI_DTD2 = datepart(EXE_SOI_DTD);
                BEN_DCD_DTE2 = datepart(BEN_DCD_DTE);
                format EXE_SOI_DTD2 BEN_DCD_DTE2 ddmmyy10.;
                drop EXE_SOI_DTD BEN_DCD_DTE;
        run;

        * Dans le PMSI - SSR;
        proc sql;

                %connectora;

                        CREATE TABLE deces_PMSI_SSR_&an. AS SELECT * FROM CONNECTION TO ORACLE (
                                                SELECT DISTINCT
                                                        a.BEN_IDT_ANO,
                                                        c.EXE_SOI_DTD,
                                                        c.EXE_SOI_DTF AS BEN_DCD_DTE
                                                FROM corresp_id_patient a
                                                        INNER JOIN T_SSR&an.C c
                                                                ON a.BEN_NIR_PSA = c.NIR_ANO_17
                                                        INNER JOIN T_SSR&an.B b
                                                                ON      c.ETA_NUM = b.ETA_NUM
                                                                AND     c.RHA_NUM = b.RHA_NUM
                                                WHERE b.SOR_MOD = '9' AND c.NIR_ANO_17 NOT IN ('&cle_inc1.',
'&cle_inc2.')
                                                        AND NIR_RET = '0' AND NAI_RET = '0' AND SEX_RET = '0'
AND SEJ_RET = '0' AND FHO_RET = '0'
                                                        AND PMS_RET = '0' AND DAT_RET = '0' %if &an. > 12 %then
%do; AND COH_NAI_RET = '0'
                                                        AND COH_SEX_RET = '0' %end;
                                                );

                        DISCONNECT FROM ORACLE;

        quit;

        data deces_PMSI_SSR_&an. (rename = (EXE_SOI_DTD2 = EXE_SOI_DTD BEN_DCD_DTE2 = BEN_DCD_DTE));
                set deces_PMSI_SSR_&an.;
                length EXE_SOI_DTD2 BEN_DCD_DTE2 4.;
                EXE_SOI_DTD2 = datepart(EXE_SOI_DTD);
                BEN_DCD_DTE2 = datepart(BEN_DCD_DTE);
                format EXE_SOI_DTD2 BEN_DCD_DTE2 ddmmyy10.;
                drop EXE_SOI_DTD BEN_DCD_DTE;
        run;
```

```
	%end;

	* On concatène toutes les tables;
	data deces_PMSI;
		set deces_PMSI_:;
	run;

	* On supprime les tables temporaires;
	proc datasets library = work memtype = data nolist;
		delete deces_PMSI_:;
	run; quit;

%mend deces_PMSI;

%deces_PMSI(
	annee_deb = &annee_2N.,
	annee_fin = &annee_N1.
	);

*
	**************************************************************************************************************
**********************************;
*	On récupère l information du décès dans le DCIR;

%macro deces_DCIR(annee_deb=, annee_fin=);

	%let annee_fin_flx = %sysevalf(&annee_fin. + 1);

	* On boucle sur les années de repérage (en flux);
	%do annee = &annee_deb. %to &annee_fin_flx.;

		* On boucle sur les 12 mois de l année;
		%do i = 1 %to 12;

			%if &i. < 10 %then %let Mois = 0&i.;
			%else %let Mois = &i.;

			%put Données de &Mois./&Annee.;

			proc sql;

				%connectora;

					CREATE TABLE deces_DCIR_&annee._&mois. AS SELECT * FROM CONNECTION
TO ORACLE (
						SELECT DISTINCT
							a.BEN_IDT_ANO,
							prs.EXE_SOI_DTD,
							prs.BEN_DCD_DTE
						FROM corresp_id_patient a
							INNER JOIN ER_PRS_F prs
							ON a.BEN_NIR_PSA = prs.BEN_NIR_PSA
						WHERE BEN_DCD_DTE > TO_DATE('16000101',
'YYYYMMDD')
							AND prs.FLX_DIS_DTD =
TO_DATE(%str(%'&Annee.&Mois.01%'), 'YYYYMMDD')
						);

				disconnect from oracle;

			quit;

			data deces_DCIR_&annee._&mois. (rename = (EXE_SOI_DTD2 = EXE_SOI_DTD BEN_DCD_DTE2 =
BEN_DCD_DTE));
				set deces_DCIR_&annee._&mois.;
				length EXE_SOI_DTD2 BEN_DCD_DTE2 4.;
				EXE_SOI_DTD2 = datepart(EXE_SOI_DTD);
				BEN_DCD_DTE2 = datepart(BEN_DCD_DTE);
				format EXE_SOI_DTD2 BEN_DCD_DTE2 ddmmyy10.;
				drop EXE_SOI_DTD BEN_DCD_DTE;
			run;
```

```
                                %arret_erreur;

                                * On empile toutes les tables mensuelles;
                                %if %sysfunc(exist(deces_DCIR)) = 1 %then
                                        %do;

                                                proc append base = deces_DCIR data = deces_DCIR_&annee._&mois.;
                                                run;

                                                %arret_erreur;

                                                proc delete data = deces_DCIR_&annee._&mois.;
                                                run; quit;

                                        %end;

                                %if %sysfunc(exist(deces_DCIR)) = 0 %then
                                        %do;

                                                data deces_DCIR;
                                                        set deces_DCIR_&annee._&mois.;
                                                run;

                                                %arret_erreur;

                                                proc delete data = deces_DCIR_&annee._&mois.;
                                                run; quit;

                                        %end;

                        %end;
                        * Fin de la boucle sur les 12 mois;

                %end;
                * Fin de la boucle par année;

                %arret_erreur;

%mend deces_DCIR;

%deces_DCIR(
                annee_deb = &annee_2N.,
                annee_fin = &annee_N.
                );

*
                ****************************************************************************************************
**********************************;
*               On concatène toutes les données et on conserve l information de manière unique;

data travail.histo_deces;
                set deces_IR_BEN_R deces_PMSI deces_DCIR;
run;

proc delete data = deces_IR_BEN_R deces_PMSI deces_DCIR;
run; quit;

proc sort data = travail.histo_deces nodupkey;
                by _all_;
run;

*
                ****************************************************************************************************
**********************************;
*               On récupère la date de décès minimale;

proc sql;

                        CREATE TABLE date_deces AS
                                SELECT
```

```
                                        BEN_IDT_ANO,
                                        MIN(BEN_DCD_DTE) AS BEN_DCD_DTE length = 4 format date9.
                                FROM travail.histo_deces
                                GROUP BY BEN_IDT_ANO;

quit;

*
        ********************************************************************************************************
************************************;
*       On ajoute l info du décès dans la table de patients;

proc sort data = pop.T_INDI_BPCO_&an_N. force;
        by BEN_IDT_ANO;
run;

proc sort data = date_deces force;
        by BEN_IDT_ANO;
run;

data pop.T_INDI_BPCO_&an_N.;
        merge   pop.T_INDI_BPCO_&an_N. (in = a drop = BEN_DCD_DTE)
                date_deces (in = b);
        by BEN_IDT_ANO;
        if a;
        format BEN_DCD_DTE date9.;
run;

*
        ********************************************************************************************************
************************************;
*       On exclut les patients qui sont décédés avant l année;

proc sql;

        ALTER TABLE pop.T_INDI_BPCO_&an_N. ADD exclus_deces VARCHAR(1);

        UPDATE pop.T_INDI_BPCO_&an_N. SET
                exclus_deces = CASE  WHEN YEAR(BEN_DCD_DTE) < &annee_N. AND BEN_DCD_DTE IS NOT NULL THEN "1"
                                                                        ELSE "0"
                                                                        END;

quit;

*
        ********************************************************************************************************
************************************;
*       On crée une table de correspondance pour les patients qu on a conservés;

* On supprime la table si elle existe;
%suppr_table(
        lib = orauser,
        table = corresp_id_patient
        );

data corresp_id_patient;
        set pop.T_INDI_BPCO_&an_N. (where = (exclus_jumeaux = "0" and exclus_deces = "0" and exclus_pas_40ans = "0" and
exclus_pas_remb = "0"
                and exclus_NIR_fictif = "0"));
run;

proc sort data = corresp_id_patient out = orauser.corresp_id_patient (keep = BEN_IDT_ANO BEN_NIR_PSA) nodupkey;
        by BEN_IDT_ANO BEN_NIR_PSA;
run;

*
        ********************************************************************************************************
************************************;
*       On supprime les tables temporaires;

proc datasets library = work memtype = data nolist;
```

```
		delete date_deces corresp_id_patient;
run; quit;

*
		*****************************************************************************************************
************************************;
*		Vérifications;

*		Variable exclus_pas_remb ;
%proc_freq(
		in_tbl = pop.T_INDI_BPCO_&an_N.,
		out_tbl = _pop_exclus_DC,
		list_var_in = exclus_deces,
		list_var_out = exclus_deces Frequency
		);

*		Variable année de BEN_DCD_DTE ;

data annee_deces;
		set pop.T_INDI_BPCO_&an_N. (keep = BEN_IDT_ANO BEN_DCD_DTE where = (BEN_DCD_DTE ne .));
		annee_deces = year(BEN_DCD_DTE);
run;

%proc_freq(
		in_tbl = annee_deces,
		out_tbl = _pop_annee_DC,
		list_var_in = annee_deces,
		list_var_out = annee_deces Frequency
		);

proc delete data = annee_deces;
run; quit;
```

3.3.6 06_Hospit_BPCO.sas

```
/*
	*************************************************************************************************************
**********************************	*/
/*

*/
/*		Sélection des patients - Hospitalisation pour BPCO en MCO, SSR ou HAD dans l'année
*/
/*

*/
/*
	*************************************************************************************************************
**********************************	*/

*
	*************************************************************************************************************
**********************************;
*	On récupère les séjours pour BPCO en MCO, SSR et HAD;

%suppr_table(
	lib = orauser,
	table = hospit_CIM_BPCO
	);

data orauser.hospit_CIM_BPCO;
	set orauser.diag_codes_CIM_BPCO (where = (reperage = 16));
run;

* On appelle la macro pour le repérage dans le PMSI MCO, HAD et SSR;
%suppr_table(
	lib = travail,
	table = reperage_hospit_BPCO_&annee_4N._&annee_N1.
	);

%extract_CIM10_PMSI(
	annee_deb = &annee_4N.,
	annee_fin = &annee_N1.,
	HAD_DP = 1,
	HAD_DAS = 1,
	HAD_MPP = 1,
	HAD_MPA = 1,
	MCO_DP = 1,
	MCO_DR = 1,
	MCO_DAS = 1,
	MCO_DP_UM = 1,
	MCO_DR_UM = 1,
	SSR_FP = 1,
	SSR_MPP = 1,
	SSR_AE = 1,
	SSR_DAS = 1,
	tbl_out = travail.reperage_hospit_BPCO_&annee_4N._&annee_N1.,
	tbl_codes = hospit_CIM_BPCO,
	tbl_patients = corresp_id_patient
	);

data travail.reperage_hospit_BPCO_&annee_4N._&annee_N1.;
	set travail.reperage_hospit_BPCO_&annee_4N._&annee_N1.;
	format id_sejour $30.;
	if domaine = "HAD" then
		id_sejour = ETA_NUM_EPMSI||"_"||RHAD_NUM||"_"||put(annee, 4.);
	if domaine = "MCO" then
		id_sejour = ETA_NUM||"_"||RSA_NUM||"_"||put(annee, 4.);
	if domaine = "SSR" then
		id_sejour = ETA_NUM||"_"||RHA_NUM||"_"||put(annee, 4.);
	if domaine ne "MCO" and date_fin = . then
		date_fin = mdy(12, 31, annee);
```

```
run;

*
	**********************************************************************************************************
**********************************;
*	Vérifications;

data verifs;
	set travail.reperage_hospit_BPCO_&annee_4N._&annee_N1. (keep = id_sejour annee date_debut date_fin CODE_CIM
domaine table variable type);
	annee_debut = year(date_debut);
	annee_fin = year(date_fin);
run;

%proc_freq(
	in_tbl = verifs,
	out_tbl = _hospit_BPCO_debut,
	list_var_in = annee_debut,
	list_var_out = annee_debut Frequency
	);

%proc_freq(
	in_tbl = verifs,
	out_tbl = _hospit_BPCO_fin,
	list_var_in = annee_fin,
	list_var_out = annee_fin Frequency
	);

%proc_freq(
	in_tbl = verifs,
	out_tbl = _hospit_BPCO_CIM,
	list_var_in = CODE_CIM,
	list_var_out = CODE_CIM Frequency
	);

%proc_freq(
	in_tbl = verifs,
	out_tbl = _hospit_BPCO,
	list_var_in = Annee*Domaine*Table*Variable*Type,
	list_var_out = Annee Domaine Table2 Variable Type Frequency
	);

proc delete data = verifs;
run; quit;

*
	**********************************************************************************************************
**********************************;
*	On compte le nombre de séjours dans l année;

proc sql undo_policy = none;

	CREATE TABLE nb_hospit_BPCO_&annee_N. AS
		SELECT
			BEN_IDT_ANO,
			COUNT(DISTINCT id_sejour) AS Nb_hospit_BPCO length = 3
		FROM travail.reperage_hospit_BPCO_&annee_4N._&annee_N1.
		WHERE annee = &annee_N.
		GROUP BY BEN_IDT_ANO;

quit;

*
	**********************************************************************************************************
**********************************;
*	On ajoute l information du nombre d hospitalisations dans l année dans la table de population;

data pop.T_INDI_BPCO_&an_N.;
	merge	pop.T_INDI_BPCO_&an_N. (in = a)
			nb_hospit_BPCO_&annee_N. (in = b);
	by BEN_IDT_ANO;
```

```
		if a;
		if not b then
			Nb_hospit_BPCO = 0;
run;

*
		*************************************************************************************************************
************************************;
*		On supprime les tables temporaires;

proc delete data = nb_hospit_BPCO_&annee_N.;
run; quit;

proc delete data = orauser.hospit_CIM_BPCO;
run; quit;

*
		*************************************************************************************************************
************************************;
*		On récupère les patients hospitalisés pour BPCO en MCO, SSR et HAD entre le 1er janvier de l année N-1 et le 31 août de l
année N;

*		On compte le nombre de séjours entre l année N-1 et l année N;
proc sql undo_policy = none;

		CREATE TABLE nb_hospit_BPCO_ind02 AS
			SELECT
				BEN_IDT_ANO,
				COUNT(DISTINCT id_sejour) AS Nb_hospit_BPCO_Ind02 length = 3
			FROM travail.reperage_hospit_BPCO_&annee_4N._&annee_N1.
			WHERE (domaine = "MCO" AND "01JAN&Annee_1N."d <= date_fin <= "31AUG&Annee_N."d) OR (domaine IN
("HAD", "SSR") AND
				(date_debut <= "31AUG&Annee_N."d AND "01JAN&Annee_1N."d <= date_fin))
			GROUP BY BEN_IDT_ANO;

quit;

*		On ajoute l information du nombre d hospitalisations dans l année dans la table de population;
data pop.T_INDI_BPCO_&an_N.;
		merge	pop.T_INDI_BPCO_&an_N. (in = a)
				nb_hospit_BPCO_ind02 (in = b);
		by BEN_IDT_ANO;
		if a;
		if not b then
			Nb_hospit_BPCO_Ind02 = 0;
run;

*
		*************************************************************************************************************
************************************;
*		On supprime les tables temporaires;

proc delete data = nb_hospit_BPCO_ind02;
run; quit;
```

3.3.7 07_Hospit_exacerbation_de_BPCO.sas

```
/*
	****************************************************************************************************
************************************			*/
/*
																			*/
/*			Sélection des patients - Nombre de séjours pour exacerbation dans l année
																			*/
/*
																			*/
/*			Séjours pour exacerbation de BPCO = Séjour hospitalier MCO terminé dans l année pour lequel :
																*/
/*				- Un code diagnostic de BPCO a été codé en DP du séjour
				*/
/*				- OU un code de pneumopathie lobaire a été codé en DP du séjour avec un diagnostic de BPCO codé en
DAS													*/
/*				- OU un code d insuffisance respiratoire aigüe a été codé en DP du séjour avec un diagnostic de BPCO
codé en DAS										*/
/*				- OU un code de grippe a été codé en DP du séjour avec un diagnostic de BPCO codé en DAS
																	*/
/*				- OU un code d infection des voies aériennes a été codé en DP du séjour avec un diagnostic de BPCO
codé en DAS											*/
/*				- OU un code de bronchopneumopathie a été codé en DP du séjour avec un diagnostic de BPCO codé
en DAS												*/
/*				- OU un code de pneumonie a été codé en DP du séjour avec un diagnostic de BPCO codé en DAS
																*/
/*				- OU un code d embolie pulmonaire a été codé en DP du séjour avec un diagnostic de BPCO codé en
DAS													*/
/*				- OU un code d insuffisance cardiaque aigue a été codé en DP du séjour avec un diagnostic de BPCO
codé en DAS											*/
/*				- OU un code d oedème aigu du poumon a été codé en DP du séjour avec un diagnostic de BPCO codé
en DAS												*/
/*				- OU un code de pneumothorax a été codé en DP du séjour avec un diagnostic de BPCO codé en DAS
																	*/
/*
																			*/
/*
	****************************************************************************************************
************************************			*/

*
	****************************************************************************************************
************************************;
*	On récupère les séjours pour exacerbation de BPCO;

%suppr_table(
	lib = orauser,
	table = diag_codes_CIM_exa_BPCO
	);

data orauser.diag_codes_CIM_exa_BPCO;
	set orauser.diag_codes_CIM_BPCO (where = (reperage in (16, 18, 17, 19, 23, 25, 24, 20, 21, 22) or (reperage = 11 and
flag_pneumo = 1)));
run;

* On appelle la macro pour le repérage dans le PMSI MCO;
%suppr_table(
	lib = travail,
	table = reperage_CIM10_exa_BPCO
	);

%extract_CIM10_PMSI(
	annee_deb = &annee_N.,
	annee_fin = &annee_N.,
```

```
		HAD_DP = 0,
		HAD_DAS = 0,
		HAD_MPP = 0,
		HAD_MPA = 0,
		MCO_DP = 1,
		MCO_DR = 1,
		MCO_DAS = 1,
		MCO_DP_UM = 1,
		MCO_DR_UM = 1,
		SSR_FP = 0,
		SSR_MPP = 0,
		SSR_AE = 0,
		SSR_DAS = 0,
		tbl_out = travail.reperage_CIM10_exa_BPCO,
		tbl_codes = diag_codes_CIM_exa_BPCO,
		tbl_patients = corresp_id_patient
		);

proc sql;

		DELETE FROM travail.reperage_CIM10_exa_BPCO
		WHERE SUBSTR(GRG_GHM, 1, 2) = "28" OR SEJ_NBJ = 0;

quit;

proc sort data = travail.reperage_CIM10_exa_BPCO;
		by ETA_NUM RSA_NUM annee;
run;

*
		*************************************************************************************************************
************************************;
*		On récupère les séjours avec un code diagnostic de BPCO a été codé en DP du séjour;

data sejours_exacerbation1;
		set travail.reperage_CIM10_exa_BPCO (where = (reperage = 16 and table = "B" and variable = "DGN_PAL"));
		length exacerbation_BPCO 3.;
		exacerbation_BPCO = 1;
run;

*
		*************************************************************************************************************
************************************;
*		On récupère les séjours avec un code diagnostic de BPCO a été codé en DAS du séjour;

proc sort data = travail.reperage_CIM10_exa_BPCO (where = (reperage = 16 and variable = "ASS_DGN")) out =
reperage_CIM10_exa_BPCO
			nodupkey;
		by ETA_NUM RSA_NUM annee;
run;

*
		*************************************************************************************************************
************************************;
*		On récupère les séjours avec un code de pneumopathie lobaire en DP et un diagnostic de BPCO en DAS;

data sejours_exacerbation2;
		merge	travail.reperage_CIM10_exa_BPCO (in = a where = (reperage = 18 and table = "B" and variable = "DGN_PAL"))
				reperage_CIM10_exa_BPCO (in = b);
		by ETA_NUM RSA_NUM annee;
		if a and b;
		length exacerbation_BPCO 3.;
		exacerbation_BPCO = 2;
run;

*
		*************************************************************************************************************
************************************;
*		On récupère les séjours avec un code d insuffisance respiratoire aigüe en DP avec un diagnostic de BPCO en DAS;

data sejours_exacerbation3;
```

```
	merge	travail.reperage_CIM10_exa_BPCO (in = a where = (reperage = 17 and table = "B" and variable = "DGN_PAL"))
			reperage_CIM10_exa_BPCO (in = b);
	by ETA_NUM RSA_NUM annee;
	if a and b;
	length exacerbation_BPCO 3.;
	exacerbation_BPCO = 3;
run;

*
	*************************************************************************************************************
***********************************;
*	On récupère les séjours avec un code de grippe en DP avec un diagnostic de BPCO en DAS;

data sejours_exacerbation4;
	merge	travail.reperage_CIM10_exa_BPCO (in = a where = (reperage = 19 and table = "B" and variable = "DGN_PAL"))
			reperage_CIM10_exa_BPCO (in = b);
	by ETA_NUM RSA_NUM annee;
	if a and b;
	length exacerbation_BPCO 3.;
	exacerbation_BPCO = 4;
run;

*
	*************************************************************************************************************
***********************************;
*	On récupère les séjours avec un code d infection des voies aériennes en DP avec un diagnostic de BPCO en DAS;

data sejours_exacerbation5;
	merge	travail.reperage_CIM10_exa_BPCO (in = a where = (reperage = 23 and table = "B" and variable = "DGN_PAL"))
			reperage_CIM10_exa_BPCO (in = b);
	by ETA_NUM RSA_NUM annee;
	if a and b;
	length exacerbation_BPCO 3.;
	exacerbation_BPCO = 5;
run;

*
	*************************************************************************************************************
***********************************;
*	On récupère les séjours avec un code de bronchopneumopathie en DP avec un diagnostic de BPCO en DAS;

data sejours_exacerbation6;
	merge	travail.reperage_CIM10_exa_BPCO (in = a where = (reperage = 25 and table = "B" and variable = "DGN_PAL"))
			reperage_CIM10_exa_BPCO (in = b);
	by ETA_NUM RSA_NUM annee;
	if a and b;
	length exacerbation_BPCO 3.;
	exacerbation_BPCO = 6;
run;

*
	*************************************************************************************************************
***********************************;
*	On récupère les séjours avec un code de pneumonie en DP avec un diagnostic de BPCO en DAS;

data sejours_exacerbation7;
	merge	travail.reperage_CIM10_exa_BPCO (in = a where = (reperage = 24 and table = "B" and variable = "DGN_PAL"))
			reperage_CIM10_exa_BPCO (in = b);
	by ETA_NUM RSA_NUM annee;
	if a and b;
	length exacerbation_BPCO 3.;
	exacerbation_BPCO = 7;
run;

*
	*************************************************************************************************************
***********************************;
*	On récupère les séjours avec un code d embolie pulmonaire en DP avec un diagnostic de BPCO en DAS;

data sejours_exacerbation8;
	merge	travail.reperage_CIM10_exa_BPCO (in = a where = (reperage = 20 and table = "B" and variable = "DGN_PAL"))
```

```
                                    reperage_CIM10_exa_BPCO (in = b);
            by ETA_NUM RSA_NUM annee;
            if a and b;
            length exacerbation_BPCO 3.;
            exacerbation_BPCO = 8;
run;

*
            ************************************************************************************************************
************************************;
*           On récupère les séjours avec un code d insuffisance cardiaque aigue en DP avec un diagnostic de BPCO en DAS;

data sejours_exacerbation9;
            merge       travail.reperage_CIM10_exa_BPCO (in = a where = (reperage = 21 and table = "B" and variable = "DGN_PAL"))
                                    reperage_CIM10_exa_BPCO (in = b);
            by ETA_NUM RSA_NUM annee;
            if a and b;
            length exacerbation_BPCO 3.;
            exacerbation_BPCO = 9;
run;

*
            ************************************************************************************************************
************************************;
*           On récupère les séjours avec un code d œdème aigu du poumon en DP avec un diagnostic de BPCO en DAS;

data sejours_exacerbation10;
            merge       travail.reperage_CIM10_exa_BPCO (in = a where = (reperage = 22 and table = "B" and variable = "DGN_PAL"))
                                    reperage_CIM10_exa_BPCO (in = b);
            by ETA_NUM RSA_NUM annee;
            if a and b;
            length exacerbation_BPCO 3.;
            exacerbation_BPCO = 10;
run;

*
            ************************************************************************************************************
************************************;
*           On récupère les séjours avec un code de pneumothorax en DP avec un diagnostic de BPCO en DAS;

data sejours_exacerbation11;
            merge       travail.reperage_CIM10_exa_BPCO (in = a where = (reperage = 11 and table = "B" and variable = "DGN_PAL"))
                                    reperage_CIM10_exa_BPCO (in = b);
            by ETA_NUM RSA_NUM annee;
            if a and b;
            length exacerbation_BPCO 3.;
            exacerbation_BPCO = 11;
run;

*
            ************************************************************************************************************
************************************;
*           On concatène toutes les tables et on compte le nombre de séjours par patient;

data travail.sejours_exacerbation_&Annee_N.;
            set sejours_exacerbation1-sejours_exacerbation11;
            length id_sejour $ 30;
            id_sejour = ETA_NUM||RSA_NUM||put(Annee, 4.);
run;

*
            ************************************************************************************************************
************************************;
*           Vérifications;

%let tmp_num_tab = 15;

%proc_freq(
            in_tbl = travail.sejours_exacerbation_&Annee_N.,
            out_tbl = _exa_BPCO,
            list_var_in = Annee*Domaine*Table*Variable*DGN_PAL*CODE_CIM,
```

```
		list_var_out = Annee Domaine Table2 Variable DGN_PAL CODE_CIM Frequency
		);

proc sql undo_policy = none;

		CREATE TABLE nb_sejours_exacerbation AS
			SELECT
				BEN_IDT_ANO,
				COUNT(DISTINCT id_sejour) AS Nb_Sej_Exacerbation length = 3
			FROM travail.sejours_exacerbation_&Annee_N.
			GROUP BY BEN_IDT_ANO;

quit;

*
		**********************************************************************************************************
**********************************;
*		On ajoute l information du nombre de séjours pour exacerbation dans l année dans la table de population;

data pop.T_INDI_BPCO_&an_N.;
		merge	pop.T_INDI_BPCO_&an_N. (in = a)
				nb_sejours_exacerbation (in = b);
		by BEN_IDT_ANO;
		if a;
		if not b then
			Nb_Sej_Exacerbation = 0;
run;

*
		**********************************************************************************************************
**********************************;
*		Vérifications;

%proc_freq(
		in_tbl = pop.T_INDI_BPCO_&an_N.,
		out_tbl = _exa_BPCO_Nb_Sej,
		list_var_in = Nb_Sej_Exacerbation,
		list_var_out = Nb_Sej_Exacerbation Frequency
		);

*
		**********************************************************************************************************
**********************************;
*		On supprime les tables temporaires;

proc datasets library = work memtype = data nolist;
		delete sejours_exacerbation: nb_sejours_exacerbation reperage_CIM10_exa_BPCO;
run; quit;

proc delete data = orauser.diag_codes_CIM_exa_BPCO;
run; quit;

proc delete data = travail.reperage_CIM10_exa_BPCO travail.sejours_exacerbation_&Annee_N.;
run; quit;
```

3.3.8 08_ALD_BPCO.sas

```
/*
	**********************************************************************************************************
**********************************	*/
/*

*/
/*		Sélection des patients - ALD BPCO active dans l année

		*/
/*

*/
/*
	**********************************************************************************************************
**********************************	*/

*
	**********************************************************************************************************
**********************************;
*	On récupère les patients avec une ALD BPCO (J44*) active dans l année;

proc sql undo_policy = none;

	%connectora;

		CREATE TABLE ALD_BPCO_&annee_N. AS
			SELECT DISTINCT * FROM CONNECTION TO ORACLE (
				SELECT DISTINCT
					a.BEN_IDT_ANO,
					b.IMB_ETM_NAT,
					b.IMB_ALD_DTD,
					b.IMB_ALD_DTF,
					b.INS_DTE,
					b.UPD_DTE,
					b.IMB_ALD_NUM,
					b.MED_MTF_COD,
					1 AS ALD_BPCO
				FROM corresp_id_patient a
					INNER JOIN IR_IMB_R b
						ON a.BEN_NIR_PSA = b.BEN_NIR_PSA
					INNER JOIN IR_CIM_V c
						ON SUBSTR(b.MED_MTF_COD, 1, 3) = SUBSTR(c.CIM_COD, 1, 3)
				WHERE IMB_ETM_NAT IN (41, 43, 45) AND SUBSTR(TRIM(b.MED_MTF_COD), 1, 3) = 'J44'
					AND b.INS_DTE < TO_DATE(%str(%'&Annee_N1.0501%'), 'YYYYMMDD')
				);

	disconnect from oracle;

quit;

proc sort data = ALD_BPCO_&annee_N.;
		by BEN_IDT_ANO IMB_ETM_NAT IMB_ALD_NUM MED_MTF_COD INS_DTE UPD_DTE IMB_ALD_DTF descending
IMB_ALD_DTD;
run;

data ALD_BPCO_&annee_N.;
	set ALD_BPCO_&annee_N.;
	by BEN_IDT_ANO IMB_ETM_NAT IMB_ALD_NUM MED_MTF_COD INS_DTE UPD_DTE IMB_ALD_DTF descending
IMB_ALD_DTD;
	if last.MED_MTF_COD then
		output;
run;

data ALD_BPCO_&annee_N. (rename = (IMB_ALD_DTD2 = IMB_ALD_DTD IMB_ALD_DTF2 = IMB_ALD_DTF));
	set ALD_BPCO_&annee_N.;
	length IMB_ALD_DTD2 IMB_ALD_DTF2 4. ALD_BPCO_&an_N. 3.;
	IMB_ALD_DTD2 = datepart(IMB_ALD_DTD);
```

```
            IMB_ALD_DTF2 = datepart(IMB_ALD_DTF);
            ALD_BPCO_&an_N. = ALD_BPCO;
            format IMB_ALD_DTD2 IMB_ALD_DTF2 ddmmyy10.;
            drop IMB_ALD_DTD IMB_ALD_DTF ALD_BPCO;
            if IMB_ALD_DTD2 <= "31DEC&Annee_N."d and (IMB_ALD_DTF2 = "01JAN1600"d or IMB_ALD_DTF2 >= "01JAN&Annee_N."d)
                        then output;
run;

*
            ************************************************************************************************************
***********************************;
*           On ajoute l information de l ALD BPCO dans la table de population;

proc sql;

            ALTER TABLE pop.T_INDI_BPCO_&an_N. ADD ALD_BPCO_&an_N. INT length = 3;

            UPDATE pop.T_INDI_BPCO_&an_N. a
                        SET ALD_BPCO_&an_N. = CASE   WHEN BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM ALD_BPCO_&annee_N.)
THEN 1
                                                                                                ELSE 0
                                                                                                END;

quit;

*
            ************************************************************************************************************
***********************************;
*           Patients avec une ALD BPCO (J44*) active au 1er septembre;

data ALD_BPCO_septembre;
            set ALD_BPCO_&annee_N.;
            where IMB_ALD_DTD <= "01SEP&annee_N."d <= IMB_ALD_DTF or IMB_ALD_DTD <= "01SEP&annee_N."d and
IMB_ALD_DTF = "01JAN1600"d;
run;


proc sql;

            ALTER TABLE pop.T_INDI_BPCO_&an_N. ADD ALD_BPCO_septembre INT length = 3;

            UPDATE pop.T_INDI_BPCO_&an_N. a
                        SET ALD_BPCO_septembre = CASE            WHEN BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM
ALD_BPCO_septembre) THEN 1
                                                                                                ELSE 0
                                                                                                END;

quit;

*
            ************************************************************************************************************
***********************************;
*           Patients avec une ALD BPCO (J44*) active au 31 décembre;

data ALD_BPCO_decembre;
            set ALD_BPCO_&annee_N.;
            where (IMB_ALD_DTD <= "31DEC&annee_N."d <= IMB_ALD_DTF) or (IMB_ALD_DTD <= "31DEC&annee_N."d and
IMB_ALD_DTF = "01JAN1600"d);
run;


proc sql;

            ALTER TABLE pop.T_INDI_BPCO_&an_N. ADD ALD_BPCO_decembre INT length = 3;

            UPDATE pop.T_INDI_BPCO_&an_N. a
                        SET ALD_BPCO_decembre = CASE            WHEN BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM
ALD_BPCO_decembre) THEN 1
                                                                                                ELSE 0
                                                                                                END;
```

```
quit;

*
	**********************************************************************************************************
**********************************;
*	Vérifications;

%proc_freq(
	in_tbl = pop.T_INDI_BPCO_&an_N.,
	out_tbl = _ALD_BPCO,
	list_var_in = ALD_BPCO_&an_N.*ALD_BPCO_septembre*ALD_BPCO_decembre,
	list_var_out = ALD_BPCO_&an_N. ALD_BPCO_septembre ALD_BPCO_decembre Frequency
	);

*
	**********************************************************************************************************
**********************************;
*	On supprime les tables temporaires;

proc datasets library = work memtype = data nolist;
	delete ALD_BPCO_&annee_N. ALD_BPCO_septembre ALD_BPCO_decembre;
run; quit;
```

3.3.9 09_Delivrances_de_traitement_BPCO.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*

*/
/*		Sélection des patients - Nombre de délivrances remboursées de traitements BPCO : BDLA, ATB et arrêt du tabac
*/
/*

*/
/*
	*****************************************************************************************************************
**********************************	*/

*
	*****************************************************************************************************************
**********************************;
*	BDLA;
*
	*****************************************************************************************************************
**********************************;

*	On récupère les délivrances remboursées de bronchodilatateur anticholinergique ou bêta-2 longue durée d action;

%suppr_table(
	lib = orauser,
	table = codes_ATC_BDLA
	);

data orauser.codes_ATC_BDLA;
	set orauser.codes_ATC_BPCO (where = (reperage in (45, 46, 47, 48, 49)));
run;

* On appelle la macro pour le repérage dans le DCIR - CIP;
%suppr_table(
	lib = travail,
	table = reperage_BDLA_&Annee_N.
	);

* Dans le DCIR - Codes CIP;
%extract_CIP(
	annee_deb = &Annee_N.,
	annee_fin = &Annee_N.,
	tbl_out = travail.reperage_BDLA_&Annee_N.,
	tbl_codes = codes_ATC_BDLA,
	tbl_patients = corresp_id_patient
	);

*
	*****************************************************************************************************************
**********************************;
*	Vérifications;

data verif;
	set travail.reperage_BDLA_&Annee_N.;
	annee_soin = year(date_debut);
run;

%proc_freq(
	in_tbl = verif,
	out_tbl = _trt_BPCO_BDLA,
	list_var_in = annee_soin*PHA_ATC_C07*PHA_CND_TOP,
	list_var_out = annee_soin PHA_ATC_C07 PHA_CND_TOP Frequency
	);

proc delete data = verif;
```

```
run; quit;

*
	************************************************************************************************************
************************************;
*	On compte le nombre de délivrances différentes par patient;

proc sql undo_policy = none;

	CREATE TABLE travail.reperage_BDLA_&Annee_N. AS
		SELECT
			BEN_IDT_ANO,
			COUNT(DISTINCT date_debut) AS Nb_Broncho_LDA length = 3
		FROM travail.reperage_BDLA_&Annee_N.
		WHERE YEAR(date_debut) = &annee_N.
		GROUP BY BEN_IDT_ANO;

quit;

*
	************************************************************************************************************
************************************;
*	On ajoute l information du nombre de délivrances remboursées de bronchodilatateur anticholinergique ou bêta-2 longue
durée d action dans
	l année dans la table de population;

proc sort data = pop.T_INDI_BPCO_&an_N.;
	by BEN_IDT_ANO;
run;

data pop.T_INDI_BPCO_&an_N.;
	merge	pop.T_INDI_BPCO_&an_N. (in = a )
			travail.reperage_BDLA_&Annee_N. (in = b);
	by BEN_IDT_ANO;
	if a;
	if not b then
		Nb_Broncho_LDA = 0;
run;

proc delete data = travail.reperage_BDLA_&Annee_N.;
run; quit;

proc delete data = orauser.codes_ATC_BDLA;
run; quit;

*
	************************************************************************************************************
************************************;
*	ATB;
*
	************************************************************************************************************
************************************;

*	On récupère les délivrances remboursées d antibiotiques;

%suppr_table(
	lib = orauser,
	table = codes_ATC_ATB
	);

data orauser.codes_ATC_ATB;
	set orauser.codes_ATC_BPCO (where = (reperage = 36));
run;

* On appelle la macro pour le repérage dans le DCIR - CIP;
%suppr_table(
	lib = travail,
	table = reperage_ATC_ATB
	);

%extract_CIP(
```

```
		annee_deb = &Annee_N.,
		annee_fin = &Annee_N.,
		tbl_out = travail.reperage_ATC_ATB,
		tbl_codes = codes_ATC_ATB,
		tbl_patients = corresp_id_patient
		);

*
		*************************************************************************************************
**********************************;
*		Vérifications;

data verif;
		set travail.reperage_ATC_ATB;
		annee_soin = year(date_debut);
run;

%proc_freq(
		in_tbl = verif,
		out_tbl = _trt_BPCO_ATB,
		list_var_in = annee_soin*PHA_ATC_C07*PHA_CND_TOP,
		list_var_out = annee_soin PHA_ATC_C07 PHA_CND_TOP Frequency
		);

proc delete data = verif;
run; quit;

*
		*************************************************************************************************
**********************************;
*		On compte le nombre de délivrances différentes par patient;

proc sql undo_policy = none;

		CREATE TABLE travail.reperage_ATC_ATB AS
			SELECT
				BEN_IDT_ANO,
				COUNT(DISTINCT date_debut) AS Nb_ATB length = 3
			FROM travail.reperage_ATC_ATB
			WHERE YEAR(date_debut) = &annee_N.
			GROUP BY BEN_IDT_ANO;

quit;

*
		*************************************************************************************************
**********************************;
*		On ajoute l information du nombre de délivrances remboursées d antibiotiques dans l année dans la table de population;

data pop.T_INDI_BPCO_&an_N.;
		merge	pop.T_INDI_BPCO_&an_N. (in = a)
				travail.reperage_ATC_ATB (in = b);
		by BEN_IDT_ANO;
		if a;
		if not b then
			Nb_ATB = 0;
run;

proc delete data = travail.reperage_ATC_ATB;
run; quit;

proc delete data = orauser.codes_ATC_ATB;
run; quit;

*
		*************************************************************************************************
**********************************;
*		Tabac;
*
		*************************************************************************************************
**********************************;
```

```
*	On récupère les délivrances remboursées de médicaments utilisés dans la dépendance tabagique;

%suppr_table(
	lib = orauser,
	table = codes_ATC_Tabac
	);

data orauser.codes_ATC_Tabac;
	set orauser.codes_ATC_BPCO (where = (reperage = 35));
run;

* On appelle la macro pour le repérage dans le DCIR - CIP;
%suppr_table(
	lib = travail,
	table = reperage_CIP_sdv_Tabac
	);

%extract_CIP(
	annee_deb = &Annee_N.,
	annee_fin = &Annee_N.,
	tbl_out = travail.reperage_ATC_Tabac,
	tbl_codes = codes_ATC_Tabac,
	tbl_patients = corresp_id_patient
	);

*
	**********************************************************************************************************
**********************************;
*	Vérifications;

data verif;
	set travail.reperage_ATC_Tabac;
	annee_soin = year(date_debut);
run;

%proc_freq(
	in_tbl = verif,
	out_tbl = _trt_BPCO_Tabac,
	list_var_in = annee_soin*PHA_ATC_C07*PHA_CND_TOP,
	list_var_out = annee_soin PHA_ATC_C07 PHA_CND_TOP Frequency
	);

proc delete data = verif;
run; quit;

*
	**********************************************************************************************************
**********************************;
*	On compte le nombre de délivrances différentes par patient;

proc sql undo_policy = none;

	CREATE TABLE travail.reperage_ATC_Tabac AS
		SELECT
			BEN_IDT_ANO,
			COUNT(DISTINCT date_debut) AS Nb_Tabac length = 3
		FROM travail.reperage_ATC_Tabac
		WHERE YEAR(date_debut) = &annee_N.
		GROUP BY BEN_IDT_ANO;

quit;

*
	**********************************************************************************************************
**********************************;
*	On ajoute l information du nombre de délivrances remboursées de médicaments utilisés dans la dépendance tabagique dans
l année dans la
	table de population;

data pop.T_INDI_BPCO_&an_N.;
```

```
		merge	pop.T_INDI_BPCO_&an_N. (in = a)
					travail.reperage_ATC_Tabac (in = b);
		by BEN_IDT_ANO;
		if a;
		if not b then
			Nb_Tabac = 0;
run;

proc delete data = travail.reperage_ATC_Tabac;
run; quit;

proc delete data = orauser.codes_ATC_Tabac;
run; quit;

*
		*************************************************************************************************************
**********************************;
*		Vérifications;

%proc_freq(
		in_tbl = pop.T_INDI_BPCO_&an_N.,
		out_tbl = _Nb_BRONCHO_LDA,
		list_var_in = Nb_BRONCHO_LDA,
		list_var_out = Nb_BRONCHO_LDA Frequency
		);

%proc_freq(
		in_tbl = pop.T_INDI_BPCO_&an_N.,
		out_tbl = _NB_ATB,
		list_var_in = NB_ATB,
		list_var_out = NB_ATB Frequency
		);

%proc_freq(
		in_tbl = pop.T_INDI_BPCO_&an_N.,
		out_tbl = _NB_TABAC,
		list_var_in = NB_TABAC,
		list_var_out = NB_TABAC Frequency
		);
```

3.3.10 10_Table_finale.sas

```
/*
	*****************************************************************************************************
**********************************	*/
/*
                                                                                                        */
/*		Sélection des patients - Table finale : Pré-sélection des populations de chaque indicateur
                                                                                                        */
/*
                                                                                                        */
/*
	*****************************************************************************************************
**********************************	*/

proc sql;

	ALTER TABLE pop.T_INDI_BPCO_&an_N. ADD indicateur_01 INT length = 3, indicateur_02a INT length = 3, indicateur_02b
INT length = 3,
		indicateur_03 INT length = 3, indicateur_04 INT length = 3, indicateur_05 INT length = 3, indicateur_06 INT length
= 3,
		indicateur_07 INT length = 3, indicateur_09 INT length = 3;

	* 01 : Patients suspects de BPCO (1 BDLA, 1 ATB, 1 TTT arrêt tabac), de 40 ans et plus;
	UPDATE pop.T_INDI_BPCO_&an_N. a
		SET indicateur_01 =	CASE	WHEN (Nb_Broncho_LDA >= 1 OR Nb_ATB >= 1 OR nb_Tabac >= 1) AND Age >=
40 THEN 1
                                                                        ELSE 0
                                                                        END;

	* 02a : Patients BPCO probable (3 BDLA), de 40 ans et plus;
	UPDATE pop.T_INDI_BPCO_&an_N. a
		SET indicateur_02a =	CASE	WHEN Nb_Broncho_LDA >= 3 AND Age >= 40 THEN 1
                                                                                ELSE 0
                                                                                END;

	* 02b : Patients BPCO diagnostiqués
                                        ALD active au 1er septembre de l année N, de 40 ans et plus
                                        hospitalisés pour BPCO en MCO, SSR ou HAD, de 40 ans et plus;
	UPDATE pop.T_INDI_BPCO_&an_N. a
		SET indicateur_02b =	CASE	WHEN (ALD_BPCO_septembre = 1 AND Age >= 40) OR (Nb_hospit_BPCO_Ind02
>= 1 AND Age >= 40) THEN 1
                                                                                ELSE 0
                                                                                END;

	* 03 : Patients BPCO probable (3 BDLA) ou diagnostiqués BPCO (ALD active au 31 décembre de l année N ou hospitalisés pour
BPCO en MCO, SSR ou HAD),
		de 40 ans et plus;
	UPDATE pop.T_INDI_BPCO_&an_N. a
		SET indicateur_03 =	CASE	WHEN (Nb_Broncho_LDA >= 3 OR ALD_BPCO_decembre = 1 OR Nb_hospit_BPCO
>= 1) AND Age >= 40 THEN 1
                                                                        ELSE 0
                                                                        END;

	* 04 : Patients avec au moins un séjour pour exacerbation de BPCO, de 40 ans et plus;
	UPDATE pop.T_INDI_BPCO_&an_N. a
		SET indicateur_04 =	CASE	WHEN Nb_Sej_Exacerbation >= 1 AND Age >= 40 THEN 1
                                                                        ELSE 0
                                                                        END;

	* 05 : Patients avec au moins un séjour pour exacerbation de BPCO, de 40 ans et plus;
	UPDATE pop.T_INDI_BPCO_&an_N. a
		SET indicateur_05 =	CASE	WHEN Nb_Sej_Exacerbation >= 1 AND Age >= 40 THEN 1
                                                                                ELSE 0
                                                                                END;

	* 06 : Patients avec au moins un séjour pour exacerbation de BPCO, de 40 ans et plus;
```

```
	UPDATE pop.T_INDI_BPCO_&an_N. a
		SET indicateur_06 =   CASE	WHEN Nb_Sej_Exacerbation >= 1 AND Age >= 40 THEN 1
									ELSE 0
									END;

	* 07 : Patients avec au moins un séjour pour exacerbation de BPCO, de 40 ans et plus;
	UPDATE pop.T_INDI_BPCO_&an_N. a
		SET indicateur_07 =   CASE	WHEN Nb_Sej_Exacerbation >= 1 AND Age >= 40 THEN 1
									ELSE 0
									END;

	* 09 : Patients avec au moins un séjour pour exacerbation de BPCO, de 40 ans et plus;
	UPDATE pop.T_INDI_BPCO_&an_N. a
		SET indicateur_09 =   CASE	WHEN Nb_Sej_Exacerbation >= 1 AND Age >= 40 THEN 1
								ELSE 0
								END;

quit;

*
	**************************************************************************************************
**********************************;
*	Vérifications;

data verif;
	set pop.T_INDI_BPCO_&an_N.;
	length age_40 3.;
	format age_40 f_age_40_.;
	* Plus ou moins de 40 ans;
	if age < 40 then
		age_40 = 0;
	else
		age_40 = 1;
run;

* Indicateur 01;
%proc_freq(
	in_tbl = verif,
	out_tbl = _ind01,
	list_var_in = Indicateur_01*Age_40*NB_BRONCHO_LDA*NB_ATB*NB_TABAC,
	list_var_out = Indicateur_01 Age_40 NB_BRONCHO_LDA NB_ATB NB_TABAC Frequency
	);

* Indicateur 02a;
%proc_freq(
	in_tbl = verif,
	out_tbl = _ind02a,
	list_var_in = Indicateur_02a*age_40*NB_BRONCHO_LDA,
	list_var_out = Indicateur_02a age_40 NB_BRONCHO_LDA Frequency
	);

* Indicateur 02b;
%proc_freq(
	in_tbl = verif,
	out_tbl = _ind02b,
	list_var_in = Indicateur_02b*age_40*ALD_BPCO_SEPTEMBRE*Nb_hospit_BPCO_Ind02,
	list_var_out = Indicateur_02b age_40 ALD_BPCO_SEPTEMBRE Nb_hospit_BPCO_Ind02 Frequency
	);

* Indicateur 03;
%proc_freq(
	in_tbl = verif,
	out_tbl = _ind03,
	list_var_in = Indicateur_03*Age_40*NB_BRONCHO_LDA*ALD_BPCO_DECEMBRE*NB_HOSPIT_BPCO,
	list_var_out = Indicateur_03 Age_40 NB_BRONCHO_LDA ALD_BPCO_DECEMBRE NB_HOSPIT_BPCO Frequency
	);

* Indicateur 04;
%proc_freq(
	in_tbl = verif,
	out_tbl = _ind04,
```

```
		list_var_in = Indicateur_04*Age_40*Nb_Sej_Exacerbation,
		list_var_out = Indicateur_04 Age_40 Nb_Sej_Exacerbation Frequency
		);

* Indicateur 05;
%proc_freq(
		in_tbl = verif,
		out_tbl = _ind05,
		list_var_in = Indicateur_05*Age_40*Nb_Sej_Exacerbation,
		list_var_out = Indicateur_05 Age_40 Nb_Sej_Exacerbation Frequency
		);

* Indicateur 06;
%proc_freq(
		in_tbl = verif,
		out_tbl = _ind06,
		list_var_in = Indicateur_06*Age_40*Nb_Sej_Exacerbation,
		list_var_out = Indicateur_06 Age_40 Nb_Sej_Exacerbation Frequency
		);

* Indicateur 07;
%proc_freq(
		in_tbl = verif,
		out_tbl = _ind07,
		list_var_in = Indicateur_07*Age_40*Nb_Sej_Exacerbation,
		list_var_out = Indicateur_07 Age_40 Nb_Sej_Exacerbation Frequency
		);

* Indicateur 09;
%proc_freq(
		in_tbl = verif,
		out_tbl = _ind09,
		list_var_in = Indicateur_09*Age_40*Nb_Sej_Exacerbation,
		list_var_out = Indicateur_09 Age_40 Nb_Sej_Exacerbation Frequency
		);

proc delete data = verif;
run; quit;

*
		**********************************************************************************************************
**********************************;
*		On crée la table de résultat et une table de correspondance pour les patients qu on a conservés pour un des indicateurs;

data res.T_INDI_BPCO_&an_N.;
		set pop.T_INDI_BPCO_&an_N. (keep = Ben_Idt_Ano BEN_NIR_PSA Age Ben_Sex_Cod Ben_Dcd_Dte BEN_RES_DPT
BEN_RES_COM RGM_GRG_COD Nb_hospit_BPCO
			Nb_hospit_BPCO_Ind02 Nb_Sej_Exacerbation ALD_BPCO_&an_N. ALD_BPCO_septembre ALD_BPCO_decembre
Nb_Broncho_LDA Nb_ATB Nb_Tabac
			exclus_pas_40ans exclus_jumeaux exclus_pas_remb exclus_NIR_fictif Indicateur_01 Indicateur_02a
Indicateur_02b Indicateur_03
			Indicateur_04 Indicateur_05 Indicateur_06 Indicateur_07 Indicateur_09 where = (exclus_pas_40ans = "0" and
exclus_jumeaux = "0"
			and exclus_pas_remb = "0" and exclus_NIR_fictif = "0" and (Indicateur_01 = 1 or Indicateur_02a = 1 or
Indicateur_02b = 1 or
			Indicateur_03 = 1 or Indicateur_04 = 1 or Indicateur_05 = 1 or Indicateur_06 = 1 or Indicateur_07 = 1 or
Indicateur_09 = 1)));
run;

* On supprime la table de correspondance si elle existe;
%suppr_table(
		lib = orauser,
		table = corresp_id_patient
		);

proc sort data = res.T_INDI_BPCO_&an_N. (keep = BEN_IDT_ANO BEN_NIR_PSA) out = orauser.corresp_id_patient nodupkey;
		by BEN_IDT_ANO BEN_NIR_PSA;
run;

proc sort data = res.T_INDI_BPCO_&an_N. (drop = BEN_NIR_PSA) nodupkey;
		by BEN_IDT_ANO;
```

```
run;

proc sql;

	CREATE INDEX BEN_IDT_ANO ON res.T_INDI_BPCO_&an_N. (BEN_IDT_ANO);

quit;

* Vérif;
proc sql;

	SELECT COUNT(*) AS nb_lignes, COUNT(DISTINCT BEN_IDT_ANO) AS nb_patients
	FROM res.T_INDI_BPCO_&an_N.;

quit;

*
	**********************************************************************************************************
**********************************;
*	On crée une table contenant les indicateurs;

data pop.indicateurs_&an_N.;
	set pop.T_INDI_BPCO_&an_N. (keep = Ben_Idt_Ano exclus_pas_40ans exclus_jumeaux exclus_pas_remb exclus_NIR_fictif
Indicateur_01
				Indicateur_02a Indicateur_02b Indicateur_03 Indicateur_04 Indicateur_05 Indicateur_06 Indicateur_07
Indicateur_09
				where = (exclus_pas_40ans = "0" and exclus_jumeaux = "0" and exclus_pas_remb = "0" and
exclus_NIR_fictif = "0" and (Indicateur_01 = 1
				or Indicateur_02a = 1 or Indicateur_02b = 1 or Indicateur_03 = 1 or Indicateur_04 = 1 or Indicateur_05
= 1 or Indicateur_06 = 1
				or Indicateur_07 = 1 or Indicateur_09 = 1)));
	drop exclus_pas_40ans exclus_jumeaux exclus_pas_remb exclus_NIR_fictif;
run;

*
	**********************************************************************************************************
**********************************;
*	On conserve des nombre de patients de 40 ans et plus ayant bénéficié de soins remboursés au moins une fois l année N
(base pour calculer
*	les % des flowcharts);

proc sql;

	CREATE TABLE travail.ref_flowcharts AS
		SELECT
			COUNT(DISTINCT BEN_IDT_ANO) AS Nb_patients_ref0 length = 3
		FROM pop.T_indi_BPCO_&an_N.
		WHERE Age >= 40 AND exclus_jumeaux = "0" AND exclus_pas_remb = "0" AND exclus_NIR_fictif = "0";

quit;

*
	**********************************************************************************************************
**********************************;
*	On supprime la table pop.T_indi_BPCO_&an_N.;

proc delete data = pop.T_indi_BPCO_&an_N.;
run; quit;
```

3.4 Informations générales sur les patients

3.4.1 01_CMUc.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*

                                                                                                                    */
/*			Patient CMUc

                                                                                                        */
/*

                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/

*
	*****************************************************************************************************************
**********************************;
*	On récupère l information de la CMUc : si le patient a au moins 1 remboursement avec BEN_CMU_TOP = 1 dans l année;

%macro CMUC_DCIR(annee_deb=, annee_fin=);

	%let annee_fin_flx = %sysevalf(&annee_fin. + 1);

	* On boucle sur les années de repérage (en flux);
	%do Annee = &annee_deb. %to &annee_fin_flx.;

		%put annee_deb = &Annee_deb.;
		* On boucle sur les 12 mois de l année;
		%do i = 1 %to 12;

			%if &i. < 10 %then %let Mois = 0&i.;
			%else %let Mois = &i.;

			%put Données de &Mois./&Annee.;

			proc sql undo_policy = none;

				%connectora;

					CREATE TABLE patients_CMUC_&Annee._&Mois. AS
						SELECT * FROM CONNECTION TO ORACLE (
							SELECT DISTINCT
								a.BEN_IDT_ANO,
								1 AS BEN_CMU_TOP
							FROM corresp_id_patient a
								INNER JOIN ER_PRS_F b
									ON a.BEN_NIR_PSA = b.BEN_NIR_PSA
							WHERE b.BEN_CDI_NIR = '00' AND b.FLX_DIS_DTD =
TO_DATE(%str(%'&Annee.&Mois.01%'), 'YYYYMMDD')
								AND b.EXE_SOI_DTD BETWEEN
TO_DATE(%str(%'&Annee_deb.0101%'), 'YYYYMMDD')
									AND
TO_DATE(%str(%'&Annee_deb.1231%'), 'YYYYMMDD')
								AND BEN_CMU_TOP = '1'
							);

				%disconnect from oracle;

			quit;

			%arret_erreur;
```

```
                %end;
                * Fin de la boucle sur les 12 mois;

        %end;
        * Fin de la boucle par année;

        * On empile toutes les tables mensuelles;
        data patients_CMUC;
                set patients_CMUC_:;
        run;

        %arret_erreur;

        proc datasets library = work memtype = data nolist;
                delete patients_CMUC_:;
        run;

%mend CMUC_DCIR;

%CMUC_DCIR(
        annee_deb = &Annee_N.,
        annee_fin = &Annee_N.
        );

*
        ************************************************************************************************************
**********************************;
*       On ajoute l information de la CMUc dans la table de population;
proc sql;

        ALTER TABLE res.T_INDI_BPCO_&an_N. ADD BEN_CMU_TOP INT length = 3;

        UPDATE res.T_INDI_BPCO_&an_N. a
                SET BEN_CMU_TOP = CASE  WHEN BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM patients_CMUC) THEN 1
                                                                                                ELSE 0
                                                                                                END;

quit;

*
        ************************************************************************************************************
**********************************;
*       On supprime les tables temporaires;

proc delete data = patients_CMUC;
run; quit;

*
        ************************************************************************************************************
**********************************;
*       Vérifications;

%proc_freq(
        in_tbl = res.T_INDI_BPCO_&an_N.,
        out_tbl = _CMUC,
        list_var_in = BEN_CMU_TOP,
        list_var_out = BEN_CMU_TOP Frequency
        );
```

3.4.2 02_IDS.sas

```
/*
	*****************************************************************************************************
**********************************	*/
/*

*/
/*			Indice de défaveur social

*/
/*

*/
/*
	*****************************************************************************************************
**********************************	*/

*
	*****************************************************************************************************
**********************************;
*	On ajoute l information de l IDS dans la table de population;

data patient_depcom;
	set res.T_INDI_BPCO_&an_N. (keep = BEN_IDT_ANO BEN_RES_DPT BEN_RES_COM);
	length code_postal $5;
	code_postal = SUBSTR(BEN_RES_DPT, 2, 2)||BEN_RES_COM;
run;

proc sql undo_policy = none;

	CREATE TABLE patient_depcom AS
		SELECT DISTINCT
			a.BEN_IDT_ANO,
			b.code_insee
		FROM patient_depcom a
			INNER JOIN rfcommun.T_FIN_GEO_LOC_FRANCE b
				ON a.code_postal = b.code_jointure
		ORDER BY code_insee, BEN_IDT_ANO;

quit;

proc sort data = consopat.DEFA_UU2013 out = DEFA_UU2013 (keep = depcom FDEP13 quintile_pop);
	by depcom;
run;

data patient_depcom (keep = BEN_IDT_ANO code_insee FDEP13 quintile_pop);
	merge	patient_depcom (in = a)
			DEFA_UU2013 (in = b rename = (depcom = code_insee));
	by code_insee;
	if a;
run;

*
	*****************************************************************************************************
**********************************;
*	On ajoute l information de l IDS dans la table de population;

proc sort data = patient_depcom nodupkey;
	by BEN_IDT_ANO;
run;

proc sort data = res.T_INDI_BPCO_&an_N. force;
	by BEN_IDT_ANO;
run;

data res.T_INDI_BPCO_&an_N.;
	merge	res.T_INDI_BPCO_&an_N. (in = a)
			patient_depcom (in = b);
```

```
		by BEN_IDT_ANO;
		if a;
run;

* On recrée l index qu on a cassé avec le proc sort;
proc sql;

		CREATE INDEX BEN_IDT_ANO ON res.T_INDI_BPCO_&an_N. (BEN_IDT_ANO);

quit;

*
		**************************************************************************************************************
**********************************;
*		On supprime la table temporaire;

proc delete data = patient_depcom DEFA_UU2013;
run; quit;

*
		**************************************************************************************************************
**********************************;
*		Vérifications;

data verif;
		set res.T_INDI_BPCO_&an_N.;
		format FDEP_renseigne f_oui_non.;
		if FDEP13 = . then
			FDEP_renseigne = 0;
		else
			FDEP_renseigne = 1;
run;

%proc_freq(
		in_tbl = verif,
		out_tbl = _FDEP13,
		list_var_in = FDEP_renseigne,
		list_var_out = FDEP_renseigne Frequency
		);

proc delete data = verif;
run; quit;
```

3.4.3 03_Infos_ALD.sas

```
/*
	**********************************************************************************************************
**********************************	*/
/*

*/
/*			Récupération des données d'ALD

*/
/*

*/
/*
	**********************************************************************************************************
**********************************	*/

*
	**********************************************************************************************************
**********************************;
*	On récupère la ligne concernant l info maximale pour l année d inclusion;

proc sql undo_policy = none;

	%connectora;

		CREATE TABLE histo_ALD_&annee_N. AS
			SELECT DISTINCT * FROM CONNECTION TO ORACLE (
				SELECT DISTINCT
					a.BEN_IDT_ANO,
					b.IMB_ETM_NAT,
					b.IMB_ALD_DTD,
					b.IMB_ALD_DTF,
					b.INS_DTE,
					b.UPD_DTE,
					b.IMB_ALD_NUM,
					b.MED_MTF_COD,
					CASE	WHEN SUBSTR(b.MED_MTF_COD, 1, 3) BETWEEN 'A00' AND 'B99'
THEN 1
						WHEN SUBSTR(b.MED_MTF_COD, 1, 3) BETWEEN 'C00'
AND 'D48' THEN 2
						WHEN SUBSTR(b.MED_MTF_COD, 1, 3) BETWEEN 'D50'
AND 'D89' THEN 3
						WHEN SUBSTR(b.MED_MTF_COD, 1, 3) BETWEEN 'E00'
AND 'E90' THEN 4
						WHEN SUBSTR(b.MED_MTF_COD, 1, 3) BETWEEN 'F00'
AND 'F99' THEN 5
						WHEN SUBSTR(b.MED_MTF_COD, 1, 3) BETWEEN 'G00'
AND 'G99' THEN 6
						WHEN SUBSTR(b.MED_MTF_COD, 1, 3) BETWEEN 'H00'
AND 'H59' THEN 7
						WHEN SUBSTR(b.MED_MTF_COD, 1, 3) BETWEEN 'H60'
AND 'H95' THEN 8
						WHEN SUBSTR(b.MED_MTF_COD, 1, 3) BETWEEN 'I00' AND
'I99' THEN 9
						WHEN SUBSTR(b.MED_MTF_COD, 1, 3) BETWEEN 'J00' AND
'J99' THEN 10
						WHEN SUBSTR(b.MED_MTF_COD, 1, 3) BETWEEN 'K00'
AND 'K93' THEN 11
						WHEN SUBSTR(b.MED_MTF_COD, 1, 3) BETWEEN 'L00'
AND 'L99' THEN 12
						WHEN SUBSTR(b.MED_MTF_COD, 1, 3) BETWEEN 'M00'
AND 'M99' THEN 13
						WHEN SUBSTR(b.MED_MTF_COD, 1, 3) BETWEEN 'N00'
AND 'N99' THEN 14
						WHEN SUBSTR(b.MED_MTF_COD, 1, 3) BETWEEN 'O00'
AND 'O99' THEN 15
```

```
                                                                WHEN SUBSTR(b.MED_MTF_COD, 1, 3) BETWEEN 'P00'
AND 'P94' THEN 16
                                                                WHEN SUBSTR(b.MED_MTF_COD, 1, 3) BETWEEN 'Q00'
AND 'Q99' THEN 17
                                                                WHEN SUBSTR(b.MED_MTF_COD, 1, 3) BETWEEN 'R00'
AND 'R99' THEN 18
                                                                WHEN SUBSTR(b.MED_MTF_COD, 1, 3) BETWEEN 'S00'
AND 'T98' THEN 19
                                                                WHEN SUBSTR(b.MED_MTF_COD, 1, 3) BETWEEN 'V01'
AND 'Y98' THEN 20
                                                                WHEN SUBSTR(b.MED_MTF_COD, 1, 3) BETWEEN 'Z00'
AND 'Z99' THEN 21
                                                                WHEN SUBSTR(b.MED_MTF_COD, 1, 3) BETWEEN 'U00'
AND 'U99' THEN 22
                                                                END AS Chapitre
                                        FROM corresp_id_patient a
                                                INNER JOIN IR_IMB_R b
                                                        ON a.BEN_NIR_PSA = b.BEN_NIR_PSA
                                        WHERE IMB_ETM_NAT IN (41, 43, 45) AND b.INS_DTE <
TO_DATE(%str(%'&Annee_N1.0501%'), 'YYYYMMDD')
                                        );

        disconnect from oracle;

quit;

proc sort data = histo_ALD_&annee_N.;
                by BEN_IDT_ANO IMB_ETM_NAT IMB_ALD_NUM MED_MTF_COD INS_DTE UPD_DTE IMB_ALD_DTF descending
IMB_ALD_DTD;
run;

data histo_ALD_&annee_N.;
        set histo_ALD_&annee_N.;
        by BEN_IDT_ANO IMB_ETM_NAT IMB_ALD_NUM MED_MTF_COD INS_DTE UPD_DTE IMB_ALD_DTF descending
IMB_ALD_DTD;
        if last.MED_MTF_COD then
                output;
run;

data histo_ALD_&annee_N. (rename = (IMB_ALD_DTD2 = IMB_ALD_DTD IMB_ALD_DTF2 = IMB_ALD_DTF));
        set histo_ALD_&annee_N.;
        length IMB_ALD_DTD2 IMB_ALD_DTF2 4.;
        IMB_ALD_DTD2 = datepart(IMB_ALD_DTD);
        IMB_ALD_DTF2 = datepart(IMB_ALD_DTF);
        format IMB_ALD_DTD2 IMB_ALD_DTF2 ddmmyy10.;
        drop IMB_ALD_DTD IMB_ALD_DTF;
        if IMB_ALD_DTD2 <= "31DEC&Annee_N."d and (IMB_ALD_DTF2 = "01JAN1600"d or IMB_ALD_DTF2 >= "01JAN&Annee_N."d)
then
                output;
run;

