TA Section

JDBC
JDBC

- Java API to access database
  1. Connect to a data source
  2. Send queries and update statements
  3. Retrieve and process results
First need to load JDBC driver

- **SQL Server Driver**
  sqljdbc4.jar

- **Postgres Driver**
  postgresql-8.4-701.jdbc4.jar

- Put on class path, then tell Java to load it

  ```java
  Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
  Class.forName("org.postgresql.Driver");
  ```
JDBC Example

Connection con = DriverManager.getConnection("jdbc:sqlserver://iisqlsrv:database=imdb_new", "myLogin", "myPassword");

Statement stmt = con.createStatement();

ResultSet rs = stmt.executeQuery("SELECT a, b, c FROM Table1");

while (rs.next()) {
    int x = rs.getInt("a");
    String s = rs.getString("b");
    float f = rs.getFloat("c");
}
Class.forName
    ("com.microsoft.sqlserver.jdbc.SQLServerDriver");

Connection con = DriverManager.getConnection
    ("jdbc:sqlserver://iisqlsrv:database=imdb_new",
    "myLogin", "myPassword");

PreparedStatement pstmt = con.prepareStatement
    ("SELECT lname FROM persons WHERE id = ?");

pstmt.setInt(1, 34);
ResultSet rs = stmt.executeQuery();

while (rs.next()) {
    String s = rs.getString("lname");
}

rs.close();
pstmt.close();
con.close();
Class.forName
("com.microsoft.sqlserver.jdbc.SQLServerDriver");

Connection con = null;

try {
    con = DriverManager.getConnection( ... );

    ...

} catch (Exception e) {
    e.printStackTrace();
} finally {
    con.close();
}
Prepared Statements

```
PreparedStatement pstmt = con.prepareStatement
    ("SELECT lname FROM persons WHERE id = ?");

... pstmt.setInt(1, 34);
ResultSet = pstmt.executeQuery();
...

pstmt.setInt(1, 63);
ResultSet = pstmt.executeQuery();
...

Statement stmt = con.createStatement();
ResultSet rs = stmt.executeQuery
    ("SELECT a, b, c FROM Table1");
```
Prepared Statements

- No need to worry about quotes ', “

```java
PreparedStatement pstmt = con.prepareStatement
  ("SELECT website FROM shops
      WHERE name = ? OR owner = ?");
...
pstmt.setString(1, "George’s");
pstmt.setString(2, "Oh "wow"!");
...
Statement stmt = con.createStatement();
ResultSet rs = stmt.executeQuery
  ("SELECT website FROM shops
      WHERE name = 'George’s' OR ...");
```
Transactions

String s1 = "BEGIN TRANSACTION READ ONLY";
String s2 = "BEGIN TRANSACTION READ WRITE";
String s3 = "COMMIT TRANSACTION";
String s4 = "ROLLBACK TRANSACTION";

PreparedStatement p1 = con.prepareStatement(s1);
PreparedStatement p2 = con.prepareStatement(s2);
PreparedStatement p3 = con.prepareStatement(s3);
PreparedStatement p4 = con.prepareStatement(s4);

... 

p1.executeUpdate();
...

if (ok) p3.executeUpdate();
else p4.executeUpdate();
Project 2

• Movie Rental Business
  – Movies from imdb (iisqlsrv - sql server)
  – Customer Information from personal database (local - postgres)
Starting Up Postgres

• Set your path variable (to find postgres)
• Create the data file (**initdb**)
  – On Z: (you created it only once)
  – On C: (it will be deleted: you re-created it every time)
• Start the server (**pg_ctl**)
• Create a database (**createdb**)
• Start the client (**psql**)

Dan Suciu -- 444 Spring 2010
Tunneling

```
ssh -L 1433:iisqlsrv.cs.washington.edu:1433
     attu.cs.washington.edu

static String connUrl =

    "jdbc:sqlserver://127.0.0.1;database=imdb_new;";

Instructions on how to set up tunneling
http://www.cs.washington.edu/education/courses/cs
   e444/CurrentQtr/tunneling-instructions.html
```
Project 2

Let’s Dive In!
Connections Expensive

• To open/close connections very expensive
• Also use resources on database server
Connection Pooling

• Fixed pool of say 30 connections
• Trick: Wrap Connection object
• getConnection() waits until connection becomes available
• close() returns Connection to pool (but does not close connection!)
• Many subtleties, dozens of products
That’s It
JDBC Architecture
Two-tier model

Java Application

DBMS

Client Machine
DBMS-proprietary protocol
Database server
JDBC Architecture
Three-tier model

- Application, Browser
- Application Server
- JDBC
- DBMS

- Client Machine
- RMI, SOAP, REST, HTTP
- Server (business logic)
- DBMS-proprietary protocol
- Database server

connection pooling
distributed transactions
...

...
JDBC Architecture
Three-tier model

application, browser

application server

hibernate

jdbc

dbms

client machine

rmi, soap, rest, http

server (business logic)
persistence & query service

dbms-proprietary protocol

database server

connection pooling

distributed transactions

...