

Professional Skills
Agile Fundamentals
Jira
Git
Databases Introduction
<div><div></div>Installing MySQL on Windows</div>
<div><div></div>Provision a MySQL Server (Google Cloud Platform)</div>
<div><div></div>Introduction to Relational Databases</div>
<div><div></div>Data Design</div>
<div><div></div>Data Definition Language (DDL)</div>
<div><div></div>Entity-Relationship Diagrams</div>
<div><div></div>Data Manipulation Language (DML)</div>
<div><div></div>Data Query Language using SELECT</div>
<div><div></div>Aggregate Functions</div>
<div><div></div>Nested Queries</div>
<div><div></div>Joins</div>
<div><div></div>Data Normalisation</div>
Java Beginner
Maven
Testing (Foundation)
Java Intermediate
HTML
CSS
Javascript
Spring Boot
Selenium
Sonarqube
Advanced Testing (Theory)
Cucumber
MongoDB
Express
NodeJS

Data Manipulation Language (DML)

Contents

- [Overview](#)
- [CRUD operations](#)
 - [Inserting Data](#)
 - [Deleting Records](#)
 - [Updating Records](#)
 - [Viewing records](#)
- [Tutorial](#)
- [Exercises](#)

Overview

Data Manipulation Language (DML) is a subset of SQL that is used to manipulate the *content* of the database.

This is different to DDL, which is used to manipulate the schema rather than the data.

DML is arguably the most widely-used subtype of MySQL, and is integrated often into applications to read/write data to/from databases.

The most well-used CRUD operations we’d expect to use in DML are:

- Inserting data to, and deleting data from, table
- Reading data from tables based on various criteria
- Updating the existing records in a table

CRUD operations

You may see applications and operations being referred to as **CRUD** applications/operations. CRUD stands for:

- Create
- Read
- Update
- Delete

In MySQL syntax, there are a few ways that we can use CRUD functionality:

Operation	SQL
<i>create</i>	INSERT INTO
<i>read</i>	SELECT
<i>update</i>	UPDATE
<i>delete</i>	DELETE

Inserting Data

The syntax for inserting records into a table breaks down into the following:

- Specify the table and columns that we're inserting data into
- Specify the values that we want to enter, in the same order that we used to identify the columns

React
Express-Testing
Networking
Security
Cloud Fundamentals
AWS Foundations
AWS Intermediate
Linux
DevOps
Jenkins Introduction
Jenkins Pipeline
Markdown
IDE Cheatsheet

If we are inserting into specific fields in a table, we must specify them:

```
INSERT INTO table_name (column_1, column_4, column_5)
VALUES (value_1, value_2, value_3);
```

However, if we're inserting into all fields, there is no need to specify them:

```
INSERT INTO table_name
VALUES (value_1, value_2, value_3, value_4, value_5);
```

If we have a field that auto increments (such as the **PRIMARY KEY** field for a table), we need to specify the rest of the fields when inserting data:

```
INSERT INTO customers (forename, surname, age)
VALUES ('Jeff', 'Cyrus', 29);
```

Deleting Records

Deleting records from a table uses the **DELETE** keyword.

If you don't specify any criteria, MySQL will delete all records, so be very careful!

```
DELETE FROM customer_archive;
```

To delete a specific record, you should specify criteria; this is done with the **WHERE** keyword.

```
DELETE FROM orders WHERE status='Cancelled';
```

Updating Records

The syntax for updating records in a table breaks down into the following:

- Outline the table that the record exists in
- Specify the value for the changed field
- Outline any conditions

```
UPDATE table_name
SET column1=value1, column2=value2
WHERE field=value;
```

This becomes more difficult as the database has more relationships within records, as constraints may prevent certain fields from being edited (such as **PRIMARY KEY** fields).

Viewing records

We can view the records within a table with the **SELECT** keyword:

```
SELECT * FROM table_name;
```

SELECT is technically its own language - **Data Query Language** - and is covered in the [Data Query Language module](#).

Tutorial

There is no tutorial for this module.

Exercises

Start by inserting at least 5 records per table. It's recommended you start with **customers** and **products**, followed by **orders**.

(note: if you have not created a schema for the games shop database yet, refer back to the [Data Definition](#) module for context)

► Solution

