INTEGRITY CONSTRAINTS

Aim:

To execute integrity constraints in SQL.

*CREATE TABLE employees(empid number, ssn number, PRIMARY KEY (empid, ssn));

```
SQL> create table employees(
  2  empid number,
  3  ssn number,
  4  primary key(empid,ssn));

Table created.

SQL> desc employees;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 EMPID                                     NOT NULL NUMBER
 SSN                                       NOT NULL NUMBER
```

ALTER TABLE sample ADD PRIMARY KEY(sid);

```
SQL> create table sample(
  2  sid number,
  3  sname varchar2(20));

Table created.

SQL> alter table sample add constraint pk_sid primary key(sid);

Table altered.

SQL> desc sample;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 SID                                       NOT NULL NUMBER
 SNAME                                              VARCHAR2(20)
```

*ALTER TABLE employees DROP PRIMARY KEY;

```
SQL> alter table employees drop primary key;

Table altered.
```

```
SQL> alter table sample drop constraint pk_sid;

Table altered.

SQL> desc sample;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 SID                                                NUMBER
 SNAME                                              VARCHAR2(20)
```

CREATE TABLE items (item_id number, qty_hand number, qty_sold number, CONSTRAINT ck_qtyCHECK(qty_hand>qty_sold));

```
SQL> create table items(
  2  item_id number,
  3  qty_hand number,
  4  qty_sold number,
  5  constraint ck_qty CHECK(qty_hand>qty_sold));

Table created.
```

CREATE TABLE students(sid number, grade number CONSTRAINT ck_grade CHECK (grade between 1 AND 10));

```
SQL> create table students(
  2  sid number,
  3  grade number CONSTRAINT ck_grade CHECK(grade between 1 AND 10));

Table created.
```

ALTER TABLE students DROP CONSTRAINT ck_grade;

```
SQL> alter table students drop constraint ck_grade;

Table altered.
```

CREATE TABLE emp_demo(empid number, ssn number, UNIQUE(empid,ssn));

```
SQL> create table emp_demo(
  2  empid number,
  3  ssn number,
  4  UNIQUE(empid,ssn));

Table created.
```

CREATE TABLE students(sid number, email varchar2(20) UNIQUE);

```
SQL> create table students(
  2  sid number,
  3  email varchar2(20) UNIQUE);

Table created.

SQL> desc students;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 SID                                                NUMBER
 EMAIL                                              VARCHAR2(20)
```

CREATE TABLE students(sid number, contact number);

ALTER TABLE students ADD CONSTRAINT un_sid_conUNIQUE(sid,contact);

```
SQL> CREATE TABLE students(sid number, contact number);

Table created.

SQL> ALTER TABLE students ADD CONSTRAINT un_sid_con UNIQUE(sid,contact);

Table altered.

SQL> desc students;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 SID                                                NUMBER
 CONTACT                                            NUMBER
```

ALTER TABLE students DROP CONSTRAINT un_sid_con;

```
SQL> alter table students drop constraint un_sid_con;

Table altered.
```

CREATE TABLE sample(id number, name varchar2(20) NOT NULL);

```
SQL> create table sample(
  2  id number,
  3  name varchar2(20) not null);

Table created.
```

CREATE TABLE person(pid number PRIMARY KEY, pname varchar2(20));

```
SQL> create table person(
  2  pid number primary key,
  3  panme varchar2(20));

Table created.
```

ALTER TABLE person MODIFY (pname varchar2(20) CONSTRAINT nn_pn NOT NULL);

```
SQL> alter table person modify(
  2  panme varchar2(20) constraint nn_pn not null);

Table altered.
```

```
SQL> alter table person modify(panme varchar(20) null);

Table altered.
```

```
SQL> create table items(
  2  item_id number  CONSTRAINT pk_item PRIMARY KEY CONSTRAINT chk_item check(item_id>0),  qty_taken number,
  3  qty_at_hand number, CHECK(qty_taken<qty_at_hand));

Table created.
```

```
SQL> alter table sample add constraint pk_id primary key(id) add constraint un_name unique(name);

Table altered.
```

```
SQL> ALTER TABLE items DROP CONSTRAINT chk_item DROP CONSTRAINT pk_item ;

Table altered.
```

Exercise

1. Alter the table sailors and enforce the following constraints on its columns:
   sid: primary key
   rating: should accept only values between 1 and 10

```
SQL> alter table sailors add constraint pkey1 primary key(sid);

Table altered.

SQL> desc sailors;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 SID                                       NOT NULL NUMBER(5)
 SNAME                                              VARCHAR2(20)
 RATING                                             NUMBER(2)
 AGE                                                NUMBER(3,1)

SQL> alter table sailors add constraint check1 check(rating between 1 and 10);

Table altered.
```

2. Alter the table boats and enforce the following constraints on its columns:
   bid: primary key

```
SQL> alter table boats add constraint pkey2 primary key (bid);

Table altered.

SQL> desc boats;
 Name                                       Null?    Type
 ----------------------------------------- -------- ----------------------------
 BID                                       NOT NULL NUMBER(5)
 BNAME                                              VARCHAR2(20)
 COLOR                                              VARCHAR2(10)
```

3. Alter the table reserves and enforce the following constraints on its columns:
   Sid: Foreign Key
   bid: Foreign key
   (sid,bid): Primary Key
   day : Not null

```
SQL> alter table reserves add constraint fkey1 foreign key(sid) references sailors;

Table altered.

SQL> desc reserves;
 Name                                       Null?    Type
 ----------------------------------------- -------- ----------------------------
 SID                                                NUMBER(5)
 BID                                                NUMBER(5)
 DAY                                                DATE
```

```
SQL> alter table reserves add constraint fkey2 foreign key(bid) references boats;

Table altered.

SQL> desc reserves;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 SID                                                NUMBER(5)
 BID                                                NUMBER(5)
 DAY                                                DATE

SQL> alter table reserves add constraint pkey3 primary key(sid, bid);

Table altered.

SQL> desc reserves;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 SID                                       NOT NULL NUMBER(5)
 BID                                       NOT NULL NUMBER(5)
 DAY                                                DATE

SQL> alter table reserves modify(day not null);

Table altered.

SQL> desc reserves;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 SID                                       NOT NULL NUMBER(5)
 BID                                       NOT NULL NUMBER(5)
 DAY                                       NOT NULL DATE
```

Result:

The execution of integrity constraints in SQL has been done successfully.

Viva Questions

1. What is a savepoint
Ans) A SAVEPOINT is a point in a transaction when you can roll the transaction back to a certain point without rolling back the entire transaction.

2. What is integrity constraint?
Ans) Integrity constraints are a set of rules. It is used to maintain the quality of information. Integrity constraints ensure that the data insertion, updating, and other processes have to be performed in such a way that data integrity is not affected.

3. Differentiate grant and revoke?
Ans) The GRANT command is used for conferring the authorization to the users whereas REVOKE command is used for withdrawing the authorization.

4. Write the syntax of enforcing a primary key constraint on a table?
Ans) CREATE TABLE <table-name>
(column-1 datatype,…,column-n datatype ,CONSTRAINT constraintName primary key(col1));
or CREATE TABLE <table-name> (column-1 datatype primary key,…,column-n datatype);

5. Differentiate drop and truncate?
Ans) The DROP command is used to remove table definition and its contents. Whereas the TRUNCATE command is used to delete all the rows from the table. ... DROP is a DDL(Data Definition Language) command. Whereas the TRUNCATE is also a DDL(Data Definition Language) command.

6. Differentiate check and unique constraint?
Ans) Unique constraints are for all rows, and there is no way to specify only some rows. Check constraints are only for validation within a single row.

7. What do you mean by table and field in SQL?
Ans) In sql, tables are database objects that contain all the data in a database. Tables contain rows and columns, where the rows are known as records and the columns are known as fields.

8. What is the difference between CHAR and VARCHAR2 datatype in SQL?
Ans) CHAR is used to store alphanumeric data of fixed length whereas VARCHAR2 is used to store alphanumeric data of variable length. Maximum limit of char is 2000bytes and that of VARCHAR2 is 4000bytes.

9. What are Entities and Relationships?
Ans) An entity is a real-world object that has some attributes or properties. A relationship is an association between two or more entities.

10. What are the different subsets of SQL?
Ans) DDL, DCL, DML, TCL.

Week4: In-built SQL functions

Aim:

To execute in-built functions of SQL.

Examples

round()-

```
SQL> select round(15.193, 1) from dual;

ROUND(15.193,1)
---------------
           15.2

SQL> select round(15.193, 3) from dual;

ROUND(15.193,3)
---------------
         15.193

SQL> select round(126.45, -1) from dual;

ROUND(126.45,-1)
----------------
             130

SQL> select round(15.193) from dual;

ROUND(15.193)
-------------
           15
```

trunc()-

```
SQL> select trunc(15.193, 1) from dual;

TRUNC(15.193,1)
---------------
           15.1

SQL> select trunc(15.193, 3) from dual;

TRUNC(15.193,3)
---------------
         15.193

SQL> select trunc(126.45, -1) from dual;

TRUNC(126.45,-1)
----------------
             120

SQL> select trunc(123.45, -2) from dual;

TRUNC(123.45,-2)
----------------
             100
```

power(), mod(), sqrt(), abs() -

```
SQL> select power(2, 4) from dual;

POWER(2,4)
----------
        16

SQL> select mod(5, 2) from dual;

  MOD(5,2)
----------
         1

SQL> select sqrt(25) from dual;

  SQRT(25)
----------
         5

SQL> select abs(-5) from dual;

   ABS(-5)
----------
         5
```

sign(), ceil(), floor()-

```
SQL> select sign(15) from dual;

  SIGN(15)
----------
         1

SQL> select sign(-15) from dual;

 SIGN(-15)
----------
        -1

SQL> select sign(0) from dual;

   SIGN(0)
----------
         0

SQL> select ceil(14.6) from dual;

CEIL(14.6)
----------
        15

SQL> select floor(14.6) from dual;

FLOOR(14.6)
-----------
         14
```

lower(), upper() -

```
SQL> select lower(sname) from sailors where sid = 22;

LOWER(SNAME)
--------------------
dustin

SQL> select lower('ORACLE') from dual;

LOWER(
------
oracle

SQL> select upper(sname) from sailors where rating = 7;

UPPER(SNAME)
--------------------
DUSTIN
HORATIO

SQL> select lower('oracle') from dual;

LOWER(
------
oracle

SQL> select upper('oracle') from dual;

UPPER(
------
ORACLE
```

initcap(), length()

```
SQL> select initcap(color) from boats;

INITCAP(CO
----------
Blue
Green
Blue

SQL> select initcap('read') from dual;

INIT
----
Read

SQL> select length('oracle') from dual;

LENGTH('ORACLE')
----------------
               6

SQL> select sname, length(sname) from sailors;

SNAME                LENGTH(SNAME)
-------------------- -------------
Dustin                           6
Brutus                           6
Lubber                           6
Horatio                          7
Art                              3
Bob                              3

6 rows selected.
```

concat()-

```
SQL> select concat(sid, sname) from sailors;

CONCAT(SID,SNAME)
------------------------------------------------------------
22Dustin
29Brutus
31Lubber
64Horatio
85Art
95Bob

6 rows selected.

SQL> select concat('oracle', 'corp') from dual;

CONCAT('OR
----------
oraclecorp
```

|| operation

```
SQL> select sid || sname as details from sailors;

DETAILS
------------------------------------------------------------
22Dustin
29Brutus
31Lubber
64Horatio
85Art
95Bob

6 rows selected.

SQL> select 'oracle'||'corp' from dual;

'ORACLE'||
----------
oraclecorp

SQL> select sname || 'has the raiting' || rating from sailors;

SNAME||'HASTHERAITING'||RATING
------------------------------------------------------------
Dustinhas the raiting7
Brutushas the raiting1
Lubberhas the raiting8
Horatiohas the raiting7
Arthas the raiting3
Bobhas the raiting3

6 rows selected.
```

lpad(), rpad()

```
SQL> select lpad(sname, 10, '*') from sailors;

LPAD(SNAME,10,'*')
----------------------------------------
****Dustin
****Brutus
****Lubber
***Horatio
*******Art
*******Bob

6 rows selected.

SQL> select rpad(sname, 10, '*') from sailors;

RPAD(SNAME,10,'*')
----------------------------------------
Dustin****
Brutus****
Lubber****
Horatio***
Art*******
Bob*******

6 rows selected.
```

ltrim(), rtrim(), trim() ascii(), chr()

```
SQL> select ltrim('  oracle  ') from dual;

LTRIM('O
--------
oracle

SQL> select rtrim('  oracle  ') from dual;

RTRIM('O
--------
  oracle

SQL> select trim('S' from 'MTHSS') from dual;

TRI
---
MTH

SQL> select trim('S' from 'SMITHS') from dual;

TRIM
----
MITH

SQL> select sname, ascii(sname) from sailors;

SNAME                ASCII(SNAME)
-------------------- ------------
Dustin                         68
Brutus                         66
Lubber                         76
Horatio                        72
Art                            65
Bob                            66

6 rows selected.

SQL> select chr(65) from dual;

C
-
A
```

