

Overview

|  |  |
| --- | --- |
| *Project Name* | Baza de date “Hotel Management” |
| *Project Manager* | Judele Roberto Carlo |

Cuprins

|  |  |
| --- | --- |
| *Descrierea modelului real, a utilității acestuia și a regulilor de funcționare* | * Tema * Descrierea modelului real * Utilitatea * Funcționalități |
| *Prezentarea constrângerilor (restricții, reguli) impuse asupra modelului* | * Tabela Hotel * Tabela Room * Tabela Staff * Tabela RoomType * Tabela Booking * Tabela Guest |
| Descrierea entităților incluzând precizarea cheii primare | * Entitatea Hotel * Entitatea Room * Entitatea Staff * Entitatea RoomType * Entitatea Booking * Entitatea Guest |
| *Descrierea relațiilor incluzând precizarea cardinalității acestora* | Describe the approaches considered, the decisions ultimately made, and why. |
| *Descrierea atributelor incluzând tipul de date și eventualele constrângeri, valori implicite, valori posibile ale atributelor* |  |
| Realizarea diagramei entitate-relație corespunzătoare descrierii de la punctele 3-5 |  |
| Realizarea diagramei conceptuale corespunzătoare diagramei entitate-relație proiectate la punctul 6 |  |
| Enumerarea schemelor relaționale corespunzătoare diagramei conceptuale proiectate la punctul 7 |  |
| Realizarea normalizării până la forma normală 3 (FN1-FN3) |  |
| Crearea unei secvențe ce va fi utilizată în inserarea înregistrărilor în tabele (punctul 11) |  |
| *Crearea tabelelor în SQL și inserarea de date coerente în fiecare dintre acestea (minimum 5 înregistrări în fiecare tabel neasociativ; minimum 10 înregistrări în tabelele asociative; maxim 30 de înregistrări în fiecare tabel)* |  |
| *Formulați în limbaj natural și implementați 5 cereri SQL complexe ce vor utiliza, în ansamblul lor, următoarele elemente:* | a) subcereri sincronizate în care intervin cel puțin 3 tabele  b) subcereri nesincronizate în clauza FROM  c) grupări de date, funcții grup, filtrare la nivel de grupuri cu subcereri nesincronizate (în clauza de HAVING) în care intervin cel puțin 3 tabele (in cadrul aceleiași cereri)  d) ordonări și utilizarea funcțiilor NVL și DECODE (în cadrul aceleiași cereri)  e) utilizarea a cel puțin 2 funcții pe șiruri de caractere, 2 funcții pe date calendaristice, a cel puțin unei expresii CASE  f) utilizarea a cel puțin 1 bloc de cerere (clauza WITH) |
| *Implementarea a 3 operații de actualizare și de suprimare a datelor utilizând subcereri* |  |
| *Crearea unei vizualizări complexe. Dați un exemplu de operație LMD permisă pe vizualizarea respectivă și un exemplu de operație LMD nepermisă.* |  |
| *Formulați în limbaj natural și implementați în SQL: o cerere ce utilizează operația outer-join pe minimum 4 tabele, o cerere ce utilizează operația division și o cerere care implementează analiza top-n.* |  |
| *Optimizarea unei cereri, aplicând regulile de optimizare ce derivă din proprietățile operatorilor algebrei relaționale. Cererea va fi exprimată prin expresie algebrică, arbore algebric și limbaj (SQL), atât anterior cât și ulterior optimizării.* |  |
| *Realizarea normalizării BCNF, FN4, FN5* |  |
| *Aplicarea denormalizării, justificând necesitatea acesteia* |  |

1. Descrierea modelului real, a utilitatii acestuia si a regulilor de functionare

|  |  |
| --- | --- |
| Tema | * Proiectul presupune realizarea unei baze de date pentru gestionarea unui hotel. Aceasta va conține informații despre hoteluri, camere, personal, tipuri de camere, rezervări și clienți. |
| Descrierea modelului real | * Modelul include următoarele entități: Hotel, Room, Staff, RoomType, Booking, Guest. Fiecare entitate are un set specific de atribute și relații cu alte entități. |
| Utilitatea | * Baza de date este utilă pentru gestionarea eficientă a operațiunilor unui hotel, inclusiv administrarea camerelor, personalului și rezervărilor. |
| Funcționalități | * Gestionarea informațiilor despre hoteluri și camere * Administrarea rezervărilor și clienților * Evidența personalului hotelului * Gestionarea tipurilor de camere și a prețurilor acestora |

2. Prezentarea constrângerilor (restricții, reguli) impuse asupra modelului

1.Tabela Hotel:

* HotelID este cheia primară
* Nume
* Adresă
* Telefon
* Email
* Stele

Tabela Hotel:

* HotelID: Identificatorul unic al hotelului, cheia primară a tabelului.
* Nume: Numele hotelului.
* Adresă: Adresa fizică a hotelului.
* Telefon: Numărul de telefon al hotelului.
* Email: Adresa de email a hotelului.
* Stele: Numărul de stele atribuit hotelului, probabil o evaluare a calității sau a categoriei hotelului.

Tabela Customer

* CustomerID: Identificatorul unic al clientului, cheia primară a tabelului.
* FirstName: Prenumele clientului.
* LastName: Numele de familie al clientului.
* Email: Adresa de email a clientului, trebuie să fie unică.
* PhoneNumber: Numărul de telefon al clientului.
* Address: Adresa fizică a clientului.
* City: Orașul în care locuiește clientul.
* State: Statul sau provincia în care locuiește clientul.
* PostalCode: Codul poștal al adresei clientului.
* Country: Țara în care locuiește clientul.

Tabela Room

* RoomID: Identificatorul unic al camerei, cheia primară a tabelului.
* HotelID: Cheia externă care face referire la tabelul Hotel, indicând în ce hotel se află camera.
* Type: Cheia externă care face referire la tabelul RoomType, specificând tipul de cameră.
* Status: Starea camerei, cum ar fi "Disponibil", "Ocupat", "Curățenie", etc.

Tabela Staff

* StaffID: Identificatorul unic al personalului, cheia primară a tabelului.
* HotelID: Cheia externă care face referință la tabelul Hotel, indicând în ce hotel lucrează angajatul.
* FirstName: Prenumele angajatului.
* LastName: Numele angajatului.
* Position: Poziția ocupată de angajat în cadrul hotelului (ex: manager, recepționer, camerist etc.).
* Salary: Salariul angajatului.
* DateOfBirth: Data de naștere a angajatului.
* Phone: Numărul de telefon al angajatului.
* Email: Adresa de email a angajatului.
* HireDate: Data la care angajatul a fost angajat în hotel.

Tabela RoomType

* TypeID: Identificatorul unic al tipului de cameră, cheia primară a tabelului.
* Name: Numele tipului de cameră (ex: single, double, suite etc.).
* PricePerNight: Prețul pe noapte pentru acest tip de cameră.
* Capacity: Capacitatea maximă de persoane acceptate în această cameră.

Tabela Booking

* BookingID: Identificatorul unic al rezervării, cheia primară a tabelului.
* GuestID: Cheia externă care face referire la tabelul Guest, indicând care client a făcut rezervarea.
* RoomID: Cheia externă care face referire la tabelul Room, specificând camera rezervată.
* CheckinDate: Data la care clientul urmează să facă check-in.
* CheckoutDate: Data la care clientul urmează să facă check-out.
* TotalPrice: Prețul total al rezervării.

Tabela Guest

* GuestID: Identificatorul unic al clientului, cheia primară a tabelului.
* FirstName: Prenumele clientului.
* LastName: Numele clientului.
* DateOfBirth: Data de naștere a clientului.
* Address: Adresa clientului.

Tabela Review:

* **ReviewID**: Identificatorul unic al recenziei, cheia primară a tabelului.
* **HotelID**\*: Identificatorul unic al hotelului pentru care a fost făcută recenzia, cheie externă către entitatea hotel.
* **CustomerID**\*: Identificatorul unic al clientului care a scris recenzia, cheie externă către entitatea customer.
* **Rating**: Evaluarea acordată hotelului, de obicei pe o scară de la 1 la 5.
* **Comment**: Comentariul scris de client în recenzie.

3. Descrierea entitatilor incluzând precizarea cheii primare

Entitatea Hotel:

Această entitate reprezintă informațiile despre hoteluri. HotelID este un număr unic și identificatorul unic al fiecărui hotel în baza de date, fiind cheia primară. Name reprezintă numele hotelului și este un șir de caractere cu maximum 50 de caractere. Address este adresa hotelului și este un șir de caractere cu maximum 100 de caractere. Phone reprezintă numărul de telefon al hotelului și este un șir de caractere cu maximum 15 caractere. Email reprezintă adresa de email a hotelului și este un șir de caractere cu maximum 50 de caractere. Stars indică clasificarea hotelului în funcție de numărul de stele și este un număr. Cheia primară: HotelID.

Entitatea Room:

Această entitate stochează informații despre camerele din hoteluri. RoomID este un număr unic și identificatorul unic al fiecărei camere, fiind cheia primară. HotelID este un număr care face referire la hotelul căruia îi aparține camera. Type indică tipul camerei și este un număr care face referință la entitatea RoomType. Status reprezintă starea camerei (ocupată, disponibilă etc.) și este un șir de caractere cu maximum 20 de caractere. Cheia primară: RoomID.

