COMP 3311
DATABASE MANAGEMENT SYSTEMS

TUTORIAL 4
STRUCTURED QUERY LANGUAGE (SQL)

REVIEW: GROUP BY

Motivation: Group by permits aggregate results to be displayed (e.g., count, avg, max, min, sum, stdev) for groups. For instance, **group by** x will get a result for every different value of x.

☞ **Recall: Aggregate queries without group by return just a single number.**

- An attribute in the **select** clause must also appear in the **group by** clause. ***The opposite is not true!*** There may be attributes in the **group by** clause that do not appear in the **select** clause.

- Any condition that appears in the **where** clause is applied ***before*** the formation of groups. That is, records that do not pass the **where** predicate are eliminated ***before*** the formation of groups.

- Any condition that appears in the **having** clause refers only to the groups and is applied ***after*** the formation of the groups. The **having** clause condition must involve aggregate functions or attributes that appear in the **select** clause or **group by** clause.

EXAMPLE RELATIONAL SCHEMA

Customer(customerId, name)

Account(accountId, *customerId*)

Deposit(depositId, *accountId*, *customerId*, amount)

Withdrawal(withdrawalId, *accountId*, *customerId*, amount)

Attribute names in italics are foreign key attributes.

EXERCISE 1

Find the customer id of the customers who deposited into both account A1 and A2.

Use **intersect**.

```
select distinct customerId
from Deposit
where accountId='A1'
intersect
select distinct customerId
from Deposit
where accountId='A2';
```

Deposit

depositId	accountId	customerId	amount
070940	A1	1	2000
070941	A1	1	1000
070943	A2	1	1000
070945	A2	2	3000
070959	A3	3	2000
080341	A3	2	5000
080342	A2	2	1500

Is it necessary to include distinct **in the** select **clauses to remove duplicates in the answer?**

No! Why?

☞ **The SQL set operators remove duplicates ⟹ intersect removes duplicates.**

Customer(customerId, name)

Account(accountId, *customerId*)

Deposit(depositId, *accountId*, *customerId*, amount)

Withdrawal(withdrawalId, *accountId*, *customerId*, amount)

EXERCISE 1 (CONTD)

Find the customer id of the customers who deposited into both account A1 and A2.

Use a subquery without **intersect**.

```
select distinct customerId
from Deposit
where accountId='A1'
        and customerId in
                (select distinct customerId
                 from Deposit
                 where accountId='A2');
```

Deposit

depositId	accountId	customerId	amount
070940	A1	1	2000
070941	A1	1	1000
070943	A2	1	1000
070945	A2	2	3000
070959	A3	3	2000
080341	A3	2	5000
080342	A2	2	1500

Is it necessary to include distinct in both select clauses to remove duplicates in the answer?

Yes and no! Why?

☞ **Since the SQL set membership operators do not remove duplicates, it is necessary in the outer select, but not in the inner select.**

Customer(customerId, name)

Account(accountId, *customerId*)

Deposit(depositId, *accountId*, *customerId*, amount)

Withdrawal(withdrawalId, *accountId*, *customerId*, amount)

EXERCISE 1 (CONT'D)

Find the customer id of the customers who deposited into both account A1 and A2.

Use only one select statement.

```
select distinct D1.customerId
from Deposit D1, Deposit D2
where D1.customerId=D2.customerId
        and D1.accountId='A1'
        and D2.accountId='A2';
```

Deposit

depositId	accountId	customerId	amount
070940	A1	1	2000
070941	A1	1	1000
070943	A2	1	1000
070945	A2	2	3000
070959	A3	3	2000
080341	A3	2	5000
080342	A2	2	1500

Does it matter whether D1 or D2 is specified in the select clause?

No.

Customer(customerId, name)

Account(accountId, *customerId*)

Deposit(depositId, *accountId*, *customerId*, amount)

Withdrawal(withdrawalId, *accountId*, *customerId*, amount)

EXERCISE 2

Find the ids of the accounts which have been deposited into by more than one customer.

Do not use **group by**.

```
select distinct D1.accountId
from Deposit D1, Deposit D2
where D1.customerId<>D2.customerId
      and D1.accountId=D2.accountId;
```

Deposit

depositId	accountId	customerId	amount
070940	A1	1	2000
070941	A1	1	1000
070943	A2	1	1000
070945	A2	2	3000
070959	A3	3	2000
080341	A3	2	5000
080342	A2	2	1500

How would you write the query if the condition was "more than X customers"?

Need to self-join Deposit *one more time* than the number of customers X with the appropriate conditions in the where clause.

(The condition becomes quite complicated!)

Customer(customerId, name)

Account(accountId, *customerId*)

Deposit(depositId, *accountId*, *customerId*, amount)

Withdrawal(withdrawalId, *accountId*, *customerId*, amount)

EXERCISE 2 (CONT'D)

Find the ids of the accounts which have been deposited into by more than one customer.

Use **group by**.

```
select distinct accountId
from Deposit
group by accountId
having count(distinct customerId)>=2;
```

What is the result if distinct is omitted in the having clause?

- A1 will also be included in the answer. **Why?**
- It is deposited into more than one time (*but by the same customer*).

