A brief introduction to database theory

Marcelo Arenas

PUC Chile

Outline

- Relational schemas
- Queries
- Data dependencies
- Normal forms
- Connections between normalization theory and information theory

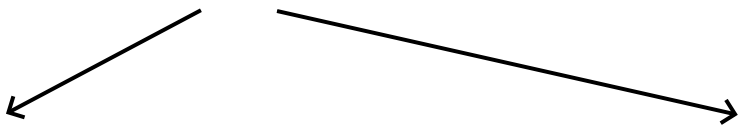
A relational database

Course

Number	Title	Section	Room
CS201	Databases	1	AL1
CS201	Databases	2	AL2
CS201	Databases	3	AL3
CS201	Databases	3	GB1
CS300	Machine learning	1	GB1

The schema of a relation

Relation schema: $R[U]$



relation name

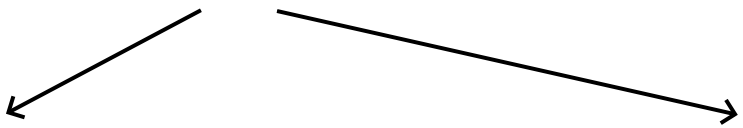
set of attributes $\{A_1, \ldots, A_k\}$

Course[{Number, Title, Section, Room}]

We use notation **Course[Number, Title, Section, Room]**

The schema of a relation

Relation schema: $R[U]$



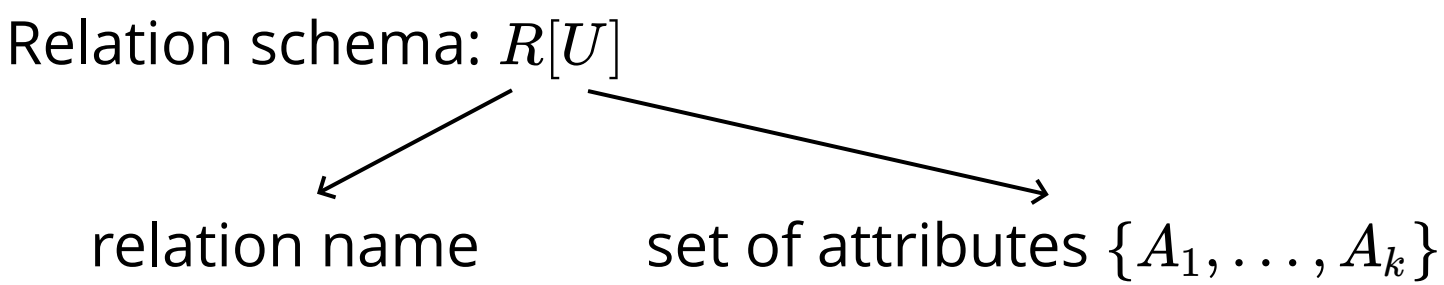
relation name

set of attributes $\{A_1, \ldots, A_k\}$

$\mathrm{dom}(A_i)$: domain of attribute A_i

U-tuple t assigns a value $t(A_i)$ to each attribute A_i, where $t(A_i) \in \mathrm{dom}(A_i)$

The schema of a relation

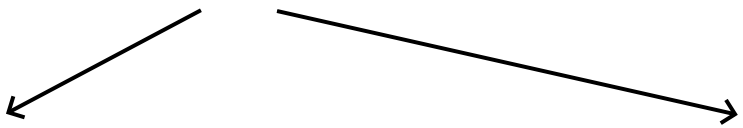


(**Number:** CS201, **Title:** Databases, **Section:** 1, **Room:** AL1)

(CS201, Databases, 1, AL1)

The schema of a relation

Relation schema: $R[U]$



relation name

set of attributes $\{A_1, \ldots, A_k\}$

An instance I of $R[U]$ is a finite set of U-tuples

CS201	Databases	1	AL1
CS201	Databases	2	AL2
CS201	Databases	3	AL3
CS201	Databases	3	GB1
CS300	Machine learning	1	GB1

A relational schema

Relational schema $S = \{R_1[U_1], R_2[U_2], \ldots, R_n[U_n]\}$

$R_i[U_i]$ is a relation schema

An instance I of S assigns to each symbol $R_i[U_i]$ an instance I^{R_i}

A relational schema

$S = \{$ **Course[Number, Title, Section, Room],**
Location[Room, Address] $\}$

$I^{\mathbf{Course}} =$

CS201	Databases	1	AL1
CS201	Databases	2	AL2
CS201	Databases	3	AL3
CS201	Databases	3	GB1
CS300	Machine learning	1	GB1

