



# University of Chittagong

Department of Computer Science & Engineering

Database Systems Lab

Name of the assignment:

**Chapters 8, 9, 10 & 18 Practice Problems**

CSE 414

Assignment 04

Submitted By:

**Debashish Chakraborty**

ID: 23701034

Submitted To:

**Dr. Rudra Pratap Deb**

**Nath**

Associate Professor

June 14, 2025

## Contents

<b>1</b>	<b>Practice 8 (Solutions)</b>	<b>2</b>
<b>2</b>	<b>Practice 9 (Solutions)</b>	<b>9</b>
<b>3</b>	<b>Practice 10 (Solutions)</b>	<b>14</b>
<b>4</b>	<b>Practice 18 (Solutions)</b>	<b>16</b>

## Practice 8 (Solutions)

- *INSERT data into the MY\_EMPLOYEE table.*

### 8.1

**Problem:** Run the statement in the lab8\_1.sql script to build the MY\_EMPLOYEE table that will be used for the lab.

**Solution:**

```
1 CREATE TABLE my_employee (  
2     ID NUMBER(4)  
3     CONSTRAINT MY_EMPLOYEE_ID_NN NOT NULL,  
4     LAST_NAME VARCHAR2(25),  
5     FIRST_NAME VARCHAR2(25),  
6     USERID VARCHAR2(8),  
7     SALARY NUMBER(9, 2)  
8 );
```

### 8.2

**Problem:** Describe the structure of the MY\_EMPLOYEE table to identify the column names.

Name	Null?	Type
ID	NOT NULL	NUMBER(4)
LAST_NAME		VARCHAR2(25)
FIRST_NAME		VARCHAR2(25)
USERID		VARCHAR2(8)
SALARY		NUMBER(9,2)

**Solution:**

```
1 DESCRIBE my_employee;
```

### 8.3

**Problem:** 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.

**Solution:**

```
1 INSERT INTO MY_EMPLOYEE VALUES ( 1,  
2     'Patel',  
3     'Ralph',  
4     'rpatel',  
5     895 );
```

ID	LAST_NAME	FIRST_NAME	USERID	SALARY
1	Patel	Ralph	rpatel	895
2	Dancs	Betty	bdancs	860
3	Biri	Ben	bbiri	1100
4	Newman	Chad	cnewman	750
5	Ropeburn	Audrey	aropebur	1550

## 8.4

**Problem:** Populate the MY\_EMPLOYEE table with the second row of sample data from the preceding list. This time, list the columns explicitly in the INSERT clause.

**Solution:**

```
1 INSERT INTO MY_EMPLOYEE (  
2     ID ,  
3     LAST_NAME ,  
4     FIRST_NAME ,  
5     USERID ,  
6     SALARY  
7 ) VALUES ( 2 ,  
8     'Dancs' ,  
9     'Betty' ,  
10    'bdancs' ,  
11    860 );
```

## 8.5

**Problem:** Confirm your addition to the table.

ID	LAST_NAME	FIRST_NAME	USERID	SALARY
1	Patel	Ralph	rpatel	895
2	Dancs	Betty	bdancs	860

**Solution:**

```
1 SELECT  
2     *  
3 FROM  
4     my_employee;
```

## 8.6

**Problem:** Write an insert statement in a text file named loademp.sql to load rows into the MY\_EMPLOYEE table. Concatenate the first letter of the first name and the first seven characters of the last name to produce the userid.

**Solution:**

```
1
2 INSERT INTO my_employee VALUES ( &p_id,
3     '&p_last_name',
4     '&p_first_name',
5     lower(substr('&p_first_name', 1, 1)
6     || substr('&p_last_name', 1, 7)),
7     &p_salary );
```

## 8.7

**Problem:** Populate the table with the next two rows of sample data by running the INSERT statement in the script that you created.

