

[Home](#) / [MySQL Views](#) / Creating MySQL Updatable Views

## Creating MySQL Updatable Views

**Summary:** in this tutorial, we will show you how to **create an updatable view** and update data in the underlying table through the view.

### Introduction to MySQL updatable views

In MySQL, views are not only query-able but also updatable. It means that you can use the **INSERT** or **UPDATE** statement to insert or update rows of the base table through the updatable view. In addition, you can use **DELETE** statement to remove rows of the underlying table through the view.

However, to create an updatable [view](#), the **SELECT** statement that defines the view must not contain any of the following elements:

- Aggregate functions such as **MIN**, **MAX**, **SUM**, **AVG**, and **COUNT**.
- **DISTINCT**
- **GROUP BY** clause.
- **HAVING** clause.
- **UNION** or **UNION ALL** clause.
- **Left join** or outer join.
- **Subquery** in the **SELECT** clause or in the **WHERE** clause that refers to the table appeared in the **FROM** clause.
- Reference to non-updatable view in the **FROM** clause.
- Reference only to literal values.
- Multiple references to any column of the base table.

If you create a view [with the TEMPTABLE algorithm](#), you cannot update the view.

Note that it is sometimes possible to create updatable views based on multiple tables using an [inner join](#).

### MySQL updatable view example

Let's create an updatable view.

First, we create a view named **officeInfo** based on the **offices** table in the [sample database](#). The view refers to three columns of the **offices** table: **officeCode**, **phone**, and **city**.

```
1 CREATE VIEW officeInfo
2 AS
3     SELECT officeCode, phone, city
4     FROM offices;
```

Next, we can query data from the **officeInfo** view using the following statement:

```
1 SELECT
2 *
3 FROM
4     officeInfo;
```

	officeCode	phone	city
▶ 1		+1 650 219 4782	San Francisco
2		+1 215 837 0825	Boston
3		+1 212 555 3000	NYC
4		+33 14 723 4404	Paris
5		+81 33 224 5000	Tokyo
6		+61 2 9264 2451	Sydney
7		+44 20 7877 2041	London

Then, we can change the phone number of the office with **officeCode** 4 through the **officeInfo** view using the following **UPDATE** statement.

```
1 UPDATE officeInfo
2 SET
3     phone = '+33 14 723 5555'
4 WHERE
5     officeCode = 4;
```

#### MYSQL QUICK START

[What Is MySQL?](#)

[Install MySQL Database Server](#)

[Connect to MySQL Server](#)

[Download MySQL Sample Database](#)

[Load Sample Database](#)

#### MYSQL VIEWS

[MySQL CREATE VIEW](#)

[MySQL View Processing Algorithms](#)

[MySQL Updatable Views](#)

[MySQL Rename View](#)

[MySQL Show View](#)

[MySQL DROP VIEW](#)

[MySQL View WITH CHECK OPTION](#)

[MySQL View LOCAL & CASCADED](#)

Finally, to verify the change, we can query the data from the `officeInfo` view by executing the following query:

```
1 SELECT
2   *
3 FROM
4   officeInfo
5 WHERE
6   officeCode = 4;
```

	officeCode	phone	city
▶	4	+33 14 723 5555	Paris

## Checking updatable view information

You can check if a view in a database is updatable by querying the `is_updatable` column from the `views` table in the `information_schema` database.

The following query gets all views from the `classicmodels` database and shows which views are updatable.

```
1 SELECT
2   table_name,
3   is_updatable
4 FROM
5   information_schema.views
6 WHERE
7   table_schema = 'classicmodels';
```

	table_name	is_updatable
▶	aboveavgproducts	YES
	customerorders	NO
	officeinfo	YES
	salepersonorder	NO

## Removing rows through the view

First, we [create a table](#) named `items`, [insert](#) some rows into the `items` table, and [create a view](#) that contains items whose prices are greater than 700.

```
1 -- create a new table named items
2 CREATE TABLE items (
3   id INT AUTO_INCREMENT PRIMARY KEY,
4   name VARCHAR(100) NOT NULL,
5   price DECIMAL(11, 2) NOT NULL
6 );
7
8 -- insert data into the items table
9 INSERT INTO items(name,price)
10 VALUES('Laptop',700.56),('Desktop',699.99),('iPad',700.50) ;
11
12 -- create a view based on items table
13 CREATE VIEW LuxuryItems AS
14 SELECT
15   *
16 FROM
17   items
18 WHERE
19   price > 700;
20 -- query data from the LuxuryItems view
21 SELECT
22   *
23 FROM
24   LuxuryItems;
```

	id	name	price
▶	1	Laptop	700.56
	3	iPad	700.50

Second, we use the `DELETE` statement to remove a row with `id` value 3.

```
1 DELETE FROM LuxuryItems
2 WHERE
3   id = 3;
```

MySQL returns a message saying that 1 row(s) affected.

Third, let's check the data through the view again.

```
1 SELECT
2   *
3 FROM
4   LuxuryItems;
```

	id	name	price
▶	1	Laptop	700.56

Fourth, we can also query the data from the base table `items` to verify if the `DELETE` statement actually deleted the row.

```
1 SELECT
2 *
3 FROM
4 items;
```

	id	name	price
▶	1	Laptop	700.56
	2	Desktop	699.99

As you see, the row with id 3 was removed from the base table.

In this tutorial, we have shown you how to create an updatable view and update data in the underlying table through the view.

## Related Tutorials

- [Understanding LOCAL & CASCADED in WITH CHECK OPTION Clause](#)
- [Managing Views in MySQL](#)

Was this tutorial helpful?

Yes

No

Previous Tutorial:  
[MySQL ALTER VIEW](#)

Next Tutorial:  
[MySQL Views & the WITH CHECK OPTION Clause](#)

### MYSQL PROGRAMMING INTERFACES

[PHP MySQL Tutorial](#)  
[Node.js MySQL Tutorial](#)  
[Python MySQL Tutorial](#)  
[Perl MySQL Tutorial](#)  
[MySQL JDBC Tutorial](#)

### OTHERS

[MySQL Cheat Sheet](#)  
[MySQL Resources](#)  
[MySQL Books](#)

### RECENT MYSQL TUTORIALS

[MySQL Stored Object Access Control](#)  
[Install MySQL on Ubuntu](#)  
[Install MySQL CentOS](#)  
[Getting Started with MySQL](#)  
[Connect to MySQL Server](#)  
[How To Lock User Accounts in MySQL](#)  
[How To Unlock User Accounts in MySQL Server](#)  
[Restart MySQL Server](#)  
[Start MySQL Server](#)  
[Stop MySQL Server](#)

### ABOUT MYSQL TUTORIAL WEBSITE

MySQLTutorial.org is a website dedicated to MySQL database. We regularly publish useful MySQL tutorials to help web developers and database administrators learn MySQL faster and more effectively.

All MySQL tutorials are practical and easy-to-follow, with SQL script and screenshots available. [More About Us](#)

### SITE LINKS

[About Us](#)  
[Contact Us](#)  
[Request a Tutorial](#)  
[Privacy Policy](#)