

Các kỹ thuật truy vấn tăng cường

4. Truy vấn nhiều bảng.

Vì không thể lưu toàn bộ dữ liệu trong một bảng đơn. Chúng ta thường cần phải kết hợp và lấy dữ liệu từ nhiều bảng khác nhau.

Có ba cách kết hợp dữ liệu từ nhiều bảng:

- **Unions** – kết hợp các hàng từ nhiều bảng dữ liệu
- **Subqueries** – đặt một truy vấn bên trong một truy vấn khác
- **Joins** – kết hợp các cột từ nhiều bảng dữ liệu

4.1 Union

Toán tử Union kết hợp kết quả của hai hay nhiều lệnh SELECT vào trong một tập kết quả duy nhất. Mỗi câu lệnh SELECT phải có cấu trúc giống nhau hoặc kiểu của các cột, số lượng các cột tương thích với nhau. Tên của các cột có thể khác nhau trong mỗi câu lệnh SELECT. Tập kết quả chỉ hiển thị những tên cột của câu lệnh SELECT đầu tiên.

Cú pháp:

```
SELECT statement  
UNION [ALL]  
SELECT statement
```

Một cách mặc định, thao tác UNION loại bỏ sự lặp lại trong tập kết quả. Nhưng nếu bạn sử dụng mệnh đề ALL cùng với câu lệnh UNION, thì truy vấn sẽ trả lại tất cả các hàng.

Xét một ứng dụng về nhà băng. Các bảng *Saving_Account* và *Current_Account* lưu thông tin về khách hàng cùng với tài khoản tiết kiệm và tài khoản vãng lai tương ứng. Để xem *Account_No* và *Name* của tất cả các khách hàng của nhà băng, bạn phải lấy các bản ghi từ cả hai bảng. Nội dung của hai bảng như sau:

Bảng *Saving_Account*

<i>Account_No</i>	<i>Name</i>
S001	James
S002	Rita
S003	Mary
S004	Valentina

Bảng *Curent_Account*

<i>Account_No</i>	<i>Name</i>
C001	Michael
C002	Robin

Truy vấn sau đây sẽ đưa ra các chi tiết của các khách hàng của nhà băng:

```
SELECT Account_No, Name FROM Saving_Account  
UNION
```

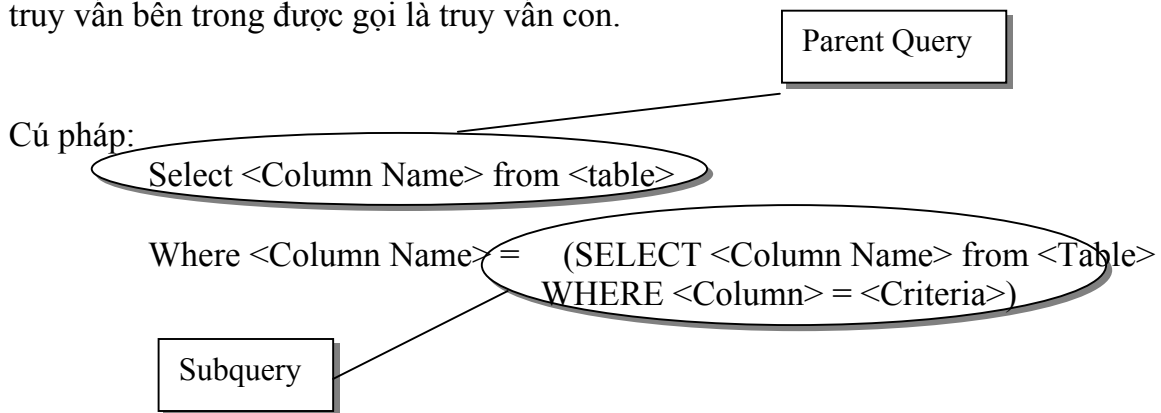
SELECT Account_No, Name FROM Current_Account

Truy vấn trên đây sẽ hiển thị kết quả như sau:

Account No	Name
S001	James
S002	Rita
S003	Mary
S004	Valentina
C001	Michael
C002	Robin

