

GTI660 - Bases de données multimédias

Présentation du laboratoire

Chargée de laboratoire: Mirna Awad

Plan

- Présentation des laboratoires
- Présentation du laboratoire 1

Projet

Objectif :

- Développer un système de base de données respectant une architecture trois tiers. Ce système proposera à l'utilisateur d'interroger, de modifier et d'interagir avec une base de données multimédia. Plus précisément, il s'agit d'un système de location de films.
- Se familiariser avec GitLab et les bonnes pratiques de Git (Git flow).
- Se familiariser avec Oracle.

Projet (suite)

Ce projet se divise en 3 laboratoires:

- Laboratoire 1 : Conception d'une base de données multimédias (9 heures)
- Laboratoire 2 : Application interactive pour la consultation de contenu multimédia (12 heures)
- Laboratoire 3 : Indexation par contenu des données multimédias (15 heures)

Laboratoire 1

- Objectifs

- Introduction à l'architecture des bases de données
- Création et modification des objets SQL
- Requêtes sur les objets de la base de données
- Validation des données

- Exigences :

- Concevoir le modèle conceptuel de la base de données comprenant:
 - Le nom des classes
 - Le nom et le type des attributs
 - Les contraintes sur les relations de spécialisation (disjointe/chevauchante, complète/incomplète)
 - Relations

Laboratoire 1 (suite 1)

- Concevoir le schéma relationnel de la base de données comprenant
 - Le nom des tables
 - Le nom, le type **Oracle** de chaque colonne
 - Les clés primaires et étrangères
 - Les relations
- Rédiger des scripts SQL créant les tables de votre schéma et les contraintes d'affaires (champs NOT NULL, UNIQUE, CHECK, FK, TRIGGERS...)
 - Un script de création de tables
 - Un script des contraintes
 - Un script de validation pour tout type de constraint créé (au moins une validation par type)
- Se connecter à la BD et exécuter votre/vos script(s) SQL pour la création des tables et des contraintes

Laboratoire 1 (suite 2)

- Rédiger un programme permettant l'insertion de données provenant des fichiers XML dans la BD. Le script peut être rédigé soit **SQL** ou en **Java** (JDBC).
- Script de validation de la base de données :
 - Le script doit valider les contraintes et l'insertion des données:
 - 21550 clients
 - 631 films
 - 4549 personnes

Évaluation

- Le travail se fait en équipes de 4 étudiants
- Date de remise : **8 Février 2021 avant 23h59**
- Votre travail devra comporter (voir grille de correction):
 - Modèle conceptuel de la BD (à remettre au début de la 1ere séance le 12 Janvier 2021).
 - Rapport de laboratoire, incluant les sections suivantes : introduction, analyse, conception (incluant le schéma relationnel), implémentation et discussion.
 - Fichiers *.sql de création des tables
 - Fichiers *.sql de contraintes
 - Programme d'insertion des données
 - Script de validation de la base de données

Procédure de remise

- La remise sera effectuée entièrement via Gitlab et Moodle (pas de courriel). La dernière version à avoir été transférée avant l'heure de la remise sera considérée comme votre remise. Si vous souhaitez qu'une autre version soit corrigée veuillez en aviser le chargé de laboratoire (en cas de retard, la pénalité prévue au plan de cours sera appliquée).
- Adresse du Gitlab. **Vous devez utiliser ce serveur continuellement afin de pouvoir suivre l'évolution du projet et y déposer tous les éléments de votre travail (rapports, scripts, B.D,...).**

À faire aujourd'hui:

Laboratoire 1:

- Lire le document de spécification SRS.pdf (Moodle)
- Lire la description et le contenu de la base de données en XML document « Contenu XML »
- Prendre connaissance du contenu du fichier requetes.pdf et en tenir compte pour la conception de la BD.
- Modèle conceptuel de la BD
- Planifier et déléguer le travail

Accès à la BD

Oracle SQL Developer :

