Experiment No. 6

Environment: Microsoft Windows
Tools/ Language: Oracle/SQL

Objective: To implement concept of various functions in SQL.

Theory & Concepts:

Oracle functions:
Functions make the basic query block more powerful and are used to manipulate data values. Functions accept one or more arguments and return one value. An argument is a user supplied constant, variable or column reference, which can be passed to a function in the following format:
function _name (argument1, argument2 ..)
Functions can be used to

- ➢ Perform calculations on data
- ➢ Modify individual data item
- ➢ Manipulate output for group of rows
- ➢ Alter date formats for display
- ➢ Convert column data types

Single Row Functions:

- ➢ Act on each row returned in a query
- ➢ Return one result per row
- ➢ Expect one or more user arguments
- ➢ May be nested

Single row functions can be of following types:
Character functions, Number Functions, Date functions and conversion functions.

ORACLE TABLE

DUAL: Dual table is owned by SYS. SYS owns the data dictionary; in DUAL is a part of data dictionary. Dual is a small Oracle worktable which consists of only one row and one column and contains the value *x* in that column. Besides arithmetic calculations, it also supports date retrieval and its formatting.

```
Select 3.14*3*3 from DUAL;
28.26

Select 4/2 from DUAL;
2

DESC DUAL;
Name                                Null?         Type
---------------------------------                 ------------------
 DUMMY                                            VARCHAR2(1)
Select * from DUAL
```

```
D
--
X
```

```
Select sysdate from DUAL
SYSDATE
------
17-FEB-14
```

(a) **Character Functions:** accept character data as input and can return both character and number values.
LOWER (col/value)
UPPER (col/value)
INITCAP (col/value)
SUBSTR (col/value, position, n)
INSTR (col/value, ‘string’)
ASCII (character)
CHR (number)

(b) **Number Functions:**
ROUND (col/val, n)
TRUNC (col/val, n)
CEIL (col/val)
FLOOR (col/val)
POWER (col/val, n)
SQRT (col/val)
EXP (n)
ABS (col/val)
MOD (value1, value2)

(c) **Date Functions:** Date functions are used to manipulate and extract values from the date column of a table.
SYSDATE
ADD_MONTHS (date, count)
LAST_DAY (date)
MONTHS_BETWEEN (date2, date1)
NEXT_DAY (date, ‘day’)

(d) **Conversion Functions:** Converts one data type into another.
TO_CHAR (input, format): Converts date or number into character string.
TO_DATE (date, format): Converts any date format to default format (dd-mon-yy).
TO_NUMBER (col/value): Converts a string into a number.

<u>ORACLE FUNCTION EXAMPLES</u>

<u>NUMERIC FUNCTIONS</u>

1. **ROUND**

```
Select ROUND(SUM(GPA)/COUNT(GPA)) as "Round Average"
from student
```

```
Round Average
-------------
        4
```

Round(*n, m)* return rounded to *m* places to the right of a decimal point. If *m* is omitted then *n* is rounded to 0 places.

```
Select ROUND(15.81,1), round(158.158),round(158.158,-2) from
DUAL
```

```
ROUND(15.81,1)
---------
15.8
```

2. **ABSOLUTE**: This function return absolute value of numbers

```
Select ABS(-15) from DUAL
```

```
 ABS(-15)
-------
  15
```

3. **POWER:** Return *m* raise to *n* where *n* must be an integer.

```
Select POWER(7,3) from DUAL
```

```
POWER(7,3)
-------
343
```

4. **SQUARE ROOT:** Return the square root of *n.* If n<0 then NULL.

```
Select SQRT(25) from DUAL
```

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

```
select ROUND(sqrt(15),2) from DUAL
```

```
ROUND(SQRT(15),2)
-----------
    3.87
```

5. **EXPONENTIAL**

```
Select EXP(5) from DUAL
```

```
   EXP(5)
------
148.413159
```

6. **EXTRACT:** Returns a value extracted from a date or an interval. A date can be used only to extract year month or day while a time span with the time zone use to extract hours and minutes.

```
Select extract(month from sysdate)
from DUAL
```

```
EXTRACT(MONTHFROMSYSDATE)
------------------------------------
                    2
```

