Rapports et requêtes 4

A la suite des formations Rapports et requêtes 1, 2 et 3, faisons un suivi des patients ARV:
éligibles, file active, protocoles
perdus de vue
survie et IO sous ARV
ruptures ARV

La version Santia 6/5/2010 ou ultérieure est requise pour les travaux pratiques

formation 5/7/2010

Eligibles, file active, protocoles 1/3

Comment calculer le nombre de patients éligibles au traitement ARV ?

Dans la consultation médicale, le médecin peut indiquer comme conduite à tenir: "A mettre sous ARV".
Comptons les consultations médicales (table **MediCons**) dont la colonne Conduite à tenir (colonne **Cond**) a la valeur "A mettre sous ARV".
En fait, la valeur "A mettre sous ARV" est stockée dans une sous-table: **MediConsCond**, et sa colonne **Desi**. (nous savons depuis les formations précédentes aller chercher une information supplémentaire dans une autre table, avec l'identifiant **Nume** de la table de départ)

```
nb_cons_elig_arv<-SELECT COUNT( DISTINCT MediCons.Nume ) FROM
MediCons,MediConsCond WHERE MediCons.Cond=MediConsCond.Nume AND
MediConsCond.Desi="A mettre sous ARV"
```

Mais c'est le nombre de patients que je voudrais, pas le nombre de consultations.

Ok, voici:

```
nb_doss_elig_arv<-SELECT COUNT( DISTINCT Doss.Nume ) FROM
Doss,MediCons,MediConsCond WHERE Doss.Nume=MediCons.Doss AND
MediCons.Cond=MediConsCond.Nume AND MediConsCond.Desi="A mettre sous
ARV"
```

Il reste à déduire ceux ont été mis sous traitement ARV: les patients éligibles ne sont pas encore sous ARV.

En effet. La colonne **Arv_Desi** de la table **Doss** contient la désignation du traitement ARV. On ajoute la condition que cette désignation est vide:

```
nb_doss_elig_arv<-SELECT COUNT( DISTINCT Doss.Nume ) FROM
Doss,MediCons,MediConsCond WHERE Doss.Nume=MediCons.Doss AND
MediCons.Cond=MediConsCond.Nume AND MediConsCond.Desi="A mettre sous
ARV" AND Doss.Arv_Desi=""
```

Il faut déduire aussi les patients décédés, non ?

C'est juste. On teste que la date de décès est nulle. C'est la colonne **RensDeceDat_** de la table des dossiers:

```
nb_doss_elig_arv<-SELECT COUNT( DISTINCT Doss.Nume ) FROM
Doss,MediCons,MediConsCond WHERE Doss.Nume=MediCons.Doss AND
MediCons.Cond=MediConsCond.Nume AND MediConsCond.Desi="A mettre sous
ARV" AND Doss.Arv_Desi="" AND Doss.RensDeceDat_=0
```

Note: il a d'autres moyens de compter les patients éligibles au traitement ARV, comme le taux de CD4 et/ou le stade clinique OMS.

! Copier/coller le texte de paramétrage et supprimer le saut de ligne éventuel dans la requête SQL.

Eligibles, file active, protocoles 2/3

Comme la colonne **Arv_Desi** de la table **Doss** contient la désignation du traitement ARV, il est plus

facile de calculer le nombre de patients sous ARV cette fois.

Oui, nous n'avons pas besoin de passer par les consultations médicales pour connaître la prescription d'ARV. L'information est copiée dans la table des dossiers:

```
nb_doss_sous_arv<-SELECT COUNT(*) FROM Doss WHERE Arv_Desi<>"" AND
RensDeceDat_=0
```

Pour une répartition par sexe et âge, je suppose qu'on ajoute les conditions déjà vues dans la formation sur l'Accueil.

Exact. Par exemple pour les hommes dont l'âge est inférieur à 15:

```
nb_arv_hm_enf<-SELECT COUNT(*) FROM Doss WHERE Arv_Desi<>"" AND
RensDeceDat_=0 AND RensSexe=1 AND RensAge_<15
```

Pour les femmes dont l'âge est supérieur ou égal à 15:

```
nb_arv_fm_adu<-SELECT COUNT(*) FROM Doss WHERE Arv_Desi<>"" AND
RensDeceDat_=0 AND RensSexe=2 AND RensAge_>=15
```

4/17

Eligibles, file active, protocoles 3/3

Comment compter les patients en 1ère ou 2ème ligne d'ARV ?

