CS262- Problem Set 1

CS262- Database Systems
2021-CS-187 — Saqib Shehzad

February 16, 2023

Consider the following schema.
Company(name, city)
Description Relation list the company name and location of company in city attribute.

Product(name, maker, cost, year)
Description Each product has name, and manufacturer of product in maker, cost as purchase price, and year as the launch year of that particular product. product name is unique for all problems except problem No.4

Purchase(id, product, buyer, price)
Description Relation list the purchases made by customer listed in buyer columns, price as sale price, and product as name of product.

To-Do For each of the problems given below you are required to provide Relational algebra expression and at least five equivalent solutions in SQL, out of which one solution should be performed using

1. Cartesian product
2. Joins
3. Subquery

If any of the above solutions is not possible provide the reason as well.

Solution should be written in latex, use the following document as template for solution, https://www.overleaf.com/read/fkrzmpgybjnq. Submit pdf and tex file on eduko.

Problem 1. Find the products(names only) whose cost is more than the average cost.

Solution.

Solution 1 :

select name from Product where cost > (select AVG(cost) from Product)

Relational Algebra :

$\sigma_{name}(\Pi_{cost>avg(cost)}(Product))$

Solution 2 :

select DISTINCT p.name
from Product p,Product p1

where P.cost > (select AVG(cost) from Product)

Relational Algebra :

$\pi_{name}(\sigma_{cost>avg(cost)}(Product \times Product))$

Solution 3 :

```
SELECT NAME
from Product
except
select name
from Product
where cost <= (select AVG(cost) from Product)
```

Relational Algebra :

$\rho_{name \in Product}(\pi_{name}(\sigma_{cost>\bar{cost}_{Product}}(Product)))$

Solution 4 :

```
select p1.name
from Product p1
join Product p2
on p1.name = p2.name
where p1.cost > (select AVG(cost) from Product)
```

Relational Algebra :

$\pi_{Name}(\sigma_{Cost>\bar{Cost}}(Product \bowtie Product))$

Solution 5 :

```
SELECT NAME
from Product
intersect
select name
from Product
where cost > (select AVG(cost) from Product)
```

Relational Algebra :

$\rho_{name}(Product \cap \sigma_{cost>AVG(cost)}(Product))$

□

Problem 2. List the name of companies whose products are bought by Aslam.

Solution.

Solution 1 :

select Company.name
from Company
join Product
on Company.name = Product.maker
join Purchase
on product.name = Purchase.product
where Purchase.buyer = 'Aslam'

Relational Algebra :

$\pi_{Company.name}(Company \bowtie_{Company.name=Product.maker} Product \bowtie_{Product.name=Purchase.product} Purchase \bowtie_{Purchase.buyer='A}$
)

Solution 2 :

select Company.name
from Company
where Company.name in (select Product.maker from Product where Product.name in (select Purchase.product
from Purchase where buyer = 'Aslam'))

Relational Algebra :

$\pi_{name}(\sigma_{name \in \pi_{maker}(\sigma_{name \in Purchase.product \wedge buyer='Aslam'}(Product))}(Company))$

Solution 3 :

select c.name from Company c,Product,Purchase where buyer = 'Aslam' and (product.name = Purchase.product)
and (c.name = maker)

Relational Algebra :

$\rho_{c.name}(\sigma_{buyer='Aslam' \wedge (product.name=Purchase.product) \wedge (c.name=maker)}(Product \bowtie Purchase))$

Solution 4 :

select Company.name
from Company
intersect
select c.name
from Company c,Product,Purchase
where buyer = 'Aslam' and (product.name = Purchase.product) and (c.name = maker)

Relational Algebra :

$\pi_{name}(Company \cap \pi_{name}(Company \times Product \times Purchase)_{buyer=Aslam \wedge product=Purchase.product \wedge maker=c.name})$

Solution 5 :

```
select Company.name
from Company
except
select Company.name
from Company
join Product
on Company.name = Product.maker
join Purchase
on product.name = Purchase.product
where Purchase.buyer <> 'Aslam'
```

Relational Algebra :

$$\rho_{Company.name}(Company) \setminus \pi_{Company.name}(\sigma_{Purchase.buyer \neq Aslam}(Product \bowtie Purchase \bowtie Company))$$

□

Problem 3. List the name of products that are more expensive that all the products produced by Unilever.

