CSC343
Course Wrap-Up

csc343, Introduction to Databases
Nosayba El-Sayed
Fall 2015
Database Design – Recap & Comments

- Closure +
- Projection
- Minimal Cover
- Finding Keys for R
- BCNF Decomposition
- 3NF Decomposition

Bogdan’s section (L0101) used a slightly different algorithm. It’s ok if you use either approach! Just get it right :-)

Tutorial examples posted on course website are important. If you only rely on course slides, you don’t get the full picture!
Minimal Cover

• Step 1: Reduce RHS to Singletons
• Step 2: Remove redundant FDs
• Step 3: Reduce LHS whenever possible
• Step 4: Repeat steps 2, 3 until no changes occur

✓ In the other approach, reverse steps 2, 3, eliminate 4.
✓ Both are cited as valid Minimal Cover algorithms.
✓ This one (above) since is consistent with the text book we’re using for this course.
Database Design – Recap & Comments

• Closure +
• Projection
• Minimal Cover
• Finding Keys for R
• BCNF Decomposition
• 3NF Decomposition

There are *techniques* to help you limit the scope of attributes you use to find a relation’s Keys
Finding Keys from FDs – Helping techniques

• *Prime* attribute: if it’s part of any key
• Example #1: \( R(ABC), \text{ FDs} = \{A \rightarrow B, \ B \rightarrow C\} \)
  • \( A \) is clearly a key
  => \( A \) is prime, \( B \) and \( C \) are non-prime
• Example #2: \( R(ABC), \text{ FDs} = \{AB \rightarrow C, \ C \rightarrow A\} \)
  • \( AB \) and \( BC \) are the keys  (Check: closure test!)
  => \( A, B, \) and \( C \) are prime  (but not keys alone!)
• How do I know \( AB \) and \( BC \) are keys?
Keys from FDs – Helping technique (Example 1)

- $R(ABC)$, FDs = $\{A \rightarrow B, \ B \rightarrow C\}$

- Table of where attributes appear in FDs

<table>
<thead>
<tr>
<th>Only Left</th>
<th>Middle (appears both left/right)</th>
<th>Only Right</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td>C</td>
</tr>
</tbody>
</table>

- Only on Left => must be part of any key
- Only on Right => cannot be part of any key
- Middle => maybe (in combination with LEFT atts); maybe not.

- In this case: $A$
- $A^+ = ABC$ => $A$ is the key
- $A$ is part of any key, and it happens to be a key
  => no need to look at $B$
Keys from FDs – Helping technique (Example 2)

- R(ABC), FDs = \{AB \rightarrow C, \ C \rightarrow B, \ C \rightarrow D\}
- Table of where attributes appear in FDs:

<table>
<thead>
<tr>
<th>Left</th>
<th>Middle</th>
<th>Right</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B, C</td>
<td>D</td>
</tr>
</tbody>
</table>

- A must be part of any key; might be a key on its own, might not (test!)
- => A+ = A
- Add one from Middle: AB
- => AB+ = ABCD
- Must try it with the other Middles too: AC
- => AC+ = ACBD

=> both AB, AC are keys!

- What if all attributes of a relation were in the middle?
- Try one at a time, then combinations of two, etc.
The Final

- Comprehensive (covers the whole term), including:
  - RA
  - SQL (DDL, DML); JDBC
  - DTDs, XML, XQuery
  - FD theory and normalization
  - ER modelling and DB design
Preparing for the exam

• Re-solve parts of the assignments where you didn’t get full marks.
• For topics you aren’t fully confident in, re-do the lecture prep and in-class exercises.
• Go over solutions for in-class exercises
• Make up your own queries in RA, SQL, XQuery to hit on query types and language features you need practice in.
• Solve old tests and finals.
The Final

• You need to know the syntax of each language.
• You don’t need to memorize function/method APIs. We will provide what you need and/or be forgiving when marking those details.
• SQL: views are always welcome, as long as correct.
• Comments are never necessary unless we say otherwise.
• Questions may be similar to previous tests and final exams, but don’t count on that!
The Final

• It’s about 24 pages long, but
  • A page towards the end is empty (for rough work)
  • Last 2 pages: the schemas for reference (you can detach this last sheet, for convenience – do not detach anything else though)
  • Page 1 is the cover
  • Lots of empty space to fill in your answers

• So it’s really 20 pages, with lots of white space
• You need 40% on the final to pass the course, regardless of the rest of the term marks
Final Exam – Logistics

- When and where:
  - [http://www.artsci.utoronto.ca/current/exams/dec15](http://www.artsci.utoronto.ca/current/exams/dec15)

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Section</th>
<th>Date</th>
<th>Time</th>
<th>Room</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSC343H1F</td>
<td>A - MC</td>
<td>TUE 15 DEC</td>
<td>EV 7:00 - 10:00</td>
<td>BN 2N</td>
</tr>
<tr>
<td>CSC343H1F</td>
<td>ME - Z</td>
<td>TUE 15 DEC</td>
<td>EV 7:00 - 10:00</td>
<td>BN 2S</td>
</tr>
<tr>
<td>CSC369H1F</td>
<td>A - T</td>
<td>FRI 11 DEC</td>
<td>EV 7:00 - 10:00</td>
<td>BN 3</td>
</tr>
<tr>
<td>CSC369H1F</td>
<td>U - Z</td>
<td>FRI 11 DEC</td>
<td>EV 7:00 - 10:00</td>
<td>ST VLAD</td>
</tr>
<tr>
<td>CSC373H1F</td>
<td></td>
<td>THU 17 DEC</td>
<td>EV 7:00 - 10:00</td>
<td>BN 3</td>
</tr>
</tbody>
</table>

