RAPPORT PROJET DB :
AYOUB ENEJJAR / 2333876 / GR02
HAMZA FELLAH / GRR02

Algèbre Relationnelle:

3-a)

Π Id_Reservation, Nom_complet, Ville_Hotel, Date_arrivee, Date_depart (ρ Nom_Client←Nom_complet, Ville_Hotel←Hotel.Ville ((Client ⟕Client.Id_Client = Reservation.Id_Client Reservation) ⟕Reservation.Id_Chambre = Chambre.Id_Chambre (Chambre ⟕Chambre.Id_Hotel = Hotel.Id_Hotel Hotel)))

3-b)

Π Nom_complet, Adresse, Email (σ Ville = 'Paris' (Client))

3-c)

Client.Id_Client, Nom_complet G COUNT(Id_Reservation) → Nombre_Reservations (Client ⟕Client.Id_Client Reservation.Id_Client Reservation)

//Note : ⟕représente la jointure externe gauche

3-d)

Type_Chambre.Id_Type, Type G COUNT(Id_Chambre) → Nombre_Chambres (Type_Chambre ⟕Type_Chambre.Id_Type =Chambre.Id_TypeChambre)

//Alternative textuelle : "Agrégation sur le résultat de la jointure externe gauche entre Type_Chambre et Chambre, groupée par Id_Type et Type, en comptant le nombre d'Id_Chambre."

3-e)
3-e-1)
Chambres_Reservees_Periode (CRP) ←
Π Id_Chambre (
σ (Date_arrivee < P_FIN) AND(Date_depart >P_DEBUT)(Reservation))

3-e-2)
Π Attributs_Chambre_Souhaités (
(Chambre ⋈Hotel ⋈Type_Chambre) - ((Chambre ⋈Hotel ⋈ Type_Chambre) ⋈Chambre.Id_Chambre = CRP.Id_Chambre CRP)
)

4) Question Théorique : SQLite vs MySQL

SQLite:

Est un moteur de base de données relationnelle embarqué, sans serveur (serverless), autonome, transactionnel et sans configuration ("zero-configuration").
La base de données entière (définitions, tables, index, et données) est stockée dans un unique fichier sur le disque.
Il est très populaire pour les applications de bureau, les navigateurs web (pour le stockage local), les applications mobiles (Android, iOS), et les petits sites web à faible trafic.
Il est écrit en C et est disponible en tant que bibliothèque que l'on intègre directement dans l'application.

Différences avec MySQL:

Architecture:

MySQL: Système de gestion de base de données client-serveur. Un processus serveur MySQL s'exécute en permanence, et les applications clientes s'y connectent (souvent via le réseau).
SQLite: Embarqué. La bibliothèque SQLite s'exécute dans le même processus que l'application. Pas de serveur séparé.
Stockage:
MySQL: Stocke les données dans plusieurs fichiers gérés par le serveur, avec une structure de répertoires complexe.
SQLite: Stocke toute la base de données dans un seul fichier .sqlite ou .db.
Typage des données:

MySQL: Typage statique et strict. Le type d'une colonne est fixe, et MySQL essaiera de convertir les données au type de la colonne, générant une erreur si ce n'est pas possible.
SQLite: Typage dynamique (ou "flexible typing"). Une colonne peut stocker différents types de données. Le type déclaré pour une colonne est une "affinité" (par exemple, une colonne déclarée INTEGER peut stocker du texte, mais SQLite essaiera de le convertir en entier si possible).
Concurrence:
MySQL: Conçu pour une forte concurrence, gérant de multiples connexions clientes simultanées avec des mécanismes de verrouillage sophistiqués (au niveau des lignes, des tables).
SQLite: Gère la concurrence au niveau du fichier de base de données (un seul processus peut écrire à la fois, bien que plusieurs puissent lire). Moins adapté pour les applications à forte charge d'écriture concurrente.
Scalabilité:
MySQL: Peut gérer des bases de données très volumineuses et un grand nombre d'utilisateurs. Supporte la réplication, le clustering.
SQLite: Mieux adapté pour des bases de données plus petites et des applications où la base de données est locale ou pour un utilisateur/petit groupe.
Fonctionnalités:
MySQL: Riche en fonctionnalités : procédures stockées, triggers (SQLite en a aussi, mais moins développés), vues matérialisées (selon version), gestion fine des utilisateurs et des permissions, etc.
SQLite: Ensemble de fonctionnalités SQL plus restreint, bien que très complet pour la plupart des besoins. Par exemple, ALTER TABLE a des limitations.
Administration:
MySQL: Nécessite une administration (installation, configuration, sauvegarde, gestion des utilisateurs, optimisation).
SQLite: Aucune administration requise. Le fichier de la base de données peut être simplement copié pour une sauvegarde.
Cas d'usage typiques:
MySQL: Applications web et d'entreprise, systèmes nécessitant une haute disponibilité et une forte concurrence.
SQLite: Applications mobiles, applications de bureau, stockage de cache, bases de données de configuration, tests unitaires pour des applications qui utilisent d'autres SGBD.