République Algérienne Démocratique et Populaire
الجمهورية الجزائرية الديمقراطية الشعبية
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

جامعة هواري بومدين للعلوم و التكنولوجيا
Université des Sciences et de la Technologie
Houari Boumediene

Module : Base de données avancées

Rapport de projet de TP

Spécialité : Systèmes Informatiques Intelligents

Réalisé par :

— Djenane Nihad

— M'Barek Lydia

Travail demandé par :

S.BOUKHEDOUMA - Cours

Promotion : 2022/2023

Partie 1 :

Relationnel - Objet

1 Modélisation orientée objet

1.1 Transformez ce schéma relationnel en un schéma Objet (diagramme de classes)

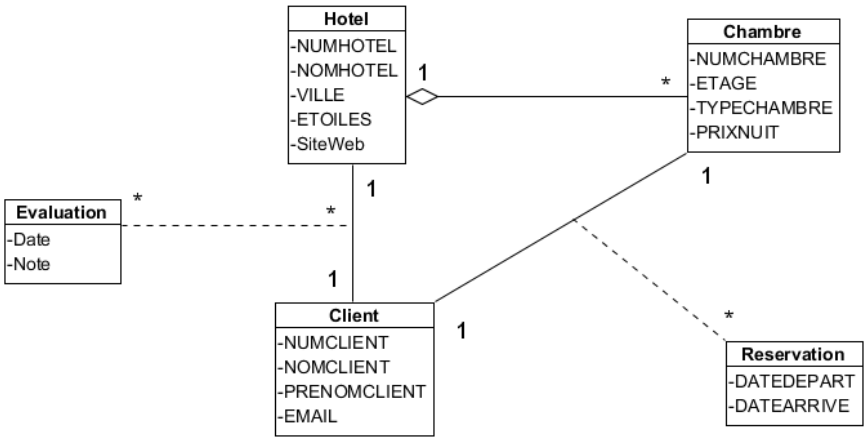

FIGURE 1 – Diagramme de classe

2 Création des TableSpaces et utilisateur

2.1 Créer deux TableSpaces SQL3_TBS et SQL3_TempTBS

```
SQL> CREATE TABLESPACE SQL3_TBS DATAFILE 'C:\SQL3_TBS.dat'  SIZE 100M  AUTOEXTEND ON   ONLINE;

Tablespace crÚÚ.

SQL> CREATE TEMPORARY TABLESPACE SQL3_TEMPTBS TEMPFILE 'C:\SQL3_TEMPTBS.dat' SIZE 100M  AUTOEXTEND ON;

Tablespace crÚÚ.
```

FIGURE 2 – Executer la commande

2.2 Créer un utilisateur SQL3 en lui attribuant les deux tablespaces créés précédemment

```
SQL> create user SQL identified by 1976 Default Tablespace SQL3_TBS Temporary Tablespace SQL3_TEMPTBS;

Utilisateur crÚÚ.
```

FIGURE 3 – Executer la commande

2.3 Donner tous les privilèges à cet utilisateur

```
SQL> grant all privileges to SQL;

Autorisation de privilÞges (GRANT) acceptÚe.
```

FIGURE 4 – Executer la commande

3 Langage de définition de données

3.1 Définition des types

```
SQL> create type THotel ;
  2  /

Type crÚÚ.

SQL> create type TClient ;
  2  /

Type crÚÚ.

SQL> create type TChambre ;
  2  /

Type crÚÚ.

SQL> create type TEvaluation as object (dateEv date, note integer, evaluation_hotel ref THotel, evaluation_client ref TClient);
  2  /

Type crÚÚ.

SQL> create or replace type TReservation as object (dateArr date, dateDepart date, reservation_client ref TClient, reservation_chambre ref TChambre);
  2  /

Type crÚÚ.

SQL> create type ref_evaluation as table of ref TEvaluation;
  2  /

Type crÚÚ.

SQL> create type ref_reservation as table of ref TReservation;
  2  /

Type crÚÚ.

SQL> create type ref_chambre as table of ref TChambre;
  2  /

Type crÚÚ.
```

FIGURE 5 – Executer la commande

```
SQL> create or replace type THotel as object (
  2      numHotel INTEGER,
  3      nomHotel varchar(50),
  4      ville varchar(50),
  5      etoile INTEGER,
  6      siteWeb varchar(50),
  7      hotel_evaluation ref_evaluation,
  8      hotel_chambre ref_chambre
  9  )Not final;
 10  /

Type crÚÚ.

SQL> create or replace type TClient as object (
  2      num_client INTEGER,
  3      nom_client varchar(50),
  4      prenom_client varchar(50),
  5      email varchar(50),
  6      client_evaluation ref_evaluation,
  7      client_reservation ref_reservation
  8  )Not final;
  9  /

Type crÚÚ.

SQL> create or replace type TChambre as object (
  2      num_chambre INTEGER,
  3      etage INTEGER,
  4      typeChambre varchar(50),
  5      prix_nuit INTEGER,
  6      chambre_hotel ref THotel,
  7      chambre_reservation ref_reservation
  8  )Not final;
  9  /

Type crÚÚ.
```

