INFO-340: Database Management & Information Retrieval

David Hendry
INFO-340 C09

Topics

- Normalization
  - Update anomalies
- Functional dependencies
- Process of normalization
  - First normal form
  - Second normal form
  - Third normal form

Normalization

- A technique for producing a set of relations with desirable properties given the data requirements of an enterprise
- Basic idea:
  - You simplify (decompose) relations to avoid data redundancy

Motivation

- To avoid **update anomalies** relations need to be in **third normal form**

The Information School of the University of Washington

Third Normal Form

A relation is in 3rd normal form if, for all time, each tuple consists of a **primary key** value that identifies some entity, together with a set of zero or more **mutually independent attribute values** that describe the entity in some way

Common sense, sort of …

The Information School of the University of Washington

Update Anomalies

- Caused by **data redundancy** with relations
- Three kinds of anomalies
  - INSERT
  - DELETE
  - UPDATE
- Let's turn to an example

The Information School of the University of Washington

Example Relation

PARTS (S#, Status, City, P#, QTY)
PRIMARY KEY (S#, P#)

Constraint:
Status is **functional dependent** on city

The Information School of the University of Washington

Example Table

S#	Status	City	P#	Qty
S1	20	London	P1	300
S1	20	London	P2	200
S1	20	London	P3	400
S1	20	London	P4	200
S2	10	Paris	P1	300
S2	10	Paris	P2	400
S3	10	Paris	P2	200
S4	20	London	P2	200
S4	20	London	P4	300

Insert Anomalies

<u>S#</u>	Status	City	<u>P#</u>	Qty
S1	20	London	P1	300
S1	20	London	P2	200

•To insert supplier information we must insert part information

•Q: Another example?

The Information School of the University of Washington

Delete Anomalies

<u>S#</u>	Status	City	<u>P#</u>	Qty
S1	20	London	P1	300
S1	20	London	P2	200
S2	10	Paris	P1	300

•Q: What happens of we delete P1?

The Information School of the University of Washington

Update Anomalies

<u>S#</u>	Status	City	<u>P#</u>	Qty
S1	20	London	P1	300
S1	20	London	P2	200
S2	10	Paris	P1	300

•Q: What happens if we need to change the city of the S1?

The Information School of the University of Washington

Summary

This design just seems poor…

PARTS (S#, Status, City, P#, QTY)
PRIMARY KEY (S#, P#)

… but exactly why is it bad?

The Information School of the University of Washington

Functional Dependencies

The Information School of the University of Washington

Functional Dependency

- Describes the relationship between attributes in a relation. If A and B are attributes of R, B is functionally dependent on A (A → B), if each value A is associated with exactly on value of B

Example:

A cityID **functionally determines** a city name

Written:

cityID → cityName

The Information School of the University of Washington

Functional Dependency

cityId → cityName

The determinant

The dependent

cityID **functionally determines** cityName

or

cityName **is functionally dependent on** cityID

The Information School of the University of Washington

Exercise

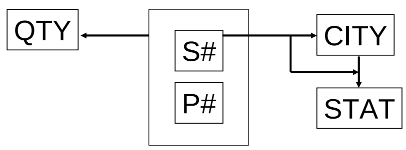
- Draw a diagram showing the functional dependencies in this relation

PARTS (S#, Stat, City, P#, QTY)

PRIMARY KEY (S#, P#)

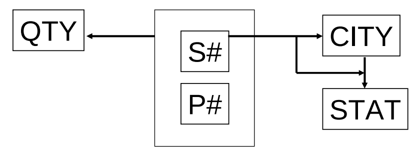
S#	Status	City	P#	Qty
S1	20	London	P1	300
S1	20	London	P2	200
S2	10	Paris	P1	300

As a Diagram



PARTS (S#, Stat, City, P#, QTY)
PRIMARY KEY (S#, P#)

As Sets



{S#,P#} → {QTY}
{S#} → {CITY}
{S#} → {STAT}
{S#} → {CITY,STAT}
{CITY} → {STAT}
{S#,CITY} → {STAT}

Trivial Dependencies

{S#,P#} → {S#} SP determines S
{S#,P#} → {P#} SP determines P

Example of 'reflexivity'

Key Properties of Functional Dependences

1. One-to-one relationship between determinant and dependent
   {S#,P#} → {QTY}
2. Relationship holds for all time (extension of relation; must consider domain)
3. Non-trivial
   Not this: {S#,P#} → {S#}

Ready for Normalization

1st Normal Form

All domains of a relation contain atomic values

Or

The intersection of each row and column contains one and only one value

2nd Normal Form

A relation in 1st normal form and every non-primary-key attribute is **fully functionally dependent** on the **primary key**

What is fully functionally dependent?

