Chapter 3 - Tutorial

SQL

COMP3278C
Introduction to Database Management Systems

Dr. CHEN, Yi
Email: chenyi1@hku.hk

School of Computing & Data Science, The University of Hong Kong

Acknowledgement: Dr. Chui Chun Kit, Dr. Reynold Cheng, Dr. Ping Luo

Learning Outcomes

- Outcome 1: **Information Modeling**
  - Able to understand the modeling of real-life information in database systems.
- Outcome 2: **Query Languages**
  - Able to use the languages designed for data access.
- Outcome 3: **System Design**
  - Able to design an efficient and reliable database system.
- Outcome 4: **Application Development**
  - Able to implement a practical application on a real database.

A database of train stations

customer

customer_id	name
1	Alan
2	Bob
3	Cindy
4	David

card

card_id	balance
1	-10
2	-4
3	30
4	-2
5	-1
6	10

owner

customer_id	card_id
1	1
1	2
2	3
2	4
3	6
4	5

station

station_id	name
1	Wan Chai
2	Central
3	Admiralty

latest_usage

customer_id	card_id	station_id	latest_expense	latest_date
1	1	2	12	2025-02-25
1	2	2	8	2025-02-25
2	3	3	4	2025-02-26
2	4	3	6	2025-02-24
3	6	3	4	2025-02-24
4	5	2	12	2025-02-24
4	5	1	10	2025-02-24
4	5	1	12	2025-02-24

A database of train stations

customer

customer_id	name
1	Alan
2	Bob
3	Cindy
4	David

card

card_id	balance
1	-10
2	-4
3	30
4	-2
5	-1
6	10

owner

customer_id	card_id
1	1
1	2
2	3
2	4
3	6
4	5

station

station_id	name
1	Wan Chai
2	Central
3	Admiralty

latest_usage

customer_id	card_id	station_id	latest_expense	latest_date
1	1	2	12	2025-02-25
1	2	2	8	2025-02-25
2	3	3	4	2025-02-26
2	4	3	6	2025-02-24
3	6	3	4	2025-02-24
4	5	2	12	2025-02-24
4	5	1	10	2025-02-24
4	5	1	12	2025-02-24

- Q1: Find the customer who has the maximum number of cards
- Return: customer_id

A database of train stations

customer

customer_id	name
1	Alan
2	Bob
3	Cindy
4	David

card

card_id	balance
1	-10
2	-4
3	30
4	-2
5	-1
6	10

owner

customer_id	card_id
1	1
1	2
2	3
2	4
3	6
4	5

station

station_id	name
1	Wan Chai
2	Central
3	Admiralty

latest_usage

customer_id	card_id	station_id	latest_expense	latest_date
1	1	2	12	2025-02-25
1	2	2	8	2025-02-25
2	3	3	4	2025-02-26
2	4	3	6	2025-02-24
3	6	3	4	2025-02-24
4	5	2	12	2025-02-24
4	5	1	10	2025-02-24
4	5	1	12	2025-02-24

- Q1: Find the customer who has the maximum number of cards
- Return: customer_id

- SQL 1

```
SELECT customer_id
FROM owner
GROUP BY customer_id
HAVING count(*) >= ALL(
    SELECT count(*)
    FROM owner
    GROUP BY customer_id
);
```

A database of train stations

customer

customer_id	name
1	Alan
2	Bob
3	Cindy
4	David

card

card_id	balance
1	-10
2	-4
3	30
4	-2
5	-1
6	10

owner

customer_id	card_id
1	1
1	2
2	3
2	4
3	6
4	5

station

station_id	name
1	Wan Chai
2	Central
3	Admiralty

latest_usage

customer_id	card_id	station_id	latest_expense	latest_date
1	1	2	12	2025-02-25
1	2	2	8	2025-02-25
2	3	3	4	2025-02-26
2	4	3	6	2025-02-24
3	6	3	4	2025-02-24
4	5	2	12	2025-02-24
4	5	1	10	2025-02-24
4	5	1	12	2025-02-24

- Q1: Find the customer who has the maximum number of cards
- Return: customer_id

- SQL 2

```
select customer_id
from owner
group by customer_id
having count(*) = (
select max(number_of_cards)
from (
      select count(*) as number_of_cards
      from owner
      group by customer_id
      ) as T
);
```

