Introduction to Databases

(INFR10080)

(Normal Forms)

Instructor: Yang Cao

(Fall 2025)

THE UNIVERSITY *of* EDINBURGH

Changelog

v25.0 Initial version

Example of bad design

BAD

Title	Director	Theatre	Address	Time	Price
Inferno	Ron Howard	Vue	Omni Centre	20:00	11.50
Inferno	Ron Howard	Vue	Omni Centre	22:30	10.50
Inferno	Ron Howard	Odeon	Lothian Rd	20:00	10.00
Inferno	Ron Howard	Cineworld	Fountain Park	18:20	9.50
Inferno	Ron Howard	Cineworld	Fountain Park	21:00	11.00
Trolls	Mike Mitchell	Vue	Omni Centre	16:10	9.50
Trolls	Mike Mitchell	Vue	Omni Centre	19:30	10.00
Trolls	Mike Mitchell	Odeon	Lothian Rd	15:00	8.50
Trolls	Mike Mitchell	Cineworld	Fountain Park	17:15	9.00

$\{ \text{Title} \to \text{Director};\ \text{Theatre,Title,Time} \to \text{Price};\ \text{Theatre} \to \text{Address} \}$

Why is BAD bad?

Redundancy

Many facts are repeated

- For every showing we list both director and title
- For every movie playing we repeat the address

Update anomalies

- Address must be changed for all movies and showtimes
- If a movie stops playing, association title-director is lost
- Cannot add a movie before it starts playing

Good design

Movies: Title → Director

Title	Director
Inferno	Ron Howard
Trolls	Mike Mitchell

Theatres: Theatre → Address

Theatre	Address
Vue	Omni Centre
Odeon	Lothian Rd
Cineworld	Fountain Park

Showings: Theatre,Title,Time → Price

Theatre	Title	Time	Price
Vue	Inferno	20:00	11.50
Vue	Inferno	22:30	10.50
Odeon	Inferno	20:00	10.00
Cineworld	Inferno	18:20	9.50
Cineworld	Inferno	21:00	11.00
Vue	Trolls	16:10	9.50
Vue	Trolls	19:30	10.00
Odeon	Trolls	15:00	8.50
Cineworld	Trolls	17:15	9.00

Why is GOOD good?

No redundancy

Every FD defines a key

No information loss

$$\begin{aligned} \text{Movies} &= \pi_{\text{Title,Director}}(\text{BAD}) \\ \text{Theatres} &= \pi_{\text{Theatre,Address}}(\text{BAD}) \\ \text{Showings} &= \pi_{\text{Theatre,Title,Time,Price}}(\text{BAD}) \\ \text{BAD} &= \text{Movies} \bowtie \text{Theatres} \bowtie \text{Showings} \end{aligned}$$

No constraints are lost

All of the original FDs appear as constraints in the new tables

Boyce-Codd Normal Form (BCNF)

Problems with bad designs are caused by non-trivial FDs $X \to Y$ where **X is not a key**

> A relation with FDs Σ is in BCNF if for every $X \to Y$ in Σ
>
> - $Y \subseteq X$ (the FD is trivial), or
> - X is a key

A database is in BCNF if all relations are in BCNF

Decompositions

Given a set of attributes U and a set of FDs Σ,
a decomposition of (U, Σ) is a set

$$(U_1, \Sigma_1), \ldots, (U_n, \Sigma_n)$$

such that $U = \bigcup_{i=1}^{n} U_i$ and Σ_i is a set of FDs over U_i

BCNF decomposition if each (U_i, Σ_i) is in BCNF

Criteria for good decompositions

Losslessness: no information is lost

Dependency preservation: no constraints are lost

Good decompositions

A decomposition of (U, Σ) into $(U_1, \Sigma_1), \ldots, (U_n, \Sigma_n)$ is

Lossless if for every relation R over U that satisfies Σ

- each $\pi_{U_i}(R)$ satisfies Σ_i, and
- $R = \pi_{U_1}(R) \bowtie \cdots \bowtie \pi_{U_n}(R)$

Dependency preserving if Σ and $\bigcup_{i=1}^{n} \Sigma_i$ are equivalent
(that is, they have the same closure)

Projection of FDs

Let Σ be a set of FDs over attributes U

The **projection** of Σ on $V \subseteq U$

$$\pi_V(\Sigma) = \{(X \to Y) \in \Sigma^+ \mid X, Y \subseteq V\}$$

is the set of all FDs over V that are implied by Σ

BCNF decomposition algorithm (non-examinable)

Input: A set of attributes U and a set of FDs Σ

Output: A database schema S

1. $S := \{(U, \Sigma)\}$
2. While there is $(U_i, \Sigma_i) \in S$ not in BCNF:
   Replace (U_i, Σ_i) by **decompose**(U_i, Σ_i)
3. Remove any (U_i, Σ_i) for which there is (U_j, Σ_j) with $U_i \subseteq U_j$
4. Return S

Subprocedure **decompose**(U, Σ):

1. Choose $(X \to Y) \in \Sigma$ that violates BCNF
2. Set $V := C_\Sigma(X)$ and $Z := U - V$
3. Return $\big(V, \pi_V(\Sigma)\big)$ and $\big(XZ, \pi_{XZ}(\Sigma)\big)$