*
        ************************************************************************************************************
************************************;
*       On récupère une ligne par patient et par chapitre;

proc sort data = histo_ALD_&annee_N. out = chapitre_&annee_N. (keep = BEN_IDT_ANO Chapitre) nodupkey;
        by BEN_IDT_ANO Chapitre;
run;

* On ajoute l information dans la table de population;
%macro ajout_chapitres_ALD;

        %do ALD = 1 %to 22;

                %put Ajout de l ALD no &ALD. ;

                proc sql;

                        ALTER TABLE res.T_INDI_BPCO_&an_N. ADD ALD_chapitre_&ALD. INT length = 3;
```

```
                                UPDATE res.T_INDI_BPCO_&an_N.
                                        SET ALD_chapitre_&ALD. =
                                                CASE        WHEN BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM
chapitre_&annee_N. WHERE Chapitre = &ALD.) THEN 1
                                                                                ELSE 0
                                                                                END;

                        quit;

                %end;

%mend ajout_chapitres_ALD;
%ajout_chapitres_ALD;

*
                ************************************************************************************************************
***********************************;
*               On ajoute l information d au moins une ALD dans l année dans la table de population;

proc sql;

                ALTER TABLE res.T_INDI_BPCO_&an_N. ADD ALD INT length = 3;

                UPDATE res.T_INDI_BPCO_&an_N.
                        SET ALD =
                                CASE        WHEN BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM histo_ALD_&annee_N.) THEN 1
                                        ELSE 0
                                        END;

quit;

*
                ************************************************************************************************************
***********************************;
*               On supprime les tables temporaires;

proc datasets library = work memtype = data nolist;
                delete histo_ALD_&annee_N. chapitre_&annee_N.;
run; quit;

*
                ************************************************************************************************************
***********************************;
*               Vérifications;

* On ne passe pas par la macro car trop de variables;
proc sql;

                CREATE TABLE verif._0&tmp_num_tab._ALD AS
                        SELECT
                                ALD,
                                ALD_Chapitre_1,
                                ALD_Chapitre_2,
                                ALD_Chapitre_3,
                                ALD_Chapitre_4,
                                ALD_Chapitre_5,
                                ALD_Chapitre_6,
                                ALD_Chapitre_7,
                                ALD_Chapitre_8,
                                ALD_Chapitre_9,
                                ALD_Chapitre_10,
                                ALD_Chapitre_11,
                                ALD_Chapitre_12,
                                ALD_Chapitre_13,
                                ALD_Chapitre_14,
                                ALD_Chapitre_15,
                                ALD_Chapitre_16,
                                ALD_Chapitre_17,
                                ALD_Chapitre_18,
                                ALD_Chapitre_19,
                                ALD_Chapitre_20,
```

```
                                ALD_Chapitre_21,
                                ALD_Chapitre_22,
                                COUNT(*) AS Frequency
                        FROM res.T_INDI_BPCO_&an_N.
                        GROUP BY ALD, ALD_Chapitre_1, ALD_Chapitre_2, ALD_Chapitre_3, ALD_Chapitre_4, ALD_Chapitre_5,
ALD_Chapitre_6, ALD_Chapitre_7,
                                ALD_Chapitre_8, ALD_Chapitre_9, ALD_Chapitre_10, ALD_Chapitre_11, ALD_Chapitre_12,
ALD_Chapitre_13, ALD_Chapitre_14,
                                ALD_Chapitre_15, ALD_Chapitre_16, ALD_Chapitre_17, ALD_Chapitre_18, ALD_Chapitre_19,
ALD_Chapitre_20, ALD_Chapitre_21,
                                ALD_Chapitre_22;

quit;

proc sql noprint; SELECT SUM(Frequency) INTO: nb_tot FROM verif._0&tmp_num_tab._ALD; quit;

data verif._0&tmp_num_tab._ALD;
        set verif._0&tmp_num_tab._ALD;
        length percent 4.;
        format percent nlpct7.1 Frequency commafr_0ch.;
        percent = Frequency/&nb_tot.;
run;

%let tmp_num_tab = %sysevalf(&tmp_num_tab. + 1);

%macro verif_ALD;

        %do i = 1 %to 22;

                %proc_freq(
                        in_tbl = res.T_INDI_BPCO_&an_N.,
                        out_tbl = _ALD&i.,
                        list_var_in = ALD_Chapitre_&i.,
                        list_var_out = ALD_Chapitre_&i. Frequency
                        );

        %end;

%mend verif_ALD;
%verif_ALD;
```

3.4.4 04_Score_de_Charlson.sas

```
/*
	*************************************************************************************************************
**********************************	*/
/*
                                                                                                                */
/*			Calcul du score de Charlson
                                                                                                                */
/*
                                                                                                                */
/*
	*************************************************************************************************************
**********************************	*/

*
	*************************************************************************************************************
**********************************;
*	Récupération des soins - Actes CCAM;

* On appelle la macro pour le repérage dans le DCIR;
%suppr_table(
	lib = work,
	table = reperage_CCAM_sdv_Charlson
	);

%extract_CCAM_DCIR(
	annee_deb = &annee_1N.,
	annee_fin = &annee_N.,
	tbl_out = reperage_CCAM_sdv_Charlson,
	tbl_codes = codes_CCAM_Charlson,
	tbl_patients = corresp_id_patient
	);

* On appelle la macro pour le repérage dans le PMSI;
%suppr_table(
	lib = work,
	table = reperage_CCAM_PMSI_Charlson
	);

%extract_CCAM_PMSI(
	annee_deb = &annee_1N.,
	annee_fin = &annee_N.,
	HAD = 1,
	MCO = 1,
	RIP = 1,
	SSR = 1,
	tbl_out = reperage_CCAM_PMSI_Charlson,
	tbl_codes = codes_CCAM_Charlson,
	tbl_patients = corresp_id_patient
	);

* On concatène dans une table finale;
%suppr_table(
	lib = travail,
	table = reperage_CCAM_Charlson
	);

data travail.reperage_CCAM_Charlson;
	length type $12.;
	set	reperage_CCAM_sdv_Charlson
		reperage_CCAM_PMSI_Charlson;
run;

proc delete data = reperage_CCAM_sdv_Charlson reperage_CCAM_PMSI_Charlson;
run; quit;
```

```
*
	*****************************************************************************************************
***********************************;
*	Récupération des soins - Codes CIP & UCD;

* On appelle la macro pour le repérage dans le DCIR - CIP;
%suppr_table(
	lib = work,
	table = reperage_CIP_sdv_Charlson
	);

%extract_CIP(
	annee_deb = &annee_1N.,
	annee_fin = &annee_N.,
	tbl_out = reperage_CIP_sdv_Charlson,
	tbl_codes = codes_ATC_Charlson,
	tbl_patients = corresp_id_patient
	);

* On appelle la macro pour le repérage dans le DCIR - UCD;
%suppr_table(
	lib = work,
	table = reperage_UCD_sdv_Charlson
	);

%extract_UCD_DCIR(
	annee_deb = &annee_1N.,
	annee_fin = &annee_N.,
	tbl_out = reperage_UCD_sdv_Charlson,
	tbl_codes = codes_ATC_Charlson,
	tbl_patients = corresp_id_patient
	);

* On appelle la macro pour le repérage dans le PMSI;
%suppr_table(
	lib = work,
	table = reperage_UCD_PMSI_Charlson
	);

%extract_UCD_PMSI(
	annee_deb = &annee_1N.,
	annee_fin = &annee_N.,
	HAD = 1,
	MCO = 1,
	RIP = 1,
	SSR = 1,
	tbl_out = reperage_UCD_PMSI_Charlson,
	tbl_codes = codes_ATC_Charlson,
	tbl_patients = corresp_id_patient
	);

* On concatène dans une table finale;
%suppr_table(
	lib = travail,
	table = reperage_ATC_Charlson
	);

data travail.reperage_ATC_Charlson;
	length type $12.;
	set	reperage_CIP_sdv_Charlson
		reperage_UCD_sdv_Charlson
		reperage_UCD_PMSI_Charlson;
run;

proc delete data = reperage_CIP_sdv_Charlson reperage_UCD_sdv_Charlson reperage_UCD_PMSI_Charlson;
run; quit;

*
	*****************************************************************************************************
*********************************;
*	Récupération des soins - Codes CIM10;
```

```
* Dans les ALD;
%suppr_table(
		lib = work,
		table = reperage_CIM10_ALD_Charlson
		);

%extract_ALD_CIM(
		tbl_out = reperage_CIM10_ALD_Charlson,
		tbl_codes = codes_CIM_Charlson,
		tbl_patients = corresp_id_patient
		);



* Dans les hospitalisations;
%suppr_table(
		lib = work,
		table = reperage_CIM10_PMSI_Charlson
		);

%extract_CIM10_PMSI(
		annee_deb = &annee_1N.,
		annee_fin = &annee_N.,
		HAD_DP = 0,
		HAD_DAS = 0,
		HAD_MPP = 0,
		HAD_MPA = 0,
		MCO_DP = 1,
		MCO_DR = 1,
		MCO_DAS = 1,
		MCO_DP_UM = 1,
		MCO_DR_UM = 1,
		SSR_FP = 0,
		SSR_MPP = 0,
		SSR_AE = 0,
		SSR_DAS = 0,
		tbl_out = reperage_CIM10_PMSI_Charlson,
		tbl_codes = codes_CIM_Charlson,
		tbl_patients = corresp_id_patient
		);

data travail.reperage_CIM10_Charlson ;
		length type $12.;
		set		reperage_CIM10_ALD_Charlson (in = a)
				reperage_CIM10_PMSI_Charlson (in = b);
		if b then
				source = "PMSI";
		if a then
				source = "ALD";
run;

proc delete data = reperage_CIM10_ALD_Charlson reperage_CIM10_PMSI_Charlson;
run; quit;

*
		**********************************************************************************************************
**********************************;
*		Récupération des soins - GHM;

%suppr_table(
		lib = travail,
		table = reperage_GHM_Charlson
		);

%extract_GHM_PMSI(
		annee_deb = &annee_1N.,
		annee_fin = &annee_N.,
		tbl_out = travail.reperage_GHM_Charlson,
		tbl_codes = codes_GHM_Charlson,
		tbl_patients = corresp_id_patient
```

```
	);

*
	**********************************************************************************************************
**********************************;
*	À partir des données repérées, sélectionner les lignes pour lesquels les soins ont lieu dans l année précédant le 1er janvier de
l année
	de repérage;

* Actes CCAM;
data travail.reperage_CCAM_Charlson;
	set travail.reperage_CCAM_Charlson (where = (
				intnx("year", "01JAN&Annee_N."d, -1, 'same') <= date_debut <= "01JAN&Annee_N."d
		or intnx("year", "01JAN&Annee_N."d, -1, 'same') <= date_fin <= "01JAN&Annee_N."d
		or date_debut < intnx("year", "01JAN&Annee_N."d, -1, 'same') and date_fin > "01JAN&Annee_N."d));
run;

* Codes CIM 10;
data travail.reperage_CIM10_Charlson;
	set travail.reperage_CIM10_Charlson (where = ((source = "PMSI" and
				intnx("year", "01JAN&Annee_N."d, -1, 'same') <= date_debut <= "01JAN&Annee_N."d
		or intnx("year", "01JAN&Annee_N."d, -1, 'same') <= date_fin <= "01JAN&Annee_N."d
		or date_debut < intnx("year", "01JAN&Annee_N."d, -1, 'same') and date_fin > "01JAN&Annee_N."d) or (source =
"ALD" and
		date_debut <= intnx("year", "31DEC&Annee_N."d, -1, 'same') and (date_fin = "01JAN1600"d or date_fin is null
		or date_fin >= intnx("year", "01JAN&Annee_N."d, -1, 'same')))));
run;

* GHM;
data travail.reperage_GHM_Charlson;
	set travail.reperage_GHM_Charlson (where = (
				intnx("year", "01JAN&Annee_N."d, -1, 'same') <= date_debut <= "01JAN&Annee_N."d
		or intnx("year", "01JAN&Annee_N."d, -1, 'same') <= date_fin <= "01JAN&Annee_N."d
		or date_debut < intnx("year", "01JAN&Annee_N."d, -1, 'same') and date_fin > "01JAN&Annee_N."d));
run;

* Codes CIP & UCD;
data travail.reperage_ATC_Charlson;
	set travail.reperage_ATC_Charlson (where = (
				intnx("year", "01JAN&Annee_N."d, -1, 'same') <= date_debut <= "01JAN&Annee_N."d
		or intnx("year", "01JAN&Annee_N."d, -1, 'same') <= date_fin <= "01JAN&Annee_N."d
		or date_debut < intnx("year", "01JAN&Annee_N."d, -1, 'same') and date_fin > "01JAN&Annee_N."d));
run;

*
	**********************************************************************************************************
**********************************;
*	Pour les médicaments, on regarde les nombres de délivrances;

* Pour les traitements de la démence, il faut au moins 3 délivrances;
proc sql;

	CREATE TABLE travail.Charlson_DCIR_demence AS
		SELECT *
		FROM travail.reperage_ATC_Charlson
		WHERE reperage = 5
		GROUP BY BEN_IDT_ANO
		HAVING COUNT(DISTINCT date_debut) >= 3;

quit;

* Pour les traitements de reperage pulmonaire chronique, il faut au moins 3 délivrances;
proc sql;

	CREATE TABLE travail.Charlson_DCIR_pulmonaire AS
		SELECT *
		FROM travail.reperage_ATC_Charlson
		WHERE reperage = 6
		GROUP BY BEN_IDT_ANO
		HAVING COUNT(DISTINCT date_debut) >= 3;
```

```
quit;

* Pour les traitements de diabète sans complication, il faut au moins 3 délivrances (ou 2 en cas de grand conditionnement);
proc sql;

	CREATE TABLE travail.Charlson_DCIR_diabete2 AS
		SELECT *
		FROM travail.reperage_ATC_Charlson
		WHERE PHA_CND_TOP = "GC" AND reperage = 10
		GROUP BY BEN_IDT_ANO
		HAVING COUNT(DISTINCT date_debut) >= 2;

	CREATE TABLE travail.Charlson_DCIR_diabete3 AS
		SELECT *
		FROM travail.reperage_ATC_Charlson
		WHERE PHA_CND_TOP NE "GC" AND reperage = 10
		GROUP BY BEN_IDT_ANO
		HAVING COUNT(DISTINCT date_debut) >= 3;

quit;

*
	**********************************************************************************************************
**********************************;
*	Création de la table contenant le score de Charlson;

data travail.reperages_Charlson;
	set	travail.reperage_CCAM_Charlson (keep = BEN_IDT_ANO reperage)
		travail.reperage_CIM10_Charlson (keep = BEN_IDT_ANO reperage)
		travail.reperage_GHM_Charlson (keep = BEN_IDT_ANO reperage)
		travail.Charlson_DCIR_demence (keep = BEN_IDT_ANO reperage)
		travail.Charlson_DCIR_pulmonaire (keep = BEN_IDT_ANO reperage)
		travail.Charlson_DCIR_diabete2 (keep = BEN_IDT_ANO reperage)
		travail.Charlson_DCIR_diabete3 (keep = BEN_IDT_ANO reperage);
run;

* On récupère 1 info unique par patient;
proc sort data = travail.reperages_Charlson nodupkey;
	by BEN_IDT_ANO reperage;
run;

proc delete data = travail.reperage_CCAM_Charlson travail.reperage_CIM10_Charlson travail.reperage_GHM_Charlson
travail.Charlson_DCIR_demence
	travail.Charlson_DCIR_pulmonaire travail.Charlson_DCIR_diabete2 travail.Charlson_DCIR_diabete3;
run; quit;

*
	**********************************************************************************************************
**********************************;
*	Correction des reperages;

* Les patients repérés pour Diabète sans complication + Patho cérébrovasculaire ou patho rénale modérée ou sévère ou infarctus du
myocarde;
*	=> Diabète avec complication;
data patients_Diabete_sans;
	set travail.reperages_Charlson;
	where reperage = 10;
run;

data patients_complications;
	set travail.reperages_Charlson;
	where reperage in (1, 4, 12);
run;

proc sql;

	DELETE FROM travail.reperages_Charlson
		WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM patients_Diabete_sans)
			AND BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM patients_complications)
			AND reperage = 10;
```

```
	INSERT INTO travail.reperages_Charlson
		SELECT	BEN_IDT_ANO,
				13
		FROM patients_Diabete_sans
			WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM patients_complications);

quit;

proc delete data = patients_Diabete_sans patients_complications;
run; quit;

* Les patients repérés pour Diabète sans complication et Diabète avec complication => Diabète avec complication;
data patients_Diabete_sans;
	set travail.reperages_Charlson;
	where reperage = 10;
run;

data patients_Diabete_avec;
	set travail.reperages_Charlson;
	where reperage = 13;
run;

proc sql;

	DELETE FROM travail.reperages_Charlson
		WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM patients_Diabete_sans)
			AND BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM patients_Diabete_avec)
			AND reperage = 10;

quit;

proc delete data = patients_Diabete_sans patients_Diabete_avec;
run; quit;

* Les patients repérés pour Cancer et Pathologie métastatique => Pathologie métastatique;
data patients_Cancer;
	set travail.reperages_Charlson;
	where reperage = 14;
run;

data patients_Metastatique;
	set travail.reperages_Charlson;
	where reperage = 16;
run;

proc sql;

	DELETE FROM travail.reperages_Charlson
		WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM patients_Cancer)
			AND BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM patients_Metastatique)
			AND reperage = 14;

quit;

proc delete data = patients_Cancer patients_Metastatique;
run; quit;

* Les patients repérés pour Pathologie hépatique légère et Pathologie hépatique modérée ou sévère => Pathologie hépatique modérée
ou sévère;
data patients_hepatique_leg;
	set travail.reperages_Charlson;
	where reperage = 9;
run;

data patients_hepatique_sev;
	set travail.reperages_Charlson;
	where reperage = 15;
run;

proc sql;
```

```
	DELETE FROM travail.reperages_Charlson
		WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM patients_hepatique_leg)
			AND BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM patients_hepatique_sev)
			AND reperage = 9;

quit;

proc delete data = patients_hepatique_leg patients_hepatique_sev;
run; quit;

*
	**********************************************************************************************************
***********************************;
*	Ajout des repérage des CMA dans la table de population;

%macro maj_Charlson_population(num_CMA=, nom_CMA=);

	proc sql;

		ALTER TABLE res.T_INDI_BPCO_&an_N. ADD &nom_CMA. INT length = 3;

		UPDATE res.T_INDI_BPCO_&an_N.
			SET &nom_CMA. =
				CASE	WHEN BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM
travail.reperages_Charlson WHERE reperage = &num_CMA.)
					THEN 1
					ELSE 0
					END ;

	quit;

%mend maj_Charlson_population;

%maj_Charlson_population(
	num_CMA = 1,
	nom_CMA = Charlson_IDM
	);

%maj_Charlson_population(
	num_CMA = 2,
	nom_CMA = Charlson_Insuf_Cardiaque
	);

%maj_Charlson_population(
	num_CMA = 3,
	nom_CMA = Charlson_Malad_Vasculaire
	);

%maj_Charlson_population(
	num_CMA = 4,
	nom_CMA = Charlson_Malad_CerebroVasc
	);

%maj_Charlson_population(
	num_CMA = 5,
	nom_CMA = Charlson_Demence
	);

%maj_Charlson_population(
	num_CMA = 6,
	nom_CMA = Charlson_Malad_Pulmonaires
	);

%maj_Charlson_population(
	num_CMA = 7,
	nom_CMA = Charlson_Malad_Rhumatisme
	);

%maj_Charlson_population(
	num_CMA = 8,
	nom_CMA = Charlson_Ulcere
```

```
		);

%maj_Charlson_population(
		num_CMA = 9,
		nom_CMA = Charlson_Hepatite
		);

%maj_Charlson_population(
		num_CMA = 10,
		nom_CMA = Charlson_Diabete_SansCompl
		);

%maj_Charlson_population(
		num_CMA = 11,
		nom_CMA = Charlson_Hemiplegie_Para
		);

%maj_Charlson_population(
		num_CMA = 12,
		nom_CMA = Charlson_Malad_Renale
		);

%maj_Charlson_population(
		num_CMA = 13,
		nom_CMA = Charlson_Diabete_Compl
		);

%maj_Charlson_population(
		num_CMA = 14,
		nom_CMA = Charlson_Cancer
		);

%maj_Charlson_population(
		num_CMA = 15,
		nom_CMA = Charlson_Patho_Foie
		);

%maj_Charlson_population(
		num_CMA = 16,
		nom_CMA = Charlson_Tumeur
		);

%maj_Charlson_population(
		num_CMA = 17,
		nom_CMA = Charlson_Malad_VIH
		);

*
		*************************************************************************************************************
***********************************;
*		Calcul du score de Charlson;

proc sql;

		ALTER TABLE res.T_INDI_BPCO_&an_N. ADD score_Charlson_CMA INT length = 3;

		UPDATE res.T_INDI_BPCO_&an_N.
			SET score_Charlson_CMA = SUM
				(CASE	WHEN Charlson_IDM = 1 THEN 0
					WHEN Charlson_IDM = 0 THEN 0
					ELSE .
					END,
				CASE	WHEN Charlson_Insuf_Cardiaque = 1 THEN 2
					WHEN Charlson_Insuf_Cardiaque = 0 THEN 0
					ELSE .
					END,
				CASE	WHEN Charlson_Malad_Vasculaire = 1 THEN 1
					WHEN Charlson_Malad_Vasculaire = 0 THEN 0
					ELSE .
					END,
				CASE	WHEN Charlson_Malad_CerebroVasc = 1 THEN 1
```

```
                                        WHEN Charlson_Malad_CerebroVasc = 0 THEN 0
                                        ELSE .
                                        END,
                        CASE        WHEN Charlson_Demence = 1 THEN 2
                                        WHEN Charlson_Demence = 0 THEN 0
                                        ELSE .
                                        END,
                        CASE        WHEN Charlson_Malad_Pulmonaires = 1 THEN 1
                                        WHEN Charlson_Malad_Pulmonaires = 0 THEN 0
                                        ELSE .
                                        END,
                        CASE        WHEN Charlson_Malad_Rhumatisme = 1 THEN 0
                                        WHEN Charlson_Malad_Rhumatisme = 0 THEN 0
                                        ELSE .
                                        END,
                        CASE        WHEN Charlson_Ulcere = 1 THEN 0
                                        WHEN Charlson_Ulcere = 0 THEN 0
                                        ELSE .
                                        END,
                        CASE        WHEN Charlson_Hepatite = 1 THEN 2
                                        WHEN Charlson_Hepatite = 0 THEN 0
                                        ELSE .
                                        END,
                        CASE        WHEN Charlson_Diabete_SansCompl = 1 THEN 0
                                        WHEN Charlson_Diabete_SansCompl = 0 THEN 0
                                        ELSE .
                                        END,
                        CASE        WHEN Charlson_Hemiplegie_Para = 1 THEN 2
                                        WHEN Charlson_Hemiplegie_Para = 0 THEN 0
                                        ELSE .
                                        END,
                        CASE        WHEN Charlson_Malad_Renale = 1 THEN 1
                                        WHEN Charlson_Malad_Renale = 0 THEN 0
                                        ELSE .
                                        END,
                        CASE        WHEN Charlson_Diabete_Compl = 1 THEN 0
                                        WHEN Charlson_Diabete_Compl = 0 THEN 0
                                        ELSE .
                                        END,
                        CASE        WHEN Charlson_Cancer = 1 THEN 2
                                        WHEN Charlson_Cancer = 0 THEN 0
                                        ELSE .
                                        END,
                        CASE        WHEN Charlson_Patho_Foie = 1 THEN 3
                                        WHEN Charlson_Patho_Foie = 0 THEN 0
                                        ELSE .
                                        END,
                        CASE        WHEN Charlson_Tumeur = 1 THEN 11
                                        WHEN Charlson_Tumeur = 0 THEN 0
                                        ELSE .
                                        END,
                        CASE        WHEN Charlson_Malad_VIH = 1 THEN 1
                                        WHEN Charlson_Malad_VIH = 0 THEN 0
                                        ELSE .
                                        END
                        ) ;

    * Ajout du score pondéré par l âge;
    ALTER TABLE res.T_INDI_BPCO_&an_N. ADD Charlson INT length = 3;

    UPDATE res.T_INDI_BPCO_&an_N.
                SET Charlson =
                        CASE        WHEN Age < 50 then score_Charlson_CMA
                                        WHEN Age BETWEEN 50 AND 59 THEN COALESCE(score_Charlson_CMA, 0) + 1
                                        WHEN Age BETWEEN 60 AND 69 THEN COALESCE(score_Charlson_CMA, 0) + 2
                                        WHEN Age BETWEEN 70 AND 79 THEN COALESCE(score_Charlson_CMA, 0) + 3
                                        WHEN Age BETWEEN 80 AND 89 THEN COALESCE(score_Charlson_CMA, 0) + 4
                                        WHEN Age >= 90 THEN COALESCE(score_Charlson_CMA, 0) + 5
                                        END ;

    * Suppression du score de Charlson non pondéré;
```

```
		ALTER TABLE res.T_INDI_BPCO_&an_N. DROP score_Charlson_CMA;

quit;

*
		*************************************************************************************************************
************************************;
*		Vérifications;

* On ne passe pas par la macro car trop de variables;
proc sql;

		CREATE TABLE verif._0&tmp_num_tab._Charlson AS
			SELECT
				Charlson,
				Charlson_IDM,
				Charlson_Insuf_Cardiaque,
				Charlson_Malad_Vasculaire,
				Charlson_Malad_CerebroVasc,
				Charlson_Demence,
				Charlson_Malad_Pulmonaires,
				Charlson_Malad_Rhumatisme,
				Charlson_Ulcere,
				Charlson_Hepatite,
				Charlson_Diabete_SansCompl,
				Charlson_Hemiplegie_Para,
				Charlson_Malad_Renale,
				Charlson_Diabete_Compl,
				Charlson_Cancer,
				Charlson_Patho_Foie,
				Charlson_Tumeur,
				Charlson_Malad_VIH,
				COUNT(*) AS Frequency
			FROM res.T_INDI_BPCO_&an_N.
			GROUP BY Charlson, Charlson_IDM, Charlson_Insuf_Cardiaque, Charlson_Malad_Vasculaire,
Charlson_Malad_CerebroVasc, Charlson_Demence,
				Charlson_Malad_Pulmonaires, Charlson_Malad_Rhumatisme, Charlson_Ulcere, Charlson_Hepatite,
Charlson_Diabete_SansCompl, Charlson_Hemiplegie_Para,
				Charlson_Malad_Renale, Charlson_Diabete_Compl, Charlson_Cancer, Charlson_Patho_Foie,
Charlson_Tumeur, Charlson_Malad_VIH;

quit;

proc sql noprint; SELECT SUM(Frequency) INTO: nb_tot FROM verif._0&tmp_num_tab._Charlson; quit;

data verif._0&tmp_num_tab._Charlson;
		set verif._0&tmp_num_tab._Charlson;
		length percent 4.;
		format percent nlpct7.1 Frequency commafr_0ch.;
		percent = Frequency/&nb_tot.;
run;

%let tmp_num_tab = %sysevalf(&tmp_num_tab. + 1);

*
		*************************************************************************************************************
************************************;
*		On supprime les tables temporaires;

proc delete data = travail.reperages_Charlson travail.reperage_ATC_Charlson;
run; quit;
```

3.4.5 05_Table_T_INDI_BPCO.sas

```
/*
	****************************************************************************************************
***********************************		*/
/*
																								*/
/*			Table finale - Ajout de formats et de labels
										*/
/*
																								*/
/*
	****************************************************************************************************
***********************************		*/

*
	****************************************************************************************************
***********************************;
*	Création de la table finale;

data res.T_INDI_BPCO_&an_N.;
	set res.T_INDI_BPCO_&an_N. (keep = Ben_Idt_Ano Age Ben_Sex_Cod Ben_Dcd_Dte BEN_RES_DPT BEN_RES_COM
RGM_GRG_COD ALD ALD_Chapitre_1
				ALD_Chapitre_2 ALD_Chapitre_3 ALD_Chapitre_4 ALD_Chapitre_5 ALD_Chapitre_6 ALD_Chapitre_7
ALD_Chapitre_8 ALD_Chapitre_9
				ALD_Chapitre_10 ALD_Chapitre_11 ALD_Chapitre_12 ALD_Chapitre_13 ALD_Chapitre_14
ALD_Chapitre_15 ALD_Chapitre_16 ALD_Chapitre_17
				ALD_Chapitre_18 ALD_Chapitre_19 ALD_Chapitre_20 ALD_Chapitre_21 ALD_Chapitre_22
ALD_BPCO_&an_N. Nb_Broncho_LDA Charlson
				Charlson_IDM Charlson_Insuf_Cardiaque Charlson_Malad_Vasculaire Charlson_Malad_CerebroVasc
Charlson_Demence Charlson_Malad_Pulmonaires
				Charlson_Malad_Rhumatisme Charlson_Ulcere Charlson_Hepatite Charlson_Diabete_Compl
Charlson_Diabete_SansCompl Charlson_Hemiplegie_Para
				Charlson_Malad_Renale Charlson_Cancer Charlson_Patho_Foie Charlson_Tumeur
Charlson_Malad_VIH FDEP13 code_insee quintile_pop
				Ben_Cmu_Top Nb_Sej_Exacerbation
				Indicateur_01 Indicateur_02a Indicateur_02b Indicateur_03 Indicateur_04 Indicateur_05 Indicateur_06
Indicateur_07 Indicateur_09 nb_ATB
				ALD_BPCO_Septembre ALD_BPCO_decembre nb_hospit_BPCO nb_hospit_BPCO_Ind02 nb_tabac);
	label
		BEN_IDT_ANO = "Numéro d'individu"
		Age = "Age du patient au 1er janvier &Annee_N."
		Ben_Sex_Cod = "Sexe du patient"
		BEN_DCD_DTE = "Date de décès"
		BEN_RES_DPT = "Département de résidence du patient"
		BEN_RES_COM = "Commune de résidence"
		RGM_GRG_COD = "Grand régime d'affiliation"
		ALD = "Au moins une ALD active en &Annee_N."
		ALD_Chapitre_1 = "ALD A00-B99 : Certaines maladies infectieuses et parasitaires"
		ALD_Chapitre_2 = "ALD C00-D48 : Tumeurs"
		ALD_Chapitre_3 = "ALD D50-D89 : Maladies du sang et des organes hématopoiétiques et certains troubles du
système immunitaire"
		ALD_Chapitre_4 = "ALD E00-E90 : Maladies endocriniennes, nutritionnelles et métaboliques"
		ALD_Chapitre_5 = "ALD F00-F99 : Troubles mentaux et du comportement"
		ALD_Chapitre_6 = "ALD G00-G99 : Maladies du système nerveux"
		ALD_Chapitre_7 = "ALD H00-H59 : Maladies de l'œil et de ses annexes"
		ALD_Chapitre_8 = "ALD H60-H95 : Maladies de l'oreille et de l'apophyse mastoïde"
		ALD_Chapitre_9 = "ALD I00-I99 : Maladies de l'appareil circulatoire"
		ALD_Chapitre_10 = "ALD J00-J99 : Maladies de l'appareil respiratoire"
		ALD_Chapitre_11 = "ALD K00-K93 : Maladies de l'appareil digestif"
		ALD_Chapitre_12 = "ALD L00-L99 : Maladies de la peau et du tissu cellulaire sous-cutané"
		ALD_Chapitre_13 = "ALD M00-M99 : Maladies du système ostéo-articulaire, des muscles et du tissu conjonctif"
		ALD_Chapitre_14 = "ALD N00-N99 : Maladies de l'appareil génito-urinaire"
		ALD_Chapitre_15 = "ALD O00-O99 : Grossesse, accouchement et puerpéralité"
		ALD_Chapitre_16 = "ALD P00-P96 : Certaines affections dont l'origine se situe dans la période périnatale"
		ALD_Chapitre_17 = "ALD Q00-Q99 : Malformations congénitales et anomalies chromosomiques"
```

```
                        ALD_Chapitre_18 = "ALD R00-R99 : Symptômes, signes et résultats anormaux d'examens cliniques et de
laboratoire, non classés ailleurs"
                        ALD_Chapitre_19 = "ALD S00-T98 : Lésions traumatiques, empoisonnements et certaines autres conséquences de
causes externes"
                        ALD_Chapitre_20 = "ALD V01-Y98 : Causes externes de morbidité et de mortalité"
                        ALD_Chapitre_21 = "ALD Z00-Z99 : Facteurs influant sur l'état de santé et motifs de recours aux services de santé"
                        ALD_Chapitre_22 = "ALD U00-U99 : Codes d'utilisation particulière"
                        ALD_BPCO_&an_N. = "Patient ayant une ALD BPCO active en &Annee_N."
                        Nb_Broncho_LDA = "Nombre de délivrances remboursées de bronchodilatateur anticholinergique ou Bêta-2
longue durée d'action en &Annee_N."
                        Charlson = "Indice de Charlson en &Annee_N."
                        Charlson_IDM = "Composante de l'indice de Charlson - Infarctus du myocarde"
                        Charlson_Insuf_Cardiaque = "Composante de l'indice de Charlson - Insuffisance cardiaque congestive"
                        Charlson_Malad_Vasculaire = "Composante de l'indice de Charlson - Maladie vasculaire périphérique"
                        Charlson_Malad_CerebroVasc = "Composante de l'indice de Charlson - Maladies cérébrovasculaires"
                        Charlson_Demence = "Composante de l'indice de Charlson - Démence"
                        Charlson_Malad_Pulmonaires = "Composante de l'indice de Charlson - Maladies pulmonaires chroniques"
                        Charlson_Malad_Rhumatisme = "Composante de l'indice de Charlson - Maladies rhumatismales"
                        Charlson_Ulcere = "Composante de l'indice de Charlson - Ulcère de l'estomac"
                        Charlson_Hepatite = "Composante de l'indice de Charlson - Hépatite"
                        Charlson_Diabete_Compl = "Composante de l'indice de Charlson - Diabète avec complications chroniques"
                        Charlson_Diabete_SansCompl = "Composante de l'indice de Charlson - Diabète sans complications chroniques"
                        Charlson_Hemiplegie_Para = "Composante de l'indice de Charlson - Hémiplégies ou paraplégies"
                        Charlson_Malad_Renale = "Composante de l'indice de Charlson - Maladies rénales"
                        Charlson_Cancer = "Composante de l'indice de Charlson - Cancers"
                        Charlson_Patho_Foie = "Composante de l'indice de Charlson - Pathologies du foie de sévère à modérer"
                        Charlson_Tumeur = "Composante de l'indice de Charlson - Tumeurs malignes métastasique"
                        Charlson_Malad_VIH = "Composante de l'indice de Charlson - Maladies dues au VIH"
                        FDEP13 = "Indice de défavorisation sociale"
                        code_insee = "Code INSEE"
                        quintile_pop = "Répartition en quintile des communes pour l'indice de défavorisation 2013"
                        BEN_CMU_TOP = "Patient bénéficiant de la CMUC en &Annee_N."
                        Nb_Sej_Exacerbation = "Nombre de séjours pour exacerbation en &Annee_N."
                        ;
        format BPCO_: ALD: Charlson_: BEN_CMU_TOP f_oui_non. BEN_DCD_DTE date9.;
run;

* Vérif;
proc sql;

        SELECT COUNT(*) AS nb_lignes, COUNT(DISTINCT BEN_IDT_ANO) AS nb_patients
        FROM res.T_INDI_BPCO_&an_N.;

quit;

*
        ************************************************************************************************************
***********************************;
*       Création d une table de correspondance et d'une table de population avec les indicateur;

data travail.corresp_id_patient;
        set orauser.corresp_id_patient;
run;
```

3.5 Tables communes

3.5.1 01_Sejours_pour_exacerbation_de_BPCO.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*

*/
/*		Séjours pour exacerbation de BPCO

*/
/*

*/
/*
	*****************************************************************************************************************
**********************************	*/

*
	*****************************************************************************************************************
**********************************;
*	On récupère les séjours pour exacerbation de BPCO;

%macro exacerb_BPCO(annee = );

	%suppr_table(
		lib = orauser,
		table = diag_codes_CIM_exa_BPCO
		);

	data orauser.diag_codes_CIM_exa_BPCO;
		set orauser.diag_codes_CIM_BPCO (where = (reperage in (16, 18, 17, 19, 23, 25, 24, 20, 21, 22) or (reperage = 11
and flag_pneumo = 1)));
	run;

	* On appelle la macro pour le repérage dans le PMSI MCO;
	%extract_CIM10_PMSI(
		annee_deb = &annee.,
		annee_fin = &annee.,
		HAD_DP = 0,
		HAD_DAS = 0,
		HAD_MPP = 0,
		HAD_MPA = 0,
		MCO_DP = 1,
		MCO_DR = 0,
		MCO_DAS = 1,
		MCO_DP_UM = 0,
		MCO_DR_UM = 0,
		SSR_FP = 0,
		SSR_MPP = 0,
		SSR_AE = 0,
		SSR_DAS = 0,
		tbl_out = travail.reperage_CIM10_exa_BPCO,
		tbl_codes = diag_codes_CIM_exa_BPCO,
		tbl_patients = corresp_id_patient
		);

	proc sql;

		DELETE FROM travail.reperage_CIM10_exa_BPCO
		WHERE SUBSTR(GRG_GHM, 1, 2) = "28" OR SEJ_NBJ = 0;

	quit;

	proc sort data = travail.reperage_CIM10_exa_BPCO;
```

```
		by ETA_NUM RSA_NUM annee;
	run;

	*
	************************************************************************************************************
*********************************;
	*	On récupère les séjours avec un code diagnostic de BPCO a été codé en DP du séjour;

	data sejours_exacerbation1;
		set travail.reperage_CIM10_exa_BPCO (where = (reperage = 16 and table = "B" and variable = "DGN_PAL"));
		length exacerbation_BPCO 3.;
		exacerbation_BPCO = 1;
	run;

	*
	************************************************************************************************************
*********************************;
	*	On récupère les séjours avec un code diagnostic de BPCO a été codé en DAS du séjour;

	proc sort data = travail.reperage_CIM10_exa_BPCO (keep = ETA_NUM RSA_NUM annee reperage variable where =
(reperage = 16 and variable = "ASS_DGN"))
			out = reperage_CIM10_exa_BPCO nodupkey;
		by ETA_NUM RSA_NUM annee;
	run;

	*
	************************************************************************************************************
*********************************;
	*	On récupère les séjours avec un code de pneumopathie lobaire en DP et un diagnostic de BPCO en DAS;

	data sejours_exacerbation2;
		merge	travail.reperage_CIM10_exa_BPCO (in = a where = (reperage = 18 and table = "B" and variable =
"DGN_PAL"))
				reperage_CIM10_exa_BPCO (in = b);
		by ETA_NUM RSA_NUM annee;
		if a and b;
		length exacerbation_BPCO 3.;
		exacerbation_BPCO = 2;
	run;

	*
	************************************************************************************************************
*********************************;
	*	On récupère les séjours avec un code d insuffisance respiratoire aigüe en DP avec un diagnostic de BPCO en DAS;

	data sejours_exacerbation3;
		merge	travail.reperage_CIM10_exa_BPCO (in = a where = (reperage = 17 and table = "B" and variable =
"DGN_PAL"))
				reperage_CIM10_exa_BPCO (in = b);
		by ETA_NUM RSA_NUM annee;
		if a and b;
		length exacerbation_BPCO 3.;
		exacerbation_BPCO = 3;
	run;

	*
	************************************************************************************************************
*********************************;
	*	On récupère les séjours avec un code de grippe en DP avec un diagnostic de BPCO en DAS;

	data sejours_exacerbation4;
		merge	travail.reperage_CIM10_exa_BPCO (in = a where = (reperage = 19 and table = "B" and variable =
"DGN_PAL"))
				reperage_CIM10_exa_BPCO (in = b);
		by ETA_NUM RSA_NUM annee;
		if a and b;
		length exacerbation_BPCO 3.;
		exacerbation_BPCO = 4;
	run;
```

```
	*
	*****************************************************************************************************************************;
	*	On récupère les séjours avec un code d infection des voies aériennes en DP avec un diagnostic de BPCO en DAS;

	data sejours_exacerbation5;
		merge	travail.reperage_CIM10_exa_BPCO (in = a where = (reperage = 23 and table = "B" and variable =
"DGN_PAL"))
				reperage_CIM10_exa_BPCO (in = b);
		by ETA_NUM RSA_NUM annee;
		if a and b;
		length exacerbation_BPCO 3.;
		exacerbation_BPCO = 5;
	run;

	*
	*****************************************************************************************************************************;
	*	On récupère les séjours avec un code de bronchopneumopathie en DP avec un diagnostic de BPCO en DAS;

	data sejours_exacerbation6;
		merge	travail.reperage_CIM10_exa_BPCO (in = a where = (reperage = 25 and table = "B" and variable =
"DGN_PAL"))
				reperage_CIM10_exa_BPCO (in = b);
		by ETA_NUM RSA_NUM annee;
		if a and b;
		length exacerbation_BPCO 3.;
		exacerbation_BPCO = 6;
	run;

	*
	*****************************************************************************************************************************;
	*	On récupère les séjours avec un code de pneumonie en DP avec un diagnostic de BPCO en DAS;

	data sejours_exacerbation7;
		merge	travail.reperage_CIM10_exa_BPCO (in = a where = (reperage = 24 and table = "B" and variable =
"DGN_PAL"))
				reperage_CIM10_exa_BPCO (in = b);
		by ETA_NUM RSA_NUM annee;
		if a and b;
		length exacerbation_BPCO 3.;
		exacerbation_BPCO = 7;
	run;

	*
	*****************************************************************************************************************************;
	*	On récupère les séjours avec un code d embolie pulmonaire en DP avec un diagnostic de BPCO en DAS;

	data sejours_exacerbation8;
		merge	travail.reperage_CIM10_exa_BPCO (in = a where = (reperage = 20 and table = "B" and variable =
"DGN_PAL"))
				reperage_CIM10_exa_BPCO (in = b);
		by ETA_NUM RSA_NUM annee;
		if a and b;
		length exacerbation_BPCO 3.;
		exacerbation_BPCO = 8;
	run;

	*
	*****************************************************************************************************************************;
	*	On récupère les séjours avec un code d insuffisance cardiaque aigue en DP avec un diagnostic de BPCO en DAS;

	data sejours_exacerbation9;
		merge	travail.reperage_CIM10_exa_BPCO (in = a where = (reperage = 21 and table = "B" and variable =
"DGN_PAL"))
				reperage_CIM10_exa_BPCO (in = b);
		by ETA_NUM RSA_NUM annee;
		if a and b;
```

```
                        length exacerbation_BPCO 3.;
                        exacerbation_BPCO = 9;
            run;

            *
            ************************************************************************************************
*********************************;
            *           On récupère les séjours avec un code d œdème aigu du poumon en DP avec un diagnostic de BPCO en DAS;

            data sejours_exacerbation10;
                        merge       travail.reperage_CIM10_exa_BPCO (in = a where = (reperage = 22 and table = "B" and variable =
"DGN_PAL"))
                                                reperage_CIM10_exa_BPCO (in = b);
                        by ETA_NUM RSA_NUM annee;
                        if a and b;
                        length exacerbation_BPCO 3.;
                        exacerbation_BPCO = 10;
            run;

            *
            ************************************************************************************************
*********************************;
            *           On récupère les séjours avec un code de pneumothorax en DP avec un diagnostic de BPCO en DAS;

            data sejours_exacerbation11;
                        merge       travail.reperage_CIM10_exa_BPCO (in = a where = (reperage = 11 and table = "B" and variable =
"DGN_PAL"))
                                                reperage_CIM10_exa_BPCO (in = b);
                        by ETA_NUM RSA_NUM annee;
                        if a and b;
                        length exacerbation_BPCO 3.;
                        exacerbation_BPCO = 11;
            run;

            *
            ************************************************************************************************
*********************************;
            *           On concatène toutes les tables et on compte le nombre de séjours par patient;

            data travail.sejours_exacerbation_&annee.;
                        set sejours_exacerbation1-sejours_exacerbation11;
                        length id_sejour $30;
                        id_sejour = ETA_NUM||"_"||RSA_NUM||"_"||put(Annee, 4.);
            run;

            data travail.sejours_cible_exacerbation_&annee.;
                        set travail.sejours_exacerbation_&annee.;
            run;

%mend exacerb_BPCO;

%exacerb_BPCO(annee = &annee_1N.);
%exacerb_BPCO(annee = &annee_N.);
%exacerb_BPCO(annee = &annee_N1.);

*
            ************************************************************************************************
*********************************;
*           On exclut les séjours qui se chevauchent (= transferts) - Pour cela, on repère les séjours de l année précédente et suivante;

data sejours_exacerbation_all;
            format variable $11.;
            set travail.sejours_cible_exacerbation_&annee_1N. travail.sejours_cible_exacerbation_&annee_N.
travail.sejours_cible_exacerbation_&annee_N1.;
run;

*
            ************************************************************************************************
*********************************;
*           Vérifications;
```

```
%proc_freq(
		in_tbl = sejours_exacerbation_all,
		out_tbl = _sej_exacerb,
		list_var_in = ANNEE*DOMAINE*TABLE*REPERAGE*VARIABLE*DGN_PAL,
		list_var_out = ANNEE DOMAINE TABLE2 REPERAGE VARIABLE DGN_PAL Frequency
		);

data verif;
		set sejours_exacerbation_all;
		annee_fin = year(date_fin);
run;

%proc_freq(
		in_tbl = verif,
		out_tbl = _sej_exacerb_annee_fin,
		list_var_in = annee_fin,
		list_var_out = annee_fin Frequency
		);

proc delete data = verif;
run; quit;

data sejours_exacerbation_all_copie;
		format variable $11.;
		set travail.sejours_cible_exacerbation_&annee_1N. travail.sejours_cible_exacerbation_&annee_N.
travail.sejours_cible_exacerbation_&annee_N1.;
run;

proc sql;

		CREATE TABLE sejours_transferts AS
			SELECT DISTINCT
				a.BEN_IDT_ANO,
				a.id_sejour,
				b.id_sejour AS index_transfert,
				a.date_debut,
				a.date_fin,
				b.date_debut as date_debut_transfert,
				b.date_fin as date_fin_transfert
			FROM sejours_exacerbation_all_copie a
				LEFT JOIN sejours_exacerbation_all b
					ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
			WHERE b.date_debut >= a.date_debut AND a.id_sejour NE b.id_sejour;

quit;

data sejours_transferts;
		set sejours_transferts;
		if (date_debut_transfert < date_fin and date_fin_transfert < date_fin)
				or (date_debut_transfert < date_fin and date_fin_transfert > date_fin)
				or (date_debut_transfert < date_fin and date_fin_transfert = date_fin) then
			output;
run;

proc sql;

		SELECT COUNT(DISTINCT id_sejour) AS nb_sejours, COUNT(DISTINCT index_transfert) AS nb_transferts
		FROM sejours_transferts;

quit;

* On supprime les 2 séjours de les tables travail.sejours_exacerbation_XXXX;
proc sql;

		DELETE FROM travail.sejours_cible_exacerbation_&annee_1N.
		WHERE id_sejour IN (SELECT id_sejour FROM sejours_transferts)
			OR id_sejour IN (SELECT index_transfert FROM sejours_transferts);

		DELETE FROM travail.sejours_cible_exacerbation_&annee_N.
		WHERE id_sejour IN (SELECT id_sejour FROM sejours_transferts)
			OR id_sejour IN (SELECT index_transfert FROM sejours_transferts);
```

```
        DELETE FROM travail.sejours_cible_exacerbation_&annee_N1.
        WHERE id_sejour IN (SELECT id_sejour FROM sejours_transferts)
                OR id_sejour IN (SELECT index_transfert FROM sejours_transferts);

quit;

proc datasets library = work memtype = data ;
        delete sejours: reperage_CIM10_exa_BPCO;
run; quit;

proc delete data = travail.reperage_CIM10_exa_BPCO;
run; quit;
```

3.5.2 02_Delivrances_remboursees_de_BDLA.sas

```
/*
	*************************************************************************************************************
**********************************	*/
/*
                                                                                                                */
/*			Délivrance remboursée de BDLA
                                                                                                */
/*
                                                                                                                */
/*
	*************************************************************************************************************
**********************************	*/

%suppr_table(
	lib = orauser,
	table = codes_ATC
	);

data orauser.codes_ATC;
	set orauser.codes_ATC_BPCO (where = (reperage in (46, 45, 47, 48, 49)));
run;

%suppr_table(
	lib = travail,
	table = reperage_BDLA_&Annee_2N._&Annee_N1.
	);

* Dans le DCIR - Codes CIP;
%extract_CIP(
	annee_deb = &Annee_2N.,
	annee_fin = &Annee_N1.,
	tbl_out = travail.reperage_BDLA_&Annee_2N._&Annee_N1.,
	tbl_codes = codes_ATC,
	tbl_patients = corresp_id_patient
	);

proc delete data = orauser.codes_ATC;
run; quit;

*
	*************************************************************************************************************
**********************************;
*	Vérifications;

data verif;
	set travail.reperage_BDLA_&Annee_2N._&Annee_N1.;
	annee_soin = year(date_debut);
run;

%proc_freq(
	in_tbl = verif,
	out_tbl = _BDLA,
	list_var_in = annee_soin*PHA_ATC_C07*PHA_CND_TOP,
	list_var_out = annee_soin PHA_ATC_C07 PHA_CND_TOP Frequency
	);

proc delete data = verif;
run; quit;
```

3.5.3 03_Contacts_medecin_generaliste_ou_traitant.sas

```
/*
	*************************************************************************************************************
**********************************	*/
/*
                                                                                                                */
/*			Contacts avec un médecin généraliste ou médecin traitant

				*/
/*
                                                                                                                */
/*
	*************************************************************************************************************
**********************************	*/

*
	*************************************************************************************************************
**********************************;
*	Table contenant les patients des indicateurs 4, 5 et 6 uniquement;

data indicateurs_04et05 (keep = BEN_IDT_ANO);
	set pop.indicateurs_&an_N. (where = (indicateur_04 = 1 or indicateur_05 = 1 or indicateur_09 = 1));
run;

%suppr_table(
	lib = orauser,
	table = corresp_MT_MG
	);

proc sql;

	CREATE TABLE orauser.corresp_MT_MG AS
		SELECT *
		FROM travail.corresp_id_patient
		WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM indicateurs_04et05);

quit;

proc delete data = indicateurs_04et05;
run; quit;

*
	*************************************************************************************************************
**********************************;
*	Contact avec le médecin généraliste ou le médecin traitant;

* MG dans le PMSI;
%suppr_table(
	lib = orauser,
	table = actes_MG
	);

%suppr_table(
	lib = travail,
	table = contact_MG_PMSI_&Annee_N._&Annee_N1.
	);

data orauser.actes_MG;
	set orauser.codes_actes_BPCO (where = (reperage = 33));
run;

%extract_ACT_COD_PMSI(
	annee_deb = &Annee_N.,
	annee_fin = &Annee_N1.,
	MCO = 1,
	SSR = 1,
```

```
	tbl_out = travail.contact_MG_PMSI_&Annee_N._&Annee_N1.,
	tbl_codes = actes_MG,
	tbl_patients = corresp_MT_MG
	);

data travail.contact_MG_PMSI_&Annee_N._&Annee_N1.;
	set travail.contact_MG_PMSI_&Annee_N._&Annee_N1. (where = (index(type, "ACE") >= 1 and EXE_SPE in ("01", "22",
"23")));
run;

*
	**********************************************************************************************************
**********************************;
*	Vérifications;

%proc_freq(
	in_tbl = travail.contact_MG_PMSI_&Annee_N._&Annee_N1.,
	out_tbl = _MG_PMSI,
	list_var_in = Annee*Domaine*Table*EXE_SPE*ACT_COD,
	list_var_out = Annee Domaine Table2 EXE_SPE ACT_COD Frequency
	);

* MG et MT dans le DCIR;
%suppr_table(
	lib = travail,
	table = contact_MT_&Annee_N._&Annee_N1.
	);

%extract_MT(
	annee_deb = &Annee_N.,
	annee_fin = &Annee_N1.,
	tbl_out = travail.contact_MT_&Annee_N._&Annee_N1.,
	tbl_patients = corresp_MT_MG
	);

*
	**********************************************************************************************************
**********************************;
*	Vérifications;

data verif;
	set travail.contact_MT_&Annee_N._&Annee_N1.;
	annee = year(date_debut);
run;

%proc_freq(
	in_tbl = verif,
	out_tbl = _MG_DCIR,
	list_var_in = Annee*PSE_SPE_COD*BSE_PRS_NAT,
	list_var_out = Annee PSE_SPE_COD BSE_PRS_NAT Frequency
	);

%proc_freq(
	in_tbl = verif,
	out_tbl = _MG_DCIR_qte,
	list_var_in = quantite,
	list_var_out = quantite Frequency
	);

proc delete data = verif;
run; quit;
```

3.5.4 04_Contacs_pneumologues.sas

```
	/*
	*************************************************************************************************************
**********************************	*/
	/*


	*/
	/*			Contacts avec un pneumologue

								*/
	/*


	*/
	/*
	*************************************************************************************************************
**********************************	*/

	* Pneumologue dans le DCIR;
	%suppr_table(
		lib = orauser,
		table = presta_pneumo
		);

	data orauser.presta_pneumo;
		set orauser.codes_presta_BPCO (where = (reperage = 32));
	run;

	%extract_PRS_NAT_REF(
		annee_deb = &Annee_N.,
		annee_fin = &Annee_N1.,
		sel_spe = AND PSE_SPE_COD = 13,
		tbl_out = pneumo_sdv,
		tbl_codes = presta_pneumo,
		tbl_patients = corresp_MT_MG,
		regul = 1
		);

	* Pneumologue dans le PMSI;
	%suppr_table(
		lib = orauser,
		table = actes_pneumo
		);

	data orauser.actes_pneumo;
		set orauser.codes_actes_BPCO (where = (reperage = 32));
	run;

	%extract_ACT_COD_PMSI(
		annee_deb = &Annee_N.,
		annee_fin = &Annee_N1.,
		MCO = 1,
		SSR = 1,
		tbl_out = pneumo_PMSI,
		tbl_codes = actes_pneumo,
		tbl_patients = corresp_MT_MG
		);

	%suppr_table(
		lib = travail,
		table = contact_pneumo_&Annee_N._&Annee_N1.
		);

	data travail.contact_pneumo_&Annee_N._&Annee_N1.;
		format type $12.;
		set	pneumo_sdv
			pneumo_PMSI (where = (EXE_SPE = "13" and index(type, "ACE") >= 1));
```

```
	run;

	proc delete data = pneumo_sdv pneumo_PMSI;
	run; quit;

	*
	**********************************************************************************************************
**********************************;
	*		Vérifications;

	data verif;
		set travail.contact_pneumo_&Annee_N._&Annee_N1.;
		if type = "DCIR" then
			annee = year(date_debut);
	run;

	%proc_freq(
		in_tbl = verif,
		out_tbl = _pneumo,
		list_var_in = Annee*TYPE*EXE_SPE*PSE_SPE_COD*BSE_PRS_NAT*ACT_COD,
		list_var_out = Annee TYPE EXE_SPE PSE_SPE_COD BSE_PRS_NAT ACT_COD Frequency
		);

	proc delete data = verif;
	run; quit;
```

