How to Insert Data into the Database

How to use the commit() method to insert data into the database

- INSERT, UPDATE, and DELETE statements work much like executing a SELECT statement in Python, but these statements don’t return a result set. Instead, they modify the data in the database.

- When you execute one of these SQL statements, the database is changed but it isn’t saved to the database. To save the changes, you must call the commit() method of the connection object. Otherwise, the changes are lost.

How to Insert Data into the Database

How to use the commit() method to insert data into the database

This example shows how to use an INSERT statement to add a new employee to the database.

```
name = "Sam"
username = "ssupport"
password = "mypassword"
email = "sam@vectacorp.com"
roleid = 2


with closing(conn.cursor()) as cursor:
    sql = "INSERT INTO employees (name, username, password, email, roleid)
        VALUES (?, ?, ?, ?, ?)"
    cursor.execute(sql, (name, username, password, email, roleid))
    conn.commit()
```

How to Insert Data into the Database

How to use the commit() method to insert data into the database

In the previous example, five variables are set for the new employee that will be added to the database including name, username, password, email, and roleid. Next, a with statement is used to open a cursor object named cursor. Inside the with statement, the code creates a string named sql that stores the INSERT statement. This statement uses question mark placeholders for five values. Then, the execute() method is called with a tuple argument that supplies the values for these placeholders. As a result, the INSERT statement that the database actually runs becomes:

```
INSERT INTO employees (name, username, password, email, roleid)
VALUES ("Sam", "ssupport", "mypassword", "sam@vectacorp.com", 2)
```

After this statement is executed, the commit() method is called to save the new row to the database.