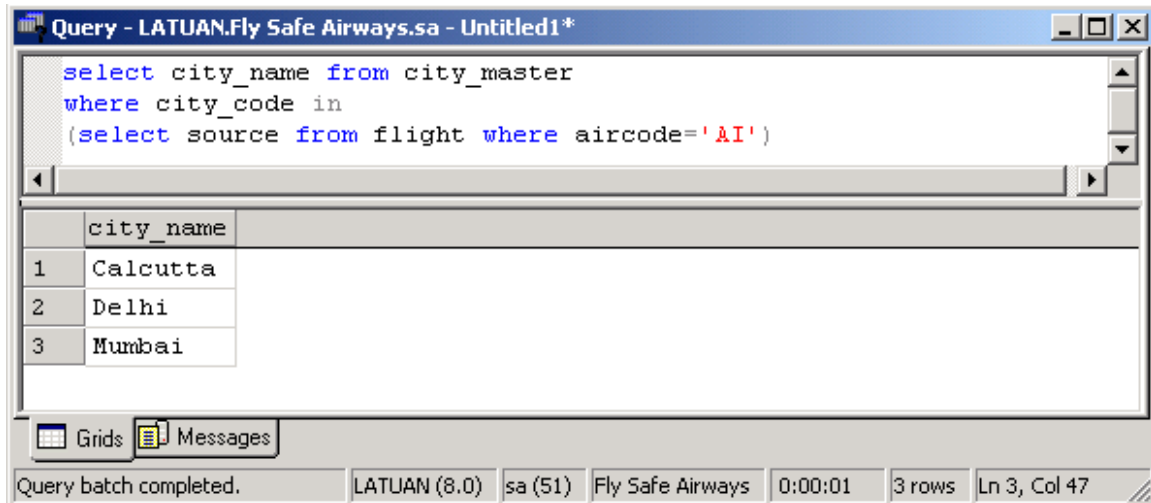
4.2 Truy vấn con

Bạn có thể sử dụng một câu lệnh SELECT để trả về các bản ghi mà một câu lệnh SELECT khác sẽ sử dụng. Truy vấn bao bên ngoài được gọi là truy vấn cha và truy vấn bên trong được gọi là truy vấn con.



Truy vấn con sẽ trả lại cột được sử dụng trong điều kiện của câu lệnh SELECT thứ nhất. Câu lệnh SELECT trong cùng sẽ được thực thi đầu tiên.

Ví dụ, giả sử rằng bạn muốn biết điểm khởi đầu (tên thành phố bắt đầu chuyến bay) của các chuyến bay Air India cung cấp. Bảng Flight lưu giữ city_code, nhưng nó không được sử dụng được trong trường hợp này, vì bạn muốn biết tên thành phố. Thông tin về thành phố được lưu giữ trong bảng City_master. Truy vấn và kết quả của nó được mô tả như sau:



Hình 4. 1

Ghi nhớ những điểm sau đây khi sử dụng truy vấn con:

- Bạn có thể sử dụng truy vấn con như một sự thay thế cho một giá trị trong mệnh đề SELECT, hoặc như thành phần của mệnh đề WHERE.
- Khi một truy vấn con được đưa vào cùng với một toán tử, có những hạn chế trên các cột và các hàng mà truy vấn con trả về. Những hạn chế đó được tổng kết trong bảng dưới đây:

Một cột		Nhiều cột
Một hàng	Sử dụng =, >, < và các toán tử so sánh khác	Sử dụng EXISTS
Nhiều hàng	Sử dụng ANY, ALL, IN và EXISTS	Sử dụng EXISTS

- Nếu một toán tử so sánh được sử dụng cùng với truy vấn con, và truy vấn con trả lại nhiều hơn một hàng, SQL Server sẽ trả về một lỗi.
- Có thể có nhiều truy vấn con được viết lồng nhau.

