



Queries

NỘI DUNG

- Tổng quan
- Cơ sở dữ liệu mẫu
- Ngôn ngữ truy vấn dữ liệu

TỔNG QUAN

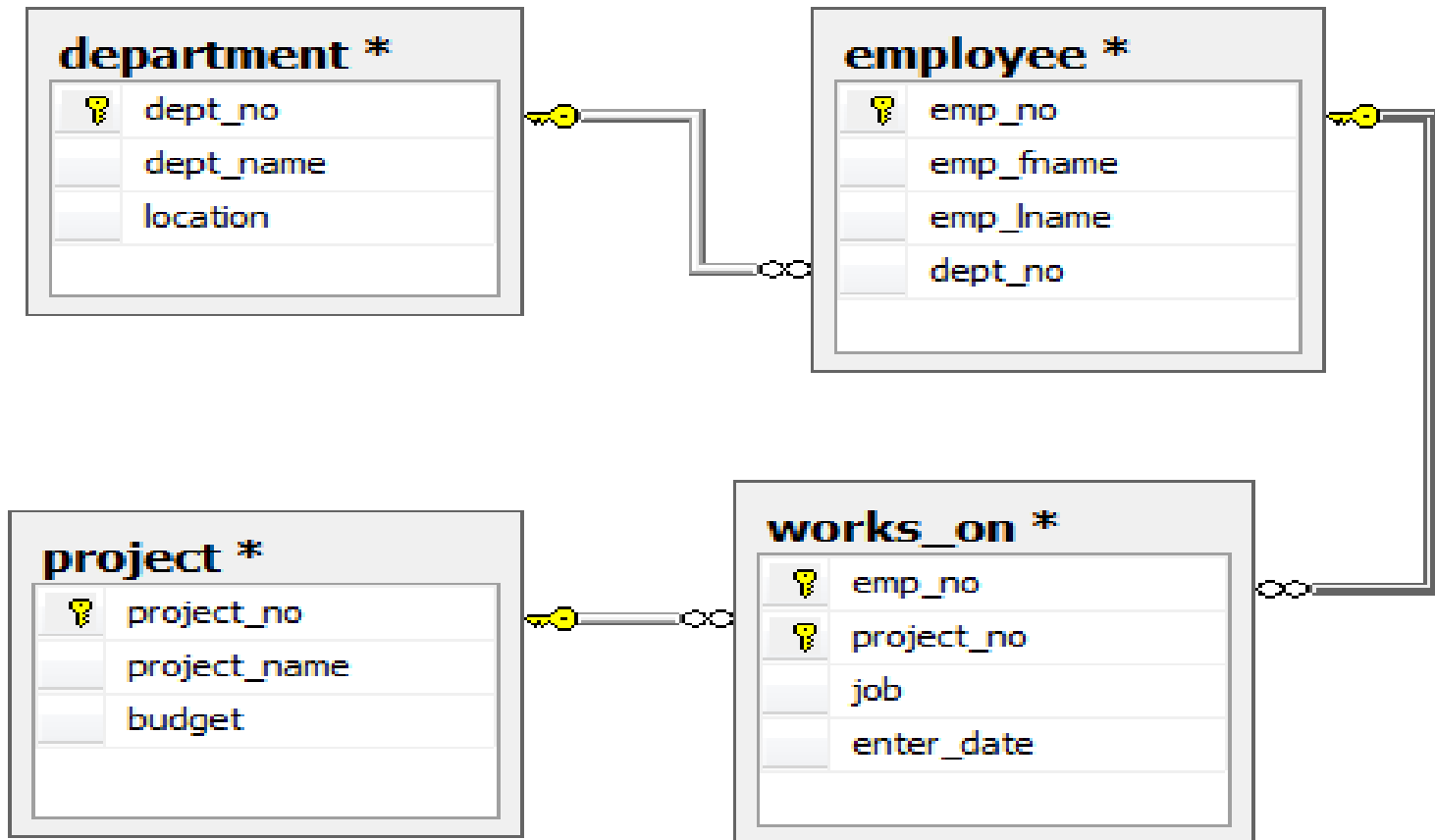
- Mỗi hệ quản trị CSDL đều phải có ngôn ngữ giao tiếp giữa người sử dụng với cơ sở dữ liệu.
- Ngôn ngữ giao tiếp CSDL gồm các loại sau:
 - ***Ngôn ngữ định nghĩa dữ liệu (Data Definition Language – DDL)***
 - Cho phép khai báo cấu trúc các bảng của CSDL, khai báo các mối liên hệ của dữ liệu (relationship) và các quy tắc áp đặt lên các dữ liệu đó.
 - ***Ngôn ngữ thao tác dữ liệu (Data Manipulation Language - DML):***
 - Cho phép Cập nhật: thêm (insert), sửa (update), xóa (delete) dữ liệu
 - ***Ngôn ngữ truy vấn dữ liệu (Structure Query Language):***
 - Cho phép trích lọc dữ liệu

