**ORACLE(SQL)**

**DAY1:**

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

SELECT dept\_name,salary,dept\_code FROM department\_masters where dept\_code in (20,30,40);

**DEPT\_NAME SALARY DEPT\_CODE**

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

Electricals 25000 20

Electronics 30000 30

Mechanics 25000 40

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

DESC department\_masters;

**Name Null? Type**

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

DEPT\_CODE NOT NULL NUMBER(2)

DEPT\_NAME VARCHAR2(50)

SALARY NUMBER

**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,Maths,Chemistry,Physics,(Maths+Chemistry+Physics) as Total\_Marks FROM student\_marks;

STUDENT\_CODE MATHS CHEMISTRY PHYSICS TOTAL\_MARKS

**------------ ---------- ---------- ---------- -----------**

1001 55 45 78 178

1002 66 74 88 228

1003 87 54 65 206

1004 65 64 90 219

1005 78 88 65 231

1006 65 86 54 205

1007 67 79 49 195

1008 72 55 55 182

1009 71 59 58 188

1010 68 44 92 204

1011 89 96 78 263

**STUDENT\_CODE MATHS CHEMISTRY PHYSICS TOTAL\_MARKS**

**------------ ---------- ---------- ---------- -----------**

1012 78 56 55 189

1013 75 58 65 198

1014 73 74 65 212

1015 66 45 74 185

1016 68 78 74 220

1017 69 44 52 165

1018 65 78 56 199

1019 78 58 74 210

1020 45 55 65 165

1021 78 79 78 235

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

select \* from staff\_masters where designation in('Professor','lecturer');

STAFF\_CODE STAFF\_NAME DEPT\_CODE STAFF\_SAL DESIGNATION

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

101 Arvind 102 17000 Professor

102 Surya 103 15000 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 (sysdate-hiredate)>(18\*265);

**STAFF\_CODE STAFF\_NAME DEPT\_CODE**

**------- ------- --------------- ----------------**

100001 Arvind 102

100002 Shyam 102

100005 John 106

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

select staff\_name,designation from staff\_masters where hiredate < to\_date ('01-jan-2003');

**STAFF\_NAME DESIGNATION**

**--------------- --------------------------------------------------**

Arvind Manager

Shyam Lecturer

John Sr.consultant

roja Consultant

reeta Consultant

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

select emp\_name,designation,salary\*120 as income from emp\_table where dept\_no in(20,30);

EMP\_NAME DESIGNATION INCOME

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

sushma Consultant 4800000

chandu HR 3600000

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

SQL> select staff\_name,((sysdate-hiredate)/365) as experience from staff\_masters where designation ='Lecturer';

STAFF\_NAME EXPERIENCE

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

Arvind 16.9793419

Anil 18.8286569

Allen 18.7108487

Smith 17.8259172

Rahul 16.0752323

Ram 17.9738624

6 rows selected.

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

select concat(student\_name,concat(',',dept\_code)) as student\_info from student\_masters;

**STUDENT\_INFO**

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

Amit,10

Ravi,10

Ajay,20

Raj,30

Arvind,40

Rahul,50

Mehul,20

Dev,10

Vijay,30

Rajat,40

Sunder,50

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

SQL> select staff\_name,staff\_sal from staff\_masters where staff\_sal between 12000 and 25000;

STAFF\_NAME STAFF\_SAL

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

Arvind 17000

Mohan 24000

Anil 20000

Raviraj 18000

Rahul 22000

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

SQL> select staff\_name from staff\_masters where mgr\_code is null;

STAFF\_NAME

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

Arvind

John

Raviraj

Ram

**12. 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).**

SQL> select staff\_name,dept\_code,staff\_dob from staff\_masters where staff\_dob between '2-Jan-81' and '31-Mar-83' order by staff\_dob;

STAFF\_NAME DEPT\_CODE STAFF\_DOB

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

Allen 30 18-MAR-81

Smith 20 28-MAR-82

Ram 30 28-APR-82

Anil 20 18-FEB-83

Raviraj 40 28-MAR-83

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

SQL> select dept\_code,sum(staff\_sal) as Sum from staff\_masters where staff\_code not in (select distinct mgr\_code from staff\_masters) and staff\_sal >20000 group by dept\_code;

DEPT\_CODE SUM

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

30 32000

20 84000

**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)**

SQL> select student\_name,NVL(to\_char (dept\_code),'No Department') Department from student\_masters;

STUDENT\_NAME DEPARTMENT

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

Amit 10

Ravi No Department

Ajay 20

Raj 30

Arvind 40

Rahul 50

Mehul 20

Dev 10

Vijay 30

Rajat No Department

Sunder 50

Rajesh 30

Anil 20

Sunil 10

Kapil 40

Ashok 40

Ramesh 30

Amit Raj 50

Ravi Raj 50

Amrit 10

Sumit 20

21 rows selected.

