Normal Forms

Dr Paolo Guagliardo

THE UNIVERSITY of EDINBURGH
informatics

Fall 2020 (v20.2.0)

This page is intentionally left blank

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

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

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 $\to$ Director

Title	**Director**
Inferno	Ron Howard
Trolls	Mike Mitchell

Theatres: Theatre $\to$ Address

Theatre	**Address**
Vue	Omni Centre
Odeon	Lothian Rd
Cineworld	Fountain Park

Showings: Theatre, Title, Time $\to$ 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 F is in BCNF if for every $X \to Y$ in F

- $Y \subseteq X$ (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 F,
a decomposition of (U, F) is a set

$$(U_1, F_1), \ldots, (U_n, F_n)$$

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

BCNF decomposition if each (U_i, F_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, F) into $(U_1, F_1), \ldots, (U_n, F_n)$ is

Lossless if for every relation R over U that satisfies F

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

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

Projection of FDs

Let F be a set of FDs over attributes U

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

$$\pi_V(F) = \{X \to Y \mid X, Y \subseteq V,\ Y \subseteq C_F(X)\}$$

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

Can be often represented compactly as a set of FDs F' over V s.t.

$$\forall X, Y \subseteq V \quad F' \models X \to Y \iff F \models X \to Y$$

BCNF decomposition algorithm

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

Output: A database schema S

1. $S := \{(U, F)\}$
2. While there is $(U_i, F_i) \in S$ not in BCNF:

   Replace (U_i, F_i) by **decompose**(U_i, F_i)
3. Remove any (U_i, F_i) for which there is (U_j, F_j) with $U_i \subseteq U_j$
4. Return S

Subprocedure **decompose**(U, F):

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

Properties of the BCNF algorithm

- 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 $F = \{C \to P, PT \to C\}$

(CPT, F) is **not in BCNF**:

$$(C \to P) \in F \text{ is non-trivial and } C \text{ is not a key}$$

If we decompose using the BCNF algorithm we get

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

We lose the constraint $PT \to C$

Third Normal Form (3NF)

(U, F) is in 3NF if for every FD $X \to Y$ in F
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 $F = \{C \to P, PT \to C\}$

(CPT, F) 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 F and G be sets of FDs

G is a cover of F if $G^+ = F^+$

Minimal if

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

Intuition: G is a small representation of all FDs in F

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 $(X \to A) \in F$ and $X' \subset X$ check whether $A \in C_F(X')$
   If yes, replace $X \to A$ by $X' \to A$ and repeat

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

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 F

Output : A database schema S

1. $S := \varnothing$
2. Find a minimal cover G of F
3. Replace all FDs $X \to A_1, \ldots, X \to A_n$ in G by $X \to A_1 \cdots A_n$ (note that the FDs have the same l.h.s.)
4. For each FD $(X \to Y) \in G$, add $(XY, X \to Y)$ to S
5. If no (U_i, F_i) in S is such that U_i is key for (U, F), find a key K for (U, F) and add $(K, \varnothing)$ to S
6. If S contains (U_i, F_i) and (U_j, F_j) with $U_i \subseteq U_j$, replace them by $(U_j, F_i \cup F_j)$
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 : $\{\ (CB, \{C \to B\}),$

$(ABD, \{BD \to A, A \to B\}),$

$(CD, \varnothing)\ \}$

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