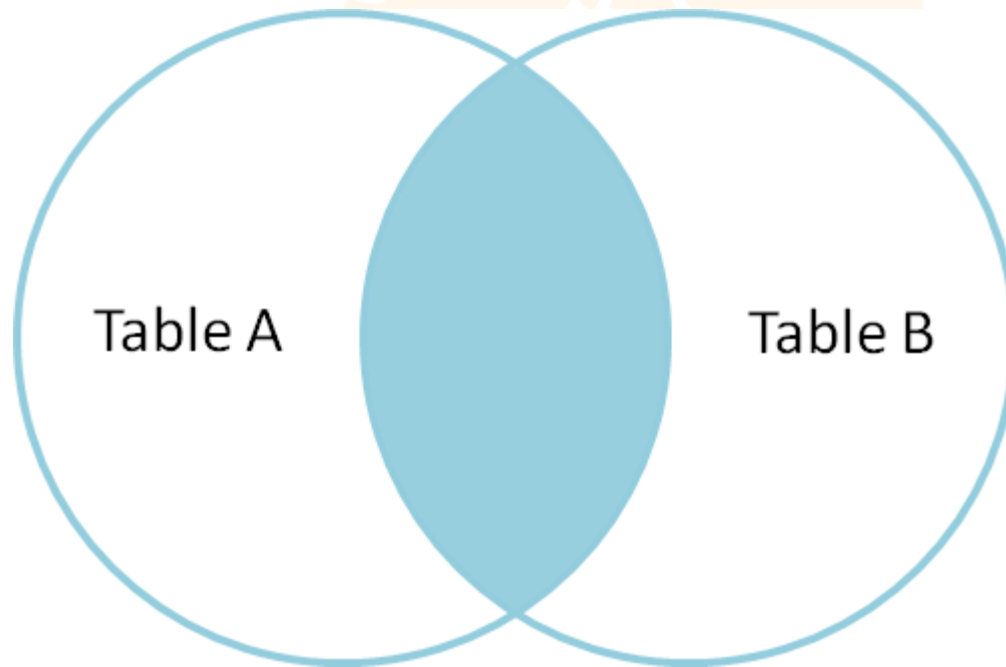


Bài 09

Truy vấn nâng cao và các phép kết nối (JOIN)

Tóm tắt

- Giới thiệu
- Nhóm và thống kê dữ liệu
- Truy vấn con
- Phép ghép nối (join)



Giới thiệu

- SQL Server 2014 đưa vào nhiều tính năng truy vấn mạnh mẽ giúp bạn có thể **lấy dữ liệu nhanh chóng và hiệu quả**.
- Dữ liệu có thể được **nhóm và/hoặc tổng hợp** (aggregated) cùng nhau để thể hiện thông tin tổng kết.
- Sử dụng khái niệm **truy vấn con**, tập kết quả của một câu lệnh SELECT có được sử dụng **làm điều kiện cho câu lệnh SELECT hoặc truy vấn khác**.
- Ghép nối (Joins) giúp bạn **kết hợp các cột dữ liệu từ hai hay nhiều bảng** dựa trên mối quan hệ giữa các bảng.

Nhóm và thống kê

Nhóm dữ liệu:

- Mệnh đề GROUP BY **phân vùng (partitions)** tập kết quả thành một hoặc nhiều tập con. Mỗi tập kết quả có các giá trị và biểu thức chung.
- Từ khóa **GROUP BY**, theo sau nó là danh sách các cột được dùng để nhóm. Mỗi cột được nhóm giới hạn số lượng dòng của tập kết quả.
- **Mỗi cột nhóm, chỉ có một dòng.** Mệnh đề GROUP BY có thể nhiều hơn một cột nhóm.

Nhóm và thống kê

Cú pháp:

```
SELECT <select list>  
FROM <table source>  
WHERE <search condition>  
GROUP BY <group by list>
```

Trong đó:

<group by list>: danh sách các cột được chọn để nhóm các dòng.

Nhóm và thống kê

Ví dụ: khách hàng có thể **mua nhiều sản phẩm**, truy vấn sau tới bảng dữ liệu hóa đơn sẽ **thống kê số sản phẩm** bằng lệnh GROUP BY

Code:

-- Truy vấn lấy về tổng số sản phẩm đã bán được NHÓM theo KHÁCH HÀNG mua

```
SELECT tenkh, COUNT(idsp) AS 'Tổng số sản phẩm mua'
```

```
FROM tblBanHang, tblKhachHang
```

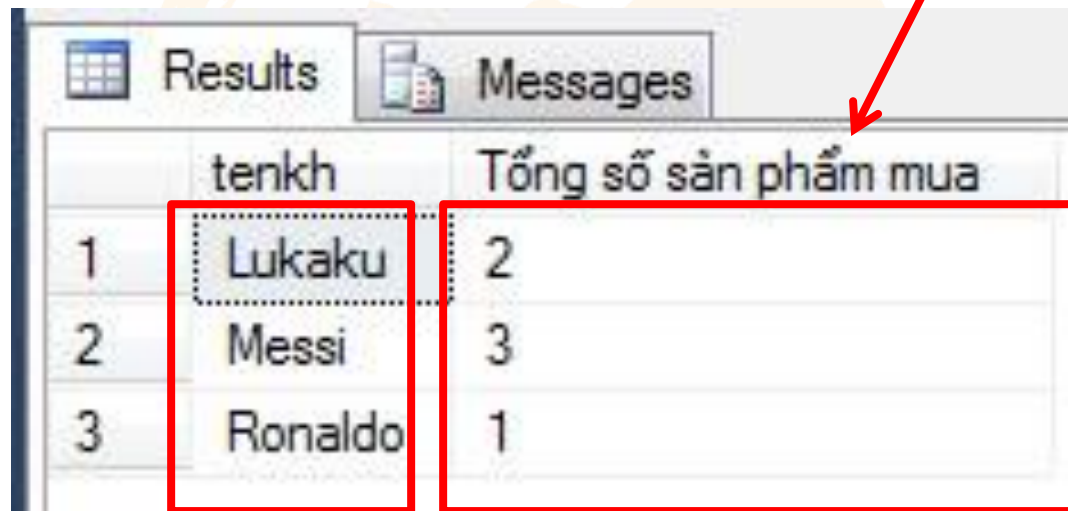
```
WHERE tblBanHang.idkh = tblKhachHang.idkh
```

```
GROUP BY tblKhachHang.tenkh -- nhóm kết quả truy vấn theo tenkh
```

