***PROIECT DATAWAREHOUSE***

***SISTEM DE GESTIUNE AL REZERVĂRILOR HOTELIERE***

***MODUL ANALIZĂ***

***1. Descrierea modelului ales și a obiectivelor aplicației…………………………………………………………………………………….…………………..…3***

***2. Diagramele bazei de date OLTP***

***a. Diagrama entitate – relație a bazei de date OLTP……………………………………………………………………………………………….………...…4***

***b. Diagrama conceptuală a bazei de date OLTP……………………………………………………………………………………………………………...…5***

***3. Diagrama stea/fulg a bazei de date deposit………………………………………………………………………………………………………………….…....7***

***4. Descrierea câmpurilor necesare pentru fiecare tabel din baza de date depozit și***

***modul de populare al acestora cu informații din baza de date OLTP ………………………………………………………………………………......……8***

***5. Identificarea constrângerilor specifice depozitelor de date ce trebuie definite………………………………………………………………………….…11***

***6. Identificarea indecșilor specifici depozitelor de date ce trebuie definiți asupra modelului.***

***Formularea unei cereri în limbaj natural care va determina utilizarea indecșilor specificați***

***și va fi implementată în următoarea etapă……………………………………………………………………………………………………………………………………………………………….......12***

***7. Identificarea obiectelor de tip dimensiune ce trebuie definite asupra modelului………………………………………………………………….……...13***

***8. Identificarea tabelelor care vor fi partiționate și a tipului de partiționare.***

***Formularea unei cereri în limbaj natural care va determina utilizarea lor și va fi***

***implementată în următoarea etapă………………………………………………………………………………….……………………………………………………………………………………………14***

***9. Formularea în limbaj natural a unei cereri SQL complexe care va fi optimizată în următoarea etapă,***

***folosind tehnici specifice bazelor de date depozit.Precizarea tehnicilor de optimizare ce ar putea fi***

***utilizate pentru această cerere particular…………………………………………………………………...…………………………………………………….15***

***10. Formularea în limbaj natural a cel puțin 5 cereri cu grad de complexitate diferit, concretizate***

***în rapoarte (grafice) ce vor fi create în următoarele etape……………………………………………………………….…………………………………….15***

***MODUL IMPLEMENTARE BAZA DE DATE***

***1. Crearea bazei de date OLTP și a utilizatorilor. ………………………………………………………………………….……………………………………….16***

***2. Generarea datelor și inserarea acestora în tabele……………………………………...................…………………………………………………………23***

***3. Crearea bazei de date depozit și a utilizatorilor.......................……………………………………………………………...................……………….31***

***4. Popularea cu informații a bazei de date depozit folosind ca sursă datele din baza de date OLTP………………...................……………………38***

***5. Definirea constrângerilor…………………………………………………………………………………………………...................…………………………47***

***6. Definirea indecșilor și a cererilor SQL însoțite de planul de execuție al acestora……………………………...................…………………………..48***

***7. Definirea obiectelor de tip dimensiune, validarea acestora……………………………………………………………...................…………………….50***

***8. Definirea partițiilor; definirea cererilor SQL însoțite de planul de execuție al***

***acestora din care să reiasă ca optimizorul utilizează eficient partițiile………………………………………………………………...................…..…52***

***9. Optimizarea cererii SQL propusă în etapa de analiză***

***a. planul de execuție ales de optimizorul bazat pe cost (explicație etape parcurse)…………………………………...................………………….52***

***b. sugestii de optimizare a cererii, specificând planul de execuție obținut…………………………………………………...................…………….53***

***10. Crearea rapoartelor cu complexitate diferită (la acest nivel vor fi scripturi SQL, fără reprezentare grafică)……………...................………….55***

***ETAPA IMPLEMENTARE APLICATIE***

***1. Modul aplicație prin care se introduc și gestionează informații la nivelul bazei de date OLTP…………………………...................……………56***

***2. Posibilitatea de vizualizare a datelor introduse/actualizate la nivelul bazei de date OLTP și posibilitatea de a verifica***

***propagarea acestor operații asupra datelor din baza de date depozit………………………………………………...................…………………......66***

***3. Rapoartele grafice asociate cererilor definite în etapele anterioare (punctul 10)………………………………................…………………….....…70***

***PROIECT DATAWAREHOUSE***

~ Modul analiza ~

1. Descrierea modelului ales și a obiectivelor aplicației.

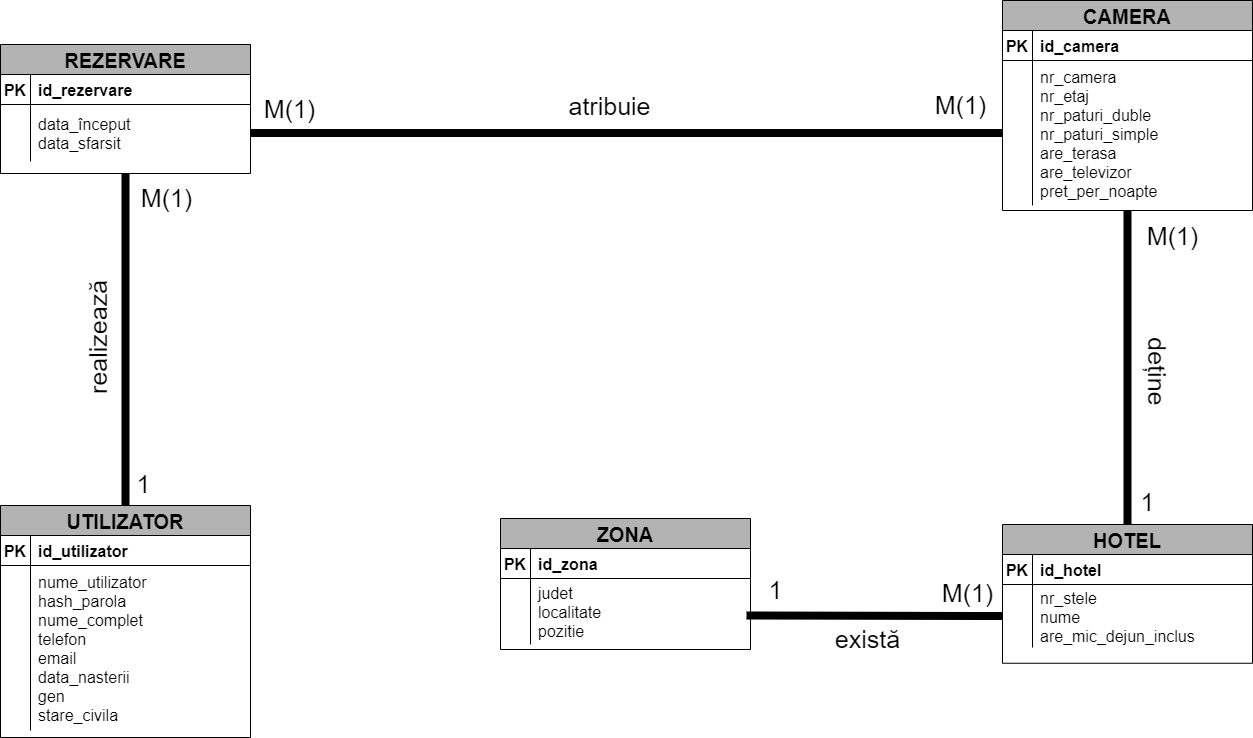
Pentru a înțelege funcționalitatea aplicației noastre, vom începe cu ***descrierea modelului***. Aplicația oferă posibilitatea utilizatorilor de a intra în aplicație prin intermediul completării unor detalii cu caracter personal cum ar numărul de telefon, adresa de email, data\_nasterii, genul, starea\_civilă, etc.

Clientul poate realiza una sau mai multe rezervări. Aceste rezervări atribuie una sau mai multe camere în funcție de perioada disponibilă. Un hotel partener al aplicației noastre deține un număr de stele și poate avea mai multe camere. O camera poate avea o serie de caracteristici importante cum ar fi etajul, numărul de paturi duble sau simple, terasă, televizor. La finalul sejurului, clientul trebuie să platescă o sumă de bani în funcție pretul camerei per noapte și de numărul de zile de cazare. Fiecare hotel se diferențiază în funcție de zona în care se află.

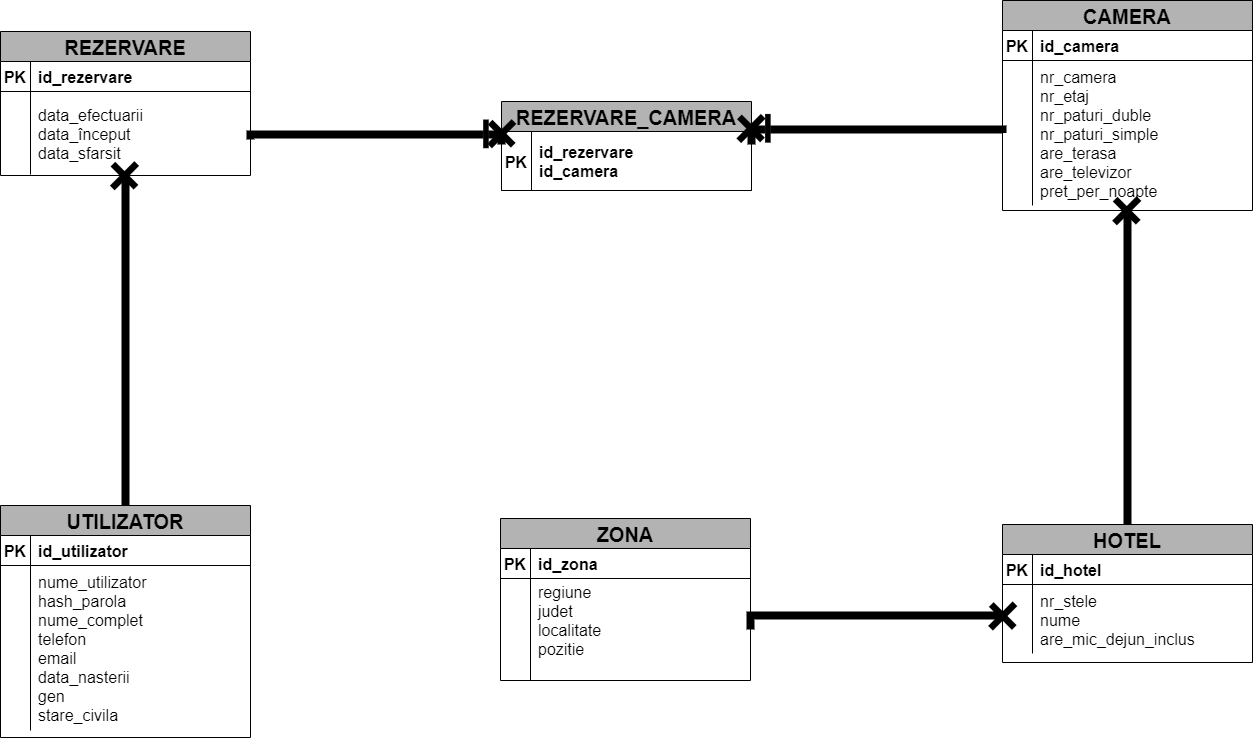
Obiectivele aplicației sunt de a oferi o interfață cât mai intuitivă clientului pentru a putea să își aleagă cu ușurință locul unde dorește să își petreacă vacanța și de a oferi managerilor o serie de rapoarte bogate în informații utile pentru a putea să își înțeleagă mai bine afacerea. Printre aceste rapoarte putem întâlni informații cu privire la perioadele de aglomerare a clienților, varsta clientilor majoritari în funcție de perioadă și informații cu privire la zonele de amplasare a viitoarelor hoteluri, etc.

**2.** Diagramele bazei de date OLTP

**a.** Diagrama entitate – relație a bazei de date OLTP.



**b.** Diagrama conceptuală a bazei de date OLTP.

****

Schemele relaționale corespunzătoare acestei diagrame conceptuale sunt următoarele:

**UTILIZATOR** ( #id\_utilizator, nume\_utilizator, hash\_parola, nume\_complet, telefon, email,

data\_nasterii, gen, stare\_civilă)

**REZERVARE** ( #id\_rezervare, id\_utilizator, data\_început, data\_sfârșit, data efectuării)

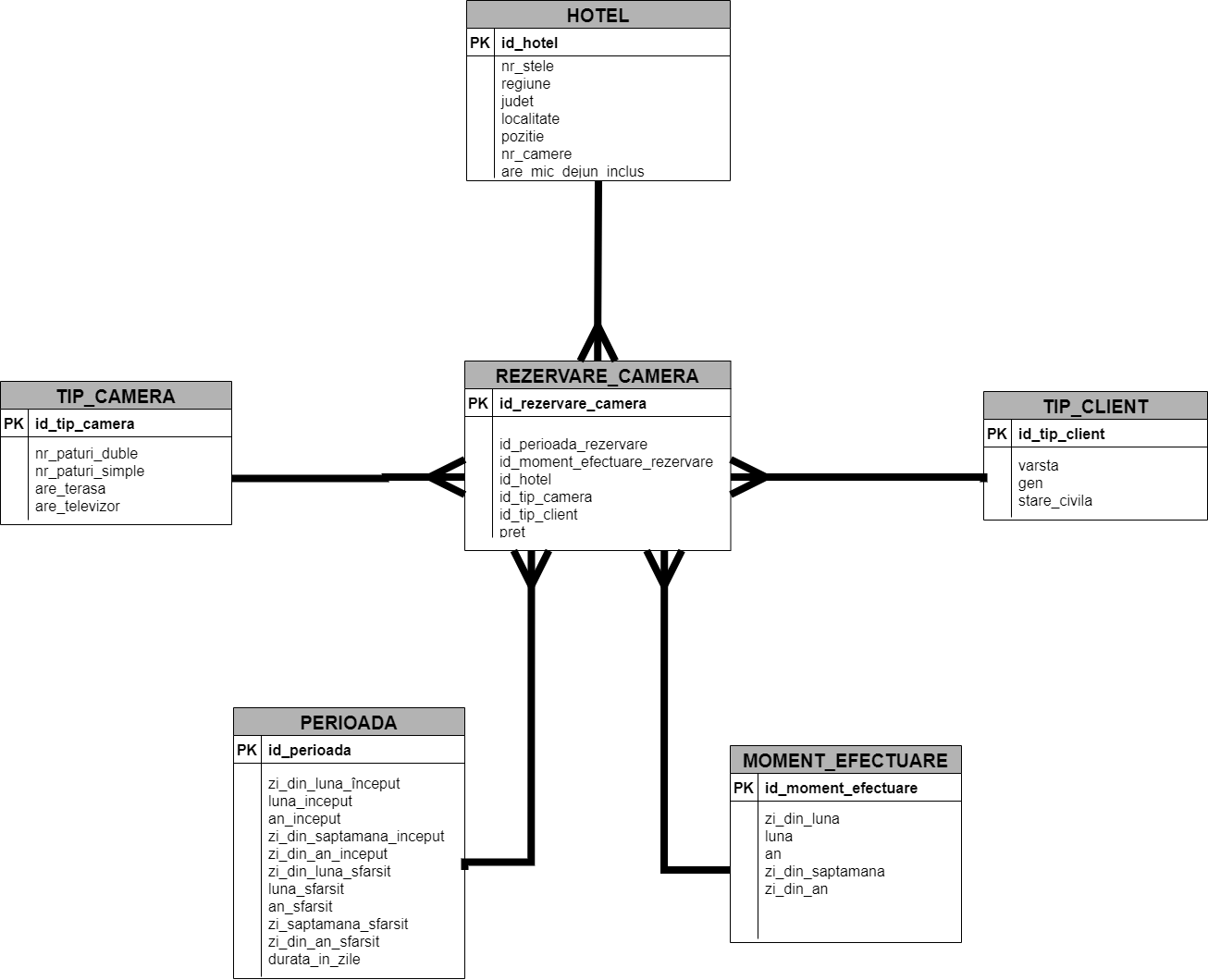
**REZERVARE\_CAMERA** ( id\_rezervare, id\_camera).

**CAMERA** ( #id\_camera, id\_hotel, nr\_camera, nr\_etaj, nr\_paturi\_duble, nr\_paturi\_simple, are\_terasa, are\_televizor, pret\_per\_noapte)

**HOTEL** ( #id\_hotel, id\_zona, nume, nr\_stele, are\_mic\_dejun)

**ZONA** ( #id\_zona, regiune,judet, localitate, pozitie).

**3.** Diagrama stea/fulg a bazei de date depozit (un tabel de fapte și cel puțin 5 tabele dimensiune).

****

**4. (1p)** Descrierea câmpurilor necesare pentru fiecare tabel din baza de date depozit și modul de populare al acestora cu informații din baza de date OLTP

**HOTEL** ( #id\_hotel, nr\_stele, judet, localitate, pozitie, nr\_camere, are\_mic\_dejun\_inclus)

id\_hotel - cheie primara autogenerata

nr\_stele - se va prelua din tabela hotel a bazei de date OLTP

judet - se va utiliza join intre tabela hotel din OLTP si tabela locatie din OLTP

localitate- se va utiliza join intre tabela hotel din OLTP si tabela locatie din OLTP

pozitie - se va utiliza join intre tabela hotel din OLTP si tabela locatie din OLTP

nr\_camere - se va prelua din tabela hotel a bazei de date OLTP

are\_mic\_dejun\_inclus - se va prelua din tabela hotel a bazei de date OLTP

**TIP\_CAMERA** ( #id\_tip\_camera, nr\_paturi\_duble, nr\_paturi\_simple, are\_terasa,are\_televizor)

id\_tip\_camera - cheie primara autogenerata

nr\_paturi\_duble - se va prelua din tabela camera a bazei de date OLTP

nr\_paturi\_simple - se va prelua din tabela camera a bazei de date OLTP

are\_terasa- se va prelua din tabela camera a bazei de date OLTP

are\_televizor- se va prelua din tabela camera a bazei de date OLTP

**PERIOADA** ( #id\_perioada, zi\_luna\_inceput, luna\_inceput, an\_inceput, zi\_din\_saptamana\_inceput, zi\_din\_an\_inceput, zi\_luna\_sfarsit, luna\_sfarsit, an\_sfarsit, zi\_din\_saptamana\_sfarsit, zi\_din\_an\_sfarsit, durata\_in\_zile)

id\_perioada - cheie primara autogenerata

luna\_inceput - se va prelua din tabela rezervare din OLTP

an\_inceput - se va prelua din tabela rezervare din OLTP

zi\_din\_saptamana\_inceput - se va prelua din tabela rezervare din OLTP

zi\_din\_an\_inceput - se va prelua din tabela rezervare din OLTP

zi\_luna\_sfarsit - se va prelua din tabela rezervare din OLTP

luna\_sfarsit - se va prelua din tabela rezervare din OLTP

an\_sfarsit - se va prelua din tabela rezervare din OLTP

zi\_din\_saptamana\_sfarsit - se va prelua din tabela rezervare din OLTP

zi\_din\_an\_sfarsit - se va prelua din tabela rezervare din OLTP

durata\_in\_zile - se va prelua din tabela rezervare din OLTP

**MOMENT\_EFECTUARE** ( #id\_moment\_efectuare, zi\_luna, luna, an, zi\_din\_saptamana, zi\_din\_an)

id\_moment\_efectuare - cheie primara autogenerata

zi\_luna-

luna-

an -

zi\_din\_saptamana-

**TIP CLIENT** ( #id\_tip\_client, varsta, gen, stare\_civila)

id\_tip\_client -

varsta - diferenta intre sysdate si data nasterii preluata din tabela client OLTP

gen - preluat din tabela client OLTP

stare\_civila - preluat din tabela client OLTP

Pentru popularea bazei de date depozit cu date din baza de date OLTP se va utiliza o procedură.

Au fost create funcții ajutătoare pentru obținerea mai multor informații, precum calculul vârstei clientului.

**5. (1p)** Identificarea constrângerilor **specifice depozitelor de date** ce trebuie definite,

justificând alegerea făcută

* Pentru toate id-urile prezente în tabele se va aplica constrângerea de cheie primară (valoare unică și nenulă pentru fiecare tuplu).
* Pentru toate datele introduse în tabela “perioada” se vor aplica restricții de tipul NOT NULL, întrucât o rezervare fără date de început și final nu poate fi considerată validă.
* Pentru toate datele introduse în tabela “hotel” se vor aplica constrângeri de tipul NOT NULL. Vor fi necesare toate datele solicitate pentru a se considera un hotel valid. Aceste date sunt utile atât pentru clienți cât și pentru analiștii de date.
* Pentru ca un client să fie considerat valid, este necesar ca toate câmpurile tabelei asociate să aibă restricția NOT NULL. Aceasta este importantă și pentru a evita rezervările false.
* Pentru câmpurile unde este necesară introducerea datelor de tip zi din lună, se poate impune restricția ca valoarea acestora să fie cuprinsă între 1 și 31.
* Pentru câmpurile unde este necesară introducerea datelor de tip zi din an, se poate impune restricția ca valoarea acestora să fie cuprinsă între 1 si 365.

**6. (0,5p)** Identificarea indecșilor **specifici depozitelor de date** ce trebuie definiți asupra modelului (**minim 2** dacă echipa este formată din 4 persoane); formularea unei cereri în limbaj natural care va determina utilizarea indecșilor specificați și va fi implementată în următoarea etapă.

* **Indexare după ID-ul rezevării**

Întrucât rezervarea este pionul central al bazei de date, este util să definim un astfel de index pentru a accesa mai ușor detaliile unei rezervări. De asemenea, vrem să vedem câte camere au fost rezervate simultan de către același client. Poate fi de ajutor pentru raportare. Câmpul id\_rezervare este preluat din tabelul “rezervare camera” din baza de date OLTP și astfel putem identifica ce camere au fost rezervate în cadrul acestui ID.

* **Idexare după luna efectuării rezervării**

Acest index poate fi utilizat pentru un rapot în care se dorește observarea lunii din an în care se efectuează rezervările. Spre exemplu, o agenție de turism ar avea nevoie de astfel de date pentru a ști când se pot lansa oferte noi.

* **Indexare după localitatea în care se află hotelul și poziția acestuia față de centrul localității**

Acest index va fi construit pentru a putea sorta și grupa rezervările după locație. Locația are rol semnificativ în rapoartele și statisticile destinațiilor de vacanță preferate de către clienți.

* **Indexare după tip camera**

Acest index este construit pentru a sorta rezervările în funcție de tipul de cameră ales. Poate fi utilizat pentru a vedea mai ușor ce tip de cameră preferă clienții.

**7. (0,5p)** Identificarea obiectelor de tip dimensiune ce trebuie definite asupra modelului

* **Obiect de tip dimensiune pentru tabelul perioada rezervare**

Această dimensiune evidențiază dependențele dintr zi zi (\_inceput/\_sfarsit) - zi\_luna (\_inceput/\_sfarsit) - zi\_an (\_inceput/\_sfarsit). Mai exact, ziua în care începe (/se sfârșește) rezervarea poate determina valorile atributelor zi din lună și zi din an.

* **Obiect de tip dimensiune pentru locație**

Am putea considera ierarhia localitate județ. Totuși, această dimensiune este problematică deoarece același nume de localitate se poate regăsi în mai multe județe. Este un caz bun pentru obiecte de tip dimensiune dar utilizat în raportări poate duce la erori grave.

* **Obiect de tip dimensiune pentru id rezevare**

Atributul id\_rezervare determină atributele id\_perioada\_rezervare, id\_moment\_efectuare\_rezervare, id\_hotel, id\_tip\_client.

**8. (1p)** Identificarea tabelelor care vor fi partiționate și a tipului de partiționare. Formularea unei cereri în limbaj natural care va determina utilizarea lor și va fi implementată în următoarea etapă.

* **Partitionare prin range pret pentru tabelul rezervare\_camera**

Se pot împărți rezervările în funcție de un interval de preț. Aceste intervale pot fi utlizate în raportările pentru bugetele clienților sau raportările pentru stabilirea ofertelor în piață.

* **Partitionare prin listă după numărul de stele pentru tabelul hotel**

Acest tip de patiționare se poate utiliza pentru situațiile în care se dorește obținerea unor informații în funcție de tipul hotelului indicat de numărul de stele. Spre exemplu, “Care este pretul mediu al unei camere cu un anumit numar de locuri la un hotel cu x nr de stele?”.

**9. (0,5p)** Formularea în limbaj natural a unei cereri SQL complexe care va fi optimizată în următoarea etapă, folosind tehnici specifice bazelor de date depozit. Precizarea tehnicilor de optimizare ce ar putea fi utilizate pentru această cerere particulară (avantaje / dezavantaje de utilizare pentru o anumită tehnică)

Construiți o cerere care să afișeze numele hotelurilor ce au valoarea rezervărilor cuprinse între două valori.

**10. (2p)** Formularea în limbaj natural a cel puțin 5 cereri cu grad de complexitate

diferit, concretizate în rapoarte (grafice) ce vor fi create în următoarele etape

* Realizați un raport care arată numărul de camere rezervate în fiecare lună.
* Raport grafic care arată câte rezervări au fost făcute în anumite perioade ale anului.
* Raport grafic ce arată cele mai scumpe 5 hoteluri din România.

***PROIECT DATAWAREHOUSE***

~ Modul implementare baza de date ~

1. Crearea bazei de date OLTP și a utilizatorilor

--Baza noastra de date va avea 3 tipuri de utilizatori in aplicatie

-- utilizatorul manager care va introduce schema si cu toate datele din aplicatie.

-- acesta va avea acces la toate datele din schema oltp prin SELECT,UPDATE,DELETE,INSERT

-- Managerul are posibilitatea de a introduce rezervari,modifica utilizatori, update pe rezervari.

-- Practic are acces sa faca tot ce vrea pe schema

-- acesta va avea acces si in olap pentru a vizualiza rapoartele scoase dar

--si de a modifica datele din tabele dupa propriul interes.

show con\_name;

alter session set container= orclpdb;

show con\_name;

ALTER PLUGGABLE DATABASE orclpdb open;

CREATE USER dw\_manager IDENTIFIED BY mng\_pass;

GRANT CREATE SESSION TO dw\_manager;

GRANT CREATE ANY TABLE TO dw\_manager;

GRANT CREATE ANY INDEX TO dw\_manager;

GRANT CREATE VIEW TO dw\_manager;

GRANT CREATE TRIGGER TO dw\_manager;

GRANT SELECT ANY TABLE TO dw\_manager;

GRANT DELETE ANY TABLE TO dw\_manager;

GRANT UPDATE ANY TABLE TO dw\_manager;

GRANT ALTER ANY TABLE TO dw\_manager;

GRANT UNLIMITED TABLESPACE TO dw\_manager;

--Pentru a vizualiza privilegiile adaugate putem folosi aceasta cerere asupra utlizatorului creat.

SELECT \*

FROM session\_privs;

-- intro schema si datele furnizate pentru schema

--SCHEMA CU TABELE TREBUIE RULATE IN dw\_manager

--script creare schema oltp.txt

--DATELE DIN TABELE TREBUIE INTRODUSE IN dw\_manager

--script inserare date oltp.txt

-- urmatorul tip va fi de tip admin care va avea posibilitatea de UPDATE peste toate tabele din schema OLTP

-- fara a avea posibilitatea de a sterge orice tip de inregistrare deoarece si inregistrarile neconforme pot reprezenta

-- un interes pentru manager.

CREATE USER dw\_admin IDENTIFIED BY admin\_pass;

GRANT CREATE SESSION TO dw\_admin;

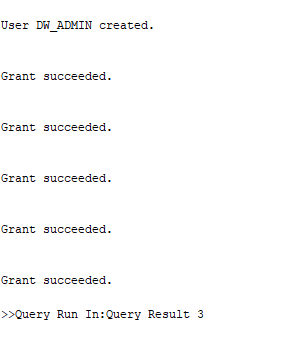
GRANT SELECT ANY TABLE TO dw\_admin;

GRANT DELETE ANY TABLE TO dw\_admin;

GRANT UPDATE ANY TABLE TO dw\_admin;

GRANT ALTER ANY TABLE TO dw\_admin;

--daca dorim sa oferim doar anumite privilegi mai restrictive asupra anumitor tabele putem folosi comanda urmatoare

--GRANT UPDATE ON dw\_manager.rezervare TO dw\_admin;

--sau daca dorim sa nu mai folosim anumite privilegii precum cel de mai putem folosi comanda

--REVOKE DELETE ON dw\_manager.rezervare FROM dw\_admin;

--pentru a accesa un tabel trebuie sa folosim dw\_manager.nume\_tabel deoarece altfel nu merge

-- iar ultimul tip de utilizator este cel cel de utilizator care are

-- posibilitatea sa vizualizeze hotelurile si sa introduce date in rezervari.

-- acesta nu avea acces la baza de date.

--ca si SYS putem rula urmatoarea cerere pentru a vizualiza care sunt care sunt privilegiile oferite

SELECT substr(grantee,1,20) grantee, owner,substr(table\_name,1,15) table\_name, grantor, privilege

FROM DBA\_TAB\_PRIVS

WHERE grantee like 'DW\_%';

--cu aceasta comanda putem vedea doar privilegiile mai restrictive.

