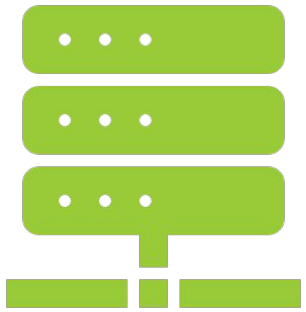


Database Normalization

In this Lecture you will Learn about:

Database Redundancies and Anomalies

Normalization

Normalization Types

Normalization Process

First Normal Form (1NF)

Second Normal Form (2NF)

Third Normal Form (3NF)

BCNF

Database Redundancies and Anomalies

- Data redundancy in database management systems (DBMS) refers to the unnecessary duplication of data within a database.
- It occurs when the same piece of data is stored in multiple places or multiple times within a database.
- Real-life Analogy: It's like writing your address on every page of your notebook — it's repetitive and wasteful.
- Data redundancy can be in various forms, such as:
  1. **Duplicate Records:** Two or more records in a database contain identical or very similar information. For example, if there are multiple entries for the same customer with slight variations in spelling or formatting.
  2. **Repeated Attributes:** The same attribute or set of attributes is stored in multiple tables. This can happen when the same piece of information is needed in different contexts but is stored separately each time. For instance, storing the address of a customer in both an "Orders" table and a "Customers" table.
  3. **Repetitive Values:** Certain values are repeated unnecessarily within a single table. For example, if a product table includes the same description for multiple products rather than referencing a centralized list of descriptions.

Database Redundancies and Anomalies

Example - Problem Of Data Redundancy In Single Tabale Database

Employee Number	First Name	Last Name	Date of Birth	Department Code	Department Name	Department Head
1001	Steave	Jakson	25-09-1985	SA001	Sales	Paul Colgan
1002	Kitty	Mathew	06-04-1998	ACC008	Accounts	Jerry Mathew
1003	Meena	Patel	11-05-1992	SA001	Sales	Paul Colgan
1004	Nancy	Samual	02-12-1996	ACC008	Accounts	Jerry Mathew
1005	Michael	Smith	28-03-1995	SA001	Sales	Paul Colgan
1006	James	Garcia	22-01-1994	SA002	Sales	David Smith
1007	Nancy	Samual	11-02-1996	ACC008	Accounts	Charles Williams

Redundant Data In Table

SA001	Sales	Paul Colgan
ACC008	Accounts	Jerry Mathew

Database Redundancies and Anomalies

Implications of Data Redundancy:

1. **Increased Storage Requirements:** Redundant data consumes additional storage space within the database, leading to inefficiency and increased storage costs.
2. **Data Inconsistency:** Redundant data introduces the risk of inconsistency, as updates made to one copy of the data may not be reflected in all instances. This can result in inaccuracies within the database.
3. **Data Anomalies**

Anomalies in the relational model refer to inconsistencies or errors that can arise when working with relational databases, specifically in the context of data insertion, deletion, and modification. When redundancy is present, these three types of update anomalies can occur:

1. **Insertion Anomalies:** These anomalies occur when it is not possible to insert data into a database because the required fields are missing or because the data is incomplete.

   For example, if a database requires that every record has a primary key, but no value is provided for a particular record, it cannot be inserted into the database.

Database Redundancies and Anomalies

2. **Deletion Anomalies:** These anomalies occur when deleting a record from a database and can result in the unintentional loss of data.

   For example, if a database contains information about customers and orders, deleting a customer record may also delete all the orders associated with that customer.

2. **Update Anomalies:** These anomalies occur when modifying data in a database and can result in inconsistencies or errors.

   For example, if a database contains information about employees and their salaries, updating an employee's salary in one record but not in all related records could lead to incorrect calculations and reporting.

These anomalies can be removed with the process of Normalization, which generally splits the database which results in reducing the anomalies in the database.

Addressing data redundancy through normalization and proper database design can help mitigate these anomalies and ensure data integrity and consistency within the database.

Database Redundancies and Anomalies

STUDENT Table

STUD_NO	STUD_NAME	STUD_PHONE	STUD_STATE	STUD-COUNTRY	STUD_AGE
1	RAM	9716271721	Haryana	India	20
2	RAM	9898291281	Punjab	India	19
3	SUJIT	7898291981	Rajasthan	India	18
4	SURESH		Punjab	India	21

Table 1

STUDENT_COURSE

STUD_NO	COURSE_NO	COURSE_NAME
1	C1	DBMS
2	C2	Computer Networks
1	C2	Computer Networks

Table 2

- **Insertion Anomaly:** If a tuple is inserted in referencing relation and referencing attribute value is not present in referenced attribute, it will not allow insertion in referencing relation.

