Institut supérieur d'informatique, de modélisation et de leurs applications - ISIMA
Filière 4 : Calcul et modélisation scientifiques

Gestion d'une Documentation Technique Discipline : Base de Données

Isaías FARIA

Documentation du travail pratique Gestion d'une Documentation Technique quant à la discipline des Base des Données du Institut supérieur d'informatique, de modélisation et de leurs applications - ISIMA

Aubière

Février - 2017

Table des matières

1 Introduction

L'objectif de ce travail est la création d'un système pour la gestion d'une documentation technique pour une entreprise comme travail pratique du cours de Base de Données.

Pour ce travail deux versions du système ont été créée. La première - *Version 1* - utilise des outils plus avancées. La deuxième version - *Version 2* a été créée après la révision par le professeur, cette version est plus simple et utilise seulement outils simples, comme Bootstrap et jQuery.

1.1 Contexte du problème

Le système a était crée pour une entreprise pour gérer la collection d'articles et de rapports. Pour ce problème il y a la notion de confidentialité. Un document peut être lu seulement pour leur auteur et par l'outres utilisateurs qu'ont le permission pour lire les documents confidentiels. Pour ce travail on a considéré que tous les utilisateurs peuvent écrire documents, comme rapports.

Un document peut avoir plusieurs auteurs et doit être écrit après 2005. Chaque emprunt doit être retourné au plus tard deux mois.

1.2 Code et démo

La démonstration du système est disponible sur le lien http ://isaiasfaria.com.br/isima_base_donne_tp1 et le code source sur GitHub, que peut être consulté par le lien https ://github.com/isaiasfsilva/isima_base_donne_tp1 .

2 Détails techniques

2.1 Technologies Utilisées

Pour ce travail les technologies suivantes on été utilisées : MySQL v5.5 comme le Système de gestion de base de données ; PHP 5.4 comme langage de programmation ; Apache 2.2.24 comme logiciel pour le serveur web.

Les outils de développement suivantes on été utilisées pour la *Version 1* du système : CodeIgniter comme *framework* base pour la langage de programmation PHP ; Grocery CRUD comme *CRUD (create, read, update and delete) framework* pour PHP ;

Bootstrap comme bibliothèque CSS ; Flexbox Grid comme l'interface de preséntation des donnés and JQuery comme bibliotèque JavaScript.

Pour la *Version 2* du système, les outils de développement suivantes on été utilisées : Bootstrap comme bibliothèque CSS et JQuery comme bibliotèque JavaScript.

2.2 Configuration Initiale

Le système est installé en quatre étapes :

1 - Décompressez le paquet.

2 - Ajouter les dossiers et les fichiers sur votre serveur. Le fichier *index.php* sera votre racine.

3 - Importer le fichier database.sql à votre serveur MySQL.

Pour la *Version 1 - OLD Version*

4 - Ouvrez le fichier *application/config/config.php* avec un éditeur de texte et définissez votre URL de base. Pour ce travail le url a été utilisée : *http ://isaiasfaria.com.br/isima_base_donne_tp1/.*

5 - Ouvrez le fichier *application/config/database.php* avec un éditeur de texte et définissez vos paramètres de base de données.

5.1 - Assurez-vous que votre serveur web apache est avec le module mod_rewrite activé pour les fichiers *.htaccess.*

Pour la *Version 2 - Nouvelle Version*

4 - Ouvrez le fichier *v2/classes/dbconfig.php* avec un éditeur de texte et définissez vos paramètres de base de données.

3 Modélisation

Dans la figure 1, vous pouvez voir le schéma conceptuel pour ce projet. Toutes les exigences du client ont été respectées.

Il a également été créé le schéma relationnel pour une meilleure représentation du modèle. C'est possible vérifier le type de chaque attribut. Le schéma relationnel est définie par les règles suivantes :

— DOCUMENTS(ID, ANNE,TITRE, NIV_CONFIG, TYPE)
— UTILISATEUR(ID, NOM, PRENOM, CONFIG)
— LIVRES_AUTEURS(*ID_DOC, ID_USER*, ORDRE)