Fully Functional Dependent

If A and B are attributes of a relation, B is fully functionally dependent on A if B is functionally dependent on A, but not on any proper subset of A

Example

{S#,CITY} → {STAT}

(Not fully functional dependent)

Fully Functional Dependent

Example

{S#,CITY} → {STAT}

Each value of (s#,CITY) is associated with a signal value of (STAT)

S# → {STAT}

{CITY} → {STAT}

The Information School of the University of Washington

3rd Normal Form

- A relation that is in first and second normal form and in which no non-primary-key is **transitively dependent** on the primary key

What is transitive dependency?

The Information School of the University of Washington

Transitive Dependency

- If A, B, and C are attributes of a relation such that A → B and B→ C, then C is transitively dependent on A via B
- Example

  {S#} → {CITY} → {STAT}

The Information School of the University of Washington

Why does this matter?

The Information School of the University of Washington

The Information School of the University of Washington

Recall some Definitions

- Atomic Values
- Fully functionally dependent
- Transitive dependency

- 1st normal form
- 2nd normal form
- 3rd normal form

The Information School of the University of Washington

Exercise

- Normalize this relation

PARTS (S#, Stat, City, P#, QTY)
PRIMARY KEY (S#, P#)

The Information School of the University of Washington

Answer

P (P#, pName, color, city)
PRIMARY KEY (P#)

S (S#, Stat, City) * 2nd Normal Form
PRIMARY KEY (S#)

SP (P#,S#,QTY)
PRIMARY KEY (P#,S#)
FOREIGN KEY (P#) REFERENCES P
FOREIGN KEY (S#) REFERENCES S

The Information School of the University of Washington

Continued..

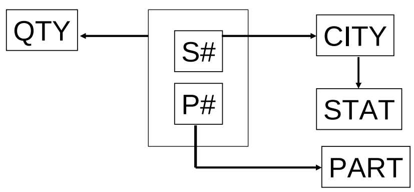
City (CityId, cityName, Stat)
PRIMARY KEY (City)

SC (S#, CityId)
PRIMARY KEY (S#)
FOREIGN KEY (CityId) REFERENCES City

P (P#, pName, color, city)
PRIMARY KEY (P#)

SP (P#,S#,QTY)
PRIMARY KEY (P#,S#)
FOREIGN KEY (P#) REFERENCES P
FOREIGN KEY (S#) REFERENCES SC

New Functional Dependency Diagram



Next Time

- Additional topics with Database Design

Summary

- Third normal form is very important because it minimizes update anomalies by removing redundancies

Third normal form
- A primary key
- All other attributes functional depend on key
- All other attributes are mutually independent

A database used in an order-entry system is to contain information about customers, items, and orders. The following information is to be included:

For each customer:
- Customer number (unique)
- Ship-to addresses (several per customer)
- Balance
- Credit limit
- Discount

For each order:
- Heading information:
  - Customer number
  - Ship-to address
  - Date of order
- Detail lines (several per order):
  - Item number
  - Quantity ordered

For each item:
- Item number (unique)
- Manufacturing plants
- Quantity on hand at each plant
- Stock danger level for each plant
- Item description (See handout)

The Information School of the University of Washington

Possible Answer

- Cust # → {Bal, Credlim, Discount}
- Address → Cust#
- Ord# → {Address, Date}
- {Ord#,Line#} → {Qtyord, Item#}
- Item# → Descn
- {Item#,Plant#} → {Qtyoh, Danger}

The Information School of the University of Washington

INFO-340: Database Management & Information Retrieval

David Hendry
INFO-340 C10

The Information School of the University of Washington

Database Methodology

The Information School of the University of Washington

Design Methodology

- A structured approach to designing a database that relies on
  - Procedures
  - Tools
  - Documentation formats
- Key idea: Follow a process to think rigorously and reduce the risk of errors

The Information School of the University of Washington

Three Phases

1. Conceptual database design
2. Logical database design
3. Physical database design

The Information School of the University of Washington

Conceptual Database Design

The process of constructing a model of the information used in an enterprise, independent of all physical considerations

Key idea: What entities, relationships, and attributes are needed? How are this things related?