substr()

```
SQL> select substr('computer', 4, 3) from dual;

SUB
---
put

SQL> select substr('computer', 4) from dual;

SUBST
-----
puter

SQL> select substr('computer', -3) from dual;

SUB
---
ter
```

instr()

```
SQL> select sname, instr(sname, 'b') from sailors
  2  ;

SNAME                INSTR(SNAME,'B')
-------------------- ----------------
Dustin                              0
Brutus                              0
Lubber                              3
Horatio                             0
Art                                 0
Bob                                 3

6 rows selected.

SQL> select sname, instr(sname, 'b', 4) from sailors;

SNAME                INSTR(SNAME,'B',4)
-------------------- ------------------
Dustin                                0
Brutus                                0
Lubber                                4
Horatio                               0
Art                                   0
Bob                                   0

6 rows selected.

SQL> select sname, instr(sname, 'b', 3) from sailors;

SNAME                INSTR(SNAME,'B',3)
-------------------- ------------------
Dustin                                0
Brutus                                0
Lubber                                3
Horatio                               0
Art                                   0
Bob                                   3

6 rows selected.
```

translate()

```
SQL> select translate(sname, 'a', '*') from sailors;

TRANSLATE(SNAME,'A','*')
------------------------------------------------------------------
Dustin
Brutus
Lubber
Hor*tio
Art
Bob

6 rows selected.

SQL> select translate('corporation', 'aro', '123') from dual;

TRANSLATE('
-----------
c32p321ti3n

SQL> select translate('corporation', 'aro', '12') from dual;

TRANSLAT
--------
c2p21tin

SQL> select translate('corporation', 'aro', null) from dual;

T
-


SQL> select translate('corporation', 'aro', '') from dual;

T
-


SQL> select translate('corporation', 'aro', '1234') from dual;

TRANSLATE('
-----------
c32p321ti3n
```

replace()

```
SQL> select replace('corporation', 'aro', '123') from dual;

REPLACE('CO
-----------
corporation

SQL> select replace('corporation', 'ora', '123') from dual;

REPLACE('CO
-----------
corp123tion

SQL> select replace('corporation', 'ora', '13') from dual;

REPLACE('C
----------
corp13tion

SQL> select replace('corporation', 'ora') from dual;

REPLACE(
--------
corption
```

sysdate, last_day()

```
SQL> select sysdate from dual;

SYSDATE
---------
23-APR-21

SQL> select last_day(sysdate) from dual;

LAST_DAY(
---------
30-APR-21

SQL> select last_day(sysdate - 15) from dual;

LAST_DAY(
---------
30-APR-21
```

next_day()

```
SQL> select next_day(sysdate, 'sunday') from dual;

NEXT_DAY(
---------
25-APR-21

SQL> select next_day(sysdate, 'Sunday') from dual;

NEXT_DAY(
---------
25-APR-21

SQL> select next_day(sysdate, 'SuNday') from dual;

NEXT_DAY(
---------
25-APR-21

SQL> select next_day(sysdate, 'SuN') from dual;

NEXT_DAY(
---------
25-APR-21

SQL> select next_day(sysdate, 1) from dual;

NEXT_DAY(
---------
25-APR-21
```

add_months(), add_months()

```
SQL> select add_months(sysdate, 5) from dual;

ADD_MONTH
---------
23-SEP-21

SQL> select months_between(sysdate, '01-dec-12') from dual;

MONTHS_BETWEEN(SYSDATE,'01-DEC-12')
-----------------------------------
                          100.740202

SQL> select months_between(sysdate, '01-dec-2012') from dual;

MONTHS_BETWEEN(SYSDATE,'01-DEC-2012')
-------------------------------------
                            100.740208
```

round and trunc

```
SQL> select round(sysdate, 'month') from dual;

ROUND(SYS
---------
01-MAY-21

SQL> select round(sysdate + 10, 'month') from dual;

ROUND(SYS
---------
01-MAY-21

SQL> select trunc(sysdate, 'month') from dual;

TRUNC(SYS
---------
01-APR-21

SQL> select trunc(sysdate + 10, 'month') from dual;

TRUNC(SYS
---------
01-MAY-21
```

rowid()-

```
SQL> select rowid,s.* from sailors s;

ROWID                      SID SNAME                    RATING        AGE
------------------ ---------- -------------------- ---------- ----------
AAAWvWAAEAAACecAAD          29 Brustus                       1         33
AAAWvWAAEAAACecAAE          22 Lubber                        8       55.5
AAAWvWAAEAAACecAAF          64 Horatio                       7         35
AAAWvWAAEAAACecAAG          32 Andy                          8       25.5
AAAWvWAAEAAACecAAI          85 Art                           3       25.5
AAAWvWAAEAAACecAAJ          95 Bob                           3       63.5
AAAWvWAAEAAACecAAK         100 RAM                           8         20

7 rows selected.
```

rownum()-

```
SQL> select rownum,sname,rating from sailors where rownum<5;

    ROWNUM SNAME                RATING
---------- -------------------- ----------
         1 Brustus                     1
         2 Lubber                      8
         3 Horatio                     7
         4 Andy                        8

SQL> select rownum,sname,rating from sailors where rownum>5;

no rows selected
```

greatest()-

```
SQL> select sname,sid,rating,greatest(sid,nvl(rating,0)) from sailors;

SNAME                      SID     RATING GREATEST(SID,NVL(RATING,0))
-------------------- ---------- ---------- --------------------------
Brustus                     29          1                         29
Lubber                      22          8                         22
Horatio                     64          7                         64
Andy                        32          8                         32
Art                         85          3                         85
Bob                         95          3                         95
RAM                        100          8                        100

7 rows selected.
```

least()-

```
SQL> select sname,sid,rating,least(sid,nvl(rating,0)) from sailors;

SNAME                      SID     RATING LEAST(SID,NVL(RATING,0))
-------------------- ---------- ---------- -----------------------
Brustus                     29          1                       1
Lubber                      22          8                       8
Horatio                     64          7                       7
Andy                        32          8                       8
Art                         85          3                       3
Bob                         95          3                       3
RAM                        100          8                       8

7 rows selected.
```

decode()

```
SQL> select sname, rating, decode(rating, 10, 'High', 7, 'Medium
  2  ', 3, 'Low') as rating_level from sailors;

SNAME                    RATING RATING_
-------------------- ---------- -------
Dustin                        7 Medium
Brutus                        1
Lubber                        8
Horatio                       7 Medium
Art                           3 Low
Bob                           3 Low

6 rows selected.
```

```
SQL> select sname, rating, decode(rating, 10, 'High', 7, 'Medium
  2  ', 3, 'Low', 'Others') as rating_level from sailors;

SNAME                    RATING RATING_
-------------------- ---------- -------
Dustin                        7 Medium
Brutus                        1 Others
Lubber                        8 Others
Horatio                       7 Medium
Art                           3 Low
Bob                           3 Low

6 rows selected.
```

case

```
SQL> select rating, case rating when 10 then 'High'
  2                              when 7 then 'Medium'
  3                              when 3 then 'Low'
  4                              else 'Others'
  5                              end
  6  from sailors
  7  ;

    RATING CASERA
---------- ------
         7 Medium
         1 Others
         8 Others
         7 Medium
         3 Low
         3 Low

6 rows selected.

SQL> select rating, case when rating >= 7 then 'High'
  2                      when rating < 7 then 'Low'
  3                      else 'Others'
  4                      end
  5  from sailors;

    RATING CASEWH
---------- ------
         7 High
         1 Low
         8 High
         7 High
         3 Low
         3 Low

6 rows selected.
```

to_char(), to_number(), to_date()

```
SQL> select to_char(100000, '$99,9999.99') from dual;

TO_CHAR(1000
------------
 $10,0000.00

SQL> select '100' + '2,000' from dual;
select '100' + '2,000' from dual
             *
ERROR at line 1:
ORA-01722: invalid number


SQL> select '100' + to_number('2,000', '9,999') from dual;

'100'+TO_NUMBER('2,000','9,999')
--------------------------------
                            2100

SQL> select to_date('01-jan-2015') from dual;

TO_DATE('
---------
01-JAN-15

SQL> select to_date('01/01/2015', 'dd/mm/yyyy') from dual;

TO_DATE('
---------
01-JAN-15
```

to_char()

```
SQL> select to_char(sysdate, 'hh:mm:ss pm') from dual;

TO_CHAR(SYS
-----------
11:04:02 pm
```

Exercise:

1. Retrieve even rows from sailors table

A) select * from (select rownumrnum, sid, sname, rating, age from sailors) where mod(rnum, 2) = 0;

```
SQL> select * from (select rownum rnum, sid, sname, rating, age from sailors) where mod(rnum, 2) = 0;

      RNUM        SID SNAME                    RATING        AGE
---------- ---------- -------------------- ---------- ----------
         2         29 Brutus                        1         33
         4         64 Horatio                       7         35
         6         95 Bob                           3       63.5
```

2. Write a query to retrieve Nth row where N is entered at runtime

A) select * from (select rownumrnum, sid, sname, rating, age from sailors) where rnum = &rnum;

```
SQL> select * from (select rownum rnum, sid, sname, rating, age from sailors) where rnum = &rnum;
Enter value for rnum: 4
old   1: select * from (select rownum rnum, sid, sname, rating, age from sailors) where rnum = &rnum
new   1: select * from (select rownum rnum, sid, sname, rating, age from sailors) where rnum = 4

      RNUM        SID SNAME                    RATING        AGE
---------- ---------- -------------------- ---------- ----------
         4         64 Horatio                       7         35
```

3. Find details of the reservations made by sailor 22 in the month of AUG and OCT.

A) select * from reserves where sid = 22 and day between '31-jul-98' and '01-sep-98' or day between '30-sep-98' and '01-nov-98';

```
SQL> select * from reserves where sid = 22 and day between '31-jul-98' and '01-sep-98' or day between '30-sep-98'
and '01-nov-98';

       SID        BID DAY
---------- ---------- ---------
        22        101 10-OCT-98
        22        103 08-OCT-98
```

4. Find sailors details (do not use wildcards)

a. Whose name contains ‘S’ as a 3rd character.

b. Whose name contains ‘E’ as a 2nd character from the end of the string.

c. Whose name contains the letter ‘A’.

d. Whose name contains the letter ‘O’ only once.

A) a) select * from sailors where instr(sname, 's') = 3;

```
SQL> select * from sailors where instr(sname, 's') = 3;

       SID SNAME                    RATING        AGE
---------- -------------------- ---------- ----------
        22 Dustin                        7         45
```

b) select * from sailors where instr(sname, 'e') = length(sname) - 2 + 1;

```
SQL> select * from sailors where instr(sname, 'e') = length(sname) - 2 + 1;

       SID SNAME                    RATING        AGE
---------- -------------------- ---------- ----------
        31 Lubber                        8       55.5
```

c) select * from sailors where instr(sname, 'a') != 0 or instr(sname, 'A') != 0;

```
SQL> select * from sailors where instr(sname, 'a') != 0 or instr(sname, 'A') != 0;

       SID SNAME                    RATING        AGE
---------- -------------------- ---------- ----------
        64 Horatio                       7         35
        85 Art                           3       25.5
```

5. Write a query to display today's date in dd-month yyyyhh:mi:ss format.

A) select to_char(sysdate, 'dd-month yyyyhh:mi:ss') from dual;

```
SQL> select to_char(sysdate, 'dd-month yyyy hh:mi:ss') from dual;

TO_CHAR(SYSDATE,'DD-MONTHYYYYHH:MI:SS')
--------------------------------------------------
26-april     2021 11:50:57
```

6. Write a query to display today's date in the given format: Monday 3rd August 2015
A) select to_char(sysdate, 'DAY DD"th" MONTH, YYYY') from dual;

```
SQL> select to_char(sysdate, 'day dd"th" month yyyy') from dual;

TO_CHAR(SYSDATE,'DAYDD"TH"MONTHYYYY')
-------------------------------------------------------------------
tuesday   27th april     2021
```

7. Write a query to display reservation sorted by month
A) select * from reserves order by day;

```
SQL> select * from reserves order by day;

       SID        BID DAY
---------- ---------- ---------
        64        101 05-SEP-98
        22        103 08-OCT-98
        22        101 10-OCT-98
```

8. Write a query to display : Today the date is: 03.08.2015
A) select 'Today the date is ' || to_char(sysdate, 'dd.mm.yyyy') from dual;

```
SQL> select 'Today the date is ' || to_char(sysdate, 'dd.mm.yyyy') from dual;

'TODAYTHEDATEIS'||TO_CHAR(SY
----------------------------
Today the date is 26.04.2021
```

Result: The execution of in-built SQL functions has been done successfully

Viva questions

1) Give the syntax of any 4 character built-in functions
A) select upper(string/columnName) from dual/tableName;
select lower(string/columnName) from dual/tableName;
select initcap(string/columnName) from dual/tableName;
select length(string/columnName) from dual/tableName;

2) Give the syntax and purpose of any 6 built-in number functions
A) select round(number, n[, m]) from dual; - rounds to the nearest according to the given n
select trunc(number, n) from dual; - simply removes all the extra digits as per n
select power(base, exp) from dual; - to calculate power of a number
select mod(num, denom) from dual; - to find modulus of a number when divided by another number
select sqrt(number) from dual; - to find square root of a number
select abs(number) from dual; - to find absolute of a number

3) Give the syntax for any two in-built date functions in SQL

A) select sysdate from dual;
select last_day(sysdate) from dual;

4) Give the syntax and formats for string conversion function in SQL (to_char)
A)) TO_CHAR(number, FORMAT): Number is converted to string as
specified by the format
Formats for Numbers:
. Prints the Decimal Point
, Prints the comma to represent thousands
9 Each 9 represents one digit in the result
0 Represents a leading zero to be displayed
$ dollar sign printed to the left of number

5) What is the difference between the functions translate and replace.
A) translate replaces strings character by character, where as replace function replaces the entire substring with a given set of characters.
ex: replace('corporation', 'ora', '13') = corp13tion
and translate('corporation', 'ora', '13') = c13p13ti1n

6) Write the syntax for the decode function in SQL
A) select DECODE(string/column,search1,replace1[,search2,replace2,…][default_value]);
ex: Select sname, rating,decode(rating,10,'High',7,'Medium',3,'Low') as rating_level from sailors;

7) Give the syntax and formats for number conversion function in SQL (to_number)
A) select TO_NUMBER(string[,format]) from dual; string is converted to a number as specified by the format

8) What is the difference between the NVL and the NVL2 functions?
A) The nvl function only has two parameters while the nvl parameter has three arguments. The nvl2 like like combining an nvl with a decode because you can transform a value: NVL (expr1 , expr2): If expr1 is null, then NVL returns expr2.

