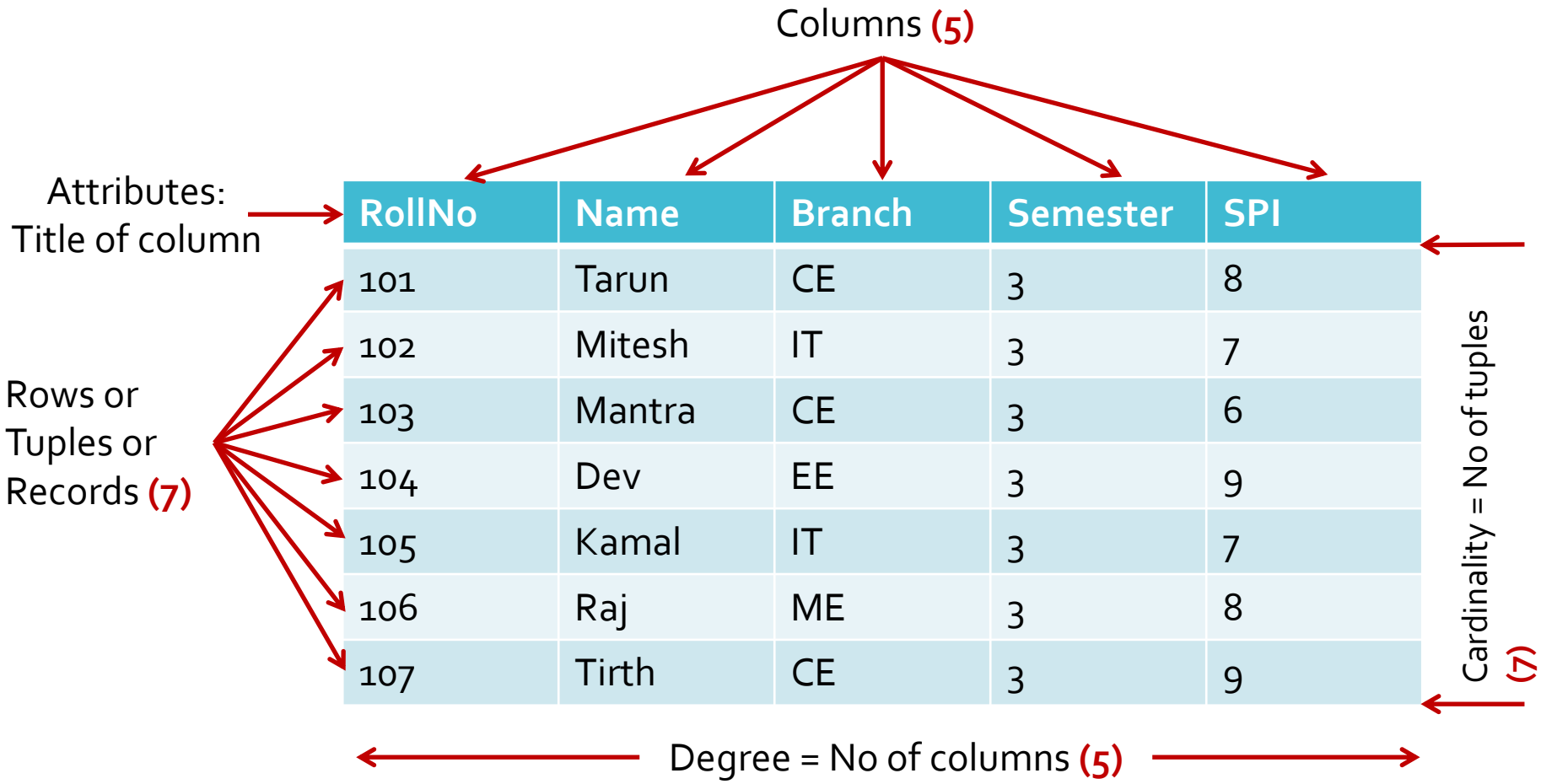
Parul® University

Database Management System (303105203)

Unit – 4: Relational Query Languages: Relational Algebra

Rina kumari,
Assistant Professor, Computer Science & Engineering

- Structure of relational databases,
- Domains, Relations, keys
- Relational algebra fundamental operators and syntax,
- Relational algebra queries
- Tuple relational calculus
- Open Source and Commercial DBMS



- **Domain** is a set of all possible unique values for a specific column.
- Domain of Branch attribute is **(CE, IT, ME, EE)**

Structure of relational databases

- **Table (Relation):** A database object that holds a collection of data for a specific topic. Table consist of rows and columns.
- **Column (Attribute):** The vertical component of a table. A column has a name and a particular data type; e.g. varchar, decimal, integer, datetime etc.
- **Record (Tuple):** The horizontal component of a table, consisting of a sequence of values, one for each column of the table. It is also known as row.
- A database consists of a collection of tables (relations), each having a unique name.