OR

- An insertion anomaly occurs when adding a new row to a table leads to inconsistencies.
- Example: If we try to insert a record into the STUDENT_COURSE table with STUD_NO = 7, it will not be allowed because there is no corresponding STUD_NO = 7 in the STUDENT table.

Database Redundancies and Anomalies

STUDENT Table

STUD_NO	STUD_NAME	STUD_PHONE	STUD_STATE	STUD-COUNTRY	STUD_AGE
1	RAM	9716271721	Haryana	India	20
2	RAM	9898291281	Punjab	India	19
3	SUJIT	7898291981	Rajasthan	India	18
4	SURESH		Punjab	India	21

Table 1

STUDENT_COURSE

STUD_NO	COURSE_NO	COURSE_NAME
1	C1	DBMS
2	C2	Computer Networks
1	C2	Computer Networks

Table 2

- **Deletion and Updation Anomaly:** If a tuple is deleted or updated from referenced relation and the referenced attribute value is used by referencing attribute in referencing relation, it will not allow deleting the tuple from referenced relation.
- Example: If we want to update a record from STUDENT_COURSE with STUD_NO =1, We have to update it in both rows of the table. If we try to delete a record from the STUDENT table with STUD_NO = 1, it will not be allowed because there are corresponding records in the STUDENT_COURSE table referencing STUD_NO = 1.
- Deleting the record would violate the foreign key constraint, which ensures data consistency between the two tables.

Database Redundancies and Anomalies

- Example:

Anomalies, Redundancy*

EmpDept

EID	Name	DeptID	DeptName
A01	Ali	12	Wing
A12	Eric	10	Tail
A13	Eric	12	Wing
A03	Tyler	12	Wing

- What anomalies are associated with EmpDept?
- Update Anomalies:

  If the Wing department changes its name, we must change multiple rows in EmpDept

- Insertion Anomalies:

  If a department has no employees, where do we store its name?

- Deletion Anomalies:

  If A12 Eric quits, the information about the Tail department will be lost.

Database Redundancies and Anomalies

- Example: Find redundancy and anomalies that can occur.

rollno	name	branch	hod	office_tel
401	Ali	CSE	Mr. X	53337
402	Aisha	CSE	Mr. X	53337
403	Sana	CSE	Mr. X	53337
404	Salman	EE	Mr. Y	53336

Normalization

- Normalization is the process of organizing the data in the database.
- Normalization is used to minimize the redundancy from a relation or set of relations. It is also used to eliminate undesirable characteristics like Insertion, Update, and Deletion Anomalies.
- Normalization divides the larger table into smaller and links them using relationships.

Normalization

large database defined as a single relation may result in data duplication. This repetition of data may result in:

- Making relations very large.
- It isn't easy to maintain and update data as it would involve searching many records in relation.
- Wastage and poor utilization of disk space and resources.
- The likelihood of errors and inconsistencies increases.

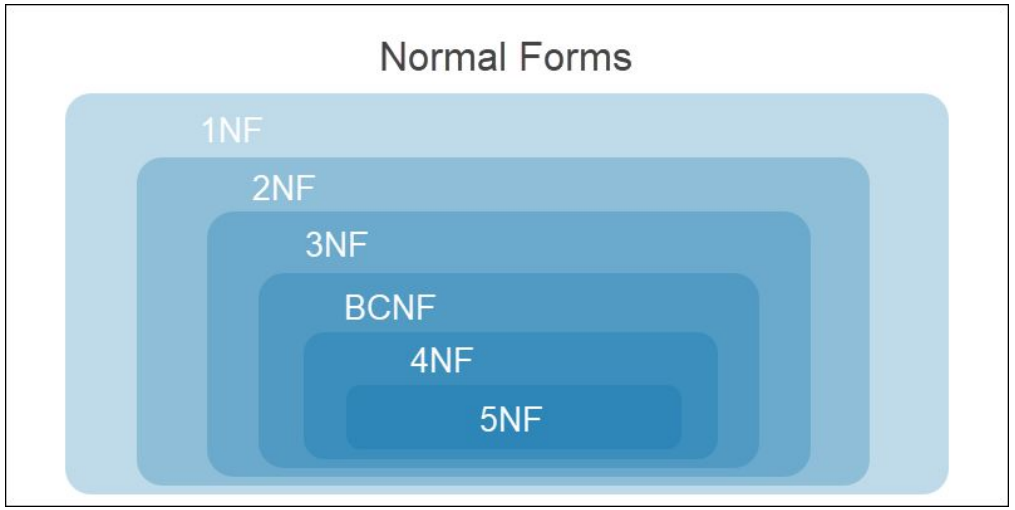
So to handle these problems, we should analyze and decompose the relations with redundant data into smaller, simpler, and well-structured relations that are satisfy desirable properties. Normalization is a process of decomposing the relations into relations with fewer attributes.

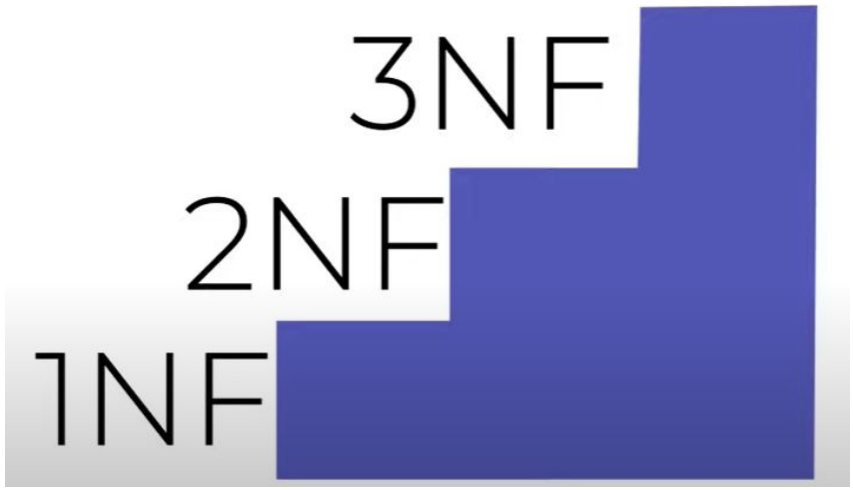
Normalization

Normalization in DBMS provides these advantages:

1. **Data Redundancy Elimination**: Reduces redundancy by organizing data efficiently.
2. **Data Consistency**: Minimizes anomalies, ensuring data remains consistent.
3. **Data Integrity Improvement**: Enforces referential integrity, maintaining accurate data.
4. **Simplified Maintenance**: Facilitates easy schema modifications without disrupting the system.
5. **Query Performance Optimization**: Can lead to faster query retrieval times.
6. **Facilitates Database Design**: Guides systematic organization of data for better understanding.
7. **Storage Space Reduction**: Despite additional tables, it often optimizes storage usage.

In summary, normalization ensures efficient, scalable, and maintainable databases by enhancing data integrity, reducing redundancy, and improving query performance.

