Bases de Données Relationnelles

TP3 :Gestion d'un emploi du temps

SI2 et MAM2

1 Exposé du problème

On se propose de réaliser un système d'information pour gérer l'emploi du temps des étudiants des départements SI et MAM. Il s'agit de concevoir la base, de générer les différentes tables et de les remplir avec un jeu de données initial (cf les fichiers **creneaux.csv** et **edt_49.csv** dans `~rueher/BDR/TP3a`).

1.1 Fichiers de données

Le fichier **creneaux.csv** contient les caractéristiques générales des créneaux d'enseignement. Il comporte des lignes avec les champs suivants :

1. **code_module** varchar(20) : le code d'un module d'enseignement tel que 'BD', 'LFA' ou 'SYSTEME';
2. **libelle** varchar(50) : le nom détaillé du module;
3. **cursus** varchar(10): le nom d'un groupe d'étudiants pour lequel on veut pouvoir afficher un edt, comme "SI1", "MAM2".
4. **groupe** varchar(10) : le nom d'un groupe d'étudiants qui suivent ensemble un ou plusieurs enseignements, comme 'SI2', 'MA2', 'GR3' ou 'SM2'.
5. **durée** : la durée du créneau type, en nombre entier d'heures, et qui sera la même chaque fois que le créneau aura lieu.

Les fichiers **edt_x.csv** , que l'on reçoit chaque semaine pour alimenter la base, permettent de définir l'emploi du temps de la semaine créneau par créneau. Ils comportent des lignes avec les champs suivants:

1. **code_module** varchar(20) : code de module
2. **groupe** varchar(10)
3. **semaine** numeric(2,0) : numéro de la semaine (de 1 à 52)
4. **jour** numeric(1,0) : numéro du jour de la semaine (de 1 à 5)
5. **heured** numeric(1,0) : numéro de la tranche horaire du démarrage du créneau (de 1 à 8)
6. **salle** varchar(10) : un nom de salle

1.2 Contraintes et dépendances fonctionnelles

L'association **(code_module, groupe)** constitue un créneau type d'enseignement, se déroulant certaines semaines de l'année.
On suppose que les contraintes suivantes sont respectées:

1. Un créneau type (code_module, groupe) se déroule au plus une fois par semaine, et en une seule fois (une seule journée et sans interruption).

2. Pour une semaine donnée, un créneau type se déroule dans la même salle.
3. Chaque module est défini par son code. Il a un seul libellé.
4. un *groupe* est rattaché à un ou plusieurs cursus.
5. Chaque créneau type (code_module, groupe) a une seule durée.

Les dépendances fonctionnelles suivantes doivent être vérifiées:

1. code_module → libelle
2. code_module, groupe → duree
3. code_module, groupe, sem → jour, heured
4. code_module, groupe, sem → salle

La dépendance 4 ne se déduit pas de la dépendance 3 : si le même cours est assuré par deux enseignants différents le même jour à la même heure mais dans deux salles différentes, la dépendance 3 est vérifiée mais pas la 4.
Les dépendances 3 et 4 permettent de déduire la dépendance:

5. code_module, groupe, sem → jour, heured, salle

1.3 Travail demandé

On demande d'abord de concevoir le schéma, c'est à dire d'indentifier les dépendances fonctionnelles et de définir les tables en 3NF.

Une décomposition en 3NF est :
module(code_module, libelle)
creneau_type (code_module, groupe, duree)
cursus (cursus) % éventuellement on peut ajouter un libellé
groupe (groupe) % éventuellement on peut ajouter un libellé
cursus_creneau(cursus, groupe)
edt_sem (code_module, groupe, sem,jour, heured, salle)

Il faut aussi écrire les scripts SQL suivants :