Solution.

Solution 1 :

```
select name
from Product
where cost > (select sum(cost) from Product where maker = 'Unilever')
```

Relational Algebra :

$$\rho_{name}(\sigma_{cost > \sum_{\forall maker = Unilever}(cost)}(Product))$$

Solution 2 :

```
select p1.name
from Product p1
join Product p2
on p1.name = p2.name
where p1.cost > (select sum(cost) from Product where maker = 'Unilever')
```

Relational Algebra :

$$\rho_{p1.name}(\sigma_{p1.cost > \sum_{p.maker = Unilever}(p.cost)}(Product \times Product))$$

Solution 3 :

```
select distinct p.name
from Product p,Product p1
where P.cost > (select sum(cost) from Product where maker = 'Unilever')
```

Relational Algebra :

$$\pi_{name}(\sigma_{p.cost > \sum_{p1.maker = 'Unilever'}(p1.cost)}(Product \times Product))$$

Solution 4 :

SELECT NAME
from Product
intersect
select name
from Product
where cost > (select sum(cost) from Product where maker = 'Unilever')

Relational Algebra :

$\pi_{NAME}(\sigma_{cost>\sum_{maker='Unilever'} cost(Product)}(Product)) \cap \pi_{NAME}(Product)$

Solution 5 :

SELECT NAME
from Product
except
select name
from Product
where cost <= (select AVG(cost) from Product where maker = 'Unilever')

Relational Algebra :

$\pi_{Name}(Product \setminus \sigma_{Cost \leq avg(Cost|\ Maker=Unilever)}(Product))$ □

Problem 4. List the copy cat products along with manufacturer, i.e. the products that have the same name as produced by Unilever.

Solution.

Solution 1 :

select name,maker from Product where name in (select name from Product where maker = 'Unilever') and maker <> 'Unilever'

Relational Algebra :

$\rho_{name,maker}(\sigma_{name \in \rho_{name}(Product) \wedge maker='Unilever'}(Product) \wedge maker \neq' Unilever')$

Solution 2 :

select p1.name,p1.maker
from product p1
join product p2
on (p1.name = p2.name) and (p2.maker = 'Unilever')
where p1.maker <> 'Unilever'

Relational Algebra :

$\pi_{name,maker}(\sigma_{maker \neq Unilever}(\rho_{p1}(Product \times_{name} \rho_{p2}(Product \times_{maker=Unilever}))))$

Solution 3 :

select p1.name,p2.maker from product p1,product p2 where(p1.name = p2.name) and (p1.maker = 'Unilever') and p2.maker <> 'Unilever'

Relational Algebra :

$\pi_{name,maker}(\sigma_{name=p2.name,maker='Unilever'}(Product_1) \bowtie Product_2)$

□

Problem 5. Buyers of products produced in Lahore.

Solution.

Solution 1 :

```
select Purchase.buyer
from purchase
join Product
on purchase.product = Product.name
join Company
on Company.name = Product.maker
where Company.city = 'lahore'
```

Relational Algebra :

$\pi_{buyer}(Purchase \bowtie Product \bowtie Company)\sigma_{Company.city=\textbf{"lahore"}}(Purchase \bowtie Product \bowtie Company)$

Solution 2 :

```
select Purchase.buyer
from Purchase
where Purchase.product in (select Product.name from Product where Product.maker in (select Company.name
from Company where city = 'lahore'))
```

Relational Algebra :

$\pi_{buyer}(\sigma_{product \in \pi_{name}(Product) \wedge maker \in \pi_{name}(Company) \wedge city='lahore'}(Purchase))$

Solution 3 :

```
select p.buyer
from Purchase p,Product,Company
where Company.city = 'lahore' and (p.product = Product.name) and (Product.maker = Company.name)
```

Relational Algebra :

