|  |  |  |
| --- | --- | --- |
| **Ex. No. 6** | **S Q L F U N C T I O N S** | **Date :** |

SQL functions are of two types

# Single row functions or scalar functions

* + Returns only one value for every row queried in the table
  + Can be used in Select clause and where clause
  + It can be broadly classified into 5 categories
    - Date Functions
    - Character Functions
    - Conversion functions
    - Numeric functions
    - Miscellaneous functions

# Group functions or multiple-row functions

**Explore the following functions and write the output**

**Note : The exercises that follow mostly uses system table ‘dual’. It is a table which is automatically created by Oracle along with the data dictionary. Dual table has one column defined to be of varchar2 type and contains only one row with value ‘x’.**

1. **Date Functions**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Functions** | **Value Returned** | **Input** | | | **Output** |
| to\_date(str,’format’) | Converts the string ina given format into Oracle date. | Select to\_date(’10-02-09’,’dd-mm-yy’) from dual; | | |  |
| to\_char(date,’format’) | Reformats date according to format | Select to\_char(sysdate,’dy yyyy’) from dual; | dd | mon |  |
| months\_between(d1,d2) | No. of months between two dates | Select months\_between(sysdate, to\_date(’10-10-07’,’dd-mm-yy’) )  from dual; | | |  |

# Character Functions

|  |  |  |  |
| --- | --- | --- | --- |
| **Functions** | **Value Returned** | **Input** | **Output** |
| initcap(char) | First letter of each word capitalized | Select initcap(‘jesus christ’) from dual; |  |
| lower(char) | Lower case | Select lower(‘DIED’) from dual; |  |
| upper(char) | Upper case | Select upper(‘for Us’) from dual; |  |
| ltrim(char, set) | Initial characters removed up to the character not in set. | Select ltrim(‘lordourgod’,’lord’) from dual; |  |
| rtrim(char, set) | Final characters removed after the last character not in set. | Select rtrim(‘godlovesyou’,’you’) from dual; |  |
| replace(char, search, repl) | Replace ‘search’ string by ‘repl’ string in ‘char’. | Select replace(‘jack and jue’,’j’,’bl’) from dual; |  |
| substr(char, m, n) | Substring of ‘char’ at ‘m’ of size ‘n’ char long. | Select  substr(‘wages of sin is death’,10,3) from dual; |  |

# Numeric Functions

|  |  |  |  |
| --- | --- | --- | --- |
| **Functions** | **Value Returned** | **Input** | **Output** |
| Abs(n) | Absolute value of n | Select abs(-15) from dual; |  |
| Ceil(n) | Smallest int >= n | Select ceil(33.645) from dual; |  |
| Exp(n) | en | Select exp(2) from dual; |  |
| Floor(n) | Largest int <= n | Select floor(100.2) from dual; |  |
| Mod(m,n) | Remainder of m divided by n | Select mod(17,3) from dual; |  |
| Power(m,n) | m power n | Select power(5,3) from dual; |  |
| Round(m,n) | m rounded to n decimal places | Select round(125.67854,2) from dual; |  |
| Sign(n) | If n<0, -1 if n=0, 0  otherwise 1. | Select sin(-19) from dual; |  |
| Trunc(m,n) | m truncated to n decimal places | Select trunc(125.5764,2) from dual; |  |

**GROUP FUNCTIONS**

**Common Group Functions**

* AVG : Average value of a set
* COUNT : Numbers of non null values
* MAX : Maximum of a set
* MIN : Minimum of a set
* STDDEV : Standard Deviation of a set
* SUM : Sum of a set
* VARIANCE : Variance of a set

# Syntax :

SELECT *column*, *group\_function(column)*

FROM *table*

[WHERE *condition*]

[GROUP BY *group\_column\_or\_expression*] [HAVING *group\_condition*]

[ORDER BY *column*];

* Group functions ignore null values
* *Group by* Clause is used to modularize rows in a table into smaller groups
* Columns that are not a part of the Group Functions should be included in the Group by clause
* Any column or expression in the SELECT list that is not an aggregate function must be in the GROUP BY clause
* Group Functions cannot be placed in the where clause
* HAVING clause is to restrict groups Groups satisfying the HAVING condition are displayed
* Order of evaluation of the clauses :
* WHERE clause
* GROUP BY clause
* HAVING clause

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
| 7369 | SMITH | CLERK | 7902 | 17-DEC-80 | 800 |  | 20 |
| 7499 | ALLEN | SALESMAN | 7698 | 20-FEB-81 | 1600 | 300 | 30 |
| 7521 | WARD | SALESMAN | 7698 | 22-FEB-81 | 1250 | 500 | 30 |
| 7566 | JONES | MANAGER | 7839 | 02-APR-81 | 2975 |  | 20 |
| 7654 | MARTIN | SALESMAN | 7698 | 28-SEP-81 | 1250 | 1400 | 30 |
| 7698 | BLAKE | MANAGER | 7839 | 01-MAY-81 | 2850 |  | 30 |
| 7782 | CLARK | MANAGER | 7839 | 09-JUN-81 | 2450 |  | 10 |
| 7788 | SCOTT | ANALYST | 7566 | 19-APR-87 | 3000 |  | 20 |
| 7839 | KING | PRESIDENT |  | 17-NOV-81 | 5000 |  | 10 |
| 7844 | TURNER | SALESMAN | 7698 | 08-SEP-81 | 1500 | 0 | 30 |
| 7876 | ADAMS | CLERK | 7788 | 23-MAY-87 | 1100 |  | 20 |
| 7900 | JAMES | CLERK | 7698 | 03-DEC-81 | 950 |  | 30 |
| 7902 | FORD | ANALYST | 7566 | 03-DEC-81 | 3000 |  | 20 |
| 7934 | MILLER | CLERK | 7782 | 23-JAN-82 | 1300 |  | 10 |

**Data for DEPT table**

Q1) Find number of rows in the table EMP

# SQL >

**Q2)** Find number of designations available in EMP table.

# SQL>

**Q3)** What is the difference between the following queries

**SQL > select *count(comm)* from *emp*;**

**SQL > select *count(nvl(comm,0))* from *emp*;**

**Q4)** Find maximum, minimum and average salary in EMP table.

# SQL>

**Q5)** Find number of employees who work in department number 30

**SQL>**

**Q6)** Find the maximum salary paid to a ‘CLERK’

# SQL>

**Q7)** List the jobs and number of employees in each job. The result should be in the descending order of the number of employees.

**SQL>**

**Q8)** List the total salary, maximum and minimum salary and average salary of the employees jobwise.

# SQL>

**Q9)** List the total salary, maximum and minimum salary and average salary of the employees jobwise, for department 20 and display only those rows having an average salary > 1000.

**SQL>**

**Q10)** List the job and total salary of employees jobwise, for jobs other than ‘PRESIDENT’ and display only those rows having total salary > 5000.

# SQL>