



TÀI LIỆU HƯỚNG DẪN THỰC HÀNH MÔN HỌC IT004 - CƠ SỞ DỮ LIỆU (DATABASE)

NỘI DUNG THỰC HÀNH TUẦN 2

Hướng dẫn thực hành

Lê Võ Đình Kha - khalvd@uit.edu.vn

GIỚI THIỆU NỘI DUNG THỰC HÀNH TUẦN 2



NỘI DUNG

1. Toán tử (Operator) và Hàm (Functions) trong SQL.
2. Mệnh đề Select trong SQL.
3. Các phép kết trong SQL (Joins).
4. Các phép toán tập hợp (Set Operations).
5. Truy vấn lồng (Subqueries).
6. Bài tập thực hành và hỏi đáp.

1

TOÁN TỬ VÀ HÀM TRONG SQL

TOÁN TỬ VÀ HÀM TRONG SQL

■ Tổng quan về toán tử trong SQL

- Toán tử (Operator) trong SQL là các ký hiệu hoặc từ khóa được sử dụng để thực hiện các phép toán trên các giá trị và biểu thức trong câu truy vấn. Các toán tử giúp kết hợp, so sánh hoặc thao tác dữ liệu trong các cột của bảng.
- **Các loại toán tử:**
 - Toán tử số học (Arithmetic Operators)
 - Toán tử so sánh (Comparison Operators)
 - Toán tử logic (Logical Operators)
 - Toán tử tập hợp (Set Operators)
 - Toán tử đặc biệt (Special Operators)

TOÁN TỬ VÀ HÀM TRONG SQL

Toán tử số học (Arithmetic Operators)

- Các toán tử số học được sử dụng để thực hiện các phép toán cơ bản như cộng, trừ, nhân, chia.

Toán tử	Mô tả
+	Cộng hai giá trị
-	Trừ giá trị
*	Nhân hai giá trị
/	Chia giá trị
%	Lấy phần dư (Modulo)

TOÁN TỬ VÀ HÀM TRONG SQL

Toán tử so sánh (Comparison Operators)

- Toán tử so sánh được sử dụng để so sánh hai giá trị hoặc biểu thức, trả về kết quả là TRUE hoặc FALSE.

Toán tử	Mô tả
=	So sánh bằng
!= or <>	So sánh khác
>	Lớn hơn
<	Nhỏ hơn
>=	Lớn hơn hoặc bằng
<=	Nhỏ hơn hoặc bằng

TOÁN TỬ VÀ HÀM TRONG SQL

Toán tử logic (Logical Operators)

- Được sử dụng để kết hợp nhiều điều kiện trong mệnh đề WHERE hoặc HAVING.

Toán tử	Mô tả	Ví dụ
AND	Kết hợp hai hoặc nhiều điều kiện, tất cả phải đúng	<pre>SELECT * FROM NhanVien WHERE Luong > 5000000 AND MaPB = 'PB01';</pre>
OR	Kết hợp hai hoặc nhiều điều kiện, chỉ cần một điều kiện đúng	<pre>SELECT * FROM NhanVien WHERE Luong > 5000000 OR MaPB = 'PB01';</pre>
NOT	Đảo ngược kết quả điều kiện, trả về ngược lại	<pre>SELECT * FROM NhanVien WHERE NOT Luong = 5000000;</pre>

TOÁN TỬ VÀ HÀM TRONG SQL

Toán tử tập hợp (Set Operators)

- Toán tử tập hợp được sử dụng để kết hợp kết quả của hai hoặc nhiều truy vấn.

Toán tử	Mô tả
UNION	Kết hợp các kết quả truy vấn, loại bỏ các giá trị trùng lặp
INTERSECT	Trả về các kết quả chung giữa hai truy vấn
EXCEPT	Trả về các kết quả có trong truy vấn thứ nhất nhưng không có trong truy vấn thứ hai

TOÁN TỬ VÀ HÀM TRONG SQL

Toán tử đặc biệt (Special Operators)

Toán tử	Mô tả	Ví dụ
IN	Kiểm tra nếu giá trị thuộc một danh sách hoặc tập kết quả con.	<pre>SELECT HoTen FROM NhanVien WHERE MaPB IN ('PB01', 'PB02');</pre>
BETWEEN	Kiểm tra nếu giá trị nằm trong một khoảng giá trị.	<pre>SELECT HoTen, Luong FROM NhanVien WHERE Luong BETWEEN 5000000 AND 10000000;</pre>
LIKE	Tìm kiếm chuỗi theo mẫu (pattern). Thường sử dụng % để đại diện cho nhiều ký tự.	<pre>SELECT HoTen FROM NhanVien WHERE HoTen LIKE 'Nguyen%';</pre>