SET FEEDBACK 1

SET NUMWIDTH 10

SET LINESIZE 80

SET TRIMSPOOL ON

SET TAB OFF

SET PAGESIZE 100

SET ECHO OFF

REM \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

REM Create the UTILIZATOR table to hold users information for application

Prompt \*\*\*\*\*\* Creating UTILIZATOR table ....

CREATE TABLE utilizator

( id\_utilizator NUMBER GENERATED ALWAYS as IDENTITY(START WITH 1 INCREMENT BY 1),

nume\_utilizator VARCHAR(30) CONSTRAINT nume\_utilizator\_nn NOT NULL,

hash\_parola VARCHAR(25) CONSTRAINT hash\_parola\_utilizator\_nn NOT NULL,

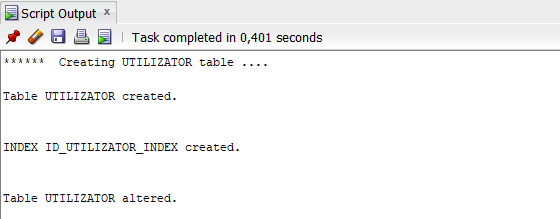
nume\_complet VARCHAR(30) CONSTRAINT nume\_complet\_utilizator\_nn NOT NULL,

telefon VARCHAR(15) CONSTRAINT telefon\_utilizator\_nn NOT NULL,

email VARCHAR(50) CONSTRAINT email\_utilizator\_nn NOT NULL,

data\_nasterii DATE CONSTRAINT data\_nasterii\_utilizator\_nn NOT NULL,

gen VARCHAR(20) DEFAULT NULL,

**** stare\_civila VARCHAR(20) DEFAULT NULL);

CREATE UNIQUE INDEX id\_utilizator\_index

ON utilizator (id\_utilizator);

ALTER TABLE utilizator

ADD ( CONSTRAINT id\_nume\_utilizator\_pk PRIMARY KEY (id\_utilizator)) ;

CREATE UNIQUE INDEX id\_utilizator\_index

ON utilizator (id\_utilizator);

ALTER TABLE utilizator

ADD ( CONSTRAINT id\_nume\_utilizator\_pk PRIMARY KEY (id\_utilizator)) ;

REM \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

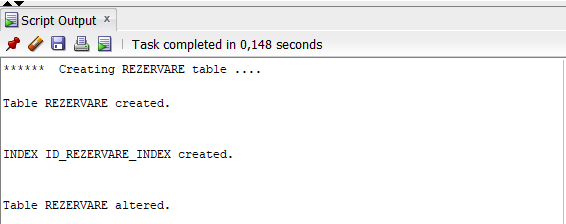
REM Create the REZERVARE table to hold information for reservation of users

Prompt \*\*\*\*\*\* Creating REZERVARE table ....

CREATE TABLE rezervare

( id\_rezervare NUMBER GENERATED ALWAYS as IDENTITY(START WITH 1 INCREMENT BY 1),

id\_client NUMBER CONSTRAINT id\_client\_utilizator\_nn NOT NULL,

 data\_inceput DATE CONSTRAINT data\_inceput\_rezervare\_nn NOT NULL,

data\_sfarsit DATE CONSTRAINT data\_sfarsit\_rezervare\_nn NOT NULL,

data\_efectuarii DATE);

CREATE UNIQUE INDEX id\_rezervare\_index

ON rezervare (id\_rezervare);

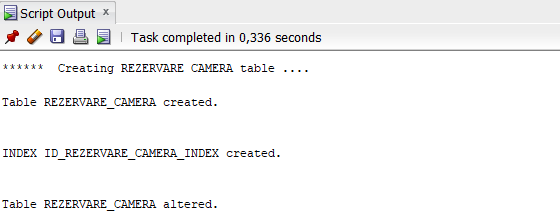
ALTER TABLE rezervare

ADD ( CONSTRAINT id\_rezervare\_pk PRIMARY KEY (id\_rezervare)) ;

REM \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

REM Create the REZERVARE CAMERA table to hold information about rooms of users

Prompt \*\*\*\*\*\* Creating REZERVARE CAMERA table ....



CREATE TABLE rezervare\_camera

( id\_rezervare NUMBER CONSTRAINT id\_rezervare\_atribuie\_nn NOT NULL,

id\_camera NUMBER CONSTRAINT id\_camera\_atribuie\_nn NOT NULL);

CREATE UNIQUE INDEX id\_rezervare\_camera\_index

ON rezervare\_camera (id\_rezervare,id\_camera);

ALTER TABLE rezervare\_camera

ADD ( CONSTRAINT id\_rezervare\_camera\_\_pk PRIMARY KEY (id\_rezervare,id\_camera));

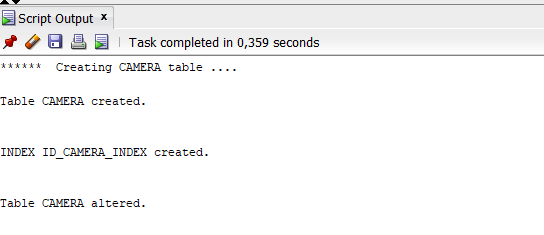
REM \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

REM Create the CAMERA table to hold informations about rooms

Prompt \*\*\*\*\*\* Creating CAMERA table ....

CREATE TABLE camera

( id\_camera NUMBER GENERATED ALWAYS as IDENTITY(START WITH 1 INCREMENT BY 1),

 id\_hotel NUMBER CONSTRAINT id\_hotel\_camera\_nn NOT NULL,

nr\_camera NUMBER,

nr\_etaj NUMBER,

nr\_paturi\_duble NUMBER CONSTRAINT nr\_paturi\_duble\_camera\_nn NOT NULL,

nr\_paturi\_simple NUMBER CONSTRAINT nr\_paturi\_simple\_camera\_nn NOT NULL,

are\_terasa NUMBER(1) CONSTRAINT are\_terasa\_camera\_nn NOT NULL,

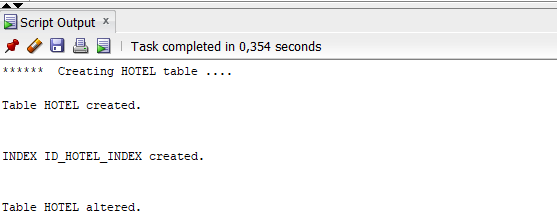
are\_televizor NUMBER(1) CONSTRAINT are\_televizor\_camera\_nn NOT NULL,

pret\_per\_noapte NUMBER CONSTRAINT pret\_per\_noapte\_camera\_nn NOT NULL);

CREATE UNIQUE INDEX id\_camera\_index

ON camera (id\_camera);

ALTER TABLE camera

ADD ( CONSTRAINT id\_camera\_camera\_pk PRIMARY KEY (id\_camera)) ;

REM \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

REM Create the HOTEL table to hold information of hotels

Prompt \*\*\*\*\*\* Creating HOTEL table ....

CREATE TABLE hotel

( id\_hotel NUMBER GENERATED ALWAYS as IDENTITY(START WITH 1 INCREMENT BY 1),

nume VARCHAR2(50) CONSTRAINT nume\_hotel\_hotel\_nn NOT NULL,

nr\_stele NUMBER CONSTRAINT nr\_stele\_nn NOT NULL,

id\_zona NUMBER CONSTRAINT id\_zona\_hotel\_nn NOT NULL,

are\_mic\_dejun\_inclus NUMBER(1) CONSTRAINT are\_mic\_dejun\_inclus\_hotel\_nn NOT NULL);

CREATE UNIQUE INDEX id\_hotel\_index ON hotel (id\_hotel);

ALTER TABLE hotel ADD ( CONSTRAINT id\_hotel\_hotel\_pk PRIMARY KEY (id\_hotel)) ;

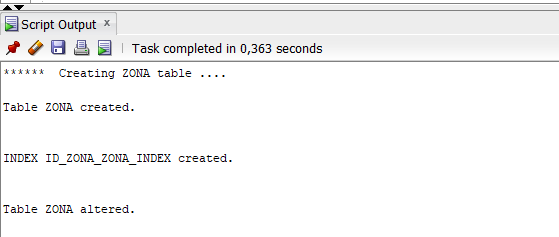
REM \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

REM Create the ZONA table to hold informatation the zones where the hotels was build

Prompt \*\*\*\*\*\* Creating ZONA table ....

CREATE TABLE zona

( id\_zona NUMBER GENERATED ALWAYS as IDENTITY(START WITH 1 INCREMENT BY 1),

 regiune VARCHAR2(50),

judet VARCHAR2(50) CONSTRAINT judet\_zona\_nn NOT NULL,

localitate VARCHAR2(50) CONSTRAINT localitate\_zona\_nn NOT NULL,

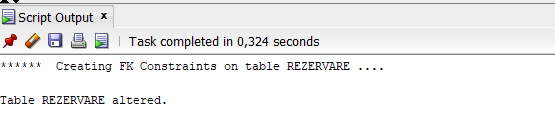
pozitie VARCHAR2(50) CONSTRAINT pozitie\_zona\_nn NOT NULL);

CREATE UNIQUE INDEX id\_zona\_zona\_index

ON zona (id\_zona);

ALTER TABLE zona

ADD ( CONSTRAINT id\_zona\_zona\_pk PRIMARY KEY (id\_zona)) ;

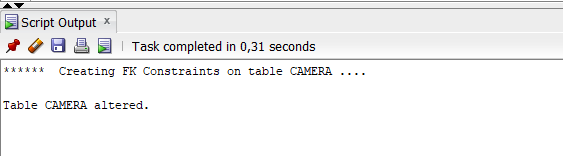


REM \*\*\*\*\*\*\*\*\*Introducerea de FK tabelului\*\*\*\* REZERVARE\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Prompt \*\*\*\*\*\* Creating FK Constraints on table REZERVARE ....

ALTER TABLE rezervare

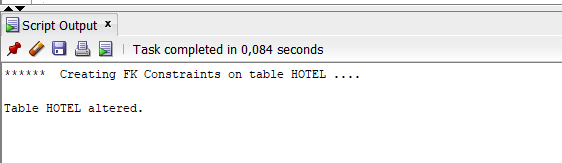
add constraint fk\_id\_client\_id\_utilizator FOREIGN KEY(id\_client) REFERENCES utilizator(id\_utilizator);

REM \*\*\*\*\*\*\*\*\*Introducerea de FK tabelului\*\*\*\* CAMERA\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Prompt \*\*\*\*\*\* Creating FK Constraints on table CAMERA ....

ALTER TABLE camera

add constraint fk\_camera\_hotel FOREIGN KEY(id\_hotel) REFERENCES hotel(id\_hotel);



REM \*\*\*\*\*\*\*\*\*Introducerea de FK tabelului\*\*\*\* HOTEL\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Prompt \*\*\*\*\*\* Creating FK Constraints on table HOTEL ....

ALTER TABLE hotel

add constraint fk\_hotel\_zona FOREIGN KEY(id\_zona) REFERENCES zona(id\_zona);

REM \*\*\*\*\*\*\*\*\*Introducerea de FK tabelului\*\*\*\* REZERVARE\_CAMERA\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

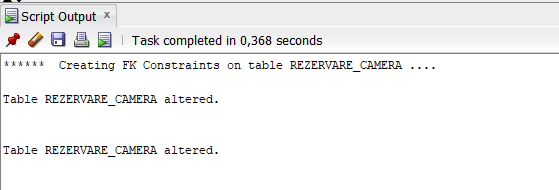
Prompt \*\*\*\*\*\* Creating FK Constraints on table REZERVARE\_CAMERA ....

ALTER TABLE rezervare\_camera

ADD CONSTRAINT fk\_rezervare\_camera\_rezervare FOREIGN KEY(id\_rezervare) REFERENCES rezervare(id\_rezervare);

ALTER TABLE rezervare\_camera

ADD CONSTRAINT fk\_rezervare\_camera\_camera FOREIGN KEY (id\_camera) REFERENCES camera(id\_camera);



2. (0,25p) Generarea datelor și inserarea acestora în tabele.

---Inserarea datelor in tabelul ZONA---

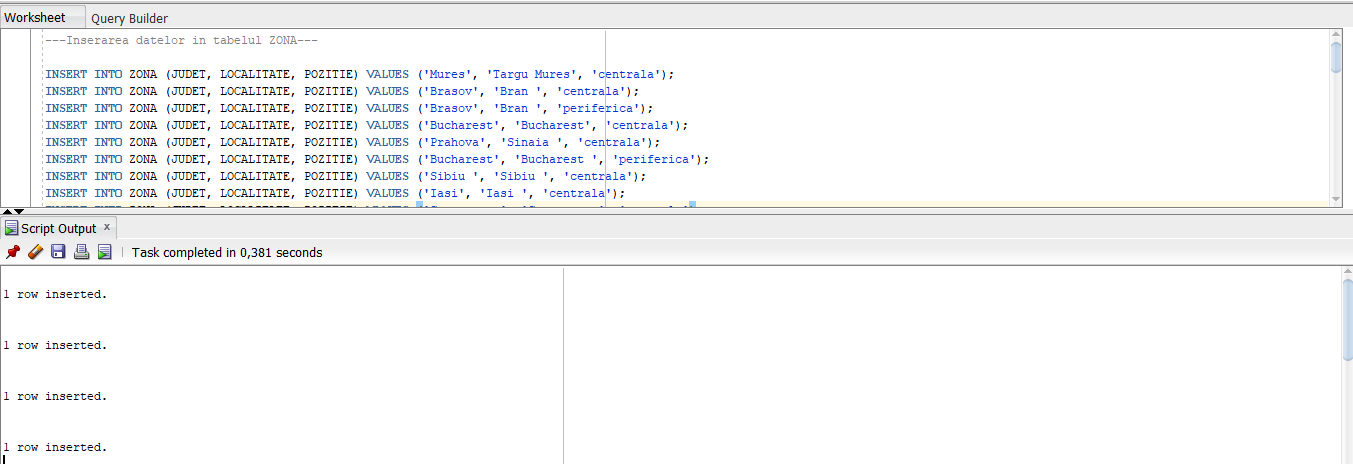
INSERT INTO ZONA (JUDET, LOCALITATE, POZITIE) VALUES ('Mures','Targu Mures','centrala'); COMMIT;

INSERT INTO ZONA (JUDET, LOCALITATE, POZITIE) VALUES ('Brasov','Bran','centrala'); COMMIT;

INSERT INTO ZONA (JUDET, LOCALITATE, POZITIE) VALUES ('Brasov','Bran','periferica'); COMMIT;

INSERT INTO ZONA (JUDET, LOCALITATE, POZITIE) VALUES ('Bucuresti','Bucuresti','centrala'); COMMIT;

INSERT INTO ZONA (JUDET, LOCALITATE, POZITIE) VALUES ('Prahova','Sinaia','centrala'); COMMIT;



---Inserarea datelor in tabelul HOTEL(dependent de ZONA)---

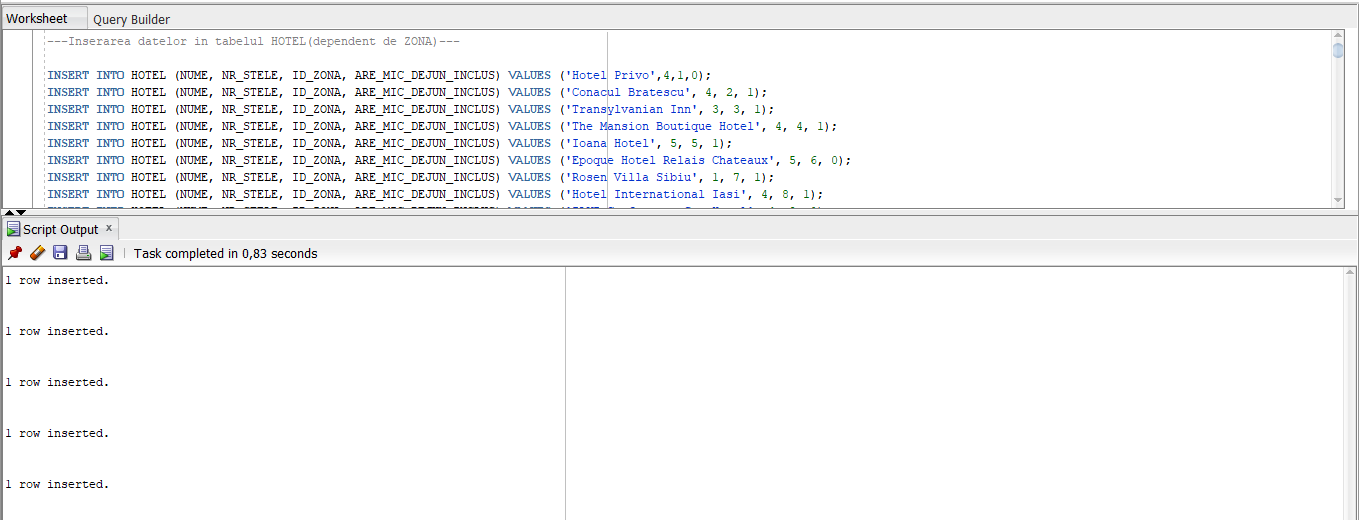
INSERT INTO HOTEL (NUME, NR\_STELE, ID\_ZONA, ARE\_MIC\_DEJUN\_INCLUS) VALUES ('Hotel Privo',4,1,0); COMMIT;

INSERT INTO HOTEL (NUME, NR\_STELE, ID\_ZONA, ARE\_MIC\_DEJUN\_INCLUS) VALUES ('Conacul Bratescu', 4, 2, 1); COMMIT;

INSERT INTO HOTEL (NUME, NR\_STELE, ID\_ZONA, ARE\_MIC\_DEJUN\_INCLUS) VALUES ('Transylvanian Inn', 3, 3, 1); COMMIT;

INSERT INTO HOTEL (NUME, NR\_STELE, ID\_ZONA, ARE\_MIC\_DEJUN\_INCLUS) VALUES ('The Mansion Boutique Hotel', 4, 4, 1); COMMIT;

INSERT INTO HOTEL (NUME, NR\_STELE, ID\_ZONA, ARE\_MIC\_DEJUN\_INCLUS) VALUES ('Ioana Hotel', 5, 5, 1); COMMIT;



---Inserarea datelor in tabelul CAMERA(dependent de HOTEL)---

INSERT INTO CAMERA (ID\_HOTEL, NR\_CAMERA, NR\_ETAJ, NR\_PATURI\_DUBLE, NR\_PATURI\_SIMPLE, ARE\_TERASA, ARE\_TELEVIZOR, PRET\_PER\_NOAPTE)

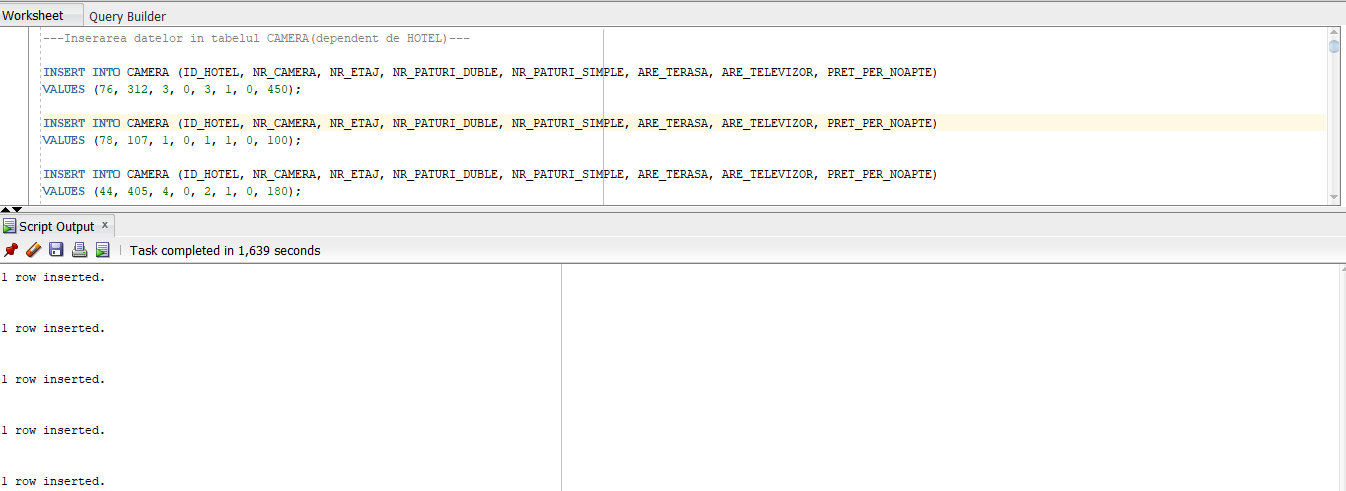
VALUES (76, 312, 3, 0, 3, 1, 0, 450); COMMIT;

INSERT INTO CAMERA (ID\_HOTEL, NR\_CAMERA, NR\_ETAJ, NR\_PATURI\_DUBLE, NR\_PATURI\_SIMPLE, ARE\_TERASA, ARE\_TELEVIZOR, PRET\_PER\_NOAPTE)

VALUES (78, 107, 1, 0, 1, 1, 0, 100); COMMIT;

INSERT INTO CAMERA (ID\_HOTEL, NR\_CAMERA, NR\_ETAJ, NR\_PATURI\_DUBLE, NR\_PATURI\_SIMPLE, ARE\_TERASA, ARE\_TELEVIZOR, PRET\_PER\_NOAPTE)

VALUES (44, 405, 4, 0, 2, 1, 0, 180); COMMIT;



---Inserarea datelor in tabelul UTILIZATOR---

INSERT INTO UTILIZATOR (NUME\_UTILIZATOR, HASH\_PAROLA, NUME\_COMPLET, TELEFON, EMAIL, DATA\_NASTERII, GEN, STARE\_CIVILA)

VALUES ('bernard\_noble', 'kr1O2h8', 'Bernard Noble', '+40 710 024 027', 'kstoltenberg@yahoo.com', to\_date('13-Oct-1976', 'DD-MON-RR'), 'masculin', 'necasatorit '); COMMIT;

INSERT INTO UTILIZATOR (NUME\_UTILIZATOR, HASH\_PAROLA, NUME\_COMPLET, TELEFON, EMAIL, DATA\_NASTERII, GEN, STARE\_CIVILA)

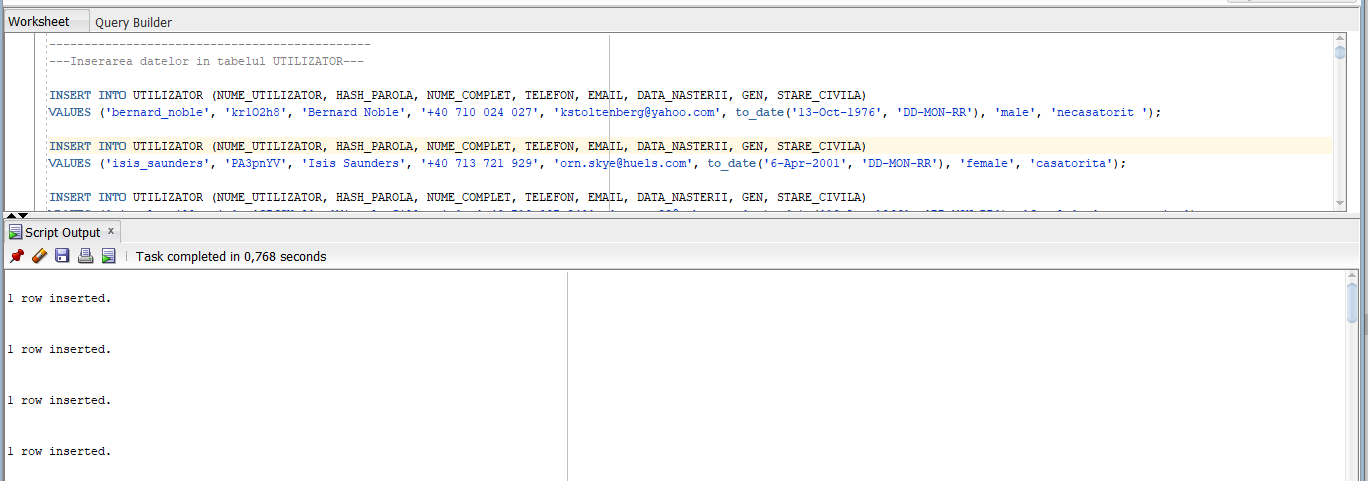
VALUES ('isis\_saunders', 'PA3pnYV', 'Isis Saunders', '+40 713 721 929', 'orn.skye@huels.com', to\_date('6-Apr-2001', 'DD-MON-RR'), 'feminin', 'casatorita'); COMMIT;

INSERT INTO UTILIZATOR (NUME\_UTILIZATOR, HASH\_PAROLA, NUME\_COMPLET, TELEFON, EMAIL, DATA\_NASTERII, GEN, STARE\_CIVILA)

VALUES ('micaela\_gillespie', 'G7fSXy8', 'Micaela Gillespie', '+40 713 037 240', 'stuart22@yahoo.com', to\_date('19-Apr-1966', 'DD-MON-RR'), 'feminin', 'necasatorita'); COMMIT;

INSERT INTO UTILIZATOR (NUME\_UTILIZATOR, HASH\_PAROLA, NUME\_COMPLET, TELEFON, EMAIL, DATA\_NASTERII, GEN, STARE\_CIVILA)

VALUES ('gilbert\_mccarty', 'ek1Rwcq', 'Gilbert Mccarty', '+40 711 666 147', 'gskiles@altenwerth.com', to\_date('31-Jul-1969', 'DD-MON-RR'), 'masculin', 'necasatorit '); COMMIT;



---Inserarea datelor in tabelul REZERVARE(dependent de UTILIZATOR)---

INSERT INTO REZERVARE (ID\_CLIENT, DATA\_INCEPUT, DATA\_SFARSIT)

VALUES (89, to\_date('2-Jul-2023', 'DD-MON-RR'), to\_date('9-Jul-2023', 'DD-MON-RR')); COMMIT;

INSERT INTO REZERVARE (ID\_CLIENT, DATA\_INCEPUT, DATA\_SFARSIT)

VALUES (49, to\_date('4-Feb-2025', 'DD-MON-RR'), to\_date('13-Feb-2025', 'DD-MON-RR')); COMMIT;

INSERT INTO REZERVARE (ID\_CLIENT, DATA\_INCEPUT, DATA\_SFARSIT)

VALUES (4, to\_date('24-Apr-2023', 'DD-MON-RR'), to\_date('27-Apr-2023', 'DD-MON-RR')); COMMIT;



---Inserarea datelor in tabelul REZERVARE\_CAMERA---

INSERT INTO REZERVARE\_CAMERA (ID\_REZERVARE, ID\_CAMERA)

VALUES (95, 167);

INSERT INTO REZERVARE\_CAMERA (ID\_REZERVARE, ID\_CAMERA)

VALUES (93, 54);

INSERT INTO REZERVARE\_CAMERA (ID\_REZERVARE, ID\_CAMERA)

VALUES (47, 140);

INSERT INTO REZERVARE\_CAMERA (ID\_REZERVARE, ID\_CAMERA)

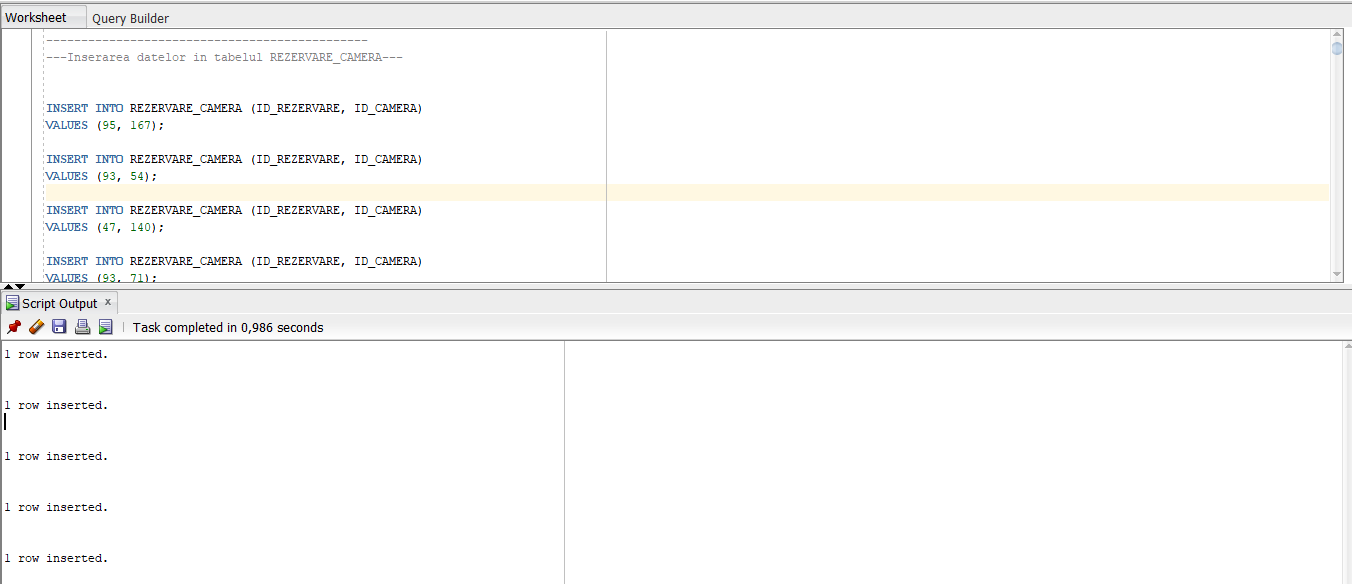
VALUES (93, 71);

