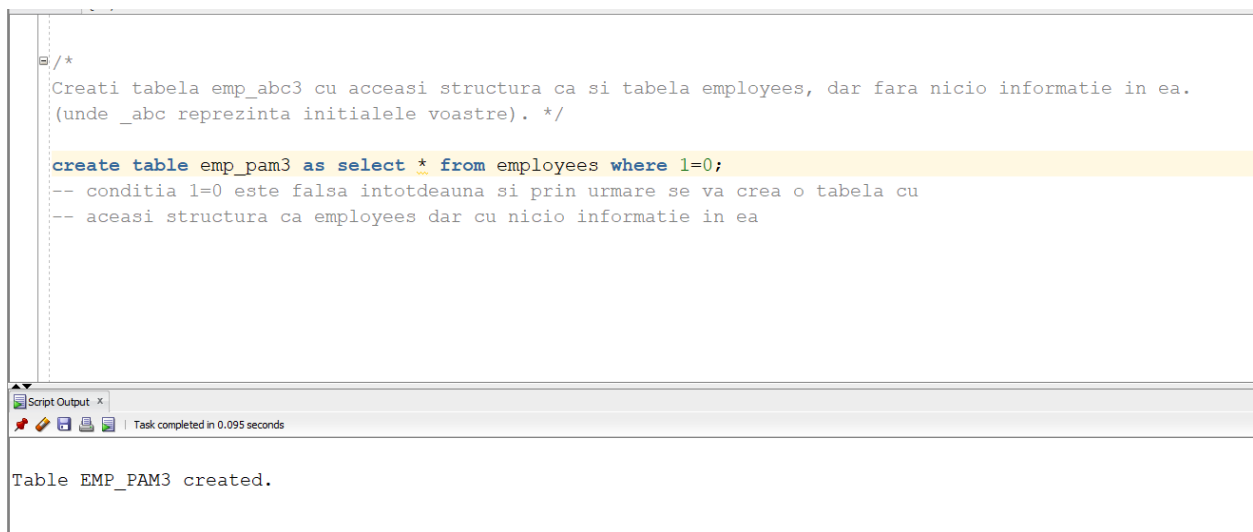


Tema 2

Exercitiul 1

```
create table emp_pam3 as select * from employees where 1=0;
-- conditia 1=0 este falsa intotdeauna si prin urmare se va crea o tabela cu
-- aceasi structura ca employees dar cu nicio informatie in ea
```



The screenshot shows a SQL script editor with a light gray background. The script content is as follows:

```
/*
Creati tabela emp_abc3 cu aceasi structura ca si tabela employees, dar fara nicio informatie in ea.
(unde _abc reprezinta initialele voastre). */

create table emp_pam3 as select * from employees where 1=0;
-- conditia 1=0 este falsa intotdeauna si prin urmare se va crea o tabela cu
-- aceasi structura ca employees dar cu nicio informatie in ea
```

Below the script editor, there is a 'Script Output' window with a status bar indicating 'Task completed in 0.095 seconds'. The output text reads: 'Table EMP_PAM3 created.'

Exercitiul 2

```
insert into emp_pam3
select * from employees where salary > (select salary as Salary_PAM
                                         from employees
                                         where first_name like 'Adam'); -- 31 de inserari

commit;
rollback;
```

-- Pentru a rezolva aceasta cerere am facut o subcerere care selecteaza salariul angajatului cu numele Adam

-- si apoi am selectat toti angajatii care au salariul mai mare decat salariul din subcerere.

-- Am folosit subcereri necorelate. Au fost inserate 31 de informatii. Dupa ce am dat commit am permanentizat modificarile efectuate. Daca dupa ultimul commit

-- dam si un rollback nu se va modifica nimic pentru ca aceasta comanda se intoarce la ultima sesiune comisa renuntand

-- la modificarile ulterioare si nu au mai fost facute modificari dupa commit.

```

/*
Inserati in tabela emp_abc3 angajatii care au salariul mai mare ca cel al salariatului cu prenumele Adam.
Cate informatii au fost inserate? Ce fel de subcereri ati folosit? Dati commit. Ce se intampla daca apoi apelam rollback?
*/

insert into emp_pam3
select * from employees where salary > (select salary as Salary_PAM
from employees
where first_name like 'Adam'); -- 31 de rezultate

commit;
rollback;

-- Pentru a rezolva aceasta cerere am facut o subcerere care selecteaza salariul angajatului cu numele Adam
-- si apoi am selectat toti angajatii care au salariul mai mare decat salariul din subcerere.
-- Am folosit subcereri necorelate. Dupa ce am dat commit am permanentizat modificarile efectuate. Daca dupa ultimul commit
-- dam si un rollback nu se va modifica nimic pentru ca aceasta comanda se intoarce la ultima vesiune comisa renuntand
-- la modificarile ulterioare si nu au mai fost facute modificari dupa commit.

