COSC265 — Relational Database Systems

Neville Churcher

Department of Computer Science & Software Engineering
University of Canterbury

2021

Things we know

- ☆ FDs represent facts about a relation scheme and constraints on extensions
- ☆ Keys are (sets of) attributes which functionally determine scheme R
- ☆ If K is a key, and a subset of K is also a key, then K is a *superkey*
- ☆ If a relation has more than one key, then each is a *candidate key*
- ☆ One candidate key is chosen to be the *primary key* (PK) and the others become *secondary keys*
- ☆ A *prime attribute* is a member of some candidate key

Normalisation Overview

- ☆ If we just have one enormous relation (the *universal relation*) then our database will suffer from update anomalies
- ☆ Normalisation is the process of obtaining relations whose schemas contain attributes which "belong" together
- ☆ Data dependencies (such as FDs) allow us to formalise the process and determine whether desirable properties hold
- ☆ *Normal forms* are statements about relation properties and provide tests we can apply

Decomposition v Synthesis

- ☆ Decompose (by projection) universal relation to produce database schema
- ☆ In practice, begin with set of relations resulting from analysis and decompose 'non-normal' relations
- ☆ Alternative is *synthesis*—start with FDs and construct normalised relations

- ☞ Normalisation (normally) leads to more relations in database
- ☞ Normalisation can involve *trade-offs*. Having many smaller relations may remove the risk of update anomalies but require more (expensive) join operations in queries
- ☞ *Denormalisation* involves deliberately storing data in lower normal forms — generally *for performance* reasons

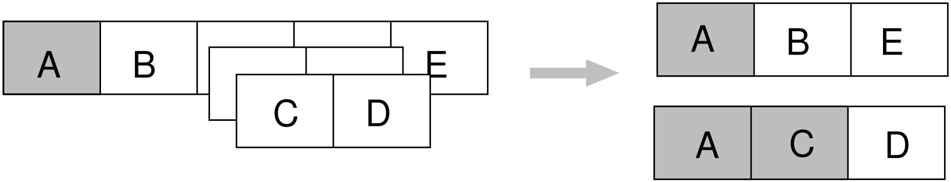
The Golden Rule

The Golden Rule

"The key,
the whole key,
and nothing but the key"

The Key

1NF: Remove repeating groups

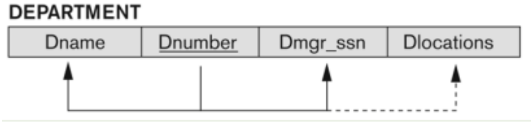


- ☆ Attribute values are single atomic values
- ☆ "Flat" tuples — no array, list, . . . fields or nested relations
- ☆ Essentially the definition of a relation
- ☆ All attributes depend on the key

1NF Example

See text . . .

Schema



Extension

DEPARTMENT

Dname	Dnumber	Dmgr_ssn	Dlocations
Research	5	333445555	{Bellaire, Sugarland, Houston}
Administration	4	987654321	{Stafford}
Headquarters	1	888665555	{Houston}

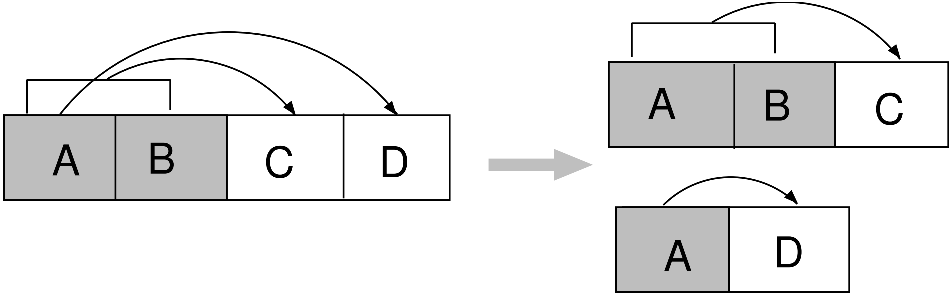
1NF

DEPARTMENT

Dname	Dnumber	Dmgr_ssn	Dlocation
Research	5	333445555	Bellaire
Research	5	333445555	Sugarland
Research	5	333445555	Houston
Administration	4	987654321	Stafford
Headquarters	1	888665555	Houston