INSERT INTO REZERVARE\_CAMERA (ID\_REZERVARE, ID\_CAMERA)

VALUES (10, 154);

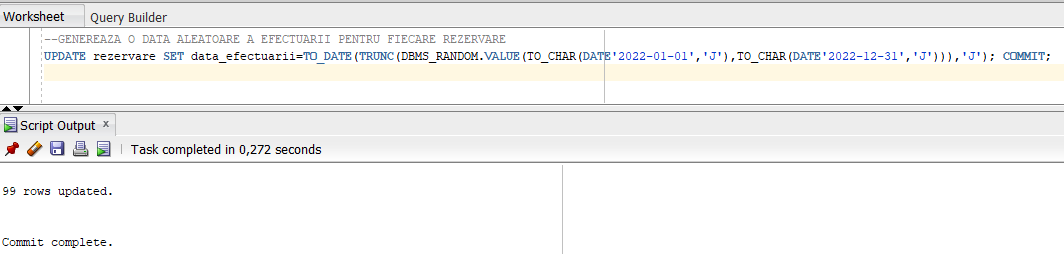
INSERT INTO REZERVARE\_CAMERA (ID\_REZERVARE, ID\_CAMERA)

VALUES (4, 120);



--GENEREAZA O DATA ALEATOARE A EFECTUARII PENTRU FIECARE REZERVARE

UPDATE rezervare SET data\_efectuarii=TO\_DATE(TRUNC(DBMS\_RANDOM.VALUE(TO\_CHAR(DATE'2022-01-01','J'),TO\_CHAR(DATE'2022-12-31','J'))),'J'); COMMIT;



--SETEAZA REGIUNILE ASOCIATE JUDETELOR

UPDATE zona

SET regiune='Transilvania'

WHERE judet='Cluj' OR judet='Brasov' OR judet='Sibiu' OR judet='Mures'; COMMIT;

UPDATE zona

SET regiune='Maramures'

WHERE judet='Maramures' OR judet='Satu Mare'; COMMIT;

UPDATE zona

SET regiune='Muntenia'

WHERE judet='Prahova' OR judet='Ilfov' OR judet='Arges' OR judet='Buzau' OR judet='Bucuresti'; COMMIT;

UPDATE zona

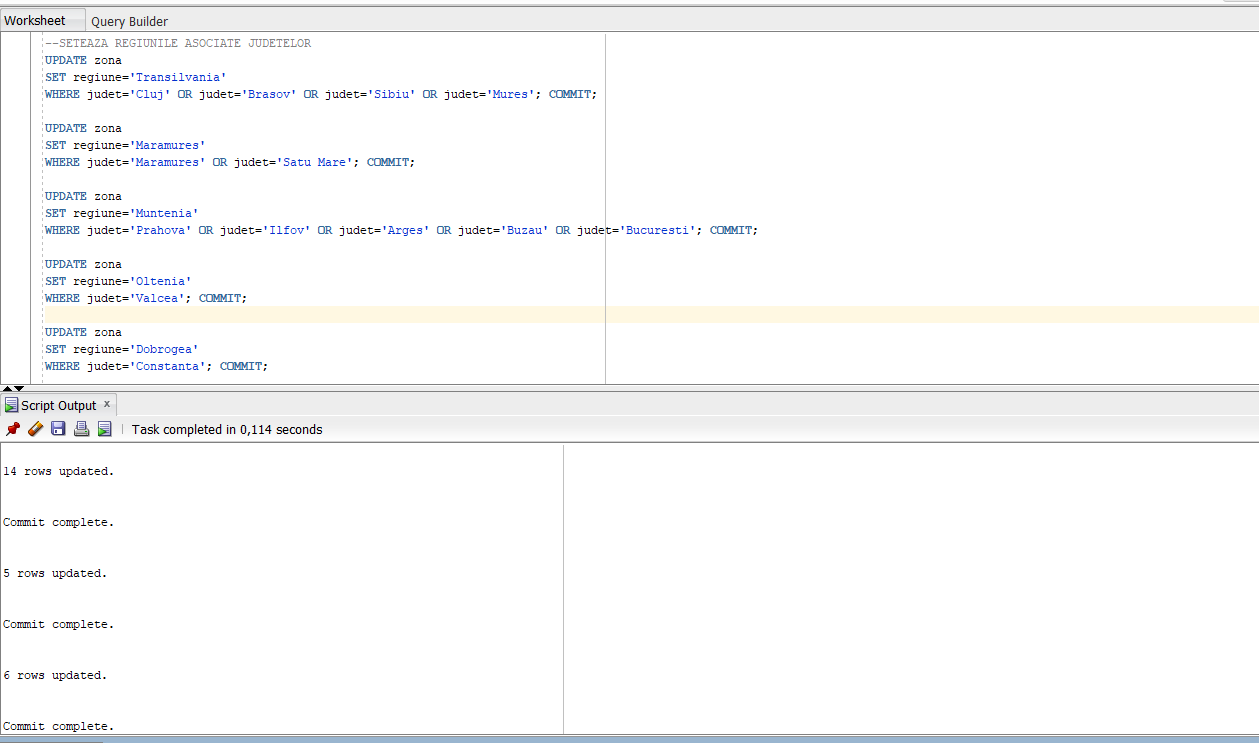
SET regiune='Oltenia'

WHERE judet='Valcea'; COMMIT;

UPDATE zona

SET regiune='Dobrogea'

WHERE judet='Constanta'; COMMIT;



3. (0,5p) Crearea bazei de date depozit și a utilizatorilor

Schema bazei de date depozit a fost introdusă în conexiunea utilizatorului dw\_manager. Acest utilizator a fost creat asemănător cu utlizatorul pentru baza de date OLTP.

CREATE TABLE perioada\_rezervare\_OLAP(

id\_perioada NUMBER(8,0) GENERATED ALWAYS as IDENTITY(START WITH 1 INCREMENT BY 1),

zi\_din\_luna\_inceput NUMBER(2,0) CONSTRAINT zi\_din\_luna\_inceput\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

luna\_inceput CHAR(3) CONSTRAINT luna\_inceput\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

an\_inceput NUMBER(4,0)CONSTRAINT an\_inceput\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

zi\_din\_saptamana\_inceput CHAR(3) CONSTRAINT zi\_din\_saptamana\_inceput\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

zi\_din\_an\_inceput NUMBER(3,0) CONSTRAINT zi\_din\_an\_inceput\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

zi\_din\_luna\_sfarsit NUMBER(2,0) CONSTRAINT zi\_din\_luna\_sfarsit\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

luna\_sfarsit CHAR(3) CONSTRAINT luna\_sfarsit\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

an\_sfarsit NUMBER(4,0) CONSTRAINT an\_sfarsit\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

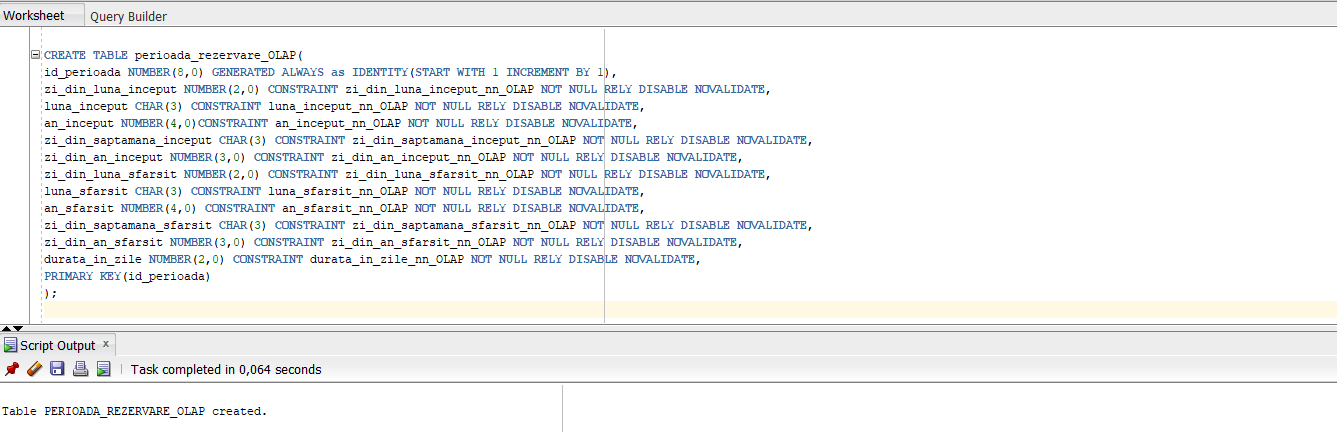
zi\_din\_saptamana\_sfarsit CHAR(3) CONSTRAINT zi\_din\_saptamana\_sfarsit\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

zi\_din\_an\_sfarsit NUMBER(3,0) CONSTRAINT zi\_din\_an\_sfarsit\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

durata\_in\_zile NUMBER(2,0) CONSTRAINT durata\_in\_zile\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

PRIMARY KEY(id\_perioada)

);



CREATE TABLE tip\_camera\_OLAP(

id\_tip\_camera NUMBER(8)GENERATED ALWAYS as IDENTITY(START WITH 1 INCREMENT BY 1),

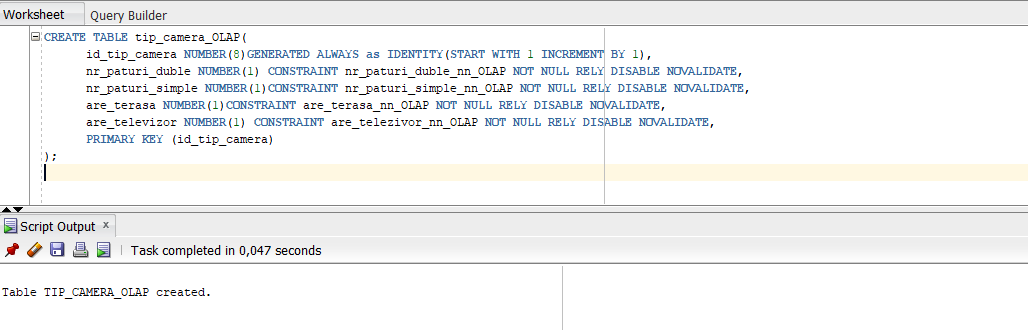
nr\_paturi\_duble NUMBER(1) CONSTRAINT nr\_paturi\_duble\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

nr\_paturi\_simple NUMBER(1)CONSTRAINT nr\_paturi\_simple\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

are\_terasa NUMBER(1)CONSTRAINT are\_terasa\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

are\_televizor NUMBER(1) CONSTRAINT are\_telezivor\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

PRIMARY KEY (id\_tip\_camera));



CREATE TABLE hotel\_OLAP

(

id\_hotel NUMBER(8),

nume VARCHAR2(50) CONSTRAINT nume\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

nr\_stele NUMBER(1) CONSTRAINT nr\_stele\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

regiune VARCHAR2(50),

judet VARCHAR2(20) CONSTRAINT judet\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

localitate VARCHAR(20) CONSTRAINT localitate\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

pozitie VARCHAR(20) CONSTRAINT pozitie\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

nr\_camere NUMBER(3),

are\_mic\_dejun\_inclus NUMBER(1) CONSTRAINT are\_mic\_dejun\_inclus\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

PRIMARY KEY (id\_hotel))

PARTITION BY LIST (nr\_stele)

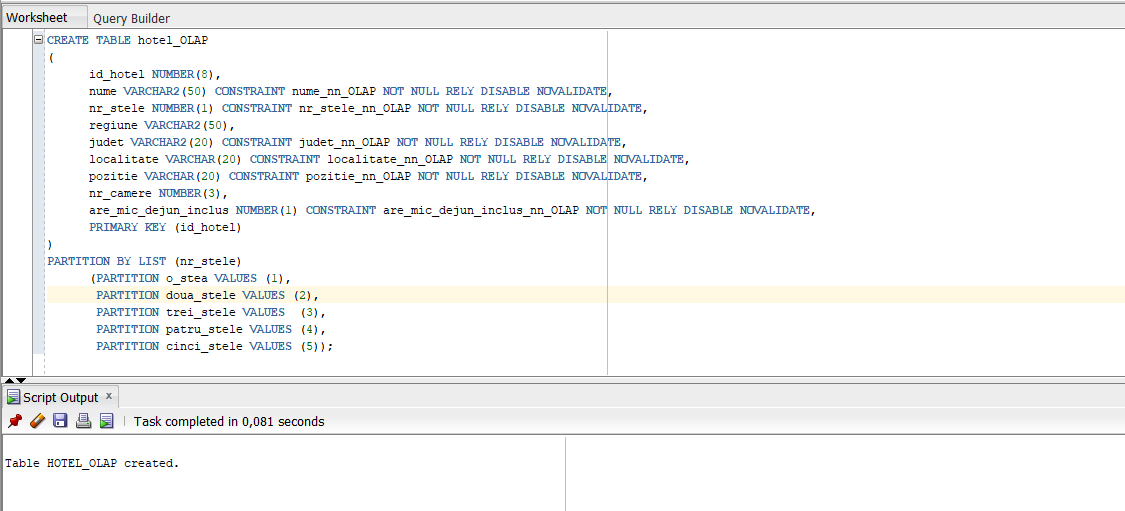
(PARTITION o\_stea VALUES (1),

PARTITION doua\_stele VALUES (2),

PARTITION trei\_stele VALUES (3),

PARTITION patru\_stele VALUES (4),

PARTITION cinci\_stele VALUES (5));



CREATE TABLE tip\_client\_OLAP(

id\_tip\_client NUMBER(8) GENERATED ALWAYS as IDENTITY(START WITH 1 INCREMENT BY 1),

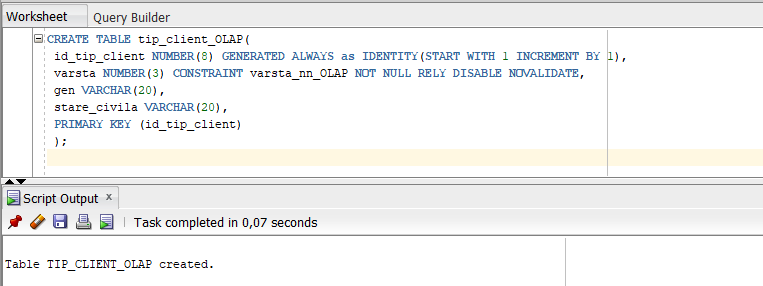
varsta NUMBER(3) CONSTRAINT varsta\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

gen VARCHAR(20),

stare\_civila VARCHAR(20),

PRIMARY KEY (id\_tip\_client)

);



CREATE TABLE moment\_efectuare\_rezervare\_OLAP(

id\_moment\_efectuare NUMBER(8,0) GENERATED ALWAYS as IDENTITY(START WITH 1 INCREMENT BY 1),

zi\_din\_luna NUMBER(2,0) CONSTRAINT zi\_din\_luna\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

luna CHAR(3) CONSTRAINT luna\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

an NUMBER(4,0) CONSTRAINT an\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

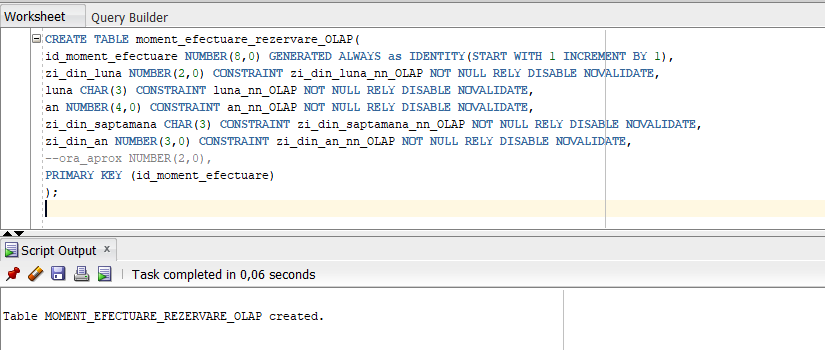
zi\_din\_saptamana CHAR(3) CONSTRAINT zi\_din\_saptamana\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

zi\_din\_an NUMBER(3,0) CONSTRAINT zi\_din\_an\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

--ora\_aprox NUMBER(2,0),

PRIMARY KEY (id\_moment\_efectuare)

);



DROP TABLE rezervare\_camera\_OLAP;

CREATE TABLE rezervare\_camera\_OLAP(

id\_rezervare NUMBER(8,0) CONSTRAINT id\_rezervare\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

id\_hotel NUMBER(8,0) CONSTRAINT id\_hotel\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

id\_perioada NUMBER(8,0) CONSTRAINT id\_perioada\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

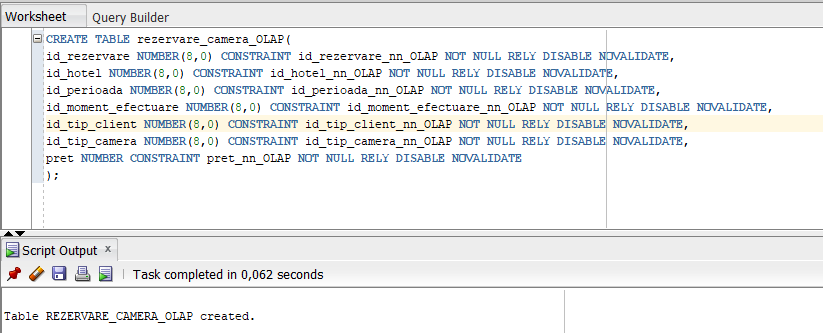
id\_moment\_efectuare NUMBER(8,0) CONSTRAINT id\_moment\_efectuare\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

id\_tip\_client NUMBER(8,0) CONSTRAINT id\_tip\_client\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

id\_tip\_camera NUMBER(8,0) CONSTRAINT id\_tip\_camera\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

pret NUMBER CONSTRAINT pret\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE

);



ALTER TABLE rezervare\_camera\_OLAP

ADD FOREIGN KEY(id\_hotel) REFERENCES hotel\_OLAP(id\_hotel);

ALTER TABLE rezervare\_camera\_OLAP

ADD FOREIGN KEY(id\_perioada) REFERENCES perioada\_rezervare\_OLAP(id\_perioada);

ALTER TABLE rezervare\_camera\_OLAP

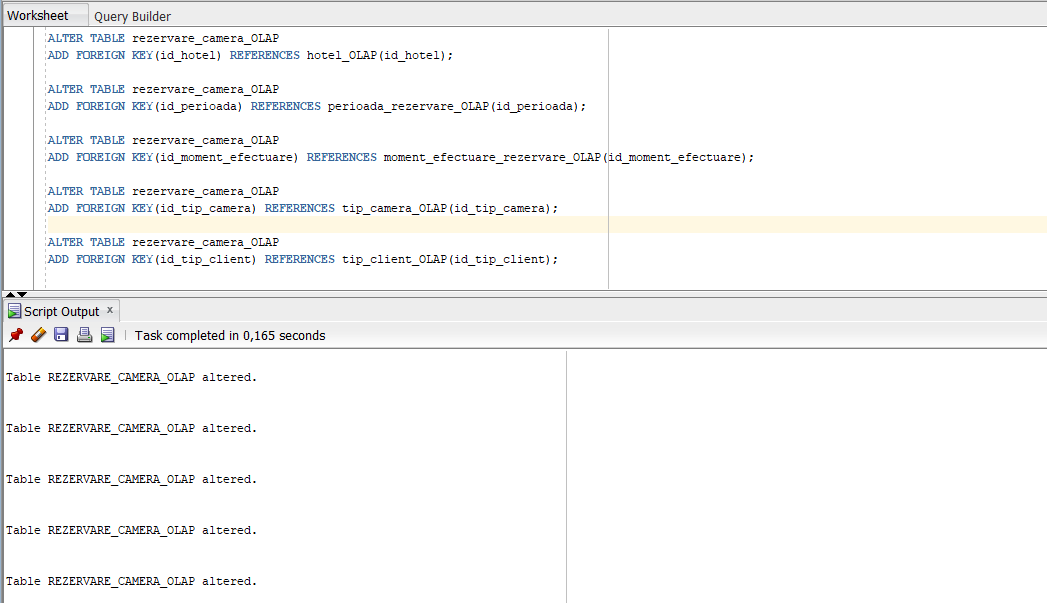
ADD FOREIGN KEY(id\_moment\_efectuare) REFERENCES moment\_efectuare\_rezervare\_OLAP(id\_moment\_efectuare);

ALTER TABLE rezervare\_camera\_OLAP

ADD FOREIGN KEY(id\_tip\_camera) REFERENCES tip\_camera\_OLAP(id\_tip\_camera);

ALTER TABLE rezervare\_camera\_OLAP

ADD FOREIGN KEY(id\_tip\_client) REFERENCES tip\_client\_OLAP(id\_tip\_client);



4. (0,5p) Popularea cu informații a bazei de date depozit folosind ca sursă datele

din baza de date OLTP

INSERT INTO perioada\_rezervare\_OLAP(zi\_din\_luna\_inceput, luna\_inceput, an\_inceput, zi\_din\_saptamana\_inceput, zi\_din\_an\_inceput, zi\_din\_luna\_sfarsit, luna\_sfarsit, an\_sfarsit, zi\_din\_saptamana\_sfarsit, zi\_din\_an\_sfarsit, durata\_in\_zile)

SELECT DISTINCT TO\_NUMBER(TO\_CHAR(data\_inceput,'DD')), TO\_CHAR(data\_inceput,'MON'), TO\_NUMBER(TO\_CHAR(data\_inceput,'YYYY')),

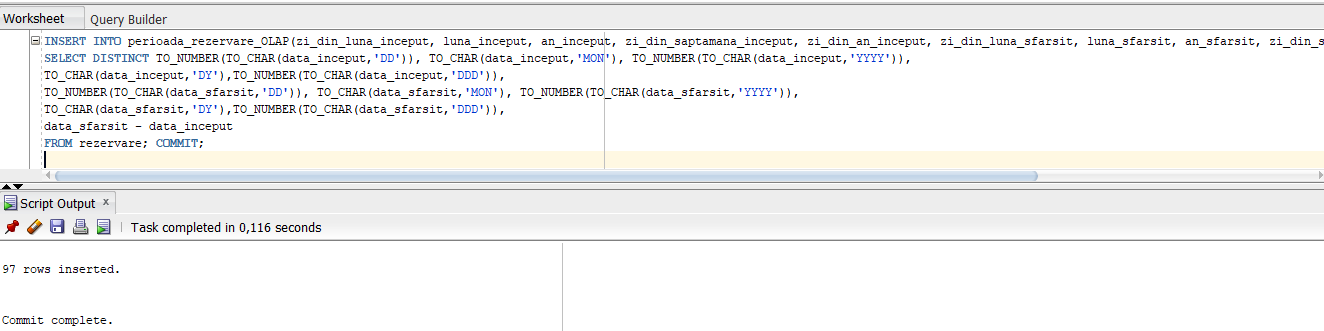
TO\_CHAR(data\_inceput,'DY'),TO\_NUMBER(TO\_CHAR(data\_inceput,'DDD')),

TO\_NUMBER(TO\_CHAR(data\_sfarsit,'DD')), TO\_CHAR(data\_sfarsit,'MON'), TO\_NUMBER(TO\_CHAR(data\_sfarsit,'YYYY')),

TO\_CHAR(data\_sfarsit,'DY'),TO\_NUMBER(TO\_CHAR(data\_sfarsit,'DDD')),

data\_sfarsit - data\_inceput

FROM rezervare; COMMIT;



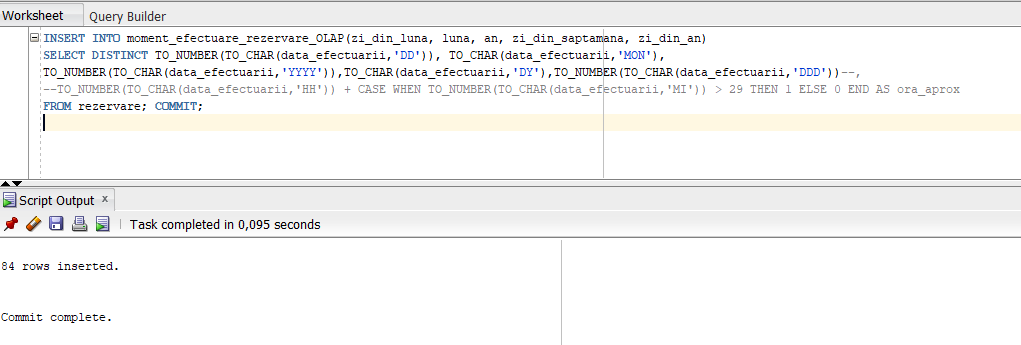
INSERT INTO moment\_efectuare\_rezervare\_OLAP(zi\_din\_luna, luna, an, zi\_din\_saptamana, zi\_din\_an)

SELECT DISTINCT TO\_NUMBER(TO\_CHAR(data\_efectuarii,'DD')), TO\_CHAR(data\_efectuarii,'MON'),

TO\_NUMBER(TO\_CHAR(data\_efectuarii,'YYYY')),TO\_CHAR(data\_efectuarii,'DY'),TO\_NUMBER(TO\_CHAR(data\_efectuarii,'DDD'))--,

--TO\_NUMBER(TO\_CHAR(data\_efectuarii,'HH')) + CASE WHEN TO\_NUMBER(TO\_CHAR(data\_efectuarii,'MI')) > 29 THEN 1 ELSE 0 END AS ora\_aprox

FROM rezervare; COMMIT;

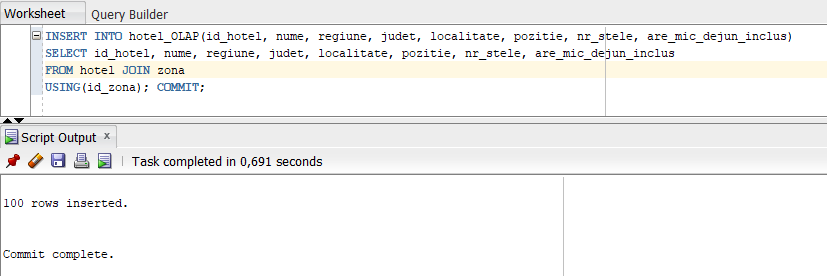


INSERT INTO hotel\_OLAP(id\_hotel, nume, regiune, judet, localitate, pozitie, nr\_stele, are\_mic\_dejun\_inclus)

SELECT id\_hotel, nume, regiune, judet, localitate, pozitie, nr\_stele, are\_mic\_dejun\_inclus

FROM hotel JOIN zona

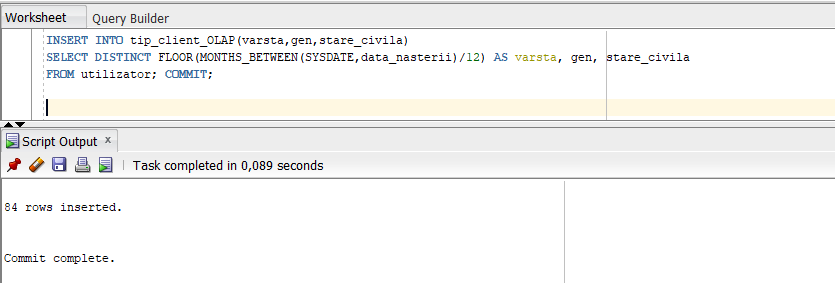
USING(id\_zona); COMMIT;



INSERT INTO tip\_client\_OLAP(varsta,gen,stare\_civila)

SELECT DISTINCT FLOOR(MONTHS\_BETWEEN(SYSDATE,data\_nasterii)/12) AS varsta, gen, stare\_civila

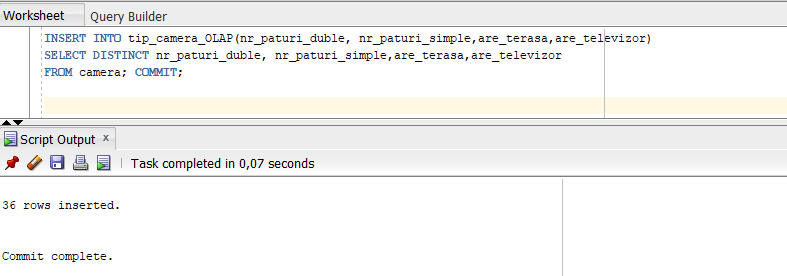
FROM utilizator; COMMIT;



INSERT INTO tip\_camera\_OLAP(nr\_paturi\_duble, nr\_paturi\_simple,are\_terasa,are\_televizor)

SELECT DISTINCT nr\_paturi\_duble, nr\_paturi\_simple,are\_terasa,are\_televizor

FROM camera; COMMIT;



