SQL Views

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 statements and functions to a view and present the data as if the data were coming from one single table.

A view is created with the CREATE VIEW statement.

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 view, every time a user queries it.

SQL CREATE VIEW Examples

The following SQL creates a view that shows all productnames with orderno greater than 2:

Example

CREATE VIEW market AS  
SELECT \*  
FROM sales  
WHERE orderno>2;

We can query the view above as follows:

Example

SELECT \* FROM market;

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 market3 AS  
SELECT \*  
FROM product  
WHERE price > (SELECT AVG(price) FROM product);

We can query the view above as follows:

Example

SELECT \* FROM market3;

Note:

select \* from market3 where pid>2;

SQL Updating a View

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

SQL CREATE OR REPLACE VIEW Syntax

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

The following SQL shows the "pid and pname" in the "market3" view:

Example

CREATE OR REPLACE VIEW market3 AS  
SELECT pid,pname  
FROM Customers  
WHERE price > (SELECT AVG(price) FROM product);

SQL Dropping a View

A view is deleted with the DROP VIEW statement.

SQL DROP VIEW Syntax

DROP VIEW *view\_name*;

The following SQL drops the "Market3" view:

Example

DROP VIEW market3;

## What is a NULL Value?

A field with a NULL value is a field with no value.

It is not possible to test for NULL values with comparison operators, such as =, <, or <>.

We will have to use the IS NULL and IS NOT NULL operators instead.

SELECT column\_namesFROM table\_name  
WHERE column\_name IS NULL;

SELECT column\_namesFROM table\_name  
WHERE column\_name IS not NULL;