Société de location de films

PROJET BASE DE DONNÉES

Dylan GRAVA
Farouk SAHRAOUI

L3 Informatique

Université de Cergy-Pontoise

Encadrants: M. Dominique Laurent
M. Tao Yuan JEN
M. Marc LEMAIR

SOMMAIRE

Introduction

Dans le cadre de notre projet base de données nous avons pour mission de créer une base de données nous permettant de gérer une société de location de films.

Notre équipe est constituée de deux membres ayant suivis tous le module base de données au cours du semestre 5. L'équipe se compose de Dylan GRAVA et Farouk SAHRAOUI.
L'encadrement du projet à été réalisé par M. Tao Yuan JEN enseignent de base de données au département informatique à l'université de Cergy-Pontoise.

Objectif du projet

L'objectif de ce projet est de s'habituer à la modélisation d'un système de gestion de base de données à partir d'un cahier des charges.
Ce cahier des charge décrit les besoins pour la création d'un système de gestion de films d'une société de location. Ensuite nous réaliserons le modèle dans une base de données MySQL, afin que nous puissions nous familiariser avec les requêtes SQL.

Organisation

Nous avons décidé de réaliser le modèle conceptuel de données et le modèle logique de notre base de données à deux afin d'être en accord entre nous tout au long du projet.
Pour les requêtes, nous avons préféré nous diviser les tâches dans un premier temps et ensuite mettre nos résultats en commun.

Notre base de données

Listes des entités

Ci-dessous la liste des entités composant notre base de données:

- Actor
- Plays_in
- Movie
- Contents_in_sup
- Support
- Content_support
- Shop
- Works_in
- Employee
- Occupied
- Booking
- Reserved
- Member

Modèle Logique (MLD)

- Actor

id_actor	identifiant de l'acteur
firstname	nom de l'acteur
lastname	prénom de l'acteur

- Plays_in

id_actor	identifiant de l'acteur
id_movie	identifiant du film

- Movie

id_movie	identifiant du film
title	titre du film
director_movie	nom du metteur en scène
production_year	année de sortie
purchase_date	date d'achat
categorie	catégorie de film

- Contents_in_Sup

id_support	identifiant du support
id_movie	identifiant du film

- Support

id_support	identifiant du support
type	type de support
languages	langue
subtitling	sous-titrage
type_of_stereo	type de stéréophonie

- Contents_Supports

id_suports	identifiant du support
id_shop	identifiant du magasin

- Shop

id_shop	identifiant du magasin
name_shop	nom du magasin
adress_shop	adresse du magasin

rental_unit	machine de location automatique
zone	zone du magasin

- Workes_in

id_employee	identifiant de l'employé
id_shop	identifint du magasin

- Employee

id_employee	identifiant de l'employé
firstname	nom de l'employé
lastname	prénom de l'employé
adress	adresse de l'employé
starting_date	date d'embauche
type	type d'emplyé

- Occupied

id_support	identifiant du support
id_booking	identifiant de la réservation

- Booking

id_booking	identifiant de la réservation
start_date_rental	date début de location
end_date_rental	date fin de location
rental_shop	magasin où la location a été effectué
restitution_shop	magasin où le support a été déposé
id_movie	identifiant du film

- Reserved

id_booking	identifiant de la réservation
id_member	identifiant du membre

- Member

id_member	identifiant du membre
firstname	nom du membre
lastname	prénom du membre

adress	adresse du membre

Associations

Plays_in

Elle relie un acteur et un film. Elle est de cardinalité 1-N car un film peut avoir plusieurs acteurs.

Content_in_Sup

Elle relie un film et un support. Elle est de cardinalité 1-N car on peut avoir plusieurs support du même film.

Contents_support

Elle relie un support et un magasin. Elle est de cardinalité N-N car on peut avoir plusieurs support dans un seul magasin.

Works_in

Elle relie un magasin et un employé. Elle est de cordialité 1-N car dans un magasin il y a plusieurs employés.