Entitatea Staff:

Această entitate conține informații despre personalul hotelului. StaffID este un număr unic și identificatorul unic al fiecărui membru al personalului, fiind cheia primară. HotelID este un număr care face referire la hotelul la care lucrează membrul personalului. FirstName și LastName reprezintă numele și prenumele membrului personalului și sunt șiruri de caractere cu maximum 50 de caractere fiecare. Position indică poziția ocupată de membrul personalului și este un șir de caractere cu maximum 50 de caractere. Salary reprezintă salariul membrului personalului și este un număr. DateOfBirth, Phone, Email și HireDate sunt informații suplimentare despre membrul personalului. Cheia primară: StaffID.

Entitatea RoomType:

Această entitate stochează tipurile diferite de camere disponibile în hoteluri. TypeID este un număr unic și identificatorul unic al fiecărui tip de cameră, fiind cheia primară. Name reprezintă numele tipului de cameră și este un șir de caractere cu maximum 50 de caractere. PricePerNight indică prețul pe noapte pentru acest tip de cameră și este un număr. Capacity reprezintă capacitatea maximă a camerei și este un număr. Cheia primară: TypeID.

Entitatea Booking:

Această entitate conține informații despre rezervările făcute de către clienți. BookingID este un număr unic și identificatorul unic al fiecărei rezervări, fiind cheia primară. GuestID este un număr care face referire la clientul care a făcut rezervarea. RoomID este un număr care face referire la camera care a fost rezervată. CheckinDate și CheckoutDate indică datele de check-in și check-out ale rezervării. TotalPrice reprezintă prețul total al rezervării și este un număr. Cheia primară: BookingID.

Entitatea Guest:

Această entitate conține informații despre clienții care fac rezervări. GuestID este un număr unic și identificatorul unic al fiecărui client, fiind cheia primară. FirstName și LastName reprezintă numele și prenumele clientului și sunt șiruri de caractere cu maximum 50 de caractere fiecare. DateOfBirth indică data nașterii clientului. Address reprezintă adresa clientului și este un șir de caractere cu maximum 100 de caractere. Cheia primară: GuestID.

Entitatea Review:

Entitatea "Review" are un identificator unic pentru fiecare recenzie (ReviewID), iar legăturile către entitățile "Hotel" și "Customer" sunt realizate prin chei externe (HotelID și CustomerID). De asemenea, păstrează informații despre evaluarea (Rating) și comentariul (Comment) asociat fiecărei recenzii.

Descrierea relatiilor incluzând precizarea cardinalitatii acestora:

Hotel - Room:

* Cardinalitate: One-to-Many
* Fiecare hotel poate avea multiple camere, dar fiecare cameră aparține unui singur hotel.

Hotel - Staff:

* Cardinalitate: One-to-Many
* Fiecare hotel poate avea multiple angajați, dar fiecare angajat aparține unui singur hotel.

RoomType - Room:

* Cardinalitate: One-to-Many
* Fiecare tip de cameră poate fi asociat cu multiple camere, dar fiecare cameră are un singur tip.

"Review" - "Guest"

* Cardinalitate: Many-to-One
* Descriere: Fiecare recenzie este scrisă de un singur oaspete, dar un oaspete poate scrie multiple recenzii.

Booking - Guest:

* Cardinalitate: Many-to-One
* Fiecare rezervare este asociată cu un singur client, dar un client poate avea multiple rezervări.

Review - Hotel

* Cardinalitate: Many-to-One
* Descriere: Fiecare recenzie este asociată cu un singur hotel, dar un hotel poate avea multiple recenzii.

Booking - Room:

* Cardinalitate: Many-to-One
* Fiecare rezervare este asociată cu o singură cameră, dar o cameră poate avea multiple rezervări.

Descrierea atributelor incluzând tipul de date și eventualele constrângeri, valori implicite, valori posibile ale atributelor

