Learning SQL

May 13, 2023

Contents

Chapter 1

Creating and Populating a Database

1.1 Using the mysql Command-Line Tool

```
mysql -u root -p;
mysql -u root -p database_name;
```

1.2 MySQL Data Types

1.2.1 Character Data

Character data can be stored as either fixed-length or variable-length strings; the difference is that fixed-length strings are right-padded with spaces and always consume the same number of bytes, and variable-length strings are not right-padded with spaces and don't always consume the same number of bytes.

The maximum length for char columns is currently 255 bytes, whereas varchar columns can be up to 65,535 bytes. If you need to store longer strings (such as emails, XML documents, etc.), then you will want to use one of the text types (mediumtext and longtext).

> An exception is made in the use of varchar for Oracle Database. Oracle users should use the varchar2 type when defining variable-length character columns.

Character sets

You may choose to use a different character set for each character column in your database, and you can even store different character sets within the same table. To choose a character set other than the default when defining a column, simply name one of the supported character sets after the type definition.

```
varchar(20) character set latin1
```

With MySQL, you may also set the default character set for your entire database:

```
create database european_sales character set latin1;
```

Text data

If you need to store data that might exceed the 64 KB limit for varchar columns, you will need to use one of the text types.

When choosing to use one of the text types, you should be aware of the following:

- If the data being loaded into a text column exceeds the maximum size for that type, the data will be truncated.
- Trailing spaces will not be removed when data is loaded into the column.

Table 1.1: MySQL text types

Text type	Maximum number of bytes
tinytext	255
text	65,535
mediumtext	16,777,215
longtext	4,294,967,295

- When using text columns for sorting or grouping, only the first 1,024 bytes are used, although this limit may be increased if necessary.

1.3 Table Creation

1.4 Populating and Modifying Tables

1.4.1 Inserting Data

There are three main components to an insert statement:

- The name of the table into which to add the data
- The names of the columns in the table to be populated
- The values with which to populate the columns

Generating numeric key data

If you are running these statements in your database, you will first need to disable the foreign key constraint on the table, and then re-enable the constraints when finished. The progression of statements would be:

```
set foreign_key_checks=0;
```

Chapter 2

Query Primer

2.1 Query Clauses

Several components or clauses make up the select statement. While only one of them is mandatory when using MySQL (the select clause), you will usually include at least two or three of the six available clauses. Table 2.1 shows the different clauses and their purposes.

2.2 The select Clause

The job of the select clause is as follows:

> The select clause determines which of all possible columns should be included in the query's result set.

If you were limited to including only columns from the table or tables named in the from clause, things would be rather dull. However, you can spice things up in your select clause by including things such as:

- Literals, such as numbers or strings
- Expressions, such as transaction.amount * -1
- Built-in function calls, such as ROUND(transaction.amount, 2)
- User-defined function calls

If you only need to execute a built-in function or evaluate a simple expression, you can skip the from clause entirely.

Removing Duplicates

> Keep in mind that generating a distinct set of results requires the data to be sorted, which can be time consuming for large result sets. Don't fall into the trap of using distinct just to be sure there are no duplicates; instead, take the time to understand the data you are working with so that you will know whether duplicates are possible.

2.3 The from Clause

2.3.1 Tables

Four different types of tables meet this relaxed definition:

- Permanent tables (i.e., created using the create table statement)

Table 2.1: Query clauses Clause name Purpose

Clause name	Purpose
select	Determines which columns to include in the query's result set
from	Identifies the tables from which to retrieve data and how the tables should be joined
where	Filters out unwanted data
group by	Used to group rows together by common column values
having	Filters out unwanted groups
order by	Sorts the rows of the final result set by one or more columns

- Derived tables (i.e., rows returned by a subquery and held in memory)
- Temporary tables (i.e., volatile data held in memory)
- Virtual tables (i.e., created using the create view statement)

Temporary tables

These tables look just like permanent tables, but any data inserted into a temporary table will disappear at some point (generally at the end of a transaction or when your database session is closed).

Views

A view is a query that is stored in the data dictionary. It looks and acts like a table, but there is no data associated with a view (virtual table). When you issue a query against a view, your query is merged with the view definition to create a final query to be executed.

2.4 The where Clause

> The where clause is the mechanism for filtering out unwanted rows from your result set.

2.5 The order by Clause

> The order by clause is the mechanism for sorting your result set using either raw column data or expressions based on column data.

2.5.1 Sorting via Numeric Placeholders

If you are sorting using the columns in your select clause, you can opt to reference the columns by their position in the select clause rather than by name. This can be especially helpful if you are sorting on an expression.