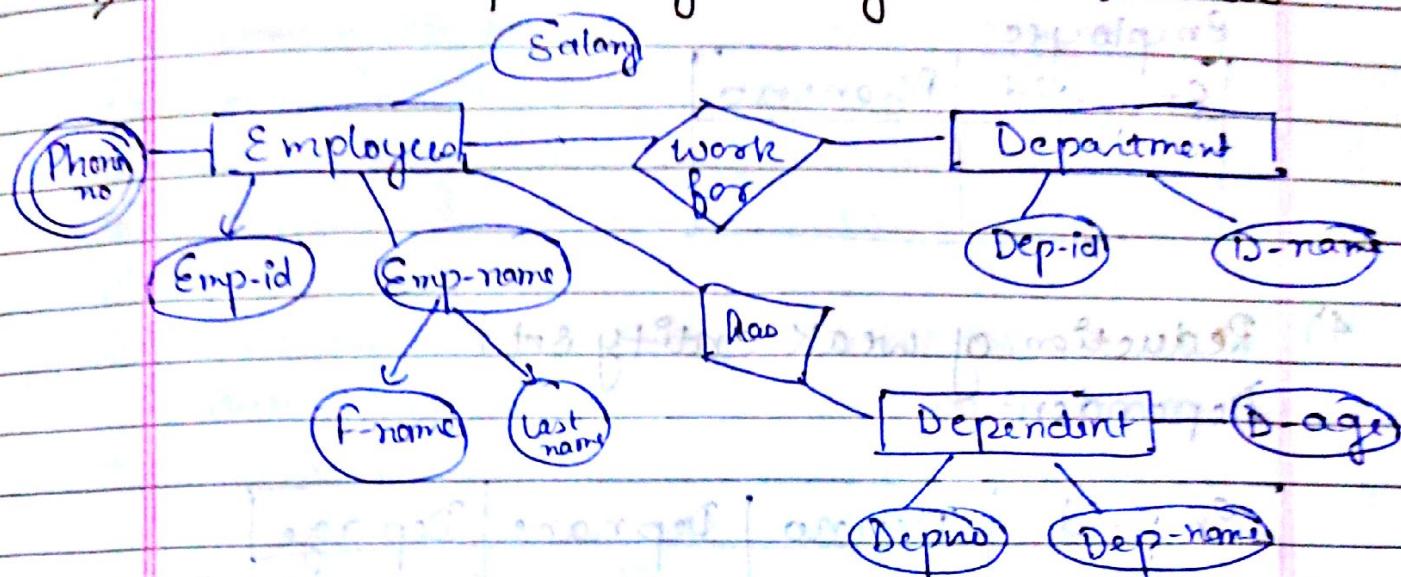


Q5 Construct an E-R diagram of University system.

## Reduction of E-R diagram into tables

### i) Reduction of Strong entity Set into table



Key attributes of Employee

| Empid | E-name | Salary |
|-------|--------|--------|
|       |        |        |
|       |        |        |

Department

| Dep-id | Dept-name |
|--------|-----------|
|        |           |
|        |           |

Reduction of Composite attribute into table

Employee

| Emp-id | f-name | l-name | Salary |
|--------|--------|--------|--------|
|        |        |        |        |
|        |        |        |        |

3) Reduction of multi valued attributes  
Phoneno.

Employee

| Emp-id | Phoneno |
|--------|---------|
|        |         |

4) Reduction of weak entity set +  
Dependent

| Emp-id | Dep no | Depname | Dep age |
|--------|--------|---------|---------|
|        |        |         |         |

5) Reduction of relationship Sets

work for

| Emp-id | Department id |
|--------|---------------|
|        |               |

RDBMS :-

- 3 Components :-
- Data Structure
- Data integrity
- Data Manipulation

Candidate key is a subset of superkey.

Key :- It is a set of one or more columns whose combined values are unique among all the

① Candidate key:

Occurrences in a given table for the all

Types of keys?

- 1) Candidate key.
- 2) Super key
- 3) Primary key
- 4) Foreign key
- 5) Alternate key
- 6) Artificial key
- 7) Composite key

i) Candidate Key: They are those attributes of the relational, which have the properties of uniqueness and irreducibility.

- { a) Uniqueness: No legal value of R ever contains two distinct tuples with same values for K.
- b) Irreducibility: No proper subset of K has the uniqueness property.

For example if the combination of (Name, class) is unique, then it can be identified as the candidate key if and only if Name and class individually are not unique.