Nhóm và thống kê

Kết quả:

COUNT(idsp) AS 'Tổng số sản phẩm mua'



	tenkh	Tổng số sản phẩm mua
1	Lukaku	2
2	Messi	3
3	Ronaldo	1

GROUP BY tblKhachHang.tenkh

Nhóm và thống kê

GROUP BY với WHERE:

- Có thể sử dụng mệnh đề WHERE cùng với mệnh đề GROUP BY để **giới hạn số dòng** trước khi thực hiện nhóm.
- Các dòng thỏa mãn điều kiện sẽ được xem xét cho việc nhóm.
- Các dòng dữ liệu **không phù hợp** với điều kiện trong mệnh đề WHERE sẽ **bị loại bỏ** trước khi công việc nhóm được thực hiện.

Nhóm và thống kê

Ví dụ: vẫn với yêu cầu trên, giờ bổ sung thêm chỉ lọc giá bán sản phẩm có **giá lớn hơn \$200**.

Code:

-- Truy vấn lấy về tổng số sản phẩm có giá > \$200 đã bán được NHÓM theo KHÁCH HÀNG mua

```
SELECT tenkh, COUNT(idsp) AS 'Tổng số sản phẩm mua'
```

```
FROM tblBanHang, tblKhachHang, tblSanPham
```

```
WHERE
```

```
tblBanHang.idkh = tblKhachHang.idkh
```

```
ANDtblBanHang.idsp = tblSanPham.id
```

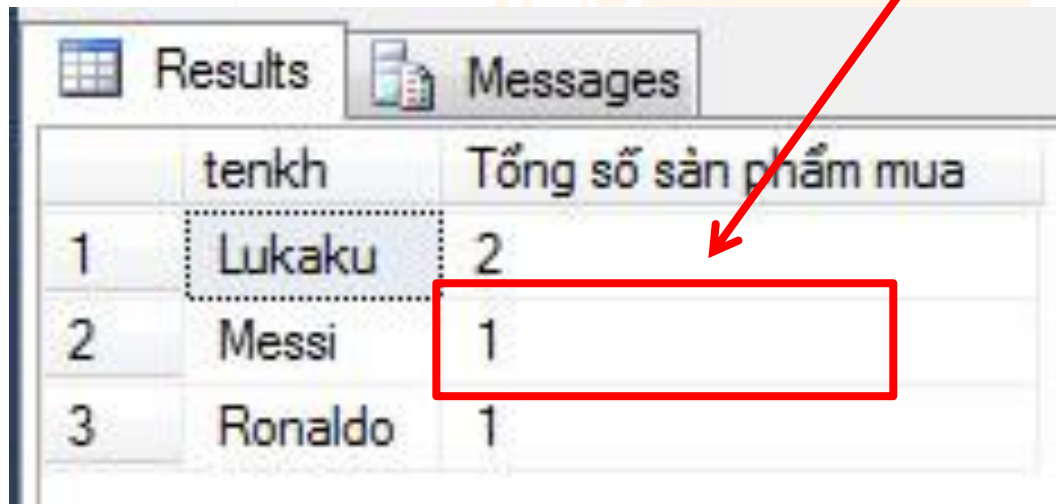
```
AND tblSanPham.giaban > 200
```

```
GROUP BY tblKhachHang.tenkh -- nhóm kết quả truy vấn theo tenkh
```

Nhóm và thống kê

Kết quả:

Khách hàng “Messi” chỉ mua 1 sản phẩm có giá trên \$200



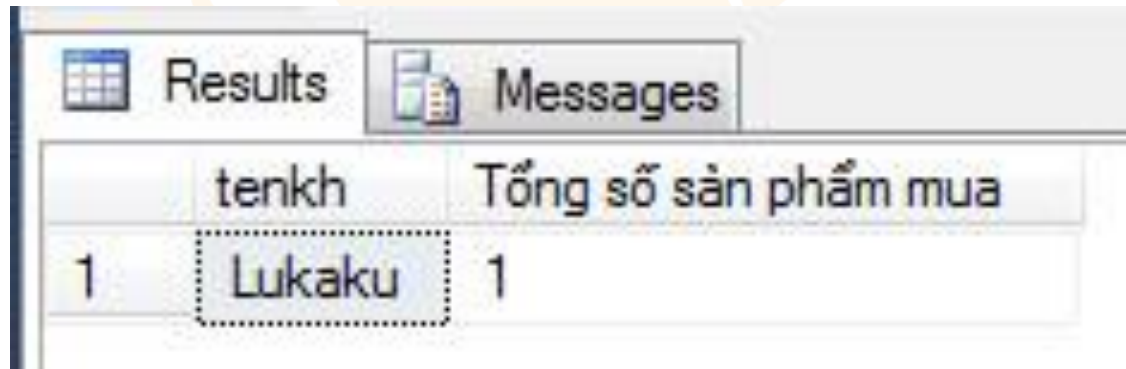
	tenkh	Tổng số sản phẩm mua
1	Lukaku	2
2	Messi	1
3	Ronaldo	1

Nhóm và thống kê

GROUP BY ALL:

- Có một số tình huống, dữ liệu sau khi bị lọc loại bỏ bởi mệnh đề WHERE **vẫn muốn hiển thị** trong tập kết quả trả về.

