**SQL CREATE INDEX Statement**

The CREATE INDEX statement is used to create indexes in tables.

Indexes are used to **retrieve data from the database more quickly than otherwise**. The users cannot see the indexes, **they are just used to speed up searches/queries.**

**Note:** Updating a table with indexes takes more time than updating a table without (because the indexes also need an update). So, only create indexes on columns that will be frequently searched against.

### CREATE INDEX Syntax

Creates an index on a table. Duplicate values are allowed:

CREATE INDEX index\_name  
ON table\_name (column1, column2, ...);

### CREATE UNIQUE INDEX Syntax

Creates a unique index on a table. Duplicate values are not allowed:

CREATE UNIQUE INDEX index\_name  
ON table\_name (column1, column2, ...);

**Note:** The syntax for creating indexes varies among different databases. Therefore: Check the syntax for creating indexes in your database.

## CREATE INDEX Example

The SQL statement below creates an index named "idx\_lastname" on the "LastName" column in the "Persons" table:

CREATE INDEX idx\_lastname  
ON Persons (LastName);

If you want to create an index on a combination of columns, you can list the column names within the parentheses, separated by commas:

CREATE INDEX idx\_pname  
ON Persons (LastName, FirstName);

## DROP INDEX Statement

The DROP INDEX statement is used to delete an index in a table.

**MS Access:**

DROP INDEX index\_name ON table\_name;

**SQL Server:**

DROP INDEX table\_name.index\_name;

**DB2/Oracle:**

DROP INDEX index\_name;

**MySQL:**

ALTER TABLE table\_nameDROP INDEX index\_name;

## SQL CREATE VIEW Statement

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.

### CREATE VIEW Syntax

CREATE VIEW view\_name AS  
SELECT column1, column2, ...  
FROM table\_name  
WHERE condition;

**Note:** A view always shows up-to-date data! The database engine recreates the data, using the view's SQL statement, every time a user queries a view.

## SQL CREATE VIEW Examples

The following SQL creates a view that shows all customers from Brazil:

### Example

CREATE VIEW [Brazil Customers] AS  
SELECT CustomerName, ContactName  
FROM Customers  
WHERE Country = "Brazil";

OR

CREATE VIEW `new\_view` AS

SELECT CustomerName, ContactName

FROM Customers

WHERE Country = "Brazil";

We can query the view above as follows:

### Example

SELECT \* FROM [Brazil Customers];

The following SQL creates a view that selects every product in the "Products" table with a price higher than the average price:

### Example

CREATE VIEW [Products Above Average Price] AS  
SELECT ProductName, Price  
FROM Products  
WHERE Price > (SELECT AVG(Price) FROM Products);

We can query the view above as follows:

### Example

SELECT \* FROM [Products Above Average Price];

## SQL Updating a View

A view can be updated with the CREATE OR REPLACE VIEW command.

### SQL CREATE OR REPLACE VIEW Syntax

CREATE OR REPLACE VIEW view\_name AS  
SELECT column1, column2, ...  
FROM table\_name  
WHERE condition;

The following SQL adds the "City" column to the "Brazil Customers" view:

### Example

CREATE OR REPLACE VIEW [Brazil Customers] AS  
SELECT CustomerName, ContactName, City  
FROM Customers  
WHERE Country = "Brazil";

## SQL Dropping a View

A view is deleted with the DROP VIEW command.

### SQL DROP VIEW Syntax

DROP VIEW view\_name;

The following SQL drops the "Brazil Customers" view:

### Example

DROP VIEW [Brazil Customers];