Tabela Booking

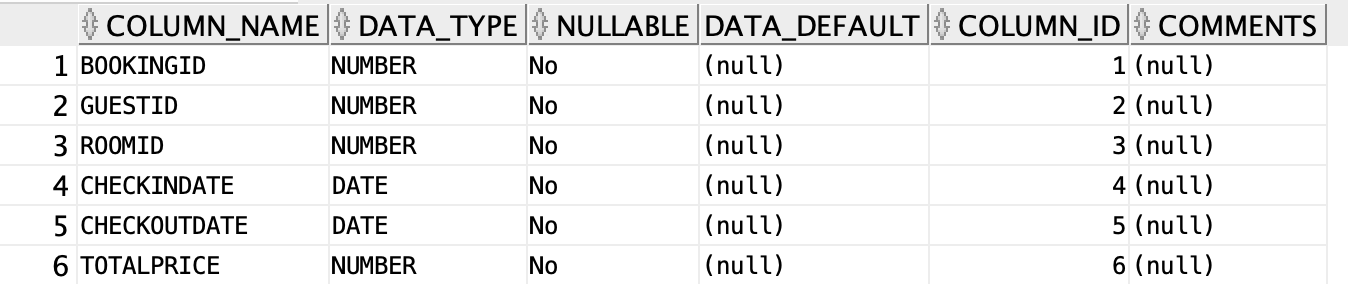


Tabela Guest

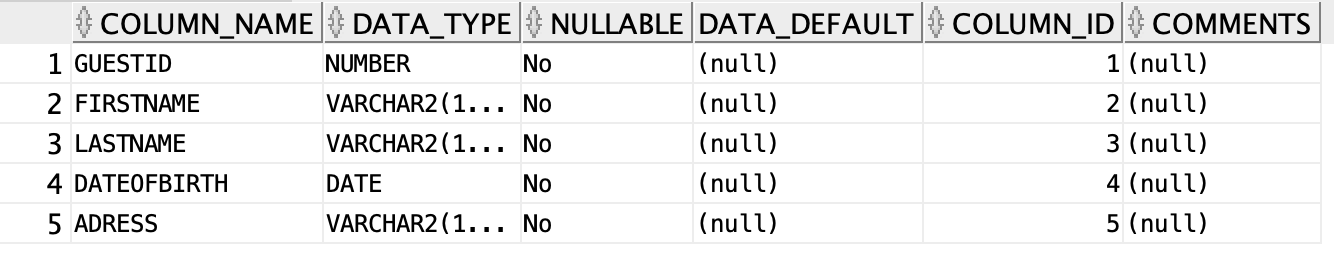


Tabela Hotel

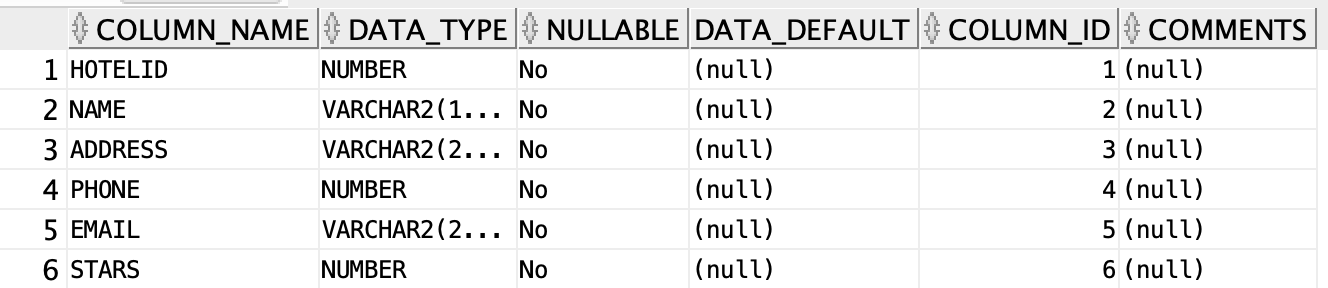


Tabela Room

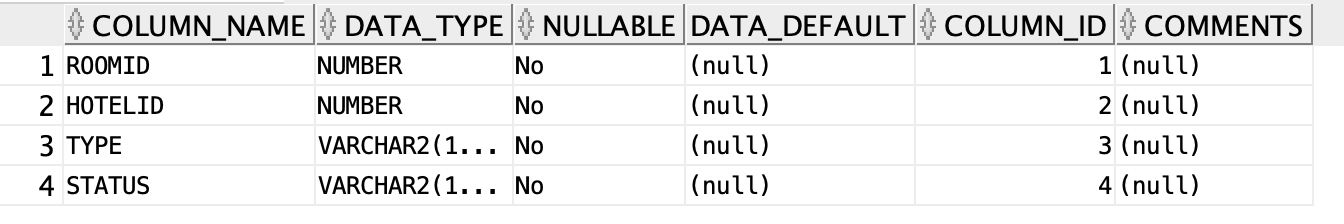


Tabela RoomType

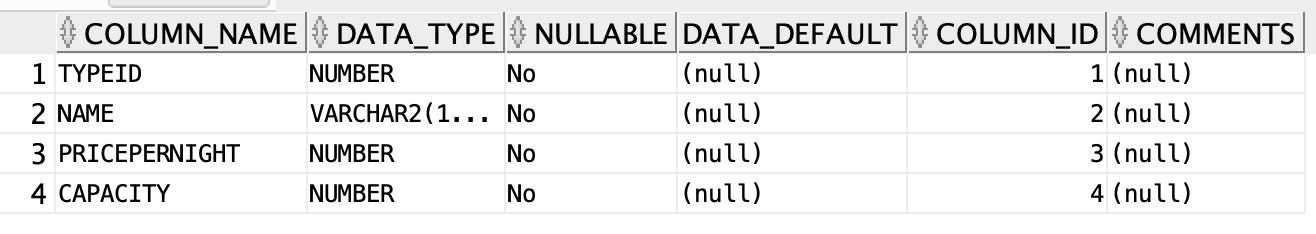


Tabela Staff

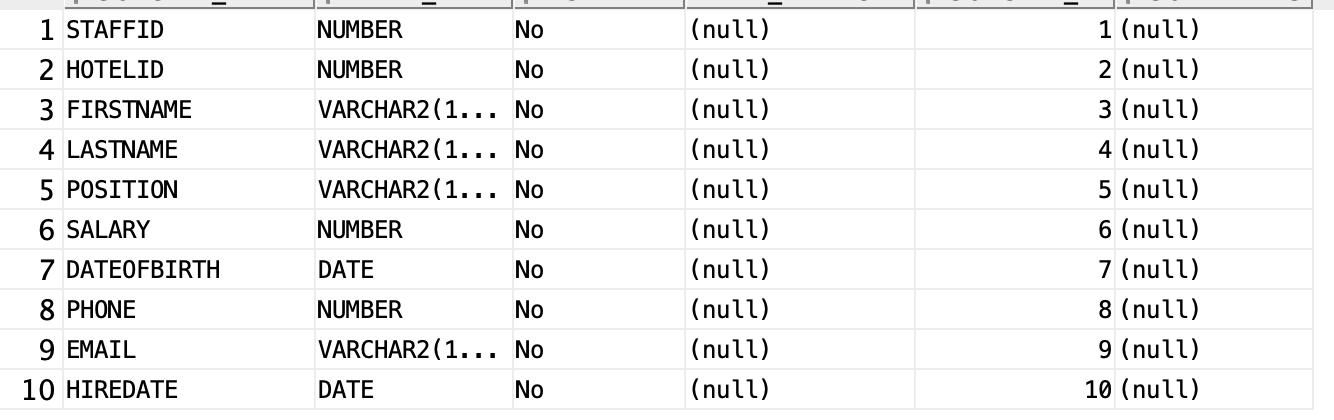
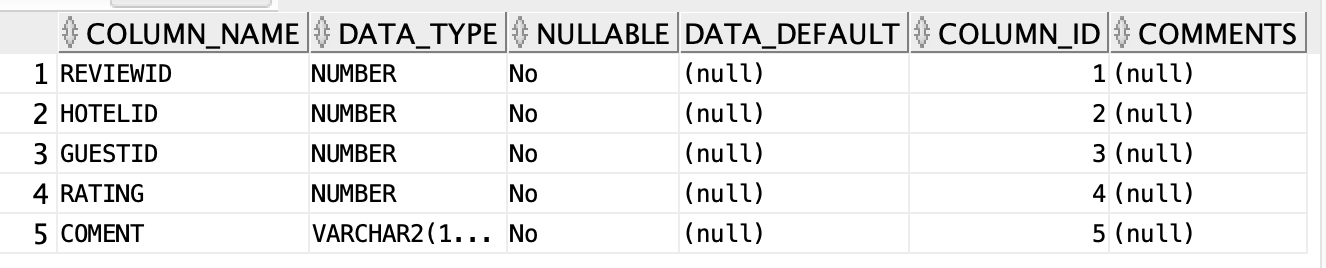
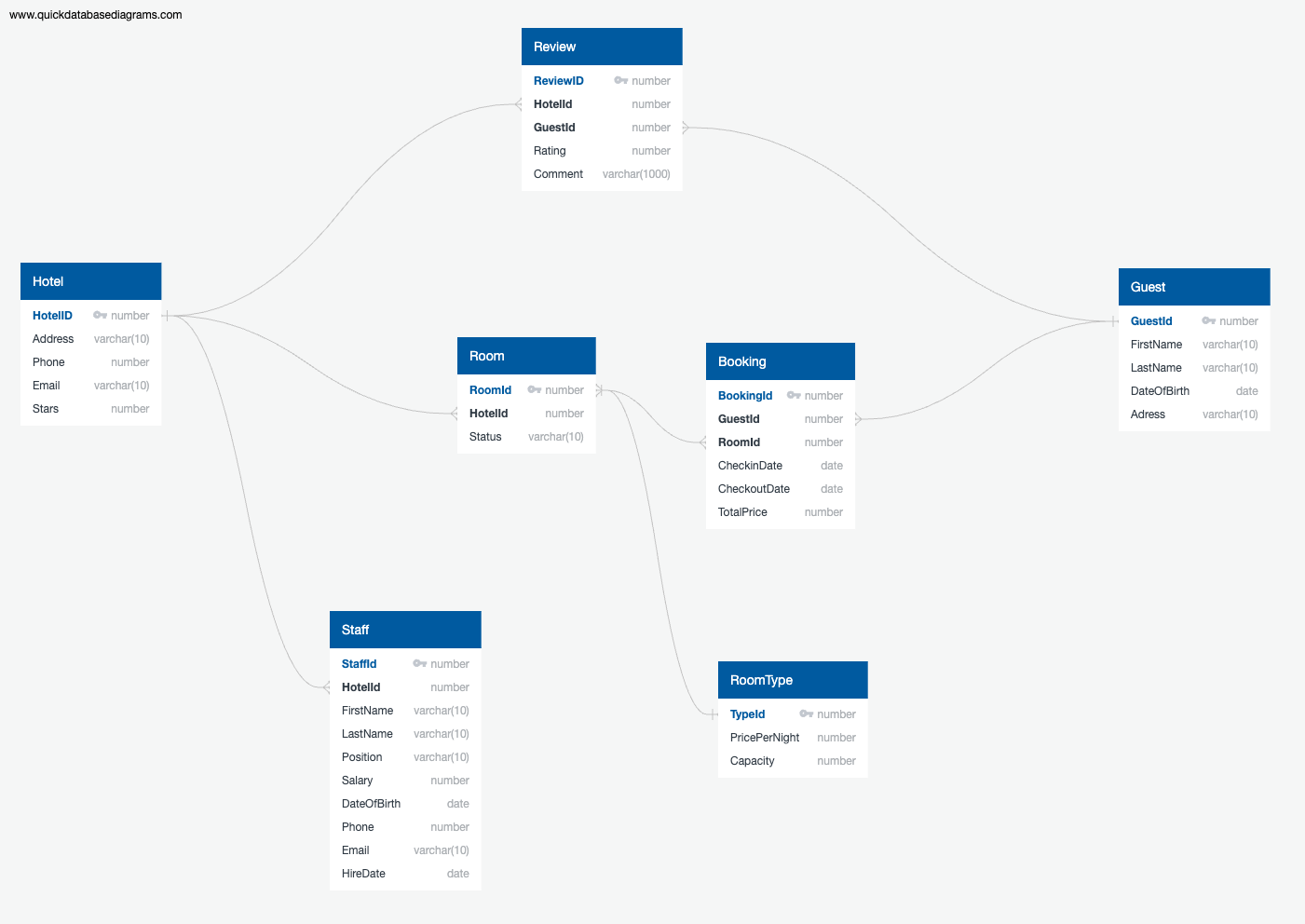


Tabela Review



6. Realizarea diagramei entitate-relatie corespunzatoare descrierii de la punctele 3-5

7. Realizarea diagramei conceptuale corespunzatoare diagramei entitate-relatie proiectate la punctul 6



8. Enumerarea schemelor relationale corespunzătoare diagramei conceptuale proiectate la punctul 7

Hotel (HotelID, Name, Address, Phone, Email, Stars)

Room (RoomID, HotelID, Type, Status)

Staff (StaffID, HotelID, FirstName, LastName, Position, Salary, DateOfBirth, Phone, Email, HireDate)

RoomType (TypeID, Name, PricePerNight, Capacity)

Booking (BookingID, GuestID, RoomID, CheckinDate, CheckoutDate, TotalPrice)

Guest (GuestID, FirstName, LastName, DateOfBirth, Address)

10.Crearea unei secvențe ce va fi utilizată în inserarea înregistrărilor în tabele (punctul 11)

-- Secventă pentru tabela Hotel

CREATE SEQUENCE hotel\_seq

START WITH 1

INCREMENT BY 1

NOCACHE;

-- Secventă pentru tabela Room

CREATE SEQUENCE room\_seq

START WITH 1

INCREMENT BY 1

NOCACHE;

-- Secventă pentru tabela Staff

CREATE SEQUENCE staff\_seq

START WITH 1

INCREMENT BY 1

NOCACHE;

-- Secventă pentru tabela RoomType

CREATE SEQUENCE roomtype\_seq

START WITH 1

INCREMENT BY 1

NOCACHE;

-- Secventă pentru tabela Booking

CREATE SEQUENCE booking\_seq

START WITH 1

INCREMENT BY 1

NOCACHE;

-- Secventă pentru tabela Guest

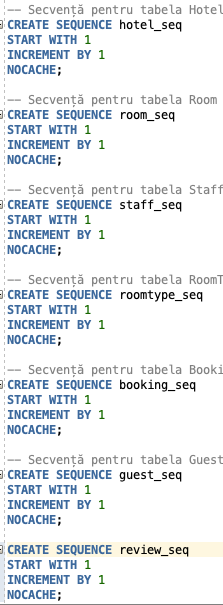
CREATE SEQUENCE guest\_seq

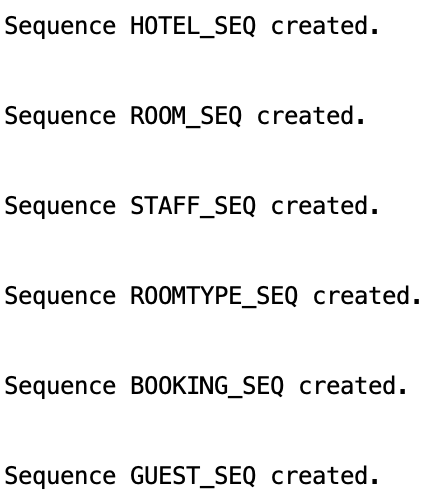
START WITH 1

INCREMENT BY 1

NOCACHE;