4.3 Các hàm tính gộp trong các truy vấn

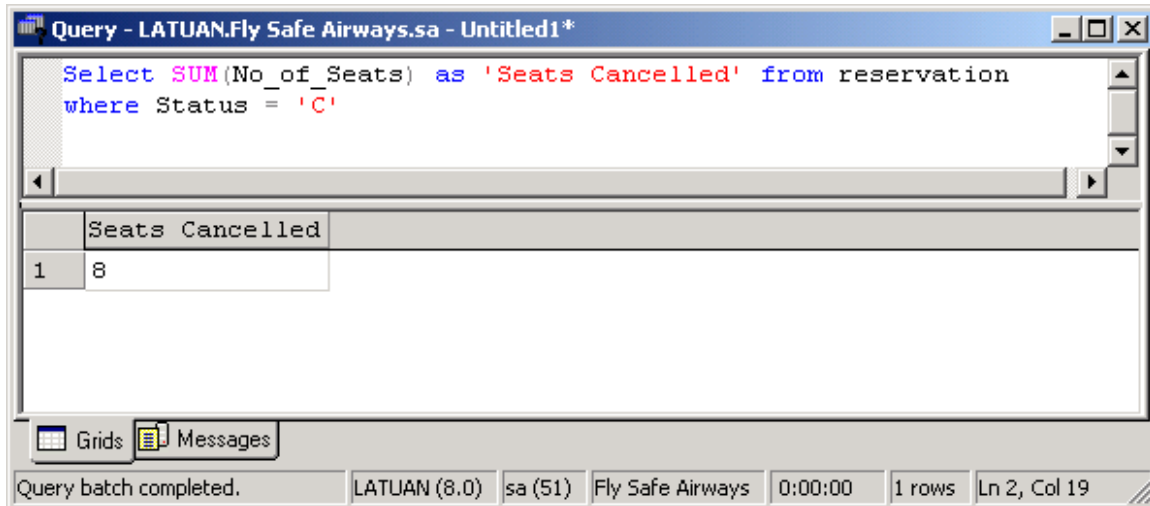
4.3.1 SUM

Trả lại tổng của tất cả các giá trị trong một biểu thức. SUM hỗ trợ việc sử dụng DISTINCT để chỉ tính tổng những giá trị duy nhất trong một biểu thức, trong khi bỏ qua các giá trị NULL. SUM chỉ có thể được sử dụng cho các cột kiểu số.

Cú pháp:

SUM (*Expression*)

Ví dụ, chúng ta sử dụng mệnh đề SUM cùng với câu lệnh SELECT để tìm số các ghế ngồi bị hủy bỏ. Truy vấn và kết quả của nó được minh họa trong hình sau:



The screenshot shows a query window titled "Query - LATUAN.Fly Safe Airways.sa - Untitled1*". The SQL query is: `Select SUM(No_of_Seats) as 'Seats Cancelled' from reservation where Status = 'C'`. Below the query, a table displays the result with one row:

	Seats Cancelled
1	8

. The status bar at the bottom indicates "Query batch completed.", "LATUAN (8.0)", "sa (51)", "Fly Safe Airways", "0:00:00", "1 rows", and "Ln 2, Col 19".

Hình 4. 2

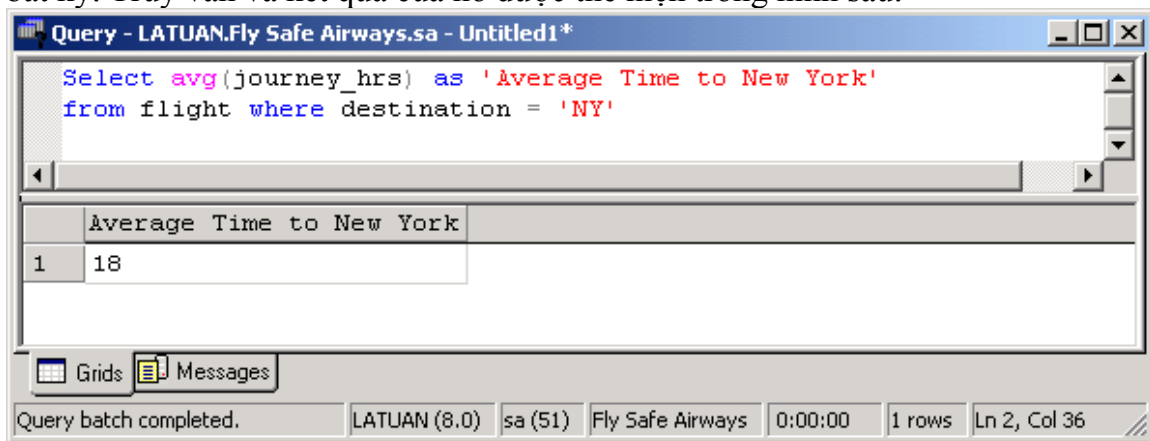
4.3.2 AVG

Hàm AVG trả lại giá trị trung bình của tất cả các giá trị trong một biểu thức. Hàm này chỉ có thể được sử dụng với các cột kiểu số. Nó tự động bỏ đi những giá trị NULL.

