Oracle JDBC

- JDBC an API used for database connectivity
- Creates Portable Applications
- Basic Steps to develop JDBC Application
  - Import JDBC classes (java.sql.*).
  - Load JDBC drivers
  - Connect and Interact with database
  - Disconnect from database

Oracle JDBC

- DriverManager provides basic services to manage set of JDBC drivers
- Connection object sends queries to database server after a connection is set up
- JDBC provides following three classes for sending SQL statements to server
  - *Statement* SQL statements without parameters
  - *PreparedStatement* SQL statements to be executed multiple times with different parameters
  - *CallableStatement* Used for stored procedures

Oracle JDBC

- SQL query can be executed using any of the objects.
  (Statement,PreparedStatement,CallableStatement)
- Syntax (Statement Object)

Public abstract ResultSet executeQuery(String sql) throws SQLException

- Syntax (PreparedStatement,CallableStatement Object)

Public abstract ResultSet executeQuery() throws SQLException

- Method executes SQL statement that returns ResultSet object (ResultSet maintains cursor pointing to its current row of data.)

Oracle JDBC (Example)

```
Import java.sql.*;
Import java.io;
Class simple{
    public static void main(String[] args) throws Exception{
      Connection conn=null;
      try{
            String conStr = "jdbc:oracle:thin:@oracle.cs.purdue.edu:1521:orb";
            DriverManager.registerDriver(new oracle.jdbc.driver.OracleDriver());
            conn =   DriverManager.getConnection(conStr,”username”,”passwd");
            Statement cursor = conn.createStatement();     // Connection Est.
             ResultSet rset = cursor.executeQuery(“Select* from table_name”);
             while(rset.next()){
                    System.out.println(“Printing column value ”+rset.getString(1));
                }
        }Catch(ClassNotFoundException e){}
         cursor.close();
         conn.close();
   }
}
```

XML Parsing

- W3C provides specifications for following parsers

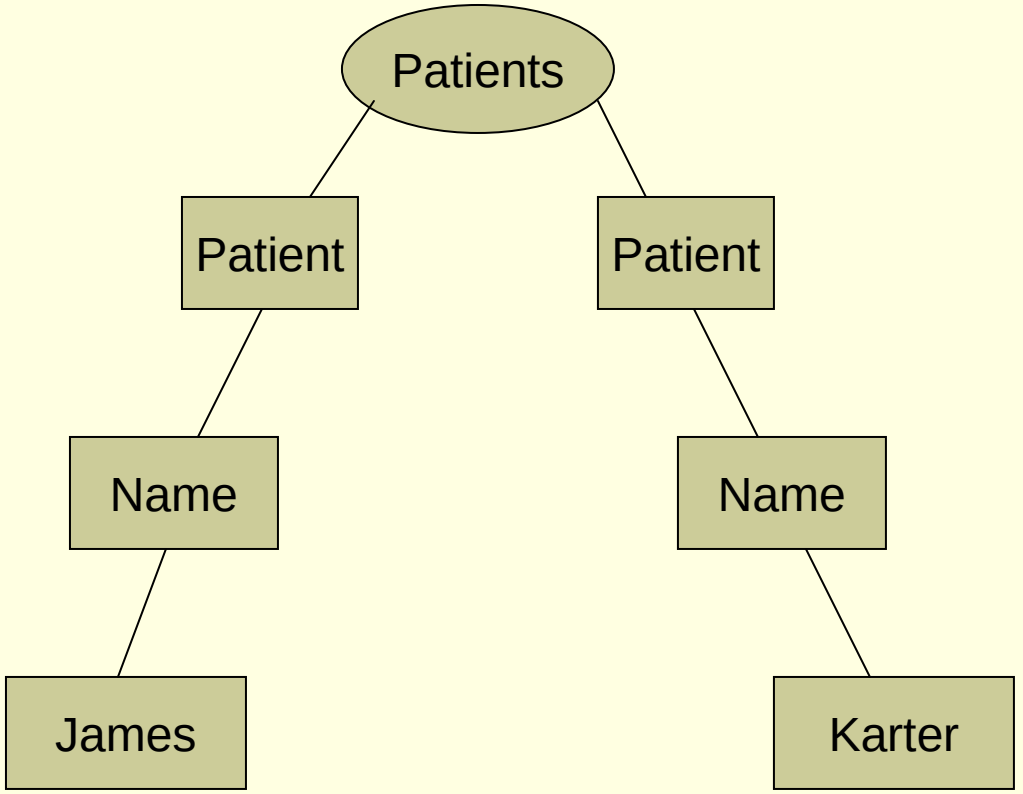
  SAX (Event Based)

  DOM (Tree Based)

- Oracle provide XML parser package In java called oracle.xml.parser.v2

DOM Tree

```
<?xml version="1.0"?>
<Patients>
<Patient>
<Name>James</Name>
</Patient>
<Patient>
<Name>Karter</Name>
</Patient>
</Patients>
```



DOM Parsing

- **DOMParser()**
- **parse(URL url)**
- **XMLDocument getDocument()**
- **XMLDocument contains**
  - **NodeList getElementsByTagName(String tagName)**
- **NodeList contains**
  - **Node item(i);**
  - **int getLength();**
- **XMLNode**
  - **Node getFirstChild()**
  - **Node getNodeValue()**
  - **Node getNodeName()**

DOM Parsing

```
DOMParser parser  = new DOMParser();
parser.parse(URL url);
/* Get the patient XML document */
XMLDocument xmlFile = parser.getDocument();
NodeList list1 =    xmlFile.getElementsByTagName("CreatedAfter");
int len = list1.getLength();
String CreatedAfter=list1.item(0).getFirstChild().getNodeValue();
```

Project3 Part 2

- Project Documents

  privacyPolicy Schema

  authPolicy Schema

  patient Schema

Project3 Part 2

- Create Following Tables
- Patient Document Table

  JprTable(JpatientRecord XMLType)

- PrivacyPolicy Document Table

  JprivacyPolicyTable(Jprivacy XMLType)

- AuthPolicy Document Table

  JauthTable(Jauth XMLType)

Project3 Part2

- JGetNumAuthorizedRecords(p_id)

  number of records physician(p_id) is authorized
- JGetMostAuthorizedRecords()

  which physician authorized to view max number of records
- JGetPermittedPhysician(p_id)

  id of physician permitted to view record of patient (p_id)
- JGetAllowedRecords(p_id,d_id)

  records id's of patient(p_id) a physician(d_id) is both authorized & permitted

References

- [1] Database Management Systems by Ramakrishnan and Gehrke

Thank You