La colonne **Arv_Lign** de la table des dossiers reprend le numéro de ligne saisi par le médecin dans la consultation.
Notez que le médecin peut oublier de saisir ce champ alors qu'un traitement ARV est bien prescrit. Dans ce cas, Arv_Lign est égal à zéro. J'ajoute donc la condition inférieur ou égal à 1 pour les 1ères lignes:

```
nb_doss_arv1<-SELECT COUNT(*) FROM Doss WHERE Arv_Desi<>"" AND
RensDeceDat_=0 AND Arv_Lign<=1
```

Pour les 2èmes lignes:

```
nb_doss_arv2<-SELECT COUNT(*) FROM Doss WHERE Arv_Desi<>"" AND
RensDeceDat_=0 AND Arv_Lign=2
```

Si je veux la liste des protocoles ARV utilisés ?

Dans la version actuelle, les rapports paramétrés font des calculs, c'est-à-dire renvoient un nombre. Pour obtenir une liste comme résultat (liste de protocoles ARV, liste de dossiers, etc..), utilisons Paramètres > SQL. On saisit les mêmes requêtes SQL, mais sans la partie "nb_etc<-".
Soit dans le cadre de saisie au milieu de l'écran, soit dans le cadre supérieur pour enregistrer la requête et la rendre accessible aux utilisateurs depuis Analyse (voir documentation > utilisation > paramètres > SQL).

```
SELECT DISTINCT Arv_Desi FROM Doss WHERE Arv_Desi<>""
```

La mention "1ère ligne" ou "2ème ligne" est incluse dans la désignation du protocole.
DISTINCT évite de lister plusieurs fois les même protocoles.

5/17

La colonne Arv_Desi de la table Doss contient la designation du traitement ARV du patient

---

6/17

Perdus de vue (PDV) 1/2

Je voudrais le nombre de patients sous ARV qui sont perdus de vue (PDV).

Accordons-nous sur la définition de PDV: c'est un patient qui n'est pas passé au centre depuis un nombre x de mois et dont le décès n'est pas renseigné.

Pour chaque dossier, il faudrait chercher la dernière consultation médicale, le dernier examen de laboratoire, la dernière arrivée à l'accueil, la dernière dispensation, etc. Cela obligerait SQL à parcourir toutes ces tables pour chaque dossier or la table des dispensations par ex. est souvent très longue.
Pour accélérer le calcul, nous allons procéder autrement et compter seulement les dossiers des consultations récentes, des dispensations récentes, etc. puis déduire les autres, c'est-à-dire les PDV.
Pour cela, on va créer une table temporaire des patients passés récemment au centre. SQL s'occupe de créer une table en mémoire, juste le temps du calcul. Cela n'interfère pas avec les données de la base. Les tables temporaires sont détruites après le calcul.
! On va utiliser les mots clé **CREATE**, **INSERT**, **DROP** sur des tables temporaires. Ne les employez pas sur les données de la base, les pertes et les modifications de données seraient définitives.

Dans mon centre, un patient ARV est considéré comme perdu de vue après 3 mois.

C'est noté. Commençons par créer une table temporaire pour les dossiers passés récemment au centre. Cette table **Temp** ne contient qu'une colonne **Doss** car on n'a besoin que de stocker l'identifiant des dossiers:

```
DROP TABLE IF EXISTS Temp
CREATE TEMPORARY TABLE Temp (Doss INT)
```

DROP supprime la table au cas où elle a déjà été créée.
Doss INT indique la colonne à créer dans la table et son type de valeur. Ici, INT (= integer) désigne un nombre entier. C'est le type qui convient pour l'identifiant de dossier.

Ensuite, on remplit la table **Temp** avec les dossiers qui ont eu une consultation médicale ces 90 derniers jours:

```
INSERT INTO Temp (Doss) SELECT DISTINCT Doss FROM MediCons WHERE
Dat_>=DATE_SUB(CURDATE(),INTERVAL 90 DAY)
```

Oh, encore de nouveaux mots SQL, on part en terre inconnue...