$I^{\mathbf{Location}} =$

AL1	95 Queen's Park
AL2	95 Queen's Park
AL3	95 Queen's Park
GB1	35 St. George Street

Querying a database

We introduce a widely used query language for relational databases: **relational algebra**

It has the same expressive power as **first-order logic**

Selection: $\sigma_{A=a}(I)$, $\sigma_{A=B}(I)$

$\sigma_{\mathbf{Room}=\mathrm{GB1}}\left(\right.$

Number	Title	Section	Room
CS201	Databases	1	AL1
CS201	Databases	2	AL2
CS201	Databases	3	AL3
CS201	Databases	3	GB1
CS300	Machine learning	1	GB1

$\left.\right) =$

Number	Title	Section	Room
CS201	Databases	3	GB1
CS300	Machine learning	1	GB1

Projection: $\pi_X(I)$, where X is a set of attributes

$\pi_{\{\mathbf{Number},\mathbf{Title}\}}$ (

Number	Title	Section	Room
CS201	Databases	1	AL1
CS201	Databases	2	AL2
CS201	Databases	3	AL3
CS201	Databases	3	GB1
CS300	Machine learning	1	GB1

) =

Set semantics ⟶

Number	Title
CS201	Databases
CS300	Machine learning

Projection: $\pi_X(I)$, where X is a set of attributes

$\pi_{\{\mathbf{Number},\mathbf{Title}\}}$ (

Number	Title	Section	Room
CS201	Databases	1	AL1
CS201	Databases	2	AL2
CS201	Databases	3	AL3
CS201	Databases	3	GB1
CS300	Machine learning	1	GB1

) =

Bag semantics ⟶

Number	Title
CS201	Databases
CS201	Databases
CS201	Databases
CS201	Databases
CS300	Machine learning

Rename: $\rho_{A \to B}(I)$

$\rho_{\mathbf{Number} \to \mathbf{ID}}$ (

Number	Title	Section	Room
CS201	Databases	1	AL1
CS201	Databases	2	AL2
CS201	Databases	3	AL3
CS201	Databases	3	GB1
CS300	Machine learning	1	GB1

) =

ID	Title	Section	Room
CS201	Databases	1	AL1
CS201	Databases	2	AL2
CS201	Databases	3	AL3
CS201	Databases	3	GB1
CS300	Machine learning	1	GB1

Join: $I \bowtie J$

Number	Title	Section	Room
CS201	Databases	1	AL1
CS201	Databases	2	AL2
CS201	Databases	3	AL3
CS201	Databases	3	GB1
CS300	Machine learning	1	GB1

$\bowtie$

Room	Address
AL1	95 Queen's Park
AL2	95 Queen's Park
AL3	95 Queen's Park
GB1	35 St. George Street

$=$

ID	Title	Section	Room	Address
CS201	Databases	1	AL1	95 Queen's Park
CS201	Databases	2	AL2	95 Queen's Park
CS201	Databases	3	AL3	95 Queen's Park
CS201	Databases	3	GB1	35 St. George Street
CS300	Machine learning	1	GB1	35 St. George Street

Union and difference

Union and difference operators are defined as the usual set theoretic operators

Notation: $I \cup J$ and $I - J$

- I and J are instances of the same relation schema $R[U]$

Relational algebra is compositional

Room	Address
AL1	95 Queen's Park
AL2	95 Queen's Park
AL3	95 Queen's Park
GB1	35 St. George Street

$$\bowtie \quad \rho_{\mathbf{Address}\to\mathbf{B}}\left(\rho_{\mathbf{Room}\to\mathbf{A}}\left(\right.\right.$$

Room	Address
AL1	95 Queen's Park
AL2	95 Queen's Park
AL3	95 Queen's Park
GB1	35 St. George Street

$$\left.\left.\right)\right) =$$

Room	Address	A	B
AL1	95 Queen's Park	AL1	95 Queen's Park
AL1	95 Queen's Park	AL2	95 Queen's Park
AL1	95 Queen's Park	AL3	95 Queen's Park
AL1	95 Queen's Park	GB1	35 St. George Street
AL2	95 Queen's Park	AL1	95 Queen's Park
...	...	...	...