- 2) Super key: Super key follow the property of uniqueness, but not irreducibility. A super key has a uniqueness property but not necessarily the irreducibility property. A candidate key is a special case of a super key.

e.g. If Roll-no is unique in relation STUDENT then, the set of attribute (Roll-no, Name, Class) is a Super key for a relation STUDENT, these set of attributes are also unique, but this combination of key is not having the property of irreducibility.

→ Relation of patient in which Patient-number is unique. The patient-number is a candidate key and (Patient-number, Patient name) is a superkey.

'A superset of a candidate key is a superkey.'

3) Primary Key: The primary key is an attribute or a set of attributes that uniquely identify a specific instance of an entity. Primary key cannot contain any null value because we cannot uniquely identify multiple null values. ~~so~~ primary key is a candidate which is chosen by database designer as primary key.

4) Alternate Key: Exactly one of those candidate keys is chosen as the primary key and the remainder, if any, are then called Alternate Keys. An alternate key is a function of all candidate key minus primary key.

5) Composite keys: A primary key that is made up of more than one attribute is known as Composite key.

## Ex :- WORK

| Employee ID | Project ID | Hours Worked |
|-------------|------------|--------------|
| 01          | 01         | 200          |
| 01          | 02         | 120          |
| 02          | 01         | 50           |
| 02          | 03         | 120          |
| 03          | 03         | 100          |
| 03          | 04         | 200          |

It is subset of Superkey.

- 6) Artificial keys :- An artificial key is one that has no meaning to the business or organization and perform function of a primary key in relation.
- Artificial key are permitted when
- no attribute has all the primary key properties
  - the primary key is large and complex

Eg:-

| Enrollment |       |        |
|------------|-------|--------|
| Student    | Class | Row-id |
| AK         | PK    |        |

- 7) Foreign key :- Foreign key are the attributes of the table which refers to primary key of some another table. Foreign key are used to link together two or more different tables which have some form of relationship with each other. These foreign keys is a reference to the tuple of a table from which it was taken, this tuple being called as Reference or target tuple.

Target  
Attribute

Foreign key

Department → Target table

Cg: Employee

| Emp-id | Name | Salary | Dep-id |
|--------|------|--------|--------|
| 1      | Abc  | 30000  | 10     |
| 2      | Xyz  | 40000  | 20     |
| 3      | Pqr  | 50000  | 30     |

| Dep-id | Dep-Name |
|--------|----------|
| 10     | Sales    |
| 20     | Market   |
| 30     | Product  |

eg: Student

| Rno | Name | Class-Code |
|-----|------|------------|
| 1   | A    | 2          |
| 2   | B    | 1          |
| 3   | C    | -          |

class

| Classcode | Name   |
|-----------|--------|
| 1         | B.TECH |
| 2         | B.TECH |
| 3         | BBA    |

- ② Data integrity :- Basically it consists of two rules
- Entity Integrity Rule
  - Referential Integrity Rule

① Entity Integrity Rule :- This rule states that in a relation, the value of the attribute of a primary key cannot be null.

② Referential Integrity :- It states that if a foreign key exists in a relation, either the foreign key value must match the primary key value of some tuple in its home relation or the foreign key value must be completely null.

| Employee |      |        |        | Department |           |
|----------|------|--------|--------|------------|-----------|
| Emp-id   | Name | Salary | Dep-id | Dep-id     | Dept name |
| 1        | A    | 3000   | 10     | 10         | Sales     |
| 2        | B    | 4000   | 20     | 20         | Market    |
| 3        | C    | 5000   | 30     | 30         | Product   |
| 4        | D    | 6000   | 40     |            |           |

either should match

**Enterprise Constraints:** These are the additional rules which are given by the user & DBA of a database. e.g.  $<$ ,  $>$ .

Codd's Rule: Founder of RDBMS is Dr. E. F. Codd  
There are 12 rule are as following:

- ① Information rule: All information represented in form of tables.
- ② Guaranteed Access rule: To Access any info we use key.
- ③ Comprehensive data Sublanguage rule: There should be a particular language to support RDBMS rule.
- ④ View updating rules: If we Change any data in Database then it should be automatically updated in all records or tables.
- ⑤ High level Insert, update & Delete: The language we used it should contain these rule insert, update & Delete and perform all function.
- ⑥ Physical data independency: Change in lower level not effect higher level and this rule support.
- ⑦ Logical Data independency: Change in conceptual level and does not effect external level.
- ⑧ Integrity Independency: RDBMS support all Integrity rule.
- ⑨ NonSubversion rule: Any language we use to access the database and that language will support our Integrity independency.
- ⑩ Systematic treatment of null Value: There should be special treatment of null value.
- ⑪ Data Description rule: Data we describe it should be in form of table.

(12) Distribution & Independence  $\therefore$  It should be platform independent.

