Introduction to Database Systems

Schema Refinement: Functional Dependencies

Lee Mong Li

NUS
National University of Singapore

Database Design Process

1. **Requirements Analysis**
   - What does the user want from the database?
   - Find out the data/application/performance requirements
2. **Conceptual Database Design**
   - Capture data requirements using a conceptual data model, e.g. ER model
   - High level description of data to be stored in database, and constraints that hold over the data
3. **Logical Database Design**
   - Convert conceptual database design to a logical schema supported by DBMS
4. **Schema Refinement**
   - Use data constraints to improve the logical schema.
   - Theory of normalizing relations
5. **Physical Database Design**
   - Use performance criteria to design physical schema, e.g. build indexes
6. **Application and Security Design**
   - Specify access control policies

DB Schema = Relation Schemas + Constraints

- **Data represented by schemas have application-dependent constraints relating to attribute values**

MovieList Database

title	director	address	phone	Time
Schlinder’s List	Spielberg	Holland	3355	1130
Saving Private Ryan	Spielberg	Holland	3355	1430
Noth by Northwest	Hitchcock	Orchard	1234	1400
The Godfather	Coppola	Orchard	1234	1700
Saving Private Ryan	Spielberg	Orchard	1234	2130

- **Data constraints:**
  - Each movie has one director
  - Each cinema has one phone number
  - Each cinema screens one movie at a time

Good and Bad Schema Design

- **Problems with the MovieList database**
  - **Redundant storage:** Some information is stored repeatedly
  - **Insertion anomaly:** Cannot store information about a new movie if the screening place and time are not known

MovieList Database

title	director	address	phone	Time
Schlinder's List	Spielberg	Holland	3355	1130
Saving Private Ryan	Spielberg	Holland	3355	1430
Noth by Northwest	Hitchcock	Orchard	1234	1400
The Godfather	Coppola	Orchard	1234	1700
Saving Private Ryan	Spielberg	Orchard	1234	2130

Good and Bad Schema Design

- **Problems with the MovieList database**
  - **Deletion anomaly:** Lose information about cinema at Holland if we delete all movies directed by Spielberg
  - **Update anomaly:** Inconsistent updates may occur if the phone number of a cinema changes

MovieList Database

title	director	address	phone	Time
Schlinder's List	Spielberg	Holland	3355	1130
Saving Private Ryan	Spielberg	Holland	3355	1430
Noth by Northwest	Hitchcock	Orchard	1234	1400
The Godfather	Coppola	Orchard	1234	1700
Saving Private Ryan	Spielberg	Orchard	1234	2130

Good and Bad Schema Design

- **Refine a bad schema by decomposing it into multiple good ones**

Movie

title	director
Schlinder's List	Spielberg
Saving Private Ryan	Spielberg
Noth by Northwest	Hitchcock
The Godfather	Coppola

Screens

address	time	title
Holland	1130	Schlinder's List
Holland	1430	Saving Private Ryan
Orchard	1400	Noth by Northwest
Orchard	1700	The Godfather
Orchard	1430	Saving Private Ryan

Cinema

address	phone
Holland	3355
Orchard	1234

- **Refined schema allows**
  - Insertion of new movies without knowing their screening details
  - Deletion of movies without losing information about cinemas
  - Updating a single record to change a cinema's phone number

Schema Design Issues

- **Two main problems:**
  - How to determine whether a schema design is good or bad?
  - How to transform a bad design into a good one?

- **Theory of functional dependencies provide a systematic approach to address these issues**

- **Introduced by E.F. Codd**
  - *A relational model for large shared data banks*, in Communications of the ACM, Vol. 13, No. 6, 1970.

Functional Dependencies (FDs)

- **Let X and Y be subsets of attributes of a relation R**
- **A functional dependency $X \rightarrow Y$ holds over R if and only if for any instance *r* of R, whenever two tuples t_1 and t_2 of *r* agree on attributes X, they also agree on attributes Y**

$$t_1.X = t_2.X \Rightarrow t_1.Y = t_2.Y$$

- **We say that X functionally determines Y (or Y functionally depends on X)**

Example

MovieList (title, director, address, phone, time)

title	director	address	phone	time
Schlinder's List	Spielberg	**Holland**	**3355**	1130
Saving Private Ryan	**Spielberg**	**Holland**	**3355**	1430
Noth by Northwest	Hitchcock	Orchard	1234	1400
The Godfather	Coppola	Orchard	1234	1700
Saving Private Ryan	**Spielberg**	Orchard	1234	2130

Functional dependencies on MovieList:

title → director

address → phone

address, time → title

Example

MovieList (title, director, address, phone, time)

FD: ***title → director***

In DRC:

