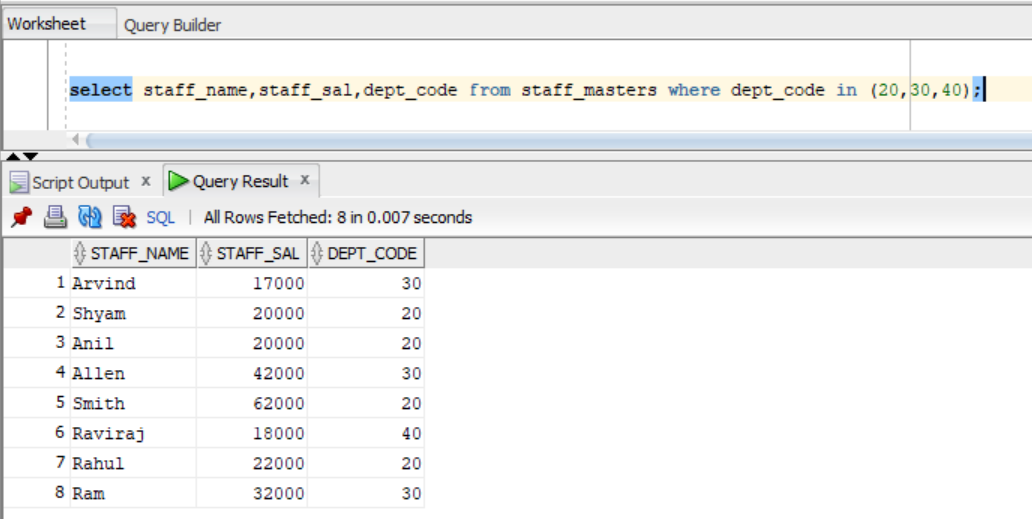
Lab1 -- Basic SQL Command

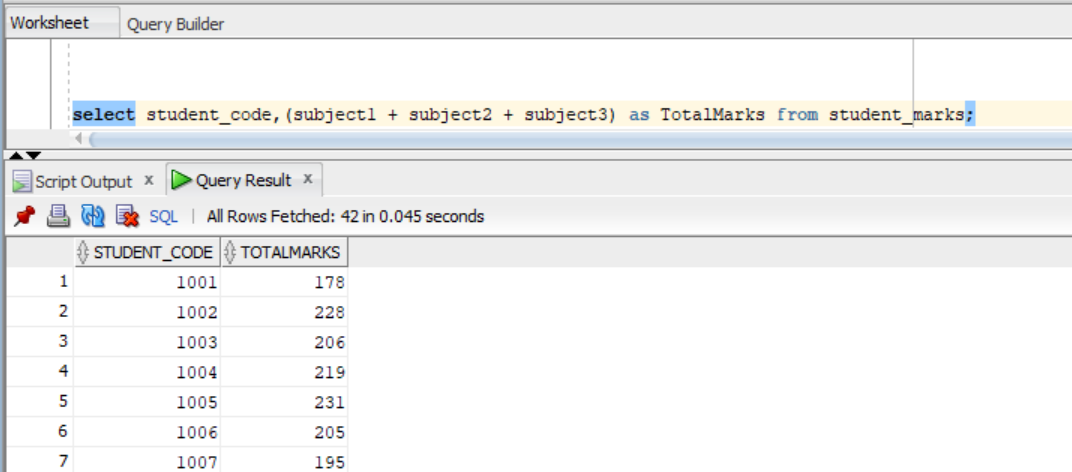
1)Retrieve the details (Name, Salary and dept code) of the staff who are working in department code 20, 30 and 40.

>select staff\_name,staff\_sal,dept\_code from staff\_masters where dept\_code in (20,30,40);

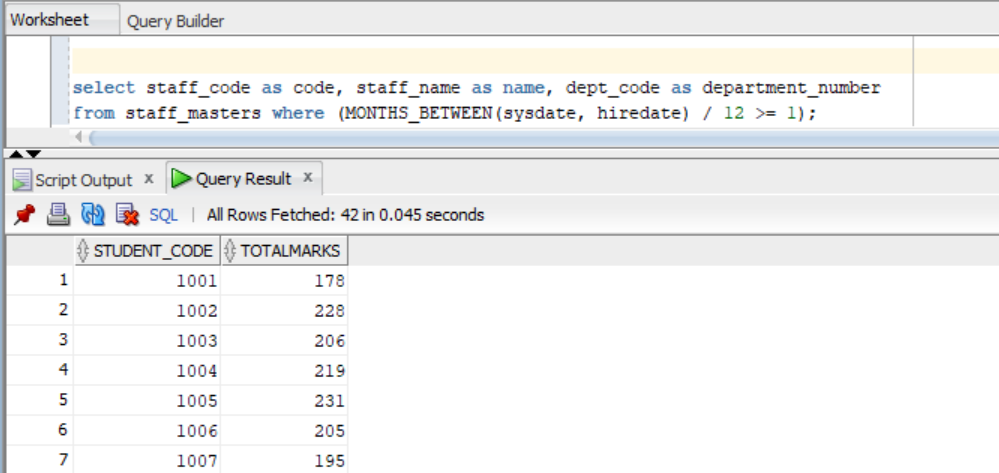


2)Display the code and total marks for every student. Total Marks will be calculated as subject1+subject2+subject3 .(Refer Student\_marks table )

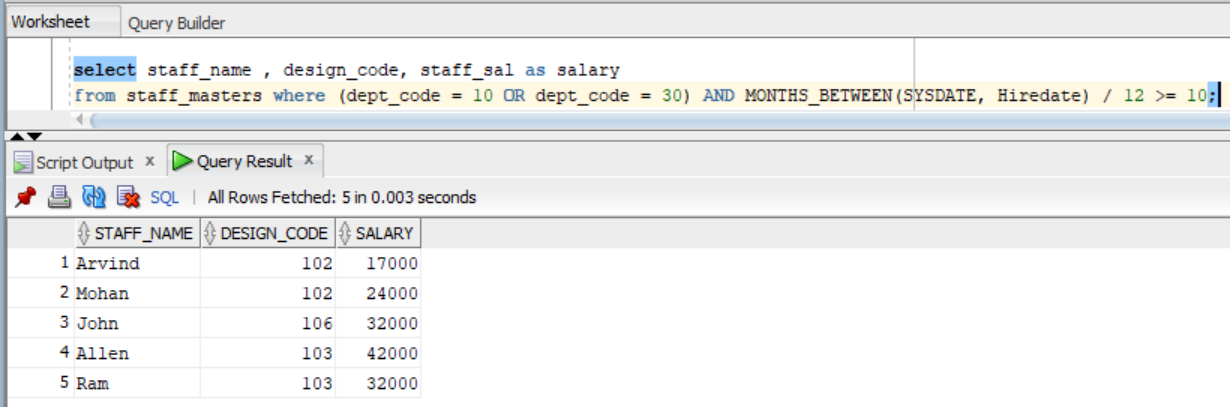
.>select student\_code,(subject1 + subject2 + subject3) as TotalMarks from student\_marks;



4. List the code, name, and department number of the staff who have experience of 18 or more years and sort them based on their experience  
> select staff\_code as code, staff\_name as name, dept\_code as department\_number from staff\_masters where (MONTHS\_BETWEEN(sysdate, hiredate) / 12 >= 1);



5. List the name, designation code, and salary for 10 years of the staff who are working in departments 10 and 30.   
> select staff\_name , design\_code, staff\_sal as salary  
from staff\_masters where (dept\_code = 10 OR dept\_code = 30) AND MONTHS\_BETWEEN(SYSDATE, Hiredate) / 12 >= 10;

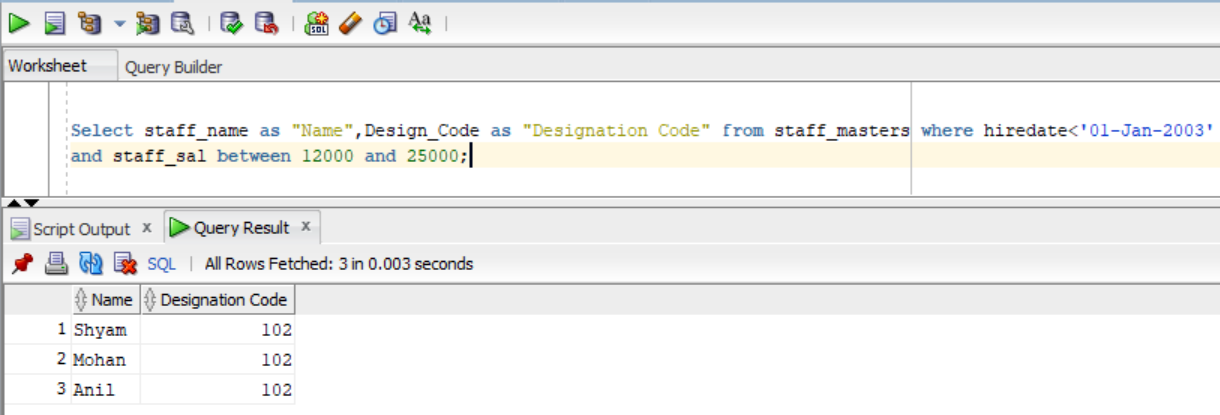


3)List the Name and Designation code of the staff who have joined before Jan 2003 and whose salary range is between 12000 and 25000.

Display the columns with user-defined Column headers. Hint: Use As clause along with other operators.

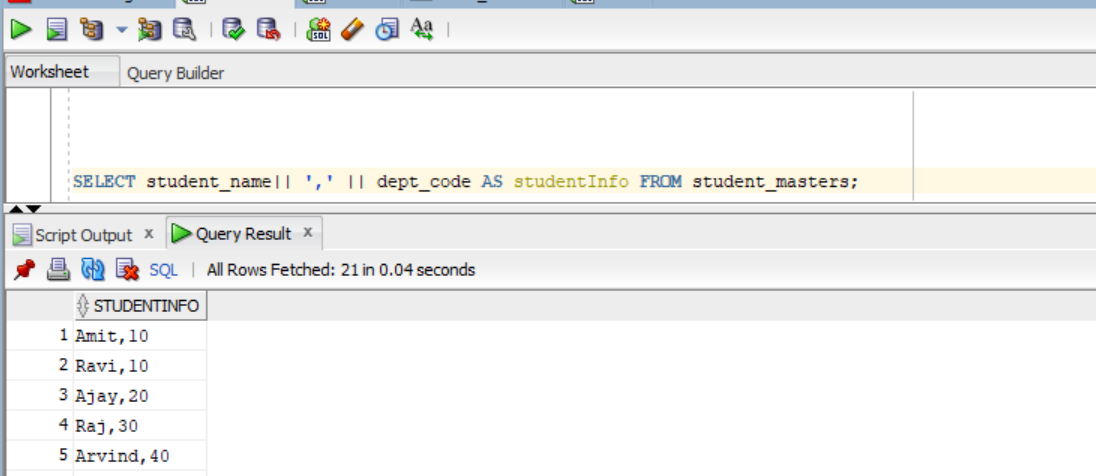
>Select staff\_name as "Name",Design\_Code as "Designation Code" from staff\_masters where hiredate<'01-Jan-2003'

and staff\_sal between 12000 and 25000;



6)Display name concatenated with dept code separated by comma and space. Name the column as ‘Student Info’.

>SELECT student\_name|| ',' || dept\_code AS studentInfo FROM student\_masters;



7)Display the staff details who do not have manager. Hint: Use is null

>SELECT \* FROM staff\_masters WHERE mgr\_code IS NULL;

8)Write a query which will display name, department code and date of birth of all students who were born between January 1, 1981 and March 31, 1983. Sort it based on date of birth (ascending).Hint: Use between operator.

select student\_name, dept\_code, student\_dob

from student\_masters

where student\_dob between '1-JAN-1981' AND '31-MAR-1983'

order by student\_dob asc;

9)Display the Book details that were published during the period of 2001 to 2004. Also display book details with Book name having the character ‘&’ anywhere.