Cú pháp:

AVG([ALL|DISTINCT] *Expression*)

Ví dụ, bạn muốn tìm thời gian trung bình để bay tới New York bằng chuyến bay bất kỳ. Truy vấn và kết quả của nó được thể hiện trong hình sau:



The screenshot shows a query window titled "Query - LATUAN.Fly Safe Airways.sa - Untitled1*". The SQL query is: `Select avg(journey_hrs) as 'Average Time to New York' from flight where destination = 'NY'`. Below the query, a table displays the result with one row:

	Average Time to New York
1	18

. The status bar at the bottom indicates "Query batch completed.", "LATUAN (8.0)", "sa (51)", "Fly Safe Airways", "0:00:00", "1 rows", and "Ln 2, Col 36".

Hình 4. 3

4.3.3. COUNT

COUNT trả về số các giá trị khác NULL trong biểu thức được cung cấp. Nếu được sử dụng với DISTINCT, COUNT tìm số các giá trị duy nhất. COUNT hỗ trợ cho cả cột số và cột ký tự. Cột PRIMARY KEY và FOREIGN KEY có thể được tin cậy để sử dụng với COUNT, vì chúng không chứa các giá trị NULL.

Bạn có thể sử dụng một dấu sao (*) làm biểu thức COUNT. Nếu bạn sử dụng một dấu sao, bạn không cần chỉ ra một tên cột cụ thể, và tất cả các hàng sẽ được đếm.

Cú pháp:

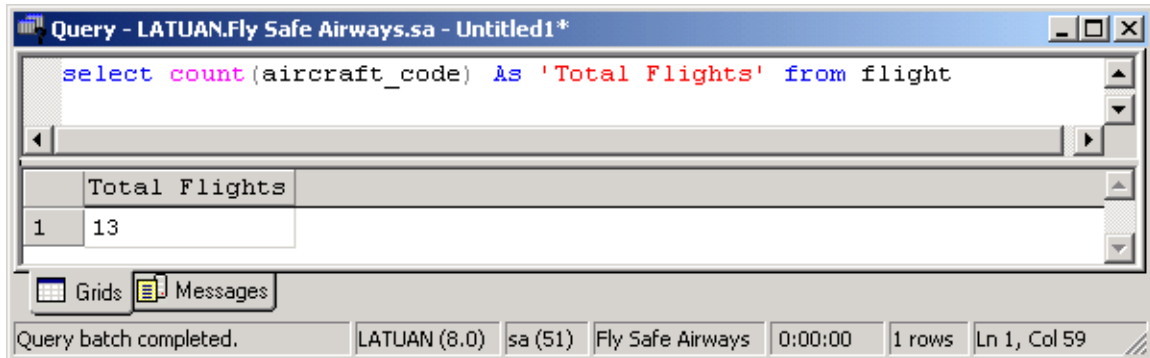
COUNT(*Expression*)

hoặc

COUNT(*)

Ví dụ, bạn muốn xác định tổng số máy bay.

Truy vấn và kết quả của nó được hiển thị trong hình sau:



Hình 4. 4

4.3.4 MAX

MAX trả lại giá trị lớn nhất trong một biểu thức. MAX có thể được sử dụng với các kiểu dữ liệu số, ký tự, và ngày/giờ. Nó bỏ qua các giá trị NULL.

Cú pháp:

MAX(*Expression*)

Ví dụ, bạn muốn tìm giá vé lớn nhất của tất cả các hãng hàng không.

Truy vấn và kết quả của nó được hiển thị trong hình sau:

The screenshot shows a SQL query window titled "Query - LATUAN.Fly Safe Airways.sa - Untitled1*". The query is `select MAX(Fare) as 'Maximum Fare' from flight_details`. The result is displayed in a table with one row and one column, showing the value 81215. The status bar at the bottom indicates "Query batch completed.", "LATUAN (8.0)", "sa (51)", "Fly Safe Airways", "0:00:00", "1 rows", and "Ln 1, Col 55".

