Podstawy JDBC oraz SQL.

Paweł Frankowski

1. Cel ćwiczenia

Zapoznanie się z podstawami JDBC oraz SQL. Porównanie czasów wykonania zapytań dla róznej wielkości rekordów z indeksami oraz bez.

2. Wiadomości teoretyczne

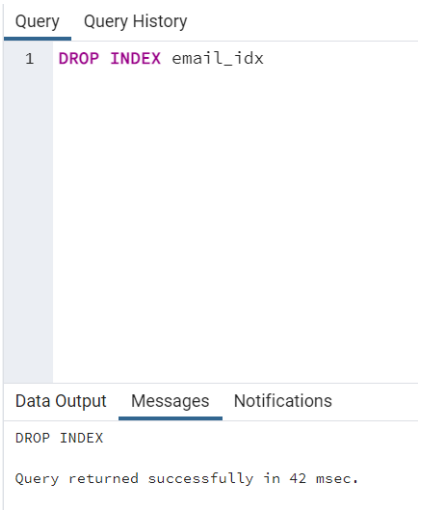
JDBC – (Java Database Connectivity) Interfejs programowania aplikacji (API), który umożliwia połączenie aplikacji Java z bazami danych oraz wykonywanie na nich operacji.
SQL – (Structered Query Language) jest językiem zapytań służącym do komunikacji z bazami danych relacyjncych.
pgAdmin – Narzędzie do zarządzania bazami danych opartym na systemie postgreSQL.
postgreSQL – System zarządzania bazami danych relacyjnych (RDBMS).
Transakcje – zbiór operacji, które zmieniają stan systemu. Wykonanie wyłącznie wszystkich kroków wchodzących w skład transakcji może być uznane za sukces (z ang. commit).

3. Przebieg ćwiczenia

3.1 Pomiary zapytań SELECT z indeksami oraz bez

```
Executing query: SELECT email FROM public."dataThousand" WHERE email LIKE 'p%'
connected to DB
Time taken: 580ms
```

kod 1. czas SELECT dla 1000 rekordów z indeksem



kod 2. Komenda SQL za pomocą której usuwamy index z kolumny.

```
Executing query: SELECT email FROM public."dataThousand" WHERE email LIKE 'p%'
connected to DB
Time taken: 742ms

Process finished with exit code 0
```

kod 3. czas SELECT dla 1000 rekordów bez indeksu

3.2 Wykonanie kilku zapytań w jednej transakcji

```java
String ktosUpdate = String.format(UPDATE_QUERY, 1000, "ktos");
String ktos1Update = String.format(UPDATE_QUERY, 2000, "ktos1");

QueryExecutor.executeUpdate(ktos1Update);
QueryExecutor.executeUpdate(ktosUpdate);
```

kod 4. 2 odzielne zapytania

Zagrożeniem może być, gdy w jednym zapytaniu wystąpi wyjątek, a drugi się wykona. W tym celu należy

```java
public static void executeUpdateTransaction(List<String> queries, Connection connection){
    try {
        connection.setAutoCommit(false);
        queries.forEach(query -> executeUpdate(query));
        connection.commit();
        connection.close();
    } catch (SQLException e) {
        throw new RuntimeException("Error: ",e);
    }
}
```

kod 5. Jedna transakcja z kilkoma zapytaniami.

4. Wnioski

Kiedy mamy do dyspozycji dużą tabelę powinniśmy używać index'u. Zwiększa to wydajność bazy danych.Im większa ilość rekordów tym więcej czasu zajmuje wykonanie zapytań.

Transakcje - domyślnie w JDBC każde pojedyńcze zapytanie zostaje wykonane w jednej transakcji (auto commit). Do zarządzania transakcjami służy klasa Connection.

Za pomocą metody setAutoCommit(false) możemy wyłączyć automatyczny commit. Commit() służy do wykonania transakcji bazodanowej a rollback() do wycofania bieżącej transakcji.

W przypadku, gdy chcemy przesłać pieniądze z konta A do konta B, powinniśmy wykonać 2 zapytania w jednej transakcji.