The Whole Key

2NF: Remove non-full dependence



- ☆ Every non-prime attribute is *fully functionally dependent* on the PK (and other candidate keys)

2NF Example

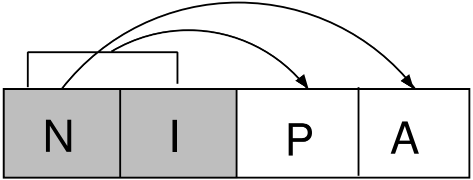
The whole key

Consider scheme SNAME (N), SADDRESS (A), ITEM (I), PRICE (P)

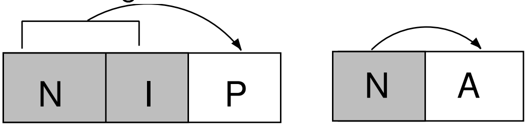
$R = \{NAIP\}$ $\mathcal{F} = \{N \rightarrow A, NI \rightarrow P\}$ PK (K) is NI

- ☆ $K \rightarrow P \equiv NI \rightarrow P$ is *full* since $N \not\rightarrow P$ and $I \not\rightarrow P$
- ☆ $K \rightarrow A \equiv NI \rightarrow P$ is *partial* since $N \rightarrow A$

1NF but not 2NF

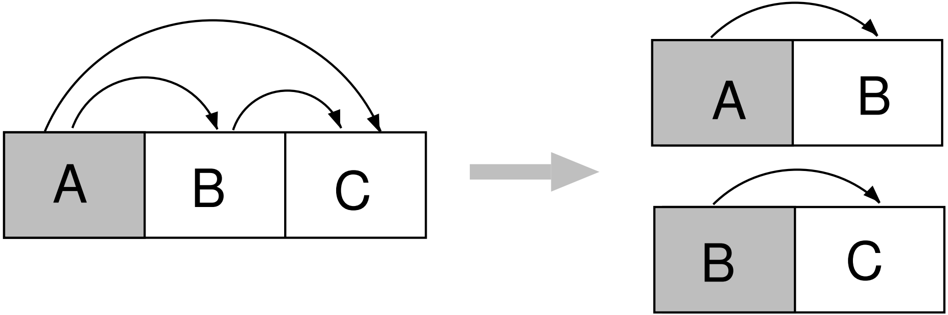


Remove non-full dependence to get



And Nothing But The Key

3NF: Remove transitive dependence



- ☆ In 2NF and no non-prime attribute is transitively dependent on the PK
- ☆ Whenever FD $X \rightarrow A$ folds in R, then either:
  - ★ X is a superkey of R or
  - ★ A is a prime attribute of R

Third Normal Form

Definition (3NF)

- ☆ In 2NF and no non-prime attribute is transitively dependent on the PK
- ☆ Whenever FD $X \rightarrow A$ folds in R, then either:
  - ★ X is a superkey of R or
  - ★ A is a prime attribute of R

Example

$$R = ABCDEFGHIJ \quad \mathcal{F} = \{AB \to C, A \to DE, B \to F, F \to GH, D \to IJ\}$$

Key

What is the key of R?

2NF

A ‖ DEIJ	$\{A \to DE, D \to IJ\}$
B ‖ FGH	$\{B \to F, F \to GH\}$
AB ‖ C	$\{AB \to C\}$

3NF

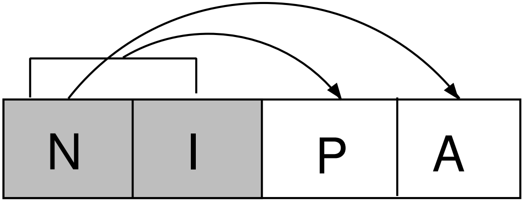
A ‖ DE	$\{A \to DE\}$
D ‖ IJ	$\{D \to IJ\}$
B ‖ F	$\{B \to F\}$
F ‖ GH	$\{F \to GH\}$
AB ‖ C	$\{AB \to C\}$

Decomposition Example

Can anything go wrong?

