1

Subject : DBMS
Topic : RELATIONAL ALGEBRA

2

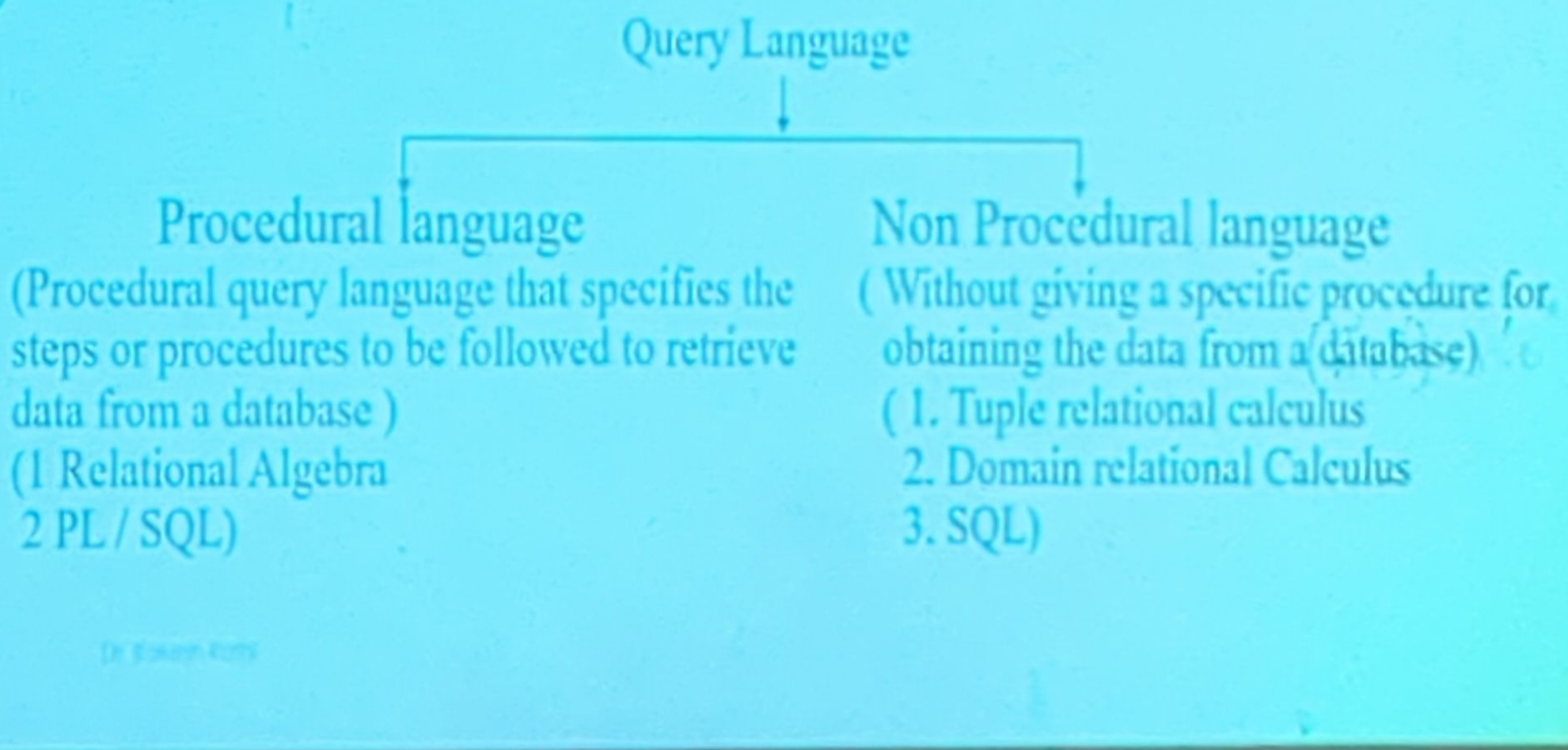
Relational Model

Query language is a language in which a user request to retrieve information from the database.

Relation → Table, Entity Set

Row → Tuple, Entity

Column → Attributes, Fields



3 Relational Algebra

- The **Relational Algebra** is a **Procedural Query Language**
- It consists of a set of operations that take one or two relations as input and produce a new relation as their result.

Cont.

- The fundamental operations in the relational algebra are *Select, Project, Union, Set difference, Cartesian product, and Rename*

- The Select, Project, and Rename operations are called *Unary Operations, because* they operate on one relation

- The Cartesian Product, Union, Set difference, Set intersection, Natural Join, Division, Assignment operations operate on pairs of relations and are, therefore, called *Binary Operations*.