9) What is basic purpose of dual in oracle?
A) Dual is a special one-row, one-column table present by default in Oracle. It has a column called DUMMY of type VARCHAR2(1), and has a value 'X' as its row. It is suitable for use in selecting a pseudo column.
10)Differentiate LPAD and RPAD?
A) lpad is basically right justifying the given string, and then filling the extra space in the left (left padding) with a given character (by default space)
rpad is the opposite of lpad. it is left justifying the given string, and then filling the extra space in the right (right padding) with a given character (by default space)

JOINS IN SQL

Aim:

To execute the different types of joins in SQL, sorting and group by clause.

Examples

```
SQL> select s.sid, s.sname, r.bid
  2  from sailors s INNER JOIN reserves r
  3  ON s.sid = r.sid;

       SID SNAME                          BID
---------- -------------------- ----------
        22 Dustin                         101
        22 Dustin                         103
        64 Horatio                        101

SQL> select s.sid, s.sname, r.bid
  2  from sailors s, reserves r
  3  where s.sid = r.sid;

       SID SNAME                          BID
---------- -------------------- ----------
        22 Dustin                         101
        22 Dustin                         103
        64 Horatio                        101

SQL> select sid, sname, bid
  2  from sailors NATURAL JOIN reserves;

       SID SNAME                          BID
---------- -------------------- ----------
        22 Dustin                         101
        22 Dustin                         103
        64 Horatio                        101
```

```
SQL> select s.sid, s.sname, r.bid
  2  from sailors s LEFT OUTER JOIN reserves r
  3  on s.sid = r.sid;

       SID SNAME                          BID
---------- -------------------- ----------
        22 Dustin                       101
        22 Dustin                       103
        29 Brutus
        31 Lubber
        64 Horatio                      101
        85 Art
        95 Bob

7 rows selected.

SQL> select s.sid, s.sname, r.bid
  2  from sailors s, reserves r
  3  where s.sid = r.sid (+);

       SID SNAME                          BID
---------- -------------------- ----------
        22 Dustin                       101
        22 Dustin                       103
        29 Brutus
        31 Lubber
        64 Horatio                      101
        85 Art
        95 Bob

7 rows selected.
```

```
SQL> select r.sid, r.bid, b.bname
  2  from reserves r RIGHT OUTER JOIN boats b
  3  on r.bid = b.bid;

       SID        BID BNAME
---------- ---------- --------------------
        22        101 Interlake
        64        101 Interlake
        22        103 Clipper
                      SeaBird

SQL> select r.sid, r.bid, b.bname
  2  from reserves r, boats b
  3  where r.bid (+) = b.bid;

       SID        BID BNAME
---------- ---------- --------------------
        22        101 Interlake
        64        101 Interlake
        22        103 Clipper
                      SeaBird
```

```
SQL> select r.sid, r.bid, b.bname
  2  from reserves r FULL OUTER JOIN boats b
  3  on r.bid = b.bid;

       SID        BID BNAME
---------- ---------- --------------------
        64        101 Interlake
        22        101 Interlake
        22        103 Clipper
                      SeaBird
```

```
SQL> select sid, sname, rating, age from sailors order by age;

       SID SNAME              RATING        AGE
---------- -------------- ---------- ----------
        71 zobra                  10         16
        85 art                     3         26
        32 ady                     8         26
        29 brutus                  1         33
        58 rusty                  10         35
        64 horatio                 7         35
        74 horatio                 9         36
        22 dustin                  7         45
        31 lubber                  8         56
        95 bob                     3         64

10 rows selected.
```

```
SQL> select sid, sname, rating, age from sailors order by 4;

       SID SNAME              RATING        AGE
---------- -------------- ---------- ----------
        71 zobra                  10         16
        85 art                     3         26
        32 ady                     8         26
        29 brutus                  1         33
        58 rusty                  10         35
        64 horatio                 7         35
        74 horatio                 9         36
        22 dustin                  7         45
        31 lubber                  8         56
        95 bob                     3         64

10 rows selected.
```

```
SQL> select sid, sname, rating, age from sailors order by rating desc;

       SID SNAME                RATING        AGE
---------- -------------- ---------- ----------
        58 rusty                  10         35
        71 zobra                  10         16
        74 horatio                 9         36
        31 lubber                  8         56
        32 ady                     8         26
        64 horatio                 7         35
        22 dustin                  7         45
        95 bob                     3         64
        85 art                     3         26
        29 brutus                  1         33

10 rows selected.
```

```
SQL> select sid, sname, rating, age from sailors order by 3 desc;

       SID SNAME                RATING        AGE
---------- -------------- ---------- ----------
        58 rusty                  10         35
        71 zobra                  10         16
        74 horatio                 9         36
        31 lubber                  8         56
        32 ady                     8         26
        64 horatio                 7         35
        22 dustin                  7         45
        95 bob                     3         64
        85 art                     3         26
        29 brutus                  1         33

10 rows selected.
```

```
SQL> select sid, sname, rating R, age from sailors order by R;

       SID SNAME                     R        AGE
---------- -------------- ---------- ----------
        29 brutus                  1         33
        85 art                     3         26
        95 bob                     3         64
        64 horatio                 7         35
        22 dustin                  7         45
        31 lubber                  8         56
        32 ady                     8         26
        74 horatio                 9         36
        71 zobra                  10         16
        58 rusty                  10         35

10 rows selected.
```

```
SQL> select * from sailors order by 2;

       SID SNAME                    AGE     RATING
---------- -------------- ---------- ----------
        32 ady                      26          8
        85 art                      26          3
        95 bob                      64          3
        29 brutus                   33          1
        22 dustin                   45          7
        64 horatio                  35          7
        74 horatio                  36          9
        31 lubber                   56          8
        58 rusty                    35         10
        71 zobra                    16         10

10 rows selected.
```

```
SQL> select s.rating, avg(s.age) as avg_age
  2  from sailors s
  3  where s.age >= 40
  4  group by s.rating;

    RATING    AVG_AGE
---------- ----------
         8         56
         7         45
         3         64
```

```
SQL> select s.rating, avg(s.age) as avg_age
  2  from sailors s
  3  group by s.rating
  4  having avg(s.age) >= 40;

    RATING    AVG_AGE
---------- ----------
         8         41
         7         40
         3         45
```

Exercise

1. Find the names of Sailors who have reserved boat number 103
A) select sname
from sailors s, reserves r
where r.sid = s.sid and r.bid = 103;
(OR)
select sname
from sailors NATURAL JOIN reserves
where bid = 103;
(OR)
select sname
from sailors s INNER JOIN reserves r

on s.sid = r.sid
where r.bid = 103;

```
SQL> select sname
  2  from sailors s, reserves r
  3  where r.sid = s.sid and r.bid = 103;

SNAME
--------------------
Lubber
Dustin

SQL> select sname
  2  from sailors NATURAL JOIN reserves
  3  where bid = 103;

SNAME
--------------------
Lubber
Dustin

SQL> select sname
  2  from sailors s INNER JOIN reserves r
  3  on s.sid = r.sid
  4  where r.bid = 103;

SNAME
--------------------
Lubber
Dustin
```

2. Find the names of the Sailors who have reserved a red boat
A) select sname
from sailors s, boats b, reserves r
where s.sid = r.sid and r.bid = b.bid and b.color = 'red';
(OR)
select sname
from (select * from sailors NATURAL JOIN reserves) NATURAL JOIN boats
where color = 'red';

```
SQL> select sname
  2  from sailors s, boats b, reserves r
  3  where s.sid = r.sid and r.bid = b.bid and b.color = 'red';


SNAME
--------------------
Dustin
Lubber

SQL> select sname
  2  from (select * from sailors NATURAL JOIN reserves) NATURAL
 JOIN boats
  3  where color = 'red';

SNAME
--------------------
Dustin
Lubber
```

3. Find the names and colors of the boats reserved by the sailor named Lubber

A) select bname, color
from (select * from sailors NATURAL JOIN reserves) NATURAL JOIN boats
where sname = 'Lubber';

(OR)

select bname, color
from boats b, reserves r, sailors s
where s.sid = r.sid and r.bid = b.bid and sname = 'Lubber';

```
SQL> select bname, color
  2  from (select * from sailors NATURAL JOIN reserves) NATURAL JOIN boats
  3  where sname = 'Lubber';

BNAME                COLOR
-------------------- ----------
Interlake            red
Clipper              green

SQL> select bname, color
  2  from boats b, reserves r, sailors s
  3  where s.sid = r.sid and r.bid = b.bid and sname = 'Lubber';

BNAME                COLOR
-------------------- ----------
Interlake            red
Clipper              green
```

4. Find the Sailors who have reserved at least one boat
A) select distinct sname
 from sailors s INNER JOIN reserves r
 on s.sid = r.sid;
(OR)
select distinct sname
from sailors s NATURAL JOIN reserves r;
(OR)
select distinct s.sname
from sailors s, reserves r
where s.sid = r.sid;

```
SQL> select distinct sname
  2  from sailors s INNER JOIN reserves r
  3  on s.sid = r.sid;

SNAME
--------------------
Lubber
Dustin
Horatio

SQL> select distinct sname
  2  from sailors s NATURAL JOIN reserves r
  3  ;

SNAME
--------------------
Lubber
Dustin
Horatio

SQL> select distinct s.sname
  2  from sailors s, reserves r
  3  where s.sid = r.sid;

SNAME
--------------------
Lubber
Dustin
Horatio
```

5. Compute Increments for the ratings of sailors who have sailed two different boats on the same day.
A) update sailors set rating = rating + 1
where sid in

select s.sid
from sailors s, reserves r1, reserves r2
where s.sid = r1.sid and r1.sid = r2.sid and r1.bid != r2.bid and r1.day = r2.day);

```
SQL> update sailors set rating = rating + 1
  2  where sid in
  3     (select s.sid
  4     from sailors s, reserves r1, reserves r2
  5     where s.sid = r1.sid and r1.sid = r2.sid and r1.bid != r2.bid and r1.day = r2.day);

1 row updated.
```

6. Find the sailors who have reserved both a red and a green boat.
A) select s.sid, s.sname
from sailors s, reserves r, boats b
where s.sid = r.sid and r.bid = b.bid and b.color = ‘red’)
intersect
select s.sid, s.sname
from sailors s, reserves r, boats b
where s.sid = r.sid and r.bid = b.bid and b.color = ‘green’;

```
SQL> select s.sid, s.sname
  2  from sailors s, reserves r, boats b
  3  where s.sid = r.sid and r.bid = b.bid and b.color = 'red'
  4  intersect
  5  select s.sid, s.sname
  6  from sailors s, reserves r, boats b
  7  where s.sid = r.sid and r.bid = b.bid and b.color = 'green';

       SID SNAME
---------- ---------------
        22 dustin
        31 lubber
```

7. Find the age of the youngest sailor for each rating level.
A) select rating, min(age)
from sailors
group by rating;

```
SQL> select rating, min(age)
  2  from sailors
  3  group by rating;

    RATING   MIN(AGE)
---------- ----------
         1         33
         8         26
         7         35
         3         26
        10         16
         9         36

6 rows selected.
```

