Gaël Reblochon et sa chaîne de restaurants

Introduction

Le projet auquel vous allez participer a pour but de mettre en œuvre vos compétences en systèmes de gestion de bases de données relationnelles. Vous devrez concevoir et implanter le schéma relationnel pour une application permettant de gérer une chaîne de restaurants. De plus, vous vous essaierez à la programmation d'application utilisant une base de données (à travers des transactions). Le développement sera fait en Java en utilisant l'API JDBC.

Le projet est à faire en **quadrinômes** et donnera lieu à deux réunions de suivi intermédiaires, ainsi qu'à une soutenance en fin de projet. La constitution des équipes et la remise des livrables (rapport final et code source) se fera sur l'application *Teide*. L'évaluation se fera sur **les aspects bases de données** du projet en l'état au moment de la soutenance.

1. Description de l'application

Le célèbre chef Gaël Reblochon désire informatiser la gestion de sa chaîne de restaurants, des réservations de ses clients, ainsi que de ses plats et menus. Il veut donc faire appel à une équipe de (futurs) professionnels pour concevoir une base de données et une application répondant à ses besoins.

G. Reblochon gère plusieurs restaurants. Chacun de ces restaurants est identifié par un numéro unique, et est décrit par son nom, son adresse (rue, code postal, ville et pays), son numéro de téléphone et sa catégorie (exemple: étoilé, restauration rapide, etc.) De plus, chaque restaurant annonce au moins une spécialité culinaire qui dépend des compétences du ou des chefs travaillant dans ce restaurant. Ces spécialités peuvent être par exemple « cuisine savoyarde », «cuisine italienne», «cuisine chinoise», etc.

Un restaurant fait au plus deux services par jour (un le midi et un le soir) et propose une carte pour chacun de ces services. Les cartes sont redéfinies chaque jour pour chaque service. Une carte comprend des articles variés. Ces articles peuvent être des entrées, des plats principaux, des desserts ou des boissons. Chacun de ces articles possède un nom unique et un prix, et correspond à une spécialité. Bien sûr, un restaurant ne peut proposer un article que si ses spécialités annoncées le lui permettent. Un client a le choix de commander un ou plusieurs de ces articles indépendamment, ou de commander un menu à prix fixe – bien entendu, ce prix est inférieur à la somme des prix des articles que le menu contient. Chaque menu propose de choisir un plat principal et une boisson parmi une sélection d'articles, et peut également offrir une entrée et/ou un dessert au choix. Un menu est forcément dédié à une spécialité particulière, et tous les articles qu'il propose doivent correspondre à cette spécialité.

Dans ses restaurants, Gaël Reblochon accepte des clients uniquement sur réservation. Toute réservation est identifiée par un numéro unique, est effectuée par un client particulier et concerne une date et un service particuliers. De même, sur toute réservation figure le nombre de participants au repas (bien entendu, on ne demande pas les noms et coordonnées de tous les participants). Un client est identifié par un numéro unique, et on demande son nom et son numéro de téléphone (en cas d'éventuelle modification ou annulation). Bien entendu, un client peut faire plusieurs réservations dans des restaurants du groupe tout en gardant le même numéro de client. Pour la bonne gestion du service

dans les restaurants, on associe à chaque réservation et à chaque personne présente la liste des articles commandés. Pour une bonne gestion de la comptabilité des restaurants, on associe également le prix total payé par le client (sans tenir compte des sombres histoires de répartition entre personnes présentes, ni des éventuels pourboires laissés sur la table et réservés aux serveurs). Ce prix peut éventuellement être inférieur à la somme des prix des articles commandés par les personnes présentes (si le chef accepte de faire une ristourne ou offre un cadeau), mais ne peut jamais être supérieur.

Un restaurant a un nombre de places limité. Ainsi, afin d'éviter d'accepter trop de réservations, on répertorie les tables de chaque restaurant. Une table a un numéro unique dans le restaurant et est caractérisée par son emplacement (exemple: terrasse, salle, étage, etc.) et son nombre de places. Parfois, un client fait une réservation pour un grand nombre de personnes. Dans ce cas, on peut envisager d'accoler des tables voisines. Cependant, le nombre de places d'une table accolée à une autre est inférieur à son nombre de places lorsqu'elle est isolée (pour simplifier, on peut supposer que le nombre de places d'une table accolée reste le même quel que soit le nombre de tables avec lesquelles elle est accolée). Pour chaque table, on connaît donc également son nombre de places lorsqu'elle est accolée, ainsi que les tables voisines avec lesquelles elle peut être accolée. Évidemment, une table ne peut être voisine d'une autre que si elle se trouve dans le même emplacement, et on suppose aussi qu'une table ne peut avoir que deux voisines au maximum (les tables sont rectangulaires et ne peuvent être accolées que par leurs petits côtés). Lors de la réservation, le client peut indiquer sa préférence de placement, et on lui attribue une ou plusieurs tables selon ses souhaits et les disponibilités. G. Reblochon a fait le choix de ne jamais partager une même table entre plusieurs réservations, et s'il n'y a pas suffisamment de places disponibles pour une réservation, la politique est de la refuser.