INSERT INTO ajoute des données à une table. Là, on ajoute des données à la table Temp dans sa colonne Doss.
Ce qu'on ajoute est défini par SELECT: c'est la colonne Doss de la table MediCons. On élimine les doublons de Doss avec DISTINCT.
De plus, une condition est posée sur la date de la table MediCons des consultations:
CURDATE() demande à SQL la date du jour (= current date)
DATE_SUB(a,b) soustrait b à a. Ici, on soustrait 90 jours (= day) à la date du jour.

Comme on va utiliser plusieurs fois le délai de 90 jours (pour l'accueil, les dispensations, etc.), je demande à SQL de le mémoriser une fois au début, à l'aide du mot clé **SET**. Le nom des valeurs mémorisées par SQL doit toujours commencer par "@". Je modifie mon texte pour utiliser dorénavant **@nb_jour** au lieu de **90**:

```
SET @nb_jour=90
INSERT INTO Temp (Doss) SELECT DISTINCT Doss FROM MediCons WHERE
Dat_>=DATE_SUB(CURDATE(),INTERVAL @nb_jour DAY)
```

7/17

Perdus de vue (PDV) 2/2

Donc la table **Temp** contient maintenant une copie de la liste des dossiers qui ont assisté à une consultation médicale dans les 90 derniers jours ?

Oui, cette liste est une sorte de copie en mémoire des données de la base, juste pour la durée du calcul.
Il faut compléter la liste pour ajouter les dossiers qui ont eu une consultation sociale, psychologique ou une séance d'éducation thérapeutique (ETP):
SociCons consultations sociales
Psy_Cons consultations psychologiques
ObseCons séances d'ETP

```
INSERT INTO Temp (Doss) SELECT DISTINCT Doss FROM SociCons WHERE
Dat_>=DATE_SUB(CURDATE(),INTERVAL @nb_jour DAY)
INSERT INTO Temp (Doss) SELECT DISTINCT Doss FROM Psy_Cons WHERE
Dat_>=DATE_SUB(CURDATE(),INTERVAL @nb_jour DAY)
INSERT INTO Temp (Doss) SELECT DISTINCT Doss FROM ObseCons WHERE
Dat_>=DATE_SUB(CURDATE(),INTERVAL @nb_jour DAY)
```

Passons à l'ajout des dossiers qui sont passés à l'accueil quel que soit le motif. On a déjà vu la table **Entr** pour l'Accueil dans la formation 3.

```
INSERT INTO Temp (Doss) SELECT DISTINCT Doss FROM Entr WHERE
ArriHoro>=DATE_SUB(CURDATE(),INTERVAL @nb_jour DAY)
```

Ajoutons les dossiers dont un examen biologique a été saisi. Les examens de laboratoire sont stockés dans cette même table **Entr**. Mais le patient n'est pas forcément passé par l'accueil: certains centres ne gère pas l'écran Accueil, et dans tous les cas, le laboratoire peut faire une saisie directe. On pose donc une condition sur la désignation de l'examen **LaboDesi** et sa date **LaboDat_** dans la table.

```
INSERT INTO Temp (Doss) SELECT DISTINCT Doss FROM Entr WHERE
LaboDesi<>"" AND LaboDat_>=DATE_SUB(CURDATE(),INTERVAL @nb_jour DAY)
```

Enfin, on ajoute les patients qui ont reçu une dispensation de médicaments. La table **Item** stocke les entrées/sorties de produits pharmaceutiques. Lorsque la colonne **Dest** contient la valeur 6, il s'agit d'une délivrance à un patient:

```
INSERT INTO Temp (Doss) SELECT DISTINCT Doss FROM Item WHERE
Dat_>=DATE_SUB(CURDATE(),INTERVAL @nb_jour DAY) AND Dest=6
```

Donc maintenant la table **Temp** contient tous les patients passés au centre quelque soit le motif. Est-ce que les patients qui ne sont pas dans cette table sont perdus de vue ?