TOÁN TỬ VÀ HÀM TRONG SQL

Toán tử đặc biệt (Special Operators)

Toán tử	Mô tả	Ví dụ
IS NULL	Kiểm tra xem giá trị có phải là NULL.	<pre>SELECT HoTen FROM NhanVien WHERE DiaChi IS NULL;</pre>
EXISTS	Kiểm tra sự tồn tại của một tập hợp con. Trả về TRUE nếu tập hợp con có ít nhất một giá trị.	<pre>SELECT * FROM PhongBan WHERE EXISTS (SELECT * FROM NhanVien WHERE NhanVien.MaPB = PhongBan.MaPB);</pre>

TOÁN TỬ TRONG SQL

Toán tử đặc biệt (Special Operators) (tt)

Toán tử	Mô tả	Ví dụ
ANY	So sánh với bất kỳ giá trị nào trong tập kết quả con.	<pre>SELECT HoTen FROM NhanVien WHERE Luong >ANY (SELECT Luong FROM NhanVien WHERE MaPB = 'PB01');</pre>
ALL	So sánh với tất cả các giá trị trong tập kết quả con.	<pre>SELECT HoTen FROM NhanVien WHERE Luong >ALL (SELECT Luong FROM NhanVien WHERE MaPB = 'PB01');</pre>

TOÁN TỬ VÀ HÀM TRONG SQL

Tổng quan về hàm trong SQL

- Hàm trong SQL là các công cụ mạnh mẽ giúp thao tác và xử lý dữ liệu một cách hiệu quả. Các hàm có thể được sử dụng để thực hiện các phép tính toán học, thao tác trên chuỗi, xử lý ngày giờ, cũng như tính toán trên các nhóm dữ liệu.
- **Các loại hàm:**
 - Hàm xử lý số (Numeric Functions)
 - Hàm xử lý chuỗi (String Functions)
 - Hàm xử lý ngày giờ (Date Functions)
 - Hàm tập hợp (Aggregate Functions)

TOÁN TỬ VÀ HÀM TRONG SQL

Hàm xử lý số (Numeric Functions)

Hàm	Mô tả	Ví dụ
ABS(x)	Trả về giá trị tuyệt đối của số x.	ABS(-5); -- Trả về 5
ROUND(x, d)	Làm tròn số x với d số thập phân.	ROUND(3.14159, 2); -- Trả về 3.14
CEIL(x)	Làm tròn lên đến số nguyên gần nhất lớn hơn hoặc bằng x.	CEIL(2.3); -- Trả về 3
FLOOR(x)	Làm tròn xuống đến số nguyên gần nhất nhỏ hơn hoặc bằng x.	FLOOR(2.9); -- Trả về 2
MOD(x, y)	Trả về phần dư khi chia x cho y.	MOD(10, 3); -- Trả về 1
SQRT(x)	Trả về căn bậc hai của số x.	SQRT(16); -- Trả về 4
POWER(x, y)	Trả về giá trị x mũ y.	POWER(2, 3); -- Trả về 8
LOG(x)	Trả về logarit tự nhiên (cơ số e) của x.	LOG(2); -- Trả về 0.693147

TOÁN TỬ VÀ HÀM TRONG SQL

Hàm xử lý chuỗi (String Functions)

Hàm	Mô tả
CONCAT()	Hàm nối chuỗi, tương tự toán tử `
LENGTH()	Trả về độ dài chuỗi (số ký tự trong chuỗi).
SUBSTRING()	Trích xuất một phần của chuỗi từ vị trí bắt đầu đến vị trí kết thúc.
TRIM()	Loại bỏ các khoảng trắng hoặc ký tự cụ thể ở đầu và cuối chuỗi.
UPPER()	Chuyển đổi toàn bộ chuỗi thành chữ hoa.
LOWER()	Chuyển đổi toàn bộ chuỗi thành chữ thường.
POSITION()	Tìm vị trí xuất hiện đầu tiên của một chuỗi con trong chuỗi chính.
REPLACE()	Thay thế tất cả các lần xuất hiện của chuỗi con bằng chuỗi mới.