FIGURE 6 – Executer la commande - suite

Calculer pour chaque client, le nombre de réservations effectuées

```
SQL> alter type TClient add member function nb_reservation return INTEGER cascade;

Type modifiÚ.

SQL> create or replace type body TClient
  2  as member function nb_reservation return INTEGER
  3  is
  4  nbreservation integer;
  5  Begin
  6  select count(*)into nbreservation
  7  from table(self.client_reservation);
  8  return nbreservation;
  9  END;
 10  END;
 11  /

Corps de type crÚÚ.
```

FIGURE 7 – Executer la commande

Calculer pour chaque hôtel, le nombre de chambres

```
SQL> alter type THotel add member function nb_hotel return INTEGER cascade;

Type modifiÚ.

SQL> create or replace type body THotel
  2  as member function nb_hotel return integer
  3  is
  4  nbhotel integer;
  5  Begin
  6  select count(*) into nbhotel
  7  from table(self.hotel_chambre);
  8  return nbhotel;
  9  END;
 10  END;
 11  /

Corps de type crÚÚ.
```

FIGURE 8 – Executer la commande

Calculer pour chaque chambre, son chiffre d'affaire

```
SQL> alter type TChambre add member function chiffre_affaire return numeric cascade;

Type modifiÚ.
```

FIGURE 9 – Executer la commande

```
SQL> create or replace type body TChambre
  2  as member function chiffre_affaire return numeric
  3  is
  4  chiffreAffaire number;
  5  Begin
  6  select sum((value(deref(self.chambre_reservation).dateDepart) - value(deref(self.chambre_reservation).dateArr)) * self.prix_nuit) into chiffreAffaire
  7  from table(self.chambre_reservation) ;
  8  return chiffreAffaire;
  9  END;
 10  END;
 11  /
```

FIGURE 10 – Executer la commande - suite

Calculer pour chaque hôtel, le nombre d'évaluations reçues à une date donnée (01-01-2022)

```
SQL> alter type THotel add member function nb_evaluation return INTEGER cascade;

Type modifiÚ.
```

FIGURE 11 – Executer la commande

```
SQL> create or replace type body THotel
  2  as member function  nb_evaluation return INTEGER
  3  is
  4  nbevaluation integer;
  5  Begin
  6  select count(*) into nbevaluation
  7  from table(self.hotel_evaluation) e
  8  where deref(e.dateEv) = '01/01/2022';
  9  END;
 10  END;
 11  /
```

FIGURE 12 – Executer la commande

3.2 Définir les tables nécessaires à la base de données

```
SQL> create table Hotel of thotel (PRIMARY KEY(numHotel),constraint ck_type check (etoile >= 1 and etoile <= 10 ))
  2  nested table hotel_evaluation store as table_hotel_evaluation ,
  3  nested table Hotel_chambre store as table_Hotel_chambre ;

Table crÚÚe.

SQL> create table Chambre of tchambre(foreign key(chambre_Hotel) references Hotel,constraint ck_et check (typeChambre in ( 'simple', 'double', 'triple','suite','autre')))
  2  nested table chambre_reservation store as table_chambre_reservation ;

Table crÚÚe.

SQL> create table Client of tclient (PRIMARY KEY(num_client))
  2  nested table client_reservation store as table_client_reservation,
  3  nested table client_evaluation store as table_client_evaluation ;

Table crÚÚe.

SQL> create table Evaluation of tevaluation (foreign key(evaluation_hotel) references Hotel, foreign key (evaluation_client) references Client, constraint ck_note check (no
te >= 1 and note <= 10 ));

Table crÚÚe.

SQL> create table Reservation of treservation(foreign key(reservation_client) references Client,foreign key(reservation_chambre) references Chambre,constraint ck_date check
 (dateDepart >= dateArr));

Table crÚÚe.
```

FIGURE 13 – Executer la commande

4 Langage de manipulation de données

Voici quelques exemples d'insertion de données dans les tables tables :

```
SQL> insert into Hotel values (4,'Saint George dAlger', 'Alger', 5, '', ref_evaluation(), ref_chambre());
1 ligne crÚÚe.
SQL> insert into Hotel values (5,'Ibis Alger Aéroport', 'Alger', 2, '', ref_evaluation(), ref_chambre());
1 ligne crÚÚe.
SQL> insert into Hotel values (6,'El Mountazah Annaba', 'Annaba', 3, '', ref_evaluation(), ref_chambre());
1 ligne crÚÚe.
SQL> insert into Hotel values (7,'Hôtel Albert 1er', 'Alger', 3, '', ref_evaluation(), ref_chambre());
1 ligne crÚÚe.
SQL> insert into Hotel values (8,'Chems', 'Oran', 2, '', ref_evaluation(), ref_chambre());
```

FIGURE 14 – Exemple d'insertion de données dans la table hotel

```
SQL> insert into Client values (59 ,'IGOUDJIL', 'Redouane', '', ref_evaluation(), ref_reservation());
1 ligne crÚÚe.
SQL> insert into Client values (60 ,'GHEZALI', 'Lakhdar', '', ref_evaluation(), ref_reservation());
1 ligne crÚÚe.
SQL> insert into Client values (61 ,'KOULA', 'Brahim', '', ref_evaluation(), ref_reservation());
1 ligne crÚÚe.
SQL> insert into Client values (62 ,'BELAID', 'Layachi', '', ref_evaluation(), ref_reservation());
1 ligne crÚÚe.
SQL> insert into Client values (63 ,'CHALABI' ,'Mourad', '', ref_evaluation(), ref_reservation());
1 ligne crÚÚe.
```