A database of train stations

customer

customer_id	name
1	Alan
2	Bob
3	Cindy
4	David

card

card_id	balance
1	-10
2	-4
3	30
4	-2
5	-1
6	10

owner

customer_id	card_id
1	1
1	2
2	3
2	4
3	6
4	5

station

station_id	name
1	Wan Chai
2	Central
3	Admiralty

latest_usage

customer_id	card_id	station_id	latest_expense	latest_date
1	1	2	12	2025-02-25
1	2	2	8	2025-02-25
2	3	3	4	2025-02-26
2	4	3	6	2025-02-24
3	6	3	4	2025-02-24
4	5	2	12	2025-02-24
4	5	1	10	2025-02-24
4	5	1	12	2025-02-24

- Q2: Find the card that has the maximum balance.
- Return: card_id, card balance

A database of train stations

customer

customer_id	name
1	Alan
2	Bob
3	Cindy
4	David

card

card_id	balance
1	-10
2	-4
3	30
4	-2
5	-1
6	10

owner

customer_id	card_id
1	1
1	2
2	3
2	4
3	6
4	5

station

station_id	name
1	Wan Chai
2	Central
3	Admiralty

latest_usage

customer_id	card_id	station_id	latest_expense	latest_date
1	1	2	12	2025-02-25
1	2	2	8	2025-02-25
2	3	3	4	2025-02-26
2	4	3	6	2025-02-24
3	6	3	4	2025-02-24
4	5	2	12	2025-02-24
4	5	1	10	2025-02-24
4	5	1	12	2025-02-24

- Q2: Find the card that has the maximum balance.
- Return: card_id, card balance

- SQL 1

```
SELECT card_id, balance
FROM card
WHERE balance >= ALL(
    SELECT MAX(balance)
    FROM card
);
```

A database of train stations

customer

customer_id	name
1	Alan
2	Bob
3	Cindy
4	David

card

card_id	balance
1	-10
2	-4
3	30
4	-2
5	-1
6	10

owner

customer_id	card_id
1	1
1	2
2	3
2	4
3	6
4	5

station

station_id	name
1	Wan Chai
2	Central
3	Admiralty

latest_usage

customer_id	card_id	station_id	latest_expense	latest_date
1	1	2	12	2025-02-25
1	2	2	8	2025-02-25
2	3	3	4	2025-02-26
2	4	3	6	2025-02-24
3	6	3	4	2025-02-24
4	5	2	12	2025-02-24
4	5	1	10	2025-02-24
4	5	1	12	2025-02-24

- Q2: Find the card that has the maximum balance.
- Return: card_id, card balance

- SQL 2

```
SELECT card_id, balance
FROM card
WHERE balance = (
    SELECT MAX(balance)
    FROM card
);
```

A database of train stations

customer

customer_id	name
1	Alan
2	Bob
3	Cindy
4	David

card

card_id	balance
1	-10
2	-4
3	30
4	-2
5	-1
6	10

owner

customer_id	card_id
1	1
1	2
2	3
2	4
3	6
4	5

station

station_id	name
1	Wan Chai
2	Central
3	Admiralty

latest_usage

customer_id	card_id	station_id	latest_expense	latest_date
1	1	2	12	2025-02-25
1	2	2	8	2025-02-25
2	3	3	4	2025-02-26
2	4	3	6	2025-02-24
3	6	3	4	2025-02-24
4	5	2	12	2025-02-24
4	5	1	10	2025-02-24
4	5	1	12	2025-02-24

- Q3: find each customer's card, which has the largest balance.
- Return: customer_id, card_id, card balance

A database of train stations

customer

customer_id	name
1	Alan
2	Bob
3	Cindy
4	David

card

card_id	balance
1	-10
2	-4
3	30
4	-2
5	-1
6	10

owner

customer_id	card_id
1	1
1	2
2	3
2	4
3	6
4	5

station

station_id	name
1	Wan Chai
2	Central
3	Admiralty

latest_usage

customer_id	card_id	station_id	latest_expense	latest_date
1	1	2	12	2025-02-25
1	2	2	8	2025-02-25
2	3	3	4	2025-02-26
2	4	3	6	2025-02-24
3	6	3	4	2025-02-24
4	5	2	12	2025-02-24
4	5	1	10	2025-02-24
4	5	1	12	2025-02-24

