Unit-III

Relational Database Design

Functional Dependencies

Functional dependencies are the result of interrelationship b/w attributes or in b/w tuples in any relation.

Defination: In relation R, X & Y are the two subsets of the set of attributes, Y is said to be functionally dependent on X if a given value of X (all attributes in X) uniquely determines the value of Y (all attribute in Y)

It is denoted by $X \rightarrow Y$ (Y depends upon X)

Determinant: Here X is known as determinant of functional dependency.

eg:

Employee

Eid	Name	Salary
1	A	10,000
2	B	20,000
3	C	30,000
4	D	40,000

Here,

$X(Eid) \rightarrow Y(Name, Salary)$

$X(Eid) : Y(Name, Salary)$

With the help of Eid, we can easily access info name, salary. (dependent) so, name & salary depends on eid.

→ Functional Dependency Chart:

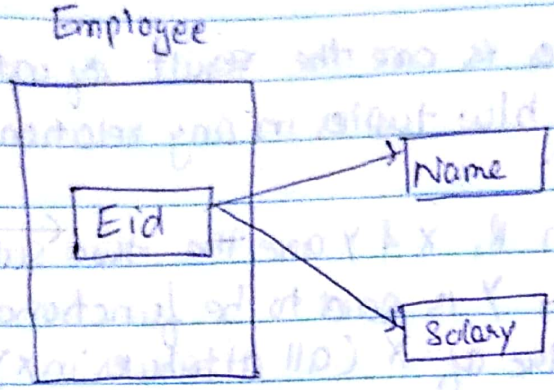
It is the graphical representation of functional dependencies among attributes in any relation.

The following steps to follow to draw FD Chart:

① Find out the primary key attribute.

Scanned by CamScanner

2. Make a rectangle & write all the primary key attributes inside it
3. Write all non prime key attributes outside the rectangle.
4. Use arrow to show FD among attributes



→ Types of Functional Dependencies:

There are 4 major types of FD's.

① Partial Dependency & Fully Functional Dependency

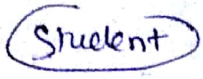
- Partial Dependency:

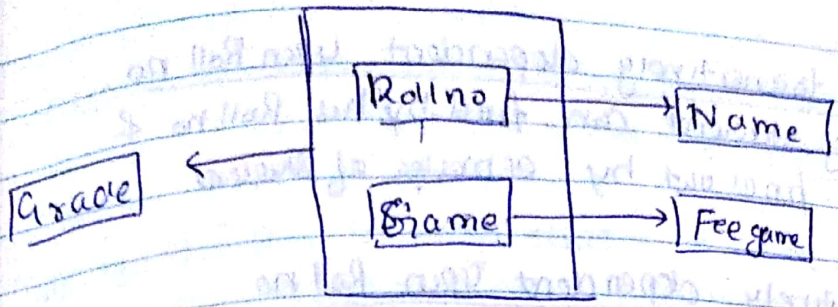
* Suppose we have more than one attribute in primary key * Let A be the non-prime key attribute * If value of A is not dependent upon all prime key attributes then A is said to partially dependent.

- Fully Functional Dependency:

* Suppose we have more than 1 attribute in primary key
* Let A be the non prime key attribute & value of A is dependent upon all prime key attributes
* Then A is said to be fully functional dependent.

Scanned by CamScanner





here
Prime key = Rollno & attribute = Game
Non-prime attribute = Name, Fee game, Grade.

Here, Name & Fee are partial dependent bec. we can find name by Rollno & Fee game by Game.

& Grade is Fully Functional Dependent bec. we can find the grade of any student in a particular game if we know Rollno & game of that student.

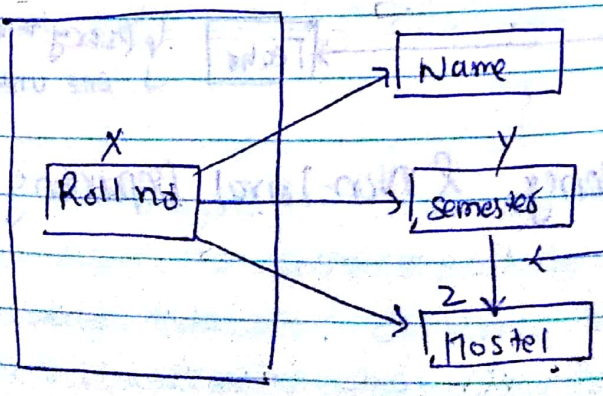
(2) Transitive Dependency & Non-Transitive Dependency:

- Transitive Dependency:

* It is due to dependency b/w non-prime key attributes.
* Suppose in a relation R, $X \to Y$ (Y depends upon X), $Y \to Z$ (Z depends upon Y) then $X \to Z$ (Z depends upon X).
* ∴ Z is said to be transitively dependent upon X.