**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**

SQL> select staff\_name,lpad('x',floor(staff\_sal/10000),'x') as salary from staff\_masters;

STAFF\_NAME SALARY

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

Arvind x

Mohan xx

Anil xx

John xxx

Allen xxxx

Smith xxxxxx

Raviraj x

Rahul xx

Ram xxx

9 rows selected.

**Lab 2. Single Row Functions**

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

SQL> select \* from staff\_masters where staff\_name like '%s' and staff\_name like 'a%';

STAFF\_CODE STAFF\_NAME DEPT\_CODE STAFF\_SAL DESIGNATION

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

100001 adams 30 17000 lecturer

100009 andres 20 22000 lecturer

100010 atticus 30 32000 l and t

**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’.**

SQL> select staff\_name,designation from staff\_masters where(staff\_name like '\_n%' or staff\_name like '\_\_n%') and (staff\_name like '%n' or staff\_name like '%s');

STAFF\_NAME DESIGNATION

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

andres lecturer

**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 ‘$’.**

SQL> select staff\_name,lpad(staff\_sal,15,'$') from staff\_masters;

STAFF\_NAME LPAD(STAFF\_SAL,15,'$')

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

adams $$$$$$$$$$17000

Mohan $$$$$$$$$$24000

Anil $$$$$$$$$$20000

John $$$$$$$$$$32000

Allen $$$$$$$$$$42000

Smith $$$$$$$$$$62000

Raviraj $$$$$$$$$$18000

andres $$$$$$$$$$22000

atticus $$$$$$$$$$32000

9 rows selected.

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

|  |  |
| --- | --- |
| **Salary** | **Grade** |
| **Salary >=50000** | **A** |
| **Salary >= 25000 < 50000** | **B** |
| **Salary>=10000 < 25000** | **C** |
| **OTHERS** | **D** |

SQL> select staff\_name,staff\_sal,

2 case

3 when staff\_sal>=50000 then 'A'

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

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

6 else 'D'

7 end as grade

8 from staff\_masters;

STAFF\_NAME STAFF\_SAL GRADE

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

adams 17000 C

Mohan 24000 C

Anil 20000 C

John 32000 B

Allen 42000 B

Smith 62000 A

Raviraj 18000 C

andres 22000 C

atticus 32000 B

9 rows selected.

**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.**

SQL> select min(staff\_sal) as min\_sal,max(staff\_sal) as max\_sal,sum(staff\_sal) as total,round(avg(staff\_sal)) as average\_sal from staff\_masters;

MIN\_SAL MAX\_SAL TOTAL AVERAGE\_SAL

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

17000 62000 269000 29889

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

SQL> select dept\_code,min(staff\_sal) as min\_sal,max(staff\_sal) as max\_sal,sum(staff\_sal) as total,round(avg(staff\_sal)) as average\_sal from staff\_masters where staff\_sal is not null group by dept\_code;

DEPT\_CODE MIN\_SAL MAX\_SAL TOTAL AVERAGE\_SAL

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

30 17000 42000 91000 30333

20 20000 62000 104000 34667

40 18000 18000 18000 18000

10 24000 32000 56000 28000

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

SQL> select dept\_code, count(\*) as no\_of\_people from staff\_masters where dept\_code is not null group by dept\_code;

DEPT\_CODE NO\_OF\_PEOPLE

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

30 3

20 3

40 1

10 2

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

SQL> select count(designation) as total\_no\_of\_managers from staff\_masters where designation='manager' group by designation;

TOTAL\_NO\_OF\_MANAGERS

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

3

**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.**

SQL> select staff\_code,staff\_name,staff\_sal from staff\_masters where staff\_sal>=10000 and designation ='manager' order by staff\_sal desc;

STAFF\_CODE STAFF\_NAME STAFF\_SAL

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

100007 Smith 62000

100006 Allen 42000

100003 Mohan 24000

**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.**

SQL> select s.staff\_name,d.dept\_code,d.dept\_name,s.staff\_sal from staff\_masters s join department\_masters d on d.dept\_code=s.dept\_code where s.staff\_sal>20000;

STAFF\_NAME DEPT\_CODE DEPT\_NAME STAFF\_SAL

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

Mohan 10 Computer Science 24000

John 10 Computer Science 32000

Allen 30 Electronics 42000

Smith 20 Electricals 62000

andres 20 Electricals 22000

atticus 30 Electronics 32000

6 rows selected.