- Q3: find each customer's card, which has the largest balance.
- Return: customer_id, card_id, card balance

- SQL 1

```
select o.customer_id, o.card_id, c.balance
from card c, owner o
where c.card_id = o.card_id and
c.balance = (
      select max(c2.balance)
      from card c2, owner o2
      where c2.card_id = o2.card_id and
      o.customer_id = o2.customer_id
);
```

A database of train stations

customer

customer_id	name
1	Alan
2	Bob
3	Cindy
4	David

card

card_id	balance
1	-10
2	-4
3	30
4	-2
5	-1
6	10

owner

customer_id	card_id
1	1
1	2
2	3
2	4
3	6
4	5

station

station_id	name
1	Wan Chai
2	Central
3	Admiralty

latest_usage

customer_id	card_id	station_id	latest_expense	latest_date
1	1	2	12	2025-02-25
1	2	2	8	2025-02-25
2	3	3	4	2025-02-26
2	4	3	6	2025-02-24
3	6	3	4	2025-02-24
4	5	2	12	2025-02-24
4	5	1	10	2025-02-24
4	5	1	12	2025-02-24

- Q3: find each customer's card, which has the largest balance.
- Return: customer_id, card_id, card balance

- SQL 2

```
select o.customer_id, o.card_id, c.balance
from card c, owner o,
(
select o.customer_id as customer_id,
        max(c.balance) as max_balance
from card c, owner o
where c.card_id = o.card_id
group by o.customer_id
) as T
where
c.card_id = o.card_id and
c.balance = T.max_balance and
o.customer_id = T.customer_id;
```

A database of train stations

customer

customer_id	name
1	Alan
2	Bob
3	Cindy
4	David

card

card_id	balance
1	-10
2	-4
3	30
4	-2
5	-1
6	10

owner

customer_id	card_id
1	1
1	2
2	3
2	4
3	6
4	5

station

station_id	name
1	Wan Chai
2	Central
3	Admiralty

latest_usage

customer_id	card_id	station_id	latest_expense	latest_date
1	1	2	12	2025-02-25
1	2	2	8	2025-02-25
2	3	3	4	2025-02-26
2	4	3	6	2025-02-24
3	6	3	4	2025-02-24
4	5	2	12	2025-02-24
4	5	1	10	2025-02-24
4	5	1	12	2025-02-24

- Q3: find each customer's card, which has the largest balance.
- Return: customer_id, card_id, card balance

- SQL 3

```
select o.customer_id, o.card_id, c.balance
from card c join owner o
on c.card_id = o.card_id
join (
      select o.customer_id as customer_id,
            max(c.balance) as max_balance
      from card c, owner o
      where c.card_id = o.card_id
      group by o.customer_id) as T
on
c.balance = T.max_balance and
o.customer_id = T.customer_id;
```

A database of train stations

customer

customer_id	name
1	Alan
2	Bob
3	Cindy
4	David

card

card_id	balance
1	-10
2	-4
3	30
4	-2
5	-1
6	10

owner

customer_id	card_id
1	1
1	2
2	3
2	4
3	6
4	5

station

station_id	name
1	Wan Chai
2	Central
3	Admiralty

latest_usage

customer_id	card_id	station_id	latest_expense	latest_date
1	1	2	12	2025-02-25
1	2	2	8	2025-02-25
2	3	3	4	2025-02-26
2	4	3	6	2025-02-24
3	6	3	4	2025-02-24
4	5	2	12	2025-02-24
4	5	1	10	2025-02-24
4	5	1	12	2025-02-24

- Q3: find each customer's card, which has the largest balance.
- Return: customer_id, card_id, card balance

- SQL 4

```
select o.customer_id, c.card_id, c.balance
from owner o join card c
on o.card_id = c.card_id
where EXISTS(
      select 1
      from owner o2 join card c2
      on o2.card_id = c2.card_id
      where o2.customer_id = o.customer_id
      group by o2.customer_id
      having c.balance = max(c2.balance)
);
```