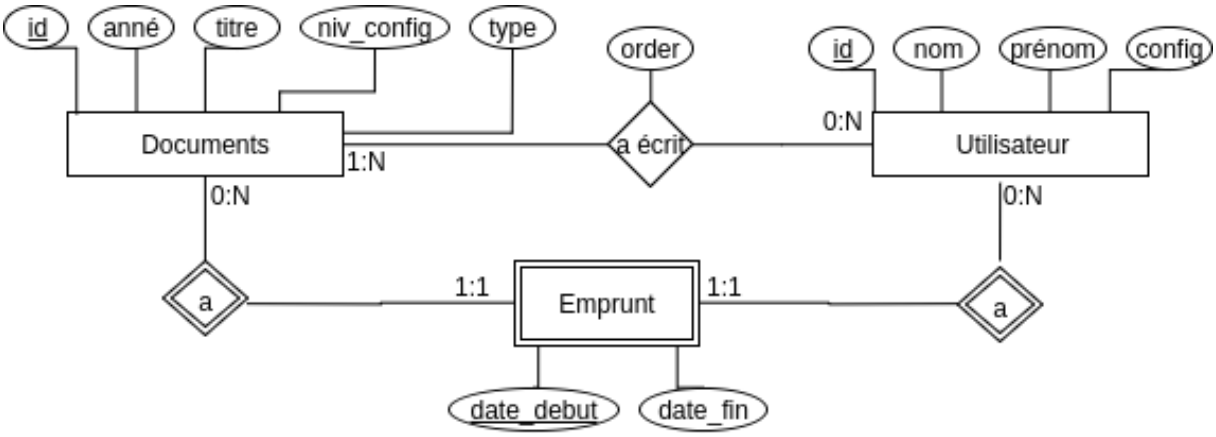

FIGURE 1 – Schéma conceptuel

— EMPRUNT(*ID_DOC*, DATE_DEBUT, *ID_USER*, DATE_FIN)
— EMPRUNT(ID_DOC) ⊆ DOCUMENTS(ID)
— EMPRUNT(ID_USER) ⊆ UTILISATEUR(ID)
— LIVRES_AUTEURS(ID_DOC) ⊆ DOCUMENTS(ID)
— LIVRES_AUTEURS(ID_USER) ⊆ UTILISATEUR(ID)

3.1 Normalisation

Pour une base de données bien structuré, c'est nécessaire normaliser dans la forme 3FN, et se possible dans la forme 3FNBCK. Pour la modélisation proposé c'est possible vérifier que tous les tableaux sont dans la forme 3FNBCK.

Tous les attributs de tous les tableaux sont des valeurs atomiques, alors ils sont dans la forme 1FN. C'est possible vérifier aussi que tous les attributs dépendent de tout le clé de chaque tableau, alors ils sont dans la forme 2FN aussi. Pour tout les dépendances fonctionnelles, la parte à gauche est toujours une superclé entière, alors le modèle est dans la forme 3FN et 3FNBCK.

3.2 Code pour la création

Après la création du modèle relationnel le code SQL a été créé pour la mise en œuvre de la base de données MySQL. Le code généré est le suivant :

```
-- Code pour les tableaux

CREATE TABLE `Documents` (
  `id` int(11) NOT NULL,
```

```
  `anee` year(4) DEFAULT NULL,
  `titre` varchar(255) COLLATE utf8_unicode_ci NOT NULL,
  `niv_config` tinyint(1) NOT NULL DEFAULT '0',
  `type` enum('Rapport','Lettre','Plan','Note','Contrat') COLLATE utf8_unicode_ci
  NOT NULL
) ENGINE=InnoDB AUTO_INCREMENT=1 DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

CREATE TABLE `Emprunt` (
  `id_doc` int(11) NOT NULL,
  `date_debut` date NOT NULL DEFAULT '0000-00-00',
  `date_fin` date DEFAULT NULL,
  `id_user` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

CREATE TABLE `Livres_auteurs` (
  `id_doc` int(11) NOT NULL,
  `id_user` int(11) NOT NULL,
  `ordre` int(11) NOT NULL DEFAULT '0'
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

CREATE TABLE `Utilisateur` (
  `id` int(11) NOT NULL,
  `nom` varchar(255) COLLATE utf8_unicode_ci NOT NULL,
  `prenom` varchar(255) COLLATE utf8_unicode_ci NOT NULL,
  `config` tinyint(1) NOT NULL
) ENGINE=InnoDB AUTO_INCREMENT=1 DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

3.3 Triggers et Functions

Pour garantir l'intégrité du modèle, il était nécessaire la création de *triggers* et *functions*. Chaque *trigger* présenté ici a été créée dans l'opération d'*insert* et dans l'opération d'*update*.

Deux *triggers* ont été créées : Une pour le tableau *Emprunt* pour vérifier la relation entre les dates et si l'utilisateur a la permission pour lire une document spécifié ; et la deuxième pour vérifier les informations des *Documents* insérés, comme la date de publication.

Une *function* a été créée pour faire la vérification si l'utilisateur a permission d'emprunter le document. Cette *function* est appelée par la première *trigger* présenté.

Procedures ont été créée pour calculer la quantité de Documents, Utilisateurs et Emprunts. Une *function* a été crée pour calculer la quantité de emprunts qui a la date_debut = aujourd'hui.

Le code pour créer les triggers, les fonctions, les procédures et la définition des clé est avec dans le fichier fariasilva_tp1.sql.

4 Conclusion

Pour ce travail était possible de vérifier que le travail de modélisation est vraiment important pour garantir l'intégrité des donnés et du système. Un petit schéma comme un système pour emprunt des documents il y a déjà beaucoup de contraints et restrictions pour vérifier.

C'est possible de vérifier qu'est un travail complexe et que demande temps et une bonne modélisation est essentielle pour éviter problèmes futurs.