	Maximum Fare
1	81215

Hình 4. 5

4.3.5 MIN

MIN trả lại giá trị bé nhất trong một biểu thức. MIN có thể được sử dụng với các cột số, ký tự, ngày/giờ. Khi MIN được sử dụng với các cột ký tự, nó trả lại giá trị bé nhất theo trật tự so sánh. Hàm này bỏ qua các giá trị NULL.

Cú pháp:

MIN(Expression)

Ví dụ, bạn muốn tìm giá vé bé nhất.

Truy vấn và kết quả của nó được hiển thị trong hình sau:

The screenshot shows a SQL query window titled "Query - LATUAN.Fly Safe Airways.sa - Untitled1*". The query is `select MIN(Fare) as 'Minimum Fare' from flight_details`. The result is displayed in a table with one row and one column, showing the value 3409. The status bar at the bottom indicates "Query batch completed.", "LATUAN (8.0)", "sa (51)", "Fly Safe Airways", "0:00:00", "1 rows", and "Ln 1, Col 11".

	Minimum Fare
1	3409

Hình 4. 6

4.4 Mệnh đề GROUP BY

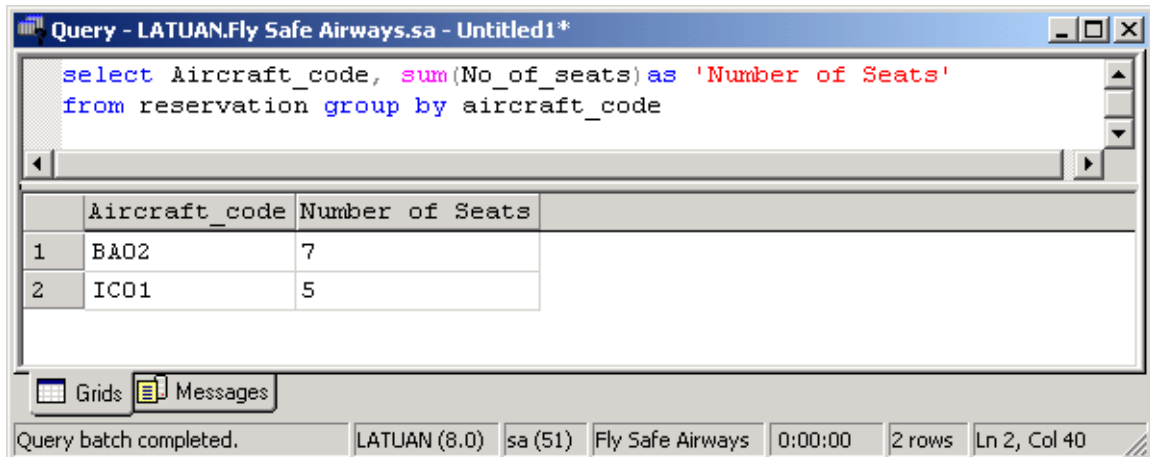
Mệnh đề GROUP BY phân chia tập kết quả thành một hoặc nhiều tập con.

Mỗi tập con có những giá trị và biểu thức chung. Nếu một hàm tính gộp được sử dụng trong một câu lệnh SELECT, mệnh đề GROUP BY cung cấp một giá trị cho mỗi hàm tính gộp.

Cú pháp:

GROUP BY <Column name>

Ví dụ, giả sử rằng bạn muốn tìm ra số số lượng ghế ngồi đã được đặt trên mỗi máy bay. Mệnh đề GROUP BY sẽ nhóm các hàng theo Aircraft_code, và tính tổng No_of_seats cho mỗi chuyến bay. Truy vấn và kết quả của nó được thể hiện trong hình sau:



Query - LATUAN.Fly Safe Airways.sa - Untitled1*

```
select Aircraft_code, sum(No_of_seats) as 'Number of Seats'
from reservation group by aircraft_code
```

	Aircraft_code	Number of Seats
1	BA02	7
2	IC01	5

Query batch completed. LATUAN (8.0) sa (51) Fly Safe Airways 0:00:00 2 rows Ln 2, Col 40

