**Function**

Single Row: We pass multiple records as input and get 1 row as output.

Multiple Row: We pass multiple records as input and get same number of rows as output.

Single:

Character

i. Case Manipulation: a Lower b. Upper c. initcap: Only first in upper

ii: Data Manipulation:

a. Length(); Length of String

b. concat: pass two agrs and combine

c. replace:

replace substring in a string Syntax: replace('OriginolString', 'Originol', 'New')// Input: Originol String and outptut: NewString

iii :Number

iv. Conversion

v. General

Multi:

Aggregate:

1. Max: Multiple Record input, single Value output

2. Min: Multiple Record input, single Value output

3. Avg

4. Sum

5. Count

**Substring**: This function is used to extract the substring from the given main string.

Syntax: Substring (Main String, Starting Pos, Ending Pos)

Eg. 1. Substring(‘SQL Developers’, 1, 5); // Lopers

Eg. 2. Substring(‘SQL’, 3, 2); // L<space><space>

**InString:** This function is usedto extract the position of the substring from the given main string

InString (‘Main String’, Substring , start pos, #of Occurrence)

e.g.

InString(‘SQL Developers’, ‘E’, 1, 1); //6

InString(‘SQL Developers’, ‘E’, 1, 2); //8

InString(‘SQL Developers’, ‘E’, 1, 3); //12

InString(‘SQL Developers’, ‘E’, 1, 4); //null

**Reverse():**  This function is used to reverse the given main string.

e.g. Reverse(‘Hello’) // olleH

Reverse(‘MOM’) // MOM

**TCL Commands:**

1. Commit🡪 It is TCL command which is used to save the changes permanently In the database. Usually commit command is used after performing DML operations (Insert, Delete and update).

Syntax: Commit;

1. Rollback🡪 It is a TCL command which is used to restore all the changes which are performed on the database. Using Rollback we can either rollback to the savepoint or rollback to the previous commit.

Examples:

**Rollback: savepoint-name;** //This will rollback or restore the changes till the specified savepoint.

**Rollback**; // If we don’t mention save point name then it goes till last commit.

* 1. Commit

Insert

Update id

Uddate marks

**Savepoint s1;**

Commit

Insert

Update id

Uddate marks

**Savepoint s2;**

Commit

Insert

Update id

Update marks

**Savepoint s3;**

*Rollback s1*; (It will remove data of s2 and s3)

*Rollback*; (It will remove s1, s2, s3 all and will take to last commit)

* 1. Insert;

Delete;

Commit;

Rollback; // Rollback will not have any affect.

* 1. Insert;

Delete;

Rollback;

Commit ; // Rollback will not have any affect.

* 1. Insert;

Savepoint s1;

Delete;

Savepoint s2;

Delete;

Savepoint s3;

Commit;

Rollback s1; // It will go to last commit. So no affect.

1. SavePoint: Since commit is a costly operation and making use of commit repeatedly will slow down performance of the application. To avoid this we make use of savepoint to save the changes temporarily in the database. We can use savepoint after every transaction (Set of operations on the same record).

This will save that particular transaction for time being, if we want to save these transactions permanently in the database then we have to perform Commit, before closing the application.

Syntax: Savepoint savepoint-name;

Savepoint s1;

1. FlashBack

**Distinct:** It is a keyword which is used to display the unique value of the selected column.

Usually we apply distinct on Non-Primary key and non unique columns.

Select job from emp; // Gives all values present in column

Select distinct job from emp; //Given only distinct values.

|  |  |
| --- | --- |
| **Unique** | **Distinct** |
| It is a constraint | It is a keyword. |
| Unique are used while creating the table. | It is used while displaying the data from the table. |
| It insures no duplicate values are inserted for the specified column. | It insures no duplicate values are displayed for the specified column. |

|  |  |  |
| --- | --- | --- |
| **Delete** | **Drop** | **Truncate** |
| It is a DML command. | It is DDL command. | It is DDL command. |
| Explicit commit is required to save the changes permanently in the database. | It is implicit commit command.  The changes will be saved automatically after performing this operation. | It is implicit commit command.  The changes will be saved automatically after performing this operation. |
| It is used to delete a specific row (record) or all the records present in the table. | It will delete all records as well as the structure of the table. (Table will be removed permanently from the database). | It will delete all the records of the table. |
| It can be rollback. | We can perform Flashback to get back the dropped table. | Once we perform truncate, we cannot get the records back. |

**IN, NOTIN, ANY, ALL Operators:**

Using equal operator, we can compare only one value at a time. If we have multiple values to be compared with the same column, then we go for IN/ Notin/ Any/ All.

**Question**: WAQ to display the emp details who are working for dept 10 and 20.

Ans1. Select \* from emp where dept= (10,20); //It will not work as = is used.

Ans2. Select \* from emp where deptno= 10 or deptno = 20;

Ans3. Select \* from emp where deptno in (10, 20, 30);

**Question 1:** WAQ to display the emp details who are working as salesman and manager.

**Question 2:** WAQ to display the emp details who belongs to deptno 10 or 20, and his job contain ‘a’ in it.

**Question 3**: WAQ to display the emp details who is not working in dept 10 or 30, and is getting some commission.

**Question 4:** WAQ to display the emp details who is earning salary between 1000 and 4000. Also is not getting any commission. And his name’s 2nd character contains ‘L’.

