BCNF and JDBC

CSE 444 section, July 8, 2010
Today

• BCNF decompositions
• JDBC for project 2
What is BCNF?

A relation R is in BCNF if:

If $A_1,... A_n \rightarrow B$ is a non-trivial dependency in R, then $\{A_1, ..., A_n\}$ is a superkey for R
Why do BCNF decompositions?
BCNF decomposition algorithm

```
BCNF_Decompose(R)

    find X s.t.: X ≠X+ ≠ [all attributes]

    if (not found) then “R is in BCNF”

    let Y = X+ - X
    let Z = [all attributes] - X+
    decompose R into R1(X ∪ Y) and R2(X ∪ Z)
    continue to decompose recursively R1 and R2
```
BCNF example: table R(A, B, C, D, E)

Consider the following FDs:
• CD → E   BAD
• D → B    BAD
• A → CD

Which one are the bad dependences?

Note: a set of attributes X is a superkey if X+ = ABCDE
BCNF example: table R(A, B, C, D, E)

Consider the following FDs:
- CD → E  BAD
- D → B  BAD
- A → CD

R(A,B,C,D,E)
[CD+ = BCDE ≠ ABCDE]

R2(B,C,D,E)
[D+ = BD ≠ BCDE]

R4(B,D)
[BCNF]

R5(C,D,E)
[BCNF]

R3(A,C,D)
[BCNF]

Note: a set of attributes X is a superkey if X+ = ABCDE
Another example: R(A,B,C,D)

Consider the following FDs:

• C → D, C+ = ACD \textbf{BAD}
• C → A, C+ = ACD \textbf{BAD}
• B → C, B+ = ABCD

Note: a set of attributes X is a superkey if X+ = ABCD
A third example: $S(A,B,C,D,E)$

Consider the following FDs:

- $AB \rightarrow C$, $AB^+ = ABCD$ BAD
- $DE \rightarrow C$, $DE^+ = CDE$ BAD
- $B \rightarrow D$, $B^+ = BD$ BAD

1st Solution:

- $S(A,B,C,D,E)$ [AB+ ≠ ABCD ≠ ABCDE]
- $S2(A,B,C,D)$ [B+ ≠ ABCD]
- $S3(A,B,E)$ [BCNF]
- $S4(B,D)$ [BCNF]
- $S5(A,B,C)$ [BCNF]

Note: a set of attributes X is a superkey if $X^+ = ABCDE$
A third example: S(A,B,C,D,E)

Consider the following FDs:

- AB → C, AB+ = ABCD  BAD
- DE → C, DE+ = CDE  BAD
- B → D, B+ = BD  BAD

2\textsuperscript{nd} Solution:

- S(A,B,C,D,E)  [DE+ = CDE ≠ ABCDE]
- S2(A,B,D,E)  [B+ = BD ≠ ABDE]
- S3(C,D,E)  [BCNF]
- S4(B,D)  [BCNF]
- S5(A,B,E)  [BCNF]

Note: a set of attributes X is a superkey if X+ = ABCDE
A third example: \( S(A,B,C,D,E) \)

Consider the following FDs:

- \( AB \rightarrow C, \ AB+ = ABCD \)  BAD
- \( DE \rightarrow C, \ DE+ = CDE \)  BAD
- \( B \rightarrow D, \ B+ = BD \)  BAD

3\textsuperscript{rd} Solution:

- \( S(A,B,C,D,E) \)
  \[ B+ = BD \neq ABCDE \]
  BAD

- \( S2(A,B,C,E) \)
  \[ AB+ = ABC \neq ABCE \]
  BAD

- \( S4(A,B,C) \)
  [BCNF]

- \( S3(B,D) \)
  [BCNF]

- \( S5(A,B,E) \)
  [BCNF]

Note: a set of attributes \( X \) is a superkey if \( X+ = ABCDE \)
A table with “real” data

### JobInterviews

<table>
<thead>
<tr>
<th>CandID</th>
<th>Date</th>
<th>Time</th>
<th>EmplID</th>
<th>RoomNo</th>
</tr>
</thead>
<tbody>
<tr>
<td>C21</td>
<td>16-April-09</td>
<td>9.30 AM</td>
<td>E211</td>
<td>CSE550</td>
</tr>
<tr>
<td>C21</td>
<td>17-April-09</td>
<td>11.00 AM</td>
<td>E211</td>
<td>CSE550</td>
</tr>
<tr>
<td>C5</td>
<td>16-April-09</td>
<td>11.00 AM</td>
<td>E51</td>
<td>CSE218</td>
</tr>
<tr>
<td>C2</td>
<td>1-May-09</td>
<td>9.30 AM</td>
<td>E211</td>
<td>CSE218</td>
</tr>
</tbody>
</table>

Note: a set of attributes X is a superkey if \( X^+ = \) all attributes
A table with “real” data

**JobInterviews**

<table>
<thead>
<tr>
<th>CandID</th>
<th>Date</th>
<th>Time</th>
<th>EmpID</th>
<th>RoomNo</th>
</tr>
</thead>
<tbody>
<tr>
<td>C21</td>
<td>16-April-09</td>
<td>9.30 AM</td>
<td>E211</td>
<td>CSE550</td>
</tr>
<tr>
<td>C21</td>
<td>17-April-09</td>
<td>11.00 AM</td>
<td>E211</td>
<td>CSE550</td>
</tr>
<tr>
<td>C5</td>
<td>16-April-09</td>
<td>11.00 AM</td>
<td>E51</td>
<td>CSE218</td>
</tr>
<tr>
<td>C2</td>
<td>1-May-09</td>
<td>9.30 AM</td>
<td>E211</td>
<td>CSE218</td>
</tr>
</tbody>
</table>

What are the FDs?