—--functie ce calculeaza varsta clientului-

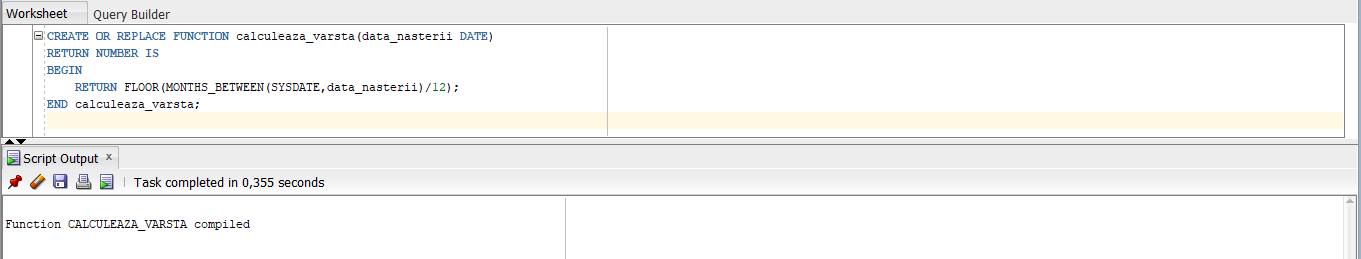
CREATE OR REPLACE FUNCTION calculeaza\_varsta(data\_nasterii DATE)

RETURN NUMBER IS

BEGIN

RETURN FLOOR(MONTHS\_BETWEEN(SYSDATE,data\_nasterii)/12);

END calculeaza\_varsta;



—--functie ce cauta id-ul perioadei de rezervare—

CREATE OR REPLACE FUNCTION gaseste\_id\_perioada\_OLAP(p\_zi\_din\_an\_inceput perioada\_rezervare\_OLAP.zi\_din\_an\_inceput%TYPE, p\_an\_inceput perioada\_rezervare\_OLAP.an\_inceput%TYPE,

p\_zi\_din\_an\_sfarsit perioada\_rezervare\_OLAP.zi\_din\_an\_sfarsit%TYPE, p\_an\_sfarsit perioada\_rezervare\_OLAP.an\_sfarsit%TYPE)

RETURN perioada\_rezervare\_OLAP.id\_perioada%TYPE IS

v\_id\_gasit perioada\_rezervare\_OLAP.id\_perioada%TYPE;

BEGIN

SELECT id\_perioada

INTO v\_id\_gasit

FROM perioada\_rezervare\_OLAP

WHERE zi\_din\_an\_inceput=p\_zi\_din\_an\_inceput

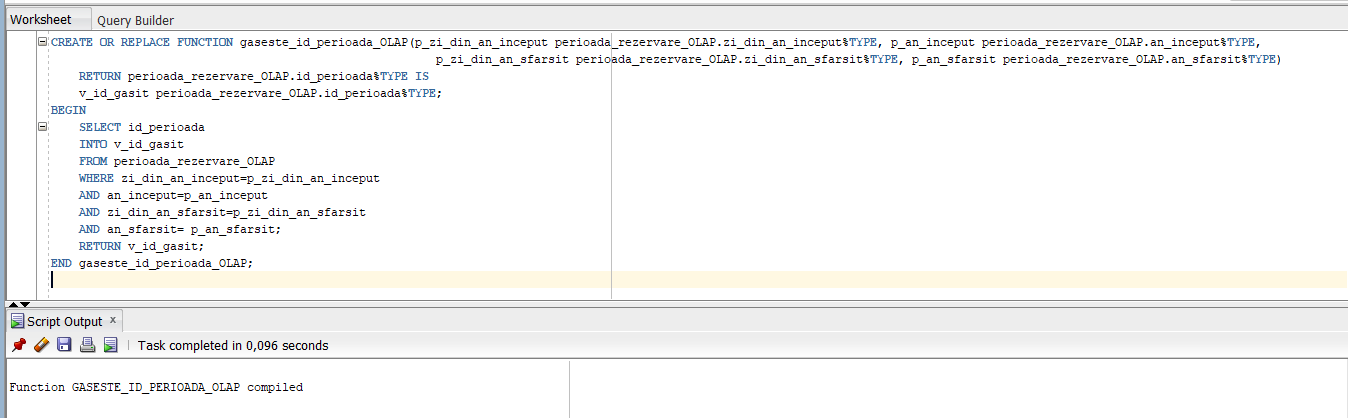
AND an\_inceput=p\_an\_inceput

AND zi\_din\_an\_sfarsit=p\_zi\_din\_an\_sfarsit

AND an\_sfarsit= p\_an\_sfarsit;

RETURN v\_id\_gasit;

END gaseste\_id\_perioada\_OLAP;



—--functie ce cauta id-ul momentului rezervarii—

CREATE OR REPLACE FUNCTION gaseste\_id\_moment\_efectuare\_OLAP(p\_zi\_din\_an moment\_efectuare\_rezervare\_OLAP.zi\_din\_an%TYPE, p\_an moment\_efectuare\_rezervare\_OLAP.an%TYPE)

RETURN moment\_efectuare\_rezervare\_OLAP.id\_moment\_efectuare%TYPE IS

v\_id\_gasit moment\_efectuare\_rezervare\_OLAP.id\_moment\_efectuare%TYPE;

BEGIN

SELECT id\_moment\_efectuare

INTO v\_id\_gasit

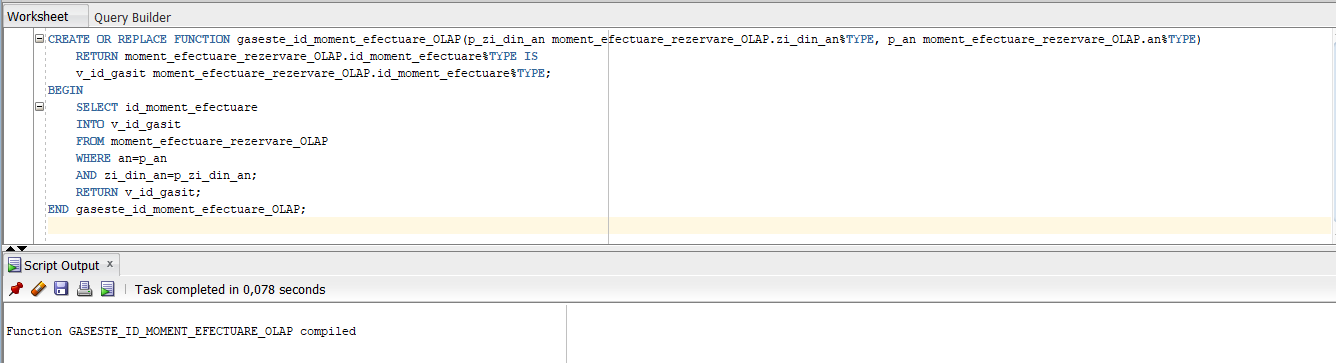
FROM moment\_efectuare\_rezervare\_OLAP

WHERE an=p\_an

AND zi\_din\_an=p\_zi\_din\_an;

RETURN v\_id\_gasit;

END gaseste\_id\_moment\_efectuare\_OLAP;



—--functie ce cauta id-ul camerei—

CREATE OR REPLACE FUNCTION gaseste\_id\_tip\_camera\_OLAP(p\_nr\_paturi\_duble tip\_camera\_OLAP.nr\_paturi\_duble%TYPE, p\_nr\_paturi\_simple tip\_camera\_OLAP.nr\_paturi\_duble%TYPE,

p\_are\_terasa tip\_camera\_OLAP.are\_terasa%TYPE, p\_are\_televizor tip\_camera\_OLAP.are\_televizor%TYPE)

RETURN tip\_camera\_OLAP.id\_tip\_camera%TYPE IS

v\_id\_gasit tip\_camera\_OLAP.id\_tip\_camera%TYPE;

BEGIN

SELECT id\_tip\_camera

INTO v\_id\_gasit

FROM tip\_camera\_OLAP

WHERE nr\_paturi\_duble=p\_nr\_paturi\_duble

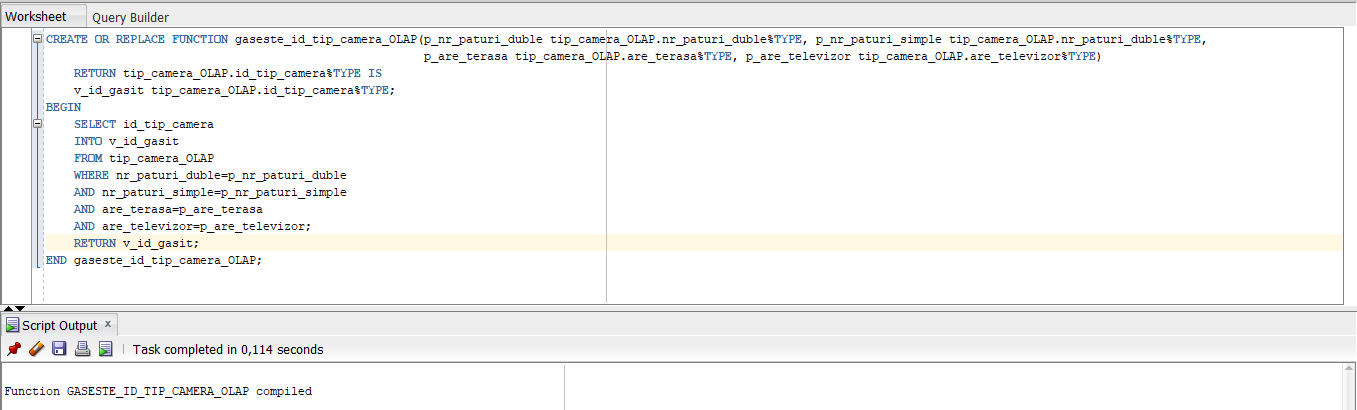
AND nr\_paturi\_simple=p\_nr\_paturi\_simple

AND are\_terasa=p\_are\_terasa

AND are\_televizor=p\_are\_televizor;

RETURN v\_id\_gasit;

END gaseste\_id\_tip\_camera\_OLAP;



—--functie ce cauta id-ul clientului—

CREATE OR REPLACE FUNCTION gaseste\_id\_tip\_client\_OLAP (p\_varsta tip\_client\_OLAP.varsta%TYPE, p\_gen tip\_client\_OLAP.gen%TYPE,p\_stare\_civila tip\_client\_OLAP.stare\_civila%TYPE)

RETURN tip\_client\_OLAP.id\_tip\_client%TYPE IS

v\_id\_gasit tip\_client\_OLAP.id\_tip\_client%TYPE;

BEGIN

SELECT id\_tip\_client

INTO v\_id\_gasit

FROM tip\_client\_OLAP

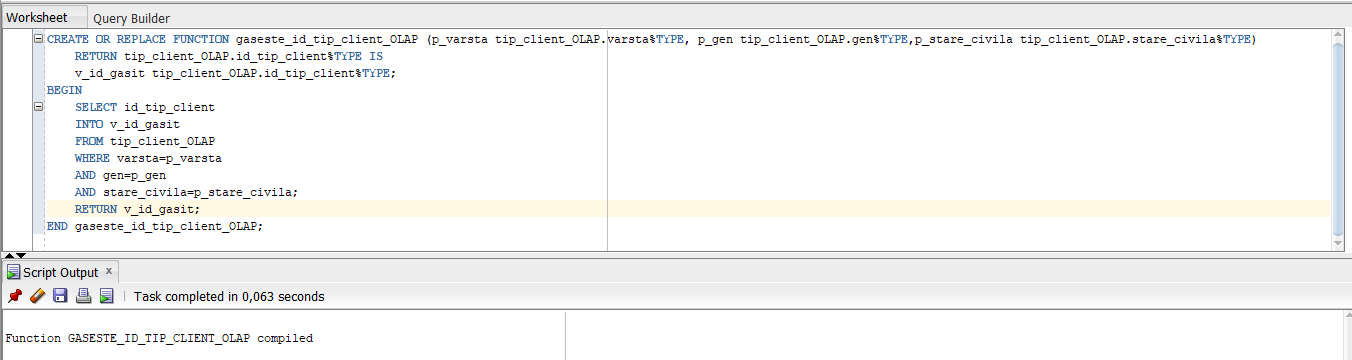
WHERE varsta=p\_varsta

AND gen=p\_gen

AND stare\_civila=p\_stare\_civila;

RETURN v\_id\_gasit;

END gaseste\_id\_tip\_client\_OLAP;



INSERT INTO rezervare\_camera\_OLAP(id\_rezervare,id\_hotel,id\_perioada,id\_moment\_efectuare,id\_tip\_camera,id\_tip\_client,pret)

SELECT DISTINCT id\_rezervare,id\_hotel,

gaseste\_id\_perioada\_OLAP(TO\_NUMBER(TO\_CHAR(data\_inceput,'DDD')),TO\_NUMBER(TO\_CHAR(data\_inceput,'YYYY')),

TO\_NUMBER(TO\_CHAR(data\_sfarsit,'DDD')),TO\_NUMBER(TO\_CHAR(data\_sfarsit,'YYYY'))) AS id\_perioada,

gaseste\_id\_moment\_efectuare\_OLAP(TO\_NUMBER(TO\_CHAR(data\_efectuarii,'DDD')),TO\_NUMBER(TO\_CHAR(data\_efectuarii,'YYYY'))) AS id\_moment\_efectuare,

gaseste\_id\_tip\_camera\_OLAP(camera.nr\_paturi\_duble,camera.nr\_paturi\_simple,camera.are\_terasa,camera.are\_televizor) AS id\_tip\_camera,

gaseste\_id\_tip\_client\_OLAP(calculeaza\_varsta(utilizator.data\_nasterii),utilizator.gen,utilizator.stare\_civila) AS id\_tip\_client,

camera.pret\_per\_noapte \* (rezervare.data\_sfarsit - rezervare.data\_inceput) AS pret

FROM utilizator JOIN rezervare

ON utilizator.id\_utilizator=rezervare.id\_client

JOIN rezervare\_camera

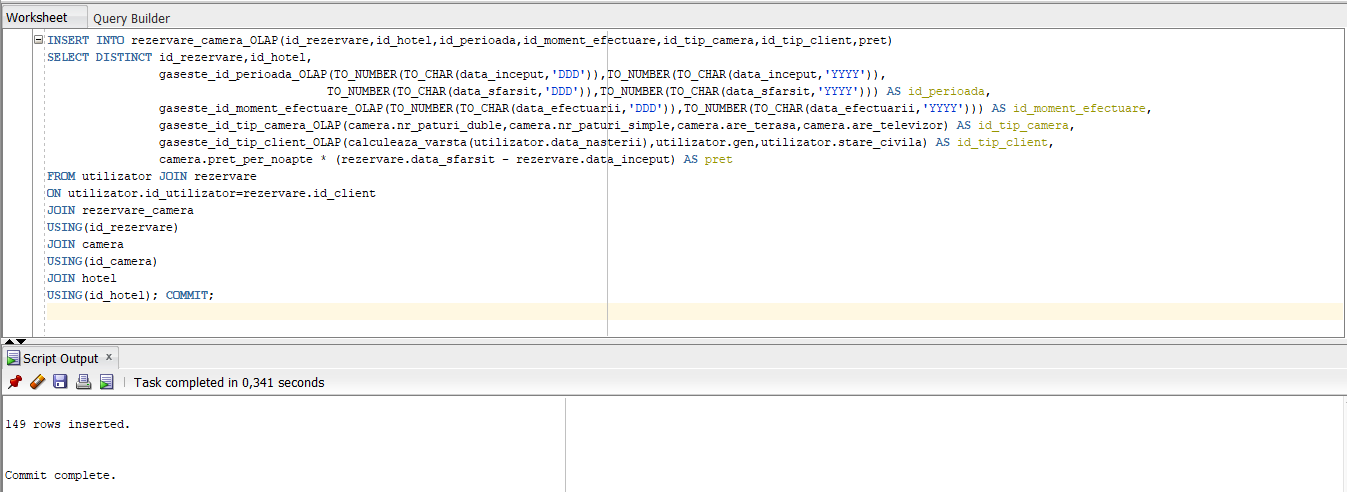
USING(id\_rezervare)

JOIN camera

USING(id\_camera)

JOIN hotel

USING(id\_hotel); COMMIT;



5. (0,5p) Definirea constrângerilor

Constrângerile au fost definite încă de la momentul creării tabelelor bazei de date depozit.

Exemple:

luna\_inceput CHAR(3) CONSTRAINT luna\_inceput\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

nr\_paturi\_duble NUMBER(1) CONSTRAINT nr\_paturi\_duble\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

nr\_stele NUMBER(1) CONSTRAINT nr\_stele\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

varsta NUMBER(3) CONSTRAINT varsta\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

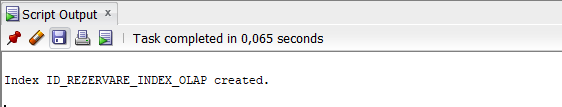
zi\_din\_an NUMBER(3,0) CONSTRAINT zi\_din\_an\_nn\_OLAP NOT NULL RELY DISABLE NOVALIDATE,

6. (1p) Definirea indecșilor și a cererilor SQL însoțite de planul de execuție al

acestora (din care să reiasă ca optimizorul utilizează eficient indecșii definiți)

CREATE INDEX id\_rezervare\_index\_OLAP

ON rezervare\_camera\_OLAP(id\_rezervare);



CREATE BITMAP INDEX luna\_an\_efectuare\_bjindex\_OLAP

ON rezervare\_camera\_OLAP(moment\_efectuare\_OLAP.luna,moment\_efectuare\_OLAP.an)

FROM moment\_efectuare\_OLAP,rezervare\_camera\_OLAP

WHERE rezervare\_camera\_OLAP.id\_moment\_efectuare= moment\_efectuare\_OLAP.id\_moment\_efectuare;

CREATE BITMAP INDEX localitate\_pozitie\_bjindex\_OLAP

ON rezervare\_camera\_OLAP(hotel.localitate,hotel.pozitie)

FROM hotel\_OLAP, rezervare\_camera\_OLAP

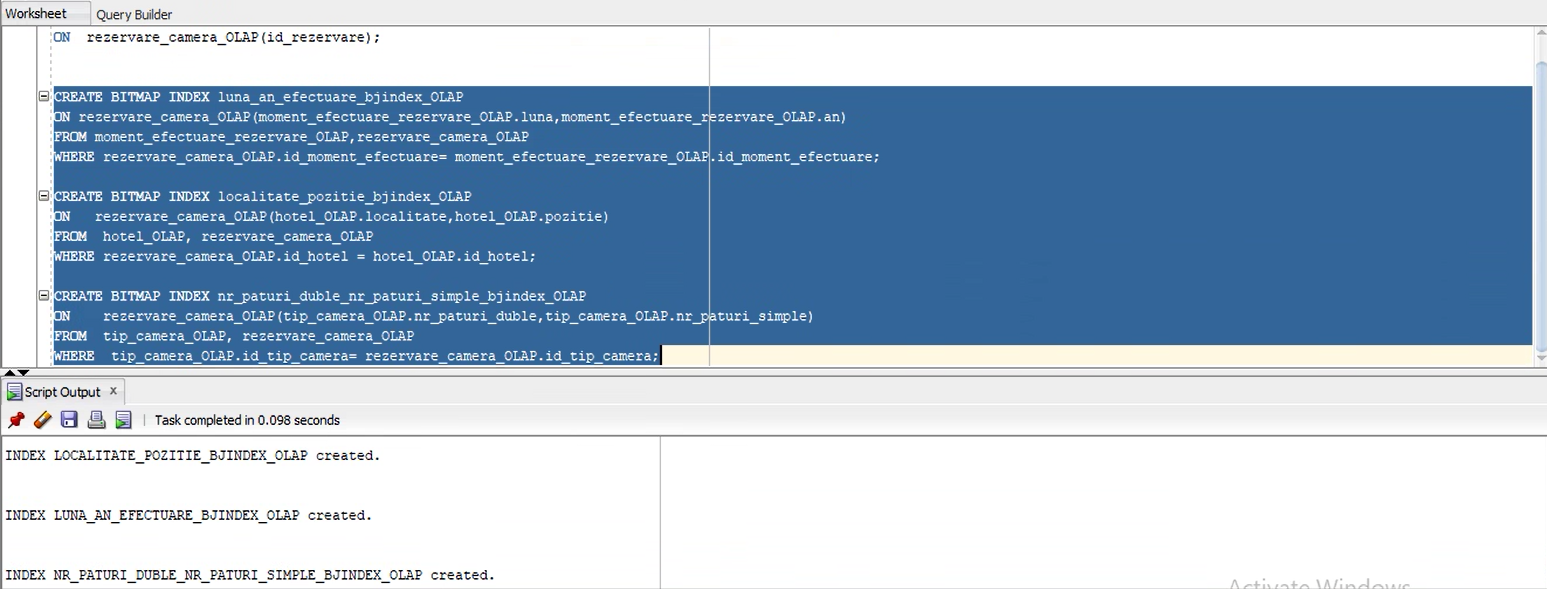
WHERE rezervare\_camera\_OLAP.id\_hotel = hotel\_OLAP.id\_hotel;

CREATE BITMAP INDEX nr\_paturi\_duble\_nr\_paturi\_simple\_bjindex\_OLAP

ON rezervare\_camera\_OLAP(tip\_camera\_OLAP.nr\_paturi\_duble,tip\_camera\_OLAP.nr\_paturi\_simple)

FROM tip\_camera\_OLAP, rezervare\_camera\_OLAP

WHERE tip\_camera\_OLAP.id\_tip\_camera= rezervare\_camera\_OLAP.id\_tip\_camera;



7. (1p) Definirea obiectelor de tip dimensiune, validarea acestora (din care să

reiasă că datele respectă constrângerile impuse prin aceste tipuri de obiecte)

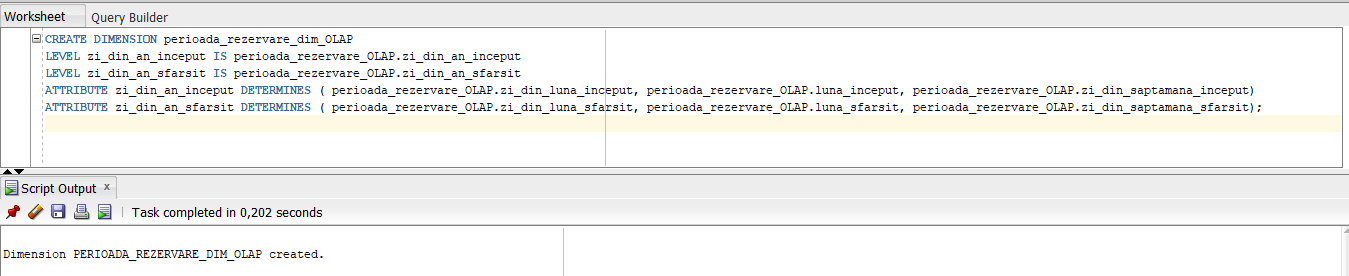
CREATE DIMENSION perioada\_rezervare\_dim\_OLAP

LEVEL zi\_din\_an\_inceput IS perioada\_rezervare\_OLAP.zi\_din\_an\_inceput

LEVEL zi\_din\_an\_sfarsit IS perioada\_rezervare\_OLAP.zi\_din\_an\_sfarsit

ATTRIBUTE zi\_din\_an\_inceput DETERMINES ( perioada\_rezervare\_OLAP.zi\_din\_luna\_inceput, perioada\_rezervare\_OLAP.luna\_inceput, perioada\_rezervare\_OLAP.zi\_din\_saptamana\_inceput)

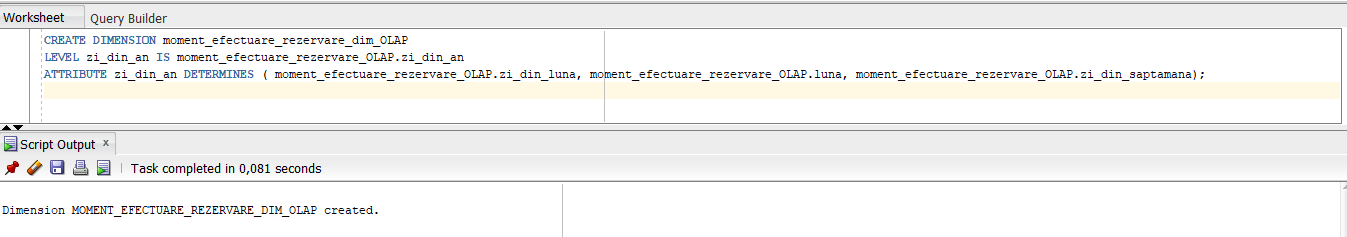
ATTRIBUTE zi\_din\_an\_sfarsit DETERMINES ( perioada\_rezervare\_OLAP.zi\_din\_luna\_sfarsit, perioada\_rezervare\_OLAP.luna\_sfarsit, perioada\_rezervare\_OLAP.zi\_din\_saptamana\_sfarsit);



CREATE DIMENSION moment\_efectuare\_rezervare\_dim\_OLAP

LEVEL zi\_din\_an IS moment\_efectuare\_rezervare\_OLAP.zi\_din\_an

ATTRIBUTE zi\_din\_an DETERMINES ( moment\_efectuare\_rezervare\_OLAP.zi\_din\_luna, moment\_efectuare\_rezervare\_OLAP.luna, moment\_efectuare\_rezervare\_OLAP.zi\_din\_saptamana);



--Ierarhia de mai jos este problematica deoarece exista posibilitatea ca numele a doua localitati din judete diferite sa coincida

CREATE DIMENSION hotel\_dim\_OLAP

LEVEL regiune IS hotel\_OLAP.regiune

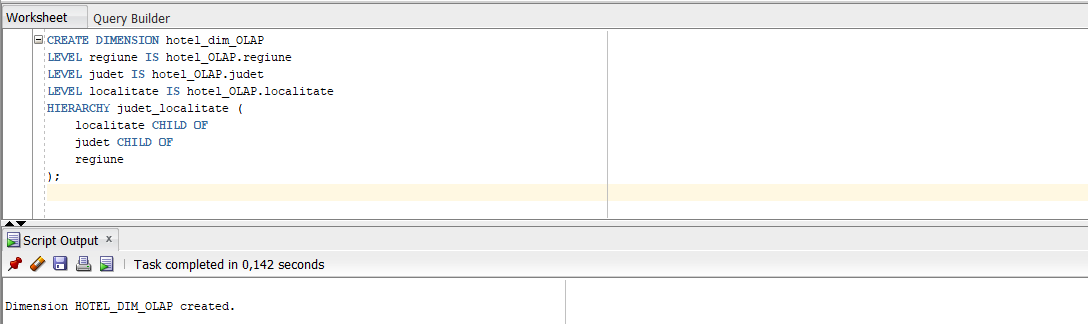
LEVEL judet IS hotel\_OLAP.judet

LEVEL localitate IS hotel\_OLAP.localitate

HIERARCHY judet\_localitate (

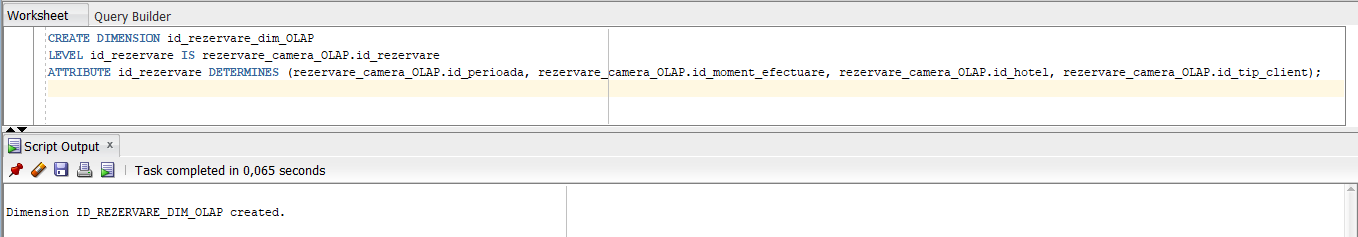
localitate CHILD OF

judet CHILD OF

 regiune);

CREATE DIMENSION id\_rezervare\_dim\_OLAP

LEVEL id\_rezervare IS rezervare\_camera\_OLAP.id\_rezervare

ATTRIBUTE id\_rezervare DETERMINES (rezervare\_camera\_OLAP.id\_perioada\_rezervare, rezervare\_camera\_OLAP.id\_moment\_efectuare\_rezervare, rezervare\_camera\_OLAP.id\_hotel, rezervare\_camera\_OLAP.id\_tip\_client);

8. (1p) Definirea partițiilor; definirea cererilor SQL însoțite de planul de execuție al acestora din care să reiasă ca optimizorul utilizează eficient partițiile.

