Projet « todolist » 

Document de conception de la base de données

(Version de travail du 22/01/2022)

Table des matières

[Le cahier des charges 2](#_Toc125295799)

[Le dictionnaire de données 3](#_Toc125295800)

[Le modèle Conceptuel de données (MCD) 5](#_Toc125295801)

[Le modèle logique de données (modèle relationnel) 7](#_Toc125295802)

[Interroger la base « todolist » 9](#_Toc125295803)

[Code de génération du dataset 9](#_Toc125295804)

[Requêtes SQL de test de la base de données générée 9](#_Toc125295805)

# Le cahier des charges

Le projet « Todolist » vise à créer un espace membre dans lequel chaque utilisateur peut créer ses listes de tâches, les consulter, les modifier et les supprimer.

Le système fournit les éléments suivants :

* Inscription d’un membre sur le site
* Connexion du membre sur le site
* Création de sa liste de tâches
* Gestion de sa liste de tâches
* Consultation d’un forum où le membre peut parler de ses difficultés avec ses tâches
* Réponse des membres ou modérateurs/administrateurs

**Requêtes à faire :**

1. Liste des listes de tâches pour un membre particulier
2. Liste des tâche pour une liste particulière d’un membre particulier
3. Liste des tâches en retard pour un membre
4. Liste des tâches faites pour un membre
5. Nombre de tâches en retard pour un membre

# Le dictionnaire de données

|  |  |  |
| --- | --- | --- |
| **Nom** | **Description** | **Type** |
| **Membres** | **Liste des membres** | **Entité** |
| * id | Identifiant du membre | AI |
| * pseudo | Pseudo du membre | Varchar |
| * mdp | Mot de passe du membre | Varchar |
| * photo | Chemin vers l’avatar du membre | Varchar |
| * role | Indique le rôle de l’utilisateur (a pour « admin », « m » pour membre) | Char |
| **Categories** | **Catégories des tâches envoyés par des membres** | **Entité** |
| * id | Identifiant du membre | AI |
| * categorie | Nom de la catégorie contenant des tâches | Varchar |
| **Tâches** | **Ensemble des tâches que chaque membre planifie** | **Entité** |
| * id | Identifiant de la tâche | AI |
| * nom\_tache | Nom de la tâche planfiiée | Varchar |
| * description | Description de la tâche planifiée | Varchar |
| * date | Date de la réalisation de la tâche | date |
| * date\_rappel | Date de rappel de la réalisation de la tâche | datetime |
| * importance | Importance de la tâche, de 1 à 3 | Int |
| * complete | Indique si la tâche est terminée ou non | bool |
| **Repertorier** | Une liste répertorie une ou plusieurs tâches alors qu’une tâche est répertoriée dans une seule liste | **Association** |
| Listes | 1,n | Cardinalité |
| Taches | 1,1 | Cardinalité |
| **Creer** | Un membre crée plusieurs listes et une liste est créée par un seul membre | **Association** |
| Membre | 1, n | Cardinalité |
| Liste | 1,1 | Cardinalité |

**Règles de gestion :**

* Chaque membre peut avoir autant de listes qu’il souhaite
* Chaque membre peut avoir autant de tâches qu’il souhaite
* Chaque tâche n’est rattachée qu’à une unique liste
* (Partie forum UNDER PROGRESS)

# Le modèle Conceptuel de données (MCD)

<MCD>  
 <entités>

\*Membres  
- id (AI)   
- identifiant (varchar(500))  
- mot\_de\_passe (varchar(500))  
- photo (varchar(500))  
- role bool(1)

\* Listes  
- id (AI)  
- nom (varchar(500))  
- idenifiant (varchar(500))  
- mot\_de\_passe (varchar(500))  
- email (varchar(500))

\* Taches  
- id (AI)  
- statut (int)  
- nom (varchar(500))  
- adresse (varchar(500))  
- departement (int)  
- description (text)  
- longitude (float)  
- latitude (float)  
</entités>  
<association>  
Prendre  
- Client 1,n  
- Réservation 1,1

Conclure  
- Reservation 1,1  
- Hotel 1,n  
</association>

</MCD>

**(refaire le graphique)**

*MCD graphique*

# Le modèle logique de données (modèle relationnel)

<MLD>  
client  
 - cli\_id (AI)  
 - cli\_nom (varchar(500))  
 - cli\_identifiant (varchar(500))  
 - cli\_mdp (varchar(500))  
 - cli\_email (varchar(500))

personnel  
 - per\_id (AI)  
 - per\_nom (varchar(500))  
 - per\_idenifiant (varchar(500))  
 - per\_mdp (varchar(500))  
 - per\_email (varchar(500))  
 - per\_role (varchar(500))  
 - per\_hotel (FK)

</MLD>

**Créer le moduèle graphique**

*Modèle logique de données graphique*

**Script SQL :** voir “vivehotel.sql”

# Interroger la base « todolist »

## Code de génération du dataset

Dans le sous-dossier « dataset » du dossier de projet, nous avons créé un fichier nommé « index.php ». Ce fichier permet de créer la base de données sur le serveur SQL, à l’adresse spécifiée dans la classe « monSQL » disponible dans le dossier « » Le script inclut ensuite le code de génération de chaque table de la base de données, disponible dans le même dossier. Les scripts de générations de chaque table de la base de données sont dépendants les uns des autres. **Il est alors nécessaire de lancer le fichier « index.php » au lieu de les lancer un par un.**

Une fois ce fichier installé, nous avons obtenu ce nombre d’insertions pour chaque table

|  |  |  |
| --- | --- | --- |
| Table | Nombre entrées | Nombre fie ou variable |
| [chambre](http://localhost/phpmyadmin/index.php?route=/sql&db=vivehotel&table=chambre&pos=0) | 500 | fixe |
| [chcategorie](http://localhost/phpmyadmin/index.php?route=/sql&db=vivehotel&table=chcategorie&pos=0) | 4 |  |
| [client](http://localhost/phpmyadmin/index.php?route=/sql&db=vivehotel&table=client&pos=0) | 20 | fixe |

*Taille (en nombre d’entrées) de chaque table*

Le script de la base de données semble s’exécuter en 6 secondes en moyenne.

Nous allons maintenant lancer des requêtes SQL pour interroger la donnée que nous avons généré par les scripts de génération de table.

## Requêtes SQL de test de la base de données générée

Ci-dessous, l’aperçu des résultats des requêtes SQL qui sont dans le fichier « requetes.sql » :