FIGURE 15 – Exemple d'insertion de données dans la table client

```
SQL> insert into Chambre values (7, 0, 'simple', 7800, (select ref(h) from Hotel h where h.numHotel=2), ref_reservation());
1 ligne crÚÚe.
SQL> insert into Chambre values (8, 2, 'double', 8600, (select ref(h) from Hotel h where h.numHotel=11), ref_reservation());
1 ligne crÚÚe.
SQL> insert into Chambre values (8, 0, 'simple', 7800, (select ref(h) from Hotel h where h.numHotel=2), ref_reservation());
1 ligne crÚÚe.
SQL> insert into Chambre values (9, 3, 'suite', 16000, (select ref(h) from Hotel h where h.numHotel=11), ref_reservation());
1 ligne crÚÚe.
```

FIGURE 16 – Exemple d'insertion de données dans la table chambre

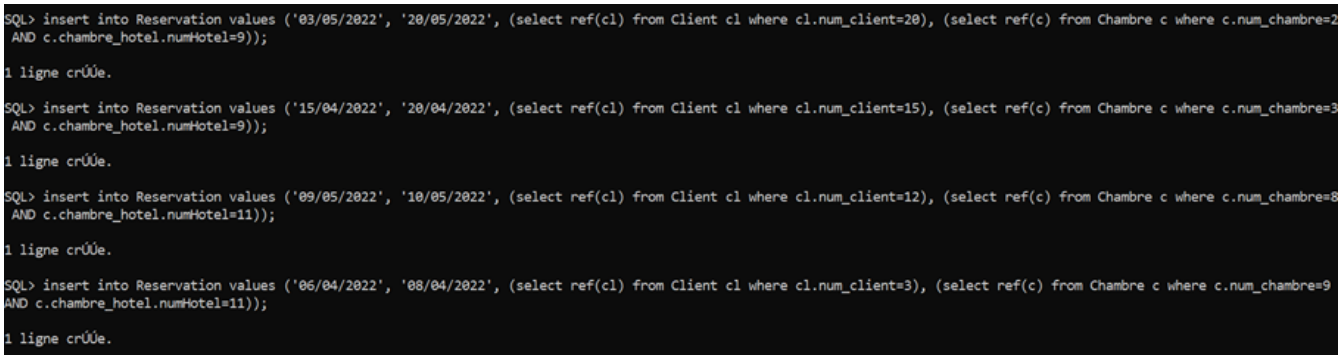

FIGURE 17 – Exemple d'insertion de données dans la table reservation

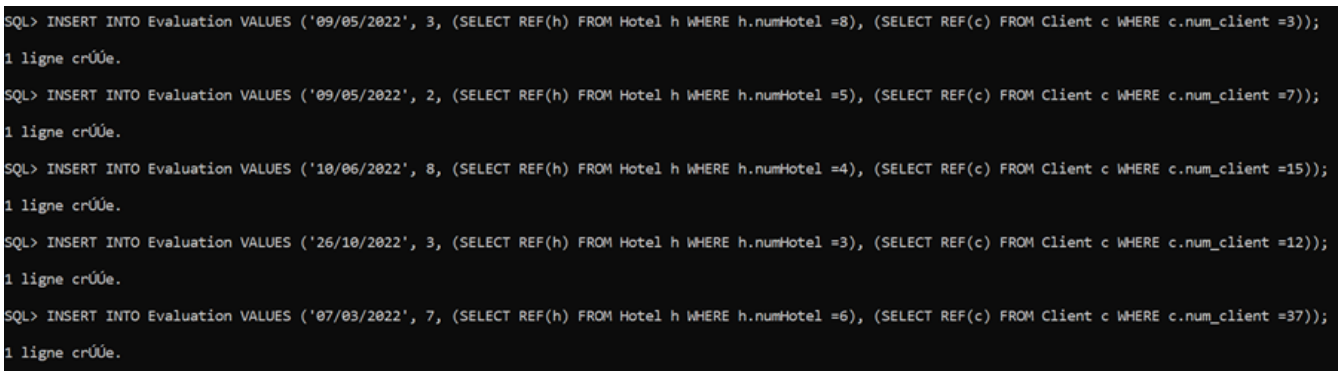

FIGURE 18 – Exemple d'insertion de données dans la table evaluation

5 Langage d'interrogation de données

5.1 Lister les noms d'hôtels et leurs villes respectives

```
SQL> select nomHotel, ville from Hotel order by nomHotel;

NOMHOTEL
--------------------------------------------------
VILLE
--------------------------------------------------
Chems
Oran

Colombe
Oran

El Mountazah Annaba
Annaba


NOMHOTEL
--------------------------------------------------
VILLE
--------------------------------------------------
Hôtel Albert 1er
Alger

Hôtel Novotel
Constantine

Hôtel Sofitel
Alger


NOMHOTEL
--------------------------------------------------
VILLE
--------------------------------------------------
Ibis Alger Aéroport
Alger

Le Méridien
Oran
```

