# آموزش اتصال جدول به دیتابیس

اکنون که به دیتابیس متصل شده اید، گام بعدی دست یابی به جدول در دیتابیس می باشد. به این خاطر نیاز به اجرای یک SQL Statementو سپس اصلاح همه ی ردیف ها و ستون هایی دارید که بازگردانده شده اند.

برای اجرای یک SQL Statement روی جدول خود، یک آبجکت Statement تنظیم کنید. بنابراین این خط را به بالای کد خود اضافه کنید:

```
import java.sql.Statement;
```

در بخش try از بلوک try … catch خط زیر را وارد کنید ) آن را درست در زیر خط Connection وارد کنید:(.

```
Statement stmt = con.createStatement( );
```

در اینجا در حال ایجاد یک آبجکت Statement به نام stmt هستیم. این آبجکت به یک آبجکت Connection به همراه متود createStatment نیاز دارد.

ما نیاز به یک عبارت SQL برای آبجکت Statement هم داریم. بنابراین این خط را به کد خود اضافه کنید:

```
String SQL = "SELECT * FROM Workers";
```

در عبارات بالا همه ی رکوردها را از جدول دیتابیس به نام Workers انتخاب کنید.

می توانیم این SQL query را به متود از آبجکت Statement به نام executeQueryانتقال دهیم. آبجکت Statement کار جمع آوری همه ی رکورهایی را مه با query ما هماهنگ هستند، آغاز می کند.

به هرحال متود executeQuery همه ی رکوردها را در موردی به نام ResultSet بازمی گرداند.

قبل از اینکه این موترد را توضیح دهیم، خط زیر را به بالای کد خود اضافه کنید:

```
import java.sql.ResultSet;
```

بنابراین ResultSet به عنوان rs مطرح می شود. این آبجکت تمام رکوردهای جدول دیتابیس را در خود حفظ می کند. قبل از اینکه پیش روی کنیم، در اینجا توضیحاتی در مورد ResultSets را مشاهده می کنید.

### ResultSetدر جاوا:

یک ResultSet راهی برای ذخیره و اصلاح رکوردهای بازگردانده شده از یک SQL queryمی باشد ResultSet .ها سه نوع می باشند. نوعی که استفاده می کنید، بستگی به کاری دارد که می خواهید با داده انجام دهید:

1.آیا فقط می خواهید از طریق رکوردها پیش بروید، از ابتدا تا انتها؟

2.آیا می خواهید از طریق رکوردها به جلو و عقب بروید و همچنین تغییرات ایجاد شده در رکوردها را کشف کنید؟

3.آیا می خواهید از طریق رکوردها به جلو و عقب بروید اما به خاطر تغییرات ایجاد شده در رکوردها اذیت نشوید؟

در لیست بالا به نامTYPE_FORWARD_ONLY ResultSet ، عدد 1 را تایپ کنید .عدد در لیست یک TYPE_SCROLL_SENSITIVE ResultSetمی باشد. گزینه ی سوم از ResultSetبا عنوان TYPE_SCROLL_INSENSITIVEمطرح می شود.

نوع ResultSet بین پرانتزهای createStement قرار می گیرد:

```
Statement stmt = con.createStatement( );
```

از آنجایی که پرانتزها را خالی گذاشته ایم، RecordSetپیش فرض را دریافت می کنیم که TYPE_FORWARD_ONLYمی باشد. در بخش بعدی از یک نوع دیگر استفاده خواهیم کرد. اما از آنها به شکل زیر استفاده کنید:

```
Statement stmt = con.createStatement( RecordSet.TYPE_SCROLL_SENSITIVE );
```

بنابراین ابتدا لغت RecordSet را تایپ کنید. بعد از یک نقطه(dot) ، نوع RecordSet مورد استفاده را تایپ کنید.