Data Manipulation  $\therefore$

(1) Relational Algebra

(2) Relational calculus.

Q Difference b/w Relational Algebra & Relational Calculus.

Relational Algebra

(1) It is procedural language

In this we follow the procedure & step

Relational Calculus.

(1) It is a non procedural language

In this we automatically get output

(2) We can combine 2 or more table to get an another table

Both are non user friendly.

Relational Algebra  $\therefore J. A \oplus B$

Relational operation  $\therefore$  Type of Relational operator

(1) Traditional set operators

(2) Special operators

(1) Traditional Set Operators  $\therefore \rightarrow$  Union

$\rightarrow$  Intersection

$\rightarrow$  Difference

$\rightarrow$  Cartesian Product

Union  $\therefore$  Combine two table Employee Dep-id.

| Emp-id | Name | Dept-id | Depid | Name |
|--------|------|---------|-------|------|
|        |      |         |       |      |

→ Intersection: Common element

→ Difference: The Difference between two sets  $S_1$  &  $S_2$  produces a set, which contains all the members of one set, which are not in the other.

R

| Cust-name | Cust-status |
|-----------|-------------|
| Ram       | Good        |
| Shyam     | Excellent   |

| Cust-name | Cust-status |
|-----------|-------------|
| Karan     | Average     |
| Ram       | Good        |

① RUS

| Cust-name | Cust-status |
|-----------|-------------|
| Ram       | Good        |
| Shyam     | Excellent   |
| Karan     | Average     |

②

RNS

| Cust-name | Cust-status |
|-----------|-------------|
| Shyam     |             |
| Ram       | Good        |

③

$R - S \Rightarrow$  Shyam.

④

$R \times S = \{(R, S) : (R, S) \in R \text{ and } (S, R) \in S\}$

| Name | Rollno |
|------|--------|
| Ami  | 10     |
| Mym  | 8      |

| Name  | Roll |
|-------|------|
| Key   | 11   |
| Shonu | 12   |

$$R \times S = \{(A, B) : A \in R \text{ and } B \in S\}$$

ii)

Special operators

→ Selection

→ Projection

→ Join

→ Division

①

Selection operation: It yields a horizontal subset of a given relation that is the subset of row which should be selected within the given relation for which a particular

Condition is satisfied. Sign of Selection is ( $\sigma$ )

Employee :-

| Emp_id | Name  | Salary |
|--------|-------|--------|
| 1001   | John  | 10000  |
| 1002   | Mike  | 12000  |
| 1003   | David | 15000  |
| 1004   | Steve | 18000  |

Pooroint :-  $\sigma_{\text{salary} > 10,000}$  (Employee)

② Projection :- The Projection operation on a table, simply form another table by copying Specified columns from the original table. Symbol of projection is ( $\Pi$ )

To Select name of employee :-

$\Pi_{\text{name}} (\text{Employee})$

for salary :-

$\Pi_{\text{salary}} (\text{Employee})$

We want the name of all the employee having salary less than 70000

$\sigma_{\text{salary} < 70000} \Pi_{\text{name}} (\text{Employee})$

## Codd's Rules:

- ① **Information Rule:** All information in a relational database including tables names, column names is represented in the form of tables. This simple view of data speeds up design and learning process; user productivity is improved since knowledge of only one language is necessary to access all data such as description of the table and attribute definitions, integrity constraints and action are taken when constraint are violated. Access of data is restricted.
- ② **Guaranteed Access Rule:** Every piece of data in relational database can be accessed by using a primary key value that identifies the row name and column name. User productivity is improved since there is no need to resort to using physical pointers or address. It provides data independence or makes it possible to retrieve each piece of data in a relational database.
- ③ **Comprehensive Data Sub-language Rule:** The RDBMS may support several languages. But at least one of them should allow the user to do all following define table<sup>view</sup>, query & update data, set integrity constraints, set authorizations & define transaction. User productivity is improved since there is just one approach that can be used for all database operations.
- ④ **View Updating Rule:** Any view that can be updated theoretically can be updated using the RDBMS. Data consistency is ensured since the changes made in the view are

transmitted to base table and vice-versa.

- ⑤ High Level Insert, update & Delete: The RDBMS support insertion, updating and deletion at a table level. The performance is improved since the commands acts on a set of records rather than one record at a time.

- Application program is not affected by changes in the physical access or storage method. Database administrators can make changes to the physical access and storage method which improve performance and do not require changes in the application programs or request.

- ⑥ Logical Data independence: Logical changes in tables and views such as adding/deleting columns or changing field lengths need not necessitate modifications in the programs.

- ⑦ Integrity Independence: Integrity constraints are stored in the one-line catalog or data dictionary and can therefore be changed without necessitating changes in the application programs.

