



HỌC VIỆN KỸ THUẬT MẬT MÃ

KHOA CÔNG NGHỆ THÔNG TIN

Bài giảng: Truy vấn dữ liệu



ThS. Thái Thanh Vân
TTVanCNTT@gmail.com

“Tri thức là chìa khóa mở cánh cửa tương lai”



Truy vấn trong cơ sở dữ liệu

Truy vấn từ một bảng

Truy vấn từ nhiều bảng



Truy vấn từ một bảng trong cơ sở dữ liệu



Câu lệnh SELECT

- Ý nghĩa Lệnh **SELECT** là lệnh cho phép truy vấn dữ liệu mà không làm thay đổi dữ liệu hoặc các đối tượng trong CSDL (database).

- Cú pháp

```
SELECT [DISTINCT / Top n / *] <biểu thức/ cột [AS <tên mới>],...>  
[INTO <tên bảng mới>]  
FROM <tên bảng> [<bí danh>],...  
[WHERE <điều kiện chọn>]  
[GROUP BY <ds tên cột gom nhóm>]  
[HAVING <điều kiện lọc nhóm>]  
[ORDER BY <tên cột>[ASC/DESC],...];
```



Truy vấn đơn giản

➤ Cú pháp

SELECT < danh sách các cột >

FROM < danh sách các bảng >

[WHERE < biểu thức điều kiện >]

Trong đó

< danh sách các cột >:

Tên các thuộc tính (cột) sẽ được hiển thị trong kết quả truy vấn.

< danh sách các bảng >:

Tên các bảng liên quan để lấy kết quả

< biểu thức điều kiện >:

là điều kiện đưa vào để chọn lọc dữ liệu, thường gồm:

- ✓ Các phép toán so sánh: < , > , ≤ , ≥ , ≠ , =
- ✓ Các phép toán logic: AND, OR, và NOT
- ✓ Các từ khóa: BETWEEN ... AND, IN, EXISTS, LIKE...



Câu lệnh Select

1. Dùng **SeLect** để chọn tất cả các đối tượng trong bảng

➤ **Cú pháp** **SELECT * FROM <tenbang>**

➤ **Ví dụ** **SELECT * FROM SINHVIEN**

➤ **Kết quả**

	MaSV	HOVATEN	NAMSINH	SDT	QUEQUAN
1	1	Nguyễn Văn A	1998-10-09	0169881689	Hà Nam
2	2	Hoàng Mai B	1998-09-09	0163481689	Hà Nội
3	3	Nguyễn Thị C	1998-08-12	096988169	Thái Bình
4	4	Lê Thị D	1998-03-15	096341356	Nam Định
5	5	Trần Mai K	1998-05-09	0985613589	Thái Bình
6	6	Lưu Văn H	1998-09-20	0984562579	Hà Nội



Câu lệnh Select

2. Dùng **Select** để chọn một số cột trong bảng

➤ **Cú pháp**

SELECT <tencot1>, <tencot2> ...

FROM <tenbang>

➤ **Ví dụ**

SELECT HOVATEN FROM SINHVIEN

➤ **Kết quả**

Results		Messages
HOVATEN		
1	Nguyễn Văn A	
2	Hoàng Mai B	
3	Nguyễn Thị C	
4	Lê Thị D	
5	Trần Mai K	
6	Lưu Văn H	



Truy vấn đơn giản

Một số toán tử trong SQL

- DISTINCT
- IN
- LIKE
- IS [NOT] NULL
- BETWEEN
- EXISTS,...

