Rapport Projet HLIN511

HENRIKSEN LAREZ Leif, SIMIONE Jeremy

Décembre 2019

DEPARTEMENT INFORMATIQUE
DE LA FACULTE DES SCIENCES

Table des matières

1 Introduction

Dans ce rapport, nous allons présenter notre projet du module HLIN511. Nous avons fait une base de données pour gérer des évènements. L'objectif de cette base est principalement d'enregistrer et manipuler des données sur les évènements et les visiteurs.

Nous allons parler, en premier lieu, des différentes tables, triggers et procédures que nous avons créés pour mettre en place cette base de données.

En second lieu, nous allons montrer et expliquer le modèle Entité/ Association de notre base et en dernier lieu nous allons donner une conclusion.

Pour ce projet, nous avons dû utiliser MySQL qui est un SGBD (i.e. un Système de Gestion de Base de Données) et est largement connu et exploité comme système de gestion de base de données pour des applications utilisant PHP. Pour collaborer à distance nous avons utiliser Github.

FIGURE 1 – Logo de MySQL

2 Fonctionnalités implémentées

2.1 Gestion des utilisateurs

Pour gérer le système d'utilisateurs et de rôles, nous avons mis au point une table nommée 'utilisateur' qui nous permet de créer des utilisateurs qui ont un nom, un prénom, un mot de passe, une adresse, un mail, et surtout un type d'utilisateur permettant d'attribuer un rôle à chaque utilisateur à savoir un rôle administrateur, contributeur ou tout simplement utilisateur.

2.2 Gestion des évènements

Pour la gestion d'évènements, nous avons réalisé une table 'événements' dans laquelle la création d'évènements est possible pour un contributeur ou un administrateur. Ensuite, nous expliquerons en détail comment nous avons empêché, grâce à la réalisation d'un trigger, la création d'un évènement par un simple utilisateur. Cette table contient aussi le thème de l'évènement qui est lié à la table thème et qui nous permettra donc de faire des recherches par thèmes des évènements. Les positions des évènements sont stockées dans la table via deux attributs qui correspondent à la longitude et la latitude de l'évènement et qui nous servira par la suite pour la partie web afin de placer des marqueurs sur une carte qui correspondra à la position de chaque évènement.

2.3 Inscription à un évènement

Pour qu'un utilisateur puisse s'inscrire à un évènement nous avons implémenté une table nommée visite qui permet de lier un utilisateur a un évènement en stockant l'identifiant de l'utilisateur souhaitant s'inscrire et l'identifiant de l'évènement auquel il veut s'inscrire. Cette liaison a été possible grâce à deux clés étrangères référencées sur les tables évènements et utilisateur.

2.4 Notation d'un évènement et moyenne

Sur notre site web, un utilisateur peut noter un évènement s'il est inscrit et si la date du-dit évènement est bien passée. Afin qu'un utilisateur puisse noter un évènement, nous avons créée une table rating qui va contenir les données relatives à un évènement et un utilisateur et une note de 1 à 5 mais aussi la date de l'évènement. La date nous sert à vérifier que l'évènement est bien passé, cette vérification est possible à nouveau grâce à un trigger dont l'explication sera détaillée par la suite. La table note est aussi liée via un trigger qui va, dès qu'une insertion est réalisée, dans la table rating, calculer la note moyenne de l'évènement.

2.5 Thèmes

Pour gérer les thèmes nous avons créée une forêt. Si un thème n'a pas de père (autrement dit la valeur de son père est 'NULL') alors ce thème est un thème racine. Nous avons aussi créé une table 'niveaux' qui contient les niveaux d'un thème dans un arbre. Cette table est remplie au fur et à mesure grâce à un trigger qui donne à chaque nouveau thème un niveau. Le niveau a pour valeur 0 si le thème est un thème racine, sinon le niveau de son père plus un. Nous avons fait ce système pour pouvoir accéder à tous les fils, grand fils, etc., d'un thème avec une seule requête.

```
/*trouver tous les fils d'un thème père*/
SELECT NT.NOM_THEME AS FILS
FROM THEME T, NIVEAU_THEME NT
WHERE T.NOM_THEME = NT.NOM_THEME
AND T.THEME_RACINE LIKE 'ancêtre commun'
AND NIVEAU > (SELECT NIVEAU FROM NIVEAU_THEME WHERE NOM_THEME LIKE père')
```

En gardant seulement le père d'un thème, nous n'avons pas réussi à trouver les grand fils sans faire des requêtes récursives avec PL/SQL.

