M2106

Programmation et administration des bases de données

COURS 1 :

RAPPELS SGBD ET SQL.

Février 02, 2017

BELFODIL Adnene – BENDIMERAD Anes

Lyon 1

* Cours extraits principalement des cours de Guillaume Cabanac, Fouad Dahak, Yacine Bouhabel, Amrouche Karima, Richard Grin

SOMMAIRE

SOMMAIRE

OBJECTIFS ET CONTENU DU MODULE

M2106

Objectifs

- Approfondissement du langage SQL.
- Maîtriser le langage procédural PL/SQL.
- Gérer des bases de données relationnelles à travers un langage de troisième génération.
- Initiation à l'administration des SGBD.

Contenu

- Le langage SQL : révisions et approfondissements (requêtes complexes) (~4H)
- L'extension procédurale de SQL : le langage PL/SQL (~32H)
- L'accès aux bases de données via JDBC.(~6H)
- Initiation à l'administration des SGBD : vues, utilisateurs et privilèges (~2H)

SOMMAIRE

DÉROULEMENT DES SÉANCES

Partie	Durée	Objectifs
Cours	25% - 30 %	Rappels, explication des notions ...
TD	25%	1 à 2 Exercices dirigées avec solution
TP	45% - 50 %	Exercices avec compte rendu à la fin de séance -> *@adnene.belfodil@insa-lyon.fr*

Contrôle des connaissances

DS de promo (2/3) + DS de groupe (1/3)

Malus-Bonus : -1.5 to +1.5 (*Life is too short to give Malus :-)*)

SOMMAIRE

A. Relations et SGBD

1. Relations : Schéma et extensions
2. Clé primaire et clé étrangère
3. Objectifs d'un SGBD

A1 – RELATIONS ET EXTENSIONS)

Définition : une relation R est définie par :

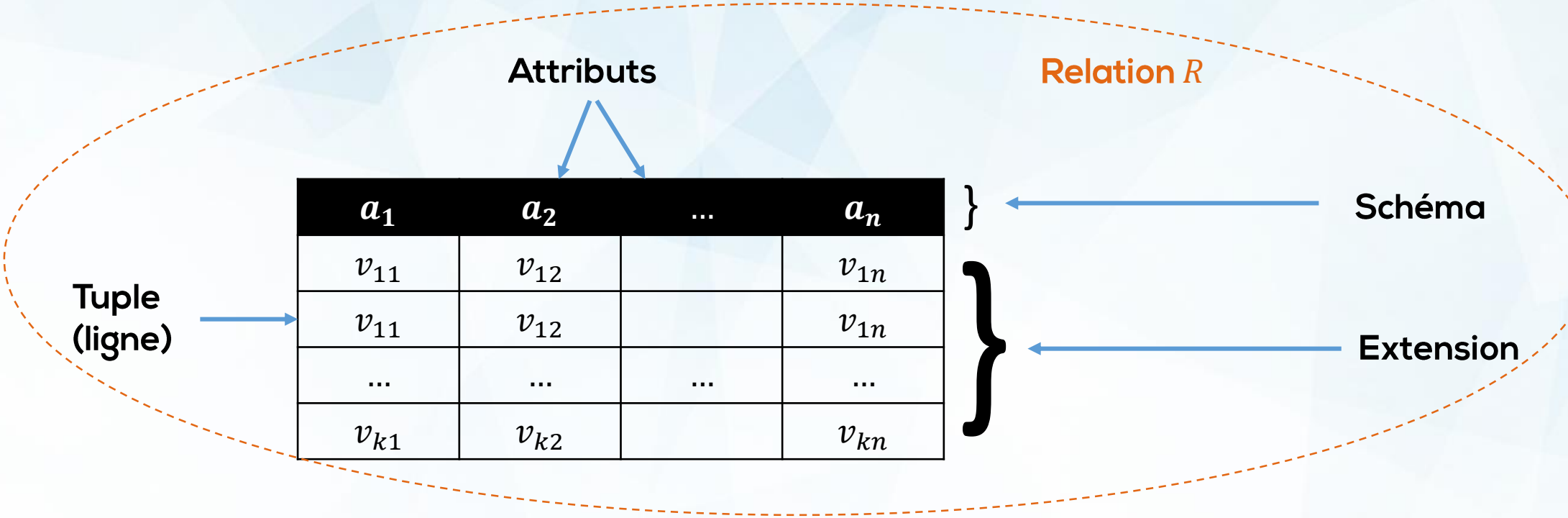
1- un schéma composé d'une liste de n attributs typés :

$$R = [a_1, a_2, \dots, a_n]$$

2- une extension composé d'un ensemble de tuples (n-uplets) dont l'ordre n'a pas d'importance

A1 – RELATIONS ET EXTENSIONS)

On peut représenter une relation R sous forme tabulaire :



A1 – RELATIONS ET EXTENSIONS)

Exemple : Soit la relation

MEPs = [*ep_id, full_name, national_party, eu_group, country, gender, dateofbirth*]

Typage des attributs :

- **ep_id** : nombre entier unique obligatoirement renseigné
- **Full_name** : chaine de 100 caractères au maximum
- **National_party** : chaine de 100 caractères au maximum
- **Eu_group** : chaine de 100 caractères au maximum
- **Country** : chaine de 30 caractères au maximum
- **Gender** : 1 caractère
- **Dateofbirth** : Date (facultative)

A1 – RELATIONS ET EXTENSIONS)

Exemple des tuples de la relation MEPs:

ep_id	Full_name	National_party	Eu_group	country	gender	Date_of_birth
1204	Alain LAMASSOURE	LR	PPE	France	M	
1023	Jean-Marie LE PEN	FN	(Null)	France	M	1928-06-20
124744	Isabelle THOMAS	PS	S&D	France	W	1961-11-26
124740	Joélle BERGERON	-	EFDD	France	W	

