Akhila K.S
18/06/23

* why do we need normalization in DBMS

- **Normalization**

Process of minimizing redundancy
(Data redundancy → basically reputation of data)

For example

Id	Name	age	Branch	HOD
1	Anu	18	CS	Shibly
2	Babu	18	CS	Shibly
3	Celin	19	CS	Shibly

As you can see {Id Name age} these datas are different but the datas of Branch, HOD repeated every time (Repeating the same data every time)

∴ Normalization basically removes the repeated datas.

Also Normalization removes
→ Insertion anomaly
→ Updation anomaly
→ Deletion anomaly.
} understand later.

∴ Normalization divides the larger table into smaller tables and link them using relationship.

When you have a really big table all the information is in just one single table. Which is wrong ∴ Normalization removes that big table into smaller tables. and links them.

Id	Name	age	Branch	HOD
1	Anu	18	CS	Shibly
2	Babu	18	CS	Shibly
3	Celin	19	CS	Shibly

Actually this one table is equivalent to student & Branch table

Id	Name	age	branchid
1	Anu	18	101
2	Babu	18	101
3	Celin	19	101

Branch	HOD	Branchid
CS	Shibly	101
~~CS~~	~~Shibly~~	
~~CS~~	~~Shibly~~	

Foreign key links thes 2 table.

These 3 anomaly occurs when the table is not in the normalized form, when the table is the big of only one table. So this over here

(i) Insertion Anomaly

* — Repetition of Data (Branch & HOD Info repeated every single time)
  — when table is in normalized form it will remove (instead of writing branch id while removing)
* If branch is not alloted, it has to be put as 'NULL' (new id over id Keep it null of can't insert that id on table)

(Branch column has a restriction that you can't insert a null value over there. So you will not be able to insert that id information at all)

(ii) Updation Anomaly

* — What if HOD is changed?
  — If HOD has change you have to change every single column of the table over here and update HOD.
  — But if it is in normalized form you want to change only over the updated table. that's it removed.

(iii) Deletion Anomaly

— 2 different informations are kept together
If the student records are deleted also branch record will be deleted (id remove Branch will loss)

Types of Normalization

Normalization
- 1NF
- 2NF
- 3NF
- BCNF
- 4NF
- 5NF.

1NF

1st Normal form

* Rules for 1NF

① Single valued attributes (atomic values) (no multi-valued attributes)
Every column should have atomic values (only 1 value = each cell)
Eg: Subject {DBMS, OS} it will be seperated; 2 times

② Attribute domain should not change
Eg id → int only enter.
Another Attribute DOB — only dd/mm/yr not other.

③ Unique name for attributes/columns
No 2 columns can have same name.
(id → another have id)

④ Order doesn't matter.
in any order we can write

id	name	age	Subject
1	PQR	18	DBMS, OS
2	XYZ	18	DBMS
3	ABC	19	Java

→

id	name	age	Subject
1	PQR	18	DBMS
1	PQR	18	OS
2	XYZ	18	DBMS
3	ABC	19	Java

2NF

2nd normal form

Rules for 2NF
① Table should be in 1NF
② No partial dependency

Dependency ≈ Functional dependency ≈ all colums are dependent on one Primary key in table.

Partial dependency

↳ when there are 2 or more primary keys in on table (candidate key)

- Any attribute depends only on a part of ~~primary key~~ candidate key

FD depends on Student id

Student

Stud-id	Stud.-name	address

Subject

Sub-id	Sub-name

links 2 tables candidate key

Score

Score-id	Stud-id	Sub-id	marks	teacher

Basic Primary Key

Dependens on all id

where only depend on one key — partial dependency

no depend on stud id

shifting one columns to other table.

Student

Stud-id	Student-name	address

Subject

Sub-id	Sub-name	teacher

Score

Score-id	Stud-id	Sub-id	marks

No partial dependency
INF form
} 2NF form.

