JDBC – ResultSet

ResultSet and its attributes

- ResultSet object contains the records/rows returned by query execution
- Each record in a ResultSet contains same number of columns
- ResultSet object maintains a cursor that points to a record/row

IN	India
AU	Australia

RESUL
TSET

- ResultSet has the following attributes
  - Type
  - Concurrency
  - Holdability

ResultSet Types

- Type determines characteristic and abilities of the ResultSet as described below

TYPE_FORWARD_ONLY	• ResultSet can only be navigated forward • Cursor cannot be moved backward • **Default** type of ResultSet
TYPE_SCROLL_INSENSITIVE	• ResultSet can be navigated both forward and backward (Scrollable) • ResultSet is insensitive to changes in the underlying data source while it is open • You can jump to a absolute position or a position relative to the current position
TYPE_SCROLL_SENSITIVE	• ResultSet can be navigated both forward and backward (Scrollable) • ResultSet is sensitive to changes in the underlying data source while it is open • You can jump to a absolute position or a position relative to the current position

ResultSet Concurrency

- Concurrency determines whether the ResultSet can be updated, or only read

CONCUR_READ_ONLY	• ResultSet can only be Read • **Default** concurrency of ResultSet
CONCUR_UPDATABLE	• ResultSet can be both read and updated

ResultSet Holdability

- Holdability determines if a ResultSet is closed when the commit() method of the underlying connection is called

CLOSE_CURSORS_AT_COMMIT	• ResultSet instances are closed when commit() is called on the connection that created the ResultSet
HOLD_CURSORS_OVER_COMMIT	• ResultSet is kept open when commit() is called on the connection that created the ResultSet.

Setting ResultSet attributes

- The ResultSet attribute's are set while creating the Statement objects
- Connection Interface methods for Statement creation are used to set ResultSet attributes

```
createStatement() // defaults are set
createStatement(int type, int concurrency)
createStatement(int type, int concurrency, int holdability)
```

- To make a ResultSet read only, scrollable and insensitive, create the statement as below

```
Statement stmt = conn.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
ResultSet.CONCUR_READ_ONLY);
```

- To make a ResultSet updatable, scrollable, sensitive and close on commit, create

```
Statement stmt = conn.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,
ResultSet.CONCUR_UPDATABLE, ResultSet.CLOSE_CURSORS_AT_COMMIT);
```

Setting ResultSet attributes

- ResultSet attributes can also be set while creating the PreparedStatement and Callable Statement objects
- Connection Interface methods to set ResultSet attributes
  - While creating PreparedStatement

```
prepareStatement(String sql)
prepareStatement(String sql, int type, int concurrency)
prepareStatement(String sql, int type, int concurrency, int holdability)
```

  - While creating CallableStatement

```
prepareCall(String sql)
prepareCall(String sql, int type, int concurrency)
prepareCall(String sql, int type, int concurrency, int holdability)
```

Methods in ResultSet Interface

- Methods of ResultSet Interface can be divided into following three categories :
  - Navigational methods - Used to move the cursor to a row in the Resultset
  - Retrieval methods - Used to retrieve the data from the current row
  - Update methods - Used to insert/update/delete the data in current row of ResultSet. Updates done in the ResultSet can be transferred to the underlying database

ResultSet – Navigational Methods

```
next() : boolean - moves the cursor forward by one row
previous() : boolean     - moves the cursor backwards by one row
first() : boolean        - positions the cursor to the first row in the ResultSet
last() : boolean - positions the cursor to the last row in the ResultSet
beforeFirst():void       - positions the cursor before the first row of ResultSet
afterLast() :void        - positions the cursor after the last row of ResultSet

Note: All the above methods except next() throw SQLException, if called on a
ResultSet of type TYPE_FORWARD_ONLY
```

```
getRow() : int           - Returns an int with the current position of the cursor
isFirst() : boolean      - Returns true if cursor is on the first row
isLast() : boolean       - Returns true if cursor is on the last row
isBeforeFirst() : boolean - Returns true if cursor is before the first row
isAfterLast() : boolean   - Returns true if cursor is after the last row
```