FIGURE 19 – Executer la commande

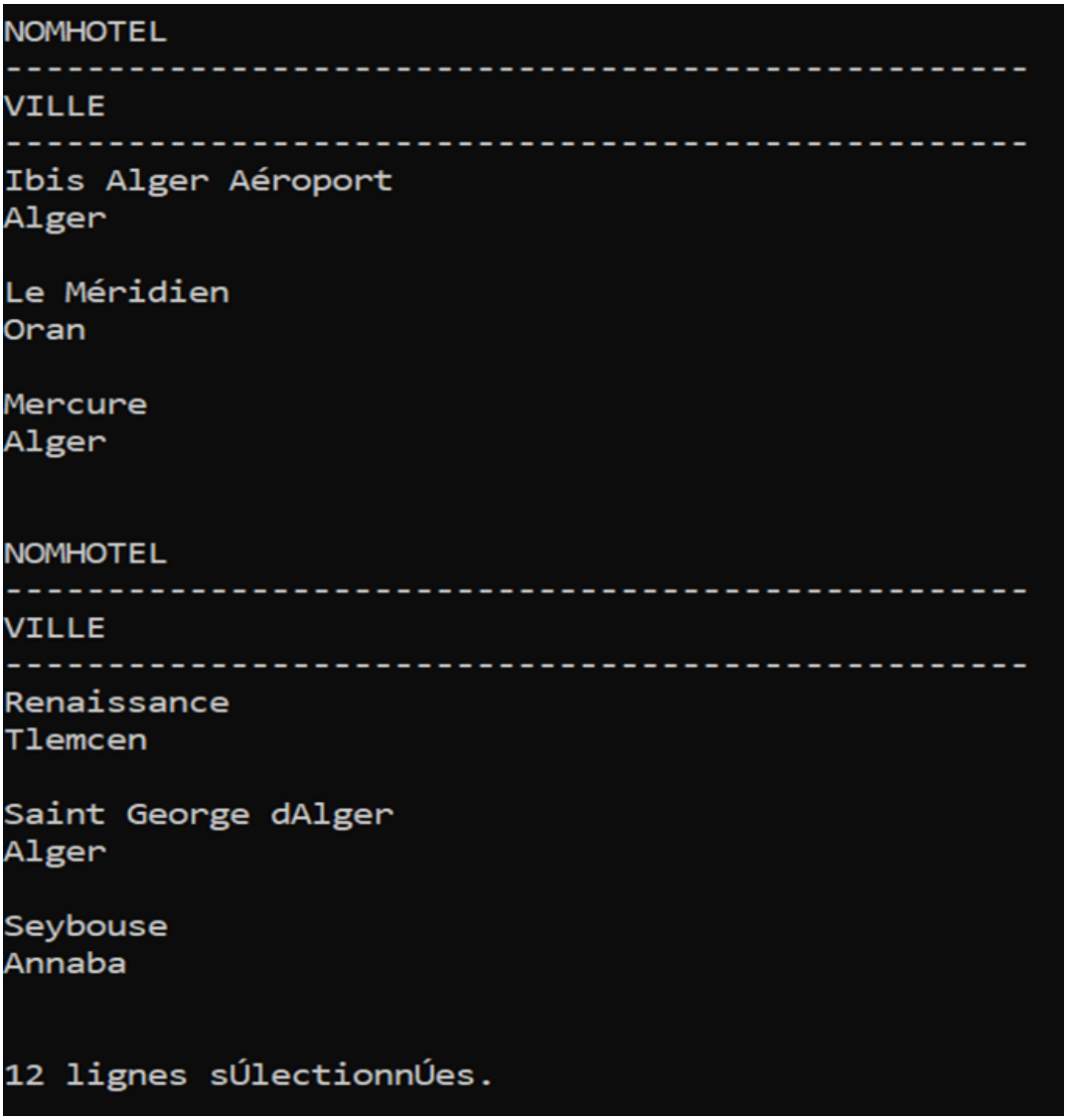

FIGURE 20 – Executer la commande

5.2 Lister les hôtels sur lesquels porte au moins une réservation

```
SQL> select distinct H.numHotel, H.nomHotel, H.ville  from Hotel H, Reservation R
  2  where deref(deref(R.reservation_chambre).chambre_hotel).numHotel = H.numHotel;

  NUMHOTEL NOMHOTEL
---------- --------------------------------------------------
VILLE
--------------------------------------------------
         2 Seybouse
Annaba

         4 Saint George dAlger
Alger

         5 Ibis Alger Aéroport
Alger


  NUMHOTEL NOMHOTEL
---------- --------------------------------------------------
VILLE
--------------------------------------------------
         6 El Mountazah Annaba
Annaba

         8 Chems
Oran

         9 Colombe
Oran


  NUMHOTEL NOMHOTEL
---------- --------------------------------------------------
VILLE
--------------------------------------------------
        10 Mercure
Alger

         10 Mercure
Alger

         11 Le Méridien
Oran

         12 Hôtel Sofitel
Alger


9 lignes sÚlectionnÚes.
```

