MySQL INSERT IGNORE Statement

Summary: in this tutorial, you will learn how to use the MySQL `INSERT IGNORE` statement to insert data into a table.

Introduction to MySQL INSERT IGNORE statement

When you use the `INSERT` (https://www.mysqltutorial.org/mysql-insert-statement.aspx) statement to add multiple rows to a table and if an error occurs during the processing, MySQL terminates the statement and returns an error. As the result, no rows are inserted into the table.

However, if you use the `INSERT IGNORE` statement, the rows with invalid data that cause the error are ignored and the rows with valid data are inserted into the table.

The syntax of the `INSERT IGNORE` statement is as follows:

```
INSERT IGNORE INTO table(column_list)
VALUES( value_list),
      ( value_list),
      ...
```

Note that the `IGNORE` clause is an extension of MySQL to the SQL standard.

MySQL INSERT IGNORE example

We will create a new table (https://www.mysqltutorial.org//mysql-create-table/) called `subscribers` for the demonstration.

```
CREATE TABLE subscribers (
    id INT PRIMARY KEY AUTO_INCREMENT,
    email VARCHAR(50) NOT NULL UNIQUE
);
```

The `UNIQUE` (https://www.mysqltutorial.org/mysql-unique-constraint/) constraint ensures that no duplicate email exists in the `email` column.

The following statement inserts a new row into the `subscribers` table:

```
INSERT INTO subscribers(email)
VALUES('john.doe@gmail.com');
```

It worked as expected.

Let’s execute another statement that inserts two rows into the `subscribers` table:

```
INSERT INTO subscribers(email)
VALUES('john.doe@gmail.com'),
      ('jane.smith@ibm.com');
```

It returns an error.

```
Error Code: 1062. Duplicate entry 'john.doe@gmail.com' for key 'email'
```

As indicated in the error message, the email `john.doe@gmail.com` violates the `UNIQUE` constraint.

However, if you use the `INSERT IGNORE` statement instead.

```
INSERT IGNORE INTO subscribers(email)
VALUES('john.doe@gmail.com'),
      ('jane.smith@ibm.com');
```

MySQL returned a message indicating that one row was inserted and the other row was ignored.

```
1 row(s) affected, 1 warning(s): 1062 Duplicate entry 'john.doe@gmail.com' for key 'email' R
```

To find the detail of the warning, you can use the `SHOW WARNINGS` command as shown below:

```
SHOW WARNINGS;
```

In conclusion, when you use the `INSERT IGNORE` statement, instead of issuing an error, MySQL issued a warning in case an error occurs.

If you query data from `subscribers` table, you will find that only one row was actually inserted and the row that causes the error was not.

MySQL INSERT IGNORE and STRICT mode

When the strict mode (https://dev.mysql.com/doc/refman/8.0/en/sql-mode.html#sql-mode-strict) is on, MySQL returns an error and aborts the `INSERT` statement if you try to insert invalid values into a table.

However, if you use the `INSERT IGNORE` statement, MySQL will issue a warning instead of an error. In addition, it will try to adjust the values to make them valid before adding the value to the table.

Consider the following example.

First, we create a new table (https://www.mysqltutorial.org/mysql-create-table/) named `tokens`:

```
CREATE TABLE tokens (
    s VARCHAR(6)
);
```

In this table, the column `s` accepts only string whose lengths are less than or equal to six.

Second, insert a string whose length is seven into the `tokens` table.

```
INSERT INTO tokens VALUES('abcdefg');
```

MySQL issued the following error because the strict mode is on.

```
Error Code: 1406. Data too long for column 's' at row 1
```

Third, use the `INSERT IGNORE` statement to insert the same string.

```
INSERT IGNORE INTO tokens VALUES('abcdefg');
```

MySQL truncated data before inserting it into the `tokens` table. In addition, it issues a warning.

In this tutorial, you have learned how to use the MySQL `INSERT IGNORE` statement to insert rows into a table and ignore error for rows that cause errors.