Relational Database Design and SQL Basics

CPS 216

Advanced Database Systems

Relational design: a review

- Identifying tuples: keys
- Generalizing the key concept: FDs
- Non-key FDs: redundancy
- Avoiding redundancy: BCNF decomposition
- Preserving FDs: 3NF

2

BNCF = no redundancy?

- *Student* (*SID*, *CID*, *club*)
  - Suppose your classes have nothing to do with the clubs you join
  - FDs?
  - BNCF?
  - Redundancies?

SID	*CID*	*club*
142	CPS 216	ballet
142	CPS 216	sumo
142	CPS 214	ballet
142	CPS 214	sumo
123	CPS 216	chess
123	CPS 216	golf
…	…	…

3

Multi-valued dependencies

- A multi-valued dependency (MVD) has the form $X \rightarrow\rightarrow Y$, where X and Y are sets of attributes in a relation R
- $X \rightarrow\rightarrow Y$ means that whenever two tuples in R agree on all the attributes of X, then we can swap their Y components and get two new tuples that are also in R

4

MVD examples

Student (*SID*, *CID*, *club*)

5

Complete MVD + FD rules

- FD reflexivity, augmentation, and transitivity
- MVD complementation:
  If $X \rightarrow\rightarrow Y$, then $X \rightarrow\rightarrow \text{attrs}(R) - X - Y$
- MVD augmentation:
  If $X \rightarrow\rightarrow Y$ and $V \subseteq W$, then $XW \rightarrow\rightarrow YV$
- MVD transitivity:
  If $X \rightarrow\rightarrow Y$ and $Y \rightarrow\rightarrow Z$, then $X \rightarrow\rightarrow Z - Y$
- Replication (FD is MVD):
  If $X \rightarrow Y$, then $X \rightarrow\rightarrow Y$
- Coalescence:
  If $X \rightarrow\rightarrow Y$ and $Z \subseteq Y$ and there is some W disjoint from Y such that $W \rightarrow Z$, then $X \rightarrow Z$

Try proving dependencies with these!?

6

An elegant solution: chase

- Given a set of FDs and MVDs D, does another dependency d (FD or MVD) follow from D?
- Procedure
  - Start with the hypotheses of d, and treat them as "seed" tuples in a relation
  - Apply the given dependencies in D repeatedly
    - If we apply an FD, we infer equality of two symbols
    - If we apply an MVD, we infer more tuples
  - If we infer the conclusion of d, we have a proof
  - Otherwise, if nothing more can be inferred, we have a counterexample

7

Proof by chase

- In $R(A, B, C, D)$, does $A \rightarrow\rightarrow B$ and $B \rightarrow\rightarrow C$ imply $A \rightarrow\rightarrow C$?

8

Counterexample by chase

- In $R(A, B, C, D)$, does $A \rightarrow\rightarrow BC$ and $CD \rightarrow B$ imply $A \rightarrow B$?

9

4NF

- A relation R is in Fourth Normal Form (4NF) if
  - For every non-trivial MVD $X \rightarrow\rightarrow Y$ in R, X is a super key
  - That is, all FDs and MVDs follow from "key $\rightarrow$ other attributes"
- 4NF is stronger than BCNF

10

4NF decomposition algorithm

- Find a 4NF violation
  - A non-trivial MVD $X \rightarrow\rightarrow Y$ in R where X is not a super key
- Decompose R into R_1 and R_2, where
  - R_1 has attributes $X \cup Y$
  - R_2 has attributes $X \cup Z$ (Z contains attributes not in X or Y)
- Repeat until all relations are in 4NF
- Almost identical to BCNF decomposition algorithm
- Any decomposition on a 4NF violation is lossless

11

4NF decomposition example

Student (*SID*, *CID*, *club*)

SID	*CID*	*club*
142	CPS 216	ballet
142	CPS 216	sumo
142	CPS 214	ballet
142	CPS 214	sumo
123	CPS 216	chess
123	CPS 216	golf
…	…	…

12

3NF, BCNF, and 4NF

	3NF	BCNF	4NF
Preserves FDs?			
Redudancy due to FDs?			
Redundancy due to MVDs?			

13

Recap

- Another source of redundancy: MVDs
- Reasoning about FDs and MVDs: chase
- Avoiding redundancy due to MVDs: 4NF

14

A complete design example

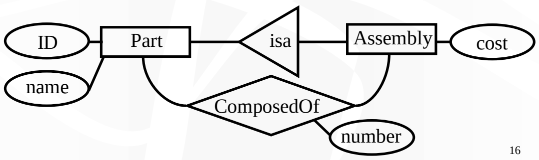
- Information about parts and assemblies for a manufacturing company; e.g.:
  - A bicycle consists of one frame and two wheels; the cost of assembly is $30
  - A frame is just a basic part
  - A wheel consists of one tire, one rim, and 48 spokes; the cost of assembly is $40
  - Everything has a part ID and a name

