Mise en œuvre d'une application de gestion de Vélos en Libre-Service

1 Organisation du projet

Le projet "administration de bases de données" a lieu du 8/01 au 30/01 2015. Il doit être réalisé par des groupes de quatre personnes. Les salles affectées à cette activité sont indiquées sur le planning ADE (merci de respecter ces salles ou bien d'informer l'enseignement de votre localisation).

Des permanences sont organisées par les enseignants pour chacune des demi-journées consacrées au projet: en général de 9h à 11h30 le matin et de 14H à 17H l'après-midi (variable selon les jours). Un pointage des gens présents sera réalisé pour chaque demi-journée.
Des ressources et un forum sont disponibles sur Moodle IM^{2}AG. Les différents rapports intermédiaires et l'archive contenant le rapport final et les sources de l'application prêtes à l'emploi (scripts SQL, déclencheurs et programmes Java, scripts de lancement, tests automatisés) devront être déposés sur Moodle.

L'évaluation portera sur le travail réalisé et sur les rapports de projet: la présentation des documents (clarté, expression, orthographe), l'état d'avancement de l'application réalisée, la qualité de programmation (lisibilité, justification, pertinence et couverture des tests), ainsi que la soutenance (présentation et réponses aux questions). En outre, la prise de recul de chaque groupe par rapport à ses réussites et à ses lacunes sera également appréciée par les enseignants.

Les différents rapports devront fournir des réponses claires et synthétiques aux questions de l'énoncé, une présentation de la solution aux problèmes à traiter avec des jeux de tests complets le cas échéant. Le rapport final devra en plus présenter la mise en œuvre des solutions proposées, un bilan sur le projet et le travail effectué, ainsi que le code Java commenté.

2 Objectifs du projet

Le projet consiste en la réalisation de trois applications clientes Java accédant via le protocole JDBC à une base de données sur un serveur distant. Plusieurs clients peuvent accéder à la base de manière concurrente. L'accent est mis sur le maintien de la cohérence des données et en aucun cas sur l'interface homme-machine : une interface en mode texte offrant un minimum d'interactivité (menus, affichage des résultats, trace de la gestion des transactions concurrentes, etc.) est demandée. L'évaluation ne tiendra pas compte de la présence d'une interface graphique avancée mais uniquement de la facilité à suivre les tâches effectuées pour mener à bien une transaction et à vérifier le bon fonctionnement de l'application.

Les technologies utilisées sont : Java et JDBC pour la partie applicative, Oracle (SQL, PL/SQL) pour la partie base de données.

L'objectif visé est une meilleure compréhension des rôles respectifs de ces technologies lors de leur utilisation dans la perspective de l'implantation d'une application de gestion de données. Un aspect modélisation, inhérent à tout projet de spécification d'un système d'information, fera partie intégrante de ce projet.

3 Position du problème

La ville de Nantes souhaite se doter d'un service de vélos en libre-service sur le modèle du Vé'lib parisien, du Vélo'v lyonnais ou du VéCUB bordelais. Ce service, nommé VéPick, sera géré par une entreprise locale qui se lance dans le domaine de la location de vélos. Vous êtes chargés du système de gestion de données de VéPick, basé sur une base de données relationnelle.

Cette base de données doit gérer à la fois le parc de vélos, les stations, les clients, ainsi que les employés qui conduisent des véhicules de régulations (et qui déplacent les vélos d'une station à l'autre). Trois applications doivent permettre d'accéder à la base de données : une pour les clients, une pour les conducteurs des véhicules de régulation et une pour les superviseurs, avec des droits d'accès différents.

Chaque station est identifiée par son adresse. Elle possède plusieurs emplacements, appelés "bornettes", chacune pouvant accueillir un vélo. Chaque bornette d'une station est identifiée par un numéro et son état (OK, HS). Chaque vélo est décrit par son numéro identifiant, son modèle et la date de sa mise en service et son état (OK, HS). A chaque instant, un vélo peut être en station associée à une bornette, loué par un client, embarqué par un véhicule de régulation ou au centre de réparation. L'association d'un vélo a une bornette est automatique grâce à la puce RFID présente dans chaque vélo. Chaque station communique instantanément les changements d'état de ses bornettes à la base de données.

Le cout de location des vélos est identique quelque soit le modèle. Ce cout est fixé à l'année et renseigné dans la base de données.

