POWERING
POTENTIAL

The SELECT keyword

Module 5: Databases

Databases

Module 5: Contents

- Data Definition Language.
- The SELECT keyword.
- Conditions with SELECT.
- Limit and order.

Objectives

The SELECT keyword

Read record information from a database using DML:

- What does the syntax look like? How is the data useful?

Add conditionals to DML reads to refine results:

- How can we easily refine our data?

CRUD: Reading

Databases

Module 5: The SELECT keyword

CRUD: Reading with DDL

Since we’ll need to access data in our database, reading from it is obviously handy.

- To see what databases are in our entire MySQL server, use the command:

```
SHOW databases;
```

We can also see all tables, and their metadata, with the following commands:

```
SHOW tables;
DESCRIBE table_name;
```

Reading with DML: The SELECT statement

The SELECT statement, oddly enough, is used to select data from a database:

- You specify the table you want to select from by using the FROM keyword.
- * is a wildcard which will simply select everything.

```
SELECT * FROM table_name;
```

You can select specific columns by listing them instead:

```
SELECT field_1, field_4 FROM table_name;
```

SELECT DISTINCT

We can add a number of other keywords to our SELECT statement to make the data returned more specific to what we need:

- One way of doing this is by using the **DISTINCT** keyword, which will only return unique values of a particular column.

For instance, perhaps we want to see all the cities which GAME's customers live in:

- First, we select all records from the customer's table.
- If we use what we have before, then we'll have an address and postcode column, but no city.

Reading the GAME database

Outcome:

- Edit our table and use SELECT statements to get the specific information we need.

Steps:

10 minutes, solo

- Select all records from the customer table and follow this up by displaying a return set for name, age, and email address.
- Add a **city** field to the customer table and edit it so that two records have the same city value.
- Display all the unique city values for customers in our database.

Reading from the GAME database

```
SELECT * FROM customers;

SELECT name, age, email FROM customers;

ALTER TABLE customers ADD city varchar(40);
UPDATE customers SET city='Edinburgh';
UPDATE customers SET city='Manchester' WHERE age=25;

SELECT DISTINCT city FROM customers;
```

Conditional statements

Databases

Module 5: The SELECT keyword

The WHERE clause

As seen in some queries already, we can use WHERE to match specified criteria:

```
SELECT column_name FROM table_name WHERE expression;
```

```
SELECT name FROM customers WHERE name='Simon';
SELECT age FROM customers WHERE email='simon@nomis.co';
```

The WHERE clause can also use several different operators for comparisons:

- = (equal), != (not equal), < and >, <= and >=
- **BETWEEN:** Within an inclusive range.
- **LIKE:** Searching for a pattern.
- **[NOT] IN:** Specifying multiple possible values for a column.
- **IS [NOT] NULL:** Select everything where the specified field is(n't) null.

WHERE and our GAME database

Outcome:

- Find a specific set of records within the customer table.

Steps:

10 minutes, solo

› Find all records in the games table with an age rating of over 12.

› Do the same, but for all records of customers that live in either Edinburgh or London.

› Find all records in the customers table where the name contains an ‘s’.

WHERE statements

```
SELECT * FROM games WHERE age_rating>12;

SELECT * FROM customers WHERE city='Edinburgh' OR city='London’;

SELECT * FROM customers WHERE name LIKE '%s%';
```

SELECT: Ordering

The ORDER BY keyword allows us to filter our records more specifically.

- For instance, reordering our games table according to price would look like this:

```
SELECT title, price FROM games ORDER BY price;
```

- ORDER BY will always present in ascending order unless specified:

```
SELECT title, price FROM games ORDER BY price DESC;
```

- This will work even with more complex querying, for instance in our stock value example earlier:

```
SELECT title, quantity, price, quantity*price AS stock_value
FROM games
ORDER BY stock_value DESC;
```

SELECT: Limiting

We can use the LIMIT keyword alongside ORDER BY to see a specific snapshot of the data we want.

- By using LIMIT, we can see just a few records outputted rather than an entire set:

```
SELECT * FROM customers LIMIT 1;
```

- Very useful for answering questions like “what are the top 5 x in table y?”:

```
SELECT * FROM games ORDER BY price LIMIT 5;
```

- It’s also useful for getting a general idea of what the data looks like:

```
SELECT * FROM orders WHERE date_placed='2016-09-25' LIMIT 10;
```

Operations overview

	CREATE	READ	UPDATE	DELETE
DDL	CREATE	SHOW DESCRIBE	ALTER	DROP
DML	INSERT INTO	SELECT	UPDATE	DELETE

Summary

Databases: Module 5

Read schema information from a database using DDL:

› SHOW and DESCRIBE are the two DDL statements which expose schema information.

Read record information from a database using DML:

› SELECT, and its conditionals, are the go-to statements for reading data with DML.

Add conditionals to DML reads to refine results:

› Using WHERE, AS, ORDER BY, and LIMIT refines and reduces results down to the specific data we need.

› There are plenty more ways to combine these together, of course.

Thank you for listening

Any questions?