**SIDDHARTH RAMAKRISHNAN\_142897 MODULE-1 ORACLE\_DBMS - ASSIGNMENT**

**LAB 1: DATA QUERY LANGUAGE**

* 1. **Data Query Language**

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

- SELECT STAFF\_NAME AS NAME, STAFF\_SAL AS SALARY, DEPT\_CODE

FROM STAFF\_MASTERS

WHERE DEPT\_CODE IN (20, 30, 40);

**2) List the details of the employees with user defined Column headers.**

-SELECT EMPNO AS EMPID,

ENAME AS NAME,

JOB AS JOBNAME,

MGR AS MANAGER,

HIREDATE AS DATE,

SAL AS SALARY,

COMM AS COMMISSION,

DEPTNO AS DEPT\_CODE;

**3) Display the code, subjects 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,(SUBJECT1+SUBJECT2+SUBJECT3) AS TOTAL\_MARKS

FROM STUDENT\_MARKS;

**4) List the details of the staff whose designations are either PROFESSOR or LECTURER.**

-SELECT \* FROM STAFF\_MASTERS

WHERE DESIGN\_CODE IN

(SELECT DESIGN\_CODE FROM DESIGNATION\_MASTERS

WHERE DESIGN\_NAME='Professor' OR DESIGN\_NAME='Lecturer');

**5) List the code, name, and department number of the employees who have experience of more than 18 years.**

-SELECT STAFF\_CODE, STAFF\_NAME, DEPT\_CODE

FROM STAFF\_MASTERS

WHERE ROUND (MONTHS\_BETWEEN (SYSDATE, HIREDATE)/12)>18;

**6) List the name and Designations of the staff who have joined before Jan 2003.**

-SELECT S.STAFF\_NAME, D.DESIGN\_NAME, D.DESIGN\_CODE

FROM STAFF\_MASTERS S, DESIGNATION\_MASTERS D

WHERE S.DESIGN\_CODE=D.DESIGN\_CODE

AND HIREDATE<TO\_DATE ('01-JAN-03','DD-MON-YY');

**7) List the name, designation, and income for 10 years of the employees who are working in departments 10 and 30.**

-SELECT S.STAFF\_NAME, D.DESIGN\_NAME (S.STAFF\_SAL\*120) AS TOTAL\_SAL

FROM STAFF\_MASTERS S, DESIGNATION\_MASTERS D

WHERE S.DESIGN\_CODE=D.DESIGN\_CODE

AND S.DEPT\_CODE IN (10, 30);

**8) List the name and experience (in years) of employees who are working as LECTURER.**

-SELECT S.STAFF\_NAME, ROUND (MONTHS\_BETWEEN (HIREDATE,SYSDATE)/12)AS EXPERIENCE,D.DESIGN\_NAME

FROM STAFF\_MASTERS S, DESIGNATION\_MASTERS D

WHERE S.DESIGN\_CODE=D.DESIGN\_CODE

AND DESIGN\_NAME='Lecturer';

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

-SELECT CONCAT(CONCAT(Student\_name,', '),Dept\_Code)CONCAT

FROM STUDENT\_MASTERS;

**10). List the Name and Salary of the staff who are earning between 12000 and 25000. Sort them based on their salaries and name.**

-SELECT STAFF\_NAME,STAFF\_SAL

FROM STAFF\_MASTERS

WHERE STAFF\_SAL BETWEEN 12000 AND 25000

ORDER BY STAFF\_SAL,STAFF\_NAME;

**11)Display employees who do not have manager.**

-SELECT STAFF\_NAME

FROM STAFF\_MASTERS

WHERE MGR\_CODE IS NULL;

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

-SELECT STUDENT\_NAME, DEPT\_CODE,STUDENT\_DOB

FROM STUDENT\_MASTERS

WHERE STUDENT\_DOB BETWEEN '01-JAN-81' AND '31-MAR-83'

ORDER BY STUDENT\_DOB ASC;

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

-SELECT DEPT\_CODE,SUM (STAFF\_SAL)

FROM STAFF\_MASTERS

WHERE STAFF\_CODE!= MGR\_CODE

GROUP BY DEPT\_CODE

HAVING SUM (STAFF\_SAL)>20000;

**14) Display the details of books that have not been returned and expected return date was last Monday.**

- SELECT B.\* FROM BOOK\_MASTERS B, BOOK\_TRANSACTIONS T

WHERE T.BOOK\_ACTUAL\_RETURN\_DATE IS NULL;

**15) Display the 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, NVL (TO\_CHAR (DEPT\_CODE),'No Department') AS Department

FROM STUDENT\_MASTERS;

**16) Display the name and salary of the staff. Salary should be represented as X. Each X represents a 1000 in salary.**

**Sample Output**

**JOHN 10000 XXXXXXXXXX**

**ALLEN 12000 XXXXXXXXXXXX**

-SELECT STAFF\_NAME, STAFF\_SAL,LPAD ('X',(STAFF\_SAL/1000),'X')AS SALARY FROM SRAFF\_MASTERS;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**LAB 2: SINGLE ROW FUNCTIONS**

**2.1: SINGLE ROW FUNCTIONS**

**1) 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,'MONTH, DD YYYY') AS DOB

FROM STUDENT\_MASTERS

WHERE TO\_CHAR (STUDENT\_DOB,'DY') IN ('SAT','SUN');

**2) 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. Round the number of months to closest whole number.**

-SELECT STAFF\_NAME, ROUND (MONTHS\_BETWEEN (SYSDATE, HIREDATE)) AS MONTHS\_WORKED FROM STAFF\_MASTERS

ORDER BY MONTHS\_WORKED;

**3) List the details of the employees, whose names start with ‘A’ and end with ‘S’.**

- SELECT \*FROM STAFF\_MASTERS

WHERE STAFF\_NAME LIKE 'A%S';

**4) List the name and job of the employees whose names should contain N as the second or third character, and ending with either ‘N’ or ‘S’.**

-SELECT S.STAFF\_NAME, D.DESIGN\_NAME

FROM STAFF\_MASTERS S, DESIGNATION\_MASTERS D

WHERE S.DESIGN\_CODE=D.DESIGN\_CODE

AND S.STAFF\_NAME LIKE '\_n%s' OR S.STAFF\_NAME LIKE '\_n%s'OR S.STAFF\_NAME LIKE '\_\_n%n' OR S.STAFF\_NAME LIKE '\_\_n%s';

**5) 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 (STAFF\_SAL,15,'$')

FROM STAFF\_MASTERS;

**6) List the names of the Employees having ‘\_’ character in their name.**

-SELECT STAFF\_NAME FROM STAFF\_MASTERS

WHERE STAFF\_NAME LIKE '%/\_%'ESCAPE'/';