Les clients peuvent être ou non abonnés au service VéPick. Les abonnés ont renseigné leur nom, prénom, date de naissance, sexe, adresse, numéro de CB, ainsi qu'un code secret leur permettant de s'identifier (en pratique, ces codes sont cryptés dans la base de données, mais nous ne considérerons pas ces détails ici). Plusieurs abonnés différents peuvent avoir le même numéro de CB (une disposition utile pour les familles). Chaque abonnement a une durée d'un an. Chaque abonné peut louer un vélo en le prenant et le ramenant dans un parc après s'être identifié. Les clients non abonnés ne renseignent que leur numéro de carte CB. La transaction bancaire ne sera pas considérée dans ce projet. Il impossible à un client de louer plusieurs vélos en même temps sur la même période. Pour chaque location par un non-abonné, un code secret est attribué au locataire, qu'il devra mémoriser pour s'identifier quand il rendra son vélo. Toute location de plus de douze heures est sanctionnée par une amende. Un client ayant 2 amendes en cours non régularisées ou une amende de plus d'un mois non régularisée ne peut plus louer de vélos, mais il peut toujours rendre un vélo.

A noter qu'en plus de la borne en station, les clients peuvent utiliser une application sur leur smartphone pour entrer leur code secret afin de prendre ou rendre un vélo. L'application fonctionne par géo-localisation mais cette fonctionnalité sera considérée comme gérée par une autre couche logicielle (non prise en compte dans ce projet). Le travail se focalisera sur la gestion concurrentielle qui résulte cette application pour smartphone (plusieurs personnes peuvent demander ou rendre un vélo en même temps).

Un client peut également renseigner par l'application smartphone/la borne de l'état d'un vélo: le vélo rendu est en panne ou le vélo qui vient d'être récupéré est en panne (3mn de délais dans ce dernier cas pour déclarer le problème).

Dans tous les cas le système ouvre la première bornette occupée par un vélo dans un état utilisable lors d'une demande de location et la première bornette libre lors d'une retour. L'algorithme soit être simple et laisser à votre convenance.

Pour un usage futur d'analyse de données et de statistiques, on prendra soin d'enregistrer dans la base de données l'ensemble des locations passées afin de mieux connaître le comportement des usagers du service.

Les clients abonnés disposent en plus d'un système de réservation qui leur permet de réserver un vélo dans une station précise pour un jour, un jour répété sur une période, ou tous les jours sur une période. Un client peut avoir plusieurs réservations en cours mais sans recouvrement : il est toujours impossible de louer plusieurs vélos dans une même période. Si une réservation est en conflit avec les réservations d'autres clients (plus de vélos disponibles) alors la réservation ne peut aboutir (même pour un seul jour sur une période).

Certains trajets sont plus courants que d'autres (par exemple, vers le campus universitaire le matin, et depuis le campus universitaire le soir). Afin d'assurer, à chaque station, la présence de vélos disponibles et d'emplacements libres pour y déposer son vélo, VéPick dispose de deux moyens de régulation.

La "remise Vplus" encourage les usagers à assurer eux-mêmes la régulation des vélos en donnant une prime (sous forme de crédit-temps pour une future location) aux usagers effectuant certains trajets bénéfiques pour la régulation. Pour bénéficier de la prime, un usager doit emprunter un vélo à une station "Vmoins" et le déposer à une station "Vplus". La définition de ces stations change en fonction des plages horaires, mais chaque station est toujours soit "Vmoins" soit "Vplus" soit "Vnul" (sans effet). La classification en Vmoins/Vplus est décidée par les superviseurs du service. Pour un abonné qui bénéficie de la remise Vplus, cette remise sera appliquée immédiatement sur son prochain trajet. Les remises Vplus ne sont pas cumulables. Un non-abonné reçoit un code identifiant, valable un mois, lui permettant d'avoir droit à la remise.

L'autre moyen de régulation dont dispose VéPick est le déploiement de véhicules de régulation. Chaque véhicule de régulation est identifié par un numéro, un modèle et une date de mise en service. Il peut accueillir un nombre de vélos indépendamment de son modèle. Les conducteurs des véhicules de régulation sont chargés de plusieurs tâches : identifier et signaler les vélos endommagés, et embarquer/débarquer les vélos dans leur véhicule. Les conducteurs des véhicules suivent une routine (à quelles stations aller et combien récupérer/déposer de vélos) qui est décidée par les superviseurs du service. La routine d'un véhicule est une suite ordonnée d'ordres du type "aller à telle station / s'assurer qu'il y ait x vélos non endommagés après mon passage et x places disponibles dans la station" ou "aller au centre de réparation / déposer les vélos endommagés et récupérer x vélos réparés". Chaque véhicule effectue les tâches de la routine dans l'ordre. Chaque action doit être validée avant de passer à la suivante. Si un ordre est impossible alors le véhicule notifie ses superviseurs et passe à l'ordre suivant. Il ne peut passer un ordre sans le valider ou notifier d'une annulation, sinon il reçoit une alerte.

