Задание ПБД: „Коли под Наем“

Момчил Милков
20621635

20.12.2022 г.

—

Програмиране за Бази от Данни

—

ас. А. Иванова

Задание на Проекта

Изисквания

Да се проектира база от данни за коли под наем, която да съхранява следната информация:

- Автомобил – марка, модел, година, цвят, изминати километри, вид цена за ден
- Клиент – име, адрес, телефон
- Служител – име, позиция, телефон
- Заемане – клиент, автомобил, служител, дата, брой дни

Правила:

- Всеки клиент не може да наема повече от една кола в едно и също време
- Всеки автомобил може да бъде само една марка и един модел

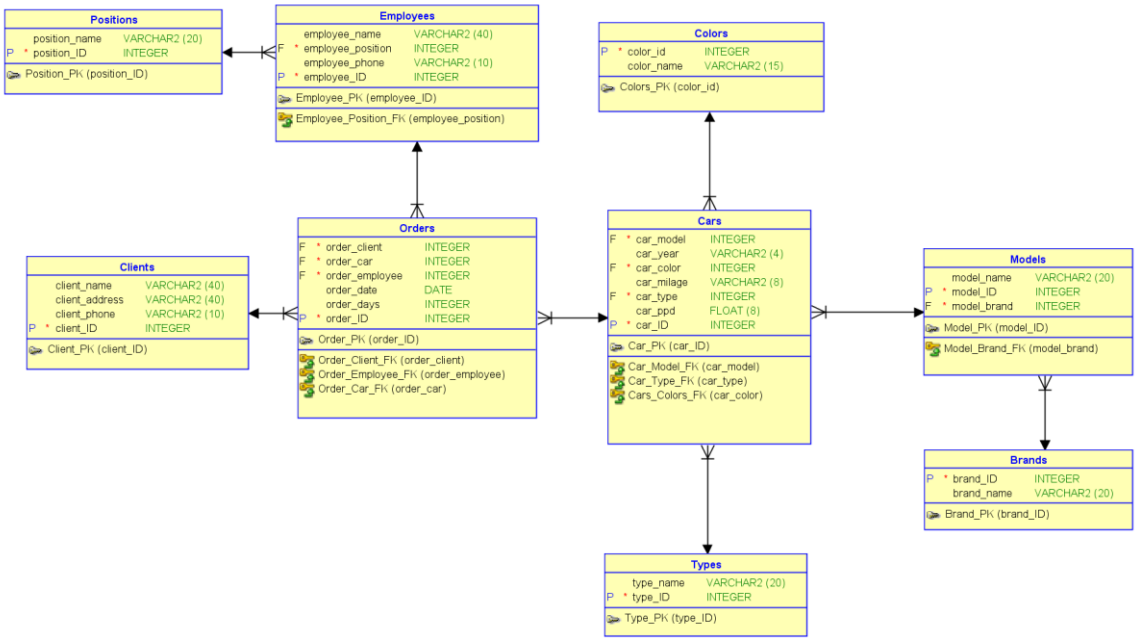
Базата данни трябва да е нормализирана и да позволява:

1. Въвеждане и корекция на данни
2. Търсене на автомобили по вид, модел, марка, цена за ден
3. Справки за:
    - Отдадени автомобили от служител, подредени по вид и дата
    - Последните 10 заемания, подредени по дни на заемане
    - Наети автомобили от клиент, подредени по дата
    - Отдадени на автомобили за период, подредени по клиенти

Проектиране на БД

Таблици

БД съдържа 9 таблици: Brands, Cars, Clients, Colors, Employees, Models, Orders, Positions, Types. За всяка една от тях съществуват Insert Procedure и Sequence, които се грижат за автоматичното инкрементиране и записване на ID стойностите на всеки ред от таблицата. Релационен модел на БД:



Тригери и Sequence-и

За всяка една от таблиците съществува “Before Insert” тригер и sequence (Нека за пример да вземем таблица Brands: Тригер- BR_TR_AI, Sequence- BRAND_SEQ; Синтаксисът за останалите

таблици е аналогичен.), които се грижат за автоматичната инкрементация на ID(PK) стойностите на всеки ред от таблицата. Съществува и тригер, който следи за записване на наемане на кола в период, в който тя вече е наета, като съответно предодвратява наемането (CHECK_DATES).

Процедури

За всяка една от таблиците съществуват процедури за Записване и Обновяване на данни, които приемат от потребителя данните за реда, който той иска да въведе, или обнови (Пример: BRA_INS, BRA_UPD). Също така, за всяка една от справките, предвидени в проекта, присъстват процедури, приемащи като входни данни критерият, по който ще се извършва избраната справка(Пример: FIND_CAR_BY_BRAND).