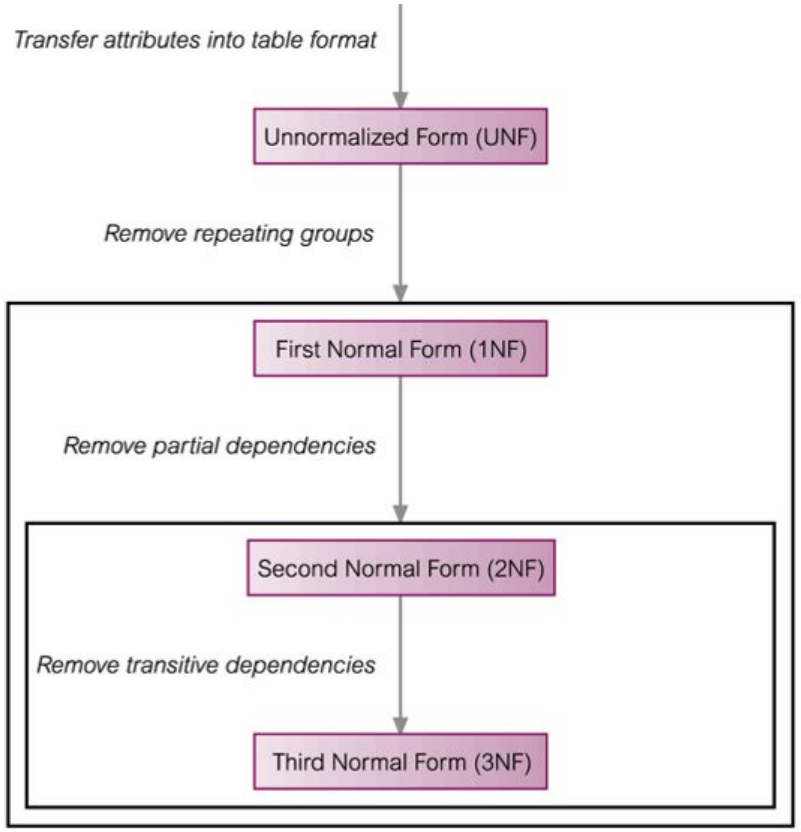


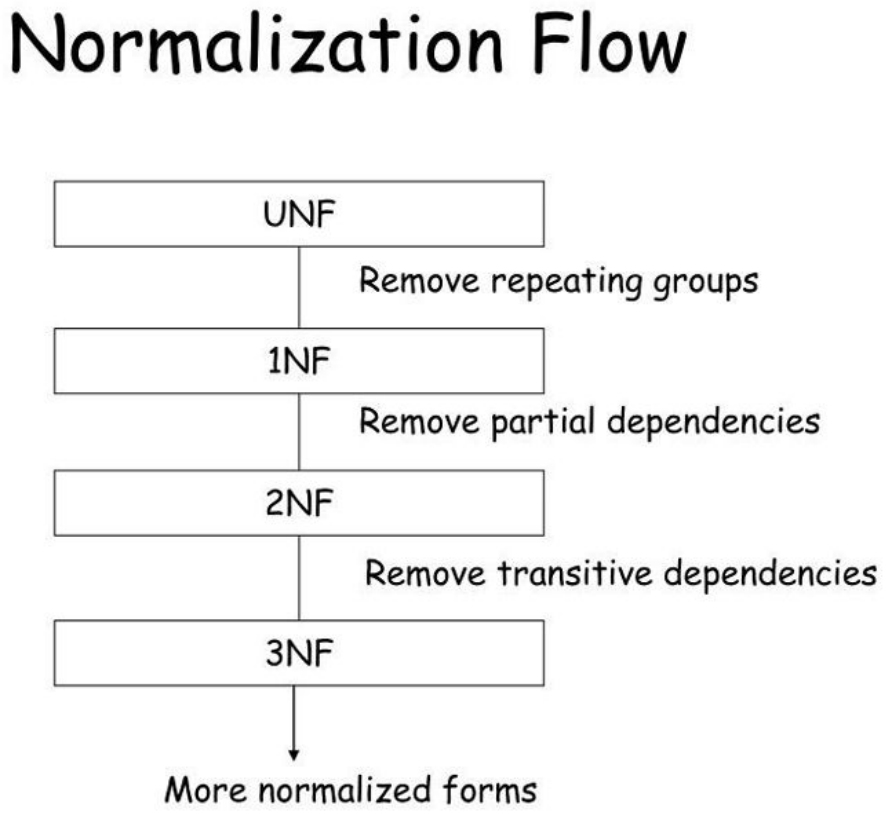


Normalization Types

Normalization rules are divided into the following normal forms:

1. **First Normal Form**
2. **Second Normal Form**
3. **Third Normal Form**
4. **BCNF**
5. **Fourth Normal Form**
6. **Fifth Normal Form**





Normalization Process

First Normal Form (1NF)

For a table to be in the First Normal Form, it should follow the following rules:

1. **It should only have single(atomic) valued attributes/columns.**
2. **No repeating values in a group.**
3. **No repeating group.**

Repeating columns means same kind of info in different columns.

Repeating columns or repeating values = violates 1NF.

Employee

EmployeeId	Name	Address	Phone
201	Saghir	R288 Karachi	033255 ,033674
202	Harris	G25 Yorkshire	033543
203	Maxwell	K87 Surrey	035872 ,036536 ,035972
204	Andy	Y78 NewCastle	038896
205	Simon	R288 London	038745
206	Sam	F7 Manchester	031210 ,033825
207	Jim	R88 London	031247 ,033111 ,033755
208	Taylor	A4 Manchester	033351 , 033751

Primary

Repeating values violating 1NF

Employee

EmployeeId	Name	Address	Phone 1	Phone 2	Phone 3
201	Saghir	R288 Karachi	033255	033674	
202	Harris	G25 Yorkshire	033543		
203	Maxwell	K87 Surrey	035872	036536	035972
204	Andy	Y78 NewCastle	038896		
205	Simon	R288 London	038745		
206	Sam	F7 Manchester	031210	033825	
207	Jim	R88 London	031247	033111	033755
208	Taylor	A4 Manchester	033351	033751	

Primary key

Repeating Columns

Employee

EmployeeId	Name	Address
201	Saghir	R288 Karachi
202	Harris	G25 Yorkshire
203	Maxwell	K87 Surrey
204	Andy	Y78 NewCastle
205	Simon	R288 London
206	Sam	F7 Manchester
207	Jim	R88 London
208	Taylor	A4 Manchester

Phone

Phone	Name
033255	Saghir
033674	Saghir
033543	Harris
035872	Maxwell
036536	Maxwell
035972	Maxwell
038896	Andy
038745	Simon
031210	Sam
033825	Sam
031247	Jim
033111	Jim
033755	Jim
033351	Taylor
033751	Taylor

Database can easily handel more than one columns But dont need voilation of normalization

First Normal Form (1NF)

Example:

Example 1 – address and name fields are composite

Name	Address	Phone
Sally Singer	123 Broadway New York, NY, 11234	(111) 222-3345
Jason Jumper	456 Jolly Jumper St. Trenton NJ, 11547	(222) 334-5566