$\forall T\ \forall D1\ \forall A1\ \forall P1\ \forall M1\ \forall D2\ \forall A2\ \forall P2\ \forall M2$

$((MovieList\ (T, D1, A1, P1, M1) \wedge MovieList\ (T, D2, A2, P2, M2))$
$\Rightarrow (D1 = D2))$

In TRC:

$\forall L1\ \forall L2\ (L1 \in MovieList \wedge L2 \in MovieList \wedge L1.title = L2.title$
$\Rightarrow L1.director = L2.director)$

In SQL:

```
CHECK (NOT EXISTS ( SELECT * FROM MovieList R1, MovieList R2
      WHERE R1.title = R2.title AND R1.director <> R2.director ) )
```

Definitions

- **Let *r* be a relation instance of relation schema *R***
- ***r* satisfies FD X $\rightarrow$ Y if for every pair of tuples t_1 and t_2 in *r* such that $t_1.X = t_2.X$, it is also true that $t_1.Y = t_2.Y$**
- **An FD *f* holds on *R* if and only if for any relation instance *r* of *R*, *r* satisfies *f***
- ***r* violates an FD *f* if *r* does not satisfy *f***
- ***r* is a legal instance of *R* if *r* satisfies all FDs that hold on *R***

Trivial, Non-trivial, Completely Non-trivial FDs

- **An FD $X \rightarrow Y$ is a trivial FD if $Y \subseteq X$; otherwise it is a non-trivial FD (i.e., $Y \not\subseteq X$)**
- **An FD $X \rightarrow Y$ is a completely non-trivial FD if $Y \cap X = \emptyset$**
- **Example**
  - $AB \rightarrow B$ is a trivial FD
  - $AB \rightarrow AC$ is a non-trivial FD
  - $AB \rightarrow C$ is a completely non-trivial FD

Example

- Consider relation schema Movie (title, director, producer)
- Let r be a legal relation instance of Movie

title	director	producer
Angela's Ashes	Parker	Williams
Saving Private Ryan	Spielberg	Williams
Noth by Northwest	Hitchcock	Harris
Schindler's List	Spielberg	Williams
Vertigo	Hitchcock	Harris

- FD producer $\rightarrow$ director does not hold on Movie
- r satisfies the FD director $\rightarrow$ producer, but we cannot conclude that director $\rightarrow$ producer holds on Movie
- Based on legal instances of R, we can tell which FDs do not hold on R, but we <u>cannot</u> deduce which non-trivial FDs hold on R!

Quiz

Consider the relation instance r of schema R(A, B, C)

A	B	C
0	0	0
2	1	2
1	1	2
0	0	1

List all non-trivial FDs that are satisfied by r

Reasoning about FDs

- **Implication problem:**
  - Given a set of FDs *F* that hold on *R*, and an FD *f*, does *f* also hold on *R*?
- **Example in MovieList, we have FDs**

  F = { {title} $\rightarrow$ {director}, {address} $\rightarrow$ {phone}, {address, time} $\rightarrow$ {title} }
- **Which of the following FD also hold on MovieList?**
  - {address, time} $\rightarrow$ {title}
  - {time} $\rightarrow$ {title}

Reasoning about FDs

- Let *F* and *G* denote sets of FDs, and *f* denote an FD
- ***F* implies G** if *F* implies *g* for each $g \in G$
- **Closure of F**, denoted by F^+, is the set of all FDs implied by F.
- Two sets of FDs, F and G, are **equivalent**, denoted by $F \equiv G$, if $F^+ = G^+$.

Axioms for FDs

- **A collection of formal rules used to derive an FD from a set of FDs**
- **Armstrong's Axioms: Let *X, Y, Z* $\subseteq$ R**
  - Reflexivity: If $Y \subseteq X$, then $X \rightarrow Y$
  - Augmentation: If $X \rightarrow Y$, then $XZ \rightarrow YZ$
  - Transitivity: If $X \rightarrow Y$ and $Y \rightarrow Z$, then $X \rightarrow Z$
- **Armstrong's Axioms are both sound and complete**
  - Sound: Any derived FD is implied by F
  - Complete: All FDs in F^+ can be derived

Additional Inference Rules

Union: If $X \rightarrow Y$ and $X \rightarrow Z$, then $X \rightarrow YZ$

Decomposition: If $X \rightarrow YZ$, then $X \rightarrow Y$ and $X \rightarrow Z$

Example

Let $F = \{A \rightarrow BCD\}$ and $G = \{A \rightarrow B, A \rightarrow C, A \rightarrow D\}$

Show that F and G are equivalent.