$\Pi_{buyer}(Purchase \ltimes_{product=Product.name} Product \ltimes_{maker=Company.name} Company \ \sigma_{city='lahore'}(Company))$

Solution 4 :

```
select purchase.buyer
from Purchase
intersect
select p.buyer
from Purchase p,Product,Company
where Company.city = 'lahore' and (p.product = Product.name) and (Product.maker = Company.name)
```

Relational Algebra :

$\pi_{buyer}(Purchase \cap \pi_{buyer}(Purchase \times Product \times Company\ \sigma\ Company.city =' lahore' \wedge Product.name = p.product \wedge Company.name = Product.maker))$

Solution 5 :

```
select purchase.buyer
from Purchase
except
select p.buyer
from Purchase p,Product,Company
where Company.city <> 'lahore' and (p.product = Product.name) and (Product.maker = Company.name)
```

Relational Algebra :

$\Pi_{buyer}(\sigma_{Company.city \neq' lahore'}(\rho_p(Purchase \bowtie Product \bowtie Company)))$ □

Problem 6. List of buyers, who only buy the products 'Made in Karachi'.

Solution.

Solution 1 :

```
SELECT buyer
FROM Purchase ph
WHERE ph.product IN(
SELECT name
FROM Product pd
WHERE pd.maker IN (
SELECT Name
FROM Company C
WHERE C.city = 'Karachi' ))
Except
SELECT buyer
FROM Purchase ph
WHERE ph.product IN(
SELECT name
FROM Product pd
WHERE pd.maker IN (
SELECT Name
FROM Company C
WHERE C.city <> 'Karachi' ))
```

Relational Algebra :

$\pi_{buyer}(\sigma_{c.city=' Karachi' \wedge Pr1.maker \in C.Name \wedge P1.product \in Pr1.name}(Purchase \bowtie Product \bowtie Company)) - \pi_{buyer}(\sigma_{c.city \neq' Karachi' \wedge}$
$Product \bowtie Company))$

Solution 2 :

```
select ph.buyer from Purchase ph
join Product p
```

on p.name = ph.product
join Company c
on c.name = p.maker
except
select ph.buyer from Purchase ph
join Product p
on p.name = ph.product
join Company c
on c.name = p.maker
where c.city ¡¿ 'karachi'

Relational Algebra :

$\rho_{ph.buyer}(\sigma_{c.city \neq' karachi'}(\Pi_{ph.buyer,p.name,c.name}(Purchase \bowtie Product \bowtie Company)))$

□

Problem 7. Name and price of products bought by more than five customers.

Solution.

Solution 1 :

select Purchase.product,Purchase.price
from Purchase
group by Purchase.product,Purchase.price
having COUNT(Purchase.buyer) > 1

Relational Algebra :

$\pi_{product,price}(\sigma_{COUNT(buyer)>1}(Purchase))$

Solution 2 :

select p.product,p1.price
from Purchase p
join Purchase p1
on (p.product = p1.product) and p.price = p1.price
group by p.product,p1.price
having COUNT(p.buyer) > 1

Relational Algebra :

$\rho_{product,price}(\sigma_{COUNT(buyer)>1}(Purchase \bowtie Purchase))$

Solution 3 :

select p.product, p1.price
from Purchase p,Purchase p1
where (p.product = p1.product) and p.price = p1.price
group by p.product,p1.price
having COUNT(p.buyer) > 1

Relational Algebra :

$\rho_{product,price}(_{COUNT(buyer)>1}(P \bowtie P1))$

Solution 4 :

```
select p.product,p1.price
from Purchase p , Purchase p1
where (p.product = p1.product) and p.price = p1.price
group by p.product,p1.price
intersect
select p.product, p1.price
from Purchase p,Purchase p1
where (p.product = p1.product) and p.price = p1.price
group by p.product,p1.price
having COUNT(p.buyer) > 1
```

Relational Algebra :

$\sigma_{product,price}(Purchase \times Purchase)(p.product = p1.product \wedge p.price = p1.price) \wedge \pi_{product,price}(\gamma_{product,price}(Purchase) \geq 2)$

□

Problem 8. List of products that are more expensive that all the products made by same company before 2015.