Example 1

ID	First	Last	Street	City	State	Zip	Phone
564	Sally	Singer	123 Broadway	New York	NY	11234	(111) 222-3345
565	Jason	Jumper	456 Jolly Jumper St.	Trenton	NJ	11547	(222) 334-5566

- Address field has been expressed in terms of constituent parts, such as street, city and postcode

Name field has been expressed in terms of last name and first name

First Normal Form (1NF)

Example 2 – repeating columns for each client & composite name field

Rep ID	Representative	Client 1	Time 1	Client 2	Time 2	Client 3	Time 3
TS-89	Gilroy Gladstone	US Corp.	14 hrs	Taggarts	26 hrs	Kilroy Inc.	9 hrs
RK-56	Mary Mayhem	Italiana	67 hrs	Linkers	2 hrs		

Example 2

- Table structure has been changed
- Data related to representative repeated
- Representative name expressed in terms of last name and first name

Rep ID	Rep First Name	Rep Last Name	Client	Time With Client
TS-89	Gilroy	Gladstone	US Corp	14 hrs
TS-89	Gilroy	Gladstone	Taggarts	26 hrs
TS-89	Gilroy	Gladstone	Kilroy Inc.	9 hrs
RK-56	Mary	Mayhem	Italiana	67 hrs
RK-56	Mary	Mayhem	Linkers	2 hrs

Any non key field should entirely depend on its primary key

1 - Should be in 1NF

2 - No PARTIAL DEPENDENCY

3 - Occurs when there is composite key

Result

StudentId	Course	Name	Marks	Teacher
201	Software Architecture	Saghir	85	A
202	Software Design	Harris	90	B
201	Quality Assurance	Saghir	75	G
204	Enlish Language	Andy	63	O
205	History	Simon	74	L
206	Project Managent	Sam	93	G
205	Software Architecture	Simon	70	A
208	Quality Assurance	Taylor	61	G

COMPOSITE KEY

Non Key Columns

violating 2NF

Result

StudentId	Course	Name	Marks
201	Software Architecture	Saghir	85
202	Software Design	Harris	90
201	Quality Assurance	Saghir	75
204	Enlish Language	Andy	63
205	History	Simon	74
206	Project Managent	Sam	93
205	Software Architecture	Simon	70
208	Quality Assurance	Taylor	61

now this is in 2NF

Both tables have no partial dependecny

Teacher

Course	Teacher
Software Architecture	A
Software Design	B
Quality Assurance	G
Enlish Language	O
History	L
Project Managent	G

take away partailly dependent to new table

Second Normal Form (2NF)

For a table to be in the Second Normal Form,

1. It should be in the First Normal form.
2. And, it should not have **Partial Dependency**. Means all non-prime attributes should be fully functionally dependent on CK.

- Partial Dependency exists, when for a composite primary key or CK, any attribute in the table depends only on a part of the primary key and not on the complete primary key.
- Part of CK determines non-prime attribute (partial dependency)
- To remove Partial dependency, we can divide the table, remove the attribute which is causing partial dependency, and move it to some other table where it fits in well.

Second Normal Form (2NF)

Example: If we have two tables Students and Subjects, to store student information and information related to subjects.

Student table:

student_id	student_name	branch
1	Akon	CSE
2	Bkon	Mechanical

Subject Table:

subject_id	subject_name
1	C Language
2	DSA
3	Operating System

Second Normal Form (2NF)

Example: And we have another table **Score** to store the marks scored by students in any subject like this

student_id	subject_id	marks	teacher_name
1	1	70	Miss. C
1	2	82	Mr. D
2	1	65	Mr. Op

Now in the above table, the primary key is student_id + subject_id, because both these information are required to select any row of data.

But in this table we have a column teacher_name, which depends on the subject information or just the subject_id, so we should not keep that information in the Score table.

Second Normal Form (2NF)

Example:

The column **teacher_name** should be in the **Subjects** table. And then the entire system will be Normalized as per the Second Normal Form.

Updated **Subject** table:

subject_id	subject_name	teacher_name
1	C Language	Miss. C
2	DSA	Mr. D
3	Operating System	Mr. Op

Updated **Score** table:

student_id	subject_id	marks
1	1	70
1	2	82
2	1	65

Second Normal Form (2NF)

Example:

- A new field ClientID introduced
- RepId and ClientID combination acts as the primary key

Rep ID*	Rep First Name	Rep Last Name	Client ID*	Client	Time With Client
TS-89	Gilroy	Gladstone	978	US Corp	14 hrs
TS-89	Gilroy	Gladstone	665	Taggarts	26 hrs
TS-89	Gilroy	Gladstone	782	Kilroy Inc.	9 hrs
RK-56	Mary	Mayhem	221	Italiana	67 hrs
RK-56	Mary	Mayhem	982	Linkers	2 hrs

Example 2NF

Rep ID*	Client ID*	Time With Client
TS-89	978	14 hrs
TS-89	665	26 hrs
TS-89	782	9 hrs
RK-56	221	67 hrs
RK-56	982	2 hrs
RK-56	665	4 hrs

Rep ID*	First Name	Last Name
TS-89	Gilroy	Gladstone
RK-56	Mary	Mayhem

Client ID*	Client Name
978	US Corp
665	Taggarts
782	Kilroy Inc.
221	Italiana
982	Linkers

- Original table decomposed into smaller tables
- Each of them are in 2NF

Second Normal Form (2NF)

Example :

STUD_NO	COURSE_NO	COURSE_FEE
1	C1	1000
2	C2	1500
1	C4	2000
4	C3	1000
4	C1	1000
2	C5	2000

Table 2

COURSE_NO	COURSE_FEE
C1	1000
C2	1500
C3	1000
C4	2000
C5	2000

Table 1

STUD_NO	COURSE_NO
1	C1
2	C2
1	C4
4	C3
4	C1
2	C5

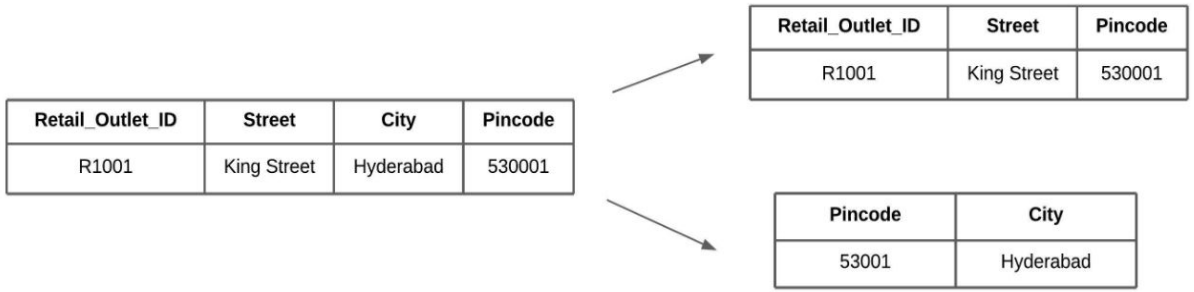
Third Normal Form (3NF)

Any non key field dependent on other non key field

1 - Should be in 1NF,2NF

2 - No TRANSITIVE DEPENDENCY

3 - Occurs when you can guess value of any column from non key colum



A relation R is said to be in 3 NF (Third Normal Form) if and only if:

1. R is **already in 2 NF**
2. There is no **transitive dependency** for non-prime attributes.

Here **transitive dependency** means non prime attribute-> non prime attribute

A transitive dependency exists when another non-key attribute determines a non-key attribute.

In other words, If A determines B and B determines C, then automatically, A determines C.

Exam

StudentId	Name	ExamType	MaxMarks
201	Saghir	Viva	20
202	Harris	Theroy	100
203	Maxwell	Practical	50
204	Andy	Practical	50
205	Simon	Viva	20
206	Sam	Theroy	100
207	Jim	Theroy	100
208	Taylor	Practical	50

Primary key

violating 3NF

MaxMarks transitively depends on Examtype

we can Guess value of MaxMarks from ExamType

VIVA = 20

THERORY=100

PRACTICAL=50

Exam

StudentId	Name	ExamType
201	Saghir	Viva
202	Harris	Theroy
203	Maxwell	Practical
204	Andy	Practical
205	Simon	Viva
206	Sam	Theroy
207	Jim	Theroy
208	Taylor	Practical

Marks

ExamType	MaxMarks
Viva	20
Theroy	100
Practical	50

Linke both tables by ExamType common table

Employee

EmployeeId	Name	Designation	Salary
201	Saghir	Manager	80000
202	Harris	Leturer	40000
203	Maxwell	Manager	80000
204	Andy	Lecturer	40000
205	Simon	Worker	10000
206	Sam	Worker	10000
207	Jim	Lecturer	40000
208	Taylor	Lab Assistant	25000

Salary transitively depends on Designation

Employee

EmployeeId	Name	Designation
201	Saghir	Manager
202	Harris	Leturer
203	Maxwell	Manager
204	Andy	Lecturer
205	Simon	Worker
206	Sam	Worker
207	Jim	Lecturer
208	Taylor	Lab Assistant

Salary

Designation	Salary
Manager	80000
Leturer	40000
Worker	10000
Lab Assistant	25000

Whenever you can guess the value from non key column its violating 3nf

Match

Match#	Teams	Ground	Capicity
01	Aus v Eng	MCG	80000
02	Zim v Eng	Hobart	30000
03	Sl v Ind	SCG	65000
04	Pak v Wi	MCG	80000
05	SA v NZ	Hobart	30000
06	Ausv Ind	Perth	45000
07	Pak v Ind	Canbera	10000
08	Eng v Sl	MCG	80000