7. **MOD:** Return the remainder of first number divided by second number passed as a parameter

```
Select MOD(15,7) from DUAL
```

```
 MOD(15,7)
-------
       1
```

8. **TRUNCATE:** Returns a truncated number to a certain number of decimal places
 TRUNC(number, decimal places)

```
Select TRUNC(125.815,1) from dual
```

```
TRUNC(125.815,1)
-----------
         125.8
```

```
Select TRUNC(125.815,-2) from dual
```

```
TRUNC(125.815,-2)
-----------
            100
```

9. **FLOOR:** Returns the largest value that is equal to or less than the number.

```
Select floor(24.92), floor(3.1) from dual
```

```
FLOOR(24.92) FLOOR(3.1)
---------------
        24       3
```

10. **CEILING:** Return the largest value that is equal to or greater than the number.

```
Select ceil(24.92), ceil(3.1) from dual
```

```
CEIL(24.92)  CEIL(3.1)
---------------
       25       4
```

STRING FUNCTIONS:

1. **Lower:** Returns char, with letters in lower case.

```
Select LOWER(‘RAJeEv’) from dual
```

```
LOWER(
----
Rajeev
```

2. **UPPER:** Return char with letters in UPPER case

```
Select LOWER(‘rAJeEv’)
from dual
```

```
LOWER(
----
RAJEEV
```

3. **SUBSTRING:**
   SUBSTR(string, start_pos, length)

```
Select SUBSTR(‘Prateek’, 4, 3) from dual
```

```
SUB
---
tee
```

```
Select SUBSTR(‘Prateek’, 4) from dual
```

```
SUB
---
teek
```

4. **INSTR:** Returns the location of substring in the string
   INSTR(string, string2, start_position, *n*th appearance)

```
Select instr(‘Rahul Pradhan’, ‘a’, 2, 3)
from dual
```

```
INSTR('RAHULPRADHAN','A',2,3)
-------------------
                 12
```

```
Select instr(‘Rahul Pradhan’, ‘a’, 2)as INSTR
from dual
```

```
INSTR
------
  2
```

String: HTC launch a new phone on net
Search t.

```
Select instr('HTC launch a new phone on net', 't')
from dual
```

```
INSTR('HTCLAUNCHANEWPHONEONNET','T')
-----------------------
                     29
```

5. **TRANSLATE**

```
Select translate('HTC launch a new phone on net', 'net',
'web')as Translate from dual
```

```
TRANSLATE
-------------------
HTC lauwch a wew phowe ow web
```

6. **LENGTH**

```
Select length('Pratyush Mehrotra') as Length from dual
```

```
   LENGTH
-------
      17
```

7. **TO_CHAR(n,fmt):** Converts a number to a character

```
Select to_char(17475,'$099,999')as FMT from dual
```

```
FMT
------
 $017,475
```

```
Select TO_CHAR(sysdate, 'Month DD, YYYY') from dual
```

```
TO_CHAR(SYSDATE,'M
------------
February  17, 2014
```

```
Select TO_CHAR(date '2014-02-20', 'DD/MM/YYYY') from dual;
```

```
TO_CHAR(DA
-------
20/02/2014
```

8. **TO_DATE:**

```
Select to_date('23/02/1988', 'DD/MM/YYYY') from DUAL
```

```
TO_DATE('
------
23-FEB-88
```

9. **ADD_MONTHS:**

```
Select add_months(sysdate,4) from dual
```

```
ADD_MONTH
------
17-JUN-14
```

```
Select add_months(date '1994-09-20',4) from dual
```

```
ADD_MONTH
------
20-JAN-95
```

10. LAST_DAY

```
Select sysdate, last_day(sysdate) from dual
```

```
SYSDATE LAST_DAY(
------------
17-FEB-14 28-FEB-14
s
```