- ⑧ Non Subversion Rule: If the RDBMS has a language that accesses the information of records at a time, this language should not be used to bypass the integrity constraints. This is necessary for data integrity.

(10) Systematic Treatment of Null values: In RDBMS null values should be supported for the representation of missing information and in applicable information, the database management must have a consistent method for representing null values.

(11) Database Description Rule: The description of database is stored and maintained in the form of tables. It allows the user with appropriate authority to query information in a similar way or language.

(12) Distribution Independence: The RDBMS package must have distribution independence. Thus RDBMS

Packet must make it possible for the database to be distributed across multiple

Computers

### Relation Algebra

① It is a procedural method for solving Queries

② The solution to the database access problem using a relational algebra is obtained by stating what is required and what are the steps to obtain that information.

It is used as vehicle for implementation of Relational Calculus

### Relational Calculus

① It is non-procedural method of solving the Queries

The solution to the database access problem using a relational calculus is obtained simply by stating what is required and letting the system find the answer.

Relation Calculus Queries are converted into equivalent relational algebra format by using Codd's reduction algorithm and then it is implemen-

with the help of relational algebra operators.

- ④ Relational algebra Operators are used as a yardstick for measuring the expressive power of any given language.

A lang. is said to be Complete if it is at least powerful as the Calculus that is, if any relation definable by some expression of the calculus is also definable by some expression of the language in question.

## Difference :-

- ① • Relational Algebra is a procedural language that can be used to tell the DBMS how to build a new relation from one or more relations in the database.
- Relational Calculus is non-procedural language that can be used to formulate the definition of relation in terms of one or more database relations.
- ② Relation Algebra : user has to specify what is required and what are the procedure or steps to obtain output.
- Relational Calculus : user just specifies what is required and need not to specify how to obtain it.

### Strong Entity Set

- ① It has its own primary key
- ② It is represented by rectangle
- ③ It contains primary key represented by an Underline
- ④ The member of Strong entity set is called dominant entity set
- ⑤ The relationship between two strong entity set is represented by diamond System
- ⑥ The line connecting Strong entity set with the relationship is single
- ⑦ Total participation in the relationship may or may not exist

### weak Entity Set

It does not have sufficient attributes to form a primary key on its own.

It is represented by double rectangle.

It contains Partial key represented by dashed underline

The member of weak entity set is called Subordinate entity set

The relationship b/w one strong and ~~one~~ weak entity set is represented by a double diamond sign. i.e Identifying relationship

The line connecting weak entity set with identifying relationship is double

Total participation in the identifying relationship always exist

## Relational Algebra:

Joins: General form:  $\pi_{\text{selection}} \sigma_{\text{join condition}}$

$\hookrightarrow$  Theta Join.

Theta Joins: It is a Cartesian product operation on the two tables followed by a restriction operation on the resultant tables.

Eg:

| Employee Product | Name   | Product |
|------------------|--------|---------|
|                  | Raja   | Pen     |
|                  | Sparsh | Pen     |
|                  | Raja   | Pencil  |
|                  | Sparsh | Rubber  |

Product Customers

| C-Product | Customer |
|-----------|----------|
| Pen       | Karan    |
| Pen       | Suneet   |
| Pencil    | Suneet   |

$\exists$  Raja Pen

| Name | Product | Customer |
|------|---------|----------|
| Raja | Pen     | Karan    |

## Operations

$\hookrightarrow$  Comparison operator

$\rightarrow$  equal =

$\rightarrow$  Not equal to a

$\rightarrow$  Greater than >

|           | Name   | Product | C-Product     | Customer         |
|-----------|--------|---------|---------------|------------------|
| Equal     | Raja   | Pen     | Pen           | Karan            |
|           | "      | "       | Pen           | Suneet           |
| Not equal | "      | "       | Pencil        | Suneet           |
|           | Sparsh | Pen     | Pen           | Karan            |
|           | "      | "       | Pen<br>Pencil | Suneet<br>Suneet |

## Equi Joins :-

Natural Joins :- The projection operation which eliminates one of the duplicated column resulting from the equijoin, the natural join is obtained.

| Name  | Product | Customer |
|-------|---------|----------|
| Raju  | Pen     | Karan    |
| Sunil | "       | Suneet   |

## Inner Joins :-

→ Outer Joins :- If it requires stack the value exist in only one table must appear in the output then the solution is outer joins.

### Normal Product

These are of two type

- 1) Left outer joins
- 2) Right outer joins.