Capicity transitively depends on Ground

Match

Match#	Teams	Ground
01	Aus v Eng	MCG
02	Zim v Eng	Hobart
03	Sl v Ind	SCG
04	Pak v Wi	MCG
05	SA v NZ	Hobart
06	Ausv Ind	Perth
07	Pak v Ind	Canbera
08	Eng v Sl	MCG

Capicity

Ground	Capicity
MCG	80000
Hobart	30000
SCG	65000
Perth	45000
Cambera	10000

Third Normal Form (3NF)

Example:

Roll no	State	City
1	Sindh	Hyderabad
2	Punjab	Lahore
3	Sindh	Karachi
4	Punjab	Faislabad
5	KPK	Peshawar
6	Sindh	Sukkur

CK is roll no

Roll no-> State

State -> City

Here Prime Attribute is Roll no

Non Prime Attributes are State, City

Here transitive dependency exists as Roll no ->State and State -> City so we can say Roll no -> City.

So this relation is not in 3rd Normal form

Third Normal Form (3NF)

Example: Before 3NF

Roll no	State	City
1	Sindh	Hyderabad
2	Punjab	Lahore
3	Sindh	Karachi
4	Punjab	Faislabad
5	KPK	Peshawar
6	Sindh	Sukkur

After Normalization

Roll no	State ID	City ID

State ID	State

City ID	City

RollNo	Name	VoterID	Age
1	Kamran	V123	20
2	Ahmed	V223	21
3	Suhail	V330	23

BCNF

C.K : { RollNo, VoterID }

F.D :

{ RollNo □ Name } = Valid

{ RollNo □ VoterID } = Valid

{ VoterID □ Age } = Valid

{ VoterID □ RollNo} = Valid

It is an upgraded version of the 3rd Normal form. It is also called as 3.5 Normal Form.

A relation R is said to be in BCNF if and only if:

- R is **already in 3 NF**
- L.H.S of each Functional Dependency should be Candidate Key or Super Key.
- Yes above relation in BCNF as candidate keys are Rollno and Voteid and for each FD LHS is CK

4NF and 5NF (4NF Problem)

A table is in **4NF** if:

1. It is in **Boyce-Codd Normal Form (BC**
2. It has **no non-trivial multi-valued dependencies (MVDs)** unless they are superkeys.

A table is in **5NF** if:

3. It is in **4NF**.
4. It cannot be decomposed further without **losing data**

Course	Teacher	Textbook
Math	Mr. A	Algebra
Math	Mr. A	Calculus
Math	Mr. B	Algebra
Math	Mr. B	Calculus

Problem:

- **Course →→ Teacher** (A course has multiple teachers).
- **Course →→ Textbook** (A course uses multiple textbooks).
- But **Teacher** and **Textbook** are independent (a teacher doesn't determine the textbook).

This leads to **redundant combinations** (e.g., if a new teacher is added, we must pair them with all textbooks).

Example NFs

UNF

managerID	managerName	area	employeeID	employeeName	sectorID	sectorName
1	Adam A.	East	1	David D.	4	Finance
			2	Eugene E.	3	IT
2	Betty B.	West	3	George G.	2	Security
			4	Henry H.	1	Administration
			5	Ingrid I.	4	Finance
3	Carl C.	North	6	James J.	1	Administration
			7	Katy K.	4	Finance

Example NFs

1NF

managerID	managerName	area	employeeID	employeeName	sectorID	sectorName
1	Adam A.	East	1	David D.	4	Finance
1	Adam A.	East	2	Eugene E.	3	IT
2	Betty B.	West	3	George G.	2	Security
2	Betty B.	West	4	Henry H.	1	Administration
2	Betty B.	West	5	Ingrid I.	4	Finance
3	Carl C.	North	6	James J.	1	Administration
3	Carl C.	North	7	Katy K.	4	Finance

Example NFs

2NF

managerID	managerName	area
1	Adam A.	East
2	Betty B.	West
3	Carl C.	North

Example NFs

2NF

employeeID	employeeName	managerID	sectorID	sectorName
1	David D.	1	4	Finance
2	Eugene E.	1	3	IT
3	George G.	2	2	Security
4	Henry H.	2	1	Administration
5	Ingrid I.	2	4	Finance
6	James J.	3	1	Administration
7	Katy K.	3	4	Finance