8. Find the age of the youngest sailor who is eligible to vote, for each rating level with at least two such sailors.
A) select min(age), rating
from sailors s
where age >= 18
group by rating
having count(*) > 1;

```
SQL> select min(age), rating
  2  from sailors s
  3  where age >= 18
  4  group by rating
  5  having count(*) > 1;

  MIN(AGE)     RATING
---------- ----------
        26          8
        35          7
        26          3
```

9. For each red boat, find the number of reservations for the boat.
A) select count(*), r.bid
from boats b, reserves r
where b.bid = r.bid and b.color='red'
group by r.bid;

```
SQL> select count(*), r.bid
  2  from boats b, reserves r
  3  where b.bid = r.bid and b.color='red'
  4  group by r.bid;

  COUNT(*)        BID
---------- ----------
         3        102
         2        104
```

10. Write a query to give bid and the number of reservations for each boat in the Boats table.
A) select count(*), b.bid
from boats b, reserves r
where b.bid = r.bid
group by b.bid;

```
SQL> select count(*), b.bid
  2  from boats b, reserves r
  3  where b.bid = r.bid
  4  group by b.bid;

  COUNT(*)        BID
---------- ----------
         3        102
         2        101
         2        104
         3        103
```

11. Find the average age of sailors for each rating level that has at least two sailors.
A) select avg(age), rating
from sailors
group by rating
having count(*) > 1;

```
SQL> select avg(age), rating
  2  from sailors
  3  group by rating
  4  having count(*) > 1;

  AVG(AGE)     RATING
---------- ----------
        41          8
        40          7
        45          3
      25.5         10
```

12. Display duplicate rows
A)selectsid
from sail
group by sid
having count(sid) >1;

```
SQL> select sid
  2  from sail
  3  group by sid
  4  having count(sid) > 1;

       SID
----------
        22
```

14 a) Find the details of the sailors with - Top 3 ratings
b) Find the details of the Top 3 sailors (Top-N analysis)
A) a) select sid, sname, rating, age
from (select rownum, sid, sname, rating, age
from sailors
where rownum<= 3
order by rating desc);

```
SQL> select sid, sname, rating, age
  2  from (select rownum, sid, sname, rating, age
  3  from sailors
  4  where rownum <= 3
  5  order by rating desc);

       SID SNAME              RATING        AGE
---------- -------------- ---------- ----------
        31 lubber                  8         56
        22 dustin                  7         45
        29 brutus                  1         33
```

b) select sid, sname, rating, age
from (select rownum, sid, sname, rating, age
from sailors
where rownum<= &n
order by rating desc);

```
SQL> select sid, sname, rating, age
  2  from (select rownum, sid, sname, rating, age
  3  from sailors
  4  where rownum <= &n
  5  order by rating desc);
Enter value for n: 6
old   4: where rownum <= &n
new   4: where rownum <= 6

       SID SNAME              RATING        AGE
---------- -------------- ---------- ----------
        58 rusty                  10         35
        31 lubber                  8         56
        32 ady                     8         26
        22 dustin                  7         45
        64 horatio                 7         35
        29 brutus                  1         33

6 rows selected.
```

15. Display the details of the sailors with the Top 3rd rating
A) select * from sailors
where sid in (select decode(rownum, 3, sid)
from (select * from sailors order by rating desc));

```
SQL> select * from sailors
  2  where sid in (select decode(rownum, 3, sid)
  3  from (select * from sailors order by rating desc));

       SID SNAME                AGE     RATING
---------- -------------- ---------- ----------
        74 horatio                 36          9
```

16. Find those rating(s) for which the average age of sailors is the minimum overall ratings.
A)SELECT sailors1.rating, sailors1.avgage

```
FROM (SELECT rating, avg(age) as avgage
            FROM Sailors
            GROUP BY rating) sailors1
WHERE sailors1.avgage = (SELECT MIN(avgage)
                                FROM (SELECT rating, avg(age) as avgage
                                        FROM Sailors
                                       GROUP BY rating) sailors1);
```

```
SQL> select s.rating, s.avgage
  2  from (select rating, avg(age) as avgage
  3             from sailors
  4             group by rating) s
  5  where s.avgage = (select min(avgage)
  6                    from (select rating, avg(age) as avgage
  7                          from sailors
  8                          group by rating) s);

    RATING     AVGAGE
---------- ----------
        10       25.5
```

Result:

Execution of types of joins, sorting and group by has been done successfully.

Viva Questions

1. What is a JOIN Clause?
A) A JOIN clause is used to combine rows from two or more tables, based on a related column between them

2. What are the different types of SQL Join clauses and how are they used?
A) INNER JOIN: To return records that have matching values in both tables
LEFT OUTER JOIN: To return all records from the left table, and the matched records from the right table
RIGHT OUTER JOIN: To returns all records from the right table, and the matched records from the left table
FULL OUTER JOIN: To return all records when there is a match in either left or right table

3. Differentiate inner join and outer join
A) Inner joins result in the intersection of two tables, whereas outer joins result in the union of two tables.

4. Differentiate left join and right join
A) The LEFT JOIN includes all records from the left side and matched rows from the right table, whereas RIGHT JOIN returns all rows from the right side and unmatched rows from the left table.

5. What are different types of outer join?
A) Right outer join, left outer join, full outer join.

6. What is the difference between joins and union?
A) Joins combine data into new columns. If two tables are joined together, then the data from the first table is shown in one set of columns alongside the second table's column in the same row. The schema is changed in case of joins.

Unions combine data into new rows. If two tables are "unioned" together, then the data from the first table is in one set of rows, and the data from the second table in another set. The rows are in the same result. the schema is not changed in case of union.

7. Can you join table by itself? If Yes how? If no Why?
A) Yes we can, we can join a table by itself by making two copies of the same table.

8. What is Cartesian join?
A) Cartesian join is a special type of join in which we join each row of one table to every row of another table.

9. What is a tuple? Are tuples and records related to each other? If yes how?
A) A tuple is very similar to a record. Both of them represent values of a table in SQL. A record has fields, and these are named. While tuples are an ordered collection of data values,a tuple is an unordered collection of labeled data values.

10. Define a relation?
A) The term relation schema refers to a heading paired with a set of constraints defined in terms of that heading. Relations are represented by tables, where each column refers to the attributes of the table and each row of a table represents a single tuple or record.

Nested Queries, SQL Operators

Aim:

To execute nested queries, sub-queries and SQL operators.

Examples:

```
SQL> SELECT sname, age FROM sailors
  2  WHERE rating=7
  3  UNION
  4  SELECT sname, age FROM sailors
  5  where rating = 10;

SNAME                 AGE
--------------- ----------
horatio                35
rusty                  35
zobra                  16
```

```
SQL> SELECT bid FROM boats
  2  UNION ALL
  3  SELECT bid FROM reserves;

       BID
----------
       101
       102
       103
       104
       101
       102
       103
       104
       102
       103
       104

       BID
----------
       101
       102
       103

14 rows selected.
```

```
SQL> select bname
  2  from boats
  3  where color = 'red'
  4  intersect
  5  select bname
  6  from boats
  7  where color = 'blue';

BNAME
--------------
interlake
```

```
SQL> SELECT sname FROM sailors
  2  where rating = 7
  3  minus
  4  select sname FROM sailors
  5  where rating = 9;

no rows selected
```

```
SQL> SELECT * FROM sailors WHERE age BETWEEN 25 AND 35;

       SID SNAME                    AGE     RATING
---------- --------------- ---------- ----------
        29 brutus                   33          1
        32 ady                      26          8
        58 rusty                    35         10
        64 horatio                  35          7
        85 art                      26          3
```

```
SQL>  SELECT * FROM sailors WHERE rating IN (7,10);

       SID SNAME                    AGE     RATING
---------- --------------- ---------- ----------
        58 rusty                    35         10
        64 horatio                  35          7
        71 zobra                    16         10
```

```
SQL> UPDATE sailors SET age=NULL WHERE rating=7;

1 row updated.

SQL> SELECT * FROM sailors WHERE age IS NULL;

       SID SNAME                    AGE     RATING
---------- --------------- ---------- ----------
        64 horatio                             7
```

```
SQL>  SELECT UNIQUE rating FROM sailors;

    RATING
----------
         1
         8
         7
         3
         9
        10

6 rows selected.
```

```
SQL> SELECT S.sname, S.rating
  2  FROM Sailors S
  3  WHERE S.rating > ANY (SELECT S2.rating
  4  FROM Sailors S2
  5  WHERE S2.sname = 'Horatio');

no rows selected
```

```
SQL> SELECT S.sname, S.rating
  2  FROM Sailors S
  3  WHERE S.rating > ALL (SELECT S2.rating
  4  FROM Sailors S2
  5  WHERE S2.sname = 'Horatio');

SNAME                RATING
-------------- ----------
brutus                    1
art                       3
bob                       3
horatio                   7
lubber                    8
ady                       8
dustin                    9
horatio                   9
zobra                    10
rusty                    10

10 rows selected.
```

```
SQL> SELECT S.sid, S.sname, S.age
  2  FROM Sailors S
  3  WHERE S.age <= ALL (SELECT age
  4  FROM Sailors);

no rows selected
```

Exercise:

1. Find the names of sailors who have reserved boat 103.
A) SELECT S.sname
FROM Sailors S
WHERE S.sid IN (SELECTR.sid

FROM Reserves R
WHERE R.bid = 103);

```
SQL> SELECT S.sname
  2  FROM Sailors S
  3  WHERE S.sid IN ( SELECT R.sid
  4                   FROM Reserves R
  5                   WHERE R.bid = 103 );

SNAME
---------------
dustin
lubber
horatio
```

2.Find the names of the sailors who have reserved a red boat.

```
A) select s.sname
from sailors s
where s.sid in (select r.sid
            from reserves r
            where r.bid in (select b.bid
                  from boats b
                  where b.color = 'red'));
```

```
SQL> select s.sname
  2  from sailors s
  3  where s.sid in (select r.sid
  4                  from reserves r
  5                  where r.bid in (select b.bid
  6                          from boats b
  7                          where b.color = 'red'));

SNAME
---------------
lubber
dustin
horatio
```

3. Find the names of the sailors who have not reserved a red boat.

```
A) select s.sname
 from sailors s
 where s.sid not in (select r.sid
              from reserves r
              where r.bid in (select b.bid
                    from boats b
                    where b.color = 'red'));
```

```
SQL> select s.sname
  2  from sailors s
  3  where s.sid not in (select r.sid
  4                      from reserves r
  5                      where r.bid in (select b.bid
  6                              from boats b
  7                              where b.color = 'red'));

SNAME
---------------
zobra
art
rusty
ady
brutus
horatio
bob

7 rows selected.
```

4. Find the names of Sailors who have reserved boat#103.

A) SELECT S.sname FROM Sailors S WHERE S.sid IN (SELECT R.sid FROM Reserves R WHERE R.bid = 103)

```
SQL> SELECT S.sname
  2  FROM Sailors S
  3  WHERE S.sid IN ( SELECT R.sid
  4                    FROM Reserves R
  5                    WHERE R.bid = 103 );

SNAME
---------------
dustin
lubber
horatio
```

5. Find Sailors whose rating is better than every Sailor called Horatio.

A) SELECT S.sname, S.rating
FROM Sailors S WHERE S.rating> ALL (SELECT S2.rating
FROM Sailors S2
WHERE S2.sname = ‘Horatio’);

```
SQL> SELECT S.sname, S.rating
  2  FROM Sailors S WHERE S.rating > ALL (SELECT S2.rating
  3                                       FROM Sailors S2
  4                                       WHERE S2.sname = 'Horatio');

SNAME               RATING
--------------- ----------
brutus                   1
art                      3
bob                      3
horatio                  7
lubber                   8
ady                      8
dustin                   9
horatio                  9
zobra                   10
rusty                   10

10 rows selected.
```

6. Find the Sailor with the highest rating.

A) SELECT S.sname
FROM Sailors S
WHERE S.rating>= ALL (SELECT rating
FROM Sailors);

```
SQL> SELECT S.sname
  2  FROM Sailors S
  3  WHERE S.rating >= ALL (SELECT rating
  4                         FROM Sailors);

SNAME
---------------
zobra
rusty
```

7. Find the names of sailors who have reserved all boats.

A) SELECT S.sname
FROM Sailors S
WHERE NOT EXISTS ((SELECT B.bid
FROM Boats B) MINUS (SELECT R.bid
FROM Reserves R
WHERE R.sid = S.sid));

```
SQL> SELECT S.sname
  2  FROM Sailors S
  3  WHERE NOT EXISTS ((SELECT B.bid
  4  FROM Boats B) MINUS (SELECT R.bid
  5                       FROM Reserves R
  6                       WHERE R.sid = S.sid));

SNAME
---------------
dustin
```

8.Find the average age of all sailors.
A) select avg(age)
from sailors;

```
SQL> select avg(age)
  2  from sailors;

  AVG(AGE)
----------
      37.2
```

9. Find the average age of sailors with a rating of 10.
A) select avg(age)
from sailors
where rating = 10;

```
SQL> select avg(age)
  2  from sailors
  3  where rating = 10;

  AVG(AGE)
----------
      25.5
```

10. Find the name and the age of the oldest sailor.
A) select s.sname, s.age
from sailors s
where s.age>= all (select s2.age
from sailors s2);

```
SQL> select s.sname, s.age
  2  from sailors s
  3  where s.age >= all (select s2.age
  4                      from sailors s2);

SNAME                 AGE
--------------- ----------
bob                   64
```

11. Count the number of sailors.
A) select count(*)
from sailors;

```
SQL> select count(*)
  2  from sailors;

  COUNT(*)
----------
        10
```

12. Count the number of different sailor names.
A) select count(*)

from (select distinct sname from sailors);

```
SQL> select count(*)
  2  from (select distinct sname from sailors);

  COUNT(*)
----------
         9
```

13. Find the names of Sailors who are older than the oldest sailor with a rating of 10.
A) select sname, age
from sailors
where age > (select max(age)
from sailors
where rating = 10);

```
SQL> select sname, age
  2  from sailors
  3  where age > (select max(age)
  4                      from sailors
  5                      where rating = 10);

SNAME                    AGE
--------------- ----------
dustin                   45
lubber                   56
horatio                  36
bob                      64
```

Result:

Execution of nested queries, sub queries and SQL operators has been done successfully.