**Solution:**

```
1
2 INSERT INTO my_employee VALUES ( &p_id,
3     '&p_last_name',
4     '&p_first_name',
5     lower(substr('&p_first_name', 1, 1)
6     || substr('&p_last_name', 1, 7)),
7     &p_salary );
```

## 8.8

**Problem:** Confirm your additions to the table.

ID	LAST_NAME	FIRST_NAME	USERID	SALARY
1	Patel	Ralph	rpatel	895
2	Dancs	Betty	bdancs	860
3	Biri	Ben	bbiri	1100
4	Newman	Chad	cnewman	750

**Solution:**

```
1 SELECT
2     *
3 FROM
4     my_employee;
```

## 8.9

**Problem:** Make the data additions permanent. Update and delete data in the MY\_EMPLOYEE table.

**Solution:**

```
1 COMMIT;
```

- *UPDATE and DELETE data in the MY\_EMPLOYEE table.*

## 8.10

**Problem:** Change the last name of employee 3 to Drexler.

**Solution:**

```
1 UPDATE my_employee
2 SET
3     last_name = 'Drexler'
4 WHERE
5     id = 3;
```

## 8.11

**Problem:** Change the salary to 1000 for all employees with a salary less than 900.

**Solution:**

```
1 UPDATE my_employee
2 SET
3     salary = 1000
4 WHERE
5     salary < 900;
```

## 8.12

**Problem:** Verify your changes to the table.

ID	LAST_NAME	FIRST_NAME	USERID	SALARY
1	Patel	Ralph	rpatel	1000
2	Dancs	Betty	bdancs	1000
3	Drexler	Ben	bbiri	1100
4	Newman	Chad	cnewman	1000

**Solution:**

```
1 SELECT
2     last_name ,
3     salary
4 FROM
5     my_employee;
```

## 8.13

**Problem:** Delete Betty Dancs from the MY\_EMPLOYEE table.

**Solution:**

```
1 DELETE FROM my_employee
2 WHERE
3     last_name = 'Dancs';
```

## 8.14

**Problem:** Confirm your changes to the table.

ID	LAST_NAME	FIRST_NAME	USERID	SALARY
1	Patel	Ralph	rpatel	1000
3	Drexler	Ben	bbiri	1100
4	Newman	Chad	cnewman	1000

**Solution:**

```
1 SELECT
2     *
3 FROM
4     my_employee;
```

## 8.15

**Problem:** Commit all pending changes. Control data transaction to the MY\_EMPLOYEE table.

**Solution:**

```
1 COMMIT;
```

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

## 8.16

**Problem:** Populate the table with the last row of sample data by modifying the statements in the script that you created in step 6. Run the statements in the script.

**Solution:**

```
1
2 INSERT INTO my_employee VALUES ( &p_id,
3     '&p_last_name',
4     '&p_first_name',
5     lower(substr('&p_first_name', 1, 1)
6     || substr('&p_last_name', 1, 7)),
7     &p_salary );
```

## 8.17

**Problem:** Confirm your addition to the table.

ID	LAST_NAME	FIRST_NAME	USERID	SALARY
1	Patel	Ralph	rpatel	1000
3	Drexler	Ben	bbiri	1100
4	Newman	Chad	cnewman	1000
5	Ropeburn	Audrey	aropebur	1550

**Solution:**

```
1 SELECT
2     *
3 FROM
4     my_employee;
```

## 8.18

**Problem:** Mark an intermediate point in the processing of the transaction.

**Solution:**

```
1 SAVEPOINT step_18;
```

## 8.19

**Problem:** Empty the entire table.

**Solution:**

```
1 DELETE FROM my_employee;
```



## 8.20

**Problem:** Confirm that the table is empty.

**Solution:**

```
1 SELECT
2     *
3 FROM
4     my_employee;
```

## 8.21

**Problem:** Discard the most recent DELETE operation without discarding the earlier INSERT operation.

**Solution:**

```
1 ROLLBACK TO step_18;
```

## 8.22

**Problem:** Confirm that the new row is still intact.

ID	LAST_NAME	FIRST_NAME	USERID	SALARY
1	Patel	Ralph	rpatel	1000
3	Drexler	Ben	bbiri	1100
4	Newman	Chad	cnewman	1000
5	Ropeburn	Audrey	aropebur	1550

**Solution:**

```
1 SELECT
2     *
3 FROM
4     my_employee;
```

## 8.23

**Problem:** Make the data addition permanent.

**Solution:**

```
1 COMMIT;
```

## Practice 9 (Solutions)

- *Create, Alter, Drop, Rename, Truncate and adding Comment to a table.*

### 9.1

**Problem:** Create the DEPT table based on the following table instance chart. Place the syntax in a script called lab9.1.sql, then execute the statement in the script to create the table. Confirm that the table is created.