Реализация

SQL команди – DDL, DML

- Create Tables:

```
CREATE TABLE brands (
    brand_id     INTEGER NOT NULL,
    brand_name   VARCHAR2(20)
);

ALTER TABLE brands ADD CONSTRAINT brand_pk PRIMARY KEY (
brand_id );

CREATE TABLE cars (
    car_model    INTEGER NOT NULL,
    car_year     VARCHAR2(4),
    car_color    INTEGER NOT NULL,
    car_milage   VARCHAR2(8),
    car_type     INTEGER NOT NULL,
    car_ppd      FLOAT(8),
    car_id       INTEGER NOT NULL
);

ALTER TABLE cars ADD CONSTRAINT car_pk PRIMARY KEY ( car_id );

CREATE TABLE clients (
    client_name      VARCHAR2(40),
    client_address   VARCHAR2(40),
    client_phone     VARCHAR2(10),
    client_id        INTEGER NOT NULL
);

ALTER TABLE clients ADD CONSTRAINT client_pk PRIMARY KEY (
client_id );

CREATE TABLE colors (
    color_id     INTEGER NOT NULL,
```

```
    color_name   VARCHAR2(15)
);

ALTER TABLE colors ADD CONSTRAINT colors_pk PRIMARY KEY (
color_id );

CREATE TABLE employees (
    employee_name       VARCHAR2(40),
    employee_position   INTEGER NOT NULL,
    employee_phone      VARCHAR2(10),
    employee_id         INTEGER NOT NULL
);

ALTER TABLE employees ADD CONSTRAINT employee_pk PRIMARY KEY (
employee_id );

CREATE TABLE models (
    model_name    VARCHAR2(20),
    model_id      INTEGER NOT NULL,
    model_brand   INTEGER NOT NULL
);

ALTER TABLE models ADD CONSTRAINT model_pk PRIMARY KEY (
model_id );

CREATE TABLE orders (
    order_client     INTEGER NOT NULL,
    order_car        INTEGER NOT NULL,
    order_employee   INTEGER NOT NULL,
    order_date       DATE,
    order_days       INTEGER,
    order_id         INTEGER NOT NULL
);

ALTER TABLE orders ADD CONSTRAINT order_pk PRIMARY KEY (
order_id );

CREATE TABLE positions (
    position_name   VARCHAR2(20),
    position_id     INTEGER NOT NULL
);

ALTER TABLE positions ADD CONSTRAINT position_pk PRIMARY KEY (
position_id );

CREATE TABLE types (
    type_name   VARCHAR2(20),
    type_id     INTEGER NOT NULL
);

ALTER TABLE types ADD CONSTRAINT type_pk PRIMARY KEY ( type_id
);
```

```
ALTER TABLE cars
    ADD CONSTRAINT car_model_fk FOREIGN KEY ( car_model )
        REFERENCES models ( model_id );

ALTER TABLE cars
    ADD CONSTRAINT car_type_fk FOREIGN KEY ( car_type )
        REFERENCES types ( type_id );

ALTER TABLE cars
    ADD CONSTRAINT cars_colors_fk FOREIGN KEY ( car_color )
        REFERENCES colors ( color_id );

ALTER TABLE employees
    ADD CONSTRAINT employee_position_fk FOREIGN KEY (
employee_position )
        REFERENCES positions ( position_id );

ALTER TABLE models
    ADD CONSTRAINT model_brand_fk FOREIGN KEY ( model_brand )
        REFERENCES brands ( brand_id );

ALTER TABLE orders
    ADD CONSTRAINT order_car_fk FOREIGN KEY ( order_car )
        REFERENCES cars ( car_id );

ALTER TABLE orders
    ADD CONSTRAINT order_client_fk FOREIGN KEY ( order_client
)
        REFERENCES clients ( client_id );

ALTER TABLE orders
    ADD CONSTRAINT order_employee_fk FOREIGN KEY (
order_employee )
     REFERENCES employees ( employee_id );
```