Exercise

1. Find out following for the given Student table:
   i. No of columns
   ii. No of records
   iii. Different attributes
   iv. Degree
   v. Cardinality

Enroll.no	Name	Branch	Age	Birthdate	City
S1	Dev	IT	18	01-05-2002	Rajkot
S2	Karan	CE	20	19-03-2000	Mumbai
S3	Jatin	ME	20	04-02-2000	Baroda
S4	Meet	IT	17	16-08-1999	Delhi
S5	Niraj	EC	19	20-02-2001	Patna

Super Key

A super key is a set of one or more attributes whose values uniquely identifies each record

Super Key
EnrollNo

Super Key
(RollNo, Name, Branch)

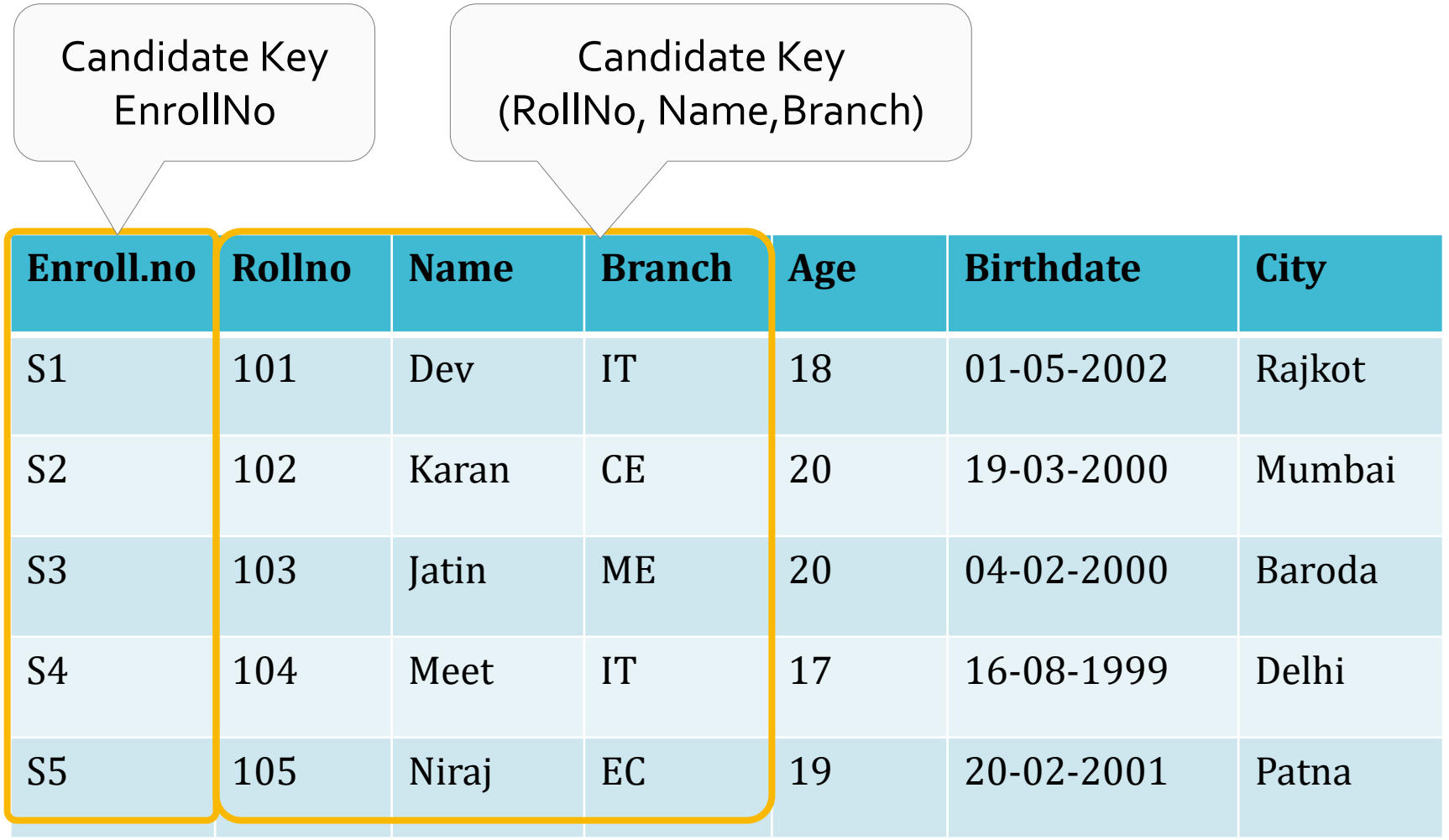
Super Key
(Age, Birthdate, City)

