Université Paris Nanterre

Année Universitaire 2021-2022
UFR SEGMI, L2 MIASH

Bases de Données Relationnelles 2, TD 4

TD4 : Normalisation d'une BD sous SQL

Objectifs :

1- Approfondir les compétences dans le domaine de normalisation d'un schéma relationnel selon les 3 formes normales FN1, FN2 et FN3.

2- Écrire en SQL des requêtes de correction d'une BDR dans un objectif de normalisation

I. Découverte du SGBD MySQL et l'éditeur « Workbench »

Pour ce cours, nous utilisé le SGBD MySQL car c'est le SGBD le plus répandu sur le marché et il est disponible sur les machines de l'université. Vous pouvez aussi l'installer sur vos machines personnelles en suivant le guide fourni dans ce lien :
Installer le SGBD MySQL - OpenClassroom

Lancer MySQL sur vos machines, lancer Workbench et essayer de découvrir votre environnement de travail. Créer ensuite votre première base « **rentcars** ».

II. Création des tables de la base du TD

Vous travaillerez pour ce TD avec une base de données d'une société de location de voiture. Cette base est composée de deux tables (voir Annexe):

- Une table appelée `customers` qui contient la liste des clients avec des références sur les voitures louées et les factures correspondantes.
- Une table appelée `customer_rentals` qui contient des informations sur les locations de voitures effectuées par les clients.

Vous disposez en ligne d'un script SQL qui permet de créer et peupler ces deux tables (ci-dessous). Ouvrez et exécutez ce script sous Workbench ou copiez directement le code SQL ci-dessous. Vérifiez que les deux tables ont été bien créées et peuplées.

```
#Créer la base customer_rentals
DROP TABLE IF EXISTS customer_rentals;
DROP TABLE IF EXISTS customers

CREATE TABLE IF NOT EXISTS customers(
customer_id INT NOT NULL PRIMARY key,
customer_name VARCHAR( 128 ),
cars_rented  VARCHAR( 128 ),
invoice_id VARCHAR( 128 ),
premium_member VARCHAR( 128 ),
salutation VARCHAR( 128 )
);
CREATE TABLE IF NOT EXISTS
customer_rentals(
customer_id INT NOT NULL,
car_id VARCHAR( 128 ) NOT NULL,
start_date DATE,
end_date DATE,
model VARCHAR( 128 ),
manufacturer VARCHAR( 128 ),
type_car VARCHAR( 128 ),
condition VARCHAR( 128 ),
color VARCHAR( 128 ),
CONSTRAINT pk PRIMARY KEY (customer_id,
car_id, start_date)
);
#Peupler la table customers
INSERT INTO customers VALUES (1453,"Kelly Brennan","4KL298","4534","FALSE", "Dr");
INSERT INTO customers VALUES (1454,"Tom Nguyen", "5PL4YY" , "9832", "FALSE", "Mr");
INSERT INTO customers VALUES (1455,"Georgia Kim","5H9OP5, 9PH8GF, 499ERW", "2903, 3490,
1021","TRUE", "Ms");
INSERT INTO customers VALUES (1456,"Jean Ford","4KL298, 9PH8GF", "7890, 4494","TRUE", "Mrs");
#Peupler la table customer_rentals
```

```
INSERT INTO customer_rentals VALUES(1453, "4KL298", 2019-01-08, 2019-01-10, "Golf 2017",
"Volkswagen", "hatchback", "fair", "blue");
INSERT INTO customer_rentals VALUES(1454, "5PL4YY", 2019-03-18, 2019-03-21, "Camaro 2019",
"Chevrolet", "convertible", "excellent", "red");
INSERT INTO customer_rentals VALUES(1455, "5H9OP5", 2019-05-02, 2019-05-16, "CRV 2018", "Honda",
"SUV", "good", "grey");
INSERT INTO customer_rentals VALUES (1455, "499ERW", 2019-01-12, 2019-01-13, "CRV 2018", "Honda",
"SUV", "excellent", "black");
INSERT INTO customer_rentals VALUES (1456, "4KL298", 2019-02-17, 2019-02-22, "Golf 2017",
"Volkswagen", "hatchback", "fair",  "blue");
```