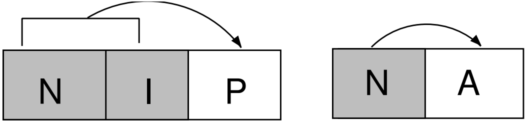
Remember SNAME (N), SADDRESS (A), ITEM (I), PRICE (P)

$R = \{NAIP\} \quad \mathcal{F} = \{N \to A, NI \to P\}$



1NF but not 2NF

Remove non-full dependence to get



☆ Do these relations contain the same information as the original one?

Non-Loss Decomposition

- Consider $r_(R)$, where $R = ABC, \mathcal{F} = \{A \rightarrow B, C \rightarrow B\}$
- Decompose into $r_1(R_1)$ and $r_2(R_2)$ where $R_1 = AB, \mathcal{F}_1 = \{A \rightarrow B\}$ and $R_2 = BC, \mathcal{F}_2 = \{C \rightarrow B\}$
- Re-join using $r' = r_1 \underset{B}{\bowtie} r_2$ — recover original relation?

A	B	C
a1	b1	c1
a3	b1	c2
a2	b2	c3
a4	b2	c4

r_1

A	B
a1	b1
a3	b1
a2	b2
a4	b2

r_2

B	C
b1	c1
b1	c2
b2	c3
b2	c4

A	B	C	
a1	b1	c1	
a1	b1	c2	☆
a3	b1	c1	☆
a3	b1	c2	
a2	b2	c3	
a2	b2	c4	☆
a4	b2	c3	☆
a4	b2	c4	

Lossless Joins

Some descriptions & definitions

- ☆ R is a relation scheme, with FDs $\mathcal{F}$, decomposed into $\{R_1, R_2, \ldots, R_k\}$
- ☆ This is a *lossless-join decomposition with respect to $\mathcal{F}$* if $\forall r(R, \mathcal{F})$

$$r = \pi_{R_1}(r) \bowtie \pi_{R_2}(r) \bowtie \ldots \pi_{R_{k-1}}(r) \bowtie \pi_{R_k}(r)$$

- ☆ In other words, r is the natural join of its projections r_i onto the R_i
- ☆ If $\rho = \{R_1, R_2, \ldots, R_k\}$ then can define the *project-join mapping*

$$m_\rho(r) = \mathop{\bowtie}_{i=1}^{k} \pi_{R_i}(r)$$

- ☆ If ρ non-loss then $r = m_\rho(r)$. Otherwise, ρ is lossy i.e. $r \subseteq m_\rho(r)$
- ☆ $\pi_{R_i} m_\rho(r) = r_i$
- ☆ $m_\rho(r)$ is idempotent

Testing for Lossy Joins

The *chase* test

- ☆ Construct a *tableau* with a column for each attribute A_j and a row for each relation scheme R_i in the decomposition
- ☆ Symbols in column based on attribute name (i.e. x in the X column)
- ☆ Symbol in row i, column j is:

  unsubscripted if A_j is in R_i

  subscripted otherwise
- ☆ If a row contains all unsubscripted symbols then the decomposition is lossless

Example

$R = ABCD \quad \rho = \{AD, BCD, AC\}$

	A	B	C	D
(AD)	a	b_1	c_1	d
(BCD)	a_2	b	c	d
(AC)	a	b_3	c	d_3

Testing for Lossy Joins

The *chase* continues ...