Print screen sursa:





b6c45e8beff30523b1b71d26db32911b.png

9. Realizarea normalizării până la forma normală 3(FN1-FN3).

Exemplu de NON-FN1

Tabelul NON-FN1: Room

Considerăm că în tabelul Room, coloana Type conține o listă de tipuri de camere pentru fiecare înregistrare, ceea ce nu respectă FN1:

**NON-FN1:**

CREATE TABLE Room (

RoomId NUMBER PRIMARY KEY,

HotelId NUMBER NOT NULL,

Type VARCHAR2(100), -- Conține o listă de tipuri

Status VARCHAR2(20) NOT NULL

);

INSERT INTO Room (RoomId, HotelId, Type, Status)

VALUES (1, 1, 'Single, Double', 'Available');

Transformarea în FN1

Pentru a normaliza în FN1, trebuie să descompunem coloana Type astfel încât fiecare înregistrare să conțină doar un singur tip:

**FN1:**

CREATE TABLE Room (

RoomId NUMBER PRIMARY KEY,

HotelId NUMBER NOT NULL,

Type VARCHAR2(20) NOT NULL,

Status VARCHAR2(20) NOT NULL,

CONSTRAINT fk\_Room\_HotelId FOREIGN KEY (HotelId) REFERENCES Hotel (HotelID)

);

INSERT INTO Room (RoomId, HotelId, Type, Status)

VALUES (1, 1, 'Single', 'Available');

INSERT INTO Room (RoomId, HotelId, Type, Status)

VALUES (2, 1, 'Double', 'Available');

Exemplu de NON-FN2

Tabelul NON-FN2: Staff

Considerăm că în tabelul Staff, Position și Salary sunt dependente parțial de cheia primară StaffId, ceea ce înseamnă că nu respectă FN2:

**NON-FN2:**

CREATE TABLE Staff (

StaffId NUMBER PRIMARY KEY,

HotelId NUMBER NOT NULL,

FirstName VARCHAR2(20) NOT NULL,

LastName VARCHAR2(20) NOT NULL,

Position VARCHAR2(20) NOT NULL,

Salary NUMBER NOT NULL,

DateOfBirth DATE NOT NULL,

Phone NUMBER NOT NULL,

Email VARCHAR2(30) NOT NULL,

HireDate DATE NOT NULL

);

INSERT INTO Staff (StaffId, HotelId, FirstName, LastName, Position, Salary, DateOfBirth, Phone, Email, HireDate)

VALUES (1, 1, 'John', 'Doe', 'Receptionist', 2500, TO\_DATE('1990-05-15', 'YYYY-MM-DD'), 123456789, 'john.doe@example.com', '2020-01-10');

Transformarea în FN2

Pentru a normaliza în FN2, vom separa Position și Salary într-un tabel separat numit PositionSalary:

**FN2:**

CREATE TABLE PositionSalary (

PositionID NUMBER PRIMARY KEY,

Position VARCHAR2(20) NOT NULL,

Salary NUMBER NOT NULL

);

INSERT INTO PositionSalary (PositionID, Position, Salary)

VALUES (1, 'Receptionist', 2500);

CREATE TABLE Staff (

StaffId NUMBER PRIMARY KEY,

HotelId NUMBER NOT NULL,

FirstName VARCHAR2(20) NOT NULL,

LastName VARCHAR2(20) NOT NULL,

PositionID NUMBER NOT NULL,

DateOfBirth DATE NOT NULL,

Phone NUMBER NOT NULL,

Email VARCHAR2(30) NOT NULL,

HireDate DATE NOT NULL,

CONSTRAINT fk\_Staff\_HotelId FOREIGN KEY (HotelId) REFERENCES Hotel (HotelID),

CONSTRAINT fk\_Staff\_PositionID FOREIGN KEY (PositionID) REFERENCES PositionSalary (PositionID)

);

INSERT INTO Staff (StaffId, HotelId, FirstName, LastName, PositionID, DateOfBirth, Phone, Email, HireDate)

VALUES (1, 1, 'John', 'Doe', 1, TO\_DATE('1990-05-15', 'YYYY-MM-DD'), 123456789, 'john.doe@example.com', '2020-01-10');

Exemplu de NON-FN3

Tabelul NON-FN3: Booking

Considerăm că în tabelul Booking, TotalPrice este dependent de CheckinDate și CheckoutDate, ceea ce nu respectă FN3:

**NON-FN3:**

CREATE TABLE Booking (

BookingId NUMBER PRIMARY KEY,

GuestId NUMBER NOT NULL,

RoomId NUMBER NOT NULL,

CheckinDate DATE NOT NULL,

CheckoutDate DATE NOT NULL,

TotalPrice NUMBER NOT NULL

);

INSERT INTO Booking (BookingId, GuestId, RoomId, CheckinDate, CheckoutDate, TotalPrice)

VALUES (1, 1, 1, TO\_DATE('2024-07-10', 'YYYY-MM-DD'), TO\_DATE('2024-07-15', 'YYYY-MM-DD'), 600);

Transformarea în FN3

Pentru a normaliza în FN3, vom elimina dependența tranzitivă a TotalPrice:

**FN3:**

CREATE TABLE Booking (

BookingId NUMBER PRIMARY KEY,

GuestId NUMBER NOT NULL,

RoomId NUMBER NOT NULL,

CheckinDate DATE NOT NULL,

CheckoutDate DATE NOT NULL,

CONSTRAINT fk\_Booking\_GuestId FOREIGN KEY (GuestId) REFERENCES Guest (GuestId),

CONSTRAINT fk\_Booking\_RoomId FOREIGN KEY (RoomId) REFERENCES Room (RoomId)

);

CREATE TABLE BookingPrice (

BookingId NUMBER PRIMARY KEY,

TotalPrice NUMBER NOT NULL,

CONSTRAINT fk\_BookingPrice\_BookingId FOREIGN KEY (BookingId) REFERENCES Booking (BookingId)

);

INSERT INTO Booking (BookingId, GuestId, RoomId, CheckinDate, CheckoutDate)

VALUES (1, 1, 1, TO\_DATE('2024-07-10', 'YYYY-MM-DD'), TO\_DATE('2024-07-15', 'YYYY-MM-DD'));

INSERT INTO BookingPrice (BookingId, TotalPrice)

VALUES (1, 600);

11. Crearea tabelelor în SQL si inserarea de date coerente în fiecare dintre acestea

CREATE TABLE Hotel (

HotelID number NOT NULL,

Name varchar(10) NOT NULL,

Address varchar(10) NOT NULL,

Phone number NOT NULL,

Email varchar(10) NOT NULL,

Stars number NOT NULL,

CONSTRAINT pk\_Hotel PRIMARY KEY (

HotelID

)

);

INSERT INTO Hotel (HotelID, Name, Address, Phone, Email, Stars)

VALUES (1, 'Hotel Example', 'Str. Exemplu nr. 1', 123456789, 'hotel@example.com', 5);

INSERT INTO Hotel (HotelID, Name, Address, Phone, Email, Stars)

VALUES (2, 'Hotel Sample', 'Str. Sample nr. 2', 987654321, 'samplehotel@example.com', 4);

INSERT INTO Hotel (HotelID, Name, Address, Phone, Email, Stars)

VALUES (3, 'Luxury Resort', 'Str. Lux nr. 123', 555666777, 'luxuryresort@example.com', 5);

INSERT INTO Hotel (HotelID, Name, Address, Phone, Email, Stars)