By the decomposition rule, we have

$$F \text{ implies } A \rightarrow B, A \rightarrow C, A \rightarrow D$$

Therefore, F implies G

By the union rule, we have

$$\{A \rightarrow B, A \rightarrow C\} \text{ implies } A \rightarrow BC \text{ and}$$

$$\{A \rightarrow BC, A \rightarrow D\} \text{ implies } A \rightarrow BCD$$

Therefore, G implies F

Hence, $F \equiv G$

Quiz

Let F = {$A \rightarrow B$, $AB \rightarrow C$, $D \rightarrow AC$, $D \rightarrow E$} and

G = {$A \rightarrow BC$, $D \rightarrow AE$}.

Show that F and G are equivalent

Superkeys, Keys & Prime Attributes

- **A set of attributes X is a superkey of schema R (with FDs F) if F implies $X \rightarrow R$**
- **A set of attributes X is a key of schema R if**
  - X is a superkey, and
  - No proper subset of X is a superkey
- **An attribute A in R is a prime attribute if A is contained in some key of R; otherwise, it is a nonprime attribute**

Example

Consider MovieList (title, director, address, phone, time) with FDs

$$\{address, time\} \rightarrow \{title\}$$
$$\{address\} \rightarrow \{phone\}$$
$$\{title\} \rightarrow \{director\}$$

- **{address, time} is the only key of MovieList**
- **{address, time} are the only prime attributes in MovieList**
- **Any superset of {address, time} is a superkey of MovieList**

Example

Consider R(A, B, C, D) with FDs F = {$A \rightarrow C$, $B \rightarrow D$}

Is AB a superkey?

- $AB \rightarrow ABC$ (augmentation of $A \rightarrow B$ with AB)
- $ABC \rightarrow ABCD$ (augmentation of $B \rightarrow D$ with ABC)
- $AB \rightarrow ABCD$ (transitivity with (1) and (2))
- Hence, AB is a superkey.

Closure of a Set of FDs

- Computing F^+ for a set of FDs F is not efficient as the size of F^+ could be exponentially large
- Consider R(A,B,C) with F = {{A} →{B}, {B} →{C}}

Completely Non-trivial FDs in F^+

$A \rightarrow B$	$B \rightarrow C$
$A \rightarrow C$	$AB \rightarrow C$
$A \rightarrow BC$	$AC \rightarrow B$

Remaining Non-trivial FDs in F^+

$A \rightarrow AB$	$AB \rightarrow BC$
$A \rightarrow AC$	$AB \rightarrow ABC$
$A \rightarrow ABC$	$AC \rightarrow AB$
$B \rightarrow BC$	$AC \rightarrow BC$
$AB \rightarrow AC$	$AC \rightarrow ABC$

Trivial FDs in F^+

$\emptyset \rightarrow \emptyset$	$BC \rightarrow \emptyset$
$A \rightarrow \emptyset$	$BC \rightarrow B$
$A \rightarrow A$	$BC \rightarrow C$
$B \rightarrow \emptyset$	$BC \rightarrow BC$
$B \rightarrow B$	$ABC \rightarrow \emptyset$
$C \rightarrow \emptyset$	$ABC \rightarrow A$
$C \rightarrow C$	$ABC \rightarrow B$
$AB \rightarrow \emptyset$	$ABC \rightarrow C$
$AB \rightarrow A$	$ABC \rightarrow AB$
$AB \rightarrow B$	$ABC \rightarrow AC$
$AB \rightarrow AB$	$ABC \rightarrow BC$
$AC \rightarrow \emptyset$	$ABC \rightarrow ABC$
$AC \rightarrow A$	
$AC \rightarrow C$	
$AC \rightarrow AC$	

Attribute Closure

- More efficient to compute the closure of a set of attributes
- Let $X \subseteq R$ and F be a set of FDs that hold on R
- Closure of X (with respect to F), denoted by X^+, is the set of attributes that are functionally determined by X with respect to F

Algorithm: Computing Attribute Closure

Input: Set of attributes $X \subseteq R$

Set of FDs F on R

Output: X^+ w.r.t. F

Initialize $X_0 = X$ and $i = 0$

Repeat

$X_{i+1} = X_i \cup Z$ such that there is some FD $Y \rightarrow Z \in F$ and $Y \subseteq X_i$

Until when $X_{i+1} = X_i$

Return X_i

Example

Let F = { AB → C, C → A, BC → D, ACD → B, D → EG, BE → C, CG → BD, CE → AG }.

Compute the closure of BD.

i	**X_i**	**FD used**
0	BD	Given
1	BDEG	D → EG
2	BCDEG	BE → C
3	ABCDEG	CE → AG
4	ABCDEG	none

Thus, BD^+ = ABCDEG

Example

Let $F = \{A \rightarrow C, B \rightarrow C, CD \rightarrow E\}$.