NỘI DUNG

- Tổng quan
- Cơ sở dữ liệu mẫu
- Ngôn ngữ truy vấn dữ liệu

Cơ sở dữ liệu mẫu

- Cơ sở dữ liệu **SAMPLE** có cấu trúc sau



Một số kiểu dữ liệu

Kiểu dữ liệu	Giải thích
Bit	Nhận giá trị 0, 1 hoặc null, 1 bit
Char(n)	Cột có chiều dài cố định chứa chính xác n Byte thông tin (nếu nhập ít hơn thì sẽ đệm các khoảng trắng. Nếu nhập nhiều hơn sẽ xén bỏ)
VarChar(n)	Cột có chiều dài biến đổi chứa tối đa n Byte thông tin. (nếu nhập ít hơn thì sẽ không đệm các khoảng trắng, điều này hiệu quả hơn về bộ nhớ. Nếu nhập nhiều hơn dữ liệu sẽ bị xén bớt)
Nchar(n)	Cột có chiều dài cố định lưu trữ dữ liệu kiểu Unicode
Nvarchar(n)	Cột có chiều dài biến đổi lưu trữ dữ liệu kiểu Unicode
Binary(n)	Cột có chiều dài cố định chứa chính xác n Byte thông tin.
VarBinary(n)	Cột có chiều dài biến đổi chứa tối đa n Byte thông tin.
Text	Lưu dữ liệu text trên 8.000 ký tự
Ntext	Lưu dữ liệu text trên 8.000 ký tự, kiểu Unicode

MỘT SỐ KIỂU DỮ LIỆU (tt)

Kiểu dữ liệu	Giải thích
Tinyint	0 – 255, 1 byte
Smallint	$\pm 32,767$, 2 byte
Int	$\pm 2,147,483,467$, 4 byte
Bigint	$\pm 2^{63}$, 8 byte
Numeric(p [, s])	$\pm 10^{38}$, p : tổng số ký số, s : số ký số lẻ, $0 \leq s \leq p$
Float	$\pm 1.79E+308$
Real	$\pm 3.40E+38$
Datetime	1/1/1753 - 12/31/9999, 8 byte
Smalldatetime	1/1/1900 - 06/06/2079, 4 byte
Money	$\pm 922.337.203.685.447,5808$, 8 byte
Smallmoney	$\pm 214.748,3642$, 4 byte
.....	

NỘI DUNG

- Tổng quan
- Cơ sở dữ liệu mẫu
- **Ngôn ngữ truy vấn dữ liệu**

Ngôn ngữ truy vấn dữ liệu

1. Câu lệnh SELECT cơ bản
2. Truy vấn con
3. Mệnh đề GROUP BY
4. Mệnh đề HAVING
5. Mệnh đề ORDER BY
6. Các phép toán về tập hợp
7. Phép kết

Câu lệnh SELECT cơ bản

```
SELECT [ ALL | DISTINCT] column_list  
FROM {table1 [tab_alias1] } ,...
```

- **table1**: Tên bảng cần truy vấn.
- **tab_alias1**: Đặt tên lại cho table1
- **column_list**: Có thể là một trong các trường hợp sau:
 - Dấu * : Lấy tất cả các cột trong table1
 - Tên cột cần truy xuất
 - Tên cột [AS] đặt lại tên cột
 - Biểu thức hoặc hàm

Câu lệnh SELECT cơ bản

- Ví dụ: Hiển thị tất cả thông tin của bảng department

USE sample;

SELECT dept_no, dept_name, location

FROM department;

- Hoặc sử dụng dấu *

USE sample;

SELECT *

FROM department;

	dept_no	dept_name	location
1	d1	research	Dallas
2	d2	accounting	Seattle
3	d3	marketing	Dallas
4	d4	Development	NULL

SELECT với mệnh đề WHERE

Mệnh đề WHERE dùng để lọc dữ liệu xuất

```
SELECT select_list  
[INTO new_table_]  
FROM table  
[WHERE search_condition]
```

SELECT với mệnh đề WHERE – Ví dụ

- Hiển thị tên và mã phòng ban đối với những phòng ban có địa chỉ ở **Dallas**

USE sample;

SELECT dept_name, dept_no

FROM department

WHERE location = 'Dallas';

	dept_name	dept_no
1	research	d1
2	marketing	d3

SELECT với mệnh đề WHERE

➡ Mệnh đề WHERE có thể chứa các toán tử sau:

<> (hoặc !=)