A2 – CLÉ PRIMAIRE ET CLÉ ÉTRANGÈRE

Définition : une clé primaire (PK) :

1- doit être définie pour chaque relation

2- porte sur un ou plusieurs attributs (clé simple vs composée)

3- a les caractéristiques suivantes :

- unique : un tuple est identifié sans ambiguïté
- non-nulle : tous ses constituants sont renseignés

Exemple : Soit la relation votes:

Votes = [*vote_id*, *themes*, *vote_date*…]

Clé primaire (souligné)

A2 – CLÉ PRIMAIRE ET CLÉ ÉTRANGÈRE

Définition : une clé étrangère (FK) :

1- garantit l'intégrité référentielle,

2- a une source composée d'un ou plusieurs attributs s_i

3- a une cible composée d'un ou plusieurs attributs c_j

4- est matérialisée par une propriété d'inclusion : $s_i \subseteq c_j$

Exemple : Soit la relation `position`:

`position = [#vote_id, #ep_id, pos…]`

Clé primaire composée de deux clés étrangère (# + souligné)

A3 – OBJECTIFS D'UN SGBD

Définition d'un base de données

Une base de données (BD) est un ensemble structuré de données enregistrées sur des supports accessibles par l'ordinateur pour satisfaire simultanément plusieurs utilisateurs de manière sélective et en un temps opportun.

A3 – OBJECTIFS D'UN SGBD

Objectifs :

1. Élimination de la redondance des donnée
2. Centralisation et organisation correcte des données
3. Apports du Système de Gestion de Bases de Données (SGBD)
   - Factorisation des modules de contrôle des applications
   - Interrogation, cohérence, partage, gestion des pannes. . .
   - Administration facilitée des données

Exemples des SGBDRs

- Oracle
- mySQL
- Postgresql
- SQLite

SOMMAIRE

B - RAPPELS SQL((LDD, LMD, LID)

B. Relations et SGBD

1. LDD : Langage de définition de données (Opération sur les tables)
2. LMD : Langage de manipulation de données (Opération sur les lignes)
3. LID : Langage d'interrogation de données (Requêtage des données)
4. LCD : Langage de contrôle sur les données(gestion des accès multiutilisateurs aux données)
5. LCT : Langage de contrôle des transactions (validation et annulation des transactions)

LDD : Langage de définition de données

Partie de SQL qui permet de créer des bases de données, des tables, des index, des contraintes ..., Elle traite de la création des schémas de bases de données.

(Create, Alter, Rename, Drop …)

B1 – LDD : CREATE / DROP DATABASE

Syntaxe (réduite) – Création d'une base

```
CREATE DATABASE [IF NOT EXISTS] $db_name
USE $db_name
```

Syntaxe (réduite) – Suppression d'une base

```
DROP DATABASE [IF EXISTS] $db_name
```

Exemple

```
> CREATE DATABASE IF NOT EXISTS 'VOTES'
> USE 'VOTES'
> DROP DATABASE 'VOTES'
```

B1 – LDD : CREATE / DROP TABLE

Syntaxe (réduite) – Création d'une Table

```
CREATE TABLE [IF NOT EXISTS] $table_name (
        $column_name $type [k times] ,
        Constraint $cstr_name ... ,
);
```

Syntaxe (réduite) – Supression d'une Table

```
DROP TABLE [IF NOT EXISTS] $table_name
```

B1 – LDD : CREATE / DROP TABLE

Exemple

```
CREATE TABLE 'MEPS' (
    ep_id number(10),
    full_name varchar2(100),
    national_party varchar2(100) default '-',
    eu_group varchar2(100),
    country varchar2(30),
    gender char(1),
    dateofbirth date
)
```

valeur par défaut

Types élémentaires

Remarque : les contraintes sont nécessaires pour assurer l'intégrité de la base

B1 – LDD : CONTRAINTES

Il existe 5 types de contraintes

PK : Les valeurs d'une colonne **Primary Key** (clé primaire) sont uniques, non nulles et indexées.

FK : Les valeurs d'une colonne **Foreign Key** (clé étrangère) sont incluses dans les valeurs de la cible (qui est une colonne Unique).

UN : Les valeurs d'une colonne **Unique** n'ont pas de doublons (null autorisés)

CK : Les valeurs d'une colonne **Check** respectent une condition booléenne spécifiée.

NN : Les valeurs d'une colonne **Not Null** sont toutes renseignées (null interdit).

Syntaxe (réduite) – Création d'une Table avec contraintes

```
CREATE TABLE [IF NOT EXISTS] $table_name (
        $column_name $type [k times] ,
        Constraint $pk_cstr primary key (column_1, …),
        Constraint $fk_cstr foreign key (column_1, …) references $table_name,
        Constraint $un_cstr unique (column_1,…)
        Constraint $ck_cstr check ($condition_expression)
        Constraint $nn_cstr not null (column_name)
);
```

Exemple

```
CREATE TABLE ‘VOTES’ (
    v_id number(10),
    themes varchar2(100),
    vote_date date constraint nn_vote_date not null,
     constraint pk_v_id primary key (v_id)
);
```

Exemple

```
CREATE TABLE ‘POSITION’ (
    v_id number(10),
    ep_id number(10),
    Postion varchar(10),
    constraint ck_position check (Postion in (‘for’,’against’,’abstain’)),
    constraint fk_v_id foreign key (v_id) references votes,
    constraint fk_ep_id foreign key (ep_id) references meps,
    constraint pk_vep primary key (v_id,ep_id)
);
```

Remarque : création de la table en plusieurs étapes

Il est possible d'ajouter les contraintes ou de les modifier après la création de la table en utilisation la commande :

`ALTER TABLE $table_name add` pour ajoute des contraintes ou des nouvelles colonnes

`ALTER TABLE $table_name modify` pour modifier les définition d'une colonne ou d'une contrainte

`ALTER TABLE $table_name drop` pour supprimer une colonne ou une contrainte

LMD : Langage de définition de données

Partie de SQL qui permet de traiter les données (*LID généralement est compris sous LMD*). Il permet l'insertion, la suppression et la modification des données.