A database of train stations

customer

customer_id	name
1	Alan
2	Bob
3	Cindy
4	David

card

card_id	balance
1	-10
2	-4
3	30
4	-2
5	-1
6	10

owner

customer_id	card_id
1	1
1	2
2	3
2	4
3	6
4	5

station

station_id	name
1	Wan Chai
2	Central
3	Admiralty

latest_usage

customer_id	card_id	station_id	latest_expense	latest_date
1	1	2	12	2025-02-25
1	2	2	8	2025-02-25
2	3	3	4	2025-02-26
2	4	3	6	2025-02-24
3	6	3	4	2025-02-24
4	5	2	12	2025-02-24
4	5	1	10	2025-02-24
4	5	1	12	2025-02-24

- Q4: Find the third highest card balance.
- Return: card balance

A database of train stations

customer

customer_id	name
1	Alan
2	Bob
3	Cindy
4	David

card

card_id	balance
1	-10
2	-4
3	30
4	-2
5	-1
6	10

owner

customer_id	card_id
1	1
1	2
2	3
2	4
3	6
4	5

station

station_id	name
1	Wan Chai
2	Central
3	Admiralty

latest_usage

customer_id	card_id	station_id	latest_expense	latest_date
1	1	2	12	2025-02-25
1	2	2	8	2025-02-25
2	3	3	4	2025-02-26
2	4	3	6	2025-02-24
3	6	3	4	2025-02-24
4	5	2	12	2025-02-24
4	5	1	10	2025-02-24
4	5	1	12	2025-02-24

- Q4: Find the third highest card balance.
- Return: card balance

- SQL 1

```
select balance
from card
order by balance desc
limit 1 offset 2;
```

A database of train stations

customer

customer_id	name
1	Alan
2	Bob
3	Cindy
4	David

card

card_id	balance
1	-10
2	-4
3	30
4	-2
5	-1
6	10

owner

customer_id	card_id
1	1
1	2
2	3
2	4
3	6
4	5

station

station_id	name
1	Wan Chai
2	Central
3	Admiralty

latest_usage

customer_id	card_id	station_id	latest_expense	latest_date
1	1	2	12	2025-02-25
1	2	2	8	2025-02-25
2	3	3	4	2025-02-26
2	4	3	6	2025-02-24
3	6	3	4	2025-02-24
4	5	2	12	2025-02-24
4	5	1	10	2025-02-24
4	5	1	12	2025-02-24

- Q4: Find the third highest card balance.
- Return: card balance

- SQL 2

```
select balance
from card
order by balance desc
limit 2,1;
```

A database of train stations

customer

customer_id	name
1	Alan
2	Bob
3	Cindy
4	David

card

card_id	balance
1	-10
2	-4
3	30
4	-2
5	-1
6	10

owner

customer_id	card_id
1	1
1	2
2	3
2	4
3	6
4	5

station

station_id	name
1	Wan Chai
2	Central
3	Admiralty

latest_usage

customer_id	card_id	station_id	latest_expense	latest_date
1	1	2	12	2025-02-25
1	2	2	8	2025-02-25
2	3	3	4	2025-02-26
2	4	3	6	2025-02-24
3	6	3	4	2025-02-24
4	5	2	12	2025-02-24
4	5	1	10	2025-02-24
4	5	1	12	2025-02-24

- Q4: Find the third highest card balance.
- Return: card balance

- SQL 3

```
select max(balance)
from card
where balance < (
    select max(balance)
    from card
    where balance < (
        select max(balance)
        from card
    )
);
```

A database of train stations

customer

customer_id	name
1	Alan
2	Bob
3	Cindy
4	David

card

card_id	balance
1	-10
2	-4
3	30
4	-2
5	-1
6	10

owner

customer_id	card_id
1	1
1	2
2	3
2	4
3	6
4	5

station

station_id	name
1	Wan Chai
2	Central
3	Admiralty

latest_usage

customer_id	card_id	station_id	latest_expense	latest_date
1	1	2	12	2025-02-25
1	2	2	8	2025-02-25
2	3	3	4	2025-02-26
2	4	3	6	2025-02-24
3	6	3	4	2025-02-24
4	5	2	12	2025-02-24
4	5	1	10	2025-02-24
4	5	1	12	2025-02-24

- Q5: Find the list of customers, who took train at the same station more than once in the same day, and date must be ordered from the most recent date to oldest date.
- Return: customer_id, station_name, latest_date, number of visits