<

>

>=

<=

!>

!<

SELECT với mệnh đề WHERE – Ví dụ

- Hiển thị tên và họ của nhân viên có mã số là 22222 :

```
SELECT emp_lname, emp_fname  
FROM employee  
WHERE emp_no = 22222;
```

	emp no	emp fname	emp lname	dept no
	11111	Ann ...	Smith ...	d2
	10102	Ann ...	Jones ...	d3
	22222	Matthew ...	Jones ...	d4
	33333	John ...	Barrimore ...	d2
	9031	Elsa	Bertoni ...	d2
	2581	Elke	Hansel ...	d2

	emp_lname	emp_fname
1	Jones	Matthew

SELECT với mệnh đề WHERE – ví dụ

- Hiển thị tên project có ngân sách > 60000£. Biết tỷ giá là 0.51 £ tương đương \$1.

```
SELECT project_name  
FROM project  
WHERE budget*0.51 > 60000;
```

- Hiển thị mã nhân viên của những nhân viên làm việc cho dự án **p1** hoặc **p2**

```
SELECT emp_no  
FROM works_on  
WHERE project_no = 'p1' OR project_no = 'p2';
```


SELECT với DISTINCT

- Nếu kết quả chứa nhiều giá trị trùng nhau, ta có thể dùng từ khóa DISTINCT để lọc trùng:

```
SELECT DISTINCT emp_no
```

```
FROM works_on
```

```
WHERE project_no = 'p1' OR project_no = 'p2';
```

- DISTINCT chỉ dùng 1 lần trong câu lệnh SELECT, và chỉ dùng cho **cột đầu tiên** trong danh sách.

```
SELECT emp_fname, DISTINCT emp_no
```

```
FROM employee
```

```
WHERE emp_lname = 'Moser'
```

SELECT với NOT AND OR

- ▶ Thứ tự thực hiện trong mệnh đề WHERE

NOT → AND → OR

- ▶ Ví dụ

```
SELECT emp_no, emp_fname, emp_lname  
FROM employee  
WHERE emp_no = 25348 AND emp_lname = 'Smith'  
OR emp_fname = 'Matthew' AND dept_no = 'd1';
```

SELECT với IN

- Hiển thị thông tin của tất cả nhân viên có mã số là 29346, 28559, hoặc 25348

```
SELECT emp_no, emp_fname, emp_lname  
FROM employee  
WHERE emp_no IN (29346, 28559, 25348);
```

SELECT với : NOT IN

- Hiển thị thông tin của tất cả nhân viên có mã số khác 10102 và 9031:

```
SELECT emp_no, emp_fname, emp_lname, dept_no  
FROM employee  
WHERE emp_no NOT IN (10102, 9031);
```

SELECT với : BETWEEN ... AND

- Hiển thị tên và ngân sách của những dự án có ngân sách trong khoảng \$95,000 và \$120,000

```
SELECT project_name, budget  
FROM project  
WHERE budget BETWEEN 95000 AND 120000;
```

SELECT với : NOT BETWEEN... AND

- Hiển thị tên và ngân sách của những dự án có ngân sách nhỏ hơn \$100,000 hoặc lớn hơn \$150,000:

```
SELECT project_name  
FROM project  
WHERE budget NOT BETWEEN 100000 AND 150000;
```

SELECT với NULL VALUE

- Hiển thị mã nhân viên và mã dự án tương ứng của những nhân viên thuộc dự án p2 mà chưa được giao công việc:

```
SELECT emp_no, project_no  
FROM works_on  
WHERE project_no = 'p2'  
AND job IS NULL;
```

SELECT với NULL VALUE

- ▶ Trong trường hợp lọc dữ liệu với điều kiện khác rỗng ta dùng IS NOT NULL.

```
SELECT project_no, job  
FROM works_on  
WHERE job <> NULL; → Sai
```


→ Đúng: **job IS NOT NULL**

SELECT với NULL VALUE

- Hàm hệ thống ISNULL cho phép hiển thị giá trị thay thế cho NULL

```
SELECT emp_no, ISNULL(job, 'Job unknown') AS task  
FROM works_on  
WHERE project_no = 'p1';
```

	emp_no	project_no	job	enter_date
1	10102	p1	NULL	2006-10-01
2	10102	p3	NULL	2008-01-01
3	25348	p2	clerk	2007-02-15
4	22334	p2	NULL	2007-01-15
5	29346	p2	NULL	2006-12-15
6	2581	p3	analyst	2007-10-15
7	22335	p2	NULL	2007-01-15
8	9031	p3	clerk	2006-11-15



	emp_no	task
1	10102	Job unknown

SELECT với : LIKE Operator

- Dạng căn bản

column [NOT] LIKE 'pattern'

