COL362/632 Introduction to Database Management Systems

Database Design – Functional Dependencies

Kaustubh Beedkar

Department of Computer Science and Engineering
Indian Institute of Technology Delhi

Example

- Consider the following relation

ActorFilm						
id	**firstname**	**lastname**	**dob**	**title**	**year**	**language**
1	Priyanka	Chopra	1992	Don	2006	Hindi
1	Priyanka	Chopra	1992	Don-II	2011	Hindi
2	Anthony	Hopkins	1937	MI-IV	2011	English
2	Anthony	Hopkins	1937	Valkyrie	2008	English
3	Bill	Nighty	1949	Valkyrie	2008	English

- Problems with this design
  1. Redundancy
  2. Update Anomalies

- `update ActorFilm set firstname = 'Priyankaa' where id = 1;`
- `insert into ActorFilm values (null, null, null, null, Avatar, 2009, English);`
- `delete from ActorFilm where title = 'Valkyrie';`
- `delete from ActorFilm where firstname = 'Anthony' and lastname = 'Hopkins';`

Normalization

The process of systematically eliminating redundancy and anomalies.

Functional Dependencies

- A **functional dependency** is another kind of constraint

- If two tuples in a relation agree on the values of one set of attributes, then they must also agree on the values of another set of attributes

Given two attribute sets A and B with $A, B \subseteq [R]$

$A \rightarrow B$ is called functional dependency (FD) if a function exists
$f(a) := b \quad \forall a \in \pi_A(R), b \in \pi_B(R)$ for all possible instances of R.

- In other words, values of A determine the values of B

Example

ActorFilm						
id	**firstname**	**lastname**	**dob**	**title**	**year**	**language**
1	Priyanka	Chopra	1992	Don	2006	Hindi
1	Priyanka	Chopra	1992	Don-II	2011	Hindi
2	Anthony	Hopkins	1937	MI-IV	2011	English
2	Anthony	Hopkins	1937	Valkyrie	2008	English
3	Bill	Nighty	1949	Valkyrie	2008	English

Functional Dependencies (FDs)

- { id, firstname, lastname } $\rightarrow$ { dob }
- { title } $\rightarrow$ { language }
- { firstname, lastname } $\rightarrow$ { title, year }

Incorrect FD!

Functional Dependencies

- Important to differentiate between instance and schema
- FD defines a constraint for the possible instances of R
  - Instances that satisfy FD are called **legal instances**
- FD do not define dependencies for concrete instance
  - Single counter example is enough!

Functional Dependencies

Example

ActorFilm						
id	**firstname**	**lastname**	**dob**	**title**	**year**	**language**
1	Priyanka	Chopra	1992	Don	2006	Hindi
1	Priyanka	Chopra	1992	Don-II	2011	Hindi
2	Anthony	Hopkins	1937	MI-IV	2011	English
2	Anthony	Hopkins	1937	Valkyrie	2008	English
3	Bill	Nighty	1949	Valkyrie	2008	English

- Functional Dependencies
  - $\{$ id $\} \rightarrow \{$ firstname, lastname, dob $\}$
  - $\{$ firstname $\} \rightarrow \{$ lastname $\}$ **not an FD**
  - $\{$ dob $\} \rightarrow \{$ id, firstname, lastname $\}$ **not an FD**

Functional Dependencies

- Not just a theoretical DB concept!
- Play a crucial role in applications beyond DB design
  - **Feature engineering and selection in ML** (e.g., dimensionality reduction, feature generation)
  - **Data cleaning and anomaly detection** (e.g., missing value imputation, duplicate detection, data errors)
  - **Causal discovery and data understanding in ML**
  - **Query Optimization in Data pipelines**
  - **Data integration and schema matching**
  - **Balancing data privacy and data utility**

Super Keys

- A **superkey** is a set of one or more attributes (columns) in a relation (table) such that the values of these attributes are sufficient to uniquely identify a tuple (row) in the relation.
- Any subset $S \subseteq [R]$ for which $S \rightarrow [R]$ holds is called a **super key** of R
  - $[R] \rightarrow [R]$
  - $\exists a \in S : S \setminus \{ a \} \rightarrow [R]$
- Super key functionally determines all other attributes
  - Not necessarily minimal!
- **Example**
  - ActorFilm(id, firstname, lastname, dob, title, year, language)
  - $S =$ { id, firsname, lastname, dob, title}