**Question 5:** WAQ to display the emp details who’s names last 3rd character is ‘I’ and job contains ‘R’ in it and getting salary between 1000 and 3000 and is getting some commission and belongs to department 10 or 20.

**Foreign Key:**

1. It is a constraint which is used to establish relationship between two tables.
2. To establish a relationship between two tables, we make use of the keyword “References”.

Usually foreign key of one table is primary key of other table.

1. Foreign Key Column will allow only those values which are present in the primary key column of the other table.

|  |  |
| --- | --- |
| **Book\_Id (pk)** | **Author** |
| 111 | Akhil |
| 112 | Ashok |
| 113 | Kishor |
| 114 | Shravya |
| 115 | Deepak |

|  |  |  |
| --- | --- | --- |
| **std\_Id (pk)** | **s\_name** | **Book\_id (fk)** |
| 11 | Dinga | 111 |
| 12 | Dingi | 112 |
| 13 | Pinga | 111 |
| 14 | Pingi |  |

|  |  |  |
| --- | --- | --- |
| **fc\_Id (pk)** | **fc\_name** | **std\_id (fk)** |
| 1 | Vishnu | 11 |
| 2 | Ranjit | 12 |
| 3 | Pratap | 12 |
| 4 | Anjali | 11 |
| 5 | Madhu | 14 |

SQL> create table library (bookid number(3) primary key, author varchar(20) not null);

SQL> create table student (sid number(2) primary key, sname varchar(20) not null, bookid number(2) references library(bookid));

SQL> create table faculty (fid number(1) primary key, fname varchar(20) not null, sid number(2) references student(sid));

SQL> insert into library values(111, 'Akhil');

SQL> insert into library values(112, 'Ashok');

SQL> insert into library values(113, 'Kishor');

SQL> insert into library values(114, 'Shreya');

SQL> insert into library values(115, 'Deepak');

SQL> insert into student values(11, 'Dinga', 111);

SQL> insert into student values(12, 'Dingi', 111);

SQL> insert into student values(13, 'Pinga', 112);

SQL> insert into student values(14, 'Pingi', null);

SQL> insert into faculty values(1, 'Vishnu', 11);

SQL> insert into faculty values(2, 'Ranjith', 12);

SQL> insert into faculty values(3, 'Pratap', 12);

SQL> insert into faculty values(4, 'Anji', 11);

SQL> insert into faculty values(5, 'Madhu', 14);

1 row created.

BOOKID AUTHOR

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

111 Akhil

112 Ashok

113 Kishor

114 Shreya

115 Deepak

SID SNAME BOOKID

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

11 Dinga 111

12 Dingi 111

13 Pinga 112

14 Pingi

FID FNAME SID

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

1 Vishnu 11

1. Ranjith 12

3 Pratap 12

4 Anji 11

5 Madhu 14

**Joins:** If we want to retrieve data from more than 1 table, we use JOINS. Usually we join the tables using primary key and foreign key, where primary key column of 1 table will be foreign key column of the other table.

Types of Joins:

1. Inner Join: It helps us to retrieve the matching records between 2 tables. It is also known as Simple join.

|  |  |
| --- | --- |
| Book\_Id (pk) | Author |
| 111 | Akhil |
| 112 | Ashok |
| 113 | Kishor |
| 114 | Shravya |
| 115 | Deepak |

|  |  |  |
| --- | --- | --- |
| std\_Id (pk) | s\_name | Book\_id (fk) |
| 11 | Dinga | 111 |
| 12 | Dingi | 112 |
| 13 | Pinga | 111 |
| 14 | Pingi |  |

Syntax:

Select c1, c2, cn from table1 t1 inner join table2 t2 On t1.fk, t2.pk;

Example:

Query: select s.sname, l.author from student s inner join library l on s.bookid= l.bookid;

Output:

SNAME AUTHOR

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

Dingi Akhil

Dinga Akhil

Pinga Ashok

1. ***Outer Join:***
2. ***Right Outer Join:*** The right outer join will display the matching records of both the tables and unmatched details of Right Table.

Syntax: Select c1, c2, cn from table1 t1 right outer join table2 t2 On t1.fk, t2.pk;

**Example:**

Query: select s.sname, l.author from student s right outer join library l on s.bookid= l.bookid;

Output:

SNAME AUTHOR

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

Dinga Akhil

Dingi Akhil

Pinga Ashok

Shreya

Kishor

Deepak

1. ***Left Outer Join:*** The left outer join will give matching details of both tables and along with it the unmatched details of left table.

**Syntax**: Select c1, c2, cn from table1 t1 left outer join table2 t2 On t1.fk, t2.pk;

**Example:**

Query: select s.sname, l.author from student s left outer join library l on s.bookid= l.bookid;

SNAME AUTHOR

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

Dinga Akhil

Dingi Akhil

Pinga Ashok

Pingi

1. ***Full Outer Join:*** It gives the common details of both the tables and along with it, the unmatched details of right table as well as left table.

**Syntax**: Select c1, c2, cn from table1 t1 full outer join table2 t2 On t1.fk, t2.pk;

**Example:**

Query: select s.sname, l.author from student s full outer join library l on s.bookid= l.bookid;

SNAME AUTHOR

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

Dinga Akhil

Dingi Akhil

Pinga Ashok

Pingi

Shreya

Kishor

Deepak