2. Travail à réaliser

2.1. Modélisation du problème

La modélisation se décompose en deux temps. Dans un premier temps, vous aurez à **analyser le problème posé** pour en extraire les concepts, les dépendances fonctionnelles reliant ces concepts, ainsi que toutes les autres contraintes. De cette analyse, vous devrez proposer ensuite un **schéma Entités/Associations** représentant les données nécessaires à l'application et leurs liens sémantiques (ce qui correspond à l'état cohérent de la base de données).

2.2. Implantation de la base de données

Vous devrez ensuite implémenter le schéma Entités/Associations en un **schéma relationnel** sur le SGBD Oracle 11g disponible sur *Ensibm*. Vous devrez insérer suffisamment de données pertinentes pour la suite du projet. Vous préciserez également la forme normale des relations obtenues.

2.3. Analyse des fonctionnalités

Vous devrez définir les requêtes SQL2 nécessaires pour réaliser les fonctionnalités suivantes en les regroupant en transactions, ce qui permettra d'assurer la cohérence globale de la base de données, même en cas d'accès concurrents :

- Création d'un restaurant, y-compris positionnement des tables
- Création et modification des cartes et menus
- Consultation de la carte du jour :
    - consultation complète
    - Par type de plat (entrée, plat principal, dessert, boisson)
    - Uniquement les menus

- Enregistrement d'une réservation (comprenant l'affectation de la (des) table(s) et éventuellement la saisie d'un nouveau client)
- Modification d'une réservation (exemple : changement du nombre de participants)
- Facturation d'une réservation

Ces requêtes et transactions peuvent (doivent !) être testées sur Oracle (SQL*Plus) pour en vérifier leur bon fonctionnement, y compris pour des exécutions concurrentes.

2.4. Implantation des fonctionnalités

Les fonctionnalités précédemment étudiées devront être implantées en Java/JDBC. Vous pouvez choisir une interface texte ou graphique, cela n'a pas d'importance vu que **seuls les aspects Bases de Données seront évalués**.

3. Déroulement du projet

Le projet sera constitué de 15 heures en séances encadrées et 3 heures réservées pour les soutenances.

3.1. Séances encadrées

Les séances encadrées sont **obligatoires**. Lors de ces séances, les quadrinômes devront avancer sur le projet et pourront poser des questions à leur encadrant. Attention : posez bien vos questions. L'encadrant jouera soit le rôle du client (et, dans ce cas, ne pourra traiter les aspects BD), soit le rôle d'un expert en bases de données (mais, dans ce cas, il ne connaît rien à l'application). En fin de chaque séance, chaque équipe devra déposer une version actualisée de la documentation du projet **sans la valider** (validation uniquement en fin de projet). Ceci afin de pouvoir suivre le déroulement du projet et de pouvoir intervenir au plus tôt en cas de grosses erreurs/anomalies.

3.2. Outils

Vous disposez de deux outils principaux pour le bon déroulement du projet :

- Le **Kiosk** : vous y trouverez les documentations techniques pour accéder à Oracle et pour utiliser JDBC, des liens Internet utiles, ainsi qu'une **Foire Aux Questions** dans laquelle vous trouverez les réponses aux questions importantes pouvant concerner toutes les équipes (à consulter souvent, donc).
- **Teide** : l'application de gestion de projet. Vous devrez utiliser Teide pour constituer vos équipes, déposer vos rendus (documentation à chaque séance, code source Java et SQL en fin de projet) et vous inscrire aux créneaux de soutenance.

3.3. Suivis

Le projet donnera lieu à deux réunions de suivi intermédiaires entre l'encadrant et chacune des équipes. Les réunions seront provoquées par les équipes elles-mêmes ou par l'encadrant en fonction de l'avancement du travail. Les thèmes discutés lors des suivis seront les suivants :

- Analyse et modélisation Entités/Associations (~3ème séance encadrée)
- Analyse et implantation SQL des fonctionnalités (~6ème séance encadrée)

3.4. Livrables

- **Documentation du projet** : Vous devrez maintenir la documentation du projet tout au long de son déroulement. La documentation doit comprendre l'analyse du problème, la conception Entités/Associations, sa traduction en relationnel, l'analyse des fonctionnalités, leur implantation en SQL2, ainsi qu'un bilan du projet (organisation, points difficiles rencontrés, etc.) Un petit mode d'emploi de votre application est également le bienvenu.
  La documentation doit comprendre les explications nécessaires à sa compréhension et à la justification de vos choix.
- **Sources java et SQL2** : Vous devrez rendre en fin de projet le code source Java, l'implantation SQL des fonctionnalités, ainsi qu'un script SQL permettant de créer votre schéma relationnel.

3.5. Soutenance

Le projet se termine par une soutenance. Chaque équipe devra s'inscrire sur un créneau de soutenance via l'application *Teide*. La soutenance se compose de la façon suivante :

- 20 minutes pendant lesquelles vous devrez présenter votre projet (conception, implantation), faire une démonstration convaincante de votre prototype et faire un bilan du projet.
- 10 minutes de questions pour le jury.

Les soutenances sont courtes, vous devez donc bien les préparer (scénario pour la démonstration, répétition, etc.)