`(Insert, Delete, Update)`

B2 – LMD : INSERTION DE DONNÉES

Syntaxe (réduite) – insertion

```
Insert into $table_name [(coln1,coln2 … , colnn)]
Values (val_1,val_2…val_n)
```

Exemple

```
> Insert into position (v_id,ep_id,position)
  values ('100','2107','for')
```

Remarque : insertion avec un résultat d'interrogation

Il est possible d'insérer un ensemble de tuples dans une table à partir du résultat d'une requête select. Comme le montre la requête suivante :

```
> insert into table_name [(coln1,coln2 … , colnn)]
  select val1,val2,…,valn from …
```

B2 – LMD : MODIFICATION DES DONNÉES

Syntaxe (réduite) – modification

```
Update $table_name
Set col1 = val1, […,coln=valn]
[where $condition_expression]
```

Exemple

```
> Update meps
  set national_party=‘non étiqueté’
  where ep_id=2107
```

Remarque : modification avec un résultat d'interrogation

De la même manière que pour l'insertion, il est possible d'utiliser le résultat d'une interrogation pour faire un update d'une ou plusieurs lignes, comme le montre la requête ci-dessous : *

```
> Update $table_name
  set col1,col2 …, coln = (select val1,val2 … valn from …)
  [where $condition_expression]
```

NB : la requête d'interrogation doit retourner une ligne au plus

B2 – LMD : SUPPRESSION DES DONNÉES

Syntaxe (réduite) – suppression

```
Delete from $table_name
[where $condition_expression]
```

NB : la clause de condition peut être le résultat d'une requête

Exemple

```
> Delete from 'votes'
  where themes='immigration'
```

LID : Langage d'interrogation de données

Partie de SQL qui permet d'interroger (extraire) les données stockées dans la base. Il permet l'insertion. Il permet entre autres de filtrer (`where`), d'agréger (`Group by`), et de joindre (`Join`) les données des différentes tables dans une base.

`(Select)`

B3 – SYNTAXE GÉNÉRALE D'UNE REQUÊTE SELECT

Syntaxe select (réduite)

```
Select col1 as rn_1,col2 as rn_2 … coln as rn_n
   From table_1,table_2, … table_k
   Where Cond1 {and | or } … Condl
   Group by colX, …, colY
   Having Condg1 {and | or } … condgt
   Order by colT {asc | desc}
```

B3 – SYNTAXE GÉNÉRALE D'UNE REQUÊTE SELECT

Conditions

- `oper1 = | != | > | < | >= | <= oper2`
- `oper is [not] null`
- `oper [not] between b_min and b_max`
- `oper like 'reg_exp'`
- `oper [not] in (query)`
- …

B3 – SYNTAXE GÉNÉRALE D'UNE REQUÊTE SELECT

Exemple

```
> Select * from 'MEPS'
  where national_party in ('FN','LR')
  order by full_name

> Select count(ep_id) from 'MEPS'
  group by country
```

NB : Au fur et à mesure on fait un rappel de requêtes à travers le TD

Exemple de jointure

```
Select *
from 'MEPS' JOIN 'POSITION' using 'EP_ID'
where country in ('UK','GR')
```

Equivalente à : ⇔ (a une projection prêt)

```
Select *
from 'MEPS' JOIN 'POSITION' on MEPS.ep_id = POSITION.ep_id
```

Exemple imbrication de requêtes

```
> Select *
  from 'MEPS'
  where national_party in
  (
     select national_party
     from droite_national_parties
  )
```

Définition : Vue

Une vue est une table virtuelle dont le schéma et le contenu sont dérivés de la base réelle par un ensemble de requêtes [Gardarin, 2003].

B3 – SYNTAXE GÉNÉRALE D'UNE REQUÊTE SELECT

Exemple création d'une vue

```
Create View meps_droite as
Select *
    from 'MEPS'
    where national_party in
    (
       select national_party
       from droite_national_parties
    )
```

SOMMAIRE

TD – ORGANISATION DU TD

Sous-Partie	Durée	Objectifs
TD-REF	30 mn	Réflexion et application des notions de cours
TD-SOL	30 mn	Solution participatif

Connectez vous sur SQL-DEVELOPER en utilisant vos comptes

- Host : iutdoua-oracle.univ-lyon1.fr
- SID : orcl

TD – ENONCÉ DU TD

Relations

etat = {code, nom, capitale, nbHab}

prenom = {idP, libelle, estCelebre}

naissance = {#idP, #code, sexe, annee, nb}

TD – ENONCÉ DU TD

Exercice 1

Créer la table `etat` qui mémorise le code postal de l'état, son nom, sa capitale et son nombre d'habitants. Par exemple, 38 041 430 personnes vivent en Californie, dont la capitale est Sacramento.

etat = {code, nom, capitale, nbHab}

Exercice 2

Créer la table `prenom` qui stocke les prénoms des personnes affiliées à la sécurité sociale des USA. Pour chaque prénom, on mémorise son identifiant (nombre entier), son libellé et s'il est célèbre ou pas (3 valeurs possibles : oui, non ou inconnu).

prenom = {idP, libelle, estCelebre}

Exercice 3

Créer la table `naissance` qui enregistre le nombre de naissances annuelles. Le sexe des nouveaux nés est précisé.

naissance = {#idP, #code, sexe, annee, nb}

Exercice 4

Remplir la table `etat` avec les données suivantes :