Ví dụ: nếu chỉ thống kê tổng sản phẩm có giá trên \$250 thì ta sẽ chỉ có 1 khách hàng là "Lukaku"



Results		Messages	
	tenkh	Tổng số sản phẩm mua	
1	Lukaku	1	

Nhóm và thống kê

Còn muốn lấy tất cả:

-- Truy vấn lấy về tổng số sản phẩm có giá > \$250 đã bán được NHÓM theo KHÁCH HÀNG mua, thống kê cả khách hàng không thỏa mãn

```
SELECT tenkh, COUNT(idsp) AS 'Tổng số sản phẩm mua'
```

```
FROM tblBanHang, tblKhachHang, tblSanPham WHERE
```

```
tblBanHang.idkh = tblKhachHang.idkh
```

```
AND tblBanHang.idsp = tblSanPham.id
```

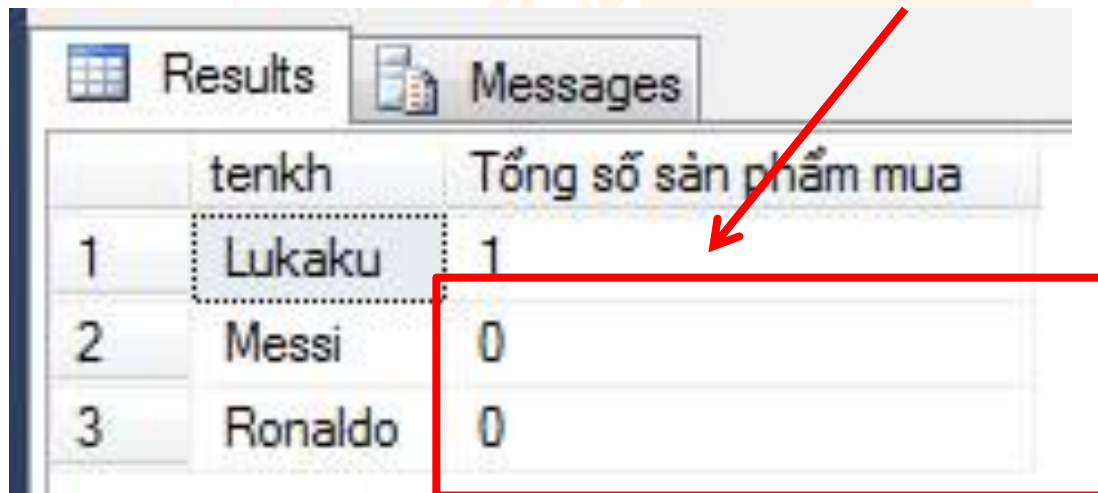
```
AND tblSanPham.giaban > 250
```

```
GROUP BY ALL tblKhachHang.tenkh -- nhóm kết quả truy vấn  
theo tenkh
```

Nhóm và thống kê

Kết quả:

Khách hàng mua sản phẩm không thỏa
mãn điều kiện vẫn thống kê



	tenkh	Tổng số sản phẩm mua
1	Lukaku	1
2	Messi	0
3	Ronaldo	0

Nhóm và thống kê

GROUP BY với HAVING:

- Mệnh đề HAVING chỉ được sử dụng cùng với mệnh đề GROUP BY trong câu lệnh SELECT để **chỉ ra điều kiện lọc dữ liệu sau khi đã tính toán** theo nhóm.
- Mệnh đề HAVING có **vai trò như** mệnh đề WHERE, nhưng nó được dùng ở những nơi mà mệnh đề WHERE không thể sử dụng được với các hàm tổng hợp như SUM ().
- Sau khi bạn đã tạo ra các nhóm bằng mệnh đề GROUP BY, và bạn muốn tiếp tục lọc các nhóm kết quả nữa.
- Mệnh đề HAVING đóng vai trò như **bộ lọc trên các nhóm**, tương tự cách mệnh đề WHERE lọc trên các dòng được trả về bởi mệnh đề FROM.

Nhóm và thống kê

Ví dụ: thống kê tổng số tiền khách hàng đã chi và chỉ lọc lấy khách hàng có **tổng tiền lớn hơn \$500**.

Code:

- Truy vấn lấy về tổng số sản phẩm đã bán được
- NHÓM theo KHÁCH HÀNG mua và tổng số tiền mà khách hàng đã chi
- và lấy về KHÁCH HÀNG VIP có tổng số tiền lớn hơn 500\$

```
SELECT tenkh, COUNT(idsp) AS 'Tổng số sản phẩm mua', SUM(giaban) AS  
'Tổng tiền' FROM tblBanHang, tblKhachHang, tblSanPham
```

```
WHERE
```

```
tblBanHang.idkh = tblKhachHang.idkh AND
```

```
tblBanHang.idsp = tblSanPham.id
```

```
GROUP BY tblKhachHang.tenkh -- nhóm kết quả truy vấn theo tenkh
```

```
HAVING SUM(giaban) > 500
```

Nhóm và thống kê

Kết quả:

	tenkh	Tổng số sản phẩm mua	Tổng tiền
1	Lukaku	2	580
2	Messi	3	470
3	Ronaldo	1	230

Tổng chi phí của
tất cả khách
hàng

	tenkh	Tổng số sản phẩm mua	Tổng tiền
1	Lukaku	2	580

Lọc khách hàng
có tổng hóa đơn
> \$500

Nhóm và thống kê

Hàm thống kê:

- Các hàm thống kê thực hiện tính toán trên **tập giá trị** và **trả về giá trị duy nhất**
- Các hàm thống kê **đều bỏ qua giá trị NULL**, nhưng trừ hàm COUNT(*).
- Các hàm thống kê thường được sử dụng với mệnh đề GROUP BY của câu lệnh SELECT.
- Khi sử dụng hàm thống kê trong phần danh sách cột của câu lệnh SELECT **thường tạo ra một cột không có bí danh (alias)**, do vậy bạn có thể dùng mệnh đề AS cung cấp cho nó một bí danh.