Occupied

Elle relie un support à une réservation. Elle est de cardinalité N-N car on peut avoir plusieurs location du même support.

Reserved

Elle relie une réservation à un membre. Elle est de cardinalité 1-N car un membre peut faire plusieurs réservations.

Modèle conceptuel de données (MCD)

SCHÉMA MCD

Amélioration faite sur notre base de données:

Grâce aux remarques de M. Tao Yuan JEN, nous avons pu réaliser quelques modifications sur notre base de données.

Nous avons créer une seule table qui regroupe les trois type de support: ‘DVD’, ‘VHS’, ‘VCD’. Cela nous à permis de réduire la table dans un premier lieu mais aussi de limiter la demande de ressources.

Nous avons supprimer la table zone et ajouter un attribut ‘zone’ dans la table ‘Shop’.

Cela nous permet d’affecter directement une zone à un employé et d’éviter une répétition d’attributs.

Nous avons créer une seule table pour les employés avec un attribut ‘type’ qui nous permet de leur affecter un type dans la liste des types d’employés.

Les requêtes

1)

```
SELECT firstname, lastname, adress
FROM booking JOIN reserved USING (id_booking) JOIN member USING (id_member) JOIN occupied USING (id_booking) JOIN support USING (id_support)
        WHERE id_movie IN ( SELECT id_movie FROM booking where type = 'DVD')
   AND type = 'VHS';
```

2)

```
SELECT title
FROM booking JOIN occupied USING (id_booking) JOIN support USING (id_support) JOIN contents_in_sup USING (id_support) JOIN movie ON contents_in_sup.id_movie = movie.id_movie
WHERE
        type = 'DVD'
        AND start_date_rental IS NOT NULL
        AND title NOT IN (SELECT title
                FROM booking
                JOIN occupied USING (id_booking)
                JOIN support USING (id_support)
```

```
JOIN contents_in_sup USING (id_support)
JOIN movie ON contents_in_sup.id_movie = movie.id_movie
WHERE type = 'VHS'
AND start_date_rental IS NOT NULL);
```

3)

```
SELECT firstname, lastname, adress
    FROM booking JOIN reserved USING (id_booking) JOIN member USING (id_member)
    WHERE rental_shop != restitution_shop;
```

4)

```
SELECT firstname, lastname
    FROM employee JOIN Works_in USING (id_employee) JOIN shop USING (id_shop)
    WHERE id_shop != (SELECT id_shop AS id_shop_animation FROM movie JOIN
contents_in_sup USING (id_movie) JOIN support USING (id_support) JOIN contents_support
USING (id_support) JOIN shop USING (id_shop) WHERE categorie = 'animation')
AND employee.type ='Director';
```

5)

```
SELECT firstname, lastname, adress, COUNT(id_booking) AS id_booking_count
    FROM booking JOIN reserved USING (id_booking) JOIN member USING (id_member)
    WHERE DATEDIFF(NOW(), 'start_date_rental') < 30 GROUP BY id_member ORDER BY
id_booking_count DESC LIMIT 3;
```

6)

```
SELECT director_movie_2013.name_shop, director_movie_2013.firstname,
director_movie_2013.lastname, movie_2013.id_booking_count
    FROM (SELECT rental_shop, COUNT(id_booking) AS id_booking_count
    FROM booking WHERE DATEDIFF(`end_date_rental`, `start_date_rental`) > 3
        AND extract(year FROM `end_date_rental`) =2013 GROUP BY
rental_shop)movie_2013 JOIN (SELECT lastname, firstname, rental_shop, name_shop
            FROM booking JOIN occupied USING (id_booking) JOIN support USING
(id_support) JOIN contents_support USING (id_support) JOIN shop USING (id_shop) JOIN
works_in USING (id_shop) JOIN employee USING (id_employee)
```

WHERE employee.type = 'Director')director_movie_2013 ON movie_2013.rental_shop = director_movie_2013.rental_shop;

7)…

Conclusion

Ce projet nous a permis de comprendre le fonctionnement d'une base de données mais aussi d'améliorer notre gestion de projet.