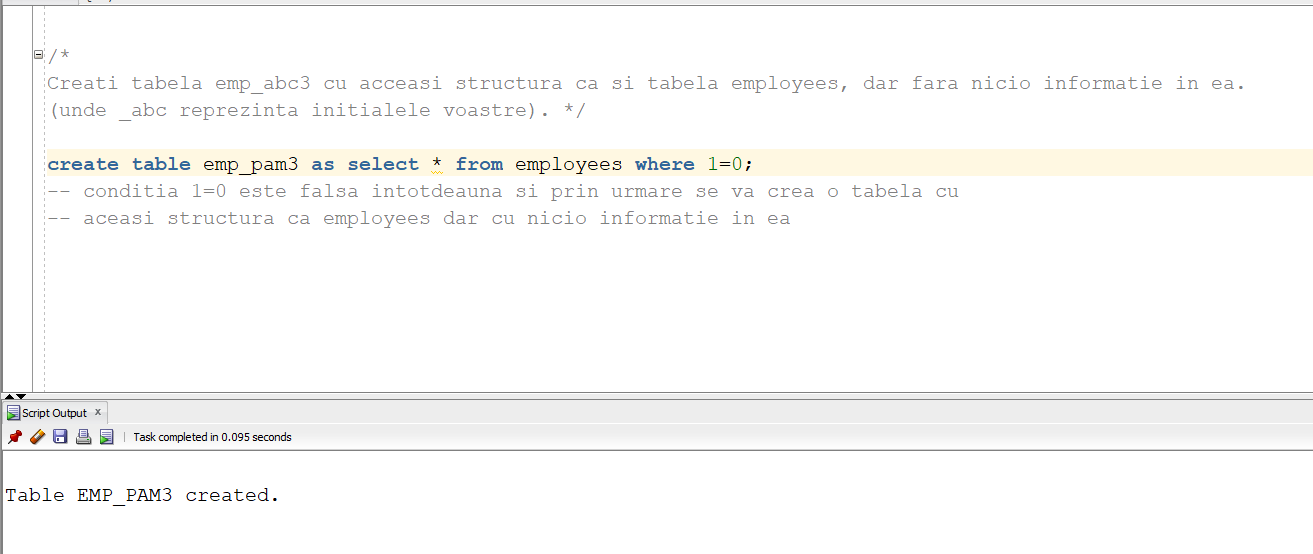
# Tema 2

## 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



## 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;