3NF

3rd normal Form

$a \rightarrow b$ (a depence on b)

$a \leftarrow b$ (not allowed)

Rules for 3NF

1. Table should be in 2NF
2. No Transetive dependency

Transetive dependency → when one column depends on a column which is not primary key

[Non primary key can take null values → X]

When non-prime → dependence on another non-form

- Prime attributes are primary keys which all other columns depence
- non-prime — other column attributes which depence primary key (it can be null values)

Transetive-Key

Score

Score-id	Stud-id	sub-id	marks	exam-name (non-Prime)	total-marks (non-Prime)
101	201	1	16	mid	20
102	201	2	69	final	80
103	202	1	19	mid	20

NULL (if condition) — This will also become NULL.

Score

Score-id	Stud-id	sub-id	marks	exam-id

+

Exam

Exam-id	exam-name	total-marks

(exam-id depends on total marks) → Primary key.

* BCNF 3.5 Normal Form

Boyce Codd Normal Form

Rules for BCNF

it should be in 3NF

Determinants should be a Superkey.

(OR) for every FD, LHS attrib must be a Superkey.

Not in BCNF.

A → B
↳ Superkey

A Can't be non-prime attribute with B being in Prime attribute

~~Customer id~~ ~~Brand~~ ~~Seller~~

Functional Dependency

Prime attribute → nonprime attribute

non-prime attribute → non-prime attribute

Transitive Dependency

Partial Dependency

Part of Primary Key → non-prime attribute

* non prime attribute → prime attribute

not in BCNF

Eg: College Enrollment Table

Stud-id	Subject	Professor
101	Java	P. Java
101	C++	P. CPP
102	Java	P. Java2
103	C#	P. chash
104	Java	P. Java

Stud-id – subject opted by the student Proffessor name assigned for that particular subject for a particular student

[Stud-id + subject] → Primary Key (can find other columns)

Proffesor column → Find subject (one Proffesor teaches one particular subject)

1NF ✓

- Satisfies first Normal Form
  - no - repeated values
  - unique names

- 2NF ✓ no partial Dependency

  (Stud-id, subject) → Professor
  Professor → subject } Dependencies

- 3NF ✓ no Transitive Dependency

- BCNF ✗ Professor can lead us to subject and subject, is a part of Candidate Key or Primary key hence it's a prime attibute while professor is a non-prime attibute.

Candidate key

Stud-id, Subject → Professor

Professor → Subject

not a superkey

Dependence

non-prime attibute.

Prime attibute

★ convert to BCNF

(College Enrollment Table
↓
Student Table + Professor Table)

Stud-id	Proff-id

Proff-id	Professor	Subject

4 NF

4th Normal Form

Rules for 4 NF

1. It should satisfy BCNF
2. It should not have Multi-valued Dependency

What multi-valued Dependency?

Any dependency:

$A \rightarrow B$ is a multi-Valued Dependency

I. If a single value of A more than one value of B

$A_1 < B_1, B_2$

II. At least table should have 3 columns

III. For a table with A B C columns $A \rightarrow B$ is multi-valued Dependency

Then B and C should be independent of each other.

* Why? Distribute multivalued data to a multiple rows will solve the problem

A	B
A_1	B_1, B_2

A	B
A_1	B_1
A_1	B_2

I + II + III → True so multi-valued Dependency

Multi-valued dependency between $A \Rightarrow B$ and $A \Rightarrow C$

{Stud-id, course opted by student & Stud hobby

Eg:

S-id	Course	hobby
1	Science	Cricket
1	Maths	Hockey
1	C#	Cricket
2	Php	Hockey
2		

no-relationship

Science Hockey

Maths Cricket

∴ Seperate Independent table.

Remove multi-valued Dependency

Student Enrollment Table

Course Opted Table (S_id & Course) + Hobbies Table (Si-id & hobby)

Course Opted	TABLE
S-id	Course
1	Science
1	Maths
2	C#
2	PhP

Hobbies	TABLE
S-id	Hobby
1	Cricket
1	Hockey
2	Cricket
2	Hockey

A table can have both F.D and multi-valued Dependency together.