2.6 Demande pour devenir contributeur

Dans notre site web, si un utilisateur le souhaite il peut devenir contributeur pour contrôler les demandes et éviter un trop grand nombre de contributeurs. Nous avons mis en place un système de contrôle, ce système consiste à rédiger une lettre de motivation et l'envoyer aux administrateurs. Les administrateurs peuvent ensuite lire les différentes demandes et ainsi les accepter ou les refuser. Pour gérer ce système nous avons créé une table 'DEMANDE' dans laquelle on garde la lettre de motivation et l'identifiant de l'utilisateur. Cet identifiant nous permet d'afficher des informations supplémentaires de l'utilisateur et aussi nous permet de faire un éventuel changement de statut.

3 Triggers et fonctions

3.1 Ton trigger (Leif)

3.2 Trigger pour la notation d'un évènement

Comme nous en avons parlé précédemment dans notre rapport, pour vérifier que la notation d'un évènement concerne bien un évènement passé nous avons créée un trigger.

Ce trigger va tout simplement vérifier **avant insertion** dans la table que la date de l'évènement est passée grâce à une comparaison de la date de l'insertion par rapport à la date du jour. Si l'évènement n'est pas passé il sera impossible de l'insérer dans la table rating et un message d'erreur sera affiché disant que l'évènement n'est pas encore passé.

3.3 Trigger pour la création de moyennes

Pour remplir automatiquement une table qui correspond à la moyenne des évènements, nous avons fait un trigger qui va **après insertion** dans la table rating calculer et insérer automatiquement dans la table moyenne la note moyenne de l'évènement.

Ce trigger fonctionne grâce à deux variables créés dans le trigger qui font différentes requêtes sur la table rating :
- une variable nommée id afin de connaître l'id de l'évènement pour savoir s'il avait déjà une moyenne et le comparer au nouvel id (de l'insertion)
- une autre nommée qui récupère la moyenne des notes de l'évènement.
Ensuite on supprime la ligne contenant l'ancienne moyenne de l'évènement s'il y en a un et insère la note moyenne mise à jour. 5

4 Modèle Entité/Association

Notre modèle contient deux entités principales, l'entité 'EVENEMENT' et l' entité 'UTILISATEUR'. L'entité 'UTILISATEUR' contient toutes les informations concernant un utilisateur, un utilisateur peut visiter plusieurs ou aucun évènements, il peut noter plusieurs évènements ou aucun, il peut aussi demander une seule fois de devenir contributeur, la lettre de motivation de cette demande est enregistrée dans l'entité 'DEMANDE_CONTRIBUTEUR' et chaque lettre peut appartenir à un unique utilisateur. L'entité 'EVENEMENT' contient tous les informations concernant un évènement, un évènement peut être visité par aucun ou plusieurs utilisateurs, il peut recevoir des notes par aucun ou plusieurs utilisateurs, il a aussi un unique thème. Notre modèle contient une autre entité importante, c'est l'entité THEME, cette entité nous permettra de réaliser des requêtes complexes sur les évènements. Un thème peut être le thème de aucun ou plusieurs évènements, il peut ne pas avoir de racine, ne pas avoir de père, il peut être le père de zéro ou plusieurs thèmes, il peut être la racine de plusieurs thèmes ou d'aucun, un thème contient aussi un niveau qui est le niveau du père plus un ou zéro si le thème n'a pas de père. Des qu'un utilisateur donne une note à un évènement cela remplit aussi l'entité 'MOYENNE' qui correspond à la moyenne des évènements. Une moyenne est donnée par zéro ou plusieurs notes en fonction du nombre de notes a l'évènement et un ensemble de note d'un évènement ne peut être liée qu'a une et une seule moyenne.