3.5.5 05_Rehabilitation_respiratoire.sas

```
	/*
	****************************************************************************************************
**********************************	*/
	/*


	*/
	/*		Réhabilitation respiratoire

									*/
	/*


	*/
	/*
	****************************************************************************************************
**********************************	*/

	*
	****************************************************************************************************
**********************************	;
	*	1.	Réadaptation respiratoire réalisée lors d'un séjour en SSR : les informations sont recherchées dans le
PMSI SSR
	*		•	qui a commencé au plus tard dans les 3 mois (date index incluse) suivant la  date
de fin du séjour (date index) pour
	*		exacerbation en MCO

	*		•	quel que soit la durée du séjour en SSR

	*		•	et quel que soit le type d'hospitalisation (HC, HDJ, séance) en SSR

		;
	*
	****************************************************************************************************
**********************************	;

	%extract_sej_PMSI(
		annee_deb = &Annee_1N.,
		annee_fin = &Annee_N1.,
		HAD = 0,
		MCO = 0,
		SSR = 1,
		RIP = 0,
		tbl_out = travail.sejours_SSR_&Annee_1N._&Annee_N1.,
		tbl_patients = corresp_id_patient
		);

	data travail.sejours_SSR_&Annee_1N._&Annee_N1.;
		set travail.sejours_SSR_&Annee_1N._&Annee_N1.;
		if date_fin = . then
			date_fin = mdy(12, 31, annee);
	run;

	proc sort data = travail.sejours_SSR_&Annee_1N._&Annee_N1.;
		by id_sejour;
	run;

	* ********** Codes GME ********** ;

	%suppr_table(
		lib = orauser,
		table = codes_GME
		);

	data orauser.codes_GME;
```

```
		set orauser.codes_GME_BPCO (where = (reperage = 7));
run;

%extract_GME_PMSI(
		annee_deb = &Annee_1N.,
		annee_fin = &Annee_N1.,
		tbl_out = reperage_CM04_&Annee_1N._&Annee_N1.,
		tbl_codes = codes_GME,
		tbl_patients = corresp_id_patient
		);

data reperage_CM04_&Annee_1N._&Annee_N1.;
		set reperage_CM04_&Annee_1N._&Annee_N1.;
		id_sejour = ETA_NUM||"_"||RHA_NUM||"_"||put(annee, 4.);
run;

proc sql undo_policy = none;

		CREATE TABLE reperage_CM04_&Annee_1N._&Annee_N1. AS
			SELECT
				id_sejour,
				MAX(CASE WHEN CODE_GME = "04" THEN 1
									ELSE .
									END) AS CM_04 length = 3
			FROM reperage_CM04_&Annee_1N._&Annee_N1.
			GROUP BY id_sejour
			ORDER BY id_sejour;

quit;

* ********** Codes CCAM ********** ;
%suppr_table(
		lib = orauser,
		table = codes_CCAM
		);

data orauser.codes_CCAM;
		set orauser.codes_CCAM_BPCO (where = (reperage = 7));
run;

%extract_CCAM_PMSI(
		annee_deb = &Annee_1N.,
		annee_fin = &Annee_N1.,
		HAD = 0, MCO = 0, RIP = 0, SSR = 1,
		tbl_out = reperage_CCAM,
		tbl_codes = codes_CCAM,
		tbl_patients = corresp_id_patient
		);

proc delete data = orauser.codes_CCAM;
run; quit;

* ********** Codes CSARR ********** ;
%suppr_table(
		lib = orauser,
		table = codes_CSARR
		);

data orauser.codes_CSARR;
		set orauser.codes_CSARR_BPCO (where = (reperage = 7));
run;

%extract_CSARR_PMSI(
		annee_deb = &Annee_1N.,
		annee_fin = &Annee_N1.,
		tbl_out = reperage_CSARR,
		tbl_codes = codes_CSARR,
		tbl_patients = corresp_id_patient
		);

proc delete data = orauser.codes_CSARR;
```

```
run; quit;

* On repère pour info les séjours actes CCAM ou CSARR de la liste, peu importe la CM et le type UM;
data reperage_CCAM_CSARR;
	length type_UM $2.;
	set reperage_CSARR reperage_CCAM;
	where type = "PMSI séjours";
	id_sejour = ETA_NUM||"_"||RHA_NUM||"_"||put(annee, 4.);
run;

* RR réalisée en cours d hospitalisation : on flag si présence d au moins un acte CCAM ou au moins un acte CSARR;

proc sql undo_policy = none;

	CREATE TABLE reperage_CCAM_CSARR AS
		SELECT
			id_sejour,
			MAX(CASE WHEN code_ccam NE "" THEN 1
								ELSE .
								END) AS CCAM length = 3,
			MAX(CASE WHEN code_csarr NE "" THEN 1
								ELSE .
								END) AS CSARR length = 3
		FROM reperage_CCAM_CSARR
		GROUP BY id_sejour
		ORDER BY id_sejour;

quit;

data travail.RR_sejours_SSR_&Annee_1N._&Annee_N1.;
	merge	travail.sejours_SSR_&Annee_1N._&Annee_N1. (in = a)
			reperage_CM04_&Annee_1N._&Annee_N1.
			reperage_CCAM_CSARR;
	by id_sejour;
	if a;
run;

proc delete data = reperage_CM04_&Annee_1N._&Annee_N1. reperage_CCAM_CSARR reperage_CSARR reperage_CCAM;
run; quit;


*
**********************************************************************************************************
**********************************;
*	Vérifications;

* On ne passe pas par la macro car trop de variables;
proc sql;

	CREATE TABLE verif._0&tmp_num_tab._CCAM_CSARR AS
		SELECT
			Annee,
			Type,
			Domaine,
			type_um,
			Ccam,
			Csarr,
			CM_04,
			COUNT(*) AS Frequency
		FROM travail.RR_sejours_SSR_&Annee_1N._&Annee_N1.
		GROUP BY Annee, Type, Domaine, type_um, ccam, csarr, CM_04;

quit;

proc sql noprint; SELECT SUM(Frequency) INTO: nb_tot FROM verif._0&tmp_num_tab._CCAM_CSARR; quit;

data verif._0&tmp_num_tab._CCAM_CSARR;
	set verif._0&tmp_num_tab._CCAM_CSARR;
	length percent 4.;
	format percent nlpct7.1 Frequency commafr_0ch.;
	percent = Frequency/&nb_tot.;
```

```
	run;

	%let tmp_num_tab = %sysevalf(&tmp_num_tab. + 1);


	*
	************************************************************************************************************
**********************************	;
	*	2.	Réadaptation respiratoire réalisée en dehors d'une hospitalisation : les informations sont recherchées
dans le DCIR et dans le PMSI	;
	*		MCO et SSR au niveau des actes et consultations externes

		;
	*
	************************************************************************************************************
**********************************	;

	* Dans le PMSI - ACE;
	%suppr_table(
		lib = orauser,
		table = codes_actes
		);

	data orauser.codes_actes;
		set orauser.codes_actes_BPCO (where = (reperage = 7));
	run;

	%extract_ACT_COD_PMSI(
		annee_deb = &Annee_1N.,
		annee_fin = &Annee_N1.,
		MCO = 1, SSR = 1,
		tbl_out = travail.reperage_actes,
		tbl_codes = codes_actes,
		tbl_patients = corresp_id_patient
		);

	data travail.reperage_actes;
		set travail.reperage_actes (where = (type = "PMSI ACE"));
	run;

	*
	************************************************************************************************************
**********************************;
	*	Vérifications;

	%proc_freq(
		in_tbl = travail.reperage_actes,
		out_tbl = _reperage_actes,
		list_var_in = Annee*Type*Domaine*table*ACT_COD,
		list_var_out = Annee Type Domaine table2 ACT_COD Frequency
		);

	proc delete data = orauser.codes_actes;
	run; quit;

	* Dans le DCIR;
	%suppr_table(
		lib = orauser,
		table = codes_presta_RR
		);

	data orauser.codes_presta_RR;
		set orauser.codes_presta_BPCO (where = (reperage = 7));
	run;

	%extract_PRS_NAT_REF(
		annee_deb = &Annee_1N.,
		annee_fin = &Annee_N1.,
		tbl_out = travail.reperage_presta,
		tbl_codes = codes_presta_RR,
		tbl_patients = corresp_id_patient
```

```
		);

	*
	*****************************************************************************************************************
*********************************;
	*	Vérifications;

	data verif;
		set travail.reperage_presta;
		annee = year(date_debut);
	run;

	%proc_freq(
		in_tbl = verif,
		out_tbl = _reperage_presta,
		list_var_in = Annee*BSE_PRS_NAT*ETE_MCO_DDP,
		list_var_out = Annee BSE_PRS_NAT ETE_MCO_DDP Frequency
		);

	proc delete data = orauser.codes_presta_RR verif;
	run; quit;

	* On concatène les 2 sources;
	data travail.reperage_actes;
		set	travail.reperage_actes
			travail.reperage_presta;
	run;

	proc delete data = travail.reperage_presta;
	run; quit;

	*	********************************** POPULATION 1 ************************************	;
	*	Tous patients et actes réalisés avant le 1er juillet 2018
		;
	*	*******************************************************************************************	;


	* AMS, AMC ou AMK 9,5 + AMS, AMC ou AMK 8 délai maximum entre les 2 actes <= 30 jours avant ou après le code 9,5 mais
le remboursement
	ne doit pas être antérieur à la date index et les actes doivent être réalisé tous les 2 soit en ville, soit en MCO, soit en SSR;

	* On récupère les AMS (BSE_PRS_NAT = 3125), AMC (BSE_PRS_NAT = 3121) ou AMK (BSE_PRS_NAT = 3122) avec un
coefficient 9.5;
	data pop1_reperage1_95;
		set travail.reperage_actes (where = (date_debut < "01JUL2018"d and ((type = "DCIR" and BSE_PRS_NAT in (3125,
3121, 3122) and
			PRS_ACT_CFT = 9.5) or (type = "PMSI ACE" and ACT_COD in ("AMS", "AMC", "AMK") and ACT_COE =
9.5))));
		rename
			PRS_ACT_CFT = PRS_ACT_CFT_95
			ACT_COE = ACT_COE_95;
	run;

	* On récupère les AMS (BSE_PRS_NAT = 3125), AMC (BSE_PRS_NAT = 3121) ou AMK (BSE_PRS_NAT = 3122) avec un
coefficient 8;
	data pop1_reperage1_8;
		set travail.reperage_actes (where = (date_debut < "01JUL2018"d and ((type = "DCIR" and BSE_PRS_NAT in (3125,
3121, 3122) and
			PRS_ACT_CFT = 8) or (type = "PMSI ACE" and ACT_COD in ("AMS", "AMC", "AMK") and ACT_COE =
8))));
		rename
			PRS_ACT_CFT = PRS_ACT_CFT_8
			ACT_COE = ACT_COE_8;
	run;

	* On récupère la jointure des 2 : 30 jours max entre les 2 codes;
	proc sql;

		CREATE TABLE travail.pop1_reperage1 AS
			SELECT
				a.*,
```

```
                                        b.PRS_ACT_CFT_8,
                                        b.ACT_COE_8,
                                        b.date_debut AS date_debut2
                                FROM pop1_reperage1_95 a
                                        INNER JOIN pop1_reperage1_8 b
                                                ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
                                                AND a.type = b.type
                                                AND a.domaine = b.domaine
                                WHERE -30 <= (a.date_debut - b.date_debut) <= 30;

        quit;

        data travail.pop1_reperage1;
                set travail.pop1_reperage1;
                date_RR = MIN(date_debut, date_debut2);
        run;

        *
        ********************************************************************************************************
************************************;
        *       Vérifications;

        %proc_freq(
                in_tbl = travail.pop1_reperage1,
                out_tbl = _RR_pop1_reperage1,
                list_var_in = PRS_ACT_CFT_8*ACT_COE_8*PRS_ACT_CFT_95*ACT_COE_95,
                list_var_out = PRS_ACT_CFT_8 ACT_COE_8 PRS_ACT_CFT_95 ACT_COE_95 Frequency
                );

        proc delete data = pop1_reperage1_95 pop1_reperage1_8;
        run; quit;


        * AMS, AMC ou AMK 9,5 + AMS, AMC ou AMK 8/2 et les actes doivent être réalisé tous les 2 soit en ville, soit en MCO, soit en
SSR et les 2 actes réalisés le même jour;

        * On récupère les AMS (BSE_PRS_NAT = 3125), AMC (BSE_PRS_NAT = 3121) ou AMK (BSE_PRS_NAT = 3122) avec un
coefficient 9.5;
        data pop1_reperage2_95;
                set travail.reperage_actes (where = (date_debut < "01JUL2018"d and ((type = "DCIR" and BSE_PRS_NAT in (3125,
3121, 3122) and
                        PRS_ACT_CFT = 9.5) or (type = "PMSI ACE" and ACT_COD in ("AMS", "AMC", "AMK") and ACT_COE =
9.5))));
                rename
                        PRS_ACT_CFT = PRS_ACT_CFT_95
                        ACT_COE = ACT_COE_95;
        run;

        * On récupère les AMS (BSE_PRS_NAT = 3125), AMC (BSE_PRS_NAT = 3121) ou AMK (BSE_PRS_NAT = 3122) avec un
coefficient 4 (8/2);
        data pop1_reperage2_8;
                set travail.reperage_actes (where = (date_debut < "01JUL2018"d and ((type = "DCIR" and BSE_PRS_NAT in (3125,
3121, 3122) and
                        PRS_ACT_CFT = 4) or (type = "PMSI ACE" and ACT_COD in ("AMS", "AMC", "AMK") and ACT_COE =
4))));
                rename
                        PRS_ACT_CFT = PRS_ACT_CFT_4
                        ACT_COE = ACT_COE_4;
        run;

        * On récupère la jointure des 2 : 30 jours max entre les 2 codes;
        proc sql;

                CREATE TABLE travail.pop1_reperage2 AS
                        SELECT
                                a.*,
                                b.PRS_ACT_CFT_4,
                                b.ACT_COE_4,
                                b.date_debut AS date_debut2
                        FROM pop1_reperage2_95 a
                                INNER JOIN pop1_reperage2_8 b
```

```
                                                ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
                                                AND a.type = b.type
                                                AND a.domaine = b.domaine
                                                AND a.date_debut = b.date_debut;

        quit;

        data travail.pop1_reperage2;
                set travail.pop1_reperage2;
                date_RR = MIN(date_debut, date_debut2);
        run;

        *
        ********************************************************************************************************************
**********************************;
        *       Vérifications;

        %proc_freq(
                in_tbl = travail.pop1_reperage2,
                out_tbl = _RR_pop1_reperage2,
                list_var_in = PRS_ACT_CFT_4*ACT_COE_4*PRS_ACT_CFT_95*ACT_COE_95,
                list_var_out = PRS_ACT_CFT_4 ACT_COE_4 PRS_ACT_CFT_95 ACT_COE_95 Frequency
                );

        proc delete data = pop1_reperage2_95 pop1_reperage2_8;
        run; quit;


        * AMS, AMC ou AMK 13.5;

        data travail.pop1_reperage3;
                set travail.reperage_actes (where = (date_debut < "01JUL2018"d and ((type = "DCIR" and BSE_PRS_NAT in (3125,
3121, 3122) and
                        PRS_ACT_CFT = 13.5) or (type = "PMSI ACE" and ACT_COD in ("AMS", "AMC", "AMK") and ACT_COE =
13.5))));
                length date_RR 4.;
                date_RR = date_debut;
                rename
                        PRS_ACT_CFT = PRS_ACT_CFT_135
                        ACT_COE = ACT_COE_135;
        run;


        * AMS, AMC ou AMK 8;

        data travail.pop1_reperage4;
                set travail.reperage_actes (where = (date_debut < "01JUL2018"d and ((type = "DCIR" and BSE_PRS_NAT in (3125,
3121, 3122) and
                        PRS_ACT_CFT = 8) or (type = "PMSI ACE" and ACT_COD in ("AMS", "AMC", "AMK") and ACT_COE =
8))));
                length date_RR 4.;
                date_RR = date_debut;
                rename
                        PRS_ACT_CFT = PRS_ACT_CFT_8
                        ACT_COE = ACT_COE_8;
        run;


        * BPC (BSE_PRS_NAT = 9570);

        data travail.pop1_reperage5;
                set travail.reperage_actes (where = (date_debut < "01JUL2018"d and ((type = "DCIR" and BSE_PRS_NAT = 9570)
or (type = "PMSI ACE"
                        and ACT_COD in ("BPC")))));
                length date_RR 4.;
                date_RR = date_debut;
        run;

        data travail.population1;
                set travail.pop1_reperage1 travail.pop1_reperage2 (in = a) travail.pop1_reperage3 travail.pop1_reperage4
travail.pop1_reperage5 (in = b);
```

```
		length flag 3.;
		if a then
			flag = 1;
		if b then
			flag = 2;
	run;

	proc delete data = travail.pop1_reperage1 travail.pop1_reperage2 travail.pop1_reperage3 travail.pop1_reperage4
travail.pop1_reperage5;
	run; quit;


	*	********************************** POPULATION 2 ************************************	;
	*	Patients sans ALD (quelque soit l'ALD) et actes réalisés après le 1er juillet 2018		;
	*	********************************************************************************	;


	* AMS, AMC ou AMK 9,5 + AMS, AMC ou AMK 8 délai maximum entre les 2 actes <= 30 jours avant ou après le code 9,5 mais
le remboursement
	ne doit pas être antérieur à la date index et les actes doivent être réalisé tous les 2 soit en ville, soit en MCO, soit en SSR;

	* On récupère les AMS (BSE_PRS_NAT = 3125), AMC (BSE_PRS_NAT = 3121) ou AMK (BSE_PRS_NAT = 3122) avec un
coefficient 9.5;
	data pop2_reperage1_95;
		set travail.reperage_actes (where = (date_debut >= "01JUL2018"d and ((type = "DCIR" and BSE_PRS_NAT in (3125,
3121, 3122) and
			PRS_ACT_CFT = 9.5) or (type = "PMSI ACE" and ACT_COD in ("AMS", "AMC", "AMK") and ACT_COE =
9.5))));
		rename
			PRS_ACT_CFT = PRS_ACT_CFT_95
			ACT_COE = ACT_COE_95;
	run;

	* On récupère les AMS (BSE_PRS_NAT = 3125), AMC (BSE_PRS_NAT = 3121) ou AMK (BSE_PRS_NAT = 3122) avec un
coefficient 8;
	data pop2_reperage1_8;
		set travail.reperage_actes (where = (date_debut >= "01JUL2018"d and ((type = "DCIR" and BSE_PRS_NAT in (3125,
3121, 3122) and
			PRS_ACT_CFT = 8) or (type = "PMSI ACE" and ACT_COD in ("AMS", "AMC", "AMK") and ACT_COE =
8))));
		rename
			PRS_ACT_CFT = PRS_ACT_CFT_8
			ACT_COE = ACT_COE_8;
	run;

	* On récupère la jointure des 2 : 30 jours max entre les 2 codes;
	proc sql;

		CREATE TABLE travail.pop2_reperage1 AS
			SELECT
				a.*,
				b.PRS_ACT_CFT_8,
				b.ACT_COE_8,
				b.date_debut AS date_debut2
			FROM pop2_reperage1_95 a
				INNER JOIN pop2_reperage1_8 b
					ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
					AND a.type = b.type
					AND a.domaine = b.domaine
			WHERE -30 <= (a.date_debut - b.date_debut) <= 30;

	quit;

	data travail.pop2_reperage1;
		set travail.pop2_reperage1;
		length date_RR 4.;
		date_RR = MIN(date_debut, date_debut2);
	run;
```

```
	*
	*************************************************************************************************************
***********************************;
	*	Vérifications;

	%proc_freq(
		in_tbl = travail.pop2_reperage1,
		out_tbl = _RR_pop2_reperage1,
		list_var_in = PRS_ACT_CFT_8*ACT_COE_8*PRS_ACT_CFT_95*ACT_COE_95,
		list_var_out = PRS_ACT_CFT_8 ACT_COE_8 PRS_ACT_CFT_95 ACT_COE_95 Frequency
		);

	proc delete data = pop2_reperage1_95 pop2_reperage1_8;
	run; quit;


	* AMS, AMC ou AMK 9,5 + AMS, AMC ou AMK 8/2 et les actes doivent être réalisé tous les 2 soit en ville, soit en MCO, soit en
SSR et les 2 actes réalisés le même jour;

	* On récupère les AMS (BSE_PRS_NAT = 3125), AMC (BSE_PRS_NAT = 3121) ou AMK (BSE_PRS_NAT = 3122) avec un
coefficient 9.5;
	data pop2_reperage2_95;
		set travail.reperage_actes (where = (date_debut >= "01JUL2018"d and ((type = "DCIR" and BSE_PRS_NAT in (3125,
3121, 3122) and
			PRS_ACT_CFT = 9.5) or (type = "PMSI ACE" and ACT_COD in ("AMS", "AMC", "AMK") and ACT_COE =
9.5))));
		rename
			PRS_ACT_CFT = PRS_ACT_CFT_95
			ACT_COE = ACT_COE_95;
	run;

	* On récupère les AMS (BSE_PRS_NAT = 3125), AMC (BSE_PRS_NAT = 3121) ou AMK (BSE_PRS_NAT = 3122) avec un
coefficient 4 (8/2);
	data pop2_reperage2_8;
		set travail.reperage_actes (where = (date_debut >= "01JUL2018"d and ((type = "DCIR" and BSE_PRS_NAT in (3125,
3121, 3122) and
			PRS_ACT_CFT = 4) or (type = "PMSI ACE" and ACT_COD in ("AMS", "AMC", "AMK") and ACT_COE =
4))));
		rename
			PRS_ACT_CFT = PRS_ACT_CFT_4
			ACT_COE = ACT_COE_4;
	run;

	* On récupère la jointure des 2 : 30 jours max entre les 2 codes;
	proc sql;

		CREATE TABLE travail.pop2_reperage2 AS
			SELECT
				a.*,
				b.PRS_ACT_CFT_4,
				b.ACT_COE_4,
				b.date_debut AS date_debut2
			FROM pop2_reperage2_95 a
				INNER JOIN pop2_reperage2_8 b
					ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
					AND a.type = b.type
					AND a.domaine = b.domaine
					AND a.date_debut = b.date_debut;

	quit;

	data travail.pop2_reperage2;
		set travail.pop2_reperage2;
		length date_RR 4.;
		date_RR = MIN(date_debut, date_debut2);
	run;

	*
	*************************************************************************************************************
***********************************;
	*	Vérifications;
```

```
	%proc_freq(
		in_tbl = travail.pop2_reperage2,
		out_tbl = _RR_pop2_reperage2,
		list_var_in = PRS_ACT_CFT_4*ACT_COE_4*PRS_ACT_CFT_95*ACT_COE_95,
		list_var_out = PRS_ACT_CFT_4 ACT_COE_4 PRS_ACT_CFT_95 ACT_COE_95 Frequency
		);

	proc delete data = pop2_reperage2_95 pop2_reperage2_8;
	run; quit;


	* AMS, AMC ou AMK 13.5;

	data travail.pop2_reperage3;
		set travail.reperage_actes (where = (date_debut >= "01JUL2018"d and ((type = "DCIR" and BSE_PRS_NAT in (3125,
3121, 3122) and
			PRS_ACT_CFT = 13.5) or (type = "PMSI ACE" and ACT_COD in ("AMS", "AMC", "AMK") and ACT_COE =
13.5))));
		length date_RR 4.;
		date_RR = date_debut;
		rename
			PRS_ACT_CFT = PRS_ACT_CFT_135
			ACT_COE = ACT_COE_135;
	run;


	* AMS, AMC ou AMK 8;

	data travail.pop2_reperage4;
		set travail.reperage_actes (where = (date_debut >= "01JUL2018"d and ((type = "DCIR" and BSE_PRS_NAT in (3125,
3121, 3122) and
			PRS_ACT_CFT = 8) or (type = "PMSI ACE" and ACT_COD in ("AMS", "AMC", "AMK") and ACT_COE =
8))));
		length date_RR 4.;
		date_RR = date_debut;
		rename
			PRS_ACT_CFT = PRS_ACT_CFT_8
			ACT_COE = ACT_COE_8;
	run;


	* BPC (BSE_PRS_NAT = 9570);

	data travail.pop2_reperage5;
		set travail.reperage_actes (where = (date_debut >= "01JUL2018"d and ((type = "DCIR" and BSE_PRS_NAT = 9570)
or (type = "PMSI ACE"
			and ACT_COD in ("BPC")))));
		length date_RR 4.;
		date_RR = date_debut;
	run;

	data travail.population2;
		set travail.pop2_reperage1 travail.pop2_reperage2 (in = a) travail.pop2_reperage3 travail.pop2_reperage4
travail.pop2_reperage5 (in = b);
		length flag 3.;
		if a then
			flag = 1;
		if b then
			flag = 2;
	run;

	* On supprime les patients avec une ALD;
	proc sql;

		DELETE FROM travail.population2
		WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM res.T_INDI_BPCO_&an_N. WHERE ALD = 1);

	quit;
```

```
	proc delete data = travail.pop2_reperage1 travail.pop2_reperage2 travail.pop2_reperage3 travail.pop2_reperage4
travail.pop2_reperage5;
	run; quit;

	*	********************************** POPULATION 3 ***********************************	;
	*	Patients avec ALD (quelque soit l'ALD) et actes réalisés après le 1er juillet 2018	;
	*	*******************************************************************************	;


	* AMS, AMC ou AMK 9,5 + AMS, AMC ou AMK 8 délai maximum entre les 2 actes <= 30 jours avant ou après le code 9,5 mais
le remboursement
	ne doit pas être antérieur à la date index et les actes doivent être réalisé tous les 2 soit en ville, soit en MCO, soit en SSR;

	* On récupère les AMS (BSE_PRS_NAT = 3125), AMC (BSE_PRS_NAT = 3121) ou AMK (BSE_PRS_NAT = 3122) avec un
coefficient 9.5;
	data pop3_reperage1_95;
		set travail.reperage_actes (where = (date_debut >= "01JUL2018"d and ((type = "DCIR" and BSE_PRS_NAT in (3125,
3121, 3122) and
			PRS_ACT_CFT = 9.5) or (type = "PMSI ACE" and ACT_COD in ("AMS", "AMC", "AMK") and ACT_COE =
9.5))));
		rename
			PRS_ACT_CFT = PRS_ACT_CFT_95
			ACT_COE = ACT_COE_95;
	run;

	* On récupère les AMS (BSE_PRS_NAT = 3125), AMC (BSE_PRS_NAT = 3121) ou AMK (BSE_PRS_NAT = 3122) avec un
coefficient 8;
	data pop3_reperage1_8;
		set travail.reperage_actes (where = (date_debut >= "01JUL2018"d and ((type = "DCIR" and BSE_PRS_NAT in (3125,
3121, 3122) and
			PRS_ACT_CFT = 8) or (type = "PMSI ACE" and ACT_COD in ("AMS", "AMC", "AMK") and ACT_COE =
8))));
		rename
			PRS_ACT_CFT = PRS_ACT_CFT_8
			ACT_COE = ACT_COE_8;
	run;

	* On récupère la jointure des 2 : 30 jours max entre les 2 codes;
	proc sql;

		CREATE TABLE travail.pop3_reperage1 AS
			SELECT
				a.*,
				b.PRS_ACT_CFT_8,
				b.ACT_COE_8,
				b.date_debut AS date_debut2
			FROM pop3_reperage1_95 a
				INNER JOIN pop3_reperage1_8 b
					ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
					AND a.type = b.type
					AND a.domaine = b.domaine
			WHERE -30 <= (a.date_debut - b.date_debut) <= 30;

	quit;

	data travail.pop3_reperage1;
		set travail.pop3_reperage1;
		length date_RR 4.;
		date_RR = MIN(date_debut, date_debut2);
	run;

	proc delete data = pop3_reperage1_95 pop3_reperage1_8;
	run; quit;

	*
	*************************************************************************************************************
***********************************;
	*	Vérifications;

	%proc_freq(
		in_tbl = travail.pop3_reperage1,
```

```
		out_tbl = _RR_pop3_reperage1,
		list_var_in = PRS_ACT_CFT_8*ACT_COE_8*PRS_ACT_CFT_95*ACT_COE_95,
		list_var_out = PRS_ACT_CFT_8 ACT_COE_8 PRS_ACT_CFT_95 ACT_COE_95 Frequency
		);


	* AMS, AMC ou AMK 9,5 + AMS, AMC ou AMK 8/2 et les actes doivent être réalisé tous les 2 soit en ville, soit en MCO, soit en
SSR et les 2 actes réalisés le même jour ;

	* On récupère les AMS (BSE_PRS_NAT = 3125), AMC (BSE_PRS_NAT = 3121) ou AMK (BSE_PRS_NAT = 3122) avec un
coefficient 9.5;
	data pop3_reperage2_95;
		set travail.reperage_actes (where = (date_debut >= "01JUL2018"d and ((type = "DCIR" and BSE_PRS_NAT in (3125,
3121, 3122) and
			PRS_ACT_CFT = 9.5) or (type = "PMSI ACE" and ACT_COD in ("AMS", "AMC", "AMK") and ACT_COE =
9.5))));
		rename
			PRS_ACT_CFT = PRS_ACT_CFT_95
			ACT_COE = ACT_COE_95;
	run;

	* On récupère les AMS (BSE_PRS_NAT = 3125), AMC (BSE_PRS_NAT = 3121) ou AMK (BSE_PRS_NAT = 3122) avec un
coefficient 4 (8/2);
	data pop3_reperage2_8;
		set travail.reperage_actes (where = (date_debut >= "01JUL2018"d and ((type = "DCIR" and BSE_PRS_NAT in (3125,
3121, 3122) and
			PRS_ACT_CFT = 4) or (type = "PMSI ACE" and ACT_COD in ("AMS", "AMC", "AMK") and ACT_COE =
4))));
		rename
			PRS_ACT_CFT = PRS_ACT_CFT_4
			ACT_COE = ACT_COE_4;
	run;

	* On récupère la jointure des 2 : 30 jours max entre les 2 codes;
	proc sql;

		CREATE TABLE travail.pop3_reperage2 AS
			SELECT
				a.*,
				b.PRS_ACT_CFT_4,
				b.ACT_COE_4,
				b.date_debut AS date_debut2
			FROM pop3_reperage2_95 a
				INNER JOIN pop3_reperage2_8 b
					ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
					AND a.type = b.type
					AND a.domaine = b.domaine
					AND a.date_debut = b.date_debut;

	quit;

	data travail.pop3_reperage2;
		set travail.pop3_reperage2;
		length date_RR 4.;
		date_RR = MIN(date_debut, date_debut2);
	run;

	*
	**********************************************************************************************************
**********************************;
	*	Vérifications;

	%proc_freq(
		in_tbl = travail.pop3_reperage2,
		out_tbl = _RR_pop3_reperage2,
		list_var_in = PRS_ACT_CFT_4*ACT_COE_4*PRS_ACT_CFT_95*ACT_COE_95,
		list_var_out = PRS_ACT_CFT_4 ACT_COE_4 PRS_ACT_CFT_95 ACT_COE_95 Frequency
		);

	proc delete data = pop3_reperage2_95 pop3_reperage2_8;
	run; quit;
```

```
	* AMS, AMC ou AMK 13.5;

	data travail.pop3_reperage3;
		set travail.reperage_actes (where = (date_debut >= "01JUL2018"d and ((type = "DCIR" and BSE_PRS_NAT in (3125,
3121, 3122) and
			PRS_ACT_CFT = 13.5) or (type = "PMSI ACE" and ACT_COD in ("AMS", "AMC", "AMK") and ACT_COE =
13.5))));
		length date_RR 4.;
		date_RR = date_debut;
		rename
			PRS_ACT_CFT = PRS_ACT_CFT_135
			ACT_COE = ACT_COE_135;
	run;


	* AMS, AMC ou AMK 8;

	data travail.pop3_reperage4;
		set travail.reperage_actes (where = (date_debut >= "01JUL2018"d and ((type = "DCIR" and BSE_PRS_NAT in (3125,
3121, 3122) and
			PRS_ACT_CFT = 8) or (type = "PMSI ACE" and ACT_COD in ("AMS", "AMC", "AMK") and ACT_COE =
8))));
		length date_RR 4.;
		date_RR = date_debut;
		rename
			PRS_ACT_CFT = PRS_ACT_CFT_8
			ACT_COE = ACT_COE_8;
	run;


	* BPC (BSE_PRS_NAT = 9570);

	data travail.pop3_reperage5;
		set travail.reperage_actes (where = (date_debut >= "01JUL2018"d and ((type = "DCIR" and BSE_PRS_NAT = 9570)
or (type = "PMSI ACE"
			and ACT_COD in ("BPC")))));
		length date_RR 4.;
		date_RR = date_debut;
	run;


	* AMC ou AMK 20;

	data travail.pop3_reperage6;
		set travail.reperage_actes (where = (date_debut >= "01JUL2018"d and ((type = "DCIR" and BSE_PRS_NAT in (3121,
3122) and
			PRS_ACT_CFT = 20) or (type = "PMSI ACE" and ACT_COD in ("AMC", "AMK") and ACT_COE = 20))));
		length date_RR 4.;
		date_RR = date_debut;
	run;


	* AMC ou AMK 28;

	data travail.pop3_reperage7;
		set travail.reperage_actes (where = (date_debut >= "01JUL2018"d and ((type = "DCIR" and BSE_PRS_NAT in (3121,
3122) and
			PRS_ACT_CFT = 28) or (type = "PMSI ACE" and ACT_COD in ("AMC", "AMK") and ACT_COE = 28))));
		length date_RR 4.;
		date_RR = date_debut;
	run;

	data travail.population3;
		set travail.pop3_reperage1 travail.pop3_reperage2 (in = a) travail.pop3_reperage3 travail.pop3_reperage4
travail.pop3_reperage5 (in = b)
			travail.pop3_reperage6 (in = c) travail.pop3_reperage7 (in = d);
		length flag 3.;
		if a then
			flag = 1;
```

```
		if b then
			flag = 2;
		if c then
			flag = 20;
		if d then
			flag = 28;
	run;

	* On supprime les patients avec une ALD;
	proc sql;

		DELETE FROM travail.population3
		WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM res.T_INDI_BPCO_&an_N. WHERE ALD = 0);

	quit;

	proc delete data = travail.pop3_reperage1 travail.pop3_reperage2 travail.pop3_reperage3 travail.pop3_reperage4
travail.pop3_reperage5
			travail.pop3_reperage6 travail.pop3_reperage7;
	run; quit;


	*
	************************************************************************************************************
************************************;
	*	RR réalisée en dehors d une hospitalisation;
	data travail.reperage_RR_&Annee_1N._&Annee_N1.;
		set travail.population1 (in = c)
			travail.population2 (in = d)
			travail.population3 (in = e);
		length date_RR 4. lieu_RR KINE_AMK_AMC_AMS_8 KINE_AMK_13_5 KINE_AMK_AMC_AMS_9_5_8
KINE_AMK_AMC_AMS_9_5_4 KINE_BPC 3.;
		format type_reperage $18. date_RR ddmmyy10.;
		KINE_AMK_AMC_AMS_8 = 0;
		KINE_AMK_13_5 = 0;
		KINE_AMK_AMC_AMS_9_5_8 = 0;
		KINE_AMK_AMC_AMS_9_5_4 = 0;
		KINE_BPC = 0;
		* Type de repérage;
		if flag = 28 then
			type_reperage = "AMK AMC 28";
		if flag = 20 then
			type_reperage = "AMK AMC 20";
		if ACT_COE_8 = 8 or PRS_ACT_CFT_8 = 8 then
			KINE_AMK_AMC_AMS_8 = 1;
		if ACT_COE_135 = 13.5 or PRS_ACT_CFT_135 = 13.5 then
			KINE_AMK_13_5 = 1;
		if date_debut2 ne . and flag = . and (ACT_COE_95 = 9.5 or PRS_ACT_CFT_95 = 9.5) and (ACT_COE_8 = 8 or
PRS_ACT_CFT_8 = 8) then
			KINE_AMK_AMC_AMS_9_5_8 = 1;
		if date_debut2 ne . and flag = 1 and (ACT_COE_95 = 9.5 or PRS_ACT_CFT_95 = 9.5) and (ACT_COE_4 = 4 or
PRS_ACT_CFT_4 = 4) then
			KINE_AMK_AMC_AMS_9_5_4 = 1;
		if flag = 2 then
			KINE_BPC = 1;
		* Lieu RR;
		if type = "DCIR" then
			lieu_RR = 1;
		if domaine = "SSR" then
			lieu_RR = 3;
		if domaine = "MCO" then
			lieu_RR = 2;
	run;

	proc datasets library = travail memtype = data nolist;
		delete population1 population2 population3 reperage_actes;
	run; quit;
```

3.5.6 06_Oxygenotherapie.sas

```
/*
	*************************************************************************************************************
**********************************	*/
/*
                                                                                                                */
/*			Oxygénothérapie
                                                                                                        */
/*
                                                                                                                */
/*
	*************************************************************************************************************
**********************************	*/

* Repérage via les actes CCAM;
%suppr_table(
	lib = orauser,
	table = codes_CCAM
	);

data orauser.codes_CCAM;
	set orauser.codes_CCAM_BPCO (where = (reperage = 5));
run;

%extract_CCAM_PMSI(
	annee_deb = &Annee_N.,
	annee_fin = &Annee_N1.,
	HAD = 1, MCO = 0, RIP = 0, SSR = 0,
	tbl_out = reperage_CCAM_PMSI,
	tbl_codes = codes_CCAM,
	tbl_patients = corresp_id_patient
	);

* Repérage via les codes LPP;
%suppr_table(
	lib = orauser,
	table = codes_LPP
	);

data orauser.codes_LPP;
	set orauser.codes_LPP_BPCO (where = (reperage = 5));
run;

%extract_LPP_DCIR(
	annee_deb = &Annee_N.,
	annee_fin = &Annee_N1.,
	tbl_out = reperage_LPP_sdv,
	tbl_codes = codes_LPP,
	tbl_patients = corresp_id_patient
	);

* Concaténation des tables;
data travail.reperage_Oxygeno_&Annee_N._&annee_N1.;
	set reperage_LPP_sdv (in = a drop = type)
		reperage_CCAM_PMSI (in = b drop = type);
	if a and date_fin = . then
		date_fin = date_debut;
	if b and date_fin = . then
		date_fin = mdy(12, 31, annee);
run;

proc delete data = reperage_LPP_sdv;
run; quit;
```

```
*
	**********************************************************************************************************
**********************************;
*	Vérifications;

data verif;
	set travail.reperage_Oxygeno_&Annee_N._&annee_N1.;
	annee_debut = year(date_debut);
run;

%proc_freq(
	in_tbl = verif,
	out_tbl = _oxygeno,
	list_var_in = Domaine*Annee*annee_debut*Reperage*TIP_PRS_IDE*Code_CCAM,
	list_var_out = Domaine Annee annee_debut Reperage TIP_PRS_IDE Code_CCAM Frequency
	);

proc delete data = verif;
run; quit;
```

3.5.7 07_Delivrances_remboursees_d_antibiotherapie.sas

```
/*
	*****************************************************************************************************
**********************************	*/
/*
                                                                                                        */
/*		Délivrance remboursée d'antibiothérapie respiratoire

				*/
/*
                                                                                                        */
/*
	*****************************************************************************************************
**********************************	*/

%suppr_table(
	lib = orauser,
	table = codes_ATC
	);

data orauser.codes_ATC;
	set orauser.codes_ATC_BPCO (where = (reperage in (36)));
run;

* Dans le DCIR - Codes CIP;
%suppr_table(
	lib = travail,
	table = reperage_antibio_&Annee_1N._&Annee_N1.
	);

%extract_CIP(
	annee_deb = &Annee_1N.,
	annee_fin = &Annee_N1.,
	tbl_out = travail.reperage_antibio_&Annee_1N._&Annee_N1.,
	tbl_codes = codes_ATC,
	tbl_patients = corresp_id_patient
	);

proc delete data = orauser.codes_ATC;
run; quit;

*
	*****************************************************************************************************
**********************************;
*	Vérifications;

data verif;
	set travail.reperage_antibio_&Annee_1N._&Annee_N1.;
	annee_debut = year(date_debut);
run;

%proc_freq(
	in_tbl = verif,
	out_tbl = _atb,
	list_var_in = TYPE*annee_debut*PHA_ATC_C07*PHA_CND_TOP,
	list_var_out = TYPE annee_debut PHA_ATC_C07 PHA_CND_TOP Frequency
	);

proc delete data = verif;
run; quit;
```

3.5.8 08_Delivrances_remboursees_de_vaccin_anti_grippal.sas

```
/*
	*************************************************************************************************************
**********************************	*/
/*

*/
/*			Délivrance remboursée de vaccin anti grippal

*/
/*

*/
/*
	*************************************************************************************************************
**********************************	*/

%suppr_table(
	lib = orauser,
	table = codes_ATC
	);

data orauser.codes_ATC;
	set orauser.codes_ATC_BPCO (where = (reperage in (40)));
run;

* Dans le DCIR - Codes CIP;
%extract_CIP(
	annee_deb = &Annee_N.,
	annee_fin = &Annee_N1.,
	tbl_out = travail.reperage_vacc_grippe_&Annee_N._&Annee_N1.,
	tbl_codes = codes_ATC,
	tbl_patients = corresp_id_patient
	);

proc sql;

	DELETE FROM travail.reperage_vacc_grippe_&Annee_N._&Annee_N1.
	WHERE code_CIP13 IN (3400939886091, 3400939886213, 3400939886152, 3400939925745, 3400939925806,
3400939925974);

quit;

proc delete data = orauser.codes_ATC;
run; quit;

*
	*************************************************************************************************************
**********************************;
*	Vérifications;

data verif;
	set travail.reperage_vacc_grippe_&Annee_N._&Annee_N1.;
	annee_debut = year(date_debut);
run;

%proc_freq(
	in_tbl = verif,
	out_tbl = _vaccin_grippe,
	list_var_in = TYPE*annee_debut*PHA_ATC_C07*PHA_CND_TOP,
	list_var_out = TYPE annee_debut PHA_ATC_C07 PHA_CND_TOP Frequency
	);

proc delete data = verif;
run; quit;
```

3.5.9 09_Sejours_avec_sortie_pour_fuite.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*

*/
/*	Séjours avec un code diagnostic de sortie contre avis médical ou fuite codé en DAS
*/
/*

*/
/*
	*****************************************************************************************************************
**********************************	*/


* On récupère les séjours en MCO avec un DAS Z53.2;
data code_CIM_fuite;
retain code_CIM;
length code_CIM $ 4;
input code_CIM $;
datalines;
Z532
;
run;

* On supprime la table si elle existe;
%suppr_table(
	lib = orauser,
	table = code_CIM_fuite
	);

* On ajoute la longueur de code_CIM;
data orauser.code_CIM_fuite;
	set code_CIM_fuite;
	length taille 3.;
	taille = length(compress(code_CIM));
run;

proc delete data = code_CIM_fuite;
run;

* On appelle la macro pour le repérage dans le PMSI MCO - DAS;
%extract_CIM10_PMSI(
	annee_deb = &annee_N.,
	annee_fin = &annee_N.,
	HAD_DP = 0,
	HAD_DAS = 0,
	HAD_MPP = 0,
	HAD_MPA = 0,
	MCO_DP = 0,
	MCO_DR = 0,
	MCO_DAS = 1,
	MCO_DP_UM = 0,
	MCO_DR_UM = 0,
	SSR_FP = 0,
	SSR_MPP = 0,
	SSR_AE = 0,
	SSR_DAS = 0,
	tbl_out = travail.fuite_&annee_N.,
	tbl_codes = code_CIM_fuite,
	tbl_patients = corresp_id_patient
	);

data travail.fuite_&annee_N.;
	set travail.fuite_&annee_N. (where = (substr(GRG_GHM, 1, 2) ne "28" or SEJ_NBJ > 0));
	id_sejour = ETA_NUM||"_"||RSA_NUM||"_"||put(Annee, 4.);
```

```
run;

*
	**********************************************************************************************************
**********************************;
*	Vérifications;

%proc_freq(
	in_tbl = travail.fuite_&annee_N.,
	out_tbl = _fuite_&annee_N.,
	list_var_in = Annee*Table*Variable*Code_CIM,
	list_var_out = Annee Table2 Variable Code_CIM Frequency
	);
```

3.5.10 10_Sejours_soins_palliatifs.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*			Séjours avec un diagnostic de soins palliatif

                                                */
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/

* On récupère les séjours en MCO, SSR et HAD avec un diagnostic de soins palliatif;

* On supprime la table si elle existe;
%suppr_table(
	lib = orauser,
	table = code_CIM_soin_palliatif
	);

data orauser.code_CIM_soin_palliatif;
	set orauser.diag_codes_CIM_BPCO (where = (reperage = 13));
run;

* On appelle la macro pour le repérage dans le PMSI;
%extract_CIM10_PMSI(
	annee_deb = &annee_2N.,
	annee_fin = &annee_N1.,
	HAD_DP = 0,
	HAD_DAS = 0,
	HAD_MPP = 0,
	HAD_MPA = 0,
	MCO_DP = 1,
	MCO_DR = 1,
	MCO_DAS = 1,
	MCO_DP_UM = 1,
	MCO_DR_UM = 1,
	SSR_FP = 1,
	SSR_MPP = 1,
	SSR_AE = 1,
	SSR_DAS = 1,
	tbl_out = travail.soin_palliatif_&annee_2N._&annee_N1.,
	tbl_codes = code_CIM_soin_palliatif,
	tbl_patients = corresp_id_patient
	);

* Repérage dans le HAD avec PEC_PAL = « 04 » ou PEC_ASS = « 04 »;
%macro HAD_palliatif(annee_deb=, annee_fin=, tbl_out=, tbl_patients=);

	%let an_deb_pmsi = %sysevalf(&annee_deb. - 2000);
	%let an_fin_pmsi = %sysevalf(&annee_fin. - 2000);

	%do i = &an_deb_pmsi. %to &an_fin_pmsi.;

		%if &i. < 10 %then %let an = 0&i.;
		%else %let an = &i.;

		%let var_join1 = ETA_NUM_EPMSI;
		%let var_join2 = RHAD_NUM;

		proc sql;

			%connectora;
```

```
                                        CREATE TABLE HAD_palliatif_&an. AS
                                                SELECT *
                                                FROM CONNECTION TO ORACLE (
                                                        SELECT DISTINCT
                                                                pop.BEN_IDT_ANO,
                                                                c.EXE_SOI_DTD,
                                                                c.EXE_SOI_DTF,
                                                                c.&var_join1.,
                                                                c.&var_join2.,
                                                                'HAD' AS domaine,
                                                                20&an. AS annee
                                                        FROM &tbl_patients. pop
                                                                INNER JOIN T_HAD&an.C c
                                                                        ON      pop.BEN_NIR_PSA = c.NIR_ANO_17
                                                                INNER JOIN T_HAD&an.B b
                                                                        ON      c.&var_join1. = b.&var_join1.
                                                                        AND     c.&var_join2. = b.&var_join2.
                                                        WHERE (b.PEC_PAL = '04' OR b.PEC_ASS = '04')
                                                                AND c.NIR_ANO_17 NOT IN ('&cle_inc1.', '&cle_inc2.')
                                                                AND NIR_RET = '0' AND NAI_RET = '0' AND SEX_RET = '0'
AND SEJ_RET = '0' AND FHO_RET = '0'
                                                                AND PMS_RET = '0' AND DAT_RET = '0' %if &an. > 12 %then
%do; AND COH_NAI_RET = '0'
                                                                AND COH_SEX_RET = '0' %end;
                                                        );

                                DISCONNECT FROM ORACLE;

                        quit;

                        data HAD_palliatif_&an.;
                                set HAD_palliatif_&an.;
                                length date_debut date_fin 4.;
                                date_debut = datepart(EXE_SOI_DTD);
                                date_fin = datepart(EXE_SOI_DTF);
                                format date_debut date_fin ddmmyy10.;
                                drop EXE_SOI_DTD EXE_SOI_DTF;
                        run;

                        %arret_erreur;

                %end;

                * Empilage de la table temporaire à la table Résultat;
                data &tbl_out.;
                        set HAD_palliatif_:;
                run;

%mend HAD_palliatif;

%HAD_palliatif(
                annee_deb = &annee_2N.,
                annee_fin = &annee_N1.,
                tbl_out = soins_palliatif_HAD,
                tbl_patients = corresp_id_patient
                );

* Table finale;
data travail.soin_palliatif_&annee_2N._&annee_N1.;
                set     travail.soin_palliatif_&annee_2N._&annee_N1.
                        soins_palliatif_HAD;
                if domaine = "MCO" then
                        id_sejour = ETA_NUM||"_"||RSA_NUM||"_"||put(Annee, 4.);
                else if domaine = "HAD" then
                        id_sejour = ETA_NUM_EPMSI||"_"||RHAD_NUM||"_"||put(Annee, 4.);
                else if domaine = "SSR" then
                        id_sejour = ETA_NUM||"_"||RHA_NUM||"_"||put(Annee, 4.);
                if domaine ne "MCO" and date_fin = . then
                        date_fin = mdy(12, 31, annee);
run;
```

```
*
	**********************************************************************************************************
**********************************;
*	Vérifications;

%proc_freq(
	in_tbl = travail.soin_palliatif_&annee_2N._&annee_N1.,
	out_tbl = _soin_pall,
	list_var_in = Annee*Domaine*Table*Variable*Code_CIM,
	list_var_out = Annee Domaine Table2 Variable Code_CIM Frequency
	);
```

3.5.11 11_Sejours_PMSI.sas

```
/*
	**********************************************************************************************************
**********************************	*/
/*

*/
/*		Séjours en HAD, MCO, SSR ou RIP

*/
/*

*/
/*
	**********************************************************************************************************
**********************************	*/

%extract_sej_PMSI(
	annee_deb = &annee_N.,
	annee_fin = &annee_N1.,
	tbl_out = travail.sejours_&Annee_N._&Annee_N1.,
	tbl_patients = corresp_id_patient
	);

data travail.sejours_&Annee_N._&Annee_N1.;
	set travail.sejours_&Annee_N._&Annee_N1.;
	if domaine in ("HAD", "RIP", "SSR") then
		do;
			if date_fin = . then
				date_fin = mdy(12, 31, annee);
		end;
run;

*
	**********************************************************************************************************
**********************************;
*	Vérifications;

%proc_freq(
	in_tbl = travail.sejours_&Annee_N._&Annee_N1.,
	out_tbl = _sejours_PMSI,
	list_var_in = Annee*Domaine*Type,
	list_var_out = Annee Domaine Type Frequency
	);
```

3.5.12 12_Historique_des_ALD.sas

```
	/*
	*************************************************************************************************************
************************************	*/
	/*


	*/
	/*			Repérage de toutes les ALD

											*/
	/*


	*/
	/*
	*************************************************************************************************************
************************************	*/

	%extract_ALD_CIM(
		tbl_out = travail.Histo_ALD,
		tbl_codes = ALD_codes_CIM_BPCO,
		tbl_patients = corresp_id_patient
		);

	*
	*************************************************************************************************************
************************************;
	*	Vérifications;

	%proc_freq(
		in_tbl = travail.Histo_ALD,
		out_tbl = _Histo_ALD,
		list_var_in = IMB_ETM_NAT*MED_MTF_COD,
		list_var_out = IMB_ETM_NAT MED_MTF_COD Frequency
		);
```

3.5.13 13_Traitements_asthmatiques.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*

                                                                                                                */
/*		Délivrance remboursée de traitements asthmatiques :

                                                */
/*				- SCI seuls (reperage = 41)

                                                                                        */
/*				- Traitements anti IgE (reperage = 37)

                                                                */
/*				- Traitements anti IL5 (reperage = 38)

                                                                */
/*				- Antileucotriène (reperage = 39)

                                                                        */
/*

                                                                                                                */
/*
	*****************************************************************************************************************
**********************************	*/

%suppr_table(
	lib = orauser,
	table = codes_ATC
	);

data orauser.codes_ATC;
	set orauser.codes_ATC_BPCO (where = (reperage in (37, 38, 39, 41)));
run;

* Dans le DCIR - Codes CIP;
%extract_CIP(
	annee_deb = &Annee_2N.,
	annee_fin = &Annee_N1.,
	tbl_out = travail.reperage_med_asthme_&Annee_2N._&Annee_N1.,
	tbl_codes = codes_ATC,
	tbl_patients = corresp_id_patient
	);

proc delete data = orauser.codes_ATC;
run; quit;

*
	*****************************************************************************************************************
**********************************;
*	Vérifications;

data verif;
	set travail.reperage_med_asthme_&Annee_2N._&Annee_N1.;
	annee_debut = year(date_debut);
run;

%proc_freq(
	in_tbl = verif,
	out_tbl = _med_asthme,
	list_var_in = TYPE*annee_debut*PHA_ATC_C07*PHA_CND_TOP,
	list_var_out = TYPE annee_debut PHA_ATC_C07 PHA_CND_TOP Frequency
	);

proc delete data = verif;
```

```
run; quit;
```

3.5.14 14_Sejours_pour_asthme.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*

*/
/*			Séjours avec un code diagnostic pour asthme et état de mal asthmatique

	*/
/*

*/
/*
	*****************************************************************************************************************
**********************************	*/

%suppr_table(
	lib = orauser,
	table = codes_CIM
	);

data orauser.codes_CIM;
	set orauser.diag_codes_CIM_BPCO (where = (reperage in (14)));
run;

* On appelle la macro pour le repérage dans le PMSI - Asthme;
%extract_CIM10_PMSI(
	annee_deb = &annee_1N.,
	annee_fin = &annee_N1.,
	HAD_DP = 1,
	HAD_DAS = 1,
	HAD_MPP = 1,
	HAD_MPA = 1,
	MCO_DP = 1,
	MCO_DR = 1,
	MCO_DAS = 1,
	MCO_DP_UM = 1,
	MCO_DR_UM = 1,
	SSR_FP = 1,
	SSR_MPP = 1,
	SSR_AE = 1,
	SSR_DAS = 1,
	tbl_out = hospit_asthme1,
	tbl_codes = codes_CIM,
	tbl_patients = corresp_id_patient
	);

proc delete data = orauser.codes_CIM;
run; quit;

%suppr_table(
	lib = orauser,
	table = codes_CIM
	);

data orauser.codes_CIM;
	set orauser.diag_codes_CIM_BPCO (where = (reperage in (15)));
run;

* On appelle la macro pour le repérage dans le PMSI - État de mal asthmatique;
%extract_CIM10_PMSI(
	annee_deb = &annee_1N.,
	annee_fin = &annee_N1.,
	HAD_DP = 0,
	HAD_DAS = 0,
	HAD_MPP = 0,
	HAD_MPA = 0,
```

```
		MCO_DP = 1,
		MCO_DR = 0,
		MCO_DAS = 0,
		MCO_DP_UM = 1,
		MCO_DR_UM = 0,
		SSR_FP = 0,
		SSR_MPP = 0,
		SSR_AE = 0,
		SSR_DAS = 0,
		tbl_out = hospit_asthme2,
		tbl_codes = codes_CIM,
		tbl_patients = corresp_id_patient
		);

proc sql;

		DELETE FROM hospit_asthme2
		WHERE SUBSTR(GRG_GHM, 1, 2) = "28" OR SEJ_NBJ = 0;

quit;

proc delete data = orauser.codes_CIM;
run; quit;

* On concatène les deux tables;
data travail.diag_asthme_&annee_1N._&annee_N1.;
		set hospit_asthme1 hospit_asthme2;
		if domaine in ("HAD", "RIP", "SSR") then
			do;
				if date_fin = . then
					date_fin = mdy(12, 31, annee);
			end;
run;

proc delete data = hospit_asthme1 hospit_asthme2;
run; quit;

*
		*************************************************************************************************************
**********************************;
*		Vérifications;

data verif;
		set travail.diag_asthme_&annee_1N._&annee_N1. (where = (reperage = 14));
run;

%proc_freq(
		in_tbl = verif,
		out_tbl = _sejours_asthme14,
		list_var_in = annee*domaine*table*variable,
		list_var_out = annee domaine table2 variable Frequency
		);

proc delete data = verif;
run; quit;

data verif;
		set travail.diag_asthme_&annee_1N._&annee_N1. (where = (reperage = 15));
run;

%proc_freq(
		in_tbl = verif,
		out_tbl = _sejours_asthme15,
		list_var_in = annee*domaine*table*variable,
		list_var_out = annee domaine table2 variable Frequency
		);

proc delete data = verif;
run; quit;
```

3.5.15 15_Thermoplastie_bronchique.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*			Thermoplastie bronchique en MCO
                                                                                                                    */
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/

%suppr_table(
	lib = orauser,
	table = codes_CCAM
	);

data orauser.codes_CCAM;
	set orauser.codes_CCAM_BPCO (where = (reperage = 4));
run;

* Dans le PMSI;
%extract_CCAM_PMSI(
	annee_deb = &Annee_1N.,
	annee_fin = &Annee_N1.,
	HAD = 0, MCO = 1, RIP = 0, SSR = 0,
	tbl_out = travail.reperage_thermo_bronch_&annee_1N._&annee_N1.,
	tbl_codes = codes_CCAM,
	tbl_patients = corresp_id_patient
	);

proc sql;

	DELETE FROM travail.reperage_thermo_bronch_&annee_1N._&annee_N1.
	WHERE type = "PMSI ACE";

quit;

proc delete data = orauser.codes_CCAM;
run; quit;

*
	*****************************************************************************************************************
**********************************;
*	Vérifications;

%proc_freq(
	in_tbl = travail.reperage_thermo_bronch_&annee_1N._&annee_N1.,
	out_tbl = _thermo_bronch,
	list_var_in = Annee*Domaine*Table*reperage*code_CCAM,
	list_var_out = Annee Domaine Table2 code_CCAM reperage Frequency
	);
```

3.5.16 16_Polyarthrite_rhumoatoide.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*			Polyarthrite rhumatoïde
                                                                                                                    */
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/

* On supprime la table si elle existe;
%suppr_table(
	lib = orauser,
	table = code_CIM
	);

data orauser.code_CIM;
	set orauser.diag_codes_CIM_BPCO (where = (reperage = 26));
run;

* On appelle la macro pour le repérage dans le PMSI;
%extract_CIM10_PMSI(
	annee_deb = &annee_4N.,
	annee_fin = &annee_N1.,
	HAD_DP = 0,
	HAD_DAS = 0,
	HAD_MPP = 0,
	HAD_MPA = 0,
	MCO_DP = 1,
	MCO_DR = 1,
	MCO_DAS = 1,
	MCO_DP_UM = 1,
	MCO_DR_UM = 1,
	SSR_FP = 0,
	SSR_MPP = 1,
	SSR_AE = 1,
	SSR_DAS = 1,
	tbl_out = travail.polyarthrite_rhum_&annee_4N._&annee_N1.,
	tbl_codes = code_CIM,
	tbl_patients = corresp_id_patient
	);

data travail.polyarthrite_rhum_&annee_4N._&annee_N1.;
	set travail.polyarthrite_rhum_&annee_4N._&annee_N1.;
	if domaine = "SSR" then
		do;
			if date_fin = . then
				date_fin = mdy(12, 31, annee);
		end;
run;

proc delete data = orauser.code_CIM;
run; quit;

*
	*****************************************************************************************************************
**********************************;
*	Vérifications;

%proc_freq(
	in_tbl = travail.polyarthrite_rhum_&annee_4N._&annee_N1.,
```

```
out_tbl = _polyarthrite_rhum,
list_var_in = Annee*Domaine*Table*Variable*reperage*code_CIM,
list_var_out = Annee Domaine Table2 Variable reperage code_CIM Frequency
);
```

3.5.17 17_Spondylarthrite_ankylosante.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*			Spondylarthrite ankylosante
                                                                                                                    */
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/

* On supprime la table si elle existe;
%suppr_table(
	lib = orauser,
	table = code_CIM
	);

data orauser.code_CIM;
	set orauser.diag_codes_CIM_BPCO (where = (reperage = 27));
run;

* On appelle la macro pour le repérage dans le PMSI;
%extract_CIM10_PMSI(
	annee_deb = &annee_4N.,
	annee_fin = &annee_N1.,
	HAD_DP = 0,
	HAD_DAS = 0,
	HAD_MPP = 0,
	HAD_MPA = 0,
	MCO_DP = 1,
	MCO_DR = 1,
	MCO_DAS = 1,
	MCO_DP_UM = 1,
	MCO_DR_UM = 1,
	SSR_FP = 0,
	SSR_MPP = 1,
	SSR_AE = 1,
	SSR_DAS = 1,
	tbl_out = travail.Spondylarthrite_&annee_4N._&annee_N1.,
	tbl_codes = code_CIM,
	tbl_patients = corresp_id_patient
	);

data travail.Spondylarthrite_&annee_4N._&annee_N1.;
	set travail.Spondylarthrite_&annee_4N._&annee_N1.;
	if domaine = "SSR" then
		do;
			if date_fin = . then
				date_fin = mdy(12, 31, annee);
		end;
run;

proc delete data = orauser.code_CIM;
run; quit;

*
	*****************************************************************************************************************
**********************************;
*	Vérifications;

%proc_freq(
	in_tbl = travail.Spondylarthrite_&annee_4N._&annee_N1.,
```

```
out_tbl = _Spondylarthrite,
list_var_in = Annee*Domaine*Table*Variable*reperage*code_CIM,
list_var_out = Annee Domaine Table2 Variable reperage code_CIM Frequency
);
```

3.5.18 18_Prothese_hanche_genou_chirurgie_rachis.sas

```
/*
	****************************************************************************************************
***********************************		*/
/*

										*/
/*			Acte CCAM de prise en charge d une prothèse de hanche ou de genou ou chirurgie du rachis
									*/
/*

										*/
/*
	****************************************************************************************************
***********************************		*/

%suppr_table(
	lib = orauser,
	table = codes_CCAM
	);

data orauser.codes_CCAM;
	set orauser.codes_CCAM_BPCO (where = (reperage in (8, 9, 10)));
run;

* Dans le PMSI;
%extract_CCAM_PMSI(
	annee_deb = &Annee_1N.,
	annee_fin = &Annee_N1.,
	HAD = 0,
	MCO = 1,
	RIP = 0,
	SSR = 0,
	tbl_out = travail.prothese_chir_rachis_&annee_1N._&annee_N1.,
	tbl_codes = codes_CCAM,
	tbl_patients = corresp_id_patient
	);

proc sql;

	DELETE FROM travail.prothese_chir_rachis_&annee_1N._&annee_N1.
	WHERE type = "PMSI ACE" OR SUBSTR(GRG_GHM, 1, 2) = "28";

quit;

proc delete data = orauser.codes_CCAM;
run; quit;

*
	****************************************************************************************************
***********************************;
*	Vérifications;

%proc_freq(
	in_tbl = travail.prothese_chir_rachis_&annee_1N._&annee_N1.,
	out_tbl = _prothese_chir_rachis,
	list_var_in = Annee*Reperage*Domaine*Table*GRG_GHM*Code_CCAM,
	list_var_out = Annee Reperage Domaine Table2 GRG_GHM Code_CCAM Frequency
	);
```

3.5.19 19_Insuffisance_coronarienne_aortique.sas

```
/*
	*************************************************************************************************************
**********************************	*/
/*

*/
/*		Insuffisance coronarienne et/ou aortique

	*/
/*

*/
/*
	*************************************************************************************************************
**********************************	*/

*
	*************************************************************************************************************
**********************************	;
*	Via les GHM;

* On supprime la table si elle existe;
%suppr_table(
	lib = orauser,
	table = code_GHM
	);

data orauser.code_GHM;
	set orauser.codes_GHM_BPCO (where = (reperage in (28, 29, 30)));
run;

* On appelle la macro pour le repérage dans le PMSI;
%extract_GHM_PMSI(
	annee_deb = &annee_N.,
	annee_fin = &annee_N1.,
	tbl_codes = code_GHM,
	tbl_out = GHM_&annee_N._&annee_N1.,
	tbl_patients = corresp_id_patient
	);

proc sql;

	DELETE FROM GHM_&annee_N._&annee_N1.
	WHERE reperage IN (28, 29) AND (SUBSTR(GRG_GHM, 1, 2) = "28" OR SEJ_NBJ = 0);

quit;

%suppr_table(
	lib = orauser,
	table = code_GHM
	);

*
	*************************************************************************************************************
**********************************	;
*	Via les codes CCAM;

* On supprime la table si elle existe;
%suppr_table(
	lib = orauser,
	table = code_CCAM
	);

data orauser.code_CCAM;
	set orauser.codes_CCAM_BPCO (where = (reperage in (28, 29)));
run;
```

```
* On appelle la macro pour le repérage dans le PMSI;
%extract_CCAM_PMSI(
		annee_deb = &Annee_N.,
		annee_fin = &Annee_N1.,
		HAD = 0, MCO = 1, RIP = 0, SSR = 0,
		tbl_out = CCAM_&annee_N._&annee_N1.,
		tbl_codes = code_CCAM,
		tbl_patients = corresp_id_patient
		);

proc sql;

		DELETE FROM CCAM_&annee_N._&annee_N1.
		WHERE type = "PMSI ACE"
			OR SUBSTR(GRG_GHM, 1, 2) = "28"
			OR SEJ_NBJ = 0;

quit;

* On supprime la table si elle existe;
%suppr_table(
		lib = orauser,
		table = code_CCAM
		);

data orauser.code_CCAM;
		set orauser.codes_CCAM_BPCO (where = (reperage = 30));
run;

* On appelle la macro pour le repérage dans le PMSI;
%extract_CCAM_PMSI(
		annee_deb = &Annee_N.,
		annee_fin = &Annee_N1.,
		HAD = 0, MCO = 1, RIP = 0, SSR = 0,
		tbl_out = CCAM2_&annee_N._&annee_N1.,
		tbl_codes = code_CCAM,
		tbl_patients = corresp_id_patient
		);

proc sql;

		DELETE FROM CCAM2_&annee_N._&annee_N1.
		WHERE type = "PMSI ACE";

quit;

*
		*************************************************************************************************************
************************************		;
*		On regroupe les deux tables;

data travail.insuf_coro_aortique_&annee_N._&annee_N1.;
		set		GHM_&annee_N._&annee_N1.
			CCAM_&annee_N._&annee_N1.
			CCAM2_&annee_N._&annee_N1.;
		id_sejour = ETA_NUM||"_"||RSA_NUM||"_"||put(Annee, 4.);
run;

proc delete data = GHM_&annee_N._&annee_N1. CCAM_&annee_N._&annee_N1. CCAM2_&annee_N._&annee_N1.;
run; quit;

*
		*************************************************************************************************************
************************************;
*		Vérifications;

* On ne passe pas par la macro car trop de variables;
proc sql;

		CREATE TABLE verif._0&tmp_num_tab._insuf_coro AS
			SELECT
```

```
                                    Annee,
                                    Domaine,
                                    table,
                                    SUBSTR(GRG_GHM, 1, 3) AS GRG_GHM3,
                                    type_UM,
                                    type_UM2,
                                    reperage,
                                    code_CCAM,
                                    COUNT(*) AS Frequency
                        FROM travail.insuf_coro_aortique_&annee_N._&annee_N1.
                        GROUP BY 1, 2, 3, 4, 5, 6, 7, 8;

quit;

proc sql noprint; SELECT SUM(Frequency) INTO: nb_tot FROM verif._0&tmp_num_tab._insuf_coro; quit;

data verif._0&tmp_num_tab._insuf_coro;
        set verif._0&tmp_num_tab._insuf_coro;
        length percent 4.;
        format percent nlpct7.1 Frequency commafr_0ch.;
        percent = Frequency/&nb_tot.;
run;

%let tmp_num_tab = %sysevalf(&tmp_num_tab. + 1);
```

3.5.20 20_Radiotherapie_chimiotherapie.sas

```
/*
	*************************************************************************************************************
**********************************	*/
/*

*/
/*			Radiothérapie et chimiothérapie

*/
/*

*/
/*
	*************************************************************************************************************
**********************************	*/

*
	*************************************************************************************************************
********************************** ;
*	Repérage via les codes CIM;

* On supprime la table si elle existe;
%suppr_table(
	lib = orauser,
	table = code_CIM
	);

data orauser.code_CIM;
	set orauser.diag_codes_CIM_BPCO (where = (reperage in (58, 59)));
run;

* On appelle la macro pour le repérage dans le PMSI;
%extract_CIM10_PMSI(
	annee_deb = &annee_1N.,
	annee_fin = &annee_N1.,
	HAD_DP = 0,
	HAD_DAS = 0,
	HAD_MPP = 0,
	HAD_MPA = 0,
	MCO_DP = 0,
	MCO_DR = 0,
	MCO_DAS = 1,
	MCO_DP_UM = 0,
	MCO_DR_UM = 0,
	SSR_FP = 0,
	SSR_MPP = 0,
	SSR_AE = 0,
	SSR_DAS = 0,
	tbl_out = CIM10&annee_1N._&annee_N1.,
	tbl_codes = code_CIM,
	tbl_patients = corresp_id_patient
	);

proc delete data = orauser.code_CIM;
run; quit;

*
	*************************************************************************************************************
********************************** ;
*	Repérage via les codes GHM;

* On supprime la table si elle existe;
%suppr_table(
	lib = orauser,
	table = code_GHM
	);
```

```
data orauser.code_GHM;
	set orauser.codes_GHM_BPCO (where = (reperage in (58, 59)));
run;

* On appelle la macro pour le repérage dans le PMSI;
%extract_GHM_PMSI(
	annee_deb = &annee_1N.,
	annee_fin = &annee_N1.,
	tbl_codes = code_GHM,
	tbl_out = GHM_&annee_1N._&annee_N1.,
	tbl_patients = corresp_id_patient
	);

proc delete data = orauser.code_GHM;
run; quit;

*
	************************************************************************************************************
********************************** ;
*	On concatène les 2 tables;

data travail.radio_chimio_&annee_1N._&annee_N1.;
	set	CIM10&annee_1N._&annee_N1.
		GHM_&annee_1N._&annee_N1.;
	id_sejour = ETA_NUM||"_"||RSA_NUM||"_"||put(Annee, 4.);
run;

proc delete data =  CIM10&annee_1N._&annee_N1. GHM_&annee_1N._&annee_N1.;
run; quit;

*
	************************************************************************************************************
***********************************;
*	Vérifications;

%proc_freq(
	in_tbl = travail.radio_chimio_&annee_1N._&annee_N1.,
	out_tbl = _radio_chimio,
	list_var_in = Annee*Table*Reperage*GRG_GHM*code_CIM,
	list_var_out = Annee Table2 Reperage GRG_GHM code_CIM Frequency
	);
```

3.5.21 21_EFR_spirometrie.sas

```
/*
	*************************************************************************************************************
**********************************	*/
/*
                                                                                                                */
/*			EFR ou spirométrie
                                                                                                */
/*
                                                                                                                */
/*
	*************************************************************************************************************
**********************************	*/

%suppr_table(
	lib = orauser,
	table = codes_CCAM
	);

data orauser.codes_CCAM;
	set orauser.codes_CCAM_BPCO (where = (reperage in (1, 2)));
run;

* Dans le DCIR;
%extract_CCAM_DCIR(
	annee_deb = &Annee_2N.,
	annee_fin = &Annee_N1.,
	tbl_out = reperage_CCAM_sdv,
	tbl_codes = codes_CCAM,
	tbl_patients = corresp_id_patient
	);

* Dans le PMSI;
%extract_CCAM_PMSI(
	annee_deb = &Annee_2N.,
	annee_fin = &Annee_N1.,
	HAD = 0, MCO = 1, RIP = 0, SSR = 1,
	tbl_out = reperage_CCAM_PMSI,
	tbl_codes = codes_CCAM,
	tbl_patients = corresp_id_patient
	);

data travail.reperage_EFR_spiro_&Annee_2N._&annee_N1.;
	set reperage_CCAM_sdv (in = a drop = type)
		reperage_CCAM_PMSI (in = b drop = type);
	if a and date_fin = . then
		date_fin = date_debut;
	if b then
		do;
			if table = "FMSTC" and date_fin = . then
				date_fin = date_debut;
			else if domaine ne "MCO" and date_fin = . then
				date_fin = mdy (12, 31, annee);

		end;
run;

proc delete data = reperage_CCAM_sdv reperage_CCAM_PMSI;
run; quit;

*
	*************************************************************************************************************
**********************************;
*	Vérifications;
```

```
* On ne passe pas par la macro car trop de variables;
proc sql;

	CREATE TABLE verif._0&tmp_num_tab._EFR_spiro AS
		SELECT
			Reperage,
			YEAR(date_debut) AS annee_Debut_DCIR,
			Annee,
			Table,
			Domaine,
			CAM_PRS_IDE,
			CODE_CCAM,
			COUNT(*) AS Frequency
		FROM travail.reperage_EFR_spiro_&Annee_2N._&annee_N1.
		GROUP BY 1, 2, 3, 4, 5, 6, 7;

quit;

proc sql noprint; SELECT SUM(Frequency) INTO: nb_tot FROM verif._0&tmp_num_tab._EFR_spiro; quit;

data verif._0&tmp_num_tab._EFR_spiro;
	set verif._0&tmp_num_tab._EFR_spiro;
	length percent 4.;
	format percent nlpct7.1 Frequency commafr_0ch.;
	percent = Frequency/&nb_tot.;
run;