- **pattern** có thể là chuỗi hoặc biểu thức
- Hai ký tự đại diện dùng trong pattern là:
 - **%** : Đại diện cho chuỗi ký tự
 - **_** (underscore) Đại diện cho 1 ký tự

SELECT với LIKE Operator

- Hiển thị thông tin những nhân viên có *ký tự thứ hai* của tên là chữ **a**:

```
SELECT emp_fname, emp_lname, emp_no  
FROM employee  
WHERE emp_fname LIKE '_a%';
```

SELECT với LIKE

- Cặp ngoặc vuông **[]** xác định dãy ký tự

- Hiển thị thông tin tất cả các phòng ban có địa chỉ bắt đầu bởi một trong các ký tự từ C đến F:

```
SELECT *  
FROM department  
WHERE location LIKE '[C-F]%';
```

- Ký tự **^** dùng phủ định dãy ký tự.

- Hiển thị thông tin những nhân viên có họ không bắt đầu bởi các ký tự J, K, L, M, N, hoặc O và tên không bắt đầu bởi E hoặc Z:

```
SELECT emp_no, emp_fname, emp_lname  
FROM employee  
WHERE emp_lname LIKE '[^J-O]%'  
AND emp_fname LIKE '[^EZ]%';
```

SELECT với NOT LIKE

- Hiển thị thông tin của những nhân viên mà tên không kết thúc bởi ký tự **n**:

```
SELECT emp_no, emp_fname, emp_lname  
FROM employee  
WHERE emp_fname NOT LIKE '%n';
```

Bài tập

Sử dụng database **Sample**, thực hiện các truy vấn sau:

1. Hiển thị tất cả thông tin của bảng `employee`
2. Hiển thị mã nhân viên của nv có job là *analyst* trong bảng `works_on`.
3. Hiển thị mã nhân viên của nv thuộc dự án `p1` và có mã nhân viên < 12000 .
4. Hiển thị mã nhân viên của nv tham gia vào các dự án trong năm 2006 nhưng chưa được giao việc.
5. Hiển thị mã nhân viên của lãnh đạo (Analyst hoặc Manager) thuộc dự án `p1`.
6. Hiển thị ngày bắt đầu làm việc của nhân viên thuộc dự án `p2` mà chưa được giao việc.
7. Hiển thị tất cả thông tin của nv có tên kết thúc bởi ký tự `n`.
8. Hiển thị tất cả thông tin của nv có họ bắt đầu bởi 2 ký tự `el` và ký tự thứ tư là **a** hoặc **e**.
9. Hiển thị tất cả thông tin của các phòng ban có địa chỉ ở Seattle.
10. Hiển thị mã nhân viên tham gia các dự án trong năm 2007.

Ngôn ngữ truy vấn dữ liệu

1. Câu lệnh SELECT cơ bản
2. Truy vấn con
3. Mệnh đề GROUP BY
4. Mệnh đề HAVING
5. Mệnh đề ORDER BY
6. Các phép toán về tập hợp
7. Phép kết

Truy vấn con (Subqueries)

- Câu lệnh SELECT được đặt trong mệnh đề WHERE gọi là truy vấn con.
- Câu lệnh truy vấn con thường được thực hiện trước
- Truy vấn con thường dùng với các toán tử sau:
 1. Toán tử so sánh
 2. Toán tử **IN**
 3. EXISTS

Subqueries và toán tử so sánh

- Ví dụ: Cho biết thông tin về dự án có ngân sách (budget) lớn nhất

```
SELECT *  
FROM project  
WHERE budget = (select MAX(budget) from project)
```

Subqueries và IN

- Ví dụ: Lập danh sách nhân viên thuộc phòng ban có địa chỉ ở Dallas:

```
SELECT *  
FROM employee  
WHERE dept_no IN (SELECT dept_no  
                  FROM department  
                  WHERE location = 'Dallas');
```

Ngôn ngữ truy vấn dữ liệu

1. Câu lệnh SELECT cơ bản
2. Truy vấn con
3. Mệnh đề GROUP BY
4. Mệnh đề HAVING
5. Mệnh đề ORDER BY
6. Các phép toán về tập hợp
7. Phép kết

Mệnh đề GROUP BY

- Mệnh đề GROUP BY dùng để nhóm dữ liệu theo yêu cầu truy vấn.