A database of train stations

customer

customer_id	name
1	Alan
2	Bob
3	Cindy
4	David

card

card_id	balance
1	-10
2	-4
3	30
4	-2
5	-1
6	10

owner

customer_id	card_id
1	1
1	2
2	3
2	4
3	6
4	5

station

station_id	name
1	Wan Chai
2	Central
3	Admiralty

latest_usage

customer_id	card_id	station_id	latest_expense	latest_date
1	1	2	12	2025-02-25
1	2	2	8	2025-02-25
2	3	3	4	2025-02-26
2	4	3	6	2025-02-24
3	6	3	4	2025-02-24
4	5	2	12	2025-02-24
4	5	1	10	2025-02-24
4	5	1	12	2025-02-24

- Q5: Find the list of customers, who took train at the same station more than once in the same day, and date must be ordered from the most recent date to oldest date.
- Return: customer_id, station_name, latest_date, number of visits

- SQL 1

```
select T.customer_id, S.name, T.latest_date, T.cnt
from station S join
(select customer_id, station_id, latest_date, count(*) as cnt
from latest_usage
group by customer_id, station_id, latest_date
having count(*) > 1
order by latest_date DESC) as T
on S.station_id = T.station_id;
```

A database of train stations

customer

customer_id	name
1	Alan
2	Bob
3	Cindy
4	David

card

card_id	balance
1	-10
2	-4
3	30
4	-2
5	-1
6	10

owner

customer_id	card_id
1	1
1	2
2	3
2	4
3	6
4	5

station

station_id	name
1	Wan Chai
2	Central
3	Admiralty

latest_usage

customer_id	card_id	station_id	latest_expense	latest_date
1	1	2	12	2025-02-25
1	2	2	8	2025-02-25
2	3	3	4	2025-02-26
2	4	3	6	2025-02-24
3	6	3	4	2025-02-24
4	5	2	12	2025-02-24
4	5	1	10	2025-02-24
4	5	1	12	2025-02-24

- Q5: Find the list of customers, who took train at the same station more than once in the same day, and date must be ordered from the most recent date to oldest date.
- Return: customer_id, station_name, latest_date, number of visits

- SQL 2

```
select T.customer_id, S.name, T.latest_date, T.cnt
from station S join
(select customer_id, station_id, latest_date, count(*) as cnt
from latest_usage
group by customer_id, station_id, latest_date
having count(*) > 1
) as T
on S.station_id = T.station_id
order by T.latest_date DESC;
```

A database of train stations

customer

customer_id	name
1	Alan
2	Bob
3	Cindy
4	David

card

card_id	balance
1	-10
2	-4
3	30
4	-2
5	-1
6	10

owner

customer_id	card_id
1	1
1	2
2	3
2	4
3	6
4	5

station

station_id	name
1	Wan Chai
2	Central
3	Admiralty

latest_usage

customer_id	card_id	station_id	latest_expense	latest_date
1	1	2	12	2025-02-25
1	2	2	8	2025-02-25
2	3	3	4	2025-02-26
2	4	3	6	2025-02-24
3	6	3	4	2025-02-24
4	5	2	12	2025-02-24
4	5	1	10	2025-02-24
4	5	1	12	2025-02-24

- Q5: Find the list of customers, who took train at the same station more than once in the same day, and date must be ordered from the most recent date to oldest date.
- Return: customer_id, station_name, latest_date, number of visits

- SQL 3

```
select L.customer_id, S.name, L.latest_date, count(*)
from latest_usage L, station S
where S.station_id = L.station_id
group by L.customer_id, S.name, L.latest_date
having count(*) > 1
order by L.latest_date DESC;
```

A database of train stations

customer

customer_id	name
1	Alan
2	Bob
3	Cindy
4	David

card

card_id	balance
1	-10
2	-4
3	30
4	-2
5	-1
6	10

owner

customer_id	card_id
1	1
1	2
2	3
2	4
3	6
4	5

station

station_id	name
1	Wan Chai
2	Central
3	Admiralty

latest_usage

customer_id	card_id	station_id	latest_expense	latest_date
1	1	2	12	2025-02-25
1	2	2	8	2025-02-25
2	3	3	4	2025-02-26
2	4	3	6	2025-02-24
3	6	3	4	2025-02-24
4	5	2	12	2025-02-24
4	5	1	10	2025-02-24
4	5	1	12	2025-02-24