- Insert:

```
REM INSERTING into BRANDS

Insert into BRANDS (BRAND_ID,BRAND_NAME) values (1,'BMW');
Insert into BRANDS (BRAND_ID,BRAND_NAME) values (2,'Audi');
Insert into BRANDS (BRAND_ID,BRAND_NAME) values
(3,'Mercedes');
Insert into BRANDS (BRAND_ID,BRAND_NAME) values (4,'Peugeot');
Insert into BRANDS (BRAND_ID,BRAND_NAME) values (5,'Tesla');
REM INSERTING into CARS

Insert into CARS
(CAR_MODEL,CAR_YEAR,CAR_COLOR,CAR_MILAGE,CAR_TYPE,CAR_PPD,CAR_
ID) values (1,'1990',1,'1234',1,50,1);
Insert into CARS
(CAR_MODEL,CAR_YEAR,CAR_COLOR,CAR_MILAGE,CAR_TYPE,CAR_PPD,CAR_
ID) values (2,'1989',2,'123',2,10,2);
```

```
Insert into CARS
(CAR_MODEL,CAR_YEAR,CAR_COLOR,CAR_MILAGE,CAR_TYPE,CAR_PPD,CAR_
ID) values (3,'2008',3,'12',3,12.2,3);
Insert into CARS
(CAR_MODEL,CAR_YEAR,CAR_COLOR,CAR_MILAGE,CAR_TYPE,CAR_PPD,CAR_
ID) values (4,'1999',4,'1243',4,124,4);
Insert into CARS
(CAR_MODEL,CAR_YEAR,CAR_COLOR,CAR_MILAGE,CAR_TYPE,CAR_PPD,CAR_
ID) values (5,'2007',5,'4452',3,11.2,5);
Insert into CARS
(CAR_MODEL,CAR_YEAR,CAR_COLOR,CAR_MILAGE,CAR_TYPE,CAR_PPD,CAR_
ID) values (6,'2001',6,'1123',1,121,6);
REM INSERTING into CLIENTS

Insert into CLIENTS
(CLIENT_NAME,CLIENT_ADDRESS,CLIENT_PHONE,CLIENT_ID) values
('Johnny Depp','Havanna, momi str.','0456194412',1);
Insert into CLIENTS
(CLIENT_NAME,CLIENT_ADDRESS,CLIENT_PHONE,CLIENT_ID) values
('Bruce Willis','USA, varna','0781249562',2);
Insert into CLIENTS
(CLIENT_NAME,CLIENT_ADDRESS,CLIENT_PHONE,CLIENT_ID) values
('Momi Groba','Bulgaria, stidentska','0476594412',3);
Insert into CLIENTS
(CLIENT_NAME,CLIENT_ADDRESS,CLIENT_PHONE,CLIENT_ID) values
('Merlyn Monroe','romana, cooulia str.','1236194412',4);
REM INSERTING into COLORS

Insert into COLORS (COLOR_ID,COLOR_NAME) values (1,'red');
Insert into COLORS (COLOR_ID,COLOR_NAME) values (2,'blue');
Insert into COLORS (COLOR_ID,COLOR_NAME) values (3,'white');
Insert into COLORS (COLOR_ID,COLOR_NAME) values (4,'black');
Insert into COLORS (COLOR_ID,COLOR_NAME) values (5,'green');
Insert into COLORS (COLOR_ID,COLOR_NAME) values (6,'yellow');
REM INSERTING into EMPLOYEES

Insert into EMPLOYEES
(EMPLOYEE_NAME,EMPLOYEE_POSITION,EMPLOYEE_PHONE,EMPLOYEE_ID)
values ('Kaloqn Koko',1,'123456789',1);
Insert into EMPLOYEES
(EMPLOYEE_NAME,EMPLOYEE_POSITION,EMPLOYEE_PHONE,EMPLOYEE_ID)
values ('Boris III',2,'123456123',2);
Insert into EMPLOYEES
(EMPLOYEE_NAME,EMPLOYEE_POSITION,EMPLOYEE_PHONE,EMPLOYEE_ID)
values ('Han Asparuh',3,'123456345',3);
Insert into EMPLOYEES
(EMPLOYEE_NAME,EMPLOYEE_POSITION,EMPLOYEE_PHONE,EMPLOYEE_ID)
values ('Kazuto Kirigaya',3,'123456567',4);
REM INSERTING into MODELS

Insert into MODELS (MODEL_NAME,MODEL_ID,MODEL_BRAND) values
('Sedan',1,1);
```

```
Insert into MODELS (MODEL_NAME,MODEL_ID,MODEL_BRAND) values
('A5',2,2);
Insert into MODELS (MODEL_NAME,MODEL_ID,MODEL_BRAND) values
('c220',3,3);
Insert into MODELS (MODEL_NAME,MODEL_ID,MODEL_BRAND) values
('X',4,5);
Insert into MODELS (MODEL_NAME,MODEL_ID,MODEL_BRAND) values
('A3',5,2);
Insert into MODELS (MODEL_NAME,MODEL_ID,MODEL_BRAND) values
('307',6,4);
REM INSERTING into ORDERS

Insert into ORDERS
(ORDER_CLIENT,ORDER_CAR,ORDER_EMPLOYEE,ORDER_DATE,ORDER_DAYS,O
RDER_ID) values (1,1,1,to_date('17-OCT-19','DD-MON-RR'),11,1);
Insert into ORDERS
(ORDER_CLIENT,ORDER_CAR,ORDER_EMPLOYEE,ORDER_DATE,ORDER_DAYS,O
RDER_ID) values (1,2,2,to_date('04-OCT-22','DD-MON-RR'),12,2);
Insert into ORDERS
(ORDER_CLIENT,ORDER_CAR,ORDER_EMPLOYEE,ORDER_DATE,ORDER_DAYS,O
RDER_ID) values (2,3,3,to_date('29-OCT-22','DD-MON-RR'),1,3);
Insert into ORDERS
(ORDER_CLIENT,ORDER_CAR,ORDER_EMPLOYEE,ORDER_DATE,ORDER_DAYS,O
RDER_ID) values (3,4,4,to_date('15-JUN-22','DD-MON-RR'),3,4);
Insert into ORDERS
(ORDER_CLIENT,ORDER_CAR,ORDER_EMPLOYEE,ORDER_DATE,ORDER_DAYS,O
RDER_ID) values (4,5,1,to_date('13-SEP-22','DD-MON-RR'),5,5);
Insert into ORDERS
(ORDER_CLIENT,ORDER_CAR,ORDER_EMPLOYEE,ORDER_DATE,ORDER_DAYS,O
RDER_ID) values (4,6,2,to_date('19-FEB-22','DD-MON-RR'),3,6);
REM INSERTING into POSITIONS

Insert into POSITIONS (POSITION_NAME,POSITION_ID) values
('Director',1);
Insert into POSITIONS (POSITION_NAME,POSITION_ID) values
('Manager',2);
Insert into POSITIONS (POSITION_NAME,POSITION_ID) values
('Cashier',3);
REM INSERTING into TYPES

Insert into TYPES (TYPE_NAME,TYPE_ID) values ('Cabrio',2);
Insert into TYPES (TYPE_NAME,TYPE_ID) values ('Jeep',3);
Insert into TYPES (TYPE_NAME,TYPE_ID) values ('Van',4);
Insert into TYPES (TYPE_NAME,TYPE_ID) values ('Sedan',1);
```

- Update:

```
UPDATE brands
SET brand_name = 'updated'
WHERE brand_id = 2;

UPDATE cars
SET car_model = 1, car_year = 0000, car_color = 1, car_milage
= '0', car_type = 1, car_ppd = 1
```

```
WHERE car_id = 2;

UPDATE clients
SET client_name = 'updated', client_address = 'updated',
client_phone = '0000000000'
WHERE client_id = 2;

UPDATE colors
SET color_name = 'updated'
WHERE color_id = 2;

UPDATE employees
SET employee_name = 'updated', employee_position = 1,
employee_phone = '0000000000'
WHERE employee_id = 2;

UPDATE models
SET model_name = 'updated', model_brand = 1
WHERE model_id = 2;

UPDATE orders
SET order_client = 1, order_car = 1, order_employee = 1,
order_date = '01-OCT-19', order_days = 1
WHERE order_id = 2;

UPDATE positions
SET position_name = 'updated'
WHERE position_id = 2;

UPDATE types
SET type_name = 'updated'
WHERE type_id = 2;
```