Enroll.no	Rollno	Name	Branch	Age	Birthdate	City
S1	101	Dev	IT	18	01-05-2002	Rajkot
S2	102	Karan	CE	20	19-03-2000	Mumbai
S3	103	Jatin	ME	20	04-02-2000	Baroda
S4	104	Meet	IT	17	16-08-1999	Delhi
S5	105	Niraj	EC	19	20-02-2001	Patna

Candidate Key

- A candidate key is a subset of a super key.
- A candidate key is a single attribute or the least combination of attributes that uniquely identifies each record in the table.
- A candidate key is a super key for which no proper subset is a super key.
- Every candidate key is a super key but every super key is not a candidate

Candidate Key



Enroll.no	Rollno	Name	Branch	Age	Birthdate	City
S1	101	Dev	IT	18	01-05-2002	Rajkot
S2	102	Karan	CE	20	19-03-2000	Mumbai
S3	103	Jatin	ME	20	04-02-2000	Baroda
S4	104	Meet	IT	17	16-08-1999	Delhi
S5	105	Niraj	EC	19	20-02-2001	Patna

Primary Key

- Ensures that a column or a group of columns identify uniquely each row in the table.
- A primary key is a candidate key that is chosen by database designer to identify tuples uniquely in a relation (table).
- A primary key may have one or more attributes.
- There is only one primary key in the relation (table).
- A primary key attribute value cannot be null.

Primary Key

Composite Primary Key

Primary Key
EnrollNo

Candidate Key
(RollNo, Name,Branch)

Enroll.no	Rollno	Name	Branch	Age	Birthdate	City
S1	101	Dev	IT	18	01-05-2002	Rajkot
S2	102	Karan	CE	20	19-03-2000	Mumbai
S3	103	Jatin	ME	20	04-02-2000	Baroda
S4	104	Meet	IT	17	16-08-1999	Delhi
S5	105	Niraj	EC	19	20-02-2001	Patna

Foreign Key

- A foreign key is an attribute or collection of attributes in one table that refers to the primary key in another table.
- A table containing the foreign key is called the child table, and the table containing the primary key is called the parent table.
- Also known as Referential Integrity Constraint

Foreign Key

EnrollNo	Name	Branch	Sem
101	Jatin	CE	3
102	Meet	CE	3
103	Ram	CE	3

ProjectID	Title	EnrollNo
P01	Bank	101
P02	College	102
P03	School	102

Relational Algebra Operations

- Selection
- Projection
- Cross Product
- Division
- Rename
- Joins
  - Natural join / Inner join
  - Outer join
    - Left outer join
    - Right outer join
    - Full outer join
- Set Operators
  - Union
  - Intersection
  - Minus / set difference
- Aggregate Functions

Selection Operator

- Symbol : σ (SIGMA)
- Notation : σ (condition) <Relation>
- Operation : Display particular tuples from a relation that satisfy a given condition (predicate)
- Operators: =, <>, <, >, <=, >=, Λ (AND), V (OR)

Example:

- Select all deposit which belongs to branch “verse”
    - σ branch=“verse” (Deposit)

Selection Operator Example

- **Example:**
- Find out all records where A = B and D > 5

  - Relation r

A	B	C	D
α	α	1	7
α	β	5	7
β	β	12	3
β	β	23	10

  - $\sigma_{A=B \wedge D > 5}(r)$

A	B	C	D
α	α	1	7
β	β	23	10

Selection Operator Example

Student

Enroll.no	Name	Branch	Age	Birthdate	City
S1	Dev	IT	18	01-05-2002	Rajkot
S2	Karan	CE	20	19-03-2000	Mumbai
S3	Jatin	ME	20	04-02-2000	Baroda
S4	Meet	IT	17	16-08-1999	Delhi
S5	Niraj	EC	19	20-02-2001	Patna

Display the detail of students belongs to “IT” branch and having age greater than 17.

$$\sigma_{Branch='IT' \wedge age > 17}(Student)$$

Output

Enroll.no	Name	Branch	Age	Birthdate	City
S1	Dev	IT	18	01-05-2002	Rajkot

Projection Operator

- Symbol: $\prod$ (Pi)
- Notation : $\prod_{\text{(attribute list)}}$ <Relation>
- Operation: Selects specified attributes of a relation.
- Project operation selects certain columns from a table while discarding others. It removes any duplicate tuples (records) from the result.
- The result of the project operation has only the attributes specified in the attribute list and in the same order as they appear in list.

Example

- List out all from deposit with bname and amount
  - $\prod$ (bname,amount) (Deposit)

Projection Operator Example

Student

Enrollno	Name	Branch	Age	Birthdate	City
S1	Dev	IT	18	01-05-2002	Rajkot
S2	Karan	CE	20	19-03-2000	Mumbai
S3	Jatin	ME	20	04-02-2000	Baroda
S4	Meet	IT	17	16-08-1999	Delhi
S5	Niraj	EC	19	20-02-2001	Patna

Display the Enrollno,Name and city of all students.

$$\Pi_{Enrollno,\ Name,\ city}(Student)$$

Output

Enrollno	Name	City
S1	Dev	Rajkot
S2	Karan	Mumbai
S3	Jatin	Baroda
S4	Meet	Delhi
S5	Niraj	Patna

Combined Example

Student					
Enrollno	**Name**	**Branch**	**Age**	**Birthdate**	**City**
S1	Dev	IT	18	01-05-2002	Rajkot
S2	Karan	CE	20	19-03-2000	Mumbai
S3	Jatin	ME	20	04-02-2000	Baroda
S4	Meet	IT	17	16-08-1999	Delhi
S5	Niraj	EC	19	20-02-2001	Patna

Display the Enrollno,Name and city of “IT” branch students.

$$\Pi_{Enrollno,\ Name,\ city}(\sigma_{Branch='IT'}(Student))$$

Output		
Enrollno	**Name**	**City**
S1	Dev	Rajkot
S4	Meet	Delhi

Cross (Cartesian Product) Operation

- Symbol : X (Cross)
- Notation : Relation1 X Relation2
- Operation : Combines information of two relations, It is also known as Cross-Product operation and similar to mathematical Cartesian Product Operation
- Result : for Relation1 and Relation2 if they have n1 and n2 attributes respectively then resultant relation will have n1+n2 attributes.
- Combining attributes from both the input relations .

Cross (Cartesian Product) Operation

- If both relation have same name of attributes then they can be distinguished by
  - Relation1.n1
  - Relation2.n1
  - i.e. RelationName.AttributeName
- If Relation1 and Relation2 have tuples t1 and t2 respectively then the resultant relation will have t1 * t2, combining each possible pair of tuples from both the relations.

  - Attributes of Resultant Relation = Attributes of R1 + Attributes of R2
  - Tuples of Resultant Relation = Tuples of R1 * Tuples of R2

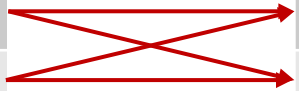
Cross Operation Example

Student

RollNo	Name	Branch
101	Raj	CE
102	Meet	ME

Result

RollNo	SPI
101	8
103	9



Student × Result

Student.RollN	Name	Branch	Result.RollN	SPI
101	Raj	CE	101	8
101	Raj	CE	103	9
102	Meet	ME	101	8
102	Meet	ME	103	9

Division Operator

- Symbol: ÷
- Notation : Relation1 ÷ Relation2
- **Note:** Attributes of relation2 is proper subset of relation1.
- Operation:
  - The output of the division operator will have attributes= All attributes of relation1- All attributes of relation2
  - The output of the division operator will have tuples= Tuples in relation1 which are associated with all tuples of relation2.

Division Operator

A

Sno	Pno
S1	P1
S1	P2
S1	P3
S1	P4
S2	P1
S2	P2
S3	P2
S4	P2
S4	P4
S5	P4

B1

Pno
P2

B2

Pno
P2
P4

B3

Pno
P1
P2
P4

A ÷ B1

Sno
S1
S2
S3
S4

A ÷ B2

Sno
S1
S4

A ÷ B3

Sno
S1

Rename Operator

- Symbol: ρ *(rho)*
- Notation : $\rho_{A\ (x1,x2,x3...Xn)}\ (R)$
- Operation:
  - It renames the existing relation.
  - $\rho_{x}\ (R)$ *: Renames relation R to x*
  - $\rho_{x(A1,A2,...AN)}\ (R)$ *: Renames relation R to x And its attributes to A1, A2, …. An*
  - $\rho_{A1,A2,...AN}\ (R)$ *:Renames all the attributes of relation.*

Rename Operator

Example
Book(Tittle,Author,Year,Price)

- Book relation with attributes Tittle,Author, Year and Price. The rename operator is used on Book relation as follows:
- Here both the relation name and attribute names are renamed

$$\rho_{\text{Temp (Bname, Aname, Pyear, Bprice)}} (\text{Book})$$

- Here only the relation name is renamed

$$\rho_{\text{Temp}} (\text{Book})$$

- In this case only the attributes names are renamed

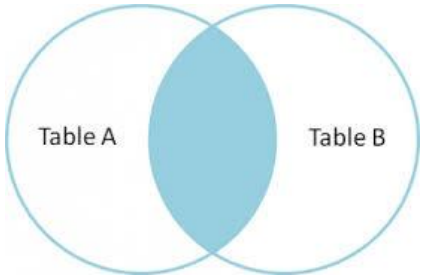
$$\rho_{\text{(Bname, Aname, Pyear, Bprice)}} (\text{Book})$$

Natural Join (Inner Join)

- Symbol: ⋈
- Notation: Relation-1 (R1) ⋈ Relation-2 (R2)
- Operation: Natural join will retrieve consistent data from multiple relations.
- It combines records from different relations that satisfy a given condition.

To perform a natural join there must be **one common attribute (column)** between two relations.

Step – 1	It performs Cartesian Product
Step – 2	Then it deletes inconsistent tuples
Step – 3	Then it removes an attribute from duplicate attributes



Natural Join (Inner Join) Example

Student

RollNo	Name	Branch
101	Raj	CE
102	Meet	ME

Result

RollNo	SPI
101	8
103	9

Student × Result

Student.RollNo	Name	Branch	Result.RollNo	SPI
101	Raj	CE	101	8
101	Raj	CE	103	9
102	Meet	ME	101	8
102	Meet	ME	103	9

Step 1 : Performs Cartesian Product

Student ⋈ Result

Step 2 : Removes inconsistent tuples

Student.RollNo	Name	Branch	Result.RollNo	SPI
101	Raj	CE	101	8

Step 3 : Removes an attribute

Student ⋈ Result

RollNo	Name	Branch	SPI
101	Raj	CE	8

Outer Join

- In natural join some records are missing, if we want that missing records than we have to use outer join.
- Types:
  - Left Outer Join ⟕
  - Right Outer Join ⟖
  - Full Outer Join ⟗

Left Outer Join

- Display all the tuples of the left relation even through there is no matching tuple in the right relation.
- For such kind of tuples having no matching, the attributes of right relation will be padded with **null** in resultant relation.
- It is denoted by $\supset\!\bowtie$ symbol.

Left Outer Join Example

Class

ID	NAME
1	Abhi
2	Adam
3	Alex
4	Anu
5	Ashish

Class_info

ID	Address
1	DELHI
2	MUMBAI
3	CHENNAI
7	NOIDA
8	PANIPAT

Class ⟕ Class_info

ID	NAME	ID	Address
1	Abhi	1	DELHI
2	Adam	2	MUMBAI
3	Alex	3	CHENNAI
4	Anu	null	null
5	Ashish	null	null

Right Outer Join

- Display all the tuples of right relation even through there is no matching tuple in the left relation.
- For such kind of tuples having no matching, the attributes of left relation will be padded with null in resultant relation.
- It is denoted by ⟕ symbol.

Right Outer Join Example

Class

ID	NAME
1	Abhi
2	Adam
3	Alex
4	Anu
5	Ashish

Class_info

ID	Address
1	DELHI
2	MUMBAI
3	CHENNAI
7	NOIDA
8	PANIPAT

Class ⟖ Class_info

ID	NAME	ID	Address
1	Abhi	1	DELHI
2	Adam	2	MUMBAI
3	Alex	3	CHENNAI
null	null	7	NOIDA
null	null	8	PANIPAT

Full Outer Join

- Display all the tuples of both of the relations. It also pads null values whenever required.
- **(Left outer join + Right outer join)**
- For such kind of tuples having no matching, it will be padded with null in resultant relation.
- It is denoted by ⟗ symbol.

