Bài 3

Ngôn ngữ SQL
Truy xuất dữ liệu phức tạp

Nội dung trình bày

- Truy xuất dữ liệu phức tạp
- Hàm tổng hợp
- Nhóm các bộ dữ liệu
- Truy vấn lồng nhau
- Phép toán kết bảng
- Tạo bảng tạm trong một truy vấn
- Biểu thức lựa chọn

Truy xuất dữ liệu phức tạp

- Các truy xuất dữ liệu phức tạp như:
  - Nhóm các bộ dữ liệu để tóm tắt dữ liệu.
  - Truy xuất dữ liệu để sử dụng trong **điều kiện chọn**.
  - Tạo bảng kết.
  - Tạo bảng tạm để sử dụng trong một lệnh truy vấn.
  - Lựa chọn các giá trị tùy vào trường hợp.
- Dạng đầy đủ của lệnh truy vấn như sau:

```
select <danh sách thuộc tính>
from (<bảng 1> join <bảng 2> on <điều kiện>)
where <điều kiện>
group by <danh sách thuộc tính nhóm>
having <điều kiện>
order by <danh sách thuộc tính sắp xếp>
```

Hàm tổng hợp trong SQL (1)

- SQL cung cấp các hàm tổng hợp để tóm tắt dữ liệu từ nhiều bộ dữ liệu thành một bộ dữ liệu.
  - **sum**(*thuộc tính*) - trả về tổng các giá trị trong cột
  - **max**(*thuộc tính*) - trả về giá trị lớn nhất trong cột
  - **min**(*thuộc tính*) - trả về giá trị nhỏ nhất trong cột
  - **avg**(*thuộc tính*) - trả về giá trị trung bình trong cột
  - **count**(*thuộc tính*) - trả về số giá trị trong cột
  - **count**(*) - trả về số bộ dữ liệu trong một nhóm
- Các NULL trong một cột sẽ được bỏ qua khi áp dụng các hàm tập hợp trên cột đó.
- Được sử dụng trong mệnh đề **select** hoặc **having**.

Hàm tổng hợp trong SQL (2)

- ***Q17*** *- Tìm tổng tiền lương của tất cả nhân viên, mức lương tối đa,mức lương tối thiểu và mức lương trung bình.*

```
select sum(Salary), max(Salary),
       min(Salary), avg(Salary)
from EMPLOYEE;
```

281000	55000	25000	35125

```
select sum(Salary) as Total_sal,
       max(Salary) as Highest_sal,
       min(Salary) as Lowest_sal,
       avg(Salary) as Average_sal
from EMPLOYEE;
```

Nhóm các bộ dữ liệu (1)

- Mệnh đề **group by**

  group by <*danh sách thuộc tính nhóm*>
  - Để nhóm các bộ dữ liệu theo giá trị của một hay nhiều thuộc tính cho việc tóm tắt dữ liệu.
  - Các bộ dữ liệu có tổ hợp giá trị của các thuộc tính trong *danh sách thuộc tính nhóm* giống nhau thì được nhóm thành một bộ để tóm tắt dữ liệu.
- *Danh sách thuộc tính nhóm* gồm các thuộc tính của các quan hệ được chỉ định trong mệnh đề **from**.
- Thuộc tính được chỉ định trong mệnh đề **select** cũng phải được chỉ định trong *danh sách thuộc tính nhóm* trừ khi nó là đối số của hàm tập hợp.

Nhóm các bộ dữ liệu (2)

- ***Q18*** *- Với mỗi phòng ban, cho biết mã phòng ban, số của nhân viên trong phòng và mức lương trung bình của họ.*

```
select Dno, count(*) as Num_of_empl,
       avg(Salary) as Average_sal
from EMPLOYEE
group by Dno;
```

Fname	Minit	Lname	Ssn	...	Salary	Super_ssn	Dno
John	B	Smith	123456789		30000	333445555	5
Franklin	T	Wong	333445555		40000	888665555	5
Ramesh	K	Narayan	666884444		38000	333445555	5
Joyce	A	English	453453453	...	25000	333445555	5
Alicia	J	Zelaya	999887777		25000	987654321	4
Jennifer	S	Wallace	987654321		43000	888665555	4
Ahmad	V	Jabbar	987987987		25000	987654321	4
James	E	Bong	888665555		55000	NULL	1

