



Chapter 4

Logical Database Design Normalization



Cont...

- Database normalization is a series of steps followed to obtain a database design that allows for consistent storage and efficient access of data in a relational database.
- These steps reduce **data redundancy** and the risk of data becoming inconsistent.

Cont...

- NORMALIZATION is the process of identifying the **logical associations** between data items and designing a database that will represent such associations but without suffering the **update anomalies** which are;
 1. Insertion Anomalies
 2. Deletion Anomalies
 3. Modification Anomalies



Cont...

- Normalization may reduce **system performance** since data will be cross referenced from many tables.
- Thus denormalization is sometimes used to improve performance, at the cost of reduced consistency guarantees.
- Normalization normally is considered as good if it is **lossless decomposition**.

Example

Stud-ID	CCode	Grade	Name	Address
100	Comp100	A	Abebe	Peda
100	Comp101	B	Abebe	Peda
100	Comp102	A	Abebe	Peda
101	Comp100	A	Tigist	Poly

- Clearly, Name and Address are redundant (larger relation + you have to update 3 rows to update the Address)
- Cannot make a record one Student unless he is not taking any classes

Cont...

Mnemonic for remembering the rationale for normalization could be the following:

1. **No Repeating or Redundancy:** no repeating fields in the table
2. **The Fields Depend Upon the Key:** the table should solely depend on the key
3. **The Whole Key:** no partial key dependency
4. **And Nothing But The Key:** no inter data dependency
5. **So Help Me Codd:** the rules of Codd

Cont...

- All the normalization rules will eventually remove the update anomalies that may exist during data manipulation after the implementation. The update anomalies are;
- **Pitfalls of Normalization**
 - Requires data to see the problems
 - May reduce performance of the system
 - Is time consuming,
 - Difficult to design and apply and
 - Prone to human error

Cont...

- The underlying ideas in normalization are simple enough. Through normalization we want to design for our relational database a set of tables that;
 1. Contain all the data necessary for the purposes that the database is to serve,
 2. Have as little redundancy as possible,
 3. Accommodate multiple values for types of data that require them,
 4. Permit efficient updates of the data in the database, and
 5. Avoid the danger of losing data unknowingly.
 - The type of problems that could occur in insufficiently normalized table is called update anomalies which includes;



Insertion anomalies

- An "insertion anomaly" is a failure to place information about a new database entry into all the places in the database where information about that new entry needs to be stored.
- In a properly normalized database, information about a new entry needs to be inserted into only one place in the database; in an inadequately normalized database, information about a new entry may need to be inserted into more than one place and, human fallibility being what it is, some of the needed additional insertions may be missed.

Deletion anomalies

- A "deletion anomaly" is a failure to remove information about an existing database entry when it is time to remove that entry.
- In a properly normalized database, information about an old, to-be-gotten-rid-of entry needs to be deleted from only one place in the database; in an inadequately normalized database, information about that old entry may need to be deleted from more than one place, and, human fallibility being what it is, some of the needed additional deletions may be missed.



Modification anomalies

- A modification of a database involves changing some value of the attribute of a table.
- In a properly normalized database table, what ever information is modified by the user, the change will be effected and used accordingly.
- The purpose of normalization is to reduce the chances for anomalies to occur in a database.

Example

EMpl D	Fname	Lname	Skil IID	Skill	Skill type	Scho ol	school Add	Skill Level
12	Abebe	Mekuria	2	SQL	Database	BiT	Poly	5
16	Lemma	Alemu	5	C++	Programm ing	High land	Kebele 14	6
28	Chane	Kebede	2	SQL	Database	Bit	poly	10
25	Abera	Taye	6	VB6	Programm ing	Blue Nile	Kebele 13	8
65	Almaz	Belay	2	SQL	Database	Blue Nile	Kebele 13	9
24	Dereje	Tamiru	8	Oracle	Database	High land	Kebele 14	5
51	Selam	Belay	4	Prolog	Programm ing	AAU	Addis Ababa	8
94	Alem	Kebede	3	Cisco	Networkin g	BiT	Poly	7
18	Girma	Dereje	1	IP	Programm ing	AAU	Addis Ababa	4



Deletion Anomalies:

- If employee with ID 16 is deleted then ever information about skill C++ and the type of skill is deleted from the database.
- Then we will not have any information about C++ and its skill type.



Insertion Anomalies:

- What if we have a new employee with a skill called Pascal? We can not decide whether Pascal is allowed as a value for skill and we have no clue about the type of skill that Pascal should be categorized as

Modification Anomalies:

- What if the address for High land is changed from kebele 14 to Kebele 7? We need to look for every occurrence of High land and change the value of School_Add from Kebele 14 to Kebele 7, which is prone to error.

Functional Dependency (FD)

Data Dependency

- The logical association between data items that point the database designer in the direction of a good database design are referred to as determinant or dependent relationships.
- Two data items **A** and **B** are said to be in a determinant or dependent relationship if certain values of data item **B** always appears with certain values of data item **A**.
- if the data item **A** is the determinant data item and **B** the dependent data item then the direction of the association is from **A** to **B** and not vice versa.

Cont...

- "B is functionally dependent on A." or "A determines B," or that "B is a function of A," or that "A functionally governs B."
- Often, the notions of functionality and functional dependency are expressed briefly by the statement, "If A, then B."
- It is important to note that the value B must be unique for a given value of A, i.e., any given value of A must imply just one and only one value of B, in order for the relationship to qualify for the name "function."

Cont...

- $X \rightarrow Y$ holds if whenever two tuples have the same value for X , they must have the same value for Y
- The notation is: **$A \rightarrow B$** which is read as; B is functionally dependent on A
- In general, a functional dependency is a relationship among attributes. In relational databases, we can have a determinant that governs one other attribute or several other attributes.

Cont...

- FDs are derived from the real-world constraints on the attributes
- Example

<i>Dinner Course</i>	<i>Type of Wine</i>
Meat	Red
Fish	White
Cheese	Rose

- Since the type of Wine served depends on the type of Dinner, we say Wine is functionally dependent on Dinner.
Dinner $\rightarrow$ Wine

Cont...

<i>Dinner Course</i>	<i>Type of Wine</i>	<i>Type of Fork</i>
Meat	Red	Meat fork
Fish	White	Fish fork
Cheese	Rose	Cheese fork

- Since both Wine type and Fork type are determined by the Dinner type, we say Wine is functionally dependent on Dinner and Fork is functionally dependent on Dinner.

Dinner $\rightarrow$ Wine

Dinner $\rightarrow$ Fork

Partial Dependency

- If an attribute which is not a member of the primary key is dependent on some part of the primary key (if we have composite primary key) then that attribute is partially functionally dependent on the primary key.
- Let $\{A,B\}$ is the Primary Key and C is no key attribute.
- Then if $\{A,B\} \rightarrow C$ and $B \rightarrow C$
- Then C is partially functionally dependent on $\{A,B\}$

Full Dependency

- If an attribute which is not a member of the primary key is not dependent on some part of the primary key but the whole key (if we have composite primary key) then that attribute is fully functionally dependent on the primary key.
- Let $\{A,B\}$ is the Primary Key and C is no key attribute
- Then if $\{A,B\} \rightarrow C$ and $B \rightarrow C$ and $A \rightarrow C$ does not hold
- Then C Fully functionally dependent on $\{A,B\}$

Transitive Dependency

- In mathematics and logic, a transitive relationship is a relationship of the following form:
- "If A implies B, and if also B implies C, then A implies C."

Example:

- If Abebe is a Human, and if every Human is an Animal, then Abebe must be an Animal.
- Generalized way of describing transitive dependency is that:
- If A functionally governs B, AND
 - If B functionally governs C
 - THEN **A** functionally governs **C**
- Provided that neither C nor B determines A ($B \not\rightarrow A$ and $C \not\rightarrow A$)
- In the normal notation:
- $\{(A \rightarrow B) \text{ AND } (B \rightarrow C)\} \implies A \rightarrow C$



Steps of Normalization:

Normalization towards a logical design consists of the following steps:

UnNormalized Form:

- Identify all data elements

First Normal Form:

- Find the key with which you can find all data

Second Normal Form:

- Remove part-key dependencies. Make all data dependent on the whole key.

Third Normal Form

- Remove non-key dependencies. Make all data dependent on nothing but the key.
- For most practical purposes, databases are considered normalized if they adhere to third normal form.

First Normal Form (1NF)

- Requires that all column values in a table are atomic (e.g., a number is an atomic value, while a list or a set is not).
- We have two ways of achieving this:
 1. Putting each repeating group into a separate table and connecting them with a primary key-foreign key relationship
 2. Moving this repeating groups to a new row by repeating the common attributes. If so then Find the key with which you can find all data

Cont...

Definition of a table (relation) in 1NF

- If
 - There are no duplicated rows in the table.
Unique identifier
 - Each cell is single-valued (i.e., there are no repeating groups).
 - Entries in a column (attribute, field) are of the same kind

Example I

- It is not in 1NF

Supplier	Part
S1	P1, P3, P4
S2	P3
S3	P2, P3

Cont...

- It is in 1NF

Supplier	Part
S1	P1
S1	P3
S1	P4
S2	P3
S3	P2
S3	P3

Example for first normal form(1NF)

UNNORMALIZED

EmpID	Fname	Lname	Skill	SkillType	School	SchoolAdd	SillLevel
12	Abebe	Mekuria	SQL, VB6	Database, Programming	BiT, Blue Nile	Poly, Kebele 13	5 8
16	Lemma	Alemu	C++, IP	Programming Programming	High land, AAU	Kebele 14, Addis Ababa	6 8
28	Chane	Kebede	SQL	Database	Bit	poly	10
25	Abera	Taye	VB6	Programming	Blue Nile	Kebele 13	8
65	Almaz	Belay	SQL Prolog Java	Database Programming Programming	Blue Nile AAU BiT	Kebele 13 Addis Ababa Poly	9 8 6
24	Dereje	Tamiru	Oracle	Database	High land	Kebele 14	5
94	Alem	Kebede	Cisco	Networking	BiT	Poly	7



FIRST NORMAL FORM (1NF)

- Remove all repeating groups.
- Distribute the multi-valued attributes into different rows and identify a unique identifier for the relation so that it can be said is a relation in relational database.

Cont...

EmpID	Fname	Lname	Skill	SkillType	School	SchoolAdd	SillLevel
12	Abebe	Mekuria	SQL	Database	BiT	Poly	5
2	Abebe	Mekuria	VB6	Programming	Blue Nile	Kebele 13	8
16	Lemma	Alemu	C++	Programming	High land	Kebele 14	6
16	Lemma	Lemma	IP	Programming	AAU	Addis Ababa	8
28	Chane	Kebede	SQL	Database	Bit	poly	10
25	Abera	Taye	VB6	Programming	Blue Nile	Kebele 13	8
65	Almaz	Belay	SQL	Database	Blue Nile	Kebele 13	9
65	Almaz	Belay	Prolog	Programming	AAU	Addis Ababa	8
65	Almaz	Belay	Java	Programming	BiT	Poly	6
24	Dereje	Tamiru	Oracle	Database	High land	Kebele 14	5
94	Alem	Kebede	Cisco	Networking	BiT	Poly	7

Second Normal form 2NF

- No partial dependency of a non key attribute on part of the primary key.
- Any table that is in 1NF and has a single-attribute (i.e., a non-composite) key is automatically also in 2NF.
- Definition of a table (relation) in 2NF
 - It is in 1NF and
 - If all non-key attributes are dependent on all of the key. i.e. no partial dependency.

Cont...

- Since a partial dependency occurs when a non-key attribute is dependent on only a part of the (composite) key, the definition of 2NF is sometimes phrased as, "A table is in 2NF if it is in 1NF and if it has no partial dependencies."
- Example for 2NF:

Cont...

- This schema is in its 1NF since we don't have any repeating groups or attributes with multi-valued property.
- To convert it to a 2NF we need to remove all partial dependencies of non key attributes on part of the primary key.

EMP_PROJ

<u>EmpID</u>	EmpName	<u>ProjNo</u>	ProjName	ProjLoc	ProjFund	ProjMangID
--------------	---------	---------------	----------	---------	----------	------------

EMP_PROJ rearranged

<u>EmpID</u>	<u>ProjNo</u>	EmpName	ProjName	ProjLoc	ProjFund	ProjMangID
--------------	---------------	---------	----------	---------	----------	------------

Cont...