- Non Transitive Dependency:

* Any Functional dependency which is not transitive is known as Non-transitive dependency.
* It exists if there is no dependency b/w non prime key attributes.



Scanned by CamScanner

Here, hostel is transitively dependent upon Roll no. Semester of any student can find by his Roll no. & hostel can be find out by semester of student.

& Name is transitively dependent upon Roll no

(3) Single valued Dependency & Multivalued Dependency.

- Single valued dependency: In any relation R, if for particular value of X, Y has single value then it is known as single valued dependency.

$X \longrightarrow Y$.

- Multivalued Dependency: In any relation R, if for a particular value of X, Y has more than one value, then it is known as MVD.

$X \longrightarrow\!\!\!\!\rightarrow Y$

ID	Teacher	Class	Days
1	A	I.T.	6
2	B	CSE	10
1	A	ECE	15

MVD

Teacher →→ Class } teacher can take more than one class

Class → Days } Class can be more than one day

SVD.

ID → Teacher } Every teacher has one unique id.

(4) Total Dependency & Non-Total Dependency.

Scanned by CamScanner

588
521

Emp-id	E-name	E-salary

- Trival F.D: In any relation R, $X \to Y$ is trival if $Y \subseteq X$ (Y is the subset of X)
- Non-Trival F.D: In any relation R, $X \to Y$ is non-trival if $Y \not\subseteq X$ (Y is not the subset of X).

Here,

Emp-id ⟶ Ename. (non-trival).

{Emp id, Ename} ⟶ {E-name}. (Trival)

Anamolies in Relational Database.

Anamolies: It refers to the unexpected results bec. of modification of data.

Eg: Employ

Eid	Ename	Salary	Dept-No	Dept-name
1	A	5,000	2	Accounts
2	B	6,000	2	Accounts
3	C	7,000	1	Sales
4	D	8,000	5	Marketing
5	E	9,000	5	Marketing
6	F	10,000	2	Accounts
7	G	11,000	5	Marketing
			3.	Finance → Can not be inserted.
8	H.	12,000	2	Accounts

→ Insertion Anamolies:

* Suppose you want to add new info in any relation but can not enter data bec. of some constraints. This is known as insertion anamolies.

Scanned by CamScanner

* In any relation employ, u can not add new department finance unless there is no employ in finance department

* Addition of this info violates entity-integrity rules that is primary key can not be NULL.

→ Deletion Anamolies:

* Deletion Anamolies occurs when you try to delete any existing info from any relation & this causes deletion of any undesirable information.

Ex: Deletion of C leads to delete sales dept. completely.

→ Updation Anamolies:

Upd. Anamolies occurs

* When you try to update any existing info in any relation & this causes inconsistency of data.

Attribute Closure $[X^+]$. (not in sgilb)

Set of attributes that is defined by X.

R(A B C D) FD. (A→B, B→C, C→D)

A → B → ABCD
B → C
C → D

i.e. (A^+) → Attribute Closure.

R(A B C D E) FD's {AB→C, C→D, B→E}

$(AB)^+$: AB → C → ABCDE
C → D
B → E

↓
Super Key & Candidate Key

(B^+) → BE
↳ Attribute Closure nahi hai Super or Cand. Key nahi hai

$(ABC)^+$ → ABCDE
↳ Best hm min attributes to key mein include karte hain)

Scanned by CamScanner

NORMALIZATION — no data loss, no data redundancy

→ Normalization is a process by which we can decompose or divide any relation into more than one relation to remove anomalies in relational database.

→ It is a step by step process & each step is known as Normal Form

→ Normalization is a reversible process.

* Properties of Normalization

(i) Remove different anomalies.
(ii) Decomposition must be lossless
(iii) Preserve necessary dependency
(iv) Reduce redundancy

Scanned by CamScanner

* Various Normal Forms -

1. First Normal Form (1NF)

A relation is in 1NF if & only if

→ This rule defines that all the attributes in a relation must have atomic domains. The

→ The values in an atomic domain are individual units.

→ It means atomicity must be present in relation.

