SQL
(Structured Query Language)

Day- 4

Dimple Waghela and Dhvani Parekh

Fractal®

String & Date Functions

STRING Functions

The following functions perform an operation on a string input value and return a string or numeric value:

ASCII	**NCHAR**	**STR**
CHAR	**REPLACE**	**SPACE**
LEFT	**REPLICATE**	**SUBSTRING**
LEN	**REVERSE**	**UPPER**
LOWER	**RIGHT**	
LTRIM	**RTRIM**	

STRING Functions

- Select ASCII('A');
- Select CHAR(6);
- Select CHARINDEX('C','ABCD',1);
- Select LEFT('SQL Session',3);
- Select RIGHT('SQL Session',7);
- Select LEN('SQL Session');
- Select LTRIM(' SQL Session');
- Select RTRIM('SQL Session ');
- Select LOWER('SQL Session');
- Select UPPER('SQL Session');
- Select SPACE(10)+ UPPER('SQL Session'); -- Concat is using + operator
- Select REPLACE('ABCD','BC','FG');
- Select REPLICATE('ABCD',2);
- Select REVERSE('ABCD');
- Select STR(65.45,2,1);--converts float, len,decimal optional
- Select SUBSTRING('ABCD',1,2);

DATE Functions

The following functions perform an operation on a string input value and return a string or numeric value:

Function	**Description**
GETDATE()	Returns the current date and time
DATEPART()	Returns a single part of a date/time
DATEADD()	Adds or subtracts a specified time interval from a date
DATEDIFF()	Returns the time between two dates
CONVERT()	Displays date/time data in different formats

Fractal® Analytics

TOP Clause

- The SQL **TOP** clause is used to fetch a TOP N number or X percent records from a table.

Syntax:

SELECT TOP number|percent column_name(s) FROM table_name WHERE [condition];

Example:

- SELECT TOP 3 * FROM CUSTOMERS;

SQL Alias

- SQL Aliases are defined for columns and tables. Basically aliases is created to make the column selected more readable.

Aliases is more useful when

- There are more than one tables involved in a query,
- Functions are used in the query,
- The column names are big or not readable,
- More than one columns are combined together

SQL Alias-Example

Example:

```
SELECT CustomerName AS Customer, ContactName AS
  [Contact Person]
  FROM Customers;
```

Displaying Data From Multiple Tables-Joins

SQL Joins

- SQL Joins are used to relate information in different tables.
- A Join condition is a part of the sql query that retrieves rows from two or more tables.
- A SQL Join condition is used in the SQL WHERE Clause of select, update, delete statements.

Types of Joins

JOINs in SQL Server can be classified as follows:

- INNER JOIN
- OUTER JOIN
- Non Eqijoins
- Self Join
- CROSS JOIN

The Syntax for joining two tables

```
SELECT col1, col2, col3...
FROM table_name1, table_name2
WHERE table_name1.col2 = table_name2.col1;
```

INNER JOIN

- Inner joins return rows only when there is at least one row from both tables that matches the join condition.
- Inner joins eliminate the rows that do not match with a row from the other table.

INNER -Example

```
SELECT orders1.ordernumber,customers.City
FROM Orders1
INNER JOIN Customers
ON
  orders1.Customernumber=customers.Customernumber;
```

LEFT OUTER JOIN

- Left Outer joins return all rows from the left table referenced with a left outer join and matching rows from other table.

- Unmatched records will be NULL.

LEFT OUTER JOIN-Example

```
SELECT customers.CustomerNumber,
   orders1.Amount,items.Description

FROM customers

LEFT OUTER JOIN orders1

ON

(customers.CustomerNumber=orders1.CustomerNumber)
```

RIGHT OUTER JOIN

- Right Outer joins return all rows from the right table referenced with a right outer join and matching rows from other table.

- Unmatched records will be NULL.

RIGHT OUTER JOIN-Example

```
SELECT customers.CustomerNumber, orders1.Amount
FROM orders1
RIGHT OUTER JOIN customers
ON
  customers.CustomerNumber=orders1.CustomerNumber
```

FULL OUTER JOIN

- Full Outer joins return all rows from both the tables. Unmatched records will be NULL.

Example:

```
SELECT customers.CustomerNumber+2, orders1.Amount
FROM customers FULL OUTER JOIN orders1 ON
customers.CustomerNumber+2=orders1.CustomerNumber
```

Non Equijoins Joins

- **SQL Non equijoins:** It is a Sql join condition which makes use of some comparison operator other than the equal sign like >, <, >=, <=

- **SELECT** name, age
  FROM employee
  WHERE salary != 30000

Self Joins

Example SQL Self Join:

```
SELECT a.sales_person_id, a.name, a.manager_id,
  b.sales_person_id, b.name
  FROM sales_person a, sales_person b
  WHERE a.manager_id = b.sales_person_id;
```

CROSS JOIN

- In cross joins, each row from first table joins with all the rows of another table.
- If there are m rows from Table1and n rows from Table2, then result set of these tables will have m*n rows.

