Database Systems I

Design Theory (1)

Instructor: Ouldooz Baghban Karimi

CMPT 354 - Summer 2019
Example

• If every course is in only one room, the relation contains redundant information

<table>
<thead>
<tr>
<th>Student</th>
<th>Course</th>
<th>Room</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mike</td>
<td>354</td>
<td>AQ3149</td>
</tr>
<tr>
<td>Mary</td>
<td>354</td>
<td>AQ3149</td>
</tr>
<tr>
<td>Sam</td>
<td>354</td>
<td>AQ3149</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Example

- If we update the room number for one tuple, we get inconsistent data. An **update anomaly**

<table>
<thead>
<tr>
<th>Student</th>
<th>Course</th>
<th>Room</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mike</td>
<td>354</td>
<td>AQ3149</td>
</tr>
<tr>
<td>Mary</td>
<td>354</td>
<td>TASC1 9204</td>
</tr>
<tr>
<td>Sam</td>
<td>354</td>
<td>AQ3149</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Example

- If everyone drops the class, we lose what room the class is in
  A delete anomaly

<table>
<thead>
<tr>
<th>Student</th>
<th>Course</th>
<th>Room</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Example

• We can not reserve a room without students
  An insert anomaly

<table>
<thead>
<tr>
<th></th>
<th>454</th>
<th>T9204</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Student</th>
<th>Course</th>
<th>Room</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Design Theory

• How to represent your data to avoid anomalies

• Anomalies: Problems that are caused in some relation schemas by the presence of certain dependencies
Functional Dependencies

• A functional dependency (FD) on a relation R
  • If two tuples of R agree on all of the attributes $A_1, A_2, \ldots, A_n$, then they must also agree on all of another list of attributes $B_1, B_2, \ldots, B_m$
    \[
    A_1 \ A_2 \ldots \ A_n \rightarrow B_1 \ B_2 \ldots \ B_m
    \]

    $A_1, A_2, \ldots, A_n$ functionally determine $B_1, B_2, \ldots, B_m$

• If we can be sure every instance of a relation R will be one in which a given FD is true, then we say that R satisfies the FD
# Functional Dependencies

If $t$ and $u$ agree here, then they must agree here.

<table>
<thead>
<tr>
<th>A1</th>
<th>...</th>
<th>An</th>
<th>B1</th>
<th>...</th>
<th>Bm</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>u</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example

The relation *Movies1*:

<table>
<thead>
<tr>
<th>title</th>
<th>year</th>
<th>length</th>
<th>genre</th>
<th>studioName</th>
<th>starName</th>
</tr>
</thead>
<tbody>
<tr>
<td>Star Wars</td>
<td>1977</td>
<td>124</td>
<td>sciFi</td>
<td>Fox</td>
<td>Carrie Fisher</td>
</tr>
<tr>
<td>Star Wars</td>
<td>1977</td>
<td>124</td>
<td>sciFi</td>
<td>Fox</td>
<td>Mark Hamill</td>
</tr>
<tr>
<td>Star Wars</td>
<td>1977</td>
<td>124</td>
<td>sciFi</td>
<td>Fox</td>
<td>Harrison Ford</td>
</tr>
<tr>
<td>Gone With the Wind</td>
<td>1939</td>
<td>231</td>
<td>drama</td>
<td>MGM</td>
<td>Vivian Leigh</td>
</tr>
<tr>
<td>Wayne’s World</td>
<td>1992</td>
<td>95</td>
<td>comedy</td>
<td>Paramount</td>
<td>Dana Carvey</td>
</tr>
<tr>
<td>Wayne’s World</td>
<td>1992</td>
<td>95</td>
<td>comedy</td>
<td>Paramount</td>
<td>Mike Meyers</td>
</tr>
</tbody>
</table>

\[
\text{title year } \rightarrow \text{ length genre studioName} \\
\text{title year } \rightarrow \neg \text{ starName}
\]
Keys of Relations

• \{A_1, A_2, \ldots, A_n\} is a key for a relation R if
  • Those attributes functionally determine all other attributes of the relation.
    • That is, it is impossible for two distinct tuples of R to agree on all of \{A_1, A_2, \ldots, A_n\}
  • No proper subset of \{A_1, A_2, \ldots, A_n\} functionally determines all other attributes of R
    • Means a key must be minimal

• Example: \{title, year, starName\} form a key for Movies1
Superkeys

- A set of attributes that contains a key is called a superkey, short for superset of a key

- Every key is a superkey

- Every superkey satisfies the first condition
  - Need not satisfy the second condition
FD Rules

• Splitting Rule

\[ A_1 A_2 \ldots A_n \rightarrow B_1 B_2 \ldots B_m \]

could be replaced with

\[ A_1 A_2 \ldots A_n \rightarrow B_i \quad \text{for } i=1, \ldots, m \]