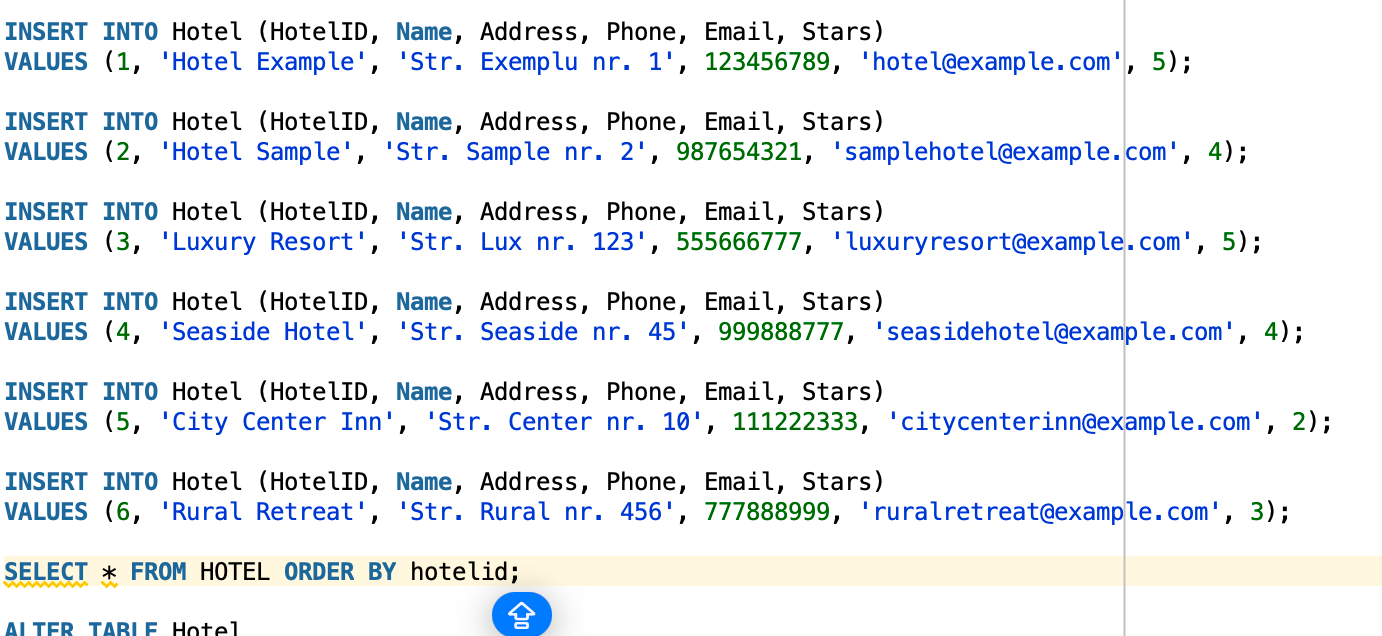
VALUES (4, 'Seaside Hotel', 'Str. Seaside nr. 45', 999888777, 'seasidehotel@example.com', 4);

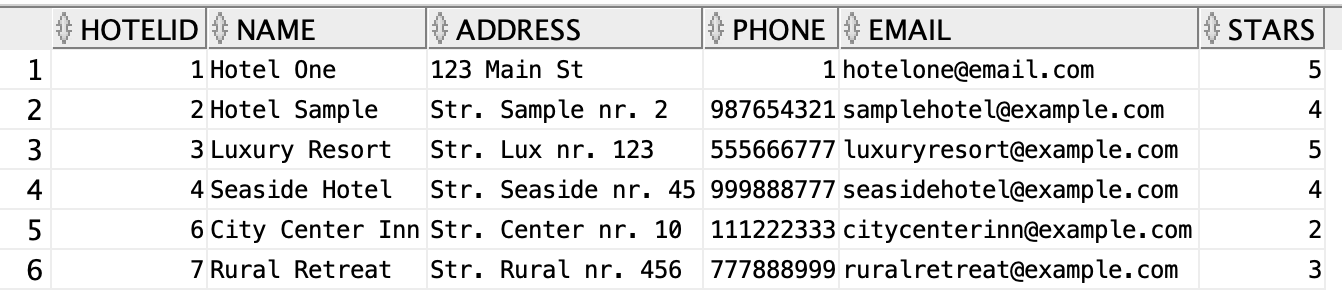
INSERT INTO Hotel (HotelID, Name, Address, Phone, Email, Stars)

VALUES (5, 'City Center Inn', 'Str. Center nr. 10', 111222333, 'citycenterinn@example.com', 2);

INSERT INTO Hotel (HotelID, Name, Address, Phone, Email, Stars)

VALUES (6, 'Rural Retreat', 'Str. Rural nr. 456', 777888999, 'ruralretreat@example.com', 3);





CREATE TABLE Room (

RoomId number NOT NULL,

HotelId number NOT NULL,

Type varchar(10) NOT NULL,

Status varchar(10) NOT NULL,

CONSTRAINT pk\_Room PRIMARY KEY (

RoomId

)

);

INSERT INTO Room (RoomId, HotelId, Type, Status)

VALUES (3, 3, 'Double', 'Available');

INSERT INTO Room (RoomId, HotelId, Type, Status)

VALUES (4, 3, 'Suite', 'Occupied');

INSERT INTO Room (RoomId, HotelId, Type, Status)

VALUES (5, 4, 'Single', 'Available');

INSERT INTO Room (RoomId, HotelId, Type, Status)

VALUES (6, 4, 'Double', 'Available');

INSERT INTO Room (RoomId, HotelId, Type, Status)

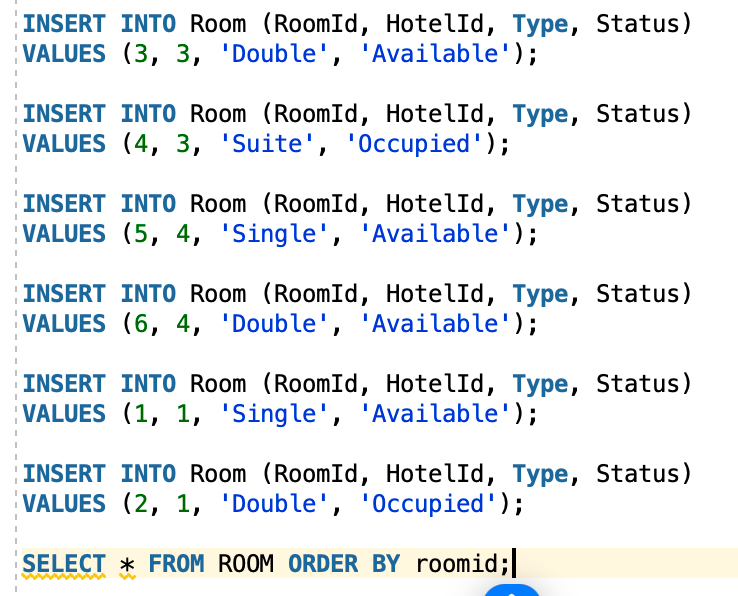
VALUES (7, 5, 'Chalet', 'Available');

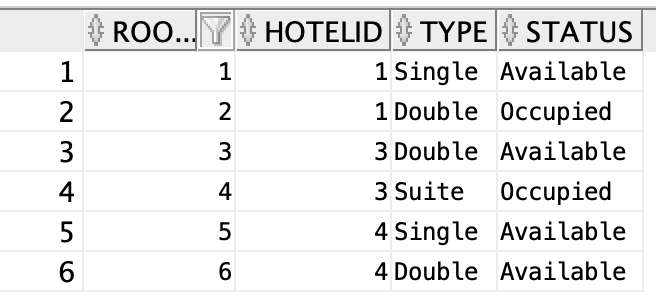
INSERT INTO Room (RoomId, HotelId, Type, Status)

VALUES (1, 1, 'Single', 'Available');

INSERT INTO Room (RoomId, HotelId, Type, Status)

VALUES (2, 1, 'Double', 'Occupied');





CREATE TABLE Staff (

StaffId number NOT NULL,

HotelId number NOT NULL,

FirstName varchar(10) NOT NULL,

LastName varchar(10) NOT NULL,

Position varchar(10) NOT NULL,

Salary number NOT NULL,

DateOfBirth date NOT NULL,

Phone number NOT NULL,

Email varchar(10) NOT NULL,

HireDate date NOT NULL,

CONSTRAINT pk\_Staff PRIMARY KEY (

StaffId

)

);

INSERT INTO Staff (StaffId, HotelId, FirstName, LastName, Position, Salary, DateOfBirth, Phone, Email, HireDate)

VALUES (1, 1, 'John', 'Doe', 'Receptionist', 2500, '1990-05-15', 123456789, 'john.doe@example.com', '2020-01-10');

INSERT INTO Staff (StaffId, HotelId, FirstName, LastName, Position, Salary, DateOfBirth, Phone, Email, HireDate)

VALUES (2, 1, 'Jane', 'Smith', 'Housekeeper', 2000, '1985-10-20', 987654321, 'jane.smith@example.com', '2019-11-05');

INSERT INTO Staff (StaffId, HotelId, FirstName, LastName, Position, Salary, DateOfBirth, Phone, Email, HireDate)

VALUES (3, 2, 'Michael', 'Johnson', 'Manager', 3500, '1978-08-25', 555123456, 'michael.johnson@example.com', '2018-06-20');

INSERT INTO Staff (StaffId, HotelId, FirstName, LastName, Position, Salary, DateOfBirth, Phone, Email, HireDate)

VALUES (4, 2, 'Emily', 'Williams', 'Concierge', 2200, '1995-03-12', 555987654, 'emily.williams@example.com', '2021-02-15');

INSERT INTO Staff (StaffId, HotelId, FirstName, LastName, Position, Salary, DateOfBirth, Phone, Email, HireDate)

VALUES (5, 3, 'Daniel', 'Brown', 'Front Desk Clerk', 2300, '1992-12-10', 111222333, 'daniel.brown@example.com', '2017-09-30');