Views, Synonyms and Sequences

Aim:

Execution of Views, Synonyms and Sequences.

Examples:

```
SQL> create view sail2 as select sid, sname, rating from sailors;

View created.

SQL> select * from sail2;

       SID SNAME              RATING
---------- -------------- ----------
        22 dustin                  9
        29 brutus                  1
        31 lubber                  8
        32 ady                     8
        58 rusty                  10
        64 horatio                 7
        71 zobra                  10
        74 horatio                 9
        85 art                     3
        95 bob                     3

10 rows selected.
```

```
SQL> insert into sail2 values (100, 'John', 8);

1 row created.

SQL> select * from sail2;

       SID SNAME               RATING
---------- -------------- ----------
        22 dustin                  9
        29 brutus                  1
        31 lubber                  8
        32 ady                     8
        58 rusty                  10
        64 horatio                 7
        71 zobra                  10
        74 horatio                 9
        85 art                     3
        95 bob                     3
       100 John                    8

11 rows selected.

SQL> select * from sailors;

       SID SNAME                   AGE     RATING
---------- -------------- ---------- ----------
        22 dustin                  45          9
        29 brutus                  33          1
        31 lubber                  56          8
        32 ady                     26          8
        58 rusty                   35         10
        64 horatio                 35          7
        71 zobra                   16         10
        74 horatio                 36          9
        85 art                     26          3
        95 bob                     64          3
       100 John                                 8

11 rows selected.
```

```
SQL> update sail2 set sname = 'Jack' where sid = 100;

1 row updated.

SQL> select * from sail2;

       SID SNAME               RATING
---------- -------------- ----------
        22 dustin                  9
        29 brutus                  1
        31 lubber                  8
        32 ady                     8
        58 rusty                  10
        64 horatio                 7
        71 zobra                  10
        74 horatio                 9
        85 art                     3
        95 bob                     3
       100 Jack                    8

11 rows selected.

SQL> select * from sailors;

       SID SNAME                   AGE     RATING
---------- -------------- ---------- ----------
        22 dustin                  45          9
        29 brutus                  33          1
        31 lubber                  56          8
        32 ady                     26          8
        58 rusty                   35         10
        64 horatio                 35          7
        71 zobra                   16         10
        74 horatio                 36          9
        85 art                     26          3
        95 bob                     64          3
       100 Jack                                 8

11 rows selected.
```

```
SQL> delete from sailors where sid = 100;

1 row deleted.

SQL> select * from sail2;

       SID SNAME               RATING
---------- -------------- ----------
        22 dustin                  9
        29 brutus                  1
        31 lubber                  8
        32 ady                     8
        58 rusty                  10
        64 horatio                 7
        71 zobra                  10
        74 horatio                 9
        85 art                     3
        95 bob                     3

10 rows selected.

SQL> select * from sailors;

       SID SNAME                   AGE     RATING
---------- -------------- ---------- ----------
        22 dustin                  45          9
        29 brutus                  33          1
        31 lubber                  56          8
        32 ady                     26          8
        58 rusty                   35         10
        64 horatio                 35          7
        71 zobra                   16         10
        74 horatio                 36          9
        85 art                     26          3
        95 bob                     64          3

10 rows selected.
```

```
SQL> drop view sail2;

View dropped.

SQL> select * from sail2;
select * from sail2
              *
ERROR at line 1:
ORA-00942: table or view does not exist
```

```
SQL> create view sail2 as select sid, sname, rating from sailors where rating = 7;

View created.

SQL> select * from sail2;

       SID SNAME               RATING
---------- -------------- ----------
        64 horatio                 7
```

```
SQL> insert into sail2 values(100, 'John', 7);

1 row created.

SQL> insert into sail2 values(101, 'Jack', 9);

1 row created.

SQL> select * from sail2;

       SID SNAME               RATING
---------- --------------- ----------
        64 horatio                  7
       100 John                     7
```

```
SQL> create view sail2 as select sid, sname, rating from sailors
  2  where rating = 7
  3  with check option;

View created.

SQL> select * from sail2;

       SID SNAME               RATING
---------- --------------- ----------
        64 horatio                  7
       100 John                     7
```

```
SQL> insert into sail2 values(102, 'John', 7);

1 row created.

SQL> insert into sail2 values(103, 'John', 9);
insert into sail2 values(103, 'John', 9)
            *
ERROR at line 1:
ORA-01402: view WITH CHECK OPTION where-clause violation


SQL> select * from sail2;

       SID SNAME               RATING
---------- --------------- ----------
        64 horatio                  7
       100 John                     7
       102 John                     7

SQL> drop view sail2;

View dropped.
```

```
SQL> create view sail2 as select sid, sname, rating from sailors
  2  where rating = 7
  3  with read only;

View created.

SQL> select * from sail2;

       SID SNAME               RATING
---------- --------------- ----------
        64 horatio                  7
       100 John                     7
       102 John                     7
```

```
SQL> insert into sail2 values(103, 'John', 7);
insert into sail2 values(103, 'John', 7)
*
ERROR at line 1:
ORA-42399: cannot perform a DML operation on a read-only view
```

```
SQL> update sail2 set sname = 'Jack'
  2  where sid = 22;
update sail2 set sname = 'Jack'
                 *
ERROR at line 1:
ORA-42399: cannot perform a DML operation on a read-only view
```

```
SQL> create force view emp
  2  as select id, name
  3  from employees;

Warning: View created with compilation errors.
```

```
SQL> select * from emp;
select * from emp
              *
ERROR at line 1:
ORA-04063: view "IT19737120.EMP" has errors


SQL> create table employees (id number, name varchar2(20));

Table created.

SQL> select * from emp;

no rows selected
```

```
SQL> insert into emp values (1, 'John');

1 row created.

SQL> select * from employees;

        ID NAME
---------- --------------------
         1 John
```

```
SQL> create synonym s for sailors;

Synonym created.

SQL> create synonym r for reserves;

Synonym created.
```

```
SQL> create table sailor_dup as (select * from sailors);

Table created.

SQL> create sequence sail_s;

Sequence created.

SQL> select sail_s.nextval from dual;

   NEXTVAL
----------
         1
```

```
SQL> insert into sailor_dup (sid, sname) values (sail_s.nextval, '&sname');
Enter value for sname: John
old   1: insert into sailor_dup (sid, sname) values (sail_s.nextval, '&sname')
new   1: insert into sailor_dup (sid, sname) values (sail_s.nextval, 'John')

1 row created.
```

```
SQL> create table persons(
  2  pid number not null,
  3  lastname varchar2(255) not null,
  4  firstname varchar(255),
  5  address varchar(255),
  6  city varchar(255) default 'Hyderabad');

Table created.
```

```
SQL> create table orders(
  2  oid number not null,
  3  orderno number not null,
  4  pid number,
  5  orderdate date default sysdate not null);

Table created.
```

Result:

Execution of Views, Synonyms and Sequences has been done successfully.

Viva questions:

1) Give the syntax for simple select.
A) SELECT [DISTINCT] select_list
FROM from_list
WHERE qualification
ORDER BY asc/desc;

2) What is the main purpose of view?
A) Views are like a logical window of tables. They are used for abstraction of data from users.

3) What are different wildcards available in oracle.
A) %(percentile) and _ (underscore)

4) How do you search for a value in a database table when you don't have the exact value to search for?
A) In such cases, the LIKE condition operator is used to select rows that match a character pattern

5) How to use LIKE operator.
A) The **SQL LIKE** operator is **used** to compare a value to similar values using wildcard operators.

6) What is the difference between WHERE and HAVING clauses of the SELECT statement
A) The condition in the where clause is applied to the rows in the from list whereas the condition in the having clause is applied according to the group by clause. This means, the where clause works on the row's data whereas the having clause works on the aggregated data.

7) What are the different kinds of joins give syntaxes
A) a) Inner join:
select select_list
from R INNER JOIN S
on r.col1 = s.col1
where condition;
b) Natural join:
select select_list
from R NATURAL JOIN S
where condition;
c) Left Outer Join:
select select_list
from R LEFT OUTER JOIN S
on r.col1 = s.col1
where condition;

d) Right Outer Join:
select select_list
from R RIGHT OUTER JOIN S
on r.col1 = s.col1
where condition;
e) Full Outer Join:
select select_list
from R FULL OUTER JOIN S
on r.col1 = s.col1
where condition;

8) What's wrong in the following query?
Select subject_code,avg(marks) from students where avg(marks)>75 group by subject_code;
A) In this instead of using where, the condition should be written in the having clause. This is because aggregate functions can only be applied in the having clause or the select list but not in the where condition.

9) Explain order by clause.
A) The order by clause is used to sort the output of the select statement based on a row, either in ascending or descending order.

10) Give the syntaxes for all the join operations.
A) a) Inner join:
select select_list
from R INNER JOIN S
on r.col1 = s.col1
where condition;
b) Natural join:
select select_list
from R NATURAL JOIN S
where condition;
c) Left Outer Join:
select select_list
from R LEFT OUTER JOIN S
on r.col1 = s.col1
where condition;
d) Right Outer Join:
select select_list
from R RIGHT OUTER JOIN S
on r.col1 = s.col1
where condition;
e) Full Outer Join:
select select_list
from R FULL OUTER JOIN S
on r.col1 = s.col1
where condition;

PLSQL

Aim:

To execute PL SQL programs

Examples:

```
SQL> set serveroutput on
SQL> Begin
  2  dbms_output.put_line('heyy');
  3  End;
  4  /
heyy

PL/SQL procedure successfully completed.
```

```
SQL> Begin
  2  dbms_output.put_line('outer block');
  3     Begin
  4     dbms_output.put_line('inner');
  5     End;
  6  End;
  7  /
outer block
inner

PL/SQL procedure successfully completed.
```

```
SQL> Declare
  2  v_num1 number(3) := &number1;
  3  v_num2 number(3) := &number2;
  4  v_sum number(5);
  5  Begin
  6  v_sum := v_num1 + v_num2;
  7  dbms_output.put_line('Total = '||v_sum);
  8  End;
  9  /
Enter value for number1: 3
old   2: v_num1 number(3) := &number1;
new   2: v_num1 number(3) := 3;
Enter value for number2: 4
old   3: v_num2 number(3) := &number2;
new   3: v_num2 number(3) := 4;
Total = 7

PL/SQL procedure successfully completed.
```

```
SQL> Declare
  2  v_sub1 number(3);
  3  v_sub2 number(3);
  4  v_total number(5);
  5  v_avg number(3);
  6  v_result varchar2(30);
  7  begin
  8  v_sub1 := &sub1Marks;
  9  v_sub2 := &sub2Marks;
 10  v_total := v_sub1 + v_sub2;
 11  v_avg := v_total / 2;
 12  if v_avg >= 75
 13     then v_result := 'Distinction';
 14  elsif (v_avg between 60 and 75)
 15     then v_result := 'First Class';
 16  elsif (v_avg between 40 and 60)
 17     then v_result := 'Second Class';
 18  else
 19     v_result := 'Fail';
 20  end if;
 21  dbms_output.put_line(v_result);
 22  end;
 23  /
Enter value for sub1marks: 90
old   8: v_sub1 := &sub1Marks;
new   8: v_sub1 := 90;
Enter value for sub2marks: 90
old   9: v_sub2 := &sub2Marks;
new   9: v_sub2 := 90;
Distinction

PL/SQL procedure successfully completed.
```

```
SQL> Declare
  2  V_num number:=&enternum;
  3  Begin
  4  Case V_num
  5  When 1 Then Dbms_output.put_line ('Violet');
  6  When 2 Then Dbms_output.put_line ('Indigo');
  7  When 3 Then Dbms_output.put_line ('Blue');
  8  When 4 Then Dbms_output.put_line ('Green');
  9  When 5 Then Dbms_output.put_line ('Yellow');
 10  When 6 Then Dbms_output.put_line ('Orange');
 11  When 7 Then Dbms_output.put_line ('Red');
 12  Else
 13   Dbms_output.put_line ('Invalid input');
 14  End case;
 15  End;
 16  /
Enter value for enternum: 3
old   2: V_num number:=&enternum;
new   2: V_num number:=3;
Blue

PL/SQL procedure successfully completed.
```

```
SQL> Declare
  2  V_no number (3):=1;
  3  V_sum number(5):=0;
  4  Begin
  5  While V_no<=10
  6  Loop
  7  Dbms_output.put_line (V_no);
  8  V_sum:= V_sum+V_no;
  9  V_no:=V_no+1;
 10  End Loop;
 11  Dbms_output.put_line ('Sum = '||V_sum);
 12  End;
 13  /
1
2
3
4
5
6
7
8
9
10
Sum = 55

PL/SQL procedure successfully completed.
```

```
SQL> Declare
  2  V_no number (3):=1;
  3  Begin
  4  While V_no<=10
  5  Loop
  6  Dbms_output.put_line (V_no);
  7  V_no:=V_no+1;
  8  If V_no=5
  9  Then Exit;
 10  END IF;
 11  End Loop;
 12  Dbms_output.put_line ('Out of loop');
 13  End;
 14  /
1
2
3
4
Out of loop

PL/SQL procedure successfully completed.
```

```
SQL> Begin
  2  For V_no in 1..10
  3  Loop
  4  If mod(V_no,2)=0
  5  Then Dbms_output.put_line (V_no);
  6  END IF;
  7  End Loop;
  8  End;
  9  /
2
4
6
8
10

PL/SQL procedure successfully completed.
```