FIGURE 21 – Executer la commande

5.3 Quels sont les clients qui ont toujours séjourné au premier étage ?

```
SQL> select C.num_client,C.nom_client,C.prenom_client from Chambre Ch, Client C, table(CH.chambre_reservation) R
  2  where deref(value(R).reservation_client).num_client = C.num_client and CH.etage = 1;

aucune ligne sÚlectionnÚe
```

FIGURE 22 – Executer la commande

5.4 Quels sont les hôtels (nom, ville) qui offrent des suites ? et donner le prix pour chaque suite

```
SQL> select distinct H.numHotel, H.nomHotel, H.ville, value(CH).num_chambre, value(CH).prix_nuit, value(CH).typeChambre from Hotel H, table(H.hotel_chambre) CH
  2  where value(CH).typeChambre = 'suite';

aucune ligne sÚlectionnÚe
```

FIGURE 23 – Executer la commande

5.5 Quel est le type de chambre le plus réservé habituellement, pour chaque hôtel d'Alger ?

```
SQL> SELECT deref(R.reservation_chambre).TYPECHAMBRE, COUNT(*) AS nb_reservation
  2  FROM Reservation R
  3  WHERE deref(deref(R.reservation_chambre).chambre_hotel).VILLE = 'Alger'
  4  GROUP BY deref(R.reservation_chambre).TYPECHAMBRE
  5  ORDER BY nb_reservation ;

DEREF(R.RESERVATION_CHAMBRE).TYPECHAMBRE           NB_RESERVATION
-------------------------------------------------- --------------
double                                                          2
triple                                                          3
simple                                                          5
```

FIGURE 24 – Executer la commande

5.6 Quels sont les hôtels (nom, ville) ayant obtenu une moyenne de notes >=6, durant l'année 2022

```
SQL> SELECT H.numHotel, H.nomHotel, H.ville, AVG(E.note) as moyennNote
  2  FROM Hotel H, Evaluation E
  3  WHERE H.numHotel = DEREF(E.evaluation_hotel).numHotel AND EXTRACT(YEAR FROM E.dateEv) = 2022
  4  GROUP BY H.numHotel, H.nomHotel, H.ville
  5  HAVING AVG(E.note) >= 6;

  NUMHOTEL NOMHOTEL
---------- --------------------------------------------------
VILLE                                              MOYENNNOTE
-------------------------------------------------- ----------
         1 Renaissance
Tlemcen                                                     9

         4 Saint George dAlger
Alger                                                    6,75

         6 El Mountazah Annaba
Annaba                                                      7


  NUMHOTEL NOMHOTEL
---------- --------------------------------------------------
VILLE                                              MOYENNNOTE
-------------------------------------------------- ----------
         7 Hôtel Albert 1er
Alger                                                       8

        12 Hôtel Sofitel
Alger                                                       9
```

FIGURE 25 – Executer la commande

5.7 Quel est l'hôtel ayant réalisé le meilleur chiffre d'affaire durant l'été 2022 (juin, juillet, aout, PS : compléter avec de nouvelles données de réservations).

```
SQL>
SQL> select deref(deref(R.RESERVATION_CHAMBRE).CHAMBRE_HOTEL).numHotel, sum(deref(R.RESERVATION_CHAMBRE).prix_nuit) as chiffre_aff
  2  from reservation R
  3  where EXTRACT(MONTH FROM DATEArr) IN (6, 7, 8)
  4  group by deref(deref(R.RESERVATION_CHAMBRE).CHAMBRE_HOTEL).numHotel
  5  ORDER BY chiffre_aff;

DEREF(DEREF(R.RESERVATION_CHAMBRE).CHAMBRE_HOTEL).NUMHOTEL CHIFFRE_AFF
---------------------------------------------------------- -----------
                                                        10        2900
                                                         5        7000
```

FIGURE 26 – Executer la commande

Partie 2 :

NoSQL – Modèle orienté documents

6 Modélisation orientée document

6.1 Proposer une modélisation orientée document

Pour proposer une modélisation orientée document de la base de données, nous allons créer deux collections : "hotel" et "client". La collection "hotel" contiendra toutes les informations relatives aux hôtels, y compris les listes d'évaluations et de réservations. La collection "client" contiendra toutes les informations relatives aux clients, telles que leur nom, leur adresse e-mail, etc.

