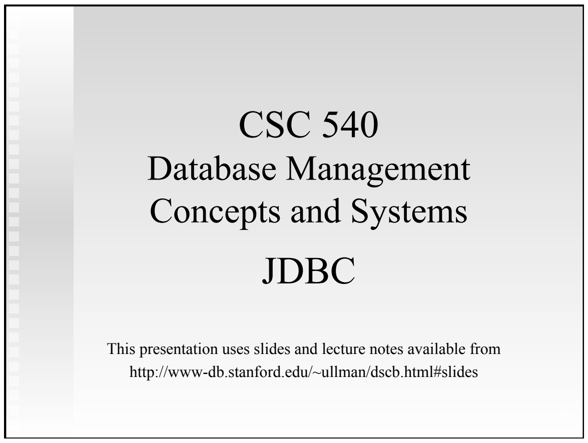
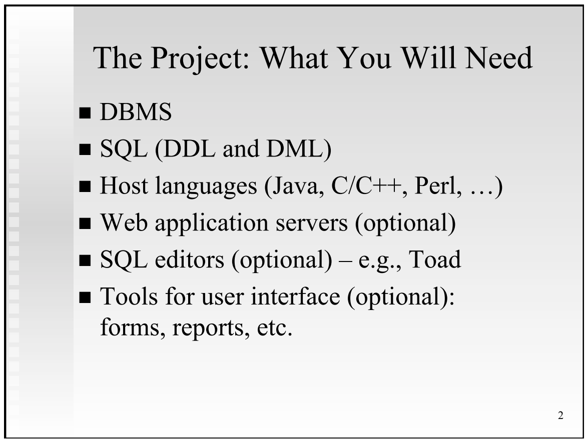


CSC 540 Database Management Concepts and Systems

JDBC

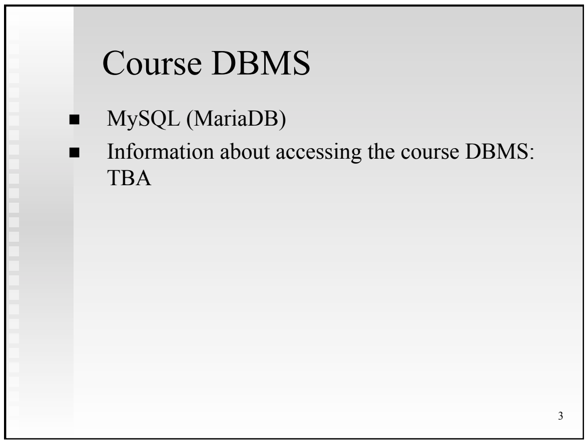
This presentation uses slides and lecture notes available from http://www-db.stanford.edu/~ullman/dscb.html#slides



The Project: What You Will Need

- DBMS
- SQL (DDL and DML)
- Host languages (Java, C/C++, Perl, …)
- Web application servers (optional)
- SQL editors (optional) – e.g., Toad
- Tools for user interface (optional): forms, reports, etc.

2



Course DBMS

- MySQL (MariaDB)
- Information about accessing the course DBMS: TBA

3

SQL

- A data-definition and data-manipulation language
- Can be used for ad-hoc queries on (relational) databases
  - Generic SQL interface: users sit at terminals and ask queries on database
- Can be used in programs in some *host* language
  - Programs access (relational) database by “calls” to SQL statements

4

Connecting SQL to Host Language

- Embedded SQL
  - Special SQL statements (not part of host language)
  - Preprocessor transforms SQL statements into host-language code
- Call-level interfaces:
  - SQL/CLI (adaptation of ODBC)
  - JDBC: links Java programs to databases