- Lập danh sách tất cả các chức danh của từng project:

```
SELECT project_no, job
```

```
FROM works_on
```

```
GROUP BY project_no, job;
```

Các hàm dùng trong group by - Aggregate Functions

1. MIN
2. MAX
3. SUM
4. AVG
5. COUNT

Aggregate Functions

- Nếu sử dụng Aggregate Functions thì trong câu lệnh select không thể chọn thêm cột khác
- Ví dụ:

```
USE sample;
```

```
SELECT emp_lname, MIN(emp_no)
```

```
FROM employee; → Sai
```

→ Sửa lại, bỏ cột emp_lname

```
USE sample;
```

```
SELECT MIN(emp_no)
```

```
FROM employee;
```

Aggregate Functions

- Nếu sử dụng mệnh đề **group by** thì tất cả các cột trong danh sách SELECT (ngoại trừ aggregate functions) phải xuất hiện trong danh sách group by
- Ví dụ: Đếm số nhân viên của từng phòng ban

USE sample;

SELECT dept_no, count(emp_no) AS SLNV

From employee

Group by dept_no

MIN and MAX – Ví dụ

- ➡ Cho biết tên và mã nhân viên của nhân viên có mã nhân viên nhỏ nhất:

```
SELECT emp_no, emp_lname  
FROM employee  
WHERE emp_no = (SELECT MIN(emp_no)  
                FROM employee);
```

- ➡ Cho biết mã nhân viên tham gia vào dự án trễ nhất:

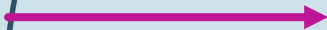
```
SELECT emp_no  
FROM works_on  
WHERE enter_date =(SELECT MAX(enter_date)  
                   FROM works_on);
```


SUM – Ví dụ

- Tính tổng ngân sách của tất cả các dự án:

```
SELECT SUM (budget) sum_of_budgets  
FROM project;
```

	project_no	project_name	budget
1	p1	Apollo	120000
2	p2	Gemini	95000
3	p3	Mercury	186500



	sum_of_budgets
1	401500

AVG – Ví dụ

- Tính ngân sách bình quân của các dự án có ngân sách lớn hơn \$100,000:

```
SELECT AVG(budget) avg_budget  
FROM project  
WHERE budget > 100000;
```

	project_no	project_name	budget
1	p1	Apollo	120000
2	p2	Gemini	95000
3	p3	Mercury	186500



	avg_budget
1	153250

COUNT – Ví dụ

- Đếm số chức danh khác nhau của từng dự án:

```
SELECT project_no, COUNT(DISTINCT job) job_count  
FROM works_on  
GROUP BY project_no;
```

- Cho biết số lượng nhân viên của từng chức danh trong tất cả các dự án:

```
SELECT job, COUNT(*) job_count  
FROM works_on  
GROUP BY job;
```

Ngôn ngữ truy vấn dữ liệu

1. Câu lệnh SELECT cơ bản
2. Truy vấn con
3. Mệnh đề GROUP BY
4. Mệnh đề HAVING
5. Mệnh đề ORDER BY
6. Các phép toán về tập hợp
7. Phép kết

Mệnh đề HAVING

- Lọc dữ liệu sau khi group by.
- Ví dụ: Cho biết mã dự án của những dự án có số lượng nhân viên ít hơn 4 người:

```
SELECT project_no  
FROM works_on  
GROUP BY project_no  
HAVING COUNT(*) < 4;
```

	emp_no	project_no	job	enter_date
1	10102	p1	analyst	2006-10-01
2	10102	p3	manager	2008-01-01
3	25348	p2	clerk	2007-02-15
4	18316	p2	NULL	2007-06-01
5	29346	p2	NULL	2006-12-15
6	2581	p3	analyst	2007-10-15
7	9031	p1	manager	2007-04-15
8	28559	p1	NULL	2007-08-01
9	28559	p2	clerk	2008-02-01
10	9031	p3	clerk	2006-11-15
11	29346	p1	clerk	2007-01-04



	project_no
1	p3

Ngôn ngữ truy vấn dữ liệu

1. Câu lệnh SELECT cơ bản
2. Truy vấn con
3. Mệnh đề GROUP BY
4. Mệnh đề HAVING
5. Mệnh đề ORDER BY
6. Các phép toán về tập hợp
7. Phép kết

Mệnh đề ORDER BY

- Sắp xếp kết quả truy vấn.

ORDER BY {[col_name | col_number [ASC | DESC]]}

- Ví dụ

```
SELECT emp_fname, emp_lname, dept_no  
FROM employee  
WHERE emp_no < 20000  
ORDER BY emp_lname, emp_fname;
```

Ngôn ngữ truy vấn dữ liệu

1. Câu lệnh SELECT cơ bản
2. Truy vấn con
3. Mệnh đề GROUP BY
4. Mệnh đề HAVING
5. Mệnh đề ORDER BY
6. Các phép toán về tập hợp
7. Phép kết

Các phép toán về tập hợp

1. UNION : Phép hội
2. INTERSECT: Phép giao
3. EXCEPT: Phép trừ

UNION

➤ Cú pháp

```
select_1 UNION [ALL] select_2 {[UNION [ALL] select_3]}...
```

- **select_1, select_2,...**: câu lệnh select dùng rút trích dữ liệu
- **ALL** : Không lọc trùng.
- Cho biết mã nhân viên của những nhân viên thuộc phòng ban **d1** hoặc tham gia vào các dự án trước ngày **01.01.2007**:

```
SELECT emp_no  
FROM employee  
WHERE dept_no = 'd1'
```