Dno	Count (*)	Avg (Salary)
5	4	33250
4	3	31000
1	1	55000

Nhóm các bộ dữ liệu (3)

- Nếu các NULL tồn tại trong *thuộc tính nhóm* thì các bộ dữ liệu có giá trị NULL trong *thuộc tính nhóm* sẽ được nhóm thành một bộ.
- ***Q19*** *- Với mỗi người giám sát, cho biết mã số nhân viên của họ và số nhân viên mà họ giám sát.*

```
select Super_ssn, count(Ssn) as Num_of_empl
from EMPLOYEE
group by Super_ssn;
```

Super_ssn	Num_of_empl
NULL	1
333445555	3
888665555	2
987654321	2

Nhóm các bộ dữ liệu (4)

- Mệnh đề **having**

  having <*điều kiện*>
  - Để truy xuất tóm tắt dữ liệu của các nhóm thỏa *điều kiện*.
  - Chỉ được đi kèm với mệnh đề **group by**.
- *Điều kiện* còn gọi là **điều kiện chọn nhóm**, là một biểu thức luận lý gồm các mệnh đề luận lý có dạng

  <hàm tập hợp> <phép toán so sánh> <hằng>

Nhóm các bộ dữ liệu (5)

- ***Q20*** *- Đối với mỗi dự án có nhiều hơn 2 nhân viên tham gia, cho biết tên của dự án và số lượng nhân viên làm việc trong dự án.*

```
select Pname, count(*) as Num_of_empl
from PROJECT, WORKS_ON
where Pnumber = Pno
group by Pnumber, Pname
having count(*) > 2;
```

Pname	Pnumber	···	Essn	Pno	Hours
ProductX	1		123456789	1	32.5
ProductX	1		453453453	1	20.0
ProductY	2		123456789	2	7.5
ProductY	2		453453453	2	20.0
ProductY	2		333445555	2	10.0
ProductZ	3		666884444	3	40.0
ProductZ	3		333445555	3	10.0
Computerization	10	···	333445555	10	10.0

These groups are not selected by the HAVING condition of Q26.

Nhóm các bộ dữ liệu (6)

Pname	Pnumber	...	Essn	Pno	Hours
ProductY	2		123456789	2	7.5
ProductY	2		453453453	2	20.0
ProductY	2		333445555	2	10.0
Computerization	10		333445555	10	10.0
Computerization	10	...	999887777	10	10.0
Computerization	10		987987987	10	35.0
Reorganization	20		333445555	20	10.0
Reorganization	20		987654321	20	15.0
Reorganization	20		888665555	20	NULL
Newbenefits	30		987987987	30	5.0
Newbenefits	30		987654321	30	20.0
Newbenefits	30		999887777	30	30.0

After applying the HAVING clause condition

Pname	Count (*)
ProductY	3
Computerization	3
Reorganization	3
Newbenefits	3

Result of Q26
(Pnumber not shown)

Truy vấn lồng nhau

- Một lệnh truy vấn **select** được đặt trong một lệnh truy vấn **select** khác.
  - Lệnh select bên trong được gọi là *truy vấn con*.
  - Lệnh select còn lại được gọi là *truy vấn cha*.
- Truy vấn con có thể xuất hiện trong mệnh đề **where**, **from**, **select** hoặc **having** của truy vấn cha.
- SQL chuẩn không chỉ định số cấp độ lồng nhau.
- Có 3 dạng truy vấn con:
  - Truy vấn con vô hướng: kết quả có một cột và một hàng.
  - Truy vấn con hàng: kết quả có nhiều cột và một hàng.
  - Truy vấn con bảng: kết quả có nhiều hàng.

Truy vấn con vô hướng

- ***Q3A*** *- Cho biết tên, địa chỉ của các nhân viên làm việc trong phòng Research.*

```
select Fname, Lname, Address
from EMPLOYEE
where Dno = (select Dnumber
             from DEPARTMENT
             where Dname = 'Research');
```

- Cho phép sử dụng với các phép toán so sánh thông thường, **like** và **between - and**.