–pentru hotel

PARTITION BY LIST (nr\_stele)

(PARTITION o\_stea VALUES (1),

PARTITION doua\_stele VALUES (2),

PARTITION trei\_stele VALUES (3),

PARTITION patru\_stele VALUES (4),

PARTITION cinci\_stele VALUES (5));

—pentru rezervare\_camera

PARTITION BY RANGE(pret)

(PARTITION pret\_sub\_1000 VALUES LESS THAN (1000),

PARTITION pret\_sub\_2500 VALUES LESS THAN (2500),

PARTITION pret\_sub\_5000 VALUES LESS THAN (5000),

PARTITION pret\_peste\_5000 VALUES LESS THAN (MAXVALUE));

9. (2p) Optimizarea cererii SQL propusă în etapa de analiză

a. (1p) planul de execuție ales de optimizorul bazat pe cost (explicație etape parcurse)

Se consideră cererea SQL ce afișează numele hotelurilor ce au avut suma rezervărilor cuprinsă între

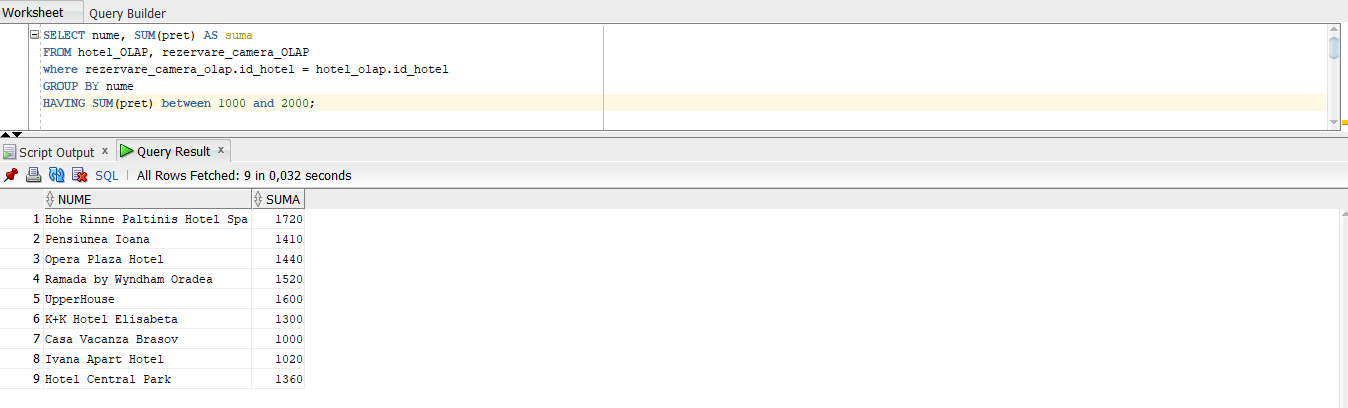
SELECT nume, SUM(pret) AS suma

FROM hotel\_OLAP, rezervare\_camera\_OLAP

where rezervare\_camera\_olap.id\_hotel = hotel\_olap.id\_hotel

GROUP BY nume

HAVING SUM(pret) between 1000 and 2000;



b. (1p) sugestii de optimizare a cererii, specificând planul de execuție obținut

Se poate crea o vizualizare materializată astfel:

CREATE MATERIALIZED VIEW pret\_rezervare\_camera\_mw

BUILD IMMEDIATE

REFRESH FORCE

ENABLE QUERY REWRITE

AS

SELECT nume,

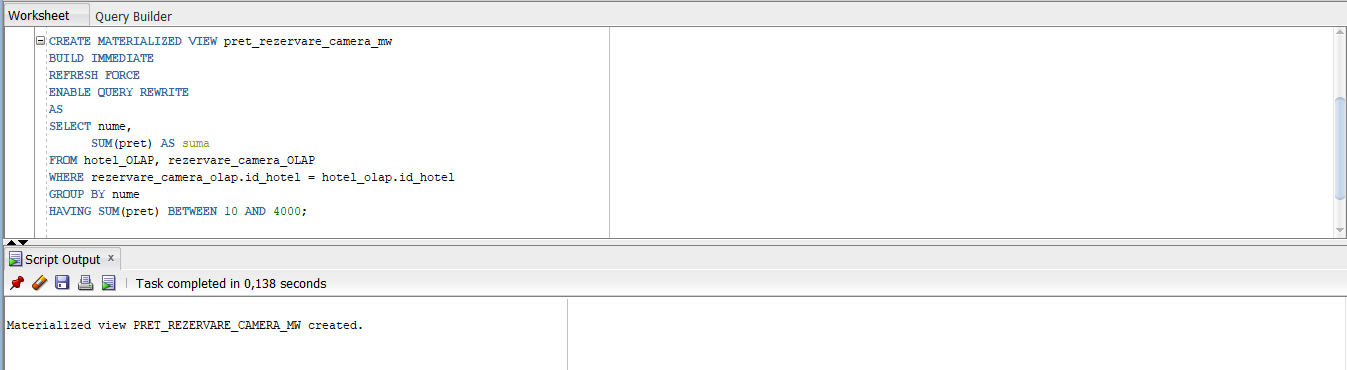
SUM(pret) AS suma

FROM hotel\_OLAP, rezervare\_camera\_OLAP

WHERE rezervare\_camera\_olap.id\_hotel = hotel\_olap.id\_hotel

GROUP BY nume

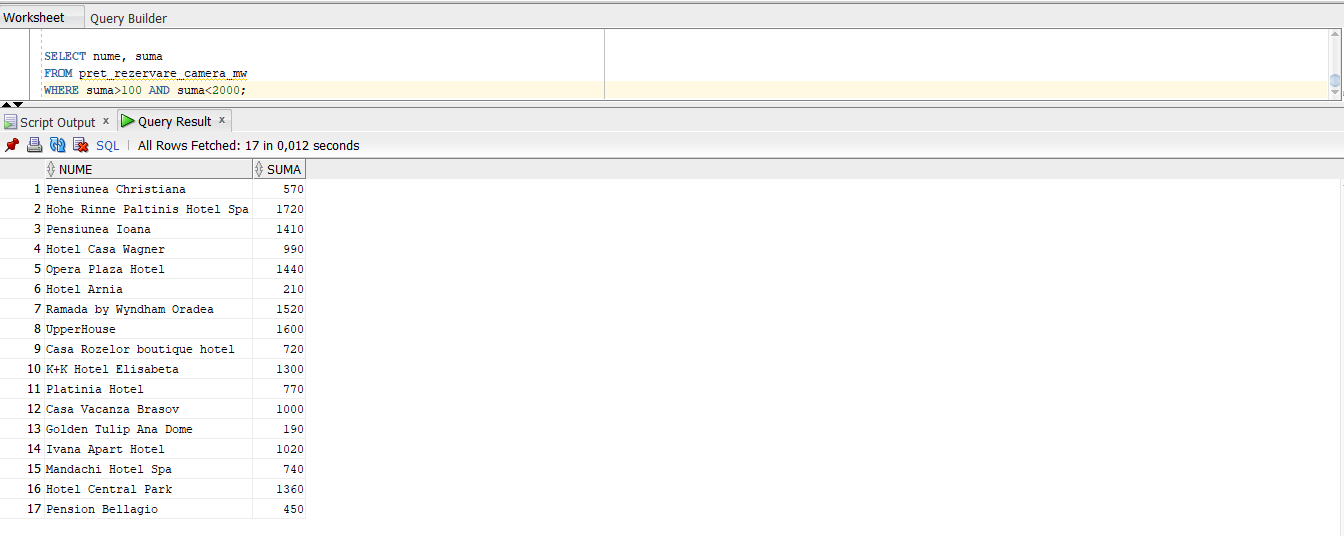
HAVING SUM(pret) BETWEEN 10 AND 4000;

Iar cererea se poate rescrie astfel:

SELECT nume, suma

FROM pret\_rezervare\_camera\_mw

WHERE suma>100 AND suma<2000;



10. (2p) Crearea rapoartelor cu complexitate diferită (la acest nivel vor fi

scripturi SQL, fără reprezentare grafică)

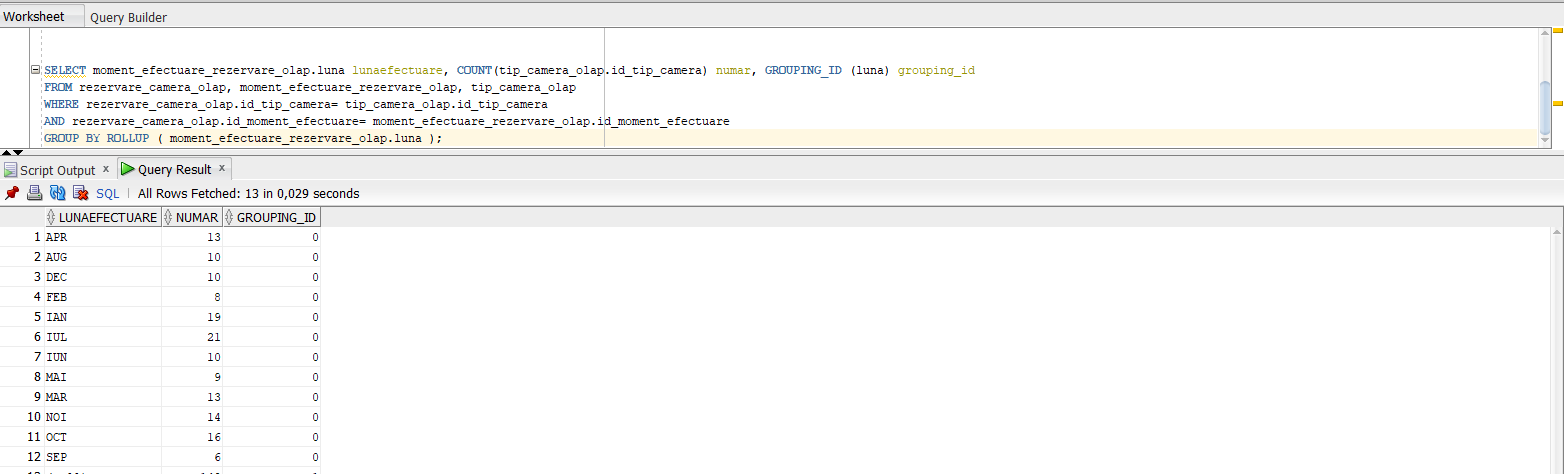
SELECT moment\_efectuare\_rezervare\_olap.luna lunaefectuare, COUNT(tip\_camera\_olap.id\_tip\_camera) numar, GROUPING\_ID (luna) grouping\_id

FROM rezervare\_camera\_olap, moment\_efectuare\_rezervare\_olap, tip\_camera\_olap

WHERE rezervare\_camera\_olap.id\_tip\_camera= tip\_camera\_olap.id\_tip\_camera

AND rezervare\_camera\_olap.id\_moment\_efectuare= moment\_efectuare\_rezervare\_olap.id\_moment\_efectuare

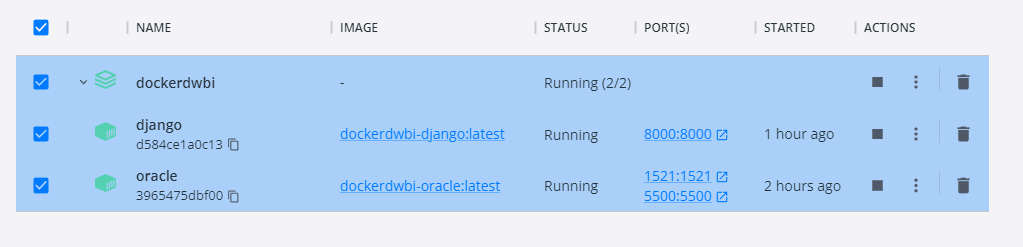
GROUP BY ROLLUP ( moment\_efectuare\_rezervare\_olap.luna );



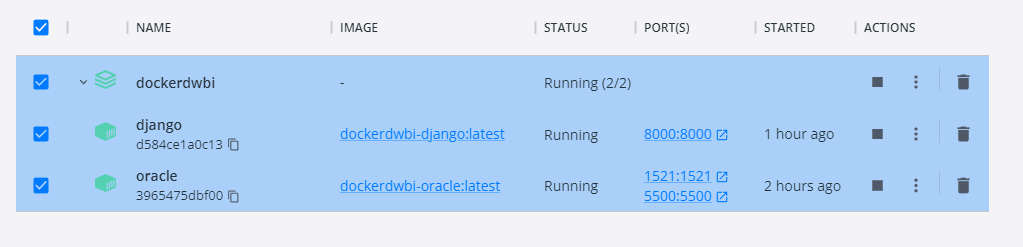
***PROIECT DATAWAREHOUSE***

~ Etapa implementare aplicație ~

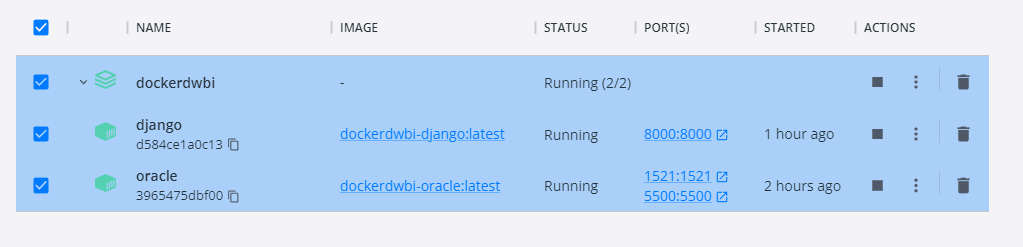
***1. Modul aplicație prin care se introduc și gestionează informații la nivelul bazei de date OLTP***

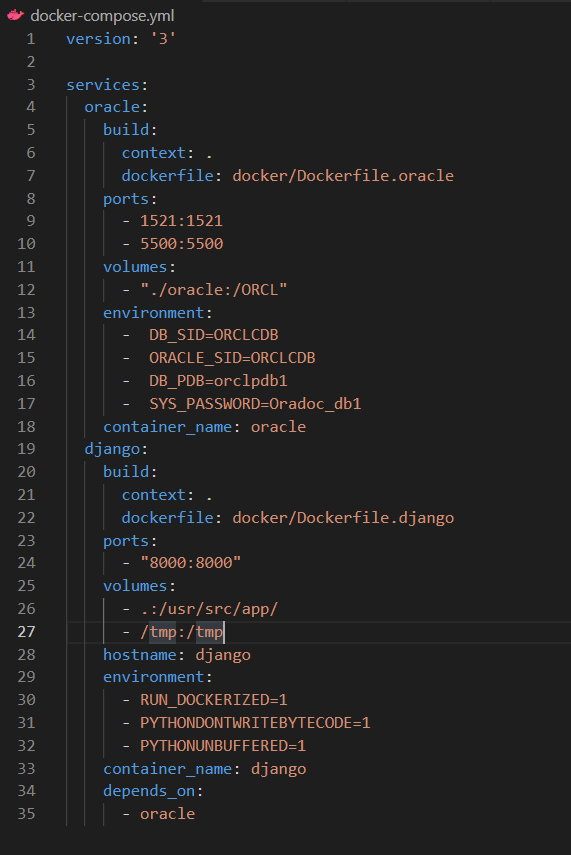
 Pentru implementarea bazei de date am folosit ca sursa baza de date Oracle, versiunea 21c. Baza de date am creat-o direct intr-un container, folosind Docker. Pentru obtinerea imaginea bazei de date de tip oracle, am folosit site-ul oficial Oracle, <https://container-registry.oracle.com/> .

Am decis sa folosim docker pentru a eficientiza procesul de instalare si configurare a bazei de date, care este utilizata de pe mai multe dispozitive.

 Pentru backend & frontend , am folosit Python, framework-ul Django, prin care sub forma unei aplicatii web putem altera si observa datele. Pentru managererea containerul, am folosit Docker Desktop, unde putem observa containerele ruland. La momentul crearii containerelor, sunt rulate scripturi si fisiere de configurare care creaza si populeaza bazele de date,respectiv tabele asociate OTLP is OLAP.

Pentru crearea containerelor, am folosit docker compose, unde avem

cele 2 servicii definite:



In script-urile de intializare, pe

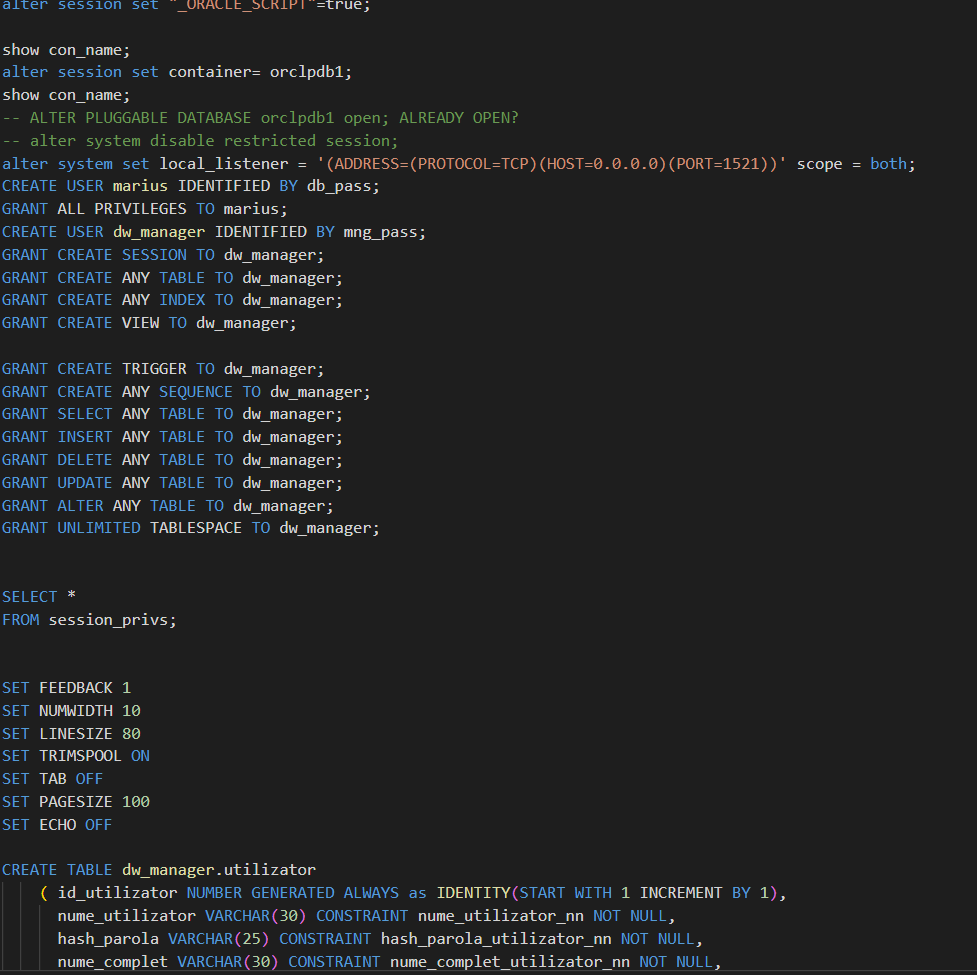
langa anumite conditii de

conectivitate,vizibilitate a bazei de

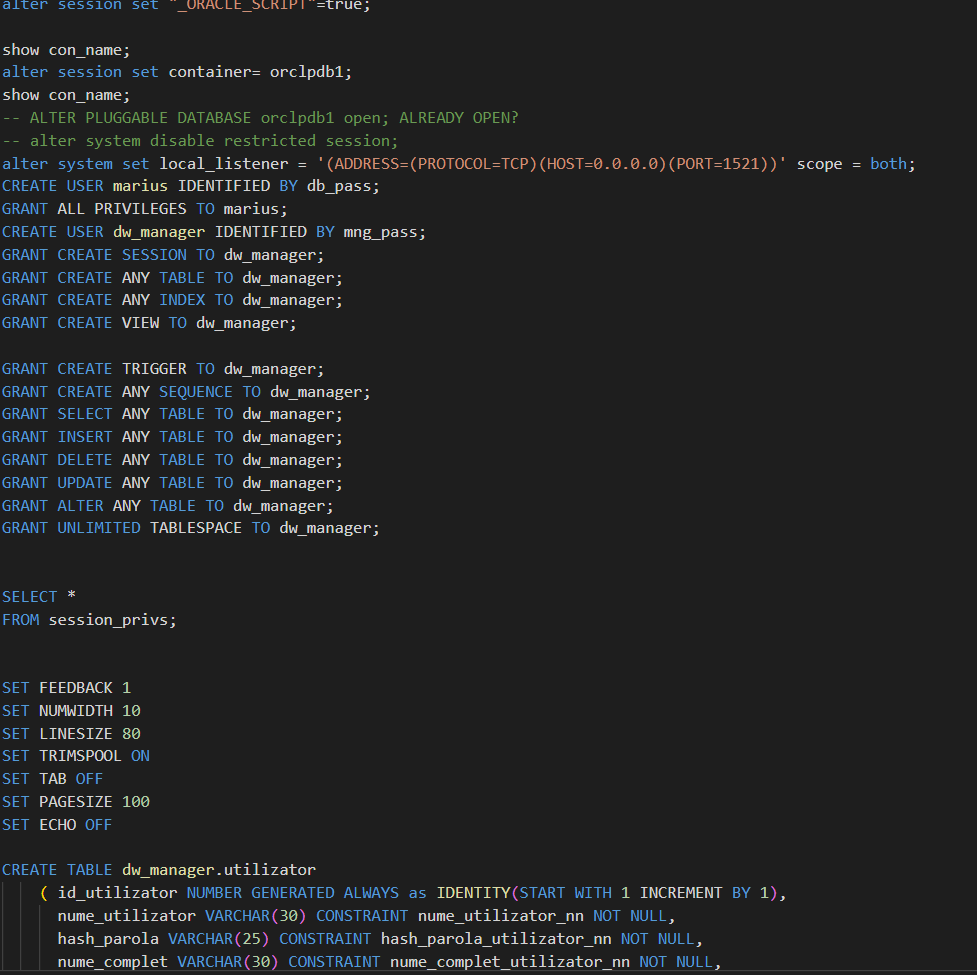
date intr-un sistem containerizat,

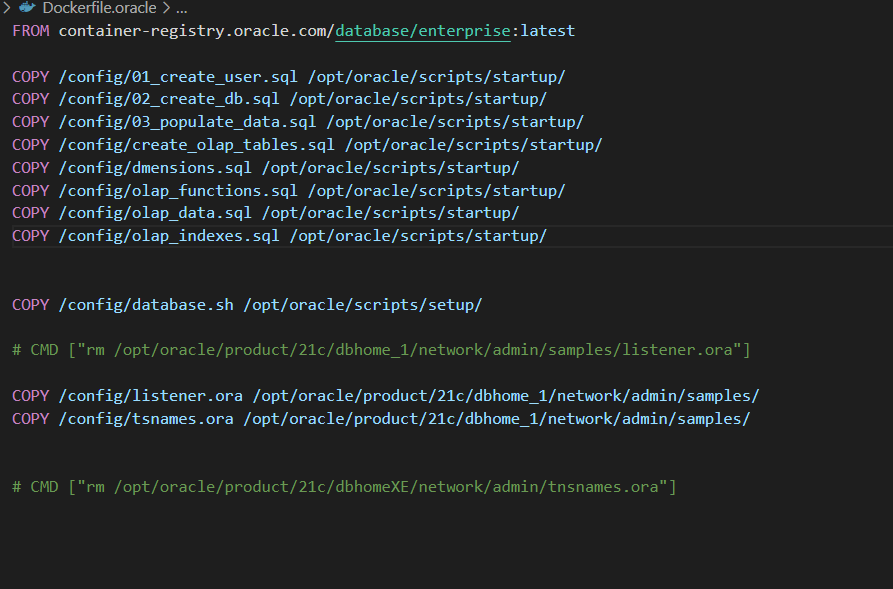
incepem prin a crea userii, si

tabelele OLTP.

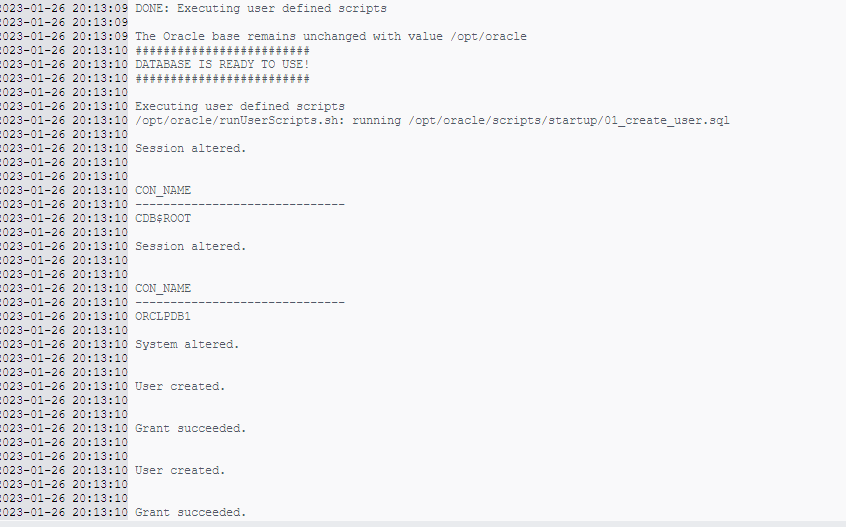


Script-urile de intializare, configurare sunt copiate in foldere predefinite de Oracle in scopul rularii acestora la startup.

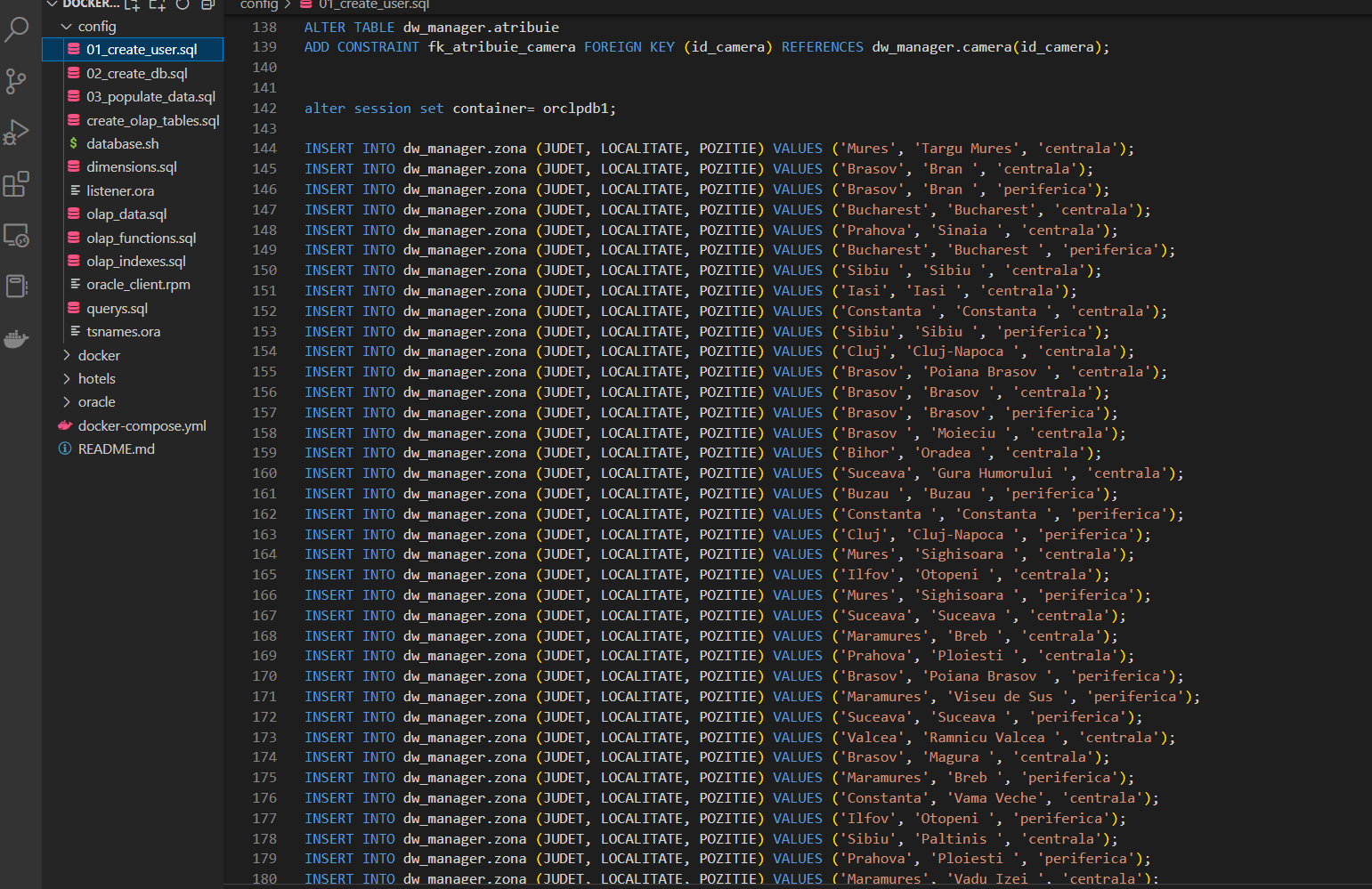


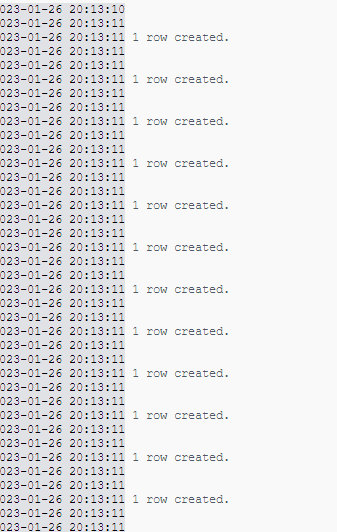
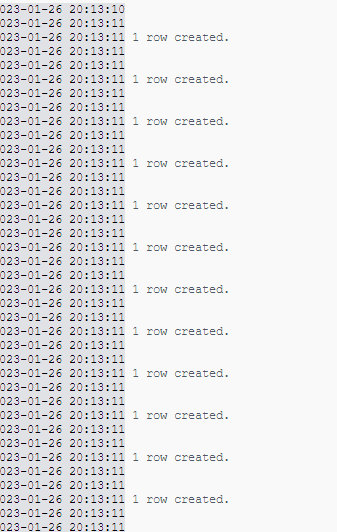


Se putea obtine acelasi rezultat folosind un volum, dar pentru vizibilitate, am copiat script-urile in folderele predefinite.