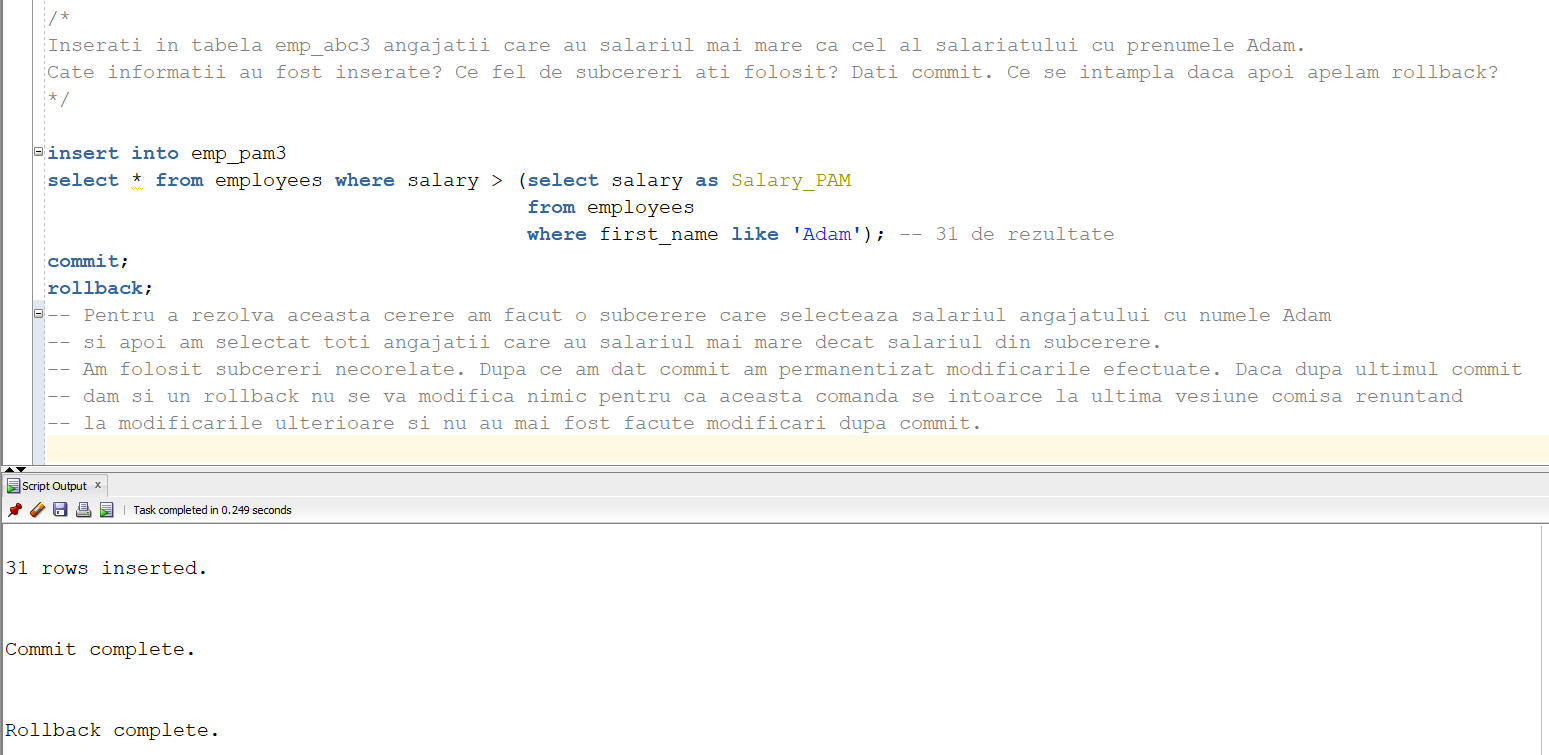
-- Pentru a rezolva aceasta cerere am facut o subcerere care selecteaza salariul angajatului cu numele Adam

-- 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 vesiune comisa renuntand

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



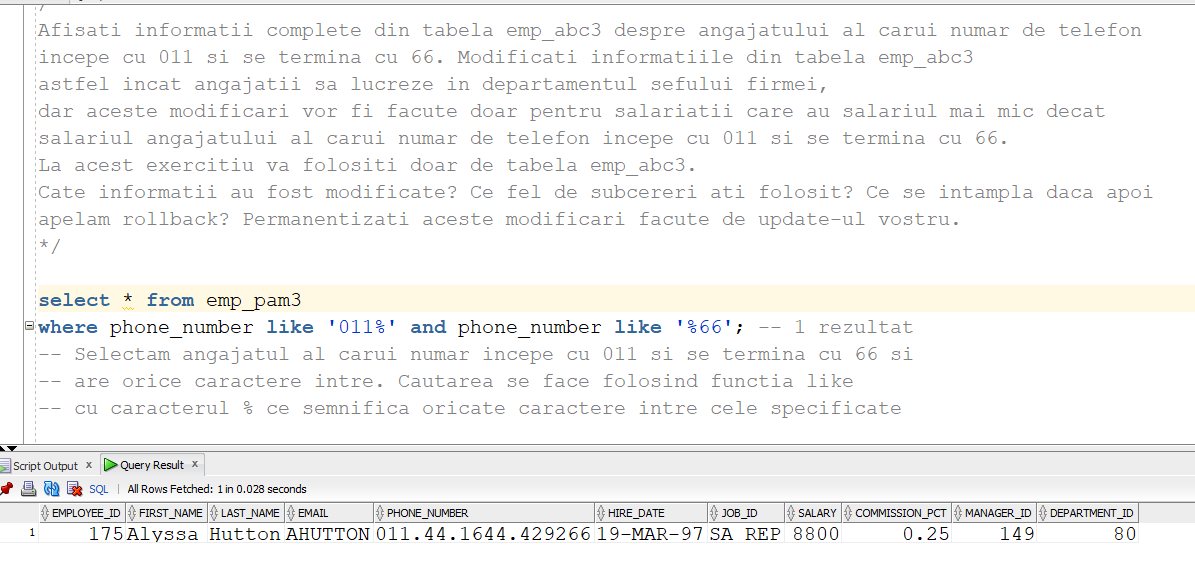
## Exercitiul 3

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

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

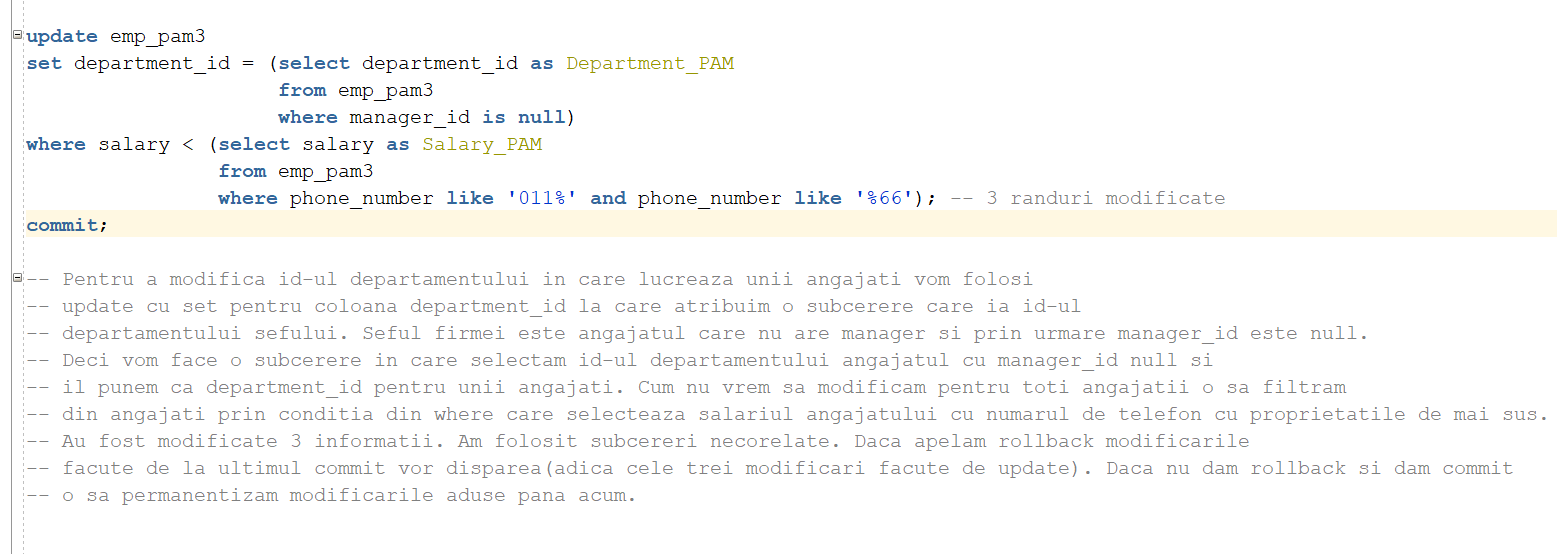
-- il punem ca department\_id pentru unii angajati. Cum nu vrem sa modificam pentru toti angajatii o sa filtram