Một số hàm trong SQL

- Hàm toán học:

- ROUND
- POWER
- CEILING
- SQRT,...
- FLOOR

- Hàm tập hợp:

- SUM(),
- AVG()
- COUNT()
- MAX(),...
- MIN()



Loại trừ dữ liệu lặp với DISTINCT

➤ Cú pháp

SELECT DISTINCT <tencot> **FROM** <tenbang>

➤ Ví dụ

MaSV	HOVATEN	LOP	NAMSINH	SDT	QUEQUAN
1	Nguyễn Văn A	AT10A	1998-10-09	0169881689	Hà Nam
2	Hoàng Mai B	AT10B	1998-09-09	0163481689	Hà Nội
3	Nguyễn Thị C	AT10A	1998-08-12	096988169	Thái Bình
4	Lê Thị D	AT10A	1998-03-15	096341356	Nam Định
5	Trần Mai K	AT10B	1998-05-09	0985613589	Thái Bình
6	Lưu Văn H	AT10A	1998-09-20	0984562579	Hà Nội

SELECT LOP FROM SINHVIEN

SELECT DISTINCT LOP FROM SINHVIEN

➤ Kết quả

	LOP
1	AT10A
2	AT10B
3	AT10A
4	AT10A
5	AT10B
6	AT10A

➤ Kết quả

	LOP
1	AT10A
2	AT10B



Lấy dữ liệu trong một khoảng với BETWEEN

➤ Cú pháp

SELECT <col1>, <col2> ...

FROM <bang>

WHERE <ten_col> *BETWEEN* giatri1 *AND* giatri2

➤ Ví dụ

SELECT *

FROM SINHVIEN

WHERE MaSV *BETWEEN* 5 *AND* 10



Tìm kiếm gần đúng với LIKE

➤ Cú pháp

```
SELECT <tencot1>, <tencot2> ...  
FROM <tenbang>  
WHERE <tencot> LIKE <du_lieu_mau>
```

Trong đó:

<du_lieu_mau> đặt sau từ khóa LIKE dùng để đại diện gần chính xác cho một dữ liệu mẫu, thông qua hai ký tự % và _

% Biểu thị một hoặc nhiều ký tự, hoặc thể hiện ký tự bằng 0.

_ Biểu thị một ký tự đơn



Một số dạng tìm kiếm gần đúng

Dạng tìm kiếm	Mệnh đề WHERE
Tìm kiếm dữ liệu bắt đầu bằng ký tự K	...WHERE <column> LIKE 'K%'
Tìm kiếm dữ liệu kết thúc bằng ký tự K	...WHERE <column> LIKE '%K'
Tìm kiếm dữ liệu có chứa ký tự Kt ở vị trí bất kỳ	...WHERE <column> LIKE '%Kt%'
Tìm kiếm dữ liệu có ký tự K ở vị trí thứ hai	...WHERE <column> LIKE '_K%'
Tìm kiếm dữ liệu bắt đầu bằng ký tự K, và có ít nhất có chiều dài là 3 ký tự	...WHERE <column> LIKE 'K_%_%'
Tìm kiếm dữ liệu bắt đầu bằng ký tự K, kết thúc bằng ký tự m	...WHERE <column> LIKE 'K%m'

➤ **Ví dụ** Xuất ra thông tin sinh viên mà tên bắt đầu bằng chữ M

```
SELECT * FROM SINHVIEN WHERE TEN Like 'M%'
```



Liên kết các cột dữ liệu trong kết quả truy vấn

➤ Cú pháp

```
SELECT <tencot> + 'string' + <tencot> + 'string' + <tencot>...  
FROM <tenbang>
```

➤ Ví dụ

```
SELECT HOVATEN + (N' học lớp ') + LOP + (N' quê ở ') +  
QUEQUAN FROM SINHVIEN
```

```
SELECT HOVATEN + (N' học lớp ') + LOP + (N' quê ở ') +  
QUEQUAN AS 'Thông tin sinh viên' FROM SINHVIEN
```

➤ Kết quả

	(No column name)
1	Nguyễn Văn A học lớp AT10A quê ở Hà Nam
2	Hoàng Mai B học lớp AT10B quê ở Hà Nội
3	Nguyễn Thị C học lớp AT10A quê ở Thái Bình
4	Lê Thị D học lớp AT10A quê ở Nam Định
5	Trần Mai K học lớp AT10B quê ở Thái Bình
6	Lưu Văn H học lớp AT10A quê ở Hà Nội