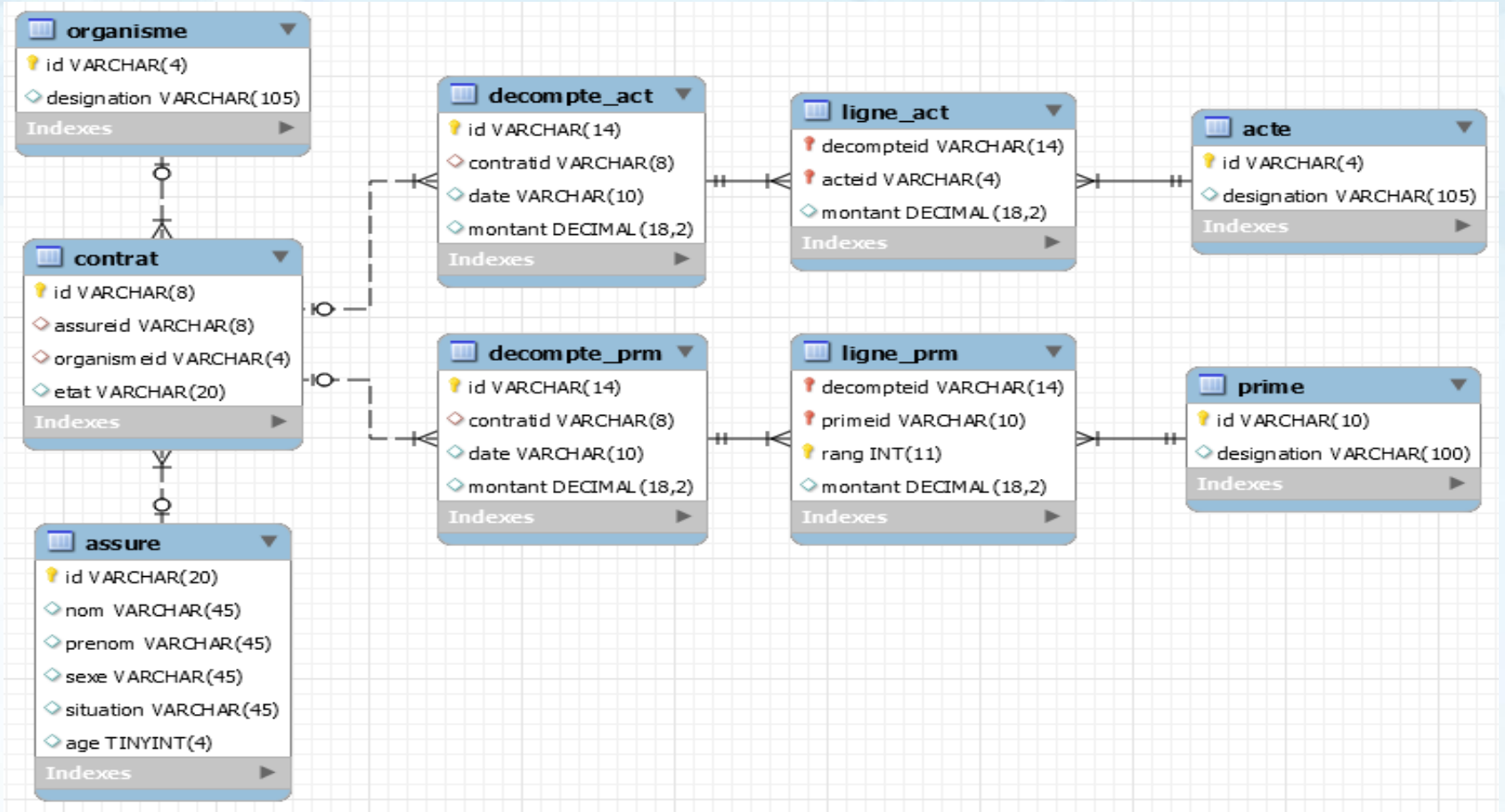
Code	Nom	Capitale	Population
CA	California	Sacramento	38 041 430
TX	Texas	Austin	26 059 203
FL	Florida	Tallahasee	19 317 568
MA	Massachussets	Boston	6 646 144
CO	Colorado	Denver	5 187 582
LA	Louisiana	Baton Rouge	4 601 893
UT	Utah	Salt Lake City	2 855 287