Votre travail consiste à explorer différents schémas et augmenter progressivement la normalisation de ces schémas à travers les différentes formes normales. Pour transformer cette base en une base normalisée selon les formes normales de 1 à 3 (FN1, FN2, FN3), il faut réorganiser les tables, créer des nouvelles tables, déplacer des données et supprimer des données.

III. Conversion de la base « rentcars » en 1NF

Soit la table "`customers`" que vous avez créé ci-dessus. Cette table, respecte-t-elle la FN1? Choisissez une réponse et justifiez.

- Oui, tous les enregistrements sont uniques ;
- Non, parce qu'il y a plusieurs valeurs dans `cars_rented` et `invoice_id` ;
- Non, parce que certaines colonnes non-clés ne dépendent pas `de customer_id`, la clé principale.

Le champ `cars_rented` contient une ou plusieurs valeurs de `car_ids`, et le champ `invoice_id` contient aussi plusieurs valeurs. Ce ne sont pas des champs atomiques. Ces deux colonnes sont à supprimer de la table `customers` pour la normaliser à la première forme normale. Les données doivent êtres donc déplacées dans une autre table.

1. Créez une nouvelle table `cust_rentals` pour enregistrer les `car_ids` et `invoice_ids` des `customer_ids` qui ont loué ces voitures. Cette table a comme clé primaire le champ invoice_id.
2. Il n'est pas possible de copier directement les données de la tables customers vers la nouvelle table `cust_rentals` parce que le champs clé primaire `invoice_id` est déclaré comme chaine de caractère et peut regrouper plusieurs valeurs (de même pour le champ `car_id`). Un traitement préalable d'extraction et de conversion est nécessaire. On vous propose alors d'exécuter les commandes d'insertion ci-dessous :

```
INSERT INTO cust_rentals values (2903, "5H9OP5", 1455);
INSERT INTO cust_rentals values (3490, "9PH8GF", 1455);
INSERT INTO cust_rentals values (1021, "499ERW", 1455);
INSERT INTO cust_rentals values (7890, "4KL298", 1456);
INSERT INTO cust_rentals values (4534, "4KL298", 1453);
INSERT INTO cust_rentals values (9832, "5PL4YY", 1454);
```

3. Supprimer les colonnes "`cars_rented`" et "`invoice_id`" de la table "`customers`".

IV. Conversion de la base « rentcars » à la FN2

La table "`customers`" est convertie à la FN1. Notre objectif dans cette question est de convertir la table "`customer_rentals`" à la FN2.

1. Affichez le contenu de la table "`customer_rentals`" et répondez à la question suivante.

Pourquoi `customer_rentals` ne répond pas aux critères FN2 ? Choisissez une réponse et justifiez.

- Parce que le champ `end_date` ne dépend pas de toute la clé primaire.
- Parce qu'il ne peut y avoir au plus que deux clés primaires.
- Parce qu'il existe des attributs non clés décrivant la voiture qui ne dépendent que d'une clé principale, `car_id`.

2. Pour convertir la table `customer_rentals` à la FN2, il faut créer une table indépendante pour les objets `rental_cars`. Quels sont les champs de `customer_rentals` à inclure dans la nouvelle table `rental_cars` ?
3. Créez la nouvelle table `rental_cars` avec comme clé primaire `car_id` et les attributs associés définis dans la question précédente. Pour définir les types des champs, vous pouvez vous inspirer de la requête de création de la table `customer_rentals` de la question I (gardez les mêmes types).
4. Copiez les données associées aux voitures de la table `customer_rentals` vers la nouvelle table `rental_cars`. Vous devrez le faire en une seule requête de type :

```
INSERT INTO rental_cars(.....) SELECT .....FROM ....;
```

5. Supprimez de la table `customer_rentals` les champs associés à la table voiture. Attention, le champ `car_id`, qui fait partie de la clé primaire ne doit pas être supprimé.