ResultSet – Navigational Methods

```
relative(int rows) : moves the cursor relative to its current position

Example: Consider a ResultSet object "rs"
         rs.relative(4) : moves the cursor 4 rows ahead of the current position
         rs.relative(-2) – moves the cursor 2 rows previous to the current position
```

```
absolute(int rows)        : positions the cursor to the given row number

Example: Consider a ResultSet object "rs"
         rs.absolute(30) – moves the cursor to the 3oth row
         rs.absolute(-5) – move the cursor to the 5th row from the end of the Resultset
                             In a resultset of 50 rows, cursor will be moved to 46th row
```

Note: relative(..) and absolute(..) methods throw SQLException, if called on a resultset of type TYPE_FORWARD_ONLY

ResultSet – Retrieval Methods

- To retrieve data from a ResultSet, the cursor has to be first positioned on the row
- After positioning, the column data can be retrieved using getXXX methods
- Every column datatype in a table has a corresponding get method
- getXXX methods take column name(String) or column index(int) as parameter
  - To retrieve data from column of datatype varchar following method can be used
  - getString(String columnName) or getString(int columnIndex)

```
Consider ResultSet 'rs' with 1st column COUNTRY_ID and 2nd column COUNTRY_NAME
After cursor positioning, value in COUNTRY_NAME column can be retrieved as below

String countryName = rs.getString(COUNTRY_NAME);

                or

String countryName = rs.getString(2);
```

rs

IN	India
AU	Australia

ResultSet – Retrieval Methods

- More retrieval methods in ResultSet

```
getShort(..)      : short
getInt(…)         : int
getFloat(..)      : float
getDouble(..)     : double
getLong(..)       : long
getDate(..)       : java.sql.Date
getTime(..)       : java.sql.Time
getTimestamp(..) : java.sql.Timestamp
getBlob(..)       : java.sql.Blob
getClob(..)       : java.sql.Clob
..

Refer Java API documentation for the complete list of retrieval methods
```

ResultSet – Update Methods

- Updates to the table can be also made by updating values in the ResultSet
- For updates, ResultSet should be defined with concurrency CONCUR_UPDATABLE
- Updating rows in a table through ResultSet is a two step process
  - Move the cursor to the row and update column values using updateXXX methods
    - (updateString(..), updateFloat(..), updateDate(..) and so on)
  - Update the changes made in ResultSet row to the Table row using updateRow() method

rs

IN	India
AU	**Astralia**

```
rs.absolute(2);
rs.updateString("NAME","Australia")
```

IN	India
AU	**Australia**

```
rs.updateRow()
```

ID *char*	NAME *varchar*	**REGION_ID** *Number*
BE	Belgium	1
IN	India	3
AU	Australia	3

ResultSet – Update Methods

updateRow()
Updates the database with the new contents of the current row of ResultSet

cancelRowUpdates()
Cancels the updates made to the current row in this ResultSet

deleteRow()
Deletes the current row from ResultSet and from the database

moveToInsertRow()
Moves the cursor to the insert row.

insertRow()
inserts the contents of the insert row into ResultSet and into the database

RowSet

- RowSet objects holds tabular data like ResultSet
- RowSet Interface is derived from the ResultSet interface and therefore share its capabilities

- RowSet adds following capabilities to ResultSet
  - Functions as java bean component with standard set of properties and an event notification mechanism
  - Add Scrollability and Updatability

- Advantages
  - It is easy and flexible to use
  - It is Scrollable and Updatable by default

Types of RowSet

- Connected RowSet
  - Makes a connection to DBMS and maintains that connection throughout its life span
    - JdbcRowSet

- Disconnected RowSet
  - Makes a connection to a DBMS only to read in data or to write data back to the data source
  - After reading data from or writing data to its data source, the RowSet object disconnects from it
    - CachedRowSet
    - WebRowSet
    - FilteredRowSet
    - JoinRowSet

JdbcRowSet Demo

```
JdbcRowSet rs = RowSetProvider.newFactory().createJdbcRowSet();
rs.setUrl("jdbc:oracle:thin:@localhost:1521:xe");
rs.setUsername("HR");
rs.setPassword("HR");

rs.setCommand("select * from countries");
rs.execute();

while(rs.next()){
    String id = rs.getString("COUNTRY_ID");
    String name = rs.getString("COUNTRY_NAME");

    System.out.println(id + "  "+ name);
}
```