TD – ENONCÉ DU TD

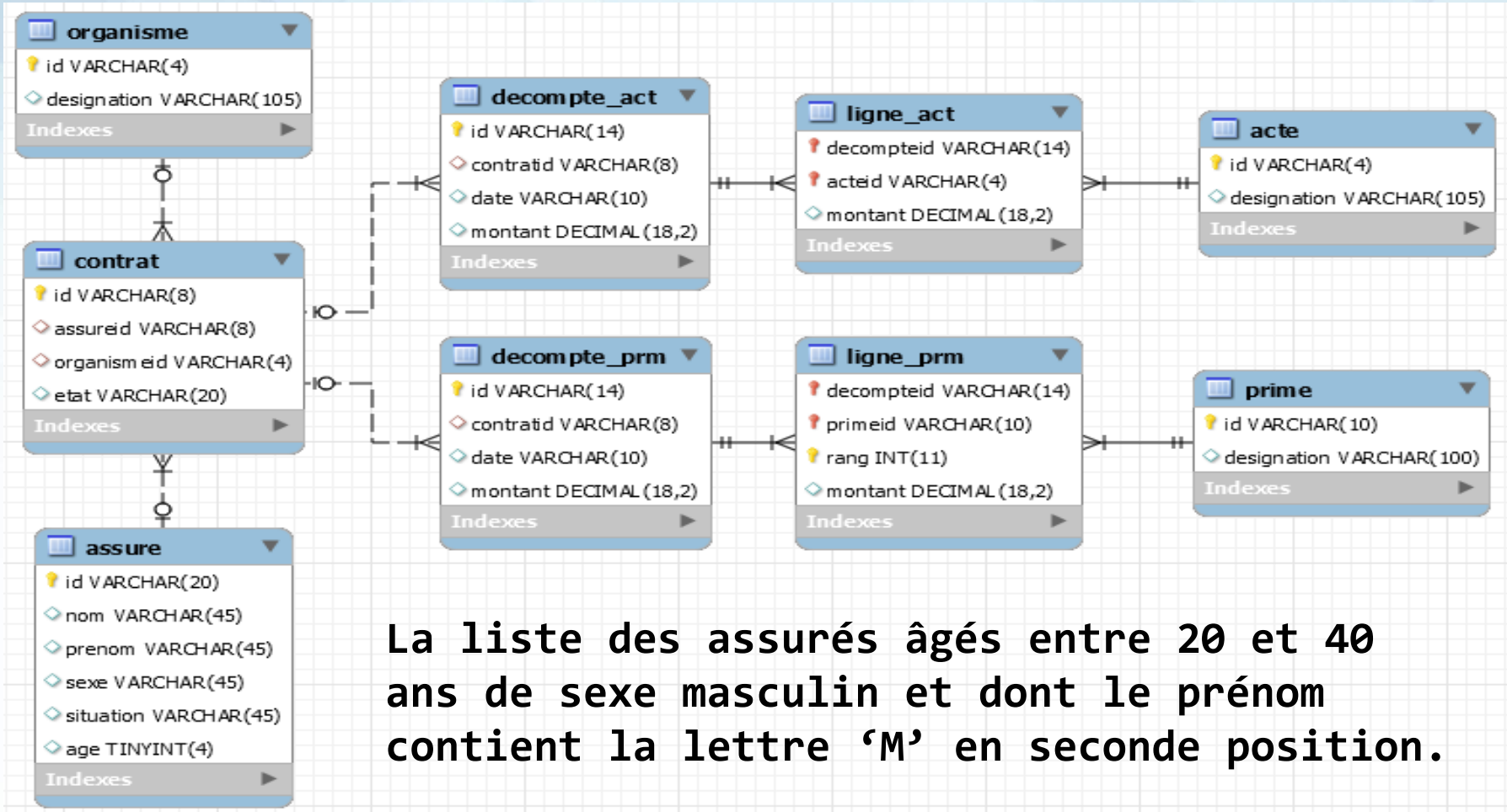
Exercice 5

Travailler sur la base mutuelle (Télécharger mutuelle.sql et l'exécuter)

Ecrire la requête d'interrogation qui permet d'extraire :