V. Conversion de la base « rentcars » à la FN3

Nous arrivons maintenant à la **FN3**. Dans l'exercice précédent, vous avez créé une table regroupant l'identifiant `car_id` et les attributs de voiture.

Or, il s'avère que la nouvelle table qui en résulte, `rental_cars`, ne respecte pas la FN3.

1. Pourquoi `rental_cars` ne répond pas aux critères de FN3 ?

- Parce qu'il y a deux colonnes qui dépendent de la colonne non-clé, model.
- Parce qu'il y a deux colonnes qui dépendent de la colonne non-clé, color.
- Parce que les critères **FN2** ne sont pas satisfaits.

Pour normaliser la nouvelle table `rental_cars`, il faut supprimer la dépendance transitive et migrer dans une nouvelle table tous les champs qui ne dépendent pas directement de la clé. Pour ce faire, on vous demande de créer une nouvelle table `car_model` contenant 3 champs de la table `'rental_cars' (model,manufacturer,type_car)`. Créez une requête LDD vous permettant de créer la table et copier les données en même temps :

```
CREAT TABLE ..... AS SELECT ..... FROM ....;
```

4. Modifiez la nouvelle table `car_model` afin de définir le champ `model` comme clé primaire.
5. Supprimez les **deux** champs (`manufacturer`,`type_car`) de la table `rental_cars` afin qu'elle respecte la FN3.

VI. Création des contraintes d'intégrité référentielle de la base « rentcars »

Maintenant que la base est normalisée, pour la finaliser, il faudra créer toutes les contraintes d'intégrité référentielle.

1. Identifiez et créez le lien logique entre les tables `cars_rented` et `customers`.
2. Identifiez et créez le lien logique entre les tables `customer_rentals` et `rental_cars`.
3. Identifiez et créez le lien logique entre les tables `rental_cars` et `car_mode`.

Exercice Complémentaire : Requêtes SQL LID

1. Quelles sont les voitures louées en Janvier 2019?
2. Qui loue des voitures de type "Chevrolet"?
3. Quels sont le type et la couleur de la (ou de les) voiture(s) louée(s) par Mr Tom Nguyen?
4. Quel est le nombre de voitures louées par Type?
5. Quel est le nombre de voitures louées par mois en 2019 ?

Annexe :

Table `customers`

customer_id	customer_name	cars_rented	invoice_id	premium_member	salutation
1453	Kelly Brennan	4KL298	4534	false	Dr
1454	Tom Nguyen	5PL4YY	9832	false	Mr
1455	Georgia Kim	5H9OP5, 9PH8GF, 499ERW	2903, 3490, 1021	true	Ms
1456	Jean Ford	4KL298, 9PH8GF	7890, 4494	true	Mrs

Table `customer_rentals`,

Customer_id	car_id	start_date	end_date	model	manufacturer	type_car	condition	color
1453	4KL298	08/01/2019	10/01/2019	Golf 2017	Volkswagen	hatchback	fair	blue

1454	5PL4YY	18/03/2019	21/03/2019	Camaro 2019	Chevrolet	convertible	excellent	red
1455	5H9OP5	14/04/2019	14/04/2019	CRV 2018	Honda	SUV	good	grey
1455	5H9OP5	02/05/2019	16/05/2019	CRV 2018	Honda	SUV	good	grey
1455	499ERW	12/01/2019	13/01/2019	CRV 2018	Honda	SUV	excellent	black
1456	4KL298	17/02/2019	22/02/2019	Golf 2017	Volkswagen	hatchback	fair	blue
1456	4KL298	05/03/2019	20/03/2019	Golf 2017	Volkswagen	hatchback	fair	blue