Dupa crearea tabelelor, si a userilor asociati, se insereaza datele, tot din cadrul fisierilor care ruleaza la startup:





Tabele pentru baza de date OLAP se populeaza din cadrul urmatorului script:

INSERT INTO perioada\_rezervare\_OLAP(zi\_din\_luna\_inceput, luna\_inceput, an\_inceput, zi\_din\_saptamana\_inceput, zi\_din\_an\_inceput, zi\_din\_luna\_sfarsit, luna\_sfarsit, an\_sfarsit, zi\_din\_saptamana\_sfarsit, zi\_din\_an\_sfarsit, durata\_in\_zile)

SELECT DISTINCT TO\_NUMBER(TO\_CHAR(data\_inceput,'DD')), TO\_CHAR(data\_inceput,'MON'), TO\_NUMBER(TO\_CHAR(data\_inceput,'YYYY')),

TO\_CHAR(data\_inceput,'DY'),TO\_NUMBER(TO\_CHAR(data\_inceput,'DDD')),

TO\_NUMBER(TO\_CHAR(data\_sfarsit,'DD')), TO\_CHAR(data\_sfarsit,'MON'), TO\_NUMBER(TO\_CHAR(data\_sfarsit,'YYYY')),

TO\_CHAR(data\_sfarsit,'DY'),TO\_NUMBER(TO\_CHAR(data\_sfarsit,'DDD')),

data\_sfarsit - data\_inceput

FROM rezervare; COMMIT;

INSERT INTO moment\_efectuare\_rezervare\_OLAP(zi\_din\_luna, luna, an, zi\_din\_saptamana, zi\_din\_an)

SELECT DISTINCT TO\_NUMBER(TO\_CHAR(data\_efectuarii,'DD')), TO\_CHAR(data\_efectuarii,'MON'),

TO\_NUMBER(TO\_CHAR(data\_efectuarii,'YYYY')),TO\_CHAR(data\_efectuarii,'DY'),TO\_NUMBER(TO\_CHAR(data\_efectuarii,'DDD'))--,

--TO\_NUMBER(TO\_CHAR(data\_efectuarii,'HH')) + CASE WHEN TO\_NUMBER(TO\_CHAR(data\_efectuarii,'MI')) > 29 THEN 1 ELSE 0 END AS ora\_aprox

FROM rezervare; COMMIT;

INSERT INTO hotel\_OLAP(id\_hotel, nume, regiune, judet, localitate, pozitie, nr\_stele, are\_mic\_dejun\_inclus)

SELECT id\_hotel, nume, regiune, judet, localitate, pozitie, nr\_stele, are\_mic\_dejun\_inclus

FROM hotel JOIN zona

USING(id\_zona); COMMIT;

INSERT INTO tip\_client\_OLAP(varsta,gen,stare\_civila)

SELECT DISTINCT FLOOR(MONTHS\_BETWEEN(SYSDATE,data\_nasterii)/12) AS varsta, gen, stare\_civila

FROM utilizator; COMMIT;

INSERT INTO tip\_camera\_OLAP(nr\_paturi\_duble, nr\_paturi\_simple,are\_terasa,are\_televizor)

SELECT DISTINCT nr\_paturi\_duble, nr\_paturi\_simple,are\_terasa,are\_televizor

FROM camera; COMMIT;

INSERT INTO rezervare\_camera\_OLAP (id\_rezervare,id\_hotel,id\_perioada,id\_moment\_efectuare,id\_tip\_camera,id\_tip\_client,pret)

SELECT DISTINCT id\_rezervare,id\_hotel,

gaseste\_id\_perioada\_OLAP(TO\_NUMBER(TO\_CHAR(data\_inceput,'DDD')),

TO\_NUMBER(TO\_CHAR(data\_inceput,'YYYY')),

TO\_NUMBER(TO\_CHAR(data\_sfarsit,'DDD')),

TO\_NUMBER(TO\_CHAR(data\_sfarsit,'YYYY'))) AS id\_perioada,

gaseste\_id\_moment\_efectuare\_OLAP(TO\_NUMBER(TO\_CHAR(data\_efectuarii,'DDD')),

TO\_NUMBER(TO\_CHAR(data\_efectuarii,'YYYY'))) AS id\_moment\_efectuare,

gaseste\_id\_tip\_camera\_OLAP(camera.nr\_paturi\_duble,camera.nr\_paturi\_simple,

camera.are\_terasa,camera.are\_televizor) AS id\_tip\_camera,

gaseste\_id\_tip\_client\_OLAP(calculeaza\_varsta(utilizator.data\_nasterii),utilizator.gen,

utilizator.stare\_civila) AS id\_tip\_client,

camera.pret\_per\_noapte \* (rezervare.data\_sfarsit - rezervare.data\_inceput)

AS pret

FROM utilizator JOIN rezervare

ON utilizator.id\_utilizator=rezervare.id\_client

JOIN rezervare\_camera

USING(id\_rezervare)

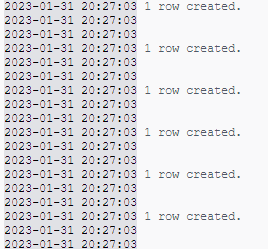
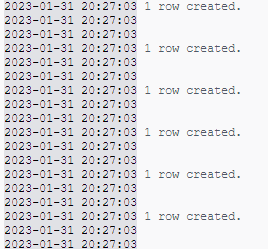
JOIN camera

USING(id\_camera)

JOIN hotel

USING(id\_hotel); COMMIT;

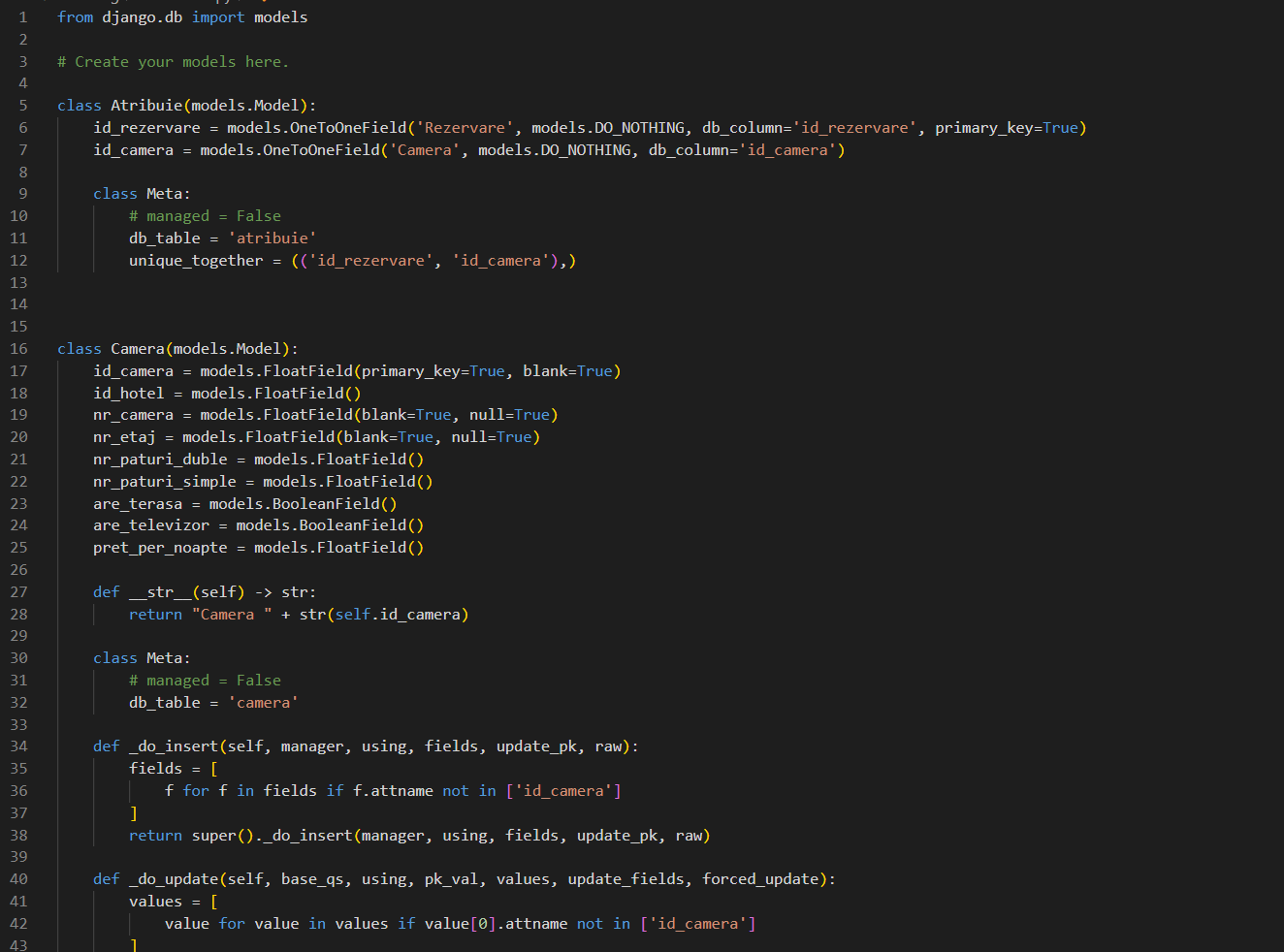
Dupa rularea script-ului, observam in docker aceleasi log-uri:



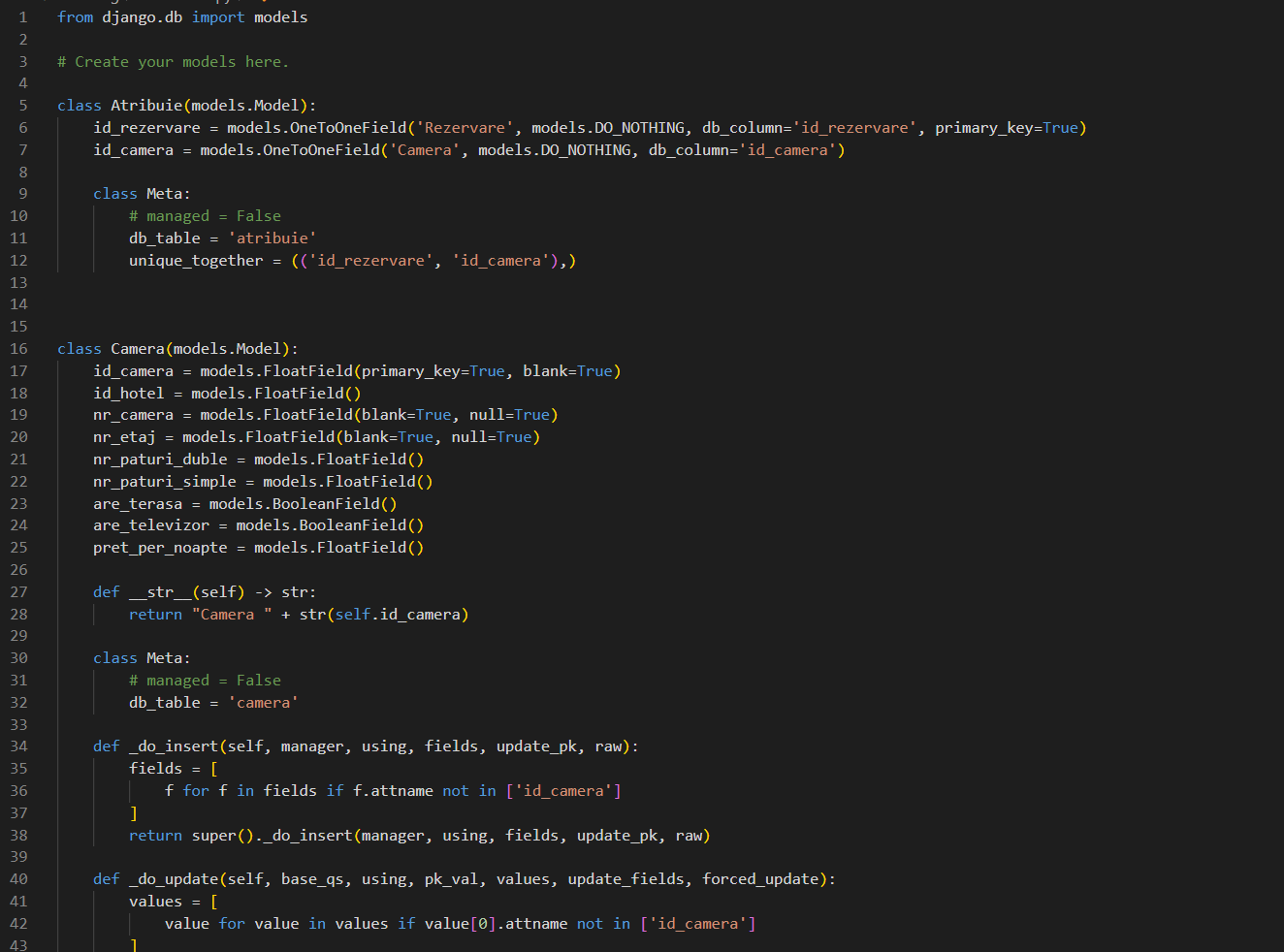
***2. Posibilitatea de vizualizare a datelor introduse/actualizate la nivelul bazei de date OLTP și posibilitatea de a verifica propagarea acestor operații asupra datelor din baza de date depozit***

# Aplicatie – Frontend,

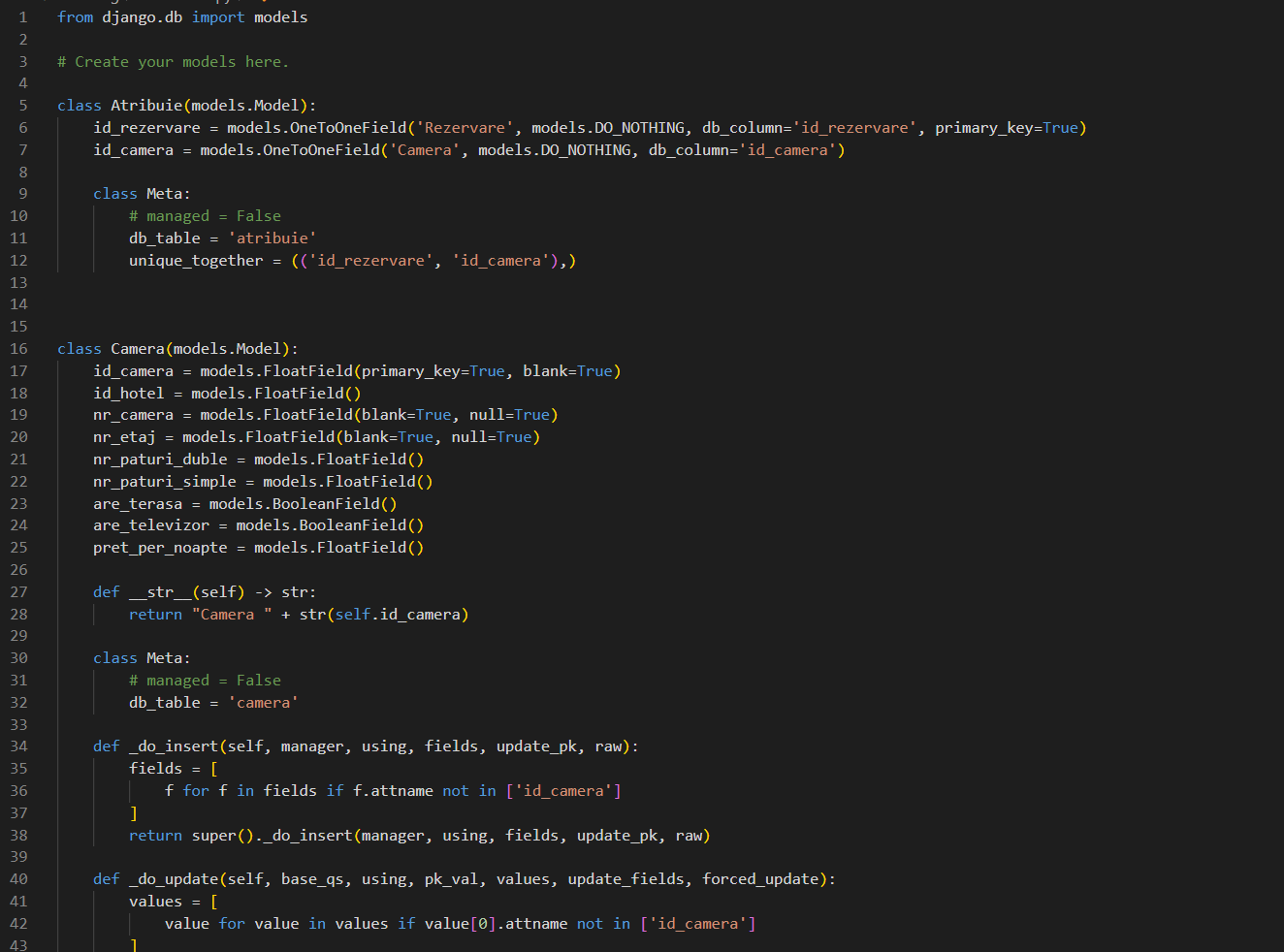
(Back end) Datele de tip OLTP au fost proiectate in back end, sub forma unor modele, folosind ORM-ul django. Comanda *python manage.py inspectdb* genereaza aceste tabele, care trebuie putin customizate pentru a functiona in diferite scenarii.

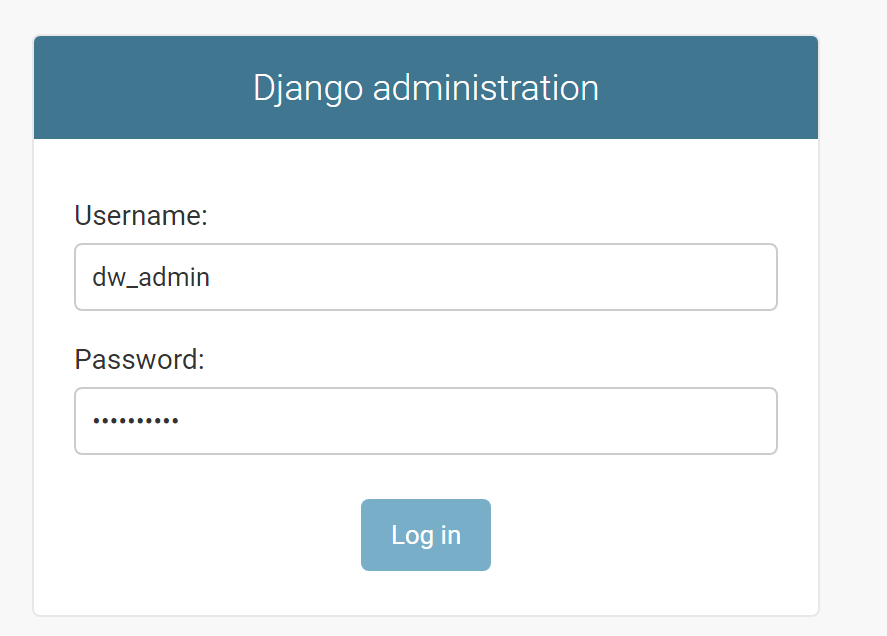


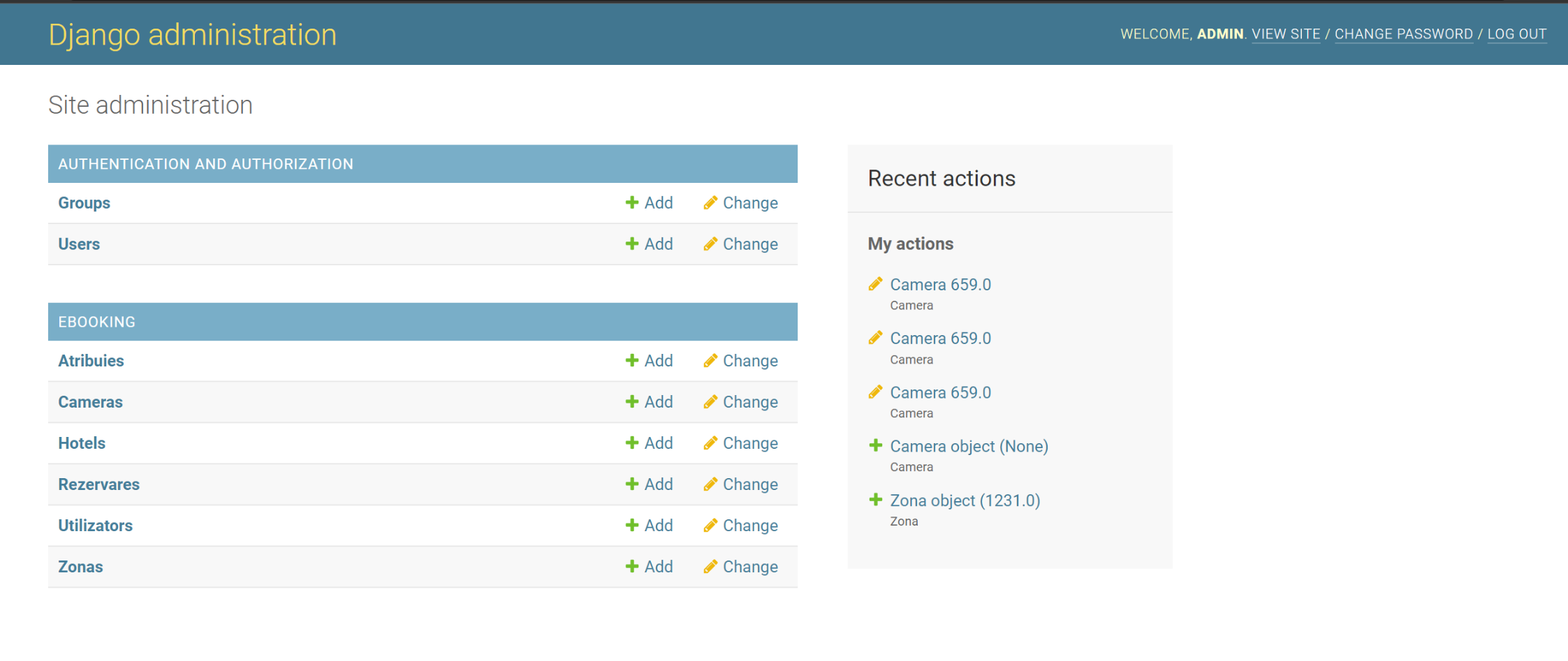
Aceste modele sunt vizibile in interfata web a aplicatiei django, printr-o pagina de management alocata administratorului bazei de date.

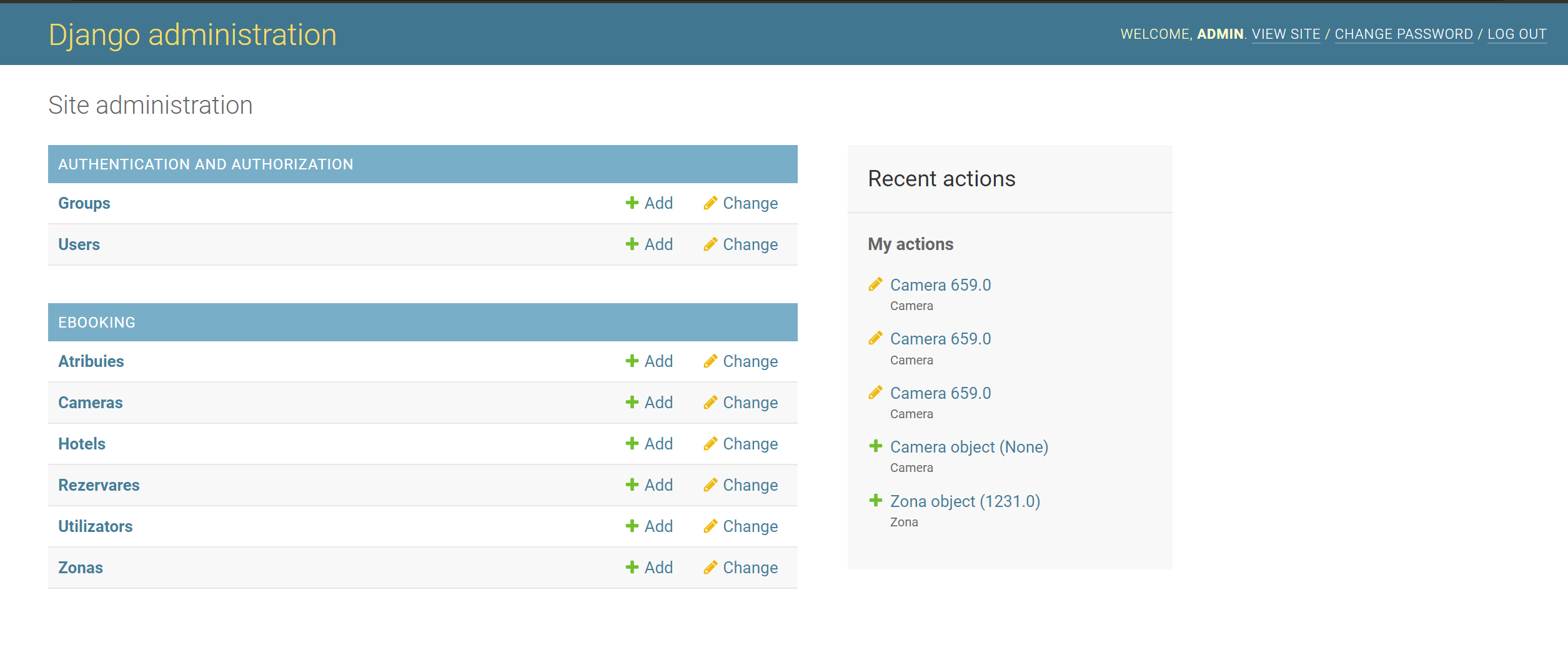
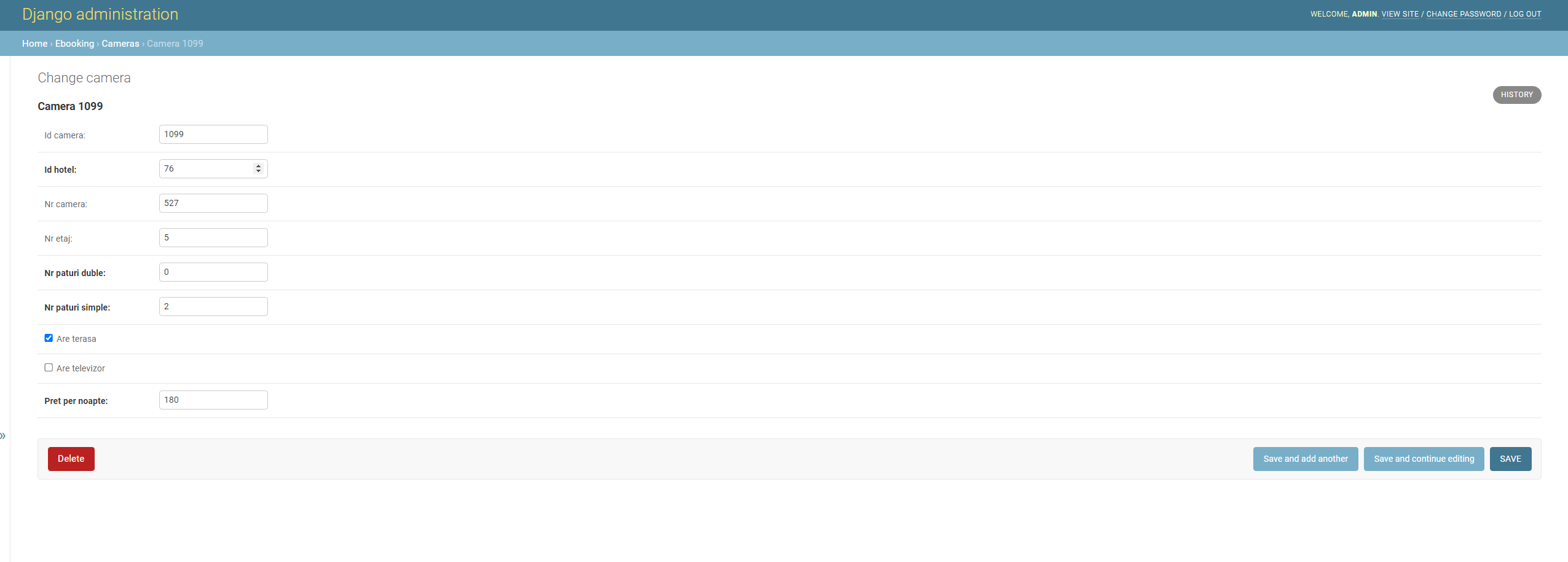


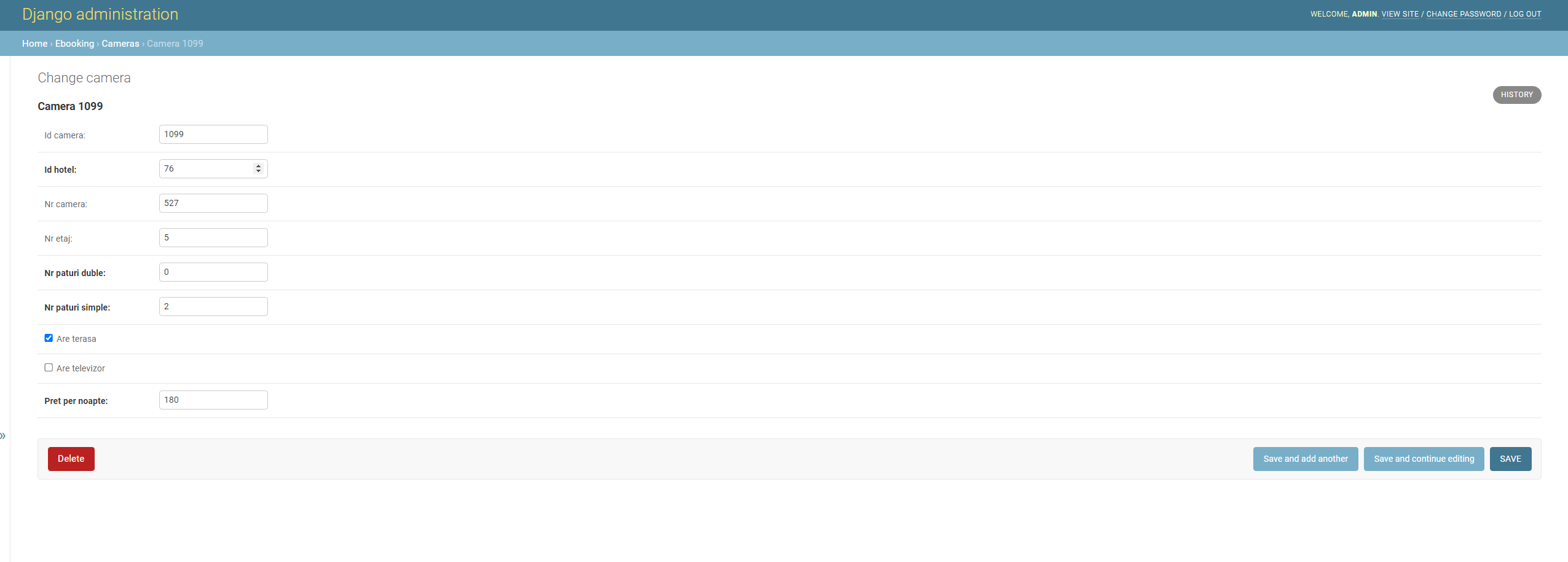
Dupa introducerea datelor de logare definite in fisierele de config ale oracle, admin-ul va putea vedea toate datele de tip OLTP din cadrul bazei de date .