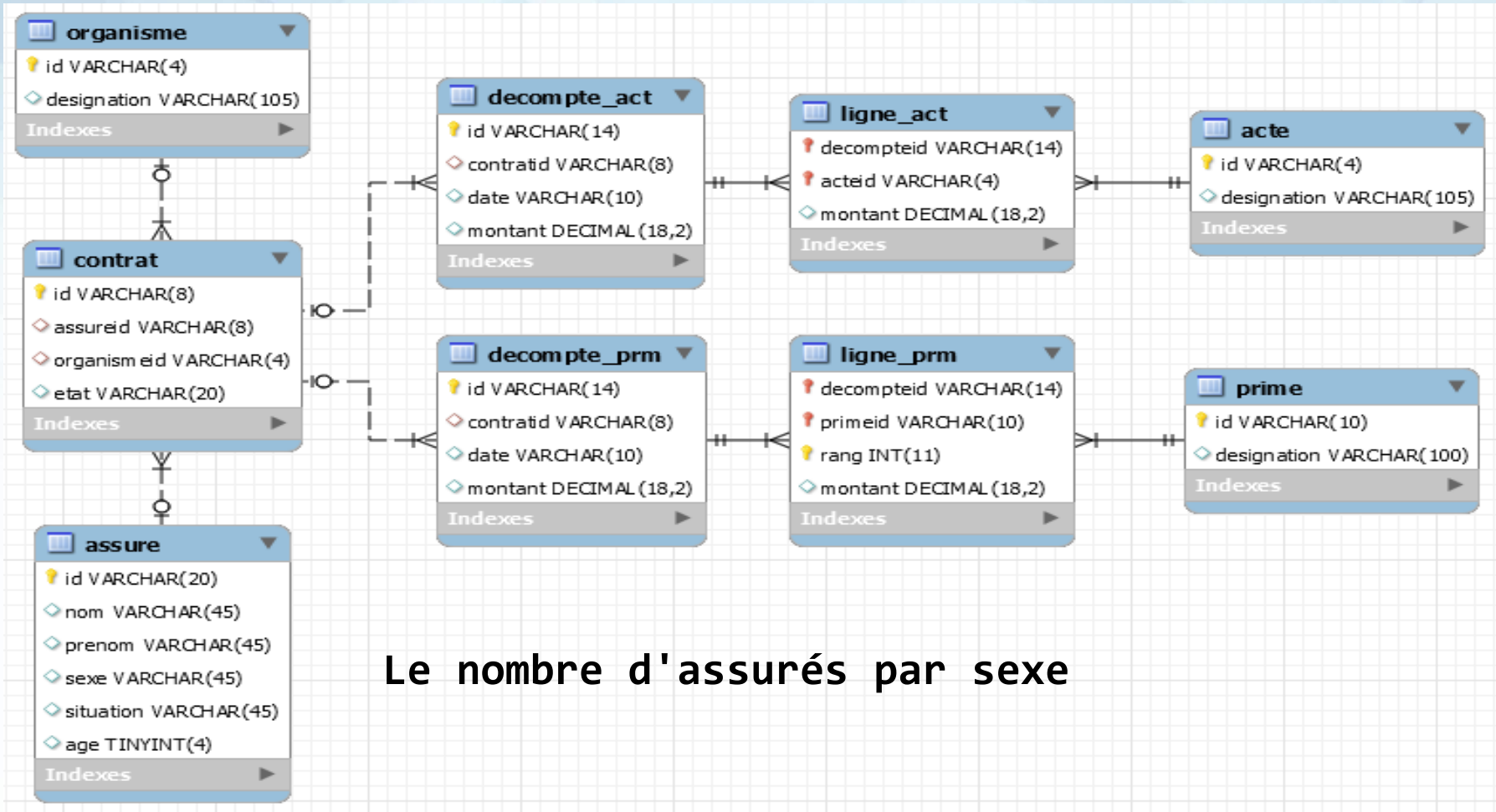


TD – ENONCÉ DU TD



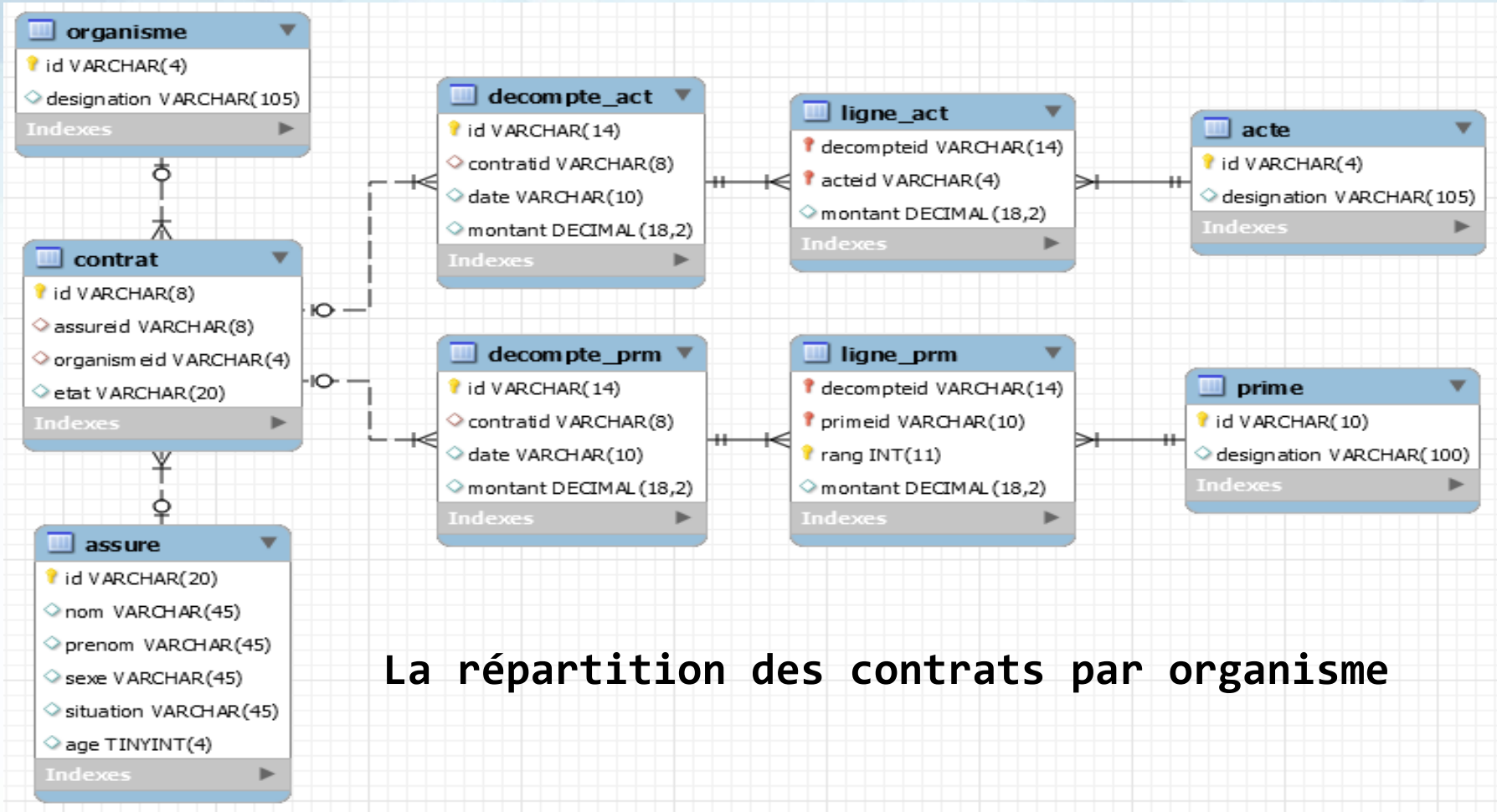
La liste des assurés âgés entre 20 et 40 ans de sexe masculin et dont le prénom contient la lettre ‘M’ en seconde position.