Exercises:

1. Write a PLSQL program to compute the Factorial of a given number and display it

A)

```
Declare
v_factnumber(3);
v_nnumber(3);
Begin
v_fact := 1;
v_n := &n;
while v_n> 0
loop
v_fact := v_fact * v_n;
v_n := v_n - 1;
End loop;
dbms_output.put_line('Factorial = ' || v_fact);
End;
/
```

```
SQL> Declare
  2  v_fact number(3);
  3  v_n number(3);
  4  Begin
  5  v_fact := 1;
  6  v_n := &n;
  7  while v_n > 0
  8  loop
  9  v_fact := v_fact * v_n;
 10  v_n := v_n - 1;
 11  End loop;
 12  dbms_output.put_line('Factorial = ' || v_fact);
 13  End;
 14  /
Enter value for n: 5
old   6: v_n := &n;
new   6: v_n := 5;
Factorial = 120

PL/SQL procedure successfully completed.
```

2. Write a PLSQL program to display the first ‘n’ digits of the Fibonacci series

A) DECLARE
   num number := &n;
   n1 number := 0;
   n2 number := 1;
   n3 number;
BEGIN
dbms_output.put_line(n1);
dbms_output.put_line(n2);
   for i in 3..num
loop
    n3 := n1 + n2;
dbms_output.put_line(n3);
    n1 := n2;
    n2 := n3;
end loop;
END;
/

```
SQL> DECLARE
  2       num number := &n;
  3       n1 number := 0;
  4       n2 number := 1;
  5       n3 number;
  6  BEGIN
  7       dbms_output.put_line(n1);
  8       dbms_output.put_line(n2);
  9       for i in 3..num
 10  loop
 11        n3 := n1 + n2;
 12        dbms_output.put_line(n3);
 13        n1 := n2;
 14        n2 := n3;
 15  end loop;
 16  END;
 17  /
Enter value for n: 5
old   2:      num number := &n;
new   2:      num number := 5;
0
1
1
2
3

PL/SQL procedure successfully completed.
```

3. Write a PLSQL program to read a number and display if it is prime or not

A) declare
n number;
inumber;
flag number;
begin
i:=2;
flag:=1;
n:=&n;
for i in 2..n/2
loop
if mod(n,i)=0
then
flag:=0;
exit;
end if;
end loop;

if flag=1
then
dbms_output.put_line('prime');
else
dbms_output.put_line('not prime');
end if;

end;
/

```
SQL> declare
  2  n number;
  3  i number;
  4  flag number;
  5
  6  begin
  7  i:=2;
  8  flag:=1;
  9  n:=&n;
 10
 11  for i in 2..n/2
 12  loop
 13  if mod(n,i)=0
 14  then
 15  flag:=0;
 16  exit;
 17  end if;
 18  end loop;
 19
 20  if flag=1
 21  then
 22  dbms_output.put_line('prime');
 23  else
 24  dbms_output.put_line('not prime');
 25  end if;
 26  end;
 27  /
Enter value for n: 5
old   9: n:=&n;
new   9: n:=5;
prime

PL/SQL procedure successfully completed.
```

4. Write a PLSQL program to read a number and display if it is an armstrong number or not

A) declare
n number:=#
s number:=0;
r number;
lennumber;
m number;
begin
m:=n;
len:=length(to_char(n));
while n>0
loop
r:=mod(n,10);
s:=s+power(r,len);
n:=trunc(n/10);
end loop;
if m=s

```
   then
dbms_output.put_line('armstrong number');
   else
dbms_output.put_line('not armstrong number');
   end if;
end;
/
```

```
SQL> declare
  2       n number:=&num;
  3       s number:=0;
  4       r number;
  5       len number;
  6       m number;
  7   begin
  8       m:=n;
  9       len:=length(to_char(n));
 10       while n>0
 11       loop
 12           r:=mod(n,10);
 13           s:=s+power(r,len);
 14           n:=trunc(n/10);
 15       end loop;
 16       if m=s
 17       then
 18           dbms_output.put_line('armstrong number');
 19       else
 20           dbms_output.put_line('not armstrong number');
 21       end if;
 22   end;
 23   /
Enter value for num: 407
old   2:       n number:=&num;
new   2:       n number:=407;
armstrong number

PL/SQL procedure successfully completed.
```

5. Program to print even numbers up to the given n value.

```
A) declare
 x number;
n number := &num;
begin
for x in 1..n loop
if mod(x,2)=0 then
dbms_output.put_line( x);
end if;
end loop;
end;
/
```

```
SQL> declare
  2   x number;
  3  n number := &num;
  4  begin
  5  for x in 1..n loop
  6  if mod(x,2)=0 then
  7  dbms_output.put_line ( x);
  8  end if;
  9  end loop;
 10  end;
 11  /
Enter value for num: 10
old   3: n number := &num;
new   3: n number := 10;
2
4
6
8
10

PL/SQL procedure successfully completed.
```

6. Program to check whether the given string is palindrome or not?

A) DECLARE
 str varchar2(50):='&string';
l number := length(str);
counter number := l;
BEGIN
while counter > 0
LOOP
if (substr(str,counter,1) != substr(str, ((l+1) - counter),1))
then exit;
end if;
counter := counter - 1;
END LOOP;
IF counter=0
THEN dbms_output.put_line(str || ' is palindrome');
ELSE
dbms_output.put_line(str || ' is not palindrome');
END IF;
END;
/

```
SQL> DECLARE
  2   str varchar2(50):='&string';
  3  l number := length(str);
  4  counter number := l;
  5  BEGIN
  6  while counter > 0
  7  LOOP
  8  if  (substr(str,counter,1) != substr(str, ( ( l+1) - counter),1))
  9  then exit;
 10  end if;
 11  counter := counter - 1;
 12  END LOOP;
 13  IF counter=0
 14  THEN dbms_output.put_line(str || ' is palindrome');
 15  ELSE
 16  dbms_output.put_line(str || ' is not palindrome');
 17  END IF;
 18  END;
 19  /
Enter value for string: madam
old   2:  str varchar2(50):='&string';
new   2:  str varchar2(50):='madam';
madam is palindrome

PL/SQL procedure successfully completed.
```

Result:

Execution of PL SQL has been done successfully.

Viva Questions:

1) Write the syntax for creating a sequence
A) CREATE SEQUENCE sequence_name
  MINVALUE value
  MAXVALUE value
  START WITH value
  INCREMENT BY value
  CACHE value;

2) What happens when an object linked to a synonym is deleted?
A) When the object linked to a synonym is dropped, the synonym does not point to anything.

3) Give the syntax for for loop and while loop in PLSQL
A) WHILE condition LOOP
<statements>
end loop;
FOR counter IN initial_value ..final_value LOOP

sequence_of_statements;
END LOOP;

4) What is the difference between ELSIF ladder and the CASE statement in PLSQL
A) Case statement can only be used for equality check where as elsif ladder can be used to check any conditions.

5) What is the difference between SQL and PL/SQL?
A) SQL is a declarative language. PL/SQL is a procedural language. SQL is used to write queries, create and execute DDL and DML statements. PL/SQL is used to write program blocks, functions, procedures, triggers and packages. The basic difference between two languages is that SQL executes the single query at a time whereas, Pl/SQL executes the block of code at once.

6) Name a few schema objects that can be created using PL/SQL?
A)

7) Write the syntax for the CASE statement in PLSQL
A) CASE selector
  WHEN 'value1' THEN S1;
  WHEN 'value2' THEN S2;
  WHEN 'value3' THEN S3;
  ELSE Sn;
END CASE;

8) What are the three basic sections of a PL/SQL block?
A) Declaration, Executable, Execution

9) What are the data types available in PL/SQL?
A) Integer, floating point, character, BOOLEAN, date, collection, reference, and large object (LOB)

10) Differentiate between syntax and runtime errors?
A) A runtime error is a program error that occurs while the program is running. Whereas, a syntax error is an error in the syntax of a sequence of characters or tokens that is intended to be written in a particular programming language.

Cursors

Aim:

PL SQL select and cursors.