***3. Rapoartele grafice asociate cererilor definite în etapele anterioare (punctul 10).***

Pe pagina principala, <http://localhost:8000/>, ne sunt prezentate graficele corespunzatoare script-urilor sql create în continuare.

**Crearea rapoartelor cu complexitate diferită**

Pentru aceste rapoarte au fost create grafice de tip chart in front-end aplicatiei Web.

1\*  - Cea mai scumpa camera per noapte  din romania

select max(pret\_per\_noapte) from

(select hotel.nume,camera.pret\_per\_noapte from hotel

INNER JOIN camera on camera.id\_hotel = hotel.id\_hotel

order by hotel.nume)

2\* Cea mai ieftina camera dintr-o zona centrala

select \* from (select hotel.nume, MIN(pret\_per\_noapte) from camera

INNER JOIN hotel

on hotel.id\_hotel=camera.id\_hotel

INNER JOIN zona on zona.id\_zona = hotel.id\_zona

WHERE zona.pozitie = 'centrala'

GROUP BY hotel.nume)

where rownum = 1

3\* Hotelul cu cele mai multe rezervari efectuate

select \* from(

SELECT MAX(rezervari) as mres, numes  FROM (SELECT  hotel.nume as numes,COUNT(atribuie.id\_rezervare) as rezervari from atribuie

inner join camera on camera.id\_camera=atribuie.id\_camera

inner join hotel on hotel.id\_hotel = camera.id\_hotel

GROUP BY hotel.nume)

group by numes

ORDER BY mres DESC)

where rownum=1;

4\* Cele mai scumpe 5 hoteluri

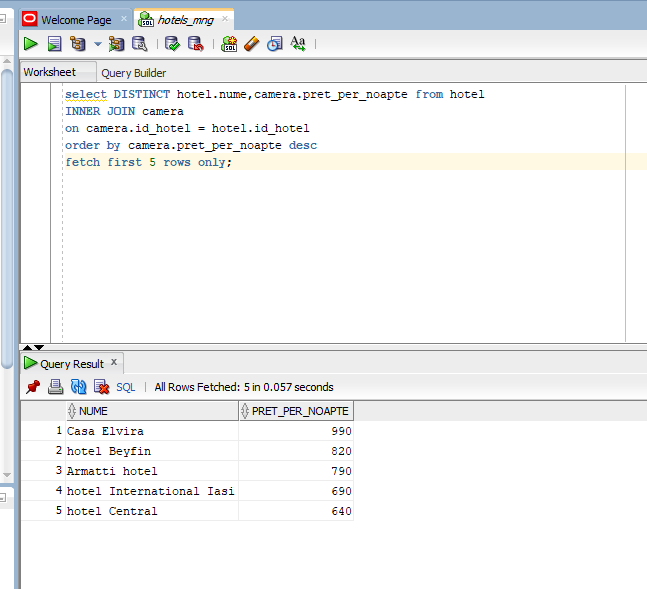
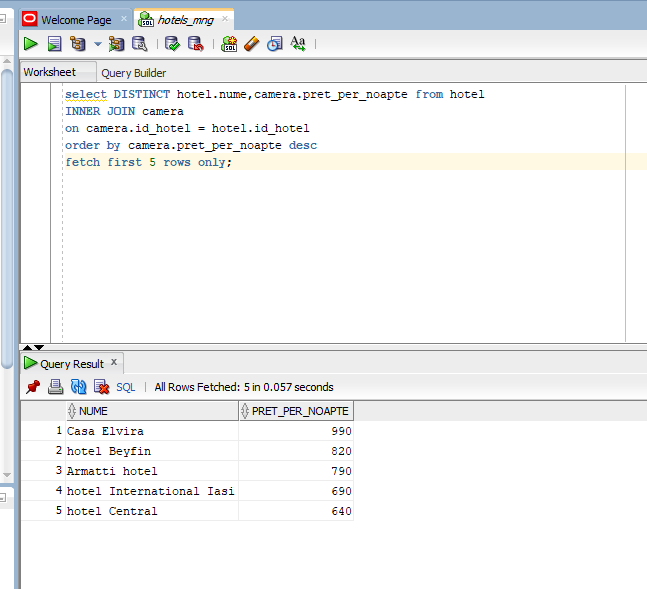
select DISTINCT hotel.nume,camera.pret\_per\_noapte from hotel

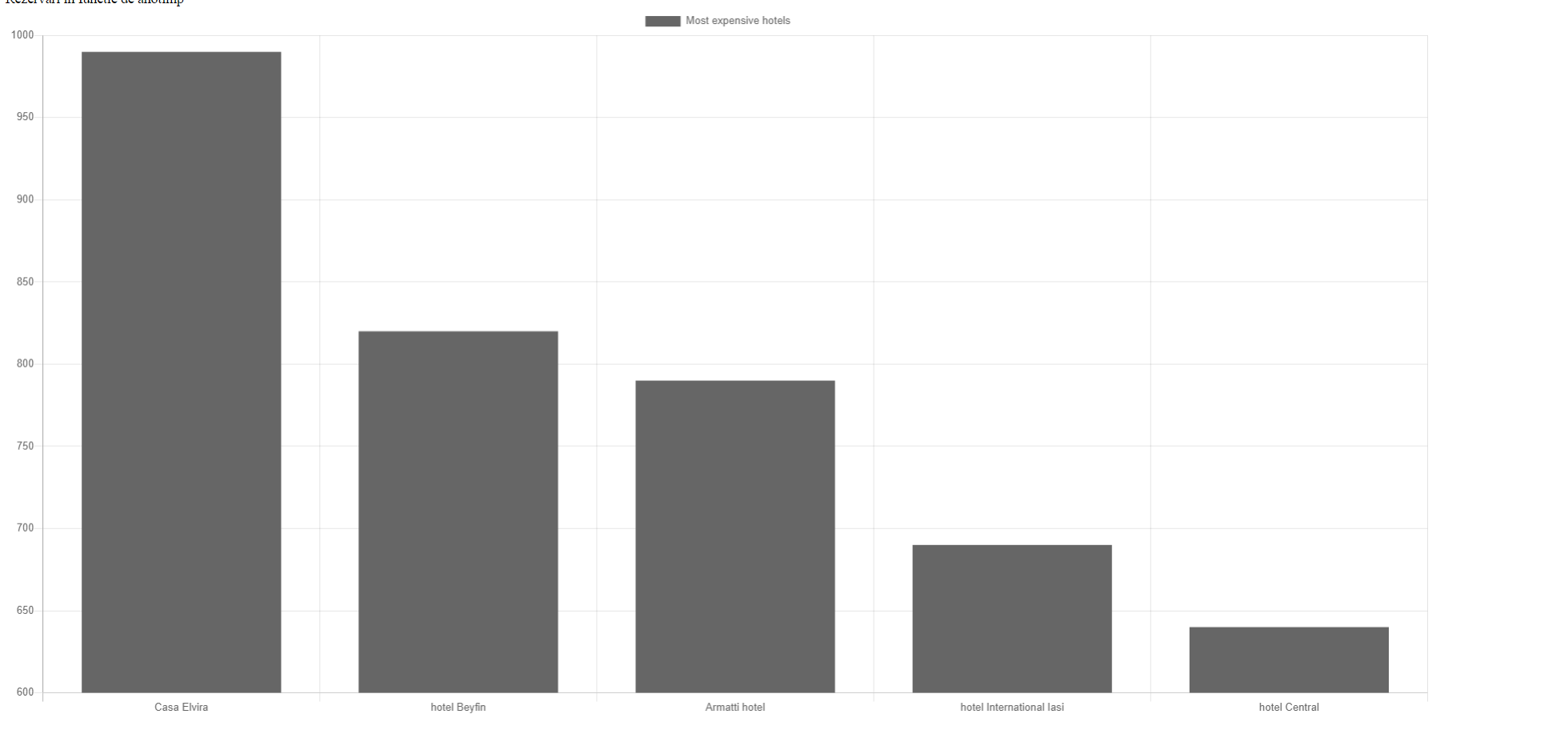
INNER JOIN camera

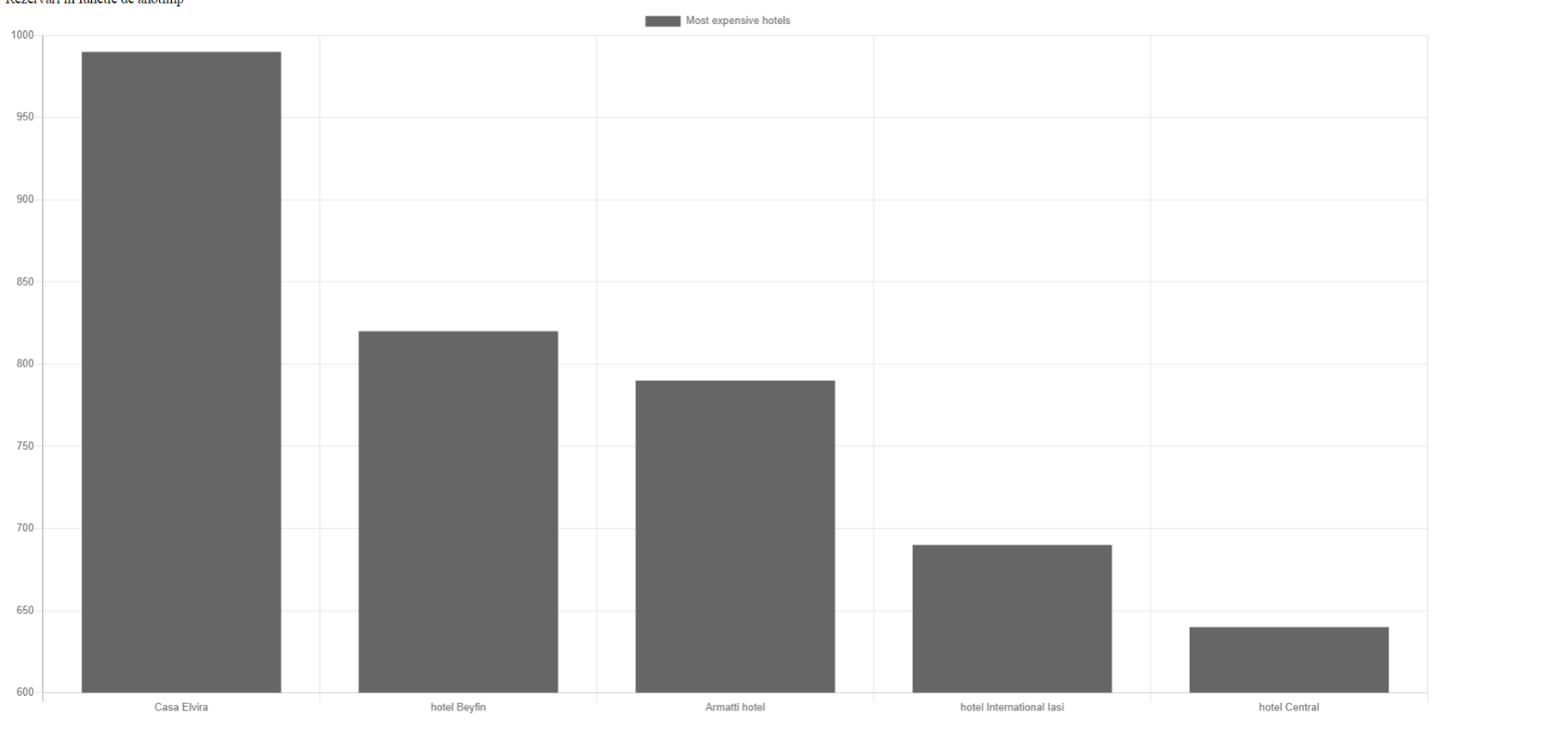
on camera.id\_hotel = hotel.id\_hotel

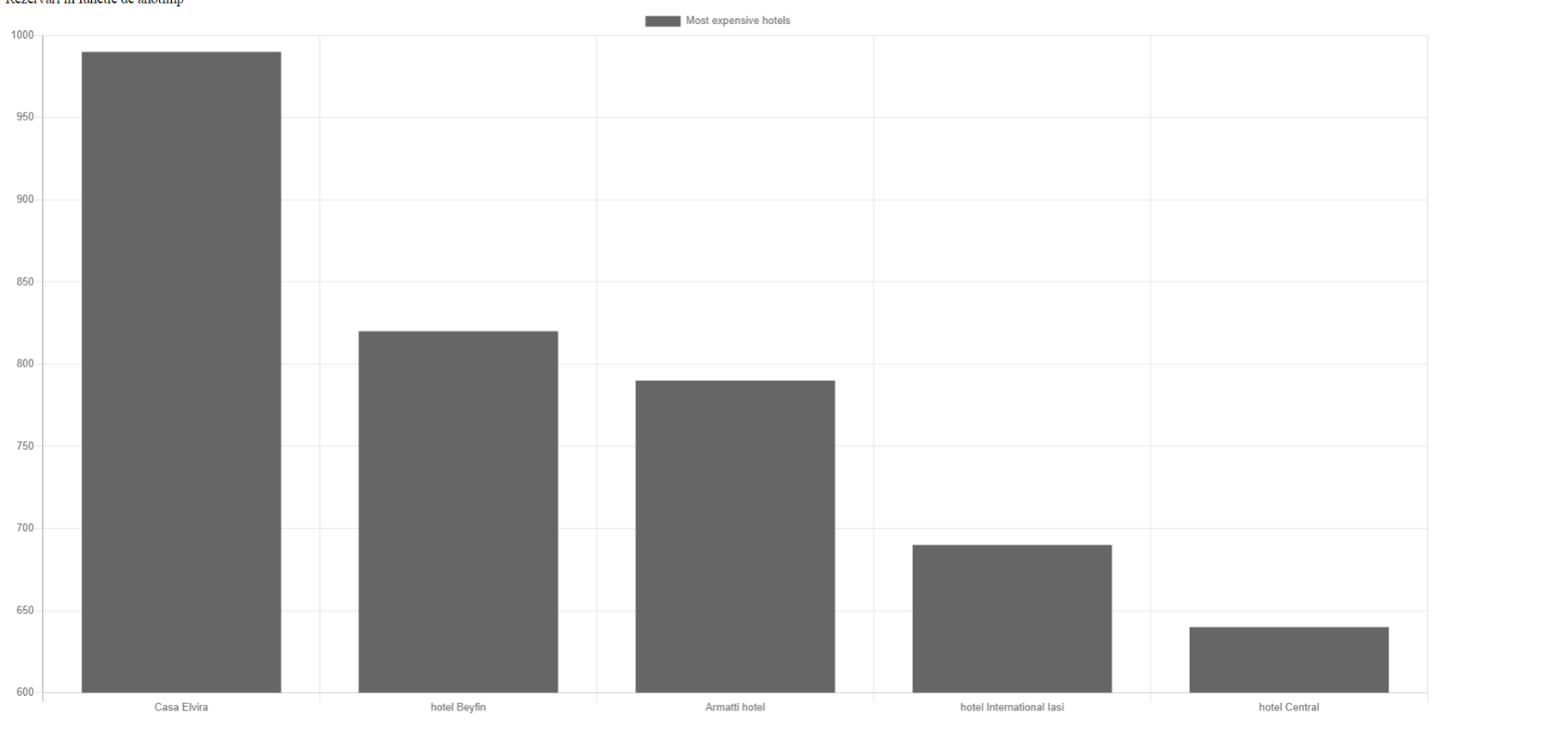
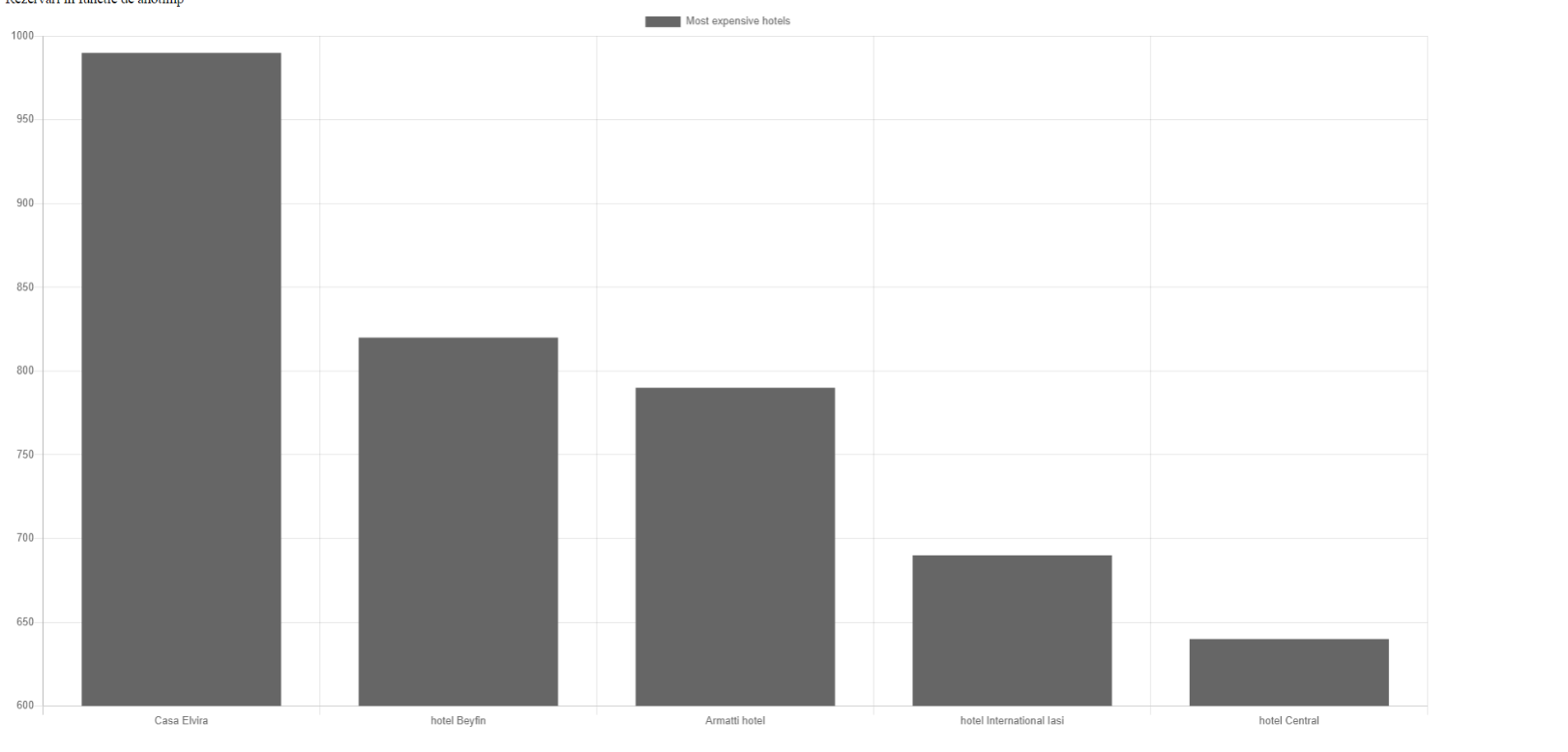
order by camera.pret\_per\_noapte desc

