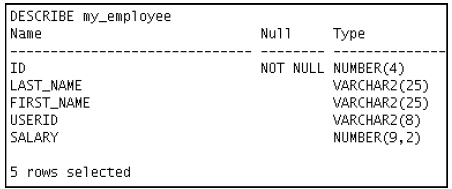
1) Run the statement in the lab\_09\_01.sql script to build the MY\_EMPLOYEE table

used in this practice.

2) Describe the structure of the MY\_EMPLOYEE table to identify the column names.



Ans. create table my\_employee

(id number(6) not null primary key,

last\_name varchar2(25),

first\_name varchar2(25),

user\_id varchar2(8),

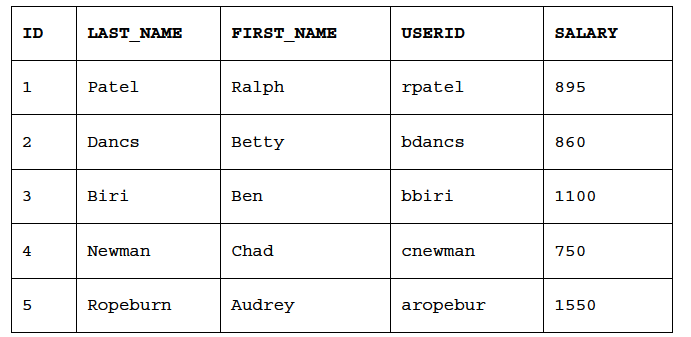
salary number(9,2));

desc my\_employee;

3) Create an INSERT statement to add the *first row* of data to the MY\_EMPLOYEE table

from the following sample data. Do not list the columns in the INSERT clause. *Do not*

*enter all rows yet.*



Insert into my\_employee

Values(1,’Patel’,’Ralph’,’rpatel’,895);

4) Populate the MY\_EMPLOYEE table with the second row of the sample data from the

preceding list. This time, list the columns explicitly in the INSERT clause.

Ans. Insert into my\_employee

values(&id,'&first\_name','&last\_name','&user\_id',salary);

5) Confirm your addition to the table.



select \*

from my\_employee;

6) Write an INSERT statement in a dynamic reusable script file to load the remaining

rows into the MY\_EMPLOYEE table. The script should prompt for all the columns (ID,

LAST\_NAME, FIRST\_NAME, USERID, and SALARY). Save this script to a

lab\_09\_06.sql file.

Ans. Insert into my\_employee

values(&id,'&first\_name','&last\_name','&user\_id',salary);

7) Populate the table with the next two rows of the sample data listed in step 3 by

running the INSERT statement in the script that you created.

Ans. Insert into my\_employee

values(&id,'&first\_name','&last\_name','&user\_id',salary);

8) Confirm your additions to the table.



Ans.

SELECT\*

FROM my\_employee;

9) Make the data additions permanent.

Ans . commit;

**Update and delete data in the MY\_EMPLOYEE table.**

10) Change the last name of employee 3 to Drexler.

UPDATE my\_employee

SET last\_name=’Drexler’

WHERE id=3;

11) Change the salary to $1,000 for all employees who have a salary less than $900.

UPDATE my\_employee

SET salary=1000

WHERE salary<900;

12) Verify your changes to the table.

Ans. Select \*

From my\_employee;

13) Delete Betty Dancs from the MY\_EMPLOYEE table

Delete from my\_employee

WHERE id=2;

14) Confirm your changes to the table.



Ans.

. Select \*

From my\_employee;

15) Commit all pending changes.

Commit;

**Control data transaction to the MY\_EMPLOYEE table.**

16) Populate the table with the last row of the sample data listed in step 3 by using the

statements in the script that you created in step 6. Run the statements in the script.

Ans. INSERT INTO my\_employee

values(&id, 'last\_name', '&first\_name','&user\_id', &salary);

17) Confirm your addition to the table.



Ans. Select \*

From my\_employee;

18) Mark an intermediate point in the processing of the transaction.

Ans. Select \*

From my\_employee

savepoint a;

19) Delete all the rows from the MY\_EMPLOYEE table.

delete from my\_employee;

20) Confirm that the table is empty.

Select \*

From my\_employee;

21) Discard the most recent DELETE operation without discarding the earlier INSERT

operation.

Rollback a;

22) Confirm that the new row is still intact.



Ans.

Select \*

From my\_employee;

23) Make the data addition permanent.

If you have the time, complete the following exercise:

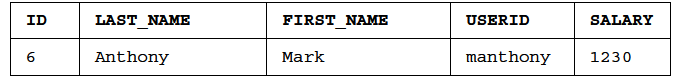
Commit;

24) Modify the lab\_09\_06.sql script such that the USERID is generated automatically

by concatenating the first letter of the first name and the first seven characters of the

last name. The generated USERID must be in lowercase. Therefore, the script should

not prompt for the USERID. Save this script to a file named lab\_09\_24.sql.



Ans.

INSERT INTO my\_employee

VALUES (&id, '&&last\_name', '&&first\_name',lower(substr('&first\_name', 1, 1) ||substr('&last\_name', 1, 7)), &salary);

26) Confirm that the new row was added with correct USERID.



Select \*

From my\_employee;