TD – ENONCÉ DU TD



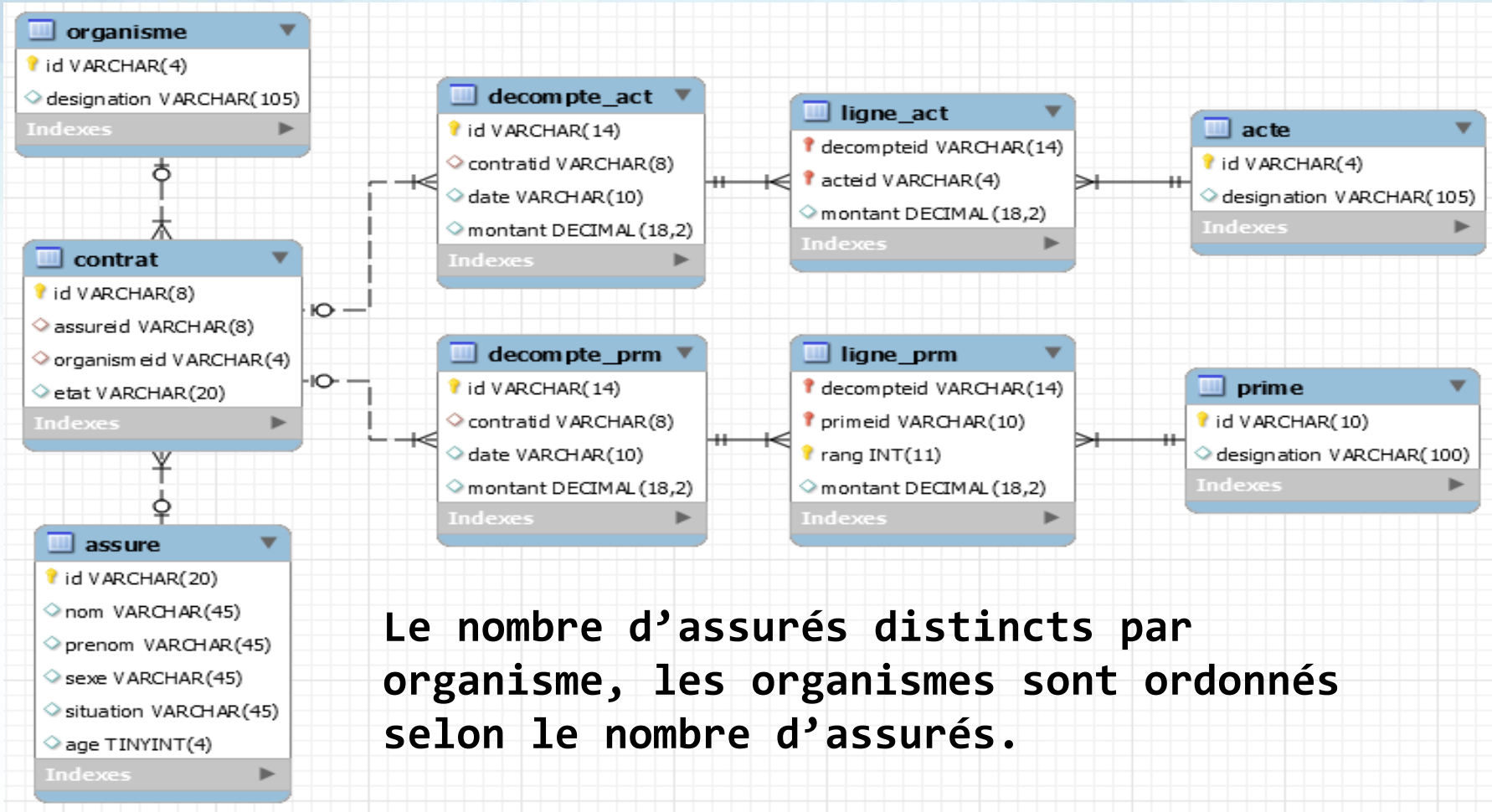
Le nombre d'assurés par sexe

TD – ENONCÉ DU TD



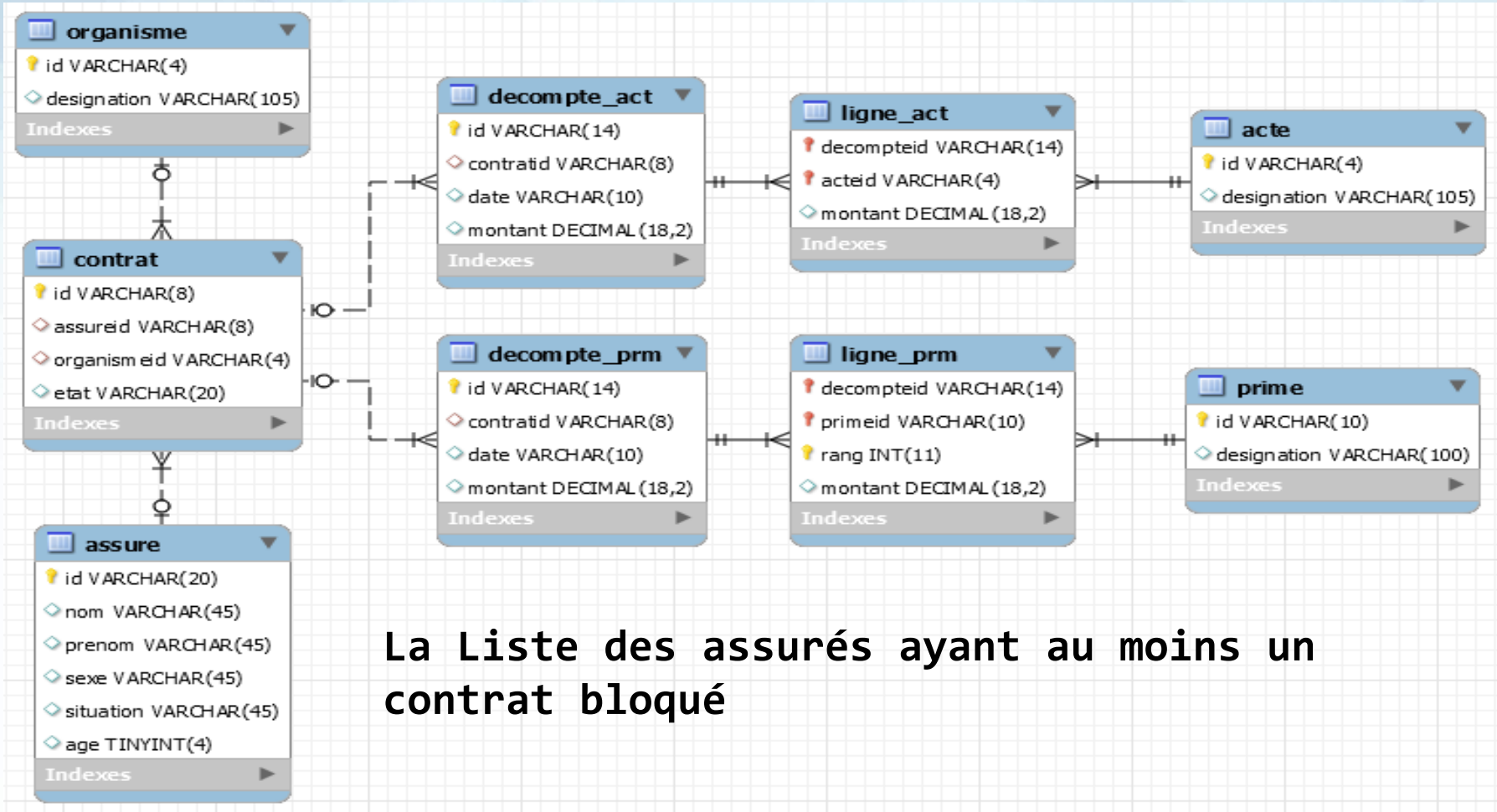
La répartition des contrats par organisme