**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**.

SQL> select c.staff\_name,d.dept\_code,d.dept\_name from staff\_masters c join department\_masters d on d.dept\_code=c.dept\_code where c.dept\_code <> 10and staff\_name like '%a%';

STAFF\_NAME DEPT\_CODE DEPT\_NAME

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

andres 20 Electricals

atticus 30 Electronics

adams 30 Electronics

Raviraj 40 Mechanics

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

SQL> select s.staff\_code staff#,s.staff\_name staff,s.dept\_code mgr#,m.mgr\_name manager from staff\_masters s join designation\_masters m on s.dept\_code=m.mgr\_code;

STAFF# STAFF MGR# MANAGER

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

100001 adams 101 HOD

100003 Mohan 102 Professor

100004 Anil 104 Sr.Lecturer

100005 John 105 Lecturer

100006 Allen 106 Director

**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.**

SQL> select s.student\_code,s.student\_name,s.dept\_name,s.subject1,s.subject2,s.subject3 from student\_marks s join department\_masters d on s.dept\_code=d.dept\_code where s.subject1>=60 and s.subject2>=60 and s.subject3>=60 and d.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.**

SQL> select c.student\_code,a.student\_name,c.book\_code,b.book\_name from student\_masters a,book\_masters b,book\_transactions c where a.student\_code=c.student\_code and b.book\_code=c.book\_code and to\_char(c.book\_expected\_return\_date)like to\_char(sysdate);

no rows selected

**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.**

SQL> select c.staff\_code,c.staff\_name,b.dept\_name,b.design\_name,d.book\_code,d.book\_name,e.book\_issue\_date from designation\_masters a,department\_masters b,staff\_masters c,book\_masters d,book\_transactions e where a.design\_code=c.dept\_code and b.dept\_code=c.dept\_code and d.book\_code=e.book\_code and book\_issue\_date>=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).**

SQL> select a.staff\_code,a.staff\_name,b.design\_name,c.dept\_name from staff\_masters a,designation\_masters b,department\_masters c where a.dept\_code=c.dept\_code and a.design\_code=b.design\_code and b.dept\_name not in '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.**

**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).**

SQL> select a.staff\_code,a.staff\_name,b.design\_name,e.dept\_name,c.book\_code ,c.book\_name,c.book\_pub\_author,5\*(d.book\_actual\_return\_date-d.book\_expected\_return\_date) Fine from staff\_masters a,designation\_masters b,book\_masters c,book\_transactions d,department\_masters e where a.design\_code=b.design\_code and c.book\_code=d.book\_code and e.dept\_code=a.dept\_code and a.staff\_code=d.staff\_code;

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

SQL> select staff\_code,staff\_name,staff\_sal from staff\_masters where staff\_sal<(select avg(staff\_sal) from staff\_masters);

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

SQL> select staff\_code,staff\_name from staff\_masters where staff\_code=any(select staff\_code from staff\_masters where designation != 'manager');

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

SQL> 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(1)>1)order by 1,2;

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

SQL> select distinct c.staff\_code ,c.staff\_name,d.dept\_name,m.book\_code from department\_masters d,staff\_masters c,book\_transactions b,book\_masters m where c.dept\_code=d.dept\_code and b.book\_code =m.book\_code and c.staff\_code=b.staff\_code and m.book\_code in(select m.book\_code from department\_masters d,staff\_masters c,book\_transactions b,book\_masters m where c.dept\_code =d.dept\_code and b.book\_code=m.book\_code and c.staff\_code=b.staff\_code group by b.book\_code having count(b.book\_code)>1);

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

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

**Subject3, Total.**

SQL> select (subject1+subject2+subject3) total c.student\_code,c.student\_name,d.dept\_name from student\_masters c,department\_masters db,student\_marks m where c.dept\_code=d.dept\_code and m.student\_code=c.student\_code and rownum<=10 order by 1 desc;

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

SQL> (select a.staff\_name,b.dept\_name,a.staff\_sal from staff\_masters a,department\_masters b,(select avg(a.staff\_sal) avg\_sal,b.dept\_code from staff\_masters a,department\_masters b where a.dept\_code=b.dept\_code group by b.dept\_code) tab where a.dept\_code=b.dept\_code and b.dept\_code=tab.dept\_code and a.staff\_sal<tab.avg\_sal);

