批量处理允许将相关的SQL语句分组到批处理中，并通过对数据库的一次调用来提交它们，一次执行完成与数据库之间的交互。

一次向数据库发送多个SQL语句时，可以减少通信开销，从而提高性能。

- 不需要JDBC驱动程序来支持此功能。应该使用DatabaseMetaData.supportsBatchUpdates()方法来确定目标数据库是否支持批量更新处理。如果JDBC驱动程序支持此功能，该方法将返回true。
- Statement，PreparedStatement和CallableStatement的addBatch()方法用于将单个语句添加到批处理。 executeBatch()用于执行组成批量的所有语句。
- executeBatch()返回一个整数数组，数组的每个元素表示相应更新语句的更新计数。
- 就像将批处理语句添加到处理中一样，可以使用clearBatch()方法删除它们。此方法将删除所有使用addBatch()方法添加的语句。 但是，无法指定选择某个要删除的语句。

使用Statement对象进行批处理

以下是使用Statement对象的批处理的典型步骤序列 -

- 使用createStatement()方法创建Statement对象。
- 使用setAutoCommit()将自动提交设置为false。
- 使用addBatch()方法在创建的Statement对象上添加SQL语句到批处理中。
- 在创建的Statement对象上使用executeBatch()方法执行所有SQL语句。
- 最后，使用commit()方法提交所有更改。

实例

以下代码片段提供了使用Statement对象的批量更新示例 -

```
// Create statement object
Statement stmt = conn.createStatement();

// Set auto-commit to false
conn.setAutoCommit(false);

// Create SQL statement
String SQL = "INSERT INTO Employees (id, first, last, age) " +
             "VALUES(200,'Ruby', 'Yang', 30)";
// Add above SQL statement in the batch.
stmt.addBatch(SQL);

// Create one more SQL statement
String SQL = "INSERT INTO Employees (id, first, last, age) " +
             "VALUES(201,'Java', 'Lee', 35)";
// Add above SQL statement in the batch.
stmt.addBatch(SQL);
```

```
// Create one more SQL statement
String SQL = "UPDATE Employees SET age = 35 " +
             "WHERE id = 100";
// Add above SQL statement in the batch.
stmt.addBatch(SQL);

// Create an int[] to hold returned values
int[] count = stmt.executeBatch();

//Explicitly commit statements to apply changes
conn.commit();
```

使用PrepareStatement对象进行批处理

以下是使用PrepareStatement对象进行批处理的典型步骤顺序 -

- 使用占位符创建SQL语句。
- 使用prepareStatement()方法创建PrepareStatement对象。
- 使用setAutoCommit()将自动提交设置为false。
- 使用addBatch()方法在创建的Statement对象上添加SQL语句到批处理中。
- 在创建的Statement对象上使用executeBatch()方法执行所有SQL语句。
- 最后，使用commit()方法提交所有更改。

以下代码段提供了使用PreparedStatement对象进行批量更新的示例 -

```
// Create SQL statement
String SQL = "INSERT INTO Employees (id, first, last, age) " +
             "VALUES(?, ?, ?, ?)";

// Create PrepareStatement object
PreparedStatemen pstmt = conn.prepareStatement(SQL);

//Set auto-commit to false
conn.setAutoCommit(false);

// Set the variables
pstmt.setInt( 1, 400 );
pstmt.setString( 2, "JDBC" );
pstmt.setString( 3, "Li" );
pstmt.setInt( 4, 33 );
// Add it to the batch
pstmt.addBatch();

// Set the variables
pstmt.setInt( 1, 401 );
pstmt.setString( 2, "CSharp" );
```

```
pstmt.setString( 3, "Liang" );
pstmt.setInt( 4, 31 );
// Add it to the batch
pstmt.addBatch();

//add more batches
.
.
.
.
//Create an int[] to hold returned values
int[] count = stmt.executeBatch();

//Explicitly commit statements to apply changes
conn.commit();
```