Les différents ordres de régulation actuellement définis sont les suivants:

- Aller à la station S
- Aller au centre de réparation
- Vérifier la présence de X vélos non endommagés en station
- Vérifier la disponibilité de Y emplacements en station
- Déposés les vélos endommagés au centre de réparation
- Récupérer X vélos au centre de réparation
- Contrôler et réparer bornettes

Chaque véhicule doit renseigner le système d'informations des actions exécutées et de l'état des différents éléments selon le contexte:

- Réparation du vélo X sur place
- Chargement du vélo X dans le véhicule

- Dépôt du vélo X au centre de réparation
- Réparation bornette Y

Cette application doit prévenir d'une erreur si deux véhicules de régulation se dirigent vers la même station.

Enfin, les vélos endommagés doivent être emmenés au centre de réparation par des véhicules de régulation. Le centre de réparation est considéré comme une station disposant d'un espace de stockage infini.

4 Cahier des charges de l'application

La liste ci-dessous énumère les fonctionnalités essentielles que doit proposer l'application sur les trois interfaces disponibles : Interface Client, Interface des conducteurs de véhicule de régulation, interface des superviseurs.

Interface Client

1 – Abonnement au service. En renseignant les informations nécessaires, un client doit pouvoir s'abonner à l'année.

2 – Emprunt de vélo. Abonné ou non, un client doit pouvoir récupérer un vélo non endommagé à une borne. L'interface doit lui signaler s'il emprunte à une borne Vmoins ou Vplus, le code secret pour rendre le vélo s'il n'est pas abonné, ainsi que l'heure à laquelle il doit avoir rendu son vélo (douze heures après). le client doit pouvoir alerter de l'état d'un vélo en panne.

3 – Rendu de vélo. Un client qui a emprunté un vélo doit pouvoir le ramener à une station s'il y a un emplacement vide et qu'il connait le code secret. Il faudra vérifier que le vélo ramené est le même que celui emprunté (on une entrée manuelle du numéro du vélo pour simuler la vérification électronique par la bornette). Le client doit pouvoir informer de l'état d'un vélo rendu en panne.

4 – Gestion des amendes. Le système doit émettre une amende au bout de douze heures si le vélo n'a pas été rendu. S'il est rendu après les douze heures, l'application doit signaler au client que l'amende a été émise.

5 – Programme Vplus. L'application doit signaler à un client qui rend son vélo s'il bénéficie d'une remise Vplus pour son prochain trajet. Si le client est un abonné, l'application doit prendre cette remise en compte automatiquement à la prochaine location (et le signaler au client). S'il n'est pas un abonné, l'application doit lui émettre un code secret valable un mois. Il doit être possible pour un client non abonné de rentrer ce code pour bénéficier de la remise à chaque nouvelle location. Ce code ne doit être valable que pour une location.

6 – Affichage des bornes Vmoins/Vplus. A tout moment un client doit pouvoir se renseigner sur le statut de bornes privilégiées.

7 – Réservation d'un vélo. Un client peut réaliser une ou plusieurs réservations. Un affichage des disponibilités dans la période de réservation pour une station peut être une fonction intéressante pour aider l'utilisateur.

Interface Conducteur

1 – Déclarer un vélo endommagé. Un vélo endommagé ne peut plus être emprunté.

2 – Consulter sa routine. La routine du véhicule doit pouvoir s'afficher avec les actions à effectuer dans l'ordre.

3 – Déplacer les vélos. Un véhicule doit pouvoir signaler au système qu'il déplace les vélos depuis une station à son véhicule et l'inverse. Un véhicule ne doit pas pouvoir déposer en station un vélo endommagé.

