CS262- Problem Set 1

CS262- Database Systems
RegNo 2022-CS-48 — Khuram Iqbal

February 7, 2024

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.

[margin=1in]geometry

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

Solution. **Relational Algebra:**

$$\pi_{\text{name}}\left(\sigma_{\text{cost}>\text{avg\_cost}}(\text{Product} \bowtie_{\text{name}=\text{product}} (\pi_{\text{product,avg\_cost}}(\gamma_{\text{avg\_cost}}(\text{Product}))))\right)$$

SQL Solutions:

1. Using Cartesian Product:

```
SELECT DISTINCT p.name FROM Product p, (SELECT AVG(cost) as     avg_cost FROM Product)
avg_p WHERE p.cost > avg_p.avg_cost;
```

2. Using Joins:

```
SELECT DISTINCT p.name
FROM Product p
JOIN (SELECT AVG(cost) as avg_cost FROM Product) avg_p
ON p.cost > avg_p.avg_cost;
```

3. Using Subquery:

```
SELECT DISTINCT name
FROM Product
WHERE cost > (SELECT AVG(cost) FROM Product);
```

4. Solution num: 4

```
SELECT DISTINCT p1.name
FROM Product p1
JOIN Product p2 ON p1.name = p2.name AND p2.cost > (SELECT AVG(cost) FROM Product);
```

5. Solution num: 5

```
SELECT DISTINCT p1.name FROM Product p1 WHERE p1.name IN
(SELECT name FROM Product WHERE cost > (SELECT AVG(cost) FROM Product)
```

□

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

Solution. **Relational Algebra:**

$$\pi_{\text{name}}\left(\sigma_{\text{buyer='Aslam'}}(\text{Purchase} \bowtie_{\text{product=name}} \text{Product})\right)$$

SQL Solutions:

1. Using WHERE Conditions:

```
SELECT DISTINCT Company.name
FROM Company, Product, Purchase
WHERE Company.name = Product.maker
  AND Product.name = Purchase.product
  AND Purchase.buyer = 'Aslam';
```

2. Using Cartesian Product:

```
SELECT DISTINCT c.name
FROM Company c, Product p, Purchase pu
WHERE p.name = pu.product
  AND c.name = p.maker
  AND pu.buyer = 'Aslam';
```

3. Using Joins:

```
SELECT DISTINCT c.name
FROM Company c
JOIN Product p ON c.name = p.maker
JOIN Purchase pu ON p.name = pu.product
WHERE pu.buyer = 'Aslam';
```

4. Using Subquery:

```
SELECT DISTINCT c.name
FROM Company c
WHERE c.name IN (SELECT p.maker
                 FROM Product p
                 JOIN Purchase pu ON p.name = pu.product
                 WHERE pu.buyer = 'Aslam');
```

[margin=1in]geometry

5. Solution: 5

```
SELECT DISTINCT Company.name
FROM Company, Product, Purchase
WHERE Company.name = Product.maker
  AND Product.name = Purchase.product
  AND Purchase.buyer = 'Aslam';
```

□

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

Solution. **Relational Algebra:**

$$\pi_{\text{name}}\left(\text{Product} - \pi_{\text{name}}\left(\sigma_{\text{maker='Unilever'}}(\text{Product})\right)\right)$$

SQL Solutions:

1. Using Cartesian Product:

```
SELECT DISTINCT p1.name
FROM Product p1, Product p2
WHERE p1.cost > p2.cost
  AND p2.maker = 'Unilever';
```

2. Using Joins:

```
SELECT DISTINCT p1.name
FROM Product p1
JOIN Product p2 ON p1.cost > p2.cost
WHERE p2.maker = 'Unilever';
```

3. Using Join and Subquery:

```
SELECT DISTINCT Product.name
FROM Product
WHERE Product.cost > ALL (
    SELECT cost
    FROM Product, Company
    WHERE Product.maker = Company.name
      AND Company.name = 'Unilever'
);
```

4. Using Max nad Join:

```
SELECT DISTINCT Product.name
FROM Product, Company
WHERE Product.maker = Company.name
  AND Company.name = 'Unilever'
  AND Product.cost > (
      SELECT MAX(cost)
      FROM Product, Company
      WHERE Product.maker = Company.name
        AND Company.name = 'Unilever'
  );
```

5. Using Subquery:

```
SELECT DISTINCT name
FROM Product p1
WHERE cost > ALL (SELECT cost
                  FROM Product p2
                  WHERE p2.maker = 'Unilever');
```

□

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

Solution. **Relational Algebra:**

$$\pi_{\text{Product.name, maker}} \left(\sigma_{\text{Product.name=UnileverProducts.name}} \left(\text{Product} \bowtie \rho_{\text{UnileverProducts(name,'Unilever')}}(\text{Product})\right)\right)$$