```

Script Output x Task completed in 0.249 seconds

31 rows inserted.

Commit complete.

Rollback complete.

Exercitiul 3

select * from emp_pam3
where phone_number like '011%' and phone_number like '%66'; -- 1 rezultat
-- Selectam angajatul al carui numar incepe cu 011 si se termina cu 66 si
-- are orice caractere intre. Cautarea se face folosind functia like
-- cu caracterul % ce semnifica oricate caractere intre cele specificate

```

/*
Afisati informatii complete din tabela emp_abc3 despre angajatului al carui numar de telefon
incepe cu 011 si se termina cu 66. Modificati informatiile din tabela emp_abc3
astfel incat angajatii sa lucreze in departamentul sefului firmei,
dar aceste modificari vor fi facute doar pentru salariatii care au salariul mai mic decat
salariul angajatului al carui numar de telefon incepe cu 011 si se termina cu 66.
La acest exercitiu va folositi doar de tabela emp_abc3.
Cate informatii au fost modificate? Ce fel de subcereri ati folosit? Ce se intampla daca apoi
apelam rollback? Permanentizati aceste modificari facute de update-ul vostru.
*/

select * from emp_pam3
where phone_number like '011%' and phone_number like '%66'; -- 1 rezultat
-- Selectam angajatul al carui numar incepe cu 011 si se termina cu 66 si
-- are orice caractere intre. Cautarea se face folosind functia like
-- cu caracterul % ce semnifica oricate caractere intre cele specificate

```

Script Output x Query Result x

All Rows Fetched: 1 in 0.028 seconds

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
1	175Alyssa	Hutton	AHUTTON	011.44.1644.429266	19-MAR-97	SA_REP	8800	0.25	149	80

update emp_pam3

set department_id = (select department_id as Department_PAM

from emp_pam3

where manager_id is null)

where salary < (select salary as Salary_PAM

from emp_pam3

where phone_number like '011%' and phone_number like '%66');

-- 3 randuri modificate

commit;

-- Pentru a modifica id-ul departamentului in care lucreaza unii angajati vom folosi

-- update cu set pentru coloana department_id la care atribuim o subcerere care ia id-ul

-- departamentului sefului. Seful firmei este angajatul care nu are manager si prin urmare manager_id este null.

-- Deci vom face o subcerere in care selectam id-ul departamentului angajatul cu manager_id null si

-- il punem ca department_id pentru unii angajati. Cum nu vrem sa modificam pentru toti angajatii o sa filtram

-- din angajati prin conditia din where care selecteaza salariul angajatului cu numarul de telefon cu proprietatile de mai sus.

-- Au fost modificate 3 informatii. Am folosit subcereri necorelate. Daca apelam rollback modificarile

-- facute de la ultimul commit vor disparea(adica cele trei modificari facute de update). Daca nu dam rollback si dam commit

-- o sa permanentizam modificarile aduse pana acum.

```
update emp_pam3
set department_id = (select department_id as Department_PAM
                    from emp_pam3
                    where manager_id is null)
where salary < (select salary as Salary_PAM
               from emp_pam3
               where phone_number like '011%' and phone_number like '%66'); -- 3 randuri modificate
commit;
```

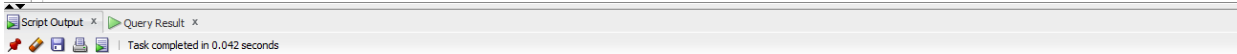
```
-- Pentru a modifica id-ul departamentului in care lucreaza unii angajati vom folosi
-- update cu set pentru coloana department_id la care atribuim o subcerere care ia id-ul
-- departamentului sefului. Seful firmei este angajatul care nu are manager si prin urmare manager_id este null.
-- Deci vom face o subcerere in care selectam id-ul departamentului angajatul cu manager_id null si
-- il punem ca department_id pentru unii angajati. Cum nu vrem sa modificam pentru toti angajatii o sa filtram
-- din angajati prin conditia din where care selecteaza salariul angajatului cu numarul de telefon cu proprietatile de mai sus.
-- Au fost modificate 3 informatii. Am folosit subcereri necorelate. Daca apelam rollback modificarile
-- facute de la ultimul commit vor disparea(adica cele trei modificari facute de update). Daca nu dam rollback si dam commit
-- o sa permanentizam modificarile aduse pana acum.
```