Evaluation of a query

Let Q be a relational algebra expression over a relational schemas S, and I be an instance of S

$Q(I)$ denotes the relation obtained by evaluating Q over I

- A tuple t is an answer to Q over I if $t \in Q(I)$

Evaluation of a query

Theorem: the problem of verifying, given Q, I, t, whether $t \in Q(I)$ is PSPACE-complete

Equivalence with first-order logic

Assume that I, J are instances of $R[A, B]$ and $S[A, C]$, respectively

$$\sigma_{A=B}(I) \longrightarrow R(x, y) \wedge x = y$$

$$\sigma_{A=a}(I) \longrightarrow R(x, y) \wedge x = a$$

$$\pi_{A}(I) \longrightarrow \exists y\, R(x, y)$$

$$I \bowtie J \longrightarrow R(x, y) \wedge S(x, z)$$

Equivalence with first-order logic

I, J are instances of $R[A, B]$ and $S[A, C]$, respectively

Return the values of attribute A that appear in R but not in S

$$\pi_A(R) - \pi_A(S) \quad \longrightarrow \quad Q(x) \;=\; \exists y\, R(x, y) \wedge \neg \exists z\, S(x, z)$$

A fundamental fragments: conjunctive queries

The fragment of relational algebra defined by the operator $\sigma_{A=a}$, $\sigma_{A=B}$, π_X, $\bowtie$

The fragment of first-order logic consisting of queries of the form

$$Q(\bar{x}) \;=\; \exists \bar{y}\,(R_1(\bar{u}_1) \wedge \cdots \wedge R_n(\bar{u}_n)),$$

where each $\bar{u}_i$ is a tuple of variables and constants

Evaluation of a conjunctive query

Theorem: the problem of verifying, given a conjunctive query Q, I, t, whether $t \in Q(I)$ is NP-complete

Why is query evaluation NP-hard?

Reduction from the graph 3-coloring problem

How to avoid this high complexity?

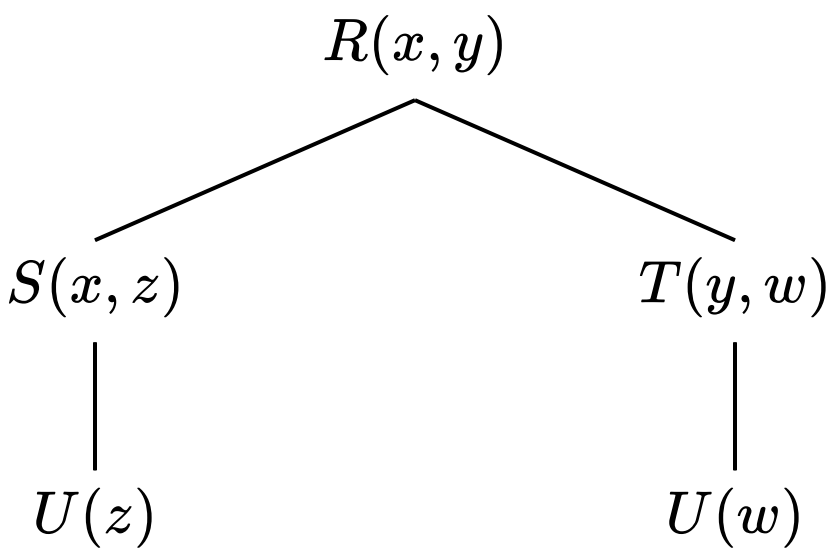
Consider the query:

$$q_1 = \exists x \exists y \exists z \exists w \left(R(x, y) \wedge S(x, z) \wedge T(y, w) \wedge U(z) \wedge U(w) \right)$$

How to avoid this high complexity?

$$\exists x \exists y \exists z \exists w \left(R(x,y) \wedge S(x,z) \wedge T(y,w) \wedge U(z) \wedge U(w) \right)$$