- If a row contains all unsubscripted symbols then the decomposition is lossless
- If not, then consider the (equality-generating) FDs
- To consider $X \to Y$: for all rows that agree on the value of X, equate symbols corresponding to Y

Example

$R = ABCD \quad \rho = \{AD, BCD, AC\} \quad \mathcal{F} = \{A \to B, B \to C, CD \to A\}$

	A	B	C	D
(AD)	a	b_1	c_1	d
(BCD)	a_2	b	c	d
(AC)	a	b_3	c	d_3

A	B	C	D
a	b_1	c_1	d
a_2	b	c	d
a	b_1	c	d_3

A	B	C	D
a	b_1	c	d
a_2	b	c	d
a	b_1	c	d_3

☞ Equating a, a_2 makes tuple 2 all unscripted symbols $\therefore \rho$ is lossless

Decomposition Example: NAIP

Recall previous example: $R = \{SNAME, SADDRESS, ITEM, PRICE\} \equiv \{NAIP\}$
$R_1 = NA, R_2 = NIP \quad \mathcal{F} = \{N \to A, NI \to P\}$

	N	**A**	**I**	**P**
(R_1)	n	a	i_1	p_1
(R_2)	n	a_2	i	p

Consider $N \to A$: replace a_2 by a

	N	**A**	**I**	**P**
(R_1)	n	a	i_1	p_1
(R_2)	n	a	i	p

- ☆ Last row all unsubscripted so decomposition $\rho(NAIP) = \{NA, NIP\}$ is lossless
- ☆ Thus 2NF better than 1NF for this example

Decomposition Example

Recall previous example $R = ABC, R1 = AB, R2 = BC$,
$\rho(R) = \{R1, R2\}, \mathcal{F} = \{A \to B, C \to B\}$

	A	B	C
(R_1)	a	b	c_1
(R_2)	a_2	b	c

1. Consider $A \to B$: no matching LHS values
2. Consider $C \to B$: no matching LHS values

$\therefore \rho(R) = \{R1 = AB, R2 = BC\}$ is lossy — consistent with previous observation

- ☆ Many other properties of tableaux have been studied
- ☆ e.g. above procedure can be proved correct
- ☆ Can be applied to other kinds of dependencies

Larger Decomposition/Chase Example

$$R = ABCDE, \quad \rho(R) = \{AD, AB, BE, CDE, AE\}$$
$$\mathcal{F} = \{A \to C, B \to C, C \to D, DE \to C, CE \to A\}$$

$A \to C$ then $B \to C$
Could have equated to c_3 or c_5

A	B	C	D	E
a	b_1	c_1	d	e_1
a	b	c_2	d_2	e_2
a_3	b	c_3	d_3	e
a_4	b_4	c	d	e
a	b_5	c_5	d_5	e

$C \to D$ then $DE \to C$
then $CE \to A$

A	B	C	D	E
a	b_1	c_1	d	e_1
a	b	c_1	d_2	e_2
a_3	b	c_1	d_3	e
a_4	b_4	c	d	e
a	b_5	c_1	d_5	e

3rd (BE) row all a symbols $\implies$ ρ non-loss

	A	B	C	D	E
	a	b_1	c	d	e_1
	a	b	c	d	e_2
➤	a	b	c	d	e
	a	b_4	c	d	e
	a	b_5	c	d	e

A Special Case

That's handy!

- ☆ Simpler test for lossless-join available for common case of decomposition into *two* schemas
- ☆ $\rho(R) = \{R_1, R_2\}$ lossless w.r.t. $\mathcal{F}$ *iff* :

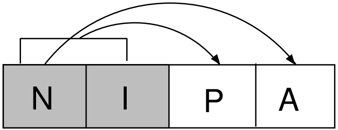
$$R_1 \cap R_2 \rightarrow R_1 - R_2 \quad \text{or} \quad R_1 \cap R_2 \rightarrow R_2 - R_1$$

- ☆ N.B. these FDs could be in $\mathcal{F}$ or $\mathcal{F}^+$ (i.e. inferred by $\mathcal{F}$)

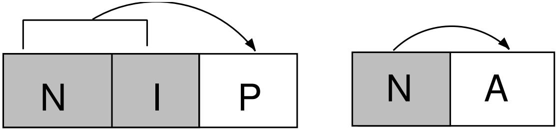
One More Time ...