- **CandID** → **EmpID, Room**  BAD
- **Date, Time** → **CandID, EmpID, RoomNo**
- **Date, EmpID** → **CandID, Time, RoomNo**
- more?

**Note**: a set of attributes X is a superkey if X+ = all attributes
A table with “real” data

**JobInterviews** (CandID, Date, Time, EmpID, RoomNo)

What are the FDs?
- CandID $\rightarrow$ EmpID, Room  **BAD**
- Date, Time $\rightarrow$ CandID, EmpID, RoomNo
- Date, EmpID $\rightarrow$ CandID, Time, RoomNo
- more?

Note: a set of attributes X is a superkey if $X^+ = \text{all attributes}$
Today

- BCNF decompositions
- JDBC for project 2
JDBC (Java Database Connectivity)

A Java API to access a database:

- connect to a data source
- send queries and update statements
- retrieve and process results

Documentation:

http://java.sun.com/javase/6/docs/technote
s/guides/jdbc/
JDBC lets Java talk to your database

Diagram:
- Java Application
- JDBC
- DBMS
- DBMS-proprietary protocol
- Client machine
- Database server
DBMS vendors make JDBC drivers...

- JDBC API for apps
- JDBC API for drivers

Java Application

DBMS

Client machine

DBMS-proprietary protocol

Database server
... letting JDBC talk to *any* database
JDBC architecture
Three-tier model

connection pooling
distributed transactions

Client machine
RMI, SOAP, REST, HTTP
Server (business logic)
DBMS-proprietary protocol
Database server
First, load the driver

- For Project 2, look in project2.tar.gz
  - SQL Server driver
    sqljdbc4.jar
  - PostgreSQL driver
    postgresql-8.4-701.jdbc4.jar
  - Already installed on Lab PCs (use 444shell.cmd)

- Put on class path, then tell Java to load it
  Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
  Class.forName("org.postgresql.Driver");
  - Class.forName() optional in current versions of Java
JDBC example

Connection con = DriverManager.getConnection
    ("jdbc:sqlserver://iisqlsrv;database=imdb",
    "username", "password");

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");
}
Modifying the database

Use `Statement.executeUpdate()`:

```java
Statement stmt = con.createStatement();

int rowsUpdated = stmt.executeUpdate(
    "UPDATE Actor " +
    "SET gender = 'F' " +
    "WHERE gender IS NULL"
);
```

- Works with any database modification, not just `UPDATE`
- Warning – will throw if you run it with a query!
Close all JDBC objects when done

Connection con = DriverManager.getConnection(...);
Statement stmt = con.createStatement();
ResultSet rs = stmt.executeQuery("SELECT a, b, c FROM Table1");

// do work with rs...

rs.close();
stmt.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();
}
Parameterized queries - PreparedStatement

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

... 
pstmt.setInt(1, 34);
ResultSet rs1 = pstmt.executeQuery();

... 
pstmt.setInt(1, 63);
ResultSet rs2 = pstmt.executeQuery();

... 
```
Parameterized queries - PreparedStatement

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\"!");
...
```
Parameterized queries - PreparedStatement

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\"!");
...
```

Parameterizing lets plan be cached!

```java
Statement stmt = con.createStatement();
ResultSet rs = stmt.executeQuery
   ("SELECT website FROM shops
       WHERE name = 'George\'s' OR ... ");
```

Must escape single quotes. What if this came from user?
Transactions

• A transaction is a logical group of SQL statements

• Each transaction is guaranteed to execute as if it was the only code running on the database

• We will talk more about them in lecture
Transactions in JDBC – option 1

Execute the SQL code to start, end transactions:

```java
PreparedStatement pBeginTx =
    con.prepareStatement("BEGIN TRANSACTION");
PreparedStatement pCommitTx =
    con.prepareStatement("COMMIT TRANSACTION");
PreparedStatement pRollbackTx =
    con.prepareStatement("ROLLBACK TRANSACTION");

...;

pBeginTx.executeUpdate();
// transaction started
...

if (ok) pCommitTx.executeUpdate();
else pRollbackTx.executeUpdate();
// transaction finished or reverted
```
Transactions in JDBC – option 2

Use JDBC methods to work with transactions:

```java
con.setAutoCommit(false);
// From now on, everything is in a transaction
...

if (ok) con.commit();
else con.rollback();
// Old transaction done/reverted, new one started
...

con.setAutoCommit(true);
// Now each statement executes by itself again
```