Select \* from book\_masters

where book\_pub\_year between '2001' And '2004' And book\_name like '%&%';

10) Display the Book details where the records have the word “COMP” anywhere in the Book name.

select \* from book\_masters

where book\_name like '%Comp%';

11) List the details of the staff, whose names start with ‘A’ and end with ‘S’ or whose names contains N as the second or third character, and ending with either ‘N’ or ‘S’.

select \* from staff\_masters

where (staff\_name like 'A%S' OR staff\_name like '\_[N][N]%[NS]');

12) List the names of the staff having ‘\_’ character in their name.

select staff\_name from staff\_masters

where staff\_name like '%\_%';

13) Create the Customer table with the following columns.

CustomerId Number(5)

Cust\_Name varchar2(20)

Address1 Varchar2(30)

Address2 Varchar2(30)

create table customer\_table

(CustomerId Number(5),Cust\_Name varchar2(20),Address1 Varchar2(30),Address2 Varchar2(30));

Table created.

SQL> desc customer\_table

Name Null? Type

-----------------------------------------------------

CUSTOMERID NUMBER(5)

CUST\_NAME VARCHAR2(20)

ADDRESS1 VARCHAR2(30)

ADDRESS2 VARCHAR2(30)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

13.a) Modify the Customer table Cust\_Name column of datatype with Varchar2(30), rename the column to CustomerName and it should not accept Nulls.

alter table customer\_table

modify(cust\_name varchar2(30) not null);

Table altered.

desc customer\_table

Name Null? Type

----------------------------------------- -------- ----------------------------

CUSTOMERID NUMBER(5)

CUST\_NAME NOT NULL VARCHAR2(30)

ADDRESS1 VARCHAR2(30)

ADDRESS2 VARCHAR2(30)

=>

alter table customer\_table

rename column cust\_name to customername;

Table altered.

SQL> desc customer\_table;

Name Null? Type

----------------------------------------- -------- ----------------------------

CUSTOMERID NUMBER(5)

CUSTOMERNAME NOT NULL VARCHAR2(30)

ADDRESS1 VARCHAR2(30)

ADDRESS2 VARCHAR2(30)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

14.a)Add the following Columns to the Customer table.

i. Gender Varchar2(1)

ii. Age Number(3)

iii. PhoneNo Number(10)

SQL> alter table customer\_table

2 add(Gender Varchar2(1),Age Number(3),PhoneNo Number(10));

Table altered.

SQL> desc customer\_table;

Name Null? Type

----------------------------------------- -------- ----------------------------

CUSTOMERID NUMBER(5)

CUSTOMERNAME NOT NULL VARCHAR2(30)

ADDRESS1 VARCHAR2(30)

ADDRESS2 VARCHAR2(30)

GENDER VARCHAR2(1)

AGE NUMBER(3)

PHONENO NUMBER(10)

==>14.b)Rename the Customer table to Cust\_Table:

-------------------------------------------------

SQL> rename customer\_table to cust\_table;

Table renamed.

SQL> desc cust\_table

Name Null? Type

----------------------------------------- -------- ----------------------------

CUSTOMERID NUMBER(5)

CUSTOMERNAME NOT NULL VARCHAR2(30)

ADDRESS1 VARCHAR2(30)

ADDRESS2 VARCHAR2(30)

GENDER VARCHAR2(1)

AGE NUMBER(3)

PHONENO NUMBER(10)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

15) Insert rows with the following data in to the Customer table.

a. Insert into customer values: (1000, ‘Allen’, ‘#115 Chicago’, ‘#115 Chicago’, ‘M’, ‘25, 7878776’)

SQL> Insert into customer\_table

values(1000, 'Allen', '#115 Chicago', '#115 Chicago', 'M', 25, 7878776);

1 row created.

===>b. In similar manner, add the below records to the Customer table:

a. 1000, Allen, #115 Chicago, #115 Chicago, M, 25, 7878776

b. 1001, George, #116 France, #116 France, M, 25, 434524

c. 1002, Becker, #114 New York, #114 New York, M, 45, 431525

------------------------------------------------------------------------------------

SQL> Insert into customer\_table

values(1001, 'George', '#116 France','#116 France', 'M', 25, 434524);

1 row created.

SQL> Insert into customer\_table

values(1002, 'Becker', '#114 New York', '#114 New York', 'M', 45, 431525);

1 row created.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

16) Insert the row given below in the Customer table and see the message generated by the Oracle server.

a. 1002, John, #114 Chicago, #114 Chicago, M, 45, 439525

SQL> Insert into customer\_table

2 values(1002, 'John', +'#114 Chicago', '#114 Chicago', 'M', 45, 439525);

1 row created.

CUSTOMERID CUSTOMERNAME ADDRESS1 ADDRESS2 G AGE PHONENO

---------- ------------------------------ ------------------------------ ------------------------------ - ---------- ----------

1001 George #116 France #116 France M 25 434524

1002 Becker #114 New York #114 New York M 45 431525

1000 Allen #115 Chicago #115 Chicago M 25 7878776

1002 John #114 Chicago #114 Chicago M 45 439525

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

17) Delete all the existing rows from Customer table, and let the structure remain itself using TRUNCATE statement.

SQL> truncate table customer\_table;

Table truncated.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

18)In the Customer table, add a column E\_mail.

SQL> alter table customer\_table

2 add(e\_mail varchar2(25));

Table altered.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

19) Drop the E\_mail column from Customer table.

alter table customer\_table

drop column e\_mail;

Table altered.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

20) Add a new column EmailId to Customer table.

alter table customer\_table

add(emaiid varchar2(20));

table altered.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

21) Mark EmailId column as unused before dropping it.

alter table customer\_table

2 set unused (emailid);

table altered.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

22) Drop the unused EmailId column from the Customer table.

alter table customer\_table drop unused column;

Table altered.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

23) Create the Suppliers table based on the structure of the Customer table. Include only the CustomerId, CustomerName, Address1, Address2, and phoneno columns.

create table Suppliers\_table

2 (CustomerId Number(5),Cust\_Name varchar2(20),Address1 Varchar2(30),Address2 Varchar2(30),PhoneNo Number(10));

Table created.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

24) Name the columns in the new table as SuppID, SName, Addr1, Addr2, and Contactno respectively.

alter table Suppliers\_table

rename column CustomerId to SuppID;

alter table Suppliers\_table

rename column CustomerName to SName;

alter table Suppliers\_table

rename column Address1 to Addr1;

alter table Suppliers\_table

rename column Address2 to Addr2;

alter table Suppliers\_table

rename column phoneno to Contactno;

desc Suppliers\_table

Name Null? Type

-----------------------------------------

SUPPID NUMBER(5)

CUST\_NAME VARCHAR2(20)

ADDR1 VARCHAR2(30)

ADDR2 VARCHAR2(30)

CONTACTNO NUMBER(10)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

25) Drop the above table and recreate the following table with the name CustomerMaster.

iv. CustomerId Number(5) Primary key(Name of constraint is CustId\_PK)

v. CustomerName Varchar2(30) Not Null

vi. Addressl Varchar2(30) Not Null

vii. Address2 Varchar2(30)

viii. Gender Varchar2(l)

ix. Age Number(3)

x. PhoneNo Number(10)

drop table suppliers\_table;

==>recreate custometmasters:-

----------------------------

create table customer\_masters

2 (CustId\_PK number(5),CustomerName Varchar2(30) Not Null,Addressl Varchar2(30) Not Null,Address2 Varchar2(30),Gender Varchar2(1),

Age Number(3),PhoneNo Number(10));

Table created.

Alter table customer\_masters add constraints Custid\_prim PRIMARY KEY (customerid);

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

26)

SQL>Create table employee as select \* from emp where 1=3

Table created.

SQL>desc employee

Name Null? Type

-----------------------------------------------------

EMPNO NOT NULL NUMBER(4)

ENAME VARCHAR2(10)

JOB VARCHAR2(9)

MGR NUMBER(4)

HIREDATE DATE

SAL NUMBER(7,2)

COMM NUMBER(7,2)

DEPTNO NUMBER(2)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

27) Write a query to populate Employee table using EMP table’s empno, ename, sal, deptno columns.

select empno,ename,job,sal,comm,deptno from employee;

EMPNO ENAME JOB SAL COMM DEPTNO

---------- ---------- --------- ---------- ---------- ----------

1111 charan sales 12500 1.3 10

1112 satya marketing 15000 2 11

1113 thaneesh clerk 10000 1 12

7839 KING PRESIDENT 5000 10

7698 BLAKE MANAGER 2850 30

7782 CLARK MANAGER 2450 10

7566 JONES MANAGER 2975 20

7788 SCOTT ANALYST 3000 20

7902 FORD ANALYST 3000 20

7369 SMITH CLERK 800 20

7499 ALLEN SALESMAN 1600 300 30

EMPNO ENAME JOB SAL COMM DEPTNO

---------- ---------- --------- ---------- ---------- ----------

7521 WARD SALESMAN 1250 500 30

7654 MARTIN SALESMAN 1250 1400 30

7844 TURNER SALESMAN 1500 0 30

7876 ADAMS CLERK 1100 20

7900 JAMES CLERK 950 30

7934 MILLER CLERK 1300 10

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

28)Write a query to change the job and deptno of employee whose empno is 7698 to the job and deptno of employee having empno 7788.

SQL>

SQL> update employee set job=(select job from employee where empno=7788),deptno=(select deptno from employee where empno=7788) where empno=7698;

1 row updated.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

29)Delete the details of department whose department name is ‘SALES’. (HR shcema)

SQL>delete from departments

2 where department\_name ='sales';

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

30)Write a query to change the deptno of employee with empno 7788 to that of employee having empno 7698.

SQL>update employee set deptno=(select deptno from employee where deptno=7788) where deptno=7698;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

31)

SQL> insert into employee (empno,ename,job,mgr,hiredate,sal,comm,deptno) values (1000,'Allen', 'Clerk',1001,'12-jan-01', 3000, 2,10);

1 row created.

SQL> insert into employee (empno,ename,job,mgr,hiredate,sal,comm,deptno) values (1001,'George', 'analyst', null, '08-Sep-92', 5000,0, 10);

1 row created.

SQL> insert into employee (empno,ename,job,mgr,hiredate,sal,comm,deptno) values (1002, 'Becker', 'Manager', 1000, '4-Nov-92', 2800,4, 20);

1 row created.

SQL>

SQL> insert into employee (empno,ename,job,mgr,hiredate,sal,comm,deptno) values (1003, 'Bill', 'Clerk', 1002, '4-Nov-92',3000, 0, 20);

1 row created.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

32)

create table project

2 (projid varchar2(10)not null,proj\_name varchar2(25),start\_date date,end\_date date);

create table employee\_project

(empno number(10),projectid number(10)primery key);

table created

insert into employee\_project

values(10,20);

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

33)

insert into customer\_table(CustomerId,CustomerName ,Address1 ,Address2,Gender,Age ,PhoneNo)

2 values(6001, 'Jack', '#116 France', '#116 France', 'M', 25, 434524,10000);

1 row created.

SQL> insert into customer\_table(CustomerId,CustomerName ,Address1 ,Address2,Gender,Age ,PhoneNo)

2 values(6000, 'John',' #115 Chicago', '#115 Chicago', 'M', 25, 7878776,20000);

1 row created.

SQL> insert into customer\_table(CustomerId,CustomerName ,Address1 ,Address2,Gender,Age ,PhoneNo)

2 values(6002, 'James', '#114 New York', '#114 New York', 'M', 45, 431525,15000.50);

1 row created.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

34)Create a Savepoint named ‘SP1’ after third record in the Customer table

insert into customer\_table(CustomerId,CustomerName ,Address1 ,Address2,Gender,Age ,PhoneNo)

2 values(6001, 'Jack', '#116 France', '#116 France', 'M', 25, 434524,10000);

1 row created.

SQL> insert into customer\_table(CustomerId,CustomerName ,Address1 ,Address2,Gender,Age ,PhoneNo)

2 values(6000, 'John',' #115 Chicago', '#115 Chicago', 'M', 25, 7878776,20000);

1 row created.

SQL> insert into customer\_table(CustomerId,CustomerName ,Address1 ,Address2,Gender,Age ,PhoneNo)

2 values(6002, 'James', '#114 New York', '#114 New York', 'M', 45, 431525,15000.50);

1 row created.

savepoint p1;

Savepoint created.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

35)Insert the below row in the Customer table.