• Combining Rule

\[ A_1 A_2 \ldots A_n \rightarrow B_i \quad \text{for } i=1, \ldots, m \]

could be replaced with

\[ A_1 A_2 \ldots A_n \rightarrow B_1 B_2 \ldots B_m \]
Example

<table>
<thead>
<tr>
<th>title</th>
<th>year</th>
<th>length</th>
<th>genre</th>
<th>studioName</th>
<th>starName</th>
</tr>
</thead>
<tbody>
<tr>
<td>Star Wars</td>
<td>1977</td>
<td>124</td>
<td>sciFi</td>
<td>Fox</td>
<td>Carrie Fisher</td>
</tr>
<tr>
<td>Star Wars</td>
<td>1977</td>
<td>124</td>
<td>sciFi</td>
<td>Fox</td>
<td>Mark Hamill</td>
</tr>
<tr>
<td>Star Wars</td>
<td>1977</td>
<td>124</td>
<td>sciFi</td>
<td>Fox</td>
<td>Harrison Ford</td>
</tr>
<tr>
<td>Gone With the Wind</td>
<td>1939</td>
<td>231</td>
<td>drama</td>
<td>MGM</td>
<td>Vivian Leigh</td>
</tr>
<tr>
<td>Wayne’s World</td>
<td>1992</td>
<td>95</td>
<td>comedy</td>
<td>Paramount</td>
<td>Dana Carvey</td>
</tr>
<tr>
<td>Wayne’s World</td>
<td>1992</td>
<td>95</td>
<td>comedy</td>
<td>Paramount</td>
<td>Mike Meyers</td>
</tr>
</tbody>
</table>

title year → length genre studioName

Could be also expressed as

- title year → length
- title year → genre
- title year → studioName
Trivial FDs

• A constraint of any kind on a relation is trivial if it holds for every instance of the relation

\[ A_1 A_2 \ldots A_n \rightarrow B_1 B_2 \ldots B_m \]
\[ \{B_1, B_2, \ldots, B_m\} \subseteq \{A_1, A_2, \ldots, A_n\} \]

• Every Trivial FD holds in every relation
Closure of Attributes

\{A_1, A_2, \ldots, A_n\} is a set of attributes and \(S\) is a Set of FDs.

The **closure** of \(\{A_1, A_2, \ldots, A_n\}\) under the FDs in \(S\) is the set of attributes \(B\) such that every relation that satisfies all the FDs in set \(S\) also satisfies \(A_1 A_2 \ldots A_n \rightarrow B\)

• We denote the closure of a set of attributes \(A_1 A_2 \ldots A_n\) by \(\{A_1, A_2, \ldots, A_n\}^+\)
Transitive Rule

• Cascade two FDs

\[
\text{If } A_1 A_2 \ldots A_n \Rightarrow B_1 B_2 \ldots B_m \\
\text{and } B_1 B_2 \ldots B_m \Rightarrow C_1 C_2 \ldots C_k \\
\text{hold in relation } R, \text{ then } A_1 A_2 \ldots A_n \Rightarrow C_1 C_2 \ldots C_k \text{ also holds in } R
\]

• Example

\[
\begin{align*}
\text{title year} & \Rightarrow \text{studioName} \\
\text{studioName} & \Rightarrow \text{studioAddress} \\
\text{title year} & \Rightarrow \text{studioAddress}
\end{align*}
\]
Closing Set of FDs

• A choice of which FDs to represent the full set of FDs for a relation

• Given set of FDs $S$, any set of FDs equivalent to $S$ is a **basis** for $S$

• **Minimal Basis** for a relation is basis $B$ that satisfies
  • All the FDs in $B$ have singleton right sides
  • If any FD is removed from $B$ the result is no longer a basis
  • If for any FD in $B$ we remove one or more attributes from the left side of $F$, the result is no longer a basis
Projecting FDs

\[ R_1 = \pi_L(R) \]

What FDs hold in \( R_1 \)?

- Projection of functional dependencies from \( S \) that
  - Follow from \( S \)
  - Involve only attributes of \( R_1 \)
Acknowledgements

I have used materials from the following resources in preparation of this course:

• **Database Systems: The Complete Book**
• Database Systems (Kiefer, Bernstein, Lewis)
• Database System Concepts: [https://www.db-book.com](https://www.db-book.com)
• Course offerings
  • **CMPT 354 (Jiannan Wang - SFU):** [https://sfu-db.github.io/cmpt354/](https://sfu-db.github.io/cmpt354/)
  • W 4111 (Eugene Wu - Columbia): [https://w4111.github.io/](https://w4111.github.io/)
  • CS 186 (Joe Hellerstein - Berkeley): [https://sites.google.com/site/cs186fall17/](https://sites.google.com/site/cs186fall17/)