Solution.

Solution 1 :

```
select * from Product p
where p.cost > (select sum(cost) from Product where year < 2015) and year > 2015
```

Relational Algebra :

$\pi_{p.cost,p.year}(\sigma_{p.cost>(\sigma_{year<2015}(cost))\wedge year>2015}(Product))$

Solution 2 :

```
select * from Product p,Product p1
where p.cost > (select sum(cost) from Product where year < 2015) and p1.year > 2015 and p.name <>
p1.name
```

Relational Algebra :

$\pi_{p.name,p.cost,p1.name,p1.cost}(\sigma_{p.cost>(\sigma_{year<2015}(\sum_{cost}(Product))\wedge p1.year>2015 \wedge p.name \neq p1.name}(Product \bowtie Product_1))$

Solution 3 :

```
select * from Product p
join Product p1
on p.name = p1.name
where p.year > 2015
group by p.year ,p.cost,p1.cost
having p.cost > sum(p1.cost)
```

Relational Algebra :

$\pi_{p.year,p.cost,p1.cost}(\sigma_{p.year>2015 \wedge p.name=p1.name}(Product \times Product)_{p.year,p.cost,p1.costp.cost>\Sigma p1.cost})$

Solution 4 :

```
select * from Product
where year > 2015
except
select sum(cost) from Product
where year < 2015
```

Relational Algebra :

$\pi_{Product}(\sigma_{\mathbf{year}>2015}(Product) - \sigma_{\mathbf{year}<2015}(Product))$

□

Problem 9. List of companies who never sale products with loss.

Solution.

Solution 1 :

```
select distinct c.name
from Company c,Product p, Purchase ph
where c.name = p.maker and p.name = ph.product and Ph.price > p.cost
```

Relational Algebra :

$\pi_{c.name}(Company \bowtie Product \bowtie Purchase) \wedge \sigma_{Ph.price>p.cost}(Product \bowtie Purchase)$

Solution 2 :

```
select distinct c.name
from Company c
join Product p
on c.name = p.maker
join Purchase ph
on p.name = ph.product
where ph.price > p.cost
```

Relational Algebra :

$\pi_{name}(\sigma_{price>cost}(Purchase \bowtie Product \bowtie Company))$

Solution 3 :

```
select distinct c.name
from Company c
join Product p
on c.name = p.maker
join Purchase ph
on p.name = ph.product
```

except
select distinct c.name
from Company c
join Product p
on c.name = p.maker
join Purchase ph
on p.name = ph.product
where ph.price < p.cost

Relational Algebra :

$\rho_{c.name}(\Pi_{c.name}(Company \bowtie_{c.name=p.maker} Product \bowtie_{p.name=ph.product} Purchase) \setminus \Pi_{c.name}(Company \bowtie_{c.name=p.maker} Product \bowtie_{p.name=ph.product} (Purchase \bowtie_{ph.price<p.cost} \phi)))$

Solution 4 :

select distinct c.name
from Company c
join Product p
on c.name = p.maker
join Purchase ph
on p.name = ph.product
intersect
select distinct c.name
from Company c
join Product p
on c.name = p.maker
join Purchase ph
on p.name = ph.product
where ph.price > p.cost

Relational Algebra :

$\pi_{c.name}(Company \bowtie Product \bowtie Purchase \cap \pi_{c.name}(Company \bowtie Product \bowtie Purchase \mid ph.price > p.cost))$

Solution 5 :

select c.name
from Company c
where C.name in (select p.maker from Product p where p.name in (select ph.product from purchase ph where ph.price > p.cost))

Relational Algebra :

$\pi_{c.name}(\sigma_{c.name \in \pi_{p.maker}(\sigma_{p.name \in \pi_{ph.product}(\sigma_{ph.price>p.cost}(Product \bowtie Purchase))})}(Company))$

□

Problem 10. List the products which have more than average revenue in 2015 but below average revenue in 2016

Solution.

This Query can not be entertained because revenue can be generated only from products saled by a company in a certain period of time .but we are provided only with the product launched date but not the date on which product has been saled .thats why this query canot be written □