6.2 Example

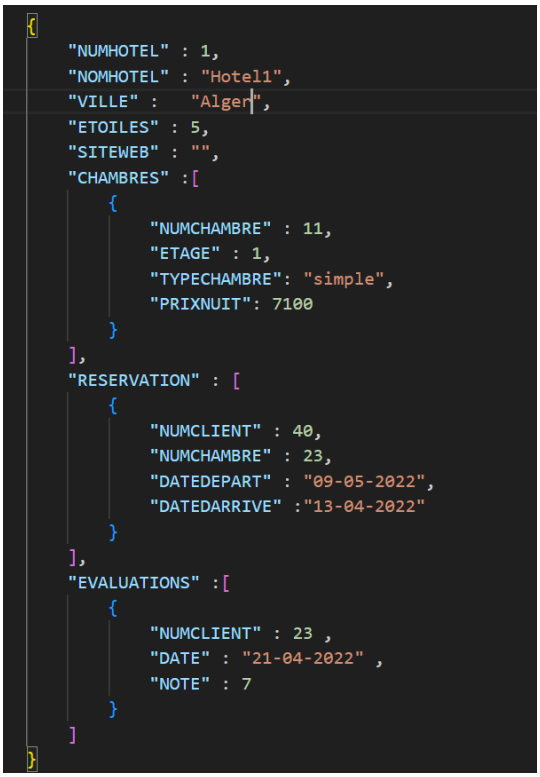

FIGURE 27 – Exemple d'un document de la collection hotels

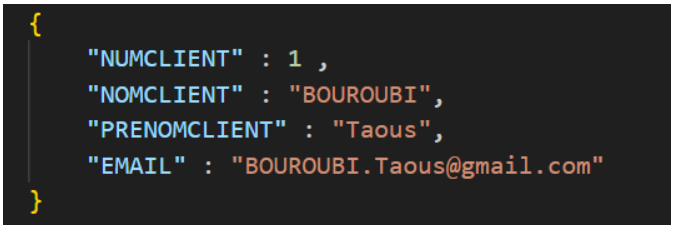

FIGURE 28 – Exemple d'un document de la collection client

6.3 Justification de notre choix

Nous avons choisi cette conception car elle permet de facilement récupérer toutes les informations relatives à un hôtel à partir d'un seul document. En outre, en ajoutant l'ID du client dans les listes d'évaluations et de réservations, nous pouvons facilement récupérer toutes les évaluations et réservations associées à un client spécifique. Cela évite le besoin de faire des jointures coûteuses entre les collections "hotels", "évaluations" et "réservations".

7 Remplir la base de données

7.1 Création de la base de données

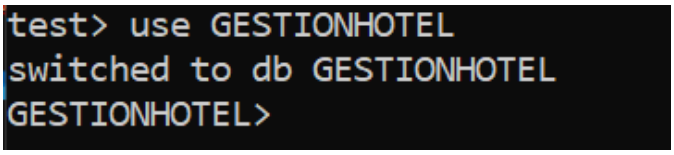

FIGURE 29 – Création de la base de données

7.2 Création des collections

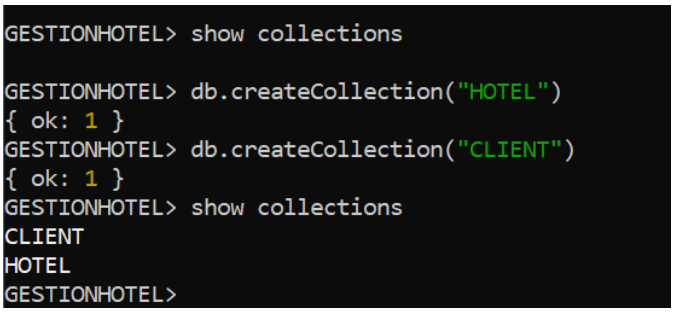

FIGURE 30 – Création des collections

7.3 Importation des données dans des collection

Nous avons créé deux fichiers JSON distincts : l'un pour les données des hôtels et l'autre pour les données des clients. Chaque fichier contient tous les documents correspondants à leurs collections respectives.
Voici un aperçu de la collection client.json qui présente un petit ensemble de données clients.

FIGURE 31 – Exemple de structure des données clients dans le fichier client.json

FIGURE 32 – Importation des données dans les collections à partir de fichiers JSON

8 Répondre aux requêtes :

8.1 Afficher tous les hôtels classés 3 étoiles

FIGURE 33 – Execution de la commande

8.2 Récupérer dans une nouvelle collection HotelsNbResv, les noms des hôtels et le nombre total de réservations par hôtel

```
GESTIONHOTEL> db.HOTEL.aggregate([
...   {
...     $project: {_id: 0,
...       hotel: "$NOMHOTEL",
...       nombreResv: { $size: "$RESERVATION" }
...     }
...   },
...   {
...     $sort: {
...       nombreResv: -1
...     }
...   },
...   {
...     $out: "HotelNbResv"
...   }
... ])
```

FIGURE 34 – Execution de la commande

```
GESTIONHOTEL> db.HotelNbResv.find()
[
  {
    _id: ObjectId("64628ec041fe5926b8751ef2"),
    hotel: 'Le Méridien',
    nombreResv: 9
  },
  {
    _id: ObjectId("64628ec041fe5926b8751ef3"),
    hotel: 'Ibis Alger Aéroport',
    nombreResv: 5
  },
  {
    _id: ObjectId("64628ec041fe5926b8751ef4"),
    hotel: 'El Mountazah Annaba',
    nombreResv: 4
  },
  {
    _id: ObjectId("64628ec041fe5926b8751ef5"),
    hotel: 'Chems',
    nombreResv: 3
  },
  {
    _id: ObjectId("64628ec041fe5926b8751ef6"),
    hotel: 'Hôtel Sofitel',
```

FIGURE 35 – Verfication du resultat

La requête utilise l'opération d'agrégation pour projeter le nom de l'hôtel et le nombre de réservations pour chaque hôtel dans la collection "HOTEL". Elle utilise l'opérateur size pour obtenir la taille du tableau des réservations de chaque hôtel. Ensuite, elle trie les documents résultants en fonction du nombre de réservations de manière décroissante. Enfin, elle écrit les résultats dans une nouvelle collection appelée "HotelNbResv" à l'aide de l'opérateur out.