Nhóm và thống kê

Hàm thống kê thường dùng:

Tên hàm	Cú pháp	Mô tả
AVG	AVG(<mệnh đề>)	Tính giá trị trung bình của dữ liệu số trên các dòng của cột chỉ định.
COUNT	COUNT(*)	Đếm tổng số dòng dữ liệu.
MAX	MAX(<mệnh đề>)	Trả về giá trị số lớn nhất hoặc ngày gần nhất.
MIN	MIN(<mệnh đề>)	Trả về giá trị số bé nhất hoặc ngày xa nhất.
SUM	SUM(<mệnh đề>)	Tính tổng giá trị của tất cả các dòng dữ liệu trong cột

Nhóm và thống kê

Hàm AVG():

```
SELECT AVG(gia_nhap) AS 'Giá trung bình' FROM  
SAN_PHAM
```

	ma_sp	ten_sp	mo_ta	thong_tin	gia_nhap
1	1	Conserve Chuck 1	fake 1	NULL	600
2	2	Gucci SNN69	mới nhập về	NULL	120
3	3	Adidas	hàng xịn của TQ	NULL	1450
4	4	Nike MG21	nhập khẩu Malay	NULL	690
5	5	Tom 300	đồ đều	NULL	55
6	6	Lacoste 066	nhanh hóng	NULL	99
1		Giá trung bình			502,333333333333

Nhóm và thống kê

Hàm COUNT():

```
SELECT COUNT(*) AS 'Tổng số sản phẩm' FROM  
SAN_PHAM
```

	ma_sp	ten_sp	mo_ta	thong_tin	gia_nhap
1	1	Conserve Chuck 1	fake 1	NULL	600
2	2	Gucci SNN69	mới nhập về	NULL	120
3	3	Adidas	hàng xịn của TQ	NULL	1450
4	4	Nike MG21	nhập khẩu Malay	NULL	690
5	5	Tom 300	đồ đều	NULL	55
6	6	Lacoste 066	nhanh hòng	NULL	99
	Tổng số sản phẩm				
1	6				

Nhóm và thống kê

Hàm MAX() và MIN():

```
SELECT MAX(gia_nhap) AS 'Giá cao nhất',  
MIN(gia_nhap) AS 'Giá thấp nhất' FROM SAN_PHAM
```

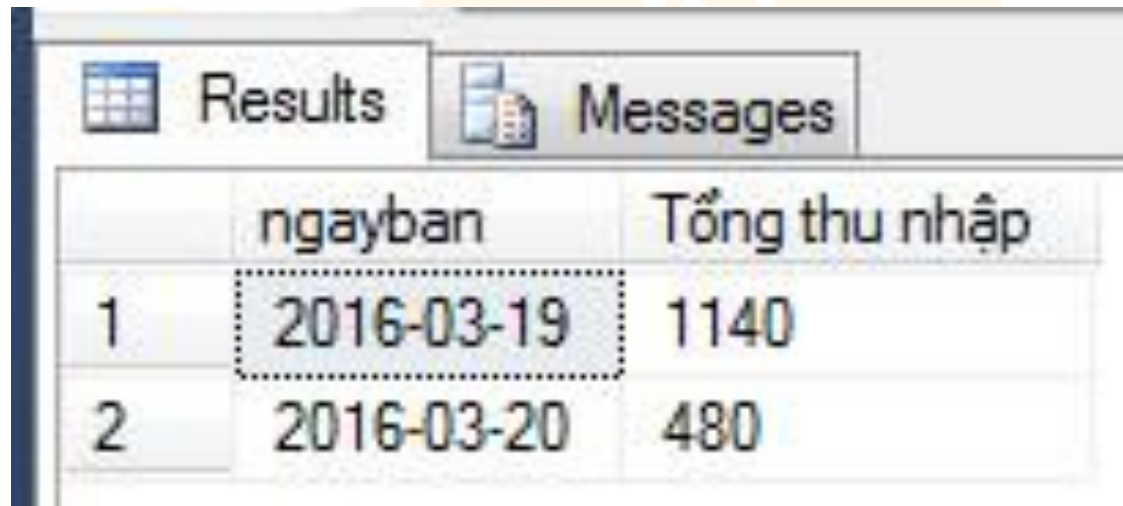
	ma_sp	ten_sp	mo_ta	thong_tin	gia_nhap
1	1	Conserve Chuck 1	fake 1	NULL	600
2	2	Gucci SNN69	mới nhập về	NULL	120
3	3	Adidas	hàng xịn của TQ	NULL	1450
4	4	Nike MG21	nhập khẩu Malay	NULL	690
5	5	Tom 300	đồ đều	NULL	55
6	6	Lacoste 066	nhanh hòng	NULL	99

	Giá cao nhất	Giá thấp nhất
1	1450	55

Nhóm và thống kê

Hàm SUM():

```
SELECT ngayban, SUM(giatien) AS 'Tổng thu  
nhập' FROM ThongKeBanHang GROUP BY ngayban
```



The screenshot shows a SQL query result window with two tabs: 'Results' and 'Messages'. The 'Results' tab is active, displaying a table with two columns: 'ngayban' and 'Tổng thu nhập'. The table contains two rows of data. The first row has '2016-03-19' under 'ngayban' and '1140' under 'Tổng thu nhập'. The second row has '2016-03-20' under 'ngayban' and '480' under 'Tổng thu nhập'. The cell containing '2016-03-19' is highlighted with a dashed border.

