

مدیریت تراکنش‌ها (Transactions) در PostgreSQL و MySQL

تراکنش‌ها در پایگاه داده تضمین می‌کنند که مجموعه‌ای از عملیات یا کاملاً انجام شوند (commit) یا در صورت خطا به حالت قبل برگردند (rollback). این قابلیت در PostgreSQL و MySQL بسیار مهم است، مخصوصاً هنگام اجرای چندین عملیات که باید به صورت اتمیک انجام شوند.

۱. تراکنش چیست؟

✓ تراکنش مجموعه‌ای از عملیات است که طبق اصل ACID اجرا می‌شود:

- Atomicity (اتمی بودن): عملیات به صورت کامل انجام می‌شود یا به حالت اولیه بازمی‌گردد.
- Consistency (یکپارچگی): پایگاه داده همیشه در یک وضعیت معتبر باقی می‌ماند.
- Isolation (ایزوله بودن): تراکنش‌ها روی یکدیگر تأثیر منفی ندارند.
- Durability (پایداری): داده‌ها پس از تأیید شدن ذخیره می‌شوند و از بین نمی‌روند.

۲. استفاده از commit () و rollback ()

📌 مراحل اجرای تراکنش در PostgreSQL و MySQL

۱. غیرفعال کردن auto-commit (به طور پیش فرض در PostgreSQL و MySQL فعال است).

۲. اجرای عملیات (INSERT, UPDATE, DELETE)

۳. ذخیره تغییرات با commit () (در صورت موفقیت آمیز بودن).

۴. برگشت تغییرات با rollback () (در صورت بروز خطا).

۳. اجرای تراکنش در MySQL

```
import mysql.connector

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

    # شروع تراکنش
    conn.start_transaction()

    # افزودن کاربران به جدول
    cursor.execute("INSERT INTO users (name, email, age) VALUES (%s, %s, %s)",
                  ("Ali Ahmadi", "ali@example.com", 30))
    cursor.execute("INSERT INTO users (name, email, age) VALUES (%s, %s, %s)",
                  ("Sara Karimi", "sara@example.com", 25))
```

```

# تأیید تراکنش
conn.commit()
print("✅ تراکنش با موفقیت انجام شد")

except mysql.connector.Error as e:
    # در صورت خطا، عملیات را لغو کن
    conn.rollback()
    print(f"❌ خطا در تراکنش: {e}")

finally:
    cursor.close()
    conn.close()

```

✅ نکات مهم:

- از `start_transaction()` برای شروع تراکنش استفاده شده است.
- اگر همه عملیات موفق باشند، `commit()` اجرا می‌شود.
- اگر خطایی رخ دهد، `rollback()` اجرا شده و تغییرات لغو می‌شوند.

۴. اجرای تراکنش در PostgreSQL

```

import psycopg2

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

    # auto-commit غیرفعال کردن
    conn.autocommit = False

    # اجرای چند عملیات در یک تراکنش
    cursor.execute("INSERT INTO users (name, email, age) VALUES (%s, %s, %s)",
                  ("Ali Ahmadi", "ali@example.com", 30))
    cursor.execute("INSERT INTO users (name, email, age) VALUES (%s, %s, %s)",
                  ("Sara Karimi", "sara@example.com", 25))

    # تأیید تغییرات
    conn.commit()
    print("✅ موفقیت‌آمیز بود PostgreSQL تراکنش در")

except psycopg2.Error as e:
    # در صورت بروز خطا، تراکنش لغو شود
    conn.rollback()
    print(f"❌ خطا در تراکنش PostgreSQL: {e}")

finally:
    cursor.close()

```

```
conn.close()
```

تفاوت‌های MySQL و PostgreSQL:

- در PostgreSQL، `autocommit = False` برای کنترل تراکنش‌ها استفاده می‌شود.
- در صورت بروز خطا، `rollback()` تمام عملیات را به حالت قبل بازمی‌گرداند.

۵. بررسی و جلوگیری از تراکنش‌های ناتمام


اگر تراکنش بدون `commit()` یا `rollback()` رها شود، پایگاه داده در حالت نامشخصی قرار می‌گیرد. برای جلوگیری از این مشکل:

- همیشه تراکنش را در `try-except-finally` مدیریت کنید تا در صورت خطا `rollback()` انجام شود.
- در PostgreSQL، اگر `autocommit = True` باشد، نیازی به `commit()` ندارید.
- در MySQL، بهتر است از `start_transaction()` برای تراکنش‌های پیچیده استفاده کنید.

۶. استفاده از WITH برای مدیریت خودکار تراکنش‌ها (PostgreSQL)

در PostgreSQL می‌توان از `with` برای اجرای امن‌تر تراکنش‌ها استفاده کرد:

```
import psycopg2

with psycopg2.connect(
    host="localhost",
    user="postgres",
    password="password",
    database="test_db"
) as conn:
    with conn.cursor() as cursor:
        cursor.execute("INSERT INTO users (name, email, age) VALUES (%s, %s, %s)",
            ("Ali Ahmadi", "ali@example.com", 30))
        cursor.execute("INSERT INTO users (name, email, age) VALUES (%s, %s, %s)",
            ("Sara Karimi", "sara@example.com", 25))
    # می‌شود تراکنش به صورت خودکار commit می‌شود
    print("تراکنش با موفقیت انجام شد )
```

مزایا: 

- نیازی به `commit()` و `rollback()` نیست.
- اگر خطایی رخ دهد، تغییرات به طور خودکار لغو می‌شوند.

- ✅ استفاده از `commit()` برای ذخیره تغییرات و `rollback()` برای بازگردانی در صورت خطا
- ✅ غیرفعال کردن `autocommit` برای مدیریت تراکنش‌ها در PostgreSQL
- ✅ استفاده از `try-except-finally` برای جلوگیری از تراکنش‌های ناتمام
- ✅ در PostgreSQL می‌توان از `with` برای مدیریت خودکار تراکنش‌ها استفاده کرد

🚀 در ادامه: کار با داده‌های JSON در MySQL و PostgreSQL!