# DBMS Laboratory

# UE19CS304

# 5th Semester, Academic Year 2021-22

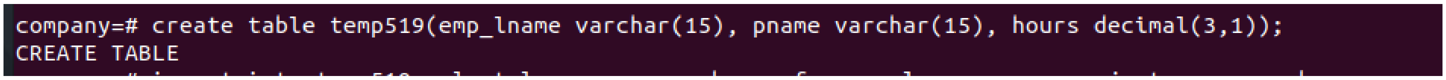
Week #: 5 - SQL DML and Transactions

Date: 9/10/2021

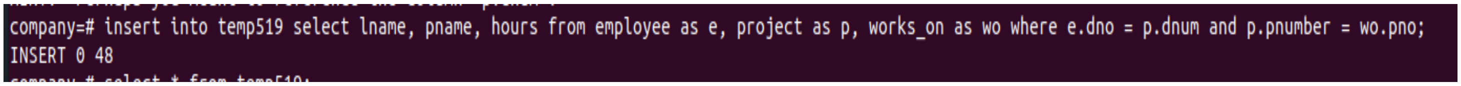
| | | |
|---|---|---|
| Name :<br>SUMUKH RAJU BHAT | SRN :<br>PES1UG19CS519 | Section :<br>H |

## 1. SQL DML

- The given files are loaded using psql -f command and then a temporary table is created in company database called “temp519” with attributes from employee, project and works_on table that are already existing by using select as a subquery in insert.

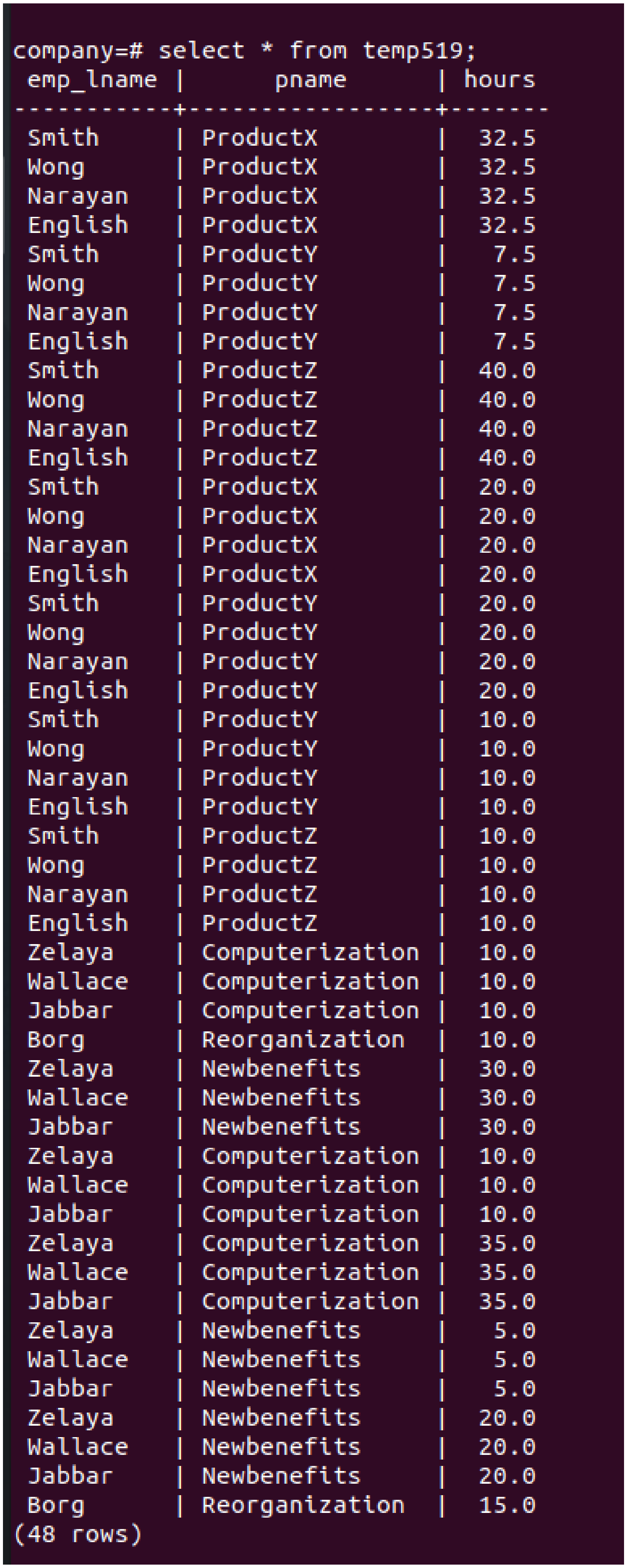


```
company=# create table temp519(emp_lname varchar(15), pname varchar(15), hours decimal(3,1));
CREATE TABLE
```



```
company=# insert into temp519 select lname, pname, hours from employee as e, project as p, works_on as wo where e.dno = p.dnum and p.pnumber = wo.pno;
INSERT 0 48
```

