انجام عملیات CRUD در MySQL و PostgreSQL (مشابه SQLite)

در این بخش یاد می‌گیریم که چگونه عملیات **ایجاد (CREATE)**، **خواندن (READ)**، **به‌روزرسانی (UPDATE)**، و **حذف (DELETE)** را در پایگاه داده **MySQL** و **PostgreSQL** انجام دهیم.

✅ **نکات کلیدی:**

- برای MySQL از `mysql-connector-python` یا `PyMySQL` استفاده می‌کنیم.
- برای PostgreSQL از `psycopg2` استفاده می‌کنیم.
- از دستورات **SQL استاندارد** برای اجرای کوئری‌ها استفاده خواهیم کرد.

---

۱. ایجاد جدول (`CREATE TABLE`)

📌 ایجاد جدول در MySQL و PostgreSQL

```
def create_table(conn):
    cursor = conn.cursor()
    cursor.execute('''
        CREATE TABLE IF NOT EXISTS users (
            id SERIAL PRIMARY KEY,  -- در MySQL به‌جای SERIAL از AUTO_INCREMENT استفاده کنید
            name VARCHAR(100) NOT NULL,
            email VARCHAR(100) UNIQUE NOT NULL,
            age INT
        )
    ''')
    conn.commit()
    print("✅ جدول users با موفقیت ایجاد شد!")
    cursor.close()
```

✅ **تفاوت‌ها:**

- در **MySQL** از `AUTO_INCREMENT` برای مقداردهی خودکار `id` استفاده کنید.
- در **PostgreSQL** از `SERIAL` برای مقداردهی خودکار `id` استفاده می‌شود.

---

۲. افزودن داده (`INSERT INTO`)

```
def insert_user(conn, name, email, age):
    cursor = conn.cursor()
    cursor.execute('''
        INSERT INTO users (name, email, age)
        VALUES (%s, %s, %s)
    ''', (name, email, age))
    conn.commit()
    print(f"✅ کاربر {name} اضافه شد!")
    cursor.close()
```

✅ **نکات:**

- از `s%` به‌جای مقداردهی مستقیم برای جلوگیری از **SQL Injection** استفاده شده است.

- در MySQL و PostgreSQL این روش امن و توصیه‌شده است.

📌 **مثال استفاده:**

```
insert_user(conn, "Ali Ahmadi", "ali@example.com", 30)
```

---

۳. خواندن داده‌ها (`SELECT`)

```
def get_users(conn):
    cursor = conn.cursor()
    cursor.execute("SELECT * FROM users")
    users = cursor.fetchall()

    print("👥 لیست کاربران:")
    for user in users:
        print(user)  # (id, name, email, age)

    cursor.close()
```

✅ **توضیح:**

- از `()fetchall` برای دریافت همه رکوردها استفاده شده است.
- می‌توان از `()fetchone` برای دریافت یک رکورد استفاده کرد.

📌 **مثال استفاده:**

```
get_users(conn)
```

---

۴. به‌روزرسانی داده‌ها (`UPDATE`)

```
def update_user_age(conn, user_id, new_age):
    cursor = conn.cursor()
    cursor.execute('''
        UPDATE users SET age = %s WHERE id = %s
    ''', (new_age, user_id))
    conn.commit()
    print(f"✅ سن کاربر {user_id} به {new_age} تغییر کرد!")
    cursor.close()
```

📌 **مثال استفاده:**

```
update_user_age(conn, 1, 35)
```

---

۵. حذف داده (`DELETE`)

```
def delete_user(conn, user_id):
    cursor = conn.cursor()
    cursor.execute('''
        DELETE FROM users WHERE id = %s
    ''', (user_id,))
    conn.commit()
    print(f"❌ کاربر {user_id} حذف شد!")
    cursor.close()
```

📌 مثال استفاده:

```
delete_user(conn, 1)
```

---

۶. اجرای کامل CRUD برای MySQL و PostgreSQL

📌 اتصال به MySQL

```
import mysql.connector

conn = mysql.connector.connect(
    host="localhost",
    user="root",
    password="password",
    database="test_db"
)

create_table(conn)
insert_user(conn, "Ali Ahmadi", "ali@example.com", 30)
get_users(conn)
update_user_age(conn, 1, 40)
delete_user(conn, 1)

conn.close()
```

📌 اتصال به PostgreSQL

```
import psycopg2

conn = psycopg2.connect(
    host="localhost",
    user="postgres",
    password="password",
    database="test_db"
)

create_table(conn)
insert_user(conn, "Ali Ahmadi", "ali@example.com", 30)
```

```
get_users(conn)
update_user_age(conn, 1, 40)
delete_user(conn, 1)

conn.close()
```

جمع‌بندی

✅ **ایجاد جدول (`CREATE TABLE`)** با استفاده از `SERIAL` در PostgreSQL و `AUTO_INCREMENT` در MySQL
✅ **افزودن داده (`INSERT INTO`)** با استفاده از `VALUES (%s, %s, %s)` برای جلوگیری از SQL Injection
✅ **خواندن داده (`SELECT`)** با `()fetchall` یا `()fetchone`
✅ **به‌روزرسانی (`UPDATE`)** و تغییر مقدار یک فیلد
✅ **حذف (`DELETE`)** بر اساس `id`

🚀 **در ادامه: استفاده از دستورات شرطی و توابع SQL در MySQL و PostgreSQL!**