- Serveur Oracle : gti660ora12c.logti.etsmtl.ca
- Port : 1521
- SID : GTI660

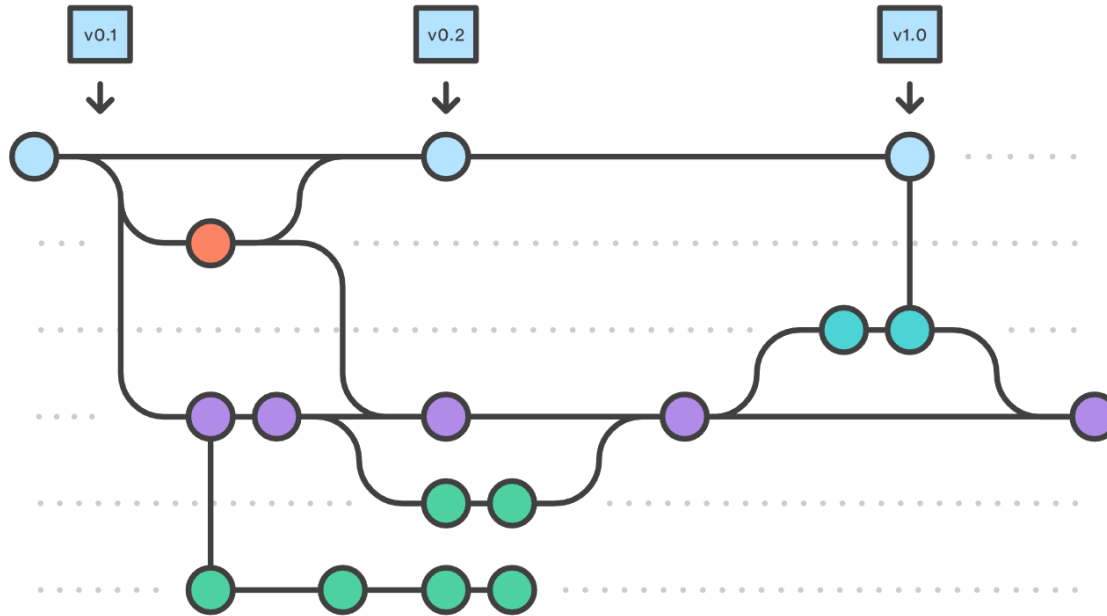
Bonnes pratiques SQL

- Utilisez des entiers comme clés primaires.
- Utilisez des jointures explicites.
- Utilisez des alias quand une expression possède plusieurs tables sources.
Indiquez les noms de colonnes dans vos requêtes d'insertions.
- Évitez les requêtes du genre `SELECT * FROM...`
- Évitez les contraintes dans la création de tables.
- Donnez un nom à chaque contrainte en suivant les standards: `PK_nomTable`, `FK_nomTable...`
- Écrivez les mots clés de SQL en majuscules.

Bonnes pratiques Développement SQL







- Assurez-vous de bien fermer vos requêtes (statement), vos connexions et vos résultats (ResultSet).
- Évitez de concaténer des chaînes de caractères pour générer des commandes SQL. Utilisez plutôt des requêtes paramétrées (prepared statement).

Bonnes pratiques Git : Git flow



Bonnes pratiques

Git : Branches

	Development branch Usually the integration branch for feature work and is often the default branch or a named branch. For pull request workflows, the branch where new feature branches are targeted.	master or develop
	Production branch Used for deploying a release. Branches from, and merges back into, the development branch. In a Gitflow-based workflow it is used to prepare for a new production release.	varies
	Feature branch Used for specific feature work or improvements. Generally branches from, and merges back into, the development branch, using pull requests.	feature/
	Release branch Used for release tasks and long-term maintenance versions. They branch from, and merge back into, the development branch.	release/
	Bugfix branch Typically used to fix Release branches.	bugfix/
	Hotfix branch Used to quickly fix a Production branch without interrupting changes in the development branch. In a Gitflow-based workflow, changes are usually merged into the production and development branches.	hotfix/

Références

Git flow : <https://fr.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow>

Branching a Repository: <https://confluence.atlassian.com/bitbucket/branching-a-repository-223217999.html>



Exemple de validation de contrainte NOT NULL

```
--client
DECLARE

    NullValues EXCEPTION;

    PRAGMA EXCEPTION_INIT(NullValues, -01400);

BEGIN

    INSERT INTO client (IDCLIENT) VALUES (NULL);
    dbms_output.put_line('NOT NULL Constraint not fonctionnal');

EXCEPTION

    WHEN NullValues
    THEN dbms_output.put_line('NOT NULL Constraint fonctionnal');

    WHEN OTHERS
    THEN dbms_output.put_line('error occured during validation');

END;
```



ÉTS

Le génie pour l'industrie