11. NEXT_DAY

```
Select sysdate, next_day(Sysdate, 'Monday')
from dual
```

```
SYSDATE    NEXT_DAY(
------      ------
17-FEB-14  24-FEB-14
```

```
Select sysdate, next_day(date '2014-09-20', 'sunday')
from dual
```

```
SYSDATE    NEXT_DAY(
------     ------
20-FEB-14  21-SEP-14
```

Practical Assignment - 6

Department: Computer Engineering & Applications

Course: B.Tech. (CSE)

Subject: Database Management System Lab (BCSC0802)

Year: 2nd **Semester:** 3rd

SQL Script for this Experiment

```
BEGIN
  FOR cur_rec IN (SELECT object_name, object_type
                  FROM user_objects
                 WHERE object_type IN
                        ('TABLE',
                         'VIEW',
                         'PACKAGE',
                         'PROCEDURE',
                         'FUNCTION',
                         'SEQUENCE'
                        ))
  LOOP
    BEGIN
      IF cur_rec.object_type = 'TABLE'
      THEN
        EXECUTE IMMEDIATE   'DROP '
                         || cur_rec.object_type
                         || ' "'
                         || cur_rec.object_name
                         || '" CASCADE CONSTRAINTS';
      ELSE
        EXECUTE IMMEDIATE   'DROP '
                         || cur_rec.object_type
                         || ' "'
                         || cur_rec.object_name
                         || '"';
      END IF;
    EXCEPTION
      WHEN OTHERS
      THEN
        DBMS_OUTPUT.put_line (  'FAILED: DROP '
                              || cur_rec.object_type
                              || ' "'
                              || cur_rec.object_name
                              || '"'
                             );
    END;
END LOOP;
END;
/
commit;
```

```
drop  table  College;
drop  table  Student;
drop table  Apply;
create table College(cName varchar2(10) primary key, state
varchar2(10), enrollment int);
create table Student(sID int primary key, sName varchar2(10), GPA
real, sizeHS int);
create table Apply(sID int, cName varchar2(10), major varchar2(20),
decision char(1), primary key(sID, major, cName), constraint sID_fk
Foreign key(sID) references Student, constraint cName_fk Foreign
key(cName) references College);

delete from Student;
delete from College;
delete from Apply;

insert into Student values (123, 'Amy', 3.9, 1000);
insert into Student values (234, 'Bob', 3.6, 1500);
insert into Student values (345, 'Craig', 3.5, 500);
insert into Student values (456, 'Doris', 3.9, 1000);
insert into Student values (567, 'Edward', 2.9, 2000);
insert into Student values (678, 'Fay', 3.8, 200);
insert into Student values (789, 'Gary', 3.4, 800);
insert into Student values (987, 'Helen', 3.7, 800);
insert into Student values (876, 'Irene', 3.9, 400);
insert into Student values (765, 'Jay', 2.9, 1500);
insert into Student values (654, 'Amy', 3.9, 1000);
insert into Student values (543, 'Craig', 3.4, 2000);
insert into College values ('Stanford', 'CA', 15000);
insert into College values ('Berkeley', 'CA', 36000);
insert into College values ('MIT', 'MA', 10000);
insert into College values ('Cornell', 'NY', 21000);
insert into College values ('Harvard', 'MA', 50040);
insert into Apply values (123, 'Stanford', 'CS', 'Y');
insert into Apply values (123, 'Stanford', 'EE', 'N');
insert into Apply values (123, 'Berkeley', 'CS', 'Y');
insert into Apply values (123, 'Cornell', 'EE', 'Y');
insert into Apply values (234, 'Berkeley', 'biology', 'N');
insert into Apply values (345, 'MIT', 'bioengineering', 'Y');
insert into Apply values (345, 'Cornell', 'bioengineering', 'N');
insert into Apply values (345, 'Cornell', 'CS', 'Y');
insert into Apply values (345, 'Cornell', 'EE', 'N');
insert into Apply values (678, 'Stanford', 'history', 'Y');
insert into Apply values (987, 'Stanford', 'CS', 'Y');
insert into Apply values (987, 'Berkeley', 'CS', 'Y');
insert into Apply values (876, 'Stanford', 'CS', 'N');
insert into Apply values (876, 'MIT', 'biology', 'Y');
insert into Apply values (876, 'MIT', 'marine biology', 'N');
insert into Apply values (765, 'Stanford', 'history', 'Y');
insert into Apply values (765, 'Cornell', 'history', 'N');
insert into Apply values (765, 'Cornell', 'psychology', 'Y');
insert into Apply values (543, 'MIT', 'CS', 'N');
commit;
```

