Tutorial 4

Structured Query Language 2

- Question: Consider the following schemas.
  CUST (*<u>cust-id</u>*, *name*), and
  WITHDRAW (*<u>w-id</u>*, *cust-id*, *acc-id*, *date*, *amount*)

- Write an SQL query to retrieve all the names of the customers who have <u>withdraw取款</u> more than 1k dollars in a single withdrawal.
  If a customer made several such withdrawals, her/his name should be reported only once.

检索单次取款超过 1k 美元的所有客户姓名。如果客户多次提款，则她/他的姓名只需报告一次。

- Answer: Consider the following schemas.
  CUST (<u>*cust-id*</u>, *name*), and
  WITHDRAW (<u>*w-id*</u>, *cust-id*, *acc-id*, *date*, *amount*)

- 检索单次取款超过 1k 美元的所有客户姓名。如果客户多次提款，则她/他的姓名只需报告一次。

select distinct *name*
from CUST **as** T1, WITHDRAW **as** T2
where T1.*cust-id* = T2.*cust-id* **and** T2.*amount* > 1k

- Question: Consider the following schemas.
  CUST (*cust-id*, *name*), and
  WITHDRAW (*w-id*, *cust-id*, *acc-id*, *date*, *amount*)

- 有时可能存在“共享”帐户，即一个账户可能具有多个owner。
  编写 SQL 查询以返回所有共享帐户的 acc-id。我们假设共享帐户的所有owner都已从该帐户中提款。

❖ Answer: Consider the following schemas.
CUST (*cust-id*, *name*), and
WITHDRAW (*w-id*, *cust-id*, *acc-id*, *date*, *amount*)

❖ 有时可能存在“共享”帐户，即一个账户可能具有多个owner。
编写 SQL 查询以返回所有共享帐户的 acc-id。我们假设共享帐户的所有owner都已从该帐户中提款。

select T1.*acc-id*
from WITHDRAW **as** T1, WITHDRAW **as** T2
where T1.*cust-id* <> T2.*cust-id* **and** T1.*acc-id* = T2.*acc-id*

w-id	acc-id	cust-id	amount
070940	A1	1	2K
070941	A1	1	1K
070943	A2	1	1K
070945	A2	2	3K
070959	A3	3	2K
080341	A3	2	5K

- Question: Consider the following schemas.
  CUST (*cust-id*, *name*), and
  WITHDRAW (*w-id*, *cust-id*, *acc-id*, *date*, *amount*)

- 我们使用“the interesting account”这个名称来指代提取金额最小的账户。
- Retrieve the *acc-id* of accounts from which withdrawals have been made, except the interesting account.

- Answer: Consider the following schemas.
  CUST (*cust-id*, *name*), and
  WITHDRAW (w-*id*, *cust-id*, *acc-id*, *date*, *amount*)

- Retrieve the *acc-id* of accounts from which withdrawals have been made, except the interesting account.

```
select distinct acc-id from WITHDRAW
where acc-id not in
        (select acc-id from WITHDRAW
         where amount =
                    (select min (amount)
                     from WITHDRAW))
```

w-id	acc-id	cust-id	amount
070940	A1	1	2K
070941	A1	1	1K
070943	A2	1	1K
070945	A2	2	3K
070959	A3	3	2K
080341	A3	2	5K

❖Consider table: *deposit* (*dep-id*, *acc-id*, *cust-id*, *amount*).
We want to retrieve the *cust-id* of the customers who deposited存款 into two accounts with *acc-id* 'A1' and 'A2', respectively.　我们想要检索分别向 acc-id 为 "A1"和"A2"的两个帐户存款的客户的 cust-id。

dep-id	acc-id	cust-id	amount
070940	A1	1	2K
070941	A1	1	1K
070943	A2	1	1K
070945	A2	2	3K
070959	A3	3	2K
080341	A3	2	5K

Write an SQL query with **intersect**.

(**select distinct** *cust-id* **from** *deposit* **where** *acc-id* = 'A1')
intersect
(**select distinct** *cust-id* **from** *deposit* **where** *acc-id* = 'A2')

Write a nested SQL query without **intersect**.

select distinct *cust-id* **from** *deposit*
where *acc-id* = 'A1' **and** *cust-id* **in**
(**select** *cust-id* **from** *deposit*
where *acc-id* = 'A2')

- Again consider table: *deposit* (*dep-id*, *acc-id*, *cust-id*, *amount*).
  We want to retrieve the *cust-id* of the customers who deposited into two accounts with *acc-id* 'A1' and 'A2', respectively

- Write an SQL query that contains only one **select** .