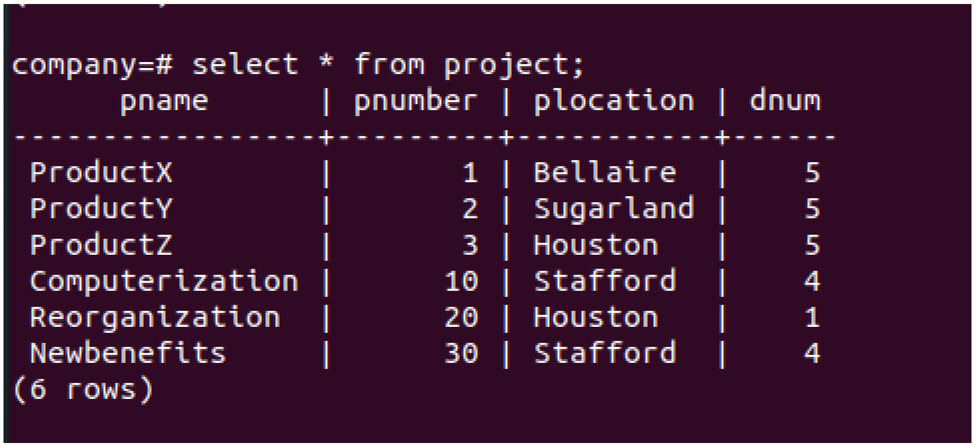
Inserted values:



```
company=# select * from temp519;
 emp_lname |      pname      | hours
-----------+-----------------+-------
 Smith     | ProductX        |  32.5
 Wong      | ProductX        |  32.5
 Narayan   | ProductX        |  32.5
 English   | ProductX        |  32.5
 Smith     | ProductY        |   7.5
 Wong      | ProductY        |   7.5
 Narayan   | ProductY        |   7.5
 English   | ProductY        |   7.5
 Smith     | ProductZ        |  40.0
 Wong      | ProductZ        |  40.0
 Narayan   | ProductZ        |  40.0
 English   | ProductZ        |  40.0
 Smith     | ProductX        |  20.0
 Wong      | ProductX        |  20.0
 Narayan   | ProductX        |  20.0
 English   | ProductX        |  20.0
 Smith     | ProductY        |  20.0
 Wong      | ProductY        |  20.0
 Narayan   | ProductY        |  20.0
 English   | ProductY        |  20.0
 Smith     | ProductY        |  10.0
 Wong      | ProductY        |  10.0
 Narayan   | ProductY        |  10.0
 English   | ProductY        |  10.0
 Smith     | ProductZ        |  10.0
 Wong      | ProductZ        |  10.0
 Narayan   | ProductZ        |  10.0
 English   | ProductZ        |  10.0
 Zelaya    | Computerization |  10.0
 Wallace   | Computerization |  10.0
 Jabbar    | Computerization |  10.0
 Borg      | Reorganization  |  10.0
 Zelaya    | Newbenefits     |  30.0
 Wallace   | Newbenefits     |  30.0
 Jabbar    | Newbenefits     |  30.0
 Zelaya    | Computerization |  10.0
 Wallace   | Computerization |  10.0
 Jabbar    | Computerization |  10.0
 Zelaya    | Computerization |  35.0
 Wallace   | Computerization |  35.0
 Jabbar    | Computerization |  35.0
 Zelaya    | Newbenefits     |   5.0
 Wallace   | Newbenefits     |   5.0
 Jabbar    | Newbenefits     |   5.0
 Zelaya    | Newbenefits     |  20.0
 Wallace   | Newbenefits     |  20.0
 Jabbar    | Newbenefits     |  20.0
 Borg      | Reorganization  |  15.0
(48 rows)
```

