Unit 3

Relational Model and relational Algebra

Content

Introduction to the Relational Model
Relational schema and concept of keys
Mapping the ER and EER Model to the Relational Model
Relational Algebra operators
Relational Algebra Queries

Introduction to the Relational Model

Relational data model is the primary data model, which is used widely around the world for data storage and processing.

This model is simple and it has all the properties and capabilities required to process data with storage efficiency.

Relational Model (RM) represents the database as a collection of relations.

A relation is nothing but a table of values.

Every row in the table represents a collection of related data values.

INFORMAL DEFINITIONS

- ## RELATION: A table of values
  - A relation may be thought of as a **set of rows.**
  - A relation may alternately be though of as a **set of columns**.
  - Each row represents a fact that corresponds to a real-world **entity** or **relationship**.
  - Each row has a value of an item or set of items that uniquely identifies that row in the table.
  - Each column typically is called by its column name or column header or attribute name.

- A **Relation** may be defined in multiple ways.
- The **Schema** of a Relation: *R* (A1, A2, ...An)
  Relation schema *R* is defined over **attributes** A1, A2, ...An
  For Example -
  CUSTOMER (Cust-id, Cust-name, Address, Phone#)

  Here, CUSTOMER is a relation defined over the four attributes Cust-id, Cust-name, Address, Phone#, each of which has a **domain** or a set of valid values. For example, the domain of Cust-id is 6 digit numbers.

- A **tuple** is an ordered set of values
- Each value is derived from an appropriate domain.
- Each row in the CUSTOMER table may be referred to as a tuple in the table and would consist of four values.

<101, “Anil", “Mumbai", “9585485858">

is a tuple belonging to the CUSTOMER relation.

- A relation may be regarded as a ***set of tuples*** (rows).
- Columns in a table are also called attributes of the relation.

- A **domain** has a logical definition: e.g., “mobile_numbers” are the set of 10 digit mobile numbers.

- A domain may have a data-type or a format defined for it. E.g., Dates have various formats such as monthname, date, year or yyyy-mm-dd, or dd mm,yyyy etc.

DEFINITION SUMMARY

Relational Model is made up of tables

- A row of table = a relational instance/tuple
- A column of table = an attribute
- A table = a schema/relation
- Cardinality = number of rows
- Degree = number of columns
- Domain = Values in a column
- Table Definition =Schema of a Relation

Concepts

Tables − In relational data model, relations are saved in the format of Tables.

A table has rows and columns, where rows represents records and columns represent the attributes.

Tuple − A single row of a table, which contains a single record for that relation is called a tuple.

Relation instance − A finite set of tuples in the relational database system represents relation instance. Relation instances do not have duplicate tuples.

Relation schema − A relation schema describes the relation name (table name), attributes, and their names.

Relation key − Each row has one or more attributes, known as relation key, which can identify the row in the relation (table) uniquely.

Relation name

Attributes

Tuples

STUDENT	Name	SSN	HomePhone	Address	OfficePhone	Age	GPA
	Benjamin Bayer	305-61-2435	373-1616	2918 Bluebonnet Lane	null	19	3.21
	Katherine Ashly	381-62-1245	375-4409	125 Kirby Road	null	18	2.89
	Dick Davidson	422-11-2320	null	3452 Elgin Road	749-1253	25	3.53
	Charles Cooper	489-22-1100	376-9821	265 Lark Lane	749-6492	28	3.93
	Barbara Benson	533-69-1238	839-8461	7384 Fontana Lane	null	19	3.25

Attribute

tuple/relational instance

SID	Name	Major	GPA
1234	John	CS	2.8
5678	Mary	EE	3.6

Cardinality = 2

4 Degree

A Schema / Relation

- **Example: STUDENT Relation**

NAME	ROLL_NO	PHONE_NO	ADDRESS	AGE
Ram	14795	7305758992	Noida	24
Shyam	12839	9026288936	Delhi	35
Laxman	33289	8583287182	Gurugram	20
Mahesh	27857	7086819134	Ghaziabad	27
Ganesh	17282	9028 9i3988	Delhi	40

In the given table, NAME, ROLL_NO, PHONE_NO, ADDRESS, and AGE are the attributes.
Relational instances: The instance of schema STUDENT has 5 tuples.
Tuple: t3=<Laxman, 33289, 8583287182, Gurugram, 20>
Relation schema: Student (Name,roll_no,phone_no,address,age)

Properties of the relational database model

- Data is presented as a collection of relations.
- Columns are attributes that belong to the entity modeled by the table

(ex. In a student table, you could have name, address, student ID, major, etc.).