%let tmp_num_tab = %sysevalf(&tmp_num_tab. + 1);
```

3.5.22 22_Pneumothorax_infarctus.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*			Pneumothorax ou infarctus
                                                                                                                    */
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/

* On supprime la table si elle existe;
%suppr_table(
	lib = orauser,
	table = code_CCAM
	);

data orauser.code_CCAM;
	set orauser.codes_CCAM_BPCO (where = (reperage = 11));
run;

* On appelle la macro pour le repérage dans le PMSI;
%extract_CCAM_PMSI(
	annee_deb = &Annee_1N.,
	annee_fin = &annee_N1.,
	HAD = 0,
	MCO = 1,
	RIP = 0,
	SSR = 1,
	tbl_out = CCAM_pneumo_&Annee_1N._&annee_N1.,
	tbl_codes = code_CCAM,
	tbl_patients = corresp_id_patient
	);

data CCAM_pneumo_&Annee_1N._&annee_N1.;
	set CCAM_pneumo_&Annee_1N._&annee_N1.;
	if table = "FMSTC" and date_fin = . then
		date_fin = date_debut;
	else if domaine ne "MCO" and date_fin = . then
		date_fin = mdy (12, 31, annee);
run;

* On supprime la table si elle existe;
%suppr_table(
	lib = orauser,
	table = code_CIM
	);

data orauser.code_CIM;
	set orauser.diag_codes_CIM_BPCO (where = (reperage = 11));
run;

* On appelle la macro pour le repérage dans le PMSI;
%extract_CIM10_PMSI(
	annee_deb = &Annee_1N.,
	annee_fin = &annee_N1.,
	HAD_DP = 0,
	HAD_DAS = 0,
	HAD_MPP = 0,
	HAD_MPA = 0,
	MCO_DP = 1,
	MCO_DR = 1,
```

```
		MCO_DAS = 1,
		MCO_DP_UM = 1,
		MCO_DR_UM = 1,
		SSR_FP = 0,
		SSR_MPP = 0,
		SSR_AE = 0,
		SSR_DAS = 0,
		tbl_out = pneumothorax_&Annee_1N._&annee_N1.,
		tbl_codes = code_CIM,
		tbl_patients = corresp_id_patient
		);

* On supprime la table si elle existe;
%suppr_table(
		lib = orauser,
		table = code_CIM
		);

data orauser.code_CIM;
		set orauser.diag_codes_CIM_BPCO (where = (reperage = 12));
run;

* On appelle la macro pour le repérage dans le PMSI;
%extract_CIM10_PMSI(
		annee_deb = &Annee_1N.,
		annee_fin = &annee_N1.,
		HAD_DP = 0,
		HAD_DAS = 0,
		HAD_MPP = 0,
		HAD_MPA = 0,
		MCO_DP = 1,
		MCO_DR = 0,
		MCO_DAS = 0,
		MCO_DP_UM = 0,
		MCO_DR_UM = 0,
		SSR_FP = 0,
		SSR_MPP = 0,
		SSR_AE = 0,
		SSR_DAS = 0,
		tbl_out = infarctus_&Annee_1N._&annee_N1.,
		tbl_codes = code_CIM,
		tbl_patients = corresp_id_patient
		);

data travail.pneumothorax_infarctus_&Annee_1N._&annee_N1.;
		set CCAM_pneumo_&Annee_1N._&annee_N1. pneumothorax_&Annee_1N._&annee_N1.
infarctus_&Annee_1N._&annee_N1.;
		if domaine = "MCO" and SEQ_NUM = "" then
			id_sejour = ETA_NUM||"_"||RSA_NUM||"_"||put(Annee, 4.);
		else if domaine = "SSR" and SEQ_NUM = ""  then
			id_sejour = ETA_NUM||"_"||RHA_NUM||"_"||put(Annee, 4.);
		if SEQ_NUM ne "" then
			id_ACE = ETA_NUM||"_"||SEQ_NUM||"_"||put(Annee, 4.);
run;

proc delete data = orauser.code_CIM orauser.code_CCAM;
run; quit;

*
		*************************************************************************************************************
**********************************;
*		Vérifications;

* On ne passe pas par la macro car trop de variables;
proc sql;

		CREATE TABLE verif._0&tmp_num_tab._pneumo_inf AS
			SELECT
				annee,
				table,
				reperage,
```

```
                                    domaine,
                                    variable,
                                    code_CIM,
                                    code_CCAM,
                                    COUNT(*) AS Frequency
                            FROM travail.pneumothorax_infarctus_&Annee_1N._&annee_N1.
                            GROUP BY 1, 2, 3, 4, 5, 6, 7;

quit;

proc sql noprint; SELECT SUM(Frequency) INTO: nb_tot FROM verif._0&tmp_num_tab._pneumo_inf; quit;

data verif._0&tmp_num_tab._pneumo_inf;
        set verif._0&tmp_num_tab._pneumo_inf;
        length percent 4.;
        format percent nlpct7.1 Frequency commafr_0ch.;
        percent = Frequency/&nb_tot.;
run;

%let tmp_num_tab = %sysevalf(&tmp_num_tab. + 1);
```

3.5.23 23_Ventilation_non_invasive.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*			Ventilation non invasive
                                                                                                                    */
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/

* Repérage via les actes CCAM;
%suppr_table(
	lib = orauser,
	table = codes_CCAM
	);

data orauser.codes_CCAM;
	set orauser.codes_CCAM_BPCO (where = (reperage = 6));
run;

%extract_CCAM_PMSI(
	annee_deb = &Annee_N.,
	annee_fin = &Annee_N1.,
	HAD = 1, MCO = 0, RIP = 0, SSR = 0,
	tbl_out = reperage_CCAM_PMSI,
	tbl_codes = codes_CCAM,
	tbl_patients = corresp_id_patient
	);

* Repérage via les codes LPP;
%suppr_table(
	lib = orauser,
	table = codes_LPP
	);

data orauser.codes_LPP;
	set orauser.codes_LPP_BPCO (where = (reperage = 6));
run;

%extract_LPP_DCIR(
	annee_deb = &Annee_N.,
	annee_fin = &Annee_N1.,
	tbl_out = reperage_LPP_sdv,
	tbl_codes = codes_LPP,
	tbl_patients = corresp_id_patient
	);

* Concaténation des tables;
data travail.reperage_VNI_&Annee_N._&annee_N1.;
	set reperage_LPP_sdv (in = a drop = type)
		reperage_CCAM_PMSI (in = b drop = type);
	if a and date_fin = . then
		date_fin = date_debut;
	if b and date_fin = . then
		date_fin = mdy(12, 31, annee);
run;

proc delete data = reperage_CCAM_PMSI reperage_LPP_sdv;
run; quit;
```

```
*
	**********************************************************************************************************
**********************************;
*	Vérifications;

data verif;
	set travail.reperage_VNI_&Annee_N._&annee_N1.;
	annee_debut = year(date_debut);
run;

%proc_freq(
	in_tbl = verif,
	out_tbl = _VNI,
	list_var_in = Domaine*Annee*Annee_Debut*Reperage*TIP_PRS_IDE*Code_CCAM,
	list_var_out = Domaine Annee Annee_Debut Reperage TIP_PRS_IDE Code_CCAM Frequency
	);

proc delete data = verif;
run; quit;
```

3.5.24 24_Delivrance_remboursee_de_BDCA_et_de_substitut_nicotinique.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                */
/*			Délivrance remboursée de BDCA et substitut nicotinique

                                        */
/*
                                                                                                                */
/*
	*****************************************************************************************************************
**********************************	*/

%suppr_table(
	lib = orauser,
	table = codes_ATC
	);

data orauser.codes_ATC;
	set orauser.codes_ATC_BPCO (where = (reperage in (35, 42, 43, 44)));
run;

* Dans le DCIR - Codes CIP;
%suppr_table(
	lib = travail,
	table = reperage_BDCA_tabac_&Annee_2N._&Annee_N1.
	);

%extract_CIP(
	annee_deb = &Annee_2N.,
	annee_fin = &Annee_N1.,
	tbl_out = travail.reperage_BDCA_tabac_&Annee_2N._&Annee_N1.,
	tbl_codes = codes_ATC,
	tbl_patients = corresp_id_patient
	);

proc delete data = orauser.codes_ATC;
run; quit;

*
	*****************************************************************************************************************
**********************************;
*	Vérifications;

data verif;
	set travail.reperage_BDCA_tabac_&Annee_2N._&Annee_N1.;
	annee = year(date_debut);
run;

%proc_freq(
	in_tbl = verif,
	out_tbl = _BDCA_tabac,
	list_var_in = annee*reperage*PHA_ATC_C07*PHA_CND_TOP,
	list_var_out = annee reperage PHA_ATC_C07 PHA_CND_TOP Frequency
	);

proc delete data = verif;
run; quit;
```

3.6 Spirométrie ou EFR à visée diagnostique chez les patients à risque de BPCO

3.6.1 00_Initialisation_de_la_table_de_resultats

```
/*
	**********************************************************************************************************
**********************************	*/
/*

*/
/*			Table de population - Patients avec indicateur_01 = 1

		*/
/*

*/
/*
	**********************************************************************************************************
**********************************	*/

* Patients pour diag de BPCO recherché;
data res.T_INDI_BPCO_DG_&an_N. (keep = BEN_IDT_ANO);
	set pop.indicateurs_&an_N. (where = (indicateur_01 = 1));
run;

proc sort data = res.T_INDI_BPCO_DG_&an_N. nodupkey;
	by BEN_IDT_ANO;
run;

*
	**********************************************************************************************************
**********************************;
*	Patients ayant eu au moins une délivrance remboursée d un bronchodilatateur anticholinergique longue durée d action ou
Bêta-2 longue
	durée d action l année N;

proc sql;

	CREATE TABLE patients_BDLA AS
		SELECT
			BEN_IDT_ANO,
			MIN(date_debut) AS Date_index_broncho format ddmmyy10. length = 4
		FROM travail.reperage_BDLA_&Annee_2N._&Annee_N1.
		WHERE YEAR(date_debut) = &Annee_N.
		GROUP BY BEN_IDT_ANO;

quit;

*	On corrige la variable broncho_LDA, qui est incohérente avec la variable nb_broncho_LDA de la table
res.T_INDI_BPCO_&an_N.;

proc sql;

	CREATE TABLE verif_LDA AS
		SELECT
			a.BEN_IDT_ANO,
			a.Date_index_broncho,
			b.Nb_Broncho_LDA
		FROM patients_BDLA a
			INNER JOIN res.T_INDI_BPCO_17 b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.Nb_Broncho_LDA = 0;

quit;
```

```
proc sql; select count(distinct BEN_IDT_ANo) from verif_LDA; quit;

proc sql;

	DELETE FROM patients_BDLA
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM verif_LDA);

quit;

proc delete data = verif_LDA;
run; quit;

proc sql;

	SELECT COUNT(DISTINCT BEN_IDT_ANO), COUNT(*) FROM patients_BDLA;

quit;

*
	*************************************************************************************************************
***********************************;
*	Patients ayant eu au moins une délivrance remboursée d antibiothérapie pour infections respiratoires l année N
*		ET précédée d au moins une délivrance remboursée d antibiothérapie pour infections respiratoires dans les 365
jours
*		ET ayant eu au moins une délivrance remboursée de bronchodilatateur anti-cholinergique courte durée d action
ou un Bêta-2
*			courte durée d action, délivrée le même jour :
*				o	que la cure d antibiothérapie réalisée l année N
*				o	OU que la cure d antibiothérapie délivrée dans les 365 jours précédant la cure d
antibiothérapie réalisée l année N;

* On récupère toutes les lignes de patients avec des délivrances d antibio l année N;
proc sql;

	CREATE TABLE patients_antibio_all AS
		SELECT DISTINCT
			BEN_IDT_ANO,
			date_debut AS date_deliv_antibio_&Annee_N. format ddmmyy10. length = 4
		FROM travail.reperage_antibio_&Annee_1N._&Annee_N1.
		WHERE YEAR(date_debut) = &Annee_N. AND reperage = 36
		ORDER BY 1, 2;

quit;

* On récupère toutes les délivrance remboursée de BDCA le même jour + précédée dans les 365 jours d au moins une antibiothérapie;
proc sql undo_policy = none;

	* Délivrance de BDCA le même jour que l antibio;
	CREATE TABLE tmp_patients_BDCA_memejour AS
		SELECT
			a.*,
			b.date_debut AS Date_index_antibio_memejour format date9. length = 4
		FROM patients_antibio_all a
			INNER JOIN travail.reperage_BDCA_tabac_&Annee_2N._&Annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE (a.date_deliv_antibio_&Annee_N. = b.date_debut) AND b.reperage in (42, 43, 44);

	* Délivrance antibio dans les 365 jours avant l antibio;
	CREATE TABLE patients_antibio_memejour AS
		SELECT
			a.*
		FROM tmp_patients_BDCA_memejour a
			INNER JOIN travail.reperage_antibio_&Annee_1N._&Annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.reperage = 36 AND (a.Date_index_antibio_memejour - 365) <= b.date_debut <
a.Date_index_antibio_memejour;

	* Date minimale;
	CREATE TABLE patients_antibio_memejour_min AS
		SELECT
			BEN_IDT_ANO,
```

```
                        MIN(Date_index_antibio_memejour) AS Date_index_antibio_memejour format date9. length = 4
                FROM patients_antibio_memejour
                GROUP BY BEN_IDT_ANO;

quit;

proc sql;

        SELECT COUNT(DISTINCT BEN_IDT_ANO), COUNT(*) FROM patients_antibio_memejour_min;

quit;

* On récupère toutes les délivrance remboursées d antibiothérapie dans les 365 jours avec une délivrance remboursée de de BDCA le
même jour;
proc sql undo_policy = none;

        * Délivrance de BDCA dans les 365 jours précédents;
        CREATE TABLE patients_BDCA_365jour AS
                SELECT
                        a.BEN_IDT_ANO,
                        a.date_deliv_antibio_&Annee_N. AS Date_index_antibio_365jour length = 4,
                        b.date_debut AS deliv_BDCA length = 4
                FROM patients_antibio_all a
                        INNER JOIN travail.reperage_BDCA_tabac_&Annee_2N._&Annee_N1. b
                                ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
                WHERE (a.date_deliv_antibio_&Annee_N. - 365) <= b.date_debut < a.date_deliv_antibio_&Annee_N. AND
b.reperage in (42, 43, 44);

        * Restreint aux antibios le même jour;
        CREATE TABLE patients_antibio_BDCA_365jour AS
                SELECT
                        a.*,
                        b.date_debut length = 4
                FROM patients_BDCA_365jour a
                        INNER JOIN travail.reperage_antibio_&Annee_1N._&Annee_N1. b
                                ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
                WHERE b.reperage = 36 AND a.deliv_BDCA = b.date_debut;

        * Date minimale;
        CREATE TABLE patients_antibio_365jour_min AS
                SELECT
                        BEN_IDT_ANO,
                        MIN(Date_index_antibio_365jour) AS Date_index_antibio_365jour format date9. length = 4
                FROM patients_antibio_BDCA_365jour
                GROUP BY BEN_IDT_ANO;

quit;

proc sql;

        SELECT COUNT(DISTINCT BEN_IDT_ANO), COUNT(*) FROM patients_antibio_365jour_min;

quit;

proc delete data = patients_antibio_all;
run; quit;

*
        ****************************************************************************************************
**********************************;
*       Patients ayant eu au moins une délivrance remboursée de substituts nicotiniques (remboursable ou forfait) ou d un
traitement d arrêt du
        tabac l année N;

proc sql undo_policy = none;

        CREATE TABLE patients_tabac AS
                SELECT
                        BEN_IDT_ANO,
                        MIN(date_debut) AS Date_index_substitut format ddmmyy10. length = 4
                FROM travail.reperage_BDCA_tabac_&Annee_2N._&Annee_N1.
```

```
                        WHERE YEAR(date_debut) = &Annee_N. AND reperage = 35
                        GROUP BY BEN_IDT_ANO;

quit;

proc sql;

        SELECT COUNT(DISTINCT BEN_IDT_ANO), COUNT(*) FROM patients_tabac;

quit;

*
        ********************************************************************************************************
**********************************;
*       La date index est le min des 4 dates;

proc sort data = tmp_patients_BDCA_memejour nodupkey;
        by BEN_IDT_ANO;
run;

data res.T_INDI_BPCO_DG_&an_N.;
        merge   res.T_INDI_BPCO_DG_&an_N. (in = z)
                        patients_BDLA (in = a keep = BEN_IDT_ANO Date_index_broncho)
                        patients_antibio_365jour_min (in = b keep = BEN_IDT_ANO Date_index_antibio_365jour)
                        patients_antibio_memejour_min (in = b keep = BEN_IDT_ANO Date_index_antibio_memejour)
                        patients_tabac (in = c keep = BEN_IDT_ANO Date_index_substitut)
                        tmp_patients_BDCA_memejour (in = d keep = BEN_IDT_ANO);
        by BEN_IDT_ANO;
        length BPCO_DG_Date_index 4. ATB_Inf_Meme_Jour ATB_Inf_365_Jour Broncho_CDA Subs_Nico 3.;
        BPCO_DG_Date_index = min(Date_index_broncho, Date_index_antibio_memejour, Date_index_antibio_365jour,
Date_index_substitut);
        ATB_Inf_Meme_Jour = 0;
        if Date_index_antibio_memejour ne . then
                ATB_Inf_Meme_Jour = 1;
        ATB_Inf_365_Jour = 0;
        if Date_index_antibio_365jour ne . then
                ATB_Inf_365_Jour = 1;
        Broncho_CDA = 0;
        if d then
                Broncho_CDA = 1;
        Subs_Nico = 0;
        if Date_index_substitut ne . then
                Subs_Nico = 1;
        format BPCO_DG_Date_index Date_index_broncho Date_index_antibio_memejour Date_index_antibio_365jour
Date_index_substitut ddmmyy10.;
        if z and BPCO_DG_Date_index ne . then
                output;
run;

*
        ********************************************************************************************************
**********************************;
*       Effectifs pour le flowchart;

proc sql;

        CREATE TABLE flowch.Flow_Chart_BPCO_DG_&an_N.  AS
                SELECT *
                FROM (
                        SELECT 1 AS indicateur length = 3, COUNT(DISTINCT BEN_IDT_ANO) AS N length = 5 FROM
res.T_INDI_BPCO_DG_&an_N. WHERE BEN_IDT_ANO
                                        IN (SELECT BEN_IDT_ANO FROM patients_BDLA)
                        UNION ALL
                        SELECT 2.5, COUNT(DISTINCT BEN_IDT_ANO) FROM res.T_INDI_BPCO_DG_&an_N. WHERE
BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM
                                        patients_antibio_365jour_min)
                        UNION ALL
                        SELECT 2, COUNT(DISTINCT BEN_IDT_ANO) FROM res.T_INDI_BPCO_DG_&an_N. WHERE
BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM
                                patients_antibio_memejour_min)
                        UNION ALL
```

```
                                SELECT 3, COUNT(DISTINCT BEN_IDT_ANO) FROM res.T_INDI_BPCO_DG_&an_N. WHERE
BEN_IDT_ANO
                                                IN (SELECT BEN_IDT_ANO FROM patients_tabac)
                                UNION ALL
                                SELECT 4, COUNT(DISTINCT BEN_IDT_ANO) FROM res.T_INDI_BPCO_DG_&an_N.
                        ) ;

quit;

proc delete data = patients_BDLA patients_antibio_365jour_min patients_antibio_memejour_min patients_tabac;
run; quit;
```

3.6.2 01_Inclusions_et_exclusions.sas

```
/*
	*****************************************************************************************************
**********************************	*/
/*

*/
/*		Indicateur 01 : Inclusion & exclusions

*/
/*

*/
/*
	*****************************************************************************************************
**********************************	*/

*
	*****************************************************************************************************
**********************************	;
*	EXCLUSION DES PATIENTS ASTHMATIQUES

;
*
	*****************************************************************************************************
**********************************	;

*
	*****************************************************************************************************
**********************************;
*	Exclusion des patients en ALD asthme active à la date index;

proc sql;

	CREATE TABLE ALD_asthme AS
		SELECT DISTINCT
			a.BEN_IDT_ANO
		FROM res.T_INDI_BPCO_DG_&an_N. a
			INNER JOIN travail.Histo_ALD b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.reperage = 53 AND (b.date_debut <= a.BPCO_DG_Date_index <= b.date_fin OR (b.date_debut <=
a.BPCO_DG_Date_index AND
			b.date_fin = "01JAN1600"d));

quit;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_DG_&an_N.
		SELECT
			5,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM res.T_INDI_BPCO_DG_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM ALD_asthme);

	* On conserve les id des patients à supprimer;
	CREATE TABLE travail.exclus1 AS
		SELECT DISTINCT
			BEN_IDT_ANO
	FROM res.T_INDI_BPCO_DG_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM ALD_asthme);

quit;

proc delete data = ALD_asthme;
run; quit;
```

```
*
	**********************************************************************************************************
**********************************;
*	Exclusion des patients en ALD mucoviscidose active à la date index;

proc sql;

	CREATE TABLE ALD_mucoviscidose AS
		SELECT DISTINCT
			a.BEN_IDT_ANO
		FROM res.T_INDI_BPCO_DG_&an_N. a
			INNER JOIN travail.Histo_ALD b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.reperage = 54 AND (b.date_debut <= a.BPCO_DG_Date_index <= b.date_fin OR (b.date_debut <=
a.BPCO_DG_Date_index AND
			b.date_fin = "01JAN1600"d));

quit;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_DG_&an_N.
		SELECT
			6,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM res.T_INDI_BPCO_DG_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM ALD_mucoviscidose);

	* On conserve les id des patients à supprimer;
	CREATE TABLE travail.exclus2 AS
		SELECT DISTINCT
			BEN_IDT_ANO
	FROM res.T_INDI_BPCO_DG_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM ALD_mucoviscidose);

quit;

proc delete data = ALD_mucoviscidose;
run; quit;

*
	**********************************************************************************************************
**********************************;
*	Exclusion des patients avec 3 délivrances de CSI sans délivrance le même jour de bronchodilatateur;

proc sql;

	CREATE TABLE delivrances_CSI AS
		SELECT DISTINCT
			a.BEN_IDT_ANO,
			b.date_debut
		FROM res.T_INDI_BPCO_DG_&an_N. a
			INNER JOIN travail.reperage_med_asthme_&Annee_2N._&Annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.reperage = 41 AND (a.BPCO_DG_Date_index - 730 <= b.date_debut <= a.BPCO_DG_Date_index - 365);

quit;

* On repère parmi ces patients, les patients avec ou sans délivrance de bronchodilatateurs le même jour;
data BDCA;
	set travail.reperage_BDCA_tabac_&Annee_2N._&Annee_N1. (where = (reperage in (42, 43, 44)));
run;

proc sql;

	CREATE TABLE delivrances_CSI_broncho AS
		SELECT
			a.*,
			CASE	WHEN b.BEN_IDT_ANO IS NOT NULL OR c.BEN_IDT_ANO IS NOT NULL THEN 1
				ELSE 0
```

```
                                                    END AS deliv_broncho length = 3
                        FROM delivrances_CSI a
                                LEFT JOIN BDCA b
                                        ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
                                        AND a.date_debut = b.date_debut
                                LEFT JOIN travail.reperage_BDLA_&Annee_2N._&Annee_N1. c
                                        ON a.BEN_IDT_ANO = c.BEN_IDT_ANO
                                        AND a.date_debut = c.date_debut;

        DELETE FROM delivrances_CSI_broncho WHERE deliv_broncho = 1;

quit;

proc delete data = BDCA;
run; quit;

proc sql undo_policy = none;

        CREATE TABLE delivrances_CSI_broncho AS
                SELECT
                        BEN_IDT_ANO,
                        COUNT(DISTINCT date_debut) AS nb_deliv length = 3
                FROM delivrances_CSI_broncho
                GROUP BY BEN_IDT_ANO;

        DELETE FROM delivrances_CSI_broncho WHERE nb_deliv < 3;

quit;


proc sql;

        * On insère l effectif dans le Flowchart;
        INSERT INTO flowch.Flow_Chart_BPCO_DG_&an_N.
                SELECT
                        7,
                        COUNT(DISTINCT BEN_IDT_ANO)
        FROM res.T_INDI_BPCO_DG_&an_N.
        WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM delivrances_CSI_broncho);

        * On conserve les id des patients à supprimer;
        CREATE TABLE travail.exclus3 AS
                SELECT DISTINCT
                        BEN_IDT_ANO
        FROM res.T_INDI_BPCO_DG_&an_N.
        WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM delivrances_CSI_broncho);

quit;

proc delete data = delivrances_CSI delivrances_CSI_broncho;
run; quit;

*
        ************************************************************************************************************
***********************************;
*       On exclut tous les patients;

data exclus;
        set     travail.exclus1-travail.exclus3;
run;

proc sql;

        * On insère l effectif dans le Flowchart;
        INSERT INTO flowch.Flow_Chart_BPCO_DG_&an_N.
                SELECT
                        8,
                        COUNT(DISTINCT BEN_IDT_ANO)
        FROM exclus;

        DELETE FROM res.T_INDI_BPCO_DG_&an_N.
```

```
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM exclus);

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_DG_&an_N.
		SELECT
			9,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM res.T_INDI_BPCO_DG_&an_N.;

quit;

proc delete data = exclus;
run; quit;

proc datasets library = travail memtype = data nolist;
	delete exclus1-exclus3;
run; quit;

*
	*************************************************************************************************************
************************************	;
*	EXCLUSION DES PATIENTS DÉJÀ DIAGNOSTIQUÉS

	;
*
	*************************************************************************************************************
************************************	;

*
	*************************************************************************************************************
************************************;
*	Exclusion des patients en ALD BPCO active à la date index;

proc sql;

	CREATE TABLE ALD_BPCO AS
		SELECT DISTINCT
			a.BEN_IDT_ANO
		FROM res.T_INDI_BPCO_DG_&an_N. a
			INNER JOIN travail.Histo_ALD b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.reperage = 52 AND (b.date_debut <= a.BPCO_DG_Date_index <= b.date_fin OR (b.date_debut <=
a.BPCO_DG_Date_index AND
			b.date_fin = "01JAN1600"d));

quit;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_DG_&an_N.
		SELECT
			10,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM res.T_INDI_BPCO_DG_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM ALD_BPCO);

	* On conserve les id des patients à supprimer;
	CREATE TABLE travail.exclus1 AS
		SELECT DISTINCT
			BEN_IDT_ANO
	FROM res.T_INDI_BPCO_DG_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM ALD_BPCO);

quit;

proc delete data = ALD_BPCO;
run; quit;
```

```
*
	**********************************************************************************************************
**********************************;
*	Exclusion des patients avec au moins une délivrance de BDLA entre 730 jours et 365 jours précédents la date index;

proc sql;

	CREATE TABLE BDLA_avant_index AS
		SELECT DISTINCT
			a.BEN_IDT_ANO
		FROM res.T_INDI_BPCO_DG_&an_N. a
			INNER JOIN travail.reperage_BDLA_&Annee_2N._&Annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE a.BPCO_DG_Date_index - 730 <= b.date_debut <= a.BPCO_DG_Date_index - 365;

quit;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_DG_&an_N.
		SELECT
			11,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM res.T_INDI_BPCO_DG_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM BDLA_avant_index);

	* On conserve les id des patients à supprimer;
	CREATE TABLE travail.exclus2 AS
		SELECT DISTINCT
			BEN_IDT_ANO
	FROM res.T_INDI_BPCO_DG_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM BDLA_avant_index);

quit;

proc delete data = BDLA_avant_index;
run; quit;

*
	**********************************************************************************************************
**********************************;
*	Exclusion des patients avec au moins un diag de BPCO codé entre 730 jours précédents la date index;

proc sql;

	CREATE TABLE hospit_MCO AS
		SELECT
			a.BEN_IDT_ANO
		FROM res.T_INDI_BPCO_DG_&an_N. a
			INNER JOIN travail.reperage_hospit_BPCO_&annee_4N._&Annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE domaine = "MCO" AND a.BPCO_DG_Date_index - 730 <= b.date_fin <= a.BPCO_DG_Date_index;

	CREATE TABLE hospit_SSR_HAD AS
		SELECT
			a.BEN_IDT_ANO
		FROM res.T_INDI_BPCO_DG_&an_N. a
			INNER JOIN travail.reperage_hospit_BPCO_&annee_4N._&Annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE domaine IN ("SSR", "HAD") AND b.date_debut <= a.BPCO_DG_Date_index AND b.date_fin >=
a.BPCO_DG_Date_index - 730;

quit;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_DG_&an_N.
		SELECT
			12,
```

```
                                COUNT(DISTINCT BEN_IDT_ANO)
        FROM res.T_INDI_BPCO_DG_&an_N.
        WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM hospit_MCO) OR BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM
hospit_SSR_HAD);

        * On conserve les id des patients à supprimer;
        CREATE TABLE travail.exclus3 AS
                SELECT DISTINCT
                        BEN_IDT_ANO
        FROM res.T_INDI_BPCO_DG_&an_N.
        WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM hospit_MCO) OR BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM
hospit_SSR_HAD);

quit;

proc delete data = hospit_MCO hospit_SSR_HAD;
run; quit;

*
        ************************************************************************************************************
************************************;
*       Exclusion des patients avec une spiro ou un EFR entre 730 jours et 365 jours précédant la date index;

*       Changement août 2021 - Nouvelle version des indicateurs : on modifie les exclusions;
*       On exclut entre 730 jours et 365 jours précédant la date index (avant = dans les 730 jours précédents la date index);

proc sql;

        CREATE TABLE Spiro_EFR AS
                SELECT
                        a.BEN_IDT_ANO
                FROM res.T_INDI_BPCO_DG_&an_N. a
                        INNER JOIN travail.reperage_EFR_spiro_&Annee_2N._&annee_N1. b
                                ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
                WHERE b.date_debut < a.BPCO_DG_Date_index - 365 AND a.BPCO_DG_Date_index - 730 <= b.date_fin;

quit;

proc sql;

        * On insère l effectif dans le Flowchart;
        INSERT INTO flowch.Flow_Chart_BPCO_DG_&an_N.
                SELECT
                        13,
                        COUNT(DISTINCT BEN_IDT_ANO)
        FROM res.T_INDI_BPCO_DG_&an_N.
        WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM Spiro_EFR);

        * On conserve les id des patients à supprimer;
        CREATE TABLE travail.exclus4 AS
                SELECT DISTINCT
                        BEN_IDT_ANO
        FROM res.T_INDI_BPCO_DG_&an_N.
        WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM Spiro_EFR);

quit;

proc delete data = Spiro_EFR;
run; quit;

*
        ************************************************************************************************************
************************************;
*       On exclut tous les patients;

data exclus;
        set     travail.exclus1-travail.exclus4;
run;

proc sql;
```

```
	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_DG_&an_N.
		SELECT
			14,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM exclus;

	DELETE FROM res.T_INDI_BPCO_DG_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM exclus);

quit;

proc delete data = exclus;
run; quit;

proc datasets library = travail memtype = data nolist;
	delete exclus1-exclus4;
run; quit;

* On insère la population d étude dans le flowchart;
proc sql;

	INSERT INTO flowch.Flow_Chart_BPCO_DG_&an_N.
		SELECT
			15,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM res.T_INDI_BPCO_DG_&an_N.;

quit;


*
	************************************************************************************************************
************************************	;
*	PATIENTS POUR LESQUELS LA SPIROMÉTRIE OU L EFR EST CONTRE INDIQUÉ

		;
*
	************************************************************************************************************
************************************	;

*	Changement août 2021 - Nouvelle version des indicateurs : on modifie les exclusions;
*	On n'exclut plus les patients pour lesquels la spirométrie ou l EFR est contre indiqué;

*
	************************************************************************************************************
************************************	;
*	PATIENTS POUR LESQUELS LA SPIROMÉTRIE OU L EFR N EST PAS RÉALISABLE

			;
*
	************************************************************************************************************
************************************	;

*
	************************************************************************************************************
************************************;
*	Exclusion des patients en ALD maladie d Alzheimer et autres démences active 365 jours après la date index;

proc sql;

	CREATE TABLE ALD_Alzheimer AS
		SELECT DISTINCT
			a.BEN_IDT_ANO
		FROM res.T_INDI_BPCO_DG_&an_N. a
			INNER JOIN travail.Histo_ALD b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.reperage IN (50, 51)
			AND (b.date_debut <= a.BPCO_DG_Date_index <= b.date_fin OR (b.date_debut <=
a.BPCO_DG_Date_index AND b.date_fin = "01JAN1600"d)
			OR a.BPCO_DG_Date_index <= b.date_debut <= (a.BPCO_DG_Date_index + 365));
```

```
quit;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_DG_&an_N.
		SELECT
			16,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM res.T_INDI_BPCO_DG_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM ALD_Alzheimer);

	* On conserve les id des patients à supprimer;
	CREATE TABLE travail.exclus1 AS
		SELECT DISTINCT
			BEN_IDT_ANO
	FROM res.T_INDI_BPCO_DG_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM ALD_Alzheimer);

quit;

proc delete data = ALD_Alzheimer;
run; quit;

*
	**********************************************************************************************************
***********************************;
*	Exclusion des patients ayant eu un diagnostic de soins palliatif codé lors d'un séjour hospitalier MCO terminé dans les 365
jours après
  la date index, ou lors d'un séjour hospitalier HAD, SSR dans les 365 jours après la date index;

proc sql undo_policy = none;

	CREATE TABLE soin_palliatif_MCO AS
		SELECT DISTINCT
			a.BEN_IDT_ANO
		FROM res.T_INDI_BPCO_DG_&an_N. a
			INNER JOIN travail.soin_palliatif_&annee_2N._&annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE a.BPCO_DG_Date_index <= b.date_fin <= a.BPCO_DG_Date_index + 365 AND b.domaine = "MCO";

	CREATE TABLE soin_palliatif_SSR_HAD AS
		SELECT DISTINCT
			a.BEN_IDT_ANO
		FROM res.T_INDI_BPCO_DG_&an_N. a
			INNER JOIN travail.soin_palliatif_&annee_2N._&annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.domaine IN ("HAD", "SSR") AND b.date_debut <= a.BPCO_DG_Date_index + 365 AND b.date_fin >=
a.BPCO_DG_Date_index;

quit;

data soin_palliatif;
	set soin_palliatif_MCO
		soin_palliatif_SSR_HAD;
run;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_DG_&an_N.
		SELECT
			17,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM res.T_INDI_BPCO_DG_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM soin_palliatif);

	* On conserve les id des séjours à supprimer;
	CREATE TABLE travail.exclus2 AS
		SELECT DISTINCT
```

```
                                        BEN_IDT_ANO
        FROM res.T_INDI_BPCO_DG_&an_N.
        WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM soin_palliatif);

quit;

proc datasets library = work memtype = data nolist;
        delete soin_palliatif_MCO soin_palliatif_SSR_HAD soin_palliatif ;
run; quit;

*
        ************************************************************************************************************
**********************************;
*       On flag les patients pour lesquels la spirométrie ou les EFR ne sont pas réalisables;

data exclus;
        set     travail.exclus1-travail.exclus2;
run;

proc sql;

        * On insère l effectif dans le Flowchart;
        INSERT INTO flowch.Flow_Chart_BPCO_DG_&an_N.
                SELECT
                        18,
                        COUNT(DISTINCT BEN_IDT_ANO)
        FROM exclus;

        DELETE FROM res.T_INDI_BPCO_DG_&an_N.
        WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM exclus);

quit;

proc delete data = exclus;
run; quit;

proc datasets library = travail memtype = data nolist;
        delete exclus1-exclus2;
run; quit;


*
        ************************************************************************************************************
**********************************;
*       Population cible;

proc sql;

        INSERT INTO flowch.Flow_Chart_BPCO_DG_&an_N.
                SELECT
                        20,
                        COUNT(DISTINCT BEN_IDT_ANO)
        FROM res.T_INDI_BPCO_DG_&an_N.;

quit;

proc sql;

        SELECT COUNT(*), COUNT(DISTINCT BEN_IDT_ANO)
        FROM res.T_INDI_BPCO_DG_&an_N.;

quit;
```

3.6.3 02_Traitements_et_population_cible.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*

*/
/*		Indicateur 01 : Repérage des traitements

*/
/*

*/
/*
	*****************************************************************************************************************
**********************************	*/

*
	*****************************************************************************************************************
**********************************;
*	Délivrance remboursée d antibiothérapie pour infections respiratoires l année N précédée de délivrances remboursées d
antibiothérapie
*	pour infections respiratoires dans les 365 jours;

proc sql;

	CREATE TABLE patients_antibio AS
		SELECT
			a.BEN_IDT_ANO,
			b.date_debut AS date_deliv_antibio_&Annee_N. format ddmmyy10. length = 4
		FROM res.T_INDI_BPCO_DG_&an_N. a
			INNER JOIN travail.reperage_antibio_&Annee_1N._&Annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE YEAR(b.date_debut) = &Annee_N. AND b.reperage = 36
		ORDER BY 1, 2;

quit;

* On récupère toutes les lignes d antibio dans les 365 jours précédants la délivrance d antibio de l année N et on compte les délivrances;
proc sql undo_policy = none;

	CREATE TABLE patients_antibio AS
		SELECT
			a.BEN_IDT_ANO,
			a.date_deliv_antibio_&Annee_N.,
			COUNT(DISTINCT b.date_debut) AS nb_deliv length = 3
		FROM patients_antibio a
			INNER JOIN travail.reperage_antibio_&Annee_1N._&Annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE (a.date_deliv_antibio_&Annee_N. - 365) <= b.date_debut < a.date_deliv_antibio_&Annee_N. AND
b.reperage = 36
		GROUP BY a.BEN_IDT_ANO, date_deliv_antibio_&Annee_N.
		ORDER BY a.BEN_IDT_ANO, date_deliv_antibio_&Annee_N.;

	CREATE TABLE patients_antibio AS
		SELECT
			BEN_IDT_ANO,
			MAX(nb_deliv) AS nb_deliv length = 3
		FROM patients_antibio a
		GROUP BY BEN_IDT_ANO;

quit;

* On ajoute l information dans la table de résultats;
data res.T_INDI_BPCO_DG_&an_N.;
	merge	res.T_INDI_BPCO_DG_&an_N. (in = a)
			patients_antibio (in = b);
	by BEN_IDT_ANO;
```

```
	if a;
	length nb_deliv ATB_Inf_Resp_2 ATB_Inf_Resp_3 3.;
	if not b then
		nb_deliv = 0;
	ATB_Inf_Resp_2 = 0;
	if nb_deliv >= 1 then
		ATB_Inf_Resp_2 = 1;
	ATB_Inf_Resp_3 = 0;
	if nb_deliv >= 2 then
		ATB_Inf_Resp_3 = 1;
run;

proc delete data = patients_antibio;
run; quit;

*
	*************************************************************************************************************
**********************************;
*	Délivrance remboursée de bronchodilatateur anticholinergique ou Bêta-2 longue durée d action délivrée l année N;

proc sql;

	CREATE TABLE patients_LDA AS
		SELECT DISTINCT
			a.BEN_IDT_ANO,
			1 AS broncho_LDA length = 3
		FROM res.T_INDI_BPCO_DG_&an_N. a
			INNER JOIN travail.reperage_BDLA_&Annee_2N._&Annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE YEAR(b.date_debut) = &Annee_N.
		ORDER BY 1, 2;

quit;

* On ajoute l information dans la table de résultats;
data res.T_INDI_BPCO_DG_&an_N.;
	merge	res.T_INDI_BPCO_DG_&an_N. (in = a)
			patients_LDA (in = b);
	by BEN_IDT_ANO;
	if a;
	if broncho_LDA = . then
		broncho_LDA = 0;
run;

proc delete data = patients_LDA;
run; quit;

*
	*************************************************************************************************************
**********************************;
*	Population cible;

data res.T_INDI_BPCO_DG_&an_N.;
	set res.T_INDI_BPCO_DG_&an_N.;
	BPCO_DG_CIBLE = 1;
run;

proc sort data = res.T_INDI_BPCO_DG_&an_N. nodupkey;
	by _all_;
run;

*
	*************************************************************************************************************
**********************************;
*	Nombre de délivrances remboursées d antibiothérapie pour infections respiratoires l année N;

proc sql;

	CREATE TABLE nb_antibio AS
		SELECT
			a.BEN_IDT_ANO,
```

```
                        COUNT(DISTINCT b.date_debut) AS Nb_ATB_Inf_Resp length = 3
                FROM res.T_INDI_BPCO_DG_&an_N. a
                        INNER JOIN travail.reperage_antibio_&Annee_1N._&Annee_N1. b
                                ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
                WHERE YEAR(b.date_debut) = &Annee_N. AND b.reperage = 36
                GROUP BY a.BEN_IDT_ANO
                ORDER BY a.BEN_IDT_ANO;

quit;

* On ajoute l information dans la table de résultats;
data res.T_INDI_BPCO_DG_&an_N.;
        merge   res.T_INDI_BPCO_DG_&an_N. (in = a)
                nb_antibio (in = b);
        by BEN_IDT_ANO;
        if a;
        if Nb_ATB_Inf_Resp = . then
                Nb_ATB_Inf_Resp = 0;
run;

proc delete data = nb_antibio;
run; quit;

*
        ************************************************************************************************************
***********************************;
*       Nombre de délivrances remboursées d'antibiothérapie pour infections respiratoires dans les 365 jours précédant la date
index;

proc sql;

        CREATE TABLE nb_antibio AS
                SELECT
                        a.BEN_IDT_ANO,
                        COUNT(DISTINCT b.date_debut) AS Nb_ATB_Inf_Resp_AV365J length = 3
                FROM res.T_INDI_BPCO_DG_&an_N. a
                        INNER JOIN travail.reperage_antibio_&Annee_1N._&Annee_N1. b
                                ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
                WHERE (a.BPCO_DG_Date_index - 365) <= b.date_debut < BPCO_DG_Date_index AND b.reperage = 36
                GROUP BY a.BEN_IDT_ANO
                ORDER BY a.BEN_IDT_ANO;

quit;

* On ajoute l information dans la table de résultats;
data res.T_INDI_BPCO_DG_&an_N.;
        merge   res.T_INDI_BPCO_DG_&an_N. (in = a)
                nb_antibio (in = b);
        by BEN_IDT_ANO;
        if a;
        if Nb_ATB_Inf_Resp_AV365J = . then
                Nb_ATB_Inf_Resp_AV365J = 0;
run;

proc delete data = nb_antibio;
run; quit;

*
        ************************************************************************************************************
***********************************;
*       Nombre de délivrances remboursées de bronchodilatateur anticholinergique ou Bêta-2 courte durée d action l année N;

proc sql;

        CREATE TABLE nb_BDCA AS
                SELECT
                        a.BEN_IDT_ANO,
                        COUNT(DISTINCT b.date_debut) AS Nb_Broncho_CDA length = 3
                FROM res.T_INDI_BPCO_DG_&an_N. a
                        INNER JOIN travail.reperage_BDCA_tabac_&Annee_2N._&Annee_N1. b
                                ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
```

```
                        WHERE YEAR(b.date_debut) = &Annee_N. AND b.reperage in (42, 43, 44)
                        GROUP BY a.BEN_IDT_ANO
                        ORDER BY 1, 2;

quit;

* On ajoute l information dans la table de résultats;
data res.T_INDI_BPCO_DG_&an_N.;
        merge   res.T_INDI_BPCO_DG_&an_N. (in = a)
                        nb_BDCA (in = b);
        by BEN_IDT_ANO;
        if a;
        if Nb_Broncho_CDA = . then
                Nb_Broncho_CDA = 0;
run;

proc delete data = nb_BDCA;
run; quit;

*
        ****************************************************************************************************
**********************************;
*       Nombre de délivrances remboursées de bronchodilatateur anticholinergique ou Bêta-2 courte durée d'action dans les 365
jours précédant la
date index;

proc sql;

        CREATE TABLE nb_BDCA AS
                SELECT
                        a.BEN_IDT_ANO,
                        COUNT(DISTINCT b.date_debut) AS Nb_Broncho_CDA_AV365j length = 3
                FROM res.T_INDI_BPCO_DG_&an_N. a
                        INNER JOIN travail.reperage_BDCA_tabac_&Annee_2N._&Annee_N1. b
                                ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
                WHERE (a.BPCO_DG_Date_index - 365) <= b.date_debut < BPCO_DG_Date_index  AND b.reperage in (42, 43, 44)
                GROUP BY a.BEN_IDT_ANO
                ORDER BY 1, 2;

quit;

* On ajoute l information dans la table de résultats;
data res.T_INDI_BPCO_DG_&an_N.;
        merge   res.T_INDI_BPCO_DG_&an_N. (in = a)
                        nb_BDCA (in = b);
        by BEN_IDT_ANO;
        if a;
        if Nb_Broncho_CDA_AV365j = . then
                Nb_Broncho_CDA_AV365j = 0;
run;

proc delete data = nb_BDCA;
run; quit;

*
        ****************************************************************************************************
**********************************;
*       Nombre de délivrances remboursées de substitus nicotiniques l année N;

proc sql;

        CREATE TABLE Nb_Subs_Nico AS
                SELECT
                        a.BEN_IDT_ANO,
                        COUNT(DISTINCT b.date_debut) AS Nb_Subs_Nico length = 3
                FROM res.T_INDI_BPCO_DG_&an_N. a
                        INNER JOIN travail.reperage_BDCA_tabac_&Annee_2N._&Annee_N1. b
                                ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
                WHERE YEAR(b.date_debut) = &Annee_N. AND b.reperage = 35
                GROUP BY a.BEN_IDT_ANO
                ORDER BY 1, 2;
```

```
quit;

* On ajoute l information dans la table de résultats;
data res.T_INDI_BPCO_DG_&an_N.;
	merge	res.T_INDI_BPCO_DG_&an_N. (in = a)
			Nb_Subs_Nico (in = b);
	by BEN_IDT_ANO;
	if a;
	if Nb_Subs_Nico = . then
		Nb_Subs_Nico = 0;
run;

proc delete data = Nb_Subs_Nico;
run; quit;
```

3.6.4 03_Realisation_EFR_ou_spiro.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*

*/
/*		Indicateur 01 : EFR ou Spirométries

*/
/*

*/
/*
	*****************************************************************************************************************
**********************************	*/

%macro EFR_Spiro(time=);

	*** Définition des bornes basses et bornes hautes pour les repérages;
	* time = 1 : Avant ou après inclusion;
	%if &time. = 1 %then
		%do;

			%let bb = BPCO_DG_Date_index - 365;
			%let bh = BPCO_DG_Date_index + 365;
			%let suf = ;

		%end;

	* time = 2 : Avant inclusion;
	%if &time. = 2 %then
		%do;

			%let bb = BPCO_DG_Date_index - 365;
			%let bh = BPCO_DG_Date_index;
			%let suf = _AV_365;

		%end;

	* time = 3 : Après inclusion;
	%if &time. = 3 %then
		%do;

			%let bb = BPCO_DG_Date_index + 1;
			%let bh = BPCO_DG_Date_index + 365;
			%let suf = _AP_365;

		%end;

	*
	*****************************************************************************************************************
********************************;
	*		On repère les patients avec une spirométrie dans les 365 jours avant ou suivant la date index;

	proc sql;

		CREATE TABLE spirometrie AS
			SELECT DISTINCT
				a.BEN_IDT_ANO,
				MIN(b.date_debut) AS date_debut length = 4,
				MIN(CASE WHEN b.date_debut < (a.&bb.) THEN a.&bb.
									ELSE b.date_debut
									END) AS Date_Spiro&suf. length = 4,
				COUNT(DISTINCT b.date_debut) AS Nb_spiro&suf. length = 3
			FROM res.T_INDI_BPCO_DG_&an_N. a
				INNER JOIN travail.reperage_EFR_spiro_&Annee_2N._&annee_N1. b
					ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
```

```
                        WHERE b.reperage = 2 AND b.date_debut <= a.&bh. AND b.date_fin >= a.&bb.
                        GROUP BY a.BEN_IDT_ANO;

                * On fait un max car si DCIR + PMSI, c est un doublon donc la spiro a lieu dans un etb;
                CREATE TABLE lieu_spiro AS
                        SELECT DISTINCT
                                a.BEN_IDT_ANO,
                                MAX(CASE WHEN b.domaine = "" THEN 1
                                                WHEN b.domaine = "MCO" THEN 2
                                                WHEN b.domaine = "SSR" THEN 3
                                                END) AS Lieu_Spiro&suf. length = 3
                                FROM spirometrie a
                                        LEFT JOIN travail.reperage_EFR_spiro_&Annee_2N._&annee_N1. b
                                                ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
                                                AND a.date_debut = b.date_debut
                                WHERE b.reperage = 2
                                GROUP BY a.BEN_IDT_ANO;

        quit;

        * On ajoute l information dans la table de résultats;

        proc sort data = res.T_INDI_BPCO_DG_&an_N.;
                by BEN_IDT_ANO;
        run;

        proc sort data = spirometrie;
                by BEN_IDT_ANO;
        run;

        proc sort data = lieu_spiro;
                by BEN_IDT_ANO;
        run;

        data res.T_INDI_BPCO_DG_&an_N.;
                merge   res.T_INDI_BPCO_DG_&an_N. (in = a)
                        spirometrie (in = b drop = date_debut)
                        lieu_spiro (in = c);
                by BEN_IDT_ANO;
                length Spiro&suf. 3.;
                Spiro&suf. = 0;
                if Nb_spiro&suf. > 0 then
                        Spiro&suf. = 1;
                if Nb_spiro&suf. = . then
                        Nb_spiro&suf. = 0;
                if a then
                        output;
        run;

        proc delete data = spirometrie lieu_spiro;
        run; quit;

        *
        ************************************************************************************************************
*********************************;
        *       On repère les patients avec une EFR dans les 365 jours avant ou suivant la date index;

        proc sql;

                CREATE TABLE EFR AS
                        SELECT DISTINCT
                                a.BEN_IDT_ANO,
                                MIN(b.date_debut) AS date_debut,
                                MIN(CASE WHEN b.date_debut < (a.&bb.) THEN a.&bb.
                                                ELSE b.date_debut
                                                END) AS Date_EFR&suf. length = 4,
                                COUNT(DISTINCT b.date_debut) AS Nb_EFR&suf. length = 3
                        FROM res.T_INDI_BPCO_DG_&an_N. a
                                INNER JOIN travail.reperage_EFR_spiro_&Annee_2N._&annee_N1. b
                                        ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
                        WHERE b.reperage = 1 AND b.date_debut <= a.&bh. AND b.date_fin >= a.&bb.
```

```
                                GROUP BY a.BEN_IDT_ANO;

                        * On fait un max car si DCIR + PMSI, c est un doublon donc l EFR a lieu dans un etb;
                        CREATE TABLE lieu_EFR AS
                                SELECT DISTINCT
                                        a.BEN_IDT_ANO,
                                        MAX(CASE WHEN b.domaine = "" THEN 1
                                                                WHEN b.domaine = "MCO" THEN 2
                                                                WHEN b.domaine = "SSR" THEN 3
                                                                END) AS Lieu_EFR&suf. length = 3
                                        FROM EFR a
                                                LEFT JOIN travail.reperage_EFR_spiro_&Annee_2N._&annee_N1. b
                                                        ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
                                                        AND a.date_debut = b.date_debut
                                        WHERE b.reperage = 1
                                        GROUP BY a.BEN_IDT_ANO;

        quit;

        * On ajoute l information dans la table de résultats;

        proc sort data = res.T_INDI_BPCO_DG_&an_N.;
                by BEN_IDT_ANO;
        run;

        proc sort data = EFR;
                by BEN_IDT_ANO;
        run;

        proc sort data = lieu_EFR;
                by BEN_IDT_ANO;
        run;

        data res.T_INDI_BPCO_DG_&an_N.;
                merge   res.T_INDI_BPCO_DG_&an_N. (in = a)
                        EFR (in = b drop = date_debut)
                        lieu_EFR (in = c);
                by BEN_IDT_ANO;
                length EFR&suf. 3.;
                EFR&suf. = 0;
                if Nb_EFR&suf. > 0 then
                        EFR&suf. = 1;
                if Nb_EFR&suf. = . then
                        Nb_EFR&suf. = 0;
                if a then
                        output;
        run;

        proc delete data = EFR lieu_EFR;
        run; quit;

%mend EFR_Spiro;

* Avant ou après inclusion : time = 1;
%EFR_Spiro (time = 1);

* Avant inclusion : time = 2;
%EFR_Spiro (time = 2);

* Après inclusion : time = 3;
%EFR_Spiro (time = 3);

* Flag BPCO_DG_OBS;
data res.T_INDI_BPCO_DG_&an_N.;
        set res.T_INDI_BPCO_DG_&an_N.;
        length BPCO_DG_OBS 3.;
        BPCO_DG_OBS = 0;
        if EFR = 1 or Spiro = 1 then
                BPCO_DG_OBS = 1;
run;
```

3.6.5 04_Pneumothorax_infartcus.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*			Indicateur 01 : Pneumothorax ou infarctus
                                                                                                                    */
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/

%macro pneumo_infar(time=);

	*** Définition des bornes basses et bornes hautes pour les repérages;
	* time = 1 : Avant ou après inclusion;
	%if &time. = 1 %then
		%do;

			%let bb = BPCO_DG_Date_index - 365;
			%let bh = BPCO_DG_Date_index + 365;
			%let suf = ;

		%end;

	* time = 2 : Avant inclusion;
	%if &time. = 2 %then
		%do;

			%let bb = BPCO_DG_Date_index - 365;
			%let bh = BPCO_DG_Date_index;
			%let suf = _AV_365;

		%end;

	* time = 3 : Après inclusion;
	%if &time. = 3 %then
		%do;

			%let bb = BPCO_DG_Date_index + 1;
			%let bh = BPCO_DG_Date_index + 365;
			%let suf = _AP_365;

		%end;

	*
	*****************************************************************************************************************
*******************************;
	*		On repère les patients avec une pneumothorax dans les 365 jours avant ou suivant la date index;

	proc sql;

		CREATE TABLE pneumothorax AS
			SELECT DISTINCT
				a.BEN_IDT_ANO
			FROM res.T_INDI_BPCO_DG_&an_N. a
				INNER JOIN travail.pneumothorax_infarctus_&Annee_1N._&annee_N1. b
					ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
			WHERE b.reperage = 11 AND b.date_debut <= a.&bh. AND b.date_fin >= a.&bb.;

	quit;

	* On ajoute l information dans la table de résultats;
```

```
	proc sort data = res.T_INDI_BPCO_DG_&an_N.;
		by BEN_IDT_ANO;
	run;

	proc sort data = pneumothorax;
		by BEN_IDT_ANO;
	run;

	data res.T_INDI_BPCO_DG_&an_N.;
		merge	res.T_INDI_BPCO_DG_&an_N. (in = a)
			pneumothorax (in = b);
		by BEN_IDT_ANO;
		length pneumothorax&suf. 3.;
		pneumothorax&suf. = 0;
		if b then
			pneumothorax&suf. = 1;
		if a then
			output;
	run;

	proc delete data = pneumothorax;
	run; quit;

	*
	**********************************************************************************************************
*********************************;
	*	On repère les patients avec un infarctus dans les 365 jours avant ou suivant la date index;

	proc sql;

		CREATE TABLE infarctus AS
			SELECT DISTINCT
				a.BEN_IDT_ANO
			FROM res.T_INDI_BPCO_DG_&an_N. a
				INNER JOIN travail.pneumothorax_infarctus_&Annee_1N._&annee_N1. b
					ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
			WHERE b.reperage = 12 AND b.date_debut <= a.&bh. AND b.date_fin >= a.&bb.;

	quit;

	* On ajoute l information dans la table de résultats;

	proc sort data = res.T_INDI_BPCO_DG_&an_N.;
		by BEN_IDT_ANO;
	run;

	proc sort data = infarctus;
		by BEN_IDT_ANO;
	run;

	data res.T_INDI_BPCO_DG_&an_N.;
		merge	res.T_INDI_BPCO_DG_&an_N. (in = a)
			infarctus (in = b);
		by BEN_IDT_ANO;
		length infarctus&suf. 3.;
		infarctus&suf. = 0;
		if b then
			infarctus&suf. = 1;
		if a then
			output;
	run;

	proc delete data = infarctus;
	run; quit;

%mend pneumo_infar;

* Avant ou après inclusion : time = 1;
%pneumo_infar (time = 1);
```

```
* Avant inclusion : time = 2;
%pneumo_infar (time = 2);

* Après inclusion : time = 3;
%pneumo_infar (time = 3);
```

3.6.6 05_Table_finale.sas

```
/*
	*****************************************************************************************************
**********************************	*/
/*

*/
/*		Indicateur 01 : Table finale - Ajout de formats et de labels

		*/
/*

*/
/*
	*****************************************************************************************************
**********************************	*/

data res.T_INDI_BPCO_DG_&an_N.;
	set res.T_INDI_BPCO_DG_&an_N. (keep = BEN_IDT_ANO Date_index_broncho Date_index_antibio_memejour
Date_index_antibio_365jour
		Date_index_substitut BPCO_DG_Date_index ATB_Inf_Resp_3 ATB_Inf_Resp_2 ATB_Inf_Meme_Jour
ATB_Inf_365_Jour Broncho_CDA Broncho_LDA Subs_Nico
		BPCO_DG_CIBLE Nb_ATB_Inf_Resp Nb_Broncho_CDA Nb_Subs_Nico BPCO_DG_OBS Spiro Nb_Spiro Date_Spiro
Lieu_Spiro EFR Nb_EFR Date_EFR Lieu_EFR
		Spiro_Av_365 Nb_Spiro_Av_365 Date_Spiro_Av_365 Lieu_Spiro_Av_365 EFR_Av_365 Nb_EFR_Av_365
Date_EFR_Av_365 Lieu_EFR_Av_365
		Spiro_AP_365 Nb_Spiro_AP_365 Date_Spiro_AP_365 Lieu_Spiro_AP_365 EFR_AP_365 Nb_EFR_AP_365
Date_EFR_AP_365 Lieu_EFR_AP_365 Pneumothorax
		Pneumothorax_AV_365 Pneumothorax_AP_365 Infarctus Infarctus_AV_365 Infarctus_AP_365
Nb_ATB_Inf_Resp_AV365J Nb_Broncho_CDA_AV365j);
	label
		BEN_IDT_ANO = "Numéro d'indivdu"
		Date_index_broncho = "Date de la première délivrance de bronchodilatateur anticholinergique longue durée
d'action ou Bêta-2 longue durée d'action en &Annee_N."
		Date_index_antibio_memejour = "Date de la première délivrance remboursée d'antibiothérapie en &Annee_N.
associée le même jour d'une délivrance remboursée de BDCA précédée dans les 365 jours d'au moins une délivrance remboursée
d'antibiothérapie"
		Date_index_antibio_365jour = "Date de la première délivrance remboursée d'antibiothérapie en &Annee_N.
précédée dans les 365 jours d'au moins une délivrance remboursée d'antibiothérapie associée le même jour d'une délivrance
remboursée de BDCA"
		Date_index_substitut = "Date de la première délivrance de substituts nicotiniques (remboursable ou forfait) ou
d'un traitement d'arrêt du tabac en &Annee_N."
		BPCO_DG_Date_index = "Date index retenue"
		ATB_Inf_Resp_3 = "Patient ayant eu au moins une délivrance remboursée d'antibiothérapie pour infections
respiratoires en &Annee_N. précédée d'au moins 2 délivrances remboursées d'antibiothérapie pour infections respiratoires dans les
365 jours"
		ATB_Inf_Resp_2 = "Patient ayant eu au moins une délivrance remboursée d'antibiothérapie pour infections
respiratoires en &Annee_N. précédée d'au moins une délivrance remboursée d'antibiothérapie pour infections respiratoires dans les
365 jours"
		ATB_Inf_meme_jour = "Patients ayant eu au moins une délivrance remboursée d'antibiothérapie &Annee_N.
associée le même jour d'une délivrance remboursée de BDCA et précédée dans les 365 jours d'au moins une délivrance remboursée
d'antibiothérapie pour infections respiratoires"
		ATB_Inf_365_Jour = "Patients ayant eu au moins une délivrance remboursée d'antibiothérapie &Annee_N.
précédée dans les 365 jours d'au moins une délivrance remboursée d'antibiothérapie associée le même jour à une délivrance
remboursée de BDCA"
		Broncho_CDA = "Patient ayant eu au moins une délivrance remboursée en &Annee_N. d'un bronchodilatateur
anticholinergique ou Bêta-2 courte durée d'action délivrée le même jour que la cure d'antibiothérapie"
		Broncho_LDA = "Patient ayant eu au moins une délivrance remboursée en Bêta-&Annee_N. d'un
bronchodilatateur anticholinergique longue durée d'action ou Bêta-2 longue durée d'action"
		Subs_Nico = "Patient ayant eu au moins une délivrance remboursée de substituts nicotiniques en &Annee_N."
		BPCO_DG_CIBLE = "Patient correspondant aux critères d'inclusion et d'exclusion de la population cible"
		Nb_ATB_Inf_Resp = "Nombre de délivrances remboursées d'antibiothérapie pour infections respiratoires en
&Annee_N."
		Nb_ATB_Inf_Resp_AV365J = "Nombre de délivrances remboursées d'antibiothérapie pour infections respiratoires
dans les 365 jours précédant la date index"
		Nb_Broncho_CDA = "Nombre de délivrances remboursées de bronchodilatateur anticholinergique ou Bêta-2
courte durée d'action en &Annee_N."
```

```
                    Nb_Broncho_CDA_AV365j = "Nombre de délivrances remboursées de bronchodilatateur anticholinergique ou
Bêta-2 courte durée d'action dans les 365 jours précédant la date index"
                    Nb_Subs_Nico = "Nombre de délivrances remboursées de substituts nicotiniques en &Annee_N. "
                    BPCO_DG_OBS = "Patient de la population cible pour lequel une spirométrie ou des EFR ont été réalisés dans les
365 jours précédant la date index ou  les 365 jours suivant la date index"
                    Spiro = "Spirométrie réalisée dans les 365 jours avant ou suivant la date index"
                    Nb_Spiro = "Nombre de spirométries réalisées dans les 365 jours avant ou suivant la date index"
                    Date_Spiro = "Date de la réalisation de la première spirométrie dans les 365 jours avant ou suivant la date index"
                    Lieu_Spiro = "Lieu de réalisation de la première spirométrie dans les 365 jours avant ou suivant la date index"
                    EFR = "EFR réalisée dans les 365 jours avant ou suivant la date index"
                    Nb_EFR = "Nombre d'EFR réalisées dans les 365 jours avant ou suivant la date index"
                    Date_EFR = "Date de la réalisation de la première EFR dans les 365 jours avant ou suivant la date index"
                    Lieu_EFR = "Lieu de réalisation de la première EFR dans les 365 jours avant ou suivant la date index"

                    Spiro_Av_365 = "Spirométrie réalisée dans les 365 jours avant la date index (incluse)"
                    Nb_Spiro_Av_365 = "Nombre de spirométries réalisées dans les 365 jours avant la date index (incluse)"
                    Date_Spiro_Av_365 = "Date de la réalisation de la première spirométrie dans les 365 jours avant la date index
(incluse)"
                    Lieu_Spiro_Av_365 = "Lieu de réalisation de la première spirométrie avant la date index"
                    EFR_Av_365 = "EFR réalisée dans les 365 jours avant la date index (incluse)"
                    Nb_EFR_Av_365 = "Nombre d'EFR réalisées dans les 365 jours avant la date index (incluse)"
                    Date_EFR_Av_365 = "Date de la réalisation de la première EFR dans les 365 jours avant la date index (incluse)"
                    Lieu_EFR_Av_365 = "Lieu de réalisation de la première EFR dans les 365 jours avant la date index (incluse)"

                    Spiro_Ap_365 = "Spirométrie réalisée dans les 365 jours suivant la date index (exclue)"
                    Nb_Spiro_Ap_365 = "Nombre de spirométries réalisées dans les 365 jours suivant la date index (exclue)"
                    Date_Spiro_Ap_365 = "Date de la réalisation de la première spirométrie dans les 365 jours suivant la date index
(exclue)"
                    Lieu_Spiro_Ap_365 = "Lieu de réalisation de la première spirométrie suivant la date index (exclue)"
                    EFR_Ap_365 = "EFR réalisée dans les 365 jours suivant la date index (exclue)"
                    Nb_EFR_Ap_365 = "Nombre d'EFR réalisées dans les 365 jours suivant la date index (exclue)"
                    Date_EFR_Ap_365 = "Date de la réalisation de la première EFR dans les 365 jours suivant la date index (exclue)"
                    Lieu_EFR_Ap_365 = "Lieu de réalisation de la première EFR dans les 365 jours suivant la date index (exclue)"
                    Pneumothorax = "Patients ayant eu un pneumothorax dans les 365 jours suivant la date index ou dans les 365
jours précédant la date index"
                    Pneumothorax_AV_365 = "Patients ayant eu un pneumothorax dans les 365 jours avant la date index (incluse)"
                    Pneumothorax_AP_365 = "Patients ayant eu un pneumothorax dans les 365 jours suivant la date index (exclue)"
                    Infarctus = "Patients ayant eu un diagnostic d'infarctus dans les 365 jours suivant la date index ou dans les 365
jours précédant la date index"
                    Infarctus_AV_365 = "Patients ayant eu un diagnostic d'infarctus dans les 365 jours avant la date index (incluse)"
                    Infarctus_AP_365 = "Patients ayant eu un diagnostic d'infarctus dans les 365 jours suivant la date index (exclue)"
                    ;
          format ATB_Inf_Resp_3 ATB_Inf_Resp_2 Broncho_CDA Subs_Nico BPCO_DG_CIBLE BPCO_DG_OBS Spiro EFR
ATB_Inf_meme_jour ATB_Inf_365_Jour
                    Broncho_LDA Spiro_Av_365 EFR_Av_365 Spiro_Ap_365 EFR_Ap_365 Pneumothorax Pneumothorax_AV_365
Pneumothorax_AP_365 Infarctus
                    Infarctus_AV_365 Infarctus_AP_365 f_oui_non. BPCO_DG_Date_index Date_index_broncho
Date_index_antibio_memejour Date_index_antibio_365jour
                    Date_index_substitut Date_Spiro Date_EFR Date_Spiro_Av_365 Date_EFR_Av_365 Date_Spiro_Ap_365
Date_EFR_Ap_365 date9. Lieu_Spiro Lieu_EFR
                    Lieu_Spiro_Av_365 Lieu_EFR_Av_365 Lieu_Spiro_Ap_365 Lieu_EFR_Ap_365 f_lieu_soin.;
run;