**7)** **List the details of the employees who have joined in December (irrespective of the year).**

-SELECT \* FROM STAFF\_MASTERS

WHERE TO\_CHAR (HIREDATE,'MON') = 'DEC';

**8) 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 BETWEEN 25000 AND 50000 THEN 'B'

WHEN STAFF\_SAL BETWEEN 10000 AND 25000 THEN 'C'

ELSE 'D' END AS GRADE FROM STAFF\_MASTERS;

**9) 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.**

-SELECT STAFF\_NAME, HIREDATE, TO\_CHAR (HIREDATE,'DY') AS DAY

FROM STAFF\_MASTERS

ORDER BY

CASE

WHEN TO\_CHAR (HIREDATE,'D') = 2 THEN 1

WHEN TO\_CHAR (HIREDATE,'D') = 3 THEN 2

WHEN TO\_CHAR (HIREDATE,'D') = 4 THEN 3

WHEN TO\_CHAR (HIREDATE,'D') = 5 THEN 4

WHEN TO\_CHAR (HIREDATE,'D') = 6 THEN 5

WHEN TO\_CHAR (HIREDATE,'D')=7 THEN 6

WHEN TO\_CHAR (HIREDATE,'D')=1 THEN 7

END;

**10) Show staff names with the respective numbers of asterisk from Staff table.**

-SELECT STAFF\_NAME,LPAD ('\*',LENGTH(STAFF\_NAME),'\*')AS ASTERISK

FROM STAFF\_MASTERS;

**11) Show staff names with the respective numbers of asterisk from Staff table except first and last characters. For example: KING will be replaced with K\*\*G.**

**-**SELECT RPAD(RPAD(SUBSTR(STAFF\_NAME,1,1),(LENGTH(STAFF\_NAME)-1),’\*’)

,LENGTH(STAFF\_NAME),SUBSTR(STAFF\_NAME,-1))

FROM STAFF\_MASTERS;

**12) Show all staffs who were hired in the first half of the month.**

-SELECT STAFF\_NAME,HIREDATE FROM STAFF\_MASTERS

WHERE TO\_CHAR(HIREDATE,'DD')<=15;

**13) Display the staff name, hire date and day of the week on which the staff joined. Order the results by the day of the week starting with Monday.**

**-**SELECT STAFF\_NAME, HIREDATE, TO\_CHAR (HIREDATE,'DY') AS DAY

FROM STAFF\_MASTERS

ORDER BY

CASE

WHEN TO\_CHAR (HIREDATE,'D') = 2 THEN 1

WHEN TO\_CHAR (HIREDATE,'D') = 3 THEN 2

WHEN TO\_CHAR (HIREDATE,'D') = 4 THEN 3

WHEN TO\_CHAR (HIREDATE,'D') = 5 THEN 4

WHEN TO\_CHAR (HIREDATE,'D') = 6 THEN 5

WHEN TO\_CHAR (HIREDATE,'D')=7 THEN 6

WHEN TO\_CHAR (HIREDATE,'D')=1 THEN 7

END;

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

-SELECT INSTR ('Mississippi','i', 2, 3) from dual;

**15) 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.**

**-**SELECT TO\_CHAR (NEXT\_DAY (SYSDATE,’FRIDAY’),’’DDspth “of”

MONTH, YYYY’)AS “PAY DATE”

FROM DUALWHERE

NEXT\_DAY(SYSDATE,’FRIDAY’)<LAST\_DAY(SYSDATE);

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**2.2: GROUP FUNCTIONS**

**16) Display the Highest, Lowest, Total & Average salary of all staff. Label the columns Maximum, Minimum, Total and Average respectively. Round the result to nearest whole number.**

-SELECT ROUND(MAX(STAFF\_SAL))AS MAXIMUM,

ROUND(MIN(STAFF\_SAL))AS MINIMUM,

ROUND(SUM(STAFF\_SAL)) AS TOTAL,

ROUND(AVG(STAFF\_SAL)) AS AVERAGE

FROM STAFF\_MASTERS;

**17) Edit the above query and display the same for each Department Name.**

-SELECT ROUND(MAX(STAFF\_SAL))AS MAXIMUM,

ROUND(MIN(STAFF\_SAL))AS MINIMUM,

ROUND(SUM(STAFF\_SAL)) AS TOTAL,

ROUND(AVG(STAFF\_SAL)) AS AVERAGE

FROM STAFF\_MASTERS

GROUP BY DEPT\_CODE;

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

-SELECT S.DEPT\_CODE,D.DEPT\_NAME,COUNT(S.DEPT\_CODE) AS NO\_OF\_PEOPLE

FROM STAFF\_MASTERS S,DEPARTMENT\_MASTERS D

WHERE S.DEPT\_CODE=D.DEPT\_CODE

GROUP BY S.DEPT\_CODE,D.DEPT\_NAME;

**19) Determine the number of managers without listing them. Label the column as ‘Total Number of Managers’.**

-SELECT COUNT (DISTINCT MGR\_CODE) AS TOTAL\_NO\_OF\_MANAGERS

FROM STAFF\_MASTERS;

**20) 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 MGR\_CODE,STAFF\_NAME,MIN(STAFF\_SAL) FROM STAFF\_MASTERS

WHERE STAFF\_CODE IN(SELECT DISTINCT MGR\_CODE FROM STAFF\_MASTERS)

GROUP BY MGR\_CODE,

STAFF\_NAME HAVING MIN(STAFF\_SAL) > 10000;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**LAB 3: JOINS AND SUBQUERIES**

**3.1: JOINS AND SUBQUERIES**

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

**//BY JOINS**

- SELECT S.STAFF\_NAME, S.DEPT\_CODE, D.DEPT\_NAME, S.STAFF\_SAL

FROM STAFF\_MASTERS S, DEPARTMENT\_MASTERS D

WHERE S.DEPT\_CODE=D.DEPT\_CODE

AND STAFF\_SAL>20000;

**//BY JOINS AND SUBQUERY**

- SELECT STAFF\_NAME,S.DEPT\_CODE,DEPT\_NAME,STAFF\_SAL

FROM STAFF\_MASTERS S,DEPARTMENT\_MASTERS D

WHERE S.DEPT\_CODE=D.DEPT\_CODE

AND STAFF\_SAL IN(SELECT STAFF\_SAL FROM STAFF\_MASTERS

WHERE STAFF\_SAL>20000);

**2) Write a query to display Staff Name, Department Code, and Department Name for all staff who do not work in Department code 10 and have ‘A’ in their name.**

- SELECT S.STAFF\_NAME, S.DEPT\_CODE, D.DEPT\_NAME

FROM STAFF\_MASTERS S, DEPARTMENT\_MASTERS D

WHERE S.DEPT\_CODE=D.DEPT\_CODE

AND S.DEPT\_CODE!=10

AND LOWER (S.STAFF\_NAME) LIKE '%a%';

