Relational Algebra (RA)

The relational algebra defines a set of operations on relations, paralleling the usual algebraic operations such as addition, subtraction, multiplication, which operates on numbers.

RA provides set of operations that take one or more relations as input and return a relation as output.

Practical query languages such as SQL are based on the relational algebra, but add a number of useful syntactic features.

Symbol	use
σ (select)	Selection of rows/tuples/records
Π (project)	Selection of domain values.
$\mathcal{G}$ (calligraphic G)	Applying aggregation fn
$\times$	Cartesian product
$\bowtie$	Natural join
$\overline{\bowtie}$	Left Outer Join
$\overline{\bowtie}$	Right Outer Join
$\overline{\bowtie}$	Full Outer Join
$\cup$	Union

Symbol	use
$\cap$	Intersection
$-$	Set difference
$\div$	Division
$\vee$	or
$\wedge$	and
$<>$	not equal to

① Select operation (σ)

Customer (cname, custst, custcity)
branch (bname, bcity, assets)
account (ano, bname, balance)
loan (lno, bname, amount)
depositor (cname, ano)
borrower (cname, lno)

(i) List the customer details.

Select * from customer;

σ (customer)

(2) Select (σ) with condition.

(i) List the accounts with more than 2000 as balance.

```
Select * from account
where balance > 2000;
```

$\sigma_{balance > 2000}(account)$

(ii) List the accounts with more than 2000 as balance & accounts opened from vandaleur branch.

```
Select * from account where
balance > 2000 and brane = "vandaleur";
```

$\sigma_{balance > 2000 \wedge brane = "vandaleur"}(account)$

(iii) List the account details opened from vandaleur or kelaubakkara branch

```
Select * from account where
brane = "vandaleur" or brane = "kelau";
```

$\sigma_{bname="randalur" \vee bname="kelam"}$ (account)

(iv) List the account from kelambakkam or randalur branch and has 2000 balance.

Select * from account where balance > 2000 and (bname = "randalur" or bname = "kelambakkam");

$\sigma_{(bname="randalur" \vee bname="kelam") \wedge balance > 2000}$ (account)

(3) Project operation (Π)

(i) List the name of all customers

Select cname from customer;

Π_{cname} (customer)

(vi) List the customer details with project operation

Select cname, cust-st, cust-city from customer;

$\Pi_{cname, custst, Cust\text{-}city}$ (customer)

(4) Combination of Select(σ) + project(Π)
↳ When a domain has to be retrieved after applying a condition.

(i) List the customer name who lives in chennai.

Select cname from customer where custCity = "chennai";

$\Pi_{cname}(\sigma_{custCity = "chennai"}(customer))$

(ii) List the loanno & branchname, which has loan amount more than 1 lakh.

Select loanno, branch from loan where amount > 100000;

$$\Pi_{lno, bname}(\sigma_{amount > 100000}(loan))$$

(4) Aggregation function (S)

Sum
avg
count
min
max

group by
having

(i) Count the number of customers

Select count(cname) from customer;

$$S_{count(cname)}(customer)$$

(vi) List the maximum loan amount.

Select max(amount) from loan;

$\mathcal{G}_{max(amount)}(loan)$

Hint 1: No attribute can be specified along with an attribute for which aggregation fn is applied.

Hint 2: An attribute for which group by is applied can be displayed along with the aggregation.

(vii) List the maximum loan amount taken at each branch.

Select max(amount) from loan group by bname;

$bname\,\mathcal{G}_{max(amount)}(loan)$

(iv) List the maximum loan amount along with the branch name, taken at each branch

```
Select bname, max(amount) from
loan group by bname;
```

$$bname\ \mathcal{S}_{bname,\ max(amount)}(loan)$$

(v) List the max loan amount taken at vandalur branch

$$\mathcal{S}_{max(amount)}\left(\sigma_{bname = \text{"vandalur"}}(loan)\right)$$

```
Select max(amount) from
loan where bname = "vandalur";
```

Hint 3: When a condition is applied for a non-aggregated attribute, "where" has to be used.

Hint 4: When a condition is applied for an aggregated attribute, "having" has to be used.

Hint 5: "having" can be used only with "group by"

(vi). List the average loan amount at each branch, if the average loan amount is greater than 30,000

```
select avg(amount) from loan
group by bname having
avg(amount)>30000;
```

$\text{bname}\, \mathcal{S}_{\text{avg(amount)} > 30000}\, (\text{loan})$

(vi) List the branchcity with the number of branches in each city which has more than 1 lakh as assets, if the count exceeds 3

```
Select bcity, count(bname) from branch
where assets > 100000 group by bcity
having count(bname) > 3;
```

$\text{bcity}\, \mathcal{S}_{\text{bcity, count(bname)} > 3}\, (\sigma_{\text{assets} > 100000}\, (\text{branch}))$

(5) Joins

- Cartesian Product (X)
- Natural Join (⋈)
- Outer Join
  - Left Outer (⟕)
  - Right Outer (⟖)
  - full Outer (⟗)

APRIL 2013

(i) Cartesian Product

a) List the name of the customer with his account balance.

```
Select customer.cname, account.balance
from customer, depositor, account
where customer.cname = depositor.
cname and depositor.ano =
account.ano;
```

$$\Pi_{customer.cname,\ account.balance}\left(\sigma_{customer.cname = depositor.cname \wedge depositor.ano = account.ano}(C \times d \times a)\right)$$

(ii) Natural Join for query a)

```
Select cname, balance from
customer natural join depositor
natural join account;
```

$\Pi_{cname, balance}$ (customer ⋈ depositor ⋈ account)

b) List the name of the customer with his account balance if the balance is greater than 10,000.

Select cname, balance from customer natural join depositor natural join account where balance > 10000;

$\Pi_{cname, balance}$ ($\sigma_{balance > 10000}$ (customer ⋈ depositor ⋈ account))

Hint 6: Representation for any joins are same for RA expression, except cartesian product.

(ppt) Full outer join for query 6)

Select cname, balance from customer natural full outer join depositor natural full outer join account where balance > 10000;

$\Pi_{cname, balance}(\sigma_{balance>10000}(customer \bowtie depositor \bowtie account))$

Hint 6: Representation of all outer joins are same for sql + ra expressions.

(6) Aggregation with Joins

(i) Count the number of customers who have account at each branch

$bname\ \mathcal{G}_{count(cname)}(depositor \bowtie account)$

Select count(cname) from depositor natural full outer join account group by bname.

(ii) Count the number of customers who live in chennai & have account.

Select count(cname) from customer natural full outer join depositor where custcity = "chennai";

$$S_{count(cname)}(\sigma_{custcity="chennai"}(depositor \bowtie customer))$$

(7) Set operations.

(i) union (∪)

List the customers who have account or loan.

(Select cname from depositor) union (Select cname from borrower)

$$\Pi_{cname}(depositor) \cup \Pi_{cname}(borrower)$$

(ii) Intersection.

List the customers who have account and loan at chennai branch

(Select cname from depositor natural full outer join account natural full outer join branch where bcity = "chennai") intersect (Select cname from borrower natural full outer join loan natural full outer join branch where bcity = "chennai")

$\Pi_{cname}(\sigma_{bcity="chennai"}(branch \bowtie account \bowtie depositor)) \cap \Pi_{cname}(\sigma_{bcity="chennai"}(branch \bowtie account \bowtie depositor))$