Truy vấn con hàng

- ***Q21*** *- Cho biết tên, địa chỉ của trưởng bộ phận Research.*

```
select Fname, Lname, Address
from EMPLOYEE
where (Dno, Ssn) = (select Dnumber, Mgr_ssn
                    from DEPARTMENT
                    where Dname = 'Research');
```

- So sánh danh sách thuộc tính với truy vấn con hàng:
  - DBMS có hỗ trợ: PostgreSQL, MySQL, SQLite, Oracle
  - Một số DBMS không hỗ trợ: SQL Server

Truy vấn con bảng

- Truy vấn con bảng thường xuất hiện trong điều kiện của mệnh đề **where** và **having** với các:
  - Phép toán so sánh tập hợp.
  - Hàm boolean **exists**.

```
select <danh sách thuộc tính>
from <danh sách bảng>
where <thuộc tính> <phép toán so sánh tập hợp> |
                   exists
      (truy vấn con)
```

Phép toán so sánh tập hợp (1)

- Phép toán **in** hoặc **not in** để so sánh một giá trị có thuộc hoặc không thuộc một tập hợp.

  <thuộc tính> **in** *(tập hợp các giá trị đơn)*

  <danh sách thuộc tính> **in** *(tập hợp các bộ dữ liệu)*
- *Tập hợp các giá trị đơn*
  - Được xác định bởi một truy vấn con bảng với kết quả chỉ có một cột.
- *Tập hợp các bộ dữ liệu*
  - Được xác định bởi một truy vấn con bảng với kết quả có số cột bằng số thuộc tính trong *danh sách thuộc tính*.

Phép toán so sánh tập hợp (2)

- ***Q9A*** *- Lập danh sách tất cả các mã số của các dự án có sự tham gia của nhân viên có họ là Smith, với tư cách là nhân viên hoặc người quản lý của phòng điều phối dự án.*

```
select distinct Pnumber
from PROJECT
where Pnumber in
      (select Pno
       from WORKS_ON, EMPLOYEE
       where Essn = Ssn and Lname = 'Smith') or
      Pnumber in
      (select Pnumber
       from PROJECT, DEPARTMENT, EMPLOYEE
       where Dnum = Dnumber and Mgr_ssn = Ssn and
             Lname = 'Smith');
```

Phép toán so sánh tập hợp (3)

- ***Q22*** *- Cho biết mã số của các nhân viên tham gia cùng dự án và bằng số giờ với nhân viên có mã số 123456789.*

```
select distinct Essn
from WORKS_ON
where (Pno, Hours) in
      (select Pno, Hours
       from WORKS_ON
       where Essn = '123456789');
```

Phép toán so sánh tập hợp (4)

- Kết hợp phép toán so sánh thông thường với các từ định lượng **all**, **any** (hoặc **some**) để so sánh một giá trị với một tập hợp.

  <thuộc tính> <phép toán so sánh> **all** | **any** *(tập hợp các giá trị đơn)*

  <danh sách thuộc tính> <phép toán so sánh> **all** | **any** *(tập hợp các bộ dữ liệu)*
- Kết quả so sánh với
  - **all** - là TRUE nếu TRUE với tất cả các phần tử của tập hợp.
  - **any** (hoặc **some**) - là TRUE nếu TRUE với một phần tử bất kì của tập hợp.

Phép toán so sánh tập hợp (5)

- ***Q23*** *- Cho biết tên các nhân có mức lương cao hơn mức lương của tất cả nhân viên bộ phận có mã số là 5.*

```
select Fname, Lname
from EMPLOYEE
where Salary > all
      (select Salary
       from EMPLOYEE
       where Dno = 5);
```

Truy vấn con tương quan (1)

- Khi điều kiện ở mệnh đề **where** của lệnh truy vấn con tham chiếu thuộc tính của một bảng của lệnh truy vấn cha thì nó được gọi là truy vấn con tương quan.
- Sử dụng bí danh (biến bộ) cho các bảng của lệnh truy vấn để thực hiện sự tham chiếu và tránh sự nhập nhằng.

Truy vấn con tương quan (2)

- ***Q24*** *- Cho biết họ tên của các nhân viên có người thân cùng tên và cùng giới tính với nhân viên đó.*

```
select Fname, Lname
from EMPLOYEE e
where e.Ssn in
      (select Essn
       from DEPENDENT d
       where e.Fname = d.Dependent_name and
             e.Sex = d.Sex);
```

- Với truy vấn con tương quan, truy vấn con được đánh giá một lần cho mỗi bộ dữ liệu (hoặc kết hợp các bộ dữ liệu) trong truy vấn cha.

