Practical Java Programming

# Java Common Using feature

## Agenda

- JDBC overview: Driver and Working procedure
- JDBC Connection
- JDBC Data access using string query
- JDBC ResultSet using for update
- JDBC Data access with parameter
- JDBC batch statement

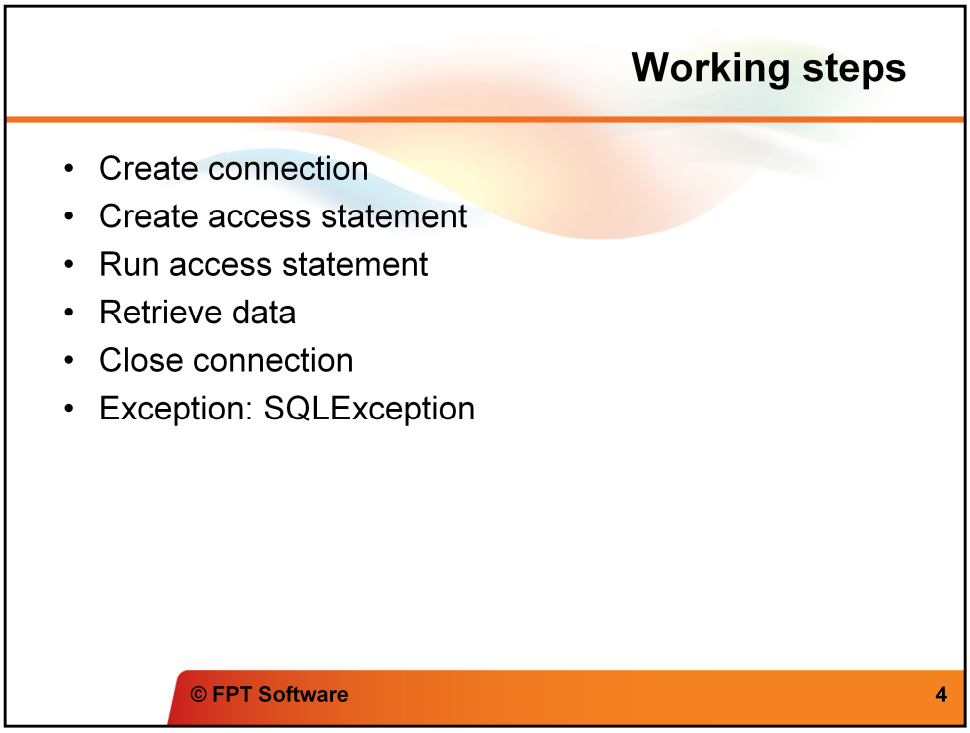
Prepare a DB for practical:

Person (ID, Name, Age),

Student(ID, PersonID, FSoftTool, RAnalyzeNote, DesignNote, CodeAndTesting, ProgrammingLanguageNote)

Professor(ID, PersonID, Seniority, Field);

# JDBC OVERVIEW




Prepare a DB for practical:

Person (ID, Name, Age),

Student(ID, PersonID, FSoftTool, RAnalyzeNote, DesignNote, CodeAndTesting, ProgrammingLanguageNote)

Professor(ID, PersonID, Seniority, Field);

## JDBC Drivers

- Driver URL

```
jdbc:oracle:thin:@<HOST>:<PORT>:<DB>
jdbc:mysql://<HOST>:<PORT>/<DB>
```

- Driver class

```
oracle.jdbc.driver.OracleDrive
org.gjt.mm.mysql.Driver
```

# JDBC CONNECTION

## JDBC Connection

```
import java.sql.DriverManager;
import java.sql.Connection;

// register the driver class with DriverManager
Class.forName(JDBCDriverClass);

// Open a new connection
Connection connection = DriverManager.
  getConnection(JDBCDriverURL, userName, pw);

// Do smth here
…
// Close connection
connection.close();
```

# JDBC SIMPLE DATA ACCESS

## JDBC data access statements

```
// Statement creation
Statement statement = connection.createStatement();
// Query string
String query = <SQLQuery>; // CRUD
// for retrieve data
ResultSet result = statement.executeQuery(query);
while (result.next()){
   result.getString(<ColName>);
   result.getInt(<ColName>);
   result.getFloat(<ColName>);
}
// for data modification: insert, update, delete
int nbUpdated = statement.executeUpdate(query);
```

Ask for CRUD with the student information and DB prepared in page 29

# JDBC RESULTSET

## JDBC ResultSet update preparation

```
// for use with ResultSet only
// No "previous" method using, no update
connection.createStatement(
    ResultSet.TYPE_FORWARD_ONLY,
    ResultSet.CONCUR_READ_ONLY);

// with "previous" method using, update
connection.createStatement(
    ResultSet.TYPE_SCROLL_SENSITIVE,
    ResultSet.CONCUR_UPDATABLE);
```




