CS262- Problem Set 1
CS262- Data Base Systems

Muhammad Mudassir 2022-CS-32

February 15, 2024

Problem 1: Products with Cost Above Average

Relational Algebra

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

SQL

Cartesian Product
SELECT name
FROM Product, (SELECT AVG(cost) AS Avg-cost FROM Product) AS Avg-table WHERE cost > Avg-cost;

Join
SELECT name
FROM Product
JOIN (SELECT AVG(cost) AS Avg-cost FROM Product) AS Avg-table
ON cost > Avg-cost;

Subquery
SELECT name
FROM Product
WHERE cost > (SELECT AVG(cost) FROM Product);

Problem 2: Companies whose Products are Bought by Aslam

Relational Algebra

$$\pi_{name}(\sigma_{buyer='Aslam'}(\text{Purchase} \rhd\lhd \text{Company}))$$

SQL

Cartesian Product
SELECT DISTINCT C.name FROM
Company C, Purchase P
WHERE C.name = P.company AND P.buyer = 'Aslam';

Join
SELECT DISTINCT C.name
FROM Company C
JOIN Purchase P ON C.name = P.company
WHERE P.buyer = 'Aslam';

Subquery
SELECT DISTINCT name
FROM Company
WHERE name IN (SELECT company FROM Purchase WHERE buyer = 'Aslam');

Problem 3: Products More Expensive than Unilever's

Relational Algebra

$$\pi_{name}(\sigma_{cost>max\_cost}(\text{Product} \times (\rho_{max\_cost}(\gamma_{max(cost)}(\sigma_{maker='Unilever'}(\text{Product}))))))$$

SQL

Cartesian Product
SELECT name
FROM Product, (SELECT MAX(cost) AS max_cost FROM Product WHERE maker = 'Unilever') AS max_table WHERE cost > max_cost;

Join
SELECT name
FROM Product
JOIN (SELECT MAX(cost) AS max_cost FROM Product WHERE maker = 'Unilever') AS max_table ON cost > max_cost;

Subquery
SELECT name
FROM Product
WHERE cost > (SELECT MAX(cost) FROM Product WHERE maker = 'Unilever');

Problem 4: Copy Cat Products

Relational Algebra

$$\pi_{Product.name,\ Product.maker}(\sigma_{Product.maker='Unilever' \wedge Product.name \in (\pi_{name}(\sigma_{maker='Unilever'}(\text{Product})))}(\text{Product}))$$

SQL

Cartesian Product SELECT
P1.name, P1.maker
FROM Product AS P1, Product AS P2
WHERE P1.maker <> 'Unilever' AND P1.name = P2.name AND P2.maker = 'Unilever';

Join
SELECT P1.name, P1.maker
FROM Product AS P1
JOIN Product AS P2 ON P1.name = P2.name
WHERE P1.maker <> 'Unilever' AND P2.maker = 'Unilever';

Subquery
SELECT name, maker
FROM Product
WHERE maker <> 'Unilever' AND name IN (SELECT name FROM Product WHERE maker = 'Unilever');

Problem 5: Buyers of Products Produced in Lahore

Relational Algebra

$$\pi_{buyer}(\sigma_{city='Lahore'}(\text{Product} \triangleright\triangleleft \text{Purchase}))$$

SQL

Cartesian Product SELECT
DISTINCT buyer

```
FROM Product, Purchase
WHERE Product.name = Purchase.product AND Product.city = 'Lahore';
```

Join

```
SELECT DISTINCT buyer
FROM Product
JOIN Purchase ON Product.name = Purchase.product
WHERE Product.city = 'Lahore';
```

Subquery

```
SELECT DISTINCT buyer
FROM Purchase
WHERE product IN (SELECT name FROM Product WHERE city = 'Lahore');
```

Problem 6: Buyers Who Only Buy Products Made in Karachi

Relational Algebra

$$\pi_{buyer}(\sigma_{city='Karachi'}(Product \triangleright\triangleleft Purchase) - \pi_{buyer}(\sigma_{city\neq'Karachi'}(Product \triangleright\triangleleft Purchase)))$$

SQL

```
Cartesian Product SELECT
DISTINCT P1.buyer
FROM Product AS P1, Product AS P2, Purchase
WHERE P1.name = Purchase.product AND P2.name = Purchase.product AND P1.city = 'Karachi' AND P2.city
```

Join

```
SELECT DISTINCT P1.buyer
FROM Product AS P1
JOIN Product AS P2 ON P1.name = P2.name
JOIN Purchase ON P1.name = Purchase.product
WHERE P1.city = 'Karachi' AND P2.city <> 'Karachi';
```

Subquery

```
SELECT DISTINCT buyer
FROM Purchase
WHERE product IN (SELECT name FROM Product WHERE city = 'Karachi')
AND buyer NOT IN (SELECT buyer FROM Purchase WHERE product IN (SELECT name FROM Product WHERE city <
```

Problem 7: Products Bought by More than Five Customers

Relational Algebra

$$\pi_{name,\ price}(\sigma_{count>5}(\gamma_{name,\ price,\ count}(Purchase)))$$

SQL

```
--  Cartesian Product
SELECT name, price
FROM Purchase GROUP
BY name, price HAVING
COUNT(*) > 5;
```

