**Lab Manual for Database Management System**

**Lab No. 9**

# VIEWS IN SQL

Objectives

Using SQL VIEWS

# *LAB 08: Views in SQL*

### INTRODUCTION

A database view is a virtual table or logical table which is defined as a [SQL SELECT query](http://www.mysqltutorial.org/mysql-select-statement-query-data.aspx) with [joins](http://www.mysqltutorial.org/mysql-inner-join.aspx). Because a database view is similar to a database table, which consists of rows and columns, so you can query data against it.

### OBJECTIVE

After the completion of this lab students shall be able to

* Create, alter and drop views

### THEORY

### Views

We should be able to use views to:

* Reduce apparent database complexity for end users
* Prevent sensitive columns from being selected, while still affording access to other important data
* Add additional indexing to your database to speed query performance

A view is, at its core, really nothing more than a stored query. What’s great is that you can mix and match your data from base tables (or other views) to create what will, in most respects, function just like another base table. You can create a simple query that selects from only one table and leaves some columns out, or you can create a complex query that joins several tables and makes them appear as one

**Syntax**

The syntax for creating a view

CREATE VIEW <view name>

AS

<SELECT statement>

**Example 1**

We’ll call this one our customer phone list, and create it as CustomerPhoneList\_vw in our Accounting database:

USE Accounting

GO

CREATE VIEW CustomerPhoneList\_vw

AS

SELECT CustomerName, Contact, Phone

FROM Customers

**Example 2**

For a more complex example, let’s briefly go back to using the Northwind database. Our manager would like to be able to do simple queries that will tell him or her. What orders have been placed for what parts and who placed them. So, we create a view that they can perform very simple queries on—remember that we are creating this one in Northwind.

USE Northwind

GO

CREATE VIEW CustomerOrders\_vw

AS

SELECT cu.CompanyName,

o.OrderID,

o.OrderDate,

od.ProductID,

p.ProductName,

od.Quantity,

od.UnitPrice,

od.Quantity \* od.UnitPrice AS ExtendedPrice

FROM Customers AS cu

INNER JOIN Orders AS o

ON cu.CustomerID = o.CustomerID

INNER JOIN [Order Details] AS od

ON o.OrderID = od.OrderID

INNER JOIN Products AS p

ON od.ProductID = p.ProductID

**Limit What’s Inserted into Views—WITH CHECK OPTION**

The WITH CHECK OPTION is one of those lesser known to almost completely unknown features in SQL Server. The rules are simple—in order to update or insert data using the view, the resulting row must qualify to appear in the view results. Restated, the inserted or updated row must meet any WHERE criterion that’s used in the SELECT statement that underlies your view.

CREATE VIEW OregonShippers\_vw

AS

SELECT ShipperID,

CompanyName,

Phone

FROM Shippers

WHERE Phone LIKE ‘(503)%’

OR Phone LIKE ‘(541)%’

OR Phone LIKE ‘(971)%’

WITH CHECK OPTION

**Editing Views with T-SQL**

The only differences between using the ALTER VIEW statement and the CREATE VIEW statement are:

* ALTER VIEW expects to find an existing view, whereas CREATE doesn’t.
* ALTER VIEW retains any permissions that have been established for the view.
* ALTER VIEW retains any dependency information.

**Dropping Views**

It doesn’t get much easier than this:

DROP VIEW <view name>, [<view name>,[ ...n]]

And it’s gone.

### ACTIVITY TIME BOXING

|  |  |  |
| --- | --- | --- |
| **Activity Name** | **Activity Time** | **Total Time** |
| **Instruments Allocation + Setting up Lab** | 10 mints | 10 mints |
| **Walk through Theory & Tasks (Lecture)** | 60 mints | 60 mints |
| **Implementation & Practice time** | 90 mints | 80 mints |
| **Evaluation Time** | 20 mints | 20 mints |
|  | Total Duration | 180 mints |

### EXERCISE:

1. Add a view called Managers in the Northwind database that shows only employees that supervise other employees.
2. Create Northwind view called “Products by Category” based on the columns CategoryName and ProductName.
3. Create a view that shows a table that list all the publishers and store names using pubs.
4. Create a view that shows a table that list all the authors and store names using pubs.
5. Create a view the amount total amount spend by customers.
6. Create a view for customers shippers information.