- Q6: Suppose the customer_id is a unique numeric increasing value, write a SQL query to select the first two even customer_id values.
- Return: customer_id

A database of train stations

customer

customer_id	name
1	Alan
2	Bob
3	Cindy
4	David

card

card_id	balance
1	-10
2	-4
3	30
4	-2
5	-1
6	10

owner

customer_id	card_id
1	1
1	2
2	3
2	4
3	6
4	5

station

station_id	name
1	Wan Chai
2	Central
3	Admiralty

latest_usage

customer_id	card_id	station_id	latest_expense	latest_date
1	1	2	12	2025-02-25
1	2	2	8	2025-02-25
2	3	3	4	2025-02-26
2	4	3	6	2025-02-24
3	6	3	4	2025-02-24
4	5	2	12	2025-02-24
4	5	1	10	2025-02-24
4	5	1	12	2025-02-24

- Q6: Suppose the customer_id is a unique numeric increasing value, write a SQL query to select the first two even customer_id values.
- Return: customer_id

- SQL 1

```
select customer_id
from customer
where customer_id%2=0
limit 2;
```

A database of train stations

customer

customer_id	name
1	Alan
2	Bob
3	Cindy
4	David

card

card_id	balance
1	-10
2	-4
3	30
4	-2
5	-1
6	10

owner

customer_id	card_id
1	1
1	2
2	3
2	4
3	6
4	5

station

station_id	name
1	Wan Chai
2	Central
3	Admiralty

latest_usage

customer_id	card_id	station_id	latest_expense	latest_date
1	1	2	12	2025-02-25
1	2	2	8	2025-02-25
2	3	3	4	2025-02-26
2	4	3	6	2025-02-24
3	6	3	4	2025-02-24
4	5	2	12	2025-02-24
4	5	1	10	2025-02-24
4	5	1	12	2025-02-24

- Q7: What are the outputs of the below SQL queries?

```
SELECT customer_id, COUNT(*), SUM(1)
FROM latest_usage
GROUP BY customer_id;
```

```
SELECT station_id, COUNT(*), SUM(2)
FROM latest_usage
GROUP BY station_id;
```

```
SELECT latest_expense, COUNT(*), SUM(3)
FROM latest_usage
GROUP BY latest_expense;
```

- Explain why?

A database of train stations

customer

customer_id	name
1	Alan
2	Bob
3	Cindy
4	David

card

card_id	balance
1	-10
2	-4
3	30
4	-2
5	-1
6	10

owner

customer_id	card_id
1	1
1	2
2	3
2	4
3	6
4	5

station

station_id	name
1	Wan Chai
2	Central
3	Admiralty

latest_usage

customer_id	card_id	station_id	latest_expense	latest_date
1	1	2	12	2025-02-25
1	2	2	8	2025-02-25
2	3	3	4	2025-02-26
2	4	3	6	2025-02-24
3	6	3	4	2025-02-24
4	5	2	12	2025-02-24
4	5	1	10	2025-02-24
4	5	1	12	2025-02-24

- Q7: What are the outputs of the below SQL queries?

```
SELECT customer_id, COUNT(*), SUM(1)
FROM latest_usage
GROUP BY customer_id;
```

```
SELECT station_id, COUNT(*), SUM(2)
FROM latest_usage
GROUP BY station_id;
```

```
SELECT latest_expense, COUNT(*), SUM(3)
FROM latest_usage
GROUP BY latest_expense;
```

- Explain why?

$$SUM(k) = COUNT(*) * k$$

A database of train stations

customer

customer_id	name
1	Alan
2	Bob
3	Cindy
4	David

card

card_id	balance
1	-10
2	-4
3	30
4	-2
5	-1
6	10

owner

customer_id	card_id
1	1
1	2
2	3
2	4
3	6
4	5

station

station_id	name
1	Wan Chai
2	Central
3	Admiralty

latest_usage

customer_id	card_id	station_id	latest_expense	latest_date
1	1	2	12	2025-02-25
1	2	2	8	2025-02-25
2	3	3	4	2025-02-26
2	4	3	6	2025-02-24
3	6	3	4	2025-02-24
4	5	2	12	2025-02-24
4	5	1	10	2025-02-24
4	5	1	12	2025-02-24

