Unit - 2

Relational Algebra

→ Set of operations selection, projection, union & join.

→ Provides mathematical framework for querying db, ensuring efficient data retrieval and manipulation.

→ It simplifies, optimize query execution for better performance.

Key concepts of Relational Algebra :

1. Relations
2. Tuples
3. Attributes
4. Domains

Relation – A relation is a table that contains rows and columns, representing data in structured form. It has unique name and each made of tuples.

Tuples – A tuple is a single row in a relation which contains set of values of each attribute.

Attributes – These are columns in relations, each representing a specific characteristics or property of the data

Ex. In student, attribute could be "Name", "age" and "grade"

Domains – A set of values that can be possibly contain by attributes.
It defines which type of data that can be stored in each column of a relation, such as integer, string, date.

~~Operations~~

Operators in Relational Algebra

Basic Operators
- Selection (σ)
- set difference (−)
- Rename (ρ)
- cartesian product (×)
- Project (π)
- Union (∪)

Derived operators
- Intersection (∩)
- Division (÷)
- Join
  - Outer join
  - Inner join

→ **Selection : (σ)** It basically filter out rows from a given table based on certain given condition. It doesn't change order of table.

Ex. relation R with attributes A, B, C

A	B	C
1	2	4
2	2	3
3	2	3
4	3	4

find $\sigma_{(C>3)}$ =

A	B	C
1	2	4
4	3	4

Projection (π) – It basically works on column. It basically allows us to pick specific columns from a given relational table based on the given condition and ignoring all other remaining columns.

$\pi_{(B,C)}(R)$

B	C
2	4
2	3
3	4

→ By default projection operation removes duplicate values.

Union (U) – It used to combine the result of two queries into a single result.

→ The only condition is that both queries must return same number of columns with same datatypes.

French

Name	R.No
Ram	01
Shyam	02

Germany

Name	R.No
Vivek	08
Mohan	09

$\pi_{(Name)}(French) \cup \pi_{(name)}(Germany)$

→

Name
Ram
Shyam
Vivek
Mohan

It removes duplicate values

Set difference (−) – It provides that are present in one table, but not in another table.

$\pi_{(name)}(FRENCH) - \pi_{(name)}(GERMAN)$

A.col1 = {1, 2, 3, 5}

B.col1 = {2, 3, 4}

Set diff

A − B = elements in A but not in B.

B − A = elements in B but not in A

A − B

From A = {1, 2, 3, 5}

B = {2, 3, 4}

Result = {1, 5}

Rename (ρ) – Rename operator basically allows you to give a temporary name to a specific relational table or to its columns.

Ex.

A	B	C
1	2	4
2	2	3
3	1	2

$\rho_{(D/B)}R$ will rename the attribute B to D.

A	D	C
1	2	4
2	2	3
3	1	2

Cartesian Product – It combines every row of one table with every row of another table, producing all possible combinations.

Relational A:

Name	Age	Sex
Ram	14	M
Sona	15	F
Kim	20	M

Relation B:

Id	Course
1	DS
2	DBMS

Cartisian product A x B

Name	Age	sex	Id	Course
Ram	14	M	1	DS
Ram	14	M	2	DBMS
Sona	15	F	1	DS
Sona	15	F	2	DBMS
Kim	20	M	1	DS
Kim	20	M	2	DBMS

If row A = 3, B = 2, X = A x B = 3 x 2 = 6

Derived operators in Relational Algebra

1. Join operator : Used to combine two relation

Inner join – It only returns matching row. If didn't match, row will be excluded

conditional join – based on condition (>, =, < etc)

Equi join – only equality (=) join

Natural join = Automatically join on same attributes and exclude duplicate columns.

Outer join – It preserves every row, it match or did not match.

→ Left outer join – All rows from the left table + matching rows from the right table) if no match, returns null on right side.

→ Right outer join – All rows from the right table + matching rows from the left table If no match, returns null on left side.

→ Full outer join – All rows from both tables (if no match – nulls on the missing side).

Examples

Inner join – Employees + department (only employees that belong to a department)

Left join – All employees are shown, if no department then null.

right join – All department are shown, if no employees then null

Full join – All department & employees are shown, missing side filled with Null.