Tree structure



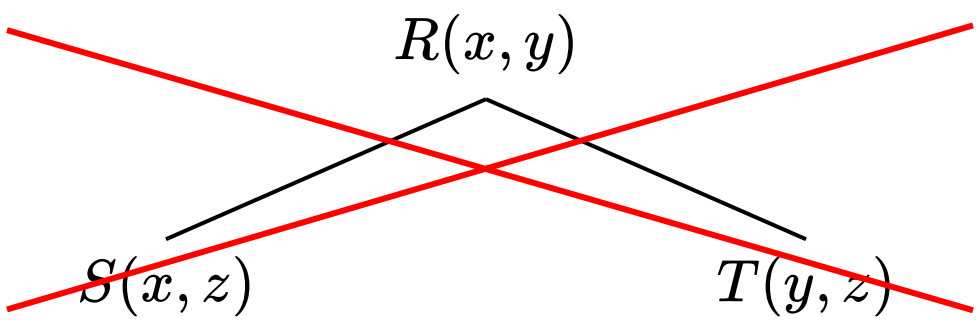
How to avoid this high complexity?

Now consider the query:

$$q_2 = \exists x \exists y \exists z \left(R(x, y) \wedge S(x, z) \wedge T(y, z) \right)$$

How to avoid this high complexity?

$$q_2 = \exists x \exists y \exists z \left(R(x,y) \wedge S(x,z) \wedge T(y,z) \right)$$



How to avoid this high complexity?

$$q_2 = \exists x \exists y \exists z \left(R(x, y) \wedge S(x, z) \wedge T(y, z) \right)$$

Several ways to measure how acyclic is a query have been studied: treewidth, hypertree width, ...

Polynomial-time evaluation for classes of queries with bounded degree of acyclicity

Data dependencies

Number	Title	Section	Room
CS201	Databases	1	AL1
CS201	Databases	2	AL2
CS201	Databases	3	AL3
CS201	Databases	3	GB1
CS300	Machine learning	1	GB1

Each course must have a unique title

Data dependencies

Number	Title	Section	Room
CS201	Databases	1	AL1
CS201	Databases	2	AL2
CS201	Databases	3	AL3
CS201	Databases	3	GB1
CS300	Machine learning	1	GB1

Number $\rightarrow$ Title

Data dependencies

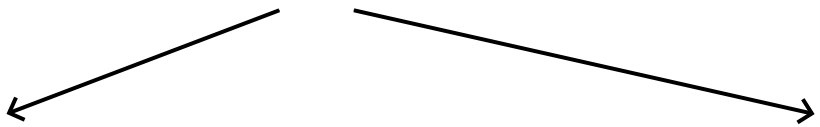
Number	Title	Section	Room
CS201	Databases	1	AL1
CS201	Databases	2	AL2
CS201	Databases	3	AL3
CS201	Databases	3	GB1
CS300	Machine learning	1	GB1

This table does **not** satisfies the data dependency

$\{\textbf{Number}, \textbf{Section}\} \rightarrow \textbf{Room}$

A complete definition of a schema

Database schema: (S, Σ)



relational schema

set of data dependencies over S

I is an instance of (S, Σ) if I is an instance of S and I satisfies each data dependency in Σ

A complete definition of a schema

We will introduce some fundamental classes of data dependencies

Functional dependencies

A functional dependency over a relation schema $R[U]$ is an expression $X \to Y$ with $X, Y \subseteq U$

For a U-tuple t, we use notation $t[X]$ for the restriction of t to the set of attribute X

If t = (**Number:** CS201, **Title:** Databases, **Section:** 1, **Room:** AL1), then

t[**Number**, **Title**] = (**Number:** CS201, **Title:** Databases)

Functional dependencies

A functional dependency over a relation schema $R[U]$ is an expression $X \to Y$ with $X, Y \subseteq U$

An instance I of $R[U]$ satisfies $X \to Y$ if for every $t_1, t_2 \in I$:

if $t_1[X] = t_2[X]$, then $t_1[Y] = t_2[Y]$

We use notation $I \models X \to Y$ to indicate that I satisfies $X \to Y$

Key dependencies

A key dependency over a relation schema $R[U]$ is a functional dependency $X \to U$ with $X \subseteq U$

- We use X to denote the key dependency

Key dependencies

Number	Title	Section	Room
CS201	Databases	1	AL1
CS201	Databases	2	AL2
CS201	Databases	3	AL3
CS201	Databases	3	GB1
CS300	Machine learning	1	GB1