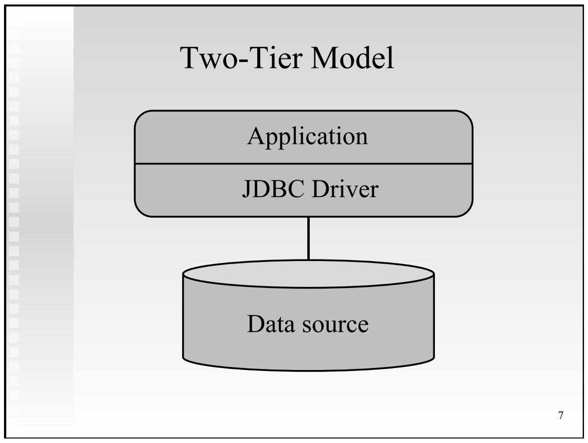
5

JDBC Basics

- Read the tutorial at

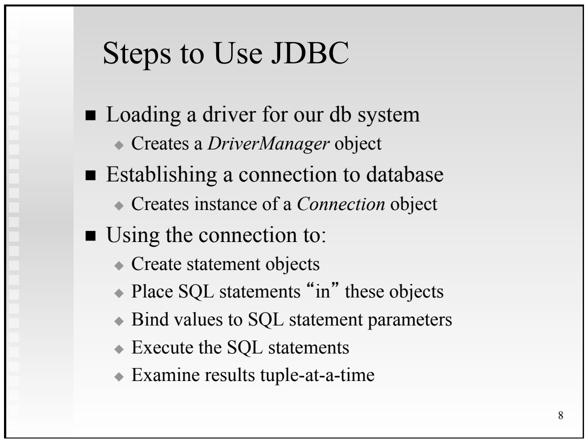
http://java.sun.com/docs/books/tutorial/jdbc/basics/

6



Two-Tier Model

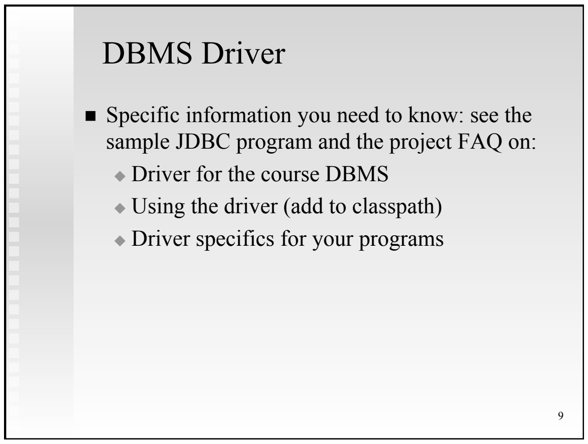
7



Steps to Use JDBC

- Loading a driver for our db system
  - Creates a *DriverManager* object
- Establishing a connection to database
  - Creates instance of a *Connection* object
- Using the connection to:
  - Create statement objects
  - Place SQL statements “in” these objects
  - Bind values to SQL statement parameters
  - Execute the SQL statements
  - Examine results tuple-at-a-time

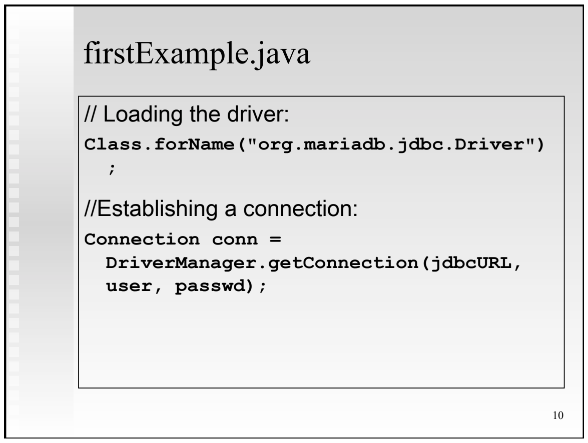
8



DBMS Driver

- Specific information you need to know: see the sample JDBC program and the project FAQ on:
  - Driver for the course DBMS
  - Using the driver (add to classpath)
  - Driver specifics for your programs

9



firstExample.java

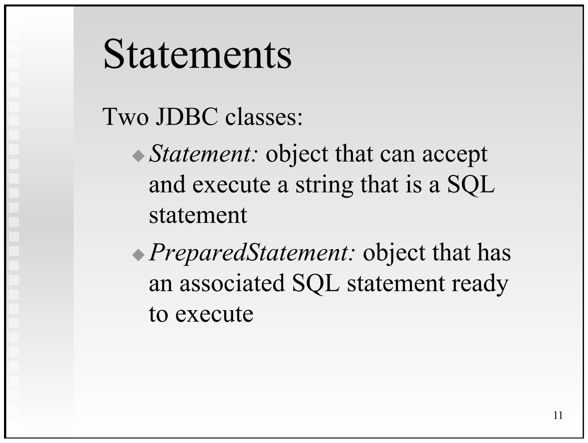
// Loading the driver:

```
Class.forName("org.mariadb.jdbc.Driver")
  ;
```