6003, John, #114 Chicago, #114 Chicago, M, 45, 439525, 19000.60

insert into customer\_table(CustomerId,CustomerName ,Address1 ,Address2,Gender,Age ,PhoneNo,price)

2 values ( 6003, 'John', '#114 Chicago', '#114 Chicago', 'M', 45, 439525, 19000.60);

1 row created.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

36. Execute rollback statement in such a way that whatever manipulations done before Savepoint sp1 are permanently implemented, and the ones after Savepoint SP1 are not stored as a part of the Customer table.

SQL>rollback p1;

**Lab -2** **Operators, Single Row Functions**

1.Create a query which will display Staff Name, Salary of each staff. Format the salary to be 15 character long and left padded with ‘$’.

select staff\_name , '$' || LPAD (to\_char(staff\_sal,999999999.99),15,'$') from staff\_masters;

2.Display name and date of birth of students where date of birth must be displayed in the format similar to “January, 12 1981” for those who were born on Saturday or Sunday.

select student\_name, to\_char(student\_dob,'FMMonth, DD YYYY') from student\_masters where to\_char(student\_dob,'D') in ('7','1');

3.Display each Staff name and number of months they worked for the organization. Label the column as ‘Months Worked’. Order your result by number of months employed. Also Round the number of months to closest whole number.

select staff\_name, round(months\_between(hiredate,sysdate)) as Months\_Worked from staff\_masters order by Months\_worked ;

4.Display the name and salary of the staff. Salary should be represented as X. Each X represents a 1000 in salary. Hint: Divide salary by 1000, use rpad to substitute an ‘X’ for every 1000

select staff\_name, RPAD('X',staff\_sal/1000,'X') from staff\_masters;

5.List the details of the staff who have joined in first half of December month (irrespective of the year).

select \* from staff\_masters where to\_char(hiredate,'MM')='12' and to\_number(to\_char(hiredate,'DD')) between 1 and 15;

6.Write a query that displays Staff Name, Salary, and Grade of all staff. Grade depends on the following table.

select staff\_name, staff\_sal,

case

when staff\_sal >= 50000 then 'A'

when staff\_sal < 50000 and staff\_sal >=25000 then 'B'

when staff\_sal >=10000 and staff\_sal < 25000 then 'C'

else 'D'

end as grade from staff\_masters;

7.Display the Staff Name, Hire date and day of the week on which staff was hired. Label the column as DAY. Order the result by the day of the week starting with Monday. Hint :Use to\_char with hiredate and formats ‘DY’ and ’D’

select staff\_name, to\_char(hiredate,'DY') as day from staff\_masters order by to\_number(to\_char(hiredate,'D'));

8.Show staff names with the respective numbers of asterisk from Staff\_Masters table except first and last characters. For example: KING will be replaced with K\*\*G. . Hint: Use substring, rpad and length functions.

select staff\_name, substr(staff\_name,1,1) || rpad('\*', length(staff\_name) -2 ,'\*') || substr(staff\_name,-1,1) from staff\_masters;

9.Write a query to find the position of third occurrence of ‘i’ in the given word ‘Mississippi’.

select instr('Mississippi','i',1,3) from dual;

10.Write a query to find the pay date for the month. Pay date is the last Friday of the month. Display the date in the format “Twenty Eighth of January, 2002”. Label the heading as PAY DATE. Hint: use to\_char,next\_day and last\_day functions

select to\_char(last\_day(next\_day(sysdate,'Friday')),'FMDay DD "of" month, yyyy') as paydate from dual;

11. Display Student code, Name and Dept Name. Display “Electricals” if dept code = 20, “Electronics” if Dept code =30 and “Others” for all other Dept codes in the Dept Name column. Hint : Use Decode

select student\_code as "student code", student\_name as name, dept\_code,

decode(dept\_code, 20, 'Electricals',

30, 'Electronics',

'others') as "dept name"

from student\_masters;

12. Display the student name and department code of students. If student does not belong to any department, display “No Department”. Label the column as “Department”. (Hint: Use NVL function)

select student\_name as "student name", nvl2(dept\_code, to\_char(dept\_code), 'no department') as "deptartment" from student\_masters;

13. Because of budget issues, the HR department needs a report that displays the last name and salary of employees who earn more than $12,000. Save your SQL statement as a file named lab\_02\_01.sql. Run your query.

SELECT last\_name, salary FROM employees WHERE salary > 12000;

14. Open a new SQL Worksheet. Create a report that displays the last name and department number for employee number 176. Run the query.

SELECT last\_name, department\_id from employees where employee\_id = 176;

15. The HR department needs to find high-salary and low-salary employees. Modify lab\_02\_01.sql to display the last name and salary for any employee whose salary is not in the range of $5,000 to $12,000. Save your SQL statement as lab\_02\_03.sql.

SELECT last\_name, salary FROM employees WHERE salary not between 5000 and 12000 order by salary;

16. Create a report to display the last name, job ID, and hire date for employees with the last names of Matos and Taylor. Order the query in ascending order by the hire date.

select last\_name, job\_id, hire\_date from employees where last\_name in ('Matos' , 'Taylor') order by hire\_date;

17. Display the last name and department ID of all employees in departments 20 or 50 in ascending alphabetical order by name.

select last\_name, department\_id from employees where department\_id in (20,50) order by Last\_name;

18. Modify to display the last name and salary lab\_02\_03.sql of employees who earn between $5,000 and $12,000, and are in department 20 or 50. Label the columns Employee and Monthly Salary, respectively. Save lab\_02\_03.sql as lab\_02\_06.sql again. Run the statement in lab\_02\_06.sql.

SELECT last\_name as Employee, salary as "Monthly Salary" FROM employees WHERE (salary not between 5000 and 12000) and (department\_id in (20, 50));

19. The HR department needs a report that displays the last name and hires date for all employees who were hired in 1994.

select last\_name, hire\_date from employees where To\_char(hire\_date, 'YYYY') = '1994';

20. Create a report to display the last name and job title of all employees who do not have a manager.

select e.last\_name as "Last Name", j.job\_title as "Job Title" from employees e join jobs j on (e.job\_id = j.job\_id) where e.manager\_id is Null;

21.Create a report to display the last name, salary, and commission of all employees who earn commissions. Sort data in descending order of salary and commissions

SELECT last\_name, salary, commission FROM employees WHERE commission IS NOT NULL ORDER BY salary DESC, commission DESC;

22. Use the column’s numeric position in the ORDER BY clause.

>select staff\_name,staff\_sal from staff\_masters order by 1 asc;

select staff\_name,staff\_sal from staff\_masters order by 2 asc;

23. Members of the HR department want to have more flexibility with the queries that you are writing. They would like a report that displays the last name and salary of employees who earn more than an amount that the user specifies after a prompt. Save this query to a file named. If you enter lab\_02\_10.sql 12000 when prompted, the report displays the following results:

-- Prompt the user for the salary threshold

ACCEPT salary\_threshold NUMBER PROMPT 'Enter the salary threshold: '

-- Create a report based on the user input

SELECT last\_name, salary FROM employees WHERE salary > &salary\_threshold;

24. The HR department wants to run reports based on a manager. Create a query that prompts the user for a manager ID and generates the employee ID, last name, salary,and department for that manager’s employees. The HR department wants the ability to sort the report on a selected column. You can test the data with the following values:

manager\_id = 103, sorted by last\_name:

manager\_id = 201, sorted by salary:

manager\_id = 124, sorted by employee\_id:

If you have time, complete the following exercises:

25. Display all employee last names in which the third letter of the name is “a.”

SELECT last\_name FROM employees WHERE SUBSTR(last\_name, 3, 1) = 'a';

26.Display the last names of all employees who have both an “a” and an “e” in their last name.

SELECT last\_name FROM employees WHERE last\_name LIKE '%a%' AND last\_name LIKE '%e%';

27. Display the last name, job, and salary for all employees whose jobs are either those of a sales representative or of a stock clerk, and whose salaries are not equal to $2,500, $3,500, or $7,000.

SELECT last\_name, job, salary FROM employees WHERE job IN ('SALESREPRESENTATIVE','STOCK CLERK') AND salary NOT IN (2500, 3500, 7000);

28.Modify to display the last name, salary, a lab\_02\_06.sql nd commission for all employees whose commission is 20%. Save lab\_02\_06.sql as lab\_02\_15.sql again. Rerun the statement in lab\_02\_15.sql.

SELECT last\_name, salary, commission FROM employees WHERE commission = 0.20;

LAB – 3 Group functions, Joins and Subqueries, Set Operators

1. Display the Highest, Lowest, Total & Average salary of all staff. Label the columns Maximum, Minimum, Total and Average respectively for each Department code. Also round the result to the nearest whole number.

select dept\_code,round(max(staff\_sal)) as "maximum", round(min(staff\_sal)) as "minimum", round(sum(staff\_sal)) as "total", round(avg(staff\_sal)) as "avarage" from staff\_masters group by dept\_code;

2. Display Department code and number of managers working in that department. Label the column as ‘Total Number of Managers’ for each department.

select deptno, count(\*) as "total number of manager" from emp where job='MANAGER' group by deptno;

3. Get the Department number, and sum of Salary of all non managers where the sum is greater than 20000.

select deptno, sum(sal) from emp where job != 'MANAGER' group by deptno having sum(sal) >20000;

join

1. Write a query which displays Staff Name, Department Code, Department Name, and Salary for all staff who earns more than 20000.

select staff\_masters.staff\_name,staff\_masters.dept\_code,department\_masters.dept\_name,staff\_masters.staff\_sal from staff\_masters join department\_masters on staff\_masters.dept\_code=department\_masters.dept\_code;

2. Display Staff Code, Staff Name, Department Name, and his manager’s number and name. Label the columns Staff#, Staff, Mgr#, Manager.

select staff\_masters.staff\_code,staff\_masters.staff\_name,department\_masters.dept\_name,emp.mgr from staff\_masters join department\_masters on staff\_masters.dept\_code=department\_masters.dept\_code join emp on staff\_masters.dept\_code=emp.deptno;

3. Create a query that will display Student Code, Student Name, Department Name, Subject1, Subject2, and Subject3 for all students who are getting 60 and above in each subject from department 10 and 20.

SELECT student\_masters.student\_code, student\_masters.student\_name, department\_masters.dept\_name, student\_marks.subject1, student\_marks.subject2, student\_marks.subject3

FROM student\_masters

JOIN department\_masters ON student\_masters.dept\_code = department\_masters.dept\_code

JOIN student\_marks ON student\_masters.student\_code = student\_marks.student\_code

WHERE student\_marks.subject1 >= 60

AND student\_marks.subject2 >= 60

AND student\_marks.subject3 >= 60

AND student\_masters.dept\_code IN (10, 20);

4. Create a query that will display Student Code, Student Name, Book Code, and Book Name for all students whose expected book return date is today

SELECT

Student\_Masters.Student\_code AS "Student Code",

Student\_Masters.Student\_name AS "Student Name",

Book\_Transactions.Book\_Code AS "Book Code",

Book\_Masters.Book\_Name AS "Book Name"

FROM Student\_Masters

JOIN Book\_Transactions ON Student\_Masters.Student\_code = Book\_Transactions.Student\_code

JOIN Book\_Masters ON Book\_Transactions.Book\_Code = Book\_Masters.Book\_Code

WHERE TO\_DATE(Book\_Transactions.Book\_expected\_return\_date, 'DD-MON-YY') = TO\_DATE(SYSDATE, 'DD-MON-YY');

5. Create a query that will display Staff Code, Staff Name, Department Name, Designation name, Book Code, Book Name, and Issue Date. For only those staff who have taken any book in last 30 days. . If required, make changes to the table to create such a scenario.