Armstrong Axioms Properties

→ These are set of interface rules developed by William Armstrong in 1974 to reason about functional dependencies.

→ This interfrence rule are used to derive or infer all functional dependencies on relational database.

Reflexibity: If $x \rightarrow y$ and $x \rightarrow x$ then y is subset of x.

Augmentation: if $x \rightarrow y$ then $xa \rightarrow ya$

Transitivity: If $x \rightarrow y$ & $y \rightarrow z$ then $x \rightarrow z$

Decomposition: if $x \rightarrow yz$ then $x \rightarrow y$ and $x \rightarrow z$

Union Mode: if $x \rightarrow y$ and $x \rightarrow z$ then $x \rightarrow yz$

Pseudotransitivity: If $x \rightarrow y$ and $yz \rightarrow a$ then $xz \rightarrow a$

Composition Rule: If $x \rightarrow y$ and $a \rightarrow b$ then $xa \rightarrow yb$

→ **Functional Dependency**: It determines the relation of one attribute to another attribute in DBMS.

→ It helps us to maintain the quality of data in database.

→ Functional dependency which is also known as non-trivial dependency occur when $A \in B$ hold true where $B \notin A$.

Multivalue Dependency: It occurs in the situation where there are multiple independent, multivalued attribute in a single table.

→ It is a complete constraint b/w two sets of attribute in a relation it require that certain tupple be present in relation.

Ex. Car, Model, color, man. year

(color, man. year — it doesn't define)

Decomposition – It is a process of breaking up or dividing single relation in two or more sub relations is called decomposition of relation

A	B	C
1	2	1
2	5	3
3	3	3

R1 →

A	B
1	2
2	5
3	3

R2 →

B	C
2	1
5	3
3	3

Normalization : It is the process of organizing the attributes of the database to reduce or eliminate data redundancy (having the same data but at different places).

1. Data redudancy
2. Data anomalies
3. Maintain consistency and integrity

Normal Forms :- 1) 1NF (First normal form)
2) 2NF (second normal form)
3) 3NF (Third normal form)
4) BCNF (Boyce codd normal form)

1NF :

→ In this type of normal form, multivalued attribute is not allowed.

For example:

S.Id	Name	Course
1	Shivam	C/ C++
2	Mohan	Java
3	Sohan	Javascript

→ It is not allowed

To fix we can do in two ways:

S.Id	Name	Course
1	Shivam	C
1	Shivam	C++
2	Mohan	Java
3	Sohan	Javascript

S.Id	Name	Course1	Course2
1	Shivam	C	C++
2	Mohan	Java	null
3	Sohan	Javascript	null

2NF :

→ It is already in 1NF.

→ No partial dependency exist

For example:

S.Id	Course Id	Student name	Course Name
1	DBMS	Shivam	Databases
2	OS	Mohan	operating systems
1	OS	Shivam	operating systems

After Normalization :

S.Id	Student Name
1	Shivam
2	Mohan
1	Shivam

Course Id	Course Name
DBMS	Database
OS	Operating System
OS	Operating System

S.Id	Course Id
1	DBMS
2	OS
1	OS

3NF :

→ The rule for 3NF is that first should be in 2NF.

→ No transitive dependency

For example :

S.Id	Student Name	Dept Id	DeptName
1	Ravi	10	computer science
2	Sita	20	mechanical
3	Amit	10	computer science

Transitive dependency here is :

S.Id ⟶ DeptId ⟶ DeptName

3NF Conversion :

S.Id	Student Name	DeptId
1	Ravi	10
2	Sita	20
3	Amit	30

DeptId	DeptName
10	computer science
20	mechanical
10	computer science

This we can implement 3NF.

BCNF

→ It should be in 3NF.

→ Every determinant should be candidate key.