$R = \{NAIP\} \quad \mathcal{F} = \{N \rightarrow A, NI \rightarrow P\}$

1NF but not 2NF



$\Longrightarrow$



- ☆ Do these relations contain the same information as the original one?
- ☆ $NIP \cap NA \rightarrow NIP \setminus NA$? $N \not\rightarrow IP$, but
- ☆ $NIP \cap NA \rightarrow NA \setminus NIP$? $N \rightarrow A$ ✔
- ☆ Lossless — 2NF (still) better that 1NF for this case

2-Schema Lossless-Join Decomposition Test

Is every 2-schema decomposition lossless?

Example (2-Schema Lossless-Join Decomposition Test)

$R = ABC, \mathcal{F} = \{A \rightarrow B\}$

$$
\begin{array}{cc}
\multicolumn{2}{c}{\rho_1} \\
R_1 & R_2 \\
AB & AC
\end{array}
\qquad\qquad
\begin{array}{cc}
\multicolumn{2}{c}{\rho_2} \\
R_1 & R_2 \\
AB & BC
\end{array}
$$

$$
\begin{array}{ccc}
AB \cap AC = A & R_1 \cap R_2 & AB \cap BC = B \\
AB - AC - B & R_1 - R_2 & AB - BC = A \\
AC - AB = C & R_2 - R_1 & BC - AB = C
\end{array}
$$

$A \rightarrow B \in \mathcal{F} \therefore \rho_1$ lossless

$B \rightarrow A \notin \mathcal{F}^+, B \rightarrow C \notin \mathcal{F}^+ \therefore \rho_2$ lossy $\square$

Dependency-Preserving Decompositions

Do the constraints described by FDs still apply after normalisation?

- ☆ Lossless-join decompositions are desirable because a relation can always be recovered from its projections (e.g. in queries involving $\bowtie$)
- ☆ Another important property of decompositions $\rho(R) = \{R_1, R_2, \ldots, R_i, \ldots, R_k\}$ is that the dependencies $\mathcal{F}$ for R are implied by their projections onto the R_i

Definition (Dependency Projection)

The projection of $\mathcal{F}$ onto attributes Z is

$$\pi_Z(\mathcal{F}) = \{X \to Y | X \to Y \in \mathcal{F}^+ \wedge XY \subseteq Z\}$$

ρ *preserves* $\mathcal{F}$ if, $\forall f \in \mathcal{F}, \quad \cup_i \pi_{R_i}(\mathcal{F}) \models f$,

- ☆ FDs represent *integrity constraints* for R. *Update anomalies* can occur if ρ does not preserve $\mathcal{F}$ — *even if ρ is lossless*.
- ☆ Consequence: can't safely update $r_i(R_i)$ without checking all $\mathcal{F}$ in R

Dependency Preservation Example

Example (**C**ity, **S**treet, **Z**ip code)

$$R = CSZ, \quad \mathcal{F} = \{CS \to Z, Z \to C\}, \quad \rho = \{SZ, CZ\}$$

- ☆ ρ is lossless as $(SZ \cap CZ) \to (CZ - SZ) \quad (\equiv Z \to C)$
- ☆ $\pi_{SZ}(\mathcal{F})$ gives only trivial FDs by reflexivity
- ☆ $\pi_{CZ}(\mathcal{F}) = \{Z \to C, + \text{ trivial FDs}\}$
- ☆ $\pi_{SZ}(\mathcal{F}) \cup \pi_{CZ}(\mathcal{F}) \not\models CS \to Z$ (prove it!)

$\therefore \rho$ does not preserve $\mathcal{F}$ $\square$

CSZ Example — Consequences

r_1

S	Z
19739 River Road	97027
19739 River Road	98119
...	...

2nd tuple could be added in error.
Satisfies $\pi_{SZ}(\mathcal{F})$

r_2

C	Z
Portland, OR	97027
Portland, OR	98119
Portland, MN	55555
...	...