## Relational Calculus:-

Difference b/w Algebra and Relational calculus

### Relational calculus

- Tuple
- Domain

## Normalization :-

- 1) Functional Dependency: In Relation R, X & Y are the two subset are the set of attributes, Y is said to be function dependent on X if a given value of X uniquely determine the value of Y.
- It is denoted by  $X \rightarrow Y$  to  
 $\downarrow \quad \swarrow$   
 where X is  
 Called determinant  
 Y is called Determined

Employee

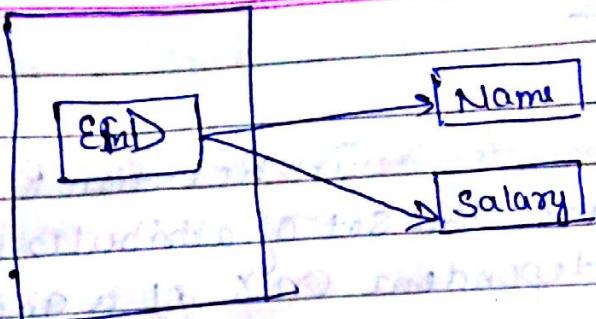
| EID | Name | Salary |
|-----|------|--------|
| 1   | XYZ  | 30,000 |
| 2   | Abc  | 20,000 |
| 3   | Pqr  | 10,000 |

$X(EID) \rightarrow Y(\text{Name}, \text{Salary})$

Functional Dependency chart: It is the graphical representation of functional dependencies among the attributes in any relation.

Steps:-

- (1) find out the primary key attribute
- (2) Make a rectangle with all the primary key attributes inside it
- (3) write all the non-primary key attribute outside the rectangle
- (4) use the arrows to show the functional dependencies among the attributes



→ Types of functional Dependency:

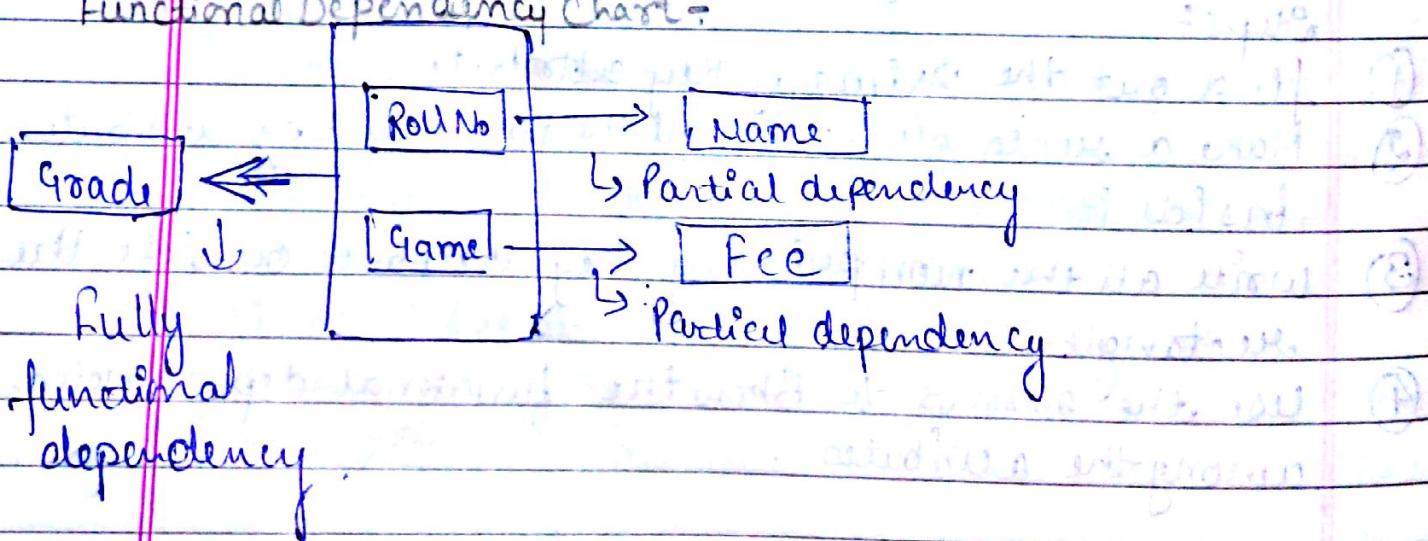
fully

### ① Partial Dependency:

- Partial Dependency: Suppose we have more than one attribute in primary key. Let A be the non primary key attributes attribute. If A is not dependent upon all the primary key attribute then partial dependency exists.

- Fully functional Dependency: Let A be the non primary key attribute if value of A is dependent upon all the primary key attributes then A is said to be fully functional dependency.