*
          ************************************************************************************************************
************************************;
*         Réalisation d EFR ou d une spirométrie annuelle;

proc sql;

          INSERT INTO flowch.Flow_Chart_BPCO_DG_&an_N.
                    SELECT
                              21,
                              COUNT(DISTINCT BEN_IDT_ANO)
          FROM res.T_INDI_BPCO_DG_&an_N.
          WHERE BPCO_DG_OBS = 1;

          * Nombre de patients pour les numérateur des premières exclusions;
          SELECT Nb_patients_ref0 into : N0 FROM travail.ref_flowcharts;
          SELECT N into : N4 FROM flowch.Flow_Chart_BPCO_DG_&an_N. WHERE indicateur = 4;
          SELECT N into : N9 FROM flowch.Flow_Chart_BPCO_DG_&an_N. WHERE indicateur = 9;
```

```
	* Population d étude;
	SELECT N into : N15 FROM flowch.Flow_Chart_BPCO_DG_&an_N. WHERE indicateur = 15;

	* Population cible;
	SELECT N into : N20 FROM flowch.Flow_Chart_BPCO_DG_&an_N. WHERE indicateur = 20;

quit;

data flowch.Flow_Chart_BPCO_DG_&an_N.;
	set flowch.Flow_Chart_BPCO_DG_&an_N.;
	if indicateur in (1, 2, 2.5, 3, 4) then
		percent = N / &N0.;
	if indicateur in (5, 6, 7, 8, 9) then
		percent = N / &N4.;
	if indicateur in (10, 11, 12, 13, 14, 15) then
		percent = N / &N9.;
	if indicateur in (16, 17, 18, 19, 20, 21) then
		percent = N / &N15.;
	if indicateur = 21 then
		percent = N / &N20.;
	format indicateur f_ind_01_flowchart. percent percent7.1;
run;

* Vérif;
proc sql;

	SELECT COUNT(*) AS nb_lignes, COUNT(DISTINCT BEN_IDT_ANO) AS nb_patients
	FROM res.T_INDI_BPCO_DG_&an_N.;
	* nb_lignes = nb_patients : OK;

	SELECT COUNT(*) AS nb_lignes, COUNT(DISTINCT BEN_IDT_ANO) AS nb_patients
	FROM res.T_INDI_BPCO_DG_&an_N.
	WHERE BPCO_DG_CIBLE = 1;
	* nb_lignes = nb_patients : OK;

quit;

*
	**********************************************************************************************************
**********************************;
*	On ajoute le flag BPCO_DG_CIBLE dans la table res.T_INDI_BPCO_&an_N.;

proc sql;

	ALTER TABLE res.T_INDI_BPCO_&an_N. DROP BPCO_DG_CIBLE;

	ALTER TABLE res.T_INDI_BPCO_&an_N. ADD BPCO_DG_CIBLE INT format = f_oui_non. length = 3
		label = "Patient appartenant à la population cible BPCO_DG";

	UPDATE res.T_INDI_BPCO_&an_N.
		SET BPCO_DG_CIBLE = CASE	WHEN BEN_IDT_ANO IN (SELECT DISTINCT BEN_IDT_ANO FROM
res.T_INDI_BPCO_DG_&an_N. WHERE

	BPCO_DG_CIBLE = 1) THEN 1
                                                                                                ELSE 0
                                                                                                END;

quit;

data res.T_INDI_BPCO_&an_N.;
	set res.T_INDI_BPCO_&an_N.;
	if BPCO_DG_CIBLE = .
		then BPCO_DG_CIBLE = 0;
run;

*
	**********************************************************************************************************
**********************************;
*	On corrige la variable broncho_LDA, qui est incohérente avec la variable nb_broncho_LDA de la table
res.T_INDI_BPCO_&an_N.;
```

```
*		En effet, certaines délivrances de médicamnts BDLA de 2017 ne remontent que dans les flux de 2019. Ils sont donc pris en
compte dans le;
*		flag broncho_LDA alors qu'ils n étaient pas pris en compte dans le dénombrement nb_broncho_LDA de la table
res.T_INDI_BPCO_&an_N.;
*		On définit broncho_LDA à 0 pour les patients ayant nb_broncho_LDA = 0;

proc sql;

		CREATE TABLE verif_LDA AS
			SELECT
				a.BEN_IDT_ANO,
				a.Broncho_LDA,
				b.Nb_Broncho_LDA
			FROM res.T_INDI_BPCO_DG_17 a
				INNER JOIN res.T_INDI_BPCO_17 b
					ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
			WHERE a.Broncho_LDA = 1 AND b.Nb_Broncho_LDA = 0;

quit;

proc sql;

		UPDATE res.T_INDI_BPCO_DG_&an_N.
			SET Broncho_LDA = 0
			WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM verif_LDA);

quit;

proc delete data = verif_LDA;
run; quit;
```

3.7 Vaccin contre la grippe chez les patients atteints de BPCO

3.7.1 00_Initialisation_du_Flowchart.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*

*/
/*			Table de population - Patients avec indicateur_02a = 1 ou indicateur_02b = 1
*/
/*

*/
/*
	*****************************************************************************************************************
**********************************	*/

* Patients BPCO probable;
data T_INDI_BPCO_VACGA_&an_N. (keep = BEN_IDT_ANO);
	set pop.indicateurs_&an_N. (where = (indicateur_02a = 1));
run;

proc sort data = T_INDI_BPCO_VACGA_&an_N. nodupkey;
	by BEN_IDT_ANO;
run;

* Patients BPCO diagnostiquée;
data T_INDI_BPCO_VACGB_&an_N. (keep = BEN_IDT_ANO);
	set pop.indicateurs_&an_N. (where = (indicateur_02b = 1));
run;

proc sort data = T_INDI_BPCO_VACGB_&an_N. nodupkey;
	by BEN_IDT_ANO;
run;

*
	*****************************************************************************************************************
**********************************;
*	Effectifs pour le flowchart;

*	Individus de 40 ans et plus avec une ALD BPCO active au 1er septembre de l année N ou individus de 40 ans et plus
hospitalisés en MCO, SSR,
	HAD avec un diagnostic BPCO codé;

*	Individus de 40 ans et plus ayant bénéficié de soins remboursés au moins une fois l année N ayant eu un traitement
spécifique de la BPCO (3 BLDLA);

proc sql;

	CREATE TABLE flowch.Flow_Chart_BPCO_VACG_&an_N.  AS
		SELECT *
		FROM (
			SELECT 1 AS indicateur length = 3, COUNT(DISTINCT BEN_IDT_ANO) AS N length = 5 FROM
T_INDI_BPCO_VACGB_&an_N. WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO
				FROM res.T_INDI_BPCO_&an_N. WHERE age >= 40 AND ALD_BPCO_septembre = 1)
			UNION ALL
			SELECT 2, COUNT(DISTINCT BEN_IDT_ANO) FROM T_INDI_BPCO_VACGB_&an_N. WHERE
BEN_IDT_ANO IN (SELECT BEN_IDT_ANO
				FROM res.T_INDI_BPCO_&an_N. WHERE age >= 40 AND Nb_hospit_BPCO_Ind02 > 0)
			UNION ALL
			SELECT 3, COUNT(DISTINCT BEN_IDT_ANO) FROM T_INDI_BPCO_VACGB_&an_N.
			UNION ALL
			SELECT 4, COUNT(DISTINCT BEN_IDT_ANO) FROM T_INDI_BPCO_VACGA_&an_N.
		);
```

```
quit;
```

3.7.2 01_Exclusions_BPCO_probable.sas

```
/*
	*****************************************************************************************************
**********************************	*/
/*

                                                                                                        */
/*			Indicateur 02 : Exclusions pour les patients BPCO probable

				*/
/*

                                                                                                        */
/*
	*****************************************************************************************************
**********************************	*/

*
	*****************************************************************************************************
**********************************	;
*	EXCLUSION DES PATIENTS ASTHMATIQUES

                                                                        ;
*
	*****************************************************************************************************
**********************************	;

*
	*****************************************************************************************************
**********************************;
*	Exclusion des patients en ALD asthme active au 1er septembre de l année N;

proc sql;

	CREATE TABLE ALD_asthme AS
		SELECT DISTINCT
			a.BEN_IDT_ANO
		FROM T_INDI_BPCO_VACGA_&an_N. a
			INNER JOIN travail.Histo_ALD b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.reperage = 53 AND (b.date_debut <= "01SEP&Annee_N."d <= b.date_fin OR (b.date_debut <=
"01SEP&Annee_N."d AND
			b.date_fin = "01JAN1600"d));

quit;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_VACG_&an_N.
		SELECT
			5,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM T_INDI_BPCO_VACGA_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM ALD_asthme);

	* On conserve les id des patients à supprimer;
	CREATE TABLE travail.exclus1 AS
		SELECT DISTINCT
			BEN_IDT_ANO
	FROM T_INDI_BPCO_VACGA_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM ALD_asthme);

quit;

proc delete data = ALD_asthme;
run; quit;
```

```
*
	****************************************************************************************************
**********************************;
*	Exclusion des patients en ALD mucoviscidose active au 1er septembre de l année N;

proc sql;

	CREATE TABLE ALD_mucoviscidose AS
		SELECT DISTINCT
			a.BEN_IDT_ANO
		FROM T_INDI_BPCO_VACGA_&an_N. a
			INNER JOIN travail.Histo_ALD b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.reperage = 54 AND (b.date_debut <= "01SEP&Annee_N."d <= b.date_fin OR (b.date_debut <=
"01SEP&Annee_N."d AND
			b.date_fin = "01JAN1600"d));

quit;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_VACG_&an_N.
		SELECT
			6,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM T_INDI_BPCO_VACGA_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM ALD_mucoviscidose);

	* On conserve les id des patients à supprimer;
	CREATE TABLE travail.exclus2 AS
		SELECT DISTINCT
			BEN_IDT_ANO
	FROM T_INDI_BPCO_VACGA_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM ALD_mucoviscidose);

quit;

proc delete data = ALD_mucoviscidose;
run; quit;

*
	****************************************************************************************************
**********************************;
*	Exclusion des patients avec au moins 3 délivrances remboursées de CSI seuls entre le 1er janvier de l année N-1 et le 31
décembre de l annee N;

proc sql;

	CREATE TABLE CSI AS
		SELECT DISTINCT
			a.BEN_IDT_ANO,
			COUNT(DISTINCT b.date_debut) AS Nb_deliv length = 3
		FROM T_INDI_BPCO_VACGA_&an_N. a
			INNER JOIN travail.reperage_med_asthme_&Annee_2N._&Annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.reperage = 41 AND ("01JAN&Annee_1N."d <= b.date_debut <= "31DEC&Annee_N."d)
		GROUP BY a.BEN_IDT_ANO;

quit;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_VACG_&an_N.
		SELECT
			7,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM T_INDI_BPCO_VACGA_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM CSI WHERE Nb_deliv >= 3);
```

```
	* On conserve les id des patients à supprimer;
	CREATE TABLE travail.exclus3 AS
		SELECT DISTINCT
			BEN_IDT_ANO
	FROM T_INDI_BPCO_VACGA_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM CSI WHERE Nb_deliv >= 3);

quit;

proc delete data = CSI;
run; quit;

*
	**********************************************************************************************************
**********************************;
*	Exclusion des patients avec au moins 1 délivrance remboursée d'anti IgE entre le 1er janvier de l annee N-1 et le 31
décembre de l annee N;

proc sql;

	CREATE TABLE anti_IGE AS
		SELECT DISTINCT
			a.BEN_IDT_ANO,
			COUNT(DISTINCT b.date_debut) AS Nb_deliv length = 3
		FROM T_INDI_BPCO_VACGA_&an_N. a
			INNER JOIN travail.reperage_med_asthme_&Annee_2N._&Annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.reperage IN (37) AND ("01JAN&Annee_1N."d <= b.date_debut <= "31DEC&Annee_N."d)
		GROUP BY a.BEN_IDT_ANO;

quit;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_VACG_&an_N.
		SELECT
			8,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM T_INDI_BPCO_VACGA_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM anti_IGE WHERE Nb_deliv >= 1);

	* On conserve les id des patients à supprimer;
	CREATE TABLE travail.exclus4 AS
		SELECT DISTINCT
			BEN_IDT_ANO
	FROM T_INDI_BPCO_VACGA_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM anti_IGE WHERE Nb_deliv >= 1);

quit;

proc delete data = anti_IGE;
run; quit;

*
	**********************************************************************************************************
**********************************;
*	Exclusion des patients avec au moins 1 délivrance remboursée d'anti IL-5 entre le 1er janvier de l annee N-1 et le 31
décembre de l annee N;

proc sql;

	CREATE TABLE anti_IL5 AS
		SELECT DISTINCT
			a.BEN_IDT_ANO,
			COUNT(DISTINCT b.date_debut) AS Nb_deliv length = 3
		FROM T_INDI_BPCO_VACGA_&an_N. a
			INNER JOIN travail.reperage_med_asthme_&Annee_2N._&Annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.reperage IN (38) AND ("01JAN&Annee_1N."d <= b.date_debut <= "31DEC&Annee_N."d)
		GROUP BY a.BEN_IDT_ANO;
```

```
quit;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_VACG_&an_N.
		SELECT
			9,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM T_INDI_BPCO_VACGA_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM anti_IL5 WHERE Nb_deliv >= 1);

	* On conserve les id des patients à supprimer;
	CREATE TABLE travail.exclus5 AS
		SELECT DISTINCT
			BEN_IDT_ANO
	FROM T_INDI_BPCO_VACGA_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM anti_IL5 WHERE Nb_deliv >= 1);

quit;

proc delete data = anti_IL5;
run; quit;

*
	**********************************************************************************************************
***********************************;
*	Exclusion des patients avec au moins 1 délivrance remboursée d'antileucotriène entre le 1er janvier de l annee N-1 et le 31
décembre de l annee N;

proc sql;

	CREATE TABLE antileucotriene AS
		SELECT DISTINCT
			a.BEN_IDT_ANO,
			COUNT(DISTINCT b.date_debut) AS Nb_deliv length = 3
		FROM T_INDI_BPCO_VACGA_&an_N. a
			INNER JOIN travail.reperage_med_asthme_&Annee_2N._&Annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.reperage = 39 AND ("01JAN&Annee_1N."d <= b.date_debut <= "31DEC&Annee_N."d)
		GROUP BY a.BEN_IDT_ANO;

quit;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_VACG_&an_N.
		SELECT
			10,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM T_INDI_BPCO_VACGA_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM antileucotriene WHERE Nb_deliv >= 1);

	* On conserve les id des patients à supprimer;
	CREATE TABLE travail.exclus6 AS
		SELECT DISTINCT
			BEN_IDT_ANO
	FROM T_INDI_BPCO_VACGA_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM antileucotriene WHERE Nb_deliv >= 1);

quit;

proc delete data = antileucotriene;
run; quit;

*
	**********************************************************************************************************
***********************************;
```

```
*		Exclusion des patients avec au moins un acte CCAM de Thermoplastie bronchique codé lors d'un sé-jour hospitalier en MCO
terminé entre
		le 1er janvier de l annee N-1 et le 31 décembre de l annee N;

proc sql;

		CREATE TABLE thermo_bronchique AS
			SELECT DISTINCT
				a.BEN_IDT_ANO,
				COUNT(DISTINCT b.date_debut) AS Nb_actes length = 3
			FROM T_INDI_BPCO_VACGA_&an_N. a
				INNER JOIN travail.reperage_thermo_bronch_&annee_1N._&annee_N1. b
					ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
			WHERE "01JAN&Annee_1N."d <= b.date_fin <= "31DEC&Annee_N."d
			GROUP BY a.BEN_IDT_ANO;

quit;

proc sql;

		* On insère l effectif dans le Flowchart;
		INSERT INTO flowch.Flow_Chart_BPCO_VACG_&an_N.
			SELECT
				11,
				COUNT(DISTINCT BEN_IDT_ANO)
		FROM T_INDI_BPCO_VACGA_&an_N.
		WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM thermo_bronchique WHERE Nb_actes >= 1);

		* On conserve les id des patients à supprimer;
		CREATE TABLE travail.exclus7 AS
			SELECT DISTINCT
				BEN_IDT_ANO
		FROM T_INDI_BPCO_VACGA_&an_N.
		WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM thermo_bronchique WHERE Nb_actes >= 1);

quit;

proc delete data = thermo_bronchique;
run; quit;

*
		**********************************************************************************************************
**********************************;
*		Exclusion des patients ayant eu au moins un diagnostic d'asthme codé entre le 1er janvier de l annee N-1 et le 31 décembre
de l annee N;


proc sql undo_policy = none;

		CREATE TABLE diag_asthme_MCO AS
			SELECT DISTINCT
				a.BEN_IDT_ANO
			FROM T_INDI_BPCO_VACGA_&an_N. a
				INNER JOIN travail.diag_asthme_&annee_1N._&annee_N1. b
					ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
			WHERE reperage = 14 AND "01JAN&Annee_1N."d <= b.date_fin <= "31DEC&Annee_N."d AND b.domaine =
"MCO";

		CREATE TABLE diag_asthme_SSR_HAD AS
			SELECT DISTINCT
				a.BEN_IDT_ANO
			FROM T_INDI_BPCO_VACGA_&an_N. a
				INNER JOIN travail.diag_asthme_&annee_1N._&annee_N1. b
					ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
			WHERE reperage = 14 AND b.domaine IN ("HAD", "SSR") AND b.date_debut <= "31DEC&Annee_N."d AND
b.date_fin >= "01JAN&Annee_1N."d;

quit;

data diag_asthme;
		set diag_asthme_MCO
```

```
                    diag_asthme_SSR_HAD;
run;

proc sql;

        * On insère l effectif dans le Flowchart;
        INSERT INTO flowch.Flow_Chart_BPCO_VACG_&an_N.
                SELECT
                        12,
                        COUNT(DISTINCT BEN_IDT_ANO)
        FROM T_INDI_BPCO_VACGA_&an_N.
        WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM diag_asthme);

        * On conserve les id des patients à supprimer;
        CREATE TABLE travail.exclus8 AS
                SELECT DISTINCT
                        BEN_IDT_ANO
        FROM T_INDI_BPCO_VACGA_&an_N.
        WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM diag_asthme);

quit;

proc delete data = diag_asthme diag_asthme_MCO diag_asthme_SSR_HAD;
run; quit;

*
        ************************************************************************************************************
************************************;
*       Exclusion des patients ayant eu au moins un diagnostic d'état de mal asthmatique  codé entre le 1er janvier de l annee N-1 et
le 31 décembre de l annee N;

proc sql;

        CREATE TABLE etat_mal_asthmatique AS
                SELECT DISTINCT
                        a.BEN_IDT_ANO
                FROM T_INDI_BPCO_VACGA_&an_N. a
                        INNER JOIN travail.diag_asthme_&annee_1N._&annee_N1. b
                                ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
                WHERE reperage = 15 AND "01JAN&Annee_1N."d <= b.date_fin <= "31DEC&Annee_N."d;

quit;

proc sql;

        * On insère l effectif dans le Flowchart;
        INSERT INTO flowch.Flow_Chart_BPCO_VACG_&an_N.
                SELECT
                        13,
                        COUNT(DISTINCT BEN_IDT_ANO)
        FROM T_INDI_BPCO_VACGA_&an_N.
        WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM etat_mal_asthmatique);

        * On conserve les id des patients à supprimer;
        CREATE TABLE travail.exclus9 AS
                SELECT DISTINCT
                        BEN_IDT_ANO
        FROM T_INDI_BPCO_VACGA_&an_N.
        WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM etat_mal_asthmatique);

quit;

proc delete data = etat_mal_asthmatique;
run; quit;

*
        ************************************************************************************************************
************************************;
*       On exclut tous les patients;

data exclus;
```

```
	set	travail.exclus1-travail.exclus9;
run;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_VACG_&an_N.
		SELECT
			14,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM exclus;

	DELETE FROM T_INDI_BPCO_VACGA_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM exclus);

quit;

proc delete data = exclus;
run; quit;

proc datasets library = travail memtype = data nolist;
	delete exclus1-exclus9;
run; quit;

*
	**********************************************************************************************************
**********************************;
*	Exclusion des patients diagnostiqués BPCO;

* Patients avec une ALD BPCO active au 1er septembre N;
proc sql;

	CREATE TABLE travail.exclus1 AS
		SELECT DISTINCT
			BEN_IDT_ANO
		FROM T_INDI_BPCO_VACGA_&an_N.
		WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM res.T_INDI_BPCO_&an_N. WHERE ALD_BPCO_septembre
= 1);

quit;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_VACG_&an_N.
		SELECT
			15,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM travail.exclus1;

quit;

* Patients avec au moins une hospit BPCO;
proc sql;

	CREATE TABLE travail.exclus2 AS
		SELECT DISTINCT
			BEN_IDT_ANO
		FROM T_INDI_BPCO_VACGA_&an_N.
		WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM res.T_INDI_BPCO_&an_N. WHERE
Nb_hospit_BPCO_Ind02 > 0);

quit;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_VACG_&an_N.
		SELECT
			16,
			COUNT(DISTINCT BEN_IDT_ANO)
```

```
	FROM travail.exclus2;

quit;

*
	*************************************************************************************************************
************************************;
*	On exclut tous les patients;

data exclus;
	set	travail.exclus1-travail.exclus2;
run;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_VACG_&an_N.
		SELECT
			17,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM exclus;

	DELETE FROM T_INDI_BPCO_VACGA_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM exclus);

quit;

proc datasets library = travail memtype = data nolist;
	delete exclus1-exclus2;
run; quit;

proc delete data = exclus;
run; quit;

*
	*************************************************************************************************************
************************************;
*	Nombre de patients dans la population d'étude : BPCO probable;

proc sql;

	INSERT INTO flowch.Flow_Chart_BPCO_VACG_&an_N.
		SELECT
			18,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM T_INDI_BPCO_VACGA_&an_N.;

quit;

proc sql;

	SELECT COUNT(*), COUNT(DISTINCT BEN_IDT_ANO)
	FROM T_INDI_BPCO_VACGA_&an_N.;

quit;
```

3.7.3 02_Populations_etude_cible.sas

```
/*
	*****************************************************************************************************
**********************************	*/
/*
                                                                                                        */
/*		Indicateur 02 : Exclusions pour les patients BPCO probable

			*/
/*
                                                                                                        */
/*
	*****************************************************************************************************
**********************************	*/

*
	*****************************************************************************************************
**********************************	;
*	EXCLUSION DES PATIENTS ASTHMATIQUES

                                                                        ;
*
	*****************************************************************************************************
**********************************	;

*
	*****************************************************************************************************
**********************************;
*	Exclusion des patients en ALD asthme active au 1er septembre de l année N;

proc sql;

	CREATE TABLE ALD_asthme AS
		SELECT DISTINCT
			a.BEN_IDT_ANO
		FROM T_INDI_BPCO_VACGA_&an_N. a
			INNER JOIN travail.Histo_ALD b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.reperage = 53 AND (b.date_debut <= "01SEP&Annee_N."d <= b.date_fin OR (b.date_debut <=
"01SEP&Annee_N."d AND
			b.date_fin = "01JAN1600"d));

quit;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_VACG_&an_N.
		SELECT
			5,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM T_INDI_BPCO_VACGA_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM ALD_asthme);

	* On conserve les id des patients à supprimer;
	CREATE TABLE travail.exclus1 AS
		SELECT DISTINCT
			BEN_IDT_ANO
	FROM T_INDI_BPCO_VACGA_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM ALD_asthme);

quit;

proc delete data = ALD_asthme;
run; quit;
```

```
*
	*****************************************************************************************************
***********************************;
*	Exclusion des patients en ALD mucoviscidose active au 1er septembre de l année N;

proc sql;

	CREATE TABLE ALD_mucoviscidose AS
		SELECT DISTINCT
			a.BEN_IDT_ANO
		FROM T_INDI_BPCO_VACGA_&an_N. a
			INNER JOIN travail.Histo_ALD b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.reperage = 54 AND (b.date_debut <= "01SEP&Annee_N."d <= b.date_fin OR (b.date_debut <=
"01SEP&Annee_N."d AND
			b.date_fin = "01JAN1600"d));

quit;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_VACG_&an_N.
		SELECT
			6,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM T_INDI_BPCO_VACGA_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM ALD_mucoviscidose);

	* On conserve les id des patients à supprimer;
	CREATE TABLE travail.exclus2 AS
		SELECT DISTINCT
			BEN_IDT_ANO
	FROM T_INDI_BPCO_VACGA_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM ALD_mucoviscidose);

quit;

proc delete data = ALD_mucoviscidose;
run; quit;

*
	*****************************************************************************************************
***********************************;
*	Exclusion des patients avec au moins 3 délivrances remboursées de CSI seuls entre le 1er janvier de l année N-1 et le 31
décembre de l annee N;

proc sql;

	CREATE TABLE CSI AS
		SELECT DISTINCT
			a.BEN_IDT_ANO,
			COUNT(DISTINCT b.date_debut) AS Nb_deliv length = 3
		FROM T_INDI_BPCO_VACGA_&an_N. a
			INNER JOIN travail.reperage_med_asthme_&Annee_2N._&Annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.reperage = 41 AND ("01JAN&Annee_1N."d <= b.date_debut <= "31DEC&Annee_N."d)
		GROUP BY a.BEN_IDT_ANO;

quit;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_VACG_&an_N.
		SELECT
			7,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM T_INDI_BPCO_VACGA_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM CSI WHERE Nb_deliv >= 3);
```

```
	* On conserve les id des patients à supprimer;
	CREATE TABLE travail.exclus3 AS
		SELECT DISTINCT
			BEN_IDT_ANO
	FROM T_INDI_BPCO_VACGA_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM CSI WHERE Nb_deliv >= 3);

quit;

proc delete data = CSI;
run; quit;

*
	**********************************************************************************************************
**********************************;
*	Exclusion des patients avec au moins 1 délivrance remboursée d'anti IgE entre le 1er janvier de l annee N-1 et le 31
décembre de l annee N;

proc sql;

	CREATE TABLE anti_IGE AS
		SELECT DISTINCT
			a.BEN_IDT_ANO,
			COUNT(DISTINCT b.date_debut) AS Nb_deliv length = 3
		FROM T_INDI_BPCO_VACGA_&an_N. a
			INNER JOIN travail.reperage_med_asthme_&Annee_2N._&Annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.reperage IN (37) AND ("01JAN&Annee_1N."d <= b.date_debut <= "31DEC&Annee_N."d)
		GROUP BY a.BEN_IDT_ANO;

quit;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_VACG_&an_N.
		SELECT
			8,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM T_INDI_BPCO_VACGA_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM anti_IGE WHERE Nb_deliv >= 1);

	* On conserve les id des patients à supprimer;
	CREATE TABLE travail.exclus4 AS
		SELECT DISTINCT
			BEN_IDT_ANO
	FROM T_INDI_BPCO_VACGA_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM anti_IGE WHERE Nb_deliv >= 1);

quit;

proc delete data = anti_IGE;
run; quit;

*
	**********************************************************************************************************
**********************************;
*	Exclusion des patients avec au moins 1 délivrance remboursée d'anti IL-5 entre le 1er janvier de l annee N-1 et le 31
décembre de l annee N;

proc sql;

	CREATE TABLE anti_IL5 AS
		SELECT DISTINCT
			a.BEN_IDT_ANO,
			COUNT(DISTINCT b.date_debut) AS Nb_deliv length = 3
		FROM T_INDI_BPCO_VACGA_&an_N. a
			INNER JOIN travail.reperage_med_asthme_&Annee_2N._&Annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.reperage IN (38) AND ("01JAN&Annee_1N."d <= b.date_debut <= "31DEC&Annee_N."d)
		GROUP BY a.BEN_IDT_ANO;
```

```
quit;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_VACG_&an_N.
		SELECT
			9,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM T_INDI_BPCO_VACGA_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM anti_IL5 WHERE Nb_deliv >= 1);

	* On conserve les id des patients à supprimer;
	CREATE TABLE travail.exclus5 AS
		SELECT DISTINCT
			BEN_IDT_ANO
	FROM T_INDI_BPCO_VACGA_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM anti_IL5 WHERE Nb_deliv >= 1);

quit;

proc delete data = anti_IL5;
run; quit;

*
	************************************************************************************************************
***********************************;
*	Exclusion des patients avec au moins 1 délivrance remboursée d'antileucotriène entre le 1er janvier de l annee N-1 et le 31
décembre de l annee N;

proc sql;

	CREATE TABLE antileucotriene AS
		SELECT DISTINCT
			a.BEN_IDT_ANO,
			COUNT(DISTINCT b.date_debut) AS Nb_deliv length = 3
		FROM T_INDI_BPCO_VACGA_&an_N. a
			INNER JOIN travail.reperage_med_asthme_&Annee_2N._&Annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.reperage = 39 AND ("01JAN&Annee_1N."d <= b.date_debut <= "31DEC&Annee_N."d)
		GROUP BY a.BEN_IDT_ANO;

quit;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_VACG_&an_N.
		SELECT
			10,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM T_INDI_BPCO_VACGA_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM antileucotriene WHERE Nb_deliv >= 1);

	* On conserve les id des patients à supprimer;
	CREATE TABLE travail.exclus6 AS
		SELECT DISTINCT
			BEN_IDT_ANO
	FROM T_INDI_BPCO_VACGA_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM antileucotriene WHERE Nb_deliv >= 1);

quit;

proc delete data = antileucotriene;
run; quit;

*
	************************************************************************************************************
***********************************;
```

```
*		Exclusion des patients avec au moins un acte CCAM de Thermoplastie bronchique codé lors d'un sé-jour hospitalier en MCO
terminé entre
		le 1er janvier de l annee N-1 et le 31 décembre de l annee N;

proc sql;

		CREATE TABLE thermo_bronchique AS
			SELECT DISTINCT
				a.BEN_IDT_ANO,
				COUNT(DISTINCT b.date_debut) AS Nb_actes length = 3
			FROM T_INDI_BPCO_VACGA_&an_N. a
				INNER JOIN travail.reperage_thermo_bronch_&annee_1N._&annee_N1. b
					ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
			WHERE "01JAN&Annee_1N."d <= b.date_fin <= "31DEC&Annee_N."d
			GROUP BY a.BEN_IDT_ANO;

quit;

proc sql;

		* On insère l effectif dans le Flowchart;
		INSERT INTO flowch.Flow_Chart_BPCO_VACG_&an_N.
			SELECT
				11,
				COUNT(DISTINCT BEN_IDT_ANO)
		FROM T_INDI_BPCO_VACGA_&an_N.
		WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM thermo_bronchique WHERE Nb_actes >= 1);

		* On conserve les id des patients à supprimer;
		CREATE TABLE travail.exclus7 AS
			SELECT DISTINCT
				BEN_IDT_ANO
		FROM T_INDI_BPCO_VACGA_&an_N.
		WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM thermo_bronchique WHERE Nb_actes >= 1);

quit;

proc delete data = thermo_bronchique;
run; quit;

*
		**********************************************************************************************************
**********************************;
*		Exclusion des patients ayant eu au moins un diagnostic d'asthme codé entre le 1er janvier de l annee N-1 et le 31 décembre
de l annee N;


proc sql undo_policy = none;

		CREATE TABLE diag_asthme_MCO AS
			SELECT DISTINCT
				a.BEN_IDT_ANO
			FROM T_INDI_BPCO_VACGA_&an_N. a
				INNER JOIN travail.diag_asthme_&annee_1N._&annee_N1. b
					ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
			WHERE reperage = 14 AND "01JAN&Annee_1N."d <= b.date_fin <= "31DEC&Annee_N."d AND b.domaine =
"MCO";

		CREATE TABLE diag_asthme_SSR_HAD AS
			SELECT DISTINCT
				a.BEN_IDT_ANO
			FROM T_INDI_BPCO_VACGA_&an_N. a
				INNER JOIN travail.diag_asthme_&annee_1N._&annee_N1. b
					ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
			WHERE reperage = 14 AND b.domaine IN ("HAD", "SSR") AND b.date_debut <= "31DEC&Annee_N."d AND
b.date_fin >= "01JAN&Annee_1N."d;

quit;

data diag_asthme;
		set diag_asthme_MCO
```

```
                        diag_asthme_SSR_HAD;
run;

proc sql;

        * On insère l effectif dans le Flowchart;
        INSERT INTO flowch.Flow_Chart_BPCO_VACG_&an_N.
                SELECT
                        12,
                        COUNT(DISTINCT BEN_IDT_ANO)
        FROM T_INDI_BPCO_VACGA_&an_N.
        WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM diag_asthme);

        * On conserve les id des patients à supprimer;
        CREATE TABLE travail.exclus8 AS
                SELECT DISTINCT
                        BEN_IDT_ANO
        FROM T_INDI_BPCO_VACGA_&an_N.
        WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM diag_asthme);

quit;

proc delete data = diag_asthme diag_asthme_MCO diag_asthme_SSR_HAD;
run; quit;

*
        ************************************************************************************************************
************************************;
*       Exclusion des patients ayant eu au moins un diagnostic d'état de mal asthmatique  codé entre le 1er janvier de l annee N-1 et
le 31 décembre de l annee N;

proc sql;

        CREATE TABLE etat_mal_asthmatique AS
                SELECT DISTINCT
                        a.BEN_IDT_ANO
                FROM T_INDI_BPCO_VACGA_&an_N. a
                        INNER JOIN travail.diag_asthme_&annee_1N._&annee_N1. b
                                ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
                WHERE reperage = 15 AND "01JAN&Annee_1N."d <= b.date_fin <= "31DEC&Annee_N."d;

quit;

proc sql;

        * On insère l effectif dans le Flowchart;
        INSERT INTO flowch.Flow_Chart_BPCO_VACG_&an_N.
                SELECT
                        13,
                        COUNT(DISTINCT BEN_IDT_ANO)
        FROM T_INDI_BPCO_VACGA_&an_N.
        WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM etat_mal_asthmatique);

        * On conserve les id des patients à supprimer;
        CREATE TABLE travail.exclus9 AS
                SELECT DISTINCT
                        BEN_IDT_ANO
        FROM T_INDI_BPCO_VACGA_&an_N.
        WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM etat_mal_asthmatique);

quit;

proc delete data = etat_mal_asthmatique;
run; quit;

*
        ************************************************************************************************************
************************************;
*       On exclut tous les patients;

data exclus;
```

```
	set	travail.exclus1-travail.exclus9;
run;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_VACG_&an_N.
		SELECT
			14,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM exclus;

	DELETE FROM T_INDI_BPCO_VACGA_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM exclus);

quit;

proc delete data = exclus;
run; quit;

proc datasets library = travail memtype = data nolist;
	delete exclus1-exclus9;
run; quit;

*
	**********************************************************************************************************
**********************************;
*	Exclusion des patients diagnostiqués BPCO;

* Patients avec une ALD BPCO active au 1er septembre N;
proc sql;

	CREATE TABLE travail.exclus1 AS
		SELECT DISTINCT
			BEN_IDT_ANO
		FROM T_INDI_BPCO_VACGA_&an_N.
		WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM res.T_INDI_BPCO_&an_N. WHERE ALD_BPCO_septembre
= 1);

quit;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_VACG_&an_N.
		SELECT
			15,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM travail.exclus1;

quit;

* Patients avec au moins une hospit BPCO;
proc sql;

	CREATE TABLE travail.exclus2 AS
		SELECT DISTINCT
			BEN_IDT_ANO
		FROM T_INDI_BPCO_VACGA_&an_N.
		WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM res.T_INDI_BPCO_&an_N. WHERE
Nb_hospit_BPCO_Ind02 > 0);

quit;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_VACG_&an_N.
		SELECT
			16,
			COUNT(DISTINCT BEN_IDT_ANO)
```

```
	FROM travail.exclus2;

quit;

*
	*************************************************************************************************************
************************************;
*	On exclut tous les patients;

data exclus;
	set	travail.exclus1-travail.exclus2;
run;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_VACG_&an_N.
		SELECT
			17,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM exclus;

	DELETE FROM T_INDI_BPCO_VACGA_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM exclus);

quit;

proc datasets library = travail memtype = data nolist;
	delete exclus1-exclus2;
run; quit;

proc delete data = exclus;
run; quit;

*
	*************************************************************************************************************
************************************;
*	Nombre de patients dans la population d'étude : BPCO probable;

proc sql;

	INSERT INTO flowch.Flow_Chart_BPCO_VACG_&an_N.
		SELECT
			18,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM T_INDI_BPCO_VACGA_&an_N.;

quit;

proc sql;

	SELECT COUNT(*), COUNT(DISTINCT BEN_IDT_ANO)
	FROM T_INDI_BPCO_VACGA_&an_N.;

quit;
```

3.7.4 03_Initialisation_de_la_table_de_resultats.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*

*/
/*		Indicateur 02 : Initialisation de la table de résultats contenant les BPCO probables et les patients diagnostiqués
BPCO	*/
/*

*/
/*
	*****************************************************************************************************************
**********************************	*/

proc sort data = T_INDI_BPCO_VACG_&an_N. out = res.T_INDI_BPCO_VACG_&an_N nodupkey;
	by BEN_IDT_ANO;
run;

*
	*****************************************************************************************************************
**********************************;
*	On flag les patients ayant eu au moins un diagnostic de BPCO codé lors d'un séjour hospitalier en MCO terminé entre le 1er
janvier de
	l année N et le 31 aout de l année N+1 ou lors d'un séjour hospitalier en SSR, HAD entre le 1er janvier de l année N et le 31
décembre
	de l année N+1;

proc sql;

	CREATE TABLE diag_BPCO AS
		SELECT DISTINCT
			BEN_IDT_ANO,
			1 AS Diag_BPCO length = 3
		FROM res.T_INDI_BPCO_VACG_&an_N. a
		WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM res.T_INDI_BPCO_&an_N. WHERE
Nb_hospit_BPCO_Ind02 > 0);

quit;

data res.T_INDI_BPCO_VACG_&an_N.;
	merge	res.T_INDI_BPCO_VACG_&an_N. (in = a)
		diag_BPCO (in = b);
	by BEN_IDT_ANO;
	if a;
	if not b then
		Diag_BPCO = 0;
run;

*
	*****************************************************************************************************************
**********************************;
*	On flag les patients en ALD BPCO active le 1er septembre de l année N;

proc sql;

	CREATE TABLE ALD_BCPO AS
		SELECT
			a.BEN_IDT_ANO,
			1 AS ALD_BPCO_avSep length = 3,
			MIN(date_debut) AS ALD_BPCO_Date format ddmmyy10. length = 4
		FROM res.T_INDI_BPCO_VACG_&an_N. a
			INNER JOIN travail.Histo_ALD b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.reperage = 52 AND a.BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM res.T_INDI_BPCO_&an_N. WHERE
ALD_BPCO_septembre = 1)
		GROUP BY a.BEN_IDT_ANO
```

```
                        ORDER BY a.BEN_IDT_ANO;

quit;

data res.T_INDI_BPCO_VACG_&an_N.;
        merge   res.T_INDI_BPCO_VACG_&an_N. (in = a)
                        ALD_BCPO (in = b);
        by BEN_IDT_ANO;
        if a;
        if ALD_BPCO_avSep = . then
                ALD_BPCO_avSep = 0;
        if ALD_BPCO_avSep = 0 then
                ALD_BPCO_Date = .;
run;

proc delete data = ALD_BCPO;
run; quit;
```

3.7.5 04_Sejours_avec_un_diag_de_BPCO.sas

```
/*
	*****************************************************************************************************
**********************************	*/
/*
                                                                                                        */
/*		Indicateur 02 : Nombre de séjours BPCO
                                                        */
/*
                                                                                                        */
/*
	*****************************************************************************************************
**********************************	*/

proc sql;

	CREATE INDEX BEN_IDT_ANO ON res.T_INDI_BPCO_VACG_&an_N. (BEN_IDT_ANO);

quit;

%macro sejours_BPCO(domaine=);

	proc sql;

		CREATE TABLE sejours_BPCO_&domaine. AS
			SELECT DISTINCT
				a.BEN_IDT_ANO,
				COUNT(DISTINCT id_sejour) AS Nb_Sej_Diag_BPCO_&domaine. length = 3
			FROM res.T_INDI_BPCO_VACG_&an_N. a
				INNER JOIN travail.reperage_hospit_BPCO_&annee_4N._&annee_N1. b
					ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
			WHERE b.domaine = "&domaine." AND
				%if &domaine. = MCO %then
					%do;
						"01JAN&Annee_1N."d <= b.date_fin <= "31AUG&Annee_N."d
					%end;
				%else
					%do;
						(b.date_debut <= "31AUG&Annee_N."d AND b.date_fin >=
"01JAN&Annee_1N."d)
					%end;
			GROUP BY a.BEN_IDT_ANO
			;

		CREATE INDEX BEN_IDT_ANO ON sejours_BPCO_&domaine. (BEN_IDT_ANO);

		ALTER TABLE res.T_INDI_BPCO_VACG_&an_N. ADD Nb_Sej_Diag_BPCO_&domaine. INT length = 3;

		UPDATE res.T_INDI_BPCO_VACG_&an_N. a
			SET Nb_Sej_Diag_BPCO_&domaine. =	(SELECT Nb_Sej_Diag_BPCO_&domaine.

	FROM sejours_BPCO_&domaine. b

	WHERE a.BEN_IDT_ANO = b.BEN_IDT_ANO);

	quit;

	data res.T_INDI_BPCO_VACG_&an_N.;
		set res.T_INDI_BPCO_VACG_&an_N.;
		if Nb_Sej_Diag_BPCO_&domaine. = . then
			Nb_Sej_Diag_BPCO_&domaine. = 0;
	run;

	proc delete data = sejours_BPCO_&domaine.;
	run; quit;
```

```
%mend sejours_BPCO;

%sejours_BPCO(domaine = MCO);
%sejours_BPCO(domaine = SSR);
%sejours_BPCO(domaine = HAD);
```

3.7.6 05_Vaccin_contre_la_grippe.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*		Indicateur 02 : Nombre de séjours BPCO
                                                                */
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/

proc sql;

	CREATE INDEX BEN_IDT_ANO ON res.T_INDI_BPCO_VACG_&an_N. (BEN_IDT_ANO);

quit;

%macro sejours_BPCO(domaine=);

	proc sql;

		CREATE TABLE sejours_BPCO_&domaine. AS
			SELECT DISTINCT
				a.BEN_IDT_ANO,
				COUNT(DISTINCT id_sejour) AS Nb_Sej_Diag_BPCO_&domaine. length = 3
			FROM res.T_INDI_BPCO_VACG_&an_N. a
				INNER JOIN travail.reperage_hospit_BPCO_&annee_4N._&annee_N1. b
					ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
			WHERE b.domaine = "&domaine." AND
				%if &domaine. = MCO %then
					%do;
						"01JAN&Annee_1N."d <= b.date_fin <= "31AUG&Annee_N."d
					%end;
				%else
					%do;
						(b.date_debut <= "31AUG&Annee_N."d AND b.date_fin >=
"01JAN&Annee_1N."d)
					%end;
			GROUP BY a.BEN_IDT_ANO
			;

		CREATE INDEX BEN_IDT_ANO ON sejours_BPCO_&domaine. (BEN_IDT_ANO);

		ALTER TABLE res.T_INDI_BPCO_VACG_&an_N. ADD Nb_Sej_Diag_BPCO_&domaine. INT length = 3;

		UPDATE res.T_INDI_BPCO_VACG_&an_N. a
			SET Nb_Sej_Diag_BPCO_&domaine. =	(SELECT Nb_Sej_Diag_BPCO_&domaine.

	FROM sejours_BPCO_&domaine. b

	WHERE a.BEN_IDT_ANO = b.BEN_IDT_ANO);

	quit;

	data res.T_INDI_BPCO_VACG_&an_N.;
		set res.T_INDI_BPCO_VACG_&an_N.;
		if Nb_Sej_Diag_BPCO_&domaine. = . then
			Nb_Sej_Diag_BPCO_&domaine. = 0;
	run;

	proc delete data = sejours_BPCO_&domaine.;
	run; quit;
```

```
%mend sejours_BPCO;

%sejours_BPCO(domaine = MCO);
%sejours_BPCO(domaine = SSR);
%sejours_BPCO(domaine = HAD);
```

3.7.7 06_Table_finale.sas

```
/*
	*************************************************************************************************************
**********************************	*/
/*
                                                                                                                */
/*		Indicateur 02 : Table finale - Ajout de formats et de labels

		*/
/*
                                                                                                                */
/*
	*************************************************************************************************************
**********************************	*/

data res.T_INDI_BPCO_VACG_&an_N.;
	set res.T_INDI_BPCO_VACG_&an_N. (keep = BEN_IDT_ANO BPCO_VACG_PROB_CIBLE Diag_BPCO ALD_BPCO_avSep
ALD_BPCO_Date BPCO_VACG_DIAG_CIBLE
		BPCO_VACG_CIBLE Nb_Sej_Diag_BPCO_MCO Nb_Sej_Diag_BPCO_SSR Nb_Sej_Diag_BPCO_HAD
BPCO_VACG_PROB_OBS BPCO_VACG_DIAG_OBS BPCO_VACG_OBS
		Date_VACG);
	label
		BEN_IDT_ANO = "Numéro d'individu"
		BPCO_VACG_PROB_CIBLE = "Patient correspondant aux critères d'inclusion et d'exclusion de la population cible
probable - BPCO probable"
		Diag_BPCO = "Patient ayant eu au moins un diagnostic de BPCO codé lors d'un séjour hospitalier en MCO terminé
entre le 1er janvier &annee_1N. et le 31 août &Annee_N. ou lors d'un séjour hospitalier en SSR, HAD entre le 1er janvier &annee_1N. et
le 31 août &Annee_N."
		ALD_BPCO_avSep = "Patient en ALD BPCO active le 1er septembre &Annee_N."
		ALD_BPCO_Date = "Date de début de mise en ALD BPCO"
		BPCO_VACG_DIAG_CIBLE = "Patient correspondant aux critères d'inclusion et d'exclusion de la population cible
diagnostiquée - BPCO diagnostiqués"
		BPCO_VACG_CIBLE = "Patient correspondant aux critères d'inclusion et d'exclusion de la population cible"
		Nb_Sej_Diag_BPCO_MCO = "Nombre de séjours avec un diagnostic BPCO terminés entre le 1er janvier
&annee_1N. et le 31 août &Annee_N. en MCO"
		Nb_Sej_Diag_BPCO_SSR = "Nombre de séjours avec un diagnostic BPCO entre le 1er janvier &annee_1N. et le 31
août &Annee_N. en SSR"
		Nb_Sej_Diag_BPCO_HAD = "Nombre de séjours avec un diagnostic BPCO entre le 1er janvier &annee_1N. et le 31
août &Annee_N. en HAD"
		BPCO_VACG_PROB_OBS = "Patient de la population cible probable ayant eu une délivrance remboursée du vaccin
contre la grippe entre le 1er septembre &Annee_N. et le 31 mai &Annee_N1."
		BPCO_VACG_DIAG_OBS = "Patient de la population cible diagnostiquée ayant eu une délivrance remboursée du
vaccin contre la grippe entre le 1er septembre &Annee_N. et le 31 mai &Annee_N1."
		BPCO_VACG_OBS = "Patient de la population cible ayant eu une délivrance remboursée du vaccin contre la grippe
entre le 1er septembre &Annee_N. et le 31 mai &Annee_N1."
		Date_VACG = "Date de la délivrance remboursée du vaccin contre la grippe entre le 1er septembre &Annee_N. et
le 31 mai &Annee_N1."
		;
	format BPCO_VACG_PROB_CIBLE Diag_BPCO ALD_BPCO_avSep BPCO_VACG_DIAG_CIBLE BPCO_VACG_PROB_OBS
BPCO_VACG_DIAG_OBS BPCO_VACG_CIBLE BPCO_VACG_OBS
		f_oui_non. ALD_BPCO_Date Date_VACG date9.;
run;

*
	*************************************************************************************************************
**********************************;
*	Vaccination contre la grippe;

proc sql;

	INSERT INTO flowch.Flow_Chart_BPCO_VACG_&an_N.
		SELECT
			26,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM res.T_INDI_BPCO_VACG_&an_N.
	WHERE BPCO_VACG_OBS = 1;
```

```
	* Nombre de patients pour les numérateur des premières exclusions;
	SELECT Nb_patients_ref0 into : N0 FROM travail.ref_flowcharts;
	SELECT N into : N1 FROM flowch.Flow_Chart_BPCO_VACG_&an_N. WHERE indicateur = 1;
	SELECT N into : N4 FROM flowch.Flow_Chart_BPCO_VACG_&an_N. WHERE indicateur = 4;

	* Population d étude;
	SELECT N into : N19 FROM flowch.Flow_Chart_BPCO_VACG_&an_N. WHERE indicateur = 19;

	* Population cible;
	SELECT N into : N25 FROM flowch.Flow_Chart_BPCO_VACG_&an_N. WHERE indicateur = 25;

quit;

data flowch.Flow_Chart_BPCO_VACG_&an_N.;
	set flowch.Flow_Chart_BPCO_VACG_&an_N.;
	if indicateur in (1, 2, 3, 4, 18, 19) then
		percent = N / &N0.;
	if indicateur in (5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17) then
		percent = N / &N4.;
	if indicateur in (20, 21, 22, 23, 25) then
		percent = N / &N19.;
	if indicateur = 26 then
		percent = N / &N25.;
	format indicateur f_ind_02_flowchart. percent percent7.1;
run;

* Vérif;
proc sql;

	SELECT COUNT(*) AS nb_lignes, COUNT(DISTINCT BEN_IDT_ANO) AS nb_patients
	FROM res.T_INDI_BPCO_VACG_&an_N.;
	* nb_lignes = nb_patients : OK;

	SELECT COUNT(*) AS nb_lignes, COUNT(DISTINCT BEN_IDT_ANO) AS nb_patients
	FROM res.T_INDI_BPCO_VACG_&an_N.
	WHERE BPCO_VACG_OBS = 1;
	* nb_lignes = nb_patients : OK;

quit;

*
	**********************************************************************************************************
**********************************;
*	On ajoute les flag BPCO_VACG_PROB_CIBLE, BPCO_VACG_DIAG_CIBLE et BPCO_VACG_CIBLE dans la table
res.T_INDI_BPCO_&an_N.;

proc sql;

	ALTER TABLE res.T_INDI_BPCO_&an_N. DROP BPCO_VACG_PROB_CIBLE, BPCO_VACG_DIAG_CIBLE, BPCO_VACG_CIBLE;

	ALTER TABLE res.T_INDI_BPCO_&an_N. ADD BPCO_VACG_PROB_CIBLE INT format = f_oui_non. length = 3
		label = "Patient appartenant à la population cible BPCO_VACG_PROB";

	ALTER TABLE res.T_INDI_BPCO_&an_N. ADD BPCO_VACG_DIAG_CIBLE INT format = f_oui_non. length = 3
		label = "Patient appartenant à la population cible BPCO_VACG_DIAG";

	ALTER TABLE res.T_INDI_BPCO_&an_N. ADD BPCO_VACG_CIBLE INT format = f_oui_non. length = 3
		label = "Patient appartenant à la population cible BPCO_VACG";

	UPDATE res.T_INDI_BPCO_&an_N.
		SET BPCO_VACG_PROB_CIBLE = CASE	WHEN BEN_IDT_ANO IN (SELECT DISTINCT BEN_IDT_ANO FROM
res.T_INDI_BPCO_VACG_&an_N. WHERE

	BPCO_VACG_PROB_CIBLE = 1) THEN 1
	                                                                                ELSE 0
	                                                                                END,
	                        BPCO_VACG_DIAG_CIBLE = CASE		WHEN BEN_IDT_ANO IN (SELECT DISTINCT BEN_IDT_ANO
FROM res.T_INDI_BPCO_VACG_&an_N. WHERE

	BPCO_VACG_DIAG_CIBLE = 1) THEN 1
```

```
                                                                                    ELSE 0
                                                                                    END,
                                    BPCO_VACG_CIBLE = CASE                  WHEN BEN_IDT_ANO IN (SELECT DISTINCT BEN_IDT_ANO
FROM res.T_INDI_BPCO_VACG_&an_N. WHERE

            BPCO_VACG_CIBLE = 1) THEN 1
                                                                                    ELSE 0
                                                                                    END;

quit;

data res.T_INDI_BPCO_&an_N.;
            set res.T_INDI_BPCO_&an_N.;
            if BPCO_VACG_PROB_CIBLE = .
                        then BPCO_VACG_PROB_CIBLE = 0;
            if BPCO_VACG_DIAG_CIBLE = .
                        then BPCO_VACG_DIAG_CIBLE = 0;
            if BPCO_VACG_CIBLE = .
                        then BPCO_VACG_CIBLE = 0;
run;
```

3.8 Réalisation d'une EFR ou spirométrie annuelle chez les patients atteints de BPCO

3.8.1 00_Initialisation_du_Flowchart.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*

*/
/*			Table de population - Patients avec indicateur_03 = 1

			*/
/*

*/
/*
	*****************************************************************************************************************
**********************************	*/

data T_INDI_BPCO_EFR_SPIRO_&an_N. (keep = BEN_IDT_ANO BPCO_Probable Diag_BPCO BPCO_diagnostique);
	set res.T_INDI_BPCO_&an_N. (where = (indicateur_03 = 1 and (nb_broncho_LDA >= 3 or ALD_BPCO_decembre = 1 or
Nb_hospit_BPCO > 0)));
	length BPCO_Probable Diag_BPCO BPCO_diagnostique 3.;
	BPCO_Probable = 0;
	Diag_BPCO = 0;
	BPCO_diagnostique = 0;
	if nb_broncho_LDA >= 3 then
		BPCO_Probable = 1;
	if Nb_hospit_BPCO > 0 then
		Diag_BPCO = 1;
	if ALD_BPCO_decembre = 1 or Nb_hospit_BPCO > 0 then
		BPCO_diagnostique = 1;
	if BPCO_diagnostique = 1 then
		BPCO_Probable = 0;
run;

proc sort data = T_INDI_BPCO_EFR_SPIRO_&an_N. nodupkey;
	by BEN_IDT_ANO BPCO_Probable Diag_BPCO BPCO_diagnostique;
run;

*
	*****************************************************************************************************************
**********************************;
*	Effectifs pour le flowchart;

*	Individus de 40 ans et plus ayant une BPCO probable ou ayant une ALD BPCO active au 31 décembre de l année N ou ayant
eu au moins un diagnostic de
	BPCO codé lors d un séjour hospitalier en MCO terminé l année N ou lors d un séjour hospitalier SSR ou HAD l année N;

proc sql;

	CREATE TABLE flowch.Flow_Chart_BPCO_EFR_SPIRO_&an_N.  AS
		SELECT *
		FROM (
			SELECT 1 AS indicateur length = 3, COUNT(DISTINCT BEN_IDT_ANO) AS N length = 5 FROM
T_INDI_BPCO_EFR_SPIRO_&an_N.
				WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM res.T_INDI_BPCO_&an_N. WHERE
age >= 40 AND nb_broncho_LDA >= 3)
			UNION ALL
			SELECT 2, COUNT(DISTINCT BEN_IDT_ANO) FROM T_INDI_BPCO_EFR_SPIRO_&an_N. WHERE
BEN_IDT_ANO IN (SELECT BEN_IDT_ANO
				FROM res.T_INDI_BPCO_&an_N. WHERE age >= 40 AND ALD_BPCO_decembre > 0)
			UNION ALL
```