Examples

```
SQL> Declare
  2  V_sid Sailors.sid%type:=&sid;
  3  V_sname Sailors.sname%type;
  4  V_rating Sailors.rating%type;
  5  V_age Sailors.age%type;
  6  Begin
  7  Select sid,sname,rating,age
  8    Into V_sid,V_sname,V_rating,V_age
  9    From Sailors
 10  Where sid=V_sid;
 11   Dbms_output.put_line ('Sailor_id: '||V_sid);
 12  Dbms_output.put_line ('Sailor_name: '||V_sname);
 13  Dbms_output.put_line ('Sailor_rating: '||V_rating);
 14  Dbms_output.put_line ('Sailor_age: '||V_age);
 15  End;
 16  /
Enter value for sid: 22
old   2: V_sid Sailors.sid%type:=&sid;
new   2: V_sid Sailors.sid%type:=22;
Sailor_id: 22
Sailor_name: dustin
Sailor_rating: 9
Sailor_age: 45

PL/SQL procedure successfully completed.
```

```
SQL> Declare
  2  V_sid Sailors.sid%type:=&sid;
  3  Rec_sailors Sailors%rowtype;
  4  Begin
  5  Select *
  6      Into Rec_sailors
  7     From Sailors
  8  Where sid=V_sid;
  9  Dbms_output.put_line ('Sailor_id: '||V_sid);
 10   Dbms_output.put_line ('Sailor_name: '||Rec_sailors.sname);
 11   Dbms_output.put_line ('Sailor_rating: '||Rec_sailors.rating);
 12   Dbms_output.put_line ('Sailor_age: '||Rec_sailors.age);
 13  End;
 14  /
Enter value for sid: 22
old   2: V_sid Sailors.sid%type:=&sid;
new   2: V_sid Sailors.sid%type:=22;
Sailor_id: 22
Sailor_name: dustin
Sailor_rating: 9
Sailor_age: 45

PL/SQL procedure successfully completed.
```

```
SQL> Declare
  2  Cursor sail_cur
  3  IS
  4  Select * from Sailors;
  5  Rec_sail Sailors%rowtype;
  6  Begin
  7  Open sail_cur;
  8  Loop
  9  Fetch sail_cur into Rec_sail;
 10  Exit when sail_cur%NOTFOUND;
 11  Dbms_output.put_line (sail_cur%rowcount||','||Rec_sail.sname||','||Rec_sa
il.rating||','||Rec_sail.age);
 12  End Loop;
 13  Close sail_cur;
 14  End;
 15  /
1,dustin,9,45
2,brutus,1,33
3,lubber,8,56
4,ady,8,26
5,rusty,10,35
6,horatio,7,35
7,zobra,10,16
8,horatio,9,36
9,art,3,26
10,bob,3,64
11,Jack,9,
12,John,7,
13,John,7,

PL/SQL procedure successfully completed.
```

```
SQL> Declare
  2  Cursor boat_cur
  3  IS
  4  Select bname,color from Boats;
  5  Begin
  6  For Rec_boat in boat_cur
  7  Loop
  8  Dbms_output.put_line  (boat_cur%rowcount||','||Rec_boat.bname||','||Rec_b
oat.color);
  9   End Loop;
 10  End;
 11  /
1,interlake,blue
2,interlake,red
3,Clipper,green
4,marine,red

PL/SQL procedure successfully completed.
```

```
SQL> Begin
  2  Update Sailors
  3  Set rating=rating+1
  4  Where age>40;
  5  If SQL%FOUND
  6  THEN
  7  Dbms_output.put_line  (SQL%ROWCOUNT|| ' rows updated');
  8  ELSE
  9  Dbms_output.put_line  ('Age group not found');
 10  End If;
 11  End;
 12  /
3 rows updated

PL/SQL procedure successfully completed.
```

```
SQL> Declare
  2  Rec_sail Sailors%rowtype;
  3  Begin
  4  SELECT * into Rec_sail
  5  FROM Sailors
  6  WHERE sid=&sid;
  7  If SQL%FOUND
  8  THEN
  9  Dbms_output.put_line  (Rec_sail.sname||','||Rec_sail.age);
 10  ELSE
 11  Dbms_output.put_line  ('Sailor not found');
 12  End If;
 13  End;
 14  /
Enter value for sid: 22
old   6: WHERE sid=&sid;
new   6: WHERE sid=22;
dustin,45

PL/SQL procedure successfully completed.
```

```
SQL> Declare
  2  Cursor sail_reserves(p_sid number)
  3  IS
  4  Select s.sname,r.bid,r.day
  5  from Sailors S, Reserves r
  6  where s.sid=r.sid
  7  and r.sid = p_sid;
  8  Rec_sail sail_reserves%rowtype;
  9  Begin
 10  Open sail_reserves(22);
 11  Loop
 12  Fetch sail_reserves into Rec_sail;
 13  Exit when sail_reserves %NOTFOUND;
 14  Dbms_output.put_line  (Rec_sail.sname||','||Rec_sail.bid||','||Rec_sail.day);
 15  End Loop;
 16  Close sail_reserves;
 17  Open sail_reserves(64);
 18  Loop
 19  Fetch sail_reserves into Rec_sail;
 20  Exit when sail_reserves %NOTFOUND;
 21  Dbms_output.put_line  (Rec_sail.sname||','||Rec_sail.bid||','||Rec_sail.day);
 22  End Loop;
 23  Close sail_reserves;
 24  End;
 25  /
dustin,101,10-OCT-98
dustin,102,10-OCT-98
dustin,103,10-AUG-98
dustin,104,10-JUL-98
horatio,101,09-MAY-98
horatio,102,09-AUG-98

PL/SQL procedure successfully completed.
```

```
SQL> Create or replace function pi
  2   return number
  3   is
  4   begin
  5   return round(22/7,2);
  6   end;
  7  /

Function created.

SQL> declare
  2  v_no number;
  3  begin
  4  v_no:=pi;
  5  dbms_output.put_line(v_no);
  6  End;
  7  /
3.14

PL/SQL procedure successfully completed.

SQL> begin
  2  dbms_output.put_line(pi);
  3  end;
  4  /
3.14

PL/SQL procedure successfully completed.
```

```
SQL> Create or replace function pi_new
  2   return number
  3   is v_pi number(3,2);
  4   begin
  5  v_pi:=22/7;
  6   return v_pi;
  7   end;
  8  /

Function created.
```

```
SQL> Create or replace function num_add(p_no1 number,p_no2 number)
  2  return number
  3  is
  4  v_add number(4);
  5  begin
  6  v_add:=p_no1+p_no2;
  7  return v_add;
  8  end;
  9  /

Function created.

SQL> select num_add(3,4) from dual
  2  /

NUM_ADD(3,4)
------------
           7
```

```
SQL> Create or replace function num_type(p_no number)
  2  return varchar2
  3  is
  4  begin
  5  if p_no>0
  6  then return 'positive';
  7  elsif p_no<0
  8  then return 'negative';
  9  else return 'zero';
 10  end if;
 11  end;
 12  /

Function created.

SQL> select num_type(9) from dual
  2  /

NUM_TYPE(9)
--------------------------------------------------------
positive
```

```
SQL> create or replace function samplefunc(a number, b number, c out number)
  2  return number
  3  is begin
  4  c:=a-b;
  5  return a+b;
  6  end;
  7  /

Function created.

SQL>
SQL> Declare
  2  x  number;
  3  z number;
  4  Begin
  5  z:=samplefunc(20,10,x);
  6  dbms_output.put_line(x||' '||z);
  7  end;
  8  /
10 30

PL/SQL procedure successfully completed.
```

```
SQL> Create or replace procedure edit_rating(p_sid number, p_rating number) is
  2  Begin
  3  update Sailors set rating=p_rating where sid=p_sid;
  4  if SQL%found then
  5  dbms_output.put_line(SQl%rowcount||' '||'rows updated');
  6  else
  7  dbms_output.put_line('no records found');
  8  end if;
  9  End;
 10  /

Procedure created.

SQL>
SQL> execute edit_rating(22,10);
1 rows updated

PL/SQL procedure successfully completed.
```

Exercise:

1. Write a program to demonstrate use of implicit cursor for a DML operation on Boats table.
A) Begin
insert into boats
values (200, ‘Interlake’, ‘green’);
If SQL%FOUND
THEN
Dbms_output.put_line (SQL%ROWCOUNT|| ' rows inserted’);
ELSE
Dbms_output.put_line ('Insertion failed due to primary key constraint');
End If;
End;
/

```
SQL> Begin
  2  insert into boats
  3  values (201, 'Interlake', 'blue');
  4  If SQL%FOUND
  5  THEN
  6  Dbms_output.put_line  (SQL%ROWCOUNT|| 'rows inserted');
  7  ELSE
  8  Dbms_output.put_line  ('Insertion failed due to primary key constraint');
  9  End If;
 10  End;
 11  /
1rows inserted

PL/SQL procedure successfully completed.
```

2. Write a program to create a stored procedure which takes sid as a parameter and increments the rating of that sailor by 1 if the sailor’s age is greater than 40
A) Create or replace procedure inc_rating(p_sid number) is

```
Begin
update Sailors set rating = rating + 1 where sid = p_sid and age >40;
if SQL % found then
dbms_output.put_line(SQL % rowcount || ' ' || 'rows updated');
else
dbms_output.put_line('no records found');
end if;
End;
/
```

execute inc_rating(22);

```
SQL> Create or replace procedure inc_rating(p_sid number) is
  2  Begin
  3  update Sailors set rating = rating + 1 where sid = p_sid and age > 40;
  4  if SQL % found then
  5  dbms_output.put_line(SQL % rowcount || ' ' || 'rows updated');
  6  else
  7  dbms_output.put_line('no records found');
  8  end if;
  9  End;
 10  /

Procedure created.

SQL>
SQL> execute inc_rating(22);
1 rows updated

PL/SQL procedure successfully completed.
```

3. Write a program to display details of all sailors from sailors table (Explicit Cursor)

```
A) Declare
Cursor sail_cur
IS
Select * from Sailors;
Rec_sailSailors%rowtype;
Begin
Open sail_cur;
Loop
Fetch sail_cur into Rec_sail;
Exit when sail_cur%NOTFOUND;
Dbms_output.put_line
(sail_cur%rowcount||','||Rec_sail.sname||','||Rec_sail.rating||','||Rec_sail.age);
End Loop;
Close sail_cur;
End;
/
```

```
SQL> Declare
  2  Cursor sail_cur
  3  IS
  4  Select * from Sailors;
  5  Rec_sail Sailors%rowtype;
  6  Begin
  7  Open sail_cur;
  8  Loop
  9  Fetch sail_cur into Rec_sail;
 10  Exit when sail_cur%NOTFOUND;
 11  Dbms_output.put_line (sail_cur%rowcount||','||Rec_sail.sname||','||Rec_sail.rating||','||Rec_sail.age);
 12  End Loop;
 13  Close sail_cur;
 14  End;
 15  /
1,dustin,11,45
2,brutus,1,33
3,lubber,9,56
4,ady,8,26
5,rusty,10,35
6,horatio,7,35
7,zobra,10,16
8,horatio,9,36
9,art,3,26
10,bob,4,64
11,Jack,9,
12,John,7,
13,John,7,

PL/SQL procedure successfully completed.
```

4. Write a program to create a procedure to edit the rating of a given sailor with the given rating.(DML- Update operation)

A) Create or replace procedure edit_rating(p_sid number, p_rating number) is
Begin
update Sailors set rating=p_rating where sid=p_sid;
if SQL%found then
dbms_output.put_line(SQl%rowcount||' '||'rows updated');
else
dbms_output.put_line('no records found');
end if;
End;
/
execute edit_rating(22,10);

```
SQL> Create or replace procedure edit_rating(p_sid number, p_rating number) is
  2  Begin
  3  update Sailors set rating=p_rating where sid=p_sid;
  4  if SQL%found then
  5  dbms_output.put_line(SQl%rowcount||' '||'rows updated');
  6  else
  7  dbms_output.put_line('no records found');
  8  end if;
  9  End;
 10  /

Procedure created.

SQL> execute edit_rating(22,10);
1 rows updated

PL/SQL procedure successfully completed.
```

Result:

Execution of PL SQL select and cursors has been done successfully.

Viva Questions

1.What is the difference between a function and a stored procedure?
Ans)Function returns a value but a stored procedure does not return a value.

2. Write the process for creating and using explicit cursors in PLSQL
Ans)Explicit Cursor has 4 parts:
Cursor declaration - - in the declaration section
Opening Cursor - - - in the begin section
Reading Cursor- - - in the begin section
Closing Cursor- - - in the begin section
Syntax:
Cursor declaration: CURSOR <cursor_name>
IS
SELECT STATEMENT;
Example: CURSOR sail_cur
IS
Select * from Sailors;

Opening Cursor: OPEN <cursor_name>;
Example: OPEN sail_cur;

Reading Cursor: FETCH <cursor_name>
INTO <variable(s)>;

Example: FETCH sail_cur
INTO rec_sail;
(rec_sailsail_cur%rowtype)

Closing Cursor: CLOSE <cursor_name>;
Example: CLOSE sail_cur;

3.What is returned by the cursor attribute SQL%NOTFOUND?
Ans)It returns FALSE if the last fetch returned a row, or TRUE if the last fetch failed to return a row.

4.What is returned by the cursor attribute SQL%FOUND?
Ans)It returns true if row is fetched and false if row is not fetched.

5. What is the purpose of %type and %rowtype in PLSQL

Ans)The %TYPE attribute allow you to declare a constant, variable, or parameter to be of the same data type as previously declared variable, record, nested table, or database column.The %ROWTYPE attribute provides a record type that represents a row in a database table. The record can store an entire row of data selected from the table or fetched from a cursor or cursor variable.

6.What do you understand by PL/SQL cursors?
Ans)Cursor is a work area to store multiple records from the database table. In PLSQL whenever select statement returns more than one row, a cursor is used.

7.List the characteristics of PL/SQL?
Ans)PL/SQL is a procedural language. PL/SQL is a combination of SQL along with the procedural features of programming languages.
PLSQL block has 3 parts:
Declaration (optional) – for variable declaration
Executable part (mandatory) – for statements
Exception part (optional) – for runtime error handling
Valid statements in PLSQL:
Plsql select statement
All DML statements
TCL statements
SQL functions, operators
User defined functions, procedures and packages
Expressions
Directly not valid in PLSQL block:
DDL statements
DCL statements

8.What are COMMIT, ROLLBACK, and SAVEPOINT?
Ans)COMMIT – Used to permanently save any transaction into Database (DML statements).
ROLLBACK- restores the database to last committed state. It is also use with savepoint command to jump to a savepoint in a transaction.
SAVEPOINT- used to temporarily save a transaction so that you can rollback to that point whenever necessary.

9.What is a Transaction?
Ans)In a database management system, a transaction is a single unit of logic or work, sometimes made up of multiple operations. Any logical calculation done in a consistent mode in a database is known as a transaction.

10. When do we go for normalizing a relation?
Ans)There are three main reasons to normalize a database. The first is to minimize duplicate data, the second is to minimize or avoid data modification issues, and the third is to simplify queries.

PL SQL – EXCEPTION HANDLING

Aim

To execute PL-SQL exception handling.

