:: DEVELOPER ZONE

MySQL Manual | 3.3.2 Creating a Table

Search the MySQL manual:

Go

- MySQL Manual

- 3 MySQL Tutorial
  - 3.3 Creating and Using a Database
    - 3.3.1 Creating and Selecting a Database
    - 3.3.2 Creating a Table
    - 3.3.3 Loading Data into a Table
    - 3.3.4 Retrieving Information from a Table

Additional languages

- French
- German
- Japanese
- Portuguese
- Russian

Additional formats

- PDF (A4)
- PDF (US letter)
- HTML, one page per chapter, tarball
- HTML, one page per chapter, ZIP
- HTML, all on one page, tarball
- HTML, all on one page, ZIP

Previous / Next / Up / Table of Contents

3.3.2 Creating a Table

Creating the database is the easy part, but at this point it's empty, as `SHOW TABLES` will tell you:

```
mysql> SHOW TABLES;
Empty set (0.00 sec)
```

The harder part is deciding what the structure of your database should be: what tables you will need and what columns will be in each of them.

You'll want a table that contains a record for each of your pets. This can be called the `pet` table, and it should contain, as a bare minimum, each animal's name. Because the name by itself is not very interesting, the table should contain other information. For example, if more than one person in your family keeps pets, you might want to list each animal's owner. You might also want to record some basic descriptive information such as species and sex.

How about age? That might be of interest, but it's not a good thing to store in a database. Age changes as time passes, which means you'd have to update your records often. Instead, it's better to store a fixed value such as date of birth. Then, whenever you need age, you can calculate it as the difference between the current date and the birth date. MySQL provides functions for doing date arithmetic, so this is not difficult. Storing birth date rather than age has other advantages, too:

- You can use the database for tasks such as generating reminders for upcoming pet birthdays. (If you think this type of query is somewhat silly, note that it is the same question you might ask in the context of a business database to identify clients to whom you'll soon need to send out birthday greetings, for that computer-assisted personal touch.)
- You can calculate age in relation to dates other than the current date. For example, if you store death date in the database, you can easily calculate how old a pet was when it died.

You can probably think of other types of information that would be useful in the `pet` table, but the ones identified so far are sufficient for now: name, owner, species, sex, birth, and death.

Use a `CREATE TABLE` statement to specify the layout of your table:

```
mysql> CREATE TABLE pet (name VARCHAR(20), owner VARCHAR(20),
    -> species VARCHAR(20), sex CHAR(1), birth DATE, death DATE);
```

`VARCHAR` is a good choice for the `name`, `owner`, and `species` columns because the column values will vary in length. The lengths of those columns need not all be the same, and need not be `20`. You can pick any length from `1` to `255`, whatever seems most reasonable to you. (If you make a poor choice and it turns out later that you need a longer field, MySQL provides an `ALTER TABLE` statement.)

Several types of values can be chosen to represent sex in animal records, such as `'m'` and `'f'`, or perhaps `'male'` and `'female'`. It's simplest to use the single characters `'m'` and `'f'`.

The use of the `DATE` data type for the `birth` and `death` columns is a fairly obvious choice.

Now that you have created a table, `SHOW TABLES` should produce some output:

```
mysql> SHOW TABLES;
+----------------------+
| Tables in menagerie |
+----------------------+
| pet                  |
+----------------------+
```

To verify that your table was created the way you expected, use a `DESCRIBE` statement:

```
mysql> DESCRIBE pet;
+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| name    | varchar(20) | YES  |     | NULL    |       |
| owner   | varchar(20) | YES  |     | NULL    |       |
| species | varchar(20) | YES  |     | NULL    |       |
| sex     | char(1)     | YES  |     | NULL    |       |
| birth   | date        | YES  |     | NULL    |       |
| death   | date        | YES  |     | NULL    |       |
+---------+-------------+------+-----+---------+-------+
```

You can use `DESCRIBE` any time, for example, if you forget the names of the columns in your table or what types they have.

Previous / Next / Up / Table of Contents

User Comments

Posted by Rizwan Omer on April 13 2003 1:41am [Delete] [Edit]

A Simple Example could be:

mysql> create table myTest(id int(3), name varchar(20));

where myTest is the name of the table to be created
id is a field of Integer type, with a width of 3,
name is a field of VarChar type, with a width of 20.

Hope it helps...Cheers!

Posted by Aljosja Beije on April 13 2003 8:53am [Delete] [Edit]

Rule one of database design: the primary key!

CREATE TABLE orders(ordernumber varchar(8) PRIMARY KEY, etc.);

Posted by Donald Axel on February 16 2004 4:24am [Delete] [Edit]

The commands
mysql-> show tables;
and
mysql-> describe pet;
are MySQL specific (not part of SQL-std.)

Add your own comment.
Top / Previous / Next / Up / Table of Contents