DESIGN THEORY FOR RELATIONAL DATABASES

1

Introduction

2

- There are always many different schemas for a given set of data.
- E.g., you could combine or divide tables.
- How do you pick a schema? Which is better? What does "better" mean?
- Fortunately, there are some principles to guide us.

2

Schemas and Constraints

3

- Consider the following sets of schemas:

  Students(macid, name, email)

  vs.

  Students(macid, name)
  Emails(macid, address)
- Consider also:

  House(street, city, value, owner, propertyTax)

  vs.

  House(street, city, value, owner)
  TaxRates(city, value, propertyTax)

Constraints are domain-dependent

Acknowledgements: R. J. Miller, M. Papagelis

3

Avoid redundancy

4

This table has redundant data, and that can lead to anomalies.

name	addr	beersLiked	manf	favBeer
Janeway	Voyager	Bud	A.B.	WickedAle
Janeway	Voyager	WickedAle	Pete's	WickedAle
Spock	Enterprise	Bud	A.B.	Bud

- Update anomaly: if Janeway is transferred to *Intrepid*, will we remember to change each of her tuples?
- Deletion anomaly: If nobody likes Bud, we lose track of the fact that Anheuser-Busch manufactures Bud.

4

Database Design Theory

5

- It allows us to improve a schema systematically.
- General idea:
  - Express constraints on the data
  - Use these to decompose the relations
- Ultimately, get a schema that is in a "normal form" that guarantees good properties, such as no anomalies.
- "Normal" in the sense of conforming to a standard.
- The process of converting a schema to a normal form is called normalization.

5

6

Part I: Functional Dependency Theory

6

Keys

7

- K is a *key* for R if K uniquely determines all of R, and no proper subset of K does.
- K is a *superkey* for relation R if K contains a key for R.
  ("superkey" is short for "superset of key".)

7

Example

8

RegNum	Surname	FirstName	BirthDate	DegreeProg
284328	Smith	Luigi	29/04/59	Computing
296328	Smith	John	29/04/59	Computing
587614	Smith	Lucy	01/05/61	Engineering
934856	Black	Lucy	01/05/61	Fine Art
965536	Black	Lucy	05/03/58	Fine Art

- `RegNum` is a key: i.e., `RegNum` is a superkey and it contains a sole attribute, so it is minimal.
- `{Surname, Firstname, BirthDate}` is another key

8

Functional Dependencies

9

- Need a special type of constraint to help us with normalization
- $X \rightarrow Y$ is an assertion about a relation *R* that whenever two tuples of *R* agree on all the attributes in set *X*, they must also agree on all attributes in set *Y*.
- E.g., suppose X = {AB}, Y = {C}

R

A	B	C
x1	y1	c2
x1	y1	c2
x2	y2	c3
x2	y2	c3

9

Functional Dependencies

10

- Say "$X \rightarrow Y$ holds in *R*."
  "*X* functionally determines *Y*."
- Convention: …, *X*, *Y*, *Z* represent sets of attributes; *A*, *B*, C,… represent single attributes.
- Convention: no braces used for sets of attributes, just *ABC*, rather than {*A*,*B*,*C* }.

10

Why "functional dependency"?

11

- "dependency" because the value of *Y* depends on the value of *X*.
- "functional" because there is a mathematical function that takes a value for *X* and gives a *unique* value for *Y*.

11

Properties about FDs

12

- Rules
  - Splitting/combining
  - Trivial FDs
  - Armstrong's Axioms
- Algorithms related to FDs
  - the closure of a set of attributes of a relation
  - a minimal basis of a relation

12

Splitting Right Sides of FDs

13

- $X \rightarrow A_1A_2\ldots A_n$ holds for R exactly when each of $X \rightarrow A_1$, $X \rightarrow A_2$,…, $X \rightarrow A_n$ hold for R.
- Example: A→BC is equivalent to A→B and A→C.
- Combining: if A→F and A→G, then A → FG
- There is no splitting rule for the left side
  - ABC → DEF is NOT the same as AB→DEF and C→DEF!
- We'll generally express FDs with singleton right sides.

13

Example: FDs

14

Drinkers(name, addr, beersLiked, manf, favBeer)

Reasonable FDs to assert:

- name → addr, favBeer.
  - Note this FD is the same as: name → addr and name → favBeer.
- beersLiked → manf

14

Example: Possible Data

15

name	addr	beersLiked	manf	favBeer
Janeway	Voyager	Bud	A.B.	WickedAle
Janeway	Voyager	WickedAle	Pete's	WickedAle
Spock	Enterprise	Bud	A.B.	Bud

Because name → addr

Because name → favBeer

Because beersLiked → manf

15

Trivial FDs

16

- Not all functional dependencies are useful
  - A → A always holds
  - ABC → A also always holds (right side is subset of left side)
- FD with an attribute on both sides
  - ABC → AD becomes ABC → D
  - Or, in singleton form, delete trivial FDs
    ABC → A and ABC → D becomes just ABC → D

16

Superkey

17

Drinkers(name, addr, beersLiked, manf, favBeer)