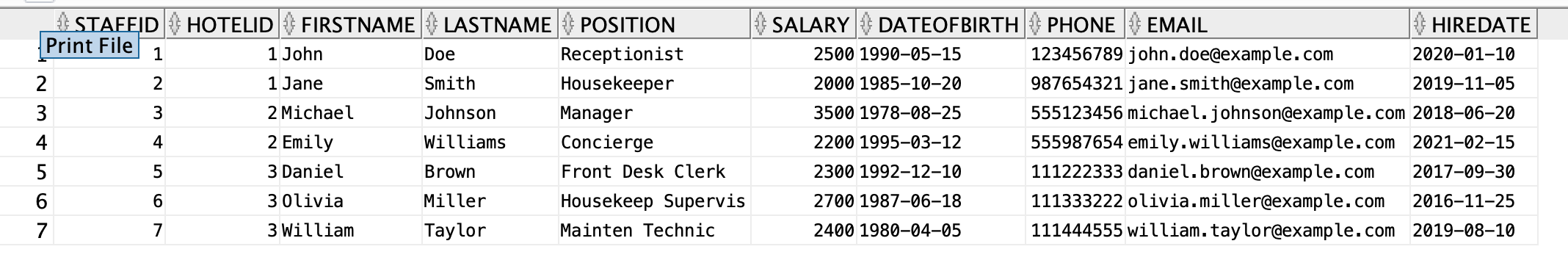
INSERT INTO Staff (StaffId, HotelId, FirstName, LastName, Position, Salary, DateOfBirth, Phone, Email, HireDate)

VALUES (6, 3, 'Olivia', 'Miller', 'Housekeeping Supervisor', 2700, '1987-06-18', 111333222, 'olivia.miller@example.com', '2016-11-25');

INSERT INTO Staff (StaffId, HotelId, FirstName, LastName, Position, Salary, DateOfBirth, Phone, Email, HireDate)

VALUES (7, 3, 'William', 'Taylor', 'Maintenance Technician', 2400, '1980-04-05', 111444555, 'william.taylor@example.com', '2019-08-10');





CREATE TABLE RoomType (

TypeId number NOT NULL,

Name varchar(10) NOT NULL,

PricePerNight number NOT NULL,

Capacity number NOT NULL,

CONSTRAINT pk\_RoomType PRIMARY KEY (

TypeId

)

);

INSERT INTO RoomType (TypeId, Name, PricePerNight, Capacity)

VALUES (1, 'Single', 100, 1);

INSERT INTO RoomType (TypeId, Name, PricePerNight, Capacity)

VALUES (2, 'Double', 150, 2);

INSERT INTO RoomType (TypeId, Name, PricePerNight, Capacity)

VALUES (3, 'Suite', 200, 4);

INSERT INTO RoomType (TypeId, Name, PricePerNight, Capacity)

VALUES (4, 'Twin', 120, 2);

INSERT INTO RoomType (TypeId, Name, PricePerNight, Capacity)

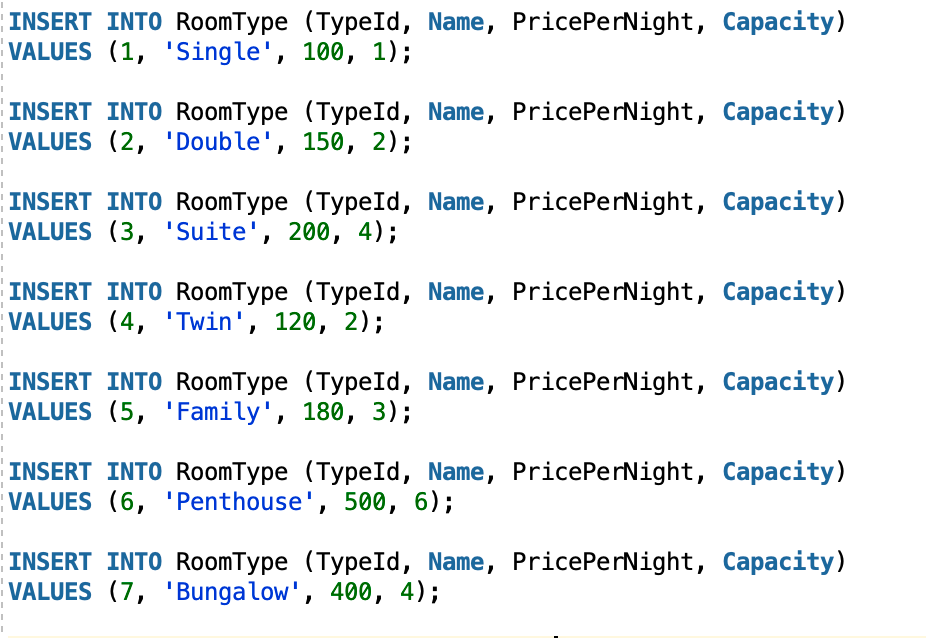
VALUES (5, 'Family', 180, 3);

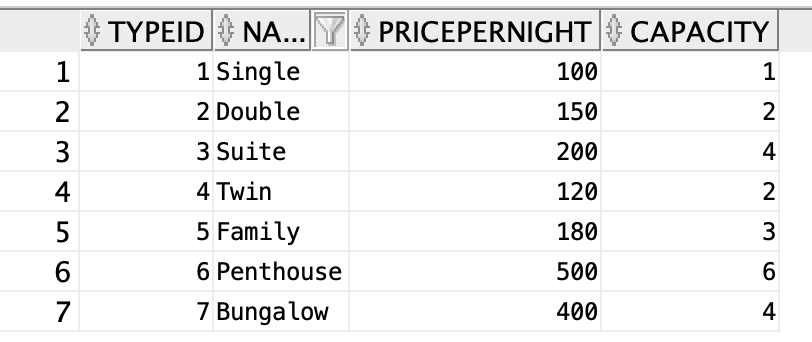
INSERT INTO RoomType (TypeId, Name, PricePerNight, Capacity)

VALUES (6, 'Penthouse', 500, 6);

INSERT INTO RoomType (TypeId, Name, PricePerNight, Capacity)

VALUES (7, 'Bungalow', 400, 4);





CREATE TABLE Booking (

BookingId number NOT NULL,

GuestId number NOT NULL,

RoomId number NOT NULL,

CheckinDate date NOT NULL,

CheckoutDate date NOT NULL,

TotalPrice number NOT NULL,

CONSTRAINT pk\_Booking PRIMARY KEY (

BookingId

)

);

INSERT INTO Booking (BookingId, GuestId, RoomId, CheckinDate, CheckoutDate, TotalPrice)

VALUES (3, 1, 1, '2024-07-10', '2024-07-15', 600);

INSERT INTO Booking (BookingId, GuestId, RoomId, CheckinDate, CheckoutDate, TotalPrice)

VALUES (4, 2, 2, '2024-08-05', '2024-08-10', 750);

INSERT INTO Booking (BookingId, GuestId, RoomId, CheckinDate, CheckoutDate, TotalPrice)

VALUES (5, 3, 3, '2024-09-01', '2024-09-05', 400);

INSERT INTO Booking (BookingId, GuestId, RoomId, CheckinDate, CheckoutDate, TotalPrice)

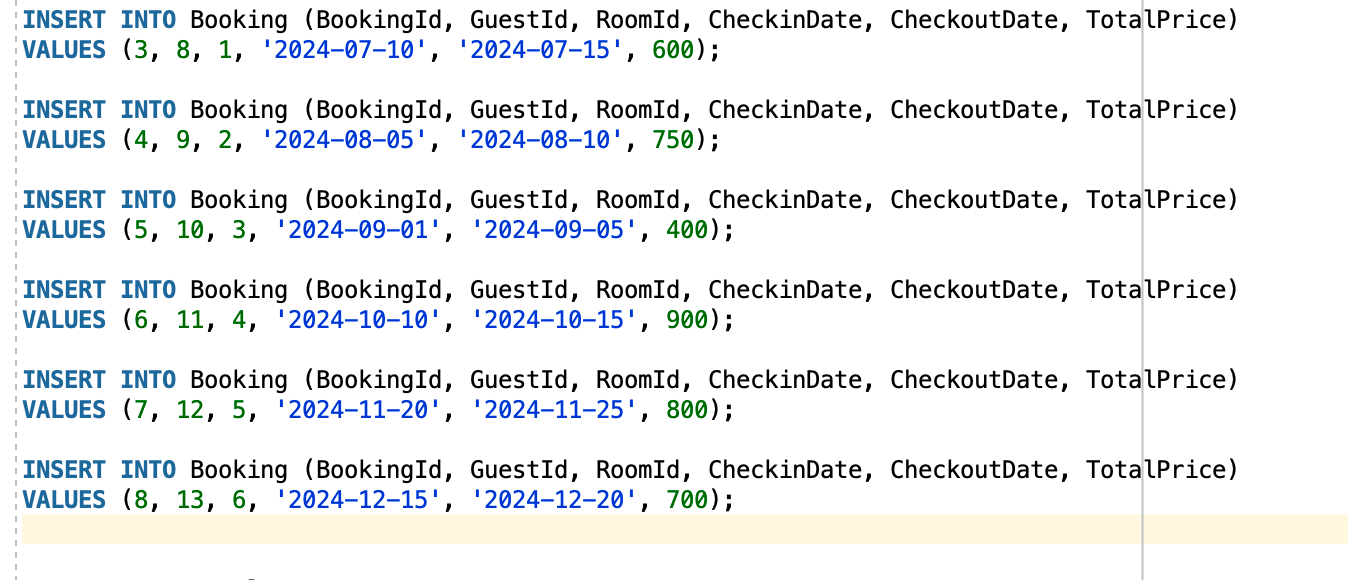
VALUES (6, 4, 4, '2024-10-10', '2024-10-15', 900);