SELECT

Staff\_Masters.Staff\_code,

Staff\_Masters.Staff\_Name,

Department\_Masters.Dept\_name,

Designation\_Masters.Design\_name,

Book\_Transactions.Book\_Code,

Book\_Masters.Book\_Name,

Book\_Transactions.Book\_Issue\_date

FROM Staff\_Masters

JOIN Department\_Masters ON Staff\_Masters.Dept\_code = Department\_Masters.Dept\_Code

JOIN Designation\_Masters ON Staff\_Masters.Design\_code = Designation\_Masters.Design\_code

JOIN Book\_Transactions ON Staff\_Masters.Staff\_code = Book\_Transactions.Staff\_code

JOIN Book\_Masters ON Book\_Transactions.Book\_Code = Book\_Masters.Book\_Code

WHERE Book\_Transactions.Book\_Issue\_date >= SYSDATE - 30;

6. Display the unique list of Book code and Book name from the Book transaction.

SELECT DISTINCT

Book\_Transactions.Book\_Code,

Book\_Masters.Book\_Name

FROM Book\_Transactions

JOIN Book\_Masters ON Book\_Transactions.Book\_Code = Book\_Masters.Book\_Code;

7. Generate a report which contains the following information.

Staff Code Staff Name Designation Name Department Name

Department Head

For all staff excluding HOD (List should not contain the details of Department head).

SELECT

Staff\_Masters.Staff\_code,

Staff\_Masters.Staff\_Name,

Designation\_Masters.Design\_name,

Department\_Masters.Dept\_name AS "Department",

Book\_Transactions.Book\_Code,

Book\_Masters.Book\_Name,

Book\_Masters.Book\_pub\_author AS "Author",

CASE

WHEN Book\_Transactions.Book\_actual\_return\_date IS NULL THEN

5 \* TRUNC(SYSDATE - Book\_Transactions.Book\_expected\_return\_date)

ELSE

0

END AS "Fine"

FROM Staff\_Masters

JOIN Department\_Masters ON Staff\_Masters.Dept\_code = Department\_Masters.Dept\_Code

JOIN Designation\_Masters ON Staff\_Masters.Design\_code = Designation\_Masters.Design\_code

JOIN Book\_Transactions ON Staff\_Masters.Staff\_code = Book\_Transactions.Staff\_code

JOIN Book\_Masters ON Book\_Transactions.Book\_Code = Book\_Masters.Book\_Code;

8. Generate a report which contains the following information

Student Code Student Name Department Name Total Marks

HOD Name

Sort the output on Department Name and Total Marks.

SELECT

SM.Student\_Code AS "Student Code",

SM.Student\_name AS "Student Name",

DM.Dept\_name AS "Department Name",

(SM.Subject1 + SM.Subject2 + SM.Subject3) AS "Total Marks",

HOD.Staff\_Name AS "HOD Name"

FROM Student\_Marks SM

JOIN Student\_Masters S ON SM.Student\_Code = S.Student\_Code

JOIN Department\_Masters DM ON S.Dept\_Code = DM.Dept\_Code

LEFT JOIN Staff\_Masters HOD ON DM.Dept\_Code = HOD.Dept\_code AND HOD.Design\_code = 'HOD'

ORDER BY "Department Name", "Total Marks";

9. Generate a report which contains the following information.

Staff Code, Staff Name, Designation Name, Department, Book Code, Book Name,

Author, Fine For the staff who has not returned the book. Fine will be calculated as Rs. 5 per day.

Fine = 5 \* (No. of days = Current Date – Expected return date). Include records in the

table to suit this problem statement

SELECT

Staff\_Masters.Staff\_code,

Staff\_Masters.Staff\_Name,

Designation\_Masters.Design\_name,

Department\_Masters.Dept\_name AS "Department",

Book\_Transactions.Book\_Code,

Book\_Masters.Book\_Name,

Book\_Masters.Book\_pub\_author AS "Author",

CASE

WHEN Book\_Transactions.Book\_actual\_return\_date IS NULL THEN

5 \* TRUNC(SYSDATE - Book\_Transactions.Book\_expected\_return\_date)

ELSE

0

END AS "Fine"

FROM Staff\_Masters

JOIN Department\_Masters ON Staff\_Masters.Dept\_code = Department\_Masters.Dept\_Code

JOIN Designation\_Masters ON Staff\_Masters.Design\_code = Designation\_Masters.Design\_code

JOIN Book\_Transactions ON Staff\_Masters.Staff\_code = Book\_Transactions.Staff\_code

JOIN Book\_Masters ON Book\_Transactions.Book\_Code = Book\_Masters.Book\_Code;

10. List Staff Code, Staff Name, and Salary for those who are getting less than the average salary of organization.

SELECT

Staff\_code AS "Staff Code",

Staff\_Name AS "Staff Name",

Staff\_sal AS "Salary"

FROM Staff\_Masters

WHERE Staff\_sal < (SELECT AVG(Staff\_sal) FROM Staff\_Masters);

11. List the Staff Code, Staff Name who are not Manager.

SELECT

Staff\_code AS "Staff Code",

Staff\_Name AS "Staff Name"

FROM Staff\_Masters join emp on staff\_masters.dept\_code=emp.deptno

WHERE JOB != 'Manager';

12. Display Author Name, Book Name for those authors who wrote more than one book.

SELECT

Book\_pub\_author AS "Author Name",

Book\_Name AS "Book Name"

FROM Book\_Masters

GROUP BY Book\_pub\_author, Book\_Name

HAVING COUNT(\*) > 1;

13. Display Staff Code, Staff Name, and Department Name for those who have taken more than one book.

SELECT

Staff\_Masters.Staff\_code AS "Staff Code",

Staff\_Masters.Staff\_Name AS "Staff Name",

Department\_Masters.Dept\_name AS "Department Name"

FROM Staff\_Masters

JOIN Department\_Masters ON Staff\_Masters.Dept\_code = Department\_Masters.Dept\_Code

WHERE Staff\_Masters.Staff\_code IN (

SELECT Staff\_code

FROM Book\_Transactions

GROUP BY Staff\_code

HAVING COUNT(\*) > 1

);

incorrect 14. Display top ten students for a specified department. Details are:

Student Code, Student Name, Department Name, Subject1, Subject2,

Subject3, Total.

SELECT

SM.Student\_Code AS "Student Code",

SM.Student\_name AS "Student Name",

DM.Dept\_name AS "Department Name",

SM.Subject1,

SM.Subject2,

SM.Subject3,

(SM.Subject1 + SM.Subject2 + SM.Subject3) AS "Total"

FROM Student\_Marks SM

JOIN Student\_Masters S ON SM.Student\_Code = S.Student\_Code

JOIN Department\_Masters DM ON S.Dept\_Code = DM.Dept\_Code

WHERE DM.Dept\_name = 'YourDepartmentName' -- Replace 'YourDepartmentName' with the specific department you're interested in

ORDER BY "Total" DESC

FETCH FIRST 10 ROWS ONLY;

15. a) Display the Staff Name, Department Name, and Salary for those staff who are getting less than average salary in their own department

SELECT

SM.Staff\_Name AS "Staff Name",

DM.Dept\_name AS "Department Name",

SM.Staff\_sal AS "Salary"

FROM Staff\_Masters SM

JOIN Department\_Masters DM ON SM.Dept\_code = DM.Dept\_Code

WHERE SM.Staff\_sal < (

SELECT AVG(SM2.Staff\_sal)

FROM Staff\_Masters SM2

WHERE SM2.Dept\_code = SM.Dept\_code

)

ORDER BY "Department Name", "Salary";

16. Create a query that will display the Staff Name, Department Name, and all the staff who work in the same department as a given staff. Give the column as appropriate label.

select s.staff\_name as "Staff Name", d.dept\_name as "Department Name" from staff\_masters s join department\_masters d on (s.dept\_code=d.dept\_code) where s.dept\_code = (select dept\_code from staff\_masters where staff\_name = 'Shyam');

17. List the Student Code, Student Name for that student who got highest marks in all three subjects in Computer Science department for a particular year according to the table data

select student\_code, student\_name from (select s.student\_code, s.student\_name,student\_year, (subject1+subject2+subject3) as total, dense\_rank() over (order by (subject1+subject2+subject3) desc) rank from student\_masters s join student\_marks m on (s.student\_code = m.student\_code) join department\_masters d on (s.dept\_code = d.dept\_code) where dept\_name = 'Computer Science' and student\_year = 2011) where rank = 1;

18. Display the Student Code, Student Name, and Department Name for that department in which there are maximum number of student are studying.

select s.student\_code, s.student\_name, d.dept\_name from student\_masters s join department\_masters d on (s.dept\_code=d.dept\_code) where s.dept\_code in (select dept\_code from (select dept\_code, count(\*), dense\_rank() over (order by count(\*) desc) as rank from student\_masters group by dept\_code) where rank = 1);

19. Display Staff Code, Staff Name, Department Name, and Designation name for those who have joined in last 3 months.

select s.staff\_code, s.staff\_name, s.hiredate, dept.dept\_name as "department name", des.design\_name as "designation name" from staff\_masters s join department\_masters dept on (s.dept\_code=dept.dept\_code) join designation\_masters des on (s.design\_code=des.design\_code) where hiredate between ADD\_MONTHS(TRUNC(SYSDATE, 'MM'), -2) and ADD\_MONTHS(TRUNC(SYSDATE, 'MM'), 1);

20. Display the Manager Name and the total strength of his/her team.

select m.ename as "Manager", count(e.ename) as "team strength" from emp m join emp e on (m.empno=e.mgr) group by m.ename;

21. Display the details of books that have not been returned and expected return date was last Monday. Book name should be displayed in proper case.. Hint: You can change /add records so that the expected return date suits this problem statement

SELECT INITCAP(BM.Book\_Name) AS "Book Name", BT.Book\_Issue\_date AS "Issue Date", BT.Book\_expected\_return\_date AS "Expected Return Date" FROM Book\_Masters BM JOIN Book\_Transactions BT ON BM.Book\_Code = BT.Book\_Code WHERE BT.Book\_actual\_return\_date IS NULL AND TRUNC(BT.Book\_expected\_return\_date) = NEXT\_DAY(TRUNC(SYSDATE) - 7, 'MONDAY');

22. Write a query to display number of people in each Department. Output should display Department Code, Department Name and Number of People.

select d.dept\_code as "department code", d.dept\_name as "department name", count(\*) as "no of people" from department\_masters d join staff\_masters s on (d.dept\_code=s.dept\_code) group by d.dept\_code, d.dept\_name;

23. Display Manager Code, Manager Name and salary of lowest paid staff in that manager’s team. Exclude any group where minimum salary is less than 10000. Order the result on descending order of salary.

select m.mgr\_code as "manager code", m.staff\_name as "manager name", min(e.staff\_sal) as "lowest paid staff" from staff\_masters e join staff\_masters m on (e.mgr\_code=m.staff\_code) group by m.mgr\_code, m.staff\_name having min(e.staff\_sal) > 10000 order by "lowest paid staff";

3.3 set operators

1. Get the details of all products irrespective of the fact whether they are in previous set or current set.

select \* from current\_products union select \* from previous\_products;

2. Get the details of all products along with the repetition of those that were present both in the previous and current sets.

select \* from current\_products union all select \* from previous\_products;

3. Get the details of only those products which were present in the previous set and are continuing.

select \* from previous\_products minus select \* from current\_products;

4. Get the details of all obsolete products (no longer continued).

select \* from previous\_products minus select \* from current\_products;

LAB- 4 Constraints, Adv. Group by, Adv. Subqueries, Managing other db objects.

4.1

CREATE TABLE Cust\_Table (

CustomerId NUMBER(5),

CustomerName VARCHAR2(30) NOT NULL,

Address1 VARCHAR2(30),

Address2 VARCHAR2(30),

Gender VARCHAR2(1),

Age NUMBER(3),

PhoneNo NUMBER(10)

);

1. Insert the row given below in the Customer table and see the message generated by the Oracle server.

1002, John, #114 Chicago, #114 Chicago, M, 45, 439525

insert into Cust\_Table values ( 1002, 'John', '#114 Chicago', '#114 Chicago', 'M', '45', '439525');

one row inserted

2. Disable the constraint on CustomerId, and insert the following data:

• 1002, Becker, #114 New York, #114 New york , M, 45, 431525

• 1003, Nanapatekar, #115 India, #115 India , M, 45, 431525

Alter table cust\_table drop PRIMARY KEY;

insert into Cust\_Table values (1002, 'Becker', '#114 New York', '#114 New york' , 'M', 45, 431525);

insert into Cust\_Table values (1003, 'Nanapatekar', '#115 India', '#115 India ' , 'M', 45, 431525);

3. Enable the constraint on CustomerId of the Customer table, and see the message generated by the Oracle server.

Alter table cust\_table add constraints Custid\_prim PRIMARY KEY (customerid);

ORA-02437: cannot validate (RDBMS.CUSTID\_PRIM) - primary key violated

02437. 00000 - "cannot validate (%s.%s) - primary key violated"

\*Cause: attempted to validate a primary key with duplicate values or null

values.

