

LAB - Distinct

In this lab, you will learn how to use the SQL Server `DISTINCT` clause to retrieve the only distinct values in a specified list of columns.

Sometimes, you may want to get only distinct values in a specified column of a table. To do this, you use the `DISTINCT` clause as follows:

```
SELECT DISTINCT
    column_name
FROM
    table_name;
```

The query returns only distinct values in the specified column. In other words, it removes the duplicate values in the column from the result set.

If you use multiple columns as follows:

```
SELECT DISTINCT
    column_name1,
    column_name2 ,
    ...
FROM
    table_name;
```

The query uses the combination of values in all specified columns in the `SELECT` list to evaluate the uniqueness.

If you apply the `DISTINCT` clause to a column that has NULL, the `DISTINCT` clause will keep only one NULL and eliminates the other. In other words, the `DISTINCT` clause treats all NULL “values” as the same value.

Examples

For the demonstration, we will use the `customers` table

sales.customers
* customer_id
first_name
last_name
phone
email
street
city
state
zip_code

A) **DISTINCT** one column example

The following statement returns all cities of all customers in the `customers` tables:

```
SELECT
    city
FROM
    sales.customers
ORDER BY
    city;
```

city
Albany
Albany
Albany
Amarillo
Amarillo
Amarillo
Amarillo
Amarillo

As you can see clearly from the output, the cities are duplicate.

To get distinct cities, you add the `DISTINCT` keyword as follows:

```
SELECT DISTINCT
    city
FROM
    sales.customers
ORDER BY
    city;
```

city
Albany
Amarillo
Amityville
Amsterdam
Anaheim
Apple Valley
Astoria
Atwater

Now, the query returns a distinct value for each group of duplicates. In other words, it removed all duplicate cities from the result set.

B) **DISTINCT** multiple columns example

This statement returns all cities and states of all customers:

```
SELECT
    city,
    state
FROM
    sales.customers
ORDER BY
    city,
    state;
```

city	state
Albany	NY
Albany	NY
Albany	NY
Amarillo	TX
Amarillo	TX
Amarillo	TX
Amarillo	TX
Amarillo	TX
Amityville	NY
Amityville	NY
Amityville	NY
Amityville	NY
Amityville	NY

The following statement finds the distinct city and state of all customers.

```
SELECT DISTINCT
    city,
    state
FROM
    sales.customers
```

city	state
Hopewell Junction	NY
Houston	TX
Howard Beach	NY
Huntington	NY
Huntington Station	NY
Ithaca	NY
Jackson Heights	NY
Jamaica	NY

In this example, the statement used the combination of values in both `city` and `state` columns to evaluate the duplicate.

C) `DISTINCT` with null values example

The following example finds the distinct phone numbers of the customers:

```
SELECT DISTINCT
    phone
FROM
    sales.customers
ORDER BY
    phone;
```

phone
NULL
(210) 436-8676
(210) 851-3122
(212) 152-6381
(212) 171-1335
(212) 211-7621
(212) 325-9145
(212) 578-2912

In this example, the `DISTINCT` clause kept only one NULL in the `phone` column and removed the other NULLs.

`DISTINCT` vs. `GROUP BY`

The following statement uses the `GROUP BY` clause to return distinct cities together with state and zip code from the `sales.customers` table:

```

SELECT
    city,
    state,
    zip_code
FROM
    sales.customers
GROUP BY
    city, state, zip_code
ORDER BY
    city, state, zip_code;

```

The following picture shows the partial output:

city	state	zip_code
Albany	NY	12203
Amarillo	TX	79106
Amityville	NY	11701
Amsterdam	NY	12010
Anaheim	CA	92806
Apple Valley	CA	92307
Astoria	NY	11102
Atwater	CA	95301
Auburn	NY	13021
Bakersfield	CA	93306
Baldwin	NY	11510
Baldwinsville	NY	13027
Ballston Spa	NY	12020

It is equivalent to the following query that uses the `DISTINCT` operator :

```

SELECT
    DISTINCT
        city,
        state,
        zip_code
FROM
    sales.customers;

```

Both `DISTINCT` and `GROUP BY` clause reduces the number of returned rows in the result set by removing the duplicates.

However, you should use the `GROUP BY` clause when you want to apply an aggregate function on one or more columns.

In this lab, you have learned how to use the SQL Server `DISTINCT` clause to retrieve the distinct values in a specified list of columns.