Column Name	ID	NAME
Key Type		
Nulls/Unique		
FK Table		
FK Column		
Data type	NUMBER	VARCHAR2
Length	7	25

  

Name	Null?	Type
ID		NUMBER(7)
NAME		VARCHAR2(25)

**Solution:**

```

1 CREATE TABLE dept (
2     id NUMBER(7),
3     name VARCHAR2(25)
4 );
5 -- Verification
6 DESCRIBE dept;
```

### 9.2

**Problem:** Populate the DEPT table with data from the DEPARTMENTS table. Include only columns that you need.

**Solution:**

```

1 INSERT INTO dept
2 SELECT
3     department_id,
4     department_name
5 FROM
6     departments;
```

### 9.3

**Problem:** Create the EMP table based on the following table instance chart. Place the syntax in a script called lab9\_3.sql, and then execute the statement in the script to create the table. Confirm that the table is created.

Column Name	ID	LAST_NAME	FIRST_NAME	DEPT_ID
Key Type				
Nulls/Unique				
FK Table				
FK Column				
Data type	NUMBER	VARCHAR2	VARCHAR2	NUMBER
Length	7	25	25	7

Name	Null?	Type
ID		NUMBER(7)
LAST_NAME		VARCHAR2(25)
FIRST_NAME		VARCHAR2(25)
DEPT_ID		NUMBER(7)

**Solution:**

```

1 CREATE TABLE emp (
2     id number(7),
3     last_name varchar2(25),
4     first_name varchar2(25),
5     dept_id number(7)
6 );
7
8 -- Verification
9 DESCRIBE emp;
```

### 9.4

**Problem:** Modify the EMP table to allow for longer employee last names. Confirm your modification.

Name	Null?	Type
ID		NUMBER(7)
LAST_NAME		VARCHAR2(50)
FIRST_NAME		VARCHAR2(25)
DEPT_ID		NUMBER(7)

**Solution:**

```
1 ALTER TABLE emp MODIFY (  
2     last_name VARCHAR2(50)  
3 );  
4 -- Verification  
5 DESCRIBE emp;
```

**9.5**

**Problem:** Confirm that both the DEPT and EMP tables are stored in the data dictionary. (Hint: USER\_TABLES)

TABLE_NAME
DEPT
EMP

**Solution:**

```
1 SELECT  
2     table_name  
3 FROM  
4     user_tables  
5 WHERE  
6     table_name IN ( 'DEPT', 'EMP' );
```

**9.6**

**Problem:** Create the EMPLOYEES2 table based on the structure of the EMPLOYEES table. Include only the EMPLOYEE\_ID, FIRST\_NAME, LAST\_NAME, SALARY, and DEPARTMENT\_ID columns. Name the columns in your new table ID, FIRST\_NAME, LAST\_NAME, SALARY, and DEPT\_ID, respectively.

**Solution:**

```
1 CREATE TABLE employees2  
2 AS  
3 SELECT  
4     employee_id id,  
5     first_name ,  
6     last_name ,  
7     salary ,  
8     department_id dept_id  
9 FROM  
10    employees;
```

## 9.7

**Problem:** Drop the EMP table.

**Solution:**

```
1 DROP TABLE emp;
```

## 9.8

**Problem:** Rename the EMPLOYEES2 table to EMP.

**Solution:**

```
1 RENAME employees2 TO emp;
```

## 9.9

**Problem:** Add a comment to the DEPT and EMP table definitions describing the tables. Confirm your additions in the data dictionary.

**Solution:**

```
1 COMMENT ON TABLE emp IS
2     'Employee Information';
3
4 COMMENT ON TABLE dept IS
5     'Department Information';
6
7 SELECT
8     *
9 FROM
10     user_tab_comments
11 WHERE
12     table_name = 'DEPT'
13 OR table_name = 'EMP';
```

## 9.10

**Problem:** Drop the FIRST\_NAME column from the EMP table. Confirm your modification by checking the description of the table.