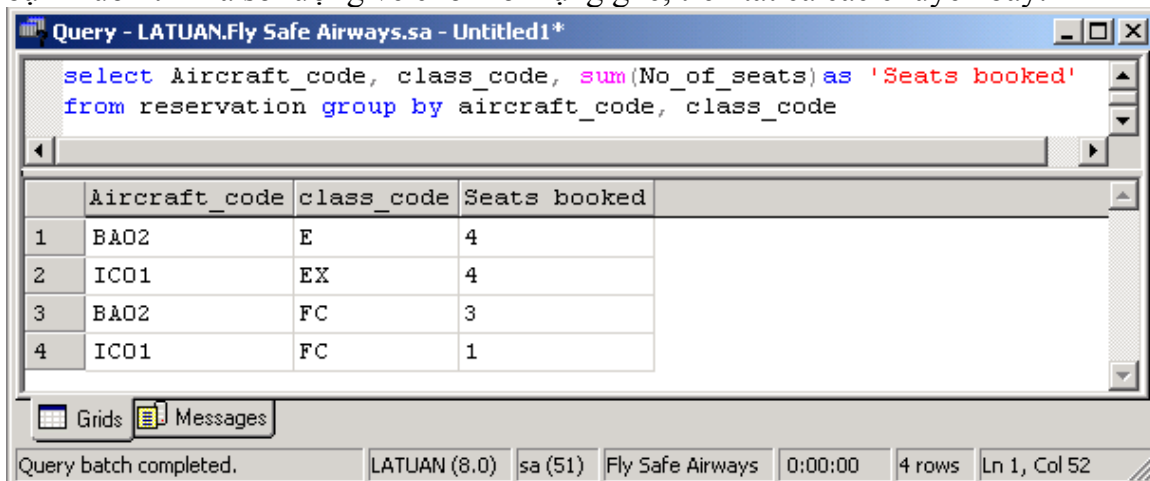
Hình 4. 7

Theo sau từ khóa GROUP BY là một danh sách các cột, được gọi là các cột nhóm hợp. Một cột nhóm hợp hạn chế số hàng của tập kết quả. Với tất cả các cột nhóm hợp, chỉ có duy nhất một hàng. Mỗi tập hàng kết quả chứa dữ liệu tổng kết liên kết với các giá trị đặc trưng.

Trong danh sách SELECT chỉ cho phép những mục sau:

- Các cột nhóm hợp
- Các biểu thức chỉ trả về một giá trị cho mỗi giá trị trong cột nhóm hợp, chẳng hạn như các hàm tính gộp..

Mệnh đề GROUP BY có thể có nhiều hơn một cột nhóm hợp. Ví dụ, giả sử rằng bạn muốn tìm ra số lượng vé cho mỗi hạng ghế, trên tất cả các chuyến bay.



Query - LATUAN.Fly Safe Airways.sa - Untitled1*

```
select Aircraft_code, class_code, sum(No_of_seats) as 'Seats booked'
from reservation group by aircraft_code, class_code
```

	Aircraft_code	class_code	Seats booked
1	BA02	E	4
2	IC01	EX	4
3	BA02	FC	3
4	IC01	FC	1

Query batch completed. LATUAN (8.0) sa (51) Fly Safe Airways 0:00:00 4 rows Ln 1, Col 52