**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.**

SQL> select c.staff\_name,d.dept\_name from staff\_masters c,department\_masters d where c.dept\_code=d.dept\_code group by c.staff\_name,d.dept\_name;

**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.**

SQL> select c.student\_code,c.student\_name,d.dept\_name,(m.subject1+m.subject2+m.subject3) total from student\_marks m,department\_masters d,student\_masters c where c.dept\_code=d.dept\_code and c.student\_code= m.student\_code and d.dept\_name='Computer Science' and (m.subject1+m.subject2+m.subject3) in (select max(m.subject1+m.subject2+m.subject3) from student\_marks m,department\_masters d,student\_masters c where c.dept\_code=d.dept\_code and c.student\_code= m.student\_code and d.dept\_name ='Computer Science');

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

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

SQL> select a.staff\_name,count(b.staff\_code) total\_team\_strength from staff\_masters a,staff\_masters b where a.dept\_code=b.staff\_code group by a.staff\_name;

**Lab 4. 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)**

SQL> CREATE TABLE customer\_table(

2 customer\_id NUMBER(5),

3 customer\_name NUMBER(10),

4 address1 VARCHAR2(30),

5 address2 VARCHAR2(30));

Table created.

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

SQL> alter table customer\_table modify customer\_name varchar2(30) not null;

Table altered.

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

**Gender Varchar2(1)**

**Age Number(3)**

**PhoneNo Number(10)**

SQL> ALTER TABLE customer\_table ADD Gender VARCHAR2(1);

Table altered.

SQL> ALTER TABLE customer\_table ADD age NUMBER(3);

Table altered.

SQL> ALTER TABLE customer\_table ADD phoneNo NUMBER(10);

Table altered.

**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**

SQL> INSERT INTO customer\_table(customer\_id,customer\_name,address1,address2,Gender,age,phoneNo) values (&customer\_id,'&customer\_name','&address1','&address2','&gender',&age,&phoneNo);

old 1: INSERT INTO customer\_table(customer\_id,customer\_name,address1,address2,Gender,age,phoneNo) values (&customer\_id,'&customer\_name','&address1','&address2','&gender',&age,&phoneNo)

new 1: INSERT INTO customer\_table(customer\_id,customer\_name,address1,address2,Gender,age,phoneNo) values (1000,'Allen','#115 chicago','#115 Chicago','M',25,7878776)

1 row created.

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

SQL> alter table customer\_table add constraint custid\_prim primary key(customer\_id);

Table altered.

**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**

SQL> insert into customer\_table (customer\_id,customer\_name,address1,address2,gender,age,phoneno) values(1002,'John','#114 chicago','#114 chicago','M',45,439525);

insert into customer\_table (customer\_id,customer\_name,address1,address2,gender,age,phoneno) values(1002,'John','#114 chicago','#114 chicago','M',45,439525)

\*

ERROR at line 1:

ORA-00001: unique constraint (SYSTEM.CUSTID\_PRIM) 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**

SQL> alter table customer\_table disable constraint custid\_prim;

Table altered.

SQL> insert into customer\_table (customer\_id,customer\_name,address1,address2,gender,age,phoneno) values(1002,'Becker','#114 New York','#114 New York','M',45,431525);

1 row created.

SQL> insert into customer\_table (customer\_id,customer\_name,address1,address2,gender,age,phoneno) values(1003,'Nanapatekar','#115 India','#115 India','M',45,431525);

1 row created.

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

SQL> alter table customer\_table add constraint custid\_prim primary key(customer\_id);

alter table customer\_table add constraint custid\_prim primary key(customer\_id)

\*

ERROR at line 1:

ORA-02260: table can have only one primary key

**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**

SQL> alter table customer\_table drop constraint custid\_prim;

Table altered.

SQL> insert into customer\_table (customer\_id,customer\_name,address1,address2,gender,age,phoneno) values(1002,'Becker','#114 New York','#114 New York','M',45,431525,15000.50);

insert into customer\_table (customer\_id,customer\_name,address1,address2,gender,age,phoneno) values(1002,'Becker','#114 New York','#114 New York','M',45,431525,15000.50)

\*

ERROR at line 1:

ORA-00913: too many values

SQL> insert into customer\_table (customer\_id,customer\_name,address1,address2,gender,age,phoneno) values(1003,'Nanapatekar','#115 India','#115 India','M',45,431525,20000.50);

insert into customer\_table (customer\_id,customer\_name,address1,address2,gender,age,phoneno) 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.**

SQL> truncate table customer\_table;

Table truncated.