8.3 Dans une collection HotelsPas-cher, récupérer les hôtels dont le prix des chambres ne dépasse pas 6000 DA

```
GESTIONHOTEL> db.HOTEL.aggregate([
...   {
...      $match : {
...        "CHAMBRES" : {
...             $not : {
...                  $elemMatch : {
...                     prix : { $gte :6000 }
...             }
...           }
...         }
...      }
...   },
...   {
...     $out: "HotelsPasCher"
...   }
... ])
```

FIGURE 36 – Execution de la commande

```
GESTIONHOTEL> db.HotelsPasCher.find()
[
  {
    _id: ObjectId("64628d4e3481340ad2ec07fd"),
    NUMHOTEL: 3,
    NOMHOTEL: 'Hôtel Novotel',
    VILLE: 'Constantine',
    ETOILES: 4,
    SITEWEB: '',
    CHAMBRES: [
      {
        NUMCHAMBRE: 101,
        ETAGE: 1,
        TYPECHAMBRE: 'simple',
        PRIXNUIT: 3200
      },
      {
        NUMCHAMBRE: 102,
        ETAGE: 1,
        TYPECHAMBRE: 'simple',
        PRIXNUIT: 3200
      },
```

FIGURE 37 – Verification du resultat

La requête ci-dessus utilise l'opération d'agrégation pour filtrer les hôtels qui n'ont aucune chambre dont le prix est supérieur ou égal à 6000. Elle utilise l'opérateur elemMatch pour rechercher les chambres qui satisfont cette condition dans le tableau des chambres de chaque hôtel. Les hôtels qui ne satisfont pas cette condition sont conservés et écrits dans une nouvelle collection appelée "HotelsPasCher" à l'aide de l'opérateur out.

8.4 Afficher tous les noms d'hôtels ayant obtenu une note moyenne >=5

```
GESTIONHOTEL> db.HOTEL.aggregate([
...   { $unwind:
...       { path: "$EVALUATIONS",
...       }
...   },
...   { $group:
...       { _id: "$NOMHOTEL",
...         moy: { $avg: "$EVALUATIONS.NOTE"}
...       }
...   },
...   { $match: { moy: {$gte: 5} }
...   },
...   { $project :
...         { _id: 0,
...           NOMHOTEL: "$_id"
...         }
...   }
...   ])
```

FIGURE 38 – Execution de la commande

```
[
  { NOMHOTEL: "Saint George d'Alger" },
  { NOMHOTEL: 'Mercure' },
  { NOMHOTEL: 'Colombe' },
  { NOMHOTEL: 'Hôtel Albert 1er' },
  { NOMHOTEL: 'Renaissance' },
  { NOMHOTEL: 'Hôtel Sofitel' },
  { NOMHOTEL: 'Ibis Alger Aéroport' },
  { NOMHOTEL: 'Le Méridien' },
  { NOMHOTEL: 'Chems' }
]
GESTIONHOTEL>
```

FIGURE 39 – Resultat de la commande

La requête ci-dessus utilise l'opération d'agrégation pour calculer la moyenne des notes d'évaluation pour chaque hôtel dans la collection "HOTEL". Elle commence par décomposer les évaluations de chaque hôtel en documents individuels. Ensuite, elle groupe les évaluations par le nom de l'hôtel et calcule la moyenne des notes. Ensuite, elle filtre les hôtels dont la moyenne des notes est supérieure ou égale à 5. Enfin, elle projette uniquement le nom de l'hôtel dans le résultat final, en excluant l'ID.

8.5 Afficher toutes les réservations d'un client donné (on donnera l'e-mail du client). On affichera le nom de l'hôtel, le numéro de chambre et la date d'arrivée

```
GESTIONHOTEL> db.HOTEL.aggregate([
...   {
...     $unwind:
...       {
...         path: "$RESERVATION",
...       }
...   },
...   {
...     $lookup:
...       {
...         from: "CLIENT",
...         localField: "RESERVATION.NUMCLIENT",
...         foreignField: "NUMCLIENT",
...         as: "CLIENTS",
...       }
...   },
...   {
...     $match:
...       {
...         "CLIENTS.EMAIL": "OUSSEDIK.Hakim@gmail.com"
...       },
...   },
...   {
...     $project:
...       {
...         _id: 0,
...         NOMHOTEL: 1,
...         "RESERVATION.NUMCHAMBRE": 1,
...         "RESERVATION.DATEDARRIVE": 1,
...       }
...   }
... ])
```

FIGURE 40 – Execution de la commande

```
[
  {
    NOMHOTEL: 'Ibis Alger Aéroport',
    RESERVATION: { NUMCHAMBRE: 2, DATEDARRIVE: '06-04-2022' }
  },
  {
    NOMHOTEL: 'Hôtel Albert 1er',
    RESERVATION: { NUMCHAMBRE: 30, DATEDARRIVE: '01-05-2022' }
  },
  {
    NOMHOTEL: 'Mercure',
    RESERVATION: { NUMCHAMBRE: 100, DATEDARRIVE: '14-06-2022' }
  }
]
GESTIONHOTEL>
```