Consider a relation "language"

Course	Content
Programming	Java, C++
Web	HTML, CSS

We re-arrange the relation (table) as below, to convert it to First Normal Form.

Course	Content
Programming	Java
Programming	C++
Web	HTML
Web	CSS

Each attribute must contain only a single value from its pre-defined domain.

2. Second Normal Form (2NF)

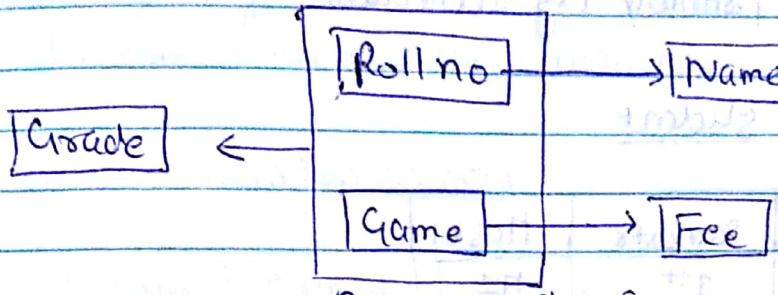
→ A relation is in 2NF if it is in 1NF & all non-prime key attributes must be fully functional dependent upon prime key attributes.

Scanned by CamScanner

Consider the relation Student

Rollno	Game	Name	Fee	Grade
1	Cricket	Akash	200	A
2	Badminton	Rohit	300	B
3	Cricket	Shiv	200	A

(a)



Dependency Chart (b)

(Student relation with anomalies)

Relation student is in 1-NF but still contains anamalies.

(i) Deletion anamoly: Suppose u want to delete Rohit but after deleting it you loose all info. of Badminton bec. he is the only player for badminton.

(ii) Insertion anamoly: Suppose u want to add new game basketball but it is u can not add this untill there is no a player for this

(iii) Updation anamoly: Suppose u want to change Fee of Cricket Then u have to search all the students participated in cricket & update fee individually Otherwise it produces inconsistency.

The soln of this problem is to separate Partial dependencies & Fully functional dependencies. So, divide Student relation into 3 parts!

Student

Rollno	Name

Games

Game	Fee

Performance

Rollno	Game	Grade

Now, deletion, insertion & updation operations can be performed without causing inconsistency.

Scanned by CamScanner

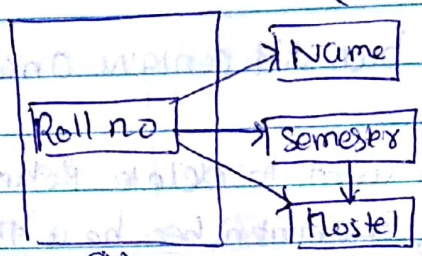
3. **Third Normal Form (3NF)**

→ A relation is in Third Normal Form if it is in 2NF & non prime key attributes must be non-transitively dependant upon primary key attributes.

Consider a relation student

Roll no	Name	semester	Hostel
1	Akash	1st	H1
2	Rohit	2nd	H2

(a)



(b)

Student relation is in 2NF but still contains anomalies

(i) Deletion anamoly: suppose u want to delete Rohit but after deleting it u loose all info. of hostel H2 bec. he is the only student in H2.

(ii) Insertion anamoly: Suppose u want to add new hostel H3 but u can not add this until there is a student for this

(iii) Updation anamoly: Suppose u want to change the hostel of 1st yr Student's then u have to search all the students of 1st sem ~~individually~~ & update them individually otherwise it causes inconsistency.

The ~~Student~~ Selin of this problem is to divide relation student into two relations:

Student.

Roll no	Name	semester

Hostel

semester	Hostel

Scanned by CamScanner

Now, deletion, insertion & updation operations can be performed without causing inconsistency.