**Solution:**

```
1 ALTER TABLE emp DROP COLUMN first_name;
2 -- Verification
3 DESCRIBE emp;
```

## 9.11

**Problem:** In the EMP table, mark the DEPT\_ID column in the EMP table as UNUSED. Confirm your modification by checking the description of the table.

**Solution:**

```
1 ALTER TABLE emp SET UNUSED ( dept_id );
2 -- Verification
3 DESCRIBE emp;
```

## 9.12

**Problem:** Drop all the UNUSED columns from the EMP table. Confirm your modification by checking the description of the table.

**Solution:**

```
1 ALTER TABLE emp DROP UNUSED COLUMNS;
2 -- Verification
3 DESCRIBE emp;
```

## Practice 10 (Solutions)

- *Creating Constraints like NOT NULL, UNIQUE, PRIMARY KEY, FOREIGN KEY, CHECK etc.*

### 10.1

**Problem:** Add a table-level PRIMARY KEY constraint to the EMP table on the ID column. The constraint should be named at creation. Name the constraint my\_emp\_id\_pk.

*Hint: The constraint is enabled as soon as the ALTER TABLE command executes successfully.*

**Solution:**

```
1 ALTER TABLE emp ADD CONSTRAINT my_emp_id_pk PRIMARY KEY (id);
```

### 10.2

**Problem:** Create a PRIMARY KEY constraint to the DEPT table using the ID column. The constraint should be named at creation. Name the constraint my\_deptid\_pk.

*Hint: The constraint is enabled as soon as the ALTER TABLE command executes successfully.*

**Solution:**

```
1 ALTER TABLE dept ADD CONSTRAINT my_deptid_pk PRIMARY KEY  
  (id);
```

### 10.3

**Problem:** Add a column DEPT\_ID to the EMP table. Add a foreign key reference on the EMP table that ensures that the employee is not assigned to a nonexistent department. Name the constraint my\_emp\_dept\_id\_fk.

**Solution:**

```
1 ALTER TABLE emp ADD (  
2     dept_id NUMBER(7)  
3 );  
4  
5 ALTER TABLE emp  
6 ADD CONSTRAINT my_emp_dept_id_fk FOREIGN KEY ( dept_id )  
7 REFERENCES dept ( id );
```

## 10.4

**Problem:** Confirm that the constraints were added by querying the USER\_CONSTRAINTS view. Note the types and names of the constraints. Save your statement text in a file called lab10.4.sql.

CONSTRAINT_NAME	C
MY_DEPT_ID_PK	P
SYS_C002541	C
MY_EMP_ID_PK	P
MY_EMP_DEPT_ID_FK	R

**Solution:**

```
1 SELECT
2     constraint_name ,
3     constraint_type
4 FROM
5     user_constraints
6 WHERE
7     table_name IN ( 'EMP' , 'DEPT' );
```

## 10.5

**Problem:** Display the object names and types from the USER\_OBJECTS data dictionary view for the EMP and DEPT tables. Notice that the new tables and a new index were created.

**Solution:**

```
1 SELECT
2     object_name ,
3     object_type
4 FROM
5     user_objects
6 WHERE
7     object_name LIKE 'EMP%'
8     OR object_name LIKE 'DEPT%';
```

## 10.6

**Problem:** Modify the EMP table. Add a COMMISSION column of NUMBER data type, precision 2, scale 2. Add a constraint to the commission column that ensures that a commission value is greater than zero.

**Solution:**

```
1 ALTER TABLE emp ADD commission NUMBER(2, 2)
2 CONSTRAINT my_emp_comm_ck CHECK ( commission >= 0 );
```



## Practice 18 (Solutions)

### 18.1

**Problem:** Write a query to display the last name, department number, and salary of any employee whose department number and salary both match the department number and salary of any employee who earns a commission.

LAST_NAME	DEPARTMENT_ID	SALARY
Taylor	80	8600
Zlotkey	80	10500
Abel	80	11000

**Solution:**

```
1 SELECT
2     last_name ,
3     department_id ,
4     salary
5 FROM
6     employees
7 WHERE
8     ( salary, department_id ) IN (
9         SELECT
10             salary, department_id
11         FROM
12             employees
13         WHERE
14             commission_pct IS NOT NULL
15     );
```

### 18.2

**Problem:** Display the last name, department name, and salary of any employee whose salary and commission match the salary and commission of any employee located in location ID 1700.