TOÁN TỬ VÀ HÀM TRONG SQL

Hàm xử lý ngày giờ (Date Functions)

Hàm	Mô tả	Ví dụ
NOW()	Trả về ngày và giờ hiện tại của hệ thống.	NOW(); -- Trả về '2024-09-29 08:23:45'
CURDATE()	Trả về ngày hiện tại.	CURDATE(); -- Trả về '2024-09-29'
YEAR(date)	Trả về năm của giá trị ngày date.	YEAR('2024-09-29'); -- Trả về 2024
MONTH(date)	Trả về tháng của giá trị ngày date.	MONTH('2024-09-29'); -- Trả về 9
DAY(date)	Trả về ngày của giá trị ngày date.	DAY('2024-09-29'); -- Trả về 29
DATE_ADD(date, INTERVAL n unit)	Cộng thêm n đơn vị thời gian vào giá trị ngày date.	DATE_ADD('2024-09-29', INTERVAL 5 DAY); -- Trả về '2024-10-04'
DATE_SUB(date, INTERVAL n unit)	Trừ đi n đơn vị thời gian từ giá trị ngày date.	DATE_SUB('2024-09-29', INTERVAL 5 DAY); -- Trả về '2024-09-24'
DATEDIFF(date1, date2)	Trả về số ngày giữa hai giá trị ngày.	DATEDIFF('2024-09-29', '2023-09-29'); -- Trả về 365

TOÁN TỬ VÀ HÀM TRONG SQL

Hàm tập hợp (Aggregate Functions)

Hàm	Mô tả	Ví dụ
COUNT(*)	Đếm tổng số hàng trong tập dữ liệu (kể cả giá trị NULL).	SELECT COUNT(*) FROM NhanVien;
SUM(x)	Tính tổng giá trị của cột x.	SELECT SUM(Luong) FROM NhanVien;
AVG(x)	Tính giá trị trung bình của cột x.	SELECT AVG(Luong) FROM NhanVien;
MAX(x)	Trả về giá trị lớn nhất của cột x.	SELECT MAX(Luong) FROM NhanVien;
MIN(x)	Trả về giá trị nhỏ nhất của cột x.	SELECT MIN(Luong) FROM NhanVien;
GROUP_CONCAT(x)	Trả về chuỗi chứa các giá trị của cột x được nối với nhau bằng dấu phẩy.	SELECT GROUP_CONCAT(HoTen) FROM NhanVien;

2

MỆNH ĐỀ SELECT TRONG SQL

MỆNH ĐỀ SELECT TRONG SQL

Tổng quan về mệnh đề SELECT trong SQL

- Mệnh đề SELECT là một trong những mệnh đề quan trọng nhất trong SQL, dùng để truy xuất dữ liệu từ một hoặc nhiều bảng.
- Cú pháp tổng quát:

```
SELECT (DISTINCT) <Tên cột 1>, <Tên cột 2>, ...  
FROM <Tên bảng>  
(WHERE <Điều kiện>)  
(GROUP BY <Tên cột 1>, <Tên cột 2>, ... )  
(HAVING <Điều kiện trên nhóm>)  
(ORDER BY <Tên cột 1> (ASC | DESC), <Tên cột 2> (ASC | DESC), ...);
```

MỆNH ĐỀ SELECT TRONG SQL

Các thành phần chính trong mệnh đề SELECT

Thành phần	Mô tả	Ví dụ
SELECT	Xác định các cột hoặc biểu thức cần truy xuất từ bảng dữ liệu.	SELECT MaNV, HoTen FROM NhanVien;
DISTINCT	Loại bỏ các dữ liệu trùng lặp trong kết quả truy vấn.	SELECT DISTINCT NoiSinh FROM KhachHang;
<Tên cột>	Danh sách các cột cần truy xuất từ bảng hoặc kết quả tính toán.	SELECT MaNV, HoTen FROM NhanVien;
FROM	Xác định bảng hoặc bảng liên quan trong truy vấn.	SELECT * FROM SanPham;
WHERE	Được sử dụng để lọc các dòng dữ liệu dựa trên điều kiện cụ thể. Điều kiện có thể kết hợp với các toán tử so sánh (=, >, <, ...) và các toán tử logic (AND, OR, NOT).	SELECT * FROM NhanVien WHERE Tuoi > 30;

MỆNH ĐỀ SELECT TRONG SQL

Các thành phần chính trong mệnh đề SELECT (tt)

Thành phần	Mô tả	Ví dụ
GROUP BY	Được sử dụng khi muốn nhóm các bản ghi lại và tính toán các giá trị tổng hợp như MIN, MAX, SUM, AVG, COUNT.	SELECT MaPB, COUNT(*) FROM NhanVien GROUP BY MaPB;
HAVING	Đặt điều kiện lọc cho các nhóm sau khi sử dụng GROUP BY .	SELECT MaPB, COUNT(*) FROM NhanVien GROUP BY MaPB HAVING COUNT(*) > 5;
ORDER BY	Sắp xếp các kết quả theo một hoặc nhiều cột theo thứ tự tăng (ASC) hoặc giảm (DESC) .	SELECT HoTen, Luong FROM NhanVien ORDER BY Luong DESC;