	Thông tin sinh viên
1	Nguyễn Văn A học lớp AT10A quê ở Hà Nam
2	Hoàng Mai B học lớp AT10B quê ở Hà Nội
3	Nguyễn Thị C học lớp AT10A quê ở Thái Bình
4	Lê Thị D học lớp AT10A quê ở Nam Định
5	Trần Mai K học lớp AT10B quê ở Thái Bình
6	Lưu Văn H học lớp AT10A quê ở Hà Nội



Giá trị được tính toán trong danh sách lựa chọn

➤ **Ví dụ** Tính điểm trung bình của mỗi sinh viên trong bảng sau

MaSV	TenSV	DToan	DLy	DHoa
1	Hoàng Mai Anh	9	9	9
2	Mai Thu Hương	8	9	10
3	Trần Thu Thủy	7	9	7
4	Đỗ Mạnh Hùng	8	6	7
5	Lê Duy Minh	7	9	8
6	Nguyễn Việt Quang	9	10	10

```
SELECT TenSV AS 'Họ và tên', Dtoan AS 'Điểm Toán', DLy AS 'Điểm Lý',  
Dhoa AS 'Điểm Hóa', (DToan+DLy+Dhoa)/3.0 AS 'Điểm trung bình' FROM  
DIEMTHI
```

➤ **Kết quả**

	Họ và tên	Điểm Toán	Điểm Lý	Điểm Hóa	Điểm trung bình
1	Hoàng Mai Anh	9	9	9	9.000000
2	Mai Thu Hương	8	9	10	9.000000
3	Trần Thu Thủy	7	9	7	7.666666
4	Đỗ Mạnh Hùng	8	6	7	7.000000
5	Lê Duy Minh	7	9	8	8.000000
6	Nguyễn Việt Quang	9	10	10	9.666666



Hạn chế tập kết quả bằng việc sử dụng TOP và PERCENT

1. Lấy n hàng trong bản ghi với TOP

➤ **Cú pháp** `SELECT TOP n <tencot> FROM <tenbang>`

➤ **Ví dụ** Lấy thông tin 3 sinh viên đầu tiên trong bảng dữ liệu

`SELECT TOP 3 * FROM SINHVIEN`

➤ **Kết quả**

	MaSV	HOVATEN	LOP	NAMSINH	SDT	QUEQUAN
1	1	Nguyễn Văn A	AT10A	1998-10-09	0169881689	Hà Nam
2	2	Hoàng Mai B	AT10B	1998-09-09	0163481689	Hà Nội
3	3	Nguyễn Thị C	AT10A	1998-08-12	096988169	Thái Bình

2. Lấy n hàng trong bản ghi với PERCENT

➤ **Cú pháp** `SELECT TOP n PERCENT <tencot> FROM <tenbang>`

➤ **Ví dụ** Lấy 20% số sinh viên đầu tiên trong bảng dữ liệu

`SELECT TOP 20 PERCENT * FROM SINHVIEN`

➤ **Kết quả**

	MaSV	HOVATEN	LOP	NAMSINH	SDT	QUEQUAN
1	1	Nguyễn Văn A	AT10A	1998-10-09	0169881689	Hà Nam
2	2	Hoàng Mai B	AT10B	1998-09-09	0163481689	Hà Nội



Sử dụng mệnh đề WHERE

- **Cú pháp** *SELECT* <tencot>
FROM <tenbang>
WHERE <dieukien>

- **Ví dụ** Chọn các sinh viên có quê quán là Hà Nội và in ra một bảng mới

	MaSV	HOVATEN	NAMSINH	SDT	QUEQUAN
1	1	Nguyễn Văn A	1998-10-09	0169881689	Hà Nam
2	2	Hoàng Mai B	1998-09-09	0163481689	Hà Nội
3	3	Nguyễn Thị C	1998-08-12	096988169	Thái Bình
4	4	Lê Thị D	1998-03-15	096341356	Nam Định
5	5	Trần Mai K	1998-05-09	0985613589	Thái Bình
6	6	Lưu Văn H	1998-09-20	0984562579	Hà Nội