Satisfies $\pi_{CZ}(\mathcal{F})$

$r_1 \bowtie r_2$

C	S	Z
Portland, OR	19739 River Road	97027
Portland, OR	19739 River Road	98119
...	...	...

- ☆ Violates $CS \rightarrow Z$
- ☆ Dependencies not preserved
- ☆ But decomposition is lossless

and *vice versa* . . .

- ☆ Have seem that ρ may be lossless-join but not preserve $\mathcal{F}$
- ☆ Could we have ρ that preserves $\mathcal{F}$ but is lossy?

Example (Dependency-Preserving Lossy Join)

Consider $R = ABCD, \quad \rho = \{AB, CD\}, \quad \mathcal{F} = \{A \rightarrow B, C \rightarrow D\}$

$\pi_{AB}(\mathcal{F}) = A \rightarrow B \quad \pi_{CD}(\mathcal{F}) = C \rightarrow D$
$\pi_{AB}(\mathcal{F}) \cup \pi_{CD}(\mathcal{F}) = \mathcal{F}$
$\therefore \rho$ is dependency-preserving

Apply two schema lossless-join decomposition test:

- ☆ $AB \cap CD \rightarrow AB - CD$ or $AB \cap CD \rightarrow CD - AB$
- ☆ $AB \cap CD = \emptyset, \quad AB - CD = AB, \quad CD - AB = CD$
- ☆ $\emptyset \not\rightarrow AB$, $\emptyset \not\rightarrow CD$
- ☆ $\therefore \rho$ is lossy, yet preserves $\mathcal{F}$ $\square$

Keys & Prime Attributes

Definition (Non-Prime Attribute)

If K_i are the candidate keys for schema R and $A \subseteq R$ then attribute A is non-prime if

$$\nexists K_i : A \subseteq K_i$$

$\square$

- ☆ If P is prime in R then $\|\{K_i | P \subseteq K_i\}\| > 0$

Example (Keys and Primes)

Given the previous definitions, show that:

- ☆ CS and SZ are both keys
- ☆ All attributes of R are prime

Third Normal Form

Several equivalent definitions in common use. Important concepts are:

- *Independence* of non-prime attributes
- Non-prime attributes are fully-dependent on key

Definition (3NF — which do you prefer?)

- 3NF = 2NF + every non-prime attribute is non-transitively dependent on PK
- No non-prime attribute is functionally dependent on another non-prime attribute
- If $X \rightarrow A$ holds in R then either X is a superkey of R or A is prime in R. (Note: this can be applied directly, without constructing 2NF first)
- Every non-prime attribute of R is:
  - Fully functionally dependent on every key
  - Non-transitively dependent on every key

Implications

- ☆ 2NF allows dependencies where a non-prime attribute determines a prime attribute
- ☆ If $\mathcal{F} \supseteq \{S\# \rightarrow City, City \rightarrow Population\}$ then an update anomaly could occur when adding new tuples because of transitive dependency $S\# \rightarrow Population$.

S#	**City**	**Population**	**. . .**
42	Christchurch	400,000	. . .
. . .	. . .	. . .	
42	Christchurch	400,366	. . .

- ☆ 3NF avoids such anomalies.

Boyce-Codd Normal Form

As far as we go with FDs

- ☆ A relation scheme is in *BCNF* if every determinant (LHS of FD) is a superkey.
- ☆ All BCNF relations are also 3NF
- ☆ A 3NF relation may not be BCNF

Example (Zip Codes)

$R = CSZ, \mathcal{F} = \{Z \to C, CS \to Z\}$
R is 3NF — all attributes are prime
R is not BCNF — $Z \to C$ but Z not a superkey (only CS, SZ, CSZ are) □

- ☆ *Any* relation scheme has a lossless-join decomposition into BCNF. However, this decomposition may not preserve dependencies.
- ☆ *Any* relation scheme has a decomposition into 3NF that is *both* lossless and dependency-preserving.