SQL Server Advanced T-SQL

By,
Srikar Reddy Gondesi,
Junior SQL Server DBA,
Database Team,
Miracle Software Systems. Inc,
Email: sgondesi@miraclesoft.com

SQL Concatenation

- Concatenation can result in more readable output while maintaining data in separate columns for greater flexibility.

- For example,

the concatenation of **'John', ' ', 'W', '. ' and 'Smith'** is **'John W. Smith'**

Sub-String

- Sub-string is useful when accessing a column that consists of meaningful sub-components, such as a telephone number that contains area code, prefix and phone number body.

- SQL SUBSTRING Syntax,

SELECT SUBSTRING(<column_name>, position, length)

FROM <table_name>

- For example,

the function **SUBSTRING('212-555-1234', 9 , 4) returns: '1234'**

It returns 4 characters starting in position 9.

Trim

- The SQL Trim functions (LTRIM and RTRIM) are useful in cleansing data that contains leading and trailing blanks.

- SQL TRIM Syntax

SELECT LTRIM(<value_1>)

SELECT RTRIM(<value_1>)

- For example,

the function **LTRIM(' Minnesota') returns: 'Minnesota'**

AND & OR

- Use SQL AND & OR operators in case when a WHERE clause requires multiple conditions. For example, we may want to select a list of CUSTOMERS whose accounts are on credit hold and whole credit balance exceeds the credit limit.

- Such a statement may look like this:

```
SELECT customer_nbr, customer_name, credit_limit_amt,
    credit_balance_amt
FROM CUSTOMER
WHERE credit_hold_ind = 'Y'
AND credit_balance_amt > credit_limit_amt
```

SQL IN

- The SQL IN operator is a clean way to check for inclusion in lists without creating a series of OR clauses.The SQL IN operator enables a comparison to a list of values or to the results of a SUBSELECT.

- For Example,

```
SELECT branch_nbr, branch_name, region_nbr
FROM branch
WHERE region_nbr IN (200, 400)
```

Between

- The SQL BETWEEN operator enables a comparison to a range of values. The SQL IN operator is a clean way to check for inclusion in a range without requiring an AND operator.

- SQL BETWEEN Syntax

SELECT <column_list>

FROM <table_name>

WHERE <column_name> BETWEEN

<lower_value> AND <higher_value>

Group By

- The SQL GROUP BY is a clause enables SQL aggregate functions for a grouping of information. For example, it could support subtotaling by region number in a sales table.
- GROUP BY supports dividing data into sets so that aggregate functions like SUM, AVG and COUNT can be performed.

- SQL GROUP BY Syntax

SELECT <column_name1>, <column_name2>
<aggregate_function>
FROM <table_name>
GROUP BY <column_name1>, <column_name2>

Aggregate Functions

- The SQL Aggregate Functions are functions that provide mathematical operations. If you need to add, count or perform basic statistics, these functions will be of great help.

The functions include:

- count() - counts a number of rows
- sum() - compute sum
- avg() - compute average
- min() - compute minimum
- max() - compute maximum

Aggregate Functions

- The SQL Aggregate Functions are useful when mathematical operations must be performed on all or a grouping of values. SQL Aggregate Functions are used as follows. If a grouping of values is needed also include the GROUP BY clause.

- SQL Aggregate Functions Syntax

SELECT <column_name1>, <column_name2>
<aggregate_function(s)>
FROM <table_name>
GROUP BY <column_name1>, <column_name2>

Having

- The SQL HAVING is a clause that enables conditions at the aggregate level. It is used instead of the WHERE clause when Aggregate Functions are used. The SQL HAVING clause supports the same operators as the WHERE clause. Grouped results can be checked using this clause.

- SQL HAVING Syntax

```
SELECT <column_name1>, <column_name2>
   <aggregate_function>
FROM <table_name>
GROUP BY <column_name1>, <column_name2>
HAVING <having_condition>
```

Order By

- The SQL ORDER BY clause is the part of the SELECT statement that controls the sequence of information returned. The SQL ORDER BY clause is really a sort feature that enables the output to be sorted by one or more columns.

- SQL ORDER BY Syntax

SELECT <column_name1>, <column_name2>

FROM <table_name>

ORDER BY <column_name1>[ASC|DESC],
<column_name2>[ASC|DESC]

JOIN

- The SQL JOIN is a clause that enables a SELECT statement to access more than one table. The JOIN clause controls how tables are linked. It is a qualifier of the SQL FROM clause.
- The standard JOIN clause (also known as the INNER JOIN clause) differs from the OUTER JOIN in that rows are returned only when there are matches for the JOIN critieria on the second table.

- SQL JOIN Syntax

SELECT <column_name1>, <column_name2>
<aggregate_function>
FROM <table_name>
JOIN <table_name> ON <join_conditions>

Distinct

- SQL DISTINCT is a SQL keyword that specifies that only distinct / non-duplicate results are to be returned. DISTINCT can apply to single columns or to multiple columns.
- The DISTINCT keyword can help to understand information. For example, a query containing DISTINCT could return a non-duplicated list of cities where a company does business.

- SQL DISTINCT Syntax

SELECT DISTINCT <column_name>

FROM <table_name>

Like

- The SQL LIKE operator is a powerful way to check for matching strings. It is often used for interactive display of lists. The SQL LIKE operator enables a comparison to a part of a string using the % wild card character.

- SQL LIKE Syntax

SELECT <column_list>

FROM <table_name>

WHERE <column_name> LIKE <like_condition>

THANK YOU

MIRACLE MSS
SOFTWARE SYSTEMS, INC.