The Information School of the University of Washington

Logical Database Design

- The process of constructing a model of the information used in the enterprise based on a specific data model, but independent of a particular DBMS and other physical considerations

- Key idea: 'Translate' conceptual model into abstractions of target database

The Information School of the University of Washington

Physical Database Design

- The process of producing a description of the implementation of the database on secondary storage;
  - Base relations & Views
  - File organizations
  - Indexes
  - Integrity constraints
  - Security measures
- Key idea: Performance & engineering issues

The Information School of the University of Washington

Conceptual Database Design

- Step 1: Build local conceptual data model for each user view
  1a. Identify entity types
  1b. Identify relationship types
  1c. Identify attributes
  1d. Determine attribute domains
  1e. Determine candidate and primary keys
  1f. Validate model against user transactions
  1g. Review model with users

The Information School of the University of Washington

Logical Database Design

- Step 2: Build and validate logical data model for each view
  2a. Remove features not compatible with the relational model
  2b. Derive relations for the logical data model
  2c. Validate relations with normalization
  2d. Define integrity constraints
  2e. Review local model with users

The Information School of the University of Washington

Logical Database Design

- Step 3: Build and validate global data model
  3a. Merge local logical data models into global model
  3b. Validate global logical data model
  3c. Check for future growth
  3d. Review global model with users

The Information School of the University of Washington

Physical Database Design

- Step 4: Translate global logical data model for target DBMS
  …
- Step 5: Design physical representation
  – Analyze transactions
  – Choose file organizations
  – Choose indexes
  – Estimate disk space requirements

The Information School of the University of Washington

Physical Database Design

- Step 6: Design user views
- Step 7: Design security mechanisms
- Step 8: Consider introduction of controlled redundancy
- Step 9: Monitor and tune the operational system

The Information School of the University of Washington

Key Ideas

- Work iteratively and with users;
- Follow a methodology to reduce the risk of making mistakes;
- Focus on the data and how it flows;
- Separate out:
  - Conceptual
  - Logical
  - Physical

The Information School of the University of Washington

Relational Databases

The Information School of the University of Washington

Topics

- Background
  - History, users, roles …
- DBMS Environment
  - Three-level architecture, data models, data independence, functions, services …
- The relational model
  - Relations, keys, DML, DDL, SQL, integrity constraints, views, relational algebra
- Relational design
  - ER & EER modeling, functional dependences, normalization

Background to databases

The Information School of the University of Washington

- What are they?
- Users and roles
- A little history…
- Functions of a databases
- Advantages and disadvantages

Database Environment

The Information School of the University of Washington

- Three-level database architecture
- Data independence
- DML & DDL
- Data models: What are they?
- Functions and services of a dbms
- Software components of a dbms
- Client-server architecture
- System catalog and Transaction Processing Monitors

The Relational Model

The Information School of the University of Washington

- Terminology
- How tables represent data
- Mathematics and relations
- Properties of relations
- Keys: super, candidate, primary, …
- Entity and referential integrity
- Purpose and use of views
- Relational algebra

SQL

The Information School of the University of Washington

- Data manipulation
  - Select
  - Set operations
  - Update operations
- Data definition
  - Integrity constraints
  - Create & Alter tables
  - Views
  - ISO transactions

ER & EER Modeling

- How to use them
- The concepts and notation
- Connection traps
  - Fan traps
  - Chasm traps

Normalization

- Purpose
- Update anomalies
- Functional dependencies
- The process of normalization
- 1^{st}, 2^{nd}, and 3^{rd} normal form

Upcoming

- Next week
  - No readings!
- Monday
  - A little more on design methodology
- Wednesday
  - Mid-term