Relational Database Design

CS 537- Big Data Analytics

Dr. Faisal Kamiran

Contents

Fundaments of relational data modeling by focusing on:

1. Normalization
2. Denormalization
3. Fact Tables
4. Dimension Tables
5. Schema Models
    1. Star Schema
    2. Snowflake Schemas

Database

- **Formal Definition:**
  - A set of related data and the way it is organized.

- Facilitates data access, management and updating

Database Management System

"…consisting of **computer software** that allows users to interact with the databases and provides access to all of the data. Because of the **close relationship** the term database is often used to refer to both the database and the DBMS used"

--Wikipedia

Importance of Relational Databases

- Invented in 1969 by researchers at IBM. Edgar R. Codd, the lead researcher proposed 12 rules of what makes a database management system a true relational system.

Rule 1: The *information* rule

All information in a relational database is represented explicitly at the logical level and in exactly one way – by values in tables

Rule 1 is what we are trying to achieve with relational modeling

Importance of Relational Databases

- Standardization of data model
  (Your data is in standard form)
- Flexibility in adding and altering tables
  (Can easily add and change tables)
- Data Integrity
  (Once a transaction is completed, change is persistent)
- Standard Query Language (SQL)
- Simplicity
- Intuitive Organization

What is OLAP vs OLTP?

Online Analytical Processing (OLAP)

Databases optimized for these workloads allow for **complex analytical and ad hoc queries.** These types of databases are optimized for reads.

Online Transactional Processing (OLTP)

Databases optimized for these workloads allow for **less complex queries in large volume.** The types of queries for these databases are read, insert, update and delete.

Example

- **OLAP queries**
  - “How many shoes were sold in Lahore in a specific month.”

- **OLTP queries**
  - “The price of the shoe.”

OLTP queries will perform very little aggregations while OLAP is designed to have heavy aggregations

Structuring Your Database

Structuring Your Database

- Normalization
    - To reduce data redundancy and increase data integrity.

- Denormalization
    - Combine multiple tables in order to facilitate faster queries
    - Must be done in read heavy workloads to increase performance

Normalization

The process of structuring a relational database in accordance with a series of **normal forms** in order to **reduce data redundancy and increase data integrity**

Data Redundancy: Goal is to remove duplicate data

Data Integrity: Make sure that you get the correct answer from the database
(update data at one place and that becomes the truth)

Normalization

Normalization

Does it follow the normalization rules?

Album ID	Album Name	Artist Name	Year	List of Songs
1	Burning Sun	Keating	1970	[Burning Sun, Feet, Moon is jealous]
2	Soul	Harvey	1960	[Hey Ma, Jenifer, Life is good]

Do you think the table is in Normal Form?

First Normal Form

- Atomic Values: Each cell contains **unique** and **single** values

There should not be any tuples or any list of values in a single cell

Album ID	Album Name	List of Songs
1	Burning Sun	[Burning Sun, Feet, Moon is jealous]
2	Soul	[Hey Ma, Jenifer, Life is good]

Album ID	Album Name	Song
1	Burning Sun	Burning Sun
1	Burning Sun	Feet
1	Burning Sun	Moon is jealous
2	Soul	Hey Ma
2	Soul	Jenifer
2	Soul	Life is good

First Normal Form

How to Reach 1st Normal Form

- Separate different relations into different tables

We do not want a single giant table

Customer and Sales table could have been merged. We could have a single table with all possible information

Customer table

Name	Email	ID	City
Amanda	jdoe@xyz.com	abc	NYC
Toby	n/a	def	NYC

Sales table

Name	Amount
Amanda	100.00
Toby	50.00

First Normal Form

How to Reach 1st Normal Form

- Keep relationships between tables together with **foreign keys**

There should be a way to link these tables together. The tables are linked together with foreign keys.

Customer table

Name	Email	ID	City
Amanda	jdoe@xyz.com	abc	NYC
Toby	n/a	def	NYC

Sales table

Name	Amount
Amanda	100.00
Toby	50.00

First Normal Form

How to Reach 1st Normal Form

- Atomic values: each cell contains unique and single values
- Keep relationships between tables together with **foreign keys**

Customer table

Name	Email	ID	City
Amanda	jdoe@xyz.com	abc	NYC
Toby	n/a	def	NYC

Sales table

Name	Amount
Amanda	100.00
Toby	50.00

Second Normal Form

How to Reach 2nd Normal Form

- Have reached 1NF
- All tables in the table must rely on the Primary Key

Customer details table

Store ID	Customer ID	Customer name	Email
001	004	Amanda	jdoe@xyz.com
001	005	Mary	mjane@yxs.com
003	006	Mike	mike@domain.com

PK

Store_Customer ID

Store ID	Customer ID
001	004
001	005
003	006

FK

Customer table

Customer ID	Customer name	Email
004	Amanda	jdoe@xyz.com
005	Mary	mjane@yxs.com
006	Mike	mike@domain.com

Third Normal Form

Awards table

Music Award	Year	Winner Record of Year	Lead Singer
Grammy	1965	The Beatles	John Lennon
CMA	2000	Faith Hill	Faith Hill
Grammy	1970	The Beatles	John Lennon
VMA	2001	U2	Bono

How to Reach 2nd Normal Form

- Have reached 2NF
- No transitive dependencies

Awards Table

Music Award	Year	Winner Record of Year
Grammy	1965	The Beatles
CMA	2000	Faith Hill
Grammy	1970	The Beatles
VMA	2001	U2