SELECT * *FROM* *SINHVIEN* *WHERE* *QUEQUAN* = 'Hà Nội'

- **Kết quả**

	MaSV	HOVATEN	LOP	NAMSINH	SDT	QUEQUAN
1	2	Hoàng Mai B	AT10B	1998-09-09	0163481689	Hà Nội
2	6	Lưu Văn H	AT10A	1998-09-20	0984562579	Hà Nội



Các toán tử so sánh, logic

1. Toán tử so sánh

Toán tử	Ý nghĩa
=	Bằng nhau
>	Lớn hơn
<	Bé hơn
>=	Lớn hơn hoặc bằng
<=	Bé hơn hoặc bằng
<>	Khác nhau
!	Phủ định

Ví dụ Liệt kê danh sách sinh viên có điểm toán lớn hơn hoặc bằng 8

```
SELECT * FROM DIEMTHI WHERE DToan >= 8
```

	MaSV	TenSV	DToan	DLy	DHoa
1	1	Hoàng Mai Anh	9	9	9
2	2	Mai Thu Hương	8	9	10
3	4	Đỗ Mạnh Hùng	8	6	7
4	6	Nguyễn Việt Quang	9	10	10

2. Toán tử logic

Các toán tử logic **AND**, **OR** và **NOT**

- ✓ **AND**, **OR** được sử dụng để kết nối điều kiện tìm kiếm trong mệnh **WHERE**;
- ✓ **NOT** phủ định điều kiện tìm kiếm.



Ví dụ về các toán tử so sánh, logic

1. Liệt kê danh sách sinh viên có điểm toán và lý lớn hơn 8

```
SELECT * FROM DIEMTHI WHERE DToan > 8 AND DLy > 8
```

	MaSV	TenSV	DToan	DLy	DHoa
1	1	Hoàng Mai Anh	9	9	9
2	6	Nguyễn Việt Quang	9	10	10

2. Liệt kê danh sách sinh viên có điểm toán hoặc điểm hóa lớn hơn 9

```
SELECT * FROM DIEMTHI WHERE DToan > 9 OR DHoa > 9
```

	MaSV	TenSV	DToan	DLy	DHoa
1	2	Mai Thu Hương	8	9	10
2	6	Nguyễn Việt Quang	9	10	10

3. Liệt kê danh sách sinh viên có điểm toán bé hơn 8

```
SELECT * FROM DIEMTHI WHERE NOT DToan >= 8
```

	MaSV	TenSV	DToan	DLy	DHoa
1	3	Trần Thu Thủy	7	9	7
2	5	Lê Duy Minh	7	9	8



Mệnh đề ORDER BY

- **Ý nghĩa** Mệnh đề **ORDER BY** trong SQL được sử dụng để sắp xếp dữ liệu theo thứ tự tăng dần (**ASC**) hoặc theo thứ tự giảm dần (**DESC**), trên một hoặc nhiều cột. Một số CSDL sắp xếp kết quả truy vấn theo thứ tự tăng dần theo mặc định.
- **Cú pháp** **SELECT** <tencot>
FROM <tenbang>
WHERE <dieukien>
ORDER BY <tencot> [**ASC** | **DESC**]
- **Ví dụ** Sắp xếp danh sách sinh viên theo thứ tự tăng dần, giảm dần của điểm toán

```
SELECT MaSV, TenSV, DToan  
FROM DIEMTHI ORDER BY DToan
```

	MaSV	TenSV	DToan
1	3	Trần Thu Thủy	7
2	5	Lê Duy Minh	7
3	4	Đỗ Mạnh Hùng	8
4	2	Mai Thu Hương	8
5	1	Hoàng Mai Anh	9
6	6	Nguyễn Việt Quang	9

```
SELECT MaSV, TenSV, DToan FROM  
DIEMTHI ORDER BY DToan DESC
```

