Algèbre Relationnelle pour les Requêtes de Gestion d'Hôtel

3. Ecrire les requêtes SQL ET en algèbre relationnelle qui permettent de:

a. Afficher la liste des réservations avec le nom du client et la ville de l'hôtel

π_{id_reservation, date_debut, date_fin, nom, ville} (

(Reservation ⋈_{Reservation.id_client = Client.id_client} Client)

⋈_{Reservation.id_reservation = Chambre.id_chambre} (

Chambre ⋈_{Chambre.id_hotel = Hotel.id_hotel} Hotel

)

)

réservé.

b. Afficher les clients qui habitent à Paris.

π_{nom, adresse, email} (σ_{ville = 'Paris'} (Client))

c. Calculer le nombre de réservations faites par chaque client.

γ_{id_client, nom, COUNT(id_reservation) → nombre_reservations} (

Client ⋈_{Client.id_client = Reservation.id_client} Reservation

)

d. Donner le nombre de chambres pour chaque type de chambre.

γ_{id_type, nom_type, COUNT(id_chambre) → nombre_chambres} (

Type_Chambre ⋈_{Type_Chambre.id_type = Chambre.id_type} Chambre

)

e. Afficher la liste des chambres qui ne sont pas réservées pour une période donnée (entre deux dates saisies par l'utilisateur).

π_{id_chambre, numero, ville, nom_type} (

(Chambre ⋈_{Chambre.id_hotel = Hotel.id_hotel} Hotel)

⋈_{Chambre.id_type = Type_Chambre.id_type} Type_Chambre

- π_{id_chambre} (

σ_{date_debut ≤ '2025-07-10' ∧ date_fin ≥ '2025-07-01'} (Reservation)

)

)

4. Qu'est ce que SQLite, quelle différence avec MySQL?

SQLite :

- Base de données légère, sans serveur, stockée dans un seul fichier (par exemple, hotel.db).
- Idéal pour les applications locales, mobiles ou embarquées.
- Pas de gestion d'utilisateurs ou de connexions concurrentes avancées.
- Syntaxe SQL simplifiée, moins de fonctionnalités avancées.

MySQL :

- Système de gestion de bases de données relationnelles avec serveur.
- Convient aux applications web avec plusieurs utilisateurs et grandes bases de données.
- Supporte des fonctionnalités avancées comme les triggers, les procédures stockées, et la gestion des privilèges.
- Nécessite une installation et une configuration du serveur.