Now that you have a database, what can you do with it?

Rachel Pottinger
Guest Lecture
CSE 490i
February 1, 2001

What the database gives you (from the last lecture)

- Storage management
- Abstract data model
- High level query and data manipulation language
- Efficient query processing
- Transaction processing
- Resiliency: recovery from crashes
- Different views of the data, security
- Interface with programming languages

What you need to do:

What the Administrator (you) needs to do:

- Design your storage/tables

You need to design your database carefully

What users/crawlers need to be able to do:

- Look at the data you have
- Add in new data

You need to interface with the query language

Database design tips

- Keep clean abstractions
  - Don't keep your data where it doesn't belong

  e.g., information about a file's location should not be stored with its artist
- When in doubt, create a separate ID as a key
- Think about the queries that will be run
- Design the database twice, put it in SQL Server once

The Impedance Mismatch Problem

The host language manipulates variables, values, pointers

SQL manipulates relations.

There is no construct in the host language for manipulating relations.

Why not use only one language?

- Forgetting SQL: "we can quickly dispense with this idea" [Ullman & Widom, pg. 363].
- SQL cannot do everything that the host language can do.

Why XML?

This is your result on html:

…<table><tr><td>Destiny's Child</td><td>Independent Woman</td><td>http://www.xyz.com/a.mp3</td></tr>…

This is your result on XML:

```
<MP3><Artist>Destiny's Child</Artist>
  <Title>Independent Woman</Title>
  <URL>http://www.xyz.com/a.mp3</URL>
</MP3>
```

Any questions?

Some language choices for interfacing with the database:

- JDBC – Java's method
- ODBC – C++'s method
- OLE DB – MS's method
- ADO – layer on top of OLE; uses Active X/Com objects
- RDO – Different MS layer on top of OLE

Interfacing SQL Server with Java

- Create the SQL Server database
- Add an ODBC connection
  - Okay, so I lied ODBC is also good for making a database appear local
  - We need a JDBC driver to access any database
- Create your Java code

Adding the ODBC source (1/2)

To make a system DSN in Windows 2000: Start → Settings → Control Panel → Administrative Tools → Data Sources (ODBC)

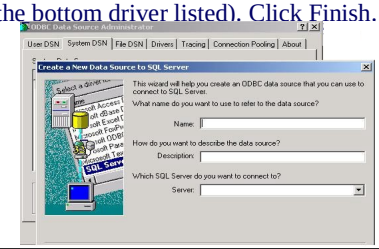
Note 1: You must have registered the Server in SQL server before doing this

Note 2: This currently doesn't work in 232; try 329

Note 3: Check with your TA on which authentication to use

Adding the ODBC source (2/2)

- Select the System DSN tab
- Click add and choose SQL server (usually the bottom driver listed). Click Finish.



Writing the code - overview

```
public static void main(String[] args){
  try{
        String url="jdbc:odbc:myDSN";
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        Connection c =
                DriverManager.getConnection(url,[userID],[password]);
        java.sql.Statement s= c.createStatement();
        java.sql.ResultSet rs;
        rs = s.executeQuery("Select * from [csepclab\\hongyu].Office");
        java.sql.ResultSetMetaData md = rs.getMetaData();
        while (rs.next()){
                for (int i = 1; i <= md.getColumnCount();i++){
                        area.append(rs.getString(i) + " | "); }}
s.executeUpdate("Insert into Paper values(13,1999,'Interesting')");
rs.close();
}catch….
```

Opening the connection

```
String url="jdbc:odbc:myDSN";
//First, we give the name of the database; the form is
//jdbc (primary connection type)
//odbc (secondary connection type)
//database name (the ODBC source you created)
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
//register the name of the driver you're going to use
Connection c =
   DriverManager.getConnection(url,[userID],[password]);
//Finally, we can open the connection
```

Show me the data!

```
java.sql.Statement s= c.createStatement();
//Create a way to talk to the database
java.sql.ResultSet rs; // Where to put the data
rs = s.executeQuery("Select * from [csepclab\\hongyu].Office");
//Execute the query
java.sql.ResultSetMetaData md = rs.getMetaData();
//Get the number of columns, etc.
while (rs.next()){  //go through the rows
    for (int i = 1; i <= md.getColumnCount();i++){ // # of columns
    area.append(rs.getString(i) + " | "); }}  // Get actual data
s.executeUpdate("Insert into Paper values(13,1999,'Interesting')");
//You can do updates, too
rs.close(); //Close the connection when done
```

Caveats

- JDBC 2.0 offers many nifty features with cursors

  You can't use them through Visual J++ ☹
- SQL Server doesn't do "Intersect"

Give me more!

- Sun's API for the sql package
  http://java.sun.com/j2se/1.3/docs/api/index.html
- See Sun's tutorial on JDBC
  http://java.sun.com/tutorial
- 444 web page
  http://www.cs.washington.edu/education/courses/cse444/CurrentQtr/help/index.htm