For example

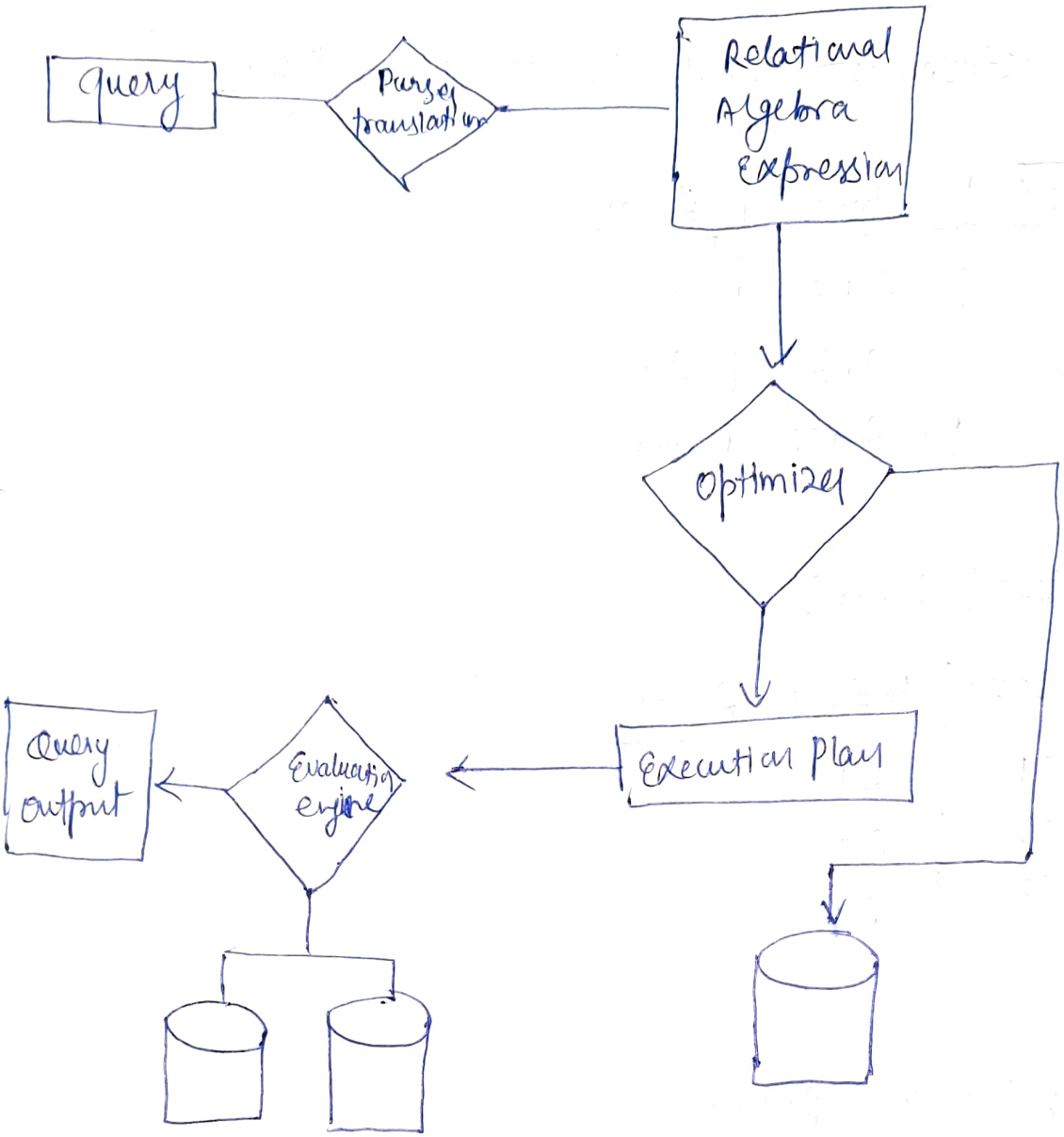
Couse Id	Instructor	Room
C1	Alice	R1
C2	Bob	R2
C3	Alice	R1

Normalized Form :

Couse Id	~~Instructor~~ Room
C1	R1
C2	R2
C3	R1

Instructor	Room
Alice	R1
Bob	R2
Alice	R1

Query Process & Optimization :



→ Query processing – It is a procedure of converting a query written in high level language like SQL into a correct and efficient execution plan expressed in low level language which is used for data manipulation.