UNION

```
SELECT emp_no  
FROM works_on  
WHERE enter_date < '01.01.2007'
```

UNION

	emp_no	emp_fname	emp_lname	dept_no
1	25348	Matthew	Smith	d3
2	10102	Ann	Jones	d3
3	18316	John	Barimore	d1
4	29346	James	James	d2
5	9031	Elsa	Bertoni	d2
6	2581	Elke	Hansel	d2
7	28559	Sybill	Moser	d1



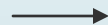
```
SELECT emp_no  
FROM  
employee  
WHERE  
dept_no = 'd1'
```



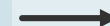
	emp_no
1	18316
2	28559

UNION

	emp_no	project_no	job	enter_date
1	10102	p1	analyst	2006-10-01
2	10102	p3	manager	2008-01-01
3	25348	p2	clerk	2007-02-15
4	18316	p2	NULL	2007-06-01
5	29346	p2	NULL	2006-12-15
6	2581	p3	analyst	2007-10-15
7	9031	p1	manager	2007-04-15
8	28559	p1	NULL	2007-08-01
9	28559	p2	clerk	2008-02-01
10	9031	p3	clerk	2006-11-15
11	29346	p1	clerk	2007-01-04



```
SELECT emp_no  
FROM works_on  
WHERE  
enter_date <  
'01.01.2007'
```



	emp_no
1	10102
2	29346
3	9031



	emp_no
1	9031
2	10102
3	18316
4	28559
5	29346

INTERSECT – Ví dụ

- Cho biết mã nhân viên của những nhân viên thuộc phòng ban **d1** và tham gia vào các dự án trước ngày **08.07.2007**

```
SELECT emp_no  
FROM employee  
WHERE dept_no = 'd1'
```

INTERSECT

```
SELECT emp_no  
FROM works_on  
WHERE enter_date < '08.07.2007';
```

	emp_no	emp_fname	emp_lname	dept_no
1	25348	Matthew	Smith	d3
2	10102	Ann	Jones	d3
3	18316	John	Barimore	d1
4	29346	James	James	d2
5	9031	Elsa	Bertoni	d2
6	2581	Elke	Hansel	d2
7	28559	Sybill	Moser	d1

```
SELECT emp_no
FROM
employee
WHERE
dept_no = 'd1'
```

	emp_no
1	18316
2	28559

INTERSECT

```
SELECT emp_no
FROM
works_on
WHERE
enter_date <
'08.07.2007';
```

	emp_no	project_no	job	enter_date
1	10102	p1	analyst	2006-10-01
2	10102	p3	manager	2008-01-01
3	25348	p2	clerk	2007-02-15
4	18316	p2	NULL	2007-06-01
5	29346	p2	NULL	2006-12-15
6	2581	p3	analyst	2007-10-15
7	9031	p1	manager	2007-04-15
8	28559	p1	NULL	2007-08-01
9	28559	p2	clerk	2008-02-01
10	9031	p3	clerk	2006-11-15
11	29346	p1	clerk	2007-01-04

	emp_no
1	10102
2	25348
3	18316
4	29346
5	9031
6	28559
7	9031
8	29346

EXCEPT – Ví dụ

```
SELECT emp_no  
FROM employee  
WHERE dept_no = 'd1'
```

EXCEPT

```
SELECT emp_no  
FROM works_on  
WHERE enter_date < '01.01.2007';
```

Bài tập

1. Lập danh sách địa chỉ của các phòng ban
2. Cho biết thông tin về nhân viên có mã số nhân viên nhỏ nhất
3. Lập danh sách dự án có ngân sách (budget) lớn hơn ngân sách trung bình
4. Cho biết các công việc (job) được thực hiện bởi nhiều hơn 2 nhân viên
5. Cho biết mã nhân viên của tất cả thư ký (clerk) làm việc cho phòng ban d3
6. Cho biết tên của các dự án (project_name) có hơn 2 thư ký đang làm việc
7. Cho biết mã (emp_no) của những nhân viên thuộc phòng Marketing

Ngôn ngữ truy vấn dữ liệu

1. Câu lệnh SELECT cơ bản
2. Truy vấn con
3. Mệnh đề GROUP BY
4. Mệnh đề HAVING
5. Mệnh đề ORDER BY
6. Các phép toán về tập hợp
7. Phép kết