Candidate Key

- A **candidate key** is a **minimal superkey**
- For a superkey $S \subseteq [R]$ for which
  - $\nexists a \in S : S \setminus \{ a \} \to [R]$ or
  - $\forall a \in S : S \setminus \{ a \} \not\to [R]$
  - is called the **candidate key** of R
- **Example**
  - ActorFilm(id, firstname, lastname, dob, title, year, language)
  - $S =$\{ firstname, lastname, title, year \}
- A relation can have **multiple candidate keys**

Primary Attribute & Primary Key

- Let C be the set of all candidate keys of $[R]$, then
- all attributes $\bigcup_i c_i$ are called **primary attributes** of $[R]$

Primary Key

- Any candidate key used to identify a tuple!
  - Cannot have null
  - Not change over time
  - Short as possible

Inferring FDs

- Given a set of FDs, which other FDs follow from it?
- Example
  - Given: $R(A, B, C)$ satisfies $A \to B$ and $B \to C$
  - Inferred: $A \to C$ through **transitivity of FDs**.

  - Given: $\{\text{aadhar}\} \to \{\text{pan}\}$, $\{\text{pan}\} \to \{\text{dob}\}$
  - Then, $\{\text{aadhar}\} \to \{\text{dob}\}$

Inferring FDs

- Given: $A_1A_2 \ldots A_n \rightarrow B_1B_2 \ldots B_m$
- Inferred $A_1A_2 \ldots A_n \rightarrow B_i$ for $i = 1, 2, \ldots, m$ through **splitting rule**

- Given $A_1A_2 \ldots A_n \rightarrow B_i$ for $i = 1, 2, \ldots, m$
- Inferred: $A_1A_2 \ldots A_n \rightarrow B_1B_2 \ldots B_m$ through **combining rule**

Equivalent FDs

- Two FDs S and T are equivalent if
  - set of relation instances satisfying S is exactly the same as the set of relation instances satisfying T
- If FD S follows from FD T, if every instance that satisfies T also satisfies S
- $S = T$ iff $S \implies T$ and $T \implies S$

Trivial and Non-trivial FDs

- Consider $R(A, B, C)$
- Given FDs
  - $AB \rightarrow A$ trivial
  - $AB \rightarrow AC$ non-trivial
  - $AB \rightarrow C$ completely non-trivial
- **Trivial FD** $X \rightarrow Y$ where $Y \subseteq X$
- **Non-trivial FD** $X \rightarrow Y$ where $Y \not\subseteq X$
- **Completely non-trivial FD** $X \rightarrow Y$ where $Y \cap X = \emptyset$

Armstrong's Axioms

1.	$B \subseteq A$	$\Longrightarrow$	$A \rightarrow B$	**Reflexivity**
2.	$A \rightarrow B$	$\Longrightarrow$	$AC \rightarrow BC$	**Augmentation**
3.	$A \rightarrow B,\ B \rightarrow C$	$\Longrightarrow$	$A \rightarrow C$	**Transitivity**
4.	$A \rightarrow BC$	$\Longrightarrow$	$A \rightarrow B,\ A \rightarrow C$	**Decomposition (splitting)**
5.	$A \rightarrow B,\ A \rightarrow C$	$\Longrightarrow$	$A \rightarrow BC$	**Union (combining)**
6.	$A \rightarrow B,\ DB \rightarrow C$	$\Longrightarrow$	$DA \rightarrow C$	**Pseudo-transitivity**

Closure of FDs

- Given: the set S of FDs
- Output: S^+, the **closure** of S containing all FDs that can be derived from S
- Example
  - Given $S = \{ AB \rightarrow C, C \rightarrow ED \}$
  - Then $S^+ = \{ AB \rightarrow ED, AB \rightarrow E, AB \rightarrow D, ABC \rightarrow ED, C \rightarrow E, C \rightarrow D, \ldots \}$
  - HW: Find all FDs

Closure of Attributes

- Let S be a set of FDs
  - Is a new FD F derivable from S?
- Example
  - Given
    $S = \{ AB \to C, BC \to AD, D \to E, CF \to B \}$
  - Does the FD $AB \to D$ hold?
  - Compute $\{ A, B \}^+$

FD	$\{ A, B \}^+$
$AB \to C$	$\{ A, B, C \}$
$BC \to AD$	$\{ A, B, C, D \}$
$D \to E$	$\{ A, B, C, D, E \}$

  - $AB \to D$ holds!

Attribute Closure Method

Given

- set S of FDs
- candidate FD $F : A \to B$

Step 1: Compute A^+

$A^+ = A$ initial set of attributes

do {
  for each FD $X \to Y \in S$ {
    if $X \subseteq A^+$ then $A^+ \leftarrow A^+ \cup Y$
  }
  if A^+ unchanged, quit
}

Step 2: Check if $B \subseteq A^+$