MỆNH ĐỀ SELECT TRONG SQL

Ví dụ về mệnh đề SELECT trong SQL

- **Ví dụ 1:** Cho biết họ tên, ngày vào làm của nhân viên có số điện thoại là 0342565758.
 - **Quan hệ:** NhanVien
 - **Thuộc tính:** HoTen, NgVL
 - **Điều kiện:** SoDT = '0342565857'
- **Cách viết:**

SELECT HoTen, NgVL

FROM NhanVien

WHERE SoDT = '0342565758';

MỆNH ĐỀ SELECT TRONG SQL

Ví dụ về mệnh đề SELECT trong SQL

- **Ví dụ 2:** Cho biết số lượng nhân viên trong mỗi phòng ban và hiển thị mã phòng ban cùng số lượng.

- **Quan hệ:** NhanVien
- **Thuộc tính:** MaPB, Count(*) AS SoLuong (Số lượng nhân viên)

Sử dụng **AS**: Đặt bí danh

- **Điều kiện:** Group by MaPB
- **Cách viết:**

SELECT MaPB, Count(*) AS SoLuong

FROM NhanVien

GROUP BY MaPB;

MỆNH ĐỀ SELECT TRONG SQL

Ví dụ về mệnh đề SELECT trong SQL

- **Ví dụ 3:** Cho biết họ tên và ngày vào làm của nhân viên mới nhất.
 - **Quan hệ:** NhanVien
 - **Thuộc tính:** HoTen, NgVL
 - **Điều kiện:** Nhân viên mới nhất (Order by NgVL DESC)
- **Cách viết:**

```
SELECT Top 1 HoTen, NgVL  
FROM NhanVien  
ORDER BY NgVL DESC;
```

MỆNH ĐỀ SELECT TRONG SQL

Ví dụ về mệnh đề SELECT trong SQL

- **Ví dụ 4:** Lấy danh sách họ tên và số điện thoại của nhân viên có lương lớn hơn 10 triệu và số điện thoại bắt đầu bằng 034.
 - **Quan hệ:** NhanVien
 - **Thuộc tính:** HoTen, SoDT
 - **Điều kiện:** LUONG > 10000000 và SODT LIKE '034%'
- **Cách viết:**

SELECT HoTen, SoDT

FROM NhanVien

WHERE Luong > 10000000 **AND** SODT **LIKE** '034%';

MỆNH ĐỀ SELECT TRONG SQL

Ví dụ về mệnh đề SELECT trong SQL

- **Ví dụ 5:** Thống kê số lượng nhân viên theo từng mức lương, chỉ hiển thị mức lương có ít nhất 3 nhân viên.
 - **Quan hệ:** NhanVien
 - **Thuộc tính:** Luong, Count(*) AS SoLuong (Số lượng nhân viên)
 - **Điều kiện:** GROUP BY Luong và HAVING COUNT(*) >= 3;
- **Cách viết:**

```
SELECT Luong, Count(*) AS SoLuong  
FROM NhanVien  
GROUP BY Luong  
HAVING Count(*) >= 3;
```

3

CÁC PHÉP KẾT TRONG SQL

CÁC PHÉP KẾT TRONG SQL

■ Tổng quan về phép kết (joins) trong SQL

- Trong SQL, phép kết (joins) là kỹ thuật được sử dụng để **kết hợp** các dòng dữ liệu từ hai hoặc nhiều bảng trong cơ sở dữ liệu dựa trên một điều kiện chung.
- **Có nhiều loại phép kết trong SQL:**
 - INNER JOIN.
 - LEFT JOIN.
 - RIGHT JOIN.
 - FULL OUTER JOIN.
 - CROSS JOIN.
 - SELF JOIN.
 - NATURAL JOIN.
 - EQUI JOIN.

CÁC PHÉP KẾT TRONG SQL

Các loại phép kết (joins) trong SQL

Phép kết	Mô tả	Ví dụ	Kết quả
INNER JOIN	Chỉ trả về các dòng dữ liệu có giá trị khớp ở cả hai bảng.	<pre>SELECT nv.HoTen, pb.TenPB FROM NhanVien nv INNER JOIN PhongBan pb ON nv.MaPB = pb.MaPB;</pre>	Chỉ những bản ghi có mã phòng ban (MaPB) khớp ở cả hai bảng sẽ xuất hiện.
LEFT JOIN (LEFT OUTER JOIN)	Trả về tất cả các dòng dữ liệu từ bảng bên trái, và các dòng dữ liệu khớp từ bảng bên phải.	<pre>SELECT nv.HoTen, pb.TenPB FROM NhanVien nv LEFT JOIN PhongBan pb ON nv.MaPB = pb.MaPB;</pre>	Tất cả nhân viên (bảng bên trái) sẽ được hiển thị, kể cả khi không có phòng ban khớp, các cột của bảng PhongBan sẽ là NULL nếu không có dòng dữ liệu khớp.