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

- SELECT E.STAFF\_CODE AS STAFF#, E.STAFF\_NAME AS STAFF, DEPT\_NAME,

M.STAFF\_CODE AS MGR#, M.STAFF\_NAME AS MANAGER FROM

STAFF\_MASTERS E, STAFF\_MASTERS M, DEPARTMENT\_MASTERS D

WHERE E.MGR\_CODE=M.STAFF\_CODE AND E.DEPT\_CODE=D.DEPT\_CODE;

**4) Create a query that will display Student Code, Student Name, Department Name, Subjec1, Subject2, and Subject3 for all students who are getting 60 and above in each subject from department 10 and 20.**

-SELECT S.STUDENT\_CODE, S.STUDENT\_NAME,

M.SUBJECT1, M.SUBJECT2, M.SUBJECT3, D.DEPT\_NAME

FROM STUDENT\_MASTERS S, STUDENT\_MARKS M, DEPARTMENT\_MASTERS D

WHERE S.STUDENT\_CODE=M.STUDENT\_CODE

AND S.DEPT\_CODE=D.DEPT\_CODE

AND SUBJECT1>60

AND SUBJECT2>60

AND SUBJECT3>60

AND S.DEPT\_CODE IN (10, 20);

**5) 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 S.STUDENT\_CODE, S.STUDENT\_NAME, B.BOOK\_CODE, B.BOOK\_NAME, T.BOOK\_EXPECTED\_RETURN\_DATE

FROM STUDENT\_MASTERS S, BOOK\_MASTERS B, BOOK\_TRANSACTIONS T

WHERE S.STUDENT\_CODE=T.STUDENT\_CODE

AND B.BOOK\_CODE=T.BOOK\_CODE

AND T.BOOK\_EXPECTED\_RETURN\_DATE='09-FEB-11';

**6) Create a query that will display Staff Code, Staff Name, Department Name, Designation, Book Code, Book Name, and Issue Date. For only those staff who have taken any book in last 30 days.**

-SELECT

S.STAFF\_NAME,S.STAFF\_CODE,D.DEPT\_NAME,DE.DESIGN\_NAME,BT.BOOK\_CODE,B.BOOK\_NAME,BT.BOOK\_ISSUE\_DATE

FROM STAFF\_MASTERS S, DEPARTMENT\_MASTERS D, DESIGNATION\_MASTERS DE, BOOK\_MASTERS B, BOOK\_TRANSACTIONS BT

WHERE S.DEPT\_CODE=D.DEPT\_CODE AND S.DESIGN\_CODE=DE.DESIGN\_CODE AND S.STAFF\_CODE=BT.STAFF\_CODE

AND B.BOOK\_CODE=BT.BOOK\_CODE AND BT.BOOK\_ISSUE\_DATE BETWEEN SYSDATE AND (SYSDATE-30);

**7) Generate a report which contains the following information.**

**Staff Code, Staff Name, Designation, Department Name , Department Head**

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

-SELECT S.STAFF\_CODE, S.STAFF\_NAME, D.DESIGN\_NAME, DEPT.DEPT\_NAME

FROM STAFF\_MASTERS S, DESIGNATION\_MASTERS D, DEPARTMENT\_MASTERS DEPT WHERE D.DESIGN\_NAME != 'HOD' ;

**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 S.STUDENT\_CODE "STUDENT CODE", S.STUDENT\_NAME "STUDENT NAME", D.DEPT\_NAME " DEPARMENT NAME", (SU.SUBJECT1+SU.SUBJECT2+SU.SUBJECT3) AS TOTALMARKS,' ' "HOD NAME"

FROM STUDENT\_MASTERS S, DEPARTMENT\_MASTERS D,STUDENT\_MARKS SU

WHERE (S.STUDENT\_CODE=SU.STUDENT\_CODE AND (D.DEPT\_CODE=S.DEPT\_CODE)) ORDER BY D.DEPT\_NAME, TOTALMARKS;

**9) Generate a report which contains the following information.**

**Staff Code, Staff Name, Designation, Department, Book Code, Book Name, Author, Fine**

**For the staff who have not return the book. Fine will be calculated as Rs. 5 per day.**

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

**-**SELECT S.STAFF\_CODE "STAFF CODE", S.STAFF\_NAME "STAFF NAME", DE.DESIGN\_NAME "DESIGNATION",D.DEPT\_NAME "DEPARTMENT",BT.BOOK\_CODE "BOOK CODE",

B.BOOK\_NAME "BOOK NAME", B.BOOK\_PUB\_AUTHOR "AUTHOR",ROUND (5\*(SYSDATE-BT.BOOK\_EXPECTED\_RETURN\_DATE)) "FINE"

FROM STAFF\_MASTERS S, DEPARTMENT\_MASTERS D,DESIGNATION\_MASTERS DE,BOOK\_MASTERS B,BOOK\_TRANSACTIONS BT

WHERE S.DEPT\_CODE=D.DEPT\_CODE AND S.DESIGN\_CODE=DE.DESIGN\_CODE AND S.STAFF\_CODE=BT.STAFF\_CODE

AND B.BOOK\_CODE=BT.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 "STAFF CODE",STAFF\_NAME "STAFF NAME",STAFF\_SAL "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,STAFF\_NAME FROM STAFF\_MASTERS

WHERE STAFF\_CODE NOT IN(SELECT DISTINCT MGR\_CODE FROM STAFF\_MASTERS);

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

-SELECT BOOK\_PUB\_AUTHOR, BOOK\_NAME FROM BOOK\_MASTERS

WHERE BOOK\_PUB\_AUTHOR IN

(SELECT BOOK\_PUB\_AUTHOR FROM BOOK\_MASTERS GROUP BY BOOK\_PUB\_AUTHOR HAVING COUNT (BOOK\_NAME)>1);

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

-SELECT S.STAFF\_CODE, S.STAFF\_NAME, D.DEPT\_NAME FROM STAFF\_MASTERS S JOIN DEPARTMENT\_MASTERS D ON S.DEPT\_CODE=D.DEPT\_CODE

WHERE S.STAFF\_CODE IN (SELECT STAFF\_CODE FROM BOOK\_TRANSACTIONS

GROUP BY STAFF\_CODE HAVING COUNT (BOOK\_CODE)>1);

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

**Student Code, Student Name, Department Name, Subject1, Subject2, Subject3, Total.**

-SELECT S.STUDENT\_CODE, S.STUDENT\_NAME, D. DEPT\_NAME,B.SUBJECT1,B.SUBJECT2,

B.SUBJECT3, B.SUBJECT1+B.SUBJECT2+B.SUBJECT3 AS TOTAL

FROM STUDENT\_MASTERS S JOIN DEPARTMENT\_MASTERS D ON S.DEPT\_CODE=D.DEPT\_CODE JOIN STUDENT\_MARKS B ON S.STUDENT\_CODE=B.STUDENT\_CODE