\*Action: remove the duplicates and null values before enabling a primary

key.

4.Drop the constraint Custld\_Prim on CustomerId and insert the following Data. Alter Customer table, drop constraint Custid\_Prim.

• 1002, Becker, #114 New York, #114 New york , M, 45, 431525, 15000.50

• 1003, Nanapatekar, #115 India, #115 India , M, 45, 431525, 20000.50

Alter table cust\_table drop PRIMARY KEY;

insert into cust\_table values (1003,'Nanapatekar', '#115 India', '#115 India' , 'M', 45, 431525, 20000.50);

SQL Error: ORA-00913: too many values

00913. 00000 - "too many values"

5.Delete all the existing rows from Customer table, and let the structure remain itself using TRUNCATE statement

truncate table cust\_table;

6. Relate AccountsMaster table and CustomerMaster table through Customerld column with the constraint name Cust\_acc.

CREATE TABLE CustomerMaster (

CustomerId NUMBER(5) PRIMARY KEY,

CustomerName VARCHAR2(30) NOT NULL,

Address1 VARCHAR2(30) NOT NULL,

Address2 VARCHAR2(30),

Gender VARCHAR2(1),

Age NUMBER(3),

PhoneNo NUMBER(10)

);

CREATE TABLE AccountsMaster (

CustomerId NUMBER(5),

AccountNumber NUMBER(10, 2),

AccountType CHAR(3),

LedgerBalance NUMBER(10, 2) NOT NULL,

CONSTRAINT Acc\_PK PRIMARY KEY (AccountNumber)

);

Alter table Accountsmaster ADD constraint ass\_fk FOREIGN KEY(customerid) REFERENCES CustomerMaster(customerid);

Lab 4.2

Determine the validity of the following three statements. Circle either True or False.

1) Group functions work across many rows to produce one result per group.

True

2) Group functions include nulls in calculations.

false

3) The clause restricts rows before inclusion WHERE in a group calculation.

True

The HR department needs the following reports:

4) Find the highest, lowest, sum, and average salary of all employees. Label the columns Maximum, Minimum, Sum, and Average, respectively. Round your results to the nearest whole number. Save your SQL statement as lab\_05\_04.sql. Run the query.

select max(salary) as Maximum, min(salary) as Minimum, sum(salary) as Sum, round(avg(salary)) as Average from employees;

5) Modify the query in lab\_05\_04.sql to display the minimum, maximum, sum, and average salary for each job type. Save lab\_05\_04.sql as lab\_05\_05.sql again.

Run the statement in lab\_05\_05.sql.c

select job\_id as "job type", max(salary) as Maximum, min(salary) as Minimum, sum(salary) as Sum, round(avg(salary)) as Average from employees group by job\_id;

Practice 5-1: Reporting Aggregated Data Using the Group Functions (continued)

6) Write a query to display the number of people with the same job.

Generalize the query so that the user in the HR department is prompted for a job title.

Save the script to a file named lab\_05\_06.sql. Run the query. Enter IT\_PROG when prompted.

select job\_id, count(job\_id) as "no of people" from employees group by job\_id having job\_id = &job;

7) Determine the number of managers without listing them. Label the column Number of Managers.

Hint: Use the MANAGER\_ID column to determine the number of managers.

select count(distinct manager\_id) as "Number of Managers" from employees;

8) Find the difference between the highest and lowest salaries. Label the column DIFFERENCE.

select max(salary) - min(salary) as DIFFERENCE from employees;

9) Create a report to display the manager number and the salary of the lowest-paid employee for that manager. Exclude anyone whose manager is not known. Exclude any groups where the minimum salary is $6,000 or less. Sort the output in descending order of salary.

select manager\_id, min(salary) as "lowest paid employee" from employees where manager\_id is not null group by manager\_id having min(salary) > 6000 order by min(salary) desc;

10) Create a query to display the total number of employees and, of that total, the number of employees hired in 1995, 1996, 1997, and 1998. Create appropriate column headings.

select

count(\*) AS total\_employees,

sum(case when extract(year from hire\_date) = 1995 then 1 else 0 end) AS employees\_1995,

sum(case when extract(year from hire\_date) = 1996 then 1 else 0 end) AS employees\_1996,

sum(case when extract(year from hire\_date) = 1997 then 1 else 0 end) AS employees\_1997,

sum(case when extract(year from hire\_date) = 1998 then 1 else 0 end) AS employees\_1998

from employees;

11) Create a matrix query to display the job, the salary for that job based on department number, and the total salary for that job, for departments 20, 50, 80, and 90, giving each column an appropriate heading.

select

job\_id AS "Job",

MAX(CASE WHEN department\_id = 20 THEN salary END) AS "Salary (Dept 20)",

MAX(CASE WHEN department\_id = 50 THEN salary END) AS "Salary (Dept 50)",

MAX(CASE WHEN department\_id = 80 THEN salary END) AS "Salary (Dept 80)",

MAX(CASE WHEN department\_id = 90 THEN salary END) AS "Salary (Dept 90)",

SUM(salary) AS "Total Salary"

from employees where department\_id in (20, 50, 80, 90) group by job\_id;

4.3

1) The HR department needs a query that prompts the user for an employee last name.

The query then displays the last name and hires date of any employee in the same department as the employee whose name they supply (excluding that employee). For example, if the user enters, find all employees Zlotkey who work with Zlotkey (excluding Zlotkey).

select \* from employees;

SELECT last\_name, hire\_date

FROM employees

WHERE department\_id = (SELECT department\_id FROM employees WHERE last\_name = 'Kochhar')

AND last\_name != 'NKOCHHAR';

2) Create a report that displays the employee number, last name, and salary of all employees who earn more than the average salary. Sort the results in order of ascending salary.

SELECT employee\_id, last\_name, salary

FROM employees

WHERE salary > (SELECT AVG(salary) FROM employees)

ORDER BY salary ASC;

3) Write a query that displays the employee number and last name of all employees who work in a department with any employee whose last name contains the letter “u.”

Save your SQL statement as lab\_07\_03.sql. Run your query.

SELECT employee\_id, last\_name

FROM employees

WHERE department\_id IN (

SELECT DISTINCT department\_id

FROM employees

WHERE last\_name LIKE '%u%'

);

4) The HR department needs a report that displays the last name, department number, and job ID of all employees whose department location ID is 1700.

Modify the query so that the user is prompted for a location ID. Save this to a file named lab\_07\_04.sql.

select \* from departments;

SELECT last\_name, department\_id, job\_id

FROM employees

WHERE department\_id IN (

SELECT department\_id

FROM departments

WHERE location\_id = '&LocationID'

);

5) Create a report for HR that displays the last name and salary of every employee who reports to King.

SELECT last\_name, salary

FROM employees

WHERE manager\_id IN (SELECT employee\_id FROM employees WHERE last\_name = 'King');

6. Create a report for HR that displays the department number, last name, and job ID for every employee in the Executive department.

SELECT department\_id, last\_name, job\_id

FROM employees

NATURAL JOIN departments

WHERE department\_name = 'Executive';

7) Create a report that displays a list of all employees whose salary is more than the salary of any employee from department 60.

SELECT employee\_id, last\_name, salary

FROM employees

WHERE salary > (

SELECT MAX(salary)

FROM employees

WHERE department\_id = 60

);

8) Modify the query in lab\_07\_03.sql to display the employee number, last name, and salary of all employees who earn more than the average salary, and who work in a department with any employee whose last name contains a “u.” Save lab\_07\_03.sql as lab\_07\_08.sql again. Run the statement in lab\_07\_08.sql.

SELECT e1.employee\_id, e1.last\_name, e1.salary

FROM employees e1

JOIN employees e2 ON e1.department\_id = e2.department\_id

WHERE e1.salary > (SELECT AVG(salary) FROM employees)

AND e2.last\_name LIKE '%u%';

Lab 5  
  
  
1) What privilege should a user be given to log on to the Oracle server? Is this a system or an object privilege?

>> TheCREATESESSION system privilege   
  
2)What privilege should a user be given to create tables?

CREATETAB privilege for the database and the USE privilege on the receiving table space

3) If you create a table, who can pass along privileges to other users in your table?

>>anyone you have given those privileges to by using the WITH GRANT OPTION.

4).  You are the DBA. You are creating many users who require the same system privileges. What should you use to make your job easier?  
>> I will group together the privileges in role. And grant this role to the user. If there are multiple set of privileges   
given based on what kind of job user does, I will create multiple roles: As in problem 5 below.

5)What command do you use to change your password?  
alter user statement  
  
6)User21is the owner of theEMPtable and grantsDELETEprivileges toUser22 by using theWITH GRANT OPTIONclause.User22then grantsDELETE   
privileges onEMPtoUser23.User21now finds thatUser23has the privilege and revokes it fromUser22  
. Which user can now delete data from the EMPtable?  
  
>>> user 21 only  
  
7 )You want to grant SCOTT the privilege to update data in the DEPARTMENTS table. You also want to enable SCOTT to grant this privilege to other users. What command do you use?  
>>GRANT UPDATE ON departments TO scott WITH GRANT OPTION;  
  
  
To complete question 8 and the subsequent ones, you need to connect to the database by using SQL Developer. If you are already not connected, do the following to connect:

1. Click the SQL Developer desktop icon.

2. In the Connections Navigator, use the oraxx account and the corresponding password provided by your instructor to log on to the database.

8. Grant another user query privilege on your table. Then, verify whether that user can use the

a) Grant another user privilege to view records in yourREGIONStable. Include an option for this user to further grant this privilege to other users.  
Team 1 executes this statement: GRANT select ON regions TO <team2\_oraxx> WITH GRANT OPTION;   
  
  
b) Have the user query yourREGIONStable. Team 2 executes this statement: SELECT \* FROM <team1\_oraxx>.regions;   
c)  Have the user pass on the query privilege to a third user (for example,ora23). Team 2 executes this statement.   
GRANT select ON <team1\_oraxx>.regions TO <team3\_oraxx>;  
d) Take back the privilege from the user who performs step b.  
Team 1 executes this statement. REVOKE select ON regions FROM <team2\_oraxx>  
  
9. Grant another user query and data manipulation privileges on yourCOUNTRIES table. Make sure the user cannot pass on these privileges to other users.   
Team 1 executes this statement. GRANT select, update, insert ON COUNTRIES TO <team2\_oraxx>;   
  
  
10) Take back the privileges on theCOUNTRIEStable granted to another user. Team 1 executes this statement.  
>> REVOKE select, update, insert ON COUNTRIES FROM <team2\_oraxx>;  
  
11) . Grant another user access to your DEPARTMENTS table. Have the user grant you query access to his or her DEPARTMENTS table.  
>>Team 2 executes the GRANT statement.  
  GRANT select  
  ON departments  
  TO <user1>;

  Team 1 executes the GRANT statement.  
  GRANT select  
  ON departments  
  TO <user2>;  
  WHERE user1 is the name of team 1 and user2 is the name of team 2.  
  
12) Query all the rows in your DEPARTMENTS table.

>> SELECT \* FROM departments;

13) Add a new row to your DEPARTMENTS table. Team 1 should add Education as department number 500   
. Team 2 should add Human Resources as department number 510. Query the other team’s table.

>>  Team 1 executes this INSERT statement.

  INSERT INTO departments(department\_id, department\_name)  
  VALUES (500, ’Education’);  
  COMMIT;

  Team 2 executes this INSERT statement.

  INSERT INTO departments(department\_id, department\_name)  
  VALUES (510, ’Administration’);  
  COMMIT;  
  
  
14) 14. Create a synonym for the other team’s DEPARTMENTS table.

>>  Team 1 creates a synonym named team2.

  CREATE SYNONYM team2  
  FOR <user2>.DEPARTMENTS;

  Team 2 creates a synonym named team1.