Properties of the BCNF algorithm (non-examinable)

- The decomposed schema is in **BCNF** and **lossless-join**
- The output depends on the FDs chosen to decompose
- Dependency preservation is **not guaranteed**

Example

Apply the BCNF algorithm to the BAD schema (blackboard)

BCNF and dependency preservation

Take the relation Lectures : **C**lass, **P**rofessor, **T**ime
with FDs $\Sigma = \{C \to P, PT \to C\}$

$(\mathcal{CPT}, \Sigma)$ is **not in BCNF**:

$(C \to P) \in \Sigma$ is non-trivial and C is not a key

If we decompose using the BCNF algorithm we get

$(\mathcal{CP}, C \to P)$ and $(CT, \varnothing)$

We lose the constraint $PT \to C$

Third Normal Form (3NF)

(U, Σ) is in 3NF if for every FD $X \to Y$ in Σ
one of the following holds:

- $Y \subseteq X$ (the FD is trivial)
- X is a key
- all of the attributes in Y are prime

Intuition: in 3NF FDs where the l.h.s. is not a key are allowed as long as the **r.h.s. consists only of prime attributes**

Every schema in BCNF is also in 3NF

3NF and redundancy

Consider again the relation Lectures : **C**lass, **P**rofessor, **T**ime
with FDs $\Sigma = \{C \to P, PT \to C\}$

(CPT, Σ) is in 3NF: PT is a candidate key, so P is prime

More redundancy than in BCNF

- each time a class appears in a tuple,
  professor's name is repeated
- we tolerate this because there is no BCNF decomposition
  that preserves dependencies

Minimal covers

Let Σ and Γ be sets of FDs

Γ is a cover of Σ if $\Gamma^+ = \Sigma^+$

Minimal if

- Each FD in Γ has the form $X \to A$
- No proper subset of Γ is a cover
  (we cannot remove FDs without losing equivalence to Σ)
- For $(X \to A) \in \Gamma$ and $X' \subset X$, $\quad A \notin C_\Sigma(X')$
  (we cannot remove attributes from the LHS of FDs in Γ)

Intuition: Γ is a small representation of all FDs in Σ

Finding minimal covers

1. Put the FDs in standard form: only one attribute on RHS
   Use Armstrong's decomposition axiom

   $X \to A_1 \cdots A_n$ is split into n FDs: $X \to A_1, \ldots, X \to A_n$

2. Minimize the LHS of each FD
   Check whether attributes in the LHS can be removed

   For $(X \to A) \in \Sigma$ and $X' \subset X$ check whether $A \in C_\Sigma(X')$
   If yes, replace $X \to A$ by $X' \to A$ and repeat

3. Delete redundant FDs
   $(X \to A) \in \Sigma$, check whether $\Sigma - \{X \to A\} \models X \to A$

Finding minimal covers: Example

Consider the FDs $\{ABCD \to E, E \to D, A \to B, AC \to D\}$

1. Already in standard form

   $$\{ABCD \to E, E \to D, A \to B, AC \to D\}$$

2. The LHS of $ABCD \to E$ can be replaced by AC

   $$\{AC \to E, E \to D, A \to B, AC \to D\}$$

3. The last FD is redundant (implied by the first two)

   $$\{AC \to E, E \to D, A \to B\}$$

3NF synthesis algorithm

Input : A set of attributes U and a set of FDs Σ

Output : A database schema S

1. $S := \varnothing$
2. Find a minimal cover Γ of Σ
3. Replace all FDs $X \to A_1, \ldots, X \to A_n$ in Γ by $X \to A_1 \cdots A_n$ (note that the FDs have the same l.h.s.)
4. For each FD $(X \to Y) \in \Gamma$, add $(XY, X \to Y)$ to S
5. If no (U_i, Σ_i) in S is such that U_i is key for (U, Σ), find a key K for (U, Σ) and add $(K, \varnothing)$ to S
6. If S contains (U_i, Σ_i) and (U_j, Σ_j) with $U_i \subseteq U_j$, replace them by $(U_j, \Sigma_i \cup \Sigma_j)$
7. Output S

Properties of the 3NF algorithm

The synthetized schema is

- in **3NF**
- **lossless-join**
- **dependency-preserving**

Example

Apply the 3NF algorithm to the Lectures schema (blackboard)
(that schema is already in 3NF, but let's do it anyway)

3NF synthesis: another example

Example

Input : $\big(ABCD, \{A \to B, C \to B, BD \to A\}\big)$

Not in 3NF (the only candidate key is CD)

The given set of FDs is already minimal

Ouput : $\Big\{ (CB, \{C \to B\}),$
$(ABD, \{BD \to A, A \to B\}),$
$(CD, \varnothing) \Big\}$

Schema design: Summary

Given the set of attributes U and the set of FDs F

Find a **lossless**, **dependency-preserving** decomposition into:

BCNF if it exists

3NF if BCNF decomposition cannot be found

Database administrators may decide to **de-normalize** tables
to reduce number of joins