ORDER BY D.DEPT\_CODE, 7 DESC;

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

-SELECT S.STAFF\_NAME,D.DEPT\_NAME,S.STAFF\_SAL

FROM STAFF\_MASTERS S,DEPARTMENT\_MASTERS D

WHERE S.DEPT\_CODE=D.DEPT\_CODE

AND S.STAFF\_SAL<(SELECT AVG(STAFF\_SAL) FROM STAFF\_MASTERS WHERE

D.DEPT\_CODE=S.DEPT\_CODE GROUP BY DEPT\_CODE);

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

-SELECT S.STAFF\_NAME,D.DEPT\_NAME FROM STAFF\_MASTERS S,DEPARTMENT\_MASTERS D

WHERE S.DEPT\_CODE=D.DEPT\_CODE;

**17) List the Student Code, Student Name for that student who got highest marks in all three subjects in Computer Science department for current year.**

-SELECT S.STUDENT\_CODE,S.STUDENT\_NAME

FROM STUDENT\_MASTERS S JOIN STUDENT\_MARKS M

ON S.STUDENT\_CODE=M.STUDENT\_CODE

WHERE S.STUDENT\_CODE IN (SELECT STUDENT\_CODE

FROM STUDENT\_MARKS

WHERE SUBJECT1 = (SELECT MAX(SUBJECT1)

FROM STUDENT\_MARKS)

AND SUBJECT2 = (SELECT MAX(SUBJECT2)

FROM STUDENT\_MARKS)

AND SUBJECT3 = (SELECT MAX(SUBJECT3)

FROM STUDENT\_MARKS))

AND S.DEPT\_CODE = 10

AND M.STUDENT\_YEAR = TO\_CHAR(SYSDATE,'YYYY');

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

-SELECT SM.STUDENT\_CODE,SM.STUDENT\_NAME,DM.DEPT\_NAME FROM STUDENT\_MASTERS SM,DEPARTMENT\_MASTERS DM

WHERE SM.DEPT\_CODE=(SELECT DEPT\_CODE FROM(SELECT SM.DEPT\_CODE,COUNT(SM.DEPT\_CODE) CT FROM STUDENT\_MASTERS SM,DEPARTMENT\_MASTERS DM

WHERE SM.DEPT\_CODE=DM.DEPT\_CODE GROUP BY SM.DEPT\_CODE)

WHERE CT=(SELECT MAX(CT)

FROM(SELECT SM.DEPT\_CODE,COUNT(SM.DEPT\_CODE) CT

FROM STUDENT\_MASTERSSM,DEPARTMENT\_MASTERS DM

WHERE SM.DEPT\_CODE=DM.DEPT\_CODE

GROUP BY SM.DEPT\_CODE)))

AND DM.DEPT\_CODE=(SELECT DEPT\_CODE

FROM(SELECT SM.DEPT\_CODE,COUNT(SM.DEPT\_CODE) CT FROM STUDENT\_MASTERS SM,DEPARTMENT\_MASTERS DM

WHERE SM.DEPT\_CODE=DM.DEPT\_CODE GROUP BY SM.DEPT\_CODE)

WHERE CT=(SELECT MAX(CT) FROM(SELECT SM.DEPT\_CODE,COUNT(SM.DEPT\_CODE) CT

FROM STUDENT\_MASTERS SM,DEPARTMENT\_MASTERS DM

WHERE SM.DEPT\_CODE=DM.DEPT\_CODE GROUP BY SM.DEPT\_CODE)));

**19) Display Staff Code, Staff Name, Department Name, and Designation for those who have joined most recently.**

-SELECT S.STAFF\_CODE, S.STAFF\_NAME, DP.DEPT\_NAME, DS.DESIGN\_NAME

FROM STAFF\_MASTERS S JOIN DEPARTMENT\_MASTERS DP

ON S.DEPT\_CODE = DP.DEPT\_CODE

JOIN DESIGNATION\_MASTERS DS

ON S.DESIGN\_CODE = DS.DESIGN\_CODE

WHERE S.HIREDATE = (SELECT MAX (HIREDATE)

FROM STAFF\_MASTERS);

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

-SELECT SM.STAFF\_NAME AS MANAGER\_NAME,COUNT(SM1.MGR\_CODE) AS TOTAL\_STRENGTH FROM STAFF\_MASTERS SM,STAFF\_MASTERS SM1

WHERE SM.STAFF\_CODE=SM1.MGR\_CODE GROUP BY SM.STAFF\_NAME;

**3.2: SET OPERATORS**

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

- SELECT \* FROM PREVIOUS\_PRODUCTS

UNION

SELECT \* FROM CURRENT\_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 PREVIOUS\_PRODUCTS

UNION ALL

SELECT \* FROM CURRENT\_PRODUCTS;

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

- SELECT \* FROM PREVIOUS\_PRODUCTS

INTERSECT

SELECT \* FROM CURRENT\_PRODUCTS;

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

**-**SELECT \* FROM CURRENT\_PRODUCTS

MINUS

SELECT \* FROM PREVIOUS\_PRODUCTS;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**LAB 04: DATABASE OBJECTS**

## 

## 4.1: DATABASE OBJECTS

1) **Create the Customer table with the following columns.**

**Customerid Number(5)**

**CustomerName Number(10)**

**Address1 Varchar2(30)**

**Address2 Varchar2(30)**

- CREATE TABLE CUSTOMER (

CUSTOMERID NUMBER (5),

CUSTOMERNAME NUMBER (10),

ADDRESS1 VARCHAR2 (30),

ADDRESS2 VARCHAR2 (30));

**2) Modify the Customer table CustomerName column of datatype with Varchar2(30). CustomerName should not accept Nulls.**

-ALTER TABLE CUSTOMER

MODIFY (CUSTOMERNAME VARCHAR2(30) NOT NULL);

**3) Add the following Columns to the Customer table.**

**Gender Varchar2(1)**

**Age Number(3)**

**PhoneNo Number(10)**

-ALTER TABLE CUSTOMER

ADD (GENDER VARCHAR2 (1),

AGE NUMBER (3),

PHONENO NUMBER (10)

);

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

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

**In similar manner, add the below records to the Customer table:**

* **1000, Allen, #115 Chicago, #115 Chicago, M, 25, 7878776**
* **1001, George, #116 France, #116 France, M, 25, 434524**
* **1002, Becker, #114 New York, #114 New York, M, 45, 431525**

- INSERT INTO CUSTOMER VALUES

(&CUSTOMERID,'&CUSTOMERNAME','&ADDRESS1','&ADDRESS2','&M', &AGE, &PHONENO);

Enter value for customerid: 1000

Enter value for customername: Allen

Enter value for address1: #115 Chicago

Enter value for address2: #115 Chicago

Enter value for m: M

Enter value for age: 25

Enter value for phoneno: 7878776

1 row created.