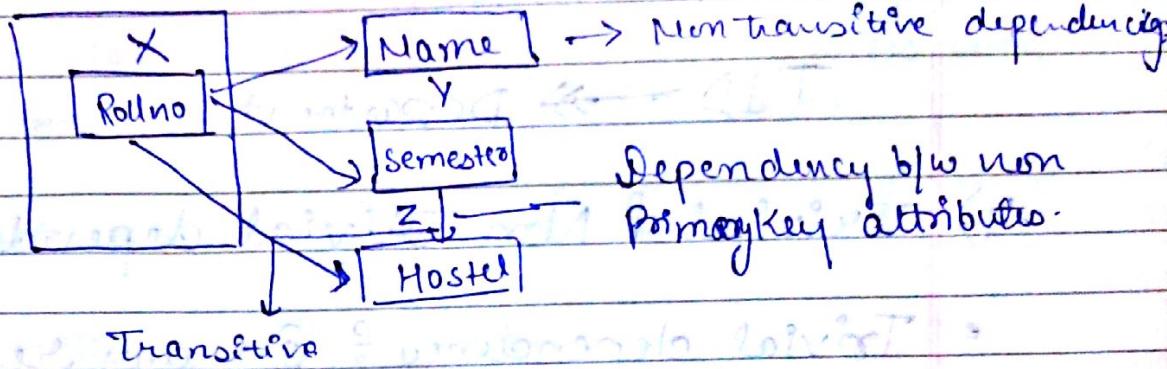
Functional Dependency Chart:



②

Transitive & Non-Transitive dependency :-

- Transitive dependency is due to dependency between the ~~known~~ primary key attributes.
- Suppose in a relation R,  $X \rightarrow Y$ ,  $Y \rightarrow Z$  then  $X \rightarrow Z$



- Non Transitive :- Any functional dependency which is not transitive is known as Non Transitive dependency.

NOTE :- Non transitive dependency exist if there is no dependency b/w the ~~known~~ primary key attributes

③

Single valued & multiple valued dependency :-

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

| Teacher ID | Name | Department |
|------------|------|------------|
| 1          | Abc  | CSE        |
| 1          | Abc  | ME         |
| 2          | Xyz  | Civil      |
| 2          | Xyz  | ECE        |
| 3          | Pqr  | E.C.E      |
| 3          | Pqr  | CSE        |

→ for single value

TeacherId → T Name

→ for multiple valued t

TID → Department

#### 4) Trivial & Non Trivial dependency

- Trivial dependency : In any relation R,  $X \rightarrow Y$  is trivial if  $Y \subseteq X$
  - Non Trivial dependency : In any relation R,  $X \rightarrow Y$  if  $Y \not\subseteq X$

Anomalies  $\hat{=}$  Undesired result

→ Inversion

→ Deletion

→ update

| EID | Name | Salary | DeptNo | Department |
|-----|------|--------|--------|------------|
| 4   | PQR  | 39,000 | 34     | Marketing  |
| 5   | ABC  | 20,000 | 3      | Marketing  |
|     |      |        |        |            |
|     |      |        |        |            |

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

Normal forms : It is a step by step process and each step is known as Normal form.

→ Properties of Normalisation :

- ① Remove different anomalies
- ② Decomposition must be lossless
- ③ Preserve the necessary dependency
- ④ Reduce Redundancy

In Normalisation there are 5 Normal form

First Normal Form :

- ↳ Composite attributes.
- ↳ Flattening the table
- ↳ Decomposition of table

| Teacher ID | Name | Subject                |
|------------|------|------------------------|
| 1          | XYZ  | ADA<br>DBMS<br>C, C++  |
| 2          | PQR  | Java, Automata<br>ADCE |

① Composite attribute

| Teacher ID | First Name | Last Name | Subject |
|------------|------------|-----------|---------|
|            |            |           |         |

(2)

## Flattening the table

| TeacherID | Name | Subject    |
|-----------|------|------------|
| 1         | XYZ  | ADA        |
| 1         | XYZ  | DBMS       |
| 1         | XYZ  | C/C++      |
| 1         | XYZ  | C++        |
| 2         | Pg1  | JAVA       |
| 2         | Pg1  | Automation |
| 2         | Pg1  | ADCP       |