TYPE_SCROLL_INSENSITIVE, TYPE_SCROLL_SENSITIVE
CONCUR_UPDATABLE

## JDBC Update using ResultSet

```
ResultSet rs =
  statement.executeQuery(query);
…
// for update
rs.updateBoolean(1, false); // change the
  first column
rs.updateInt("Age", 25); // change the
  column named "Age"
rs.updateRow();

// to delete
rs.deleteRow();
```

# JDBC WITH PARAMETER

# JDBC with parameter

```
// in string query
String query = "INSERT INTO Person " +
               "VALUES (" + <name> + ", "
                          <age> + … + ")";
// using statementPrepare
String query = "INSERT INTO Person " +
               "VALUES (?, ?)"
PreparedStatement statement = connect.prepareStatement(query);
connect.setAutoCommit(false);
statement.setString(1, "Titi");
statement.setInt(2, 25);
statement.executeQuery();         // insert 1
statement.setString(1, "Tata");
statement.setInt(2, 28);
statement.executeQuery();         // Insert 2
connect.commit();
connect.setAutoCommit(true);
```

# JDBC “BATCH” STATEMENT

## JDBC “batch” processing

```
connect.setAutoCommit(false);
// replace executeQuery by addBatch
…
statement.setString(1, “Titi”);
statement.setInt(2, 25);
statement.addBatch();        // Insert 1
statement.setString(1, “Tata”);
statement.setInt(2, 28);
statement.addBatch();        // Insert 2
// then call batch processing statement
statement.executeBatch();
// also applied for normal statement (not prepared one)
connect.commit();
connect.setAutoCommit(true);
```

## JDBC Batch with string query

```
connect.setAutoCommit(false);
Statement statement = connect.createStatement();
statement.addBatch(<Insert query>);
statement.addBatch(<Insert query>);
statement.addBatch(<Update query>);
statement.addBatch(<Delete query>);
int[] updateCounts = statement.executeBatch();
connect.commit();
statement.close();
connect.setAutoCommit(true);
```

## Lesson summary

- JDBC Driver: depend on database technology
- JDBC Working procedure: create connection, access data, close connection
- JDBC Connection: Connection class
- JDBC Data access using string query: Statement class
- JDBC ResultSet using for update: rs.updateInt…
- JDBC Data access with parameter: in query string or by PreparedStatement
- JDBC batch statement: st.addBatch, st.executeBatch

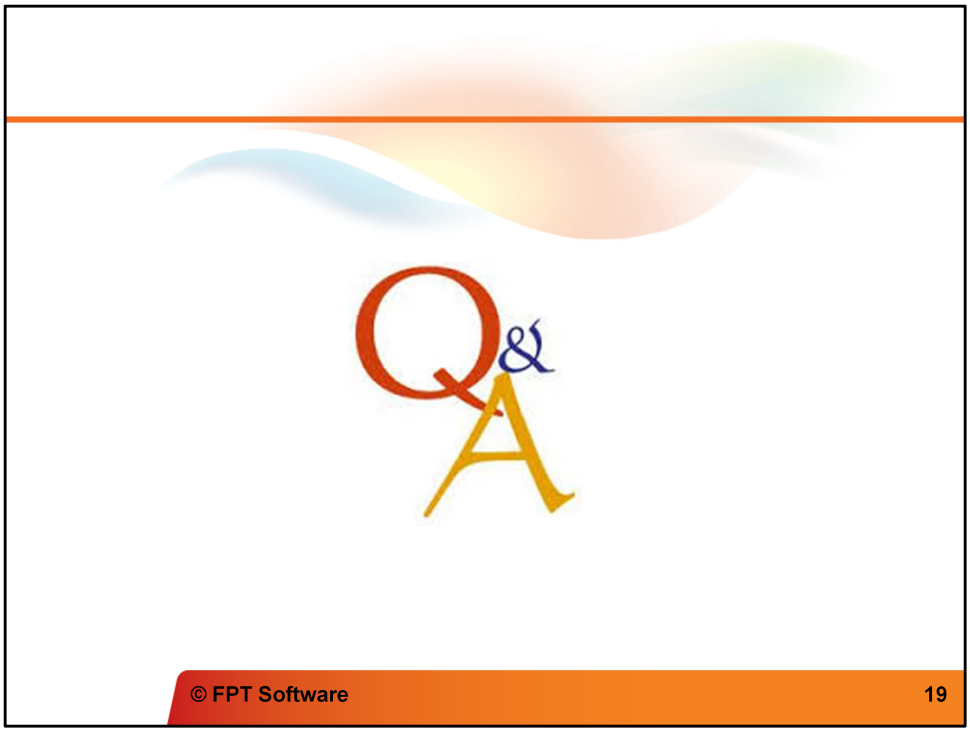



Prepare a DB for practical:

Person (ID, Name, Age),

Student(ID, PersonID, FSoftTool, RAnalyzeNote, DesignNote, CodeAndTesting, ProgrammingLanguageNote)

Professor(ID, PersonID, Seniority, Field);


Q&A
© FPT Software
19