Hàm kiểm tra exists

- Hàm **exists** để kiểm tra kết quả của một truy vấn con có dữ liệu hay không.
  - **exists (***truy vấn con***)** - là TRUE nếu kết quả của truy vấn con là có bộ dữ liệu.
  - **not exists (***truy vấn con***)** - là TRUE nếu kết quả của truy vấn con là rỗng.
- ***Q25*** *- Cho biết họ tên của các nhân viên không có người thân.*

```
select e.Fname, e.Lname
from EMPLOYEE e
where not exists (select *
                  from DEPENDENT d
                  where e.Ssn = d.Essn);
```

Phép toán kết và bảng kết trong SQL (1)

- Phép toán kết để thực hiện kết các bộ dữ liệu của hai quan hệ với nhau.

  <bảng 1> <phép toán kết> <bảng 2> **on** *<điều kiện>*
- Phép toán kết được chỉ định trong mệnh đề **from**.
- **Bảng kết** là kết quả của phép toán kết.
- Các kiểu phép toán kết
  - Phép toán kết **inner join**.
  - Phép toán kết **outer join**.
  - Phép toán kết **cross join**.

Phép toán kết và bảng kết trong SQL (2)

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Finance	6	NULL	NULL
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquaters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

Phép toán kết **inner join** (1)

- Phép kết **inner join** (hay **join**) để nối cặp bộ dữ liệu của hai bảng nếu chúng thỏa điều kiện.

```
DEPARTMENT d join
DEPT_LOCATIONS l on
d.Dnumber = l.Dnumber
```

Dname	Dnumber	Mgr_ssn	Mgr_start_date	Dnumber	Dlocation
Research	5	333445555	1988-05-22	5	Bellaire
Research	5	333445555	1988-05-22	5	Sugarland
Research	5	333445555	1988-05-22	5	Houston
Administration	4	987654321	1995-01-01	4	Stafford
Headquaters	1	888665555	1981-06-19	1	Houston

Phép toán kết **inner join** (2)

- ***Q26*** *- Cho biết tên, vị trí của bộ phận Research.*

```
select Dname, Dlocation
from DEPARTMENT d join
     DEPT_LOCATIONS l on d.Dnumber = l.Dnumber
where d.Dname = 'Research';
```

Dname	Dlocation
Research	Bellaire
Research	Sugarland
Research	Houston

- Phép kết **join** được thực hiện để tạo ra bảng kết. Sau đó truy xuất dữ liệu được thực hiện trên bảng kết.

Phép toán kết **outer join** (1)

- Phép kết **outer join** cơ bản cũng nối cặp bộ dữ liệu của hai bảng nếu chúng thỏa điều kiện. Nhưng nếu có bộ nào của bảng này không thỏa điều kiện kết với tất cả các bộ của bảng còn lại thì nó cũng được đưa vào bảng kết.
- **left outer join** (hay **left join**)
  - Mọi bộ dữ liệu của bảng bên trái đều có trong bảng kết.
- **right outer join** (hay **right join**)
  - Mọi bộ dữ liệu của bảng bên phải đều có trong bảng kết.
- **full outer join** (hay **full join**)
  - Mọi bộ dữ liệu của cả hai bảng đều có trong bảng kết.

Phép toán kết **outer join** (2)

```
DEPARTMENT d left join
DEPT_LOCATIONS l on
d.Dnumber = l.Dnumber
```

Dname	Dnumber	Mgr_ssn	Mgr_start_date	Dnumber	Dlocation
Finance	6	NULL	NULL	NULL	NULL
Research	5	333445555	1988-05-22	5	Bellaire
Research	5	333445555	1988-05-22	5	Sugarland
Research	5	333445555	1988-05-22	5	Houston
Administration	4	987654321	1995-01-01	4	Stafford
Headquaters	1	888665555	1981-06-19	1	Houston

- Bộ <'Finance', 6, NULL, NULL> được đưa vào bảng kết. Hai cột Dnumber, Dlocation không có dữ liệu phù hợp nên được đặt bằng các NULL.