SQL> select \* from customer\_table;

no rows selected

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

SQL> alter table customer\_table add E\_mail VARCHAR2(20);

Table altered.

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

SQL> ALTER TABLE customer\_table drop column E\_mail;

Table altered.

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

SQL> alter table customer\_table add EmailId VARCHAR2(20);

Table altered.

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

SQL> alter table customer\_table set unused column EmailId;

Table altered.

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

SQL> alter table customer\_table drop unused columns;

Table altered.

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

SQL> COMMENT ON table system.customer\_table IS'COMMENTS';

Comment created.

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

**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.**

SQL> CREATE table suppliers as (select customer\_id as suppid,customer\_name as sname,address1 as addr1,address2 as addr2,phoneno as contactno from customer\_table);

Table created.

**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)**

SQL> drop table suppliers;

Table dropped.

SQL> desc suppliers;

ERROR:

ORA-04043: object suppliers does not exist

SQL> create table customer\_masters( CustomerId Number(5) constraint custid\_pk primary key,customer\_name varchar2(30) not null,address1 varchar2(30) not null,address2 varchar2(30),gender varchar2(1),age number(3),phoneno number(10));

Table created.

SQL> desc customer\_masters;

Name Null? Type

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

CUSTOMERID NOT NULL NUMBER(5)

CUSTOMER\_NAME NOT NULL VARCHAR2(30)

ADDRESS1 NOT NULL VARCHAR2(30)

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**

SQL> create table AccountsMaster(customer\_id number(5),account\_number number(10,2) constraint acc\_pk primary key, accoutn\_type char(3),ledger\_balance number(10) not null);

Table created.

SQL> desc AccountsMaster;

Name Null? Type

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

CUSTOMER\_ID NUMBER(5)

ACCOUNT\_NUMBER NOT NULL NUMBER(10,2)

ACCOUTN\_TYPE CHAR(3)

LEDGER\_BALANCE NOT NULL NUMBER(10)

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

SQL> alter table AccountsMaster add constraint cust\_acc FOREIGN KEY (customer\_id) REFERENCES customer\_masters(customerid);

Table altered.

**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**

old 1: insert into customer\_masters(customerid,customer\_name,address1,address2,gender,age,phoneno) values(&customerid,'&customer\_name','&address1','&address2','&gender',&age,&phoneno)

new 1: insert into customer\_masters(customerid,customer\_name,address1,address2,gender,age,phoneno) values(1000,'Allen','#115 Chicago','#115 Chicago','M',25,7878776)

1 row created.

old 1: insert into customer\_masters(customerid,customer\_name,address1,address2,gender,age,phoneno) values(&customerid,'&customer\_name','&address1','&address2','&gender',&age,&phoneno)

new 1: insert into customer\_masters(customerid,customer\_name,address1,address2,gender,age,phoneno) values(1001,'George','#116 France','#116 France','M',25,434524)

1 row created.

old 1: insert into customer\_masters(customerid,customer\_name,address1,address2,gender,age,phoneno) values(&customerid,'&customer\_name','&address1','&address2','&gender',&age,&phoneno)

new 1: insert into customer\_masters(customerid,customer\_name,address1,address2,gender,age,phoneno) values(1002,'Becker','#114 New York','#114 New York','M',45,431525)

1 row created.

SQL> select \* from customer\_masters;

CUSTOMERID CUSTOMER\_N ADDRESS1 ADDRESS2 G AGE PHONENO

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

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

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

SQL> alter table AccountsMaster add constraint ck\_ac check (account\_type='NRI' or account\_type='IND');

Table altered.

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

SQL> insert into accountsmaster values(1000,46798738,'NRI',32000000);

1 row created.

SQL> insert into accountsmaster values(1001,46798742,'IND',32000250);

1 row created.

SQL> insert into accountsmaster values(1002,49023493,'IND',23900000);

1 row created.

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

SQL> alter table accountsmaster add constraint balance\_check check(ledger\_balance>5000);

Table altered.

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

SQL> create table AccountDetails as (select \* from AccountsMaster);

Table created.

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

SQL> alter table AccountDetails rename to Backup\_Details;

Table altered.