- Q8: Write a SQL query to calculate the sum of all positive card balances, and the sum of all negative card balances.
- Return: the sum of all positive balances and the sum of all negative balances

A database of train stations

customer

customer_id	name
1	Alan
2	Bob
3	Cindy
4	David

card

card_id	balance
1	-10
2	-4
3	30
4	-2
5	-1
6	10

owner

customer_id	card_id
1	1
1	2
2	3
2	4
3	6
4	5

station

station_id	name
1	Wan Chai
2	Central
3	Admiralty

latest_usage

customer_id	card_id	station_id	latest_expense	latest_date
1	1	2	12	2025-02-25
1	2	2	8	2025-02-25
2	3	3	4	2025-02-26
2	4	3	6	2025-02-24
3	6	3	4	2025-02-24
4	5	2	12	2025-02-24
4	5	1	10	2025-02-24
4	5	1	12	2025-02-24

- Q8: Write a SQL query to calculate the sum of all positive card balances, and the sum of all negative card balances.
- Return: the sum of all positive balances and the sum of all negative balances
- SQL 1

```
select sum( case
                when balance > 0
                then balance
                else 0
            end
        ) as sum_pos_balance,
       sum( case
                when balance < 0
                then balance
                else 0
            end
        ) as sum_neg_balance
from card;
```

A database of train stations

customer

customer_id	name
1	Alan
2	Bob
3	Cindy
4	David

card

card_id	balance
1	-10
2	-4
3	30
4	-2
5	-1
6	10

owner

customer_id	card_id
1	1
1	2
2	3
2	4
3	6
4	5

station

station_id	name
1	Wan Chai
2	Central
3	Admiralty

latest_usage

customer_id	card_id	station_id	latest_expense	latest_date
1	1	2	12	2025-02-25
1	2	2	8	2025-02-25
2	3	3	4	2025-02-26
2	4	3	6	2025-02-24
3	6	3	4	2025-02-24
4	5	2	12	2025-02-24
4	5	1	10	2025-02-24
4	5	1	12	2025-02-24

- Q8: Write a SQL query to calculate the sum of all positive card balances, and the sum of all negative card balances.
- Return: the sum of all positive balances and the sum of all negative balances

- SQL 2

```
select
(select sum(balance) from card where balance > 0)
as sum_pos_balance,
(select sum(balance) from card where balance < 0)
as sum_neg_balance;
```

A database of train stations

customer

customer_id	name
1	Alan
2	Bob
3	Cindy
4	David

card

card_id	balance
1	-10
2	-4
3	30
4	-2
5	-1
6	10

owner

customer_id	card_id
1	1
1	2
2	3
2	4
3	6
4	5

station

station_id	name
1	Wan Chai
2	Central
3	Admiralty

latest_usage

customer_id	card_id	station_id	latest_expense	latest_date
1	1	2	12	2025-02-25
1	2	2	8	2025-02-25
2	3	3	4	2025-02-26
2	4	3	6	2025-02-24
3	6	3	4	2025-02-24
4	5	2	12	2025-02-24
4	5	1	10	2025-02-24
4	5	1	12	2025-02-24

- Q9: Write a SQL query to add 20 to negative card balance, add 5 to positive card balance.
- Requirement: use the SELECT clause

A database of train stations

customer

customer_id	name
1	Alan
2	Bob
3	Cindy
4	David

card

card_id	balance
1	-10
2	-4
3	30
4	-2
5	-1
6	10

owner

customer_id	card_id
1	1
1	2
2	3
2	4
3	6
4	5

station

station_id	name
1	Wan Chai
2	Central
3	Admiralty

latest_usage

customer_id	card_id	station_id	latest_expense	latest_date
1	1	2	12	2025-02-25
1	2	2	8	2025-02-25
2	3	3	4	2025-02-26
2	4	3	6	2025-02-24
3	6	3	4	2025-02-24
4	5	2	12	2025-02-24
4	5	1	10	2025-02-24
4	5	1	12	2025-02-24

- Q9: Write a SQL query to add 20 to negative card balance, add 5 to positive card balance.
- Requirement: use the SELECT clause

- SQL 1

```
select
    case
        when balance < 0
        then balance + 20
        else balance + 5
    end
    as new_balance
from card;
```