select distinct T1.*cust-id*
from *deposit* **as** T1, *deposit* **as** T2
where T1.*cust-id* = T2.*cust-id* **and**
T1.*acc-id* = 'A1' **and**
T2.*acc-id* = 'A2'

dep-id	acc-id	cust-id	amount
070940	A1	1	2K
070941	A1	1	1K
070943	A2	1	1K
070945	A2	2	3K
070959	A3	3	2K
080341	A3	2	5K

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

❖Write a SQL query without **group by**:

select D1.*acc-id*
from deposit **as** D1, deposit **as** D2
where D1.*cust-id* <> D2.*cust-id* **and** D1.*acc-id* = D2.*acc-id*

dep-id	acc-id	cust-id	amount
070940	A1	1	2K
070941	A1	1	1K
070943	A2	1	1K
070945	A2	2	3K
070959	A3	3	2K
080341	A3	2	5K

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

- Write a SQL query with **group by**:

select *acc-id*
from *deposit*
group by *acc-id*
having count (distinct *cust-id*) >= 2

- If there is no **distinct** here, A1 will also be displayed.

dep-id	acc-id	cust-id	amount
070940	A1	1	2K
070941	A1	1	1K
070943	A2	1	1K
070945	A2	2	3K
070959	A3	3	2K
080341	A3	2	5K

- Again consider table: *deposit* (*dep-id*, *acc-id*, *cust-id*, *amount*).
  We want to retrieve the *cust-id* of the customers who deposited into the account with *acc-id* = ‘A1’ or ‘A2’ but not both.

- Write an SQL query that contains only one SELECT.

select *cust-id* **from** *deposit*
where *acc-id* = ‘A1’ **or** *acc-id* = ‘A2’
group by *cust-id*
having count (distinct *acc-id*) = 1

必须的

dep_id	acc-id	cust-id	amount
070940	A1	1	2K
070941	A1	1	1K
070943	A2	1	1K
070945	A2	2	3K
070959	A3	3	2K
080341	A3	2	5K

- *deposit* (*dep-id*, *acc-id*, *cust-id*, *amount*).
  Retrieve the *cust-id* of the customer who deposited the largest number of times.

```
select  cust-id from deposit
group by cust-id
having count  (*) >= all
            (select count (*) from deposit
              group by cust-id)
```

dep-id	acc-id	cust-id	amount
070940	A1	1	2K
070941	A1	1	1K
070943	A2	1	1K
070945	A2	2	3K
070959	A3	3	2K
080341	A3	2	5K

- 一般而言，outer join没有相对应的指令，然而我们可以用现有的指令实现他们

- CS-PROF (*prof-id*, *name*)
- SUPERVISION (*prof-id*, *stu-id*)

- Write an alternative query that returns the same information as

  select *prof-id, name, stu-id*
  from CS-PROF **left outer join** SUPERVISION
  on CS-PROF.*prof-id* = SUPERVISION.*prof-id*

- Answer:
- CS-PROF (*prof-id*, *name*)
- SUPERVISION (*prof-id*, *stu-id*)

```
select prof-id, name, stu-id
from CS-PROF left outer join SUPERVISION
        on CS-PROF.prof-id = SUPERVISION.prof-id


(select T1.prof-id, name, stu-id
 from CS-PROF as T1, SUPERVISION as T2
 where T1.prof-id = T2.prof-id)
union
(select prof-id, name, NULL
 from CS-PROF as T1
 where not exists
        (select * from SUPERVISION as T2
         where T1.prof-id = T2.prof-id))
```

- We can see that **left outer join** simplifies the query significantly.

- Question: Consider MARKS(*stu-id*, *course-id*, *score*)
- Write a query to retrieve the stu-id of every student who scored at least 80 in all the courses s/he took, but scored less than 90 in at least one course.

- Your query should not contain more than 2 **select**.

- Answer: Consider MARKS(*stu-id*, *course-id*, *score*).
- Write a query to retrieve the *stu-id* of every student who scored得分 at least 80 in all the courses s/he took, but scored less than 90 in at least one course.

```
 (select stu-id from MARKS
 where score < 90)
 except
(select stu-id from MARKS
 where score < 80)
```

- Answer: Consider MARKS (*stu-id*, *course-id*, *score*).
- Write a query to retrieve the *stu-id* of every student who scored at least 80 in all the courses s/he took, but scored less than 90 in at least one course.

select *stu-id*
from MARKS
group by *stu-id*
having min (*score*) >= 80 **and min** (*score*) < 90

- What will happen if some of values of score are NULL in table ?

- 除 count(*) 之外的所有聚合操作都会忽略聚合属性上具有空值的元组。