"Address add to Enrollment table"

S-id	address	Course	hobby

S-id → address F.D

(Sid → Course, Sid → Hobby) m.v.D

Course-opted Table (S-id & course) + Hobbies Table (S-id & hobby) + Address Table (S-id & address)

Bad Database Design → Reason for MV.D

Student Table

S-id	S-name
S1	A
S2	B

Course Table

C-id	C-name
C1	C
C2	D

When doing Cartesian Product of the table and Keep the all dates to one single table.

like; Enrolment Table (No, M.V.D)

S-id	S-name	C-id	C-name
S1	A	C1	C
S1	A	C2	D
S2	B	C1	C
S2	B	C2	D

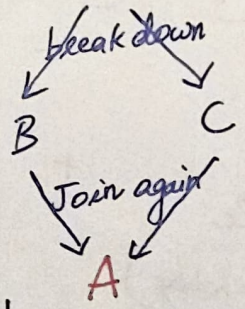
5 NF

$\boxed{5^{th} \text{ Normal Form}}$ PJNF Project Join Normal Form

Rules for 5 NF

1. Should be in 4NF
2. et Should not have Join Dependency

A a relation with JOIN dependency



get a same relation again!

IF JOIN Dependency doesn't exist then either data is lost or new entries are created

And opposite is also true. 5NF under stand

2 ways

by learning JOIN Dependency

basic concept of breaking down

Eg: SPC Table

Supplier	Product	Customer

Composite, primary Key.

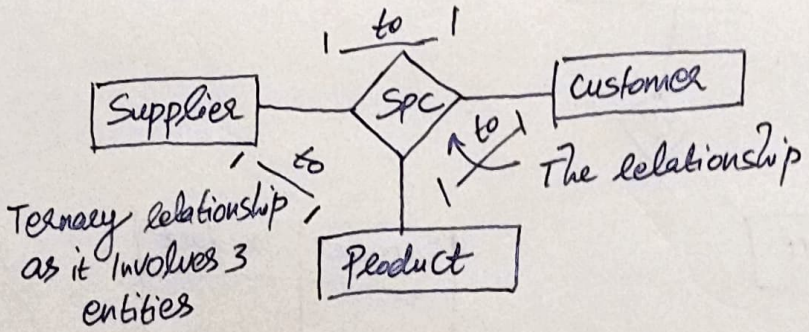
SPC Table

Supplier	Product	Customer
ACME	T2X SW	FORD
ACME	GEAR L	GM
ROBUSTO	E SWITCH	FORD
ROBUSTO	OBD II	MERCEDES
ALWAT	T2X SW	GM
ALWAT	OBD II	MERCEDES
ALWAT	GEAR L	MERCEDES

5th Normal Form (Requirements)

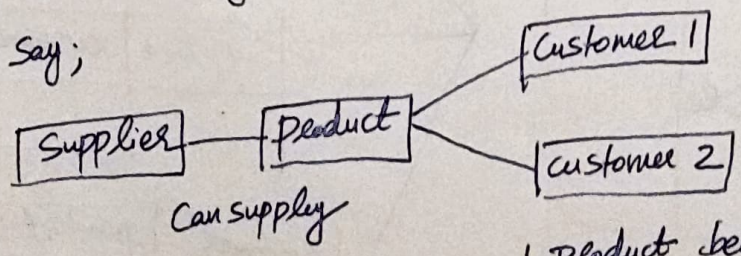
1. Table should satisfy the 4th Normal Form
2. If JOIN Dependency exist decompose the table

SPC Table ER Diagram

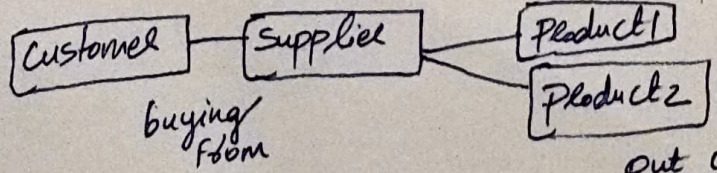


From the ER-Diagram

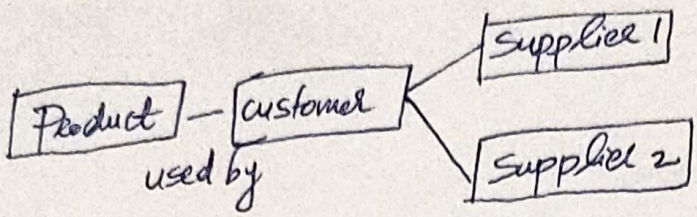
we can say;