Deposit

depositId	accountId	customerId	amount
070940	A1	1	2000
070941	A1	1	1000
070943	A2	1	1000
070945	A2	2	3000
070959	A3	3	2000
080341	A3	2	5000
080342	A2	2	1500

Customer(customerId, name)

Account(accountId, *customerId*)

Deposit(depositId, *accountId*, *customerId*, amount)

Withdrawal(withdrawalId, *accountId*, *customerId*, amount)

EXERCISE 3

Find the customer id of the customers who deposited into either account A1 or account A2 but not both accounts.

Use only one select statement.

```
select customerId
from Deposit
where accountId='A1'
        or accountId='A2'
group by customerId
having count(distinct accountId)=1;
```

Deposit

depositId	accountId	customerId	amount
070940	A1	1	2000
070941	A1	1	1000
070943	A2	1	1000
070945	A2	2	3000
070959	A3	3	2000
080341	A3	2	5000
080342	A2	2	1500

What is the result if distinct is omitted in the having clause?

- ☞ No account is selected. **Why?**
- ☞ The deposit counts for A1 and A2 for each customer is greater than one.

Customer(customerId, name)

Account(accountId, *customerId*)

Deposit(depositId, *accountId*, *customerId*, amount)

Withdrawal(withdrawalId, *accountId*, *customerId*, amount)

EXERCISE 4

Find the customer id of the customers who deposited the largest number of times.

Use aggregate functions.

```
select customerId
from Deposit
group by customerId
having count(*)=(select max(count(*))
                 from Deposit
                 group by customerId);
```

Customers who made the number of deposits equal to the largest number made by any customer.

The largest number of deposits by any customer.

Deposit

depositId	accountId	customerId	amount
070940	A1	1	2000
070941	A1	1	1000
070943	A2	1	1000
070945	A2	2	3000
070959	A3	3	2000
080341	A3	2	5000
080342	A2	2	1500

Customer(customerId, name)

Account(accountId, *customerId*)

Deposit(depositId, *accountId*, *customerId*, amount)

Withdrawal(withdrawalId, *accountId*, *customerId*, amount)

EXERCISE 4 (CONT'D)

Find the customer id of the customers who deposited the largest number of times.

Use set membership.

```
select customerId
from Deposit
group by customerId
having count(*)>=all (select count(*)
                      from Deposit
                      group by customerId);
```

Deposit

depositId	accountId	customerId	amount
070940	A1	1	2000
070941	A1	1	1000
070943	A2	1	1000
070945	A2	2	3000
070959	A3	3	2000
080341	A3	2	5000
080342	A2	2	1500

What is the result if we replace >=all **with** >all **in the** having **clause?**

☞ No customer is selected. **Why?**

☞ There is no customer who deposited more than the largest number of times!

The number of deposits made by a customer must be greater than or equal to the number of deposits made by all customers.

Customer(customerId, name)

Account(accountId, *customerId*)

Deposit(depositId, *accountId*, *customerId*, amount)

Withdrawal(withdrawalId, *accountId*, *customerId*, amount)

EXERCISE 5

Find all the names of the customers who have withdrawn more than 1000 dollars in a single withdrawal. If a customer made several such withdrawals, report her/his name only once.

```
select distinct name
from Customer, Withdrawal
where Customer.customerId=Withdrawal.customerId
        and amount>1000;
```

Customer(customerId, name)

Account(accountId, *customerId*)

Deposit(depositId, *accountId*, *customerId*, amount)

Withdrawal(withdrawalId, *accountId*, *customerId*, amount)

EXERCISE 6

While an account has only one owner, it may be shared by multiple customers who deposit money into and/or withdraw money from it. Find the account id of all the shared accounts. Assume that all shared account customers have made withdrawals from the account.

```
select distinct W1.accountId
from Withdrawal W1, Withdrawal W2
where W1.customerId<>W2.customerId
        and W1.accountId=W2.accountId;
```

Customer(customerId, name)

Account(accountId, *customerId*)

Deposit(depositId, *accountId*, *customerId*, amount)

Withdrawal(withdrawalId, *accountId*, *customerId*, amount)

EXERCISE 7

An "interesting account" is an account from which the withdrawal with the smallest amount was made.
Find the account id of accounts from which withdrawals have been made, excluding the interesting accounts.

Using **minus** operator.

```
select distinct accountId
from Withdrawal
minus
select accountId
from Withdrawal
where amount=(select min(amount)
              from Withdrawal);
```

Customer(customerId, name)

Account(accountId, *customerId*)

Deposit(depositId, *accountId*, *customerId*, amount)

Withdrawal(withdrawalId, *accountId*, *customerId*, amount)

EXERCISE 7 (CONT'D)

An "interesting account" is an account from which the withdrawal with the smallest amount was made. Find the account id of accounts from which withdrawals have been made, excluding the interesting accounts.

Using **not in** operator.

```
select distinct accountId
from Withdrawal
where accountId not in (select accountId
                        from Withdrawal
                        where amount=(select min(amount)
                                      from Withdrawal));
```

Customer(customerId, name)

Account(accountId, *customerId*)

Deposit(depositId, *accountId*, *customerId*, amount)

Withdrawal(withdrawalId, *accountId*, *customerId*, amount)