In-class exercises

Week 2023-03

SQL exercises

Given these sample tables:

```
county(no, name)
Primary Key no

dealer(id, city, county_no)
Primary Key id
Foreign Key county_no REFERENCES county(no)

car(id, make, model, year, mileage, fuel, type, price, dealer_id)
Primary Key id
Foreign Key dealer_id REFERENCES dealer(id)
```

Q1:

- What is a SQL equivalent to:

  $\sigma_{\texttt{model\_year < 2010 OR mileage > 50000}}(\texttt{car})$?

Solution:

```
SELECT *
FROM car
WHERE model_year < 2010 OR mileage > 50000
```

Q2:

- What is a SQL equivalent to:

  $\Pi_{\texttt{id, make, model, type, price}}(\sigma_{\texttt{type IN ('station wagon', 'suv') AND price BETWEEN 300000 AND 450000}}(\texttt{car}))$

Solution:

```
SELECT id, make, model, type, price
FROM car
WHERE type IN ('station wagon', 'suv')
      AND price BETWEEN 300000 AND 450000
```

Q3:

- What is a SQL equivalent to:

  $\Pi_{\texttt{city, make, model}}(\sigma_{\texttt{fuel <> 'diesel'}}(\texttt{car} \bowtie_{\texttt{dealer\_id = dealer.id}} \texttt{dealer}))$?

Solution:

```
SELECT city, make, model
FROM car
INNER JOIN dealer ON dealer_id = dealer.id
WHERE fuel <> 'diesel'
```

Q4:

- What is the relational algebra expression equivalent to:

```
SELECT id, make, model, model_year, comment
FROM car
WHERE comment IS NOT NULL
```

Solution:

$\Pi_{\text{id, make, model, model\_year, comment}}(\sigma_{\text{comment IS NOT NULL}}(\text{car}))$?

Q5:

- Write a SQL statement that will return the name of the county, the city of the dealer, the id, model_year, mileage, and comment for Volkswagen Passats where the mileage is no more than 45000. Order the result on name of county, city of dealer, model_year (descending), and mileage (descending).

Solution:

```
SELECT name, city, car.id, model_year, mileage, comment
FROM car
INNER JOIN dealer on dealer_id = dealer.id
INNER JOIN county on county_no = no
WHERE make = 'Volkswagen' AND model = 'Passat'
      AND mileage < 45000
ORDER BY name, city, model_year DESC, mileage DESC
```

Q6:

- Write a SQL statement that will return the dealer city, id, make, model, age, and average mileage per year of cars for sale in the cities of Gjøvik, Hamar, and Lillehammer. The column showing the age of the car should be named age, the column showing the average mileage should be named yearly_mileage.

Solution:

```
SELECT city, car.id, make, model, 2021 – model_year AS age,
       mileage / (2021 – model_year) AS yearly_mileage
FROM car
INNER JOIN dealer ON dealer_id = dealer.id
WHERE city IN ('Gjøvik', 'Hamar', 'Lillehammer')
```

Q7:

- Write a SQL statement that returns all information about cars having been used for demo purposes (i.e., the comment includes demo – case insensitive).

Solution:

```
SELECT *
FROM car
WHERE LOWER(comment) LIKE '%demo%'
```

Q8:

- Write a SQL statement that returns dealer city, make, model, and model_year for cars newer than 2015. Duplicates should be removed.

Solution:

```
SELECT DISTINCT city, make, model, model_year
FROM car
INNER JOIN dealer ON dealer_id = dealer.id
WHERE model_year > 2015
```

Q9:

- What is a SQL equivalent to:
  $\Pi_{\text{city, make, model}}(\text{car} \bowtie_{\text{dealer\_id = dealer.id}} \text{dealer})$?

Solution:

```
SELECT city, make, model
FROM car
RIGHT OUTER JOIN dealer ON dealer_id = dealer.id
```

Q10:

- What is the relational algebra expression equivalent to:

```
SELECT city, car.id, make
FROM dealer LEFT OUTER JOIN car ON dealer_id = dealer.id
WHERE city IN ('Jessheim', 'Elverum', 'Kongsvinger', 'Otta')
```

Solution:

$$\Pi_{\text{city, car.id, make}}(\sigma_{\text{city IN ('Jessheim', 'Elverum', 'Kongsvinger', 'Otta')}}(\text{dealer} \bowtie_{\text{dealer\_id = dealer.id}} \text{car}))$$

Q11:

- Write a SQL statement that returns county name, city name, model year for cars for sale. Duplicates should be removed. All dealers should be listed, even when there are no cars for sale at the dealer.

