The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.12741775

CREATING COMBINED QUERIES IN SQL LANGUAGE

SH. PULATOV

Doctor of Technical Sciences, Associate Professor of Kokand State Pedagogical Institute, Uzbekistan.
E-mail: psharifjon@mail.ru

S. KHAIDAROVA

Candidate of Technical Sciences, Associate Professor of Kokand State Pedagogical Institute, Uzbekistan.
E-mail: hay-vb1952@umail.uz

Annotation

This article discusses methods for creating combined queries in SQL language and using the Union operator in it.

Keywords and Expressions: SQL language, UNION and UNION ALL operators, WHERE clause and SELECT statement.

Currently, SQL (Structured Query Language) is the most popular database language. In everyday life we have to work with databases, the SQL language is designed specifically for this. Every time you select a name from your email address book, you are accessing a database. And even when you insert your plastic card into an ATM, the PIN code and account balance are checked through the database [Форта 2014].

Most SQL queries use a single SELECT statement to retrieve data from one or more tables. SQL also allows you to run multiple queries (by using the SELECT statement multiple times) and return the results as a single set. Such queries are usually called combined queries.

It should be noted that the result of combining two queries on the same table is essentially the same as the result obtained by running a single query with multiple conditions in the WHERE clause. In other words, any SELECT statement with multiple conditions WHERE can also be considered a combined query.

Queries in SQL are combined using the UNION operator, which allows you to specify a SELECT statement multiple times, returning a single set of results. Using the UNION operator is quite simple. All you have to do is specify each SELECT statement and insert between them is the UNION keyword.

Let's look at an example. Let's say you want to get a report containing information about all customers from the states of Illinois, Indiana, and Michigan (the Customers table).

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.12741775

Customers table

cust_id	cust_name	cust_address	cust_city	cust_state	cust_zip	cust_country	cust_contact	cust_email
1000000001	Village Toys	200 Maple Lane	Detroit	MI	44444	USA	John Smith	sales@villagetoys.com
1000000002	Kids Place	333 South Lake Drive	Columbus	OH	43333	USA	Michelle Green	
1000000003	Fun4All	1 Sunny Place	Muncie	IN	42222	USA	Jim Jones	jjones@fun4all.com
1000000004	Fun4All	829 Riverside Drive	Phoenix	AZ	88888	USA	Denise L. Stephens	dstephens@fun4all.com
1000000005	The Toy Store	4545 53rd Street	Chicago	IL	54545	USA	Kim Howard	

You also want to include Fun4All customer data regardless of state. Of course, you can create a WHERE condition that will satisfy these requirements, but in this case it is much more convenient to use the UNION operator. As already mentioned, using the UNION operator involves reusing SELECT statements. First, let's look at the individual components of a combined query.

The first request looks like this:

```
SELECT cust_name, cust_contact, cust_email
FROM Customers
WHERE cust_ state IN ( 'IL', 'IN', 'MI') ;
```

The first SELECT statement retrieves all rows related to the states of Illinois, Indiana, and Michigan, the abbreviations specified in the IN statement.

The result of this query is shown below:

```
cust_name        cust_contact        cust_email
- - - - - - - - - - - -   - - - - - - - - - - - -   - - - - - - - - - - - -
Village Toys     John Smith          sales@villagetoys.com
Fun4All          Jim Jones           jjones@fun4all.com
The Toy Store    Kim Howard          NULL
```

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.12741775

The second request looks like this:

```
SELECT cust_name, cust_contact, cust_email
FROM Customers
WHERE cust_name = 'Fun4All';
```

The second SELECT statement uses a simple equality test to find all occurrences of the Fun4All client. The result of this query is shown below:

cust_name	cust_contact	cust_email
Fun4All	Jim Jones	jjones@fun4all.com
Fun4All	Denise L. Stephens	dstephens@fun4all.com

To combine both queries, do the following:

```
SELECT cust_name, cust_contact, cust_email
FROM Customers
WHERE cust_state IN ('IL','IN','MI')
UNION
SELECT cust_name, cust_contact, cust_email
FROM Customers
WHERE cust_name = 'Fun4All';
```

This query contains the original SELECT statements separated by the UNION key word. It causes the DBMS to execute both statements and output the results as a single result set. The result of the combined query is shown below:

cust_name	cust_contact	cust_email
Fun4All	Jim Jones	jjones@fun4all.com
Fun4All	Denise L. Stephens	dstephens@fun4all.com
Village Toys	John Smith	sales@villagetoys.com
The Toy Store	Kim Howard	NULL

It's easy to see that when SELECT statements are executed individually, the first SELECT statement returns three rows, and the second returns two. But when these two statements are combined using the UNION key word, only four rows are returned and not five.