CÁC PHÉP KẾT TRONG SQL

Các loại phép kết (joins) trong SQL (tt)

Phép kết	Mô tả	Ví dụ	Kết quả
RIGHT JOIN (RIGHT OUTER JOIN)	Trả về tất cả các bản ghi từ bảng bên phải, và các dòng dữ liệu khớp từ bảng bên trái.	<pre>SELECT nv.HoTen, pb.TenPB FROM NhanVien nv RIGHT JOIN PhongBan pb ON nv.MaPB = pb.MaPB;</pre>	Tất cả các phòng ban (bảng bên phải) sẽ xuất hiện, kể cả những phòng ban không có nhân viên, các cột của bảng NhanVien sẽ là NULL nếu không có dòng dữ liệu khớp.
FULL OUTER JOIN	Trả về tất cả các bản ghi từ cả hai bảng, dù có khớp hay không.	<pre>SELECT nv.HoTen, pb.TenPB FROM NhanVien nv FULL OUTER JOIN PhongBan pb ON nv.MaPB = pb.MaPB;</pre>	Tất cả các dòng dữ liệu từ cả hai bảng sẽ được hiển thị, với giá trị NULL ở bảng không có dòng dữ liệu khớp.

CÁC PHÉP KẾT TRONG SQL

Các loại phép kết (joins) trong SQL (tt)

Phép kết	Mô tả	Ví dụ	Kết quả
CROSS JOIN	Tạo ra tích Cartesian, kết hợp tất cả các hàng từ hai bảng.	SELECT nv.HoTen, pb.TenPB FROM NhanVien nv RIGHT JOIN PhongBan pb ON nv.MaPB = pb.MaPB;	Tất cả các phòng ban (bảng bên phải) sẽ xuất hiện, kể cả những phòng ban không có nhân viên, các cột của bảng NhanVien sẽ là NULL nếu không có dòng dữ liệu khớp.
SELF JOIN	Kết hợp một bảng với chính nó, thường dùng để tìm quan hệ giữa các hàng.	SELECT nv.HoTen AS NhanVien, ql.HoTen AS QuanLy FROM NhanVien nv JOIN NhanVien ql ON nv.MaNV = ql.MaNV;	Hiển thị các nhân viên cùng với quản lý của họ (dùng bảng NHANVIEN hai lần với các bí danh khác nhau).

CÁC PHÉP KẾT TRONG SQL

Các loại phép kết (joins) trong SQL (tt)

Phép kết	Mô tả	Ví dụ	Kết quả
NATURAL JOIN	Tự động kết hợp các cột có cùng tên từ hai bảng.	SELECT HoTen, TenPB FROM NhanVien NATURAL JOIN PhongBan;	Tất cả các phòng ban (bảng bên phải) sẽ xuất hiện, kể cả những phòng ban không có nhân viên, các cột của bảng NhanVien sẽ là NULL nếu không có dòng dữ liệu khớp.
EQUI JOIN	So sánh các cột trong hai bảng bằng toán tử bằng (=).	SELECT nv.HoTen, pb.TenPB FROM NhanVien nv JOIN PhongBan pb ON nv.MaPB = pb.MaPB;	Trả về các bản ghi khi điều kiện bằng (=) giữa các cột trong hai bảng được thoả mãn. Đây là phép kết cơ bản nhất của INNER JOIN.

CÁC PHÉP KẾT TRONG SQL

Các loại phép kết (joins) trong SQL (tt)

Phép kết	Mô tả	Ví dụ	Kết quả
NATURAL JOIN	Tự động kết hợp các cột có cùng tên từ hai bảng.	SELECT HoTen, TenPB FROM NhanVien NATURAL JOIN PhongBan;	Tất cả các phòng ban (bảng bên phải) sẽ xuất hiện, kể cả những phòng ban không có nhân viên, các cột của bảng NhanVien sẽ là NULL nếu không có dòng dữ liệu khớp.
EQUI JOIN	So sánh các cột trong hai bảng bằng toán tử bằng (=).	SELECT nv.HoTen, pb.TenPB FROM NhanVien nv JOIN PhongBan pb ON nv.MaPB = pb.MaPB;	Trả về các bản ghi khi điều kiện bằng (=) giữa các cột trong hai bảng được thoả mãn. Đây là phép kết cơ bản nhất của INNER JOIN.