Join

```
SELECT name, price
FROM Purchase AS P
JOIN (
```

```
    SELECT product, COUNT(*) AS cnt
    FROM Purchase
    GROUP BY product
    HAVING COUNT(*) > 5
) AS P_cnt ON P.product = P_cnt.product;
```

Subquery

SELECT name

\begin{verbatim} ,

price

FROM Purchase
WHERE product IN (
SELECT product FROM
Purchase GROUP BY
product HAVING
COUNT(*) > 5);

Problem 8: Products More Expensive than Previous Years

Relational Algebra

name (Product Product.name (Product.year ¡ 2015 (Product) Product)) name (Product Product.name (Product.year¡2015 (Product)Product))

SQL

Cartesian Product
SELECT DISTINCT P1.name
FROM Product AS P1, Product AS P2
WHERE P1.name = P2.name AND P1.year >= 2015 AND P2.year < 2015 AND P1.cost > P2.cost;
Join
SELECT DISTINCT P1.name
FROM Product AS P1
JOIN Product AS P2 ON P1.name = P2.name AND P1.year >= 2015 AND P2.year < 2015 AND P1.cost > P2.cost

Subquery
SELECT DISTINCT P1.name
FROM Product AS P1
WHERE P1.year >= 2015 AND P1.cost > ALL (SELECT cost FROM Product WHERE name = P1.name AND year < 20

Problem 9: Companies Never Selling at Loss

Relational Algebra

name (Company Company.name (Purchase.price ¡ Product.cost (Purchase Product Company))
) name (Company Company.name (Purchase.price¡Product.cost (PurchaseProductCompany)))

SQL

Cartesian Product SELECT
DISTINCT C.name
FROM Company AS C, Purchase, Product
WHERE C.name = Purchase.buyer AND Purchase.product = Product.name AND Product.maker = C.name AND Pur

Join
SELECT DISTINCT C.name

```
FROM Company AS C
JOIN Purchase ON C.name = Purchase.buyer
JOIN Product ON Purchase.product = Product.name AND Product.maker = C.name
WHERE Purchase.price >= Product.cost;
```

Subquery

```
SELECT DISTINCT name
FROM Company
WHERE name NOT IN (
SELECT DISTINCT C.name
FROM Company AS C
JOIN Purchase ON C.name = Purchase.buyer
JOIN Product ON Purchase.product = Product.name AND Product.maker = C.name
WHERE Purchase.price < Product.cost
);
```

Problem 10: Products with More than Average Revenue in 2015 and Below Average Revenue in 2016

Relational Algebra

$\pi_{\text{product\_name}}(\sigma_{\text{revenue\_2015}>\text{avg\_revenue\_2015} \wedge \text{revenue\_2016}<\text{avg\_revenue\_2016}}(\text{Products} \bowtie_{\text{product\_id=product\_id}} \text{Sales}))$

SQL

```
Cartesian Product SELECT
DISTINCT P.name FROM Product
AS P, Purchase
WHERE P.name = Purchase.product AND P.year = 2015 AND Purchase.year = 2015 GROUP
BY P.name
HAVING SUM(Purchase.price) > (
SELECT AVG(revenue_2015)
FROM (
SELECT SUM(price) AS revenue_2015
FROM Purchase
WHERE year = 2015
GROUP BY product )
AS avg_table
)
AND SUM(Purchase.price) < (
SELECT AVG(revenue_2016)
FROM (
SELECT SUM(price) AS revenue_2016
FROM Purchase
WHERE year = 2016
GROUP BY product )
AS avg_table );
```

Join

```
SELECT DISTINCT P.name
FROM Product AS P
JOIN (
SELECT product, SUM(price) AS revenue_2015
```

```
FROM Purchase
WHERE year = 2015
GROUP BY product
) AS revenue_2015 ON P.name = revenue_2015.product
JOIN (
SELECT product, SUM(price) AS revenue_2016
FROM Purchase
WHERE year = 2016
GROUP BY product
) AS revenue_2016 ON P.name = revenue_2016.product
WHERE revenue_2015 > (
SELECT AVG(revenue_2015)
FROM (
SELECT SUM(price) AS revenue_2015
FROM Purchase
WHERE year = 2015
GROUP BY product
) AS avg_table
)
AND revenue_2016 < ( SELECT
AVG(revenue_2016) FROM (
SELECT SUM(price) AS revenue_2016
FROM Purchase
WHERE year = 2016
GROUP BY product
) AS avg_table
);
```

Subquery

```
SELECT DISTINCT name
FROM Product
WHERE name IN (
SELECT product
FROM Purchase
WHERE year = 2015
GROUP BY product
HAVING SUM(price) > (
SELECT AVG(revenue_2015)
FROM (
SELECT SUM(price) AS revenue_2015
FROM Purchase
WHERE year = 2015
GROUP BY product
) AS avg_table
)
)
AND name IN ( SELECT
product FROM Purchase
WHERE year = 2016 GROUP
BY product HAVING
SUM(price) < ( SELECT
AVG(revenue_2016) FROM (
SELECT SUM(price) AS revenue_2016
FROM Purchase
WHERE year = 2016
```

```
GROUP BY product
)  AS avg_table
)
);
```