15

Entities and relationships

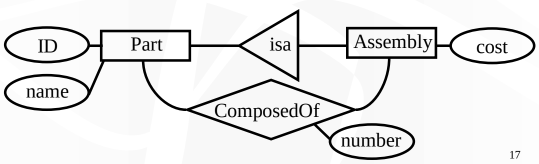
- Entities

- Relationships



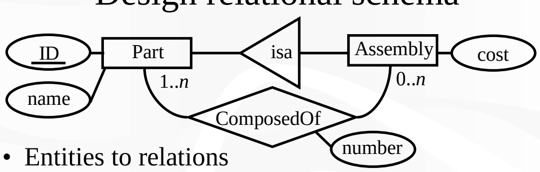
16

Identify constraints



17

Design relational schema

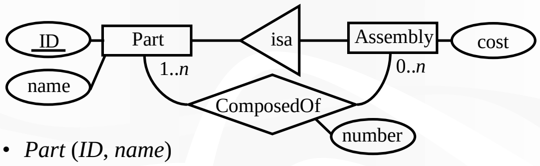


- Entities to relations

- Relationships to relations

18

Encode constraints



- *Part* (*ID*, *name*)
- *Assembly* (*ID*, *cost*)
- *ComposedOf* (*assemblyID*, *partID*, *number*)
- Any missing constraints?

19

Apply relational design theory

- *Part* (*ID*, *name*)
  - *ID* is a key
- *Assembly* (*ID*, *cost*)
  - *ID* is a key
- *ComposedOf* (*assemblyID*, *partID*, *number*)
  - {*assemblyID*, *partID*} is a key
- 3NF? BCNF? 4NF?

20

Populate schema with data

Part

ID	*name*
1	bicycle
2	frame
3	wheel
4	tire
5	rim
6	spoke
…	…

Assembly

ID	*cost*
1	30
3	40
…	…

ComposedOf

assemblyID	*partID*	*number*
1	2	1
1	3	2
3	4	1
3	5	1
3	6	48
…	…	…

21

Good design principles

- Avoid redundancy
- Avoid decomposing too much
- KISS
  – Focus on the task and avoid over-design

22

SQL

- SQL: Structured Query Language
  – Pronounced "S-Q-L" or "sequel"
  – The query language of every commercial DBMS
- A brief history
  – System R
  – SQL89
  – SQL92 (SQL2)
  – SQL3 (still under construction)

23

Table creation

- CREATE TABLE *table_name* (…, *column_name*$_i$ *column_type*$_i$, …);
- Example
  – create table Student (SID integer, name varchar(30), email varchar(30), age integer, GPA float);
  – create table Course (CID char(10), title varchar(100));
  – create table Enroll (SID integer, CID char(10));

SQL is case insensitive

24

Key declaration

- At most one PRIMARY KEY per table
  - Typically implies a primary index
  - Rows are stored inside the index, typically sorted by primary key value
- Any number of UNIQUE keys per table
  - Typically implies a secondary index
  - Pointers to rows are stored inside the index

25

Key declaration examples

- create table Student
  (SID integer primary key,
  name varchar(30),
  email varchar(30) unique,
  age integer, GPA float);
- create table Course
  (CID char(10) primary key,
  title varchar(100));
- create table Enroll
  (SID integer, CID char(10),
  primary key(SID, CID));

26

SFW queries

- SELECT A_1, A_2, …, A_n
  FROM R_1, R_2, …, R_m
  WHERE *condition*;
- Also called an SPJ (select-project-join) query
- Equivalent (more or less) to relational algebra query

27

Example: reading a table

- SELECT * FROM Student;
  - “*” is a shorthand for all columns
  - WHERE clause is optional

28

Example: selection and projection

- Names of students under 18
- When was Lisa born?

➢ SELECT list can contain calculations
➢ String literals are enclosed in single quotes (case sensitive)

29

Example: join

- SIDs and names of students taking courses with the word “Database” in their titles

➢ Okay to omit the table_name in table_name.column_name if column name is unique
➢ Many, many more built-in predicates such as LIKE

30

Example: rename

- SIDs of all pairs of classmates
  - AS is optional; in fact Oracle doesn't like it in the FROM clause

31

Set versus bag semantics

- Set
  - No duplicates
  - Relational model uses set semantics
- Bag
  - Duplicates allowed
  - Number of duplicates is significant
  - SQL uses bag semantics by default

32

Set versus bag example

SID	*CID*
142	CPS 216
142	CPS 214
123	CPS 216
857	CPS 216
857	CPS 130
456	CPS 214
…	…

π_{SID} (Enroll)   SELECT SID FROM Enroll;

33

A case for bag semantics

- Efficiency
- Which one is more useful?
- Besides, SQL provides the option of set semantics with DISTINCT

34

Example: forcing set semantics

- SIDs of all pairs of classmates
  - SELECT e1.SID as SID1, e2.SID as SID2
    FROM Enroll as e1, Enroll as e2
    WHERE e1.CID = e2.CID
    AND e1.SID > e2.SID;
    - Duplicates?
  - SELECT DISTINCT e1.SID as SID1, e2.SID as SID2
    FROM Enroll as e1, Enroll as e2
    WHERE e1.CID = e2.CID
    AND e1.SID > e2.SID;
    - No duplicates

35

Operational semantics of SFW

- SELECT [DISTINCT] E_1, E_2, …, E_n
  FROM R_1, R_2, …, R_m
  WHERE *condition*;
- For each t_1 in R_1:
  For each t_2 in R_2: … …
  For each t_m in R_m:
  If *condition* is true over t_1, t_2, …, t_m:
  Compute and output E_1, E_2, …, E_n
  If DISTINCT is present
  Eliminate duplicates in output

36

What’s next

More SQL

- Set/bag operations
- Joins
- Subqueries
- Aggregates
- NULL
- Modification statements