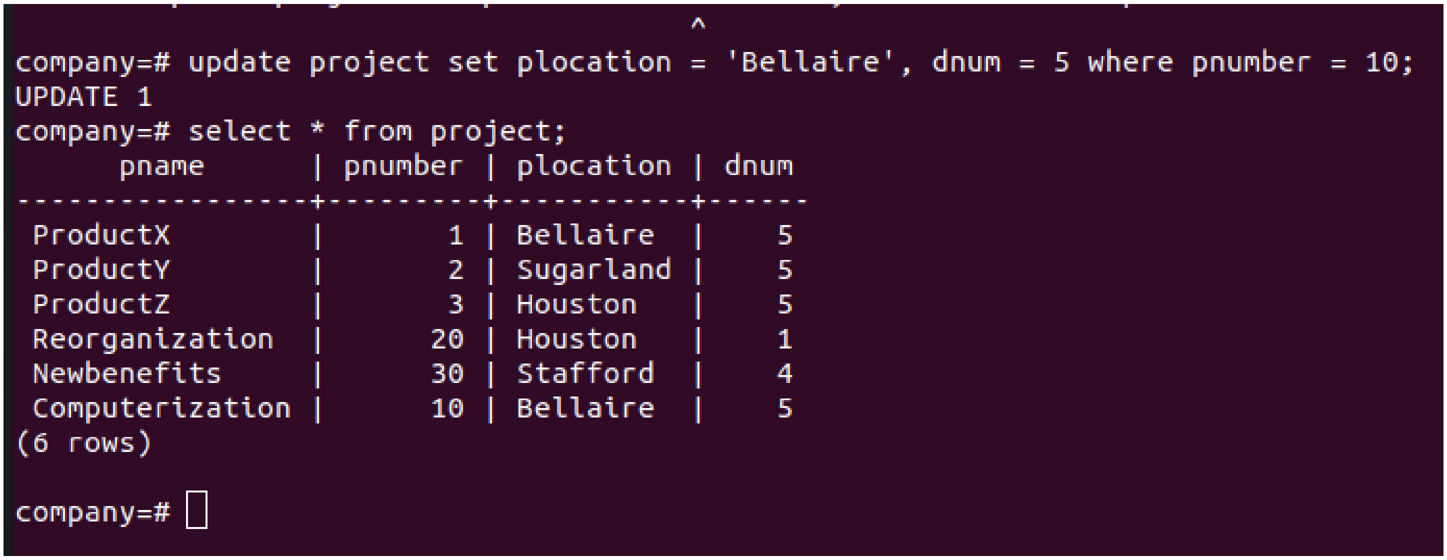
- Update project table where pnumber = 10 using update command:

Before updating:



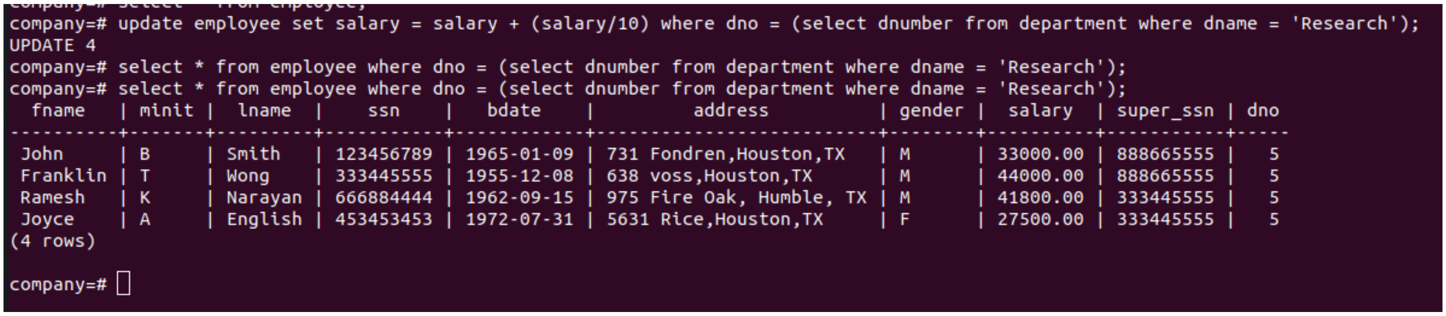
```
company=# select * from project;
      pname      | pnumber | plocation | dnum
-----------------+---------+-----------+------
 ProductX        |       1 | Bellaire  |    5
 ProductY        |       2 | Sugarland |    5
 ProductZ        |       3 | Houston   |    5
 Computerization |      10 | Stafford  |    4
 Reorganization  |      20 | Houston   |    1
 Newbenefits     |      30 | Stafford  |    4
(6 rows)
```

Updating the required entries by using the following query, we get the updated table as:



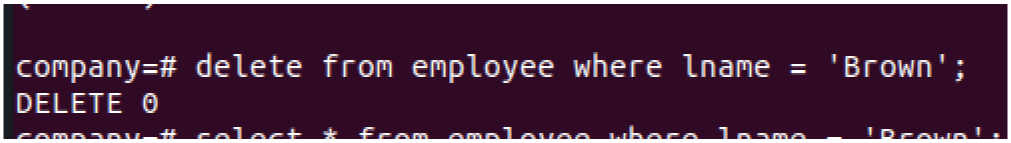
```
company=# update project set plocation = 'Bellaire', dnum = 5 where pnumber = 10;
UPDATE 1
company=# select * from project;
      pname      | pnumber | plocation | dnum
-----------------+---------+-----------+------
 ProductX        |       1 | Bellaire  |    5
 ProductY        |       2 | Sugarland |    5
 ProductZ        |       3 | Houston   |    5
 Reorganization  |      20 | Houston   |    1
 Newbenefits     |      30 | Stafford  |    4
 Computerization |      10 | Bellaire  |    5
(6 rows)

company=# 
```

- Giving all the employees in 10% raise in Research dept using update command with select as subquery. We get the following updates:



```
company=# select * from employee;
company=# update employee set salary = salary + (salary/10) where dno = (select dnumber from department where dname = 'Research');
UPDATE 4
company=# select * from employee where dno = (select dnumber from department where dname = 'Research');
company=# select * from employee where dno = (select dnumber from department where dname = 'Research');
  fname   | minit |  lname  |    ssn    |   bdate    |         address          | gender |  salary  | super_ssn | dno
----------+-------+---------+-----------+------------+--------------------------+--------+----------+-----------+-----
 John     | B     | Smith   | 123456789 | 1965-01-09 | 731 Fondren,Houston,TX   | M      | 33000.00 | 888665555 |   5
 Franklin | T     | Wong    | 333445555 | 1955-12-08 | 638 voss,Houston,TX      | M      | 44000.00 | 888665555 |   5
 Ramesh   | K     | Narayan | 666884444 | 1962-09-15 | 975 Fire Oak, Humble, TX | M      | 41800.00 | 333445555 |   5
 Joyce    | A     | English | 453453453 | 1972-07-31 | 5631 Rice,Houston,TX     | F      | 27500.00 | 333445555 |   5
(4 rows)

company=# 
```

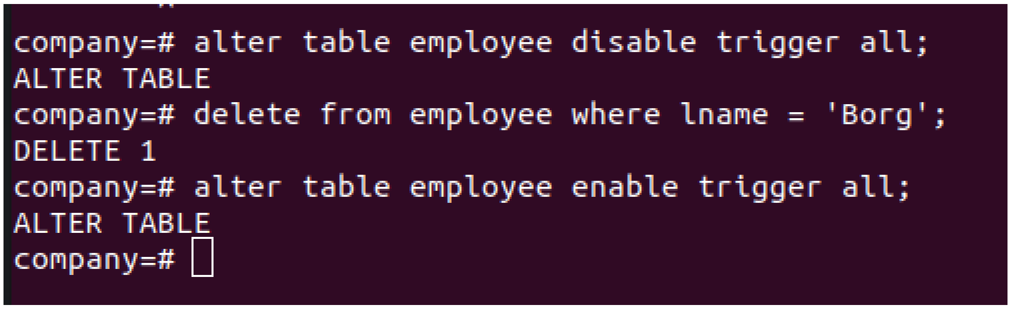
- Deleting entries in employee where lname = Brown



```
company=# delete from employee where lname = 'Brown';
DELETE 0
company=# select * from employee where lname = 'Brown';
```