PL/SQL процедури/тригери/курсори

- Тригери: Auto Increment

```
--------------------------------------------------------
--  DDL for Trigger BR_TR_AI
--------------------------------------------------------

  CREATE OR REPLACE TRIGGER "BR_TR_AI"
before insert on brands
for each row
 WHEN (NEW.brand_id is null) begin
    :NEW.brand_id := BRAND_SEQ.NEXTVAL;
end;
/
ALTER TRIGGER "BR_TR_AI" ENABLE;
--------------------------------------------------------
--  DDL for Trigger CA_TR_AI
```

```
--------------------------------------------------------

  CREATE OR REPLACE TRIGGER "CA_TR_AI"
before insert on cars
for each row
 WHEN (NEW.car_id is null) begin
    :NEW.car_id := CAR_SEQ.NEXTVAL;
end;
/
ALTER TRIGGER "CA_TR_AI" ENABLE;
--------------------------------------------------------
--  DDL for Trigger CL_TR_AI
--------------------------------------------------------

  CREATE OR REPLACE TRIGGER "CL_TR_AI"
before insert on clients
for each row
 WHEN (NEW.client_id is null) begin
    :NEW.client_id := CLIENT_SEQ.NEXTVAL;
end;
/
ALTER TRIGGER "CL_TR_AI" ENABLE;
--------------------------------------------------------
--  DDL for Trigger CO_TR_AI
--------------------------------------------------------

  CREATE OR REPLACE TRIGGER "CO_TR_AI"
before insert on colors
for each row
 WHEN (NEW.color_id is null) begin
    :NEW.color_id := COLOR_SEQ.NEXTVAL;
end;
/
ALTER TRIGGER "CO_TR_AI" ENABLE;
--------------------------------------------------------
--  DDL for Trigger EM_TR_AI
--------------------------------------------------------

  CREATE OR REPLACE TRIGGER "EM_TR_AI"
before insert on employees
for each row
 WHEN (NEW.employee_id is null) begin
    :NEW.employee_id := EMPLOYEE_SEQ.NEXTVAL;
end;
/
ALTER TRIGGER "EM_TR_AI" ENABLE;
--------------------------------------------------------
--  DDL for Trigger MO_TR_AI
--------------------------------------------------------

  CREATE OR REPLACE TRIGGER "MO_TR_AI"
before insert on models
for each row
```

```
 WHEN (NEW.model_id is null) begin
    :NEW.model_id := MODEL_SEQ.NEXTVAL;
end;
/
ALTER TRIGGER "MO_TR_AI" ENABLE;
--------------------------------------------------------
--  DDL for Trigger OR_TR_AI
--------------------------------------------------------

  CREATE OR REPLACE TRIGGER "OR_TR_AI"
before insert on orders
for each row
 WHEN (NEW.order_id is null) begin
    :NEW.order_id := ORDER_SEQ.NEXTVAL;
end;
/
ALTER TRIGGER "OR_TR_AI" ENABLE;
--------------------------------------------------------
--  DDL for Trigger PO_TR_AI
--------------------------------------------------------

  CREATE OR REPLACE TRIGGER "PO_TR_AI"
before insert on positions
for each row
 WHEN (NEW.position_id is null) begin
    :NEW.position_id := POSITION_SEQ.NEXTVAL;
end;
/
ALTER TRIGGER "PO_TR_AI" ENABLE;
--------------------------------------------------------
--  DDL for Trigger TY_TR_AI
--------------------------------------------------------

  CREATE OR REPLACE TRIGGER "TY_TR_AI"
before insert on types
for each row
 WHEN (NEW.type_id is null) begin
    :NEW.type_id := TYPE_SEQ.NEXTVAL;
end;
/
ALTER TRIGGER "TY_TR_AI" ENABLE;
```

- Sequence:

```
--------------------------------------------------------
--  DDL for Sequence BRAND_SEQ
--------------------------------------------------------

   CREATE SEQUENCE  "BRAND_SEQ"  MINVALUE 1 MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 6
NOCACHE  ORDER  NOCYCLE ;
--------------------------------------------------------
--  DDL for Sequence CAR_SEQ
```

```
--------------------------------------------------------

   CREATE SEQUENCE  "CAR_SEQ"  MINVALUE 1 MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 7
NOCACHE  ORDER  NOCYCLE ;
--------------------------------------------------------
--  DDL for Sequence CLIENT_SEQ
--------------------------------------------------------

   CREATE SEQUENCE  "CLIENT_SEQ"  MINVALUE 1 MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 5
NOCACHE  ORDER  NOCYCLE ;
--------------------------------------------------------
--  DDL for Sequence COLOR_SEQ
--------------------------------------------------------

   CREATE SEQUENCE  "COLOR_SEQ"  MINVALUE 1 MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 8
NOCACHE  ORDER  NOCYCLE ;
--------------------------------------------------------
--  DDL for Sequence EMPLOYEE_SEQ
--------------------------------------------------------

   CREATE SEQUENCE  "EMPLOYEE_SEQ"  MINVALUE 1 MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 5
NOCACHE  ORDER  NOCYCLE ;
--------------------------------------------------------
--  DDL for Sequence MODEL_SEQ
--------------------------------------------------------

   CREATE SEQUENCE  "MODEL_SEQ"  MINVALUE 1 MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 7
NOCACHE  ORDER  NOCYCLE ;
--------------------------------------------------------
--  DDL for Sequence ORDER_SEQ
--------------------------------------------------------

   CREATE SEQUENCE  "ORDER_SEQ"  MINVALUE 1 MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 7
NOCACHE  ORDER  NOCYCLE ;
--------------------------------------------------------
--  DDL for Sequence POSITION_SEQ
--------------------------------------------------------

   CREATE SEQUENCE  "POSITION_SEQ"  MINVALUE 1 MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 4
NOCACHE  ORDER  NOCYCLE ;
--------------------------------------------------------
--  DDL for Sequence TYPE_SEQ
--------------------------------------------------------
```

```
   CREATE SEQUENCE  "TYPE_SEQ"  MINVALUE 1 MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 5
NOCACHE  ORDER  NOCYCLE ;
```