Phép toán kết **outer join** (3)

- ***Q27*** *- Với mỗi bộ phận, liệt kê tên bộ phận và vị trí.*

```
select Dname, Dlocation
from DEPARTMENT d left join
     DEPT_LOCATIONS l on d.Dnumber = l.Dnumber;
```

Dname	Dlocation
Finance	NULL
Research	Bellaire
Research	Sugarland
Research	Houston
Administration	Stafford
Headquaters	Houston

Phép toán kết **cross join**

- Phép kết **cross join** để nối tất cả các cặp bộ dữ liệu của hai bảng không cần điều kiện.

```
DEPARTMENT cross join DEPT_LOCATIONS
```

Dname	Dnumber	Mgr_ssn	Mgr_start_date	Dnumber	Dlocation
Finance	6	NULL	NULL	1	Houston
Finance	6	NULL	NULL	4	Stafford
Finance	6	NULL	NULL	5	Bellaire
Finance	6	NULL	NULL	5	Sugarland
Finance	6	NULL	NULL	5	Houston
Research	5	333445555	1988-05-22	1	Houston
Research	5	333445555	1988-05-22	4	Stafford
.....					
.....					
Headquaters	1	888665555	1981-06-19	1	Houston
Headquaters	1	888665555	1981-06-19	4	Stafford
Headquaters	1	888665555	1981-06-19	5	Bellaire
Headquaters	1	888665555	1981-06-19	5	Sugarland
Headquaters	1	888665555	1981-06-19	5	Houston

Tạo bảng tạm trong một truy vấn (1)

- Mệnh đề **with** cho phép tạo bảng tạm chỉ để dùng trong một lệnh truy vấn cụ thể.

```
with <định nghĩa bảng tạm>
as (lệnh truy vấn tạo bảng tạm)
<lệnh truy vấn chính tham chiếu bảng tạm>;
```

- *Định nghĩa bảng tạm* gồm tên bảng và có thể tên các thuộc tính.
- Bảng tạm sẽ bị huy sau khi lệnh truy vấn được thực thi.

Tạo bảng tạm trong một truy vấn (2)

- ***Q28*** *- Với mỗi phòng có nhiều hơn 2 nhân viên, cho biết mã số phòng và số nhân viên có mức lương trên 30.000 của phòng đó.*

```
with BIG_DEPT
as (select Dno
    from EMPLOYEE
    group by Dno
    having count(*) > 2)
select e.Dno, count(*) as Num_of_Empl
from (BIG_DEPT b join
      EMPLOYEE e on b.Dno = e.Dno)
where e.Salary > 30000
group by e.Dno;
```

Biểu thức lựa chọn (1)

- Biểu thức **case** chỉ định một giá trị cụ thể (nếu giá trị đó có thể khác nhau) dựa trên các điều kiện nhất định.

```
case
     when <điều kiện 1> then <giá trị 1>
     ...
     when <điều kiện n> then <giá trị n>
    [else <giá trị>]
end
```

- Mệnh đề **case** có thể sử dụng
  - Trong mệnh đề **select**, điều kiện chọn của mệnh đề **where** của lệnh truy vấn.
  - Trong lệnh cập nhật dữ liệu **update**, **delete**.

Biểu thức lựa chọn (2)

- ***Q29*** *- Liệt kê danh sách các nhân viên gồm họ tên, ngày sinh, chức vụ. Nếu nhân viên là trưởng bộ phận thì chức vụ là 'Trưởng phòng'.*

```
select e.Fname, e.Lname, e.Bdate,
       case
           when d.Dnumber is not null then 'Truong phong'
       end as 'Chuc vu'
from (EMPLOYEE e left join
      DEPARTMENT d on e.Ssn = d.Mgr_ssn);
```

Biểu thức lựa chọn (3)

- ***Q30*** *- Liệt kê danh sách các nhân viên đến tuổi nghỉ hưu gồm họ tên, ngày sinh, giới tính. Biết qui định tuổi nghỉ hưu cho nam là 60, nữ là 55.*

```
select Fname, Lname, Bdate, Sex
from EMPLOYEE
where year(getdate()) - year(Bdate) >=
      case
          when Sex in ('m', 'M') then 60
          when Sex in ('f', 'F') then 55
      end;
```

Lưu ý: getdate() không phải là hàm SQL chuẩn