4 – Valider sa routine. Un véhicule doit pouvoir signaler au système que son ordre actuel vient d'être effectué. LE SYSTEME DOIT POUVOIR VERIFIER SI C'EST BIEN LE CAS, c'est-à-dire que le nombre de vélos indiqué a bel et bien été déplacé. Si l'ordre n'est pas applicable

(pas assez ou trop de vélos dans le véhicule), une erreur est envoyée au superviseur et le véhocule peut passer à l'ordre suivant.

5 – Un véhicule doit pouvoir renseigner les réparations entreprises localement.

Interface Superviseur

1 – Consulter les routines d'un véhicule, ou consulter l'ordre actuel dans la routine de chaque véhicule.

2 – Consulter le nombre de vélos dans chaque station, le nombre de vélos endommagés, et le nombre de places libres.

3 – Consulter (et modifier) les plages horaires de "Vmoins / Vplus". Chaque modification ne doit prendre effet qu'à partir du lendemain.

4 – Savoir, pour une station, quel véhicule s'y arrête dans sa routine prévue (pour y faire quoi, et avec quel ordre de priorité), combien a-t'il de vélos, depuis combien de temps n'a-t'elle pas reçu la visite d'un véhicule.

5 – Modifier la routine d'un véhicule.

5 Modélisation du schéma de base de données

Cette question donnera lieu à un rapport intermédiaire.

Question 1: Vous donnerez le schéma conceptuel des données (UML ou entité-association), puis le schéma relationnel complet (tables, identifiants, identifiants externes, typage des attributs, contraintes référentiels et de domaines).
Les choix conceptuels devront être argumentés au besoin.

6 Analyse des contraintes d'intégrité

Cette question donnera lieu à un rapport intermédiaire.

Les contraintes énoncées dans les sections précédentes se décomposent en trois catégories :

1. les contraintes simples sur les données qui sont exprimables dans la description du schéma de la base de données (schéma SQL),
2. les contraintes complexes sur les données qui ne sont pas exprimables directement dans le schéma de la base mais à l'aide de déclencheurs disponibles dans une BD dynamique,
3. les contraintes applicatives qui sont gérées au sein de l'application.

Question 2: Donner la liste des contraintes d'intégrité, spécifiées et déduites des sections 3, 4 et 5, en précisant pour chaque contrainte si elle sera gérée au niveau du SGBD ou de l'application.

Question 3: Donner le script SQL (Oracle) qui implémente le schéma de la base de données énoncé en section 5. Vous peuplerez la base à l'aide d'un jeu d'essai judicieusement choisi (et commenté). Vous prendrez un grand soin dans le choix des types pour les dates et les durées.

7 Gestion de l'aspect transactionnel

Cette question donnera lieu à un rapport intermédiaire.

Question 4: La mise en œuvre des fonctionnalités énoncées dans la section 4 demande une décomposition de ces fonctionnalités en unités transactionnelles. Il faut pour cela:

1. identifier correctement les transactions,
2. choisir le niveau d'isolation pertinent pour chaque transaction,
3. développer un ensemble de scénarios de tests pour valider le bon fonctionnement d'une fonctionnalité en mode d'accès concurrent.

L'énoncé des transactions pourra se faire dans un pseudo langage mêlant algorithme (simplifiée) et SQL. L'objectif est de donner l'idée des tâches réalisées par chaque transaction.
L'intégralité des transactions devra apparaître dans ce document. Seules des corrections mineures pourront être acceptées dans le rapport final.

8 Réalisation de l'application

Cette partie donnera lieu à un rapport final qui fera la synthèse de votre travail sur ce projet.

Vous devrez réaliser une application Java qui met en œuvre les fonctionnalités énoncées dans la section 4.

L'exécution de l'application doit faire apparaître des traces démontrant la prise en compte des différents types de contraintes d'intégrité et la gestion correcte des transactions (en mode d'accès concurrent), et cela en limitant les saisies manuelles : le jour de la démo on évitera, dans la mesure du possible, de perdre du temps à devoir saisir bêtement de nombreuses données pour montrer un point important.

Chaque opération devra réaliser une trace permettant de faciliter au maximum la visualisation de la bonne marche de l'opération (contraintes prises en compte, transactions exécutées). Au contraire, le développement d'une interface utilisateur riche n'est absolument pas demandé, l'application doit juste proposer le minimum pour faciliter la saisie des informations et fluidifier le déroulement de vos scénarios lors de la soutenance.

Evidemment, plusieurs opérations concurrentes doivent être possibles sans provoquer d'incohérence et en favorisant la concurrence d'accès.