- $\{\text{EmpID}, \text{ProjNo}\} \rightarrow \text{EmpName}, \text{ProjName}, \text{ProjLoc}, \text{ProjFund}, \text{ProjMangID}$
- But in addition to this we have the following dependencies
- $\text{EmpID} \rightarrow \text{EmpName}$
- $\text{ProjNo} \rightarrow \text{ProjName}, \text{ProjLoc}, \text{ProjFund}, \text{ProjMangID}$
- As we can see some non key attributes are partially dependent on some part of the primary key.
- Thus these collections of attributes should be moved to a new relation

Cont...

EMPLOYEE

<u><i>EmpID</i></u>	<i>EmpName</i>
---------------------	----------------

PROJECT

<u><i>ProjNo</i></u>	<i>ProjName</i>	<i>ProjLoc</i>	<i>ProjFund</i>	<i>ProjMangID</i>
----------------------	-----------------	----------------	-----------------	-------------------

EMP PROJ

<u><i>EmpID</i></u>	<u><i>ProjNo</i></u>
---------------------	----------------------

Third Normal Form (3NF)

- Eliminate Columns Not Dependent On Key
 - If attributes do not contribute to a description of the key, remove them to a separate table.
 - This level avoids update and delete anomalies.
- Definition of a Table (Relation) in 3NF
 - It is in 2NF and
 - There are no transitive dependencies between attributes.

- It is not in 3NF

Student	Course_ID	Grade	Grade_value
Erik	CIS331	A	4.00
Sven	CIS331	B	3.00
Inge	CIS331	C	2.00
Hildur	CIS362	A	4.00

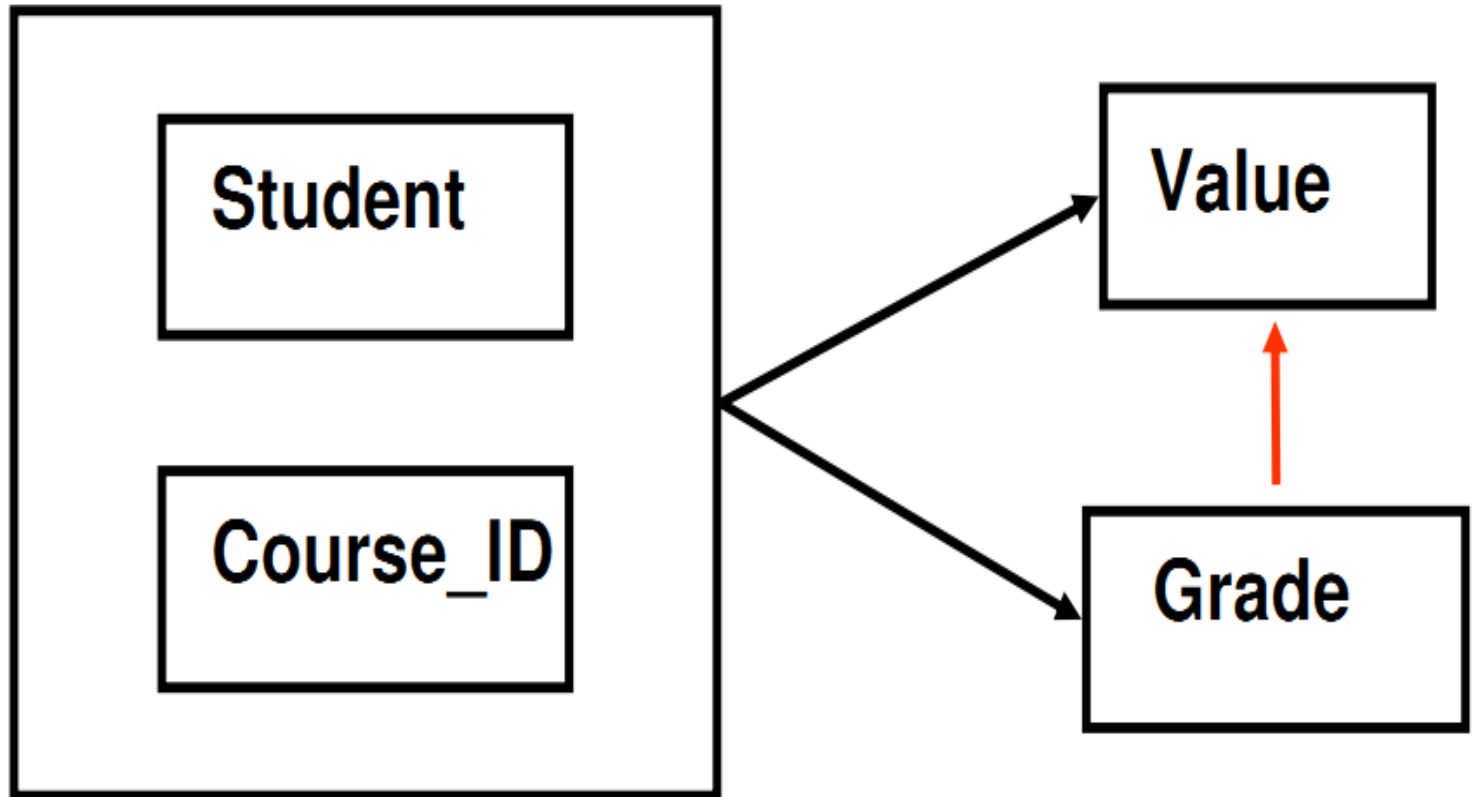
Cont...