Solution:

```
SELECT DISTINCT name, city, model_year
FROM dealer
LEFT OUTER JOIN car on dealer_id = dealer.id
INNER JOIN county on county_no = no
```

Q12:

- What is a SQL equivalent to:

$$\gamma_{\text{MIN mileage, MAX mileage}}(\text{car} \bowtie_{\text{dealer\_id = dealer.id}} \text{dealer} \bowtie_{\text{county\_no = no}} \sigma_{\text{name = 'Innlandet'}}(\text{county}))$$

Solution:

```
SELECT MIN(mileage), MAX(mileage)
FROM car
INNER JOIN dealer ON dealer_id = dealer.id
INNER JOIN county ON county_no = no
WHERE name = 'Innlandet'
```

Q13:

- Write a SQL statement that returns the number of dealers that have Volkswagen cars for sale.

Solution:

```
SELECT COUNT(DISTINCT dealer_no)
FROM car
WHERE make = 'Volkswagen'
```

Q14:

- What is a SQL equivalent to:

$$\gamma_{\text{city, make SUM price}}(\text{car} \bowtie_{\text{dealer\_id = dealer.id}} \text{dealer})$$

Solution:

```
SELECT city, make, SUM(price)
FROM car
INNER JOIN dealer ON dealer_id = dealer.id
GROUP BY city, make
```

Q15:

- What is a SQL equivalent to:

$$\sigma_{\text{SUM price > 750000}}(\gamma_{\text{city, make SUM price}}(\text{car} \bowtie_{\text{dealer\_id = dealer.id}} \text{dealer}))$$

Solution:

```
SELECT city, make, SUM(price)
FROM car
INNER JOIN dealer ON dealer_id = dealer.id
GROUP BY city, make
HAVING SUM(price) > 750000
```

Q16:

- What is the relational algebra expression equivalent to:

```
SELECT fuel, AVG(price)
```

```
FROM car
GROUP BY fuel
```

- *Solution:*

$$_{\texttt{fuel AVG price}}(\texttt{car})$$

Q17:

- Write a SQL statement that returns the number of cars for each combination of make and fuel. Order the results on number of cars in descending order, then make and fuel in alphabetic order.

Solution:

```
SELECT make, fuel, COUNT(*) FROM car GROUP BY make, fuel ORDER BY
COUNT(*) DESC, make ASC, fuel ASC;
```

Q18:

- Write a SQL statement that returns the number of cars of each make for sale in each county, but only for makes for which there are more than two cars for sale in that county.

Solution:

```
SELECT name, make, COUNT(*)
FROM car
INNER JOIN dealer ON dealer_id = dealer.id
INNER JOIN county ON county_no = no
GROUP BY no, make
HAVING COUNT(*) > 2
```

Q19:

- What is a SQL equivalent to:

$$\Pi_{\texttt{city, make}}(\texttt{dealer} \bowtie_{\texttt{dealer\_id = dealer.id}} \sigma_{\texttt{model\_year IN (2019, 2020)}}(\texttt{car})$$

Solution:

```
SELECT DISTINCT city, make
FROM dealer
LEFT OUTER JOIN (
   SELECT *
   FROM car
   WHERE model_year IN (2019, 2020)) AS T ON dealer_id = dealer.id
```

Q20:

- Write a SQL statement that lists the counties that don't have Volkswagen Passat cars.

Solution:

```
SELECT name
FROM county
WHERE no NOT IN (
   SELECT DISTINCT county_no
   FROM dealer
   INNER JOIN car ON dealer.id = dealer_id
   WHERE (make, model) = ('Volkswagen', 'Passat'))
```

Or (requiring less work for the DBMS):

```
SELECT name
FROM county
WHERE NOT EXISTS (
   SELECT *
   FROM dealer
   INNER JOIN car ON dealer.id = dealer_id
   WHERE county_no = no
   AND (make, model) = ('Volkswagen', 'Passat'))
```

Q21:

- Write a SQL statement that lists the number of dealers per county, but only for the counties having more than average number of dealers among them.

Solution:

```
SELECT name, COUNT(id)
FROM county
INNER JOIN dealer ON no = county_no
GROUP BY no
HAVING COUNT(id) > (
   SELECT AVG(cnt)
   FROM (
       SELECT COUNT(id) AS cnt
       FROM dealer
       GROUP BY county_no
       ) AS T
    )
```