Ayant identifié les contraintes d'intégrité à la charge du SGBD, vous réaliserez les déclencheurs correspondants sous la forme de triggers Oracle. Des jeux d'essais complets devront valider la gestion des contraintes par le SGBD.

Réaliser en Java les opérations énoncées dans la section 4 en considérant les déclencheurs actifs au niveau de la base Oracle.Vous mettrez en évidence la bonne gestion des transactions dans un contexte d'accès concurrents.

9 Rapport final et soutenance

Le document (au format PDF) doit répondre aux différentes questions dans un français ou anglais correct.

Rapport final. Vous devrez faire l'effort d'être à la fois précis et concis. Le contenu du rapport doit reprendre les points suivants:

- Rappeler les objectifs du projet ainsi que l'organisation du document dans une brève introduction. Décrire les opérations de l'application (explications, code commenté, protocoles de tests).
- Faire le lien avec le contenu des rapports intermédiaires afin de mettre en valeur l'implémentation des contraintes et de la gestion concurrente. Faire notamment apparaître les éventuelles corrections apportées par rapport aux spécifications des rapports intermédiaires (gestion des contraintes et des transactions)
- Décrire les opérations de l'application (explications, code commenté, protocoles de tests).

Soutenance. Vous aurez environ 15 minutes (sur 30) à votre disposition pour nous faire une démonstration de votre application. Tous les membres d'un groupe sont conviés à la soutenance (sous réserve d'absence justifiée). Vous emploierez ce temps à:

- Réaliser la démonstration de votre application en laissant libre cours à vos talents de présentation. Le jury se laisse le droit d'interrompre la démonstration pour utiliser son propre jeu de tests et poser des questions. Il est très important de préparer des scripts pour éviter la saisie manuelle des données pour illustrer des cas complexes (l'interaction utilisateur n'est pas au centre de ce projet).
- Répondre aux questions posées à chacun des membres du groupe. Une pondération de la note globale pourra être envisagée à cette occasion.

10 Dates importantes

Afin de contrôler l'avancement du travail au fur et à mesure du projet, des documents intermédiaires devront être fournis aux enseignants à différentes étapes du projet.
Les dates clés du planning sont les suivantes :

- **Jeudi 8 janvier :** lancement du projet
- **mardi 13 janvier à 20h :** remise d'un rapport décrivant votre schéma conceptuel de données, votre démarche d'analyse, et votre schéma relationnel.
- **vendredi 16 janvier à 20h :** remise d'un rapport résumant l'analyse de toutes les contraintes d'intégrité du cahier des charges. Pour chaque contrainte, indiquer à quel(s) niveau(x) elle sera gérée : schéma, déclencheur(s), application.
- **mercredi 21 janvier à 20h :** remise d'un rapport décrivant les différentes séquences transactionnelles à mettre en œuvre au niveau de l'application. Pour chaque séquence, décrire :
    - le niveau d'isolation SQL correspondant
    - les conditions de validation (commit), d'annulation (rollback) et éventuellement d'annulation partielle (rollback to savepoint), en fonction des contraintes concernées.
    - des scénarios de concurrence d'accès (entrelacement de transactions) représentatifs du bon fonctionnement de l'application.

  (Les documents intermédiaires doivent être précis, concis et techniques, dans les 5 pages. Donc pas de rédaction verbeuse ou de recopie/paraphrase de l'énoncé pour décorer)
- **Jeudi 29 janvier à 20h :**
    - remise du rapport final de projet. Il n'est pas nécessaire d'inclure les listings du code du projet en annexe (cependant, des courts extraits représentatifs de code peuvent éventuellement être intégrés dans le corps du rapport si cela facilite certaines explications). Ce rapport doit contenir principalement le descriptif des scénarios de test/démonstration qui seront présentés lors de la soutenance.
    - Remise du code complet du projet, des jeux de tests et des scénarios scriptés (fichiers Java et SQL). Ce code devra être suffisamment commenté pour être compréhensible par des personnes extérieures à votre groupe de projet. Le code sera accompagné d'un fichier README précisant (i) les scripts à lancer pour créer la base, la peupler et initialiser les déclencheurs et (ii) les commandes de compilation et de lancement pour l'application java.

  (Le rapport final ne reprend pas les documents intermédiaires, il doit juste indiquer les modifications. Il doit résumer les fonctions réalisées et présenter les scénarios associés).
- **Vendredi 30 janvier :** Soutenances. durée 30mn.