Working with Data from a PreparedStatement

Kevin Jones

@kevinrjones

What's in This Module

- Processing data using a ResultSet
- Modifying data
- Setting null values
- Getting data by column name and position

The ResultSet

Execute the query to get a ResultSet

Has a 'cursor'

Call 'next' to move the cursor

- Returns true if there is a row to process
- false otherwise

Get values from the rows

```
var sql = "select name, capacity from venues";
try (PreparedStatement ps =
        conn.prepareStatement(sql)) {
  var rs = ps.executeQuery();
  while(rs.next()) {
    var name = rs.getString("name");
    var capacity = rs.getInt("capacity");
          System.out.println(name +
                    " has capacity " +
                    capacity);
    }
}
```

- **Create query**
- **Create the PreparedStatement**
- **Execute the query**
- **Move and check the cursor**
- **Get the values by name**
- **Use the values**

ResultSet Cursor

Table: venues		
Id	**Name**	**Capacity**
1	The Arena	100
2	The Bowl	150
3	The Garage	200

ResultSet Cursor

Initial position →

Table: venues		
Id	**Name**	**Capacity**
1	The Arena	100
2	The Bowl	150
3	The Garage	200

ResultSet Cursor

rs.next() true →

Table: venues		
Id	**Name**	**Capacity**
1	**The Arena**	**100**
2	**The Bowl**	**150**
3	**The Garage**	**200**

ResultSet Cursor

rs.next() true

Table: venues		
Id	**Name**	**Capacity**
1	**The Arena**	**100**
2	**The Bowl**	**150**
3	**The Garage**	**200**

ResultSet Cursor

rs.next() true →

Table: venues		
Id	**Name**	**Capacity**
1	**The Arena**	**100**
2	**The Bowl**	**150**
3	**The Garage**	**200**

ResultSet Cursor

Table: venues		
Id	**Name**	**Capacity**
1	**The Arena**	**100**
2	**The Bowl**	**150**
3	**The Garage**	**200**

rs.next() false ⟶

Can Also Get the Data Using Column Numbers

```
var sql = "select name, capacity from venues";

try (PreparedStatement ps = conn.prepareStatement(sql)) {
  var rs = ps.executeQuery();
  while(rs.next()) {
    var name = rs.getString(1);
    var capacity = rs.getInt(2);
        System.out.println(name + " has capacity " + capacity);
    }
}
```

Suppose You Only Want to Read One Row from a Table

Maybe a count or an aggregate function

Can use 'if(rs.next())'

```
var sql = "select count(*) from venues";
try (PreparedStatement ps =
     conn.prepareStatement(sql)) {
  var rs = ps.executeQuery();
  if(rs.next()) {
    var numberOfVenues = rs.getInt(1);
    System.out.println(
       "Number of venues is: " +
                  numberOfVenues);
    }
}
```

- Create query
- Create the PreparedStatement
- Execute the query
- Move and check the cursor
- Get the value by column number
- Use the value

Using the Column Name

```
var sql = "select count(*) as count from venues";

try (PreparedStatement ps = conn.prepareStatement(sql)) {

    var rs = ps.executeQuery();


    if(rs.next()) {

        var numberOfVenues = rs.getInt("count");

        System.out.println("Number of venues is: " + numberOfVenues);

    }

}
```

Dos and Don'ts

Check the return value of rs.next()

Access any data with first calling rs.next()

Use an invalid column number

Use an invalid column name

resultSet Get Methods

Method	Parameter Type
getBoolean	boolean
getDouble	double
getInt	int
getLong	long
getObject	Object
getString	String

Using GetObject

```
String name = "";

int capacity = 0;

while(rs.next()) {

    var nameField = rs.getObject("name");

    var capacityField = rs.getObject("capacity");

    if(nameField instanceof String) { name = (String) nameField; }

    if(capacityField instanceof Integer) {capacity = (int) capacityField; }

    System.out.println(name + " has capacity " + capacity);

}
```

Binding Parameters in a Select

As shown previously can also bind parameters in the SELECT statement

```
var sql = "select name, capacity from venues where
capacity > ?";

try (PreparedStatement ps = conn.prepareStatement(sql))
{

  ps.setInt(1, 120);

  var rs = ps.executeQuery();

  while(rs.next()) {

      var name = rs.getString("name");

      var capacity = rs.getInt("capacity");

      System.out.println(name +

                        " has capacity " +

                        capacity);

    }

}
```

- **Create query**
- **Create the PreparedStatement**
- **Bind the value**
- **Execute the query**
- **Move and check the cursor**
- **Get the values**
- **Use the values**

Can Also Bind Nulls

```
var sql = "insert into Acts (name, recordlabel) values(?, ?)";
try (PreparedStatement
            ps = conn.prepareStatement(sql)) {
    ps.setString(1, name);
    if(recordLabel == null)
        ps.setNull(2, Types.CHAR);
    else
        ps.setString(2, recordLabel);
    return ps.executeUpdate();
}
```

Closing Resources

Like connections and prepared statements result sets must be closed

Use try with resources

Note that closing a prepared statement will also close any associated result sets

Summary

Executing queries with a PrepareStatement returns a ResultSet

ResultSets have a cursor

Must move the cursor before accessing the data

Access the data by column, either indexed (1-based) or name

Up Next:
Working with a CallableStatement