  CREATE SYNONYM team1  
  FOR <user1>. DEPARTMENTS;

15) Query all the rows in the other team’s DEPARTMENTS table by using your synonym.  
  
  Team 1 executes this SELECT statement.  
  
  SELECT \*  
  FROM team2;

  Team 2 executes this SELECT statement.

  SELECT \*  
  FROM team1;  
  
16. Revoke the SELECT privilege from the other team.

  Team 1 revokes the privilege.

  REVOKE select  
  ON departments  
  FROM user2;

  Team 2 revokes the privilege.

  REVOKE select  
  ON departments  
  FROM user1;  
  
17) Remove the row you inserted into the DEPARTMENTS table in step 8 and save the changes.

  Team 1 executes this INSERT statement.

  DELETE FROM departments  
  WHERE department\_id = 500;  
  COMMIT;

  Team 2 executes this INSERT statement.

  DELETE FROM departments  
  WHERE department\_id = 510;  
  COMMIT;  
  
  
...practice 2.1  Controlling User Access (continued)

1. 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

  CREATE TABLE dept  
  (id NUMBER(7),  
  name VARCHAR2(25));

2. Populate the DEPT table with data from the DEPARTMENTS table. Include only columns that  
   you need.

  INSERT INTO dept  
  SELECT department\_id, department\_name  
  FROM departments;

3. 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

  CREATE TABLE emp  
  (id NUMBER(7),  
  last\_name VARCHAR2(25),  
  first\_name VARCHAR2(25),  
  dept\_id NUMBER(7));  
  
  
  Practice 2-1: Managing Schema Objects (continued)---------------------  
  
  5) Modify the EMP2t able to allow for longer employee last names. Confirm your modification.  
  
>>ALTER TABLE emp  
  MODIFY (last\_name VARCHAR2(50));  
  
  
  Table altered.  
  
  
Column  Null?  Type  
ID   -   NUMBER(7,0)  
LAST\_NAME   -   VARCHAR2(50)  
FIRST\_NAME   -   VARCHAR2(25)  
DEPT\_ID   -   NUMBER(7,0)

  5) Create the EMPLOYEES2t able 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.  
  
>>  CREATE TABLE employees2 AS  
  SELECT employee\_id id, first\_name, last\_name, salary,  
  department\_id dept\_id  
  FROM employees;  
  
6)  Drop the EMP2 table.

DROP TABLE emp;

7)Restore the EMP2t able to a state before the DROP statement.

>>>>   FLASHBACK TABLE emp2 TO BEFORE DROP;

9). Drop the FIRST\_NAME column from the EMP table. Confirm your modification by checking the  
    description of the table.  
  
  ALTER TABLE emp  
  DROP COLUMN FIRST\_NAME;  
  
  
  Practice 2-1: Managing Schema Objects (continued)  
  
10)10. In the EMPLOYEES2t able, mark the DEPT\_ID column as UNUSED. Confirm your modification by checking the description of the table.

>>> ALTER TABLE emp  
  SET UNUSED (dept\_id);  
  
  
11. Drop all the UNUSED columns from the EMPLOYEES2t able. Confirm your modification by checking the description of the table.  
>> ALTER TABLE emp  
  DROP UNUSED COLUMNS;  
  
12) Create a PRIMARY KEY constraint to the DEPT table using the ID column. The constraint  
   should be named at creation. Name the constraint my\_dept\_id\_pk.    
>> ALTER TABLE emp  
  ADD CONSTRAINT my\_emp\_id\_pk PRIMARY KEY (id);  
  
  
13)Create a PRIMARY KEY constraint to the DEPT2 table using the ID column.   
The constraint should be named at creation. Name the constraint my\_dept\_id\_pk .

>>> ALTER TABLE emp  
   ALTER TABLE  dept2  ADD CONSTRAINT  
   my\_dept\_id\_pk PRIMARY KEY(id);  
  
  
14)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.

  ALTER TABLE emp  
  ADD (dept\_id NUMBER(7));  
  ALTER TABLE emp  
  ADD CONSTRAINT my\_emp\_dept\_id\_fk  
  FOREIGN KEY (dept\_id) REFERENCES dept(id);  
  
15) 15. Modify the EMP2t able. Add a COMMISSION column of the NUMBER data type, precision 2, scale 2.  
Add a constraint to the COMMISSION column that ensures that a commission value is greater than zero.  
  
>>  ALTER TABLE EMP  
  ADD commission NUMBER(2,2)  
  CONSTRAINT my\_emp\_comm\_ck CHECK (commission >= 0;

16) Drop the EMP2 and DEPT2 tables so that they cannot be restored. Verify the recycle bin.

   DROP TABLE emp2 PURGE;  
   DROP TABLE dept2 PURGE;

    SELECT original\_name, operation, droptime FROM recyclebin;  
  
17) Create the DEPT\_NAMED\_INDEX table based on the following table instance chart.

Name the index for the PRIMARY KEY column as DEPT\_PK\_IDX.

Column Name Deptno Dname

Primary Key Yes

Data Type Number VARCHAR2

Length 4 30

>>>> CREATE TABLE DEPT\_NAMED\_INDEX (deptno NUMBER(4) PRIMARY KEY USING INDEX (CREATE INDEX dept\_pk\_idx ON  DEPT\_NAMED\_INDEX(deptno)), dname VARCHAR2(30));

18) Create an external table library\_items\_ext. Use the ORACLE\_LOADER access driver.  Note: The emp\_dir directory and library\_items.dat are already created for this exercise.  library\_items.dat has records in the following format: 2354,    2264, 13.21, 150, 2355,    2289, 46.23, 200, 2355,    2264, 50.00, 100,     a) Open the lab\_02\_18.sql file. Observe the code snippet to create the  library\_items\_ext external table. Then, replace <TODO1>, <TODO2>,  <TODO3>, and <TODO4> as appropriate and save the file as lab\_02\_18\_soln.sql. Run the script to create the external table.  CREATE TABLE library\_items\_ext ( category\_id   number(12)                                , book\_id number(6)                                , book\_price number(8,2)                                , quantity   number(8)                                ) ORGANIZATION EXTERNAL  (TYPE ORACLE\_LOADER  DEFAULT DIRECTORY emp\_dir  ACCESS PARAMETERS (RECORDS DELIMITED BY NEWLINE                      FIELDS TERMINATED BY ',')  LOCATION ('library\_items.dat')  ) REJECT LIMIT UNLIMITED;

Practice 2-1: Managing Schema Objects (continued

  a. Open the lab\_02\_18.sql file. Observe the code snippet to create the library\_items\_ext external table. Then replace < TODO1 > , < TODO2 > ,  
< TODO3 > , and < TODO4 > as appropriate and save the file as  
lab\_02\_18\_soln.sql . Run the script to create the external table.

>>CREATE TABLE library\_items\_ext ( category\_id   number(12), book\_id number(6), book\_price number(8,2) ,  
quantity   number(8) ) ORGANIZATION EXTERNAL   
(TYPE ORACLE\_LOADER  DEFAULT DIRECTORY emp\_dir  ACCESS PARAMETERS (RECORDS DELIMITED BY NEWLINE FIELDS TERMINATED BY ',')   
LOCATION ('library\_items.dat')  ) REJECT LIMIT UNLIMITED;  
  
b) Query the library\_items\_ext table.

>>  SELECT \* FROM library\_items\_ext;

19. The HR department needs a report of addresses of all the departments. Create an external table as dept\_add\_ext using the ORACLE\_DATAPUMP access driver.  
The report should show the location ID, street address, city, state or province, and country in the output. Use a NATURAL JOIN to produce the results.  
Note: The emp\_dir directory is already created for this exercise  
  
a)   Open the lab\_02\_19.sql file. Observe the code snippet to create the  dept\_add\_ext external table. Then, replace <TODO1>, <TODO2>, and<TODO3> with appropriate code.  
Replace <oraxx\_emp4.exp> and <oraxx\_emp5.exp> with appropriate file names. For example, if you are user ora21, your file names are ora21\_emp4.exp and ora21\_emp5.exp.   
Savethe script as lab\_02\_19\_soln.sql.

>>>> CREATE TABLE dept\_add\_ext (location\_id, street\_address, city,  state\_province, country\_name) ORGANIZATION EXTERNAL( TYPE ORACLE\_DATAPUMP  
DEFAULT DIRECTORY emp\_dir LOCATION ('oraxx\_emp4.exp','oraxx\_emp5.exp'))   
PARALLEL AS  SELECT location\_id, street\_address, city, state\_province, country\_name FROM locations  NATURAL JOIN countries;

b) Query the dept\_add\_ext table.  
>  SELECT \* FROM dept\_add\_ext;

Practice 2-1: Managing Schema Objects (continued)

20) Create the emp\_books table and populate it with data. Set the primary key as deferred and observe what happens at the end of the transaction.   
a) Run the lab\_02\_20a.sql script to create the emp\_books table. Observe that the emp\_books\_pk primary key is not created as deferrable.  
>>  CREATE TABLE emp\_books (book\_id number,title varchar2(20), CONSTRAINT emp\_books\_pk  PRIMARY KEY (book\_id));   
  
b)  Run  the  lab\_02\_20b.sql  script  to  populate  data  into  the  emp\_books table What do you observe?

>> INSERT INTO emp\_books VALUES(300,'Organizations');   
INSERT INTO emp\_books VALUES(300,'Change Management');   
The first row is inserted. However, you see the ora-00001 error with the second row insertion.

c) Set the emp\_books\_pk constraint as deferred. What do you observe?

>> SET CONSTRAINT emp\_books\_pk DEFERRED;         
You see the following error: “ORA-02447: Cannot defer a constraint that is not deferrable.”   
  
d) Drop the emp\_books\_pk constraint.

>> ALTER TABLE emp\_books DROP CONSTRAINT emp\_books\_pk;

e) Modify the emp\_books table definition to add the emp\_books\_pkconstraint as deferrable this time.

>> ALTER TABLE emp\_books ADD (CONSTRAINT emp\_books\_pk PRIMARY KEY (book\_id) DEFERRABLE);

Practice 2-1: Managing Schema Objects (continued)

f) Set the emp\_books\_pk constraint as deferred.  
>> SET CONSTRAINT emp\_books\_pk DEFERRED;   
  
  
g) Run the lab\_02\_20g.sql script to populate data into the emp\_bookstable.  
What do you observe?

>>INSERT INTO emp\_books VALUES (300,'Change Management');   
>>INSERT INTO emp\_books VALUES (300,'Personality');   
>>INSERT INTO emp\_books VALUES (350,'Creativity');  
  
  
  h) Commit the transaction. What do you observe?   
  