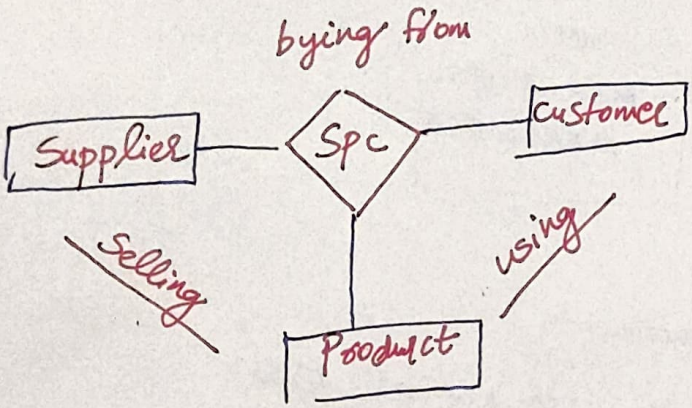
1 product being supplied to multiple customers



out of the product that a given supplier supplies.



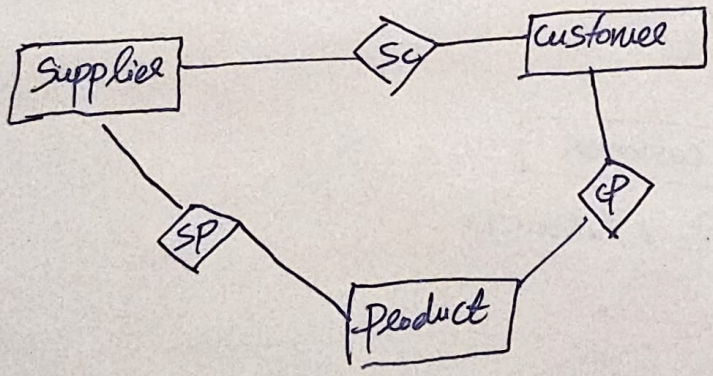
If multiple suppliers supply the same product



Binary Relationships

Supplier - customer
customer - product
Product - supplier

Decompose the main table.



Supplier	product	customer
	Spc Table	

Supplier	product

Supplier	customer

customer	product

Supplier	Product	Customer
ACME	72XSW	FORD
ACME	GEARL	GM
ROBUSTO	E-SWITCH	FORD
ROBUSTO	OBD II	MERCEDES
ALWAT	72XSW	GM
ALWAT	OBD II	MERCEDES
ALWAT	GEAR L	MERCEDES

SUPP_ PRO TABLE

Supplier	product
ACME	72XSW

SELLS

SUPP-CUST TABLE

~~ACME~~ ~~FO~~

Supplier	customer
ACME	FORD

SUPPLIES TO

CUST- PRO - TABLE

Customer ~~Supplier~~	product
FORD	72X SW

USES

ACME Sells 72X SW
FORD uses 72X SW
ACME Supplies to FORD ✓

← ACME sells 72XSW to FORD ✗

JOIN DEPENDENCY

A
break down
B C

Additional Information is created
or Information is lost
→ ∴ can't decompose

if breaking down the table doesn't lead to information loss then decompose the table → 5NF works.

Summary ★

1NF → Single valued attributes
unique domain

2NF → Partial Dependency Avoid, 1NF Form

3NF → No Transitive Dependency, 2NF Form

4NF → BCNF, Multivalued Attribute

BCNF → 3NF, Determinants Should be a superkey

5NF → 4NF, Avoid Joint Dependency