**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.**

SQL> create view acc\_view as (select c.customerid as Customer\_Code,c.Customer\_Name as Account\_Holder\_Name,a.Account\_Number,a.account\_type as Type,a.ledger\_balance as Balance from customer\_masters c,accountsmaster a where c.customerid=a.customer\_id);

View created.

SQL>desc acc\_view;

Name Null? Type

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

CUSTOMER\_CODE NOT NULL NUMBER(5)

ACCOUNT\_HOLDER\_NAME NOT NULL VARCHAR2(30)

ACCOUNT\_NUMBER NOT NULL NUMBER(10,2)

TYPE CHAR(3)

BALANCE NOT NULL NUMBER(10)

**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**

SQL> create view vAccs\_Dtls as(select c.\* from customer\_masters c,Accountsmaster a where c.customerid = a.customer\_id and a.account\_type='IND' and a.ledger\_balance >10000) with check option constraint vAccs\_Dtls\_cnst;

View created.

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

SQL> create view accsvw10 as(select c.\* from customer\_masters c,accountsmaster a where c.customerid=a.customer\_id and a.account\_type='IND' and a.ledger\_balance >10000)with read only;

View created.

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

SQL> select dept\_code from staff\_masters where staff\_sal in(select max(staff\_sal) from staff\_masters);

DEPT\_CODE

----------

40

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

**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.**

SQL> create sequence deptno\_sequen start with 40 increment by 10 maxvalue 200 cycle;

Sequence created.

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

SQL> insert into department\_masters values(deptno\_sequen.nextval,'ABC');

SQL> insert into department\_masters values(deptno\_sequen.nextval,'DEF');

SQL> insert into department\_masters values(deptno\_sequen.nextval,'XYZ');

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

SQL> alter sequence deptno\_sequen increment by 5;

Sequence altered.

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

SQL> drop sequence deptno\_sequen ;

Sequence dropped.

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

SQL> create unique index no\_name on department\_masters (dept\_code,dept\_name);

Index created.

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

SQL> select \* from all\_indexes where index\_name='No\_Name';

no rows selected

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

SQL> create synonym synemp for emp;

Synonym created.

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

SQL> select \* from all\_synonyms where synonym\_name='synemp';

no rows selected

**Lab 5. 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> CREATE TABLE employ\_table(  2 EMPNO NUMBER(4) not null,  3 ENAME VARCHAR2(10),  4 JOB VARCHAR2(50) ,  5 MGR NUMBER(4),  6 HIREDATE DATE,  7 SAL NUMBER(7,2),  8 COMM NUMBER(7,2),  9 DEPTNO NUMBER(2));  Table created. | |  | |

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

**SQL>select \* from employee**

SQL> insert into employ\_table(empno,ename,sal,deptno)values(7369,'smith',800,20);

1 row created.

SQL> insert into employ\_table(empno,ename,sal,deptno)values(7469,'allen',1600,30);

1 row created.

SQL> insert into employ\_table(empno,ename,sal,deptno)values(7521,'ward',1250,30);

1 row created.

SQL> insert into employ\_table(empno,ename,sal,deptno)values(7566,'jones',2975,20);

1 row created.

SQL> insert into employ\_table(empno,ename,sal,deptno)values(7654,’martin’,1250,30);

1 row created.

SQL> insert into employ\_table(empno,ename,sal,deptno)values(7698,’blake’,2850,30);

1. row 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 employ\_table set job=(select job from employ\_table where empno=7788),deptno=(select deptno from employ\_table where empno=7788) where empno=7698;

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

SQL> delete from department\_masters where dept\_name='sales';

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

SQL> update employ\_table set deptno=(select deptno from employee where empno=7788) where empno=7698;

**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**

SQL> insert into employ\_table 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 1: insert into employ\_table values(&empno,'&ename',&job,&mgr,'&hiredate',&sal,&comm,&deptno)

new 1: insert into employ\_table values(1000,'allen','clerk',1001,'12-jan-01',3000,2,10)

1 row created.

SQL> insert into employ\_table 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 1: insert into employ\_table values(&empno,'&ename',&job,&mgr,'&hiredate',&sal,&comm,&deptno)

new 1: insert into employ\_table values(1001,'george','analyst',null,'08-sep-92',5000,0,10)

1 row created.

**Lab 6. 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.**

Insert into customer\_table values(6000,’john’,#115 chicago,#115 chicago,’M’,25,7878776,10000);

1 row created.

Insert into customer\_table values(6001,’jack’,#116 france,#116 france,’M’,25,434524,20000);

1 row created.