SQL Solutions:

1. Using Cartesian Product:

```
SELECT DISTINCT p1.name, p1.maker
FROM Product p1, Product p2
WHERE p1.name = p2.name
  AND p2.maker = 'Unilever';
```

2. Using Joins:

```
SELECT DISTINCT p1.name, p1.maker
FROM Product p1
JOIN Product p2 ON p1.name = p2.name
WHERE p2.maker = 'Unilever';
```

3. Using Subquery:

```
SELECT DISTINCT name, maker
FROM Product
WHERE name IN (SELECT name FROM Product WHERE maker = 'Unilever');
```

4. Using Simple Join:

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

5. Using Join with NOT EXISTS:

```
SELECT Product.name, Product.maker
FROM Product, Company
WHERE Product.maker = Company.name
  AND Company.name = 'Unilever'
  AND NOT EXISTS (
      SELECT *
      FROM Product, Company
      WHERE Product.maker = Company.name
        AND Company.name <> 'Unilever'
        AND Product.name = Product.name
  );
```

□

Problem 5. Buyers of products produced in Lahore.

Solution. textbfRelational Algebra:

$$\pi_{\text{Purchase.buyer}} \left(\sigma_{\text{Product.maker='Lahore'}} \left(\text{Product} \bowtie \text{Purchase} \right) \right)$$

SQL Solutions:

1. Using Cartesian Product:

```
SELECT DISTINCT pu.buyer
FROM Product pr, Purchase pu
WHERE pr.maker = 'Lahore'
  AND pr.name = pu.product;
```

2. Using Joins:

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

3. Using Simple Join:

```
SELECT DISTINCT Purchase.buyer
FROM Purchase, Product, Company
WHERE Purchase.product = Product.name
  AND Product.maker = Company.name
  AND Company.city = 'Lahore';
```

4. Using Simple Join with IN:

```
SELECT DISTINCT buyer
FROM Purchase
WHERE product IN (
      SELECT Product.name
      FROM Product, Company
      WHERE Product.maker = Company.name
        AND Company.city = 'Lahore'
  );
```

5. Using Subquery:

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

□

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

Solution. **Relational Algebra:**

$$\pi_{\text{buyer}}\left(\sigma_{\text{product\_city}='Karachi' \wedge \text{count\_distinct\_products}=1}(\text{Purchase} \bowtie_{\text{product}=\text{name}} (\gamma_{\text{count\_distinct\_products}}(\text{Product})))\right)$$

SQL Solutions:

1. Using Cartesian Product:

```
SELECT DISTINCT pu.buyer
FROM Purchase pu, (SELECT product, COUNT(DISTINCT name) as count_distinct_products
                   FROM Product WHERE maker = 'Karachi' GROUP BY product) prod_khi
WHERE pu.product = prod_khi.product AND prod_khi.count_distinct_products = 1;
```

2. Using Joins:

```
SELECT DISTINCT pu.buyer
FROM Purchase pu
JOIN (SELECT product, COUNT(DISTINCT name) as count_distinct_products
      FROM Product WHERE maker = 'Karachi' GROUP BY product) prod_khi
ON pu.product = prod_khi.product AND prod_khi.count_distinct_products = 1;
```

3. Using Simple Join:

```
SELECT DISTINCT P1.buyer
FROM Purchase P1
WHERE P1.product IN (
      SELECT DISTINCT Product.name
      FROM Product, Company
      WHERE Product.maker = Company.name
        AND Company.city = 'Karachi'
  )
  AND NOT EXISTS (
      SELECT P2.buyer
      FROM Purchase P2
      WHERE P2.buyer = P1.buyer
        AND P2.product NOT IN (
            SELECT Product.name
            FROM Product, Company
            WHERE Product.maker = Company.name
              AND Company.city = 'Karachi'
        )
  );
```

4. Using NOT IN:

```
SELECT DISTINCT buyer
FROM Purchase
WHERE buyer NOT IN (
      SELECT DISTINCT P.buyer
      FROM Purchase P, Product, Company
      WHERE P.product = Product.name
        AND Product.maker = Company.name
        AND Company.city = 'Karachi'
        AND P.product NOT IN (
            SELECT DISTINCT Product.name
            FROM Product, Company
            WHERE Product.maker = Company.name
              AND Company.city != 'Karachi'
        )
  );
```

5. Using Subquery:

```
SELECT DISTINCT buyer
FROM Purchase
WHERE product IN (SELECT product
                  FROM Product
                  WHERE maker = 'Karachi'
                  GROUP BY product
                  HAVING COUNT(DISTINCT name) = 1);
```