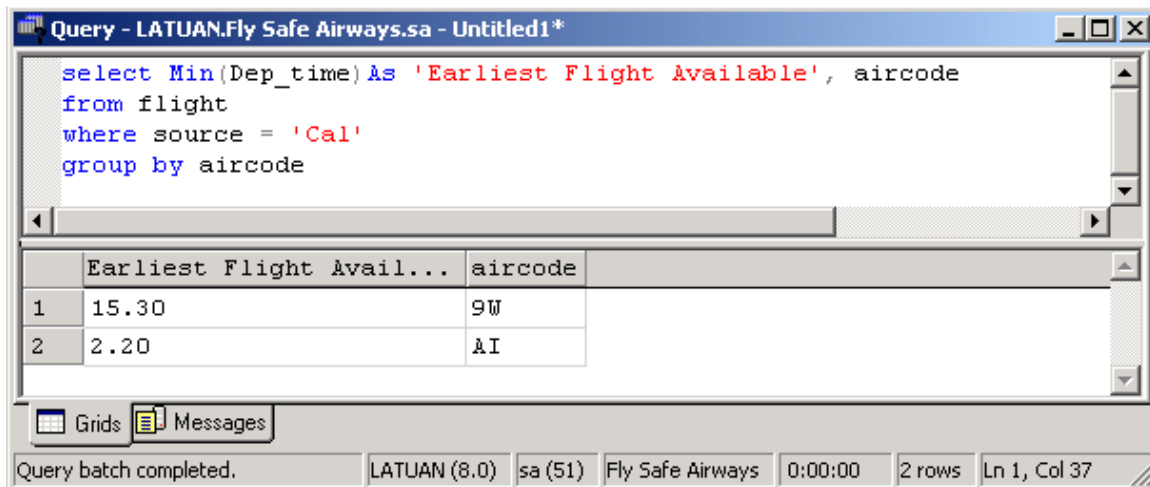
Hình 4. 8

4.4.1 GOUP BY với mệnh đề WHERE

Mệnh đề WHERE có thể được sử dụng với mệnh đề GROUP BY để hạn chế các hàng cho việc nhóm hợp. Các hàng thỏa mãn điều kiện tìm kiếm sẽ được xem xét để nhóm hợp. Các hàng không thỏa mãn điều kiện trong mệnh đề WHERE bị loại ra trước khi việc nhóm hợp xảy ra.

Ví dụ, giả sử rằng bạn muốn tìm ra chuyến bay sớm nhất từ 'Calcutta' trong mỗi hãng hàng không.

Truy vấn và kết quả của nó được thể hiện trong hình sau:



The screenshot shows a query window titled "Query - LATUAN.Fly Safe Airways.sa - Untitled1*". The query is:

```
select Min(Dep_time) As 'Earliest Flight Available', aircode
from flight
where source = 'Cal'
group by aircode
```

The results are displayed in a table with two columns: "Earliest Flight Avail..." and "aircode".

	Earliest Flight Avail...	aircode
1	15.30	9W
2	2.20	AI

At the bottom, a status bar indicates "Query batch completed.", "LATUAN (8.0)", "sa (51)", "Fly Safe Airways", "0:00:00", "2 rows", and "Ln 1, Col 37".

Hình 4. 9

4.4.2 Lựa chọn các hàng bằng việc sử dụng mệnh đề HAVING

Mệnh đề HAVING được sử dụng để lọc các hàng sau khi nhóm hợp chúng. Mệnh đề HAVING thiết lập các điều kiện trên mệnh đề GROUP BY, tương tự như cách WHERE thiết lập các điều kiện trên SELECT. Điều kiện tìm kiếm WHERE được áp dụng trước khi thao tác nhóm hợp xảy ra. Điều kiện tìm kiếm HAVING được áp dụng sau khi thao tác nhóm hợp xảy ra. Cú pháp HAVING tương tự như cú pháp WHERE, ngoại trừ việc HAVING có thể chứa những hàm tính gộp. Mệnh đề HAVING có thể tham khảo tới bất kỳ mục nào xuất hiện trong danh sách SELECT.

Ví dụ, giả sử rằng bạn muốn biết các mã bữa ăn được cung cấp trên nhiều hơn ba chuyến bay. Bạn sẽ phải sử dụng mệnh đề HAVING để thiết lập điều kiện.

Truy vấn và kết quả được thể hiện trong hình sau

The screenshot shows a SQL query window titled "Query - LATUAN.Fly Safe Airways.sa - Untitled1*". The query is:

```
select meal_code, count(*) from airline_meal
group by meal_code having count(*) > 3
```

The results are displayed in a table with the following data:

	meal_code	(No column name)
1	NV	4
2	V	4

The status bar at the bottom indicates "Query batch completed.", "LATUAN (8.0)", "sa (51)", "Fly Safe Airways", "0:00:00", "2 rows", and "Ln 2, Col 37".

Hình 4. 10

Các mệnh đề HAVING và WHERE có thể được sử dụng cùng với nhau trong một câu lệnh SELECT. Việc hiểu đúng trật tự áp dụng những mệnh đề WHERE, GROUP BY, và HAVING giúp cho việc viết những truy vấn hiệu quả cao. Các chức năng của ba mệnh đề như sau:

- WHERE – lọc các hàng do các thao tác được chỉ ra trong mệnh đề FROM
- GROUP BY – nhóm hợp đầu ra của mệnh đề WHERE
- HAVING – lọc các hàng từ kết quả nhóm hợp.