	ngayban	Tổng thu nhập
1	2016-03-19	1140
2	2016-03-20	480

Truy vấn con

- Có thể sử dụng một câu lệnh SELECT hoặc một truy vấn trả về các bản ghi được để **làm điều kiện cho câu truy lệnh SELECT hoặc truy vấn khác**.
- Truy vấn lồng (inner query) còn được gọi là truy vấn cha, và truy vấn bên trong được gọi là truy vấn con. Mục đích của truy vấn con là **để trả về kết quả** cho truy vấn bên ngoài.
- Dạng **đơn giản nhất** của một truy vấn con là **chỉ trả về một cột**. Truy vấn cha có thể sử dụng kết quả của truy vấn con này bằng một dấu =

Truy vấn con

Ví dụ: lấy về các đơn hàng cuối cùng trong bảng hóa đơn

```
SELECT * FROM ThongKeBanHang
```

```
WHERE ngayban = (SELECT MAX(ngayban) FROM  
ThongKeBanHang)
```

	sanpham	giatien	ngayban
1	Samsung Trend	120	2016-03-20
2	Nokia 1200	120	2016-03-20
3	Samsung Trend	120	2016-03-20
4	Nokia 1200	120	2016-03-20

Truy vấn con

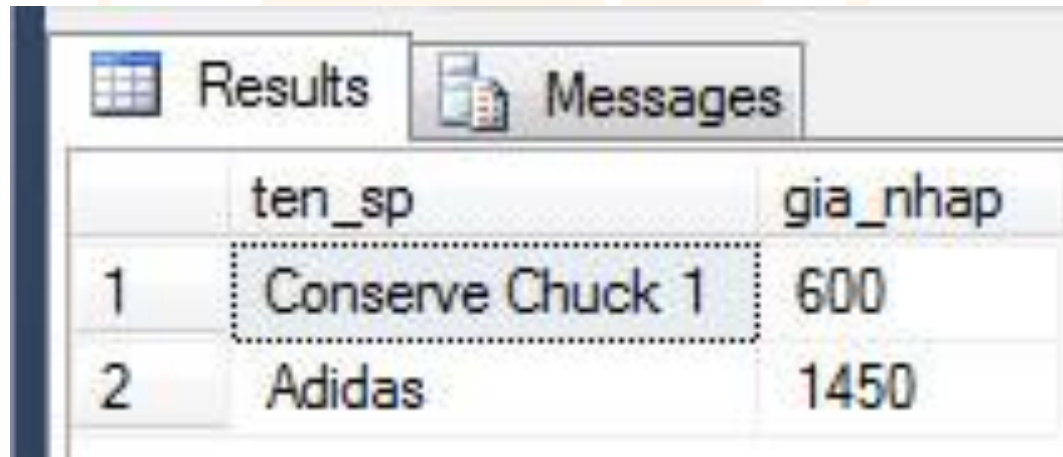
Truy vấn đa trị:

- Nếu toán tử = được sử dụng với truy vấn con, truy vấn con đó chỉ được trả về một giá trị vô hướng duy nhất.
- Nếu trả về nhiều hơn một giá trị, truy vấn sẽ bị lỗi và không được thực thi.
- Các từ khóa **ANY, ALL, IN, và EXISTS** có thể được sử dụng trong mệnh đề WHERE của câu lệnh SELECT, khi đó truy vấn trả về một cột hay có nhiều dòng.
- Những từ khóa này còn được gọi là các vị từ (**predicates**), được sử dụng với các truy vấn đa trị.

Truy vấn con

Ví dụ: lấy về tên và giá sản phẩm mà mã loại sản phẩm là 1.

```
SELECT ten_sp, gia_nhap FROM SAN_PHAM WHERE  
ma_sp IN (SELECT ma_sp FROM SAN_PHAM WHERE  
ma_lsp = 1)
```



	ten_sp	gia_nhap
1	Conserve Chuck 1	600
2	Adidas	1450

Ghép nối - join

Các phép ghép nối được sử dụng để **lấy dữ liệu từ hai hoặc nhiều bảng dựa trên mối quan hệ luận lý** (logical relationship) giữa các bảng. Nó định nghĩa cách thức hai bảng được quan hệ với nhau bằng cách:

- Chỉ ra cột từ mỗi bảng được sử dụng cho việc ghép nối. Một ghép nối đặc thù chỉ ra **khóa ngoại** từ một bảng kết hợp (associated) với **khóa chính** từ một bảng khác.
- Toán tử luận lý $=$, $<>$ được sử dụng để so sánh giá trị từ các cột.

Ghép nối - join

Cú pháp:

```
SELECT <ColumnName1>, <ColumnName2>...<ColumnNameN>  
FROM Table_A AS Table_Alias_A  
JOIN  
Table_B AS Table_Alias_B  
ON  
Table_Alias_A.<CommonColumn> = Table_Alias_B.<CommonColumn>
```

Trong đó:

- **<ColumnName1>, <ColumnName2>**: là một danh sách cột được hiển thị
- **Table_A**: là tên của bảng phía bên trái của từ khóa JOIN.
- **Table_B**: là tên của bảng phía bên phải của từ khóa JOIN.