LAST_NAME	DEPARTMENT_NAME	SALARY
Whalen	Administration	4400
Gietz	Accounting	8300
Higgins	Accounting	12000
Kochhar	Executive	17000
De Haan	Executive	17000
King	Executive	24000

6 rows selected.

**Solution:**

```
1 SELECT
2     last_name ,
3     department_name ,
4     salary
5 FROM
6     employees e,
7     departments d
8 WHERE
9     e.department_id = d.department_id
10    AND ( salary, nvl(commission_pct, 0) ) IN (
11        SELECT
12            salary, nvl(commission_pct, 0)
13        FROM
14            employees e, departments d
15        WHERE
16            e.department_id = d.department_id
17            AND d.location_id = 1700
18    );
```

**18.3**

**Problem:** Create a query to display the last name, hire date, and salary for all employees who have the same salary and commission as Kochhar.

*Note: Do not display Kochhar in the result set.*

LAST_NAME	HIRE_DATE	SALARY
De Haan	13-JAN-93	17000

**Solution:**

```
1 SELECT
2     last_name ,
3     hire_date ,
4     salary
5 FROM
6     employees
7 WHERE
8     ( salary, nvl(commission_pct, 0) ) IN (
9         SELECT
10             salary, nvl(commission_pct, 0)
11         FROM
12             employees
13         WHERE
14             last_name = 'Kochhar'
15     )
16    AND last_name != 'Kochhar';
```

## 18.4

**Problem:** Create a query to display the employees who earn a salary that is higher than the salary of all of the sales managers (JOB\_ID = 'SA\_MAN'). Sort the results on salary from highest to lowest.

LAST_NAME	JOB_ID	SALARY
King	AD_PRES	24000
Kochhar	AD_VP	17000
De Haan	AD_VP	17000
Hartstein	MK_MAN	13000
Higgins	AC_MGR	12000
Abel	SA_REP	11000

6 rows selected.

### Solution:

```
1 SELECT
2     last_name ,
3     job_id ,
4     salary
5 FROM
6     employees
7 WHERE
8     salary > ALL (
9         SELECT
10             salary
11         FROM
12             employees
13         WHERE
14             job_id = 'SA_MAN'
15     )
16 ORDER BY
17     salary DESC;
```

## 18.5

**Problem:** Display the details of the employee ID, last name, and department ID of those employees who live in cities whose name begins with T.

EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID
201	Hartstein	20
202	Fay	20

**Solution:**

```

1 SELECT
2     employee_id ,
3     last_name ,
4     department_id
5 FROM
6     employees
7 WHERE
8     department_id IN (
9         SELECT
10            department_id
11        FROM
12            departments
13        WHERE
14            location_id IN (
15                SELECT
16                    location_id
17                FROM
18                    locations
19                WHERE
20                    city LIKE 'T%'
21            )
22    );

```

**18.6**

**Problem:** Write a query to find all employees who earn more than the average salary in their departments. Display last name, salary, department ID, and the average salary for the department. Sort by average salary. Use aliases for the columns retrieved by the query as shown in the sample output.

ENAME	SALARY	DEPTNO	DEPT_AVG
Mourgos	5800	50	3500
Hunold	9000	60	6400
Hartstein	13000	20	9500
Abel	11000	80	10033.3333
Zlotkey	10500	80	10033.3333
Higgins	12000	110	10150
King	24000	90	19333.3333

**Solution:**

```

1 SELECT
2     e.last_name ename ,
3     e.salary    salary ,
4     e.department_id deptno ,
5     (SELECT AVG(salary) FROM employees WHERE department_id =
6         e.department_id) dept_avg

```

```
6 FROM employees e
7 WHERE e.salary > (
8     SELECT AVG(salary)
9     FROM employees
10    WHERE department_id = e.department_id
11 )
12 ORDER BY dept_avg;
```

## 18.7

**Problem:** Find all employees who are not supervisors.

LAST_NAME
Ernst
Lorentz
Rajs
Davies
Matos
Vargas
Abel
Taylor
Grant
Whalen
Fay
Gietz

12 rows selected.