TD – ENONCÉ DU TD



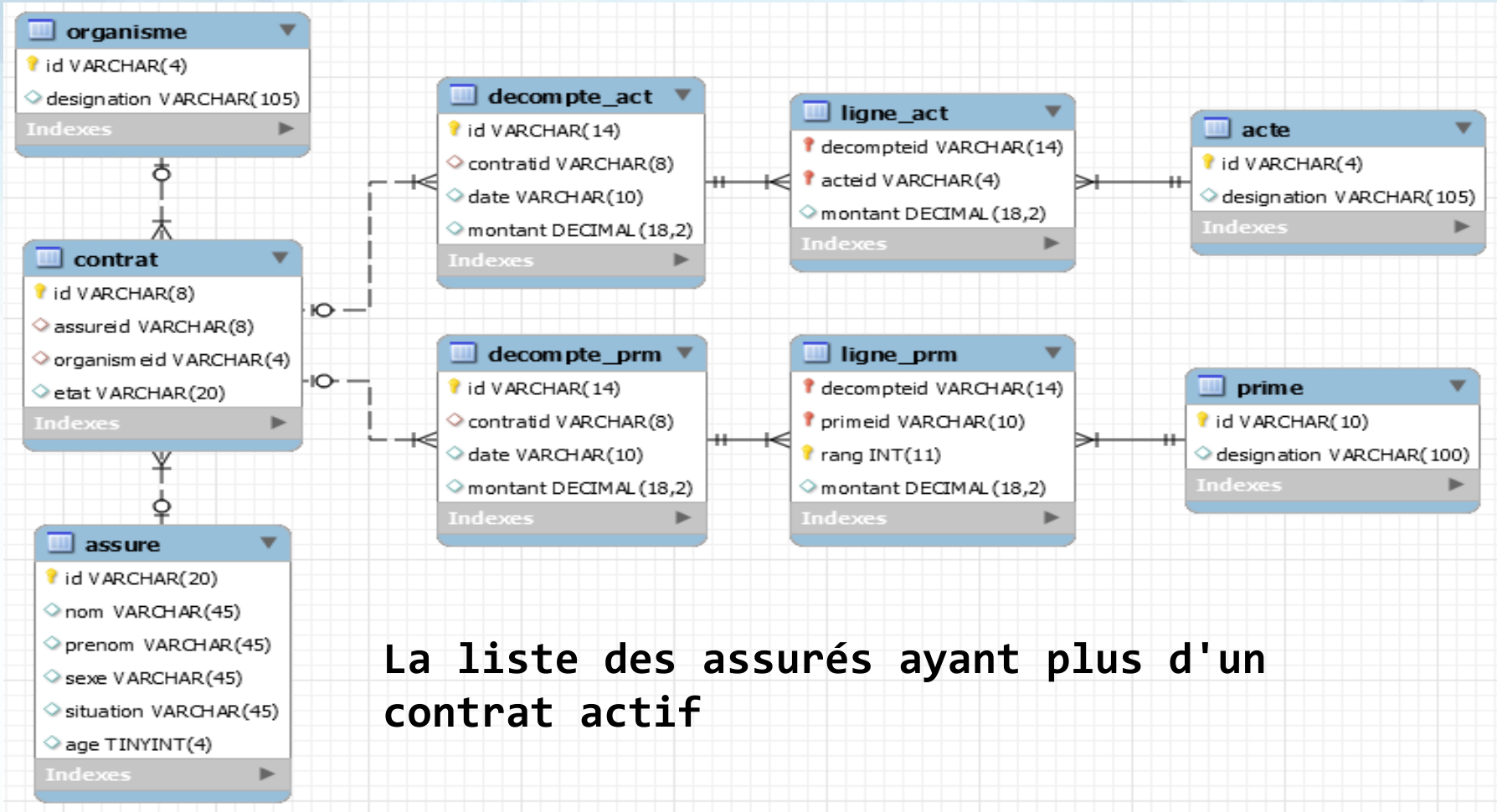
Le nombre d'assurés distincts par organisme, les organismes sont ordonnés selon le nombre d'assurés.

TD – ENONCÉ DU TD



La Liste des assurés ayant au moins un contrat bloqué

TD – ENONCÉ DU TD



La liste des assurés ayant plus d'un contrat actif

SOMMAIRE

TP – ORGANISATION DU TP

Sous-Partie	Durée	Objectifs
TP	1 heure 30 mn	Résoudre le TP, les questions sont les bienvenues
CPT_RENDU	10 mn	Envoyer les fichier.txt au mail : adnene.belfodil@insa-lyon.fr