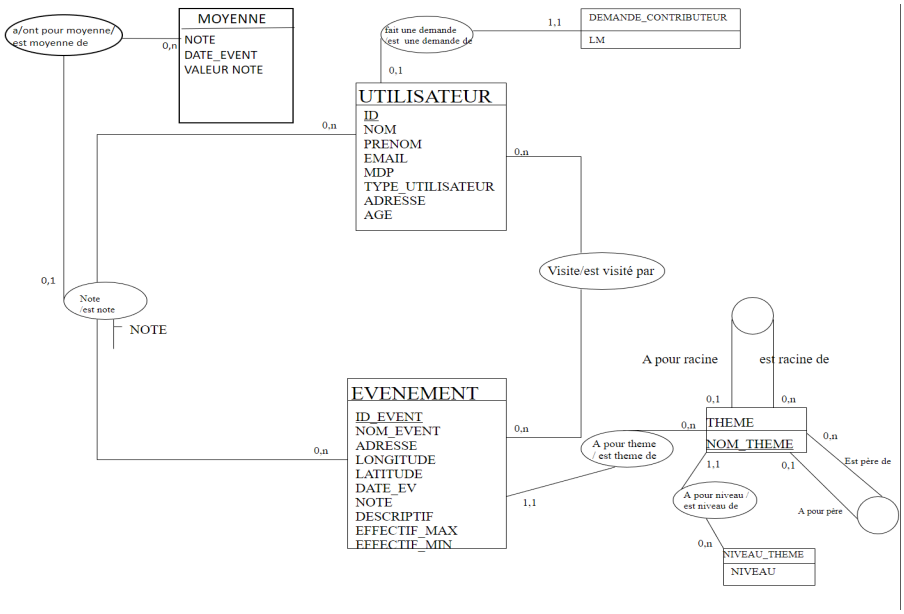

FIGURE 2 – Modèle entité-association de notre base de données

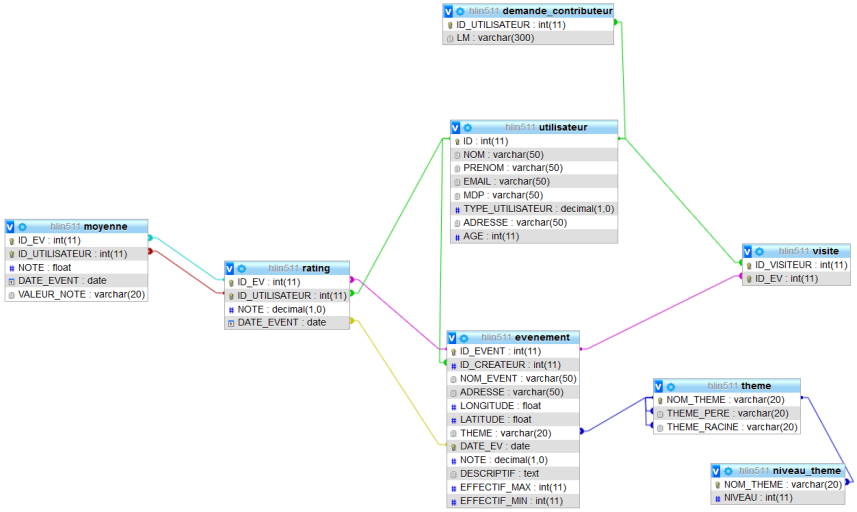

FIGURE 3 – Notre base de données modélisée par phpMyAdmin

5 Conclusion

Durant la réalisation de ce projet, nous avons beaucoup appris surtout lors de la création de triggers, nouveau concept pour nous assez difficiles à prendre en main surtout au niveau de la syntaxe. Nous avions déjà vu au cours de l'année précédente quelques exemples de créations de table en SQL, mais nous avions peu pratiqué.

Ce projet nous a donc permis d'en apprendre plus sur le langage SQL et nous a mis face à un cas concret d'utilisation de base de données. Nous avons réussi à implémenter toutes les fonctionnalités demandées et le projet web a pu ainsi être entièrement réalisé grâce à notre SGBD.