	MaSV	TenSV	DToan
1	1	Hoàng Mai Anh	9
2	6	Nguyễn Việt Quang	9
3	2	Mai Thu Hương	8
4	4	Đỗ Mạnh Hùng	8
5	5	Lê Duy Minh	7
6	3	Trần Thu Thủy	7



Mệnh đề GROUP BY và HAVING

➤ **Ý nghĩa** Mệnh đề **GROUP BY** được sử dụng kết hợp với lệnh SELECT để sắp xếp dữ liệu đồng nhất vào trong các nhóm

➤ **Cú pháp**

```
SELECT <tencot>  
FROM <tenbang>  
WHERE <dieukien>  
GROUP BY <tencot>
```

Ví dụ Đếm tổng sinh viên của mỗi tỉnh thành

```
SELECT quequan 'Quê quán', count(masv) 'Số sinh viên'  
FROM Sinhvien  
GROUP BY quequan
```

	MaSV	HotenSV	GioiTinh	NgaySinh	QueQuan
1	AT123	Nguyễn Mai Anh	Nữ	1998-11-12	Thanh Hóa
2	AT124	Hoàng Văn Bình	Nam	1998-12-08	Thái Bình
3	AT125	Trần Duy Cường	Nam	1998-10-10	Hà Nội
4	AT126	Trần Duy Dũng	Nam	1998-10-10	Hà Nội
5	AT127	Hoàng Mạnh Đạt	Nam	1998-10-10	Đà Nẵng
6	AT128	Vũ Thu Hà	Nữ	1998-10-10	Hà Nội

➤ **Kết quả**

	Quê quán	Số sinh viên
1	Đà Nẵng	1
2	Hà Nội	3
3	Thái Bình	1
4	Thanh Hóa	1



Mệnh đề GROUP BY và HAVING

- **Ý nghĩa** Mệnh đề **HAVING** được thêm vào SQL vì mệnh đề WHERE không áp dụng được đối với các hàm tập hợp (như SUM). Nếu không có HAVING, ta không thể nào kiểm tra được điều kiện với các hàm tập hợp. Nên ta có thể hiểu như HAVING dùng để thay thế WHERE khi sử dụng các hàm tập hợp.

- **Cú pháp**

```
SELECT <tencot>, SUM<tencot>  
FROM <tenbang>  
GROUP BY <tencot>  
HAVING SUM<tencot> dieukien giatri
```

- **Ví dụ** Đưa ra danh sách các tỉnh thành có số sinh viên >2

```
SELECT quequan 'Quê quán', count(masv) 'Số sinh viên'  
FROM sinhvien  
GROUP BY quequan  
HAVING count(masv) >2
```

- **Kết quả**

Results			Messages	
	Quê quán	Số sinh viên		
1	Hà Nội	3		



So sánh mệnh đề WHERE và HAVING

➤ *Giống nhau*

WHERE và **HAVING** đều là câu lệnh dùng để lọc ra các bản ghi dựa trên một hoặc nhiều điều kiện

➤ *Khác nhau*

- **WHERE** là câu lệnh dùng để đặt điều kiện lọc trên từng bộ (từng dòng)
- **HAVING** cũng là câu lệnh đặt điều kiện nhưng là ở trên 1 nhóm xác định, thường đi kèm với câu lệnh GROUP BY.

➤ *Cú pháp tổng quát*

SELECT < danh sách các cột >

FROM < danh sách các bảng >

WHERE < điều kiện >

GROUP BY < danh sách các cột gom nhóm >

HAVING < điều kiện trên nhóm >

ORDER BY < tên cột > [**ASC** | **DESC**]



Bài tập thực hành

1. Tạo một cơ sở dữ liệu
2. Tạo hai bảng trong cơ sở dữ liệu, nhập dữ liệu, xóa, thêm cột trong bảng
3. Thực hiện các lệnh truy xuất dữ liệu đã học với cơ sở dữ liệu vừa lập.