- Insert Процедури:

```
--------------------------------------------------------
--  DDL for Procedure CLI_INS
--------------------------------------------------------
set define off;

  CREATE OR REPLACE PROCEDURE "CLI_INS"
(c_name clients.client_name%type,
 c_address clients.client_address%type,
 c_phone clients.client_phone%type)
as
begin
    insert into
clients(client_name,client_address,client_phone)
    values(c_name,c_address,c_phone);
end CLI_INS;

/
--------------------------------------------------------
--  DDL for Procedure COL_INS
--------------------------------------------------------
set define off;

  CREATE OR REPLACE PROCEDURE "COL_INS"
(c_name colors.color_name%type)
as
begin
    insert into colors(color_name)
    values(c_name);
end COL_INS;

/
--------------------------------------------------------
--  DDL for Procedure EMP_INS
--------------------------------------------------------
set define off;

  CREATE OR REPLACE PROCEDURE "EMP_INS"
(e_name employees.employee_name%type,
 e_position employees.employee_position%type,
 e_phone employees.employee_phone%type)
as
begin
    insert into employees(employee_name, employee_position,
employee_phone)
    values(e_name, e_position, e_phone);
end EMP_INS;
```

```
/
------------------------------------------------------
--  DDL for Procedure MOD_INS
------------------------------------------------------
set define off;

  CREATE OR REPLACE PROCEDURE "MOD_INS"
(m_name models.model_name%type,
 m_brand models.model_brand%type)
as
begin
    insert into models(model_name, model_brand)
    values(m_name,m_brand);
end MOD_INS;

/
------------------------------------------------------
--  DDL for Procedure ORD_INS
------------------------------------------------------
set define off;

  CREATE OR REPLACE PROCEDURE "ORD_INS"
(o_client orders.order_client%type,
 o_car orders.order_car%type,
 o_employee orders.order_employee%type,
 o_date orders.order_date%type,
 o_days orders.order_days%type)
as
begin
    insert into orders(order_client, order_car,
order_employee, order_date, order_days)
    values(o_client, o_car, o_employee, o_date, o_days);
end ORD_INS;

/
------------------------------------------------------
--  DDL for Procedure POS_INS
------------------------------------------------------
set define off;

  CREATE OR REPLACE PROCEDURE "POS_INS"
(p_name positions.position_name%type)
as
begin
    insert into positions(position_name)
    values(p_name);
end POS_INS;

/
------------------------------------------------------
--  DDL for Procedure TYP_INS
------------------------------------------------------
set define off;
```

```
  CREATE OR REPLACE PROCEDURE "TYP_INS"
(t_name types.type_name%type)
as
begin
    insert into types(type_name)
    values(t_name);
end TYP_INS;

/
--------------------------------------------------------
--  DDL for Procedure BRA_INS
--------------------------------------------------------
set define off;

  CREATE OR REPLACE PROCEDURE "BRA_INS"
(b_name brands.brand_name%type)
as
begin
    insert into brands(brand_name)
    values(b_name);
end BRA_INS;

/
--------------------------------------------------------
--  DDL for Procedure CAR_INS
--------------------------------------------------------
set define off;

  CREATE OR REPLACE PROCEDURE "CAR_INS"
(c_model cars.car_model%type,
 c_year cars.car_year%type,
 c_color cars.car_color%type,
 c_milage cars.car_milage%type,
 c_type cars.car_type%type,
 c_ppd cars.car_ppd%type)
as
begin
    insert into
cars(car_model,car_year,car_color,car_milage,car_type,car_ppd)
    values(c_model,c_year,c_color,c_milage,c_type,c_ppd);
end CAR_INS;

/
```

- Update Процедури:

```
--------------------------------------------------------
--  DDL for Procedure BRA_UPD
--------------------------------------------------------
set define off;

  CREATE OR REPLACE PROCEDURE "BRA_UPD"
```

```
(b_name brands.brand_name%type, ID integer)
as
begin
    UPDATE brands
    SET brand_name = b_name
    WHERE brand_id = ID;
end BRA_UPD;


/
--------------------------------------------------------
--  DDL for Procedure CAR_UPD
--------------------------------------------------------
set define off;

  CREATE OR REPLACE PROCEDURE "CAR_UPD"
(c_model cars.car_model%type,
 c_year cars.car_year%type,
 c_color cars.car_color%type,
 c_milage cars.car_milage%type,
 c_type cars.car_type%type,
 c_ppd cars.car_ppd%type,
 ID integer)
as
begin
    UPDATE cars
    SET car_model = c_model, car_year = c_year, car_color =
c_color, car_milage = c_milage, car_type = c_type, car_ppd =
c_ppd
    WHERE car_id = ID;
end CAR_UPD;


/
--------------------------------------------------------
--  DDL for Procedure CLI_UPD
--------------------------------------------------------
set define off;

  CREATE OR REPLACE PROCEDURE "CLI_UPD"
(c_name clients.client_name%type,
 c_address clients.client_address%type,
 c_phone clients.client_phone%type,
 ID integer)
as
begin
    UPDATE clients
    SET client_name = c_name, client_address = c_address,
client_phone = c_phone
    WHERE client_id = ID;
end CLI_UPD;
```

```
/
------------------------------------------------------
--  DDL for Procedure COL_UPD
------------------------------------------------------
set define off;

  CREATE OR REPLACE PROCEDURE "COL_UPD"
(c_name colors.color_name%type,
 ID integer)
as
begin
    UPDATE colors
    SET color_name = c_name
    WHERE color_id = ID;
end COL_UPD;


/
------------------------------------------------------
--  DDL for Procedure EMP_UPD
------------------------------------------------------
set define off;

  CREATE OR REPLACE PROCEDURE "EMP_UPD"
(e_name employees.employee_name%type,
 e_position employees.employee_position%type,
 e_phone employees.employee_phone%type,
 ID integer)
as
begin
    UPDATE employees
    SET employee_name = e_name, employee_position =
e_position, employee_phone = e_phone
    WHERE employee_id = ID;
end EMP_UPD;


/
------------------------------------------------------
--  DDL for Procedure MOD_UPD
------------------------------------------------------
set define off;

  CREATE OR REPLACE PROCEDURE "MOD_UPD"
(m_name models.model_name%type,
 m_brand models.model_brand%type,
 ID integer)
as
begin
    UPDATE models
    SET model_name = m_name, model_brand = m_brand
    WHERE model_id = ID;
end MOD_UPD;
```

```
/
--------------------------------------------------------
--  DDL for Procedure ORD_UPD
--------------------------------------------------------
set define off;

  CREATE OR REPLACE PROCEDURE "ORD_UPD"
(o_client orders.order_client%type,
 o_car orders.order_car%type,
 o_employee orders.order_client%type,
 o_date orders.order_date%type,
 o_days orders.order_days%type,
 ID integer)
as
begin
    UPDATE orders
    SET order_client = o_client, order_car = o_car,
order_employee = o_employee, order_date = o_date, order_days =
o_days
    WHERE order_id = ID;
end ORD_UPD;


/
--------------------------------------------------------
--  DDL for Procedure POS_UPD
--------------------------------------------------------
set define off;

  CREATE OR REPLACE PROCEDURE "POS_UPD"
(p_name positions.position_name%type,
 ID integer)
as
begin
    UPDATE positions
    SET position_name = p_name
    WHERE position_id = ID;
end POS_UPD;


/
--------------------------------------------------------
--  DDL for Procedure TYP_UPD
--------------------------------------------------------
set define off;

  CREATE OR REPLACE PROCEDURE "TYP_UPD"
(t_name types.type_name%type,
 ID integer)
as
begin
```

```
    UPDATE types
    SET type_name = t_name
    WHERE type_id = ID;
end TYP_UPD;


/
```