Ghép nối - join

- **AS Table_Alias**: là cách đặt tên bí danh cho bảng. Có thể xác định một bí danh cho bảng trong truy vấn để không cần phải dùng đến tên đầy đủ của nó (Tên đầy đủ của bảng thường dài, nó gồm tên lược đồ với tên bảng).
- **<CommonColumn>**: là cột chung có trong hai cả bảng tham gia ghép nối. Trong trường hợp đó, ghép nối chỉ thành công nếu các cột có các giá trị khớp nhau

Ghép nối - join

	idbh	idkh	idsp	ngaymua
1	1	3	2	2016-04-30 00:00:00.000
2	2	1	1	2016-04-01 00:00:00.000
3	3	1	3	2016-04-03 00:00:00.000
4	4	2	3	2016-04-08 00:00:00.000
5	5	3	3	2016-04-09 00:00:00.000
6	6	1	1	2016-04-15 00:00:00.000

Bảng tblBanHang



Bảng tblKhachHang

	idkh	tenkh	namsinh
1	1	Messi	1986
2	2	Ronaldo	1983
3	3	Lukaku	1990

Ghép nối - join

```
SELECT idbh AS 'Mã hóa đơn', tenkh AS 'Tên khách hàng',  
ngaymua AS 'Ngày mua' FROM
```

```
tblBanHang bh JOIN tblKhachHang kh ON bh.idkh = kh.idkh
```

Bí danh

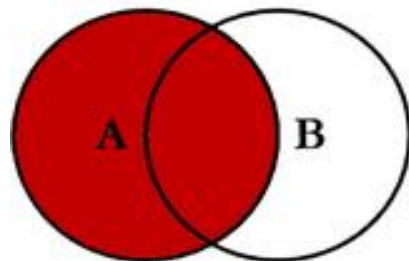
Phép nối

Điều kiện ghép nối

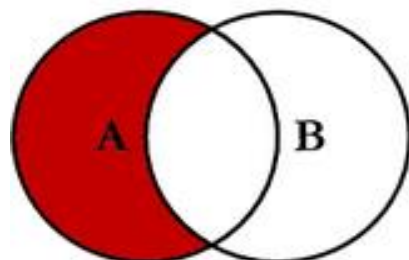
	Mã hóa đơn	Tên khách hàng	Ngày mua
1	1	Lukaku	2016-04-30 00:00:00.000
2	2	Messi	2016-04-01 00:00:00.000
3	3	Messi	2016-04-03 00:00:00.000
4	4	Ronaldo	2016-04-08 00:00:00.000
5	5	Lukaku	2016-04-09 00:00:00.000
6	6	Messi	2016-04-15 00:00:00.000

Ghép nối - join

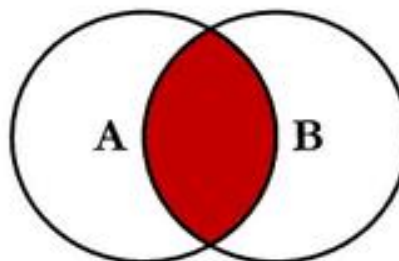
SQL JOINS



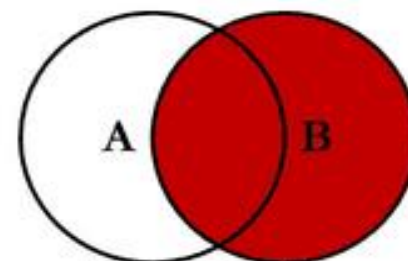
```
SELECT <select_list>  
FROM TableA A  
LEFT JOIN TableB B  
ON A.Key = B.Key
```



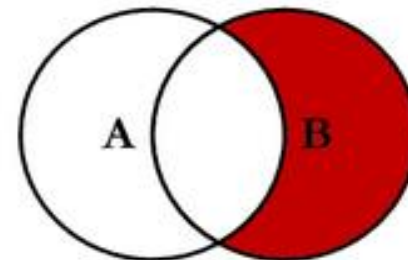
```
SELECT <select_list>  
FROM TableA A  
LEFT JOIN TableB B  
ON A.Key = B.Key  
WHERE B.Key IS NULL
```



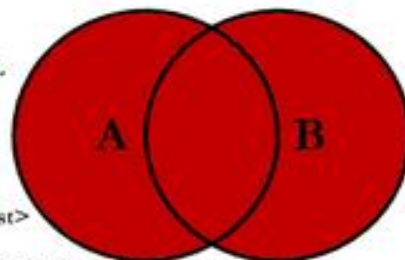
```
SELECT <select_list>  
FROM TableA A  
INNER JOIN TableB B  
ON A.Key = B.Key
```



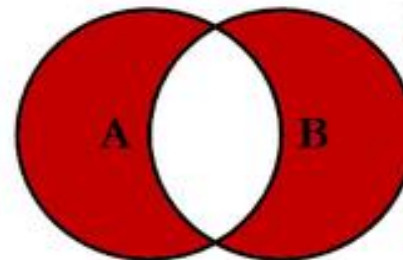
```
SELECT <select_list>  
FROM TableA A  
RIGHT JOIN TableB B  
ON A.Key = B.Key
```



```
SELECT <select_list>  
FROM TableA A  
RIGHT JOIN TableB B  
ON A.Key = B.Key  
WHERE A.Key IS NULL
```



```
SELECT <select_list>  
FROM TableA A  
FULL OUTER JOIN TableB B  
ON A.Key = B.Key
```



```
SELECT <select_list>  
FROM TableA A  
FULL OUTER JOIN TableB B  
ON A.Key = B.Key  
WHERE A.Key IS NULL  
OR B.Key IS NULL
```