Phép kết - Jions

1. Natural Join
2. Cartesian product
3. Outer Join
4. Theta Join
5. Self-Join
6. So sánh Joins và Subqueries

Natural Join – Ví dụ

- Liệt kê danh sách nhân viên gồm tất cả thông tin về nhân viên và phòng ban của nhân viên đó.

```
SELECT employee.*, department.*
```

```
FROM employee INNER JOIN department
```

```
ON employee.dept_no = department.dept_no;
```

	emp_no	emp_fname	emp_lname	dept_no	dept_no	dept_name	location
1	25348	Matthew	Smith	d3	d3	marketing	Dallas
2	10102	Ann	Jones	d3	d3	marketing	Dallas
3	18316	John	Bamimore	d1	d1	research	Dallas
4	29346	James	James	d2	d2	accounting	Seattle
5	9031	Elsa	Bertoni	d2	d2	accounting	Seattle
6	2581	Elke	Hansel	d2	d2	accounting	Seattle
7	28559	Sybill	Moser	d1	d1	research	Dallas

Natural Join – Ví dụ

- Lập danh sách nhân viên thuộc dự án Gemini, gồm các thông tin: mã nv, họ&tên nv, tên dự án, chức danh, ngày vào làm việc

```
SELECT e.emp_no,e.emp_fname,e.emp_lname, p.project_name ,w.job,  
w.enter_date  
FROM works_on w JOIN project p   ON p.project_no = w.project_no  
      JOIN employee e ON e.emp_no=w.emp_no  
WHERE project_name = 'Gemini';
```

	emp_no	emp_fname	emp_lname	project_name	job	enter_date
1	25348	Matthew	Smith	Gemini	clerk	2007-02-15 00:00:00.000
2	18316	John	Bamimore	Gemini	NULL	2007-06-01 00:00:00.000
3	29346	James	James	Gemini	NULL	2006-12-15 00:00:00.000
4	28559	Sybill	Moser	Gemini	clerk	2008-02-01 00:00:00.000

Natural Join – Ví dụ

1. Lập danh sách nhân viên tham gia vào các dự án trước ngày 15/10/2007 ?
1. Cho biết tên nhân viên có chức danh analysts và có địa chỉ phòng ban ở Seattle?
2. Cho biết tên của những project có nhân viên thuộc phòng kế toán (`dept_name = 'Accounting'`)?

Phép kết - Jions

1. Natural Join
2. Cartesian product
3. Outer Join
4. Theta Join
5. Self-Join
6. So sánh Joins và Subqueries

Cartesian product – Ví dụ

```
SELECT employee.*, department.*  
FROM employee CROSS JOIN department;
```

	emp_no	emp_fname	emp_lname	dept_no
1	25348	Matthew	Smith	d3
2	10102	Ann	Jones	d3
3	18316	John	Barimore	d1
4	29346	James	James	d2
5	9031	Elsa	Bertoni	d2
6	2581	Elke	Hansel	d2
7	28559	Sybill	Moser	d1

	dept_no	dept_name	location
1	d1	research	Dallas
2	d2	accounting	Seattle
3	d3	marketing	Dallas

	emp_no	emp_fname	emp_lname	dept_no	dept_no	dept_name	location
1	25348	Matthew	Smith	d3	d1	research	Dallas
2	10102	Ann	Jones	d3	d1	research	Dallas
3	18316	John	Barimore	d1	d1	research	Dallas
4	29346	James	James	d2	d1	research	Dallas
5	9031	Elsa	Bertoni	d2	d1	research	Dallas
6	2581	Elke	Hansel	d2	d1	research	Dallas
7	28559	Sybill	Moser	d1	d1	research	Dallas
8	25348	Matthew	Smith	d3	d2	accounting	Seattle
9	10102	Ann	Jones	d3	d2	accounting	Seattle
10	18316	John	Barimore	d1	d2	accounting	Seattle
11	29346	James	James	d2	d2	accounting	Seattle
12	9031	Elsa	Bertoni	d2	d2	accounting	Seattle
13	2581	Elke	Hansel	d2	d2	accounting	Seattle
14	28559	Sybill	Moser	d1	d2	accounting	Seattle
15	25348	Matthew	Smith	d3	d3	marketing	Dallas
16	10102	Ann	Jones	d3	d3	marketing	Dallas
17	18316	John	Barimore	d1	d3	marketing	Dallas
18	29346	James	James	d2	d3	marketing	Dallas
19	9031	Elsa	Bertoni	d2	d3	marketing	Dallas
20	2581	Elke	Hansel	d2	d3	marketing	Dallas
21	28559	Sybill	Moser	d1	d3	marketing	Dallas