به هرحال این مسئله در همین جا تمام نمی شود. اگر می خواهید از TYPE_SCROLL_SENSITIVEیا TYPE_SCROLL_INSENSITIVEاستفاده کنید، نیاز به تعیین این مسئله نیز می باشد که آیا ResultSet از نوع Read Onlyاست یا قابل آپدیت شدن می باشد. این کار را با در ثابت داخلی شامل CONCUR_READ_ONLYو CONCUR_UPDATABLEانجام می دهید. مجددا این دو نیز بعد از لغت RecordSet قرار می گیرند:

```
ResultSet.CONCUR_READ_ONLY
ResultSet.CONCUR_UPDATABLE
```

این امر منجر به ایجاد یک کد طولانی می شود:

```
Statement stmt = con.createStatement( RecordSet.TYPE_SCROLL_SENSITIVE,
ResultSet.CONCUR_UPDATABLE);
```

مسئله ی دیگری که باید در مورد ResultSets به آن عادت کنیم، موردی به نام Cursor می باشد. یک Cursor در واقع یک نشانگر به یک ردیف از جدول می باشد. وقتی که رکوردها را در یک ResultSet بارگذاری می کنید، Cursor درست به قبل از اولین ردیف در جدول اشاره می کند. سپس از متودهایی برای اصلاح Cursorاستفاده می کنید. اما منظور تشخیص یک ردیف خاص در جدول می باشد.

## استفاده از ResultSet:

وقتی که همه ی رکوردها را در یک Results set (مجموعه از نتایج) دارید، متودهایی وجود دارند که می توانید برای اصلاح رکوردهای خود استفاده کنید .در اینجا متودهایی را مشاهده می کنید که اغلب استفاده می کنید:

نشانگر را در جدول شما به ردیف بعدی حرکت می دهد. اگر در جدول ردیف دیگری وجود نداشته باشد، مقدار False گزارش داده خواهد شد.

| | |
|---|---|
| next | نشانگر را در جدول شما به ردیف بعدی حرکت می دهد. اگر در جدول ردیف دیگری وجود نداشته باشد، مقدار False گزارش داده خواهد شد. |
| Previous | نشانگر را در جدول یک ردیف به قبل بازمی گردانیم. اگر ردیف دیگری در جدول وجود نداشته باشه، مقدار False گزارش داده می شود. |
| First | نشانگر را به اولین ردیف در جدول حرکت می دهد. |
| Last | نشانگر را به آخرین ردیف در جدول حرکت می دهد. |
| * | نشانگر را به یک ردیف خاص در جدول حرکت می دهد. بنابراین (۵) absolute نشانگر را به ردیف شماره ی ۵ در جدول حرکت می دهد. |

ResultSetدارای متودهایی نیز می باشد که می توانید برای تشخیص یک ستون خاص (field) در یک ردیف استفاده کنید. می توانید این کار را یا با استفاده از نام ستون و یا با استفاده از شماره ایندکس آن انجام دهید. برای جدول Workers ما چهار ستون تنظیم کرده ایم، که دارای نام های زیر می باشند .ID, First_Name, Last_Name, and Job_Title : بنابراین شماره های ایندکس عبارتند از 1، 2، 3و 4.

ستون ID را برای نگهداری مقادیر صحیح (Integer) تنظیم کردیم. متودی که برای دریافت مفادیر صحیح در یک ستون انجام می دهید، getIntمی باشد:

```
int id_col = rs.getInt("ID");
```

در اینجا یک متغیر صحیح به نام id_col را تنظیم کرده ایم. سپس از متود getIntاز آبجکت ResultSet استفاده می کنیم که rs نامیده می شود. بین پرانتزها نام ستون را داریم که می توانستیم به جای آن از عدد ایندکس استفاده کنیم:

```
int id_col = rs.getInt(1);
```

دقت داشته باشید که عدد ایندکس دارای علامت نقل قول نمی باشد اما نام این نمادها را دارد.