- Select(Query) Процедури:

```
--------------------------------------------------------
--  DDL for Procedure FIND_CAR_BY_BRAND
--------------------------------------------------------
set define off;

  CREATE OR REPLACE PROCEDURE "FIND_CAR_BY_BRAND" (carBrand
brands.brand_name%type)
is
begin
declare
cursor CarBrandCursor is
select  c.car_year, co.color_name, b.brand_name, c.car_milage,
t.type_name, c.car_ppd, m.model_name
from CARS c join MODELS m on c.car_model=m.model_ID join
BRANDS b on m.model_brand=b.brand_ID join TYPES t on
c.car_type=t.type_ID join COLORS co on c.car_color=co.color_ID
where b.brand_name = carBrand;
begin
dbms_output.put_line('Cars with brand '||carBrand||':');
dbms_output.put_line('Year|Color|Brand|Milage|Type|PPD|Model')
;
for cust_record in CarBrandCursor
loop
dbms_output.put_line(cust_record.car_year||'|'||cust_record.co
lor_name||'|'||cust_record.brand_name||'|'||cust_record.car_mi
lage||'|'
||cust_record.type_name||'|'||cust_record.car_ppd||'|'||cust_r
ecord.model_name);
end loop;
end;
end;
```

```
begin
find_car_by_brand('BMW');
end;
```

```
Cars with brand BMW:
Year|Color|Brand|Milage|Type|PPD|Model
1234|red|BMW|1234|Sedan|123|Sedan
1990|red|BMW|1234|Sedan|50|Sedan
```

```
/
--------------------------------------------------------
--  DDL for Procedure FIND_CAR_BY_CLIENT
--------------------------------------------------------
set define off;
```

```
  CREATE OR REPLACE PROCEDURE "FIND_CAR_BY_CLIENT" (carClient
clients.client_name%type)
is
begin
declare
cursor CarClientCursor is
select cl.client_name, o.order_date, c.car_year,
co.color_name, b.brand_name, c.car_milage, t.type_name,
c.car_ppd, m.model_name
from CARS c join MODELS m on c.car_model=m.model_ID join
BRANDS b on m.model_brand=b.brand_ID join TYPES t on
c.car_type=t.type_ID
join COLORS co on c.car_color=co.color_ID join ORDERS o on
c.car_id = o.order_car join CLIENTS cl on o.order_client =
cl.client_id
where cl.client_name = carClient
order by o.order_date;
begin
dbms_output.put_line('Cars rented by '||carClient||', ordered
by date:');
dbms_output.put_line('Client|Date|Year|Color|Brand|Milage|Type
|PPD|Model');
for cust_record in CarClientCursor
loop
dbms_output.put_line(cust_record.client_name||'|'||cust_record
.order_date||'|'||cust_record.car_year||'|'||cust_record.color
_name||'|'||cust_record.brand_name||'|'||cust_record.car_milag
e||'|'
||cust_record.type_name||'|'||cust_record.car_ppd||'|'||cust_r
ecord.model_name);
end loop;
end;
end;
```

```
begin
find_car_by_client('Johnny Depp');
end;
```

```
Cars rented by Johnny Depp, ordered by date:
Client|Date|Year|Color|Brand|Milage|Type|PPD|Model
Johnny Depp|17-OCT-19|1990|red|BMW|1234|Sedan|50|Sedan
Johnny Depp|18-NOV-19|1990|red|BMW|1234|Sedan|50|Sedan
Johnny Depp|04-OCT-22|1989|blue|Audi|123|Cabrio|10|A5
Johnny Depp|29-OCT-22|2008|white|Mercedes|12|Jeep|12.2|c220
```

```
/
--------------------------------------------------------
--  DDL for Procedure FIND_CAR_BY_EMPLOYEE
--------------------------------------------------------
set define off;

  CREATE OR REPLACE PROCEDURE "FIND_CAR_BY_EMPLOYEE"
(carEmployee employees.employee_name%type)
```

```
is
begin
declare
cursor CarEmployeeCursor is
select e.employee_name, o.order_date, c.car_year,
co.color_name, b.brand_name, c.car_milage, t.type_name,
c.car_ppd, m.model_name
from CARS c join MODELS m on c.car_model=m.model_ID join
BRANDS b on m.model_brand=b.brand_ID join TYPES t on
c.car_type=t.type_ID
join COLORS co on c.car_color=co.color_ID join ORDERS o on
c.car_id = o.order_car join EMPLOYEES e on o.order_employee =
e.employee_id
where e.employee_name = carEmployee
order by t.type_name, o.order_date;
begin
dbms_output.put_line('Cars issued by '||carEmployee||',
ordered by type and date:');
dbms_output.put_line('Employee|Date|Year|Color|Brand|Milage|Ty
pe|PPD|Model');
for cust_record in CarEmployeeCursor
loop
dbms_output.put_line(cust_record.employee_name||'|'||cust_reco
rd.order_date||'|'||cust_record.car_year||'|'||cust_record.col
or_name||'|'||cust_record.brand_name||'|'||cust_record.car_mil
age||'|'
||cust_record.type_name||'|'||cust_record.car_ppd||'|'||cust_r
ecord.model_name);
end loop;
end;
end;
```

```
begin
find_car_by_employee('Kaloqn Koko');
