Lecture 9

# Database Systems

Instructor: M.Imran Khalil

Imrankhalil3@gmail.com

Resource:Imrankhalil3.wordpress.com



## Todays lecture

- Database Normalization

## Well-structured relations

- Well-structured relations contain minimal redundancy and allow insertion, modification, and deletion without errors or inconsistencies
- Anomalies are errors or inconsistencies resulting from redundancy
  - Insertion anomaly
  - Deletion anomaly
  - Modification anomaly

## Anomalies

- Goal of relational schema design is to avoid anomalies and redundancy.
  - *Update anomaly* : one occurrence of a fact is changed, but not all occurrences.
  - *Deletion anomaly* : valid fact is lost when a tuple is deleted.

4

# Insert Anomaly

Student

| sNumber | sName | pNumber | pName |
|---|---|---|---|
| s1 | ali | p1 | aslam |
| s2 | akram | p1 | akber |

Note: We cannot insert a professor who has no students.

Insert Anomaly: We are not able to insert "valid" value/(s)

# Deletion Anomaly

Student

| sNumber | sName | pNumber | pName |
|---|---|---|---|
| s1 | ali | p1 | aslam |
| s2 | akram | p1 | akber |

Note: We cannot delete a student that is the only student of a professor.

Delete Anomaly: We are not able to perform a delete without losing some
"valid" information.

# Update Anomaly

Student

| sNumber | sName | pNumber | pName |
|---|---|---|---|
| s1 | ali | p1 | aslam |
| s2 | akram | p1 | akber |

Note: To update the name of professor , we have to update the multiple tuples.

Update Anomaly: To update a value, we have to update multiple rows.

# Database Normalization

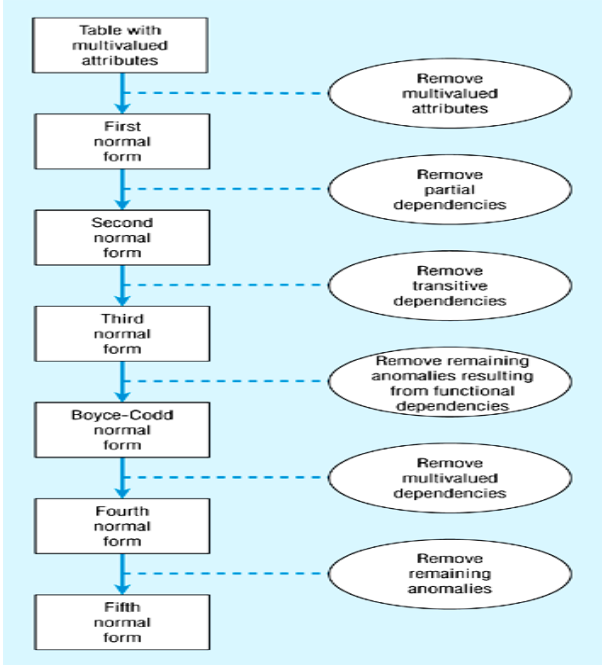
- **Database normalization** is the process of removing redundant data from your tables in to improve storage efficiency, data integrity, and scalability.
- In the relational model, methods exist for quantifying how efficient a database is. These classifications are called **normal forms** (or **NF**), and there are algorithms for converting a given database between them.
- Normalization generally involves splitting existing tables into multiple ones, which must be re-joined or linked each time a query is issued.

# Why Normalization?

- To remove potential redundancy in design
  - Redundancy causes several *anomalies: insert,* delete and update
- Normalization uses concept of dependencies
  - Functional Dependencies
- Idea used: *Decomposition*
  - Break R (A, B, C, D) into R1 (A, B) and R2 (B, C, D)
- Use decomposition judiciously.

# Data normalization

- Normalization is a formal process for deciding which attributes should be grouped together in a relation
  - Objective: to validate and improve a logical design so that it satisfies certain constraints that avoid unnecessary duplication of data
  - Definition: the process of decomposing relations with anomalies to produce smaller, well-structured relations




**Steps in normalization**

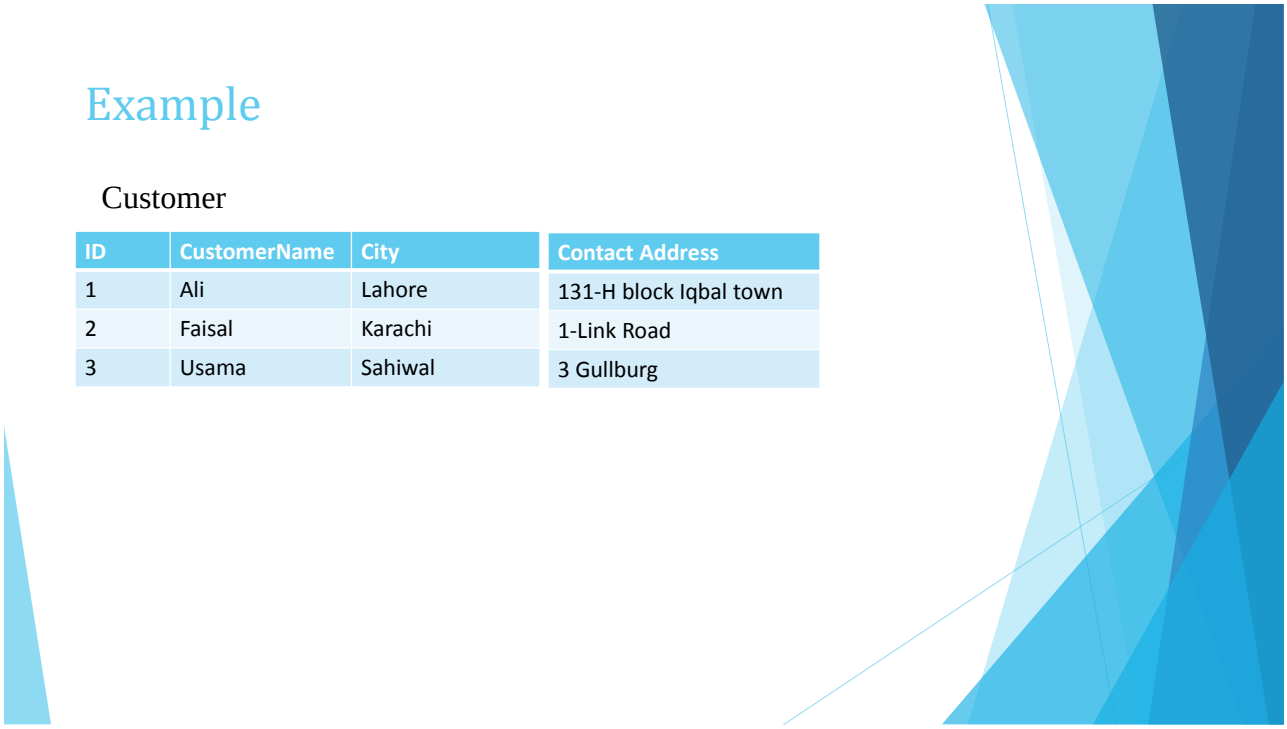
## First normal form

- No multi-valued attributes.
- Every attribute value is atomic.
- No repeating groups

# Example

Customer

| ID | CustomerName | City | Contact Address |
|---|---|---|---|
| 1 | Ali | Lahore | 131-H block Iqbal town |
| 2 | Faisal | Karachi | 1-Link Road |
| 3 | Usama | Sahiwal | 3 Gullburg |



# Example

Customer

| ID | CustomerName | City | Contact Address |
|---|---|---|---|
| 1 | Ali | Lahore | 131-H block Iqbal town |
| 2 | Faisal | Karachi | 1-Link Road |
| 3 | Usama | Sahiwal | 3 Gullburg, 3 streat , lohari Gate |

# Example

Customer

| ID | CustomerName | City | Contact Address | Contact Address 2 |
|---|---|---|---|---|
| 1 | Ali | Lahore | 131-H block Iqbal town | |
| 2 | Faisal | Karachi | 1-Link Road | |
| 3 | Usama | Sahiwal | 3 Gullburg | 1 streat Lohari Gate |

# Solution

Customer

| ID | CustomerName | City |
|---|---|---|
| 1 | Ali | Lahore |
| 2 | Faisal | Karachi |
| 3 | Usama | Sahiwal |

CustomerContact

| ID | CustID | ContactAddress |
|---|---|---|
| 1 | 1 | 131-H block Iqbal town |
| 2 | 2 | 1-Link Road |
| 3 | 3 | 2 A Gullburb III |
| 4 | 3 | Streat 2 lohari gate |
| 5 | 3 | 16 C izmir Town |

## Second normal form

- 1NF and every non-key attribute is fully functionally dependent on the primary key.
- Every non-key attribute must be defined by the entire key, not by only part of the key.
- No partial functional dependencies.