Exercitiul 4

```
delete from emp_pam3
where hire_date > (select distinct greatest((select hire_date as Hire_Date_PAM
                                           from emp_pam3
                                           where hire_date < (select hire_date Hire_Date_PAM
                                                             from emp_pam3
                                                             where lower(last_name)='kochhar'))))
from emp_pam3); -- 30 de informatii sterse
-- Pentru a selecta cea mai mare data mai mica decat data de angajare a lui Kochhar
-- vom selecta toate datele mai mici si o vom pastra doar pe cea mai mare folosind
-- functia greatest. Vom pune si distinct pentru a nu returna mai multe valori egale.
-- Pentru a selecta aceste date e nevoie de o subcerere care selecteaza
-- data de angajare a lui Kochhar si apoi de inca una care selecteaza datele mai mici ca aceasta.
-- Au fost 30 de informatii sterse. Au fost folosite subcereri necorelate.
```

```
/*
Sa se stearga din tabela emp_abc3 toti salariatii angajati dupa salariatul angajat inaintea lui Kochhar
(acesta este numele de familie). La acest exercitiu va folositi doar de tabela emp_abc3.
Cate informatii au fost sterse? Ce fel de subcereri ati folosit?
*/

delete from emp_pam3
where hire_date > (select distinct greatest((select hire_date as Hire_Date_PAM
                                           from emp_pam3
                                           where hire_date < (select hire_date Hire_Date_PAM
                                                             from emp_pam3
                                                             where lower(last_name)='kochhar'))))
from emp_pam3); -- 30 de informatii sterse
-- Pentru a selecta cea mai mare data mai mica decat data de angajare a lui Kochhar
-- vom selecta toate datele mai mici si o vom pastra doar pe cea mai mare folosind
-- functia greatest. Vom pune si distinct pentru a nu returna mai multe valori egale.
-- Pentru a selecta aceste date e nevoie de o subcerere care selecteaza
-- data de angajare a lui Kochhar si apoi de inca una care selecteaza datele mai mici ca aceasta.
-- Au fost 30 de informatii sterse. Au fost folosite subcereri necorelate.
```



30 rows deleted.

Exercitiul 5

```
select * from emp_pam3; -- un singur rezultat ramas
-- Pentru a afisa informatiile ramase facem un select la toate coloanele din tabela. A ramas un
```

-- singur rezultat in tabel.

```
/*
Afisati informatiile ramase in tabela emp_abc3. In final stergeti tabla emp_abc3.
*/
select * from emp_pam3; -- un rezultat ramas

drop table emp_pam3;
-- Pentru a afisa informatiile ramase facem un select la tabela. Am ramas cu un singur
```

Script Output x Query Result x

 All Rows Fetched: 1 in 0.012 seconds

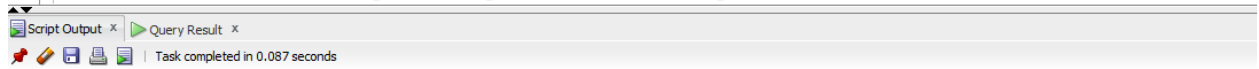
EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID	
1	100	Steven	King	SKING 515.123.4567	17-JUN-87	AD	PRES	24000	(null)	(null)	90

drop table emp_pam3;

-- Apoi stergem tabela cu drop table.

```
/*
Afisati informatiile ramase in tabela emp_abc3. In final stergeti tabla emp_abc3.
*/
select * from emp_pam3; -- un rezultat ramas

drop table emp_pam3;
-- Pentru a afisa informatiile ramase facem un select la tabela. Am ramas cu un singur
-- rezultat in tabel. Apoi stergem tabela cu drop table.
```



The screenshot shows the SQL Developer interface with a 'Task completed' message. The message indicates that the task was completed in 0.087 seconds.

Task completed in 0.087 seconds
Task completed in 0.087 seconds

Table EMP_PAM3 dropped.