□

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

Solution. **Relational Algebra:**

$$\pi_{\text{name, price}}\left(\sigma_{\text{count\_distinct\_buyers}>5}(\text{Purchase} \bowtie_{\text{product=name}} (\gamma_{\text{count\_distinct\_buyers}}(\text{Purchase})))\right)$$

SQL Solutions:

1. Using Cartesian Product:

```
SELECT DISTINCT pr.name, pr.price
FROM Purchase pu, (SELECT product, COUNT(DISTINCT buyer) as count_distinct_buyers
                   FROM Purchase GROUP BY product) pu_count
JOIN Product pr ON pr.name = pu_count.product
WHERE pu.product = pu_count.product AND pu_count.count_distinct_buyers > 5;
```

2. Using COUNT and GROUP BY:

```
SELECT Product.name, Purchase.price
FROM Product
JOIN Purchase ON Product.name = Purchase.product
WHERE Purchase.product IN (
      SELECT P.product
      FROM Purchase P
      GROUP BY P.product
      HAVING COUNT(DISTINCT P.buyer) > 5
  );
```

3. Using Subquery:

```
SELECT name, price
FROM Product
WHERE name IN (
      SELECT P.product
      FROM Purchase P
      GROUP BY P.product
      HAVING COUNT(DISTINCT P.buyer) > 5
  );
```

4. Using Joins:

```
SELECT DISTINCT pr.name, pr.price
FROM Purchase pu
JOIN (SELECT product, COUNT(DISTINCT buyer) as count_distinct_buyers
      FROM Purchase GROUP BY product) pu_count
ON pu.product = pu_count.product AND pu_count.count_distinct_buyers > 5
JOIN Product pr ON pr.name = pu_count.product;
```

5. Using Subquery:

```
SELECT DISTINCT pr.name, pr.price
FROM Product pr
WHERE pr.name IN (SELECT pu.product
```

```
FROM Purchase pu
GROUP BY pu.product
HAVING COUNT(DISTINCT pu.buyer) > 5);
```

□

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

Solution. **Relational Algebra:**

$$\pi_{\text{name}}\left(\sigma_{\text{price}>\text{max\_price\_before\_2015}}(\text{Product} \bowtie_{\text{name}=\text{product}} \left(\gamma_{\text{max\_price\_before\_2015}}(\sigma_{\text{year}<2015}(\text{Product}))\right)\right)\right)$$

SQL Solutions:

1. Using Cartesian Product:

```
SELECT DISTINCT p.name
FROM Product p, (SELECT product, MAX(price) as max_price_before_2015
                 FROM Product
                 WHERE year < 2015
                 GROUP BY product) max_p
WHERE p.name = max_p.product AND p.price > max_p.max_price_before_2015;
```

2. Using Joins:

```
SELECT DISTINCT p.name
FROM Product p
JOIN (SELECT product, MAX(price) as max_price_before_2015
      FROM Product
      WHERE year < 2015
      GROUP BY product) max_p
ON p.name = max_p.product AND p.price > max_p.max_price_before_2015;
```

3. Using NOT EXISTS:

```
SELECT p1.name
FROM Product p1
WHERE p1.cost > ALL (
      SELECT p2.cost
      FROM Product p2
      WHERE p2.maker = p1.maker AND p2.year < 2015
  );
```

4. Using Subquery:

```
SELECT name
FROM Product
WHERE cost > ALL (
      SELECT cost
      FROM Product
      WHERE maker = Product.maker AND year < 2015
  );
```

5. Subquery:

```
SELECT DISTINCT name
FROM Product p
WHERE price > (SELECT MAX(price)
               FROM Product
               WHERE year < 2015
               GROUP BY product);
```

□

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

Solution. **Relational Algebra:**

$$\pi_{\text{maker}}\left(\text{Product} - \pi_{\text{maker}}\left(\sigma_{\text{price}<\text{cost}}(\text{Product})\right)\right)$$

SQL Solutions:

1. Using Cartesian Product:

```
SELECT DISTINCT p.maker
FROM Product p
WHERE NOT EXISTS (SELECT 1
                  FROM Product
                  WHERE maker = p.maker AND price < cost);
```

2. Using Joins:

```
SELECT DISTINCT p.maker
FROM Product p
LEFT JOIN (SELECT maker, 1 as loss
           FROM Product
           WHERE price < cost) loss_p
ON p.maker = loss_p.maker
WHERE loss_p.maker IS NULL;
```

3. Using NOT EXISTS:

```
SELECT DISTINCT maker
FROM Product p1
WHERE NOT EXISTS (
      SELECT *
      FROM Purchase p2
      WHERE p2.product = p1.name AND p2.price < p1.cost
  );
```