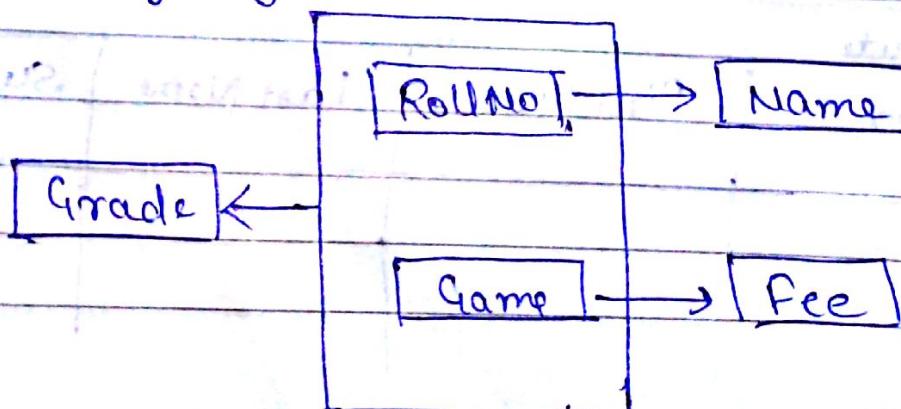
## (3) Decomposition of table :

| TeacherID | Name | Subject |
|-----------|------|---------|
|           |      |         |
|           |      |         |
|           |      |         |

A relation is in the first normal form if the domain of each attribute contains only atomic values it means atomicity must be present in the relation

(4)

Second Normal form : A relation is in the second normal form if it is in the first normal form and all the non-primary key attribute must be fully functionally dependent upon the primary key attribute



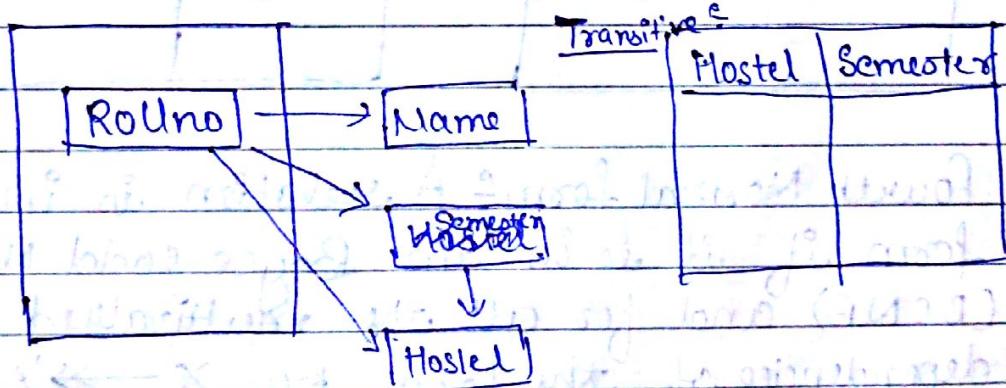
| Roll No | Name | Game       | Fee | Grade |
|---------|------|------------|-----|-------|
| 101     | John | Cricket    | 500 | A+    |
| 102     | Jane | Football   | 450 | B+    |
| 103     | Mike | Volleyball | 400 | C+    |

| Roll no | Name | Game       | Fee |
|---------|------|------------|-----|
| 101     | John | Cricket    | 500 |
| 102     | Jane | Football   | 450 |
| 103     | Mike | Volleyball | 400 |

| RollNo | Game       | Grade |
|--------|------------|-------|
| 101    | Cricket    | A+    |
| 102    | Football   | B+    |
| 103    | Volleyball | C+    |

III)

Third Normal form : A relation is in the third normal form if it is in the Second normal form and non primary key attributes must be non transitively dependent upon primary key attributes.



Nontransitive :

| Rollno | Name | Semester |
|--------|------|----------|
| 101    | John | 1st      |
| 102    | Jane | 2nd      |
| 103    | Mike | 3rd      |

BCNF (Boyce Codd Normal form) :-  
 A relation is in BCNF if and only if all the determinants are <sup>th</sup> candidate key

$X \rightarrow Y \rightarrow \text{determined}$   
 ↓  
 determinant

| Roll no. | Teacher | Subject |
|----------|---------|---------|
|          |         |         |
|          |         |         |
|          |         |         |

→ Roll no.  
 → (Rollno, Teacher)  
 → (Rollno, Subject)

| Rollno | TeacherID | TeacherID | Subject |
|--------|-----------|-----------|---------|
|        |           |           |         |
|        |           |           |         |
|        |           |           |         |

IV: Fourth Normal form: A relation is in fourth form if it is in the Boyce Codd Normal form (BCNF) and for all the multivalued functional dependencies of the form by  $X \rightarrow\!\!\! \rightarrow Y$

| Programmer | Project | Module |
|------------|---------|--------|
|            |         |        |
|            |         |        |
|            |         |        |

| Programmer | Project |
|------------|---------|
|            |         |
|            |         |
|            |         |

| Project | Module |
|---------|--------|
|         |        |
|         |        |
|         |        |