-- 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

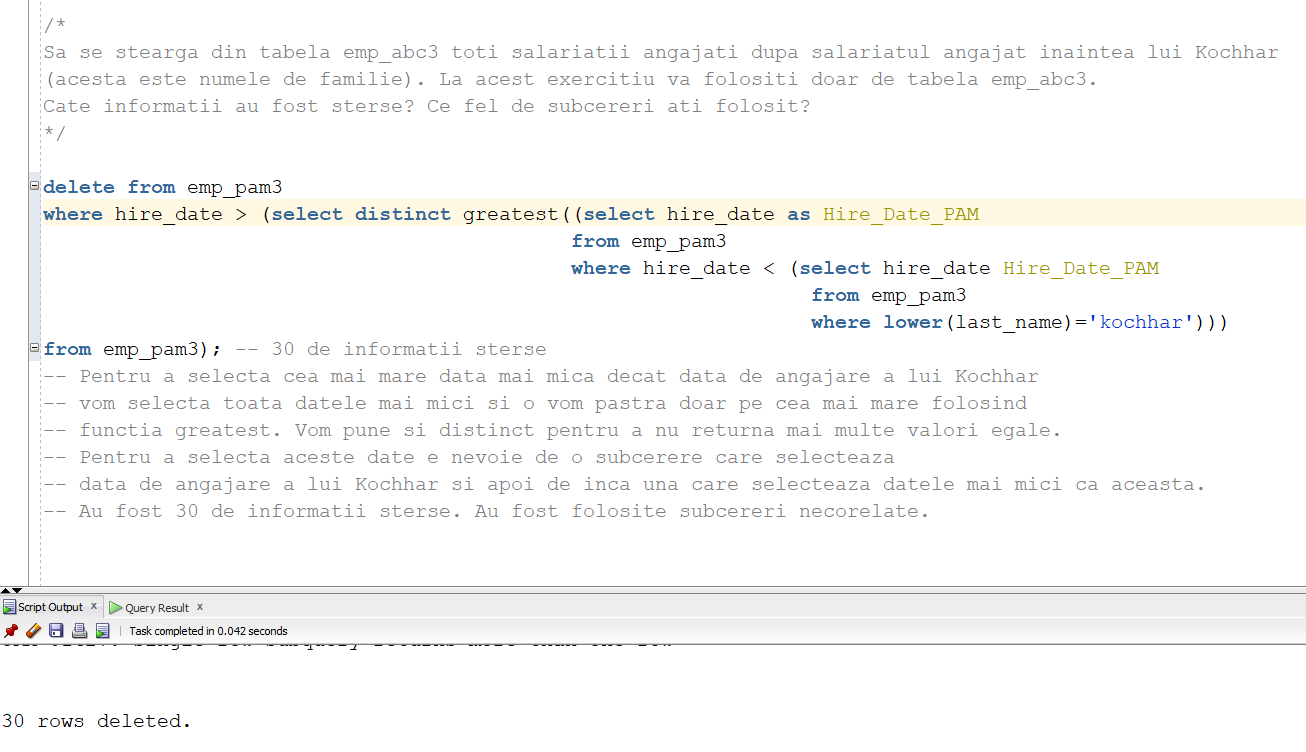
-- vom selecta toata datele mai mici si o vom pastra doar pe cea mai mare folosind

-- 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.

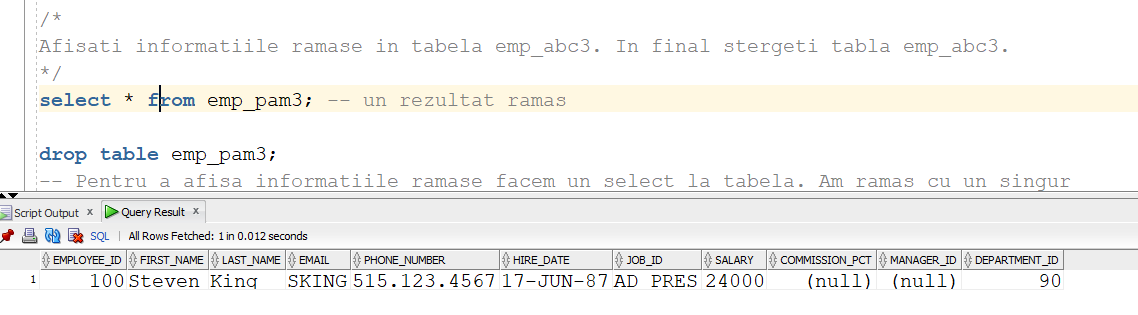
-- Au fost 30 de informatii sterse. Au fost folosite subcereri necorelate.



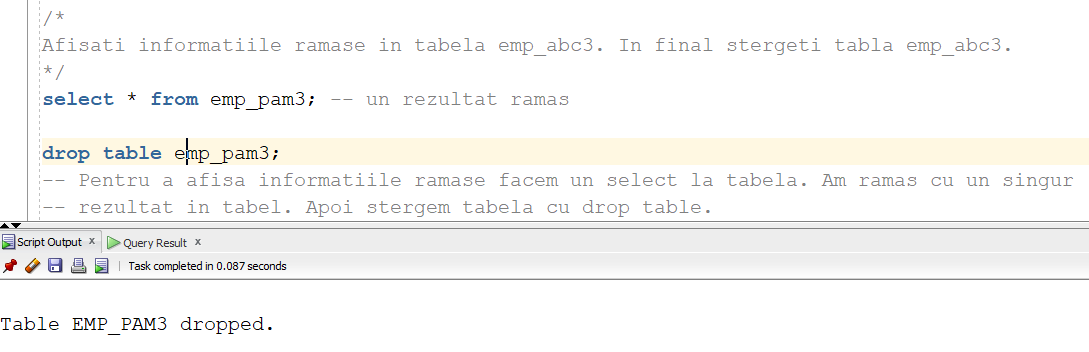
## Exercitiul 5

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.

drop table emp\_pam3;

-- Apoi stergem tabela cu drop table.