



**MCI**



**MAGIC CODE INSTITUTE**

## **BUỔI 5** TRUY VẤN DỮ LIỆU TRÊN BẢNG PHỤ



## 1

# Phương pháp sử dụng Subquery

Truy vấn con trong SQL (subquery) là gì?

Trong SQL Server, truy vấn con là một truy vấn nằm trong một truy vấn khác. Bạn có thể tạo các truy vấn trong lệnh SQL. Các truy vấn con này nằm trong mệnh đề **WHERE**, **FROM** hoặc **SELECT**

```
SELECT order_id,  
       order_quantity,  
       product_name  
FROM Orders  
WHERE order_quantity > (SELECT AVG(order_quantity)  
                          FROM Orders)
```

Lưu ý:

- Truy vấn con còn được gọi là **INNER QUERY** hay **INNER SELECT**.
- Truy vấn chính mà chứa truy vấn con được gọi là **OUTER QUERY** hay **OUTER SELECT**.

Subquery



# Phương pháp sử dụng Subquery

**Có một vài quy tắc mà Sub query phải tuân theo:**

- Sub query phải được đặt trong dấu ngoặc đơn.
- Không thể sử dụng lệnh ORDER BY trong sub query, mặc dù truy vấn chính có thể sử dụng ORDER BY.
- Lệnh GROUP BY được sử dụng bình thường trong 1 sub query.

## CTE trong SQL Server là gì?

- CTE có thể được xem như **một bảng chứa dữ liệu tạm thời** từ câu lệnh được định nghĩa trong phạm vi của nó.
- CTE tương tự như một bảng dẫn xuất (derived table) ở chỗ nó không được lưu trữ như một đối tượng và chỉ kéo dài trong suốt thời gian của câu truy vấn. Không giống như bảng dẫn xuất, CTE có thể tự tham chiếu tới bản thân của nó và có thể tham chiếu nhiều lần trong một câu truy vấn.

```
WITH engineers as (  
    select *  
    from employees  
    where dept='Engineering'  
)  
select *  
from engineers  
where ...
```

Annotations:

- WITH
- CTE name (points to `engineers`)
- CTE Body (points to the query inside the parentheses)
- CTE Usage (points to `from engineers`)

## Mục đích của CTE

- Tạo truy vấn đệ quy (recursive query).
- Thay thế View trong một số trường hợp.
- Sử dụng được nhiều CTE trong một truy vấn duy nhất

## Ưu điểm của CTE

CTE có nhiều ưu điểm như khả năng đọc dữ liệu được cải thiện và dễ dàng bảo trì các truy vấn phức tạp. Các truy vấn có thể được phân thành các khối nhỏ, đơn giản. Những khối này được sử dụng để xây dựng các CTE phức tạp hơn cho đến khi tập hợp kết quả cuối cùng được tạo ra.

Cú pháp của CTE:

```
1  WITH expression_name [ ( column_name [...n] ) ]  
2  
3  AS  
4  
5  (  
6  
7      CTE_query_definition  
8  
9  )  
10  
11  SELECT  
12  
13  FROM expression_name;
```

Ví dụ sử dụng CTE:

```
-- Xác định tên biểu thức CTE và danh sách các cột.  
WITH PivotOrders_CTE (Order_id, Total_quantity, Total_value)  
AS  
-- Xác định truy vấn CTE.  
(  
    SELECT Order_id, sum(Order_quantity) AS Total_quantity, sum(value) AS Total_value  
    FROM Orders  
    GROUP BY Order_id)  
-- Xác định truy vấn bên ngoài tham chiếu đến tên CTE.  
SELECT Order_id, Total_quantity, Total_value  
FROM PivotOrders_CTE  
WHERE Total_quantity > 100  
ORDER BY Total_quantity Desc;
```

# Phương pháp sử dụng BẢNG TẠM-TEMP TABLE

- Sẽ rất có lợi khi lưu trữ dữ liệu trong các **bảng tạm** thời của SQL Server thay vì thao tác hoặc làm việc với các bảng cố định.
- Khi bạn muốn có đầy đủ quyền truy cập vào các **bảng** trong Database, nhưng bạn lại không có. Bạn có thể sử dụng quyền truy cập đọc hiện có của mình để kéo dữ liệu vào bảng tạm thời của **SQL-Server** và thực hiện các điều chỉnh từ đó.
- Hoặc bạn không có quyền để tạo bảng trong cơ sở dữ liệu hiện có, bạn có thể tạo **bảng tạm thời** SQL Server mà bạn có thể thao tác.
- Cuối cùng, bạn có thể rơi vào tình huống chỉ cần hiển thị dữ liệu trong phiên hiện tại, và muốn update insert data trước khi hiển thị.



**Bảng tạm trong SQL server là gì ?**

**Bảng tạm** là các có cấu trúc và chức năng như một bảng cố định bình thường trong SQL Server. Nhưng thay vì tạo ra một bảng trong Database, bảng tạm được tạo ra và lưu trữ trong tempdb. Chúng ta thường tạo bảng tạm trong một câu truy vấn, trong xử lý của một procedure hoặc trong một function (chỉ sử dụng được biến kiểu bảng).

Đối với SQL Server có 2 dạng bảng tạm đó là:

- **Local temporary table (#Table1):** Sử dụng để tạo ra bảng tạm và tồn tại trong kết nối của người dùng tạo ra bảng đó và sẽ bị huỷ khi ngắt kết nối.
- **Global temporary table (##Table2):** Sử dụng để tạo ra bảng tạm và tồn tại đến khi nào tất cả các kết nối đến cơ sở dữ liệu làm việc đóng hết. Có thể sử dụng ở kết nối của người dùng khác.