- Clara Benson Building (BN), 320 Huron Street
- Next to Athletic Centre!
- **EV == evening! So, it’s at 7-10PM, not AM!**
Course wrap-up - Lessons learned

• What do I take away from this course?
  • Data models are important
  • Relational model: concept of relation/table
  • Schema vs. instance!
  • Keys, integrity constraints
  • RA: foundation of SQL
  • SQL: DML, DDL, expressive power/limitations
  • Embedded SQL: more control, addresses SQL limitations by combining it with a conventional language
  • XML/DTD + Xpath/Xquery – not all data fits a rigid schema; unstructured data needs another representation
  • Database design theory: client requirements => E/R diagram => relational schema.
  • FD theory helps bring schema into a normal form
Why should I care?

• In the era of **Big Data**, having knowledge of database systems is really important! E.g.,
  • What type of data do I have? Structured vs. unstructured
  • How do I start designing a database?
  • How do I optimize my design?
  • How do I use a database system?

• Database systems are all part of a bigger picture
Trends in DB Research

- Managing huge amounts of data: approximate querying, statistical methods, self-tuning, power management
- NoSQL technologies
- Stream processing
- Data mining
- Data privacy and security
- Different kinds of data, e.g., spatial, temporal, data from sensors, social network data, graph databases
- Visualization of data

- Top-tier database conferences: VLDB, SIGMOD, ICDE, EDBT, CIDR, CIKM, SIGSPATIAL GIS, IEEE BigData
NoSQL – why should I check it out?

DO YOU HAVE ANY EXPERTISE IN SQL?

NO

DOESN'T MATTER. WRITE: "EXPERT IN NO SQL."

Leverage the NoSQL boom
BigData – no “one-size fits all”

• Relational databases are not always a good fit

• No “one-size fits all”
  • Typical relational database: less than 1TB of data
  • Google: 900 TB of search engine data (mostly unstructured!)
  • Youtube: 80PB video data/year
  • Scientific data
    • US department of energy: 3.5PB
Need for flexible data model

• Relational schema: too rigid
  • No way to change dynamically
  • Need a DB admin to “stop the world” and change the schema, migrate the data in the new structures, etc.

• Many applications’ data: no fixed structure
  • Log processing
  • Stream processing
  • Graph processing
    (e.g. think Google Maps)
NoSQL advantages

• Data is replicated to multiple nodes (availability and fault-tolerance) and can be partitioned
  • Down nodes easily replaced
  • No single point of failure

• Can scale up and down

• Doesn't require a schema
  • Not really.. :)

University of Toronto
What are we sacrificing instead?

• Decades of database optimizations (carefully-designed query optimizers, indexing, etc.)

• Joins

• ACID Transactions

• SQL, powerful expressive query language (mostly)

• Easy integration with other applications that support SQL
Should I be using NoSQL Databases?

- NoSQL data storage systems makes sense for applications that need to deal with very large semi-structured data
  - Log Analysis
  - Social Networking Feeds

- Most of us work on organizational databases, which are not that large and have low update/query rates
  - Regular relational databases are the right solution for most such applications
Source:
https://kvaes.wordpress.com/2015/01/21/database-variants-explained-sql-or-nosql-is-that-really-the-question/
Big Picture
What next?

THE FUTURE OF THE DATABASE

1960s
First Computerized Database Models
- Hierarchical Model (IMS)
- Network Model (CODASYL)

1970s
The Dawn of the Database
- The relational model and its language SQL emerge
- E.F. Codd writes a paper on the Relational Database Model
- System R by IBM (SQL)
- Ingres (QJEL)
- Single Instance Relational Database
- Entity Relational Database

1980s
An Industry Develops
- SQL becomes the de facto standard
- Commercial offerings from IBM, Oracle grow market
- Other data models enter the scene, without much traction
- 1st commercially available RDBMS
- IBM DB2
- SAP Sybase
- Informix

1990s
Technology Shifts
- Data explodes with the Internet age
- Single server SQL databases run into resource problems
- Business Intelligence and Analytics move out of transactional database

Source: wired.com
2000s
New Players Emerge
- Data variety, velocity and volume increase
- New analytics SQL databases are introduced
- NoSQL databases fill the gap for processing unstructured data
- Hadoop gains traction for analyzing petabytes of data

Today
Databases Adapt and Evolve
- Businesses require real-time analytics on operational data
- Scale-up SQL proves too costly, but scale-out removes resource constraint
- Scale-out provides real-time analytics with high volume transactions
- Google and Clustrix are pioneers in this space

The Future
Businesses Advance with Database Innovations
- Single node SQL gets replaced by scale-out SQL
- Data warehouse type analytics will become available in real-time database
- Businesses gain a significant edge and increased agility

Winning Database Platforms
- NOSQL DATABASE
- DISTRIBUTED SQL
- HADOOP
CSC443

- “Database System Technology”
- Takes the perspective of building a DBMS.
- Internals of a DBMS
- Topics like:
  - Memory management – bufferpool
  - Query optimization – produce good query plans
  - Managing storage – row-oriented, column-oriented, etc.
  - Concurrency control
  - Tuning for performance
  - Types of workloads: OLTP, OLAP, etc.
  - Data mining
Thank you!

Hope you found this course interesting; good luck using what you learned in your future career prospects!

Good luck with the final exam :-)

That's all Folks!