Example NFs

3NF

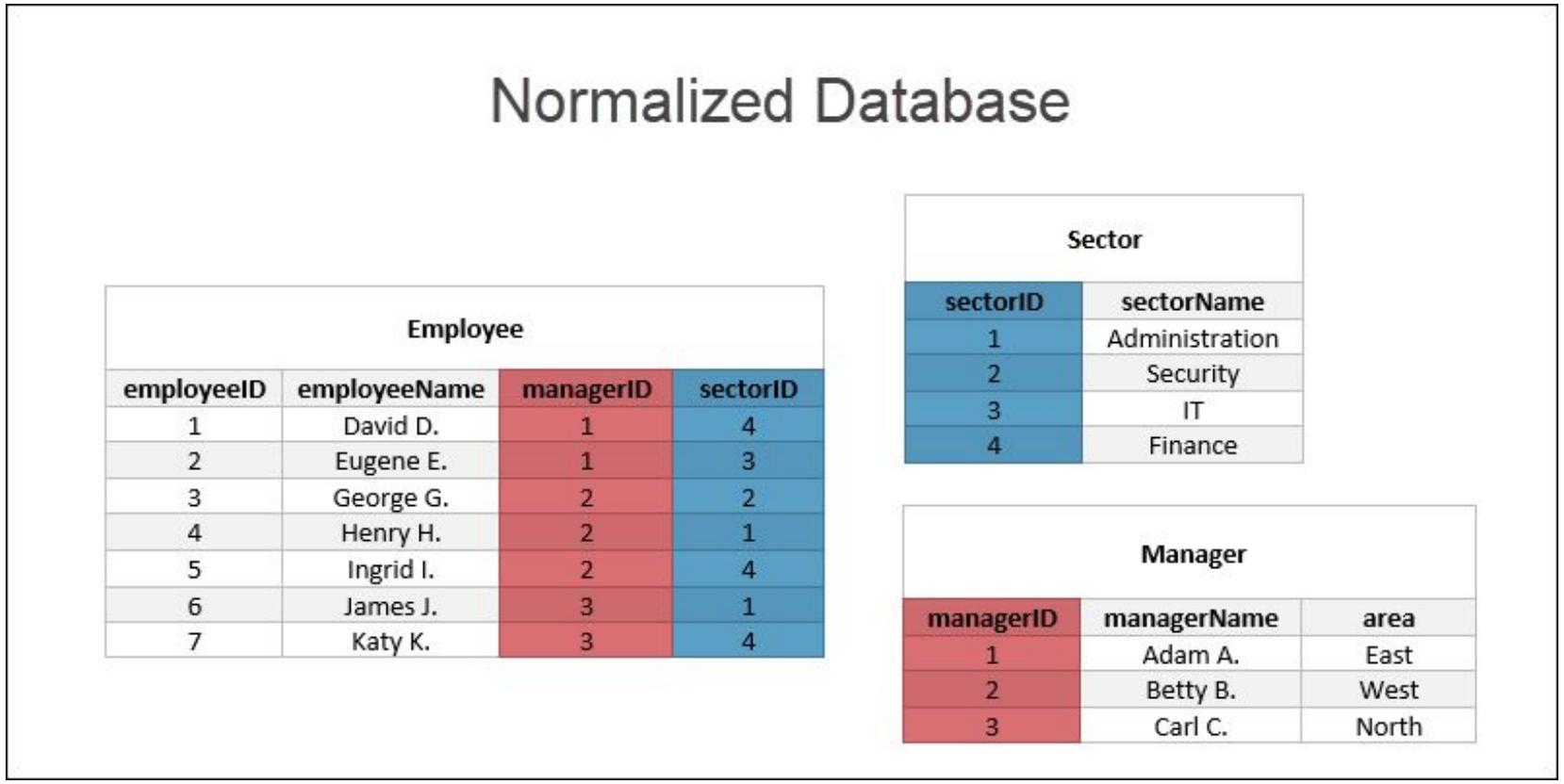
employeeID	employeeName	managerID	sectorID
1	David D.	1	4
2	Eugene E.	1	3
3	George G.	2	2
4	Henry H.	2	1
5	Ingrid I.	2	4
6	James J.	3	1
7	Katy K.	3	4

Example NFs

3NF

sectorID	sectorName
1	Administration
2	Security
3	IT
4	Finance

Example NFs


Normalized Database
Employee
employeeID | employeeName | managerID | sectorID
1 | David D. | 1 | 4
2 | Eugene E. | 1 | 3
3 | George G. | 2 | 2
4 | Henry H. | 2 | 1
5 | Ingrid I. | 2 | 4
6 | James J. | 3 | 1
7 | Katy K. | 3 | 4
Sector
sectorID | sectorName
1 | Administration
2 | Security
3 | IT
4 | Finance
Manager
managerID | managerName | area
1 | Adam A. | East
2 | Betty B. | West
3 | Carl C. | North