- Normalized to 3NF

Student	Course_ID	Grade
Erik	CIS331	A
Sven	CIS331	B
Inge	CIS331	C
Hildur	CIS362	A

Grade	Grade_value
A	4.00
B	3.00
C	2.00

Cont...



Example 2 for (3NF)

- Assumption: Students of same batch (same year) live in one building or dormitory

STUDENT

<u><i>StudID</i></u>	<i>Stud_F_Name</i>	<i>Stud_L_Name</i>	<i>Dept</i>	<i>Year</i>	<i>Dormitory</i>
125/97	Abebe	Mekuria	<i>Info Sc</i>	<i>1</i>	<i>401</i>
654/95	Lemma	Alemu	<i>Geog</i>	<i>3</i>	<i>403</i>
842/95	Chane	Kebede	<i>CompSc</i>	<i>3</i>	<i>403</i>
165/97	Alem	Kebede	<i>InfoSc</i>	<i>1</i>	<i>401</i>
985/95	Almaz	Belay	<i>Geog</i>	<i>3</i>	<i>403</i>

- This schema is in its 2NF since the primary key is a single attribute.

Cont...

- Let's take StudID, Year and Dormitory and see the dependencies.
- **StudID** → Year AND **Year** → Dormitory
Then transitively **StudID** → **Dormitory**
- To convert it to a 2NF we need to remove all partial dependencies of non key attributes on part of the primary key.

Cont...

STUDENT

<u>StudID</u>	<i>Stud F_Name</i>	<i>Stud L_Name</i>	<i>Dept</i>	<i>Year</i>
125/97	Abebe	Mekuria	<i>Info Sc</i>	<i>1</i>
654/95	Lemma	Alemu	<i>Geog</i>	<i>3</i>
842/95	Chane	Kebede	<i>CompSc</i>	<i>3</i>
165/97	Alem	Kebede	<i>InfoSc</i>	<i>1</i>
985/95	Almaz	Belay	<i>Geog</i>	<i>3</i>

DORM

<u>Year</u>	<i>Dormitory</i>
<i>1</i>	<i>401</i>
<i>3</i>	<i>403</i>

Boyce-Codd Normal Form (BCNF):

- Isolate Independent Multiple Relationships - No table may contain two or more 1:n or N:M relationships that are not directly related.
- The correct solution, to cause the model to be in 4th normal form, is to ensure that all **M:M relationships** are resolved independently if they are indeed independent, as shown below.
- Def: A table is in BCNF if it is in 3NF and if every determinant is a candidate key.

Summary

- Boyce-Codd Normal Form (BCNF):
 - “Everything should depend on the key, the whole key, and nothing but the key”
- 1NF (all attributes are atomic)
- 2NF (no partial dependencies)
- 3NF (no transitive dependencies)