Examples

```
SQL> DECLARE
  2  rec_sail Sailors%rowtype;
  3  BEGIN
  4  select * into rec_sail
  5  from Sailors
  6  where sid=&sid;
  7  dbms_output.put_line(rec_sail.sid||' '||rec_sail.sname);
  8  EXCEPTION
  9  WHEN too_many_rows THEN
 10  dbms_output.put_line('Use cursors');
 11  WHEN no_data_found THEN
 12  dbms_output.put_line('No Sailors found');
 13  WHEN others THEN dbms_output.put_line(SQLERRM);
 14  end;
 15  /
Enter value for sid: 22
old   6: where sid=&sid;
new   6: where sid=22;
22 dustin

PL/SQL procedure successfully completed.
```

```
SQL> DECLARE
  2  rec_sail Sailors%rowtype;
  3  v_sid number(3);
  4  BEGIN
  5  v_sid:=&sid;
  6  select * into rec_sail
  7  from Sailors
  8  where sid=v_sid;
  9  dbms_output.put_line(rec_sail.sid||' '||rec_sail.sname);
 10  EXCEPTION
 11  WHEN too_many_rows THEN
 12  dbms_output.put_line('User cursors');
 13  WHEN no_data_found THEN
 14  dbms_output.put_line('No Sailors found');
 15  WHEN value_error THEN
 16  dbms_output.put_line('datatype/width mismatch');
 17  WHEN others THEN dbms_output.put_line(SQLERRM);
 18  END;
 19  /
Enter value for sid: 22
old   5: v_sid:=&sid;
new   5: v_sid:=22;
22 dustin

PL/SQL procedure successfully completed.
```

```
SQL> DECLARE
  2  e_positive EXCEPTION;
  3  v_num NUMBER;
  4  BEGIN
  5  v_num:=&number;
  6  IF v_num<0
  7  THEN
  8  RAISE e_positive;
  9  ELSE
 10  dbms_output.put_line('Square root'||' '||sqrt(v_num));
 11  END IF;
 12  dbms_output.put_line('job over');
 13  EXCEPTION
 14  WHEN e_positive THEN
 15  dbms_output.put_line('Imaginary value');
 16  WHEN value_error THEN
 17  dbms_output.put_line('datatype/width mismatch');
 18  WHEN others THEN dbms_output.put_line(SQLERRM);
 19  END;
 20  /
Enter value for number: 25
old   5: v_num:=&number;
new   5: v_num:=25;
Square root 5
job over

PL/SQL procedure successfully completed.

SQL> /
Enter value for number: 3
old   5: v_num:=&number;
new   5: v_num:=3;
Square root 1.732050807568877293527446341505872366 94
job over

PL/SQL procedure successfully completed.
```

```
SQL> CREATE OR REPLACE TRIGGER message
  2  AFTER update on Boats
  3  begin
  4  dbms_output.put_line('Update success');
  5  end;
  6  /

Trigger created.

SQL>
SQL>
SQL> UPDATE Boats set bname='Marine' where bid=105;
Update success

0 rows updated.
```

```
SQL> CREATE OR REPLACE TRIGGER message1
  2  AFTER UPDATE OR INSERT OR DELETE on Sailors
  3  begin
  4  IF INSERTING then
  5  dbms_output.put_line('Insert Success');
  6  ELSIF UPDATING then
  7  dbms_output.put_line('Update Success');
  8  ELSE dbms_output.put_line('Delete Success');
  9  END IF;
 10  END;
 11  /

Trigger created.

SQL>
SQL> update Sailors set rating=7 where sid=22;
Update Success

1 row updated.

SQL> insert into Sailors values(36,'John',7,45);
Insert Success

1 row created.

SQL> delete from Sailors where sid=36;
Delete Success

1 row deleted.
```

```
SQL> CREATE OR REPLACE TRIGGER sample
  2  BEFORE DELETE ON Sailors
  3  FOR EACH ROW
  4  BEGIN
  5  IF :old.rating=7 THEN
  6  raise_application_error(-20000,'Insufficient Privileges');
  7  END IF;
  8  END;
  9  /

Trigger created.

SQL>
SQL> Delete from Sailors where rating=7;
Delete from Sailors where rating=7
            *
ERROR at line 1:
ORA-20000: Insufficient Privileges
ORA-06512: at "IT19737120.SAMPLE", line 3
ORA-04088: error during execution of trigger 'IT19737120.SAMPLE'
```

```
SQL> CREATE OR REPLACE TRIGGER sample
  2  before delete on Sailors
  3  for each row
  4  when(old.rating=7)
  5  begin
  6  raise_application_error(-20002,'No privileges');
  7  end;
  8  /

Trigger created.

SQL>
SQL> Delete from Sailors where rating=7;
Delete from Sailors where rating=7
            *
ERROR at line 1:
ORA-20002: No privileges
ORA-06512: at "IT19737120.SAMPLE", line 2
ORA-04088: error during execution of trigger 'IT19737120.SAMPLE'
```

```
SQL> CREATE OR REPLACE TRIGGER case_sample
  2  BEFORE INSERT OR UPDATE ON Sailors
  3  FOR EACH ROW
  4  BEGIN
  5  if trim(:new.sname)!=upper(trim(:new.sname))
  6  then
  7  raise_application_error(-20001,'Lower case not allowed');
  8  end if;
  9  end;
 10  /

Trigger created.

SQL> insert into Sailors values(35,'John',7,45);  // before trigger creation
  2  insert into Sailors values(35,'JOHN',7,45);
insert into Sailors values(35,'John',7,45);  // before trigger creation
                                           *
ERROR at line 1:
ORA-00911: invalid character


SQL> update Sailors set sname='Dustin' where sid=22;
update Sailors set sname='Dustin' where sid=22
       *
ERROR at line 1:
ORA-20001: Lower case not allowed
ORA-06512: at "IT19737120.CASE_SAMPLE", line 4
ORA-04088: error during execution of trigger 'IT19737120.CASE_SAMPLE'
```

```
SQL> CREATE OR REPLACE VIEW reserve_view AS
  2  SELECT s.sid,s.sname,r.bid,b.bname,r.day
  3  FROM Sailors s,Reserves R, Boats b
  4  WHERE s.sid=r.sid
  5  And r.bid=b.bid;

View created.

SQL> /

View created.

SQL>
SQL> insert into reserve_view values(50,'JOHN',105,’Cruise’,'10-OCT-98');
insert into reserve_view values(50,'JOHN',105,'Cruise','10-OCT-98')
*
ERROR at line 1:
ORA-01779: cannot modify a column which maps to a non key-preserved table


SQL> insert into reserve_view values(29,'BRUTUS',101,’Interlake’,'11-OCT-98');
insert into reserve_view values(29,'BRUTUS',101,'Interlake','11-OCT-98')
*
ERROR at line 1:
ORA-01779: cannot modify a column which maps to a non key-preserved table
```

```
SQL>  CREATE OR REPLACE TRIGGER sample2
  2  INSTEAD OF INSERT ON reserve_view
  3  FOR EACH ROW
  4  DECLARE
  5  rowcnt number;
  6  BEGIN
  7  SELECT COUNT(*) INTO rowcnt FROM Sailors WHERE sid=:new.sid;
  8  IF rowcnt=0 THEN
  9  INSERT INTO Sailors(sid, sname) values (:new.sid,:new.sname);
 10  ELSE
 11  UPDATE Sailors SET Sailors.sname=:new.sname
 12  WHERE Sailors.sid=:new.sid;
 13  END IF;
 14  SELECT COUNT(*) INTO rowcnt FROM Boats WHERE bid=:new.bid;
 15  IF rowcnt=0 THEN
 16  INSERT INTO Boats(bid,bname) values (:new.bid,:new.bname);
 17  ELSE
 18  UPDATE Boats SET Boats.bname=:new.bname
 19  WHERE Boats.bid=:new.bid;
 20  END IF;
 21  SELECT COUNT(*) INTO rowcnt FROM Reserves WHERE sid=:new.sid
 22  AND bid=:new.bid;
 23  IF rowcnt=0 THEN
 24  INSERT INTO Reserves(sid,bid,day) values (:new.sid,:new.bid,:new.day);
 25  ELSE
 26  UPDATE Reserves SET Reserves.day=:new.day
 27  WHERE Reserves.sid=:new.sid AND Reserves.bid=:new.bid;
 28  END IF;
 29  END;
 30  /

Trigger created.

SQL>
SQL> insert into reserve_view values(50,'JOHN',105,’Cruise’,'10-OCT-98');
Insert Success

1 row created.

SQL> insert into reserve_view values(29,'BRUTUS',101,’Interlake’,'11-OCT-98');
Update Success
Update success
SQL> CREATE OR REPLACE TRIGGER tab1_trig
  2  AFTER insert ON sailors
  3  DECLARE
  4  PRAGMA AUTONOMOUS_TRANSACTION;
  5  BEGIN
  6  INSERT INTO log VALUES (SYSDATE, 'Insert on Sailors');
  7  COMMIT; -- only allowed in autonomous triggers
  8  END;
  9  /

Warning: Trigger created with compilation errors.
```

Exercises

1. Create a trigger to print the total number of records in the Reserves table whenever a new row is inserted or an existing row is deleted from the table.

Ans)

```
SQL> CREATE OR REPLACE TRIGGER exercise1
  2  AFTER INSERT OR DELETE on reserves
  3  DECLARE
  4  recordcount number;
  5  BEGIN
  6  select count(*) into recordcount from reserves;
  7  dbms_output.put_line('Total number of rows are'||':'||recordcount);
  8  END;
  9  /

Trigger created.

SQL> insert into reserves values(95,103,'05-SEP-98');
Total number of rows are:6

1 row created.

SQL> select * from reserves;

       SID        BID DAY
---------- ---------- ---------
        31        102 11-NOV-98
        22        101 10-OCT-98
        22        103 08-OCT-98
        31        103 06-NOV-98
        64        101 05-SEP-98
        95        103 05-SEP-98

6 rows selected.

SQL> delete from reserves where sid=95;
Total number of rows are:5

1 row deleted.
```

2. Write a trigger that restricts the boats table from having duplicates values or null values in the column bname.

Ans)

```
SQL> create or replace trigger exercise2
  2  BEFORE INSERT OR UPDATE OF bname ON BOATS
  3  for each row
  4  begin
  5  if((trim(:new.bname)=trim(:old.bname)) OR (:new.bname IS NULL))
  6  then raise_application_error(-20000,'DUPLICATES OR NULL VALUES ARE NOT ALLOWED!!');
  7  END IF;
  8  END;
  9  /

Trigger created.

SQL> insert into boats values(108,'Interlake','red');
insert into boats values(108,'Interlake','red')
            *
ERROR at line 1:
ORA-20000: DUPLICATES AND NULL VALUES NOT ALLOWED!!
```

```
SQL> insert into boats(bid,color) values(108,'red');
insert into boats(bid,color) values(108,'red')
            *
ERROR at line 1:
ORA-20000: DUPLICATES OR NULL VALUES ARE NOT ALLOWED!!
```

Results

PL SQL exception handling has been done successfully.

Viva Questions

1) PL/SQL packages usually have two parts. What are these two parts?
Ans) The two parts of package are package specification & package body. Package Specification contains declarations that are global to the packages and local to the schema .Package Body contains actual procedures and local declaration of the procedures and cursor declarations.
2) Which command(s) are used for creating PL/SQL packages?
Ans) The CREATE PACKAGE BODY Statement is used for creating the package body.
3) Explain the process of creating user defined exceptions
Ans) PL/SQL allows you to define your own exceptions according to the need of your program. A user-defined exception must be declared and then raised explicitly, using either a RAISE statement or the procedure DBMS_STANDARD.RAISE_APPLICATION_ERROR.
BEGIN
<execution block>
EXCEPTION
WHEN <exceptionl_name>
THEN
<Exception handling code for the “exception 1 _name’’ >
WHEN OTHERS
THEN
<Default exception handling code for all exceptions >
END;

4) List any four in-built exceptions and when they are raised.
Ans) a) ZERO_DIVIDE- Dividing a number by zero
b) TOO_MANY_ROWS - When a 'SELECT' statement with INTO clause returns more than one row
c) VALUE_ERROR - Arithmetic or size constraint error (eg: assigning a value to a variable that is larger than the variable size)
d) NO_DATA_FOUND - When a 'SELECT' statement that contains INTO clause fetches no rows.
5) What is the purpose of using OTHERS exceptions?
Ans) If none of the 'WHEN' clause is present for the exception which has been raised, then PL/SQL engine will execute the 'WHEN OTHERS' part (if present). This is common for all the exceptions.

6) Write the difference between Triggers and Constraints?
Ans) Trigger affects only those rows, which are added after it is enabled. Constraints affects all the rows i.e. the ones that existed before and the ones that were newly added. A constraint is responsible only for maintaining the integrity of the database
7)Discuss the difference in execution of triggers and stored procedures?
Ans) Basic - trigger is a stored procedure that runs automatically when various events happen (eg update, insert, delete) whereas Stored procedures are a piece of the code written in PL/SQL to do some specific task.
Running Methodology - Trigger can execute automatically based on the events whereas stored procedure can be invoked explicitly by the user
Parameter - Trigger can not take input as parameter but a stored procedure can take input as a parameter
8) When do we use triggers?
Ans) Triggers can be defined to run instead of or after DML (Data Manipulation Language) actions such as INSERT, UPDATE, and DELETE. Triggers help the database designer ensure certain actions, such as maintaining an audit file, are completed regardless of which program or user makes changes to the data.
9) What is the mutating table and constraining table?
Ans) A mutating table is the one, which is modified using a DML statement or a table with defined triggers. A constraining table is the one, which is being read for a referential integrity constraint.
10) What are actual parameters and formal parameters?
Ans) The parameter used in function definition statements which contain data type on its time of declaration is called formal parameter. ... Actual Parameters are the parameters which are in the calling subprogram. Formal Parameters are the parameters which are in called subprogram