/

Enter value for customerid: 1001

Enter value for customername: George

Enter value for address1: #116 France

Enter value for address2: #116 France

Enter value for m: M

Enter value for age: 25

Enter value for phoneno: 434524

1 row created.

/

Enter value for customerid: 1002

Enter value for customername: Becker

Enter value for address1: #114 New York

Enter value for address2: #114 New York

Enter value for m: M

Enter value for age: 45

Enter value for phoneno: 431525

1 row created.

**5) Add the Primary key constraint for Customerld with the name Custld\_Prim.**

- ALTER TABLE CUSTOMER

ADD CONSTRAINT Custid\_Prim PRIMARY KEY (CUSTOMERID);

**6) 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 CUSTOMER VALUES

(&CUSTOMERID,'&CUSTOMERNAME','&ADDRESS1','&ADDRESS2','&M',&AGE,&PHONENO);

(1002,'John','#114 Chicago','#114 Chicago','M',45,439525)

ERROR at line 1:

ORA-00001: unique constraint (SCOTT.CUSTID) violated

**7) 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 CUSTOMER DISABLE CONSTRAINT CUSTID;

INSERT INTO CUSTOMER VALUES

(&CUSTOMERID,'&CUSTOMERNAME','&ADDRESS1','&ADDRESS2','&M', &AGE, &PHONENO);

Enter value for customerid: 1002

Enter value for customername: Becker

Enter value for address1: #114 New York

Enter value for address2: #114 New York

Enter value for m: M

Enter value for age: 45

Enter value for phoneno: 431525

1 row created.

/

Enter value for customerid: 1003

Enter value for customername: Nanapatekar

Enter value for address1: #115 India

Enter value for address2: #115 India

Enter value for m: M

Enter value for age: 45

Enter value for phoneno: 431525

1 row created.

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

- ALTER TABLE CUSTOMER

ENABLE CONSTRAINT CUSTID;

ERROR at line 1:

ORA-02437: cannot validate (SCOTT.CUSTID) - primary key violated

**9) 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 CUSTOMER DROP CONSTRAINT CUSTID;

INSERT INTO CUSTOMER VALUES(1002,'Becker','#114 New York','#114 New York','

M', 45, 431525, 15000.50);

ERROR at line 1:

ORA-00913: too many values

INSERT INTO CUSTOMER VALUES(1003,'Nanapatekar','#115 India','#115 India','

M', 45, 431525, 20000.50);

ERROR at line 1:

ORA-00913: too many values

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

- TRUNCATE TABLE CUSTOMER;

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

- ALTER TABLE CUSTOMER ADD(EMAIL VARCHAR2(10));

**12) Drop the E\_mail column from Customer table.**

- ALTER TABLE CUSTOMER DROP COLUMN EMAIL;

**13) Add a new column EmailId to Customer table.**

- ALTER TABLE CUSTOMER ADD(EMAILID NUMBER(10));

**14) Mark EmailId column as unused before dropping it.**

- ALTER TABLE CUSTOMER SET UNUSED COLUMN EMAILID;

**15) Drop the unused EmailId column from the Customer table.**

- ALTER TABLE CUSTOMER DROP UNUSED COLUMN;

**16) Define the COMMENT ‘Customers Details’ for Customer table.**

- COMMENT ON TABLE CUSTOMER IS 'CUSTOMER DETAILS';

**17) Use Data Dictionary USER\_TAB\_COMMENTS to view the created comment.**

- SELECT \* FROM USER\_TAB\_COMMENTS WHERE TABLE\_NAME='CUSTOMER';

**18) Define the COMMENT ‘Personal Contact no’ for the phoneno column of the Customer table.**- COMMENT ON COLUMN CUSTOMER.PHONENO IS 'PERSONAL CONTACT NO';

**19) Use Data Dictionary USER\_COL\_COMMENTS to view the created comment.**

- SELECT \* FROM USER\_COL\_COMMENTS WHERE COLUMN\_NAME='PHONENO';

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

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

- CREATE TABLE SUPPLIERS (

SUPPID NUMBER (5),

SNAME VARCHAR2 (30),

ADDRESS1 VARCHAR2 (30),

ADDRESS2 VARCHAR2 (30),

CONTACTNO NUMBER (10));

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

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

**CustomerName Varchar2(30) Not Null**

**Addressl Varchar2(30) Not Null**

**Address2 Varchar2(30)**

**Gender Varchar2(l)**

**Age Number(3)**

**PhoneNo Number(10)**

- CREATE TABLE CUSTOMERMASTER (

CUSTOMERID NUMBER (5), CONSTRAINT CUSTID\_PK PRIMARY KEY(CUSTOMERID), CUSTOMERNAME VARCHAR2(30) NOT NULL,

ADDRESS1 VARCHAR2(30) NOT NULL, ADDRESS2 VARCHAR2(30),

GENDER VARCHAR2(1), AGE NUMBER(3),

PHONENO NUMBER(10));

**22) Create the AccountsMaster table with the following Columns. Use sequence to generate Account number**

**Customerld Number(5)**

**AccountNumber Number(10,2) Primary key(Name of constraint is Acc\_PK)**

**AccountType Char(3)**

**LedgerBalance Number(10,2) Not Null**

- CREATE TABLE ACCOUNTSMASTER(

CUSTOMERID NUMBER(5),ACCOUNTNUMBER NUMBER(10,2),CONSTRAINT ACC\_PK PRIMARY KEY(ACCOUNTNUMBER),ACCOUNTTYPE CHAR(3),

LEDGERBALANCE NUMBER(10,2) NOT NULL);

-CREATE SEQUENCE SEQ\_ANO

MINVALUE 101

MAXVALUE 10000

START WITH 101

INCREMENT BY 1

CACHE 101;

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

- ALTER TABLE ACCOUNTSMASTER ADD CONSTRAINT ASS\_FK FOREIGN KEY

(CUSTOMERID) REFERENCES CUSTOMERMASTER(CUSTOMERID);

**24) Insert the following rows to the CustomerMaster table:**

* **1000, Allen, #115 Chicago, #115 Chicago, M, 25, 7878776**
* **1001, George, #116 France, #116 France, M, 25, 434524**
* **1002, Becker, #114 New York, #114 New York, M, 45, 431525**

-INSERT INTO CUSTOMERMASTER VALUES(&CUSTOMERID,'&CUSTOMERNAME','&ADDRESS1','&ADDRESS2','&GENDER',&AGE,&PHONENO);

Enter value for customerid: 1000

Enter value for customername: Allen

Enter value for address1: #115 Chicago

Enter value for address2: #115 Chicago

Enter value for gender: M

Enter value for age: 25

Enter value for phoneno: 7878776

1 row created.