>> COMMIT;  
  
  
  lab 5 Practice 3-1: Managing Objects with Data Dictionary Views  
In this practice, you query the dictionary views to find information about objects in your schema.  
1. Query the USER\_TABLES data dictionary view to see information about the tables that you own.  
SELECT table\_name, tablespace\_name, num\_rows, last\_analyzed  
FROM user\_tables;  
  
…  
2. Query the ALL\_TABLES data dictionary view to see information about all the tables that you can access. Exclude the tables that you own.  
Note: Your list may not exactly match the following list:  
…  
SELECT table\_name, owner, tablespace\_name, num\_rows, last\_analyzed  
FROM all\_tables  
WHERE owner <> USER;  
  
  
3. for a specified table, create a script that reports the column names, data types, and data types ’ lengths, as well as whether nulls are allowed. Prompt the user to enter the table name. Give appropriate aliases to the DATA\_PRECISION and DATA\_SCALE columns. Save this script in a file named lab\_03\_01.sql.  
For example, if the user enters DEPARTMENTS, the following output results:  
4. Create a script that reports the column name, constraint name, constraint type, search  
Condition and status for a specified table. You must join the USER\_CONSTRAINTS and USER\_CONS\_COLUMNS tables to obtain all this information. Prompt the user to enter the table name. Save the script in a file named lab\_03\_04.sql.  
For example, if the user enters DEPARTMENTS, the following output results:  
  
5. Add a comment to the DEPARTMENTS table. Then query the USER\_TAB\_COMMENTS view to verify that the comment is present.  
COMMENT ON TABLE DEPARTMENTS IS 'This is the DEPARTMENTS table for storing department information.';  
SELECT table\_name, comments  
FROM user\_tab\_comments  
WHERE table\_name = 'DEPARTMENTS';  
  
6. Create a synonym for your EMPLOYEES table. Call it EMP. Then find the names of all synonyms that are in your schema.  
CREATE SYNONYM EMP FOR EMPLOYEES;  
SELECT synonym\_name FROM user\_synonyms;  
  
7. Run lab\_03\_07.sql to create the dept50 view for this exercise.  
You need to determine the names and definitions of all the views in your schema.  
Create a report that retrieves view information: the view name and text from the USER\_VIEWS data dictionary view.  
SELECT view\_name, text  
FROM user\_views;  
SET LONG 10000

8. Find the names of your sequences. Write a query in a script to display the following information about your sequences:  
   sequence name, maximum value, increment size, and last number. Name the script lab\_03\_08.sql. Run the statement in your script.  
Run the lab\_03\_09\_tab.sql script as a prerequisite for exercises 9 through 11.  
Alternatively, open the script file to copy the code and paste it into your SQL Worksheet.  
Then execute the script. This script:  
• Drops if there are existing tables DEPT2 and EMP2  
• Creates the DEPT2 and EMP2 tables  
Note: In Practice 2, you should have already dropped the DEPT2 and EMP2 tables so that they cannot be restored.  
>>>> SELECT sequence\_name, max\_value, increment\_by, last\_number FROM user\_sequences;  
Drop table DEPT2;  
Drop table EMP2;  
CREATE TABLE DEPT2 (  
  Deptno NUMBER(2) PRIMARY KEY,  
  Dname VARCHAR2(14),  
  Loc VARCHAR2(13)  
);  
CREATE TABLE EMP2 (  
  Empno NUMBER(4) PRIMARY KEY,  
  Ename VARCHAR2(10),  
  Job VARCHAR2(9),  
  Mgr NUMBER(4),  
  Hiredate DATE,  
  Sal NUMBER(7, 2),  
  Comm NUMBER(7, 2),  
  Deptno NUMBER(2)  
);  
  
  
9. Confirm that both the DEPT2 and EMP2 tables are stored in the data dictionary.  
select \* from EMP2;  
select \* from DEPT2;  
alternate:-  SELECT table\_name FROM user\_tables WHERE table\_name IN ('DEPT2', 'EMP2');  
  
10. Confirm that the constraints were added by querying the USER\_CONSTRAINTS view. Note the types and names of the constraints.  
>>> select constraint\_name, constraint\_type from user\_constraints where table\_name IN ('DEPT2', 'EMP2');  
  
  
11. Display the object names and types from the USER\_OBJECTS data dictionary view for the EMP2  
and DEPT2 tables.  
>>> select object\_name, object\_type FROM user\_objects where object\_name IN ('EMP2', 'DEPT2');  
  
12. Create the SALES\_DEPT table based on the following table instance chart. Name the index for the PRIMARY KEY column SALES\_PK\_IDX . Then query the data dictionary view to find the index name, table name, and whether the index is unique.  
  
>>>  
CREATE TABLE SALES\_DEPT (  
  Team\_Id NUMBER(3),  
  Location VARCHAR2(30),  
  CONSTRAINT SALES\_PK\_IDX PRIMARY KEY (Team\_Id)  
);  
  
SELECT index\_name, table\_name, uniqueness  
FROM user\_indexes  
WHERE table\_name = 'SALES\_DEPT' AND index\_name = 'SALES\_PK\_IDX';

lab=6:-

-------

PRACTIES-1:

---------------

1)Run the lab\_04\_01.sql script in the lab folder to create the SAL\_HISTORY table.

create table sal\_history(employee\_id number(10),hiredate date,sal number(10));

Table created.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2)Display the structure of the SAL\_HISTORY table.

SQL> desc sal\_history

Name Null? Type

--------------------------------------------

EMPLOYEE\_ID NUMBER(10)

HIREDATE DATE

SAL NUMBER(10)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

3)Run the lab\_04\_03.sql script in the lab folder to create the MGR\_HISTORY table.

create table mgr\_history

2 (employee\_id number(4),mgr\_id number(2),sal number(10));

Table created.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4)Display the structure of the MGR\_HISTORY table.

SQL> desc mgr\_history

Name Null? Type

-----------------------------------------------

EMPLOYEE\_ID NUMBER(4)

MGR\_ID NUMBER(2)

SAL NUMBER(10)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

5)Run the lab\_04\_05.sql script in the lab folder to create the SPECIAL\_SAL table.

SQL> create table special\_sal

2 (employee\_id number(5),sal number(10));

Table created.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

6. Display the structure of the SPECIAL\_SAL table.

SQL> desc special\_sal

Name Null? Type

----------------------------------

EMPLOYEE\_ID NUMBER(5)

SAL NUMBER(10)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

7)a. Write a query to do the following:

- Retrieve details such as the employee ID, hire date, salary, and manager ID of those employees whose employee ID is less than 125 from the EMPLOYEES table.

SQL> select empno,hiredate,sal,mgr from employee

2 where empno <125;

EMPNO HIREDATE SAL MGR

---------- --------- ---------- ----------

120 20-JUN-00 20267 110

- If the salary is more than $20,000, insert details such as the employee ID and salary into the SPECIAL\_SAL table.

SQL> insert into special\_sal(employee\_id,sal)

2 select empno,sal

3 from employee

4 where sal >20000;

2 rows created.

SQL> select \* from special\_sal;

EMPLOYEE\_ID SAL

----------- ----------

125 28000

120 20267

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

8)a. Run the lab\_04\_08a.sql script in the lab folder to create the

SALES\_WEEK\_DATA table

create table SALES\_WEEK\_DATA

(ID number(5), week\_ID varchar2(25), sales\_quantity\_Monday number(10), sales\_quantity\_Tuesday number(10), sales\_quantity\_Wednesday number(10), sales\_quantity\_Thursday number(10),sales\_quantity\_Friday number(10));

b. Run the lab\_04\_08b.sql script in the lab folder to insert records into the SALES\_WEEK\_DATA table.

INSERT ALL INTO SALES\_WEEK\_DATA

VALUES (101,'mon\_101',50,49.8,51.1,56,62)

INTO SALES\_WEEK\_DATA

VALUES (102,'tues\_101',55,51,63,68,45)

INTO SALES\_WEEK\_DATA

VALUES (103,'wed\_101',56.9,55,52,50,58)

INTO SALES\_WEEK\_DATA

VALUES (104,'thus\_103',51,69,45,58,38)

INTO SALES\_WEEK\_DATA

VALUES (105,'fri\_103',71,65,58.3,64.1,45.8)

SELECT \* from dual;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

c. Display the structure of the SALES\_WEEK\_DATA table.

SQL> desc sales\_week\_data

Name Null? Type

------------------------------------------------------------

ID NUMBER(5)

WEEK\_ID VARCHAR2(25)

SALES\_QUANTITY\_MONDAY NUMBER(10)

SALES\_QUANTITY\_TUESDAY NUMBER(10)

SALES\_QUANTITY\_WEDNESDAY NUMBER(10)

SALES\_QUANTITY\_THURSDAY NUMBER(10)

SALES\_QUANTITY\_FRIDAY- NUMBER(10)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

d)Display the records from the SALES\_WEEK\_DATA table.

SQL> select \* from sales\_week\_data;

ID WEEK\_ID SALES\_QUANTITY\_MONDAY SALES\_QUANTITY\_TUESDAY SALES\_QUANTITY\_WEDNESDAY SALES\_QUANTITY\_THURSDAY SALES\_QUANTITY\_FRIDAY

---------- --------------------- --------------------- ---------------------- ------------------------ ----------------------- ---------------------

101 mon\_101 50 50 51 56 62

102 tues\_101 55 51 63 68 45

103 wed\_101 57 55 52 50 58

104 thus\_103 51 69 45 58 38

105 fri\_103 71 65 58 64 46

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

e)Run the lab\_04\_08\_e.sql script in the lab folder to create the EMP\_SALES\_INFO table.

SQL> create table EMP\_SALES\_INFO

2 (ID number(5), week\_ID number(4), sales\_quantity\_Monday number(10), sales\_quantity\_Tuesday number(10), sales\_quantity\_Wednesday number(10), sales\_quantity\_Thursday number(10),sales\_quantity\_Friday number(10));

Table created.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

f)Display the structure of the EMP\_SALES\_INFO table.

SQL> desc EMP\_SALES\_INFO

Name Null? Type

------------------------------------------------------------

ID NUMBER(5)

WEEK\_ID VARCHAR2(25)

SALES\_QUANTITY\_MONDAY NUMBER(10)

SALES\_QUANTITY\_TUESDAY NUMBER(10)

SALES\_QUANTITY\_WEDNESDAY NUMBER(10)

SALES\_QUANTITY\_THURSDAY NUMBER(10)

SALES\_QUANTITY\_FRIDAY- NUMBER(10)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

g)Write a query to do the following:

- Retrieve details such as ID, week ID, sales quantity on Monday, sales quantity on Tuesday, sales quantity on Wednesday, sales quantity on Thursday, and sales quantity on Friday from the SALES\_WEEK\_DATA table.

SQL> select id,week\_id,SALES\_QUANTITY\_MONDAY,SALES\_QUANTITY\_TUESDAY,SALES\_QUANTITY\_WEDNESDAY,SALES\_QUANTITY\_thursday,SALES\_QUANTITY\_FRIDAY

2 from sales\_week\_data;

ID WEEK\_ID SALES\_QUANTITY\_MONDAY SALES\_QUANTITY\_TUESDAY SALES\_QUANTITY\_WEDNESDAY SALES\_QUANTITY\_THURSDAY SALES\_QUANTITY\_FRIDAY

---------- ------------------------- --------------------- ---------------------- ------------------------ ----------------------- ---------------------

101 mon\_101 50 50 51 56 62

102 tues\_101 55 51 63 68 45

103 wed\_101 57 55 52 50 58

104 thus\_103 51 69 45 58 38

105 fri\_103 71 65 58 64 46

- Build a transformation such that each record retrieved from the SALES\_WEEK\_DATAt able is converted into multiple records for the EMP\_SALES\_INFO table.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

h)Display the records from the EMP\_SALES\_INFO table.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

9)You have the data of past employees stored in a flat file called emp.data. You want to store the names and email IDs of all employees, past and present, in a table. To do this, first create an external table called EMP\_DATA using the emp.dat source file in the emp\_dir directory. Use the lab\_04\_09.sql script to do this.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Practice 5-1: Managing Data in Different Time Zones(HR-SCHEMA):-

----------------------------------------------------------------

In this practice, you display time zone offsets, CURRENT\_DATE, CURRENT\_TIMESTAMP, and LOCALTIMESTAMP. You also set time zones and use the EXTRACT function.

1. Alter the session to set NLS\_DATE\_FORMATto DD-MON-YYYYH H24: MI:SS.

alter session set NLS\_DATE\_FORMAT = 'DD-MON-YYYY HH24:MI:SS';

Session altered.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2. Write queries to display the time zone offsets (TZ\_OFFSET) for the following time zones.

- US/Pacific-New

- Singapore

- Egypt

SQL> select TZ\_OFFSET ('US/Pacific-New') from dual;

TZ\_OFFS

-------

-08:00

SQL> select TZ\_OFFSET ('Singapore') from dual;

TZ\_OFFS

-------

+08:00

SQL> select TZ\_OFFSET ('Egypt') from dual;

TZ\_OFFS

-------

+02:00

---------------------------------------------------------------------------------------------------------------

b. Alter the session to set the TIME\_ZONE parameter value to the time zone offset of US/Pacific-New.

alter session set TIME\_ZONE = '-7:00';

Session altered.

----------------------------------------------------------------------------------------------------------------

c. Display CURRENT\_DATE, CURRENT\_TIMESTAMP, and LOCALTIMESTAMP for this session.