```
                                SELECT 3, COUNT(DISTINCT BEN_IDT_ANO) FROM T_INDI_BPCO_EFR_SPIRO_&an_N. WHERE
BEN_IDT_ANO IN (SELECT BEN_IDT_ANO
                                        FROM res.T_INDI_BPCO_&an_N. WHERE age >= 40 AND Nb_hospit_BPCO > 0)
                                UNION ALL
                                SELECT 4, COUNT(DISTINCT BEN_IDT_ANO) FROM T_INDI_BPCO_EFR_SPIRO_&an_N.
                        );

quit;
```

3.8.2 01_Inclusions_et_exclusions.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*		Indicateur 03 : Inclusion & exclusions

                                                                */
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/

data res.T_INDI_BPCO_EFR_SPIRO_&an_N.;
	set T_INDI_BPCO_EFR_SPIRO_&an_N.;
	BPCO_EFR_SPIRO_CIBLE = 1;
run;

* On ajoute la date de mise sous ALD;

proc sql;

	CREATE TABLE ALD_BCPO AS
		SELECT
			a.BEN_IDT_ANO,
			MIN(date_debut) AS ALD_BPCO_Date format ddmmyy10. length = 4
		FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N. a
			INNER JOIN travail.Histo_ALD b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.reperage = 52 AND a.BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM res.T_INDI_BPCO_&an_N. WHERE
ALD_BPCO_decembre = 1)
			AND ((b.date_debut <= "31DEC&annee_N."d <= b.date_fin) OR (b.date_debut <= "31DEC&annee_N."d
and b.date_fin = "01JAN1600"d))
		GROUP BY a.BEN_IDT_ANO
		ORDER BY a.BEN_IDT_ANO;

quit;

data res.T_INDI_BPCO_EFR_SPIRO_&an_N.;
	merge	res.T_INDI_BPCO_EFR_SPIRO_&an_N. (in = a)
		ALD_BCPO (in = b);
	by BEN_IDT_ANO;
	if a;
run;

proc delete data = ALD_BCPO;
run; quit;

*
	*****************************************************************************************************************
**********************************	;
*	EXCLUSION DES PATIENTS ASTHMATIQUES

                                                                ;
*
	*****************************************************************************************************************
**********************************	;

*
	*****************************************************************************************************************
**********************************;
*	Exclusion des patients en ALD asthme active au 28 février de l année N+1;

proc sql;
```

```
	CREATE TABLE ALD_asthme AS
		SELECT DISTINCT
			a.BEN_IDT_ANO
		FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N. a
			INNER JOIN travail.Histo_ALD b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.reperage = 53 AND (b.date_debut <= "28FEB&Annee_N1."d <= b.date_fin OR (b.date_debut <=
"28FEB&Annee_N1."d AND
			b.date_fin = "01JAN1600"d));

quit;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_EFR_SPIRO_&an_N.
		SELECT
			5,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM ALD_asthme);

	* On conserve les id des patients à supprimer;
	CREATE TABLE travail.exclus1 AS
		SELECT DISTINCT
			BEN_IDT_ANO
	FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM ALD_asthme);

quit;

proc delete data = ALD_asthme;
run; quit;

*
	**************************************************************************************************
**********************************;
*	Exclusion des patients en ALD mucoviscidose active au 28 février de l année N+1;

proc sql;

	CREATE TABLE ALD_mucoviscidose AS
		SELECT DISTINCT
			a.BEN_IDT_ANO
		FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N. a
			INNER JOIN travail.Histo_ALD b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.reperage = 54 AND (b.date_debut <= "28FEB&Annee_N1."d <= b.date_fin OR (b.date_debut <=
"28FEB&Annee_N1."d AND
			b.date_fin = "01JAN1600"d));

quit;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_EFR_SPIRO_&an_N.
		SELECT
			6,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM ALD_mucoviscidose);

	* On conserve les id des patients à supprimer;
	CREATE TABLE travail.exclus2 AS
		SELECT DISTINCT
			BEN_IDT_ANO
	FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM ALD_mucoviscidose);
```

```
quit;

proc delete data = ALD_mucoviscidose;
run; quit;

*
	**********************************************************************************************************
**********************************;
*	Exclusion des patients avec au moins 3 délivrances remboursées de CSI seuls entre le 1er janvier de l année N-1 et le 28
février de l année
	N+1;

proc sql;

	CREATE TABLE CSI AS
		SELECT DISTINCT
			a.BEN_IDT_ANO,
			COUNT(DISTINCT b.date_debut) AS Nb_deliv length = 3
		FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N. a
			INNER JOIN travail.reperage_med_asthme_&Annee_2N._&Annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.reperage = 41 AND ("01JAN&Annee_1N."d <= b.date_debut <= "28FEB&Annee_N1."d)
		GROUP BY a.BEN_IDT_ANO;

quit;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_EFR_SPIRO_&an_N.
		SELECT
			7,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM CSI WHERE Nb_deliv >= 3);

	* On conserve les id des patients à supprimer;
	CREATE TABLE travail.exclus3 AS
		SELECT DISTINCT
			BEN_IDT_ANO
	FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM CSI WHERE Nb_deliv >= 3);

quit;

proc delete data = CSI;
run; quit;

*
	**********************************************************************************************************
**********************************;
*	Exclusion des patients avec au moins 1 délivrance remboursée d'anti IgE entre le 1er janvier de l année N-1 et le 28 février de
l année N+1;

proc sql;

	CREATE TABLE anti_IGE AS
		SELECT DISTINCT
			a.BEN_IDT_ANO,
			COUNT(DISTINCT b.date_debut) AS Nb_deliv length = 3
		FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N. a
			INNER JOIN travail.reperage_med_asthme_&Annee_2N._&Annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.reperage IN (37) AND ("01JAN&Annee_1N."d <= b.date_debut <= "28FEB&Annee_N1."d)
		GROUP BY a.BEN_IDT_ANO;

quit;

proc sql;

	* On insère l effectif dans le Flowchart;
```

```
	INSERT INTO flowch.Flow_Chart_BPCO_EFR_SPIRO_&an_N.
		SELECT
			8,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM anti_IGE WHERE Nb_deliv >= 1);

	* On conserve les id des patients à supprimer;
	CREATE TABLE travail.exclus4 AS
		SELECT DISTINCT
			BEN_IDT_ANO
	FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM anti_IGE WHERE Nb_deliv >= 1);

quit;

proc delete data = anti_IGE;
run; quit;

*
	**********************************************************************************************************
**********************************;
*	Exclusion des patients avec au moins 1 délivrance remboursée d'anti IL-5 entre le 1er janvier de l année N-1 et le 28 février
de l année N+1;

proc sql;

	CREATE TABLE anti_IL5 AS
		SELECT DISTINCT
			a.BEN_IDT_ANO,
			COUNT(DISTINCT b.date_debut) AS Nb_deliv length = 3
		FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N. a
			INNER JOIN travail.reperage_med_asthme_&Annee_2N._&Annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.reperage IN (38) AND ("01JAN&Annee_1N."d <= b.date_debut <= "28FEB&Annee_N1."d)
		GROUP BY a.BEN_IDT_ANO;

quit;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_EFR_SPIRO_&an_N.
		SELECT
			9,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM anti_IL5 WHERE Nb_deliv >= 1);

	* On conserve les id des patients à supprimer;
	CREATE TABLE travail.exclus5 AS
		SELECT DISTINCT
			BEN_IDT_ANO
	FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM anti_IL5 WHERE Nb_deliv >= 1);

quit;

proc delete data = anti_IL5;
run; quit;

*
	**********************************************************************************************************
**********************************;
*	Exclusion des patients avec au moins 1 délivrance remboursée d'antileucotriène entre le 1er janvier de l année N-1 et le 28
février de l année N+1;

proc sql;

	CREATE TABLE antileucotriene AS
		SELECT DISTINCT
```

```
                                a.BEN_IDT_ANO,
                                COUNT(DISTINCT b.date_debut) AS Nb_deliv length = 3
                        FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N. a
                                INNER JOIN travail.reperage_med_asthme_&Annee_2N._&Annee_N1. b
                                        ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
                        WHERE b.reperage = 39 AND ("01JAN&Annee_1N."d <= b.date_debut <= "28FEB&Annee_N1."d)
                        GROUP BY a.BEN_IDT_ANO;

quit;

proc sql;

        * On insère l effectif dans le Flowchart;
        INSERT INTO flowch.Flow_Chart_BPCO_EFR_SPIRO_&an_N.
                SELECT
                        10,
                        COUNT(DISTINCT BEN_IDT_ANO)
        FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N.
        WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM antileucotriene WHERE Nb_deliv >= 1);

        * On conserve les id des patients à supprimer;
        CREATE TABLE travail.exclus6 AS
                SELECT DISTINCT
                        BEN_IDT_ANO
        FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N.
        WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM antileucotriene WHERE Nb_deliv >= 1);

quit;

proc delete data = antileucotriene;
run; quit;

*
        ************************************************************************************************************
************************************;
*       Exclusion des patients avec au moins un acte CCAM de Thermoplastie bronchique codé lors d'un sé-jour hospitalier en MCO
terminé entre
        le 1er janvier de l année N-1 et le 28 février de l année N+1;

proc sql;

        CREATE TABLE thermo_bronchique AS
                SELECT DISTINCT
                        a.BEN_IDT_ANO,
                        COUNT(DISTINCT b.date_debut) AS Nb_actes length = 3
                FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N. a
                        INNER JOIN travail.reperage_thermo_bronch_&annee_1N._&annee_N1. b
                                ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
                WHERE "01JAN&Annee_1N."d <= b.date_fin <= "28FEB&Annee_N1."d
                GROUP BY a.BEN_IDT_ANO;

quit;

proc sql;

        * On insère l effectif dans le Flowchart;
        INSERT INTO flowch.Flow_Chart_BPCO_EFR_SPIRO_&an_N.
                SELECT
                        11,
                        COUNT(DISTINCT BEN_IDT_ANO)
        FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N.
        WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM thermo_bronchique WHERE Nb_actes >= 1);

        * On conserve les id des patients à supprimer;
        CREATE TABLE travail.exclus7 AS
                SELECT DISTINCT
                        BEN_IDT_ANO
        FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N.
        WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM thermo_bronchique WHERE Nb_actes >= 1);

quit;
```

```
proc delete data = thermo_bronchique;
run; quit;

*
	**********************************************************************************************************
**********************************;
*	Exclusion des patients ayant eu au moins un diagnostic d'asthme codé entre le 1er janvier de l année N-1 et le 28 février de l
année N+1;

proc sql;

	CREATE TABLE diag_asthme_MCO AS
		SELECT DISTINCT
			a.BEN_IDT_ANO
		FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N. a
			INNER JOIN travail.diag_asthme_&annee_1N._&annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.domaine = "MCO" AND b.reperage = 14 AND "01JAN&Annee_1N."d <= b.date_fin <=
"28FEB&Annee_N1."d;

	CREATE TABLE diag_asthme_SSR_HAD AS
		SELECT DISTINCT
			a.BEN_IDT_ANO
		FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N. a
			INNER JOIN travail.diag_asthme_&annee_1N._&annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.domaine IN ("HAD", "SSR") AND b.reperage = 14 AND b.date_debut <= "28FEB&Annee_N1."d AND
b.date_fin >= "01JAN&Annee_1N."d;

quit;

data diag_asthme;
	set diag_asthme_MCO diag_asthme_SSR_HAD;
run;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_EFR_SPIRO_&an_N.
		SELECT
			12,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM diag_asthme);

	* On conserve les id des patients à supprimer;
	CREATE TABLE travail.exclus8 AS
		SELECT DISTINCT
			BEN_IDT_ANO
	FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM diag_asthme);

quit;

proc delete data = diag_asthme diag_asthme_MCO diag_asthme_SSR_HAD;
run; quit;

*
	**********************************************************************************************************
**********************************;
*	Exclusion des patients ayant eu au moins un diagnostic d'état de mal asthmatique codé entre le 1er janvier de l année N-1 et
le 28 février
	de l année N+1;

proc sql;

	CREATE TABLE etat_mal_asthmatique AS
		SELECT DISTINCT
			a.BEN_IDT_ANO
		FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N. a
```

```
                                INNER JOIN travail.diag_asthme_&annee_1N._&annee_N1. b
                                        ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
                        WHERE reperage = 15 AND "01JAN&Annee_1N."d <= b.date_fin <= "28FEB&Annee_N1."d;

quit;

proc sql;

        * On insère l effectif dans le Flowchart;
        INSERT INTO flowch.Flow_Chart_BPCO_EFR_SPIRO_&an_N.
                SELECT
                        13,
                        COUNT(DISTINCT BEN_IDT_ANO)
        FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N.
        WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM etat_mal_asthmatique);

        * On conserve les id des patients à supprimer;
        CREATE TABLE travail.exclus9 AS
                SELECT DISTINCT
                        BEN_IDT_ANO
        FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N.
        WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM etat_mal_asthmatique);

quit;

proc delete data = etat_mal_asthmatique;
run; quit;

*
        ************************************************************************************************************
***********************************;
*       On exclut tous les patients;

data exclus;
        set     travail.exclus1-travail.exclus9;
run;

proc sql;

        * On insère l effectif dans le Flowchart;
        INSERT INTO flowch.Flow_Chart_BPCO_EFR_SPIRO_&an_N.
                SELECT
                        14,
                        COUNT(DISTINCT BEN_IDT_ANO)
        FROM exclus;

        DELETE FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N.
        WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM exclus);

quit;

proc delete data = exclus;
run; quit;

proc datasets library = travail memtype = data nolist;
        delete exclus1-exclus9;
run; quit;

* On insère l effectif de la population d étude hors asthamtiques dans le Flowchart;
proc sql;

        INSERT INTO flowch.Flow_Chart_BPCO_EFR_SPIRO_&an_N.
                SELECT
                        15,
                        COUNT(DISTINCT BEN_IDT_ANO)
        FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N.;

quit;
```

```
*
	*************************************************************************************************************
**********************************	;
*	PATIENTS POUR LESQUELS LA SPIROMÉTRIE OU L EFR N EST PAS RÉALISABLE

				;
*
	*************************************************************************************************************
**********************************	;

*
	*************************************************************************************************************
**********************************;
*	Exclusion des patients en ALD maladie d Alzheimer et autres démences active au 28 février de l année N+1;

proc sql;

	CREATE TABLE ALD_Alzheimer AS
		SELECT DISTINCT
			a.BEN_IDT_ANO
		FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N. a
			INNER JOIN travail.Histo_ALD b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.reperage IN (50, 51) AND (b.date_debut <= "28FEB&Annee_N1."d <= b.date_fin OR (b.date_debut <=
"28FEB&Annee_N1."d AND
			b.date_fin = "01JAN1600"d));

quit;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_EFR_SPIRO_&an_N.
		SELECT
			16,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM ALD_Alzheimer);

	* On conserve les id des patients à supprimer;
	CREATE TABLE travail.exclus1 AS
		SELECT DISTINCT
			BEN_IDT_ANO
	FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM ALD_Alzheimer);

quit;

proc delete data = ALD_Alzheimer;
run; quit;

*
	*************************************************************************************************************
**********************************;
*	Exclusion des patients ayant eu un diagnostic de soins palliatif codé lors d'un séjour hospitalier MCO terminé entre le 1er
janvier de l année N
	et le 28 février de l année N+1, ou lors d'un séjour hospitalier HAD, SSR entre le 1er janvier de l année N et le 28 février de l
année N+1;

* On exclut les séjours avec un séjour MCO terminé pendant la campagne;
* On exclut les séjours avec un séjour SSR ou HAD pendant la campagne;
proc sql undo_policy = none;

	CREATE TABLE soin_palliatif_MCO AS
		SELECT DISTINCT
			a.BEN_IDT_ANO
		FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N. a
			INNER JOIN travail.soin_palliatif_&annee_2N._&annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE "01JAN&Annee_N."d <= b.date_fin <= "28FEB&Annee_N1"d AND b.domaine = "MCO";
```

```
	CREATE TABLE soin_palliatif_SSR_HAD AS
		SELECT DISTINCT
			a.BEN_IDT_ANO
		FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N. a
			INNER JOIN travail.soin_palliatif_&annee_2N._&annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.domaine IN ("HAD", "SSR") AND b.date_debut <= "28FEB&Annee_N1."d AND b.date_fin >=
"01JAN&Annee_N."d;

quit;

data soin_palliatif;
	set soin_palliatif_MCO
		soin_palliatif_SSR_HAD;
run;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_EFR_SPIRO_&an_N.
		SELECT
			17,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM soin_palliatif);

	* On conserve les id des séjours à supprimer;
	CREATE TABLE travail.exclus2 AS
		SELECT DISTINCT
			BEN_IDT_ANO
	FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM soin_palliatif);

quit;

proc datasets library = work memtype = data nolist;
	delete soin_palliatif_MCO soin_palliatif_SSR_HAD soin_palliatif ;
run; quit;

*
	**********************************************************************************************************
***********************************;
*	Exclusion des patients hospitalisés en SSR ou psychiatrie plus de 90 jours entre le 1er janvier de l année N et le 28 février de l
année N+1;

* On regarde si les patients ont un séjours > 3 mois;
proc sql;

	CREATE TABLE hospit1_3mois AS
		SELECT DISTINCT
			a.BEN_IDT_ANO
		FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N. a
			INNER JOIN travail.sejours_&Annee_N._&Annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.domaine IN ("RIP", "SSR") AND b.date_debut < "01JAN&Annee_N."d AND (b.date_fin -
"01JAN&Annee_N."d) > 90;

	CREATE TABLE hospit2_3mois AS
		SELECT DISTINCT
			a.BEN_IDT_ANO
		FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N. a
			INNER JOIN travail.sejours_&Annee_N._&Annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.domaine IN ("RIP", "SSR") AND "01JAN&Annee_N."d <= b.date_debut <= "28FEB&Annee_N1"d AND
			"01JAN&Annee_N."d <= b.date_fin <= "28FEB&Annee_N1"d AND (b.date_fin - b.date_debut) > 90;

	CREATE TABLE hospit3_3mois AS
		SELECT DISTINCT
			a.BEN_IDT_ANO
		FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N. a
			INNER JOIN travail.sejours_&Annee_N._&Annee_N1. b
```

```
                                        ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
                        WHERE b.domaine IN ("RIP", "SSR") AND b.date_debut >= "01JAN&Annee_N."d AND b.date_fin >
"28FEB&Annee_N1"d
                                AND ("28FEB&Annee_N1"d - b.date_debut) > 90;

quit;

data hospit_3mois;
        set hospit1_3mois hospit2_3mois hospit3_3mois;
run;

proc sort data = hospit_3mois (keep = BEN_IDT_ANO) nodupkey;
        by BEN_IDT_ANO;
run;

proc sql;

        * On insère l effectif dans le Flowchart;
        INSERT INTO flowch.Flow_Chart_BPCO_EFR_SPIRO_&an_N.
                SELECT
                        18,
                        COUNT(DISTINCT BEN_IDT_ANO)
        FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N.
        WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM hospit_3mois);

        * On conserve les id des séjours à supprimer;
        CREATE TABLE travail.exclus3 AS
                SELECT DISTINCT
                        BEN_IDT_ANO
        FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N.
        WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM hospit_3mois);

quit;

proc delete data = hospit_3mois;
run; quit;

*
        ************************************************************************************************************
***********************************;
*       On flag les patients pour lesquels la spirométrie ou les EFR ne sont pas réalisables;

data exclus;
        set     travail.exclus1-travail.exclus3;
run;

proc sql;

        * On insère l effectif dans le Flowchart;
        INSERT INTO flowch.Flow_Chart_BPCO_EFR_SPIRO_&an_N.
                SELECT
                        19,
                        COUNT(DISTINCT BEN_IDT_ANO)
        FROM exclus;

        DELETE FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N.
        WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM exclus);

quit;

proc delete data = exclus;
run; quit;

proc datasets library = travail memtype = data nolist;
        delete exclus1-exclus3;
run; quit;

proc sort data = res.T_INDI_BPCO_EFR_SPIRO_&an_N. nodupkey;
        by _all_;
run;
```

```
*
	**********************************************************************************************************
**********************************;
*	Population cible;

proc sql;

	INSERT INTO flowch.Flow_Chart_BPCO_EFR_SPIRO_&an_N.
		SELECT
			21,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N.;

quit;

proc sql;

	SELECT COUNT(*), COUNT(DISTINCT BEN_IDT_ANO)
	FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N.;

quit;
```

3.8.3 02_Sejours_avec_un_diag_de_BPCO.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*		Indicateur 03 : Nombre de séjours BPCO
                                                                                                                    */
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/

proc sql;

	CREATE INDEX BEN_IDT_ANO ON res.T_INDI_BPCO_EFR_SPIRO_&an_N. (BEN_IDT_ANO);

quit;

%macro sejours_BPCO(domaine=);

	proc sql;

		CREATE TABLE sejours_BPCO_&domaine. AS
			SELECT DISTINCT
				a.BEN_IDT_ANO,
				COUNT(DISTINCT id_sejour) AS Nb_Sej_Diag_BPCO_&domaine. length = 3
			FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N. a
				INNER JOIN travail.reperage_hospit_BPCO_&annee_4N._&annee_N1. b
					ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
			WHERE b.domaine = "&domaine." AND annee = &Annee_N.
			GROUP BY a.BEN_IDT_ANO
			;

		CREATE INDEX BEN_IDT_ANO ON sejours_BPCO_&domaine. (BEN_IDT_ANO);

		ALTER TABLE res.T_INDI_BPCO_EFR_SPIRO_&an_N. ADD Nb_Sej_Diag_BPCO_&domaine. INT length = 3;

		UPDATE res.T_INDI_BPCO_EFR_SPIRO_&an_N. a
			SET Nb_Sej_Diag_BPCO_&domaine. =	(SELECT Nb_Sej_Diag_BPCO_&domaine.

	FROM sejours_BPCO_&domaine. b

	WHERE a.BEN_IDT_ANO = b.BEN_IDT_ANO);

	quit;

	data res.T_INDI_BPCO_EFR_SPIRO_&an_N.;
		set res.T_INDI_BPCO_EFR_SPIRO_&an_N.;
		if Nb_Sej_Diag_BPCO_&domaine. = . then
			Nb_Sej_Diag_BPCO_&domaine. = 0;
	run;

	proc delete data = sejours_BPCO_&domaine.;
	run; quit;

%mend sejours_BPCO;

%sejours_BPCO(domaine = MCO);
%sejours_BPCO(domaine = SSR);
%sejours_BPCO(domaine = HAD);
```

3.8.4 03_Realisation_EFR_ou_spiro.sas

```
/*
	*****************************************************************************************************
**********************************	*/
/*
                                                                                                                */
/*		Indicateur 03 : EFR ou Spirométries
                                                                                */
/*
                                                                                                                */
/*
	*****************************************************************************************************
**********************************	*/

*
	*****************************************************************************************************
**********************************;
*	On repère les patients avec une spirométrie entre le 1er janvier de l année N et le 28 février de l année N+1;

proc sql;

	CREATE TABLE spirometrie AS
		SELECT DISTINCT
			a.BEN_IDT_ANO,
			MIN(b.date_debut) AS date_debut length = 4,
			MIN(CASE WHEN b.date_debut < "01JAN&Annee_N."d THEN "01JAN&Annee_N."d
					ELSE b.date_debut
					END) AS Date_Spiro length = 4,
			COUNT(DISTINCT b.date_debut) AS Nb_spiro length = 3
		FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N. a
			INNER JOIN travail.reperage_EFR_spiro_&Annee_2N._&annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.reperage = 2 AND (b.date_debut <= "28FEB&Annee_N1."d AND b.date_fin >= "01JAN&Annee_N."d)
		GROUP BY a.BEN_IDT_ANO;

	* On fait un max car si DCIR + PMSI, c'est un doublon donc la spiro a lieu dans un etb;
	CREATE TABLE lieu_spiro AS
		SELECT DISTINCT
			a.BEN_IDT_ANO,
			MAX(CASE WHEN b.domaine = "" THEN 1
					WHEN b.domaine = "MCO" THEN 2
					WHEN b.domaine = "SSR" THEN 3
					END) AS Lieu_Spiro length = 3
			FROM spirometrie a
				LEFT JOIN travail.reperage_EFR_spiro_&Annee_2N._&annee_N1. b
					ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
					AND a.date_debut = b.date_debut
			WHERE b.reperage = 2
			GROUP BY a.BEN_IDT_ANO;

quit;

* On ajoute l information dans la table de résultats;

proc sort data = res.T_INDI_BPCO_EFR_SPIRO_&an_N.;
	by BEN_IDT_ANO;
run;

proc sort data = spirometrie;
	by BEN_IDT_ANO;
run;

proc sort data = lieu_spiro;
	by BEN_IDT_ANO;
run;
```

```
data res.T_INDI_BPCO_EFR_SPIRO_&an_N.;
	merge	res.T_INDI_BPCO_EFR_SPIRO_&an_N. (in = a)
			spirometrie (in = b)
			lieu_spiro (in = c);
	by BEN_IDT_ANO;
	length Spiro 3.;
	Spiro = 0;
	if Nb_spiro > 0 then
		Spiro = 1;
	if a then
		output;
run;

proc delete data = spirometrie lieu_spiro;
run; quit;

*
	*************************************************************************************************************
***********************************;
*	On repère les patients avec une EFR entre le 1er janvier de l année N et le 28 février de l année N+1;

proc sql;

	CREATE TABLE EFR AS
		SELECT DISTINCT
			a.BEN_IDT_ANO,
			MIN(b.date_debut) AS date_debut length = 4,
			MIN(CASE WHEN b.date_debut < "01JAN&Annee_N."d THEN "01JAN&Annee_N."d
					ELSE b.date_debut
					END) AS Date_EFR length = 4,
			COUNT(DISTINCT b.date_debut) AS Nb_EFR length = 3
		FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N. a
			INNER JOIN travail.reperage_EFR_spiro_&Annee_2N._&annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.reperage = 1 AND (b.date_debut <= "28FEB&Annee_N1."d AND b.date_fin >= "01JAN&Annee_N."d)
		GROUP BY a.BEN_IDT_ANO;

	* On fait un max car si DCIR + PMSI, c'est un doublon donc l EFR a lieu dans un etb;
	CREATE TABLE lieu_EFR AS
		SELECT DISTINCT
			a.BEN_IDT_ANO,
			MAX(CASE WHEN b.domaine = "" THEN 1
					WHEN b.domaine = "MCO" THEN 2
					WHEN b.domaine = "SSR" THEN 3
					END) AS Lieu_EFR length = 3
			FROM EFR a
				LEFT JOIN travail.reperage_EFR_spiro_&Annee_2N._&annee_N1. b
					ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
					AND a.date_debut = b.date_debut
			WHERE b.reperage = 1
			GROUP BY a.BEN_IDT_ANO;

quit;

* On ajoute l information dans la table de résultats;

proc sort data = res.T_INDI_BPCO_EFR_SPIRO_&an_N.;
	by BEN_IDT_ANO;
run;

proc sort data = EFR;
	by BEN_IDT_ANO;
run;

proc sort data = lieu_EFR;
	by BEN_IDT_ANO;
run;

data res.T_INDI_BPCO_EFR_SPIRO_&an_N.;
	merge	res.T_INDI_BPCO_EFR_SPIRO_&an_N. (in = a)
```

```
                                    EFR (in = b)
                                    lieu_EFR (in = c);
        by BEN_IDT_ANO;
        length EFR BPCO_EFR_SPIRO_OBS 3.;
        EFR = 0;
        if Nb_EFR > 0 then
                EFR = 1;
        * Patient de la population cible ayant bénéficié d une EFR ou une spiro;
        BPCO_EFR_SPIRO_OBS = 0;
        if EFR = 1 or Spiro = 1 then
                BPCO_EFR_SPIRO_OBS = 1;
        if a then
                output;
run;

proc delete data = EFR lieu_EFR;
run; quit;
```

3.8.5 04_Pneumo_Infarctus_Oxygeno_VNI.sas

```
/*
	*****************************************************************************************************
**********************************	*/
/*

*/
/*		Indicateur 03 : Pneumothorax, infarctus, oxygénothérapie et VNI

		*/
/*

*/
/*
	*****************************************************************************************************
**********************************	*/

%macro add_info_ind03(table=, in_indic=, out_indic=);

	* Patients avec le soin entre le 1er janvier &an_N. et le 28 février &an_N1.;
	proc sql;

		CREATE TABLE patients AS
			SELECT DISTINCT
				a.BEN_IDT_ANO,
				1 AS &out_indic. length = 3
			FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N. a
				INNER JOIN &table. b
					ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
			WHERE b.reperage IN &in_indic. AND b.date_debut <= "28FEB&Annee_N1."d AND b.date_fin >=
"01JAN&Annee_N."d
			ORDER BY a.BEN_IDT_ANO;

	quit;

	data res.T_INDI_BPCO_EFR_SPIRO_&an_N.;
		merge	res.T_INDI_BPCO_EFR_SPIRO_&an_N. (in = a)
				patients (in = b);
		by BEN_IDT_ANO;
		if a;
		if not b then
			&out_indic. = 0;
	run;

	proc delete data = patients;
	run; quit;

%mend add_info_ind03;

* Pneumothorax;
%add_info_ind03(
	table = travail.pneumothorax_infarctus_&Annee_1N._&annee_N1.,
	in_indic = (11),
	out_indic = Pneumothorax
	);

* Infarctus;
%add_info_ind03(
	table = travail.pneumothorax_infarctus_&Annee_1N._&annee_N1.,
	in_indic = (12),
	out_indic = Infarctus
	);

* Oxygénothérapie;
%add_info_ind03(
	table = travail.reperage_Oxygeno_&Annee_N._&annee_N1.,
	in_indic = (5),
	out_indic = Oxygenotherapie
```

```
		);

* VNI;
%add_info_ind03(
		table = travail.reperage_VNI_&Annee_N._&annee_N1.,
		in_indic = (6),
		out_indic = VNI
		);
```

3.8.6 05_Table_finale.sas

```
/*
	*************************************************************************************************************
**********************************	*/
/*
                                                                                                                */
/*		Indicateur 03 : Table finale - Ajout de formats et de labels

		*/
/*
                                                                                                                */
/*
	*************************************************************************************************************
**********************************	*/

data res.T_INDI_BPCO_EFR_SPIRO_&an_N.;
	set res.T_INDI_BPCO_EFR_SPIRO_&an_N. (keep = BEN_IDT_ANO BPCO_Probable Diag_BPCO BPCO_Diagnostique
BPCO_EFR_SPIRO_CIBLE ALD_BPCO_Date
		Nb_Sej_Diag_BPCO_MCO Nb_Sej_Diag_BPCO_SSR Nb_Sej_Diag_BPCO_HAD BPCO_EFR_SPIRO_OBS Spiro
Nb_Spiro Date_Spiro Lieu_Spiro EFR Nb_EFR
		Date_EFR Lieu_EFR Pneumothorax Infarctus Oxygenotherapie VNI);
	label
		BEN_IDT_ANO = "Numéro d'individu"
		BPCO_Probable = "Patient ayant une BPCO probable"
		Diag_BPCO = "Patient ayant eu au moins un diagnostic de BPCO codé lors d'un séjour hospitalier en MCO terminé
en &Annee_N. ou lors d'un séjour hospitalier en SSR, HAD en &Annee_N."
		BPCO_Diagnostique = "Patient diagnostiqué BPCO"
		BPCO_EFR_SPIRO_CIBLE = "Patient correspondant aux critères d'inclusion et d'exclusion de la population cible"
		ALD_BPCO_Date = "Date de début de mise en ALD BPCO"
		Nb_Sej_Diag_BPCO_MCO = "Nombre de séjours avec un diagnostic BPCO terminés en &Annee_N. en MCO"
		Nb_Sej_Diag_BPCO_SSR = "Nombre de séjours avec un diagnostic BPCO en &Annee_N. en SSR"
		Nb_Sej_Diag_BPCO_HAD = "Nombre de séjours avec un diagnostic BPCO en &Annee_N. en HAD"
		BPCO_EFR_SPIRO_OBS = "Patient de la population cible ayant bénéficiés d'une exploration fonctionnelle
respiratoire ou d'une spirométrie entre le 1er janvier &Annee_N. et le 28 février &Annee_N1."
		Spiro = "Spirométrie réalisée entre le 1er janvier &Annee_N. et le 28 février &Annee_N1."
		Nb_Spiro = "Nombre de spirométries réalisées entre le 1er janvier &Annee_N. et le 28 février &Annee_N1."
		Date_Spiro = "Date de la réalisation de la première spirométrie entre le 1er janvier &Annee_N. et le 28 février
&Annee_N1."
		Lieu_Spiro = "Lieu de réalisation de la première spirométrie"
		EFR = "EFR réalisé entre le 1er janvier &Annee_N. et le 28 février &Annee_N1."
		Nb_EFR = "Nombre d'EFR réalisés entre le 1er janvier &Annee_N. et le 28 février &Annee_N1."
		Date_EFR = "Date de la première réalisation d'EFR entre le 1er janvier &Annee_N. et le 28 février &Annee_N1."
		Lieu_EFR = "Lieu de réalisation du première EFR entre le 1er janvier &Annee_N. et le 28 février &Annee_N1."
		Pneumothorax = "Patients ayant eu un pneumothorax entre le 1er janvier &Annee_N. et le 28 février
&Annee_N1."
		Infarctus = "Patients ayant un diagnostic d'infarctus entre le 1er janvier &Annee_N. et le 28 février &Annee_N1."
		Oxygenotherapie = "Patients ayant eu une délivrance remboursée d'un traitement par oxygénothérapie entre le
1er janvier &Annee_N. et le 28 février &Annee_N1."
		VNI = "Patients ayant eu une délivrance remboursée d'un traitement par VNI entre le 1er janvier &Annee_N. et le
28 février &Annee_N1."
		;
	format BPCO_Probable Diag_BPCO BPCO_Diagnostique BPCO_EFR_SPIRO_CIBLE BPCO_EFR_SPIRO_OBS Spiro EFR
Pneumothorax Infarctus Oxygenotherapie
		VNI f_oui_non. ALD_BPCO_Date Date_Spiro Date_EFR date9. Lieu_Spiro Lieu_EFR f_lieu_soin.;
run;

*
	*************************************************************************************************************
**********************************;
*	Réalisation d EFR ou d une spirométrie annuelle;

proc sql;

	INSERT INTO flowch.Flow_Chart_BPCO_EFR_SPIRO_&an_N.
		SELECT
			22,
```

```
                                    COUNT(DISTINCT BEN_IDT_ANO)
        FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N.
        WHERE BPCO_EFR_SPIRO_CIBLE = 1 AND BPCO_EFR_SPIRO_OBS = 1;


        * Nombre de patients pour les numérateur des premières exclusions;
        SELECT Nb_patients_ref0 into : N0 FROM travail.ref_flowcharts;
        * Nombre de patients  de 40 ans et plus ayant une BPCO probable ou diagnostiquée;
        SELECT N into : N4 FROM flowch.Flow_Chart_BPCO_EFR_SPIRO_&an_N. WHERE indicateur = 4;

        * population d étude;
        SELECT N into : N15 FROM flowch.Flow_Chart_BPCO_EFR_SPIRO_&an_N. WHERE indicateur = 15;

        * Population cible;
        SELECT N into : N21 FROM flowch.Flow_Chart_BPCO_EFR_SPIRO_&an_N. WHERE indicateur = 21;

quit;

data flowch.Flow_Chart_BPCO_EFR_SPIRO_&an_N.;
        set flowch.Flow_Chart_BPCO_EFR_SPIRO_&an_N.;
        if indicateur in (1, 2, 3, 4) then
                percent = N / &N0.;
        if indicateur in (5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15) then
                percent = N / &N4.;
        if indicateur in (16, 17, 18, 19, 20, 21) then
                percent = N / &N15.;
        if indicateur = 22 then
                percent = N / &N21.;
        format indicateur f_ind_03_flowchart. percent percent7.1;
run;

* Vérif;
proc sql;

        SELECT COUNT(*) AS nb_lignes, COUNT(DISTINCT BEN_IDT_ANO) AS nb_patients
        FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N.;
        * nb_lignes = nb_patients : OK;

        SELECT COUNT(*) AS nb_lignes, COUNT(DISTINCT BEN_IDT_ANO) AS nb_patients
        FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N.
        WHERE BPCO_EFR_SPIRO_CIBLE = 1;
        * nb_lignes = nb_patients : OK;

quit;

*
        ********************************************************************************************************
************************************;
*       On ajoute le flag BPCO_EFR_SPIRO_CIBLE dans la table res.T_INDI_BPCO_&an_N.;

proc sql;

        ALTER TABLE res.T_INDI_BPCO_&an_N. DROP BPCO_EFR_SPIRO_CIBLE;

        ALTER TABLE res.T_INDI_BPCO_&an_N. ADD BPCO_EFR_SPIRO_CIBLE INT format = f_oui_non. length = 3
                label = "Patient appartenant à la population cible BPCO_EFR_SPIRO";

        UPDATE res.T_INDI_BPCO_&an_N.
                SET BPCO_EFR_SPIRO_CIBLE = CASE         WHEN BEN_IDT_ANO IN (SELECT DISTINCT BEN_IDT_ANO FROM
res.T_INDI_BPCO_EFR_SPIRO_&an_N. WHERE

        BPCO_EFR_SPIRO_CIBLE = 1) THEN 1
                                                                                        ELSE 0
                                                                                        END;

quit;

data res.T_INDI_BPCO_&an_N.;
        set res.T_INDI_BPCO_&an_N.;
        if BPCO_EFR_SPIRO_CIBLE = .
                then BPCO_EFR_SPIRO_CIBLE = 0;
```

```
run;
```

3.9 Suivi médical dans les 7 jours après une hospitalisation pour exacerbation de BPCO des patients sortis à domicile

3.9.1 00_Initialisation_de_la_table_de_resultats.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*

*/
/*			Table de population - Patients avec indicateur_04 = 1

		*/
/*

*/
/*
	*****************************************************************************************************************
**********************************	*/

data T_INDI_BPCO_SMED_P_EA_&an_N. (keep = BEN_IDT_ANO);
	set pop.indicateurs_&an_N. (where = (indicateur_04 = 1));
run;

*
	*****************************************************************************************************************
**********************************;
*	Effectifs pour le flowchart;

* Séjours pour exacerbation : individus de 40 ans et plus ayant bénéficié de soins remboursés au moins une fois l année N;

proc sort data = T_INDI_BPCO_SMED_P_EA_&an_N. nodupkey;
	by BEN_IDT_ANO;
run;

proc sort data = travail.sejours_cible_exacerbation_&annee_N.;
	by BEN_IDT_ANO;
run;

data T_INDI_BPCO_SMED_P_EA_&an_N.;
	merge	T_INDI_BPCO_SMED_P_EA_&an_N. (in = a)
		travail.sejours_cible_exacerbation_&annee_N. (in = b);
	by BEN_IDT_ANO;
	if a and b;
	BPCO_SMED_P_EA_Sej_Index = id_sejour;
run;

proc sql;

	CREATE TABLE flowch.Flow_Chart_BPCO_SMED_P_EA_&an_N.  AS
		SELECT *
		FROM (
			SELECT 1 AS indicateur length = 3, COUNT(DISTINCT BPCO_SMED_P_EA_Sej_Index) AS N length = 5
FROM T_INDI_BPCO_SMED_P_EA_&an_N. WHERE exacerbation_BPCO = 1
			UNION ALL
			SELECT 2, COUNT(DISTINCT BPCO_SMED_P_EA_Sej_Index) FROM T_INDI_BPCO_SMED_P_EA_&an_N.
WHERE exacerbation_BPCO = 2
			UNION ALL
			SELECT 3, COUNT(DISTINCT BPCO_SMED_P_EA_Sej_Index) FROM T_INDI_BPCO_SMED_P_EA_&an_N.
WHERE exacerbation_BPCO = 3
			UNION ALL
			SELECT 4, COUNT(DISTINCT BPCO_SMED_P_EA_Sej_Index) FROM T_INDI_BPCO_SMED_P_EA_&an_N.
WHERE exacerbation_BPCO = 4
			UNION ALL
```

```
                                SELECT 5, COUNT(DISTINCT BPCO_SMED_P_EA_Sej_Index) FROM T_INDI_BPCO_SMED_P_EA_&an_N.
WHERE exacerbation_BPCO = 5
                                UNION ALL
                                SELECT 6, COUNT(DISTINCT BPCO_SMED_P_EA_Sej_Index) FROM T_INDI_BPCO_SMED_P_EA_&an_N.
WHERE exacerbation_BPCO = 6
                                UNION ALL
                                SELECT 7, COUNT(DISTINCT BPCO_SMED_P_EA_Sej_Index) FROM T_INDI_BPCO_SMED_P_EA_&an_N.
WHERE exacerbation_BPCO = 7
                                UNION ALL
                                SELECT 8, COUNT(DISTINCT BPCO_SMED_P_EA_Sej_Index) FROM T_INDI_BPCO_SMED_P_EA_&an_N.
WHERE exacerbation_BPCO = 8
                                UNION ALL
                                SELECT 9, COUNT(DISTINCT BPCO_SMED_P_EA_Sej_Index) FROM T_INDI_BPCO_SMED_P_EA_&an_N.
WHERE exacerbation_BPCO = 9
                                UNION ALL
                                SELECT 10, COUNT(DISTINCT BPCO_SMED_P_EA_Sej_Index) FROM T_INDI_BPCO_SMED_P_EA_&an_N.
WHERE exacerbation_BPCO = 10
                                UNION ALL
                                SELECT 11, COUNT(DISTINCT BPCO_SMED_P_EA_Sej_Index) FROM T_INDI_BPCO_SMED_P_EA_&an_N.
WHERE exacerbation_BPCO = 11
                                UNION ALL
                                SELECT 12, COUNT(DISTINCT BPCO_SMED_P_EA_Sej_Index) FROM T_INDI_BPCO_SMED_P_EA_&an_N.
                        ) ;

quit;
```

3.9.2 01_Inclusions_et_exclusions.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*		Indicateur 04 : Inclusion & exclusions
                                                                                                                    */
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/

*
	*****************************************************************************************************************
**********************************;
*	On récupère tous les séjours pour exacerbation de BPCO pour les patients de la table
res.T_INDI_BPCO_SMED_P_EA_&an_N.;

* Séjours avec BPCO en DAS;
data BPCO_DAS;
	set travail.sejours_cible_exacerbation_&annee_N. (where = (reperage = 16 and variable = "ASS_DGN"));
	BPCO_SMED_P_EA_Sej_Index = id_sejour;
run;

proc sort data = T_INDI_BPCO_SMED_P_EA_&an_N. out = reperage_BPCO nodupkey;
	by BPCO_SMED_P_EA_Sej_Index UM_ORD_NUM;
run;

data reperage_BPCO2;
	set reperage_BPCO;
	by BPCO_SMED_P_EA_Sej_Index UM_ORD_NUM;
	if first.BPCO_SMED_P_EA_Sej_Index then
		output;
run;

proc sql undo_policy = none;

	CREATE TABLE res.T_INDI_BPCO_SMED_P_EA_&an_N. AS
		SELECT DISTINCT
			BEN_IDT_ANO,
			date_debut,
			date_fin AS BPCO_SMED_P_EA_Date_index length = 4,
			BPCO_SMED_P_EA_Sej_Index,
			ETA_NUM AS Finess_PMSI,
			ETA_NUM_GEO AS Finess_GEO,
			DGN_PAL AS Dp,
			CASE	WHEN SUBSTR(DGN_PAL, 1, 4) IN ("J440", "J441", "J448", "J449") THEN 1
				WHEN SUBSTR(DGN_PAL, 1, 3) IN ("J12", "J13", "J14", "J15", "J16", "J17", "J18")
OR SUBSTR(DGN_PAL, 1, 4) = "J181" THEN 2
				WHEN SUBSTR(DGN_PAL, 1, 4) IN ("J960") THEN 3
				WHEN SUBSTR(DGN_PAL, 1, 3) IN ("J09", "J10", "J11") THEN 4
				WHEN SUBSTR(DGN_PAL, 1, 3) IN ("I26") THEN 5
				WHEN SUBSTR(DGN_PAL, 1, 3) IN ("I50") OR SUBSTR(DGN_PAL, 1, 4) IN ("I130",
"I132", "I110", "I501") THEN 6
				WHEN SUBSTR(DGN_PAL, 1, 3) IN ("J86", "J93") OR SUBSTR(DGN_PAL, 1, 4) IN
("J942") THEN 7
				END AS Dp_classe length = 3,
			CASE	WHEN BPCO_SMED_P_EA_Sej_Index IN (SELECT BPCO_SMED_P_EA_Sej_Index FROM
BPCO_DAS) THEN 1
				ELSE 0
				END AS DA_BPCO length = 3,
			GRG_GHM AS GHM,
			1 AS BPCO_SMED_P_EA_CIBLE length = 3
```

```
                        FROM reperage_BPCO2
                        WHERE UM_ORD_NUM = 1
                        ORDER BY BEN_IDT_ANO;

quit;

proc delete data = BPCO_DAS;
run; quit;

*
        ********************************************************************************************************************
**********************************;
*       On exclut les séjours dont le mode de sortie est le décès. ;

proc sql;

        * On insère l effectif dans le Flowchart;
        INSERT INTO flowch.Flow_Chart_BPCO_SMED_P_EA_&an_N.
                SELECT
                        13,
                        COUNT(DISTINCT BPCO_SMED_P_EA_Sej_Index)
        FROM res.T_INDI_BPCO_SMED_P_EA_&an_N.
        WHERE BPCO_SMED_P_EA_Sej_Index IN (SELECT BPCO_SMED_P_EA_Sej_Index FROM T_INDI_BPCO_SMED_P_EA_&an_N.
WHERE SOR_MOD = "9");

        * On conserve les id des séjours à supprimer;
        CREATE TABLE travail.exclus1 AS
                SELECT DISTINCT
                        BPCO_SMED_P_EA_Sej_Index
        FROM res.T_INDI_BPCO_SMED_P_EA_&an_N.
        WHERE BPCO_SMED_P_EA_Sej_Index IN (SELECT BPCO_SMED_P_EA_Sej_Index FROM T_INDI_BPCO_SMED_P_EA_&an_N.
WHERE SOR_MOD = "9");

quit;

proc delete data = T_INDI_BPCO_SMED_P_EA_&an_N.;
run; quit;

*
        ********************************************************************************************************************
**********************************;
*       Changement juillet 2021 - Nouvelle version des indicateurs : on modifie les exlcusions;
*       On exclut les séjours index suivi d au moins une journée d hospitalisation en MCO, SSR, HAD ou PSY dans les 7 jours suivant
la date index;

data sejours_&Annee_N._&Annee_N1.;
        set travail.sejours_&Annee_N._&Annee_N1.;
run;

proc sql;

        DELETE FROM sejours_&Annee_N._&Annee_N1.
        WHERE domaine = "MCO" AND (SUBSTR(GRG_GHM, 1, 2) = "28" OR DUREE_SEJOUR = 0);

        DELETE FROM sejours_&Annee_N._&Annee_N1.
        WHERE domaine ne "MCO" AND date_debut = date_fin;

quit;

proc sql;

        CREATE TABLE sejours_7jours AS
                SELECT DISTINCT
                        a.BPCO_SMED_P_EA_Sej_Index
                FROM res.T_INDI_BPCO_SMED_P_EA_&an_N. a
                        INNER JOIN sejours_&Annee_N._&Annee_N1. b
                                ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
                WHERE b.date_debut < (a.BPCO_SMED_P_EA_Date_index + 7) AND a.BPCO_SMED_P_EA_Date_index <
b.date_fin
                        AND a.BPCO_SMED_P_EA_Sej_Index NE b.id_sejour;
```

```
	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_SMED_P_EA_&an_N.
		SELECT
			14,
			COUNT(DISTINCT BPCO_SMED_P_EA_Sej_Index)
	FROM res.T_INDI_BPCO_SMED_P_EA_&an_N.
	WHERE BPCO_SMED_P_EA_Sej_Index IN (SELECT BPCO_SMED_P_EA_Sej_Index FROM sejours_7jours);

	* On conserve les id des séjours à supprimer;
	CREATE TABLE travail.exclus2 AS
		SELECT DISTINCT
			BPCO_SMED_P_EA_Sej_Index
	FROM res.T_INDI_BPCO_SMED_P_EA_&an_N.
	WHERE BPCO_SMED_P_EA_Sej_Index IN (SELECT BPCO_SMED_P_EA_Sej_Index FROM sejours_7jours);

quit;

*
	**********************************************************************************************************
**********************************;
*	On exclut tous les patients;

data exclus;
	set	travail.exclus1-travail.exclus2;
run;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_SMED_P_EA_&an_N.
		SELECT
			15,
			COUNT(DISTINCT BPCO_SMED_P_EA_Sej_Index)
	FROM exclus;

	DELETE FROM res.T_INDI_BPCO_SMED_P_EA_&an_N.
	WHERE BPCO_SMED_P_EA_Sej_Index IN (SELECT BPCO_SMED_P_EA_Sej_Index FROM exclus);

quit;

proc sort data = res.T_INDI_BPCO_SMED_P_EA_&an_N. nodupkey;
	by _all_;
run; quit;

proc delete data = exclus sejours_&Annee_N._&Annee_N1.;
run; quit;

proc datasets library = travail memtype = data nolist;
	delete exclus1-exclus2;
run; quit;

*
	**********************************************************************************************************
**********************************;
*	Nombre de séjours inclus;

proc sql;

	INSERT INTO flowch.Flow_Chart_BPCO_SMED_P_EA_&an_N.
		SELECT
			17,
			COUNT(DISTINCT BPCO_SMED_P_EA_Sej_Index)
	FROM res.T_INDI_BPCO_SMED_P_EA_&an_N.;

quit;

proc sql;

	SELECT COUNT(*), COUNT(DISTINCT BPCO_SMED_P_EA_Sej_Index)
	FROM res.T_INDI_BPCO_SMED_P_EA_&an_N.;
```

```
quit;
```

3.9.3 02_Contact_MG_MT.sas

```
/*
	*****************************************************************************************************************************************	*/
/*
*/
/*	Indicateur 04 : Contact avec un médecin généraliste ou traitant
	*/
/*
*/
/*
	*****************************************************************************************************************************************	*/

*
	*****************************************************************************************************************************************;
*	Contact avec le médecin généraliste ou le médecin traitant;
*	On joint avec la table res.T_INDI_BPCO_SMED_P_EA_&an_N. pour récupérer les info dans les 7 et 180 jours suivant la date index;

%macro contact_MG_MT(nb_jours=);

	proc sql;

		CREATE TABLE contact_MG_PMSI_&nb_jours.j AS
			SELECT DISTINCT
				a.BPCO_SMED_P_EA_Sej_Index,
				a.BPCO_SMED_P_EA_Date_index,
				b.date_debut
			FROM res.T_INDI_BPCO_SMED_P_EA_&an_N. a
				INNER JOIN travail.contact_MG_PMSI_&Annee_N._&Annee_N1. b
					ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
			WHERE a.BPCO_SMED_P_EA_Date_index <= b.date_debut <= (a.BPCO_SMED_P_EA_Date_index + &nb_jours.)
			ORDER BY BPCO_SMED_P_EA_Sej_Index;

		CREATE TABLE contact_MT_&nb_jours.j AS
			SELECT DISTINCT
				a.BPCO_SMED_P_EA_Sej_Index,
				a.BPCO_SMED_P_EA_Date_index,
				b.date_debut
			FROM res.T_INDI_BPCO_SMED_P_EA_&an_N. a
				INNER JOIN travail.contact_MT_&Annee_N._&Annee_N1. b
					ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
			WHERE a.BPCO_SMED_P_EA_Date_index <= b.date_debut <= (a.BPCO_SMED_P_EA_Date_index + &nb_jours.)
			ORDER BY BPCO_SMED_P_EA_Sej_Index;

	quit;

	data contact_MG_MT_&nb_jours.j;
		set	contact_MG_PMSI_&nb_jours.j contact_MT_&nb_jours.j;
	run;

	proc sql undo_policy = none;

		CREATE TABLE contact_MG_MT_&nb_jours.j AS
			SELECT DISTINCT
				BPCO_SMED_P_EA_Sej_Index,
				BPCO_SMED_P_EA_Date_index,
				1 AS Contact_Med_&nb_jours.j length = 3,
				COUNT(DISTINCT date_debut) AS Nb_Contact_Med_&nb_jours.j length = 3,
				MIN(date_debut) AS Date_Contact_Med_&nb_jours.j length = 4
			FROM contact_MG_MT_&nb_jours.j
```

```
                                GROUP BY BPCO_SMED_P_EA_Sej_Index, BPCO_SMED_P_EA_Date_index;

        quit;

%mend contact_MG_MT;

%contact_MG_MT(nb_jours = 7);
%contact_MG_MT(nb_jours = 180);

*       On ajoute les informations dans la table de résultats;
proc sort data = res.T_INDI_BPCO_SMED_P_EA_&an_N.;
        by BPCO_SMED_P_EA_Sej_Index;
run;

* Dans les 7 jours;
proc sort data = contact_MG_MT_7j;
        by BPCO_SMED_P_EA_Sej_Index;
run;

data res.T_INDI_BPCO_SMED_P_EA_&an_N.;
        merge   res.T_INDI_BPCO_SMED_P_EA_&an_N. (in = a)
                        contact_MG_MT_7j (in = b keep = BPCO_SMED_P_EA_Sej_Index Contact_Med_7j
Nb_Contact_Med_7j);
        by BPCO_SMED_P_EA_Sej_Index;
        if Contact_Med_7j ne 1 then
                Contact_Med_7j = 0;
        if Nb_Contact_Med_7j = . then
                Nb_Contact_Med_7j = 0;
        if a then
                output;
run;

* Dans les 180 jours;
proc sort data = contact_MG_MT_180j;
        by BPCO_SMED_P_EA_Sej_Index;
run;

data res.T_INDI_BPCO_SMED_P_EA_&an_N.;
        merge   res.T_INDI_BPCO_SMED_P_EA_&an_N. (in = a)
                        contact_MG_MT_180j (in = b keep = BPCO_SMED_P_EA_Sej_Index Date_Contact_Med_180j);
        by BPCO_SMED_P_EA_Sej_Index;
        if a then
                output;
run;

proc datasets library = work memtype = data nolist;
        delete contact_:;
run; quit;
```

3.9.4 03_Contact_pneumologue.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                */
/*			Indicateur 04 : Contact avec un pneumologue
                                                                */
/*
                                                                                                                */
/*
	*****************************************************************************************************************
**********************************	*/

*
	*****************************************************************************************************************
**********************************;
*	Contact avec le pneumologue;
*	On joint avec la table res.T_INDI_BPCO_SMED_P_EA_&an_N. pour récupérer les info dans les 7 et 180 jours suivant la date
index;

%macro contact_pneumo(nb_jours=);

	proc sql;

		CREATE TABLE contact_pneumo_&nb_jours.j AS
			SELECT DISTINCT
				a.BPCO_SMED_P_EA_Sej_Index,
				1 AS Contact_pneumo_&nb_jours.j length = 3,
				COUNT(DISTINCT b.date_debut) AS Nb_Contact_pneumo_&nb_jours.j length = 3,
				MIN(b.date_debut) AS Date_Contact_pneumo_&nb_jours.j length = 4
			FROM res.T_INDI_BPCO_SMED_P_EA_&an_N. a
				INNER JOIN travail.contact_pneumo_&Annee_N._&Annee_N1. b
					ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
			WHERE b.type ne "PMSI séjours" AND a.BPCO_SMED_P_EA_Date_index <= b.date_debut <=
(a.BPCO_SMED_P_EA_Date_index + &nb_jours.)
			GROUP BY BPCO_SMED_P_EA_Sej_Index
			ORDER BY BPCO_SMED_P_EA_Sej_Index;

	quit;

%mend contact_pneumo;

%contact_pneumo(nb_jours = 7);
%contact_pneumo(nb_jours = 180);

*	On ajoute les informations dans la table de résultats;
proc sort data = res.T_INDI_BPCO_SMED_P_EA_&an_N.;
	by BPCO_SMED_P_EA_Sej_Index;
run;

* Dans les 7 jours;
proc sort data = contact_pneumo_7j;
	by BPCO_SMED_P_EA_Sej_Index;
run;

data res.T_INDI_BPCO_SMED_P_EA_&an_N.;
	merge	res.T_INDI_BPCO_SMED_P_EA_&an_N. (in = a)
		contact_pneumo_7j (in = b keep = BPCO_SMED_P_EA_Sej_Index Contact_pneumo_7j
Nb_Contact_pneumo_7j);
	by BPCO_SMED_P_EA_Sej_Index;
	if Contact_pneumo_7j ne 1 then
		Contact_pneumo_7j = 0;
	if Nb_Contact_pneumo_7j = . then
		Nb_Contact_pneumo_7j = 0;
```

```
		if a then
			output;
run;

* Dans les 180 jours;
proc sort data = contact_pneumo_180j;
		by BPCO_SMED_P_EA_Sej_Index;
run;

data res.T_INDI_BPCO_SMED_P_EA_&an_N.;
		merge	res.T_INDI_BPCO_SMED_P_EA_&an_N. (in = a)
				contact_pneumo_180j (in = b keep = BPCO_SMED_P_EA_Sej_Index Date_Contact_pneumo_180j);
		by BPCO_SMED_P_EA_Sej_Index;
		if a then
			output;
run;

proc datasets library = work memtype = data nolist;
		delete contact_:;
run; quit;
```

3.9.5 04_Sejours_exacerbation_de_BPCO_en_MCO.sas

```
/*
	*****************************************************************************************************
**********************************	*/
/*

                                                                                                        */
/*		Indicateur 04 - Nombre de séjours l année N

                                                        */
/*

                                                                                                        */
/*
	*****************************************************************************************************
**********************************	*/

*
	*****************************************************************************************************
**********************************;
*	Nombre de séjours MCO d'exacerbation de BPCO codé en DP ou en DAS terminés l année N;

proc sql;

	CREATE TABLE nb_sejours_exa_BPCO_DP AS
		SELECT
			BEN_IDT_ANO,
			COUNT(DISTINCT id_sejour) AS Nb_Sej_EA_BPCO_DP_MCO length = 3
		FROM travail.sejours_exacerbation_&annee_N.
		WHERE reperage = 16 AND variable = "DGN_PAL"
		GROUP BY BEN_IDT_ANO;

	CREATE TABLE nb_sejours_exa_BPCO_DAS AS
		SELECT
			BEN_IDT_ANO,
			COUNT(DISTINCT id_sejour) AS Nb_Sej_EA_BPCO_DAS_MCO length = 3
		FROM travail.sejours_exacerbation_&annee_N.
		WHERE reperage = 16 AND variable = "ASS_DGN"
		GROUP BY BEN_IDT_ANO;

quit;

*
	*****************************************************************************************************
**********************************;
*	On ajoute l'information dans la table res.T_INDI_BPCO_SMED_P_EA_&an_N.;

proc sort data = res.T_INDI_BPCO_SMED_P_EA_&an_N.;
	by BEN_IDT_ANO;
run;

data res.T_INDI_BPCO_SMED_P_EA_&an_N.;
	merge	res.T_INDI_BPCO_SMED_P_EA_&an_N. (in = a)
			nb_sejours_exa_BPCO_DP (in = b)
			nb_sejours_exa_BPCO_DAS (in = c);
	by BEN_IDT_ANO;
	if a;
	if not b then
		Nb_Sej_EA_BPCO_DP_MCO = 0;
	if not c then
		Nb_Sej_EA_BPCO_DAS_MCO = 0;
run;
```

3.9.6 05_Table_finale.sas

```
/*
	*************************************************************************************************************
**********************************	*/
/*
                                                                                                                */
/*		Indicateur 04 : Table finale - Ajout de formats et de labels

		*/
/*
                                                                                                                */
/*
	*************************************************************************************************************
**********************************	*/

data res.T_INDI_BPCO_SMED_P_EA_&an_N.;
	set res.T_INDI_BPCO_SMED_P_EA_&an_N. (keep = BEN_IDT_ANO BPCO_SMED_P_EA_Date_index
BPCO_SMED_P_EA_Sej_Index Finess_PMSI Finess_GEO Dp Dp_classe
		DA_BPCO GHM BPCO_SMED_P_EA_CIBLE Nb_Sej_EA_BPCO_DP_MCO Nb_Sej_EA_BPCO_DAS_MCO
Contact_Med_7j Nb_Contact_Med_7j
		Date_Contact_Med_180j Contact_Pneumo_7j Nb_Contact_Pneumo_7j Date_Contact_Pneumo_180j);
	length BPCO_SMED_P_EA_OBS 3.;
	BPCO_SMED_P_EA_OBS = 0;
	if Contact_Med_7j = 1 or Contact_Pneumo_7j = 1 then
		BPCO_SMED_P_EA_OBS = 1;
	label
		BEN_IDT_ANO = "Numéro d'individu"
		BPCO_SMED_P_EA_Date_index = "Date index"
		BPCO_SMED_P_EA_Sej_Index = "Identifiant du séjour index"
		Finess_PMSI = "Finess PMSI ayant réalisé le séjour index"
		Finess_GEO = "Finess géographique du premier RUM"
		Dp = "Diagnostic principal du séjour index "
		Dp_classe = "Diagnostic principal du séjour index"
		DA_BPCO = "BPCO codé en diagnostic secondaire du séjour index"
		GHM = "GHM du séjour index"
		BPCO_SMED_P_EA_CIBLE = "Séjour correspondant aux critères d'inclusion et d'exclusion de la population cible"
		Nb_Sej_EA_BPCO_DP_MCO = "Nombre de séjours MCO d'exacerbation de BPCO codé en DP terminés en
&annee_N."
		Nb_Sej_EA_BPCO_DAS_MCO = "Nombre de séjours MCO d'exacerbation de BPCO codé en DAS terminés en
&annee_N."
		BPCO_SMED_P_EA_OBS = "Séjour de la population cible pour lequel le patient a bénéficié d'un suivi dans les 7
jours après la sortie de l'hôpital"
		Contact_Med_7j = "Séjour de la population cible pour lequel le patient a eu un contact médecin traitant ou MG
dans les 7 jours suivant la date index"
		Nb_Contact_Med_7j = "Nombre de contacts médecin traitant ou MG dans les 7 jours suivant la date index"
		Date_Contact_Med_180j = "Date du premier contact médecin traitant ou MG dans les 180 jours suivant la date
index"
		Contact_Pneumo_7j = "Séjour de la population cible pour lequel le patient a eu un contact pneumologue dans les
7 jours suivant la date index"
		Nb_Contact_Pneumo_7j = "Nombre de contacts pneumologue dans les 7 jours suivant la date index"
		Date_Contact_Pneumo_180j = "Date du premier contact pneumologue dans les 180 jours suivant la date index"
		;
	format DA_BPCO BPCO_SMED_P_EA_CIBLE BPCO_SMED_P_EA_OBS Contact_Med_7j Contact_Pneumo_7j f_oui_non.
Dp_Classe f_DP_classe. Date_Contact_Med_180j
		Date_Contact_Pneumo_180j BPCO_SMED_P_EA_Date_index date9.;
run;

*
	*************************************************************************************************************
**********************************;
*	Nombre de contacts dans les 7 jours après l hospitalisation;

proc sql;

	INSERT INTO flowch.Flow_Chart_BPCO_SMED_P_EA_&an_N.
		SELECT
```

```
                                18,
                                COUNT(DISTINCT BPCO_SMED_P_EA_Sej_Index)
        FROM res.T_INDI_BPCO_SMED_P_EA_&an_N.
        WHERE Contact_Pneumo_7j = 1;

        INSERT INTO flowch.Flow_Chart_BPCO_SMED_P_EA_&an_N.
                SELECT
                                19,
                                COUNT(DISTINCT BPCO_SMED_P_EA_Sej_Index)
        FROM res.T_INDI_BPCO_SMED_P_EA_&an_N.
        WHERE Contact_Med_7j = 1;

        INSERT INTO flowch.Flow_Chart_BPCO_SMED_P_EA_&an_N.
                SELECT
                                20,
                                COUNT(DISTINCT BPCO_SMED_P_EA_Sej_Index)
        FROM res.T_INDI_BPCO_SMED_P_EA_&an_N.
        WHERE BPCO_SMED_P_EA_OBS = 1;

        * Population d étude;
        SELECT N into : N12 FROM flowch.Flow_Chart_BPCO_SMED_P_EA_&an_N. WHERE indicateur = 12;
        * Population cible;
        SELECT N into : N17 FROM flowch.Flow_Chart_BPCO_SMED_P_EA_&an_N. WHERE indicateur = 17;

quit;

data flowch.Flow_Chart_BPCO_SMED_P_EA_&an_N.;
        set flowch.Flow_Chart_BPCO_SMED_P_EA_&an_N.;
        if indicateur in (13, 14, 15, 17) then
                percent = N / &N12.;
        if indicateur in (18, 19, 20) then
                percent = N / &N17.;
        format indicateur f_ind_04_flowchart. percent percent7.1;
run;

* Vérif;
proc sql;

        SELECT COUNT(*) AS nb_lignes, COUNT(DISTINCT BPCO_SMED_P_EA_Sej_Index) AS nb_sejours
        FROM res.T_INDI_BPCO_SMED_P_EA_&an_N.;
        * nb_lignes = nb_sejours : OK;

        SELECT COUNT(*) AS nb_lignes, COUNT(DISTINCT BPCO_SMED_P_EA_Sej_Index) AS nb_sejours
        FROM res.T_INDI_BPCO_SMED_P_EA_&an_N.
        WHERE BPCO_SMED_P_EA_OBS = 1;
        * nb_lignes = nb_sejours : OK;

quit;

*
        ************************************************************************************************
**********************************;
*       On ajoute le flag BPCO_SMED_P_EA_CIBLE dans la table res.T_INDI_BPCO_&an_N.;

proc sql;

        ALTER TABLE res.T_INDI_BPCO_&an_N. DROP BPCO_SMED_P_EA_CIBLE;

        ALTER TABLE res.T_INDI_BPCO_&an_N. ADD BPCO_SMED_P_EA_CIBLE INT format = f_oui_non. length = 3
                label = "Patient appartenant à la population cible BPCO_SMED_P_EA";

        UPDATE res.T_INDI_BPCO_&an_N.
                SET BPCO_SMED_P_EA_CIBLE =  1
                WHERE BEN_IDT_ANO IN (SELECT DISTINCT BEN_IDT_ANO FROM res.T_INDI_BPCO_SMED_P_EA_&an_N.);

quit;

data res.T_INDI_BPCO_&an_N.;
        set res.T_INDI_BPCO_&an_N.;
        if BPCO_SMED_P_EA_CIBLE = .
                then BPCO_SMED_P_EA_CIBLE = 0;
```

```
run;
```

3.10 Suivi par le pneumologue dans les 60 jours après hospitalisation pour exacerbation de BPCO des patients sortis à domicile

3.10.1 00_Initialisation_de_la_table_de_resultats.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*

*/
/*		Table de population - Patients avec indicateur_05 = 1

		*/
/*

*/
/*
	*****************************************************************************************************************
**********************************	*/

data T_INDI_BPCO_SMED_D_EA_&an_N. (keep = BEN_IDT_ANO);
	set pop.indicateurs_&an_N. (where = (indicateur_05 = 1));
run;

*
	*****************************************************************************************************************
**********************************;
*	Effectifs pour le flowchart;

* Séjours pour exacerbation : individus de 40 ans et plus ayant bénéficié de soins remboursés au moins une fois l année N;

proc sort data = T_INDI_BPCO_SMED_D_EA_&an_N. nodupkey;
	by BEN_IDT_ANO;
run;

proc sort data = travail.sejours_cible_exacerbation_&annee_N.;
	by BEN_IDT_ANO;
run;

data T_INDI_BPCO_SMED_D_EA_&an_N.;
	merge	T_INDI_BPCO_SMED_D_EA_&an_N. (in = a)
		travail.sejours_cible_exacerbation_&annee_N. (in = b);
	by BEN_IDT_ANO;
	if a and b;
	BPCO_SMED_D_EA_Sej_Index = id_sejour;
run;

proc sql;

	CREATE TABLE flowch.Flow_Chart_BPCO_SMED_D_EA_&an_N.  AS
		SELECT *
		FROM (
			SELECT 1 AS indicateur length = 3, COUNT(DISTINCT BPCO_SMED_D_EA_Sej_Index) AS N length = 5
FROM T_INDI_BPCO_SMED_D_EA_&an_N. WHERE exacerbation_BPCO = 1
			UNION ALL
			SELECT 2, COUNT(DISTINCT BPCO_SMED_D_EA_Sej_Index) FROM T_INDI_BPCO_SMED_D_EA_&an_N.
WHERE exacerbation_BPCO = 2
			UNION ALL
			SELECT 3, COUNT(DISTINCT BPCO_SMED_D_EA_Sej_Index) FROM T_INDI_BPCO_SMED_D_EA_&an_N.
WHERE exacerbation_BPCO = 3
			UNION ALL
			SELECT 4, COUNT(DISTINCT BPCO_SMED_D_EA_Sej_Index) FROM T_INDI_BPCO_SMED_D_EA_&an_N.
WHERE exacerbation_BPCO = 4
			UNION ALL
```

```
                                SELECT 5, COUNT(DISTINCT BPCO_SMED_D_EA_Sej_Index) FROM T_INDI_BPCO_SMED_D_EA_&an_N.
WHERE exacerbation_BPCO = 5
                                UNION ALL
                                SELECT 6, COUNT(DISTINCT BPCO_SMED_D_EA_Sej_Index) FROM T_INDI_BPCO_SMED_D_EA_&an_N.
WHERE exacerbation_BPCO = 6
                                UNION ALL
                                SELECT 7, COUNT(DISTINCT BPCO_SMED_D_EA_Sej_Index) FROM T_INDI_BPCO_SMED_D_EA_&an_N.
WHERE exacerbation_BPCO = 7
                                UNION ALL
                                SELECT 8, COUNT(DISTINCT BPCO_SMED_D_EA_Sej_Index) FROM T_INDI_BPCO_SMED_D_EA_&an_N.
WHERE exacerbation_BPCO = 8
                                UNION ALL
                                SELECT 9, COUNT(DISTINCT BPCO_SMED_D_EA_Sej_Index) FROM T_INDI_BPCO_SMED_D_EA_&an_N.
WHERE exacerbation_BPCO = 9
                                UNION ALL
                                SELECT 10, COUNT(DISTINCT BPCO_SMED_D_EA_Sej_Index) FROM
T_INDI_BPCO_SMED_D_EA_&an_N. WHERE exacerbation_BPCO = 10
                                UNION ALL
                                SELECT 11, COUNT(DISTINCT BPCO_SMED_D_EA_Sej_Index) FROM
T_INDI_BPCO_SMED_D_EA_&an_N. WHERE exacerbation_BPCO = 11
                                UNION ALL
                                SELECT 12, COUNT(DISTINCT BPCO_SMED_D_EA_Sej_Index) FROM
T_INDI_BPCO_SMED_D_EA_&an_N.
                        ) ;

quit;
```

3.10.2 01_Inclusions_et_exclusions.sas

```
/*
	*****************************************************************************************************************************
**********************************	*/
/*
                                                                                                                                */
/*			Indicateur 05 : Inclusion & exclusions
                                                                                                */
/*
                                                                                                                                */
/*
	*****************************************************************************************************************************
**********************************	*/

*
	*****************************************************************************************************************************
**********************************;
*	On récupère tous les séjours pour exacerbation de BPCO pour les patients de la table
res.T_INDI_BPCO_SMED_D_EA_&an_N.;

* Séjours avec BPCO en DAS;
data BPCO_DAS;
	set travail.sejours_cible_exacerbation_&annee_N. (where = (reperage = 16 and variable = "ASS_DGN"));
	BPCO_SMED_D_EA_Sej_Index = id_sejour;
run;

proc sort data = T_INDI_BPCO_SMED_D_EA_&an_N. out = reperage_BPCO nodupkey;
	by BPCO_SMED_D_EA_Sej_Index UM_ORD_NUM;
run;

data reperage_BPCO2;
	set reperage_BPCO;
	by BPCO_SMED_D_EA_Sej_Index UM_ORD_NUM;
	if first.BPCO_SMED_D_EA_Sej_Index then
		output;
run;

proc sql undo_policy = none;

	CREATE TABLE res.T_INDI_BPCO_SMED_D_EA_&an_N. AS
		SELECT DISTINCT
			BEN_IDT_ANO,
			date_debut,
			date_fin AS BPCO_SMED_D_EA_Date_index length = 4,
			BPCO_SMED_D_EA_Sej_Index,
			ETA_NUM AS Finess_PMSI,
			ETA_NUM_GEO AS Finess_GEO,
			DGN_PAL AS Dp,
			CASE	WHEN SUBSTR(DGN_PAL, 1, 4) IN ("J440", "J441", "J448", "J449") THEN 1
				WHEN SUBSTR(DGN_PAL, 1, 3) IN ("J12", "J13", "J14", "J15", "J16", "J17", "J18")
OR SUBSTR(DGN_PAL, 1, 4) = "J181" THEN 2
				WHEN SUBSTR(DGN_PAL, 1, 4) IN ("J960") THEN 3
				WHEN SUBSTR(DGN_PAL, 1, 3) IN ("J09", "J10", "J11") THEN 4
				WHEN SUBSTR(DGN_PAL, 1, 3) IN ("I26") THEN 5
				WHEN SUBSTR(DGN_PAL, 1, 3) IN ("I50") OR SUBSTR(DGN_PAL, 1, 4) IN ("I130",
"I132", "I110", "I501") THEN 6
				WHEN SUBSTR(DGN_PAL, 1, 3) IN ("J86", "J93") OR SUBSTR(DGN_PAL, 1, 4) IN
("J942") THEN 7
				END AS Dp_classe length = 3,
			CASE	WHEN BPCO_SMED_D_EA_Sej_Index IN (SELECT BPCO_SMED_D_EA_Sej_Index FROM
BPCO_DAS) THEN 1
				ELSE 0
				END AS DA_BPCO length = 3,
			GRG_GHM AS GHM,
			1 AS BPCO_SMED_D_EA_CIBLE length = 3
```

```
                        FROM reperage_BPCO2
                        WHERE UM_ORD_NUM = 1
                        ORDER BY BEN_IDT_ANO;

quit;

proc delete data = BPCO_DAS;
run; quit;

*
        ********************************************************************************************************************
**********************************;
*       On exclut les séjours dont le mode de sortie est le décès. ;

proc sql;

        * On insère l effectif dans le Flowchart;
        INSERT INTO flowch.Flow_Chart_BPCO_SMED_D_EA_&an_N.
                SELECT
                        13,
                        COUNT(DISTINCT BPCO_SMED_D_EA_Sej_Index)
        FROM res.T_INDI_BPCO_SMED_D_EA_&an_N.
        WHERE BPCO_SMED_D_EA_Sej_Index IN (SELECT BPCO_SMED_D_EA_Sej_Index FROM T_INDI_BPCO_SMED_D_EA_&an_N.
WHERE SOR_MOD = "9");