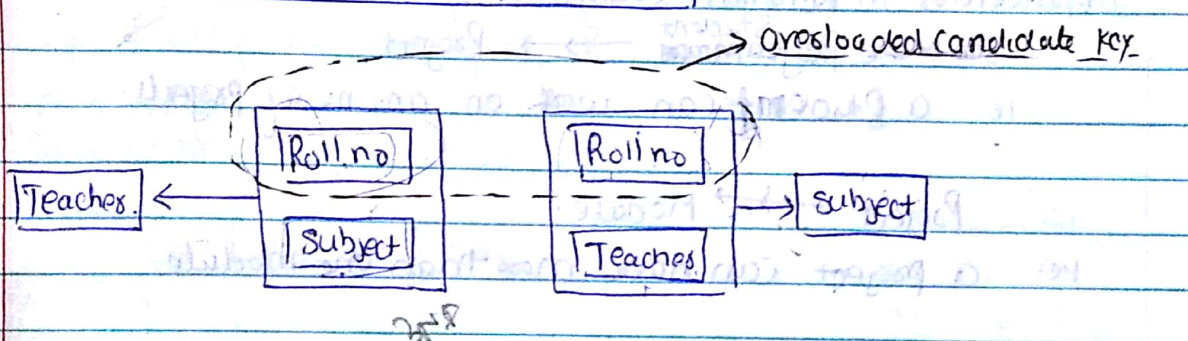
44

4. Boyce Codd Normal Form (BCNF)

→ BCNF is a strict format of 3NF
→ A relation is in BCNF if & only if all determinants are candidates Keys
→ BCNF deals with multiple candidate Keys.

Consider a relation Student

Roll no	Subject	Teacher
1	C	T1
2	C++	T2
3	C	T3
4	Java	T4



Student relation is in ~~BCNF~~ but still contains anamolies

(i) Deletion anamoly: Suppose u want to delete all info of Roll no 2 but after deleting it u loose all info. of Teacher T2 that teaches subject C++
(ii) Insertion anamoly: Suppose u want to add new subject HTML but u can't add this until there is a student for this.
(iii) Re-Updation anamoly: Suppose u want to change teacher of subject C then u have to search all the students having subject C & update them individually.

The soln of this problem is to divide relation Student into two relations:

(Student_Teacher)

Roll NO	Teacher

(Subteacher)

Teacher	Subject

Scanned by CamScanner

5 **Fourth Normal Form (4NF)**

→ A relation is in 4NF if it is in BCNF & for all multi-valued functional dependencies (MVD) of the form $X \rightarrow\rightarrow Y$ either $X \rightarrow Y$ is a trival MVD or X is a super key of relation.

Consider a relation student-details

Student Name	Project	Module
S1	1	M1
S2	1	M2
S1	2	M1

Dependencies in Relation student details are -

Student $\rightarrow\rightarrow$ Project

i.e. a student can work on any no of projects.

Project $\rightarrow\rightarrow$ Module.

i.e. a project can have more than one module.

Student-details Relation is in BCNF but still contains anomalies

(i) Deletion anamoly : Suppose u want to delete Project 2 but after deleting it u loose all info of about Student 2

(ii) Insertion anamoly : Suppose u want to add new project 3 but u can not add this until there is a student for this

(iii) Updation anamoly : suppose u want to change the name of project 2 then u have to search all the students of 2 & update them individually otherwise it causes inconsistency.

The soln of this problem is to divide relation student-details into two relations.

Scanned by CamScanner

Student - Project

Student	Project
S1	1
S2	1
S1	2

Here, Programmes is the Super Key

Project - module

Project	Module
1	M1
1	M2
2	M1

Here, Project is the super key.

6. Project Join Normal Form (PJNF) or (5NF) or Join Dependency

Sol → Join dependency: Let R be a given relation up to 4NF & it decompose (Project or divided) into $\{R_1, R_2, R_3, \ldots R_n\}$

→ The relation R satisfy the join dependency $\{R_1, R_2, R_3, \ldots R_n\}$ if & only if joining of R_1 to $R_n = R$.

Consider a XYZ relation

X#	Y#	Z#
X1	Y1	Z2
X1	Y2	Z1
X2	Y1	Z1

X1 Y1 Z1
X1 Y1 Z2
X1 Y2 Z1
X2 Y1 Z1

XY, YZ, ZX

XY

X#	Y#
X1	Y1
X1	Y2
X2	Y1

YZ

Y#	Z#
Y1	Z2
Y2	Z1
Y1	Z1

ZX

Z#	X#
Z2	X1
Z1	X1
Z1	X2

Join over Y#

Join over Z#, X#

XYZ

X#	Y#	Z#
X1	Y1	Z2
X1	Y1	Z1
X1	Y2	Z1
X2	Y1	Z2
X2	Y1	Z1

not in original XYZ

X#	Y#	Z#
X1	Y1	Z2
X1	Y2	Z1
X2	Y1	Z1
X1	Y1	Z1

original XYZ

Scanned by CamScanner

→ 5NF is the ultimate Normal Form. A relation in 5NF is guaranteed to be free of anomalies.

Scanned by CamScanner