Insert into customer\_table values(6002,’james’,#114 new york,#114 new york,’M’,45,431525,15000.50);

1 row created.

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

SQL> COMMIT;

Commit complete.

SQL> savepoint sp1;

Savepoint created.

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

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

Insert into customer\_table values(6003,’john’,#114 chicago,#114 chicago,’M’,45,439525,19000.60);

1 row created.

1. **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;

Rollback complete.

SQL> select \* from customer\_table;

**PL/SQL**

**2.1**

**Identify the problems(if any) in the below declarations:**

**DECLARE**

**V\_Sample1 NUMBER(2);**

**V\_Sample2 CONSTANT NUMBER(2) ;**

**V\_Sample3 NUMBER(2) NOT NULL ;**

**V\_Sample4 NUMBER(2) := 50;**

**V\_Sample5 NUMBER(2) DEFAULT 25;**

**2.3 Write a PL/SQL block to retrieve all staff (code, name, salary) under specific department number and display the result. (Note: The Department\_Code will be accepted from user. Cursor to be used.)**

create or replace procedure bi\_pro(input in number)

2 is

3 s\_code staff\_masters.dept\_code%type;

4 s\_name staff\_masters.staff\_name%type;

5 s\_salary staff\_masters.staff\_sal%type;

6 cursor bi\_cur is select dept\_code,staff\_name,staff\_sal from staff\_masters where dept\_code=input;

7 begin

8 open bi\_cur;

9 loop

10 fetch bi\_cur into s\_code,s\_name,s\_salary;

11

12 exit when bi\_cur%notfound;

13 dbms\_output.put\_line(s\_code || s\_name ||s\_salary);

14

15 end loop;

16 close bi\_cur;

17 end;

18 /

Procedure created.

SQL> call bi\_pro(20);

20Mohan24000

20andres22000

Call completed.

**2.4. Write a PL/SQL block to increase the salary by 30 % or 5000 whichever minimum for a given Department\_Code.**

SQL> create or replace procedure bi\_pro2(input in number)

2 is

3 s\_code staff\_masters.dept\_code%type;

4 s\_name staff\_masters.staff\_name%type;

5 s\_salary staff\_masters.staff\_sal%type;

6 cursor bi\_cur1 is select dept\_code,staff\_name,staff\_sal+5000 from staff\_masters where dept\_code=input;

7 begin

8 open bi\_cur1;

9 loop

10 fetch bi\_cur1 into s\_code,s\_name,s\_salary;

11 exit when bi\_cur1%notfound;

12 dbms\_output.put\_line(s\_code || s\_name ||s\_salary);

13

14 end loop;

15

16 close bi\_cur1;

17 end;

18 /

Procedure created.

SQL> call bi\_pro2(20);

20Mohan29000

20andres27000

Call completed.

**2.5. Write a PL/SQL block to generate the following report for a given Department code**

**Student\_Code Sudent\_Name Subject1 Subject2 Subject3 Total Percentage Grade**

**Note: Display suitable error massage if wrong department code has entered and if there is no student in the given department.**

**For Grade:**

**Student should pass in each subject individually (pass marks 60).**

**Percent >= 80 then grade= A**

**Percent >= 70 and < 80 then grade= B**

**Percent >= 60 and < 70 then grade= C**

**Else D**

declare

2 cursor c is select student\_name,subject1,subject2,subject3 from student\_marks where student\_code ='&student\_code';

3 mtotal number;

4 mstudent\_name varchar2(10);

5 msubject1 number;

6 msubject2 number;

7 msubject3 number;

8 mper number;

9 grade varchar2(2);

10 begin

11 open c ;

12 fetch c into mstudent\_name,msubject1,msubject2,msubject3;

13 mtotal:= (msubject1+msubject2+msubject3);

14 mper:=((mtotal)/300)\*100;

15 if mper >= 80 then

16 dbms\_output.put\_line(mstudent\_name||' '||msubject1||' ' ||msubject2||' ' ||msubject3||' '||mtotal||' '||mper||' '||'the grade is' || ' '|| 'A');

17 elsif mper < 80 and mper >= 70 then

18 dbms\_output.put\_line(mstudent\_name||' '||msubject1||' ' ||msubject2||' ' ||msubject3||' '||mtotal||' '||mper||' '||'the grade is' ||''|| 'B');

19 elsif mper < 70 and mper >= 60 then

20 dbms\_output.put\_line(mstudent\_name||' '||msubject1||' ' ||msubject2||' ' ||msubject3||' '||mtotal||' '||mper||' '||'the grade is' ||''|| 'C');