Relations →

borrow = (branch_name, loan_number, customer_name, amount)

deposit = (branch_name, account_number, customer_name, balance)

customer = (customer_name, street, city)

client = (customer_name, banker_name)

branch = (branch_name, branch_city, assets)

6 Unary Relational Operations

- SELECT (symbol: σ (sigma))
- Selects a tuples from relation
- PROJECT (symbol: π (pi))
- Selects columns from relation or list those attributes that we wish to appear in the result.
- RENAME (symbol: ρ (rho))
- To avoid ambiguity rename the relation.

7

SELECTION

$$\sigma_{Selection\_Criteria}(Input)$$

Manipulates data in a single relation

A relation instance

The selection operator specifies the tuples to retain through selection criteria.

A boolean combination (i.e. using V and Λ) of terms

Attribute op constant or attribute1 op attribute2

<, <=, =, ≠, >=, or >

EXAMPLE

1. Select tuples of borrow relation – where the branch is "Perryridge".

borrow = (branch_name, loan_number, customer_name, amount)

$\sigma_{branch\_name = \text{"Perryridge"}}$ (borrow)

2. Find all tuples in which the amount borrowed is more than 1200

$\sigma_{amount > 1200}$ (borrow)

3. Find those tuples pertaining to loans of more than 1200 made by the Perryridge branch.

$\sigma_{branch\_name = \text{"Perryridge"} \wedge amount > 1200}$ (borrow)

9

PROJECTION

$$\pi_{fields}(Input)$$

Allows us to extract columns from a relation

Example: S2

sid	sname	rating	age
28	yuppy	9	35.0
31	lubber	8	55.5
44	guppy	5	35.0
58	rusty	10	35.0

$\Rightarrow \pi_{age}(S2) \Rightarrow$

age
35.0
55.5

10 Composition of Relational Operations

S2

sid	sname	rating	age
28	yuppy	9	35.0
31	lubber	8	55.5
44	guppy	5	35.0
58	rusty	10	35.0

$$\pi_{sname,rating}(\sigma_{rating>8}(S2))$$

sname	rating
yuppy	9
rusty	10

EXAMPLE

borrow = (branch_name, loan_number, customer_name, amount)

1. Show customers and the branches from which they borrow.

$$\Pi_{\text{branch\_name, customer\_name}} (\text{borrow})$$

2. Find the customer_name who borrowed more than 1000

$$\Pi_{\text{customer\_name}} (\sigma_{\text{amount} > 1000} (\text{borrow}))$$

Cartesian-Product Operation : X (Cross)

- Cartesian Product is used to extract information from several relations.
- Also known as product or cross-join
- Notation $R \times S$
- Combines tuples of one relation to other relation
- $R(a_1, a_2 \ldots a_n) \times S(b_1, b_2 \ldots b_n)$
  - $T(a_1, a_2 \ldots a_n, b_1, b_2 \ldots b_n)$

Cartesian-Product Operation

- Relations *R*,*S*:

A	*B*
α	1
β	2

R

C	*D*	*E*
α	10	a
β	10	a
β	20	b
γ	10	b

S

- *R* x *S*:

A	*B*	*C*	*D*	*E*
α	1	α	10	a
α	1	β	10	a
α	1	β	20	b
α	1	γ	10	b
β	2	α	10	a
β	2	β	10	a
β	2	β	20	b
β	2	γ	10	b

EXAMPLE

client = (customer_name, banker_name)

customer = (customer_name, street, customer_city)

Find all client of banker Johnson and the city in which they live (Relations used client and customer).

$$\Pi_{\text{client. customer\_name, customer\_city}} (\sigma_{\text{client.customer\_name = customer.customer\_name} \wedge \text{banker\_name = "Johnson"}} (\text{client } X \text{ customer}))$$

15 RENAME

- The general RENAME operation ρ can be expressed by any of the following forms:
  - $\rho_S(R)$ Rename Relation :
    - *R is relation name and S is new name of relation*
  - $\rho_{(A1/B1,\ldots)}(R)$ Rename Attribute :
    - *will rename the attribute 'B1' of relation by 'A1'.*
  - $\rho_{S\,(A1/B1,\ldots)}(R)$ Rename both Relation and Attribute:
    - *the relation name to S, and*
    - *the column (attribute) names to A1.*

EXAMPLE

Find the names of all customers who live on the same street and in the same city as Smith.

customer = (customer_name, street, customer_city)

First obtained the street and city of Smith

$$\Pi_{\text{street, customer\_city}} (\sigma_{\text{customer\_name = "Smith"}} (\text{customer}))$$

$$\Pi_{\text{customer.customer\_name}} (\sigma_{\text{customer.street = customer1.street} \wedge \text{customer.customer\_city = customer1.customer\_city}} (\text{customer}) \times (\Pi_{\text{street, customer\_city}} (\sigma_{\text{customer\_name = "Smith"}} (\rho_{\text{customer1}} (\text{customer}))))))$$

17

Relational Algebra Operations Set Theory

- Union
- Intersection
- Set Difference / Minus
- Division Operation
- Outer Join
- Natural Join
- Assignment Operator

18

UNION

- It is a Binary operation, denoted by ***U***

  – The result of R ***U*** S, is a relation that includes all tuples that are either in R or in S or in both R and S

  – Duplicate tuples are eliminated.