SQL> select CURRENT\_DATE, CURRENT\_TIMESTAMP, LOCALTIMESTAMP from dual;

CURRENT\_DATE CURRENT\_TIMESTAMP LOCALTIMESTAMP

-------------------- --------------------------------------------------------------------------- -----------------------------------------------------

06-NOV-2023 00:09:46 06-NOV-23 12.09.46.000000 AM -07:00 06-NOV-23 12.09.46.000000 AM

----------------------------------------------------------------------------------------------------------------

d. Alter the session to set the TIME\_ZONE parameter value to the time zone offset of Singapore.

alter session set TIME\_ZONE = '+8:00';

Session altered.

----------------------------------------------------------------------------------------------------------------

e. Display CURRENT\_DATE, CURRENT\_TIMESTAMP, and LOCALTIMESTAMP for this session.

==>

SQL> select CURRENT\_DATE, CURRENT\_TIMESTAMP, LOCALTIMESTAMP from dual;

CURRENT\_DATE CURRENT\_TIMESTAMP LOCALTIMESTAMP

-------------------- --------------------------------------------------------------------------- ----------------------------------------------

06-NOV-2023 15:11:11 06-NOV-23 03.11.11.000000 PM +08:00 06-NOV-23 03.11.11.000000 PM

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

3. Write a query to display DBTIMEZONE and SESSIONTIMEZONE.

SQL> select DBTIMEZONE, SESSIONTIMEZONE from dual;

DBTIME SESSIONTIMEZONE

------ ---------------------------------------------------------------------------

+00:00 +08:00

----------------------------------------------------------------------------------------------------------------

4. Write a query to extract the YEAR from the HIRE\_DATE column of the EMPLOYEES table for those employees who work in department 80.

==> select last\_name, extract(YEAR from HIRE\_DATE) from employees where department\_id = 80;

----------------------------------------------------------------------------------------------------------------

Practice 5-1: Managing Data in Different Time Zones (continued)

5. Alter the session to set NLS\_DATE\_FORMAT to DD-MON-YYYY.

==> alter session set NLS\_DATE\_FORMAT = 'DD-MON-YYYY';

----------------------------------------------------------------------------------------------------------------

6. Examine and run the lab\_05\_06.sql script to create the SAMPLE\_DATES table and populate it.

a. Select from the table and view the data.

create table sample\_dates

2 (date\_col date);

Table created.

==> select \* from sample\_dates;

DATE\_COL

-----------

06-NOV-2023

----------------------------------------------------------------------------------------------------------------

b. Modify the data type of the DATE\_COL column and change it to TIMESTAMP. Select from the table to view the data.

SQL> alter table sample\_dates

2 modify(date\_col timestamp);

SQL> select \* from sample\_dates;

DATE\_COL

---------------------------------------------------------------------------

06-NOV-23 12.00.00.000000 AM

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

c. Try to modify the data type of the DATE\_COL column and change it to TIMESTAMP WITH TIME ZONE . What happens?

SQL> alter table sample\_dates

2 modify (date\_col timestamp with time zone);

modify (date\_col timestamp with time zone)

\*

ERROR at line 2:

ORA-01439: column to be modified must be empty to change datatype

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

7. Create a query to retrieve last names from the EMPLOYEES table and calculate the review status. If the year hired was 1998, display needs Review for the review status; otherwise, display not this year! Name the review status column Review. Sort the results by the HIRE\_DATE column.

Hint: Use a CASE expression with the EXTRACT function to calculate the review status.

SQL> SELECT e.last\_name,

2 (CASE extract(year from e.hire\_date)

3 WHEN 1998 THEN 'Needs Review'

4 ELSE 'not this year!'

5 END)

6 AS "Review "

7 FROM employees e

8 ORDER BY e.hire\_date;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Practice 5-1: Managing Data in Different Time Zones (continued)

8. Create a query to print the last names and the number of years of service for each employee.

If the employee has been employed for five or more years, print 5 years of service.

If the employee has been employed for 10 or more years, print 10 years of service.

If the employee has been employed for 15 or more years, print 15 years of service .

If none of these conditions match, print may be

next year! Sort the results by the HIRE\_DATE column. Use the EMPLOYEES table.

Hint: Use CASE expressions and TO\_YMINTERVAL.

SQL> select e.last\_name, hire\_date, sysdate,

2 (CASE

3 WHEN (sysdate -TO\_YMINTERVAL('15-0')) >= hire\_date THEN '15 years of service'

4 WHEN (sysdate -TO\_YMINTERVAL('10-0')) >= hire\_date THEN '10 years of service'

5 WHEN (sysdate - TO\_YMINTERVAL('5-0')) >= hire\_date THEN '5 years of service'

6 ELSE 'maybe next year!'

7 END)

8 as "Awards"

9 from employees e;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Practice 6-1: Retrieving Data by Using Subqueries

In this practice, you write multiple-column subqueries, and correlated and scalar subqueries. You also solve problems by writing the WITH clause.

1. 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.

SQL> select last\_name, department\_id, salary from employees

2 where (salary, department\_id) IN (select salary, department\_id from employees

3 where commission\_pct is not null);

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2. 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.

SQL> select e.last\_name, d.department\_name, e.salary from employees e, departments d

2 where e.department\_id = d.department\_id AND (salary, NVL(commission\_pct,0)) IN (select salary, NVL(commission\_pct,0) from employees e,departments d

3 where e.department\_id = d.department\_id AND d.location\_id = 1700);

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

3. 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.

SQL> select last\_name, hire\_date, salary from employees

2 where (salary, NVL(commission\_pct,0)) IN (SELECT salary, NVL(commission\_pct,0) from employees

3 where last\_name = 'Kochhar') AND last\_name != 'Kochhar';

LAST\_NAME HIRE\_DATE SALARY

------------------------- ----------- ----------

De Haan 13-JAN-2001 17000

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

SQL> select last\_name, job\_id, salary from employees

2 where salary > ALL (select salary from employees

3 where job\_id = 'SA\_MAN')

4 order by salary DESC;

LAST\_NAME JOB\_ID SALARY

------------------------- ---------- ----------

King AD\_PRES 24000

De Haan AD\_VP 17000

Kochhar AD\_VP 17000

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Practice 6-1: Retrieving Data by Using Subqueries (continued)

5. Display details such as the employee ID, last name, and department ID of those employees who live in cities the names of which begin with T.

SQL> select employee\_id, last\_name, department\_id from employees

2 where department\_id IN (select department\_id from departments

3 where location\_id IN (select location\_id from locations

4 where city LIKE 'T%'));

EMPLOYEE\_ID LAST\_NAME DEPARTMENT\_ID

----------- ------------------------- -------------

202 Fay 20

201 Hartstein 20

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

6. 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 and round to two decimals. Use aliases for the columns retrieved by the query as shown in the sample output.

SQL> select e.last\_name ename, e.salary salary, e.department\_id deptno, AVG(a.salary) dept\_avg from employees e, employees a

2 where e.department\_id = a.department\_id AND e.salary > (select AVG(salary) from employees

3 where department\_id = e.department\_id ) group by e.last\_name, e.salary, e.department\_id

4 order by AVG(a.salary);

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

7. Find all employees who are not supervisors.

a. First, do this using the NOT EXISTS operator.

SQL> select outer.last\_name from employees outer

2 where not exists (select 'X' from employees inner

3 where inner.manager\_id = outer.employee\_id);

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Practice 6-1: Retrieving Data by Using Subqueries (continued)

b. Can this be done by using the NOT IN operator? How, or why not?

SQL> select outer.last\_name from employees outer

2 where outer.employee\_id NOT IN (select inner.manager\_id from employees inner);

no rows selected

This alternative solution is not a good one. The subquery picks up a NULL value, so the

entire query returns no rows. The reason is that all conditions that compare a NULL

value result in NULL. Whenever NULL values are likely to be part of the value set, do

not use NOT IN as a substitute for NOT EXISTS.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

SQL> select last\_name from employees outer

2 where outer.salary < (select AVG(inner.salary) from employees inner

3 where inner.department\_id = outer.department\_id);

LAST\_NAME

-------------------------

Chen

Sciarra

Urman

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

SQL> select last\_name from employees outer

2 where exists (select 'X' from employees inner

3 where inner.department\_id = outer.department\_id AND inner.hire\_date > outer.hire\_date AND inner.salary > outer.salary);

LAST\_NAME

-------------------------

De Haan

Austin

Lorentz

Pataballa

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

10. Write a query to display the employee ID, last names, and department names of all the employees.

Note: Use a scalar subquery to retrieve the department name in the SELECT statement.

SQL> select employee\_id, last\_name, (select department\_name from departments d

2 where e.department\_id = d.department\_id ) department from employees e

3 order by department;

EMPLOYEE\_ID LAST\_NAME DEPARTMENT

----------- ------------------------- ------------------------------

205 Higgins Accounting

206 Gietz Accounting

200 Whalen Administration

100 King Executive

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

11. 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 SUMMAR.Y

SQL> WITH summary AS (SELECT d.department\_name, SUM(e.salary) AS dept\_total FROM employees e, departments d

2 WHERE e.department\_id = d.department\_id GROUP BY d.department\_name) SELECT department\_name, dept\_total FROM summary

3 WHERE dept\_total > ( SELECT SUM(dept\_total) \* 1/8 FROM summary )

4 ORDER BY dept\_total DESC;

DEPARTMENT\_NAME DEPT\_TOTAL

------------------------------ ----------

Sales 304500

Shipping 156400

lab=7:-

--------

SQL> CREATE VIEW v AS SELECT 'www.oracle-developer.net' AS string FROM dual;

View created.

SQL> SELECT \* FROM v;

STRING

------------------------

www.oracle-developer.net

SQL> SELECT REGEXP\_SUBSTR(string, '\.[a-z-]+\.') AS url\_middle\_10g FROM v;

URL\_MIDDLE\_10G

------------------

.oracle-developer.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Write a query to find the number of occurrences of "e" in this website's address given in view V.

SQL> select regexp\_count(string, 'e') as "e count" from v;

e count

----------

5

Generate the output as shown below without using the further functions such as REPLACE, LTRIM, RTRIM or even SUBSTR in the given query.

Write a query to search the EMPLOYEES table for all the employees whose first names start with “Ki” or “Ko.”

==> select ename from employee where regexp\_like (ename, '^K(i|o).');

Create a query that removes the spaces in the STREET\_ADDRESS column of the LOCATIONS table in the display. Use “ Street Address ” as the column heading.

==>select regexp\_replace (street\_address, ' ', '') as "Street Address" from locations;

Create a query that displays “ S t” replaced by “ Street ” in the STREET\_ADDRESS column of the LOCATIONS table. Be careful that you do not affect any rows that already have “Street ” in them. Display only those rows that are affected.

==>select regexp\_replace (street\_address, 'St$','Street') from locations where regexp\_like (street\_address, 'St');

Create a contacts table and add a check constraint to the p\_number column to enforce the following format mask to ensure that phone numbers are entered into the database in the following standard format: (XXX) XXX-XXXX . The table should

have the following columns:

- l\_name varchar2(30)

- p\_number varchar2 (30)

==>create table contacts(

l\_name varchar2(30),

p\_number varchar2(30),

constraint p\_number\_format check ( regexp\_like ( p\_number, '^\(\d{3}\) \d{3}-\d{4}$')));

Run the SQL script lab\_07\_05.sql to insert the following seven phone numbers

into the contacts table. Which numbers are added?

l\_name Column Value p\_number Column Value

NULL ‘(650) 555-5555’

NULL ‘(215) 555-3427’

NULL ‘650 555-5555’

NULL ‘650 555 5555’

NULL ‘650-555-5555’

NULL ‘(650)555-5555’

NULL ‘ (650) 555-5555’

SQL> select \* from contacts;

L\_NAME P\_NUMBER

------------------------------ ------------------------------

(650) 555-5555

(215) 555-3427

==> Only the first two INSERT statements use a format that conforms to the c\_contacts\_pnf constraint; the remaining statements generate CHECK constraint errors.

ORA-02290: check constraint (PCPR1.P\_NUMBER\_FORMAT) violated;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

6.Write a query to find the number of occurrences of the DNA pattern ctc in the string

gtctcgtctcgttctgtctgtcgttctg . Ignore case-sensitivity.

SQL>select regexp\_count('gtctcgtctcgttctgtctgtcgttctg','ctc') as Count\_DNA from dual;

COUNT\_DN

----------

2