end;
```

```
Cars issued by Kaloqn Koko, ordered by type and date:
Employee|Date|Year|Color|Brand|Milage|Type|PPD|Model
Kaloqn Koko|13-SEP-22|2007|green|Audi|4452|Jeep|11.2|A3
Kaloqn Koko|29-OCT-22|2008|white|Mercedes|12|Jeep|12.2|c220
Kaloqn Koko|17-OCT-19|1990|red|BMW|1234|Sedan|50|Sedan
Kaloqn Koko|18-NOV-19|1990|red|BMW|1234|Sedan|50|Sedan
```

```
/
-----------------------------------------------------------
--  DDL for Procedure FIND_CAR_BY_MODEL
-----------------------------------------------------------
set define off;

  CREATE OR REPLACE PROCEDURE "FIND_CAR_BY_MODEL" (carModel
models.model_name%type)
is
begin
```

```
declare
cursor CarModelCursor is
select  c.car_year, co.color_name, b.brand_name, c.car_milage,
t.type_name, c.car_ppd, m.model_name
from CARS c join MODELS m on c.car_model=m.model_ID join
BRANDS b on m.model_brand=b.brand_ID join TYPES t on
c.car_type=t.type_ID join COLORS co on c.car_color=co.color_ID
where m.model_name = carModel;
begin
dbms_output.put_line('Cars with model '||carModel||':');
dbms_output.put_line('Year|Color|Brand|Milage|Type|PPD|Model')
;
for cust_record in CarModelCursor
loop
dbms_output.put_line(cust_record.car_year||'|'||cust_record.co
lor_name||'|'||cust_record.brand_name||'|'||cust_record.car_mi
lage||'|'
||cust_record.type_name||'|'||cust_record.car_ppd||'|'||cust_r
ecord.model_name);
end loop;
end;
end;
```

```
begin
find_car_by_model('Sedan');
end;
```

```
Cars with model Sedan:
Year|Color|Brand|Milage|Type|PPD|Model
1234|red|BMW|1234|Sedan|123|Sedan
1990|red|BMW|1234|Sedan|50|Sedan
```

```
/
-----------------------------------------------------------
--  DDL for Procedure FIND_CAR_BY_PERIOD
-----------------------------------------------------------
set define off;

  CREATE OR REPLACE PROCEDURE "FIND_CAR_BY_PERIOD" (startDate
orders.order_date%type, endDate orders.order_date%type)
is
begin
declare
cursor CarPeriodCursor is
select cl.client_name, o.order_date, c.car_year,
co.color_name, b.brand_name, c.car_milage, t.type_name,
c.car_ppd, m.model_name
from CARS c join MODELS m on c.car_model=m.model_ID join
BRANDS b on m.model_brand=b.brand_ID join TYPES t on
c.car_type=t.type_ID
join COLORS co on c.car_color=co.color_ID join ORDERS o on
c.car_id = o.order_car join CLIENTS cl on o.order_client =
cl.client_id
where o.order_date between startDate and endDate
order by cl.client_name;
begin
```

```
dbms_output.put_line('Cars rented between '||startDate||' and
'||endDate||', ordered by client:');
dbms_output.put_line('Client|Date|Year|Color|Brand|Milage|Type
|PPD|Model');
for cust_record in CarPeriodCursor
loop
dbms_output.put_line(cust_record.client_name||'|'||cust_record
.order_date||'|'||cust_record.car_year||'|'||cust_record.color
_name||'|'||cust_record.brand_name||'|'||cust_record.car_milag
e||'|'
||cust_record.type_name||'|'||cust_record.car_ppd||'|'||cust_r
ecord.model_name);
end loop;
end;
end;
```

```
begin
find_car_by_period('18-NOV-19', '15-JUN-22');
end;
```

```
Cars rented between 18-NOV-19 and 15-JUN-22, ordered by client:
Client|Date|Year|Color|Brand|Milage|Type|PPD|Model
Johnny Depp|18-NOV-19|1990|red|BMW|1234|Sedan|50|Sedan
Merlyn Monroe|19-FEB-22|2001|yellow|Peugeot|1123|Sedan|121|307
```

```
/
------------------------------------------------------------
--  DDL for Procedure FIND_CAR_BY_PPD
------------------------------------------------------------
set define off;

  CREATE OR REPLACE PROCEDURE "FIND_CAR_BY_PPD" (carPPD
cars.car_ppd%type)
is
begin
declare
cursor CarPPDCursor is
select  c.car_year, co.color_name, b.brand_name, c.car_milage,
t.type_name, c.car_ppd, m.model_name
from CARS c join MODELS m on c.car_model=m.model_ID join
BRANDS b on m.model_brand=b.brand_ID join TYPES t on
c.car_type=t.type_ID join COLORS co on c.car_color=co.color_ID
where c.car_ppd = carPPD;
begin
dbms_output.put_line('Cars with PPD '||carPPD||':');
dbms_output.put_line('Year|Color|Brand|Milage|Type|PPD|Model')
;
for cust_record in CarPPDCursor
loop
```

```
dbms_output.put_line(cust_record.car_year||'|'||cust_record.co
lor_name||'|'||cust_record.brand_name||'|'||cust_record.car_mi
lage||'|'
||cust_record.type_name||'|'||cust_record.car_ppd||'|'||cust_r
ecord.model_name);
end loop;
end;
end;
```

```
begin