Student

sID	sName	GPA	sizeHS	DoB
123	Amy	3.9	1000	26-JUN-96
234	Bob	3.6	1500	7-Apr-95
345	Craig	3.5	500	4-Feb-95
456	Doris	3.9	1000	24-Jul-97
567	Edward	2.9	2000	21-Dec-96
678	Fay	3.8	200	27-Aug-96
789	Gary	3.4	800	8-Oct-96
987	Helen	3.7	800	27-Mar-97
876	Irene	3.9	400	7-Mar-96
765	Jay	2.9	1500	8-Aug-98
654	Amy	3.9	1000	26-May-96
543	Craig	3.4	2000	27-Aug-98

College

cName	state	enrollment
Stanford	CA	15000
Berkeley	CA	36000
MIT	MA	10000
Cornell	NY	21000
Harvard	MA	50040

Apply

sID	cName	major	decision
123	Stanford	CS	Y
123	Stanford	EE	N
123	Berkeley	CS	Y
123	Cornell	EE	Y
234	Berkeley	biology	N
345	MIT	bioengineering	Y
345	Cornell	bioengineering	N
345	Cornell	CS	Y
345	Cornell	EE	N
678	Stanford	history	Y
987	Stanford	CS	Y
987	Berkeley	CS	Y
876	Stanford	CS	N
876	MIT	biology	Y
876	MIT	marine biology	N
765	Stanford	history	Y
765	Cornell	history	N
765	Cornell	psychology	Y
543	MIT	CS	N

Write SQL queries for the following:

Q1. Create a new column DoB in Student table. (Datatype will be *date*)

Q2. Insert DoB for each Student in corresponding table using above ***instance*** of Student table.

Q3. Find average of GPA round off to 2 decimal places.

Q4. Find year of DoB of Student having less than 1000.

Q5. Compute Age of each student. (Hint: *take difference between year of sysdate and Student's DoB*)

Q6. Display name of *all* Students in uppercase and name of college they applied in lower case.

Q7. Find fourth alphabet of each student. (Hint: *use* substring)

Q8. Find sID and sName of student whose sName has string length greater than 3.

Q9. Find floor, ceiling and truncate (to one decimal place) value of average GPA.

Q10. Display details of all students whose sID is *even.*

Q11. Compute Square Root of 900 and 24^7.

Q12. Consider the string "Peter Piper picked a peck of pickled peppers. A peck of pickled peppers Peter Piper picked. If Peter Piper picked a peck of pickled peppers, Where the peck of pickled peppers Peter Piper picked?"
Find 6th occurrence of string 'pick'. (Hint: *use* INSTR)

Q13. Consider String ‘Satya Nadella’ replace this using the key (Hint: *use* translate)

a	d	e	l	N	S	t	y
1	2	3	4	5	6	7	8

Q14. Display sID, sname and DoB in this format ‘February 26, 2014’
Q15. Convert the text ’26/02/2014’ to date.
Q16. Compute on which date is next Saturday and last day of this month?

Exercise

Q1. Display sID, sname and DoB in this format ‘26th Feburary, 2014’
Q2. Display sID, sname and DoB in this format ‘26/02/2014’
Q3. Add 5 months to DoB of Edward?
Q4. Display last day of DoB of Amy?
Q5. Display next Sunday of DoB of Doris?

Pre Experiment Questions

1. How to do arithmetic calculation in SQL?
2. How to store date in SQL?

Post Experiment Questions

1. When to use User created table or to use DUAL?
2. How can we decide whether to use truncate or round?
3. Which function is use to convert text to date?