/

Enter value for customerid: 1001

Enter value for customername: George

Enter value for address1: #116 France

Enter value for address2: #116 France

Enter value for gender: M

Enter value for age: 25

Enter value for phoneno: 434524

1 row created.

/

Enter value for customerid: 1002

Enter value for customername: Becker

Enter value for address1: #114 New York

Enter value for address2: #114 New York

Enter value for gender: M

Enter value for age: 45

Enter value for phoneno: 431525

1 row created.

**25) Modify the AccountMaster table with the Check constraint to ensure AccountType should be either NRI or IND.**

- ALTER TABLE ACCOUNTSMASTER ADD CONSTRAINT CK\_AC CHECK(ACCOUNTTYPE='NRI' OR ACCOUNTTYPE='IND');

**26) Insert 5 rows into the AccountsMaster table:**

-INSERT INTO ACCOUNTSMASTER VALUES ('&CUID',ACCNUM\_SEQ.NEXTVAL, '&C', '&D');

ENTER VALUE FOR A: 1000

ENTER VALUE FOR C: NRI

ENTER VALUE FOR D: 1000

OLD 1: INSERT INTO ACCOUNTSMASTER VALUES ('&A',ACCNUM\_SEQ.NEXTVAL, '&C', '&D')

NEW 1: INSERT INTO ACCOUNTSMASTER VALUES ('1000',ACCNUM\_SEQ.NEXTVAL, 'NRI', '1000')

1 ROW CREATED.

SQL> /

ENTER VALUE FOR A: 1001

ENTER VALUE FOR C: IND

ENTER VALUE FOR D: 1001

OLD 1: INSERT INTO ACCOUNTSMASTER VALUES ('&A',ACCNUM\_SEQ.NEXTVAL, '&C', '&D')

NEW 1: INSERT INTO ACCOUNTSMASTER VALUES ('1001',ACCNUM\_SEQ.NEXTVAL, 'IND', '1001')

1 ROW CREATED.

SQL> /

ENTER VALUE FOR A: 1002

ENTER VALUE FOR C: NRI

ENTER VALUE FOR D: 2001

OLD 1: INSERT INTO ACCOUNTSMASTER VALUES ('&A',ACCNUM\_SEQ.NEXTVAL, '&C', '&D')

NEW 1: INSERT INTO ACCOUNTSMASTER VALUES ('1002',ACCNUM\_SEQ.NEXTVAL, 'NRI', '2001')

1 ROW CREATED.

SQL> /

ENTER VALUE FOR A: 1000

ENTER VALUE FOR C: NRI

ENTER VALUE FOR D: 2001

OLD 1: INSERT INTO ACCOUNTSMASTER VALUES ('&A',ACCNUM\_SEQ.NEXTVAL, '&C', '&D')

NEW 1: INSERT INTO ACCOUNTSMASTER VALUES ('1000',ACCNUM\_SEQ.NEXTVAL, 'NRI', '2001')

1 ROW CREATED.

SQL> /

ENTER VALUE FOR A: 1001

ENTER VALUE FOR C: IND

ENTER VALUE FOR D: 100000

OLD 1: INSERT INTO ACCOUNTSMASTER VALUES ('&A',ACCNUM\_SEQ.NEXTVAL, '&C', '&D')

NEW 1: INSERT INTO ACCOUNTSMASTER VALUES ('1001',ACCNUM\_SEQ.NEXTVAL, 'IND', '100000')

**27) Modify the AccountsMaster table keeping a Check constraint with the name Balance\_Check for the Minimum Balance which should be greater than 5000.**

-ALTER TABLE ACCOUNTMASTER

ADD CONSTRAINT BALANCE\_CHECK CHECK(LEDGERBALANCE > 5000);

**28) Modify the AccountsMaster table such that if Customer is deleted from Customer table then all his details should be deleted from AccountsMaster table.**

-ALTER TABLE ACCOUNTMASTER

DROP CONSTRAINT CUST\_ACC;

**29) Create Backup copy for the AccountsMaster table with the name ‘AccountDetails’.**

-CREATE TABLE ACCOUNTDETAILS AS SELECT \* FROM ACCOUNTMASTER;

**30) Change the name of the AccountDetails table to ‘BackUpTable’ table.**

-ALTER TABLE ACCOUNTDETAILS

RENAME TO BACKUPTABLE;

**31) Create a view ‘Acc\_view’ with columns Customerld, CustomerName, AccountNumber, AccountType, and LedgerBalance from AccountsMaster. In the view Acc\_view, the column names should be CustomerCode, AccountHolderName, AccountNumber, Type, and Balance for the respective columns from AccountsMaster table.**

-CREATE VIEW ACC\_VIEW

AS SELECT A.CUSTOMERID AS CUSTOMERCODE, C.CUSTOMERNAME AS ACCOUNTHOLDERNAME, A.ACCOUNTNUMBER, A.ACCOUNTTYPE AS TYPE, A.LEDGERBALANCE AS BALANCE

FROM ACCOUNTMASTER A,CUSTOMER C WHERE C.CUSTOMERID=A.CUSTOMERID;

**32) Create a view on AccountsMaster table with name vAccs\_Dtls. This view should list all customers whose AccountType is ‘IND’ and their balance amount should not be less than 10000. Using this view any DML operation should not violate the view conditions.**

**hint2Hint:** Use the With Check Option constraint.

-CREATE VIEW VACCS\_DTLS AS

SELECT \* FROM ACCOUNTMASTER WHERE ACCOUNTTYPE='IND' AND LEDGERBALANCE>10000

WITH CHECK OPTION;

**33) Create a view accsvw10 which will not allow DML statement against it.**

-CREATE VIEW ACCSVW10AS

SELECT \* FROM ACCOUNTMASTER WITH READ ONLY;

**34) Display the department from Staff table which has the highest salary by using Inline View.**

-SELECT DEPT\_CODE FROM(SELECT DEPT\_CODE FROM STAFF\_MASTERS ORDER BY STAFF\_SAL DESC) WHERE ROWNUM=1;

**35) List the top two highest earning employees in each department.**

-SELECT \* FROM(SELECT STAFF\_NAME,STAFF\_SAL,DEPT\_CODE,RANK() OVER (PARTITION BY DEPT\_CODE ORDER BY STAFF\_SAL DESC) RANK FROM STAFF\_MASTERS) WHERE RANK<=2;

