

Rapport Projet HLIN511

HENRIKSEN LAREZ Leif, SIMIONE Jeremy

Décembre 2019



**DEPARTEMENT INFORMATIQUE
DE LA FACULTE DES SCIENCES**

Table des matières

1	Introduction	2
2	Fonctionnalités implémentées	3
2.1	Gestion des utilisateurs	3
2.2	Gestion des événements	3
2.3	Inscription à un événement	3
2.4	Notation d'un événement et moyenne	3
2.5	Thèmes	4
2.6	Demande pour devenir contributeur	4
3	Triggers et fonctions	5
3.1	Ton trigger (Leif)	5
3.2	Trigger pour la notation d'un événement	5
3.3	Trigger pour la création de moyennes	5
4	Conclusion	6

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 créé pour mettre en place cette base de données, en second lieu nous allons montrer et expliquer le modèle Entité/Association de notre base, 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) qui 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, une adresse et surtout un type d'utilisateur qui nous a permis de pouvoir attribuer un rôle à chaque utilisateur à savoir un rôle admin, contributeur ou tout simplement utilisateur.

2.2 Gestion des événements

Pour la gestion d'événements nous avons fait une table 'événements' dans laquelle on peut créer des événements si c'est un contributeur ou un administrateur qui crée l'événement nous expliquerons en détail comment nous avons empêché grâce à un trigger la création d'un événement par un simple utilisateur. Cette table contient aussi le thème de l'événement qui est lié à la table thème et qui nous permettra donc de faire des recherches par thèmes des événements. Les positions des événements sont placées dans la table par deux attributs qui correspondent à la longitude et la latitude de l'événement et qui nous servira par la suite pour la partie web pour placer des marqueurs sur une carte qui correspondra à la position de chaque événement.

2.3 Inscription à un événement

Pour qu'un utilisateur puisse s'inscrire à un événement nous avons implémenté une table nommée visite qui permet de lier un utilisateur à un événement en stocker l'id de l'utilisateur souhaitant s'inscrire et l'id de l'événement auquel il veut s'inscrire. Cette liaison a été possible grâce à deux clés étrangères qui font des références sur les tables événements et utilisateur.

2.4 Notation d'un événement et moyenne

Sur notre site web un utilisateur peut noter un événement si il est inscrit et si la date dudit événement est bien passée. Afin qu'un utilisateur puisse noter un événement nous avons créé une table rating qui va contenir les données relatives à un événement et un utilisateur et une note de 1 à 5 mais aussi la date de l'événement. La date nous sert à vérifier que l'événement est bien passé, cette vérification est possible à nouveau grâce à un trigger que nous expliquerons plus en détail par la suite. La table note est aussi liée via un trigger qui va dès qu'une insertion est faite dans la table rating, calculer la note moyenne de l'événement.

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 créée une table 'niveaux', qui contient les niveaux d'un thème dans un arbre, cette table est rempli 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 niveaux de son père plus un. Nous avons fait cette système pour pouvoir accéder a 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 récursive 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 demande 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, ensuite les administrateurs peuvent lire les différentes demandes et ainsi les accepter ou les refuser. Pour gérer ce système nous avons créée 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 l'insertion dans la table** que la date de l'évènement est passé grâce a 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 encroe 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. ce trigger fonctionne grâce a deux variables propres au trigger qui font en fait différentes requetes 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 le tuple contenant l'ancienne moyenne de l'évènement s'il y en a un et insere la note moyenne mise à jour.

4 Conclusion

Au cours de ce projet nous avons beaucoup appris surtout lors de la création de triggers qui étaient un nouveau concept pour nous et qui sont assez difficiles à prendre en main surtout au niveau de la syntaxe. Nous avons déjà vu au cours de l'année précédente quelques exemples de créations de table en SQL, mais nous avons peu pratiqué. Ce projet nous a donc permis d'en apprendre plus sur le langage SQL et nous a mis face d'un cas concret d'utilisation de base de données. Nous avons réussi à implémenter toute les fonctionnalités demandés et le projet web a pu ainsi être entièrement réalisé grâce à notre SGBD.