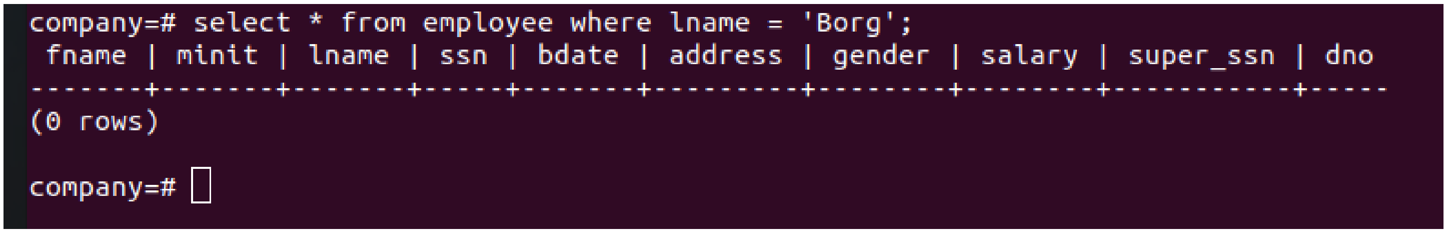
It is evident there are no such entries!

So I have demonstrated by deleting entries in employee where lname = Borg



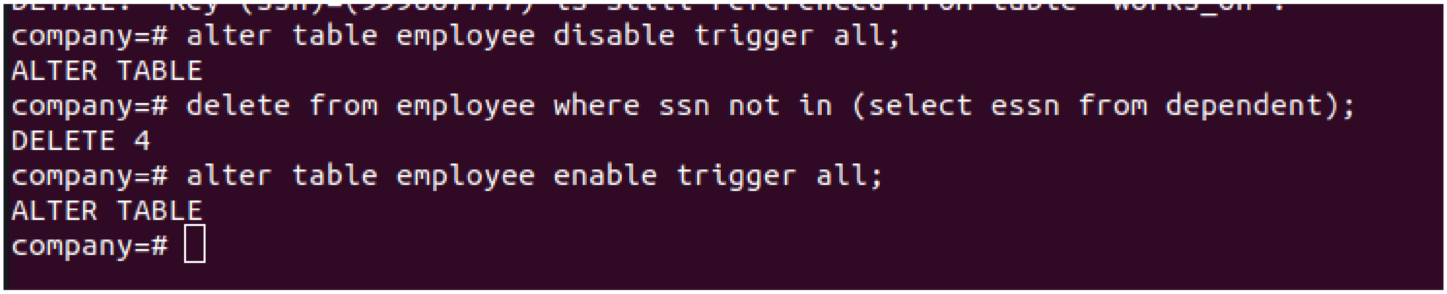
```
company=# alter table employee disable trigger all;
ALTER TABLE
company=# delete from employee where lname = 'Borg';
DELETE 1
company=# alter table employee enable trigger all;
ALTER TABLE
company=# 
```

We will have referential integrity or fk constraint violation if we simply use delete command. So we can disable all triggers before deleting, delete the entries and enable it back. After deletion is successfull we can see that no entries exist with lname = Borg:



```
company=# select * from employee where lname = 'Borg';
 fname | minit | lname | ssn | bdate | address | gender | salary | super_ssn | dno
-------+-------+-------+-----+-------+---------+--------+--------+-----------+-----
(0 rows)

company=# 
```

- ◆ Deleting all employees who don’t have dependents using select as subquery.