Full Outer Join Example

Class

ID	NAME
1	Abhi
2	Adam
3	Alex
4	Anu
5	Ashish

Class_info

ID	Address
1	DELHI
2	MUMBAI
3	CHENNAI
7	NOIDA
8	PANIPAT

Class ⟗ Class_info

ID	NAME	ID	Address
1	Abhi	1	DELHI
2	Adam	2	MUMBAI
3	Alex	3	CHENNAI
4	Anu	null	null
5	Ashish	null	null
null	Null	7	NOIDA
null	Null	8	PANIPAT

Set operators

- Set operators combine the results of two or more queries into a single result.
- Different types of set operators
  1. Union
  2. Intersection
  3. Minus (Set Difference)

Union Operator

- Symbol: $\cup$ (Union)
- Notation : Relation1 $\cup$ Relation2
- Operation:
  - *Selects tuples those are in either or both of the relation*

Union Operation Example

- **Relations *r, s:***

A	B
α	1
α	2
β	1

r

A	B
α	2
β	3

s

r ∪ s

A	B
α	1
α	2
β	1
β	3

Union removes duplicate records.

Union Operator Example

Display Name of person who are either employee or customer

Employee

ID	Name	Salary
2	Kevin	1000
3	Jay	5000

Customer

ID	Name	Balance
1	Rajesh	3000
2	Kevin	7000

$$\Pi_{Name}(Employee) \cup \Pi_{Name}(Customer)$$

Output

Name
Kevin
Jay
Rajesh

Intersection Operator

- Symbol: ∩ (Intersection)
- Notation : Relation1 ∩ Relation2
- Operation:
  - *Selects tuples those are in both relation*

Display Name of person who are employee as well as customer.

Employee

ID	Name	Salary
2	Kevin	1000
3	Jay	5000

Customer

ID	Name	Balance
1	Rajesh	3000
2	Kevin	7000

Intersection Operator Example

$$\Pi_{Name}(Employee) \cap \Pi_{Name}(Customer)$$

Output

Name
Kevin

Minus (Set difference) Operator

- Symbol: -
- Notation : Relation1 - Relation2
- Operation:
  - Returns all the records from first (left) query that are not contained in the second (right) query.

Manager

Name
Jay
Tirth
Hari

Employee

Name
Hari
Tirth
Smit

Manager – Employee

Name
Jay

Employee – Manager

Name
Smit

Aggregate Functions

- *Aggregation function takes a collection of values and returns a single value as a result.*

  avg: average value
  min: minimum value
  max: maximum value
  sum: sum of values
  count: number of values

- *Aggregate operation in relational algebra*

$$_{G_1,G_2,\ldots,G_n}\vartheta_{F_1(A_1),F_2(A_2,\ldots,F_n(A_n)}(E)$$

- *E is any relational-algebra expression*

  G_1, G_2 …, G_n is a list of attributes on which to group (can be empty)
  Each F_i is an aggregate function
  Each A_i is an attribute name

Aggregate Operation Example

Relation *r*:

A	B	C
α	α	7
α	β	7
β	β	3
β	β	10

Find out sum of attribute C

$g_{\text{sum(c)}}(r)$

sum(c)
27

Aggregate Operation Example

Relation *r*:

A	*B*	*C*
α	α	7
α	β	7
β	β	3
β	β	10

Find out maximum and minimum from attribute C

$$g_{\mathbf{max(c),min(c)}}(\mathrm{r})$$

max(*c*)	min(*c*)
10	3

Aggregate Operation Example

- *Relation account grouped by branch-name:*

branch_name	*account_number*	*balance*
Perryridge	A-102	400
Perryridge	A-201	900
Brighton	A-217	750
Brighton	A-215	750
Redwood	A-222	700

Find out total of balance branch-name wise from account:

$$branch\_name\ \mathcal{g}\ \mathbf{sum}(balance)\ (account)$$

branch_name	**sum**(*balance*)
Perryridge	1300
Brighton	1500
Redwood	700

Open source Vs Commercial DBMS

Open source	Commercial
It is available free of cost in market.	*It is available at certain cost in market.*
As it is open source, anyone can modify the code.	*It is not accessible to unauthorized person.*
Ex. MySQL, MongoDB, SQLite etc.	*Ex. Microsoft SQL server, IBM Db2 etc.*

Thanks