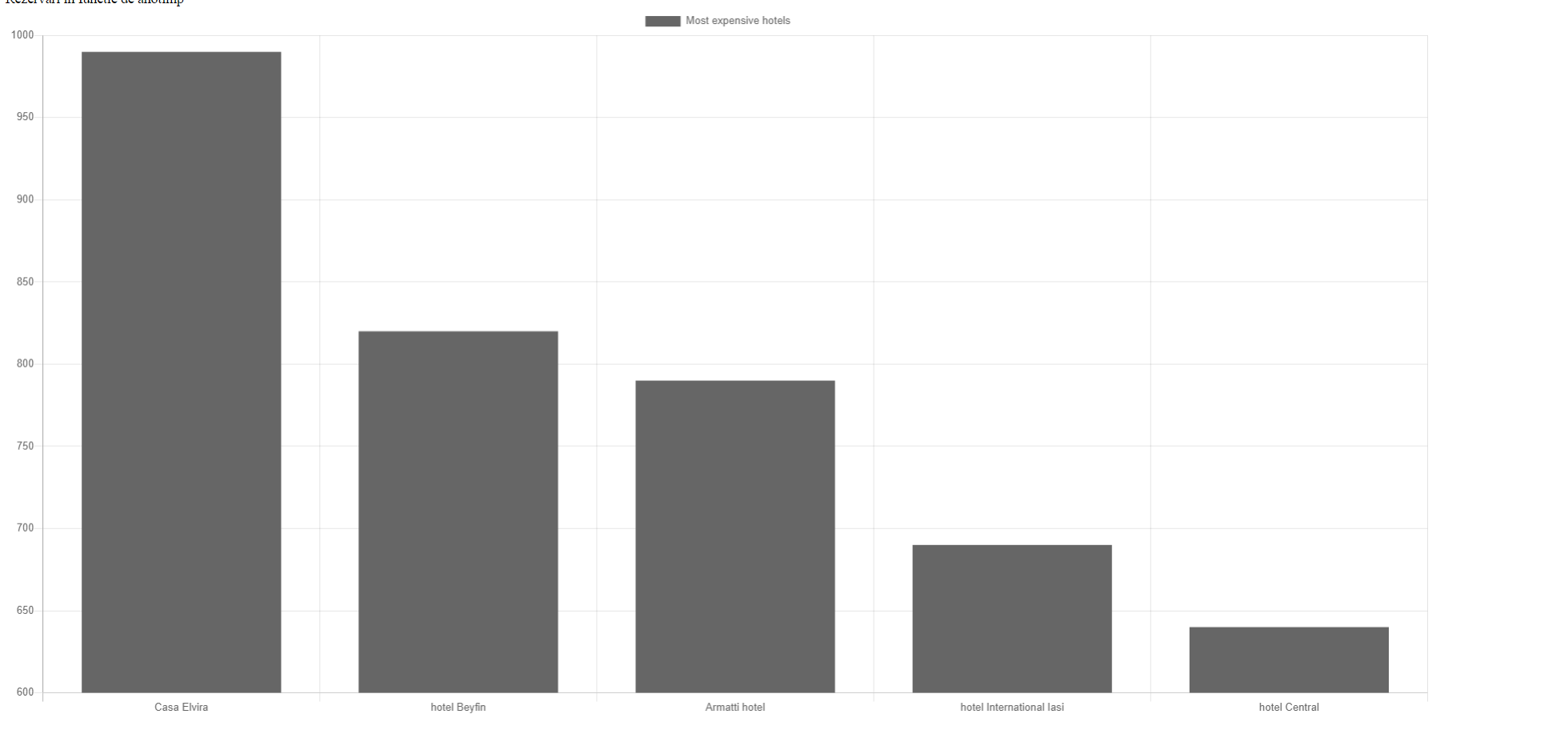
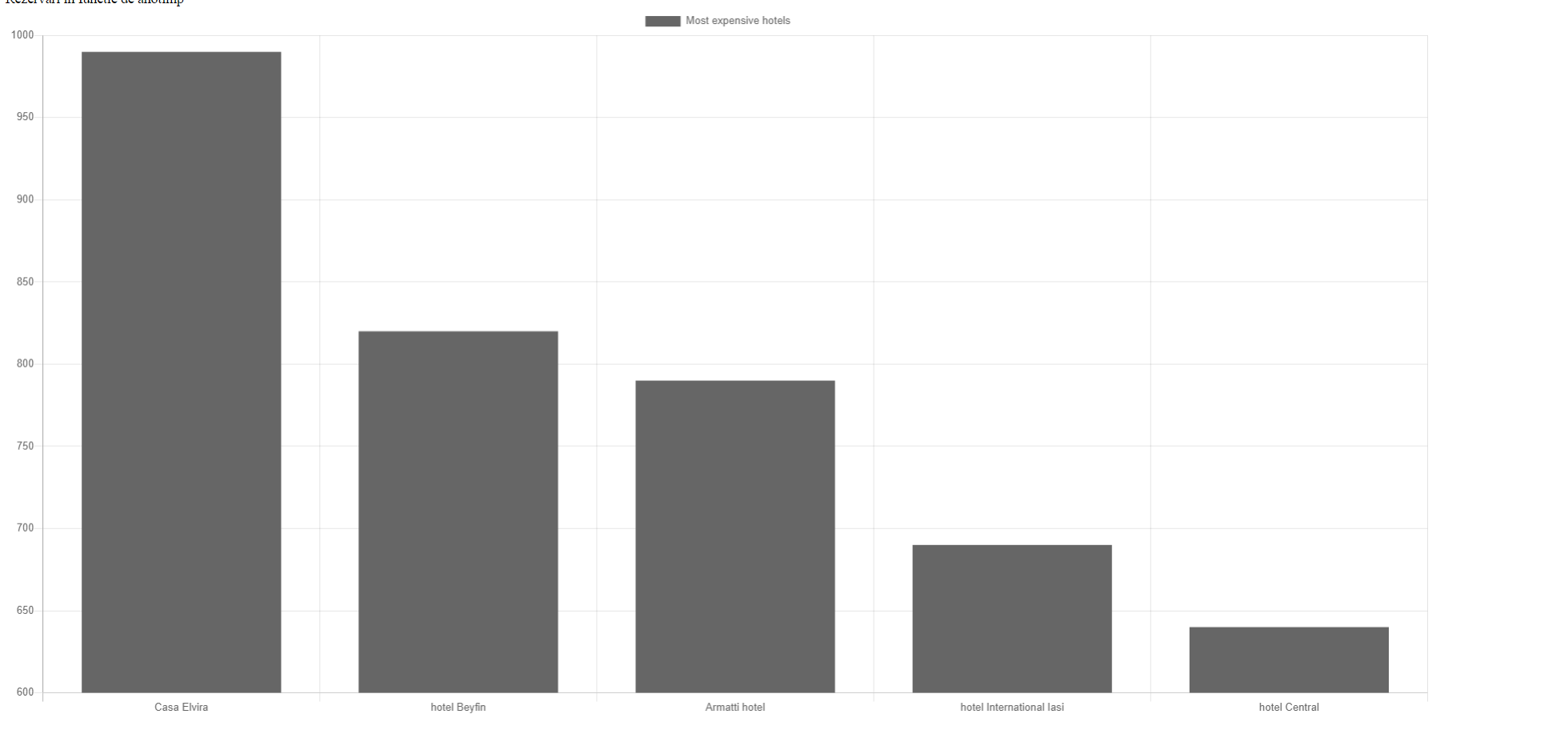
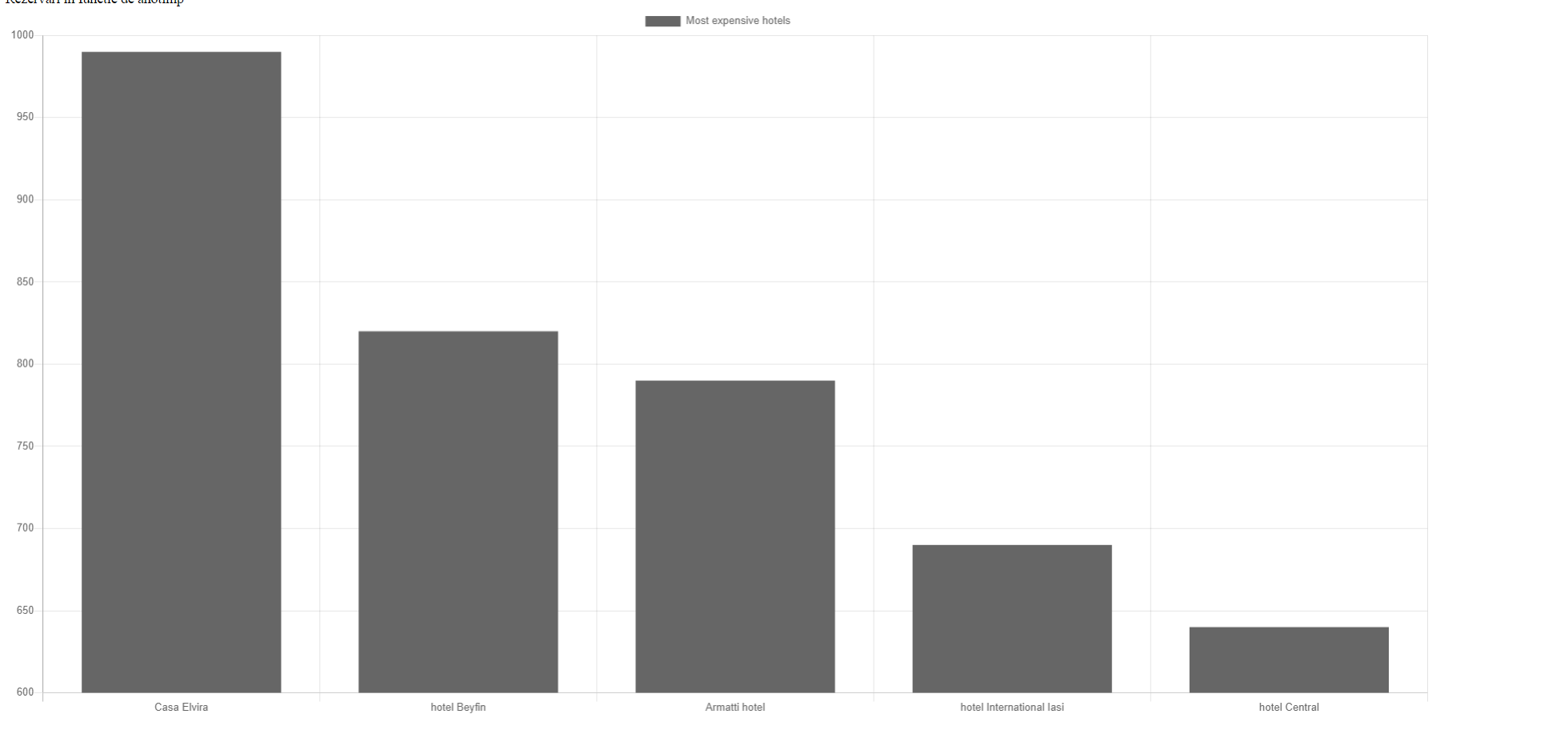
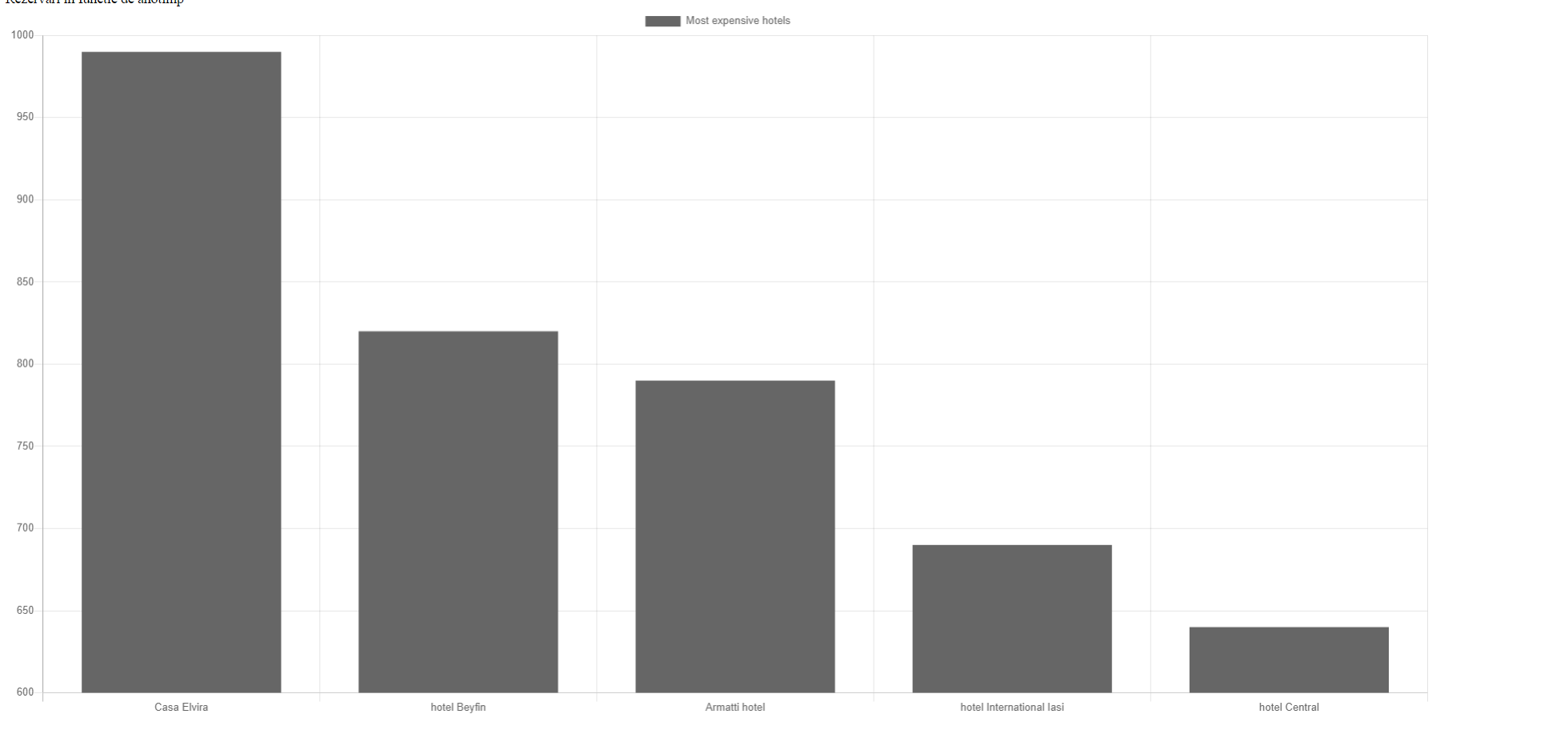
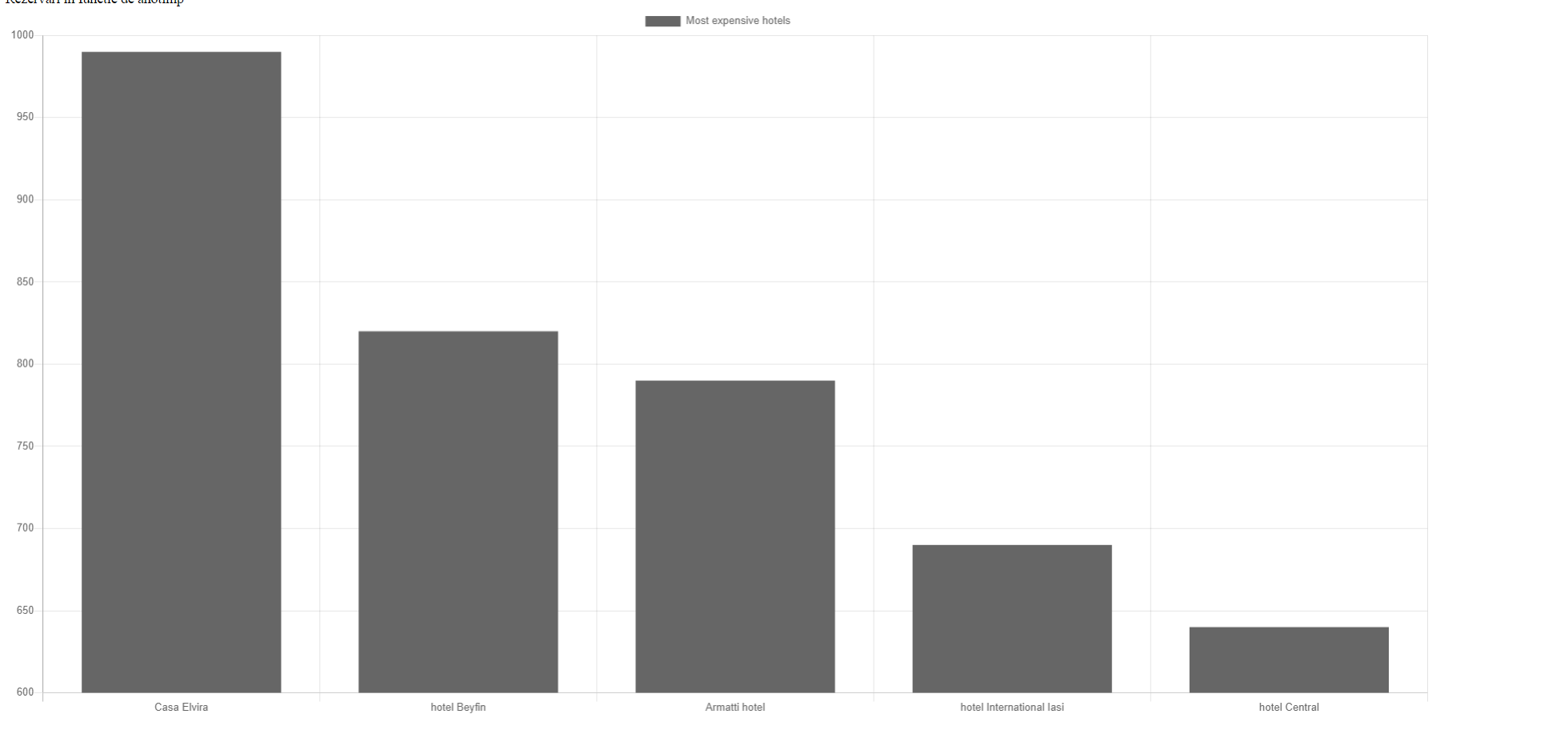
fetch first 5 rows only;





 ***Cele mai scumpe hoteluri 5 din Romania***





5\* Rezervari in functie de luna anului

SELECT COUNT(\*) FROM REZERVARE

WHERE EXTRACT(MONTH FROM data\_inceput) IN (1,4)

SELECT COUNT(\*) FROM REZERVARE

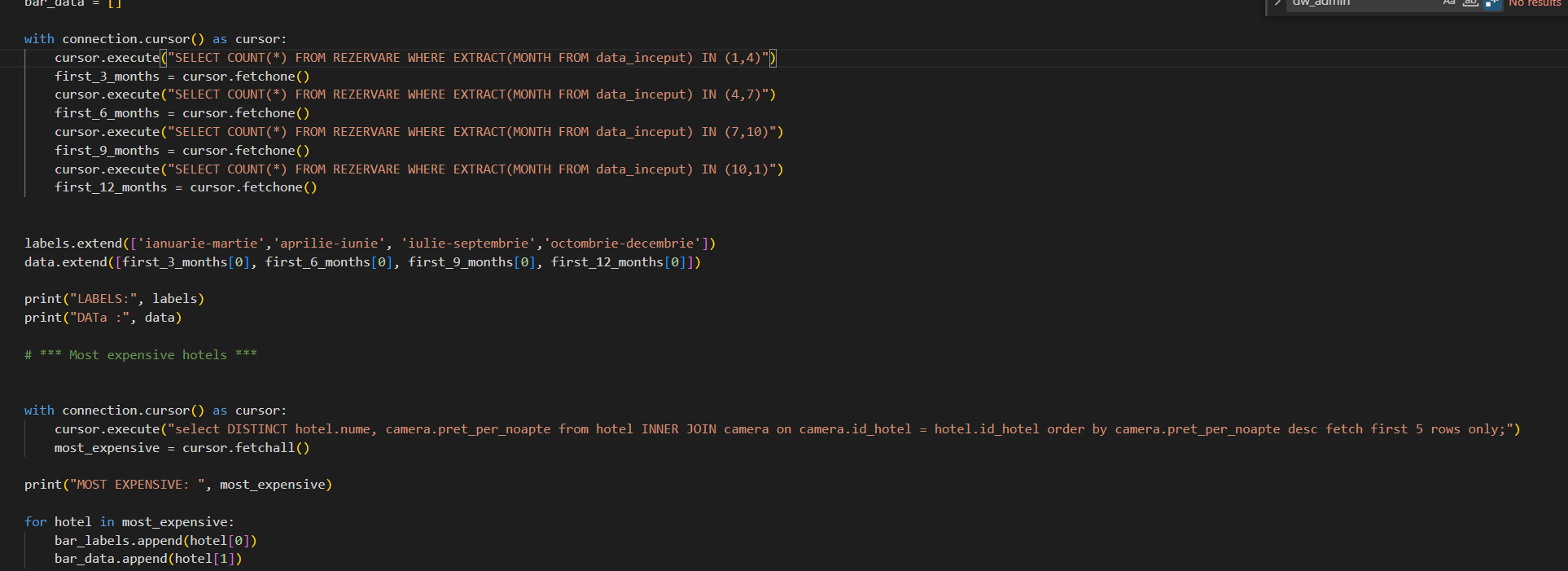
WHERE EXTRACT(MONTH FROM data\_inceput) IN (4, 7)

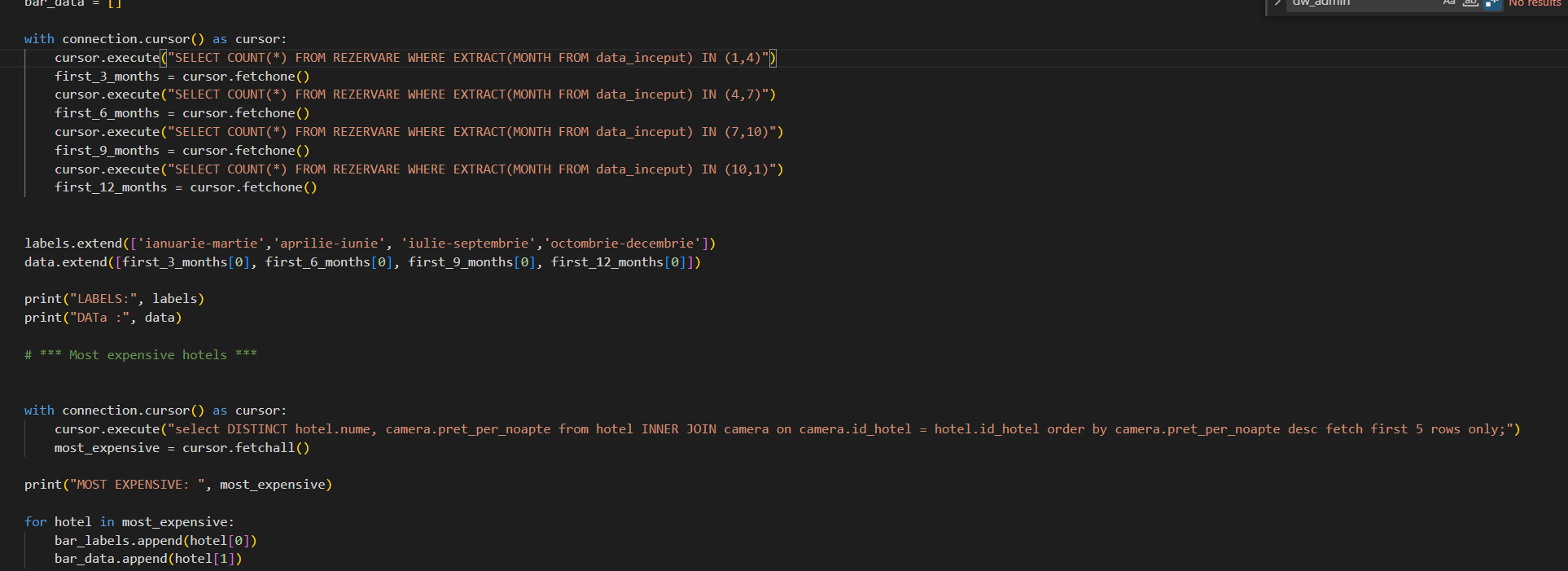
SELECT COUNT(\*) FROM REZERVARE

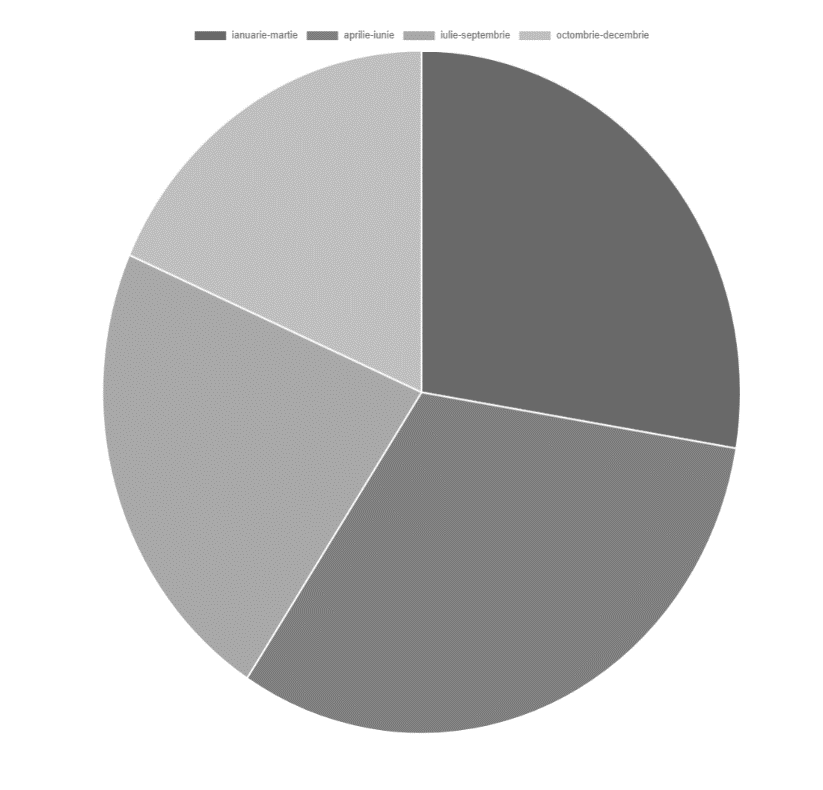
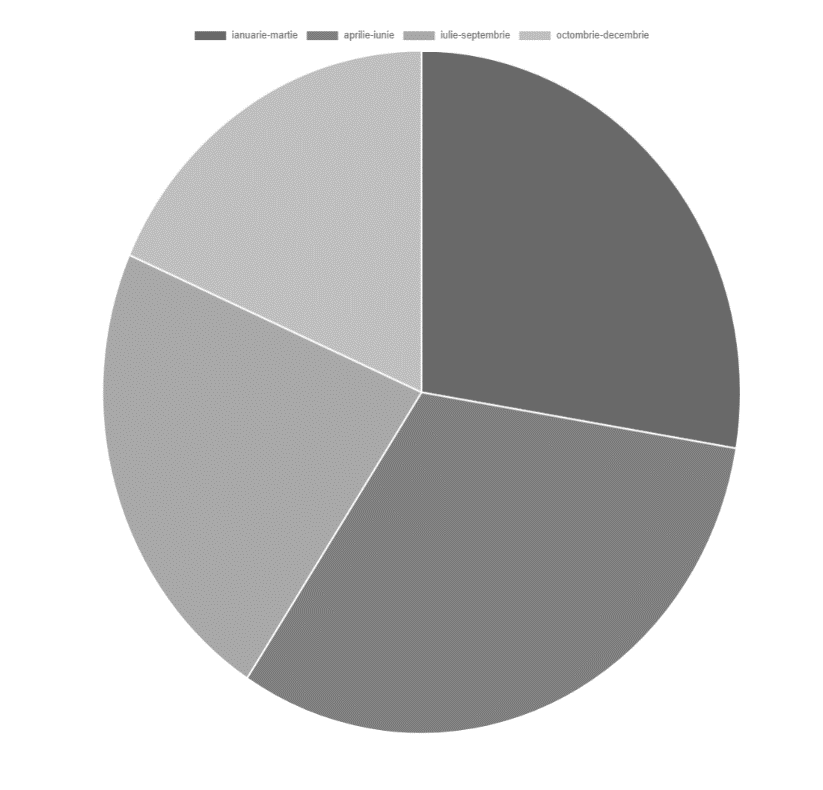
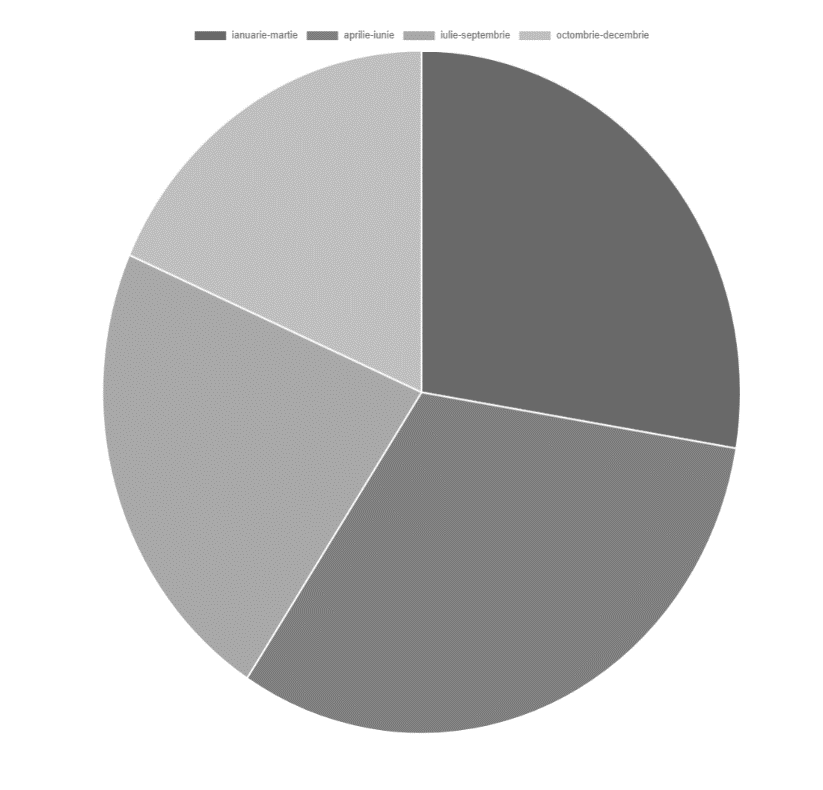
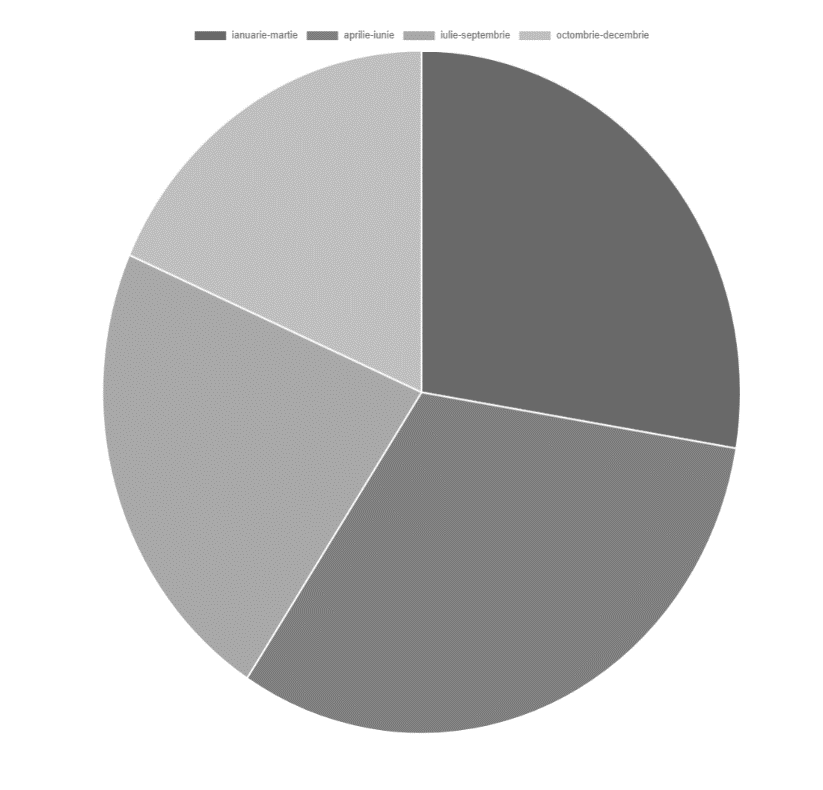
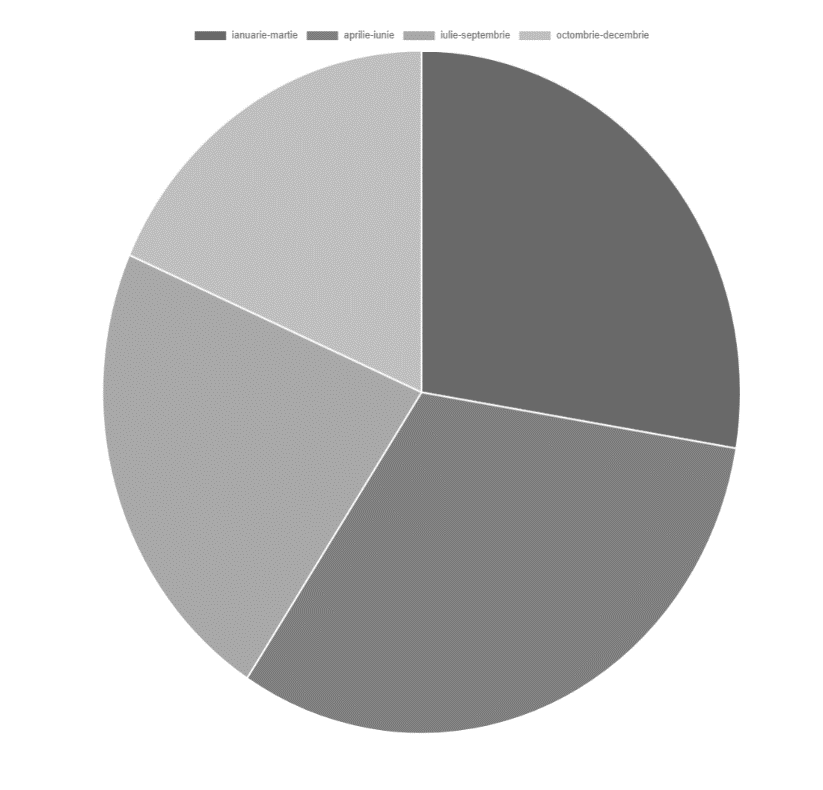
WHERE EXTRACT(MONTH FROM data\_inceput) IN (7, 10)

SELECT COUNT(\*) FROM REZERVARE

WHERE EXTRACT(MONTH FROM data\_inceput) IN (10, 1)







**Cereri in functie de perioada anului**