21 else

22 dbms\_output.put\_line(mstudent\_name||' '||msubject1||' ' ||msubject2||' ' ||msubject3||' '||mtotal||' '||mper||' '||'the grade is'||''|| 'D');

23 --exception

24 --when invalid\_number then

25 --dbms\_output.put\_line('enter proper student id');

26 end if;

27 --dbms\_output.put\_line(msubject1);

28 close c;

29 end;

30 /

Enter value for student\_code: 1019

old 2: cursor c is select student\_name,subject1,subject2,subject3 from student\_marks where student\_code ='&student\_code';

new 2: cursor c is select student\_name,subject1,subject2,subject3 from student\_marks where student\_code ='1019';

78 58 74 210 70 the grade isB

PL/SQL procedure successfully completed.

**3.4**

**Predict the output of the following block ? What corrections would be needed to make it more efficient?**

BEGIN

2 DECLARE

3 fname employ\_table.ename%TYPE;

4 BEGIN

5 SELECT ename INTO fname

6 FROM employ\_table

7 WHERE 1=2;

8 DBMS\_OUTPUT.PUT\_LINE('This statement will print');

9 EXCEPTION

10 WHEN OTHERS THEN

11 DBMS\_OUTPUT.PUT\_LINE('Some inner block error');

12 END;

13 EXCEPTION

14 WHEN NO\_DATA\_FOUND THEN

15 DBMS\_OUTPUT.PUT\_LINE('No data found in fname');

16 WHEN OTHERS THEN

17 DBMS\_OUTPUT.PUT\_LINE('Some outer block error');

18 END;

19 /

Some inner block error

PL/SQL procedure successfully completed.

**3.6: Write a PL/SQL program to check for the commission for an employee no 7369. If no commission exists, then display the error message. Use Exceptions.**

declare

2 name employ\_table.ename%type;

3 salary employ\_table.sal%type;

4 commision employ\_table.comm%type;

5 my\_exp exception;

6 begin

7 select ename,sal,nvl(comm,0) into name,salary,commision from employ\_table where empno=1002;

8 if commision = 0 then

9 raise my\_exp;

10 else

11 dbms\_output.put\_line(name || ',' ||salary || ',' || commision);

12 end if;

13

14 exception

15 when my\_exp then

16 dbms\_output.put\_line('commision is not found');

17 end;

18 /

commision is not found

PL/SQL procedure successfully completed.

**3.7: Write a PL/SQL block to drop any user defined table.**

SQL> create or replace procedure bi\_pro2(input in varchar2,input2 in varchar2)

2 is

3 begin

4 execute immediate 'drop'||input||' '||input2;

5 dbms\_output.put\_line('the row deleted');

6 end;

7 /

Procedure created.

**4.1 Write a PL/SQL block to find the maximum salary of the staff in the given department. Note: Department code should be passed as parameter to the cursor.**

SQL>create or replace procedure bi\_pro2(input in number)

2 is

3 name employ\_table.ename%type;

4 salary employ\_table.sal%type;

5 cursor cur1 is select ename,max(sal) from employ\_table where deptno=input;

6 begin

7 open cur1;

8 loop

9 fetch cur1 into name,salary;

10 exit when cur1%notfound;

11 dbms\_output.put\_line(name||' '||salary);

12 end loop;

13 close cur1;

14 end;

15 /

Procedure created.

**4.2. Write a function to compute age. The function should accept a date and return age in years.**

SQL> declare

2 sname varchar2(20) :='&name';

3 sdob date :='&dob';

4 sage number;

5 begin

6 select floor(months\_between(sysdate,sdob)/12) into sage from dual;

7 dbms\_output.put\_line(sname ||' '|| sage);

8 end;

9 /

Enter value for name: satya

old 2: sname varchar2(20) :='&name';

new 2: sname varchar2(20) :='satya';

Enter value for dob: 01-apr-98

old 3: sdob date :='&dob';

new 3: sdob date :='01-apr-98';

satya 21

PL/SQL procedure successfully completed.

4.3. Write a procedure that accept staff code and update staff name to Upper case. If the staff name is null raise a user defined exception.

create or replace procedure bi\_pro(input in number)

2 is

3 begin

4 update employ\_table set ename=lower(ename) where empno=input;

5 if sql%notfound then

6 dbms\_output.put\_line('ename not found');

7 else

8 dbms\_output.put\_line('inserted');

9 end if;

10 end;

11 /

Procedure created.