Given that **Number** $\rightarrow$ **Title** is in the schema,

{**Number**, **Section, Room}** is a key dependency

Key dependencies

Number	Title	Section	Room
CS201	Databases	1	AL1
CS201	Databases	2	AL2
CS201	Databases	3	AL3
~~CS201~~	~~Databases~~	~~3~~	~~GB1~~
CS300	Machine learning	1	GB1

Key dependencies

Number	Title	Section	Room
CS201	Databases	1	AL1
CS201	Databases	2	AL2
CS201	Databases	3	AL3
CS300	Machine learning	1	GB1

If {**Number**, **Section**} $\rightarrow$ **Room** is in the schema,
{**Number**, **Section}** is a key dependency

The implication problem for functional dependencies

Let $\Sigma \cup \{X \to Y\}$ be a set of functional dependencies over a relational schema S

$I \models \Sigma$ if $I \models \varphi$ for every $\varphi \in \Sigma$

$\Sigma \models X \to Y$ if for every instance I of S:

if $I \models \Sigma$, then $I \models X \to Y$

The implication problem for functional dependencies

$\{\textbf{Number} \rightarrow \textbf{Title},\ \{\textbf{Number, Section}\} \rightarrow \textbf{Room}\} \models$

$\{\textbf{Number}, \textbf{Section}\} \rightarrow \{\textbf{Number}, \textbf{Title}, \textbf{Section}, \textbf{Room}\}$

Number	Title	Section	Room
CS201	Databases	1	AL1
CS201	Databases	2	AL2
CS201	Databases	3	AL3
CS300	Machine learning	1	GB1

The implication problem for functional dependencies

Theorem: The implication problem for functional dependencies can be solved in linear time

Multivalued dependencies

Theater	Movie	Snack
Cineplex	CODA	coffee
Cineplex	CODA	popcorn
Cineplex	Belfast	coffee
Cineplex	Belfast	popcorn
Carlton	Dune	fries
Carlton	Dune	popcorn
Carlton	CODA	fries
Carlton	CODA	popcorn

Theater $\twoheadrightarrow$ **Movie**

Multivalued dependencies

A multivalued dependency over a relation schema $R[U]$ is an expression $X \twoheadrightarrow Y$ with $X, Y \subseteq U$

We use notation XY for $X \cup Y$

Multivalued dependencies

A multivalued dependency over a relation schema $R[U]$ is an expression $X \twoheadrightarrow Y$ with $X, Y \subseteq U$

Let $Z = U - XY$

An instance I of $R[U]$ satisfies $X \twoheadrightarrow Y$ if for every $t_1, t_2 \in I$ with $t_1[X] = t_2[X]$, there exists $t_3 \in I$ such that:

$$t_3[XY] = t_1[XY] \text{ and } t_3[XZ] = t_2[XZ]$$

The implication problem for functional and multivalued dependencies

Notice that $X \rightarrow Y \models X \twoheadrightarrow Y$

Theorem: The implication problem for functional and multivalued dependencies can be solved in polynomial time

Join dependencies

A join dependency over a relation schema $R[U]$ is an expression $\bowtie [X_1, X_2, \ldots, X_n]$ with $X_1, X_2, \ldots, X_n \subseteq U$

An instance I of $R[U]$ satisfies $\bowtie [X_1, X_2, \ldots, X_n]$ if :

$$I = \pi_{X_1}(I) \bowtie \pi_{X_2}(I) \bowtie \cdots \bowtie \pi_{X_n}(I)$$

Join dependencies

I $=$

Theater	**Movie**	**Snack**
Cineplex	CODA	coffee
Cineplex	CODA	popcorn
Cineplex	Belfast	coffee
Cineplex	Belfast	popcorn
Carlton	Dune	fries
Carlton	Dune	popcorn
Carlton	CODA	fries
Carlton	CODA	popcorn

$\bowtie \left[\{\textbf{Theater}, \textbf{Movie}\}, \{\textbf{Theater}, \textbf{Snack}\}\right]$

Join dependencies

$\pi_{\{\mathbf{Theater}, \mathbf{Movie}\}}(I)$

Theater	Movie
Cineplex	CODA
Cineplex	Belfast
Carlton	Dune
Carlton	CODA

$\pi_{\{\mathbf{Theater}, \mathbf{Snack}\}}(I)$

Theater	Snack
Cineplex	coffee
Cineplex	popcorn
Carlton	fries
Carlton	popcorn

$$I \quad = \quad \pi_{\{\mathbf{Theater}, \mathbf{Movie}\}}(I) \bowtie \pi_{\{\mathbf{Theater}, \mathbf{Snack}\}}(I)$$