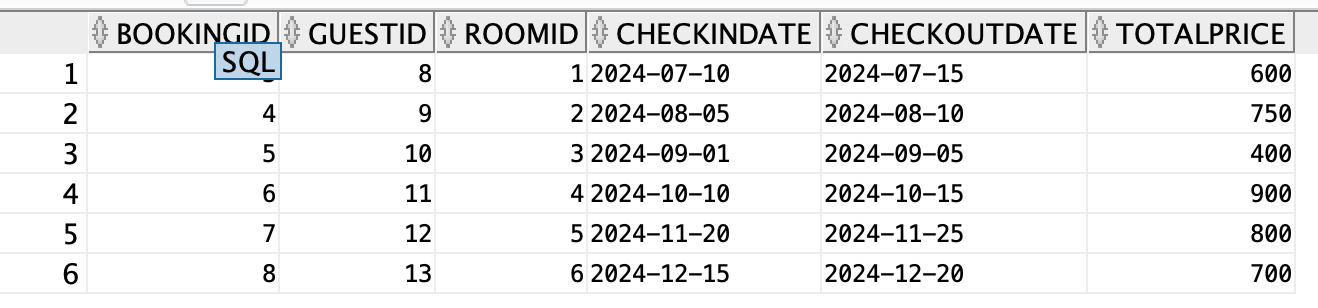
INSERT INTO Booking (BookingId, GuestId, RoomId, CheckinDate, CheckoutDate, TotalPrice)

VALUES (7, 5, 5, '2024-11-20', '2024-11-25', 800);

INSERT INTO Booking (BookingId, GuestId, RoomId, CheckinDate, CheckoutDate, TotalPrice)

VALUES (8, 6, 6, '2024-12-15', '2024-12-20', 700);





CREATE TABLE Guest (

GuestId number NOT NULL,

FirstName varchar(10) NOT NULL,

LastName varchar(10) NOT NULL,

DateOfBirth date NOT NULL,

Adress varchar(10) NOT NULL,

CONSTRAINT pk\_Guest PRIMARY KEY (

GuestId

)

);

INSERT INTO Guest (GuestId, FirstName, LastName, DateOfBirth, Address)

VALUES (8, 'Sophia', 'Wilson', '1993-03-25', 'Str. Sunshine nr. 123');

INSERT INTO Guest (GuestId, FirstName, LastName, DateOfBirth, Address)

VALUES (9, 'Ethan', 'Clark', '1980-09-15', 'Str. Sea Breeze nr. 456');

INSERT INTO Guest (GuestId, FirstName, LastName, DateOfBirth, Address)

VALUES (10, 'Ava', 'Thomas', '1975-06-30', 'Str. Mountain View nr. 789');

INSERT INTO Guest (GuestId, FirstName, LastName, DateOfBirth, Address)

VALUES (11, 'Logan', 'Lee', '1998-11-10', 'Str. River Side nr. 101');

INSERT INTO Guest (GuestId, FirstName, LastName, DateOfBirth, Address)

VALUES (12, 'Zoe', 'Garcia', '1987-04-05', 'Str. Forest Path nr. 111');

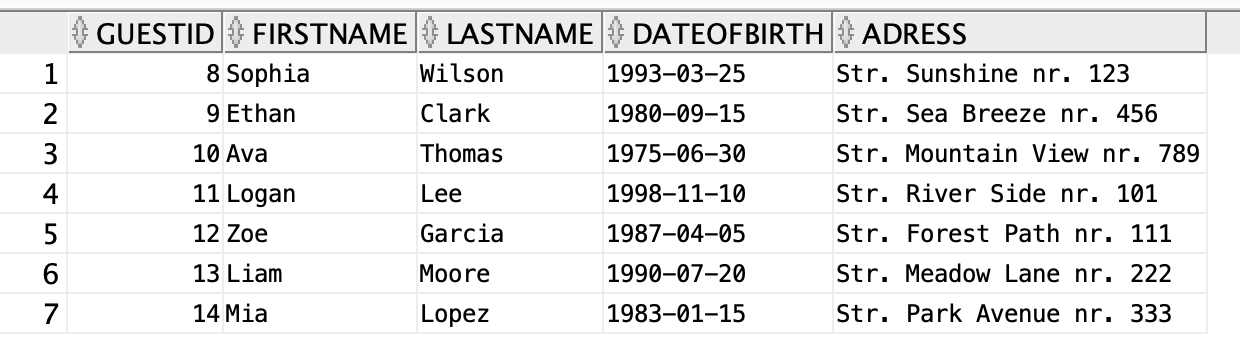
INSERT INTO Guest (GuestId, FirstName, LastName, DateOfBirth, Address)

VALUES (13, 'Liam', 'Moore', '1990-07-20', 'Str. Meadow Lane nr. 222');

INSERT INTO Guest (GuestId, FirstName, LastName, DateOfBirth, Address)

VALUES (14, 'Mia', 'Lopez', '1983-01-15', 'Str. Park Avenue nr. 333');





CREATE TABLE Review (

ReviewID number NOT NULL,

HotelId number NOT NULL,

GuestId number NOT NULL,

Rating number NOT NULL,

Coment varchar(1000) NOT NULL,

CONSTRAINT pk\_Review PRIMARY KEY (

ReviewID

)

);

INSERT INTO Review (ReviewID, HotelId, GuestId, Rating, Coment)

VALUES (8, 1, 8, 4, 'A fost o experiență plăcută. Personalul a fost foarte amabil.');

INSERT INTO Review (ReviewID, HotelId, GuestId, Rating, Coment)

VALUES (9, 2, 9, 5, 'Am fost extrem de mulțumit de servicii. Recomand cu încredere!');

INSERT INTO Review (ReviewID, HotelId, GuestId, Rating, Coment)

VALUES (10, 3, 10, 3, 'Locație frumoasă, dar camerele ar putea fi mai bine întreținute.');

INSERT INTO Review (ReviewID, HotelId, GuestId, Rating, Coment)

VALUES (11, 4, 11, 4, 'Personalul foarte prompt și atent la detalii.');

INSERT INTO Review (ReviewID, HotelId, GuestId, Rating, Coment)

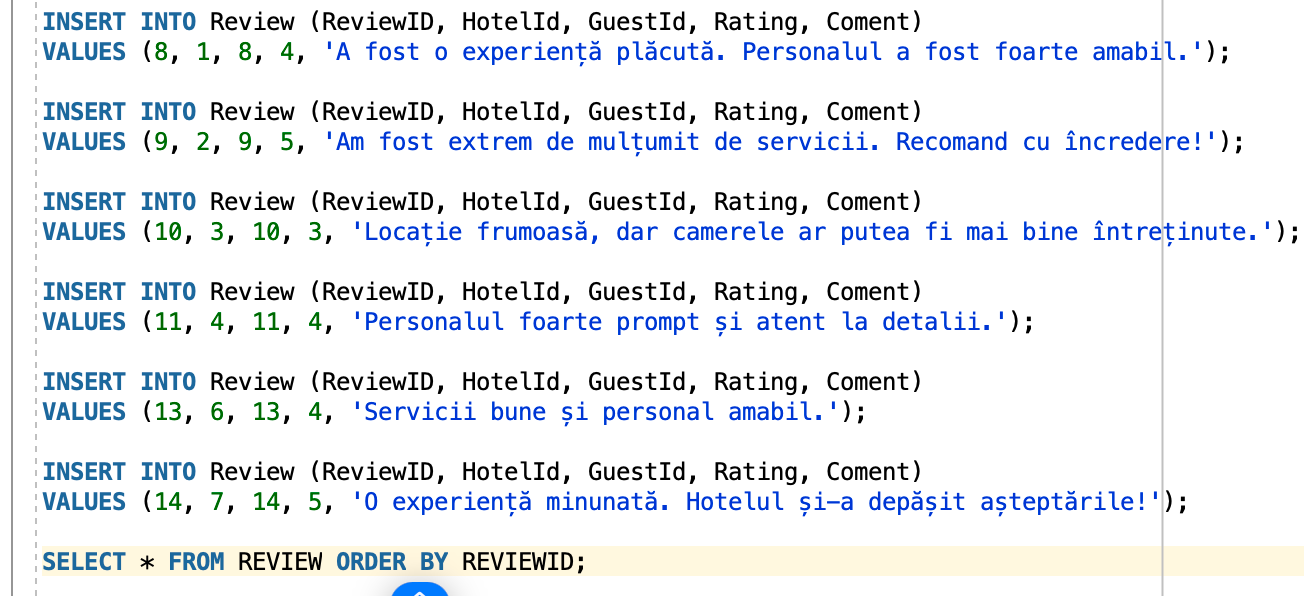
VALUES (12, 5, 12, 2, 'Camerele sunt vechi și neîngrijite.');

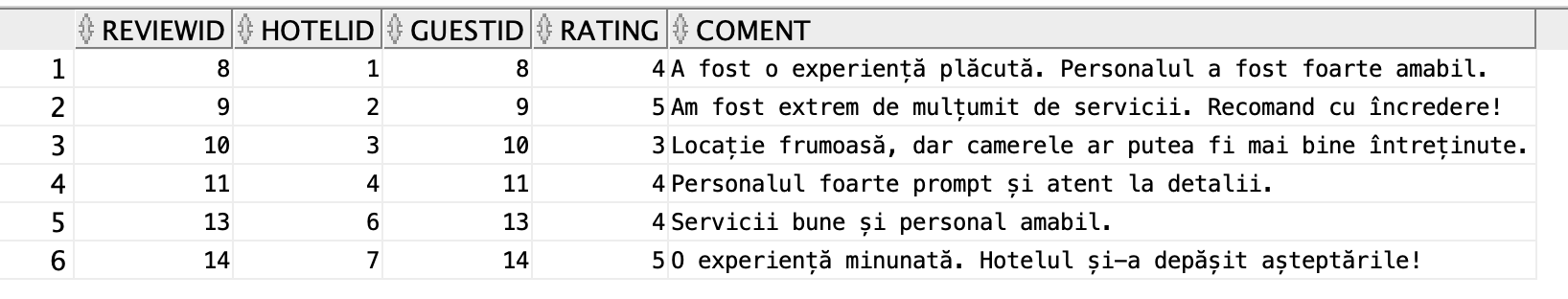
INSERT INTO Review (ReviewID, HotelId, GuestId, Rating, Coment)

VALUES (13, 6, 13, 4, 'Servicii bune și personal amabil.');

INSERT INTO Review (ReviewID, HotelId, GuestId, Rating, Coment)

VALUES (14, 7, 14, 5, 'O experiență minunată. Hotelul și-a depășit așteptările!');





12.Formulați în limbaj natural și implementați 5cereri SQL complexece vor utiliza, în ansamblul lor, următoarele elemente:

Cererea 1:

* Formularea în limbaj natural:
  + Găsește numele hotelurilor care au camere de tip 'Suite' și au primit recenzii de la oaspeți care au făcut rezervări.