**36) Create a Sequence with the name Seq\_Dept on Deptno column of Dept table. It should start from 40 and stop at 200. Increment parameter for the sequence Seq\_Dept should be in step of 10.**

-CREATE SEQUENCE SEQ\_DEPT

START WITH 40

INCREMENT BY 10

MAXVALUE 200

NOCYCLE

NOCACHE;

**37) Insert three sample rows by using the above sequence in Dept table.**

-INSERT INTO DEPARTMENT VALUES (SEQ\_DEPT.NEXTVAL, 'FINANCE');

INSERT INTO DEPARTMENT VALUES (SEQ\_DEPT.NEXTVAL, 'MARKETING');

INSERT INTO DEPARTMENT VALUES (SEQ\_DEPT.NEXTVAL, 'HUMANRESOURCES');

**38) Alter the above specified sequence with an increment by 5.**

-ALTER SEQUENCE SEQ\_DEPT

INCREMENT BY 5;

**39) Drop the Seq\_Dept sequence.**

-DROP SEQUENCE SEQ\_DEPT;

**40) Create a Unique index with the name No\_Name on DeptNo and Dname of Dept table.**

-CREATE INDEX NO\_NAME ON DEPARTMENT\_MASTERS(DEPT\_CODE,DEPT\_NAME);

**41) Get information on the index No\_Name from the Data Dictionary.**

-SELECT INDEX\_NAME,TABLE\_NAME FROM USER\_INDEXES WHERE INDEX\_NAME='NO\_NAME';

**42) Create public synonym synEmp for the EMP table.**

-CREATE PUBLIC SYNONYM SYNEMP FOR EMP;

**43) Get Information on synonym synEmp from the Data Dictionary.**

-SELECT \* FROM ALL\_SYNONYMS WHERE SYNONYM\_NAME='SYNEMP';

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**LAB 5: DATA MANIPULATION LANGUAGE**

## 5.1: DATA MANIPULATION LANGUAGE

**1) Create Employee table with same structure as EMP table.**

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

**SQL>desc employee**

|  |  |  |
| --- | --- | --- |
| **Name** | **Null?** | **Type** |
| EMPNO | NOT NULL | NUMBER(4) |
| ENAME |  | VARCHAR2(10) |
| JOB |  | VARCHAR2(50) |
| MGR |  | NUMBER(4) |
| HIREDATE |  | DATE |
| SAL |  | NUMBER(7,2) |
| COMM |  | NUMBER(7,2) |
| DEPTNO |  | NUMBER(2) |

**SQL>select \* from employee**

->> SQL:

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

Table created.

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)

SQL> select \* from employee ;

no rows selected

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

**SQL>select \* from employee**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **EMPNO** | **ENAME** | **JOB** | **MGR** | **HIREDATE** | **SAL** | **COMM** | **DEPTNO** |
| 7369 | SMITH |  |  |  | 800 |  | 20 |
| 7499 | ALLEN |  |  |  | 1600 |  | 30 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 7521 | WARD |  |  |  | 1250 |  | 30 |
| 7566 | JONES |  |  |  | 2975 |  | 20 |
| 7654 | MARTIN |  |  |  | 1250 |  | 30 |
| 7698 | BLAKE |  |  |  | 2850 |  | 30 |
| 7782 | CLARK |  |  |  | 2450 |  | 10 |
| 7788 | SCOTT |  |  |  | 3000 |  | 20 |
| 7839 | KING |  |  |  | 5000 |  | 10 |
| 7844 | TURNER |  |  |  | 1500 |  | 30 |
| 7876 | ADAMS |  |  |  | 1100 |  | 20 |
| 7900 | JAMES |  |  |  | 950 |  | 30 |
| 7902 | FORD |  |  |  | 3000 |  | 20 |
| 7934 | MILLER |  |  |  | 1300 |  | 10 |

->> SQL:

insert into employee

(empno,ename,sal,deptno)

select empno,ename,sal,deptno

from emp

where 1=1;

14 rows created.

**3) 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:

UPDATE EMPLOYEE SET JOB=(SELECT JOB FROM EMP WHERE EMPNO=7788), DEPTNO=(SELECT DEPTNO FROM EMP WHERE EMPNO=7788)

WHERE EMPNO=7698;

1 row updated.

**4) Delete the details of department whose department name is ‘SALES’.**

-DELETE FROM emp

WHERE deptno=(SELECT deptno FROM dept

WHERE dname=’SALES’);

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

**-**UPDATE emp

SET deptno=(SELECT deptno

FROM emp

WHERE empno=7698)

WHERE empno=7788;

**6) Insert the following rows to the Employee table through parameter substitution.**

* **1000,Allen, Clerk,1001,12-jan-01, 3000, 2,10**
* **1001,George, analyst, null, 08 Sep 92, 5000,0, 10**
* **1002, Becker, Manager, 1000, 4 Nov 92, 2800,4, 20**
* **1003, 'Bill', Clerk, 1002, 4 Nov 92,3000, 0, 20**

-INSERT INTO EMPLOYEE VALUES

(&EMPNO,'&ENAME','&JOB',&MGR,'&HIREDATE',&SAL,&COMM,&DEPTNO);

ENTER VALUE FOR EMPNO: 1000

ENTER VALUE FOR ENAME: ALLEN

ENTER VALUE FOR JOB: CLERK

ENTER VALUE FOR MGR: 1001

ENTER VALUE FOR HIREDATE: 12-JAN-01

ENTER VALUE FOR SAL: 3000

ENTER VALUE FOR COMM: 2

ENTER VALUE FOR DEPTNO: 10

OLD 2: (&EMPNO,'&ENAME','&JOB',&MGR,'&HIREDATE',&SAL,&COMM,&DEPTNO)

NEW 2: (1000,'ALLEN','CLERK',1001,'12-JAN-01',3000,2,10)

1 ROW CREATED.

SQL> INSERT INTO EMPLOYEE VALUES

(&EMPNO,'&ENAME','&JOB',&MGR,'&HIREDATE',&SAL,&COMM,&DEPTNO);

ENTER VALUE FOR EMPNO: 1001

ENTER VALUE FOR ENAME: GEORGE

ENTER VALUE FOR JOB: ANALYST

ENTER VALUE FOR MGR: NULL

ENTER VALUE FOR HIREDATE: 08-SEP-92

ENTER VALUE FOR SAL: 5000

ENTER VALUE FOR COMM: 0

ENTER VALUE FOR DEPTNO: 10

OLD 2: (&EMPNO,'&ENAME','&JOB',&MGR,'&HIREDATE',&SAL,&COMM,&DEPTNO)

NEW 2: (1001,'GEORGE','ANALYST',NULL,'08-SEP-92',5000,0,10)