The implication problem for functional, multivalued and join dependencies

Theorem: The implication problem for functional, multivalued and join dependencies is NP-hard, and can be solved in exponential time

Update anomalies

Number $\rightarrow$ Title

Number	Title	Section	Room
CS201	Databases	1	AL1
CS201	Databases	2	AL2
CS201	Databases	3	AL3
CS201	Databases	3	GB1
CS300	Machine learning	1	GB1

Update anomalies

Number → Title

Number	Title	Section	Room
CS201	Databases I	1	AL1
CS201	Databases	2	AL2
CS201	Databases	3	AL3
CS201	Databases	3	GB1
CS300	Machine learning	1	GB1

Update anomalies

Number $\rightarrow$ Title

Number	Title	Section	Room
CS201	Databases I	1	AL1
CS201	Databases I	2	AL2
CS201	Databases I	3	AL3
CS201	Databases I	3	GB1
CS300	Machine learning	1	GB1

Insertion anomalies

Number $\rightarrow$ Title

Number	Title	Section	Room
CS201	Databases I	1	AL1
CS201	Databases I	2	AL2
CS201	Databases I	3	AL3
CS201	Databases I	3	GB1
CS300	Machine learning	1	GB1

Insertion anomalies

Number $\rightarrow$ Title

Number	Title	Section	Room
CS201	Databases I	1	AL1
CS201	Databases I	2	AL2
CS201	Databases I	3	AL3
CS201	Databases I	3	GB1
CS300	Machine learning	1	GB1
CS202	Databases II	?	?

Deletion anomalies

Number $\rightarrow$ Title

Number	Title	Section	Room
CS201	Databases I	1	AL1
CS201	Databases I	2	AL2
CS201	Databases I	3	AL3
CS201	Databases I	3	GB1
CS300	Machine learning	1	GB1

Deletion anomalies

Number $\rightarrow$ Title

Number	Title	Section	Room
CS201	Databases I	1	AL1
CS201	Databases I	2	AL2
CS201	Databases I	3	AL3
CS201	Databases I	3	GB1
CS300	Machine learning	1	GB1

Deletion anomalies

Number $\rightarrow$ Title

Number	Title	Section	Room
CS201	Databases I	1	AL1
CS201	Databases I	2	AL2
CS201	Databases I	3	AL3
CS201	Databases I	3	GB1
~~CS300~~	~~Machine learning~~	~~1~~	~~GB1~~

Normal forms

A normal form imposes syntactic restrictions on a database schema (S, Σ)

- To avoid update, insertion and deletion anomalies

Functional dependencies: BCNF

$(R[U], \Sigma)$ is in BCNF if for every non-trivial functional dependency $X \to A$ such that $\Sigma \models X \to A$:

X is a key dependency over $R[U]$

A data dependency φ over a relation schema $R[U]$ is trivial if $I \models \varphi$ for every instance I of $R[U]$

- A functional dependency $X \to A$ is trivial if and only if $A \in X$

Functional dependencies: BCNF

(**Course**[**Number**, **Title**, **Section**, **Room**],
Number $\rightarrow$ **Title**)

is not in BCNF

since **Number** is not a key dependency

Transforming to BCNF

Number $\rightarrow$ Title

Number	Title	Section	Room
CS201	Databases	1	AL1
CS201	Databases	2	AL2
CS201	Databases	3	AL3
CS201	Databases	3	GB1
CS300	Machine learning	1	GB1

Transforming to BCNF

Number $\rightarrow$ Title

Number	Title
CS201	Databases
CS300	Machine learning

Number	Section	Room
CS201	1	AL1
CS201	2	AL2
CS201	3	AL3
CS201	3	GB1
CS300	1	GB1

Transforming to BCNF

(**Course**[**Number**, **Title**, **Section**, **Room**],
Number $\rightarrow$ **Title**)

is replaced by

(**CourseInfo**[**Number**, **Title**], **Number** $\rightarrow$ **Title**)

(**Semester**[**Number**, **Section**, **Room**], $\emptyset$)

Testing BNCF

Theorem: the problem of verifying, given a database schema $(R[U], \Sigma)$ with Σ a set of functional dependencies, whether $(R[U], \Sigma)$ is in BCNF can be solved in polynomial time