Show that F implies $AD \rightarrow E$.

i	X_i	FD used
0	AD	given
1	ACD	$A \rightarrow C$
2	ACDE	$CD \rightarrow E$
3	ACDE	none

Thus, $AD^+ = ACDE$

Since $E \in AD^+$, therefore F implies $AD \rightarrow E$

Equivalence of Sets of FDs

- **We can use attribute closure to determine if two sets of FDs F and G are equivalent**
- **For each FD $X \rightarrow Y \in F$**
  - Compute X+ with respect to G
  - $X \rightarrow Y \in G^+$ if $Y \subseteq X+$
- **Do the same for each FD in G**

Redundant Attributes in FDs

- **An attribute $A \in X$ is redundant in the FD $X \rightarrow B$ if $(F - \{X \rightarrow B\} \cup \{X - A \rightarrow B\})$ is equivalent to F**
- **How to check if $A \in X$ is redundant in $X \rightarrow B$?**
  - Compute $(X - A)^+$ w.r.t. F
  - $A \in X$ is redundant in $X \rightarrow B$ if $B \in (X - A)^+$
- **Example: What are the redundant attributes in $F = \{AB \rightarrow C, A \rightarrow B, B \rightarrow A\}$?**
  - A in AB → C is redundant since B+ = ABC
  - B in AB → C is redundant since A+ = ABC

Redundant FDs

- **A FD $f \in F$ is redundant if $(F - \{f\})$ is equivalent to F**
- **How to check if an FD $X \rightarrow A$ is redundant in F ?**
    - Compute X^+ w.r.t. $F - \{ X \rightarrow A \}$
    - $X \rightarrow A$ is redundant in F if $A \in X^+$
- **Example: What are the redundant FDs in $\{A \rightarrow B, A \rightarrow C, B \rightarrow A, B \rightarrow C, C \rightarrow A\}$?**

Minimal Cover for FDs

- **A set of FDs G is a minimal cover for a set of FDs F if and only if**
  - Every FD in G is of the form $X \rightarrow A$ where X is a set of attributes, A is a single attribute
  - For each FD $X \rightarrow A$ in G, X has no redundant attributes
  - There are no redundant FDs in G
  - F and G are equivalent

Algorithm: Computing Minimal Cover

Use decomposition rule to obtain FD with one attribute on RHS

Remove redundant attribute from LHS of each FD

Remove redundant FDs

Input: Set of FDs F

Output: G, a minimal cover for F

Initialize $G = \emptyset$

For each FD $X \rightarrow B_1 \ldots B_n \in F$

  $G = G \cup \{ X \rightarrow B_i \mid i \in [1, n] \}$

For each FD $X \rightarrow B \in G$

  initialize X' = X

  For each $A \in X$ do

   If ($B \in (X' - A)^+$ w.r.t. G) then

    replace $X' \rightarrow B$ in G by $X' - A \rightarrow B$

    $X' = X' - A$

For each FD $X \rightarrow B \in G$

  If ($B \in X^+$ w.r.t. $G - \{X \rightarrow B\}$) then

   remove $X \rightarrow B$ from G

Return G

Example

Find a minimal cover for F = {ABCD $\rightarrow$ E, E $\rightarrow$ D, A $\rightarrow$ B, AC $\rightarrow$ D}

1. **Decompose FDs**
   - All FDs in F have a single attribute on the RHS; G = F
2. **Eliminate redundant attributes**
   - B in ABCD $\rightarrow$ E is redundant since ACD^+ w.r.t. G is ABCDE

     G = {ACD $\rightarrow$ E, E $\rightarrow$ D, A $\rightarrow$ B, AC $\rightarrow$ D}
   - D in ACD $\rightarrow$ E is redundant since AC^+ w.r.t. G is ABCDE

     G = {AC $\rightarrow$ E, E $\rightarrow$ D, A $\rightarrow$ B, AC $\rightarrow$ D}
3. **Eliminate redundant FDs**
   - AC $\rightarrow$ D is redundant since AC^+ w.r.t. G – {AC $\rightarrow$ D} is ABCDE

***G = {AC $\rightarrow$ E, E $\rightarrow$ D, A $\rightarrow$ B}* is a minimal cover for F**

Extended Minimal Cover

- An extended minimal cover is obtained by using union rule to combine FDs with the same LHS

- Example: Find an extended minimal cover for

  $F = \{AB \rightarrow C,\ C \rightarrow A,\ BC \rightarrow D,\ ACD \rightarrow B,$

  $D \rightarrow EG,\ BE \rightarrow C,\ CG \rightarrow BD,\ CE \rightarrow AG\ \}$

Summary

- Bad schema designs can result in data redundancy and various update anomalies
- Schema refinement aims to eliminate bad schema designs by decomposing a bad relational schema into smaller schemas
- Functional dependencies are used to characterize properties of good/bad schemas