برای سه ستون دیگر در جدول دیتابیس، آنها را تنظیم کردیم تا Strings (رشته ها) را حفظ کنند. بنابراین نیاز به متود getStringداریم:

```
String first_name = rs.getString("First_Name");
```

یا می توانستیم از عدد ایندکس استفاده کنیم:

از آنجایی که ResultSet Cursor درست به قبل از اولین رکورد در هنگام بارگذاری داده اشاره دارد، لازم است که از متود بعدی برای جابه جایی به اولین ردیف استفاده کنیم. کد زیر اولین رکورد را از جدول دریافت می کند:

```
rs.next( );
int id_col = rs.getInt("ID");
String first_name = rs.getString("First_Name");
String last_name = rs.getString("Last_Name");
String job = rs.getString("Job_Title");
```

دقت کنید که rs.nextدر ابتدای کد قرار می گیرد. این برنامه نشانگر را به اولین رکورد در جدول حرکت می دهد.

شما می توانید برای نمایش رکورد در پنجره ی Output یک خط چاپی به کد خود اضافه کنید:

```
System.out.println( id_col + " " + first_name + " " + last_name + " " + job
);
```

اکنون کد شما باید مشابه زیر باشد ( خط چاپی را کمی تطبیق دادهایم زیرا کمی بلند می باشد.):

```
try {
    String host = "jdbc:derby://localhost:1527/Employees";
    String uName = "admin";
    String uPass = "admin";
    Connection con = DriverManager.getConnection(host, uName, uPass);

    Statement stmt = con.createStatement();
    String sql = "SELECT * FROM Workers";
    ResultSet rs = stmt.executeQuery(sql);

    rs.next();
    int id_col = rs.getInt("ID");
    String first_name = rs.getString("First_Name");
    String last_name = rs.getString("Last_Name");
    String job = rs.getString("Job_Title");

    String p = id_col + " " + first_name + " " + last_name + " " + job;
    System.out.println(p);
}
catch ( SQLException err ) {
    System.out.println( err.getMessage( ) );
}
```

اگر می خواهید تمام رکوردهای جدول را بررسی کنید، می توانید از یک loop استفاده کنید. از آنجایی که متود بعدی true یا false را بازمی گرداند، می توانید از آن به عنوان شرطی برای یک while loop استفاده کنید:

```
while ( rs.next( ) ) {
}
```

در بین پرانتزهای while می توانیم rs.next را مشاهده کنیم. این امر تازمانی که نشانگر از آخرین رکورد در جدول عبور نکرده باشد، درست خواهد بود: اگر عبور کرده باشد، rs.nextمقدار false را گزارش می دهد و while loop خاتمه خواهد یافت. با استفاده ازrs.next ، نشانگر نیز همراه یک رکورد در زمان جابجا خواهد شد. در اینجا همان کد بالا را مشاهده می کنید، اما در حالیکه از while loop استفاده می کند. کد خود را برای هماهنگی تغییر دهید:

```
try {
 String host = "jdbc:derby://localhost:1527/Employees";
 String uName = "admin";
 String uPass = "admin";
 Connection con = DriverManager.getConnection(host, uName, uPass);

 Statement stmt = con.createStatement();
 String sql = "SELECT * FROM Workers";
 ResultSet rs = stmt.executeQuery(sql);

 while (rs.next()) {
     int id_col = rs.getInt("ID");
     String first_name = rs.getString("First_Name");
     String last_name = rs.getString("Last_Name");
     String job = rs.getString("Job_Title");

     String p = id_col + " " + first_name + " " + last_name + " " + job;
     System.out.println(p);
 }

 }
 catch ( SQLException err ) {
     System.out.println( err.getMessage( ) );
 }
}
```

وقتی کد بالا را اجرا می کنید، پنجره ی Output باید مورد زیر را نمایش دهد:

```
run:
1 Helen James IT Manager
2 Eric Khan Programmer
3 Tommy Lee Systems Analyst
4 Priyanka Collins Programmer
BUILD SUCCESSFUL (total time: 2 seconds)
```

اکنون که دیدگاهی در مورد چگونگی اتصال به جدول دیتابیس و نمایش رکوردها را دارید، ما ادامه خواهیم داد و با استفاده از فرم ها و دکمه ها یک برنامه ی پیچیده تر برای ورود به رکوردها خواهیم نوشت.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

http://www.tahlildadeh.com/