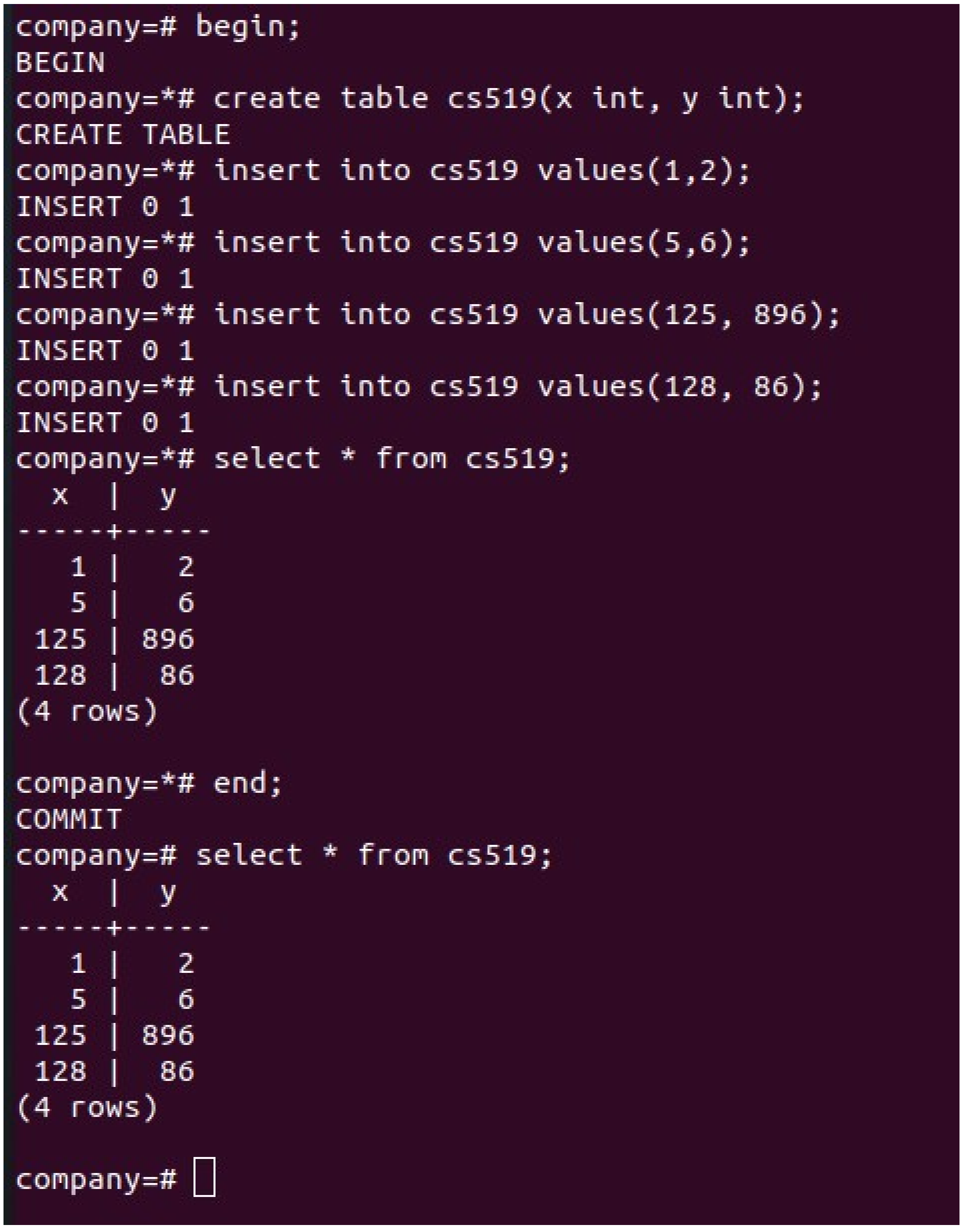


```
DETAIL:  Key (ssn)=(999887777) is still referenced from table "works_on".
company=# alter table employee disable trigger all;
ALTER TABLE
company=# delete from employee where ssn not in (select essn from dependent);
DELETE 4
company=# alter table employee enable trigger all;
ALTER TABLE
company=# 
```

Again we will have referential integrity or fk constraint violation if we simply use delete command. So we can disable all triggers before deleting, delete the entries and enable it back.

