Theory: Basic INSERT statement

10 minutes 0 / 5 problems solved

You are getting more and more familiar with databases. Let's take it a step further! To use a database, you should know how to insert new records into a database table. In this topic, we'll show you how this can be done in SQL.

§1. Basic INSERT statement

You can insert a new record into a table with a simple query using **INSERT INTO statement**.

For example, let's add a record about a new customer into the table customers with columns *name* (VARCHAR(20)), *surname* (VARCHAR(20)), *zip_code* (INT) and *city* (VARCHAR(10)).

name	surname	zip_code	city
Tom	Black	11643	New York
Hermione	Smith	20599	Washington

Our new customer's name is Bobby, his surname is Ray, his ZIP code is 60601, and he lives in Chicago. The query below will insert this information into the table:

```
INSERT INTO customers (name, surname, zip_code, city) VALUES ('Bobby', 'Ray', 6060
1, 'Chicago');
```

As you can see, what you should do is write a list of values to be inserted after the keyword **VALUES**.

Once we have executed this query, our table *customers* will have a new row:

name	surname	zip_code	city
Tom	Black	11643	New York
Hermione	Smith	20599	Washington
Bobby	Ray	60601	Chicago

If you know the exact order of the columns in the table, you can use the shorter version of INSERT INTO query without specifying the column names.

In this case, our previous SQL query can be rewritten like this:

```
INSERT INTO customers VALUES ('Bobby', 'Ray', 60601, 'Chicago');
```

§2. Insert multiple rows

If you want to insert multiple rows, you don't have to add them one by one: you can add multiple rows simultaneously.

Let's add two more rows to the table *customers*:

```
INSERT INTO customers (name, surname, zip_code, city)

VALUES ('Mary', 'West', 75201, 'Dallas'), ('Steve', 'Palmer', 33107, 'Miami');
```

In this example, we wrote two comma-separated lists of values instead of just one.

§3. Insert into specified columns

Current topic:

Basic INSERT statement

Topic depends on:

✕ Basic CREATE statement

Topic is required for:

JDBC Statements

Table of contents:

Sometimes you have to insert a record without any information. Assume we have an empty table *cats* with columns *name* (VARCHAR(20)), *age* (INT) and *owner* (VARCHAR(40)). None of these columns have a default value.

name	age	owner

We want to add information about a three-year-old homeless cat Felix. Since Felix doesn't have an owner, we can skip this column in our INSERT INTO query.

```
INSERT INTO cats (name, age) VALUES ('Felix', 3);
```

As a result, the table *cats* will have one row:

name	age	owner
Felix	3	NULL

The *owner* column value for the first row will be NULL since we didn't specify it.

If a column has a default value and you skip this column during the insertion, its value will be set to default.

§4. Conclusion

Here is a template for a basic INSERT INTO statement:

```
INSERT INTO table_name (column_1, column_2,..., column_n) VALUES
(list_of_values_1) [, (list_of_values_2), ..., (list_of_values_m)];
```

This notation means that one tuple of values is always necessary, others are optional.

When you know the order of columns and want to insert values into all the columns, you can follow the shorter INSERT INTO statement template:

```
INSERT INTO table_name VALUES (value_1, value_2,..., value_n);
```

Report a typo

297 users liked this theory. **6** didn't like it. **What about you?**

Start practicing

Comments (4) Hints (0) Useful links (0) Show discussion