Oui, s'il ne sont pas décédés.
Ci-dessous, on compte les patients sous ARV non décédés, dont on soustrait ceux de la table Temp.
LEFT JOIN...ON associé à **IS NULL** permet de soustraire à la table Doss les lignes de la table Temp (on verra l'usage de ces mots SQL une autre fois en détail).
D'autre part, je mets la condition DISTINCT sur le résultat car la table Temp contient autant de fois les dossiers passés au centre plusieurs fois:

```
nb_arv_pdv<-SELECT COUNT( DISTINCT Nume ) FROM Doss LEFT JOIN Temp ON
Doss.Nume=Temp.Doss WHERE Temp.Doss IS NULL AND Doss.Arv_Desi<>"" AND
RensDeceDat_=0
```

Et comme d'habitude pour un tri par sexe et âge (ici, garçons de moins de 15 ans):

```
nb_arv_pdv<-SELECT COUNT( DISTINCT Nume ) FROM Doss LEFT JOIN Temp ON
Doss.Nume=Temp.Doss WHERE Temp.Doss IS NULL AND Doss.Arv_Desi<>"" AND
RensDeceDat_=0 AND RensSexe=1 AND RensAge_<15
```

Pour retirer aussi les patients en voyage:

```
AND RensVoya<>1
```

On peut écrire "Doss.RensSexe" etc. au lieu de "RensSexe", mais il n'y a pas d'ambiguité avec la table Temp.

8/17

On peut soustraire aux lignes d'une table A les lignes d'une table B

9/17

Survie sous ARV

Les indicateurs UNGASS comptent le nombre de patients vivants 6, 12 et 24 mois après le début du traitement. Comment calculer cela ?

Il faut d'abord trouver la date de début du traitement.
Une solution consisterait à chercher la consultation médicale où le médecin a sélectionné "Début" dans le champ Modification traitement ARV. Mais cette valeur n'est pas toujours bien renseignée par les utilisateurs.
Pour avoir un calcul plus fiable, nous allons tout simplement chercher si une consultation avec prescription d'ARV a eu lieu il y a plus de 6 mois chez les patients vivants.

```
nb_surv6<-SELECT COUNT( DISTINCT Doss.Nume ) FROM Doss,MediCons WHERE
Doss.Nume=MediCons.Doss AND (MediCons.Arv0Prsc<>"" OR
MediCons.Arv1Prsc<>"" OR MediCons.Arv2Prsc<>"" OR MediCons.Arv3Prsc<>"")
AND MediCons.Dat_<=DATE_SUB(CURDATE(),INTERVAL 180 DAY)
```

Explications por favor ?

On sait chercher une information supplémentaire dans une autre table. Ici, à partir de la table des dossiers, on cherche les consultations médicales du dossier, avec **Doss.Nume=MediCons.Doss**.
La date de la consultation **MediCons.Dat_** doit être antérieure à 6 mois. On a vu précédemment (perdus de vue) comment déduire (DATE_SUB) des jours (180 DAY) de la date courante (CURDATE).
On ajoute encore une condition: la consultation doit avoir donné lieu à une prescription d'ARV. Les colonnes **Arv0Prsc** à **Arv3Prsc** contiennent les molécules ARV prescrites. La prescription d'une seule molécule suffit (OR = ou).

Mais ce n'est pas terminé: on vérifie que le patient est sous ARV et qu'il n'est pas décédé. Je complète donc:

```
nb_surv6<-SELECT COUNT( DISTINCT Doss.Nume ) FROM Doss,MediCons WHERE
Doss.Nume=MediCons.Doss AND (MediCons.Arv0Prsc<>"" OR
MediCons.Arv1Prsc<>"" OR MediCons.Arv2Prsc<>"" OR MediCons.Arv3Prsc<>"")
AND MediCons.Dat_<=DATE_SUB(CURDATE(),INTERVAL 180 DAY) AND
Doss.Arv_Desi<>"" AND Doss.RensDeceDat_=0
```

Idem pour la survie à 12 et 24 mois, j'imagine.

Oui, on remplace simplement 180 par 360 (= 12 mois) ou 720 (= 24 mois).

On peut ajouter les habituelles conditions sur le sexe et l'âge (ici, filles de moins de 15 ans):

```
AND Doss.RensSexe=2 AND Doss.RensAge_<15
```

Note:
- On aurait pu se baser sur les dispensations pour déterminer le début du traitement, plutôt que les consultations. Mais certains centres ne gèrent pas de pharmacie ou peuvent ne pas délivrer à un moment donné.
- Le rapport d'activité (Analyse > Rapport d'activité) utilise la valeur "Début" du champ Modification traitement ARV.