- Each row ("tuple") represents a single entity or an instance of that particular entity

(ex. In a student table, John Smith, 14 Oak St, 9002342, Accounting, would represent one student entity).

- Every table has a set of attributes that taken together as a "key" (technically, a "superkey") uniquely identifies each entity

(Ex. In the student table, “student ID” would uniquely identify each student – no two students would have the same student ID).

- **Rules**

1. The order of tuples and attributes is not important. (Ex. Attribute order not important…if you have name before address, is the same as address before name).
2. Every tuple is unique. This means that for every record in a table there is something that uniquely identifies it from any other tuple.
3. Cells contain single values. This means that each cell in a table can contain only one value.
4. All values within an attribute are from the same domain. This means that however the attribute is defined, the values for each tuple fall into that definition. For example, if the attribute is labeled as Date, you would not enter a dollar amount, shirt size, or model number in that column, only dates.
5. Table names in the database must be unique and attribute names in tables must be unique. No two tables can have the same name in a database. Attributes (columns) cannot have the same name in a table. You can have two different tables that have similar attribute names.

Operations in Relational Model

Four basic update operations performed on relational database model are Insert, update, delete and select.

- Insert is used to insert data into the relation
- Delete is used to delete tuples from the table.
- Modify allows you to change the values of some attributes in existing tuples.
- Select allows you to choose a specific range of data.

Update Operation

You can see that in the below-given relation table CustomerName= ‘Apple’ is updated from Inactive to Active.

CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
3	Apple	Inactive
4	Alibaba	Active

CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
3	Apple	Active
4	Alibaba	Active

Delete Operation

To specify deletion, a condition on the attributes of the relation selects the tuple to be deleted.

CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
3	Apple	Active
4	Alibaba	Active

CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
4	Alibaba	Active

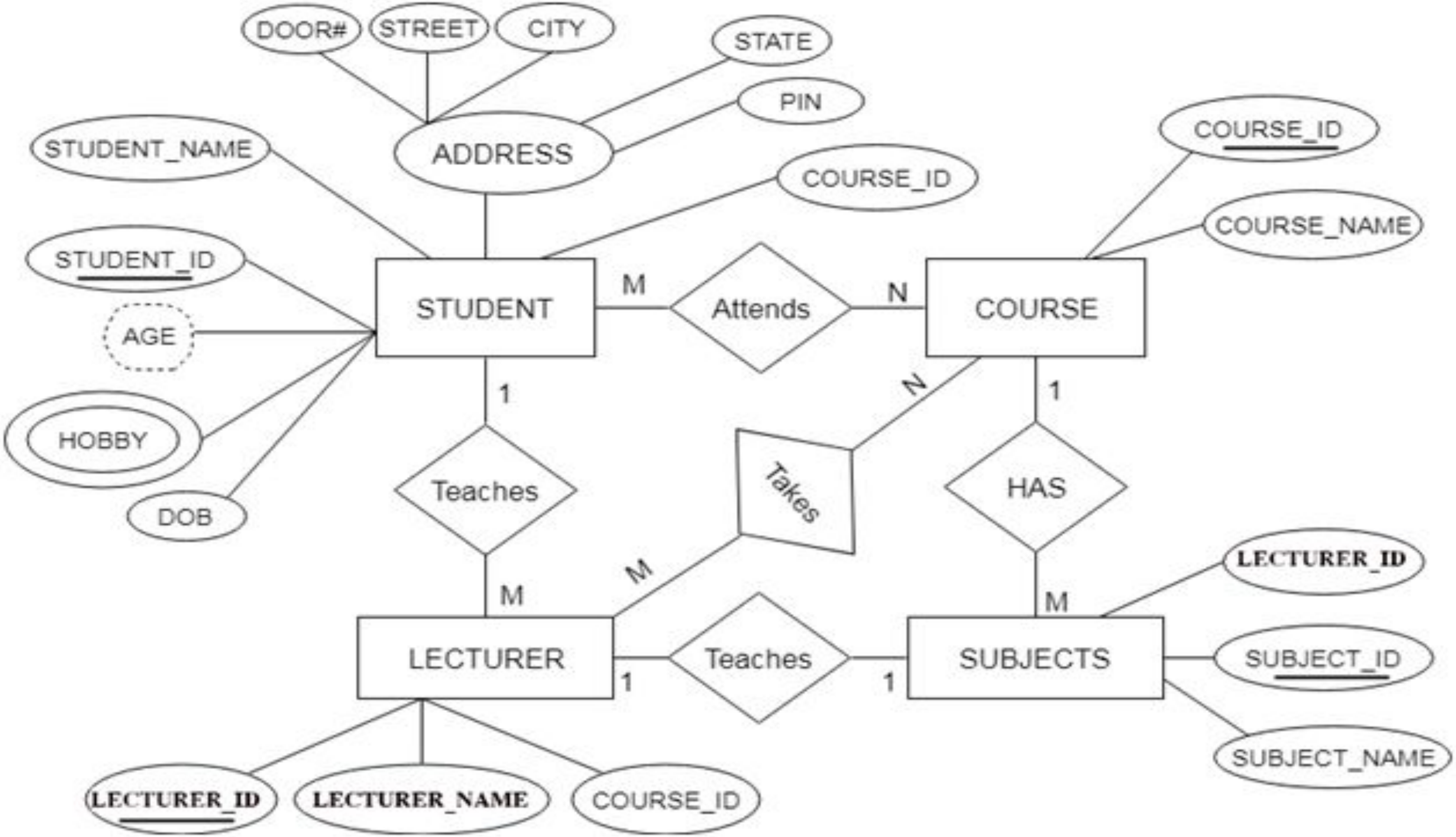
Select Operation

CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
4	Alibaba	Active

SELECT

CustomerID	CustomerName	Status
2	Amazon	Active

Mapping ER/ERR with relational model


DOOR#
STREET
CITY
STATE
PIN
ADDRESS
STUDENT_NAME
COURSE_ID
COURSE_ID
COURSE_NAME
STUDENT_ID
AGE
HOBBY
DOB
STUDENT
M
Attends
N
COURSE
1
N
1
Teaches
Takes
HAS
M
M
M
LECTURER_ID
LECTURER
Teaches
SUBJECTS
1
1
SUBJECT_ID
SUBJECT_NAME
LECTURER_ID
LECTURER_NAME
COURSE_ID

- **Entity type becomes a table.**

In the given ER diagram, LECTURE, STUDENT, SUBJECT and COURSE forms individual tables.

- **All single-valued attribute becomes a column for the table.**

In the STUDENT entity, STUDENT_NAME and STUDENT_ID form the column of STUDENT table. Similarly, COURSE_NAME and COURSE_ID form the column of COURSE table and so on.

- **A key attribute of the entity type represented by the primary key.**

In the given ER diagram, COURSE_ID, STUDENT_ID, SUBJECT_ID, and LECTURE_ID are the key attribute of the entity.