FIGURE 41 – Resultat de la commande

La requête utilise l'opération d'agrégation pour récupérer les informations des réservations d'un client spécifique à partir de la collection "HOTEL". Elle commence par décomposer la liste des réservations de chaque hôtel en documents individuels. Ensuite, elle effectue une jointure avec la collection "CLIENT" en utilisant le champ "NUMCLIENT" pour trouver les correspondances avec le client recherché. Le filtrage est ensuite appliqué pour ne conserver que les réservations du client spécifique. Enfin, les champs pertinents (nom de l'hôtel, numéro de chambre, date d'arrivée) sont projetés dans le résultat final

8.6 Afficher toutes les évaluations postées par un client donné (on donnera l'e-mail du client). On affichera le nom de l'hôtel, la date d'évaluation, la note

Similaire aux précédentes :

```
GESTIONHOTEL> db.HOTEL.aggregate([
...   {
...     $unwind:
...       {
...         path: "$EVALUATIONS",
...       }
...   },
...   {
...     $lookup:
...       {
...         from: "CLIENT",
...         localField: "EVALUATIONS.NUMCLIENT",
...         foreignField: "NUMCLIENT",
...         as: "CLIENTS",
...       }
...   },
...   {
...     $match:
...       {
...         "CLIENTS.EMAIL":
...         "OUSSEDIK.Hakim@gmail.com",
...       }
...   },
...   {
...     $project:
...       {
...         _id: 0,
...         NOMHOTEL: 1,
...         "EVALUATIONS.DATE": 1,
...         "EVALUATIONS.NOTE": 1,
...       },
...   }
... ])
```

FIGURE 42 – Execution de la commande

```
[
  {
    NOMHOTEL: 'Hôtel Albert 1er',
    EVALUATIONS: { DATE: '01-05-2022', NOTE: 7 }
  },
  { NOMHOTEL: 'Mercure', EVALUATIONS: { DATE: '14-06-2022', NOTE: 5 } }
]
GESTIONHOTEL>
```

FIGURE 43 – Resultat de la commande

8.7 Augmenter de 2000DA, le prix unitaire de toutes les chambres des hôtels classés 5 étoiles

```
GESTIONHOTEL> db.HOTEL.updateMany(
...       { ETOILES : { $eq : 5 } },
...       { $inc: { "CHAMBRES.$[].PRIXNUIT":2000 } }
... )
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 4,
  modifiedCount: 4,
  upsertedCount: 0
}
```

FIGURE 44 – Execution de la commande

8.8 Reprendre la 2ème requête à l'aide du paradigme Map-Reduce

```
GESTIONHOTEL> var finalizeFunction = function(key, reducedValue) {
...   return { nombreResv: reducedValue };
... };

GESTIONHOTEL>

GESTIONHOTEL> db.HOTEL.mapReduce(
...   mapFunction,
...   reduceFunction,
...   {
...     out: "HOTELNBRESVMR",
...     finalize: finalizeFunction
...   }
... );
{ result: 'HOTELNBRESVMR', ok: 1 }
GESTIONHOTEL>
```

FIGURE 45 – Execution de la commande

```
GESTIONHOTEL> db.HOTELNBRESVMR.find().sort({ value: -1 });
[
  { _id: 'Le Méridien', value: { nombreResv: 9 } },
  { _id: 'Ibis Alger Aéroport', value: { nombreResv: 5 } },
  { _id: 'El Mountazah Annaba', value: { nombreResv: 4 } },
  { _id: 'Chems', value: { nombreResv: 3 } },
  { _id: 'Hôtel Sofitel', value: { nombreResv: 3 } },
  { _id: 'Colombe', value: { nombreResv: 2 } },
  { _id: "Saint George d'Alger", value: { nombreResv: 1 } },
  { _id: 'Mercure', value: { nombreResv: 1 } },
  { _id: 'Renaissance', value: { nombreResv: 1 } },
  { _id: 'Hôtel Albert 1er', value: { nombreResv: 1 } },
  { _id: 'Seybouse', value: { nombreResv: 1 } },
  { _id: 'Hôtel Novotel', value: { nombreResv: 0 } }
]
GESTIONHOTEL>
```

FIGURE 46 – Execution de la commande - suite

La requête utilise la fonction de map-reduce pour calculer le nombre total de réservations par hôtel à partir de la collection "HOTEL". La fonction de mappage émet une paire clé-valeur pour chaque document, où la clé est le nom de l'hôtel et la valeur est la longueur de la liste de réservations. La fonction de réduction agrège les valeurs associées à chaque clé en effectuant une somme. Enfin, le résultat est récupéré en triant les documents par ordre décroissant du nombre de réservations.

9 Analyse

D'après notre modélisation orientée document, il semble que pour récupérer les réservations d'un client donné par son adresse e-mail, il serait nécessaire de réaliser une jointure entre les collections des clients et des réservations. Cependant, afin de simplifier les requêtes et d'améliorer les performances, nous avons pris la décision d'ajouter l'attribut d'adresse e-mail directement dans chaque document de réservation, au sein de la liste des réservations d'un hôtel. Du même pour les évaluations.