## Example

Events

| ID | Date | CourseTitle | Seats | Room |
|---|---|---|---|---|
| SQL101 | 04/02/2010 | Intro to SQL | 5 | 4 |
| SQL101 | 05/02/2015 | Intro to SQL | 6 | 7 |
| SQL101 | 06/06/2015 | Intro to SQL | 8 | 6 |
| DB101 | 05/05/2015 | Database | 6 | 5 |

# Example

Events

| ID | Date | Seats | Room |
|---|---|---|---|
| SQL101 | 04/02/2010 | 5 | 4 |
| SQL101 | 05/02/2015 | 6 | 7 |
| SQL101 | 06/06/2015 | 8 | 6 |
| DB101 | 05/05/2015 | 6 | 5 |

Courses

| ID | CourseTitle |
|---|---|
| SQL101 | Intro to SQL |
| DB101 | Database |

# Functional dependencies and keys

- Functional dependency: the value of one attribute (the *determinant*) determines the value of another attribute
  - A -> B, for every valid instance of A, that value of A uniquely determines the value of B
- Candidate key: an attribute or combination of attributes that uniquely identifies an instance
  - Uniqueness: each non-key field is functionally dependent on every candidate key
  - Non-redundancy

## Third normal form

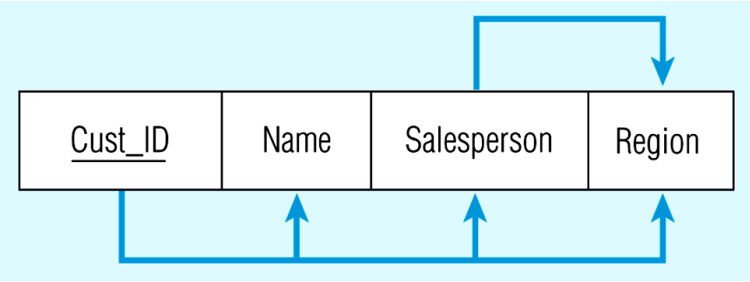
- 2NF and no transitive dependencies (functional dependency between non-key attributes.)

## Relation with transitive dependency

SALES

| Cust_ID | Name | Salesperson | Region |
|---|---|---|---|
| 8023 | Anderson | Smith | South |
| 9167 | Bancroft | Hicks | West |
| 7924 | Hobbs | Smith | South |
| 6837 | Tucker | Hernandez | East |
| 8596 | Eckersley | Hicks | West |
| 7018 | Arnold | Faulb | North |

## Transitive dependency in SALES relation




## Removing a transitive dependency

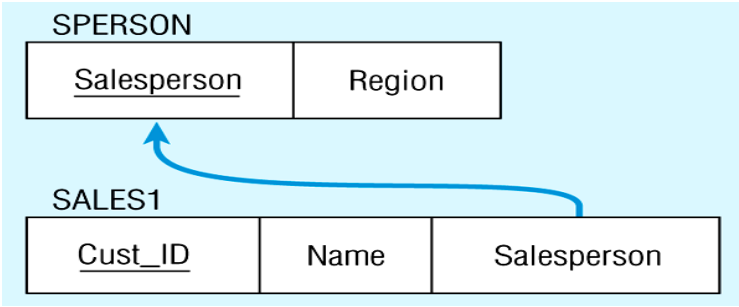
SALES1

| Cust_ID | Name | Salesperson |
|---|---|---|
| 8023 | Anderson | Smith |
| 9167 | Bancroft | Hicks |
| 7924 | Hobbs | Smith |
| 6837 | Tucker | Hernandez |
| 8596 | Eckersley | Hicks |
| 7018 | Arnold | Faulb |

SPERSON

| Salesperson | Region |
|---|---|
| Smith | South |
| Hicks | West |
| Hernandez | East |
| Faulb | North |

## Relations in 3NF




## Let's practice...

# Other considerations...

- Synonyms: different names, same meaning.
- Homonyms: same name, different meanings.