SELECT DISTINCT H.Name

FROM Hotel H

JOIN (

SELECT RoomId, HotelId

FROM Room

WHERE Type = 'Suite'

) R ON H.HotelID = R.HotelId

JOIN (

SELECT ReviewID, HotelId

FROM Review

WHERE GuestId IN (

SELECT GuestId

FROM Booking

)

) Rev ON H.HotelID = Rev.HotelId

ORDER BY H.Name;



Se respecta următoarele cerințe:

* Subcereri sincronizate cu 3 tabele
* Subcereri nesincronizate

Cererea 2:

* Formularea în limbaj natural:
  + Găsește hotelurile care au mai multe de 3 recenzii și afișează numărul de recenzii și media rating-ului.

SELECT H.Name, COUNT(Rev.ReviewID) AS TotalReviews, AVG(Rev.Rating) AS AvgRating

FROM Hotel H

JOIN (

SELECT ReviewID, HotelId, Rating

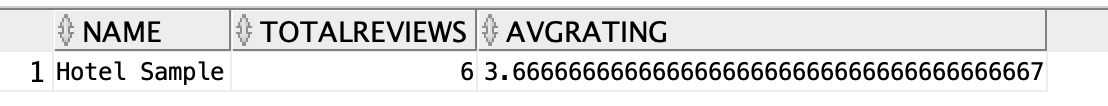
FROM Review

) Rev ON H.HotelID = Rev.HotelId

GROUP BY H.Name

HAVING COUNT(Rev.ReviewID) > 3

ORDER BY AvgRating DESC;



Se respecta următoarele cerințe:

* grupări de date cu subcereri nesincronizate in care intervin cel putin 3 tabele, funcții grup, filtrare la nivel de grupuri(in cadrul aceleiasi cereri)

Cererea 3:

* Formularea în limbaj natural:
  + Afișează numele oaspeților, data de naștere și o categorie a vârstei folosind NVL și DECODE.

SELECT G.FirstName || ' ' || G.LastName AS FullName,

G.DateOfBirth,

DECODE(

TRUNC((SYSDATE - G.DateOfBirth) / 365),

NULL, 'N/A',

0, 'Child',

1, 'Child',

2, 'Child',

3, 'Teenager',

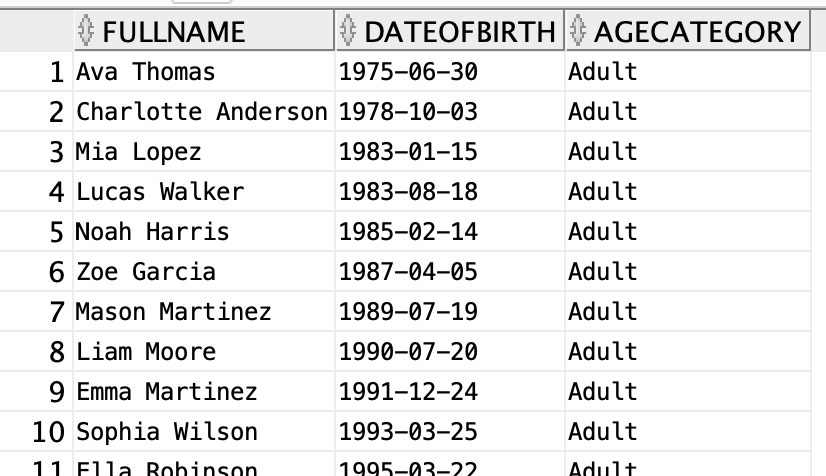
4, 'Teenager',

'Adult'

) AS AgeCategory

FROM Guest G

ORDER BY G.DateOfBirth;



Se respecta următoarele cerințe:

* ordonări si utilizarea funcțiilor NVL și DECODE (in cadrul aceleiasi cereri)

Cererea 4

* Formularea în limbaj natural:
  + Găsește numele complet al angajaților și zilele rămase până la următoarea aniversare a angajării.

SELECT S.FirstName || ' ' || S.LastName AS FullName,

ROUND(SYSDATE - S.HireDate, 0) AS DaysSinceHire,

CASE

WHEN MONTHS\_BETWEEN(ADD\_MONTHS(S.HireDate, 12 \* TRUNC(MONTHS\_BETWEEN(SYSDATE, S.HireDate) / 12) + 12), SYSDATE) <= 0 THEN

ADD\_MONTHS(S.HireDate, 12 \* TRUNC(MONTHS\_BETWEEN(SYSDATE, S.HireDate) / 12) + 24) - SYSDATE

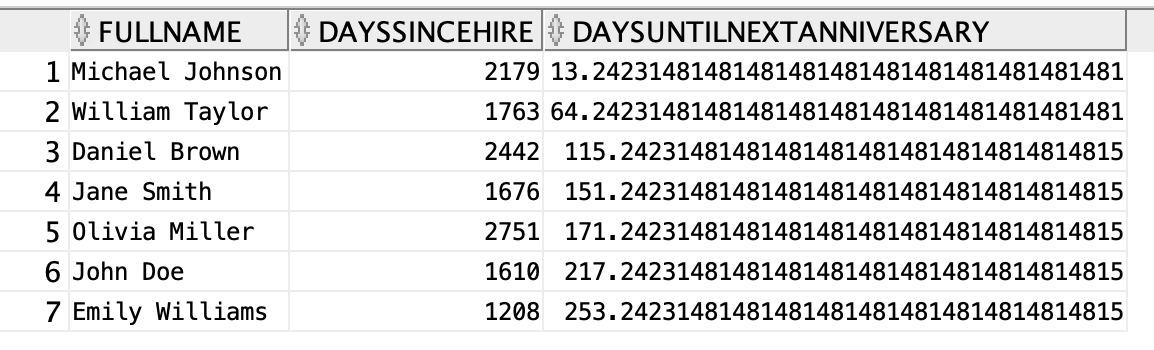
ELSE

ADD\_MONTHS(S.HireDate, 12 \* TRUNC(MONTHS\_BETWEEN(SYSDATE, S.HireDate) / 12) + 12) - SYSDATE

END AS DaysUntilNextAnniversary

FROM Staff S

ORDER BY DaysUntilNextAnniversary;



Se respecta următoarele cerințe:

* utilizarea a cel puțin 2 funcții pe șiruri de caractere, 2 funcții pe date calendaristice, a cel puțin unei expresii CASE

Cererea 5

* Formularea în limbaj natural:
  + Găsește numele hotelurilor care au cel puțin un angajat născut în luna curentă și afișează numele hotelurilor și numele angajaților.

WITH CurrentMonthBirthdays AS (

SELECT StaffId, HotelId, FirstName, LastName

FROM Staff

WHERE EXTRACT(MONTH FROM DateOfBirth) = EXTRACT(MONTH FROM SYSDATE)

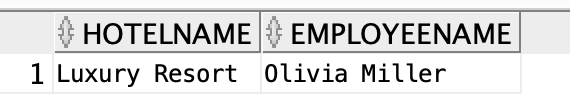
)

SELECT H.Name AS HotelName, B.FirstName || ' ' || B.LastName AS EmployeeName

FROM Hotel H

JOIN CurrentMonthBirthdays B ON H.HotelID = B.HotelId

ORDER BY H.Name, B.LastName;



Se respecta următoarele cerințe:

* Utilizarea clauzei WITH și subcereri nesincronizate

Implementarea a 3 operații de actualizare și de suprimare a datelor utilizând subcereri.

* Prima operatie:

Actualizează statusul camerelor la 'Occupied' pentru toate camerele rezervate în ultimele 30 de zile

UPDATE Room

SET Status = 'Occupied'

WHERE RoomId IN (

SELECT RoomId

FROM Booking

WHERE CheckinDate >= SYSDATE - 30

);

* A doua operație:

Șterge recenziile oaspeților care nu au făcut niciodată o rezervare

DELETE FROM Review

WHERE GuestId IN (

SELECT GuestId

FROM Guest

WHERE GuestId NOT IN (

SELECT DISTINCT GuestId

FROM Booking

)

);

* A treia operație

Actualizează prețul pe noapte pentru camerele de tip 'Suite' la 250

UPDATE RoomType

SET PricePerNight = 250

WHERE TypeId IN (

SELECT RoomId

FROM Room

WHERE Type = 'Suite'

);