- **The multivalued attribute is represented by a separate table.**

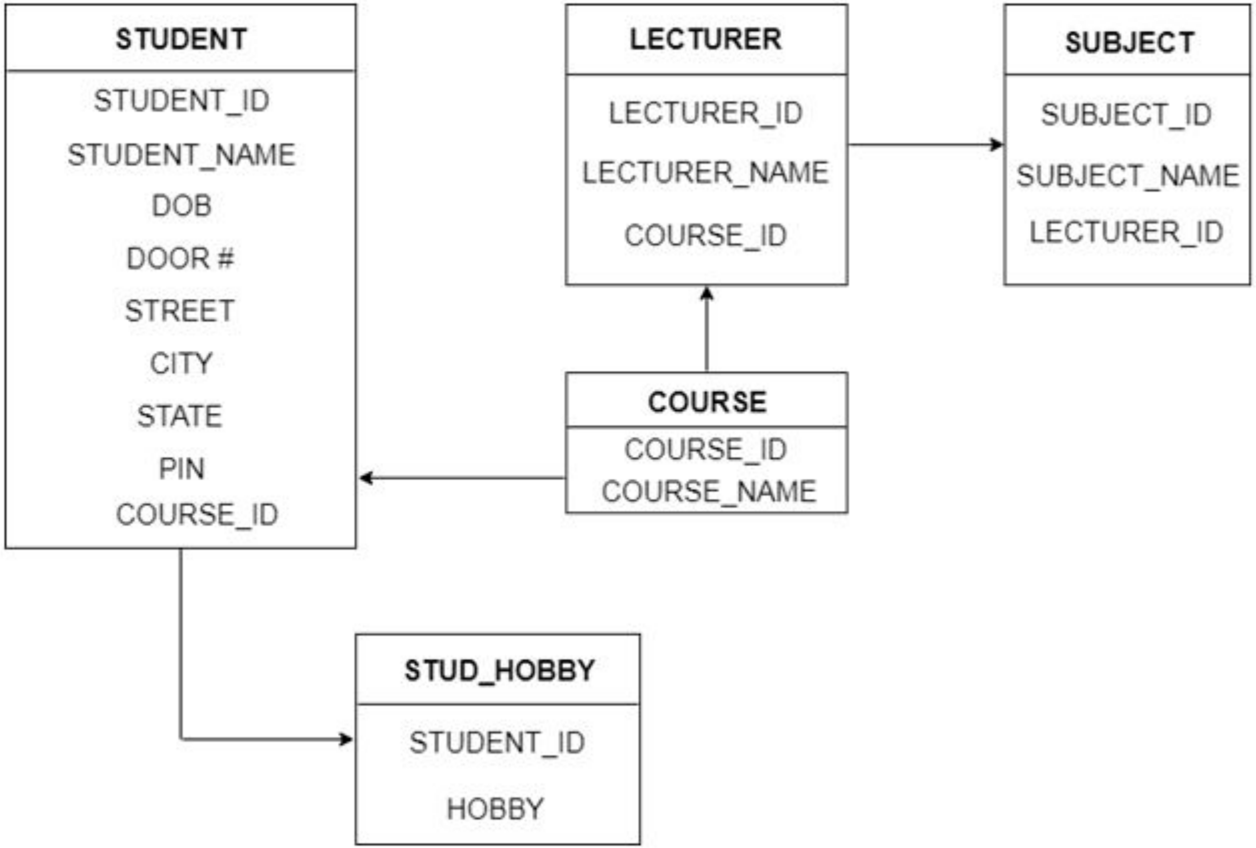
In the student table, a hobby is a multivalued attribute. So it is not possible to represent multiple values in a single column of STUDENT table. Hence we create a table STUD_HOBBY with column name STUDENT_ID and HOBBY. Using both the column, we create a composite key.

- **Composite attribute represented by components.**

In the given ER diagram, student address is a composite attribute. It contains CITY, PIN, DOOR, STREET, and STATE. In the STUDENT table, these attributes can consider as an individual column.

- **Derived attributes are not considered in the table.**

In the STUDENT table, Age is the derived attribute. It can be calculated at any point of time by calculating the difference between current date and Date of Birth.


STUDENT
STUDENT_ID
STUDENT_NAME
DOB
DOOR #
STREET
CITY
STATE
PIN
COURSE_ID
LECTURER
LECTURER_ID
LECTURER_NAME
COURSE_ID
SUBJECT
SUBJECT_ID
SUBJECT_NAME
LECTURER_ID
COURSE
COURSE_ID
COURSE_NAME
STUD_HOBBY
STUDENT_ID
HOBBY

Relational Algebra-operators

- **RELATIONAL ALGEBRA** is a widely used procedural query language.(it tells what data to be retrieved and how to be retrieved.)
- The purpose of a query language is to retrieve data from database or perform various operations such as insert, update, delete on the data.
- It uses various operations to perform action.
- SQL Relational algebra query operations are performed recursively on a relation.
- The output of these operations is a new relation, which might be formed from one or more input relations.

The fundamental operations of relational algebra are as follows −

- Select
- Project
- Union
- Set difference
- Cartesian product
- Rename

- **Unary Relational Operations**

1. SELECT (symbol: σ)
2. PROJECT (symbol: π)
3. RENAME (symbol: ρ)

- **Relational Algebra Operations From Set Theory**

1. UNION (υ)
2. INTERSECTION ( ),
3. DIFFERENCE (-)
4. CARTESIAN PRODUCT (x)

- **Binary Relational Operations**

1. JOIN

SELECT (σ)

- The SELECT operation is used for selecting a subset of the tuples according to a given selection condition.
- Sigma(σ)Symbol denotes it.
- It is used as an expression to choose tuples which meet the selection condition.
- Select operator selects tuples that satisfy a given predicate.

Notation

$\sigma_p(r)$

σ is the predicate

r stands for relation which is the name of the table

p is prepositional logic formula which may use connectors like: AND OR and NOT.

These relational can use as relational operators like =, ≠, ≥, <, >, ≤.

- **Example 1**

$$\sigma_{\text{topic = "Database"}}(\text{Tutorials})$$

Output –

Selects tuples from Tutorials where topic = ‘Database’.

- **Example 2**

$$\sigma_{\text{topic = "Database" and author = "navathe"}}(\text{Tutorials})$$

- **Output** –

Selects tuples from Tutorials where the topic is ‘Database’ and ‘author’ is navathe.

- **Example 3**

$\sigma_{\text{sales} > 50000}$ (Customers)

- **Output** –

Selects tuples from Customers where sales is greater than 50000

Q.1 Select tuples from a relation “Books” where subject is “database”

Q.2 Select tuples from a relation “Books” where subject is “database” and price is “450”

Q.3 Select tuples from a relation “Books” where subject is “database” and price is “450” or have a publication year after 2010

A.1 $\sigma_{\text{subject = “database”}}$ (Books)

- A.2 $\sigma_{\text{subject} = \text{“database”} \land \text{price} = \text{“450”}}$ (Books)

- $\sigma_{\text{subject = “database”} \wedge \text{price = “450”} \vee \text{year >”2010″}}$ (Books)

Projection(π)

- It projects column(s) that satisfy a given predicate.
- it is used to select desired columns (or attributes) from a table (or relation).
- It is denoted by ∏

- Notation −

$\prod_{A1, A2, An} (r)$

Where A_1, A_2 , A_n are attribute names of relation **r**.

- **example** –

$\Pi_{\text{subject, author}}$ (Books)

Output :

Selects and projects columns named as subject and author from the relation Books.

- Example 2:

Consider the following table

CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
3	Apple	Inactive
4	Alibaba	Active

- The projection of CustomerName and status will give

$$\Pi_{\text{CustomerName, Status}}(\text{Customers})$$

CustomerName	Status
Google	Active
Amazon	Active
Apple	Inactive
Alibaba	Active

Rename (ρ)

- Rename is a unary operation used for renaming attributes of a relation.
- It is used to assign a new name to a relation

- **Notation:**

$$\rho_X(R)$$

where the symbol 'ρ' is used to denote the RENAME operator and R is expression which is saved with the name X.

- Example 1

The student table is renamed with newstudent with the help of the following command −

ρnewstudent (student)

- **Example-2:**

Query to rename the attributes Name, Age of table Department to A,B.

$\rho_{(A,\ B)}$ **(Department)**

- **Example-3:**

Query to rename the first attribute of the table Student with attributes A, B, C to P.

$\rho_{(P,\ B,\ C)}$ **(Student)**

Union Operator (∪)

- Union operator is denoted by ∪ symbol and it is used to select all the rows (tuples) from two tables (relations).
- It also eliminates duplicate tuples.

Syntax of Union Operator (∪)

table_name1 ∪ table_name2

Example

Consider the following tables.

Table A		Table B	
column 1	column 2	column 1	column 2
1	1	1	1
1	2	1	3

A ∪ B
gives

Table A ∪ B	
column 1	column 2
1	1
1	2
1	3

- Table 1: COURSE

Course_Id	Student_Name	Student_Id
C101	Aditya	S901
C104	Aditya	S901
C106	Steve	S911
C109	Paul	S921
C115	Lucy	S931

Table 2: STUDENT

Student_Id	Student_Name	Student_Age
S901	Aditya	19
S911	Steve	18
S921	Paul	19
S931	Lucy	17
S941	Carl	16
S951	Rick	18

Query:

$\prod$ Student_Name (COURSE) $\cup$ $\prod$ Student_Name (STUDENT)

- Output:

Student_Name
Aditya
Carl
Paul
Lucy
Rick
Steve

Intersection Operator (∩)

- Intersection operator is denoted by ∩ symbol and it is used to select common rows (tuples) from two tables (relations).

- **Syntax of Intersection Operator (∩)**

  table_name1 ∩ table_name2

- Table 1: COURSE

Course_Id	Student_Name	Student_Id
C101	Aditya	S901
C104	Aditya	S901
C106	Steve	S911
C109	Paul	S921
C115	Lucy	S931

Table 2: STUDENT

Student_Id	Student_Name	Student_Age
S901	Aditya	19
S911	Steve	18
S921	Paul	19
S931	Lucy	17
S941	Carl	16
S951	Rick	18

Query:

∏ Student_Name (COURSE) ∩ ∏ Student_Name (STUDENT)

Output:

Student_Name
Aditya
Steve
Paul
Lucy

Set Difference (-)

- Set Difference is denoted by – symbol.
- Suppose we have two relations R1 and R2 and we want to select all those tuples(rows) that are present in Relation R1 but **not** present in Relation R2, this can be done using Set difference R1 – R2.

Syntax of Set Difference (-)

table_name1 - table_name2

Set Difference (-) Example

- write a query to select those student names that are present in STUDENT table but not present in COURSE table.
- ∏ Student_Name (STUDENT) - ∏ Student_Name (COURSE)

Output:

Student_Name
Carl
Rick

Cartesian product (X)

- Cartesian Product is denoted by X symbol.
- Lets say we have two relations R1 and R2 then the cartesian product of these two relations (R1 X R2) would combine each tuple of first relation R1 with the each tuple of second relation R2.

Syntax of Cartesian product (X)

R1 X R2

- Cartesian product (X) Example

Table 1: R

Col_A	Col_B
AA	100
BB	200
CC	300

Table 2: S

Col_X	Col_Y
XX	99
YY	11
ZZ	101

Lets find the cartesian product of table R and S.

R X S

Output:

Col_A	Col_B	Col_X	Col_Y
-----	------	------	------
AA	100	XX	99
AA	100	YY	11
AA	100	ZZ	101
BB	200	XX	99
BB	200	YY	11
BB	200	ZZ	101
CC	300	XX	99
CC	300	YY	11
CC	300	ZZ	101

Join Operations

- Join operation is essentially a cartesian product followed by a selection criterion.
- Join operation denoted by ⋈.
- JOIN operation also allows joining variously related tuples from different relations.

- **Types of JOIN:**

Various forms of join operation are:

- Inner Joins:

1. Natural join

- Outer join:

1. Left Outer Join
2. Right Outer Join
3. Full Outer Join

Inner Join:

- In an inner join, only those tuples that satisfy the matching criteria are included, while the rest are excluded.

1]NATURAL JOIN (⋈)

Natural join can only be performed if there is a common attribute (column) between the relations.
The name and type of the attribute must be same.

Example

Consider the following two tables

C	
Num	Square
2	4
3	9

D	
Num	Cube
2	8
3	27

$C \bowtie D$

$C \bowtie D$		
Num	Square	Cube
2	4	8
3	9	27

- **OUTER JOIN**
- In an outer join, along with tuples that satisfy the matching criteria, we also include some or all tuples that do not match the criteria.

Left Outer Join(A ⟕ B)

In the left outer join, operation allows keeping all tuple in the left relation. However, if there is no matching tuple is found in right relation, then the attributes of right relation in the join result are filled with null values.