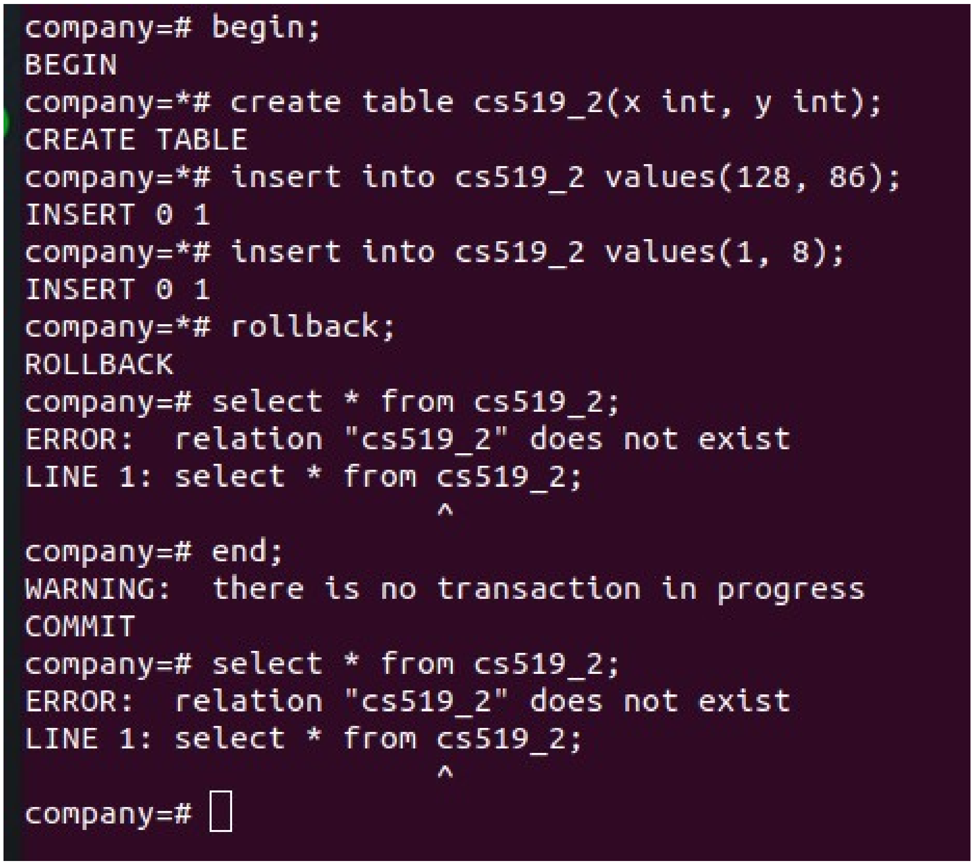
## 2. Transcations

- We create a transaction with creating a table called cs519, inserting 4 tuples to it and ending it and hence commiting it. It is evident that after commiting, the changes are reflected in original db as well:



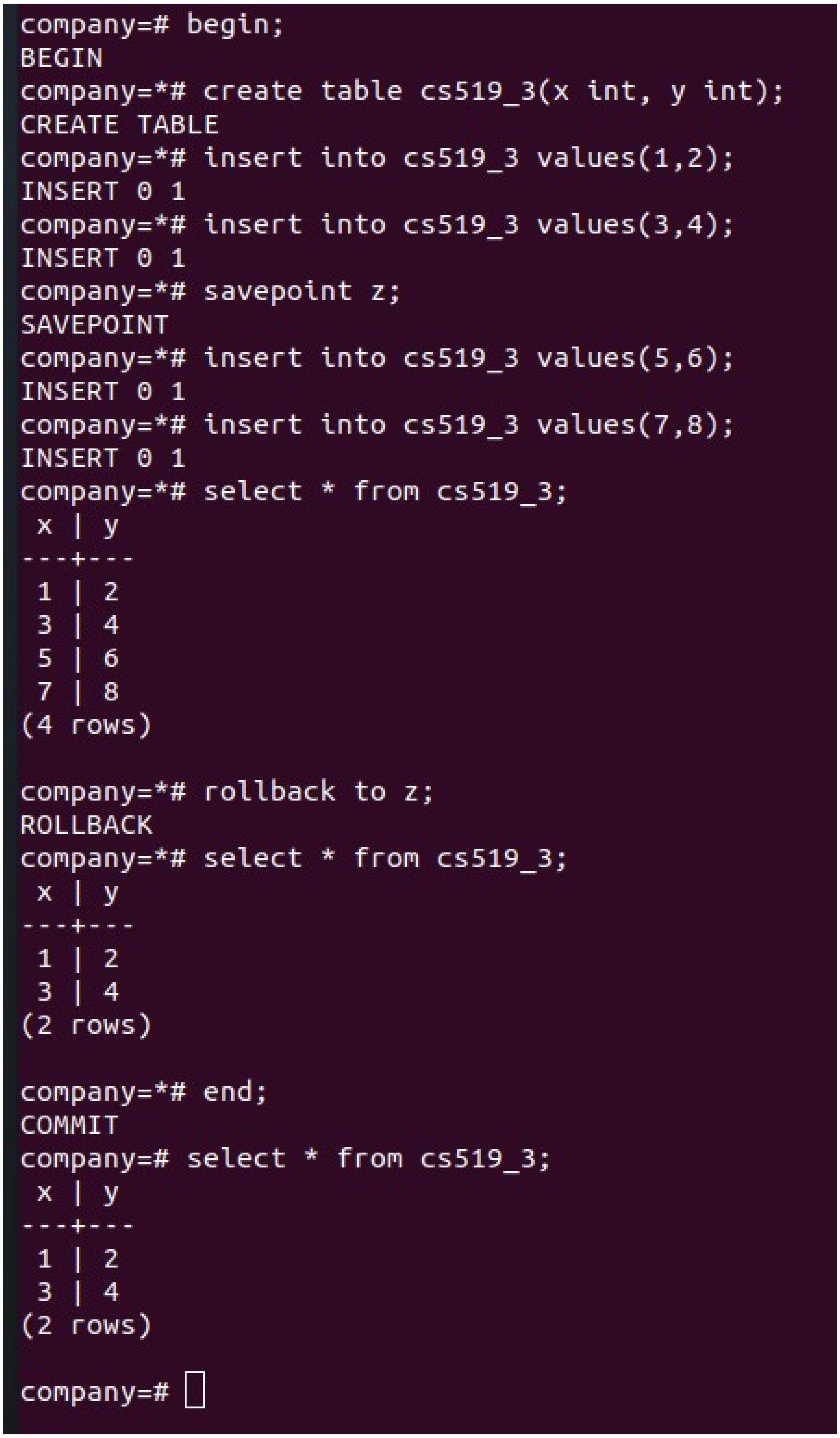
```
company=# begin;
BEGIN
company=*# create table cs519(x int, y int);
CREATE TABLE
company=*# insert into cs519 values(1,2);
INSERT 0 1
company=*# insert into cs519 values(5,6);
INSERT 0 1
company=*# insert into cs519 values(125, 896);
INSERT 0 1
company=*# insert into cs519 values(128, 86);
INSERT 0 1
company=*# select * from cs519;
  x  |  y
-----+-----
   1 |   2
   5 |   6
 125 | 896
 128 |  86
(4 rows)

company=*# end;
COMMIT
company=# select * from cs519;
  x  |  y
-----+-----
   1 |   2
   5 |   6
 125 | 896
 128 |  86
(4 rows)

company=# 
```