**Solution:**

a. First do this using the NOT EXISTS operator:

```
1 SELECT
2     outer.last_name
3 FROM
4     employees outer
5 WHERE
6     NOT EXISTS (
7         SELECT
8             'X'
9         FROM
10            employees inner
11         WHERE
12            inner.manager_id = outer.employee_id
13     );
```

b. Can this be done by using the NOT IN operator:

```
1 SELECT
2     outer.last_name
3 FROM
4     employees outer
5 WHERE
6     outer.employee_id NOT IN (
7         SELECT
8             inner.manager_id
9         FROM
10            employees inner
11     );
```

In this alternative solution, the subquery picks up a NULL value. So the entire query returns no rows. Because all conditions that compare a NULL value result in NULL. Whenever NULL values are likely to be part of the value set, we should not use NOT IN as a substitute for NOT EXISTS.

## 18.8

**Problem:** Write a query to display the last names of the employees who earn less than the average salary in their departments.

LAST_NAME
Kochhar
De Haan
Ernst
Lorentz
Davies
Matos
Vargas
Taylor
Fay
Gietz

10 rows selected.

**Solution:**

```
1 SELECT
2     last_name
3 FROM
4     employees outer
5 WHERE
6     outer.salary < (
7         SELECT
8             AVG(inner.salary)
9         FROM
10            employees inner
11         WHERE
12             inner.department_id = outer.department_id
13     );
```

## 18.9

**Problem:** Write a query to display the last names who have one or more coworkers in their departments with later hire dates but higher salaries.

LAST_NAME
Rajs
Davies
Matos
Vargas
Taylor

**Solution:**

```
1 SELECT
2     last_name
3 FROM
4     employees outer
5 WHERE
6     EXISTS (
7         SELECT
8             'X'
9         FROM
10            employees inner
11        WHERE
12            inner.department_id = outer.department_id
13            AND inner.hire_date > outer.hire_date
14            AND inner.salary > outer.salary
15    );
```

## 18.10

**Problem:** Write a query to display the employee ID, last names of the employees, and department names of all employees. *Note: Use a scalar subquery to retrieve the department name in the SELECT statement.*

EMPLOYEE_ID	LAST_NAME	DEPARTMENT
205	Higgins	Accounting
206	Gietz	Accounting
200	Whalen	Administration
100	King	Executive
101	Kochhar	Executive
102	De Haan	Executive
103	Hunold	IT
104	Ernst	IT
107	Lorentz	IT
201	Hartstein	Marketing
202	Fay	Marketing
149	Zlotkey	Sales
176	Taylor	Sales
174	Abel	Sales
EMPLOYEE_ID	LAST_NAME	DEPARTMENT
124	Mourgos	Shipping
141	Rajs	Shipping
142	Davies	Shipping
143	Matos	Shipping
144	Vargas	Shipping
178	Grant	

20 rows selected

**Solution:**

```

1 SELECT
2     employee_id ,
3     last_name ,
4     (
5         SELECT
6             department_name
7         FROM
8             departments d
9         WHERE
10             e.department_id = d.department_id
11     ) department
12 FROM
13     employees e
14 ORDER BY
15     department ;

```

**18.11**

**Problem:** Write a query to display the department names of those departments whose total salary cost is above one-eighth ( $1/8$ ) of the total salary cost of the whole company. Use the WITH clause to write this query. Name the query SUMMARY.

DEPARTMENT_NAME	DEPT_TOTAL
Executive	58000
Sales	30100



**Solution:**

```
1 WITH summary AS (  
2     SELECT  
3         department_name ,  
4         SUM(salary) AS dept_total  
5     FROM  
6         employees ,  
7         departments  
8     WHERE  
9         employees.department_id = departments.department_id  
10    GROUP BY  
11        department_name  
12 )  
13 SELECT  
14     department_name ,  
15     dept_total  
16 FROM  
17     summary  
18 WHERE  
19     dept_total > (  
20         SELECT  
21             SUM(dept_total) * 1 / 8  
22         FROM  
23             summary  
24     )  
25 ORDER BY  
26     dept_total DESC;
```