The UNION operator automatically removes all duplicate rows from the result set. Specifically, there is a record for a Fun4All client from Indiana - this row was returned by both SELECT statements. In the case of the UNION operator, the duplicate row is removed.

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.12741775

This is the default behavior of the UNION operator, but you can change it if you wish. If you want all occurrences to be returned, you must use the UNION ALL operator rather than the UNION operator.

Consider the following example.

```
SELECT cust_name,  cust_contact,  cust_email
FROM Customers
WHERE cust_state IN  ('IL', 'IN', 'MI')
UNION ALL
SELECT cust_name,  cust_contact,  cust_email
FROM Customers
WHERE cust_name = 'Fun4All';
```

The result of this query is shown below:

cust_name	cust_contact	cust_email
Village Toys	John Smith	sales@villagetoys.com
Fun4All	Jim Jones	jjones@fun4all.com
The Toy Store	Kim Howard	NULL
Fun4All	Jim Jones	jjones@fun4all.com
Fun4All	Denise L. Stephens	dstephens@fun4all.com

When using the UNION ALL operator, the DBMS does not remove duplicates. Therefore, in this example, there are five lines, and one of them is repeated twice.

This query can also be created using the WHERE clause:

```
SELECT cust_name,  cust_contact,  cust_email
FROM Customers
WHERE cust_state IN  ('IL', 'IN', 'MI')
OR cust_name = 'Fun4All';
```

Combining queries created using the UNION operator and the WHERE clause produces the same result. But comparing these examples showed that using the UNION operator can be more inconvenient than the WHERE clause. But if your filter condition is more complex or you need to retrieve data from multiple tables (not just one), then the UNION operator can greatly simplify the process.

The SQL standard does not limit the number of SELECT statements that can be combined using the UNION operator. However, it is better to refer to the documentation of your DBMS and make sure that it does not impose any restrictions on the maximum allowable number of instructions.

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.12741775

References

1) Бен Форта. Освой самостоятельно SQL за 10 минут, 4-е изд.: Пер. с англ.—М .: ООО “И.Д. Вильямс”, 2014. 288 с.

2) KHAIDAROVA, S. "CREATING SQL QUESTIONS IN RELATIONAL DATABASES." *Ethiopian International Journal of Multidisciplinary Research* 11.02 (2024): 266-269.

3) Хайдарова, Сапияхон. "Создание SQL-запросов в реляционных базах данных". *Вестник РГГУ. Серия: Информатика. Информационная безопасность. Математика* 3 (2020): 8-19.

4) Хайдарова Сапияхон. SQL тили: имкониятлари ва қўлланилиши. Ўқув қўлланма. “Инновация - Зиё” нашриёти, 2020, - 120 бет.

5) Ҳайдарова, С. "APPLICATION OF SQL LANGUAGE IN CLIENT-SERVER TECHNOLOGY." *Экономика и социум* 5-2 (2021): 1097-1101.

6) Khaidarova, S. "Sql-expressions That Manage Transactions." *JournalNX*: 307-310.

7) Pulatov, Sh, and S. Khaidarova. "CREATING SQL-SUB QUERIES IN RELATIONAL DATABASES." *湖南大学学报 (自然科学版)* 50.12 (2023).

8) Хайдарова С.. "СОЗДАНИЕ SQL-ЗАПРОСОВ В РЕЛЯЦИОННЫХ БАЗАХ ДАННЫХ" Экономика и социум, no. 11 (114)-1, 2023, pp. 1078-1082.

9) Хайдарова, Сапияхон. "Методика оптимального построения информационных баз данных для агропромышленных комплексов." (1994).

10) Хайдарова С. "СОЗДАНИЕ SQL-ПОДЗАПРОСОВ В РЕЛЯЦИОННЫХ БАЗАХ ДАННЫХ" Экономика и социум, no. 11 (114)-2, 2023, pp. 977-981.