$(A+B)' = A' \cdot B'$

19

UNION

- The two operand relations R and S must be "type compatible" (or UNION compatible)

- Two relations are *union compatible* if
  - Relation R and S should have same number of columns
  - Names of attributes and the domain type are the same in both

Set Operation: Union

S1

sid	sname	rating	age
22	dustin	7	45.0
31	lubber	8	55.5
58	rusty	10	35.0

S2

sid	sname	rating	age
28	yuppy	9	35.0
31	lubber	8	55.5
44	guppy	5	35.0
58	rusty	10	35.0

S1 U S2

sid	sname	rating	age
22	dustin	7	45.0
31	lubber	8	55.5
58	rusty	10	35.0
44	guppy	5	35.0
28	yuppy	9	35.0

EXAMPLE

1. Find all customers of the Perryridge branch (i.e. Find all customers who has a loan, an account or both in Perryridge branch).

a. $\Pi_{\text{customer\_name}}(\sigma_{\text{branch\_name} = \text{"Perryridge"}}(\text{borrow}))$

b. $\Pi_{\text{customer\_name}}(\sigma_{\text{branch\_name} = \text{"Perryridge"}}(\text{deposite}))$

$\Rightarrow \Pi_{\text{customer\_name}}(\sigma_{\text{branch\_name} = \text{"Perryridge"}}(\text{borrow})) \cup \Pi_{\text{customer\_name}}(\sigma_{\text{branch\_name} = \text{"Perryridge"}}(\text{deposite}))$

INTERSECTION

- INTERSECTION is denoted by $\cap$
- The result of the operation $R \cap S$, is a relation that includes all tuples that are in **both** R and S
- $R \cap S = R - (R - S)$
  - The attribute names in the result will be the same as the attribute names in R

- The two operand relations R and S must be "type compatible

Set Operation: Intersection

S1

sid	sname	rating	age
22	dustin	7	45.0
31	lubber	8	55.5
58	rusty	10	35.0

S2

sid	sname	rating	age
28	yuppy	9	35.0
31	lubber	8	55.5
44	guppy	5	35.0
58	rusty	10	35.0

sid	sname	rating	age
31	lubber	8	55.5
58	rusty	10	35.0

S1 ∩ S2

EXAMPLE

1. Find all customers with both a loan and an account at the Perryridge branch.

$$\Pi_{customer\_name}(\sigma_{branch\_name = \text{“Perryridge”}}(borrow)) \cap$$

$$\Pi_{customer\_name}(\sigma_{branch\_name = \text{“Perryridge”}}(deposit))$$

$(A+B)' = A'.B'$

Set Operation: Set-Difference

S1

sid	sname	rating	age
22	dustin	7	45.0
31	lubber	8	55.5
58	rusty	10	35.0

S2

sid	sname	rating	age
28	yuppy	9	35.0
31	lubber	8	55.5
44	guppy	5	35.0
58	rusty	10	35.0

S1 – S2

sid	sname	rating	age
22	dustin	7	45.0

EXAMPLE

Find all customers of the Perryridge branch who have an account there but not a loan.

$$\Pi_{\text{customer\_name}}(\sigma_{\text{branch\_name} = \text{"Perryridge"}}(\text{deposite})) - \Pi_{\text{customer\_name}}(\sigma_{\text{branch\_name} = \text{"Perryridge"}}(\text{borrow}))$$

28 Division operation (÷) " for all "

1. Find all customers who have an account at all branches located in Brooklyn.

a. $\Pi_{\text{branch\_name}} (\sigma_{\text{branch\_city} = \text{"Brooklyn"}} (\text{branch}))$

b. $\Pi_{\text{customer\_name, branch\_name}} (\text{deposit})$

$$\Pi_{\text{customer\_name, branch\_name}} (\text{deposit}) \div \Pi_{\text{branch\_name}} (\sigma_{\text{branch\_city} = \text{"Brooklyn"}} (\text{branch}))$$