- {name, beersLiked} is a superkey because together these attributes determine all the other attributes.
  - name → addr, favBeer
  - beersLiked → manf

name	addr	beersLiked	manf	favBeer
Janeway	Voyager	Bud	A.B.	WickedAle
Janeway	Voyager	WickedAle	Pete's	WickedAle
Spock	Enterprise	Bud	A.B.	Bud

17

Example: Key

18

- {name, beersLiked} is a key because neither {name} nor {beersLiked} is a key on its own.
  - name doesn't → manf; beersLiked doesn't → addr.
- There are no other keys, but lots of superkeys.
  - Any superset of {name, beersLiked}.

name	addr	beersLiked	manf	favBeer
Janeway	Voyager	Bud	A.B.	WickedAle
Janeway	Voyager	WickedAle	Pete's	WickedAle
Spock	Enterprise	Bud	A.B.	Bud

18

FDs are a generalization of keys

19

- Functional dependency: X → Y
- Superkey: X → R
- A superkey must include all the attributes of the relation on the RHS.
- An FD can involve just a subset of them
  - Example:

Houses (street, city, value, owner, tax)
- street,city → value, owner, tax *(both FD and key)*
- city,value → tax *(FD only)*

19

Identifying functional dependencies

20

- FDs are domain knowledge
  - Intrinsic features of the data you're dealing with
  - Something you know (or assume) about the data
- Database engine cannot identify FDs for you
  - Designer must specify them as part of schema
  - DBMS can only enforce FDs when told to
- DBMS cannot "optimize" FDs either
  - It has only a finite sample of the data
  - An FD constrains the entire domain

20

Coincidence or FD?

21

ID	Email	City	Country	Surname
1983	tom@gmail.com	Bern	Switzerland	Mendes
8624	jones@bell.com	London	Canada	Jones
9141	scotty@gmail.com	Winnipeg	Canada	Jones
1204	birds@gmail.com	Aachen	Germany	Lakemeyer

- In this instance:
  - Surname → Country
  - City → Country
- Are these FDs?

21

Coincidence or FD

22

- We have an FD only if it holds for every instance of the relation.
- You can't know this just by looking at one instance.
- You can only determine this based on knowledge of the domain.

22

Armstrong's Axioms

23

X, Y, Z are sets of attributes

1. **Reflexivity:** If $Y \subseteq X$, then $X \to Y$
2. **Augmentation:** If $X \to Y$, then $XZ \to YZ$ for any Z
3. **Transitivity:** If $X \to Y$ and $Y \to Z$, then $X \to Z$
4. **Union:** If $X \to Y$ and $X \to Z$, then $X \to YZ$
5. **Decomposition:** If $X \to YZ$, then $X \to Y$ and $X \to Z$

23

Inferring FDs

24

- Given a set of FDs, we can often infer further FDs.
- This will come in handy when we apply FDs to the problem of database design.

24

Dependency Inference

25

- Suppose we are given FDs
  $X_1 \rightarrow A_1$,
  $X_2 \rightarrow A_2$,
  …,
  $X_n \rightarrow A_n$.
- Does the FD $Y \rightarrow B$ also hold in any relation that satisfies the given FDs?
- Example: If $A \rightarrow B$ and $B \rightarrow C$ hold, surely $A \rightarrow C$ holds, even if we don't say so.
  $A \rightarrow C$ is ***entailed (implied)*** by $\{A \rightarrow B, B \rightarrow C\}$

25

Transitive Property

26

The transitive property holds for FDs

- Consider the FDs: $A \rightarrow B$ and $B \rightarrow C$; then $A \rightarrow C$ holds
- Consider the FDs: $AD \rightarrow B$ and $B \rightarrow CD$; then $AD \rightarrow CD$ holds or just $AD \rightarrow C$ (because of trivial FDs)

26

Method 1: Prove it from first principles

27

- To test if $Y \rightarrow B$, start by assuming two tuples agree on all attributes of Y.

← Y →

t1: aaaaa bb. . . b

t2: aaaaa ?? . . . ?

27

Example

28

ClientID	Income	OtherProd	Rate	Country	City	State
225	High	A	2.1%	USA	San Francisco	MD
420	High	A	2.1%	USA	San Francisco	CA
333	High	B	3.0%	USA	San Francisco	CA
576	High	B	3.0%	USA	San Francisco	CA
128	Low	C	4.5%	UK	Reading	Berkshire
193	Low	C	4.5%	UK	London	London
550	Low	B	3.5%	UK	London	London

F1: [Income, OtherProd] → [Rate]   F2: [Country, City] → [State]

How to prove it in the general case?

28

Closure Test for FDs

- Given attribute set Y and FD set F
  - Denote Y_F^+ or Y^+ the closure of Y relative to F
    Y_F^+ = set of all FDs given or implied by Y
- Computing the closure of Y
  - Start: $Y_F^+ = Y$, $F' = F$
  - While there exists an $f \in F'$ s.t. $LHS(f) \subseteq Y_F^+$:
    $Y_F^+ = Y_F^+ \cup RHS(f)$
    $F' = F' - f$
  - At end: $Y \rightarrow B$ for all $B \in Y_F^+$

Acknowledgements: M. Papagelis

Computing the closure Y^+ of a set of attributes Y
Given FDs F: