Unit: III
Lecture: 3
Relational Algebra
(Part-I)

Unary Relational Operations (Select and Project):

Selection operator (σ): Selection operator is used to select tuples from a relation based on some condition.
Syntax:

$$\sigma_{\text{(Cond)(Relation Name)}}$$

e.g. extract employees whose salary is <10000 from Table 1 Employee:

$$\sigma_{\text{(salary <1000)(Employee)}}$$

Result will be:

EID	Name	Salary
2E	Ramesh	5000
3E	Smith	8000
4E	Jack	6000

Projection Operator (∏): Projection operator is used to project particular columns from a relation.
Syntax:

$$\prod_{\text{(Column 1,Column 2....Column n)(Relation Name)}}$$

e.g. extract EID and Name from the relation Employee

$$\prod_{\text{(EID, Name)(Employee)}}$$

Result will be:

EID	Name
1E	John
2E	Ramesh
3E	Smith
4E	Jack
5E	Nile

Relational algebra is a procedural query language. It uses a collection of operators to compose the queries.3
Every operator in the algebra accepts wither one or two relation instances as arguments and output a resultant relation instance.

Operations in Relational Algebra:
The relational algebraic operations can be divided into:

1. **Basic Set Oriented Operations:** Union, Intersection, Set difference and Cartesian product.
2. **Relational Oriented Operations:** Selection, Projection, Division and Joins

Consider the Employee-Student database shown below. This database contain two tables Employee and Student and the relationship is that an employee can also be a student and vice-versa.

Employee		
EID	**Name**	**Salary**
1E	John	10000
2E	Ramesh	5000
3E	Smith	8000
4E	Jack	6000
5E	Nile	15000

Student		
SID	**Name**	**Fees**
1S	Smith	1000
2S	Vijay	950
3S	Gaurav	2000
4S	Nile	1500
5S	John	950

Fig: Employee and Student Relations

Basic Set-oriented Operations:

- **The union operation:** The union operation is a binary operation that is used to find union of relations. Here relations are considered as sets. So, duplicate values are eliminated.
  - **It is denoted by (U).**
  - **Conditions for Union Operation:**
    - Both the relations should have same number of attributes.
    - Data types of their corresponding attributes must be same.
  - Two relations are said to be union compatible if they follow the above two conditions.
  - e.g. if we want to find the names of all employees and names of all students together then query is:
    $\prod_{\text{Name}}$ **(Employee) U** $\prod_{\text{Name}}$ **(Student)**

Result:

Name
John
Ramesh
Smith
jack
Nile
Vijay
Gaurav

- **Set Intersection Operation:** Set intersection is used to find common tuples between two relations.
  - **It is denoted by** ∩
  - e.g. if we want to find all the employees from Relation Employee those are also student. Then query is:

    $\prod_{\text{Name}}$ **(Employee)** ∩ $\prod_{\text{Name}}$ **(Student)**

Result:

Name
John
Smith
Nile

- **Set Difference Operation:** Set-difference operation is a binary operation which is used to find tuples that are present in one relation but not in other relation.
  - **It is denoted by (-).**
  - It removes the common tuples of two relations and produce a new relation having rest of the tuples of first relation.
  - e.g. if we want the names of those employees that are not students, then query is:

    $\prod_{\text{Name}}$ **(Employee)** - $\prod_{\text{Name}}$ **(Student)**

Result:

Name
Ramesh
Jack

- **Cartesian Product Operation:** Cartesian product is a binary operation which is used to combine information of any two relations.
  - **It is denoted by (x).**
  - Suppose a relation R1 is having m tuples and other relation R2 is having n tuples then R1 x R2 has m x n tuples.

e.g. Consider two relations given below:

Employee		
EID	**Name**	**JID**
1E	Manoj	1J
2E	Deepak	2J
3E	Vinay	1J

Job	
JID	**Job**
1J	Tester
2J	Manager

Say query is: Employee x Job

The result of Cartesian product will be:

EID	**Name**	**Employee JID**	**Job JID**	**Job**
1E	Manoj	1J	1J	Tester
1E	Manoj	1J	2J	Manager
2E	Deepak	2J	1J	Tester
2E	Deepak	2J	2J	Manager
3E	Vinay	1J	1J	Tester
3E	Vinay	1J	2J	Manager