10/17

Parfois, il existe plusieurs approches pour calculer une information

---

11/17

IO sous ARV 1/2

Je voudrais le nombre de patients sous ARV depuis plus de 6 mois qui font une infection opportuniste (IO).

Qui font une infection opportuniste actuellement ? On va considérer les IO saisies dans les consultations médicales des 60 derniers jours par exemple.

On sait déjà compter les patients sous ARV depuis plus de 6 mois et non décédés (voir Survie sous ARV):

```
nb_arv6<-SELECT COUNT( DISTINCT Doss.Nume ) FROM Doss,MediCons WHERE
Doss.Nume=MediCons.Doss AND (MediCons.Arv0Prsc<>"" OR
MediCons.Arv1Prsc<>"" OR MediCons.Arv2Prsc<>"" OR MediCons.Arv3Prsc<>"")
AND MediCons.Dat_<=DATE_SUB(CURDATE(),INTERVAL 180 DAY) AND
Doss.Arv_Desi<>"" AND Doss.RensDeceDat_=0
```

Je me souviens du rapport IO dans la formation 2. On lisait la colonne **ConcCase** de la table **MediCons** pour obtenir la valeur des cases à cocher d'infections opportunistes.

Oui, dès qu'une case IO est cochée (classification OMS, palu..), la valeur de **ConcCase** change.

Toutefois, on a un petit souci. Comme on lit les consultations médicales de plus de 6 mois (pour déterminer si le patient est sous ARV depuis plus de 6 mois), on ne peut pas demander en même temps les consultations de moins de 2 mois (qui contiennent une IO récente). C'est contradictoire: une consultation médicale ne peut être à la fois ancienne (plus de 6 mois) et récente (moins de 2 mois).
On va donc passer par une table temporaire pour stocker d'abord les dossiers sous ARV depuis plus de 6 mois.
Je ne réexplique pas cette partie car nous l'avons vue précédemment pour Perdus de vue:

```
DROP TABLE IF EXISTS Temp
CREATE TEMPORARY TABLE Temp (Doss INT)
INSERT INTO Temp (Doss) SELECT DISTINCT MediCons.Doss FROM Doss,MediCons
WHERE Doss.Nume=MediCons.Doss AND (MediCons.Arv0Prsc<>"" OR
MediCons.Arv1Prsc<>"" OR MediCons.Arv2Prsc<>"" OR MediCons.Arv3Prsc<>"")
AND MediCons.Dat_<=DATE_SUB(CURDATE(),INTERVAL 180 DAY) AND
Doss.Arv_Desi<>"" AND Doss.RensDeceDat_=0
```

12/17

IO sous ARV 2/2

Ensuite ?

Ensuite, je crée une deuxième table temporaire pour les dossiers qui ont une consultation avec IO dans les 60 derniers jours. La condition est que **ConcCase** ne soit pas vide (c'est-à-dire qu'au moins une case IO soit cochée):

```
DROP TABLE IF EXISTS Temp2
CREATE TEMPORARY TABLE Temp2 (Doss INT)
INSERT INTO Temp2 (Doss) SELECT DISTINCT Doss FROM MediCons WHERE
Dat_>DATE_SUB(CURDATE(),INTERVAL 60 DAY) AND ConcCase>0 AND SUBSTR
(ConcCase,1,1)<>1
```

Il y a une double condition sur ConcCase: au moins une case est cochée, en dehors de la case n°1 "Asymptomatique" (voir Formation 2).

Pour finir, on compte le nombre de dossiers de **Temp** (ARV depuis plus de 6 mois) qui se trouvent aussi dans **Temp2** (IO depuis moins de 2 mois), sans les doublons:

```
nb_arv6_io2<-SELECT COUNT( DISTINCT Temp.Doss ) FROM Temp,Temp2 WHERE
Temp.Doss=Temp2.Doss
```

Et pour une sélection par sexe et âge des patients ?

Il faut simplement ajouter les conditions (exemple: AND Doss.RensSexe=2 AND Doss.RensAge_>=15) dans la requête qui remplit Temp (cf page précédente) car elle lit la table **Doss** et peut accéder à ces colonnes.