CÁC PHÉP KẾT TRONG SQL

■ Cách viết phép kết bằng (EQUI JOIN) để kết hợp hai hay nhiều bảng trong SQL

- Cách 1: Sử dụng INNER JOIN và điều kiện trong mệnh đề ON

- Sử dụng cú pháp INNER JOIN và đặt điều kiện kết hợp giữa các bảng trong mệnh đề ON. Đây là cách viết chuẩn khi sử dụng phép kết trong SQL.

- Cú pháp:

```
SELECT <Tên cột 1>, <Tên cột 2>, ...
```

```
FROM <Tên bảng 1> INNER JOIN <Tên bảng 2> ON <Điều kiện kết>;
```

- Ví dụ: Kết bảng NhanVien và PhongBan dựa trên cột MaPB

```
SELECT nv.HoTen, pb.TenPB
```

```
FROM NhanVien nv INNER JOIN PhongBan pb ON nv.MaPB = pb.MaPB;
```

CÁC PHÉP KẾT TRONG SQL

— Cách viết phép kết bằng (EQUI JOIN) để kết hợp hai hay nhiều bảng trong SQL

- Cách 2: Đưa điều kiện kết hợp xuống mệnh đề WHERE

- Cách này đưa điều kiện kết hợp các bảng vào mệnh đề WHERE. Mặc dù cách này có thể hoạt động, nhưng không được khuyến khích để dùng.

- Cú pháp:

```
SELECT <Tên cột 1>, <Tên cột 2>, ...  
FROM <Tên bảng 1>, <Tên bảng 2>  
WHERE <Điều kiện kết>;
```

- Ví dụ: Kết bảng NhanVien và PhongBan dựa trên cột MaPB

```
SELECT nv.HoTen, pb.TenPB  
FROM NhanVien nv, PhongBan pb  
WHERE nv.MaPB = pb.MaPB;
```

CÁC PHÉP KẾT TRONG SQL

Cách viết phép kết bằng (EQUI JOIN) để kết hợp hai hay nhiều bảng trong SQL

- **Ví dụ:** Lấy danh sách nhân viên và phòng ban, đồng thời chỉ lấy những nhân viên có ngày vào làm sau năm 2020.

- **Cách 1:**

```
SELECT nv.HoTen, pb.TenPB
```

```
FROM NhanVien nv INNER JOIN PhongBan pb ON nv.MaPB = pb.MaPB
```

```
WHERE nv.NgVL > '2020-01-01';
```

- **Cách 2:**

```
SELECT nv.HoTen, pb.TenPB
```

```
FROM NhanVien nv, PhongBan pb
```

```
WHERE nv.MaPB = pb.MaPB AND nv.NgVL > '2020-01-01';
```

4

CÁC PHÉP TOÁN TẬP HỢP TRONG SQL

CÁC PHÉP TOÁN TẬP HỢP TRONG SQL

Giới thiệu các phép toán tập hợp (Set Operations)

- Phép toán tập hợp (Set Operations) trong SQL là các kỹ thuật giúp kết hợp kết quả từ nhiều câu truy vấn khác nhau.
- **Các phép toán tập hợp:**
 - **UNION:** Kết hợp kết quả từ hai truy vấn và loại bỏ các dòng dữ liệu trùng lặp.
 - **INTERSECT:** Trả về các dòng dữ liệu tồn tại trong cả hai truy vấn.
 - **EXCEPT:** Trả về các dòng dữ liệu chỉ có trong truy vấn đầu tiên nhưng không có trong truy vấn thứ hai.

(Câu truy vấn 1)

<PHÉP TOÁN TẬP HỢP>

(Câu truy vấn 2)

CÁC PHÉP TOÁN TẬP HỢP TRONG SQL

Giới thiệu các phép toán tập hợp (Set Operations)

▪ UNION - Phép hội

Phép *UNION* được sử dụng để kết hợp kết quả của hai hoặc nhiều câu truy vấn. Nó chỉ trả về các hàng duy nhất (không lặp lại) trong kết quả.

- **Cú pháp:**
 (SELECT <Tên cột 1>, <Tên cột 2>, ...
 FROM <Tên bảng 1>
 WHERE <Điều kiện>
 UNION
 (SELECT <Tên cột 1>, <Tên cột 2>, ...
 FROM <Tên bảng 2>
 WHERE <Điều kiện>);

CÁC PHÉP TOÁN TẬP HỢP TRONG SQL

Giới thiệu các phép toán tập hợp (Set Operations)

▪ UNION - Phép hội

- Ví dụ: Liệt kê các mã nhân viên thực hiện đề án DA01 hoặc DA02

```
(SELECT MaNV  
FROM PhanCong  
WHERE MaDA = 'DA01')  
UNION  
(SELECT MaNV  
FROM PhanCong  
WHERE MaDA = 'DA02');
```

CÁC PHÉP TOÁN TẬP HỢP TRONG SQL

Giới thiệu các phép toán tập hợp (Set Operations)

▪ INTERSECT - Phép giao

Phép *INTERSECT* trả về các hàng xuất hiện ở cả hai kết quả của các câu truy vấn con. Nói cách khác, chỉ các hàng có trong cả hai tập hợp sẽ được giữ lại.

- **Cú pháp:**

```
(SELECT <Tên cột 1>, <Tên cột 2>, ...  
FROM <Tên bảng 1>  
WHERE <Điều kiện>  
INTERSECT  
(SELECT <Tên cột 1>, <Tên cột 2>, ...  
FROM <Tên bảng 2>  
WHERE <Điều kiện>);
```


CÁC PHÉP TOÁN TẬP HỢP TRONG SQL

Giới thiệu các phép toán tập hợp (Set Operations)

- **INTERSECT** - Phép giao

- Ví dụ: Liệt kê các mã nhân viên thực hiện đề án DA01 và DA02

```
(SELECT MaNV  
FROM PhanCong  
WHERE MaDA = 'DA01')  
  
INTERSECT  
  
(SELECT MaNV  
FROM PhanCong  
WHERE MaDA = 'DA02');
```

CÁC PHÉP TOÁN TẬP HỢP TRONG SQL

Giới thiệu các phép toán tập hợp (Set Operations)

▪ EXCEPT - Phép trừ

Phép *EXCEPT* trả về các hàng có trong kết quả của câu truy vấn đầu tiên nhưng không có trong kết quả của câu truy vấn thứ hai.

- **Cú pháp:**
(**SELECT** <Tên cột 1>, <Tên cột 2>, ...
FROM <Tên bảng 1>
WHERE <Điều kiện>
EXCEPT
(**SELECT** <Tên cột 1>, <Tên cột 2>, ...
FROM <Tên bảng 2>
WHERE <Điều kiện>);

CÁC PHÉP TOÁN TẬP HỢP TRONG SQL

Giới thiệu các phép toán tập hợp (Set Operations)

▪ EXCEPT - Phép trừ

- Ví dụ 1: Liệt kê các mã nhân viên thực hiện đề án DA01 nhưng không thực hiện đề án DA02

```
(SELECT MaNV  
FROM PhanCong  
WHERE MaDA = 'DA01')  
EXCEPT  
(SELECT MaNV  
FROM PhanCong  
WHERE MaDA = 'DA02');
```

CÁC PHÉP TOÁN TẬP HỢP TRONG SQL

Giới thiệu các phép toán tập hợp (Set Operations)

- **EXCEPT - Phép trừ**

- Ví dụ 2: Cho biết những nhân viên không tham gia đề án nào

```
(SELECT MaNV, HoTen
```

```
FROM NhanVien)
```

```
EXCEPT
```

```
(SELECT nv.MaNV, HoTen
```

```
FROM NhanVien nv, PhanCong pc
```

```
WHERE nv.MaNV = pc.MaNV);
```

5

TRUY VẤN LỒNG

TRUY VẤN LỒNG

Giới thiệu tổng quan về truy vấn lồng (Subqueries) trong SQL

- Truy vấn lồng (hay còn gọi là Subquery) là một kỹ thuật trong SQL cho phép lồng một câu truy vấn bên trong một câu truy vấn khác.
- Truy vấn lồng thường được sử dụng khi cần lấy dữ liệu từ một bảng dựa trên kết quả của một truy vấn khác.

- **Cú pháp:**

SELECT <Tên cột 1>, <Tên cột 2>, ...

FROM <Tên bảng 1>

WHERE <Tên cột> <Toán tử> (**SELECT** <Tên cột 1>, <Tên cột 2>, ...

FROM <Tên bảng khác>

WHERE <Điều kiện>);

TRUY VẤN LỒNG

Giới thiệu tổng quan về truy vấn lồng (Subqueries) trong SQL

- Câu truy vấn lồng cơ bản là dạng truy vấn con không phụ thuộc vào truy vấn cha.
- Câu truy vấn lồng phân cấp có thể chứa nhiều tầng truy vấn con, thường dùng trong các bài toán phức tạp.
- Câu truy vấn lồng tương quan yêu cầu sự tương quan giữa truy vấn chính và truy vấn con, sử dụng khi cần thực hiện phép so sánh giữa các hàng trong cùng một bảng hoặc bảng khác.

- **Cú pháp:**

```
SELECT (DISTINCT) <Danh sách các thuộc tính/hàm>  
FROM <Danh sách các bảng>  
WHERE <Điều kiện> (Câu truy vấn con)
```

TRUY VẤN LỒNG

Các dạng truy vấn lồng

- **Lồng phân cấp (Non-Correlated Subquery):**
 - So sánh tập hợp thường đi kèm: **IN / NOT IN, ALL, ANY**
 - Trước ALL, ANY có toán tử so sánh. Ví dụ: **> ALL, = ANY**
- **Lồng tương quan (Correlated Subquery):**
 - Kiểm tra sự tồn tại: **EXIST, NOT EXIST**

TRUY VẤN LỒNG

Các toán tử sử dụng trong truy vấn lồng trong SQL

Toán tử	Cú pháp	Ý nghĩa
IN	<biểu thức> (NOT) IN (<truy vấn con>)	Kiểm tra nếu giá trị của biểu thức có nằm trong tập hợp kết quả của truy vấn con.
ANY	<biểu thức> <phép toán so sánh> ANY (<truy vấn con>)	So sánh biểu thức với bất kỳ giá trị nào trong tập hợp kết quả của truy vấn con.
ALL	<biểu thức> <phép toán so sánh> ALL (<truy vấn con>)	So sánh biểu thức với tất cả giá trị trong tập hợp kết quả của truy vấn con.
EXISTS	EXISTS (<truy vấn con>)	Trả về TRUE nếu truy vấn con trả về ít nhất một hàng; trả về FALSE nếu không có hàng nào.
NOT EXISTS	NOT EXISTS (<truy vấn con>)	Trả về TRUE nếu truy vấn con không trả về hàng nào; trả về FALSE nếu có ít nhất một hàng.

IN/NOT IN

- Ví dụ 1: Tìm tên các sản phẩm đã được bán trong tháng 1 năm 2024.
- Cách viết:

SELECT TenSP

FROM SanPham

WHERE MaSP IN (**SELECT** MaSP

FROM HoaDon

WHERE month(NgayBan) = 1 AND year(NgayBan) = 2024);

TRUY VẤN LỒNG

ANY

- **Ví dụ 2:** Tìm các nhân viên có lương lớn hơn bất kỳ nhân viên nào làm việc trong phòng ban có MaPB là PB01, PB02 hoặc PB03.
- **Cách viết:**

SELECT HoTen, Luong

FROM NhanVien

WHERE Luong > ANY (**SELECT** Luong

FROM NhanVien

WHERE MaPB IN ('PB01', 'PB02', 'PB03'));

ALL

- **Ví dụ 3:** Tìm các sản phẩm có giá bán cao hơn tất cả các sản phẩm trong danh mục "Laptop" nhưng thuộc danh mục "Phụ kiện".
- **Cách viết:**

```
SELECT MaSP, TenSP, GiaBan
```

```
FROM SanPham
```

```
WHERE DanhMuc = 'Phu Kien'
```

```
AND GiaBan > ALL ( SELECT GiaBan
```

```
FROM SanPham
```

```
WHERE DanhMuc = 'Laptop');
```

EXISTS

- Ví dụ 4: Tìm kiếm tất cả nhân viên có ít nhất một đơn hàng.
- Cách viết:

SELECT HoTen

FROM NhanVien nv

WHERE EXISTS (**SELECT** *

FROM DonHang dh

WHERE dh.MaNV = nv.MaNV);

NOT EXISTS

- Ví dụ 5: Tìm kiếm tất cả nhân viên không có bất kỳ đơn hàng nào.
- Cách viết:

SELECT HoTen

FROM NhanVien nv

WHERE NOT EXISTS (**SELECT** *

FROM DonHang dh

WHERE dh.MaNV = nv.MaNV);

6

BÀI TẬP THỰC HÀNH VÀ HỎI ĐÁP

BÀI TẬP THỰC HÀNH VÀ HỎI ĐÁP

Yêu cầu: (Xem thêm đề bài tại file Bài tập thực hành tuần 2)

Phần bài làm lưu trữ trên một file script có tên **<MSSV>_<HoVaTen>_BTTH2.sql** (trong đó **MSSV** là mã số sinh viên, **HoVaTen** là họ và tên của sinh viên).

Sử dụng các câu lệnh SQL trong công cụ **SQL Server Management Studio** để thực hiện yêu cầu. **Phần III: Ngôn ngữ truy vấn dữ liệu**, bài tập **Quản lý bán hàng** từ câu 1 đến câu 25.



HỎI ĐÁP



Liên hệ hỗ trợ

Lê Võ Đình Kha - khalvd@uit.edu.vn