© C.L. Moffatt, 2008

Ghép nối - join

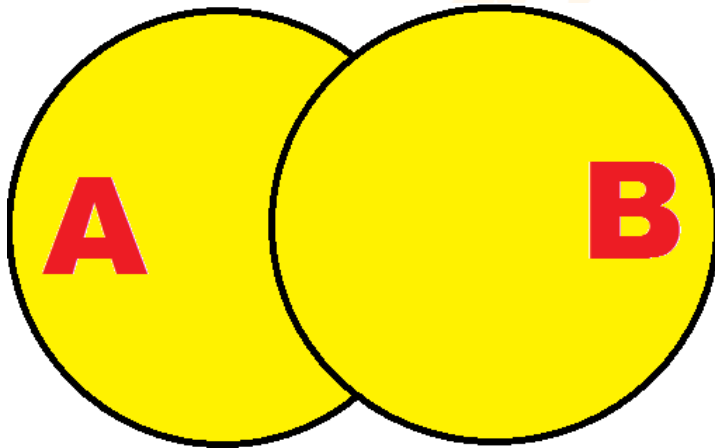
Toán tử UNION:

- Kết quả từ hai câu lệnh truy vấn khác nhau có thể được **kết hợp thành một tập kết quả** duy nhất bằng toán tử UNION.
- Các câu lệnh truy vấn **phải tương thích về kiểu và số lượng các cột**.
- Tên các cột trong mỗi câu lệnh có thể khác nhau, nhưng kiểu dữ liệu phải tương thích (**giống nhau**).

```
Query_Statement1  
UNION [ALL]  
Query_Statement2
```

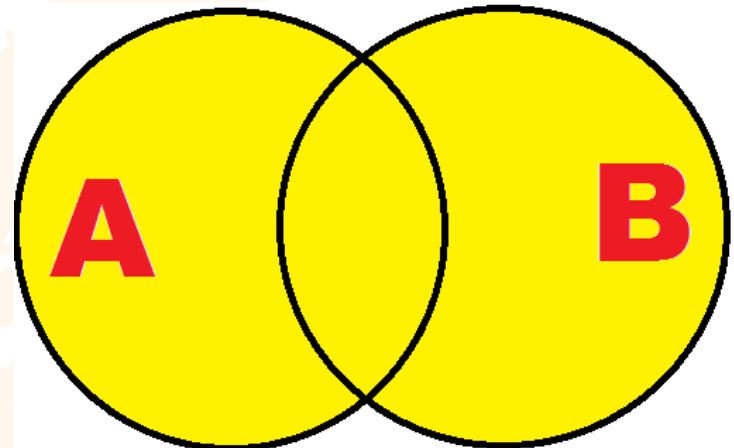
Ghép nối - join

UNION



UNION sẽ hội kết quả của 2 câu truy vấn sau khi LOẠI BỎ kết quả trùng lặp

UNION ALL



UNION ALL sẽ KHÔNG LOẠI BỎ kết quả trùng lặp

Ghép nối - join

(Lệnh A) `SELECT * FROM tblKhachHang WHERE
tenkh LIKE 'M%'`

	idkh	tenkh	namsinh
1	1	Messi	1986

(Lệnh B) `SELECT * FROM tblKhachHang WHERE
namsinh >= 1986`

	idkh	tenkh	namsinh
1	1	Messi	1986
2	3	Lukaku	1990

Ghép nối - join

```
SELECT * FROM tblKhachHang WHERE tenkh LIKE 'M%'
```

```
UNION
```

```
SELECT * FROM tblKhachHang WHERE namsinh >= 1986
```

	idkh	tenkh	namsinh
1	1	Messi	1986
2	3	Lukaku	1990

Ghép nối - join

```
SELECT * FROM tblKhachHang WHERE tenkh LIKE 'M%'
```

```
UNION ALL
```

```
SELECT * FROM tblKhachHang WHERE namsinh >= 1986
```

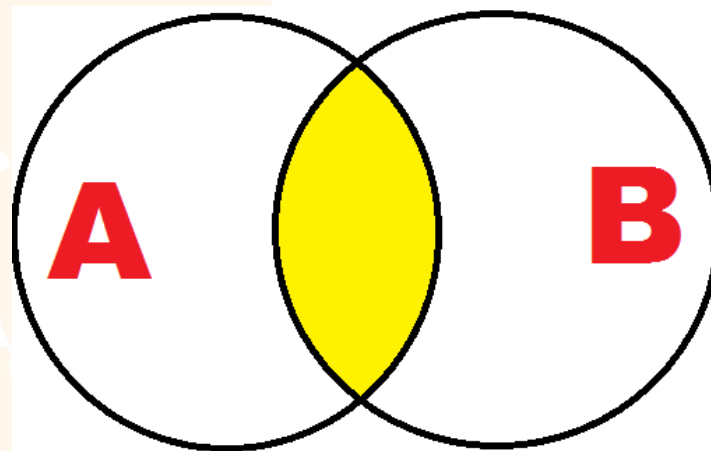
	idkh	tenkh	namsinh
1	1	Messi	1986
2	1	Messi	1986
3	3	Lukaku	1990

Ghép nối - join

Toán tử INTERSECT: được sử dụng với hai câu lệnh truy vấn để trả về tập các **dòng chung duy nhất** của cả hai câu lệnh.

Cú pháp:

```
Query_statement1  
INTERSECT  
Query_statement2
```



Nguyên tắc:

- Số lượng **cột và thứ tự** của các cột được chỉ ra trong hai lệnh truy vấn phải **giống nhau**.
- **Kiểu dữ liệu** của các cột phải **phù hợp**.