//Establishing a connection:

```
Connection conn =
  DriverManager.getConnection(jdbcURL,
  user, passwd);
```

10

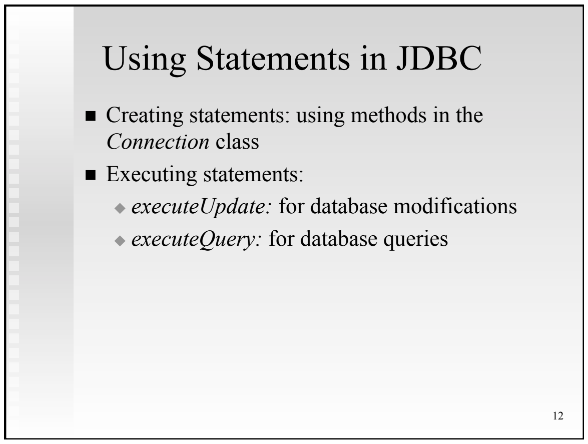


Statements

Two JDBC classes:

- *Statement:* object that can accept and execute a string that is a SQL statement
- *PreparedStatement:* object that has an associated SQL statement ready to execute

11



Using Statements in JDBC

- Creating statements: using methods in the *Connection* class
- Executing statements:
  - *executeUpdate:* for database modifications
  - *executeQuery:* for database queries

12

firstExample.java

```
// Create a statement object that will be sending your
// SQL statements to the DBMS:
Statement stmt = conn.createStatement();
// Create the COFFEES table:
stmt.executeUpdate("CREATE TABLE COFFEES " +
       "(COF_NAME VARCHAR(32), SUP_ID INTEGER, " +
       "PRICE FLOAT, SALES INTEGER, TOTAL INTEGER)");
// Populate the COFFEES table:
stmt.executeUpdate("INSERT INTO COFFEES " +
       "VALUES ('Colombian', 101, 7.99, 0, 0)");
// Get data from the COFFEES table:
ResultSet rs = stmt.executeQuery("SELECT COF_NAME,
   PRICE FROM COFFEES");
```

13

ResultSet

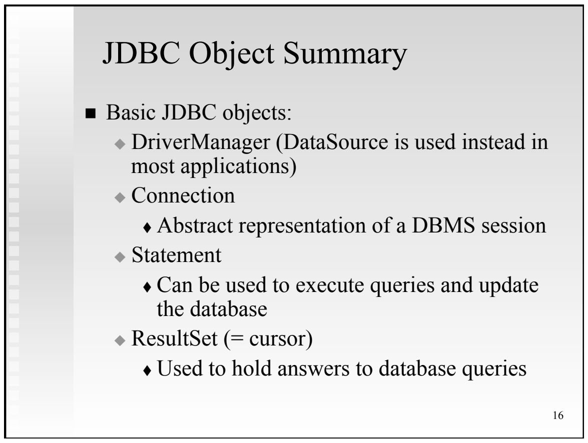
- An object of type ResultSet is like a cursor
- Method "next" advances cursor to next tuple:
  - The first time *next()* returns the first tuple
  - If no more tuples then *next()* returns FALSE
- Accessing components of tuples:
  - Method *getX(name),* where *X* is some type and *name* is an attribute name

14

firstExample.java

```
// Now rs contains the rows of coffees and prices from
// the COFFEES table. To access the data, use the
   method
// NEXT to access all rows in rs, one row at a time
while (rs.next()) {
  String s = rs.getString("COF_NAME");
  float n = rs.getFloat("PRICE");
  System.out.println(s + " " + n);
}
```

15



JDBC Object Summary

- Basic JDBC objects:
  - DriverManager (DataSource is used instead in most applications)
  - Connection
    - Abstract representation of a DBMS session
  - Statement
    - Can be used to execute queries and update the database
  - ResultSet (= cursor)
    - Used to hold answers to database queries

16