## Temporary Table



### 3

# Phương pháp sử dụng BẢNG TẠM-TEMP TABLE

## Cú pháp tạo bảng tạm

Local temporary table

```
--Create temp table
CREATE TABLE #TempTable1
(
    ID INT IDENTITY PRIMARY KEY NOT NULL,
    Name VARCHAR(10) NOT NULL,
    DOB DATETIME null
)
GO
```



## 3

# Phương pháp sử dụng BẢNG TẠM-TEMP TABLE

Cú pháp tạo bảng tạm:

Global temporary table

```
CREATE TABLE ##TempTable2
(
    ID INT IDENTITY PRIMARY KEY NOT NULL,
    Name VARCHAR(10) NOT NULL,
    DOB DATETIME null
)
GO

INSERT INTO ##TempTable2
    ( Name, DOB )
VALUES (
    'TONA', -- Name - varchar(10)
    GETDATE() -- DOB - datetime
)
GO

SELECT * FROM ##TempTable2
```



**BT1:** Từ bảng dữ liệu **Orders** và **Returns** hãy tạo ra kết quả gồm 3 cột:

Năm

Tháng

Tổng đơn hàng

Tổng đơn hàng bị trả lại

=>> Sử dụng cả phương pháp Sub query và CTE

	year	month	total_orders	total_returns
1	2009	1	133	11
2	2009	2	101	11
3	2009	3	124	9
4	2009	4	118	13
5	2009	5	119	15
6	2009	6	105	13
7	2009	7	120	13
8	2009	8	127	11
9	2009	9	123	7
10	2009	10	100	8
11	2009	11	106	6

**BT2:** Từ bảng **Orders** và **Returns** hãy tạo ra kết quả gồm các cột:

Năm

Tháng

Loại sản phẩm

Tổng giá trị (Total\_value)

Tổng giá trị hoàn hàng (Total\_value\_of\_returned)

=>> Sử dụng cả phương pháp Sub query và CTE

	year	month	product_category	total_value	total_value_return
1	2009	1	Furniture	202384.6983	39726.1084
2	2009	1	Office Supplies	107072.4041	25598.3919
3	2009	1	Technology	194022.3464	3187.8564
4	2009	2	Furniture	138050.5225	5552.3856
5	2009	2	Office Supplies	46968.3552	26726.2147
6	2009	2	Technology	145430.1075	17666.8502
7	2009	3	Furniture	106698.1748	11970.6816
8	2009	3	Office Supplies	67964.6083	882.9428
9	2009	3	Technology	227399.9161	23415.3457
10	2009	4	Furniture	142856.9672	4312.5972
11	2009	4	Office Supplies	111054.9543	7784.1959

**BT3:** Từ tập dữ liệu đã cho, truy xuất tất cả các đơn đặt hàng trong năm 2012 (từ 2012-01-01 đến 2012-12-31) và tóm tắt thông tin như sau:

	manager_name	manager_level	manager_id	number_items	total_quantity	total_value	total_profit
1	Pat	3	115	264	6463	560685.682	226740.4393
2	Erin	3	113	452	11537	818930.6175	335563.5014
3	Chris	2	111	138	3567	204636.1426	82910.1684
4	William	3	112	655	17446	1089018.9105	448907.8411
5	Sam	4	114	386	9846	557977.3138	222439.6058

Trong đó:

- number\_items: tổng số mặt hàng, không bao gồm các mặt hàng bị trả lại.
- total\_quantity: tổng số lượng mặt hàng được giao cho từng người quản lý, không bao gồm các mặt hàng bị trả lại.
- total\_value: tổng giá trị của các mặt hàng được giao cho từng người quản lý, không bao gồm các mặt hàng bị trả lại.
- total\_profit: tổng lợi nhuận của các mặt hàng được giao cho từng người quản lý, không bao gồm các mặt hàng bị trả lại.

Bài 1:

--Phương pháp CTE

with a as

(

select year(order\_date) as year, month(order\_date) as month, count(distinct order\_id) as total\_orders from orders  
group by

year(order\_date),

month(order\_date)

)

,

b as

(

select year(returned\_date) as year, month(returned\_date) as month, count(order\_id) as total\_returns from returns  
group by

year(returned\_date),

month(returned\_date)

)

select a.year, b.month, a.total\_orders , b.total\_returns

from a as a

left join b as b

on a.year = b.year and a.month=b.month

**Các bạn nên thử thêm cách sử dụng sub query nhé!  
Tương tự với BT2**



Bài 3:

--Phương pháp CTE

with a as

```
(
select o.order_date,o.order_id,o.order_quantity,o.value,o.profit, r.returned_date, r.status,m.manager_name, m.manager_id,
m.manager_level,m.manager_phone from orders as o
left join returns as r
on o.order_id=r.order_id
left join profiles as p
on o.province = p.province
left join Managers as m
on p.manager=m.manager_name
where o.order_date between '2012-1-1' and '2012-12-31'
)
```

```
select
a.manager_name,
a.manager_level,
a.manager_id,
count(a.order_id) as number_items,
sum(a.order_quantity) as total_quantity,
sum(value) as total_value,
sum(profit) as total_profit
from a
where a.status is null
group by a.manager_name,a.manager_id, a.manager_level
```

**Các bạn nên thử thêm cách sử dụng sub query nhé!**