A	
Num	Square
2	4
3	9
4	16

B	
Num	Cube
2	8
3	18
5	75

A ⟕ B

A ⋈ B		
Num	Square	Cube
2	4	8
3	9	18
4	16	–

Right Outer Join: (A ⟖ B)

In the right outer join, operation allows keeping all tuple in the right relation. However, if there is no matching tuple is found in the left relation, then the attributes of the left relation in the join result are filled with null values.

A ⋈ B

A ⋈ B		
Num	Cube	Square
2	8	4
3	18	9
5	75	–

Full Outer Join: (A ⟗ B)

In a full outer join, all tuples from both relations are included in the result, irrespective of the matching condition.

A ⟗ B		
Num	Cube	Square
2	4	8
3	9	18
4	16	–
5	–	75

Example : Relational algebra Queries

- **Question:**

Consider the following relational database schema consisting of the four relation schemas:

- **passenger** (pid, pname, pgender, pcity)
- **agency** (aid, aname, acity)
- **flight** (fid, fdate, time, src, dest)
- **booking** (pid, aid, fid, fdate)

Answer questions using relational algebra queries;

a) Get the complete details of all flights to New Delhi.
b) Get the details about all flights from Chennai and New Delhi.
c) Find only the flight numbers for passenger with pid 123 for flights to Chennai before 06/11/2020
d) Find the passenger names for passengers who have bookings on at least one flight.
e)Find the passenger names for those who do not have any bookings in any flights.
f) Get the details of flights that are scheduled on both dates 01/12/2020 and 02/12/2020 at 16:00 hours.

DBMS - Prachiti Pimple

Solution:

Relational algebra operators:

σ – selection with conditions (It selects all tuples that satisfies the conditions. Shows entire table with respect to the structure)

Π – projection operator (It selects the attributes which are listed here)

⋈ - natural join operator (Binary operator that join two relations on common attributes' values)

-, ∪, and ∩ - set operators (difference, union and intersection)

DBMS- –Prachiti Pimple

a) Get the complete details of all flights to New Delhi.

$$\sigma_{\text{destination} = \text{“New Delhi”}}(\text{flight})$$

b) Get the details about all flights from Chennai and New Delhi.

$$\sigma_{\text{src} = \text{“Chennai”} \wedge \text{dest} = \text{“New Delhi”}}(\text{flight})$$

DBMS - Prachiti Pimple

c) Find only the flight numbers for passenger with pid 123 for flights to Chennai before 06/11/2020.

$$\Pi_{fid}(\sigma_{pid = 123}(\text{booking}) \bowtie \sigma_{dest = \text{“Chennai”} \wedge fdate < 06/11/2020}(\text{flight}))$$

[**<u>Hint:</u>** ***Given conditions are pid, dest, and fdate. To get the flight id for a passenger given a pid, we have two tables flight and booking to be joined with necessary conditions. From the result, the flight id can be projected***]

DBMS - Prachiti Pimple

d) Find the passenger names for passengers who have bookings on at least one flight.

$$\Pi_{pname}(\text{passenger} \bowtie \text{booking})$$

e) Find the passenger names for those who do not have any bookings in any flights.

$$\Pi_{pname}((\Pi_{pid}(\text{passenger}) - \Pi_{pid}(\text{booking})) \bowtie \text{passenger})$$

[**Hint:** ***here applied a set difference operation. The set difference operation returns only pids that have no booking. The result is joined with passenger table to get the passenger names.***]

DBMS - Prachiti Pimple

f) Get the details of flights that are scheduled on both dates 01/12/2020 and 02/12/2020 at 16:00 hours.

$$(\sigma_{\,fdate\ =\ 01/12/2020\ \wedge\ time\ =\ 16:00}(\text{flight})) \cap (\sigma_{\,fdate\ =\ 02/12/2020\ \wedge\ time\ =\ 16:00}(\text{flight}))$$

[**Hint:** ***the requirement is for flight details for both dates in common. Hence, set intersection is used between the temporary relations generated from application of various conditions.***]

DBMS - Prachiti Pimple