- Creating a transaction by creating a table called cs519_2, inserting 2 tuples to it and reverting all the changes. On reverting we see that neither create nor inserts are reflected. Even after ending the transaction and hence committing the changes, we see that the db doesn’t exist as we had reverted back and hence we got an error on using select command.



```
company=# begin;
BEGIN
company=*# create table cs519_2(x int, y int);
CREATE TABLE
company=*# insert into cs519_2 values(128, 86);
INSERT 0 1
company=*# insert into cs519_2 values(1, 8);
INSERT 0 1
company=*# rollback;
ROLLBACK
company=# select * from cs519_2;
ERROR:  relation "cs519_2" does not exist
LINE 1: select * from cs519_2;
                      ^
company=# end;
WARNING:  there is no transaction in progress
COMMIT
company=# select * from cs519_2;
ERROR:  relation "cs519_2" does not exist
LINE 1: select * from cs519_2;
                      ^
company=# 
```

- Creating a transaction by creating a table called cs519_3 and inserting 2 tuples. Until this point a savepoint called “z” is created. After this, we add 2 more tuples to the table. Now on reverting to “z”, we can see only the 2 tuples inserted in the start. Even on ending the transaction and hence committing the transaction, we only see 2 tuples in the table instead of 4.



```
company=# begin;
BEGIN
company=*# create table cs519_3(x int, y int);
CREATE TABLE
company=*# insert into cs519_3 values(1,2);
INSERT 0 1
company=*# insert into cs519_3 values(3,4);
INSERT 0 1
company=*# savepoint z;
SAVEPOINT
company=*# insert into cs519_3 values(5,6);
INSERT 0 1
company=*# insert into cs519_3 values(7,8);
INSERT 0 1
company=*# select * from cs519_3;
 x | y
---+---
 1 | 2
 3 | 4
 5 | 6
 7 | 8
(4 rows)

company=*# rollback to z;
ROLLBACK
company=*# select * from cs519_3;
 x | y
---+---
 1 | 2
 3 | 4
(2 rows)

company=*# end;
COMMIT
company=# select * from cs519_3;
 x | y
---+---
 1 | 2
 3 | 4
(2 rows)

company=# 
```