Ghép nối - join

SELECT * FROM tblBanHang

SELECT * FROM tblSanPham

	idbh	idkh	idsp	ngaymua
1	1	3	2	2016-04-30 00:00:00.000
2	2	1	1	2016-04-01 00:00:00.000
3	3	1	3	2016-04-03 00:00:00.000
4	4	2	3	2016-04-08 00:00:00.000
5	5	3	3	2016-04-09 00:00:00.000
6	6	1	1	2016-04-15 00:00:00.000

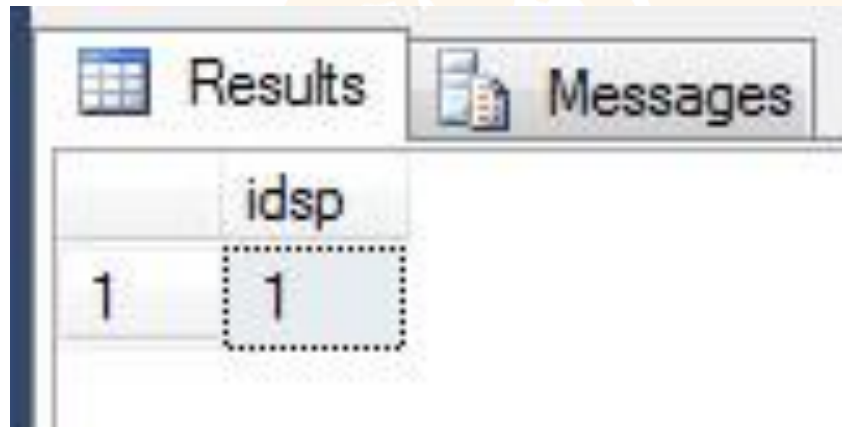
	id	tensp	giaban	mota
1	1	Nokia	120	Sản phẩm phổ thông
2	2	Samsung	350	Cao cấp
3	3	LG	230	Rẻ tiền

Ghép nối - join

```
SELECT idsp FROM tblBanHang WHERE ngaymua > '2016-04-05'
```

```
INTERSECT
```

```
SELECT id FROM tblSanPham WHERE giaban < 200
```



The screenshot shows the 'Results' pane in SQL Server Enterprise Manager. It displays a single row of data with two columns. The first column contains the value '1' and the second column contains the value '1'. The column headers are not visible in this view.

	idsp
1	1

Ghép nối - join

Toán tử PIVOT: được sử dụng với mục đích **đảo dữ liệu hàng thành cột** => dễ nhìn – dễ hiểu.

Ví dụ: trong bảng dữ liệu bán hàng, mã hóa đơn, mã nhân viên và hình thức thanh toán lưu trong mỗi dòng bản ghi.

	id	sanpham	giatien	ngayban	ma_nv	thanhtoan
1	1	Nokia 1200	100	2016-03-19	1	Tiền mặt
2	2	Samsung Trend	120	2016-03-19	2	Visa
3	3	HTC One	50	2016-03-19	1	Tiền mặt
4	4	HTC One	50	2016-03-19	2	Chuyển khoản
5	5	HTC One M8	100	2016-03-19	3	Tiền mặt
6	6	HTC One M9	150	2016-03-19	1	Tiền mặt
7	7	Samsung Trend	120	2016-03-20	3	Tiền mặt
8	8	Nokia 1200	120	2016-03-20	1	Chuyển khoản
9	9	Nokia 1200	100	2016-03-19	3	Tiền mặt
10	10	Samsung Trend	120	2016-03-19	3	Chuyển khoản
11	11	HTC One	50	2016-03-19	2	Tiền mặt
12	12	HTC One	50	2016-03-19	1	3
13	13	HTC One M8	100	2016-03-19	3	Tiền mặt
14	14	HTC One M9	150	2016-03-19	1	Visa
15	15	Samsung Trend	120	2016-03-20	3	Visa
16	16	Nokia 1200	120	2016-03-20	1	Visa

Ghép nối - join

Lệnh sau chỉ lấy về mã hóa đơn, mã nhân viên và hình thức thanh toán:

```
SELECT id, ma_nv, thanhtoan FROM ThongKeBanHang
```

	id	ma_nv	thanhtoan
1	1	1	Tiền mặt
2	2	2	Visa
3	3	1	Tiền mặt
4	4	2	Chuyển khoản
5	5	3	Tiền mặt
6	6	1	Tiền mặt
7	7	3	Tiền mặt
8	8	1	Chuyển khoản
9	9	3	Tiền mặt
10	10	3	Chuyển khoản

Ghép nối - join

Yêu cầu nghiệp vụ là: thống kê mỗi nhân viên đã thực hiện bao nhiêu hình thức thanh toán. Ví dụ: nhân viên A – tiền mặt (3 lần) | chuyển khoản (6 lần)....

Code:

```
-- PIVOT
```

```
SELECT ma_nv, [Tiền mặt], [Visa], [Chuyển khoản]  
FROM (SELECT id, ma_nv, thanhtoan FROM ThongKeBanHang)  
BangNgon  
PIVOT (COUNT(id) FOR thanhtoan IN ([Tiền mặt], [Visa],  
[Chuyển khoản])) BangDao
```

Ghép nối - join

Kết quả:

- Thống kê được hình thức thanh toán tương ứng với nhân viên
- Hình thức thanh toán chuyển sang cột => dễ nhìn

	ma_nv	Tiền mặt	Visa	Chuyển khoản
1	1	3	2	1
2	2	1	1	1
3	3	4	1	1

Tóm tắt bài học

- **Mệnh đề GROUP BY** và cá hàm thống kê cho phép tổng hợp dữ liệu để trình bày thông tin tổng hợp.
- **Truy vấn con** cho phép tập kết quả của một câu lệnh SELECT này sử dụng để làm điều kiện cho một câu lệnh SELECT khác.
- Việc **ghép nối (Joins)** giúp bạn kết hợp cá cột dữ liệu từ hai hay nhiều bảng dựa trên mối quan hệ logic giữa các bảng.
- Các **toán tử tập hợp** giúp bạn kết hợp/gộp các dòng dữ liệu từ từ hai hay nhiều bảng

TRƯỜNG ĐÀO TẠO LẬP TRÌNH VIÊN VÀ QUẢN TRỊ MẠNG QUỐC TẾ BACHKHOA-APTECH

Thank for watching!