1. Le script creer_schema.sql pour créer le schéma complet de la base;

```
/*
create_schema_edt.sql
-----------------------------------------

L'utilisation des références permet de garantir que les contraintes
liées aux dépendances fonctionnelles sont respectées
*/

-- Suppresion des tables

drop table edt_sem;
drop table cursus_groupe;
drop table cursus;
drop table groupe;
drop table creneau_type;
drop table module;
drop table edt_init;
```

```
drop table creneau_init;

-----------------------------------------
-- Tables de donnees
-----------------------------------------

create table creneau_init(
    code_module  varchar(20),
    libelle varchar(50),
    cursus  varchar(10),
    groupe   varchar(10),
    duree   numeric(1)
);

create table edt_init (
    module  varchar(20),
    groupe   varchar(10),
    sem     numeric(2),
    jour    numeric(1),
    heured  numeric(1),
    salle   varchar(10)
);

-----------------------------------------
-- Table pour le stockage de EDT
-----------------------------------------


create table module (               -- un module est défini par son code
    code_module  varchar(20) primary key,
    libelle varchar(50)
);

create table creneau_type (         -- chaque module(code_module,groupe)
                                    -- a une durée

    code_module  varchar(20) references module,
    groupe   varchar(10),
    duree   numeric(1),
    primary key (code_module, groupe)
);

create table cursus (
    nomcursus   varchar(10) primary key,
    libelle varchar(50)
);
create table groupe (
    nomcursus   varchar(10) primary key,
    libelle varchar(50)
);
create table cursus_groupe(           -- lien entre cursus et groupe
    nomcursus  varchar(10) references cursus,
    groupe   varchar(10) references groupe,
    PRIMARY KEY  (nomcursus, groupe)
);

create table edt_sem (
```

```
    code_module  varchar(20) references module,
    groupe    varchar(10),
    sem       numeric(2) CHECK ( sem >= 1 and sem <= 52),
    jour      numeric(1) CHECK ( jour  >= 1 and jour <= 5),
    heured    numeric(1) CHECK ( heured >= 1 and heured <= 8),
    salle     varchar(10),
      primary key (code_module, groupe, sem), -- code_module, groupe, sem
                                              --   -> jour, heured, durée
      unique (sem, jour, heured, salle )      -- il n'y a pas 2 cours en même
                                              -- temps dans la même salle
    );
```

---

2. Le script initialiser.sql pour alimenter la base avec les données stables indépendantes des semaines

---

```
/*
-----------------------------------------
init_schema_edt.sql
-----------------------------------------
*/

-- ---------------------------------------
-- tables de donnees
-- ---------------------------------------

\copy creneau_init from creneaux.csv delimiter ';' csv

-- ---------------------------------------
-- tables indépendantes de la semaine
-- ---------------------------------------

insert into module
 select distinct C.code_module, C.libelle from creneau_init C;


insert into creneau_type
    select distinct C.code_module, C.groupe, C.duree from creneau_init C;

insert into cursus
    select distinct C.cursus from creneau_init C;

insert into groupe
    select distinct C.groupe from creneau_init C;

insert into cursus_groupe
    select distinct C.cursus, C.groupe from creneau_init C;
```

---

3. Le script remplir_49.sql qui permet d'alimenter la base avec l'emploi du temps de la semaine 49

```
/*
```

```
-------------------------------------------
fill_semaine_edt.sql
-------------------------------------------
*/

-- ---------------------------------------
-- Tables de données
-- ---------------------------------------


\copy edt_init from edt_49.csv delimiter ';' csv


-- ---------------------------------------
-- Table Semaine
-- ---------------------------------------


insert into edt_sem   select * from edt_init;
```

L'utilisation des références devra garantir que les contraintes liées aux dépendances fonctionnelles sont respectées.

2 Remarques

1. On suppose qu'il n'y a pas deux cours différents le même jour à la même heure dans la même salle.
2. Il existe une relation (m,n) entre un groupe et un cursus
3. Les tables que l'on peut générer à partir des fichiers creneaux.csv et edt_49.csv ne contiennent pas de clé primaire. On va donc dériver des tables avec des clés primaires à partir des informations de ces tables et des dépendances fonctionnelles.
4. Récupération du contenu d'un fichier csv:

   \copy *table* from *ficher* delimiter ';' csv

   si toutefois le delimiteur est un point virgule, bien entendu.