CROSS JOIN Example

```
SELECT Zip, FirstName, amount
FROM Orders1
CROSS JOIN Customers
```

Set Operators

Fractal® Analytics

Set Operators

According to SQL Standard there are following

Set operator types:

- UNION
- UNION ALL
- INTERSECT
- EXCEPT (Minus in Oracle)

UNION Operator

- The UNION operator is used to combine the result-set of two or more SELECT statements.
- Notice that each SELECT statement within the UNION must have the same number of columns.
- The columns must also have similar data types. Also, the columns in each SELECT statement must be in the same order.

Fractal® Analytics

UNION Example

Union cities from table1 and table2

```
SELECT city FROM table1
UNION
SELECT city FROM table2;
```

Union ALL Operator

- Union all allows all duplicate values in the output set.

Example:

```
SELECT city FROM table1
UNION ALL
SELECT city FROM table2;
```

Intersect Operator

- Intersect returns only these rows, which are in both tables.

Example :

```
SELECT city FROM table1

INTERSECT

SELECT city FROM table2;
```

Except Operator (Minus)

- Except returns unique rows that are returned by the first query but are NOT returned by the second query.

Example:

```
SELECT city FROM table1

EXCEPT

SELECT city FROM table2
```

SubQueries

Subquery

- Subquery or Inner query or Nested query is a query in a query.
- A subquery is usually added in the WHERE Clause of the Sql statement.
- Subqueries can be used with the following sql statements along with the comparison operators like =, <, >, >=, <= etc.

SELECT,INSERT,UPDATE,DELETE

Fractal® Analytics

Subquery

A subquery can be used in the places of a query are

- Within the list of columns in the SELECT statement
- With the FROM clause
- With the WHERE clause
- With the HAVING clause
- With the GROUP BY clause.

Subquery Example

```
SELECT  firstname,City
FROM customers
WHERE city IN ('Plano', 'Reo');

SELECT customernumber,lastname
FROM customers
WHERE state NOT IN ('CA');
```

Subquery Example

```
SELECT FirstName, Zip, State
FROM Customers
WHERE city IN (SELECT city
FROM Customers
WHERE city= 'Plano');
```

Correlated subquery

- A query is called correlated subquery when both the inner query and the outer query are interdependent.
- For every row processed by the inner query, the outer query is processed as well.
- The inner query depends on the outer query before it can be processed.

Correlated-Example

```
SELECT zip FROM customers
WHERE customernumber =
(SELECT CustomerNumber from customers  where city
   ='Reo')
```

Correlated-Example

```
SELECT last_name, salary
FROM   employee
WHERE  salary >(SELECT salary FROM   employee
  WHERE  last_name = ‘Able');


SELECT * FROM employee WHERE salary >(SELECT
  Min(salary) FROM employees)
```

Fractal® Analytics

Other Database Objects

Views

- In SQL, a view is a virtual table based on the result-set of an SQL statement.
- A view contains rows and columns, just like a real table. The fields in a view are fields from one or more real tables in the database.
- You can add SQL functions, WHERE, and JOIN statements to a view and present the data as if the data were coming from one single table.

Types of Views -Simple View

- When we create a view on a single table, it is called simple view.

- In simple view we can insert, update, delete data. We can only insert data in simple view if we have primary key and all not null fields in the view.

Fractal® Analytics

Simple View-Example

Syntax:

```
CREATE VIEW view_name AS
  SELECT column_name(s)
  FROM table_name
  WHERE condition
```

Example :

```
CREATE VIEW mycus AS

SELECT * FROM customers
```

Types of Views -Complex View

- When we create a view on more than one table, it is called complex view.

Example:

```
Create VIEW vw_Employee_Personal_Info
As
Select e.Emp_ID,
  e.Emp_Name,e.Emp_Designation,p.DOB,p.Mobile
From Employee_Test123 e INNER JOIN Personal_Info p
On e.Emp_Name = p. Emp_Name;
Select * from vw_employee_personal_info
```

Thank You

Thank You

Fractal® Analytics

Fortune 500 companies recognize analytics as a competitive differentiator to **understand customers** and make **better decisions**.

We deliver insight, impact and innovation to them through **predictive analytics** and **visual story-telling**.

fractalanalytics.com

United States | Canada | United Kingdom | Italy | Switzerland | Singapore | UAE | India | China