Note: si le médecin saisit l'IO en tapant un texte au lieu de cocher les cases prévues, l'IO ne sera pas comptabilisée.

13/17

On peut créer et utiliser plusieurs tables temporaires

14/17

Ruptures ARV 1/2

Pour finir, comment compter les patients dont le traitement ARV a subi des ruptures de stock ?

Il y a sûrement plusieurs approches.
Je propose de chercher les prescriptions (= ordonnances) qui n'ont pas été servies par une délivrance.
Quand on active Paramètres > Application > Délivrer uniquement sur prescription (et on peut cocher Autoriser la dizaine d'unité supérieure), les quantités délivrées (écran dispensation) sont enregistrées à côté des quantités prescrites. On peut alors faire la comparaison dans les données.

Voici une première étape: le nombre de consultations, des 12 derniers mois par exemple, dont un ARV prescrit n'a pas été délivré:

```
nb_cons_rupt<-SELECT COUNT(*) FROM MediCons WHERE ((Arv0Prsc<>"" AND
(Arv0Serv IS NULL OR Arv0Serv=0)) OR (Arv1Prsc<>"" AND (Arv1Serv IS NULL
OR Arv1Serv=0)) OR (Arv2Prsc<>"" AND (Arv2Serv IS NULL OR Arv2Serv=0))
OR (Arv3Prsc<>"" AND (Arv3Serv IS NULL OR Arv3Serv=0))) AND
Dat_>DATE_SUB(CURDATE(),INTERVAL 360 DAY)
```

Je m'explique:
Dans une consultation, on peut prescrire 4 molécules (ou associations de molécules), de **Arv0** à **Arv3**.
Arv0Prsc est la molécule prescrite. **Arv0Serv** est le nombre de comprimés délivrés. On pose comme condition que si la molécule prescrite n'est pas vide (Arv0Prsc<>"") et que le nombre délivré est nul (Arv0Serv IS NULL OR Arv0Serv=0. Ici, il faut tester ces 2 cas), la prescription n'est pas servie.
Voyez l'imbrication des conditions, à l'aide de parenthèses, pour chaque ARV.

15/17

Ruptures ARV 2/2

On vient de sortir le nombre de consultations non servies en ARV, mais ce n'est pas le nombre de patients ayant subi une rupture. Encore moins répartis par sexe et âge...

Ok, reprenons le code pour chercher en même temps les dossiers. On peut alors ajouter une condition sur le sexe et l'âge:

```
nb_rupt_hm_enf<-SELECT COUNT( DISTINCT Doss.Nume ) FROM Doss,MediCons
WHERE Doss.Nume=MediCons.Doss AND ((Arv0Prsc<>"" AND (Arv0Serv IS NULL
OR Arv0Serv=0)) OR (Arv1Prsc<>"" AND (Arv1Serv IS NULL OR Arv1Serv=0))
OR (Arv2Prsc<>"" AND (Arv2Serv IS NULL OR Arv2Serv=0)) OR (Arv3Prsc<>""
AND (Arv3Serv IS NULL OR Arv3Serv=0))) AND MediCons.Dat_>DATE_SUB
(CURDATE(),INTERVAL 360 DAY) AND Doss.RensSexe=1 AND Doss.RensAge_<15
```

Notes:
- On pourait préciser "MediCons.Arv0Prsc" au lieu de "Arv0Prsc" etc, mais il n'y a pas d'ambiguité avec les colonnes de la table Doss.
- On liste les molécules prescrites qui n'ont pas été servies du tout. Les délivrances partielles ne sont pas comptées. Par ailleurs, la non délivrance peut avoir une autre cause que la rupture de stock (le patient n'est pas passé à la pharmacie, etc.).

16/17

Il faut parfois chercher un moyen astucieux de calculer l'information, comme le montre l'exemple des ruptures de stock

---

17/17

Finalement, quel est l'intérêt des tables temporaires ?

Les tables temporaires peuvent accélérer les calculs. Mais surtout, elles aident à réaliser certaines requêtes complexes. Il suffit alors de décomposer le traitement en plusieurs étapes.
La manipulation des tables temporaires est identique à celle des tables de la base de données, avec les mêmes mots SQL. Si c'est sans risque pour les tables temporaires, gare aux tables de la base !

Merci de votre attention,

A bientôt pour la suite !