        * On conserve les id des séjours à supprimer;
        CREATE TABLE travail.exclus1 AS
                SELECT DISTINCT
                        BPCO_SMED_D_EA_Sej_Index
        FROM res.T_INDI_BPCO_SMED_D_EA_&an_N.
        WHERE BPCO_SMED_D_EA_Sej_Index IN (SELECT BPCO_SMED_D_EA_Sej_Index FROM T_INDI_BPCO_SMED_D_EA_&an_N.
WHERE SOR_MOD = "9");

quit;

proc delete data = T_INDI_BPCO_SMED_D_EA_&an_N.;
run; quit;


*
        ********************************************************************************************************************
**********************************;
*       Changement juillet 2021 - Nouvelle version des indicateurs : on modifie les exlcusions;
*       On exclut les séjours index suivi d au moins une journée d hospitalisation en MCO, SSR, HAD ou PSY dans les 60 jours suivant
la date index;

data sejours_&Annee_N._&Annee_N1.;
        set travail.sejours_&Annee_N._&Annee_N1.;
run;

proc sql;

        DELETE FROM sejours_&Annee_N._&Annee_N1.
        WHERE domaine = "MCO" AND (SUBSTR(GRG_GHM, 1, 2) = "28" OR DUREE_SEJOUR = 0);

        DELETE FROM sejours_&Annee_N._&Annee_N1.
        WHERE domaine ne "MCO" AND date_debut = date_fin;

quit;

proc sql;

        CREATE TABLE sejours_60jours AS
                SELECT DISTINCT
                        a.BPCO_SMED_D_EA_Sej_Index
                FROM res.T_INDI_BPCO_SMED_D_EA_&an_N. a
                        INNER JOIN sejours_&Annee_N._&Annee_N1. b
                                ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
                WHERE b.date_debut < (a.BPCO_SMED_D_EA_Date_index + 60) AND a.BPCO_SMED_D_EA_Date_index <
b.date_fin
                        AND a.BPCO_SMED_D_EA_Sej_Index NE b.id_sejour;
```

```
	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_SMED_D_EA_&an_N.
		SELECT
			14,
			COUNT(DISTINCT BPCO_SMED_D_EA_Sej_Index)
	FROM res.T_INDI_BPCO_SMED_D_EA_&an_N.
	WHERE BPCO_SMED_D_EA_Sej_Index IN (SELECT BPCO_SMED_D_EA_Sej_Index FROM sejours_60jours);

	* On conserve les id des séjours à supprimer;
	CREATE TABLE travail.exclus2 AS
		SELECT DISTINCT
			BPCO_SMED_D_EA_Sej_Index
	FROM res.T_INDI_BPCO_SMED_D_EA_&an_N.
	WHERE BPCO_SMED_D_EA_Sej_Index IN (SELECT BPCO_SMED_D_EA_Sej_Index FROM sejours_60jours);

quit;

*
	*************************************************************************************************************
**********************************;
*	On exclut tous les patients;

data exclus;
	set	travail.exclus1-travail.exclus2;
run;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_SMED_D_EA_&an_N.
		SELECT
			15,
			COUNT(DISTINCT BPCO_SMED_D_EA_Sej_Index)
	FROM exclus;

	DELETE FROM res.T_INDI_BPCO_SMED_D_EA_&an_N.
	WHERE BPCO_SMED_D_EA_Sej_Index IN (SELECT BPCO_SMED_D_EA_Sej_Index FROM exclus);

quit;

proc sort data = res.T_INDI_BPCO_SMED_D_EA_&an_N. nodupkey;
	by _all_;
run; quit;

proc delete data = exclus sejours_&Annee_N._&Annee_N1.;
run; quit;

proc datasets library = travail memtype = data nolist;
	delete exclus1-exclus2;
run; quit;

*
	*************************************************************************************************************
**********************************;
*	Nombre de séjours inclus;

proc sql;

	INSERT INTO flowch.Flow_Chart_BPCO_SMED_D_EA_&an_N.
		SELECT
			17,
			COUNT(DISTINCT BPCO_SMED_D_EA_Sej_Index)
	FROM res.T_INDI_BPCO_SMED_D_EA_&an_N.;

quit;

proc sql;

	SELECT COUNT(*), COUNT(DISTINCT BPCO_SMED_D_EA_Sej_Index)
	FROM res.T_INDI_BPCO_SMED_D_EA_&an_N.;
```

```
quit;
```

3.10.3 02_Contact_pneumologue.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*

*/
/*			Indicateur 05 : Contact avec un pneumologue

*/
/*

*/
/*
	*****************************************************************************************************************
**********************************	*/

*
	*****************************************************************************************************************
**********************************;
*	Contact avec le pneumologue;
*	On joint avec la table res.T_INDI_BPCO_SMED_D_EA_&an_N. pour récupérer les info dans les 60 et 180 jours suivant la date
index;

%macro contact_pneumo(nb_jours=);

	proc sql;

		CREATE TABLE contact_pneumo_&nb_jours.j AS
			SELECT DISTINCT
				a.BPCO_SMED_D_EA_Sej_Index,
				1 AS BPCO_SMED_D_EA_OBS length = 3,
				COUNT(DISTINCT b.date_debut) AS Nb_Contact_pneumo_&nb_jours.j length = 3,
				MIN(b.date_debut) AS Date_Contact_pneumo_&nb_jours.j format ddmmyy10. length = 4
			FROM res.T_INDI_BPCO_SMED_D_EA_&an_N. a
				INNER JOIN travail.contact_pneumo_&Annee_N._&Annee_N1. b
					ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
			WHERE a.BPCO_SMED_D_EA_Date_index <= b.date_debut <= (a.BPCO_SMED_D_EA_Date_index +
&nb_jours.)
			GROUP BY BPCO_SMED_D_EA_Sej_Index
			ORDER BY BPCO_SMED_D_EA_Sej_Index;

	quit;

%mend contact_pneumo;

%contact_pneumo(nb_jours = 60);
%contact_pneumo(nb_jours = 180);

*	On ajoute les informations dans la table de résultats;
proc sort data = res.T_INDI_BPCO_SMED_D_EA_&an_N.;
	by BPCO_SMED_D_EA_Sej_Index;
run;

* Dans les 60 jours;
proc sort data = contact_pneumo_60j;
	by BPCO_SMED_D_EA_Sej_Index;
run;

data res.T_INDI_BPCO_SMED_D_EA_&an_N.;
	merge	res.T_INDI_BPCO_SMED_D_EA_&an_N. (in = a)
			contact_pneumo_60j (in = b keep = BPCO_SMED_D_EA_Sej_Index BPCO_SMED_D_EA_OBS
Nb_Contact_pneumo_60j);
	by BPCO_SMED_D_EA_Sej_Index;
	if BPCO_SMED_D_EA_OBS ne 1 then
		BPCO_SMED_D_EA_OBS = 0;
	if Nb_Contact_pneumo_60j = . then
		Nb_Contact_pneumo_60j = 0;
```

```
		if a then
			output;
run;

* Dans les 180 jours;
proc sort data = contact_pneumo_180j;
		by BPCO_SMED_D_EA_Sej_Index;
run;

data res.T_INDI_BPCO_SMED_D_EA_&an_N.;
		merge	res.T_INDI_BPCO_SMED_D_EA_&an_N. (in = a)
				contact_pneumo_180j (in = b keep = BPCO_SMED_D_EA_Sej_Index Date_Contact_pneumo_180j);
		by BPCO_SMED_D_EA_Sej_Index;
		if a then
			output;
run;

proc datasets library = work memtype = data nolist;
		delete contact_:;
run; quit;
```

3.10.4 03_Sejour_exacerbation_de_BPCO_en_MCO.sas

```
/*
	*****************************************************************************************************
**********************************	*/
/*
                                                                                                        */
/*		Indicateur 05 - Nombre de séjours l année N
                                                                */
/*
                                                                                                        */
/*
	*****************************************************************************************************
**********************************	*/

*
	*****************************************************************************************************
**********************************;
*	Nombre de séjours MCO d'exacerbation de BPCO codé en DP ou en DAS terminés l année N;

proc sql;

	CREATE TABLE nb_sejours_exa_BPCO_DP AS
		SELECT
			BEN_IDT_ANO,
			COUNT(DISTINCT id_sejour) AS Nb_Sej_EA_BPCO_DP_MCO length = 3
		FROM travail.sejours_exacerbation_&annee_N.
		WHERE reperage = 16 AND variable = "DGN_PAL"
		GROUP BY BEN_IDT_ANO;

	CREATE TABLE nb_sejours_exa_BPCO_DAS AS
		SELECT
			BEN_IDT_ANO,
			COUNT(DISTINCT id_sejour) AS Nb_Sej_EA_BPCO_DAS_MCO length = 3
		FROM travail.sejours_exacerbation_&annee_N.
		WHERE reperage = 16 AND variable = "ASS_DGN"
		GROUP BY BEN_IDT_ANO;

quit;

*
	*****************************************************************************************************
**********************************;
*	On ajoute l'information dans la table res.T_INDI_BPCO_SMED_D_EA_&an_N.;

proc sort data = res.T_INDI_BPCO_SMED_D_EA_&an_N.;
	by BEN_IDT_ANO;
run;

data res.T_INDI_BPCO_SMED_D_EA_&an_N.;
	merge	res.T_INDI_BPCO_SMED_D_EA_&an_N. (in = a)
			nb_sejours_exa_BPCO_DP (in = b)
			nb_sejours_exa_BPCO_DAS (in = c);
	by BEN_IDT_ANO;
	if a;
	if not b then
		Nb_Sej_EA_BPCO_DP_MCO = 0;
	if not c then
		Nb_Sej_EA_BPCO_DAS_MCO = 0;
run;
```

3.10.5 04_Table_finale.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*

*/
/*		Indicateur 05 : Table finale - Ajout de formats et de labels

		*/
/*

*/
/*
	*****************************************************************************************************************
**********************************	*/

data res.T_INDI_BPCO_SMED_D_EA_&an_N.;
	set res.T_INDI_BPCO_SMED_D_EA_&an_N. (keep = BEN_IDT_ANO BPCO_SMED_D_EA_Date_index
BPCO_SMED_D_EA_Sej_Index Finess_PMSI Finess_GEO Dp Dp_classe
		DA_BPCO GHM BPCO_SMED_D_EA_CIBLE Nb_Sej_EA_BPCO_DP_MCO Nb_Sej_EA_BPCO_DAS_MCO
BPCO_SMED_D_EA_OBS Nb_Contact_Pneumo_60j Date_Contact_Pneumo_180j);
	label
		BEN_IDT_ANO = "Numéro d'individu"
		BPCO_SMED_D_EA_Date_index = "Date index"
		BPCO_SMED_D_EA_Sej_Index = "Identifiant du séjour index"
		Finess_PMSI = "Finess PMSI ayant réalisé le séjour index"
		Finess_GEO = "Finess géographique du premier RUM"
		Dp = "Diagnostic principal du séjour index "
		Dp_classe = "Diagnostic principal du séjour index"
		DA_BPCO = "BPCO codé en diagnostic secondaire du séjour index"
		GHM = "GHM du séjour index"
		BPCO_SMED_D_EA_CIBLE = "Séjour correspondant aux critères d'inclusion et d'exclusion de la population cible"
		Nb_Sej_EA_BPCO_DP_MCO = "Nombre de séjours MCO d'exacerbation de BPCO codé en DP terminés en
&annee_N."
		Nb_Sej_EA_BPCO_DAS_MCO = "Nombre de séjours MCO d'exacerbation de BPCO codé en DAS terminés en
&annee_N."
		BPCO_SMED_D_EA_OBS = "Séjour de la population cible pour lequel le patient a bénéficié d'un suivi à distance
après la sortie de l'hôpital"
		Nb_Contact_Pneumo_60j = "Nombre de contacts pneumologue dans les 60 jours suivant la date index"
		Date_Contact_Pneumo_180j = "Date du premier contact pneumologue dans les 180 jours suivant la date index"
		;
	format DA_BPCO BPCO_SMED_D_EA_CIBLE BPCO_SMED_D_EA_OBS f_oui_non. Dp_Classe f_DP_classe.
BPCO_SMED_D_EA_Date_index Date_Contact_Pneumo_180j date9.;
run;

*
	*****************************************************************************************************************
**********************************;
*	Nombre de contacts dans les 60 jours après l hospitalisation;

proc sql;

	INSERT INTO flowch.Flow_Chart_BPCO_SMED_D_EA_&an_N.
		SELECT
			18,
			COUNT(DISTINCT BPCO_SMED_D_EA_Sej_Index)
	FROM res.T_INDI_BPCO_SMED_D_EA_&an_N.
	WHERE BPCO_SMED_D_EA_OBS = 1;

	* Population d étude;
	SELECT N into : N12 FROM flowch.Flow_Chart_BPCO_SMED_D_EA_&an_N. WHERE indicateur = 12;
	* Population cible;
	SELECT N into : N17 FROM flowch.Flow_Chart_BPCO_SMED_D_EA_&an_N. WHERE indicateur = 17;

quit;

data flowch.Flow_Chart_BPCO_SMED_D_EA_&an_N.;
```

```
		set flowch.Flow_Chart_BPCO_SMED_D_EA_&an_N.;
		if indicateur in (13, 14, 15, 17) then
			percent = N / &N12.;
		if indicateur = 18 then
			percent = N / &N17.;
		format indicateur f_ind_05_flowchart. percent percent7.1;
run;

* Vérif;
proc sql;

		SELECT COUNT(*) AS nb_lignes, COUNT(DISTINCT BPCO_SMED_D_EA_Sej_Index) AS nb_sejours
		FROM res.T_INDI_BPCO_SMED_D_EA_&an_N.;
		* nb_lignes = nb_sejours : OK;

		SELECT COUNT(*) AS nb_lignes, COUNT(DISTINCT BPCO_SMED_D_EA_Sej_Index) AS nb_sejours
		FROM res.T_INDI_BPCO_SMED_D_EA_&an_N.
		WHERE BPCO_SMED_D_EA_OBS = 1;
		* nb_lignes = nb_sejours : OK;

quit;

*
		**************************************************************************************************
***********************************;
*		On ajoute le flag BPCO_SMED_D_EA_CIBLE dans la table res.T_INDI_BPCO_&an_N.;

proc sql;

		ALTER TABLE res.T_INDI_BPCO_&an_N. DROP BPCO_SMED_D_EA_CIBLE;

		ALTER TABLE res.T_INDI_BPCO_&an_N. ADD BPCO_SMED_D_EA_CIBLE INT format = f_oui_non. length = 3
			label = "Patient appartenant à la population cible BPCO_SMED_D_EA";

		UPDATE res.T_INDI_BPCO_&an_N.
			SET BPCO_SMED_D_EA_CIBLE = 1
			WHERE BEN_IDT_ANO IN (SELECT DISTINCT BEN_IDT_ANO FROM res.T_INDI_BPCO_SMED_D_EA_&an_N.);

quit;

data res.T_INDI_BPCO_&an_N.;
		set res.T_INDI_BPCO_&an_N.;
		if BPCO_SMED_D_EA_CIBLE = .
			then BPCO_SMED_D_EA_CIBLE = 0;
run;
```

3.11 Traitement remboursé de BDLA délivré dans les 90 jours après une hospitalisation pour exacerbation de BPCO des patients sortis à domicile

3.11.1 00_Initialisation_de_la_table_de_resultats.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*		Table de population - Patients avec indicateur_06 = 1

			*/
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/

data T_INDI_BPCO_BDLA_EA_&an_N. (keep = BEN_IDT_ANO);
	set pop.indicateurs_&an_N. (where = (indicateur_06 = 1));
run;

*
	*****************************************************************************************************************
***********************************;
*	Effectifs pour le flowchart;

* Séjours pour exacerbation : individus de 40 ans et plus ayant bénéficié de soins remboursés au moins une fois l année N;

proc sort data = T_INDI_BPCO_BDLA_EA_&an_N. nodupkey;
	by BEN_IDT_ANO;
run;

proc sort data = travail.sejours_cible_exacerbation_&annee_N.;
	by BEN_IDT_ANO;
run;

data T_INDI_BPCO_BDLA_EA_&an_N.;
	merge	T_INDI_BPCO_BDLA_EA_&an_N. (in = a)
			travail.sejours_cible_exacerbation_&annee_N. (in = b);
	by BEN_IDT_ANO;
	if a and b;
	BPCO_BDLA_EA_Sej_Index = id_sejour;
run;

proc sql;

	CREATE TABLE flowch.Flow_Chart_BPCO_BDLA_EA_&an_N.  AS
		SELECT *
		FROM (
			SELECT 1 AS indicateur length = 3, COUNT(DISTINCT BPCO_BDLA_EA_Sej_Index) AS N length = 5 FROM
T_INDI_BPCO_BDLA_EA_&an_N. WHERE exacerbation_BPCO = 1
			UNION ALL
			SELECT 2, COUNT(DISTINCT BPCO_BDLA_EA_Sej_Index) FROM T_INDI_BPCO_BDLA_EA_&an_N.
WHERE exacerbation_BPCO = 2
			UNION ALL
			SELECT 3, COUNT(DISTINCT BPCO_BDLA_EA_Sej_Index) FROM T_INDI_BPCO_BDLA_EA_&an_N.
WHERE exacerbation_BPCO = 3
			UNION ALL
			SELECT 4, COUNT(DISTINCT BPCO_BDLA_EA_Sej_Index) FROM T_INDI_BPCO_BDLA_EA_&an_N.
WHERE exacerbation_BPCO = 4
			UNION ALL
```

```
                                SELECT 5, COUNT(DISTINCT BPCO_BDLA_EA_Sej_Index) FROM T_INDI_BPCO_BDLA_EA_&an_N.
WHERE exacerbation_BPCO = 5
                                UNION ALL
                                SELECT 6, COUNT(DISTINCT BPCO_BDLA_EA_Sej_Index) FROM T_INDI_BPCO_BDLA_EA_&an_N.
WHERE exacerbation_BPCO = 6
                                UNION ALL
                                SELECT 7, COUNT(DISTINCT BPCO_BDLA_EA_Sej_Index) FROM T_INDI_BPCO_BDLA_EA_&an_N.
WHERE exacerbation_BPCO = 7
                                UNION ALL
                                SELECT 8, COUNT(DISTINCT BPCO_BDLA_EA_Sej_Index) FROM T_INDI_BPCO_BDLA_EA_&an_N.
WHERE exacerbation_BPCO = 8
                                UNION ALL
                                SELECT 9, COUNT(DISTINCT BPCO_BDLA_EA_Sej_Index) FROM T_INDI_BPCO_BDLA_EA_&an_N.
WHERE exacerbation_BPCO = 9
                                UNION ALL
                                SELECT 10, COUNT(DISTINCT BPCO_BDLA_EA_Sej_Index) FROM T_INDI_BPCO_BDLA_EA_&an_N.
WHERE exacerbation_BPCO = 10
                                UNION ALL
                                SELECT 11, COUNT(DISTINCT BPCO_BDLA_EA_Sej_Index) FROM T_INDI_BPCO_BDLA_EA_&an_N.
WHERE exacerbation_BPCO = 11
                                UNION ALL
                                SELECT 12, COUNT(DISTINCT BPCO_BDLA_EA_Sej_Index) FROM T_INDI_BPCO_BDLA_EA_&an_N.
                        ) ;

quit;
```

3.11.2 01_Inclusions_et_exclusions.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                */
/*		Indicateur 06 : Inclusion & exclusions

                                                                */
/*
                                                                                                                */
/*
	*****************************************************************************************************************
**********************************	*/

*
	*****************************************************************************************************************
**********************************;
*	On récupère tous les séjours pour exacerbation de BPCO pour les patients de la table res.T_INDI_BPCO_BDLA_EA_&an_N.;

* Séjours avec BPCO en DAS;
data BPCO_DAS;
	set travail.sejours_cible_exacerbation_&annee_N. (where = (reperage = 16 and variable = "ASS_DGN"));
	BPCO_BDLA_EA_Sej_Index = id_sejour;
run;

proc sort data = T_INDI_BPCO_BDLA_EA_&an_N. out = reperage_BPCO nodupkey;
	by BPCO_BDLA_EA_Sej_Index UM_ORD_NUM;
run;

data reperage_BPCO2;
	set reperage_BPCO;
	by BPCO_BDLA_EA_Sej_Index UM_ORD_NUM;
	if first.BPCO_BDLA_EA_Sej_Index then
		output;
run;

proc sql undo_policy = none;

	CREATE TABLE res.T_INDI_BPCO_BDLA_EA_&an_N. AS
		SELECT DISTINCT
			BEN_IDT_ANO,
			date_debut,
			date_fin AS BPCO_BDLA_EA_Date_index length = 4,
			BPCO_BDLA_EA_Sej_Index,
			ETA_NUM AS Finess_PMSI,
			ETA_NUM_GEO AS Finess_GEO,
			DGN_PAL AS Dp,
			CASE	WHEN SUBSTR(DGN_PAL, 1, 4) IN ("J440", "J441", "J448", "J449") THEN 1
				WHEN SUBSTR(DGN_PAL, 1, 3) IN ("J12", "J13", "J14", "J15", "J16", "J17", "J18")
OR SUBSTR(DGN_PAL, 1, 4) = "J181" THEN 2
				WHEN SUBSTR(DGN_PAL, 1, 4) IN ("J960") THEN 3
				WHEN SUBSTR(DGN_PAL, 1, 3) IN ("J09", "J10", "J11") THEN 4
				WHEN SUBSTR(DGN_PAL, 1, 3) IN ("I26") THEN 5
				WHEN SUBSTR(DGN_PAL, 1, 3) IN ("I50") OR SUBSTR(DGN_PAL, 1, 4) IN ("I130",
"I132", "I110", "I501") THEN 6
				WHEN SUBSTR(DGN_PAL, 1, 3) IN ("J86", "J93") OR SUBSTR(DGN_PAL, 1, 4) IN
("J942") THEN 7
				END AS Dp_classe length = 3,
			CASE	WHEN BPCO_BDLA_EA_Sej_Index IN (SELECT BPCO_BDLA_EA_Sej_Index FROM BPCO_DAS)
THEN 1
				ELSE 0
				END AS DA_BPCO length = 3,
			GRG_GHM AS GHM,
```

```
                                1 AS BPCO_BDLA_EA_CIBLE length = 3
                        FROM reperage_BPCO2
                        WHERE UM_ORD_NUM = 1
                        ORDER BY BEN_IDT_ANO;

quit;

proc delete data = BPCO_DAS;
run; quit;

*
        ************************************************************************************************************
************************************;
*       On exclut les séjours dont le mode de sortie est le décès. ;

proc sql;

        * On insère l effectif dans le Flowchart;
        INSERT INTO flowch.Flow_Chart_BPCO_BDLA_EA_&an_N.
                SELECT
                        13,
                        COUNT(DISTINCT BPCO_BDLA_EA_Sej_Index) length = 3
        FROM res.T_INDI_BPCO_BDLA_EA_&an_N.
        WHERE BPCO_BDLA_EA_Sej_Index IN (SELECT BPCO_BDLA_EA_Sej_Index FROM T_INDI_BPCO_BDLA_EA_&an_N. WHERE
SOR_MOD = "9");

        * On conserve les id des séjours à supprimer;
        CREATE TABLE travail.exclus1 AS
                SELECT DISTINCT
                        BPCO_BDLA_EA_Sej_Index
        FROM res.T_INDI_BPCO_BDLA_EA_&an_N.
        WHERE BPCO_BDLA_EA_Sej_Index IN (SELECT BPCO_BDLA_EA_Sej_Index FROM T_INDI_BPCO_BDLA_EA_&an_N. WHERE
SOR_MOD = "9");

quit;

proc delete data = T_INDI_BPCO_BDLA_EA_&an_N.;
run; quit;


*
        ************************************************************************************************************
************************************;
*       Changement juillet 2021 - Nouvelle version des indicateurs : on modifie les exlcusions;
*       On exclut les séjours index suivi d au moins une journée d hospitalisation en MCO, SSR, HAD ou PSY dans les 60 jours suivant
la date index;

data sejours_&Annee_N._&Annee_N1.;
        set travail.sejours_&Annee_N._&Annee_N1.;
run;

proc sql;

        DELETE FROM sejours_&Annee_N._&Annee_N1.
        WHERE domaine = "MCO" AND (SUBSTR(GRG_GHM, 1, 2) = "28" OR DUREE_SEJOUR = 0);

        DELETE FROM sejours_&Annee_N._&Annee_N1.
        WHERE domaine ne "MCO" AND date_debut = date_fin;

quit;

proc sql;

        CREATE TABLE sejours_90jours AS
                SELECT DISTINCT
                        a.BPCO_BDLA_EA_Sej_Index
                FROM res.T_INDI_BPCO_BDLA_EA_&an_N. a
                        INNER JOIN sejours_&Annee_N._&Annee_N1. b
                                ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
                WHERE b.date_debut < (a.BPCO_BDLA_EA_Date_index + 90) AND a.BPCO_BDLA_EA_Date_index < b.date_fin
                        AND a.BPCO_BDLA_EA_Sej_Index NE b.id_sejour;
```

```
	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_BDLA_EA_&an_N.
		SELECT
			14,
			COUNT(DISTINCT BPCO_BDLA_EA_Sej_Index)
	FROM res.T_INDI_BPCO_BDLA_EA_&an_N.
	WHERE BPCO_BDLA_EA_Sej_Index IN (SELECT BPCO_BDLA_EA_Sej_Index FROM sejours_90jours);

	* On conserve les id des séjours à supprimer;
	CREATE TABLE travail.exclus2 AS
		SELECT DISTINCT
			BPCO_BDLA_EA_Sej_Index
	FROM res.T_INDI_BPCO_BDLA_EA_&an_N.
	WHERE BPCO_BDLA_EA_Sej_Index IN (SELECT BPCO_BDLA_EA_Sej_Index FROM sejours_90jours);

quit;

*
	************************************************************************************************************
************************************;
*	On exclut tous les patients;

data exclus;
	set	travail.exclus1-travail.exclus2;
run;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_BDLA_EA_&an_N.
		SELECT
			15,
			COUNT(DISTINCT BPCO_BDLA_EA_Sej_Index)
	FROM exclus;

	DELETE FROM res.T_INDI_BPCO_BDLA_EA_&an_N.
	WHERE BPCO_BDLA_EA_Sej_Index IN (SELECT BPCO_BDLA_EA_Sej_Index FROM exclus);

quit;

proc sort data = res.T_INDI_BPCO_BDLA_EA_&an_N. nodupkey;
	by _all_;
run; quit;

proc delete data = exclus sejours_&Annee_N._&Annee_N1.;
run; quit;

proc datasets library = travail memtype = data nolist;
	delete exclus1-exclus2;
run; quit;

*
	************************************************************************************************************
************************************;
*	Nombre de séjours inclus;

proc sql;

	INSERT INTO flowch.Flow_Chart_BPCO_BDLA_EA_&an_N.
		SELECT
			17,
			COUNT(DISTINCT BPCO_BDLA_EA_Sej_Index)
	FROM res.T_INDI_BPCO_BDLA_EA_&an_N.;

quit;

proc sql;

	SELECT COUNT(*), COUNT(DISTINCT BPCO_BDLA_EA_Sej_Index)
	FROM res.T_INDI_BPCO_BDLA_EA_&an_N.;
```

```
quit;
```

3.11.3 02_Traitement_apres_hospitalisation.sas

```
/*
	*****************************************************************************************************
**********************************	*/
/*
                                                                                                        */
/*		Indicateur 06 : Traitement après hospitalisation pour exacerbation

		*/
/*
                                                                                                        */
/*
	*****************************************************************************************************
**********************************	*/

*
	*****************************************************************************************************
**********************************;
*	On corrige les données de la table travail.reperage_BDLA_&Annee_2N._&Annee_N1. qui sont incohérentes avec la variable
nb_broncho_LDA de;
*	la table res.T_INDI_BPCO_&an_N.;
*	En effet, certaines délivrances de médicaments BDLA de 2017 ne remontent que dans les flux de 2019. Ils seront donc pris en
compte dans le;
*	repérage des délivrances de BDLA alors qu'ils n étaient pas pris en compte dans le dénombrement nb_broncho_LDA de la
table res.T_INDI_BPCO_&an_N.;
*	On supprime les délivrances de 2017 pour ces patients;

proc sql;

	CREATE TABLE patients_sans_BDLA AS
		SELECT
			BEN_IDT_ANO
		FROM res.T_INDI_BPCO_&an_N.
		WHERE Nb_Broncho_LDA = 0;

quit;

data reperage_BDLA_&Annee_2N._&Annee_N1.;
	set travail.reperage_BDLA_&Annee_2N._&Annee_N1.;
run;

proc sql;

	DELETE FROM reperage_BDLA_&Annee_2N._&Annee_N1.
		WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM patients_sans_BDLA) AND YEAR(date_debut) =
&Annee_N.;

quit;

proc delete data = patients_sans_BDLA;
run; quit;

*
	*****************************************************************************************************
**********************************;
*	On joint avec la table res.T_INDI_BPCO_BDLA_EA_&an_N. pour récupérer les info dans les 90 jours suivant la date index;

proc sql;

	CREATE TABLE Broncho_LDA_90j AS
		SELECT DISTINCT
			a.BPCO_BDLA_EA_Sej_Index,
			1 AS BPCO_BDLA_EA_OBS length = 3,
			COUNT(DISTINCT b.date_debut) AS Nb_Broncho_LDA_90j length = 4
		FROM res.T_INDI_BPCO_BDLA_EA_&an_N. a
			INNER JOIN reperage_BDLA_&Annee_2N._&Annee_N1. b
```

```
                                        ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
                        WHERE a.BPCO_BDLA_EA_Date_index <= b.date_debut <= (a.BPCO_BDLA_EA_Date_index + 90)
                        GROUP BY BPCO_BDLA_EA_Sej_Index;

quit;

*       On ajoute l information dans la table de résultats;
proc sort data = res.T_INDI_BPCO_BDLA_EA_&an_N.;
        by BPCO_BDLA_EA_Sej_Index;
run;

proc sort data = Broncho_LDA_90j;
        by BPCO_BDLA_EA_Sej_Index;
run;

data res.T_INDI_BPCO_BDLA_EA_&an_N.;
        merge   res.T_INDI_BPCO_BDLA_EA_&an_N. (in = a)
                        Broncho_LDA_90j (in = b);
        by BPCO_BDLA_EA_Sej_Index;
        if BPCO_BDLA_EA_OBS ne 1 then
                do;
                        BPCO_BDLA_EA_OBS = 0;
                        Nb_Broncho_LDA_90j = 0;
                end;
        if a then
                output;
run;

proc delete data = Broncho_LDA_90j;
run; quit;

*
        ***********************************************************************************************************
***********************************;
*       On joint avec la table res.T_INDI_BPCO_BDLA_EA_&an_N. pour récupérer les info dans les 180 jours suivant la date index;
proc sql;

        CREATE TABLE Broncho_LDA_180j AS
                SELECT DISTINCT
                        a.BPCO_BDLA_EA_Sej_Index,
                        MIN(b.date_debut) AS Date_Broncho_LDA_180j length = 4
                FROM res.T_INDI_BPCO_BDLA_EA_&an_N. a
                        INNER JOIN reperage_BDLA_&Annee_2N._&Annee_N1. b
                                ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
                WHERE a.BPCO_BDLA_EA_Date_index <= b.date_debut <= (a.BPCO_BDLA_EA_Date_index + 180)
                GROUP BY BPCO_BDLA_EA_Sej_Index;

quit;

*       On ajoute l information dans la table de résultats;
proc sort data = res.T_INDI_BPCO_BDLA_EA_&an_N.;
        by BPCO_BDLA_EA_Sej_Index;
run;

proc sort data = Broncho_LDA_180j;
        by BPCO_BDLA_EA_Sej_Index;
run;

data res.T_INDI_BPCO_BDLA_EA_&an_N.;
        merge   res.T_INDI_BPCO_BDLA_EA_&an_N. (in = a)
                        Broncho_LDA_180j (in = b);
        by BPCO_BDLA_EA_Sej_Index;
        if a then
                output;
run;

proc delete data = Broncho_LDA_180j;
run; quit;
```

```
*
	**********************************************************************************************************
**********************************;
*	On joint avec la table res.T_INDI_BPCO_BDLA_EA_&an_N. pour récupérer les info dans les 90 jours avant la date index;
proc sql;

	CREATE TABLE Broncho_LDA_90javant AS
		SELECT DISTINCT
			a.BPCO_BDLA_EA_Sej_Index,
			1 AS Broncho_LDA_90javant length = 3
		FROM res.T_INDI_BPCO_BDLA_EA_&an_N. a
			INNER JOIN reperage_BDLA_&Annee_2N._&Annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE (a.BPCO_BDLA_EA_Date_index - 90) <= b.date_debut < a.BPCO_BDLA_EA_Date_index
		ORDER BY BPCO_BDLA_EA_Sej_Index;

quit;

proc delete data = reperage_BDLA_&Annee_2N._&Annee_N1.;
run; quit;

*	On ajoute l information dans la table de résultats;
proc sort data = res.T_INDI_BPCO_BDLA_EA_&an_N.;
	by BPCO_BDLA_EA_Sej_Index;
run;

proc sort data = Broncho_LDA_90javant;
	by BPCO_BDLA_EA_Sej_Index;
run;

data res.T_INDI_BPCO_BDLA_EA_&an_N.;
	merge	res.T_INDI_BPCO_BDLA_EA_&an_N. (in = a)
			Broncho_LDA_90javant (in = b);
	by BPCO_BDLA_EA_Sej_Index;
	if Broncho_LDA_90javant ne 1 then
		Broncho_LDA_90javant = 0;
	if a then
		output;
run;

proc delete data = Broncho_LDA_90javant;
run; quit;
```

3.11.4 03_Sejours_d_exacerbation_deBPCO_en_MCO.sas

```
/*
	*****************************************************************************************************
**********************************	*/
/*

*/
/*	Indicateur 06 : Nombre de séjours pour exacerbation de BPCO dans l année N

	*/
/*

*/
/*
	*****************************************************************************************************
**********************************	*/

*
	*****************************************************************************************************
**********************************;
*	Nombre de séjours MCO d'exacerbation de BPCO codé en DP ou en DAS terminés l année N;

proc sql;

	CREATE TABLE nb_sejours_exa_BPCO_DP AS
		SELECT
			BEN_IDT_ANO,
			COUNT(DISTINCT id_sejour) AS Nb_Sej_EA_BPCO_DP_MCO length = 3
		FROM travail.sejours_exacerbation_&annee_N.
		WHERE reperage = 16 AND variable = "DGN_PAL"
		GROUP BY BEN_IDT_ANO;

	CREATE TABLE nb_sejours_exa_BPCO_DAS AS
		SELECT
			BEN_IDT_ANO,
			COUNT(DISTINCT id_sejour) AS Nb_Sej_EA_BPCO_DAS_MCO length = 3
		FROM travail.sejours_exacerbation_&annee_N.
		WHERE reperage = 16 AND variable = "ASS_DGN"
		GROUP BY BEN_IDT_ANO;

quit;

*
	*****************************************************************************************************
**********************************;
*	On ajoute l'information dans la table res.T_INDI_BPCO_BDLA_EA_&an_N.;

proc sort data = res.T_INDI_BPCO_BDLA_EA_&an_N.;
	by BEN_IDT_ANO;
run;

data res.T_INDI_BPCO_BDLA_EA_&an_N.;
	merge	res.T_INDI_BPCO_BDLA_EA_&an_N. (in = a)
			nb_sejours_exa_BPCO_DP (in = b)
			nb_sejours_exa_BPCO_DAS (in = c);
	by BEN_IDT_ANO;
	if a;
	if not b then
		Nb_Sej_EA_BPCO_DP_MCO = 0;
	if not c then
		Nb_Sej_EA_BPCO_DAS_MCO = 0;
run;
```

3.11.5 04_Table_finale.sas

```
/*
	*****************************************************************************************************
**********************************	*/
/*

*/
/*		Indicateur 06 : Table finale - Ajout de formats et de labels

		*/
/*

*/
/*
	*****************************************************************************************************
**********************************	*/

data res.T_INDI_BPCO_BDLA_EA_&an_N.;
	set res.T_INDI_BPCO_BDLA_EA_&an_N. (keep = BEN_IDT_ANO BPCO_BDLA_EA_Date_index BPCO_BDLA_EA_Sej_Index
Finess_PMSI Finess_GEO Dp Dp_classe
		DA_BPCO GHM BPCO_BDLA_EA_CIBLE BPCO_BDLA_EA_OBS Nb_Sej_EA_BPCO_DP_MCO
Nb_Sej_EA_BPCO_DAS_MCO Nb_Broncho_LDA_90j Date_Broncho_LDA_180j
		Broncho_LDA_90javant);
	label
		BEN_IDT_ANO = "Numéro d'individu"
		BPCO_BDLA_EA_Date_index = "Date index"
		BPCO_BDLA_EA_Sej_Index = "Identifiant du séjour index"
		Finess_PMSI = "Finess PMSI ayant réalisé le séjour index"
		Finess_GEO = "Finess géographique du premier RUM"
		Dp = "Diagnostic principal du séjour index"
		Dp_classe = "Diagnostic principal du séjour index"
		DA_BPCO = "BPCO codé en diagnostic secondaire du séjour index"
		GHM = "GHM du séjour index"
		BPCO_BDLA_EA_CIBLE = "Séjour correspondant aux critères d'inclusion et d'exclusion de la population cible"
		BPCO_BDLA_EA_OBS = "Séjour de la population cible pour lesquels le patient a eu une délivrance remboursée de
traitement bronchodilatateur bêta-2 longue durée d'action dans les 90 jours suivant la date index"
		Nb_Sej_EA_BPCO_DP_MCO = "Nombre de séjours MCO d'exacerbation de BPCO codé en DP terminés en
&Annee_N."
		Nb_Sej_EA_BPCO_DAS_MCO = "Nombre de séjours MCO d'exacerbation de BPCO codé en DAS terminés en
&Annee_N."
		Nb_Broncho_LDA_90j = "Nombre de délivrances remboursées de bronchodilatateur anticholinergique ou Bêta-2
longue durée d'action dans les 90 jours suivant la date index"
		Date_Broncho_LDA_180j = "Date de la première délivrance remboursée de bronchodilatateur anticholinergique
ou Bêta-2 longue durée d'action dans les 180 jours suivant la date index"
		Broncho_LDA_90javant = "Patient ayant eu au moins une délivrance remboursée d'un bronchodilatateur
anticholinergique ou Bêta-2 longue durée d'action dans les 90 jours précédant la date index"
		;
	format DA_BPCO BPCO_BDLA_EA_CIBLE BPCO_BDLA_EA_OBS Broncho_LDA_90javant f_oui_non. Dp_Classe f_DP_classe.
		BPCO_BDLA_EA_Date_index Date_Broncho_LDA_180j date9.;
run;

*
	*****************************************************************************************************
**********************************;
*	Nombre de traitement de BDLA dans les 3 mois après hospitalisation;

proc sql;

	INSERT INTO flowch.Flow_Chart_BPCO_BDLA_EA_&an_N.
		SELECT
			18,
			COUNT(DISTINCT BPCO_BDLA_EA_Sej_Index)
	FROM res.T_INDI_BPCO_BDLA_EA_&an_N.
	WHERE BPCO_BDLA_EA_OBS = 1;

	* Population d étude;
	SELECT N into : N12 FROM flowch.Flow_Chart_BPCO_BDLA_EA_&an_N. WHERE indicateur = 12;
	* Population cible;
```

```
	SELECT N into : N17 FROM flowch.Flow_Chart_BPCO_BDLA_EA_&an_N. WHERE indicateur = 17;

quit;

data flowch.Flow_Chart_BPCO_BDLA_EA_&an_N.;
	set flowch.Flow_Chart_BPCO_BDLA_EA_&an_N.;
	if indicateur in (13, 14, 15, 17) then
		percent = N / &N12.;
	if indicateur = 18 then
		percent = N / &N17.;
	format indicateur f_ind_06_flowchart. percent percent7.1;
run;

* Vérif;
proc sql;

	SELECT COUNT(*) AS nb_lignes, COUNT(DISTINCT BPCO_BDLA_EA_Sej_Index) AS nb_sejours
	FROM res.T_INDI_BPCO_BDLA_EA_&an_N.;
	* nb_lignes = nb_sejours : OK;

	SELECT COUNT(*) AS nb_lignes, COUNT(DISTINCT BPCO_BDLA_EA_Sej_Index) AS nb_sejours
	FROM res.T_INDI_BPCO_BDLA_EA_&an_N.
	WHERE BPCO_BDLA_EA_OBS = 1;
	* nb_lignes = nb_sejours : OK;

quit;

*
	**********************************************************************************************************
**********************************;
*	On ajoute le flag BPCO_BDLA_EA_CIBLE dans la table res.T_INDI_BPCO_&an_N.;

proc sql;

	ALTER TABLE res.T_INDI_BPCO_&an_N. DROP BPCO_BDLA_EA_CIBLE;

	ALTER TABLE res.T_INDI_BPCO_&an_N. ADD BPCO_BDLA_EA_CIBLE INT format = f_oui_non. length = 3
		label = "Patient appartenant à la population cible BPCO_BDLA_EA";

	UPDATE res.T_INDI_BPCO_&an_N.
		SET BPCO_BDLA_EA_CIBLE = CASE		WHEN BEN_IDT_ANO IN (SELECT DISTINCT BEN_IDT_ANO FROM
res.T_INDI_BPCO_BDLA_EA_&an_N.) THEN 1
									ELSE 0
									END;

quit;

data res.T_INDI_BPCO_&an_N.;
	set res.T_INDI_BPCO_&an_N.;
	if BPCO_BDLA_EA_CIBLE = .
		then BPCO_BDLA_EA_CIBLE = 0;
run;
```

3.12 Soins de rééducation dans les 90 jours après une hospitalisation pour exacerbation de BPCO des patients sortis à domicile

3.12.1 00_Initialisation_de_la_table_de_resultats.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*		Table de population - Patients avec indicateur_07 = 1

			*/
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/

data T_INDI_BPCO_RR_EA_&an_N. (keep = BEN_IDT_ANO);
	set pop.indicateurs_&an_N. (where = (indicateur_07 = 1));
run;

*
	*****************************************************************************************************************
***********************************;
*	Effectifs pour le flowchart;

* Individus de 40 ans et plus ayant bénéficié de soins remboursés au moins une fois l année N;

proc sort data = T_INDI_BPCO_RR_EA_&an_N. nodupkey;
	by BEN_IDT_ANO;
run;

proc sort data = travail.sejours_cible_exacerbation_&annee_N.;
	by BEN_IDT_ANO;
run;

data T_INDI_BPCO_RR_EA_&an_N.;
	merge	T_INDI_BPCO_RR_EA_&an_N. (in = a)
			travail.sejours_cible_exacerbation_&annee_N. (in = b);
	by BEN_IDT_ANO;
	if a and b;
	BPCO_RR_EA_Sej_Index = id_sejour;
run;

proc sort data = T_INDI_BPCO_RR_EA_&an_N. out = premier_sejour;
	by BEN_IDT_ANO date_debut date_fin;
run;

data premier_sejour (keep = BEN_IDT_ANO id_sejour);
	set premier_sejour;
	by BEN_IDT_ANO date_debut date_fin;
	if first.BEN_IDT_ANO then
		output;
run;

proc sort data = T_INDI_BPCO_RR_EA_&an_N.;
	by BEN_IDT_ANO id_sejour;
run;

proc sort data = premier_sejour;
	by BEN_IDT_ANO id_sejour;
```

```
run;

data T_INDI_BPCO_RR_EA_&an_N.;
	merge	premier_sejour (in = a)
			T_INDI_BPCO_RR_EA_&an_N. (in = b);
	by BEN_IDT_ANO id_sejour;
	if a and b;
run;

proc sql;

	CREATE TABLE flowch.Flow_Chart_BPCO_RR_EA_&an_N.  AS
		SELECT *
		FROM (
			SELECT 1 AS indicateur length = 3, COUNT(DISTINCT BEN_IDT_ANO) AS N length = 5 FROM
T_INDI_BPCO_RR_EA_&an_N. WHERE exacerbation_BPCO = 1
			UNION ALL
			SELECT 2, COUNT(DISTINCT BEN_IDT_ANO) FROM T_INDI_BPCO_RR_EA_&an_N. WHERE
exacerbation_BPCO = 2
			UNION ALL
			SELECT 3, COUNT(DISTINCT BEN_IDT_ANO) FROM T_INDI_BPCO_RR_EA_&an_N. WHERE
exacerbation_BPCO = 3
			UNION ALL
			SELECT 4, COUNT(DISTINCT BEN_IDT_ANO) FROM T_INDI_BPCO_RR_EA_&an_N. WHERE
exacerbation_BPCO = 4
			UNION ALL
			SELECT 5, COUNT(DISTINCT BEN_IDT_ANO) FROM T_INDI_BPCO_RR_EA_&an_N. WHERE
exacerbation_BPCO = 5
			UNION ALL
			SELECT 6, COUNT(DISTINCT BEN_IDT_ANO) FROM T_INDI_BPCO_RR_EA_&an_N. WHERE
exacerbation_BPCO = 6
			UNION ALL
			SELECT 7, COUNT(DISTINCT BEN_IDT_ANO) FROM T_INDI_BPCO_RR_EA_&an_N. WHERE
exacerbation_BPCO = 7
			UNION ALL
			SELECT 8, COUNT(DISTINCT BEN_IDT_ANO) FROM T_INDI_BPCO_RR_EA_&an_N. WHERE
exacerbation_BPCO = 8
			UNION ALL
			SELECT 9, COUNT(DISTINCT BEN_IDT_ANO) FROM T_INDI_BPCO_RR_EA_&an_N. WHERE
exacerbation_BPCO = 9
			UNION ALL
			SELECT 10, COUNT(DISTINCT BEN_IDT_ANO) FROM T_INDI_BPCO_RR_EA_&an_N. WHERE
exacerbation_BPCO = 10
			UNION ALL
			SELECT 11, COUNT(DISTINCT BEN_IDT_ANO) FROM T_INDI_BPCO_RR_EA_&an_N. WHERE
exacerbation_BPCO = 11
			UNION ALL
			SELECT 12, COUNT(DISTINCT BEN_IDT_ANO) FROM T_INDI_BPCO_RR_EA_&an_N.
		) ;

quit;
```

3.12.2 01_Inclusions_et_exclusions.sas

```
/*
	*****************************************************************************************************
**********************************	*/
/*

                                                                                                        */
/*			Indicateur 07 : Inclusion & exclusions

                                                                */
/*

                                                                                                        */
/*
	*****************************************************************************************************
**********************************	*/

*
	*****************************************************************************************************
**********************************;
*	On récupère le premier séjour pour exacerbation de BPCO pour les patients de la table res.T_INDI_BPCO_RR_EA_&an_N.;

proc sort data = T_INDI_BPCO_RR_EA_&an_N.;
	by BEN_IDT_ANO date_debut date_fin;
run;

data sej_inclusion;
	set T_INDI_BPCO_RR_EA_&an_N. (keep = BEN_IDT_ANO id_sejour);
	by BEN_IDT_ANO;
	if first.BEN_IDT_ANO then
		output;
run;

* On récupère l ensemble des lignes des séjours d inclusion;
proc sort data = sej_inclusion;
	by BEN_IDT_ANO id_sejour;
run;
proc sort data = T_INDI_BPCO_RR_EA_&an_N.;
	by BEN_IDT_ANO id_sejour;
run;

data T_INDI_BPCO_RR_EA_&an_N.;
	merge	T_INDI_BPCO_RR_EA_&an_N. (in = a)
			sej_inclusion (in = b);
	by BEN_IDT_ANO id_sejour;
	if a and b;
run;

* Séjours avec BPCO en DAS;
data BPCO_DAS;
	set travail.sejours_cible_exacerbation_&annee_N. (where = (reperage = 16 and variable = "ASS_DGN"));
	BPCO_RR_EA_Sej_Index = id_sejour;
run;

proc sort data = T_INDI_BPCO_RR_EA_&an_N. out = reperage_BPCO nodupkey;
	by BPCO_RR_EA_Sej_Index UM_ORD_NUM;
run;

data reperage_BPCO2;
	set reperage_BPCO;
	by BPCO_RR_EA_Sej_Index UM_ORD_NUM;
	if first.BPCO_RR_EA_Sej_Index then
		output;
run;

proc sql undo_policy = none;

	CREATE TABLE res.T_INDI_BPCO_RR_EA_&an_N. AS
```

```
                SELECT DISTINCT
                        BEN_IDT_ANO,
                        date_debut,
                        date_fin AS BPCO_RR_EA_Date_index length = 4,
                        BPCO_RR_EA_Sej_Index,
                        ETA_NUM AS Finess_PMSI,
                        ETA_NUM_GEO AS Finess_GEO,
                        DGN_PAL AS Dp,
                        CASE    WHEN SUBSTR(DGN_PAL, 1, 4) IN ("J440", "J441", "J448", "J449") THEN 1
                                WHEN SUBSTR(DGN_PAL, 1, 3) IN ("J12", "J13", "J14", "J15", "J16", "J17", "J18")
OR SUBSTR(DGN_PAL, 1, 4) = "J181" THEN 2
                                WHEN SUBSTR(DGN_PAL, 1, 4) IN ("J960") THEN 3
                                WHEN SUBSTR(DGN_PAL, 1, 3) IN ("J09", "J10", "J11") THEN 4
                                WHEN SUBSTR(DGN_PAL, 1, 3) IN ("I26") THEN 5
                                WHEN SUBSTR(DGN_PAL, 1, 3) IN ("I50") OR SUBSTR(DGN_PAL, 1, 4) IN ("I130",
"I132", "I110", "I501") THEN 6
                                WHEN SUBSTR(DGN_PAL, 1, 3) IN ("J86", "J93") OR SUBSTR(DGN_PAL, 1, 4) IN
("J942") THEN 7
                                END AS Dp_classe length = 3,
                        CASE    WHEN BPCO_RR_EA_Sej_Index IN (SELECT BPCO_RR_EA_Sej_Index FROM BPCO_DAS) THEN
1
                                ELSE 0
                                END AS DA_BPCO length = 3,
                        GRG_GHM AS GHM,
                        1 AS BPCO_RR_EA_CIBLE length = 3
                FROM reperage_BPCO2
                WHERE UM_ORD_NUM = 1
                ORDER BY BEN_IDT_ANO;

quit;

proc delete data = BPCO_DAS sej_inclusion;
run; quit;

*
        ****************************************************************************************************
**********************************;
*       Changement juillet 2021 - Nouvelle version des indicateurs : on modifie les exclusions;
*       On exclut les patients avec des séjours dont le mode de sortie est le décès. ;

proc sql;

        * On insère l effectif dans le Flowchart;
        INSERT INTO flowch.Flow_Chart_BPCO_RR_EA_&an_N.
                SELECT
                        13,
                        COUNT(DISTINCT BEN_IDT_ANO)
        FROM res.T_INDI_BPCO_RR_EA_&an_N.
        WHERE BPCO_RR_EA_Sej_Index IN (SELECT BPCO_RR_EA_Sej_Index FROM T_INDI_BPCO_RR_EA_&an_N. WHERE SOR_MOD
= "9");

        * On conserve les id des séjours à supprimer;
        CREATE TABLE travail.exclus1 AS
                SELECT DISTINCT
                        BEN_IDT_ANO
        FROM res.T_INDI_BPCO_RR_EA_&an_N.
        WHERE BPCO_RR_EA_Sej_Index IN (SELECT BPCO_RR_EA_Sej_Index FROM T_INDI_BPCO_RR_EA_&an_N. WHERE SOR_MOD
= "9");

quit;

proc delete data = T_INDI_BPCO_RR_EA_&an_N.;
run; quit;

*
        ****************************************************************************************************
**********************************;
*       Changement juillet 2021 - Nouvelle version des indicateurs : on modifie les exclusions;
*       On exclut les patients dont le séjour index suivi d au moins une journée d hospitalisation en MCO, HAD ou PSY dans les 90
jours suivant la date index;
```

```
data sejours_&Annee_N._&Annee_N1.;
	set travail.sejours_&Annee_N._&Annee_N1.;
run;

proc sql;

	DELETE FROM sejours_&Annee_N._&Annee_N1.
	WHERE domaine = "MCO" AND (SUBSTR(GRG_GHM, 1, 2) = "28" OR DUREE_SEJOUR = 0);

	DELETE FROM sejours_&Annee_N._&Annee_N1.
	WHERE domaine ne "MCO" AND date_debut = date_fin;

quit;

proc sql;

	CREATE TABLE sejours_90jours AS
		SELECT DISTINCT
			a.BPCO_RR_EA_Sej_Index
		FROM res.T_INDI_BPCO_RR_EA_&an_N. a
			INNER JOIN sejours_&Annee_N._&Annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.date_debut < (a.BPCO_RR_EA_Date_index + 90) AND a.BPCO_RR_EA_Date_index < b.date_fin
			AND a.BPCO_RR_EA_Sej_Index NE b.id_sejour AND b.domaine NE "SSR";

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_RR_EA_&an_N.
		SELECT
			14,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM res.T_INDI_BPCO_RR_EA_&an_N.
	WHERE BPCO_RR_EA_Sej_Index IN (SELECT BPCO_RR_EA_Sej_Index FROM sejours_90jours);

	* On conserve les id des séjours à supprimer;
	CREATE TABLE travail.exclus2 AS
		SELECT DISTINCT
			BEN_IDT_ANO
	FROM res.T_INDI_BPCO_RR_EA_&an_N.
	WHERE BPCO_RR_EA_Sej_Index IN (SELECT BPCO_RR_EA_Sej_Index FROM sejours_90jours);

quit;

*
	**********************************************************************************************************
**********************************;
*	On exclut les patients en ALD Alzheimer et autres démences actives 90 jours après la date index;

proc sql undo_policy = none;

	CREATE TABLE ALD_Alz_autres AS
		SELECT DISTINCT
			a.BEN_IDT_ANO
		FROM res.T_INDI_BPCO_RR_EA_&an_N. a
			INNER JOIN travail.histo_ALD b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.reperage IN (50, 51) AND
			(a.BPCO_RR_EA_Date_index <= b.date_debut <= (a.BPCO_RR_EA_Date_index + 90)
			OR a.BPCO_RR_EA_Date_index <= b.date_fin <= (a.BPCO_RR_EA_Date_index + 90)
			OR (b.date_debut <= (a.BPCO_RR_EA_Date_index + 90)  AND (b.date_fin >=
a.BPCO_RR_EA_Date_index + 90 OR b.date_fin = "01JAN1600"d)));

quit;

proc sql;

	* On insère l effectif dans le Flowchart;
	INSERT INTO flowch.Flow_Chart_BPCO_RR_EA_&an_N.
		SELECT
			15,
			COUNT(DISTINCT BEN_IDT_ANO)
	FROM res.T_INDI_BPCO_RR_EA_&an_N.
```

```
		WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM ALD_Alz_autres);

		* On conserve les id des séjours à supprimer;
		CREATE TABLE travail.exclus3 AS
			SELECT DISTINCT
				BEN_IDT_ANO
		FROM res.T_INDI_BPCO_RR_EA_&an_N.
		WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM ALD_Alz_autres);

quit;

proc delete data = ALD_Alz_autres sejours_&Annee_N._&Annee_N1.;
run; quit;

*
		*************************************************************************************************************
**********************************;
*		On exclut tous les patients;

data exclus;
		set		travail.exclus1-travail.exclus3;
run;

proc sql;

		* On insère l effectif dans le Flowchart;
		INSERT INTO flowch.Flow_Chart_BPCO_RR_EA_&an_N.
			SELECT
				16,
				COUNT(DISTINCT BEN_IDT_ANO)
		FROM exclus;

		DELETE FROM res.T_INDI_BPCO_RR_EA_&an_N.
		WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM exclus);

quit;

proc sort data = res.T_INDI_BPCO_RR_EA_&an_N. nodupkey;
		by _all_;
run; quit;

proc delete data = exclus;
run; quit;

proc datasets library = travail memtype = data nolist;
		delete exclus1-exclus3;
run; quit;

*
		*************************************************************************************************************
**********************************;
*		Nombre de patients inclus;

proc sql;

		INSERT INTO flowch.Flow_Chart_BPCO_RR_EA_&an_N.
			SELECT
				18,
				COUNT(DISTINCT BEN_IDT_ANO)
		FROM res.T_INDI_BPCO_RR_EA_&an_N.;

quit;

proc sql;

		SELECT COUNT(*), COUNT(DISTINCT BEN_IDT_ANO)
		FROM res.T_INDI_BPCO_RR_EA_&an_N.;

quit;
```

3.12.3 02_Rehabilitation_respiratoire.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*		Indicateur 07 : Réhabilitation respiratoire
                                                                                                                    */
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/

*
	*****************************************************************************************************************
**********************************;
*	On repère les patients avec un séjour en SSR dans les 90 jours suivant la date index + info sur les UM 54 et les CM04;

proc sql;

	CREATE TABLE Sej_SSR_90j AS
		SELECT DISTINCT
			a.BEN_IDT_ANO,
			1 AS Sej_SSR length = 3,
			MIN(b.date_debut) AS Date_RR_SSR format ddmmyy10. length = 4,
			MAX(CASE WHEN b.CM_04 = 1 THEN 1
                                                ELSE 0
                                                END) AS Sej_CM04 length = 3,
			MAX(CASE WHEN b.type_UM = "54" THEN 1
                                                ELSE 0
                                                END) AS Sej_UM54 length = 3,
			MAX(CASE WHEN b.CM_04 = 1 AND b.type_UM = "54" THEN 1
                                                ELSE 0
                                                END) AS Sej_CM04_UM54 length = 3,
			MAX(CASE WHEN b.type_UM = "50" THEN 1
                                                ELSE 0
                                                END) AS Sej_UM50 length = 3,
			MAX(CASE WHEN CM_04 = 1 AND b.type_UM = "50" THEN 1
                                                ELSE 0
                                                END) AS Sej_CM04_UM50 length = 3,
			MAX(CASE WHEN CCAM = 1 OR CSARR = 1 THEN 1
                                                ELSE 0
                                                END) AS Sej_CSAR_CCAM length = 3,
			MAX(CASE WHEN CM_04 = 1 AND (CCAM = 1 OR CSARR = 1) THEN 1
                                                ELSE 0
                                                END) AS Sej_CM04_CSAR_CCAM length = 3,
			MAX(CASE WHEN b.type_UM = "50" AND (CCAM = 1 OR CSARR = 1) THEN 1
                                                ELSE 0
                                                END) AS Sej_UM50_CSAR_CCAM length = 3,
			MAX(CASE WHEN b.CM_04 = 1 AND b.type_UM = "50" AND (CCAM = 1 OR CSARR = 1) THEN 1
                                                ELSE 0
                                                END) AS SEJ_CM04_UM50_CSAR_CCAM length = 3
		FROM res.T_INDI_BPCO_RR_EA_&an_N. a
			INNER JOIN travail.RR_sejours_SSR_&Annee_1N._&Annee_N1. b
                                ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE a.BPCO_RR_EA_Date_index <= b.date_debut <= (a.BPCO_RR_EA_Date_index + 90)
		GROUP BY 1;

quit;

*
	*****************************************************************************************************************
**********************************;
```

```
*		On repère les patient avec une RR kinésithérapique respiratoire pour les patients atteints de handicap respiratoire dans les
90 jours
		suivant la date index;

proc sql;

		CREATE TABLE RR_kine_90j AS
			SELECT
				a.BEN_IDT_ANO,
				MIN(b.Date_RR) AS Date_RR_kine format ddmmyy10. length = 4,
				MAX(CASE WHEN b.type_reperage = "AMK AMC 28" THEN 1
							ELSE 0
							END) AS KINE_AMK_AMC_28 length = 3,
				MAX(CASE WHEN b.type_reperage = "AMK AMC 20" THEN 1
							ELSE 0
							END) AS KINE_AMK_AMC_20 length = 3,
				MAX(CASE WHEN b.KINE_BPC = 1 THEN 1
							ELSE 0
							END) AS KINE_BPC,
				MAX(CASE WHEN b.KINE_AMK_AMC_AMS_8 = 1 THEN 1
							ELSE 0
							END) AS KINE_AMK_AMC_AMS_8,
				MAX(CASE WHEN b.KINE_AMK_13_5 = 1 THEN 1
							ELSE 0
							END) AS KINE_AMS_AMC_AMK_13_5,
				MAX(CASE WHEN  b.KINE_AMK_AMC_AMS_9_5_8 = 1 THEN 1
							ELSE 0
							END) AS KINE_AMK_AMC_AMS_9_5_8,
				MAX(CASE WHEN b.KINE_AMK_AMC_AMS_9_5_4 = 1 THEN 1
							ELSE 0
							END) AS KINE_AMK_AMC_AMS_9_5_4
			FROM res.T_INDI_BPCO_RR_EA_&an_N. a
				INNER JOIN travail.reperage_RR_&Annee_1N._&Annee_N1. b
					ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
			WHERE a.BPCO_RR_EA_Date_index <= b.Date_RR <= (a.BPCO_RR_EA_Date_index + 90)
			GROUP BY a.BEN_IDT_ANO;

quit;

* On récupère le lieu de la RR pour la RR kinésithérapique respiratoire;
proc sql;

		* On fait un max car si DCIR + PMSI, c est un doublon mal identifié, donc la RR a lieu dans un etb;
		CREATE TABLE lieu_RR AS
			SELECT
				a.BEN_IDT_ANO,
				MAX(b.Lieu_RR) AS Lieu_RR length = 3
			FROM RR_kine_90j a
				INNER JOIN travail.reperage_RR_&Annee_1N._&Annee_N1. b
					ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
					AND a.Date_RR_kine = b.Date_RR
			GROUP BY a.BEN_IDT_ANO;

quit;

data RR_kine_90j;
		merge	RR_kine_90j
				lieu_RR;
		by BEN_IDT_ANO;
run;

proc delete data = lieu_RR;
run; quit;

*
		*************************************************************************************************
**********************************;
*		On récupère la date minimale et le lieu associé de la première RR;

data premiere_RR_90j;
		merge	Sej_SSR_90j (in = a)
```

```
                                RR_kine_90j (in = b);
        by BEN_IDT_ANO;
        if Date_RR_SSR ne . then
                do;
                        Date_RR = Date_RR_SSR;
                        Lieu_RR = 3;
                end;
        else
                Date_RR = Date_RR_kine;
        format date_RR ddmmyy10.;
run;

*
        ************************************************************************************************************
**********************************;
*       On ajoute l information dans la table de résultats;

proc sort data = res.T_INDI_BPCO_RR_EA_&an_N.;
        by BEN_IDT_ANO;
run;

data res.T_INDI_BPCO_RR_EA_&an_N.;
        merge   res.T_INDI_BPCO_RR_EA_&an_N. (in = a)
                        premiere_RR_90j (in = b);
        by BEN_IDT_ANO;
        array v Sej_: KINE_:;
        do over v;
                if v = . then
                        v = 0;
        end;
        length BPCO_RR_EA_OBS 3.;
        BPCO_RR_EA_OBS = 0;
        if Date_RR ne . then
                BPCO_RR_EA_OBS = 1;
        if a then
                output;
run;

proc delete data = Sej_SSR_90j RR_kine_90j premiere_RR_90j;
run; quit;

*
        ************************************************************************************************************
**********************************;
*       On ajoute l information de la date de la RR précédente;

* RR en SSR;
proc sql;

        CREATE TABLE Sej_SSR_prec AS
                SELECT DISTINCT
                        a.BEN_IDT_ANO,
                        MAX(b.date_debut) AS Date_RR_SSR format ddmmyy10. length = 4
                FROM res.T_INDI_BPCO_RR_EA_&an_N. a
                        INNER JOIN travail.RR_sejours_SSR_&Annee_1N._&Annee_N1. b
                                ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
                WHERE (a.BPCO_RR_EA_Date_index - 365) <= b.date_debut < a.BPCO_RR_EA_Date_index
                GROUP BY a.BEN_IDT_ANO;

quit;

* RR en kinésithérapique respiratoire;
proc sql;

        CREATE TABLE RR_kine_prec AS
                SELECT
                        a.BEN_IDT_ANO,
                        MAX(b.Date_RR) AS Date_RR_kine format ddmmyy10. length = 4
                FROM res.T_INDI_BPCO_RR_EA_&an_N. a
                        INNER JOIN travail.reperage_RR_&Annee_1N._&Annee_N1. b
                                ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
```

```
                        WHERE (a.BPCO_RR_EA_Date_index - 365) <= b.Date_RR < a.BPCO_RR_EA_Date_index
                        GROUP BY a.BEN_IDT_ANO;

quit;