find_car_by_ppd(12.2);
end;
```

```
Cars with PPD 12.2:
Year|Color|Brand|Milage|Type|PPD|Model
2008|white|Mercedes|12|Jeep|12.2|c220
```

```
/
--------------------------------------------------------
--  DDL for Procedure FIND_CAR_BY_TYPE
--------------------------------------------------------
set define off;

  CREATE OR REPLACE PROCEDURE "FIND_CAR_BY_TYPE" (carType
types.type_name%type)
is
begin
declare
cursor CarTypeCursor is
select  c.car_year, co.color_name, b.brand_name, c.car_milage,
t.type_name, c.car_ppd, m.model_name
from CARS c join MODELS m on c.car_model=m.model_ID join
BRANDS b on m.model_brand=b.brand_ID join TYPES t on
c.car_type=t.type_ID join COLORS co on c.car_color=co.color_ID
where t.type_name = carType;
begin
dbms_output.put_line('Cars with type '||carType||':');
dbms_output.put_line('Year|Color|Brand|Milage|Type|PPD|Model')
;
for cust_record in CarTypeCursor
loop
dbms_output.put_line(cust_record.car_year||'|'||cust_record.co
lor_name||'|'||cust_record.brand_name||'|'||cust_record.car_mi
lage||'|'
||cust_record.type_name||'|'||cust_record.car_ppd||'|'||cust_r
ecord.model_name);
end loop;
end;
end;
```

```
begin
find_car_by_type('Jeep');
end;
```

```
Cars with type Jeep:
Year|Color|Brand|Milage|Type|PPD|Model
2008|white|Mercedes|12|Jeep|12.2|c220
2007|green|Audi|4452|Jeep|11.2|A3
```

```
/
```

```
--------------------------------------------------------
--  DDL for Procedure FIND_LAST_ORDERS
--------------------------------------------------------
set define off;

  CREATE OR REPLACE PROCEDURE "FIND_LAST_ORDERS"
is
begin
declare
cursor LastCursor is
(select cl.client_name, e.employee_name, o.order_days,
o.order_date, c.car_year, co.color_name, b.brand_name,
c.car_milage, t.type_name, c.car_ppd, m.model_name
from CARS c join MODELS m on c.car_model=m.model_ID join
BRANDS b on m.model_brand=b.brand_ID join TYPES t on
c.car_type=t.type_ID
join COLORS co on c.car_color=co.color_ID join ORDERS o on
c.car_id = o.order_car join CLIENTS cl on o.order_client =
cl.client_id join EMPLOYEES e on o.order_employee =
e.employee_id
order by o.order_date desc)
where rownum <=10
order by order_days;
begin
dbms_output.put_line('Last 10 rented cars, ordered by number
of days of rent:');
dbms_output.put_line('Client|Employee|Days|Date|Year|Color|Bra
nd|Milage|Type|PPD|Model');
for cust_record in LastCursor
loop
dbms_output.put_line(cust_record.client_name||'|'||cust_record
.employee_name||'|'||cust_record.order_days||'|'||cust_record.
order_date||'|'||cust_record.car_year||'|'||cust_record.color_
name||'|'||cust_record.brand_name||'|'||cust_record.car_milage
||'|'
||cust_record.type_name||'|'||cust_record.car_ppd||'|'||cust_r
ecord.model_name);
end loop;
end;
end;
```

```
begin
find_last_orders;
end;
```

```
Last 5 rented cars, ordered by number of days of rent:
Client|Employee|Days|Date|Year|Color|Brand|Milage|Type|PPD|Model
Bruce Willis|Han Asparuh|1|29-OCT-22|2008|white|Mercedes|12|Jeep|12.2|c220
Johnny Depp|Kaloqn Koko|3|29-OCT-22|2008|white|Mercedes|12|Jeep|12.2|c220
Momi Groba|Kazuto Kirigaya|3|15-JUN-22|1999|black|Tesla|1243|Van|124|X
Merlyn Monroe|Kaloqn Koko|5|13-SEP-22|2007|green|Audi|4452|Jeep|11.2|A3
Johnny Depp|Boris III|12|04-OCT-22|1989|blue|Audi|123|Cabrio|10|A5
```

Пример с 5 реда

```
/
```

- Тригер Check_Dates:

```
--------------------------------------------------------
--  DDL for Trigger CHECK_DATES
--------------------------------------------------------

  CREATE OR REPLACE TRIGGER "CHECK_DATES"
before insert on ORDERS
for each row
DECLARE rowCount number;
begin
select count(order_id) into rowCount from Orders
where :new.order_car = order_car
AND ((:new.order_date between order_date and
order_date+order_days)
OR (:new.order_date+:new.order_days between order_date and
order_date+order_days)
OR ((:new.order_date < order_date) AND (order_date+order_days
< :new.order_date+:new.order_days)));
if(rowCount > 0)
then
raise_application_error(-20103, 'That car has already been
rented for that period!');
end if;
end;
/
ALTER TRIGGER "CHECK_DATES" ENABLE;
```

```
begin
ORD_INS(1, 1, 1, '18-NOV-19', 5);
end;
Error report -
ORA-20103: That car has already been rented for that period!
```

Пример при въведен невалиден период на наемане