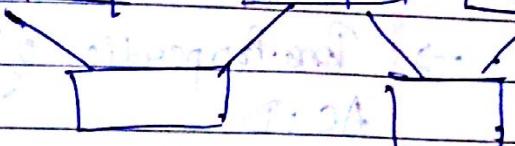
V

Fifth Normal Form (Join dependency):  
 Let  $R$  be the given relation upto fourth normal form or (4NF). Let  $R$  be and it decompose into  $(R_1, R_2, R_3, \dots, R_n)$ . The Relation  $R$  satisfy the join dependencies if and only if joining  $R_1$  to  $R_n = R$

| X     | Y     | Z     |
|-------|-------|-------|
| $x_1$ | $y_1$ | $z_1$ |
| $x_2$ | $y_2$ | $z_2$ |
| $x_3$ | $y_3$ | $z_3$ |

| X | Y | Z | X |
|---|---|---|---|
|   |   |   |   |
|   |   |   |   |
|   |   |   |   |

join dependency



join dependency

join dependency

join dependency

join dependency

all of the above shown are join dependency

foot at 2 position

left hand side is called as FDI & right hand side is called as FDII

left hand side is called as FDI & right hand side is called as FDII

## Transaction & concurrency Control

- Transaction: It can be defined as a unit or any part of any program at the time of its execution. During transaction the data items can be read or updated or both.
- Concurrency Control: It is the activity of co-ordinating the action of transactions, that operate simultaneously.

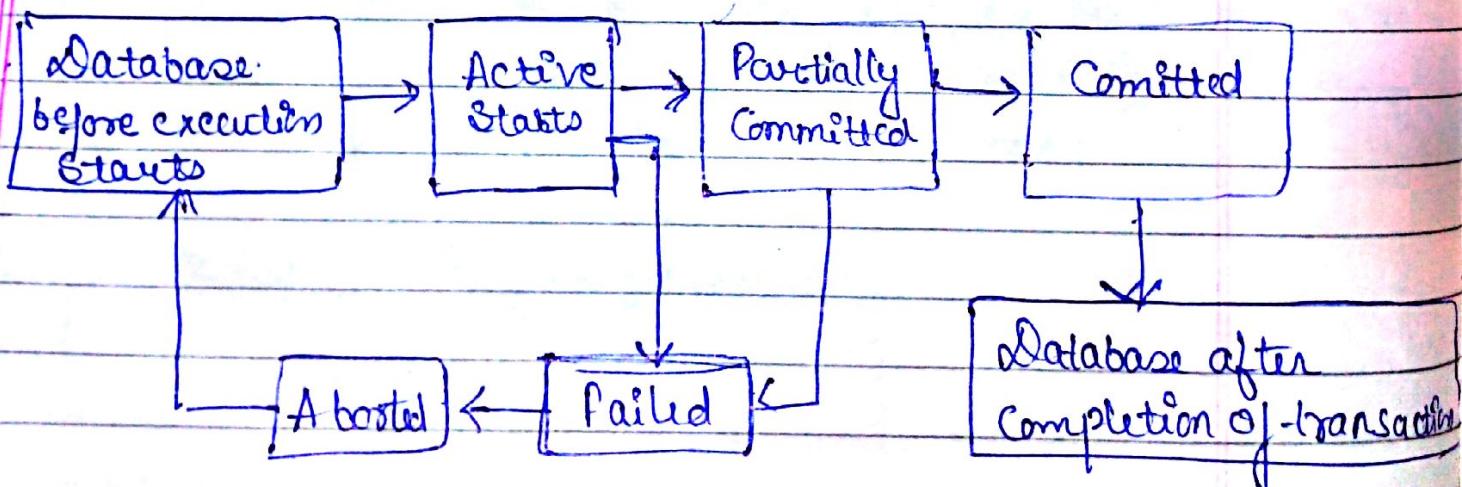
### → Four Properties of Transactions:

ACID

- ↳ Atomicity
- ↳ Consistency
- ↳ Isolation
- ↳ Durability

- Durability: Changes are made permanent to the database after successful completion of the transaction.

### → Transaction States:



- Serializability: Considered the set of transactions  $T_1, T_2, T_3 \dots T_i$ .  $S_1$  is the state of database after they are concurrently executed and successfully completed and  $S_2$  is the state of database after they are executed in any serial manner and successfully completed. If  $S_1$  and  $S_2$  are same then the database maintains the serializability.
- Concurrent Execution: If more than one transaction are executed at the same time then they are said to be executed Concurrently.
- Recoverability: To maintain the Atomicity of the database, undo effects of any transaction has to be performed in case of failure of that transaction. If undo effects is successful then that database maintains the recoverability. This process is also known as roll back.

→ Construct an ER diagram for Banking System

Construct an ER diagram for Hospital Management System

## Hospital management