*               On récupère la date de la dernière RR avant inclusion (priorité sur la date SSR);
data RR_precedente;
                merge       Sej_SSR_prec (in = a)
                                        RR_kine_prec (in = b);
                by BEN_IDT_ANO;
                length Date_RR_prec 4.;
                if Date_RR_SSR ne . then
                                Date_RR_prec = Date_RR_SSR;
                else
                                Date_RR_prec = Date_RR_kine;
                format Date_RR_prec ddmmyy10.;
run;

proc delete data = Sej_SSR_prec RR_kine_prec;
run;

proc sort data = res.T_INDI_BPCO_RR_EA_&an_N.;
                by BEN_IDT_ANO;
run;

proc sort data = RR_precedente;
                by BEN_IDT_ANO;
run;

data res.T_INDI_BPCO_RR_EA_&an_N.;
                merge       res.T_INDI_BPCO_RR_EA_&an_N. (in = a)
                                        RR_precedente (in = b);
                by BEN_IDT_ANO;
                if a then
                                output;
run;

proc delete data = RR_precedente;
run; quit;

proc sql;

                SELECT COUNT(*), COUNT(DISTINCT BEN_IDT_ANO)
                FROM res.T_INDI_BPCO_RR_EA_&an_N.;

quit;
```

3.12.4 03_Sejours_d_exacerbation_de_BPCO_en_MCO.sas

```
/*
	*****************************************************************************************************
**********************************	*/
/*

                                                                                                        */
/*			Indicateur 07 - Nombre de séjours l année N

                                                                */
/*

                                                                                                        */
/*
	*****************************************************************************************************
**********************************	*/

*
	*****************************************************************************************************
**********************************;
*	Nombre de séjours MCO d'exacerbation de BPCO codé en DP ou en DAS terminés l année N;

proc sql;

	CREATE TABLE nb_sejours_exa_BPCO_DP AS
		SELECT
			BEN_IDT_ANO,
			COUNT(DISTINCT id_sejour) AS Nb_Sej_EA_BPCO_DP_MCO length = 3
		FROM travail.sejours_exacerbation_&annee_N.
		WHERE reperage = 16 AND variable = "DGN_PAL"
		GROUP BY BEN_IDT_ANO;

	CREATE TABLE nb_sejours_exa_BPCO_DAS AS
		SELECT
			BEN_IDT_ANO,
			COUNT(DISTINCT id_sejour) AS Nb_Sej_EA_BPCO_DAS_MCO length = 3
		FROM travail.sejours_exacerbation_&annee_N.
		WHERE reperage = 16 AND variable = "ASS_DGN"
		GROUP BY BEN_IDT_ANO;

quit;

*
	*****************************************************************************************************
**********************************;
*	On ajoute l'information dans la table res.T_INDI_BPCO_RR_EA_&an_N.;

proc sort data = res.T_INDI_BPCO_RR_EA_&an_N.;
	by BEN_IDT_ANO;
run;

data res.T_INDI_BPCO_RR_EA_&an_N.;
	merge	res.T_INDI_BPCO_RR_EA_&an_N. (in = a)
			nb_sejours_exa_BPCO_DP (in = b)
			nb_sejours_exa_BPCO_DAS (in = c);
	by BEN_IDT_ANO;
	if a;
	if not b then
		Nb_Sej_EA_BPCO_DP_MCO = 0;
	if not c then
		Nb_Sej_EA_BPCO_DAS_MCO = 0;
run;
```

3.12.5 04_Polyarthrite_rhumatoides.sas

```
/*
	****************************************************************************************************
*************************************	*/
/*
																													*/
/*			Indicateur 07 - Patient pris en charge pour polyarthrite rhumatoïdes et maladies apparentées
																							*/
/*			-		Patients en ALD arthrite rhumatoïde active 90 jours après la date index

	*/
/*			-		et/ou patients ayant eu au moins un diagnostic d'arthrite rhumatoïde codé (DP ou DR du séjour) lors
d'un séjour hospitalier en		*/
/*					MCO terminé entre le 1er janvier de l année N-4 et 90 jours après la date index
																													*/
/*			-		et/ou patients ayant eu au moins un diagnostic d'arthrite rhumatoïde codé comme complication ou
morbidité associée (DAS, ou DP		*/
/*					ou DR des UM) lors d'un séjour hospitalier MCO terminé entre 275 jours avant la date index et 90 jours
après la date index		*/
/*			-		et/ou patients ayant eu au moins un diagnostic d'arthrite rhumatoïde codé (MMP, AE des RHS ou DAS)
lors d'un séjour hospitalier		*/
/*					en SSR entre 275 jours avant la date index et 90 jours après la date index.
																													*/
/*

																													*/
/*
	****************************************************************************************************
*************************************	*/

*
	****************************************************************************************************
*************************************;
*	On repère les patients avec une ALD arthrite rhumatoïde active 90 jours après la date index;

proc sql undo_policy = none;

	CREATE TABLE ALD_arthrite AS
		SELECT DISTINCT
			a.BEN_IDT_ANO
		FROM res.T_INDI_BPCO_RR_EA_&an_N. a
			INNER JOIN travail.histo_ALD b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.reperage = 56 AND
			(a.BPCO_RR_EA_Date_index <= b.date_debut <= (a.BPCO_RR_EA_Date_index + 90)
			OR a.BPCO_RR_EA_Date_index <= b.date_fin <= (a.BPCO_RR_EA_Date_index + 90)
			OR (b.date_debut <= (a.BPCO_RR_EA_Date_index + 90) AND (b.date_fin >=
a.BPCO_RR_EA_Date_index + 90 OR b.date_fin = "01JAN1600"d)));

quit;

*
	****************************************************************************************************
*************************************;
*	On repère les patients avec au moins un diagnostic d'arthrite rhumatoïde codé (DP ou DR du séjour) lors d'un séjour
hospitalier en MCO
*	terminé entre le 1er janvier &annee_4N. et 90 jours après la date index;

proc sql undo_policy = none;

	CREATE TABLE hospit_arthrite1 AS
		SELECT DISTINCT
			a.BEN_IDT_ANO
		FROM res.T_INDI_BPCO_RR_EA_&an_N. a
			INNER JOIN travail.polyarthrite_rhum_&annee_4N._&annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE domaine = "MCO" AND variable IN ("DGN_PAL", "DGN_REL") AND table NE "UM"
```

```
                                AND "01JAN&annee_4N."d <= b.date_fin <= (a.BPCO_RR_EA_Date_index + 90);

quit;

*
        ****************************************************************************************************
**********************************;
*       On repère les patients avec au moins un diagnostic d'arthrite rhumatoïde codé comme complication ou morbidité associée
(DAS, ou DP ou DR
*       des UM) lors d'un séjour hospitalier MCO terminé entre 275 jours avant la date index et 90 jours après la date index;

proc sql undo_policy = none;

        CREATE TABLE hospit_arthrite2 AS
                SELECT DISTINCT
                        a.BEN_IDT_ANO
                FROM res.T_INDI_BPCO_RR_EA_&an_N. a
                        INNER JOIN travail.polyarthrite_rhum_&annee_4N._&annee_N1. b
                                ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
                WHERE domaine = "MCO" AND (variable IN ("DGN_PAL", "DGN_REL") AND table = "UM" OR variable =
"ASS_DGN")
                        AND (a.BPCO_RR_EA_Date_index - 275) <= b.date_fin <= (a.BPCO_RR_EA_Date_index + 90);

quit;

*
        ****************************************************************************************************
**********************************;
*       On repère les patients avec au moins un diagnostic d'arthrite rhumatoïde codé (MMP, AE des RHS ou DAS) lors d'un séjour
hospitalier
*       en SSR entre 275 jours avant la date index et 90 jours après la date index;

proc sql undo_policy = none;

        CREATE TABLE hospit_arthrite3 AS
                SELECT DISTINCT
                        a.BEN_IDT_ANO
                FROM res.T_INDI_BPCO_RR_EA_&an_N. a
                        INNER JOIN travail.polyarthrite_rhum_&annee_4N._&annee_N1. b
                                ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
                WHERE domaine = "SSR" AND
                        ((a.BPCO_RR_EA_Date_index - 275) <= b.date_debut <= (a.BPCO_RR_EA_Date_index + 90)
                        OR (a.BPCO_RR_EA_Date_index - 275) <= b.date_fin <= (a.BPCO_RR_EA_Date_index + 90)
                        OR (b.date_debut < (a.BPCO_RR_EA_Date_index - 275) AND b.date_fin > (a.BPCO_RR_EA_Date_index
+ 90)));

quit;

*
        ****************************************************************************************************
**********************************;
*       On concatène toutes les informations;

data patients_arthrite;
        set ALD_arthrite hospit_arthrite1 hospit_arthrite2 hospit_arthrite3;
        length Polyarthrite_Rhuma 3.;
        Polyarthrite_Rhuma = 1;
run;

*
        ****************************************************************************************************
**********************************;
*       On insère l information dans la table de résultats;

proc sort data = patients_arthrite nodupkey;
        by BEN_IDT_ANO;
run;

proc sort data = res.T_INDI_BPCO_RR_EA_&an_N.;
        by BEN_IDT_ANO;
run;
```

```
data res.T_INDI_BPCO_RR_EA_&an_N.;
	merge	res.T_INDI_BPCO_RR_EA_&an_N. (in = a)
			patients_arthrite (in = b);
	by BEN_IDT_ANO;
	if Polyarthrite_Rhuma = . then
		Polyarthrite_Rhuma = 0;
run;

proc delete data = ALD_arthrite hospit_arthrite1 hospit_arthrite2 hospit_arthrite3 patients_arthrite;
run; quit;
```

3.12.6 05_Spondylarthrite_ankylosante.sas

```
/*
	*****************************************************************************************************
*************************************		*/
/*
																											*/
/*			Indicateur 07 - Patient pris en charge pour spondylarthrite ankylosante  et maladies apparentées
																			*/
/*		-		Patients en ALD spondylarthrite ankylosante, ou arthropathie psoriasique ou entéropathique, ou
autres spondylopathie inflammatoire	*/
/*				active 90 jours après la date index

															*/
/*		-		et/ou patients ayant eu au moins un diagnostic de spondylarthrite ankylosante, ou arthropathie
psoriasique ou entéropathique,		*/
/*				ou autres spondylopathie inflammatoire  codé (DP ou DR du séjour) lors d'un séjour hospitalier en
MCO terminé entre le 1er janvier		*/
/*				de l année N-4 et 90 jours après la date index

											*/
/*		-		et/ou patients ayant eu au moins un diagnostic de spondylarthrite ankylosante, ou arthropathie
psoriasique ou entéropathique,		*/
/*				ou autres spondylopathie inflammatoire codé comme complication ou morbidité associée (DAS, ou DP
ou DR des UM) lors d'un séjour	*/
/*				hospitalier MCO terminé entre 275 jours avant la date index et 90 jours après la date index
																			*/
/*		-		et/ou patients ayant eu au moins un diagnostic de spondylarthrite ankylosante, ou arthropathie
psoriasique ou entéropathique,		*/
/*				ou autres spondylopathie inflammatoire codé (MMP, AE des RHS ou DAS) lors d'un séjour hospitalier
en SSR entre 275 jours avant		*/
/*				la date index et 90 jours après la date index.

											*/
/*
																											*/
/*
	*****************************************************************************************************
*************************************		*/

*
	*****************************************************************************************************
*************************************;
*	On repère les patients avec une ALD arthrite rhumatoïde active 90 jours après la date index;

proc sql undo_policy = none;

	CREATE TABLE ALD_spondylarthrite AS
		SELECT DISTINCT
			a.BEN_IDT_ANO
		FROM res.T_INDI_BPCO_RR_EA_&an_N. a
			INNER JOIN travail.histo_ALD b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE b.reperage = 57 AND (a.BPCO_RR_EA_Date_index <= b.date_debut <= (a.BPCO_RR_EA_Date_index + 90)
			OR a.BPCO_RR_EA_Date_index <= b.date_fin <= (a.BPCO_RR_EA_Date_index + 90) OR b.date_debut <=
(a.BPCO_RR_EA_Date_index + 90)
			AND (b.date_fin >= a.BPCO_RR_EA_Date_index + 90 OR b.date_fin = "01JAN1600"d));

quit;

*
	*****************************************************************************************************
*************************************;
*	On repère les patients avec au moins un diagnostic de spondylarthrite ankylosante codé (DP ou DR du séjour) lors d'un
séjour hospitalier en MCO
*	terminé entre le 1er janvier &annee_4N. et 90 jours après la date index;
```

```
proc sql undo_policy = none;

	CREATE TABLE hospit_spondylarthrite1 AS
		SELECT DISTINCT
			a.BEN_IDT_ANO
		FROM res.T_INDI_BPCO_RR_EA_&an_N. a
			INNER JOIN travail.Spondylarthrite_&annee_4N._&annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE domaine = "MCO" AND variable IN ("DGN_PAL", "DGN_REL") AND table NE "UM"
			AND "01JAN&annee_4N."d <= b.date_fin <= (a.BPCO_RR_EA_Date_index + 90);

quit;

*
	**********************************************************************************************************
**********************************;
*	On repère les patients avec au moins un diagnostic de spondylarthrite ankylosante codé comme complication ou morbidité
associée (DAS,
*	ou DP ou DR des UM) lors d'un séjour hospitalier MCO terminé entre 275 jours avant la date index et 90 jours après la date
index;

proc sql undo_policy = none;

	CREATE TABLE hospit_spondylarthrite2 AS
		SELECT DISTINCT
			a.BEN_IDT_ANO
		FROM res.T_INDI_BPCO_RR_EA_&an_N. a
			INNER JOIN travail.Spondylarthrite_&annee_4N._&annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE domaine = "MCO" AND ((variable IN ("DGN_PAL", "DGN_REL") AND table = "UM") OR variable =
"ASS_DGN")
			AND (a.BPCO_RR_EA_Date_index - 275) <= b.date_fin <= (a.BPCO_RR_EA_Date_index + 90);

quit;

*
	**********************************************************************************************************
**********************************;
*	On repère les patients avec au moins un diagnostic de spondylarthrite ankylosante codé (MMP, AE des RHS ou DAS) lors d'un
séjour hospitalier
*	en SSR entre 275 jours avant la date index et 90 jours après la date index;

proc sql undo_policy = none;

	CREATE TABLE hospit_spondylarthrite3 AS
		SELECT DISTINCT
			a.BEN_IDT_ANO
		FROM res.T_INDI_BPCO_RR_EA_&an_N. a
			INNER JOIN travail.Spondylarthrite_&annee_4N._&annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE domaine = "SSR" AND ((a.BPCO_RR_EA_Date_index - 275) <= b.date_debut <=
(a.BPCO_RR_EA_Date_index + 90)
			OR (a.BPCO_RR_EA_Date_index - 275) <= b.date_fin <= (a.BPCO_RR_EA_Date_index + 90)
			OR (b.date_debut < (a.BPCO_RR_EA_Date_index - 275) AND b.date_fin > (a.BPCO_RR_EA_Date_index
+ 90)));

quit;

*
	**********************************************************************************************************
**********************************;
*	On concatène toutes les informations;

data patients_spondylarthrite;
	set ALD_spondylarthrite hospit_spondylarthrite1 hospit_spondylarthrite2 hospit_spondylarthrite3;
	length Spondylarthirte_Anky 3.;
	Spondylarthirte_Anky = 1;
run;
```

```
*
	**********************************************************************************************************
**********************************;
*	On insère l information dans la table de résultats;

proc sort data = patients_spondylarthrite nodupkey;
	by BEN_IDT_ANO;
run;

proc sort data = res.T_INDI_BPCO_RR_EA_&an_N.;
	by BEN_IDT_ANO;
run;

data res.T_INDI_BPCO_RR_EA_&an_N.;
	merge	res.T_INDI_BPCO_RR_EA_&an_N. (in = a)
			patients_spondylarthrite (in = b);
	by BEN_IDT_ANO;
	if Spondylarthirte_Anky = . then
		Spondylarthirte_Anky = 0;
run;

proc delete data = ALD_spondylarthrite hospit_spondylarthrite1 hospit_spondylarthrite2 hospit_spondylarthrite3
patients_spondylarthrite;
run; quit;
```

3.12.7 06_Protheses_et_chirurgie_du_rachis.sas

```
/*
	*****************************************************************************************************
**********************************	*/
/*

*/
/*			Indicateur 07 -

*/
/*			Patients ayant eu au moins un acte CCAM de prise en charge d'une prothèse de hanche codé lors d'un séjour
hospitalier MCO terminé		*/
/*			dans les 90 jours précédant la date index et dans les 90 jours suivant la date index.
*/
/*			Patients ayant eu au moins un acte CCAM de prise en charge d'une prothèse de genou codé lors d'un séjour
hospitalier MCO terminé		*/
/*			dans les 90 jours précédant la date index et dans les 90 jours suivant la date index
*/
/*			Patients ayant eu au moins un acte CCAM de prise en charge de la chirurgie du rachis codé lors d'un séjour
hospitalier MCO terminé		*/
/*			dans les 90 jours précédant la date index et dans les 90 jours suivant la date index
*/
/*

*/
/*
	*****************************************************************************************************
**********************************	*/

*
	*****************************************************************************************************
**********************************	;
*	On repère les patients avec au moins un acte CCAM de prise en charge d'une prothèse de hanche codé lors d'un séjour
hospitalier MCO terminé
*	dans les 90 jours précédant la date index et dans les 90 jours suivant la date index;

proc sql undo_policy = none;

	CREATE TABLE prothese_hanche AS
		SELECT DISTINCT
			a.BEN_IDT_ANO,
			1 AS Prothese_Hanche length = 3
		FROM res.T_INDI_BPCO_RR_EA_&an_N. a
			INNER JOIN travail.prothese_chir_rachis_&annee_1N._&annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE reperage = 8 AND (a.BPCO_RR_EA_Date_index - 90) <= b.date_fin <= (a.BPCO_RR_EA_Date_index + 90)
		ORDER BY a.BEN_IDT_ANO;

quit;

* On insère l information dans la table de résultats;

data res.T_INDI_BPCO_RR_EA_&an_N.;
	merge	res.T_INDI_BPCO_RR_EA_&an_N. (in = a)
			prothese_hanche (in = b);
	by BEN_IDT_ANO;
	if Prothese_Hanche = . then
		Prothese_Hanche = 0;
run;

proc delete data = prothese_hanche;
run; quit;

*
	*****************************************************************************************************
**********************************	;
*	On repère les patients avec au moins un acte CCAM de prise en charge d'une prothèse de genou codé lors d'un séjour
hospitalier MCO terminé
```

```
*		dans les 90 jours précédant la date index et dans les 90 jours suivant la date index;

proc sql undo_policy = none;

		CREATE TABLE Prothese_Genou AS
			SELECT DISTINCT
				a.BEN_IDT_ANO,
				1 AS Prothese_Genou length = 3
			FROM res.T_INDI_BPCO_RR_EA_&an_N. a
				INNER JOIN travail.prothese_chir_rachis_&annee_1N._&annee_N1. b
					ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
			WHERE reperage = 9 AND (a.BPCO_RR_EA_Date_index - 90) <= b.date_fin <= (a.BPCO_RR_EA_Date_index + 90)
			ORDER BY a.BEN_IDT_ANO;

quit;

* On insère l information dans la table de résultats;

data res.T_INDI_BPCO_RR_EA_&an_N.;
		merge	res.T_INDI_BPCO_RR_EA_&an_N. (in = a)
				Prothese_Genou (in = b);
		by BEN_IDT_ANO;
		if Prothese_Genou = . then
			Prothese_Genou = 0;
run;

proc delete data = Prothese_Genou;
run; quit;

*
		**********************************************************************************************************
********************************** ;
*		On repère les patients avec au moins un acte CCAM de prise en charge de la chirurgie du rachis codé lors d'un séjour
hospitalier MCO terminé
*		dans les 90 jours précédant la date index et dans les 90 jours suivant la date index;

proc sql undo_policy = none;

		CREATE TABLE Chir_Rachis AS
			SELECT DISTINCT
				a.BEN_IDT_ANO,
				1 AS Chir_Rachis length = 3
			FROM res.T_INDI_BPCO_RR_EA_&an_N. a
				INNER JOIN travail.prothese_chir_rachis_&annee_1N._&annee_N1. b
					ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
			WHERE reperage = 10 AND (a.BPCO_RR_EA_Date_index - 90) <= b.date_fin <= (a.BPCO_RR_EA_Date_index + 90)
			ORDER BY a.BEN_IDT_ANO;

quit;

* On insère l information dans la table de résultats;

data res.T_INDI_BPCO_RR_EA_&an_N.;
		merge	res.T_INDI_BPCO_RR_EA_&an_N. (in = a)
				Chir_Rachis (in = b);
		by BEN_IDT_ANO;
		if Chir_Rachis = . then
			Chir_Rachis = 0;
run;

proc delete data = Chir_Rachis;
run; quit;
```

3.12.8 07_Pontage_valve_cardiaque_et_stents.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*

                                                                                                                    */
/*			Indicateur 07 -

                                                                                                                */
/*			Patients ayant eu au moins un GHM ou acte CCAM de pontage coronarien codé lors d'un séjour hospitalier en
MCO terminé dans	*/
/*			les 90 jours précédant la date index et dans les 90 jours suivant la date index
                                                                                                                    */
/*			Patients ayant eu au moins un GHM ou acte CCAM de remplacement de valve cardiaque codé lors d'un séjour
hospitalier en MCO	*/
/*			terminé dans les 90 jours précédant la date index et dans les 90 jours suivant la date index
                                                                                                                */
/*			Patients ayant eu au moins un GHM ou acte CCAM de prothèse vasculaire (stents) codé lors d'un séjour
hospitalier en MCO	*/
/*			terminé dans les 90 jours précédant la date index et dans les 90 jours suivant la date index
                                                                                                                */
/*

                                                                                                                    */
/*

                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/

*
	*****************************************************************************************************************
**********************************	;
*	On repère les patients avec au moins un GHM ou acte CCAM de pontage coronarien codé lors d'un séjour hospitalier en
MCO terminé
*	dans les 90 jours précédant la date index et dans les 90 jours suivant la date index;

proc sql undo_policy = none;

	CREATE TABLE Pontage_Coro AS
		SELECT DISTINCT
			a.BEN_IDT_ANO,
			1 AS Pontage_Coro length = 3
		FROM res.T_INDI_BPCO_RR_EA_&an_N. a
			INNER JOIN travail.insuf_coro_aortique_&annee_N._&annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE reperage = 28 AND (a.BPCO_RR_EA_Date_index - 90) <= b.date_fin <= (a.BPCO_RR_EA_Date_index + 90)
		ORDER BY a.BEN_IDT_ANO;

quit;

* On insère l information dans la table de résultats;

data res.T_INDI_BPCO_RR_EA_&an_N.;
	merge	res.T_INDI_BPCO_RR_EA_&an_N. (in = a)
			Pontage_Coro (in = b);
	by BEN_IDT_ANO;
	if Pontage_Coro = . then
		Pontage_Coro = 0;
run;

proc delete data = Pontage_Coro;
run; quit;
```

```
*
	**********************************************************************************************************
**********************************	;
*	On repère les patients ayant eu au moins un GHM ou acte CCAM de remplacement de valve cardiaque codé lors d'un séjour
hospitalier en MCO
*	terminé dans les 90 jours précédant la date index et dans les 90 jours suivant la date index;

proc sql undo_policy = none;

	CREATE TABLE Rempla_Valve_card AS
		SELECT DISTINCT
			a.BEN_IDT_ANO,
			1 AS Rempla_Valve_card length = 3
		FROM res.T_INDI_BPCO_RR_EA_&an_N. a
			INNER JOIN travail.insuf_coro_aortique_&annee_N._&annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE reperage = 29 AND (a.BPCO_RR_EA_Date_index - 90) <= b.date_fin <= (a.BPCO_RR_EA_Date_index + 90)
		ORDER BY a.BEN_IDT_ANO;

quit;

* On insère l information dans la table de résultats;

data res.T_INDI_BPCO_RR_EA_&an_N.;
	merge	res.T_INDI_BPCO_RR_EA_&an_N. (in = a)
			Rempla_Valve_card (in = b);
	by BEN_IDT_ANO;
	if Rempla_Valve_card = . then
		Rempla_Valve_card = 0;
run;

proc delete data = Rempla_Valve_card;
run; quit;

*
	**********************************************************************************************************
**********************************	;
*	On repère les patients ayant eu au moins un GHM ou acte CCAM de prothèse vasculaire (stents) codé lors d'un séjour
hospitalier en MCO
*	terminé dans les 90 jours précédant la date index et dans les 90 jours suivant la date index;

proc sql undo_policy = none;

	CREATE TABLE Stent AS
		SELECT DISTINCT
			a.BEN_IDT_ANO,
			1 AS Stent length = 3
		FROM res.T_INDI_BPCO_RR_EA_&an_N. a
			INNER JOIN travail.insuf_coro_aortique_&annee_N._&annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE reperage = 30 AND (a.BPCO_RR_EA_Date_index - 90) <= b.date_fin <= (a.BPCO_RR_EA_Date_index + 90)
		ORDER BY a.BEN_IDT_ANO;

quit;

* On insère l information dans la table de résultats;

data res.T_INDI_BPCO_RR_EA_&an_N.;
	merge	res.T_INDI_BPCO_RR_EA_&an_N. (in = a)
			Stent (in = b);
	by BEN_IDT_ANO;
	if Stent = . then
		Stent = 0;
run;

proc delete data = Stent;
run; quit;
```

3.12.9 08_Radiotherapie_chimiotherapie.sas

```
/*
	*****************************************************************************************************
**********************************	*/
/*

*/
/*			Indicateur 07 -

*/
/*			Patients ayant eu au moins une séance de radiothérapie codé (DP du séjour) lors d'un séjour hospitalier en MCO
terminé dans les 90	*/
/*			jours précédant la date index et dans les 90 jours suivant la date index
*/
/*			Patients ayant eu au moins une séance de chimiothérapie codé (DP du séjour) lors d'un séjour hospitalier en MCO
terminé dans les 90	*/
/*			jours précédant la date index et dans les 90 jours suivant la date index
*/
/*

*/
/*
	*****************************************************************************************************
**********************************	*/

*
	*****************************************************************************************************
**********************************	;
*	On repère les patients avec au moins une séance de radiothérapie codé (DP du séjour) lors d'un séjour hospitalier en MCO
terminé dans les 90	*/
*	jours précédant la date index et dans les 90 jours suivant la date index;

proc sql undo_policy = none;

	CREATE TABLE Radiotherapie AS
		SELECT DISTINCT
			a.BEN_IDT_ANO,
			1 AS Radiotherapie length = 3
		FROM res.T_INDI_BPCO_RR_EA_&an_N. a
			INNER JOIN travail.radio_chimio_&annee_1N._&annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE reperage = 59 AND (a.BPCO_RR_EA_Date_index - 90) <= b.date_fin <= (a.BPCO_RR_EA_Date_index + 90)
		ORDER BY a.BEN_IDT_ANO;

quit;

* On insère l information dans la table de résultats;

data res.T_INDI_BPCO_RR_EA_&an_N.;
	merge	res.T_INDI_BPCO_RR_EA_&an_N. (in = a)
			Radiotherapie (in = b);
	by BEN_IDT_ANO;
	if Radiotherapie = . then
		Radiotherapie = 0;
run;

proc delete data = Radiotherapie;
run; quit;

*
	*****************************************************************************************************
**********************************	;
*	On repère les patients avec au moins une séance de chimiothérapie codé (DP du séjour) lors d'un séjour hospitalier en MCO
terminé dans les 90	*/
*	jours précédant la date index et dans les 90 jours suivant la date index;

proc sql undo_policy = none;
```

```
	CREATE TABLE Chimiotherapie AS
		SELECT DISTINCT
			a.BEN_IDT_ANO,
			1 AS Chimiotherapie length = 3
		FROM res.T_INDI_BPCO_RR_EA_&an_N. a
			INNER JOIN travail.radio_chimio_&annee_1N._&annee_N1. b
				ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
		WHERE reperage = 58 AND (a.BPCO_RR_EA_Date_index - 90) <= b.date_fin <= (a.BPCO_RR_EA_Date_index + 90)
		ORDER BY a.BEN_IDT_ANO;

quit;

* On insère l information dans la table de résultats;

data res.T_INDI_BPCO_RR_EA_&an_N.;
	merge	res.T_INDI_BPCO_RR_EA_&an_N. (in = a)
			Chimiotherapie (in = b);
	by BEN_IDT_ANO;
	if Chimiotherapie = . then
		Chimiotherapie = 0;
run;

proc delete data = Chimiotherapie;
run; quit;
```

3.12.10 09_Table_finale.sas

```
/*
	*************************************************************************************************************
**********************************	*/
/*
                                                                                                                */
/*			Indicateur 07 : Table finale - Ajout de formats et de labels

			*/
/*
                                                                                                                */
/*
	*************************************************************************************************************
**********************************	*/

data res.T_INDI_BPCO_RR_EA_&an_N.;
	set res.T_INDI_BPCO_RR_EA_&an_N. (keep = BEN_IDT_ANO BPCO_RR_EA_Date_index BPCO_RR_EA_Sej_Index Finess_PMSI
Finess_GEO Dp Dp_classe DA_BPCO
			GHM BPCO_RR_EA_CIBLE Sej_SSR Sej_CM04 Sej_UM54 Sej_UM50 Sej_CM04_UM54
SEJ_CM04_UM50_CSAR_CCAM KINE_AMK_AMC_28 KINE_AMK_AMC_20 KINE_BPC
			KINE_AMK_AMC_AMS_8 KINE_AMS_AMC_AMK_13_5 KINE_AMK_AMC_AMS_9_5_8
KINE_AMK_AMC_AMS_9_5_4 Sej_CSAR_CCAM Sej_CM04_UM50 Sej_CM04_CSAR_CCAM
			Sej_UM50_CSAR_CCAM BPCO_RR_EA_OBS Nb_Sej_EA_BPCO_DP_MCO Nb_Sej_EA_BPCO_DAS_MCO Date_RR
Lieu_RR Date_RR_prec Polyarthrite_Rhuma
			Spondylarthirte_Anky Prothese_Hanche Prothese_Genou Chir_Rachis Pontage_Coro Rempla_Valve_card Stent
Radiotherapie Chimiotherapie);
	label
			BEN_IDT_ANO = "Numéro d'individu"
			BPCO_RR_EA_Date_index = "Date index"
			BPCO_RR_EA_Sej_Index = "Identifiant du séjour index"
			Finess_PMSI = "Finess PMSI ayant réalisé le séjour index"
			Finess_GEO = "Finess géographique du premier RUM"
			Dp = "Diagnostic principal du séjour index"
			Dp_classe = "Diagnostic principal du séjour index"
			DA_BPCO = "BPCO codé en diagnostic secondaire du séjour index"
			GHM = "GHM du séjour index"
			BPCO_RR_EA_CIBLE = "Patient correspondant aux critères d'inclusion et d'exclusion de la population cible"
			Sej_SSR = "Patient de la population cible ayant eu un séjour en SSR dans les 90 jours suivant la date index"
			Sej_CM04 = "Patient ayant eu un séjour en SSR dans la CM 04 dans les 90 jours suivant la date index"
			Sej_UM54 = "Patient de la population cible ayant eu un séjour en SSR et pris en charge dans une UM spécialisée «
Affections respiratoires » dans les 90 jours suivant la date index"
			Sej_UM50 = "Patient de la population cible ayant eu un séjour en SSR et pris en charge dans une UM spécialisée «
SSR indifférenciés ou polyvalents » dans les 90 jours suivant la date index"
			SEJ_CM04_UM54 = "Patient ayant eu un séjour en SSR dans la CM 04 et pris en charge dans une unité médicale
spécialisée « Affections respiratoires » dans les 90 jours suivant la date index"
			SEJ_CM04_UM50_CSAR_CCAM = "Patient ayant eu un séjour en SSR dans la CM 04 et ayant eu un séjour en SSR
dans une unité médicale « SSR indifférenciés ou polyvalents » avec au moins un acte CSARR ou CCAM de la liste de RR dans les 90 jours
suivant la date index"
			KINE_AMK_AMC_28 = "Réadaptation respiratoire kinésithérapique respiratoire pour les patients atteints de
handicap respiratoire dans les 90 jours suivant la date index"
			KINE_AMK_AMC_20 = "Réadaptation respiratoire kinésithérapique respiratoire pour les patients atteints de
handicap respiratoire chronique en prise en charge de groupe de 2 à 4 personnes avec rééducation respiratoire en individuel dans les
90 jours suivant la date index"
			KINE_AMK_AMC_AMS_9_5_8 = "Réadaptation respiratoire codée en coefficient 9.5 avec des coefficients 8 dans
les 90 jours suivant la date index"
			KINE_AMK_AMC_AMS_9_5_4 = "Réadaptation respiratoire codée en coefficient 9.5 avec des coefficients 8/2
(réalisés le même jour et les actes doivent être réalisé tous les 2 soit en ville, soit en MCO, soit en SSR) dans les 90 jours suivant la date
index"
			KINE_AMK_AMC_AMS_8 = "Patient de la population cible ayant bénéficié d'une réadaptation respiratoire
kinésithérapique respiratoire codé en coefficient 8 dans les 90 jours suivant la date index"
			KINE_AMS_AMC_AMK_13_5 = "Rééducation respiratoire et motrice codé avec le coefficient 13.5 dans les 90 jours
suivant la date index"
			KINE_BPC = "Rééducation respiratoire et motrice codé avec l'acte BPC dans les 90 jours suivant la date index"
			Sej_CSAR_CCAM = "Patient de la population cible ayant eu un séjour en SSR au moins un acte CSARR ou CCAM de
la liste de RR dans les 90 jours suivant la date index"
```

```
                    SEJ_CM04_UM50 = "Patient ayant eu un séjour en SSR dans la CM 04 et pris en charge dans une unité médicale
spécialisée « SSR indifférenciés ou polyvalents » dans les 90 jours suivant la date index"
                    Sej_CM04_CSAR_CCAM = "Patient ayant eu un séjour en SSR dans la CM 04 avec au moins un acte CSARR ou
CCAM de la liste de RR dans les 90 jours suivant la date index"
                    Sej_UM50_CSAR_CCAM = "Patient de la population cible ayant eu un séjour en SSR dans une UM « Soins de suite
et de réadaptation indifférenciés ou polyvalents » avec au moins un acte CSARR ou CCAM de la liste de RR dans les 90 jours suivant la
date index"
                    BPCO_RR_EA_OBS = "Patient de la population cible ayant bénéficié d'une réadaptation respiratoire dans les 90
jours suivant la date index"
                    Nb_Sej_EA_BPCO_DP_MCO = "Nombre de séjours MCO d'exacerbation de BPCO codé en DP terminés en
&Annee_N."
                    Nb_Sej_EA_BPCO_DAS_MCO = "Nombre de séjours MCO d'exacerbation de BPCO codé en DAS terminés en
&Annee_N."
                    Date_RR = "Date de réalisation de la réadaptation respiratoire dans les 90 jours suivant la sortie du séjour index"
                    Lieu_RR = "Lieu de réalisation de la réadaptation respiratoire dans les 90 jours suivant la sortie du séjour index"
                    Date_RR_prec = "Date de la dernière RR dans les 365 jours précédant la date d'entrée du séjour index"
                    Polyarthrite_Rhuma = "Patient pris en charge pour polyarthrite rhumatoïdes et maladies apparentées"
                    Spondylarthirte_Anky = "Patient pris en charge pour spondylarthrite ankylosante et maladies apparentées"
                    Prothese_Hanche = "Patient ayant eu au moins un acte CCAM de prise en charge d'une prothèse de hanche"
                    Prothese_Genou = "Patient ayant eu au moins un acte CCAM de prise en charge d'une prothèse de genou"
                    Chir_Rachis = "Patient ayant eu au moins un acte CCAM de prise en charge de la chirurgie du rachis"
                    Pontage_Coro = "Patient ayant eu au moins un diagnostic de pontage coronarien"
                    Rempla_Valve_card = "Patient ayant eu au moins un diagnostic de remplacement de valve cardiaque"
                    Stent = "Patient ayant eu au moins un diagnostic de prothèse vasculaire (stents)"
                    Radiotherapie = "Patient ayant eu au moins une séance de radiothérapie"
                    Chimiotherapie = "Patient ayant eu au moins une séance de chimiothérapie"
                    ;
          format DA_BPCO BPCO_RR_EA_CIBLE Sej_CM04 Sej_UM54 Sej_UM50 Sej_CM04_UM54 SEJ_CM04_UM50_CSAR_CCAM
KINE_AMK_AMC_28 KINE_AMK_AMC_20 KINE_BPC
                    KINE_AMK_AMC_AMS_8 KINE_AMS_AMC_AMK_13_5 KINE_AMK_AMC_AMS_9_5_8
KINE_AMK_AMC_AMS_9_5_4 Sej_CSAR_CCAM Sej_CM04_UM50 Sej_CM04_CSAR_CCAM
                    Sej_UM50_CSAR_CCAM BPCO_RR_EA_OBS Polyarthrite_Rhuma Spondylarthirte_Anky Prothese_Hanche
Prothese_Genou Chir_Rachis Pontage_Coro
                    Rempla_Valve_card Stent Radiotherapie Chimiotherapie SEJ_SSR f_oui_non. Dp_Classe f_DP_classe.
BPCO_RR_EA_Date_index Date_RR
                    Date_RR_prec date9. Lieu_RR f_lieu_soin.;
run;

*
          ************************************************************************************************************
************************************;
*         Nombre de réadaptation respiratoire dans les 90 jours après hospitalisation;

proc sql;

          INSERT INTO flowch.Flow_Chart_BPCO_RR_EA_&an_N.
                    SELECT
                              19,
                              COUNT(DISTINCT BEN_IDT_ANO)
          FROM res.T_INDI_BPCO_RR_EA_&an_N.
          WHERE BPCO_RR_EA_OBS = 1;

          SELECT Nb_patients_ref0 into : N0 FROM travail.ref_flowcharts;

          * Population d étude;
          SELECT N into : N12 FROM flowch.Flow_Chart_BPCO_RR_EA_&an_N. WHERE indicateur = 12;
          * Population cible;
          SELECT N into : N18 FROM flowch.Flow_Chart_BPCO_RR_EA_&an_N. WHERE indicateur = 18;

quit;

data flowch.Flow_Chart_BPCO_RR_EA_&an_N.;
          set flowch.Flow_Chart_BPCO_RR_EA_&an_N.;
          if indicateur in (1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12) then
                    percent = N / &N0.;
          if indicateur in (13, 14, 15, 16, 18) then
                    percent = N / &N12.;
          if indicateur = 19 then
                    percent = N / &N18.;
          format indicateur f_ind_07_flowchart. percent percent7.1;
run;
```

```
* Vérif;
proc sql;

	SELECT COUNT(*) AS nb_lignes, COUNT(DISTINCT BEN_IDT_ANO) AS nb_patients, COUNT(DISTINCT BPCO_RR_EA_Sej_Index)
AS nb_sejours
	FROM res.T_INDI_BPCO_RR_EA_&an_N.;
	* nb_lignes = nb_sejours : OK;

	SELECT COUNT(*) AS nb_lignes, COUNT(DISTINCT BEN_IDT_ANO) AS nb_patients, COUNT(DISTINCT BPCO_RR_EA_Sej_Index)
AS nb_sejours
	FROM res.T_INDI_BPCO_RR_EA_&an_N.
	WHERE BPCO_RR_EA_OBS = 1;
	* nb_lignes = nb_sejours : OK;

quit;

*
	**********************************************************************************************************
**********************************;
*	On ajoute le flag BPCO_RR_EA_CIBLE dans la table res.T_INDI_BPCO_&an_N.;

proc sql;

	ALTER TABLE res.T_INDI_BPCO_&an_N. DROP BPCO_RR_EA_CIBLE;

	ALTER TABLE res.T_INDI_BPCO_&an_N. ADD BPCO_RR_EA_CIBLE INT format = f_oui_non. length = 3
		label = "Patient appartenant à la population cible BPCO_RR_EA";

	UPDATE res.T_INDI_BPCO_&an_N.
		SET BPCO_RR_EA_CIBLE = CASE  WHEN BEN_IDT_ANO IN (SELECT DISTINCT BEN_IDT_ANO FROM
res.T_INDI_BPCO_RR_EA_&an_N.) THEN 1
                                                                                ELSE 0
                                                                                END;

quit;

data res.T_INDI_BPCO_&an_N.;
	set res.T_INDI_BPCO_&an_N.;
	if BPCO_RR_EA_CIBLE = .
		then BPCO_RR_EA_CIBLE = 0;
run;
```

3.13 Suppression des décédés dans les tables finales

3.13.1 01_Patients_decedes_Reperage.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*

*/
/*		Exclusion des patients décédés - Sélection des patients

	*/
/*

*/
/*
	*****************************************************************************************************************
**********************************	*/

%macro verif_dates_deces(var=, sejour=);

	%if &i. = 1 %then
		%do;

			proc sql;

				CREATE TABLE travail.patients_deces_exclus AS
					SELECT DISTINCT
						a.BEN_IDT_ANO
					FROM res.T_INDI_BPCO_&an_N. a
						INNER JOIN res.&table. b
							ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
					WHERE a.BEN_DCD_DTE < b.&var. AND a.BEN_DCD_DTE NE .;

			quit;

		%end;

	%else
		%do;

			proc sql;

				INSERT INTO travail.patients_deces_exclus
					SELECT DISTINCT
						a.BEN_IDT_ANO
					FROM res.T_INDI_BPCO_&an_N. a
						INNER JOIN res.&table. b
							ON a.BEN_IDT_ANO = b.BEN_IDT_ANO
					WHERE a.BEN_DCD_DTE < b.&var. AND a.BEN_DCD_DTE NE .;

			quit;

		%end;

%mend verif_dates_deces;

*****************************************************************************;
* Table T_Indi_BPCO_DG_&an_N.;

%let table = T_Indi_BPCO_DG_&an_N.;
%let sejour_index = ;

%let i = 1;
%verif_dates_deces(var = BPCO_DG_Date_Index, sejour = 0);
```

```
%let i = %sysevalf(&i. + 1);
%verif_dates_deces(var = Date_index_broncho, sejour = 0);

%let i = %sysevalf(&i. + 1);
%verif_dates_deces(var = Date_index_antibio_memejour , sejour = 0);

%let i = %sysevalf(&i. + 1);
%verif_dates_deces(var = Date_index_antibio_365jour, sejour = 0);

%let i = %sysevalf(&i. + 1);
%verif_dates_deces(var = Date_index_substitut, sejour = 0);

%let i = %sysevalf(&i. + 1);
%verif_dates_deces(var = Date_Spiro, sejour = 0);

%let i = %sysevalf(&i. + 1);
%verif_dates_deces(var = Date_EFR, sejour = 0);

%let i = %sysevalf(&i. + 1);
%verif_dates_deces(var = Date_Spiro_Av_365, sejour = 0);

%let i = %sysevalf(&i. + 1);
%verif_dates_deces(var = Date_EFR_Av_365, sejour = 0);

%let i = %sysevalf(&i. + 1);
%verif_dates_deces(var = Date_Spiro_Ap_365, sejour = 0);

%let i = %sysevalf(&i. + 1);
%verif_dates_deces(var = Date_EFR_Ap_365, sejour = 0);

*******************************************************************************;
* Table T_Indi_BPCO_VacG_&an_N.;
%let table = T_Indi_BPCO_VacG_&an_N.;
%let sejour_index = ;

%let i = %sysevalf(&i. + 1);
%verif_dates_deces(var = Date_VacG, sejour = 0);

*******************************************************************************;
* Table T_Indi_BPCO_EFR_SPIRO_&an_N.;
%let table = T_Indi_BPCO_EFR_SPIRO_&an_N.;
%let sejour_index = ;

%let i = %sysevalf(&i. + 1);
%verif_dates_deces(var = Date_Spiro, sejour = 0);

%let i = %sysevalf(&i. + 1);
%verif_dates_deces(var = Date_EFR, sejour = 0);

*******************************************************************************;
* Table T_Indi_Smed_P_EA_&an_N.;
%let table = T_Indi_BPCO_Smed_P_EA_&an_N.;
%let sejour_index = BPCO_Smed_P_EA_Sej_Index;

%let i = %sysevalf(&i. + 1);
%verif_dates_deces(var = BPCO_Smed_P_EA_Date_Index, sejour = 1);

%let i = %sysevalf(&i. + 1);
%verif_dates_deces(var = Date_Contact_Med_180J, sejour = 1);

%let i = %sysevalf(&i. + 1);
%verif_dates_deces(var = Date_Contact_Pneumo_180J, sejour = 1);

*******************************************************************************;
* Table T_Indi_Smed_D_EA_&an_N.;
%let table = T_Indi_BPCO_Smed_D_EA_&an_N.;
%let sejour_index = BPCO_Smed_D_EA_Sej_Index;

%let i = %sysevalf(&i. + 1);
%verif_dates_deces(var = BPCO_Smed_D_EA_Date_Index, sejour = 1);
```

```
%let i = %sysevalf(&i. + 1);
%verif_dates_deces(var = Date_Contact_Pneumo_180J, sejour = 1);

*******************************************************************************;
* Table T_Indi_BPCO_BDLA_EA_&an_N.;
%let table = T_Indi_BPCO_BDLA_EA_&an_N.;
%let sejour_index = BPCO_BDLA_EA_Sej_Index;

%let i = %sysevalf(&i. + 1);
%verif_dates_deces(var = BPCO_BDLA_EA_Date_Index, sejour = 1);

%let i = %sysevalf(&i. + 1);
%verif_dates_deces(var = Date_Broncho_LDA_180J, sejour = 1);

*******************************************************************************;
* Table T_Indi_BPCO_RR_EA_&an_N.;
%let table = T_Indi_BPCO_RR_EA_&an_N.;
%let sejour_index = ;

%let i = %sysevalf(&i. + 1);
%verif_dates_deces(var = BPCO_RR_EA_Date_Index, sejour = 0);

%let i = %sysevalf(&i. + 1);
%verif_dates_deces(var = Date_RR, sejour = 0);

%let i = %sysevalf(&i. + 1);
%verif_dates_deces(var = Date_RR_Prec, sejour = 0);

*******************************************************************************;
* Table T_Indi_Reh_EA_&an_N.;
/*
%let table = T_Indi_BPCO_Reh_EA_&an_N.;
%let sejour_index = BPCO_Reh_EA_Sej_Index;

%let i = %sysevalf(&i. + 1);
%verif_dates_deces(var = BPCO_ReH_EA_Date_Index, sejour = 1);

%let i = %sysevalf(&i. + 1);
%verif_dates_deces(var = ReH_Date, sejour = 1);

%let i = %sysevalf(&i. + 1);
%verif_dates_deces(var = Date_Contact_Med_180J, sejour = 1);

%let i = %sysevalf(&i. + 1);
%verif_dates_deces(var = Date_Contact_Pneumo_180J, sejour = 1);

%let i = %sysevalf(&i. + 1);
%verif_dates_deces(var = Date_RR_180J, sejour = 1);
*/

proc sql;

	SELECT COUNT(DISTINCT BEN_IDT_ANO)
	FROM travail.patients_deces_exclus;

quit;
```

3.13.2 02 _Patients_decedes_maj_Flowchart.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*

*/
/*			Exclusion des patients décédés - Mise à jour des Flowchart

		*/
/*

*/
/*
	*****************************************************************************************************************
**********************************	*/

%macro maj_flowchart(indi_res=, indi_fl=, pop_obs=, num_pop_etude=, num_pop_DC=, num_pop_cible=, num_indic=, sejour=);

	* On compte le nombre de patients décédés;
	proc sql;

		* Nombre de patients décédés parmi la population Cible ou le nombre de séjours;
		SELECT
			%if &sejour. = 0 %then %do; COUNT(DISTINCT BEN_IDT_ANO) INTO : nb_source %end;
			%if &sejour. = 1 %then %do; COUNT(DISTINCT BPCO_&indi_res._Sej_Index) INTO : nb_source %end;
		FROM res.T_INDI_BPCO_&indi_res._&an_N.
		WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM travail.patients_deces_exclus);

		* Nombre de patients décédés parmi la population observée;
		SELECT
			%if &sejour. = 0 %then %do; COUNT(DISTINCT BEN_IDT_ANO) INTO : nb_indi %end;
			%if &sejour. = 1 %then %do; COUNT(DISTINCT BPCO_&indi_res._Sej_Index) INTO : nb_indi %end;
		FROM res.T_INDI_BPCO_&indi_res._&an_N.
		WHERE &pop_obs. = 1 AND BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM travail.patients_deces_exclus);

		* Nombre de patients parmi la population d étude (pour calcul des %);
		SELECT N into : nb_etude
		FROM flowch.Flow_Chart_BPCO_&indi_fl._&an_N.
		WHERE indicateur = &num_pop_etude.;

		* On insère le nouvel indicateur d exclusion;
		INSERT INTO flowch.Flow_Chart_BPCO_&indi_fl._&an_N.
			SELECT
				&num_pop_DC.,
				%if &sejour. = 0 %then %do;
					COUNT(DISTINCT BEN_IDT_ANO),
					COUNT(DISTINCT BEN_IDT_ANO)/&nb_etude.
				%end;
				%if &sejour. = 1 %then %do;
					COUNT(DISTINCT BPCO_&indi_res._Sej_Index),
					COUNT(DISTINCT BPCO_&indi_res._Sej_Index)/&nb_etude.
				%end;
			FROM res.T_INDI_BPCO_&indi_res._&an_N.
			WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM travail.patients_deces_exclus);

	quit;

	* On trie pour visualiser la nouvelle exclusion avant la population cible;
	proc sort data = flowch.Flow_Chart_BPCO_&indi_fl._&an_N.;
		by indicateur;
	run;

	* On met à jour les % : Pour la population Cible, on supprime les patients exclus et on recalcule le pourcentage;
	data flowch.Flow_Chart_BPCO_&indi_fl._&an_N.;
		set flowch.Flow_Chart_BPCO_&indi_fl._&an_N.;
		if indicateur = &num_pop_cible. then
			do;
				N = N - &nb_source.;
```

```
                                        percent = N / &nb_etude.;
                                end;
        run;

        * On sauvegarde la nouvelle population cible;
        proc sql;

                SELECT N INTO : nb_cible
                FROM flowch.Flow_Chart_BPCO_&indi_fl._&an_N.
                WHERE indicateur = &num_pop_cible.;

        quit;

        * On met à jour les % : Pour la population observée, on supprime les patients exclus et on recalcule le pourcentage;
        data flowch.Flow_Chart_BPCO_&indi_fl._&an_N.;
                set flowch.Flow_Chart_BPCO_&indi_fl._&an_N.;
                if indicateur = &num_indic. then
                        do;
                                N = N - &nb_indi.;
                                percent = N / &nb_cible.;
                        end;
        run;

        * Spécifique pour l indicateur de suivi médical précoce : il faut màj les indicateur de suivi MG et de suivi pneumo;
        %if &indi_res. = SMED_P_EA %then
                %do;

                        proc sql;

                                * Nombre de séjours de patients décédés parmi la population observée avec suivi méd
pneumo;
                                SELECT
                                        COUNT(DISTINCT BPCO_SMED_P_EA_Sej_Index) INTO : nb_cible_pneumo
                                FROM res.T_INDI_BPCO_SMED_P_EA_&an_N.
                                WHERE Contact_Pneumo_7j = 1 AND BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM
travail.patients_deces_exclus);

                                * Nombre de patients décédés parmi la population observée avec suivi méd MG;
                                SELECT
                                        COUNT(DISTINCT BPCO_SMED_P_EA_Sej_Index) INTO : nb_cible_MG
                                FROM res.T_INDI_BPCO_SMED_P_EA_&an_N.
                                WHERE Contact_Med_7j = 1 AND BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM
travail.patients_deces_exclus);

                        quit;

                        data flowch.Flow_Chart_BPCO_&indi_fl._&an_N.;
                                set flowch.Flow_Chart_BPCO_&indi_fl._&an_N.;
                                if indicateur = 18 then
                                        do;
                                                N = N - &nb_cible_pneumo.;
                                                percent = N / &nb_cible.;
                                        end;
                                if indicateur = 19 then
                                        do;
                                                N = N - &nb_cible_MG.;
                                                percent = N / &nb_cible.;
                                        end;
                        run;

                %end;

%mend maj_flowchart;

* Indicateur 1 - Diagnostic de BPCO recherché;
%maj_flowchart(
        indi_res = DG,
        indi_fl = DG,
        pop_obs = BPCO_DG_OBS,
        num_pop_etude = 15,
        num_pop_DC = 19,
```

```
		num_pop_cible = 20,
		num_indic = 21,
		sejour = 0
		);

* Indicateur 2 - Vaccin contre la grippe;
%maj_flowchart(
		indi_res = VACG,
		indi_fl = VACG,
		pop_obs = BPCO_VACG_OBS,
		num_pop_etude = 19,
		num_pop_DC = 24,
		num_pop_cible = 25,
		num_indic = 26,
		sejour = 0
		);

* Indicateur 3 - Réalisation d'EFR ou d une spirométrie annuelle;
%maj_flowchart(
		indi_res = EFR_SPIRO,
		indi_fl = EFR_SPIRO,
		pop_obs = BPCO_EFR_SPIRO_OBS,
		num_pop_etude = 15,
		num_pop_DC = 20,
		num_pop_cible = 21,
		num_indic = 22,
		sejour = 0
		);

* Indicateur 4 - Suivi méical précoce après hospitalisation pour exacerbation;
%maj_flowchart(
		indi_res = SMED_P_EA,
		indi_fl = SMED_P_EA,
		pop_obs = BPCO_SMED_P_EA_OBS,
		num_pop_etude = 12,
		num_pop_DC = 16,
		num_pop_cible = 17,
		num_indic = 20,
		sejour = 1
		);

* Indicateur 5 - Suivi méical à distance après hospitalisation pour exacerbation;
%maj_flowchart(
		indi_res = SMED_D_EA,
		indi_fl = SMED_D_EA,
		pop_obs = BPCO_SMED_D_EA_OBS,
		num_pop_etude = 12,
		num_pop_DC = 16,
		num_pop_cible = 17,
		num_indic = 18,
		sejour = 1
		);

* Indicateur 6 - Traitement après hospitalisation pour exacerbation;
%maj_flowchart(
		indi_res = BDLA_EA,
		indi_fl = BDLA_EA,
		pop_obs = BPCO_BDLA_EA_OBS,
		num_pop_etude = 12,
		num_pop_DC = 16,
		num_pop_cible = 17,
		num_indic = 18,
		sejour = 1
		);

* Indicateur 7 - Réadaptation respiratoire après exacerbation;
%maj_flowchart(
		indi_res = RR_EA,
		indi_fl = RR_EA,
		pop_obs = BPCO_RR_EA_OBS,
		num_pop_etude = 12,
```

```
		num_pop_DC = 17,
		num_pop_cible = 18,
		num_indic = 19,
		sejour = 0
		);
/*
* Indicateur 9 - Réhospitalisation pour exacerbation à 6 mois;
%maj_flowchart(
		indi_res = ReH_EA,
		indi_fl = ReH_EA,
		pop_obs = BPCO_ReH_EA_OBS,
		num_pop_etude = 12,
		num_pop_DC = 18.5,
		num_pop_cible = 19,
		num_indic = 20,
		sejour = 1
		);
*/
```

3.13.3 03_Patients_decedes_Suppression_decedes.sas

```
/*
	*****************************************************************************************************************
**********************************	*/
/*
                                                                                                                    */
/*		Exclusion des patients décédés - Suppression dans les tables de résultat
                                                                                                                    */
/*
                                                                                                                    */
/*
	*****************************************************************************************************************
**********************************	*/

proc sql;

	DELETE FROM res.T_INDI_BPCO_BDLA_EA_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM travail.patients_deces_exclus);

	DELETE FROM res.T_INDI_BPCO_DG_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM travail.patients_deces_exclus);

	DELETE FROM res.T_INDI_BPCO_EFR_SPIRO_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM travail.patients_deces_exclus);

	/*
	DELETE FROM res.T_INDI_BPCO_ReH_EA_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM travail.patients_deces_exclus);
	*/

	DELETE FROM res.T_INDI_BPCO_RR_EA_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM travail.patients_deces_exclus);

	DELETE FROM res.T_INDI_BPCO_SMED_D_EA_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM travail.patients_deces_exclus);

	DELETE FROM res.T_INDI_BPCO_SMED_P_EA_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM travail.patients_deces_exclus);

	DELETE FROM res.T_INDI_BPCO_VACG_&an_N.
	WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM travail.patients_deces_exclus);

quit;

proc sql;

	ALTER TABLE res.T_INDI_BPCO_&an_N. DROP exclus_DCD;

	ALTER TABLE res.T_INDI_BPCO_&an_N. ADD exclus_DCD INT length = 3;

	UPDATE res.T_INDI_BPCO_&an_N.
		SET		BPCO_DG_CIBLE = 0,
				BPCO_VACG_PROB_CIBLE = 0,
				BPCO_VACG_DIAG_CIBLE = 0,
				BPCO_VACG_CIBLE = 0,
				BPCO_RR_EA_CIBLE = 0,
				BPCO_BDLA_EA_CIBLE = 0,
				BPCO_SMED_P_EA_CIBLE = 0,
				BPCO_SMED_D_EA_CIBLE = 0,
				/*BPCO_ReH_EA_CIBLE = 0,*/
				BPCO_EFR_SPIRO_CIBLE = 0,
				exclus_DCD = 1
		WHERE BEN_IDT_ANO IN (SELECT BEN_IDT_ANO FROM travail.patients_deces_exclus);

quit;
```

3.13.4 04 _Suppression_Infos_temporaires.sas

```
/*
	*************************************************************************************************************
**********************************	*/
/*

*/
/*			Suppression de toutes les tables, variables et observations inutiles

	*/
/*

*/
/*
	*************************************************************************************************************
**********************************	*/

* Suppression des individus non repérés dans les indicateurs;
proc sql;

	DELETE FROM res.T_INDI_BPCO_&an_N.
	WHERE BPCO_DG_V1_CIBLE = 0 AND BPCO_DG_V2_CIBLE = 0 AND BPCO_VACG_PROB_CIBLE = 0 AND
BPCO_VACG_DIAG_CIBLE = 0 AND BPCO_RR_EA_CIBLE = 0 AND
		BPCO_BDLA_EA_CIBLE = 0 AND BPCO_SMED_P_EA_CIBLE = 0 AND BPCO_SMED_D_EA_CIBLE = 0 AND
BPCO_ReH_EA_CIBLE = 0 AND BPCO_EFR_SPIRO_CIBLE = 0;

quit;

* Suppression des variables de pré-sélection des indicateurs;
proc sql;

	ALTER TABLE res.T_INDI_BPCO_&an_N.
	DROP Nb_hospit_BPCO, Nb_hospit_BPCO_Ind02, ALD_BPCO_septembre, ALD_BPCO_decembre, Nb_ATB, Nb_Tabac,
indicateur_01, indicateur_02a,
		indicateur_02b, indicateur_03, indicateur_04, indicateur_05, indicateur_06, indicateur_07, indicateur_09,
CODE_INSEE, exclus_DCD;

quit;

* Suppression des tables temporaires;
proc delete data =
	pop.indicateur_&an_N.
	travail.contact_MG_PMSI_&annee_N._&annee_N1.
	travail.contact_MT_&annee_N._&annee_N1.
	travail.contact_pneumo_&annee_N._&annee_N1.
	travail.corres_id_patient
	travail.diag_asthme_&annee_1N._&annee_N1.
	travail.fruite_&annee_N.
	travail.histo_ALD
	travail.histo_deces
	travail.insuf_coro_aortique_&annee_N._&annee_N1.
	travail.patients_deces_exclus
	travail.pneumothorax_infarctus_&Annee_1N._&annee_N1.
	travail.polyarthrite_rhum_&annee_4N._&annee_N1.
	travail.prothese_chir_rachis_&annee_1N._&annee_N1.
	travail.ref_flowcharts
	travail.reperage_antibio_&annee_1N._&annee_N1.
	travail.reperage_BDCA_tabac_&annee_2N._&annee_N1.
	travail.reperage_BDLA_&annee_2N._&annee_N1.
	travail.reperage_CCAM_CSARR
	travail.reperage_EFR_spiro_&Annee_2N._&annee_N1.
	travail.reperage_hospit_BPCO_&annee_4N._&annee_N1.
	travail.reperage_med_asthme_&annee_2N._&annee_N1.
	travail.reperage_Oxygeno_&Annee_N._&annee_N1.
	travail.reperage_RR_&annee_1N._&annee_N1.
	travail.RR_sejours_SSR_&Annee_1N._&Annee_N1.
	travail.reperage_Thermo_bronch_&annee_1N._&annee_N1.
	travail.reperage_vacc_grippe_&annee_1N._&annee_N1.
```

```
            travail.reperage_VNI_&annee_1N._&annee_N1.
            travail.sejours_&annee_N._&annee_N1.
            travail.sejours_cible_exacerbation_&annee_1N.
            travail.sejours_cible_exacerbation_&annee_N.
            travail.sejours_cible_exacerbation_&annee_N1.
            travail.sejours_exacerbation_&annee_1N.
            travail.sejours_exacerbation_&annee_N.
            travail.sejours_exacerbation_&annee_N1.
            travail.soin_palliatif_&annee_2N._&annee_N1.
            travail.spondylarthrite_&annee_4N._&annee_N1.
            ;
run; quit;

proc datasets library = orauser nolist kill;
run; quit;
```

4. Annexe

- Patients pris en charge pour polyarthrite rhumatoïdes et maladies apparentées :
    - Patients en ALD arthrite rhumatoïde active 90 jours après la date index
    - ET/OU patients ayant eu au moins un diagnostic d'arthrite rhumatoïde codé (DP ou DR du séjour) lors d'un séjour hospitalier en MCO terminé entre le 1er janvier 2013 et 90 jours après la date index
    - ET/OU patients ayant eu au moins un diagnostic d'arthrite rhumatoïde codé comme complication ou morbidité associée (DAS, ou DP ou DR des UM) lors d'un séjour hospitalier MCO terminé entre 275 jours avant la date index et 90 jours après la date index
    - ET/OU patients ayant eu au moins un diagnostic d'arthrite rhumatoïde codé (MMP, AE des RHS ou DAS) lors d'un séjour hospitalier en SSR entre 275 jours avant la date index et 90 jours après la date index
- Patients pris en charge pour spondylarthrite ankylosante et maladies apparentées :
    - Patients en ALD spondylarthrite ankylosante, ou arthropathie psoriasique ou entéropathique, ou autres spondylopathie inflammatoire active 90 jours après la date index
    - ET/OU patients ayant eu au moins un diagnostic de spondylarthrite ankylosante, ou arthropathie psoriasique ou entéropathique, ou autres spondylopathie inflammatoire codé (DP ou DR du séjour) lors d'un séjour hospitalier en MCO terminé entre le 1er janvier 2013 et 90 jours après la date index
      ET/OU ayant eu au moins un diagnostic de spondylarthrite ankylosante, ou arthropathie psoriasique ou entéropathique, ou autres spondylopathie inflammatoire codé comme complication ou morbidité associée (DAS, ou DP ou DR des UM) lors d'un séjour hospitalier MCO terminé entre 275 jours avant la date index et 90 jours après la date index
    - ET/OU patients ayant eu au moins un diagnostic de spondylarthrite ankylosante, ou arthropathie psoriasique ou entéropathique, ou autres spondylopathie inflammatoire codé (MMP, AE des RHS ou DAS) lors d'un séjour hospitalier en SSR entre 275 jours avant la date index et 90 jours après la date index
- Patients ayant eu au moins un acte CCAM de prise en charge d'une prothèse de hanche codé lors d'un séjour hospitalier MCO terminé dans les 90 jours précédant la date index et dans les 90 jours suivant la date index
- Patients ayant eu au moins un acte CCAM de prise en charge d'une prothèse de genou codé lors d'un séjour hospitalier MCO terminé dans les 90 jours précédant la date index et dans les 90 jours suivant la date index
- Patients ayant eu au moins un acte CCAM de prise en charge de la chirurgie du rachis codé lors d'un séjour hospitalier MCO terminé dans les 90 jours précédant la date index et dans les 90 jours suivant la date index
- Patients ayant eu au moins un diagnostic de pontage coronarien codé (GHM du séjour OU actes CCAM pendant le séjour) lors d'un séjour hospitalier en MCO terminé dans les 90 jours précédant la date index et dans les 90 jours suivant la date index
- Patients ayant eu au moins un diagnostic de remplacement de valve cardiaque codé (GHM du séjour OU actes CCAM pendant le séjour) lors d'un séjour hospitalier en MCO terminé dans les 90 jours précédant la date index et dans les 90 jours suivant la date index
- Patients ayant eu au moins un diagnostic de prothèse vasculaire (stents) codé (GHM du séjour OU actes CCAM pendant le séjour) lors d'un séjour hospitalier en MCO terminé dans les 90 jours précédant la date index et dans les 90 jours suivant la date index

- Patients ayant eu au moins une séance de radiothérapie codé (DAS du séjour ou GHM du séjour) lors d'un séjour hospitalier en MCO terminé dans les 90 jours précédant la date index et dans les 90 jours suivant la date index
- Patients ayant eu au moins une séance de chimiothérapie codé (DAS du séjour ou GHM du séjour) lors d'un séjour hospitalier en MCO terminé dans les 90 jours précédant la date index et dans les 90 jours suivant la date index