Functional and multivalued dependencies: 4NF

$(R[U], \Sigma)$ is in 4NF if for every non-trivial mutivalued dependency $X \twoheadrightarrow Y$ such that $\Sigma \models X \twoheadrightarrow Y$:

X is a key dependency over $R[U]$

Functional dependencies: BCNF

(**Cinema**[**Theater**, **Movie**, **Snack**],
Theater $\twoheadrightarrow$ **Movie**)

is not in 4NF

since **Theater** is not a key dependency

Transforming to 4NF

Theater $\twoheadrightarrow$ Title

Theater	Movie	Snack
Cineplex	CODA	coffee
Cineplex	CODA	popcorn
Cineplex	Belfast	coffee
Cineplex	Belfast	popcorn
Carlton	Dune	fries
Carlton	Dune	popcorn
Carlton	CODA	fries
Carlton	CODA	popcorn

Transforming to 4NF

Theater	Movie
Cineplex	CODA
Cineplex	Belfast
Carlton	Dune
Carlton	CODA

Theater	Snack
Cineplex	coffee
Cineplex	popcorn
Carlton	fries
Carlton	popcorn

Transforming to 4NF

(**Cinema**[**Theater**, **Movie**, **Snack**],
Theater $\twoheadrightarrow$ **Movie**)

is replaced by

(**CinemaMovie**[**Theater**, **Movie**], $\emptyset$)

(**CinemaSnack**[**Theater**, **Snack**], $\emptyset$)

Testing 4NF

Theorem: the problem of verifying, given a database schema $(R[U], \Sigma)$ with Σ a set of functional and multivalued dependencies, whether $(R[U], \Sigma)$ is in 4NF can be solved in polynomial time

Normal forms for functional and join dependencies

Several definitions can be found in the literature:

5NF

$\Downarrow$

PJ/NF (Projection/Join NF)

$\Downarrow$

5NFR (Reduced-5NF)

Information theory to the rescue

The goal is to develop tools for testing when a normal form correspond to a *good* design

Based on information theory, a measure of information content of an element in a database is defined

Information content (or amount of uncertainty)

A	B	C
1	2	3
1		4

$\mathbf{A \rightarrow B}$

A	B	C
1	2	3
1	2	4
1		5

Defining the measure

$\mathbf{A \rightarrow B}$

A	B	C
1	2	3
1	2	4

Defining the measure

$\mathbf{A \to B}$

A	B	C
1		3
1	2	4

$$\mathrm{dom}(I) \subseteq \{1, \ldots, k\}$$

set of values occurring in I

Defining the measure

$\mathbf{A \rightarrow B}$

A	B	C
1		3
	2	4

$\mathrm{dom}(I) \subseteq \{1, \ldots, k\}$

X: set of positions

$\Pr(\square \mid \{\ \square\ \})$

Defining the measure

$\mathbf{A \to B}$

A	B	C
1		3
	2	4

$\mathrm{dom}(I) \subseteq \{1, 2, 3, 4, 5, 6\}$

$\Pr(\square \mid \{\square\})$

$$\Pr(2 \mid X) = \frac{6}{}$$

$$\Pr(a \mid X) = \frac{5}{} \qquad a \neq 2$$

Defining the measure

$\mathbf{A \to B}$

A	B	C
1		3
	2	4

$\mathrm{dom}(I) \subseteq \{1, 2, 3, 4, 5, 6\}$

$\Pr(\square \mid \{\square\})$

$$\Pr(2 \mid X) = \frac{6}{6 + 5 \cdot 5} = \frac{6}{31}$$

$$\Pr(a \mid X) = \frac{5}{6 + 5 \cdot 5} = \frac{5}{31} \qquad a \neq 2$$

Defining the measure

$\mathbf{A \rightarrow B}$

A	B	C
1	2	3
1	2	4

Defining the measure

$\mathbf{A \to B}$

A	B	C
1		3
1	2	4

$$\mathrm{dom}(I) \subseteq \{1, 2, 3, 4, 5, 6\}$$

Defining the measure

$\mathbf{A \rightarrow B}$

A	B	C
1		3
	2	

$\mathrm{dom}(I) \subseteq \{1, 2, 3, 4, 5, 6\}$

X: set of positions

$\Pr(\square \mid \{\square, \square\})$

Defining the measure

$\mathbf{A \to B}$

A	B	C
1		3
	2	

$\mathrm{dom}(I) \subseteq \{1, 2, 3, 4, 5, 6\}$

X: set of positions

$\Pr(\square \mid \{\square, \square\})$

$$\Pr(2 \mid X) = \frac{35}{35 + 5 \cdot 30}$$

$$\Pr(a \mid X) = \frac{30}{35 + 5 \cdot 30} \qquad a \neq 2$$

The use of conditional entropy

Parameters:

- A databases schema (S, Σ)
- An instance I of (S, Σ)
- A position p in I
- A value k (to restrict the domain of instances to be considered)

$\mathrm{Inf}_I^k(p \mid \Sigma)$ will be used as the measure of the amount of information in position p

The use of conditional entropy

$$\mathrm{Inf}_I^k(p \mid \Sigma) = \sum_{X\,:\,\text{set of positions}} \left(\Pr(X) \sum_{a \in \{1,\ldots,k\}} \Pr(a \mid X) \log \frac{1}{\Pr(a \mid X)} \right)$$

The measure depends on k

We would like it to be defined for an arbitrary large value of k

The general measure

The value $\mathrm{Inf}_I^k(p \mid \Sigma)$ is compared with the maximum entropy $\log k$

$$\mathrm{Inf}_I(p \mid \Sigma) \quad = \quad \lim_{k \to \infty} \frac{\mathrm{Inf}_I^k(p \mid \Sigma)}{\log k}$$

Lemma: the measure is well defined for data dependencies defined in first-order logic

A general notion of being well designed

A database schema (S, Σ) is well designed if for every instance I of (S, Σ) and every position p in I:

$$\mathrm{Inf}_I(p \mid \Sigma) = 1$$

Some basic properties

The measure does not depend on a particular representation of constraints:
If Σ_1 and Σ_2 are equivalent, then $\mathrm{Inf}_I(p \mid \Sigma_1) = \mathrm{Inf}_I(p \mid \Sigma_2)$

A	B	C
1	2	3
1		4

0.875

A → B

A	B	C
1	2	3
1	2	4
1		5

0.781

Justifying normal forms

If Σ is a set of functional dependencies:

(S, Σ) is well-designed if and only if (S, Σ) is in BCNF

If Σ is a set of functional and multivalued dependencies:

(S, Σ) is well-designed if and only if (S, Σ) is in 4NF

Justifying normal forms

If Σ is a set of functional and join dependencies:

- If (S, Σ) is in PJ/NF or in 5NFR, then (S, Σ) is well-designed. The converse is not true
- A syntactic characterization of being well-designed has been developed

Thanks!

Background material

Normalization algorithm

Transforms a database schema (S, Σ) into a database schema (S', Σ') that conforms to a normal form

Two desirable properties of the algorithm: **information losslessness** and **dependency preservation**

Information losslessness

We consider normalization algorithms that transform a database schema $(R[U], \Sigma)$ into a database schema (S', Σ') such that $S' = \{R_1[U_1], R_2[U_2], \ldots, R_k[U_k]\}$ and $U_1, U_2, \ldots, U_k \subseteq U$

Information losslessness

(S', Σ') is a lossless decomposition of $(R[U], \Sigma)$ if for every instance I of $(R[U], \Sigma)$, there exists an instance J of (S', Σ') such that:

$$J^{R_i} = \pi_{U_i}(I) \text{ for every } i \in \{1, \ldots, k\}$$

$$I = J^{R_1} \bowtie J^{R_2} \bowtie \cdots \bowtie J^{R_k}$$

The examples of decomposition shown in this presentation are lossless

Dependency preservation

For functional dependencies, we consider normalization algorithms that transform a database schema $(R[U], \Sigma)$ into a database schema $\{(R_1[U_1], \Sigma_1), \ldots, (R_k[U_k], \Sigma_k)\}$ such that $U_1, \ldots, U_k \subseteq U$

Dependency preservation

$\{(R_1[U_1], \Sigma_1), \ldots, (R_k[U_k], \Sigma_k)\}$ is a dependency preserving decomposition of $(R[U], \Sigma)$ if:

$$\Sigma \equiv \Sigma_1 \cup \cdots \cup \Sigma_k$$

For functional dependencies, the examples of decomposition shown in this presentation are dependency preserving