Phép kết - Jions

1. Natural Join
2. Cartesian product
3. Outer Join
4. Theta Join
5. Self-Join
6. So sánh Joins và Subqueries

Outer Join

```
SELECT employee_enh.*, department.location  
FROM employee_enh JOIN department  
ON domicile = location;
```

	emp_no	emp_fname	emp_lname	dept_no	domicile
1	25348	Matthew	Smith	d3	San Antonio
2	10102	Ann	Jones	d3	Houston
3	18316	John	Barimore	d1	San Antonio
4	29346	James	James	d2	Seattle
5	9031	Elsa	Bertoni	d2	Portland
6	2581	Elke	Hansel	d2	Tacoma
7	28559	Sybill	Moser	d1	Houston

	dept_no	dept_name	location
1	d1	research	Dallas
2	d2	accounting	Seattle
3	d3	marketing	Dallas



	emp_no	emp_fname	emp_lna...	dept_no	domicile	location
1	29346	James	James	d2	Seattle	Seattle

Left Outer Join

```
SELECT employee_enh.*, department.location
```

```
FROM employee_enh LEFT OUTER JOIN department
```

```
ON domicile = location;
```

	emp_no	emp_fname	emp_lname	dept_no	domicile	location
1	25348	Matthew	Smith	d3	San Antonio	NULL
2	10102	Ann	Jones	d3	Houston	NULL
3	18316	John	Barrimore	d1	San Antonio	NULL
4	29346	James	James	d2	Seattle	Seattle
5	9031	Elsa	Bertoni	d2	Portland	NULL
6	2581	Elke	Hansel	d2	Tacoma	NULL
7	28559	Sybill	Moser	d1	Houston	NULL

Right Outer Join

```
SELECT employee_enh.domicile, department.*
```

```
FROM employee_enh RIGHT OUTER JOIN department
```

```
ON domicile =location;
```

	domicile	dept_no	dept_name	location
1	NULL	d1	research	Dallas
2	Seattle	d2	accounting	Seattle
3	NULL	d3	marketing	Dallas

Phép kết - Jions

1. Natural Join
2. Cartesian product
3. Outer Join
4. Theta Join
5. Self-Join
6. So sánh Joins và Subqueries

Theta Join

```
SELECT emp_fname, emp_lname, domicile, location  
FROM employee_enh JOIN department  
ON domicile < location
```

	emp_fname	emp_lname	domicile	location
1	Matthew	Smith	San Antonio	Seattle
2	Ann	Jones	Houston	Seattle
3	John	Barrimore	San Antonio	Seattle
4	Elsa	Bertoni	Portland	Seattle
5	Sybill	Moser	Houston	Seattle

Phép kết - Jions

1. Natural Join
2. Cartesian product
3. Outer Join
4. Theta Join
5. Self-Join
6. So sánh Joins và Subqueries

Self-Join

- Hiển thị thông tin tất cả các phòng ban có cùng địa chỉ:

```
SELECT t1.dept_no, t1.dept_name, t1.location  
FROM department t1 JOIN department t2  
ON t1.location = t2.location  
WHERE t1.dept_no <> t2.dept_no
```

	dept_no	dept_name	location
1	d3	marketing	Dallas
2	d1	research	Dallas

Phép kết - Joins

1. Natural Join
2. Cartesian product
3. Outer Join
4. Theta Join
5. Self-Join
6. So sánh Joins và Subqueries

Joins hay Subqueries?

1. Subqueries

Dùng Subqueries thuận tiện hơn khi chỉ truy vấn trên cùng một bảng và trong câu lệnh có sử dụng aggregate functions.

- Ví dụ: Cho biết mã nhân viên và ngày bắt đầu làm việc của những nhân viên tham gia vào các dự án sớm nhất:

```
SELECT emp_no, enter_date
FROM works_on
WHERE enter_date = (SELECT min(enter_date)
                    FROM works_on)
```


Joins hay Subqueries?

2. Joins

Sử dụng join khi trong danh sách select liên quan đến nhiều bảng

► Ví dụ: Cho biết thông tin nhân viên tham gia vào dự án ngày 15/10/2007:

```
SELECT employee.emp_no, emp_lname, job  
FROM employee, works_on  
WHERE employee.emp_no = works_on.emp_no  
AND enter_date = '10.15.2007';
```

Bài tập

1. Cho biết mã nhân viên, chức danh của những nhân viên làm việc cho dự án Gemini.
2. Cho biết tên nhân viên làm việc cho phòng Research hoặc Accounting.
3. Cho biết ngày bắt đầu làm việc của các thư ký thuộc phòng d1.
4. Cho biết tên nhân viên là managers của dự án Mercury.
5. Cho biết tên nhân viên tham gia vào các dự án cùng ngày.
6. Cho biết mã của những nhân viên ở cùng địa chỉ và làm việc cùng một phòng ban
7. Cho biết mã của những nhân viên thuộc phòng Marketing bằng 2 cách:
 - a. Dùng JOIN operator
 - b. Dùng Subquery