Lead Singer

Band Name	Lead Singer
The Beatles	John Lennon
Faith Hill	Faith Hill
U2	Bono

Lead singer is related to the name of the band. Changing the band will change the lead singer

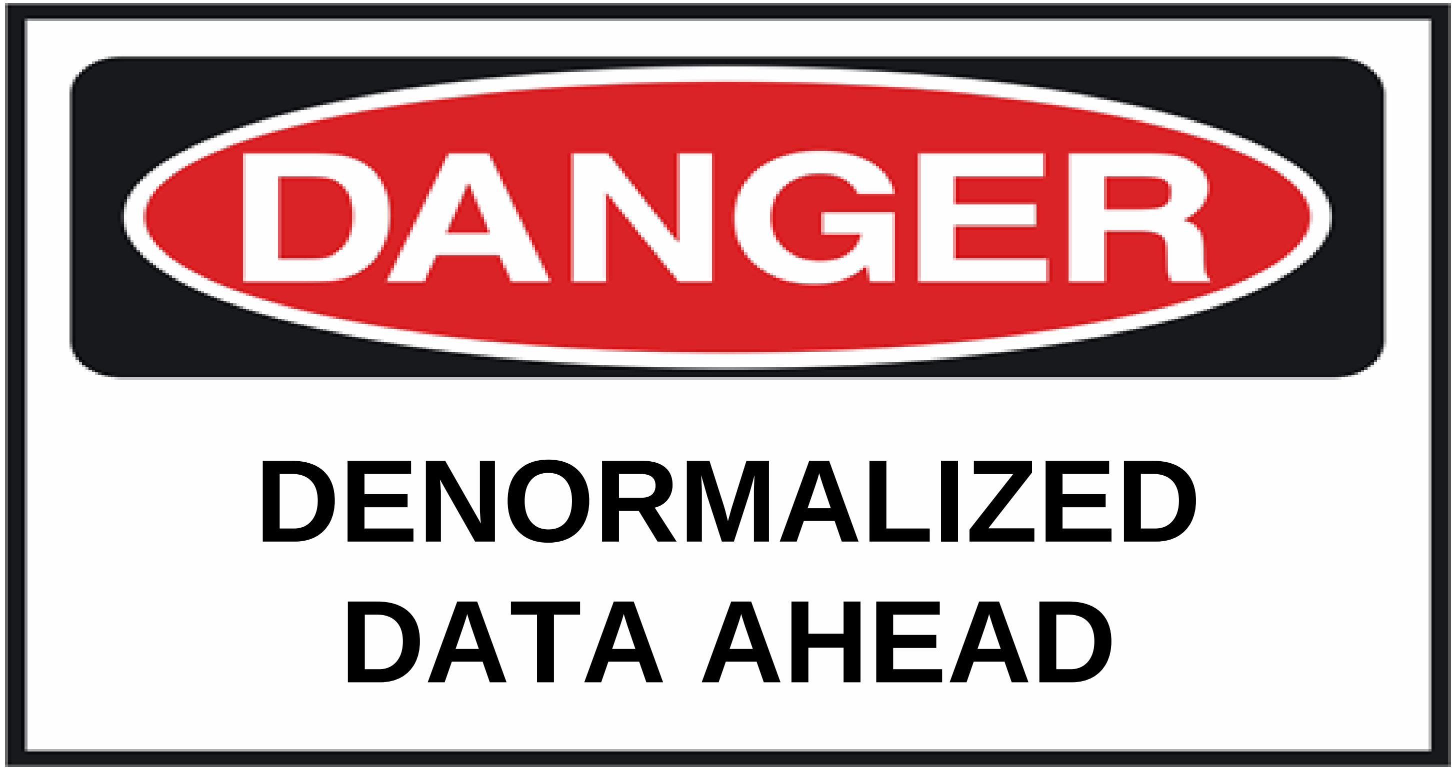
Denormalization

Consequences of Normalization

- Data redundancy is reduced or eliminated.
- Relations are broken into smaller, related tables.
- Using all the attributes from the original relation requires joining these smaller tables.

Denormalization

Deliberately reintroducing some redundancy, so that we can access data faster.



Denormalization

Objective: To improve the read performance of a database at the expense of losing some write performance by adding redundant copies of data.

- JOINS allow outstanding flexibility but are extremely slow.
- Denormalization is preferred for databases with heavy reads
- Denormalization is done **after** normalization
- Denormalization utilizes more space as multiple copies of the data are stored

When should denormalization be done?

We want a logical design change

- We want to model our data differently
- Reads will be faster (select)
- Writes will be slower (insert, update, delete)

Data Consistency

- There are multiple copies of data so each copy should be updated or deleted at the same time
- City and Name should be inserted or updated in both tables

Customer

Name	City	Amount
Amanda	NYC	100.00
Toby	NYC	30.00

Shipping

Name	City	Item
Amanda	NYC	Shirt
Toby	NYC	Pants

Denormalization

- Denormalization is all about performance.
- You do not need heavy joins to answer queries.
- We have separate tables with duplicate copies of data to increase performance.
- We first perform normalization and then denormalization.

How much a customer spent?

Customer

Name	City	Amount
Amanda	NYC	100.00
Toby	NYC	30.00

Shipping

Name	City	Item
Amanda	NYC	Shirt
Toby	NYC	Pants

The type of items we need to ship?

Normalization & Denormalization

Normalization	Denormalization
Redundancy and inconsistency is reduced	Redundancy is added for quick execution of queries
Number of tables increases	Number of tables decreases
Data integrity is maintained	Does not maintain data integrity
Optimizes memory usage	Does not optimize memory usage

DEMO