A database of train stations

customer

customer_id	name
1	Alan
2	Bob
3	Cindy
4	David

card

card_id	balance
1	-10
2	-4
3	30
4	-2
5	-1
6	10

owner

customer_id	card_id
1	1
1	2
2	3
2	4
3	6
4	5

station

station_id	name
1	Wan Chai
2	Central
3	Admiralty

latest_usage

customer_id	card_id	station_id	latest_expense	latest_date
1	1	2	12	2025-02-25
1	2	2	8	2025-02-25
2	3	3	4	2025-02-26
2	4	3	6	2025-02-24
3	6	3	4	2025-02-24
4	5	2	12	2025-02-24
4	5	1	10	2025-02-24
4	5	1	12	2025-02-24

- Q9: Write a SQL query to add 20 to negative card balance, add 5 to positive card balance.
- Requirement: use the SELECT clause

- SQL 2

```
select balance + 20 * (balance < 0) + 5 * (balance > 0)
as new_balance
from card;
```

A database of train stations

customer

customer_id	name
1	Alan
2	Bob
3	Cindy
4	David

card

card_id	balance
1	-10
2	-4
3	30
4	-2
5	-1
6	10

owner

customer_id	card_id
1	1
1	2
2	3
2	4
3	6
4	5

station

station_id	name
1	Wan Chai
2	Central
3	Admiralty

latest_usage

customer_id	card_id	station_id	latest_expense	latest_date
1	1	2	12	2025-02-25
1	2	2	8	2025-02-25
2	3	3	4	2025-02-26
2	4	3	6	2025-02-24
3	6	3	4	2025-02-24
4	5	2	12	2025-02-24
4	5	1	10	2025-02-24
4	5	1	12	2025-02-24

- Q10: Write a SQL query to find the customer, who has the maximum number of cards, and all his/her cards have been deducted to negative balance at the same station within the last ten days.
- Return: customer_id, customer_name, number of all his/her cards satisfied the above query, station_name

A database of train stations

customer

customer_id	name
1	Alan
2	Bob
3	Cindy
4	David

card

card_id	balance
1	-10
2	-4
3	30
4	-2
5	-1
6	10

owner

customer_id	card_id
1	1
1	2
2	3
2	4
3	6
4	5

station

station_id	name
1	Wan Chai
2	Central
3	Admiralty

latest_usage

customer_id	card_id	station_id	latest_expense	latest_date
1	1	2	12	2025-02-25
1	2	2	8	2025-02-25
2	3	3	4	2025-02-26
2	4	3	6	2025-02-24
3	6	3	4	2025-02-24
4	5	2	12	2025-02-24
4	5	1	10	2025-02-24
4	5	1	12	2025-02-24

- Q10: Write a SQL query to find the customer, who has the maximum number of cards, and all his/her cards have been deducted to negative balance at the same station within the last ten days.
- Return: customer_id, customer_name, number of all his/her cards satisfied the above query, station_name

```
select L.customer_id, C.name, count(distinct L.card_id), S.name
from latest_usage L, customer C, station S
where L.latest_date > date_sub(curdate(), interval 10 day)
and C.customer_id = L.customer_id
and S.station_id = L.station_id
group by L.customer_id, L.station_id
having L.customer_id in
(

select customer_id
from customer
where

customer_id in (
        select customer_id
        from owner
        group by customer_id
        having count(*) >= all(
                select count(*)
            from owner
            group by customer_id
                )
)

and customer_id in (
select l.customer_id
from latest_usage l
where l.latest_date > date_sub(curdate(), interval 10 day)
group by l.customer_id
having count(distinct station_id) = 1 and count(distinct card_id) = (
        select count(*)
        from owner o
        where o.customer_id = l.customer_id
)
)

and customer_id in (
select customer_id
from owner o, card c
where o.card_id = c.card_id
group by customer_id
having max(balance) < 0
)
);
```

Chapter 3 - Tutorial

END

COMP3278C
Introduction to Database Management Systems

Dr. CHEN, Yi
Email: chenyi1@hku.hk

School of Computing & Data Science, The University of Hong Kong

Acknowledgement: Dr. Chui Chun Kit, Dr. Reynold Cheng, Dr. Ping Luo