1 ROW CREATED.

SQL> INSERT INTO EMPLOYEE VALUES

(&EMPNO,'&ENAME','&JOB',&MGR,'&HIREDATE',&SAL,&COMM,&DEPTNO);

ENTER VALUE FOR EMPNO: 1002

ENTER VALUE FOR ENAME: BECKER

ENTER VALUE FOR JOB: MANAGER

ENTER VALUE FOR MGR: 1000

ENTER VALUE FOR HIREDATE: 04-NOV-92

ENTER VALUE FOR SAL: 2800

ENTER VALUE FOR COMM: 4

ENTER VALUE FOR DEPTNO: 20

OLD 2: (&EMPNO,'&ENAME','&JOB',&MGR,'&HIREDATE',&SAL,&COMM,&DEPTNO)

NEW 2: (1002,'BECKER','MANAGER',1000,'04-NOV-92',2800,4,20)

1 ROW CREATED.

SQL> INSERT INTO EMPLOYEE VALUES

(&EMPNO,'&ENAME','&JOB',&MGR,'&HIREDATE',&SAL,&COMM,&DEPTNO);

ENTER VALUE FOR EMPNO: 1003

ENTER VALUE FOR ENAME: BILL

ENTER VALUE FOR JOB: CLERK

ENTER VALUE FOR MGR: 1002

ENTER VALUE FOR HIREDATE: 04-NOV-92

ENTER VALUE FOR SAL: 3000

ENTER VALUE FOR COMM: 0

ENTER VALUE FOR DEPTNO: 20

OLD 2: (&EMPNO,'&ENAME','&JOB',&MGR,'&HIREDATE',&SAL,&COMM,&DEPTNO)

NEW 2: (1003,'BILL','CLERK',1002,'04-NOV-92',3000,0,20)

1 ROW CREATED.

>> OUTPUT:

EMPNO ENAME JOB MGR HIREDATE SAL COMM

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

DEPTNO

----------

7369 SMITH 800

20

7499 ALLEN 1600

30

7521 WARD 1250

30

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**LAB 6: TRANSCATION CONTROL LANGUAGE STATEMENTS**

## 6.1: Transaction Control Language Statements

**1) Insert rows with the following data into the Customer table.**

* **6000, John, #115 Chicago, #115 Chicago, M, 25, 7878776, 10000**
* **6001, Jack, #116 France, #116 France, M, 25, 434524, 20000**
* **6002, James, #114 New York, #114 New York, M, 45, 431525, 15000.50**

**Use parameter substitution.**

->> SQL:

INSERT INTO CUSTOMER VALUES(&CUSTOMERID,'&CUSTOMERNAME','&ADDRESS1','&ADDRE

SS2','&GENDER',&AGE,&PHONENO,&SALARY);

ENTER VALUE FOR CUSTOMERID: 6000

ENTER VALUE FOR CUSTOMERNAME: JHON

ENTER VALUE FOR ADDRESS1: #115 CHICAGO

ENTER VALUE FOR ADDRESS2: #115 CHICAGO

ENTER VALUE FOR GENDER: M

ENTER VALUE FOR AGE: 25

ENTER VALUE FOR PHONENO: 7878776

ENTER VALUE FOR SALARY: 10000

OLD 1: INSERT INTO CUSTOMER VALUES(&CUSTOMERID,'&CUSTOMERNAME','&ADDRESS1','&A

DDRESS2','&GENDER',&AGE,&PHONENO,&SALARY)

NEW 1: INSERT INTO CUSTOMER VALUES(6000,'JHON','#115 CHICAGO','#115 CHICAGO','

M',25,7878776,10000)

1 ROW CREATED.

SQL> /

ENTER VALUE FOR CUSTOMERID: 6001

ENTER VALUE FOR CUSTOMERNAME: JACK

ENTER VALUE FOR ADDRESS1: #116 FRANCE

ENTER VALUE FOR ADDRESS2: #116 FRANCE

ENTER VALUE FOR GENDER: M

ENTER VALUE FOR AGE: 25

ENTER VALUE FOR PHONENO: 434524

ENTER VALUE FOR SALARY: 20000

OLD 1: INSERT INTO CUSTOMER VALUES(&CUSTOMERID,'&CUSTOMERNAME','&ADDRESS1','&A

DDRESS2','&GENDER',&AGE,&PHONENO,&SALARY)

NEW 1: INSERT INTO CUSTOMER VALUES(6001,'JACK','#116 FRANCE','#116 FRANCE','M'

,25,434524,20000)

1 ROW CREATED.

SQL> /

ENTER VALUE FOR CUSTOMERID: 6002

ENTER VALUE FOR CUSTOMERNAME: JAMES

ENTER VALUE FOR ADDRESS1: #114 NEW YORK

ENTER VALUE FOR ADDRESS2: #114 NEW YORK

ENTER VALUE FOR GENDER: M

ENTER VALUE FOR AGE: 45

ENTER VALUE FOR PHONENO: 431525

ENTER VALUE FOR SALARY: 15000.50

OLD 1: INSERT INTO CUSTOMER VALUES(&CUSTOMERID,'&CUSTOMERNAME','&ADDRESS1','&A

DDRESS2','&GENDER',&AGE,&PHONENO,&SALARY)

NEW 1: INSERT INTO CUSTOMER VALUES(6002,'JAMES','#114 NEW YORK','#114 NEW YORK

','M',45,431525,15000.50)

1 ROW CREATED.

**2)Create a Savepoint named ‘SP1’ after third record in the Customer table .**

->> SQL:

SAVEPOINT SP1;

**3)Insert the below row in the Customer table.**

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

->> SQL:

> INSERT INTO CUSTOMER VALUES(&CUSTOMERID,'&CUSTOMERNAME','&ADDRESS1','&ADDRE

SS2','&GENDER',&AGE,&PHONENO,&SALARY);

ENTER VALUE FOR CUSTOMERID: 6003

ENTER VALUE FOR CUSTOMERNAME: JHON

ENTER VALUE FOR ADDRESS1: #114 CHICAGO

ENTER VALUE FOR ADDRESS2: #114 CHICAGO

ENTER VALUE FOR GENDER: M

ENTER VALUE FOR AGE: 45

ENTER VALUE FOR PHONENO: 439525

ENTER VALUE FOR SALARY: 19000.60

OLD 1: INSERT INTO CUSTOMER VALUES(&CUSTOMERID,'&CUSTOMERNAME','&ADDRESS1','&A

DDRESS2','&GENDER',&AGE,&PHONENO,&SALARY)

NEW 1: INSERT INTO CUSTOMER VALUES(6003,'JHON','#114 CHICAGO','#114 CHICAGO','

M',45,439525,19000.60)

**4)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 TO SP1;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*