4. Using LEFT Join:

```
SELECT DISTINCT p.maker
FROM Product p
LEFT JOIN Purchase pu ON p.name = pu.product AND pu.price < p.cost
WHERE pu.id IS NULL;
```

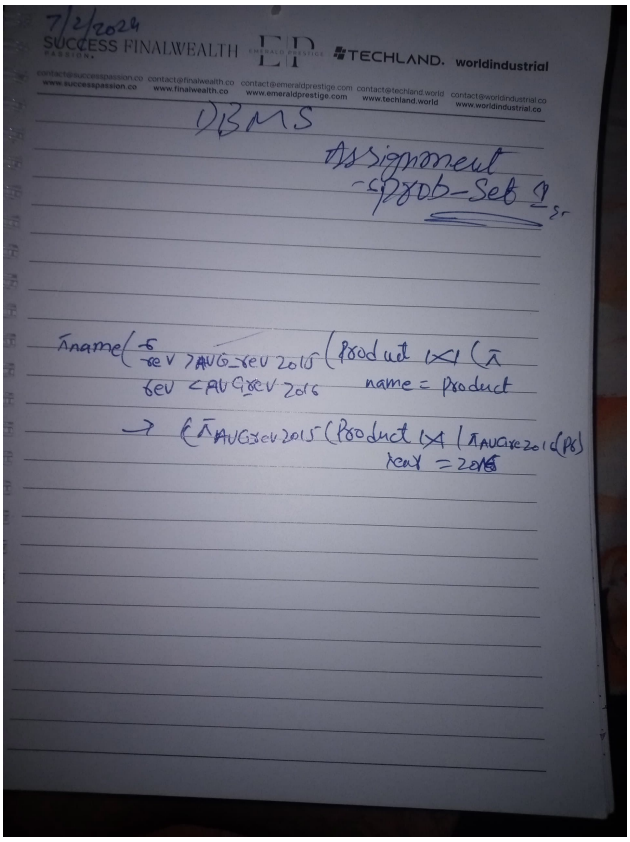

Figure 1: Due to unstable margins

5. Using Subquery:

```
SELECT DISTINCT maker
FROM Product
WHERE maker NOT IN (SELECT maker
                    FROM Product
                    WHERE price < cost);
```

□

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

Solution. **Relational Algebra:**

$$\pi_{\text{name}}\,(\sigma_{\text{revenue}>\text{avg\_revenue\_2015}\wedge\text{revenue}<\text{avg\_revenue\_2016}}(\text{Product} \bowtie_{\text{name=product}} (\pi_{\text{product,}}$$

$\text{revenue}(\gamma_{\text{avg\_revenue\_2015}}(\text{Product}) \bowtie_{\text{year=2016}} \gamma_{\text{avg\_revenue\_2016}}(\text{Product}))))$

SQL Solutions:

1. Using Cartesian Product:

```
SELECT DISTINCT p.name
FROM Product p,
     (SELECT AVG(revenue) as avg_revenue_2015
```

```
      FROM Product
      WHERE year = 2015) avg_2015,
     (SELECT AVG(revenue) as avg_revenue_2016
      FROM Product
      WHERE year = 2016) avg_2016
WHERE p.revenue > avg_2015.avg_revenue_2015
  AND p.revenue < avg_2016.avg_revenue_2016;
```

2. Using Joins:

```
SELECT DISTINCT p.name
FROM Product p
JOIN (SELECT AVG(revenue) as avg_revenue_2015
      FROM Product
      WHERE year = 2015) avg_2015
ON p.year = 2015
JOIN (SELECT AVG(revenue) as avg_revenue_2016
      FROM Product
      WHERE year = 2016) avg_2016
ON p.year = 2016
WHERE p.revenue > avg_2015.avg_revenue_2015
  AND p.revenue < avg_2016.avg_revenue_2016;
```

3. Using Subquery:

```
SELECT DISTINCT name
FROM Product
WHERE revenue > (SELECT AVG(revenue)
                 FROM Product
                 WHERE year = 2015)
  AND revenue < (SELECT AVG(revenue)
                 FROM Product
                 WHERE year = 2016);
```

4. Using FULL JOIN:

```
SELECT DISTINCT p.name
FROM Product p
FULL JOIN Purchase pu ON p.name = pu.product
WHERE pu.price > (SELECT AVG(pu1.price) FROM Purchase pu1 WHERE YEAR(pu1.date) = 2015)
AND pu.price < (SELECT AVG(pu2.price) FROM Purchase pu2 WHERE YEAR(pu2.date) = 2016);
```

□