Subqueries in SQL

Mary, Sanchit, Andrew

DESCRIPTION

Subqueries (also known as inner queries or nested queries) are a tool for performing operations in multiple steps. For example, if you wanted to take the sums of several columns, then average all those values, you’d need to do each aggregation in a distinct step.

A subquery is a query that is nested inside a SELECT, INSERT, UPDATE, or DELETE statement or inside another subquery. A subquery can return a set of rows or just one row to its parent query. A scalar subquery is a query that returns exactly one value: a single row, with a single column. Scalar subqueries can be used in most places in a SQL statement where you could use an expression or a literal value.

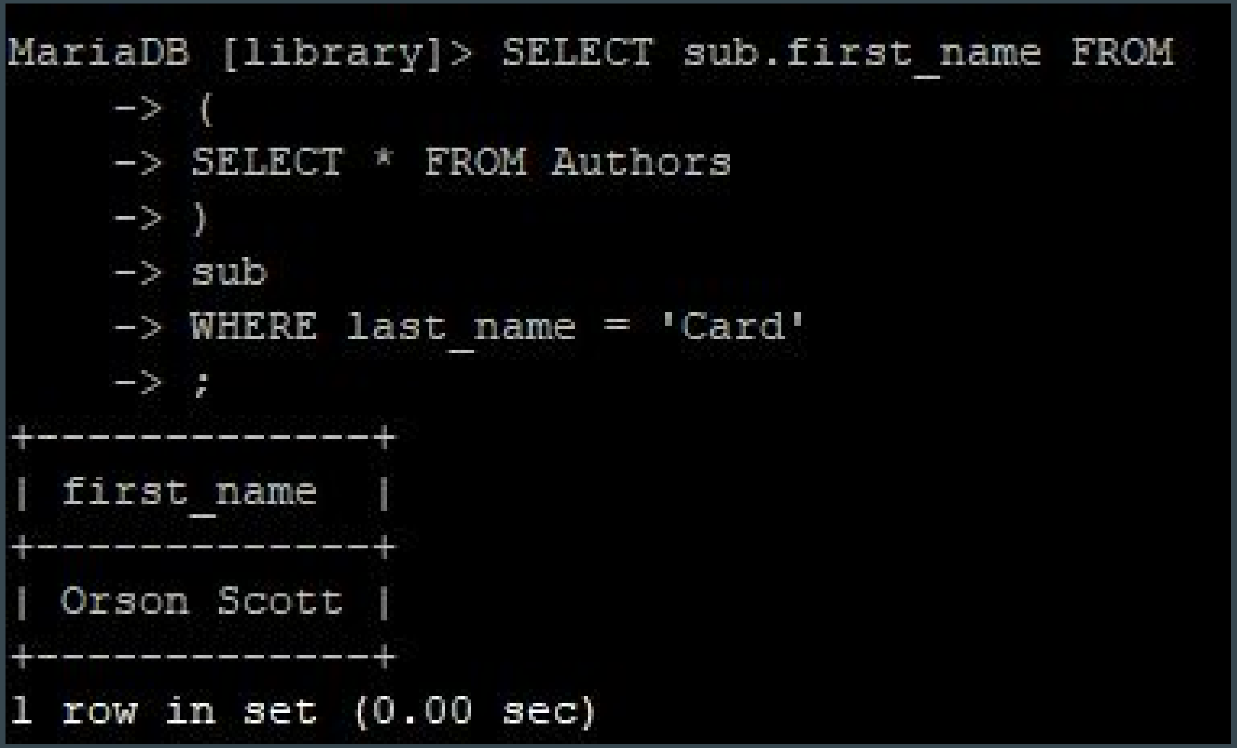
EXAMPLE WITH LIBRARY DB

The subquery creates a result named 'sub' which is the same as that generated by the query:

```
>SELECT * FROM Authors;
```

then the main query is run on those results:

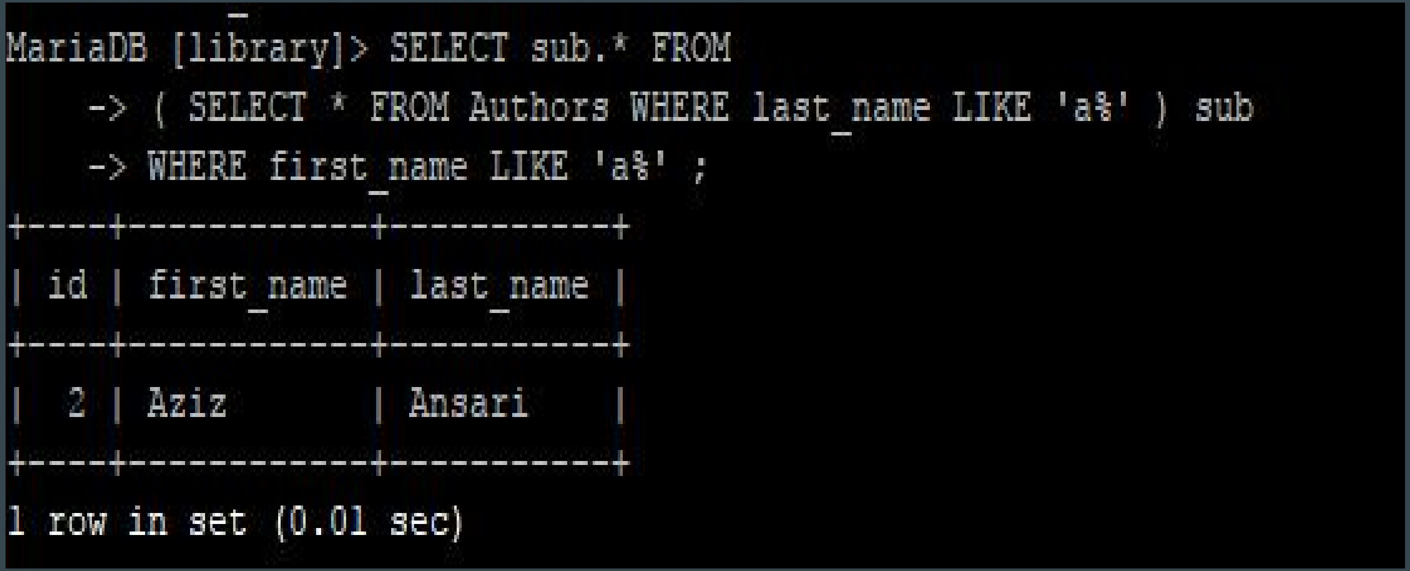
```
>SELECT sub.first_name
>FROM sub WHERE
>last_name = 'Card';
```



```
MariaDB [library]> SELECT sub.first_name FROM
    -> (
    -> SELECT * FROM Authors
    -> )
    -> sub
    -> WHERE last_name = 'Card'
    -> ;
+--------------+
| first_name   |
+--------------+
| Orson Scott  |
+--------------+
1 row in set (0.00 sec)
```

EXAMPLE WITH LIBRARY DB

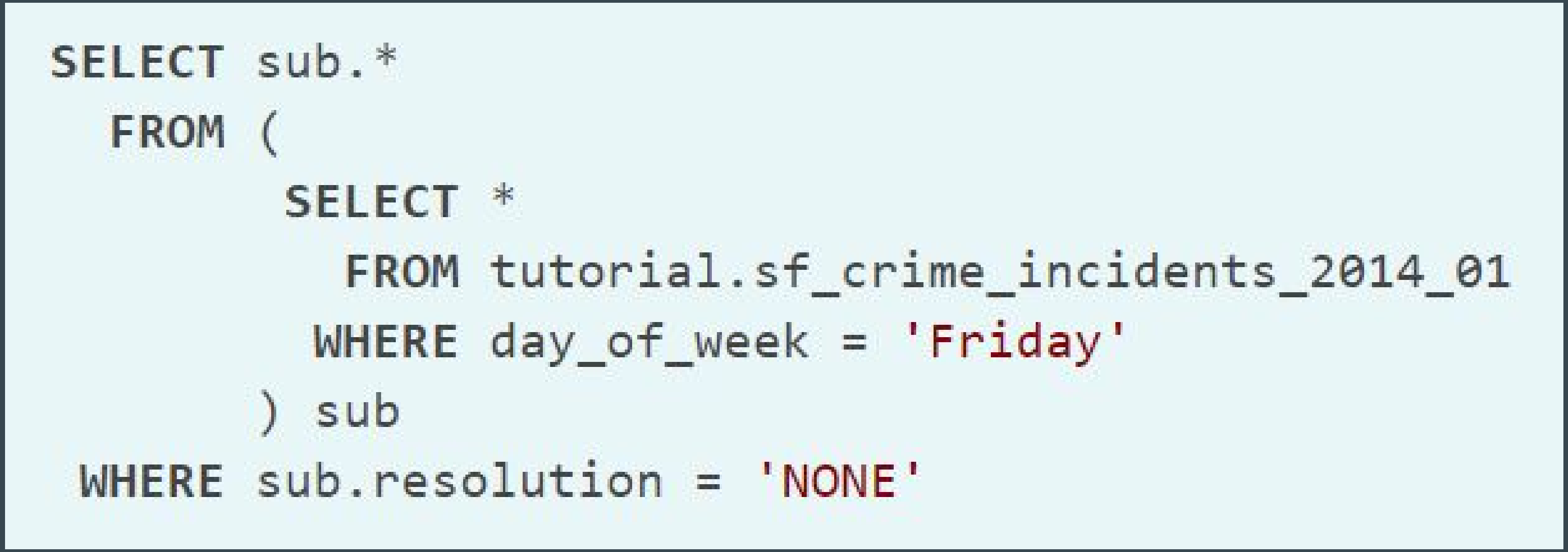
This subquery finds authors with last names beginning with A and the main query finds authors within that set whose first name also begins with A.



```
MariaDB [library]> SELECT sub.* FROM
    -> ( SELECT * FROM Authors WHERE last_name LIKE 'a%' ) sub
    -> WHERE first_name LIKE 'a%' ;
+----+------------+-----------+
| id | first_name | last_name |
+----+------------+-----------+
|  2 | Aziz       | Ansari    |
+----+------------+-----------+
1 row in set (0.01 sec)
```

HOW IT MIGHT BE USED

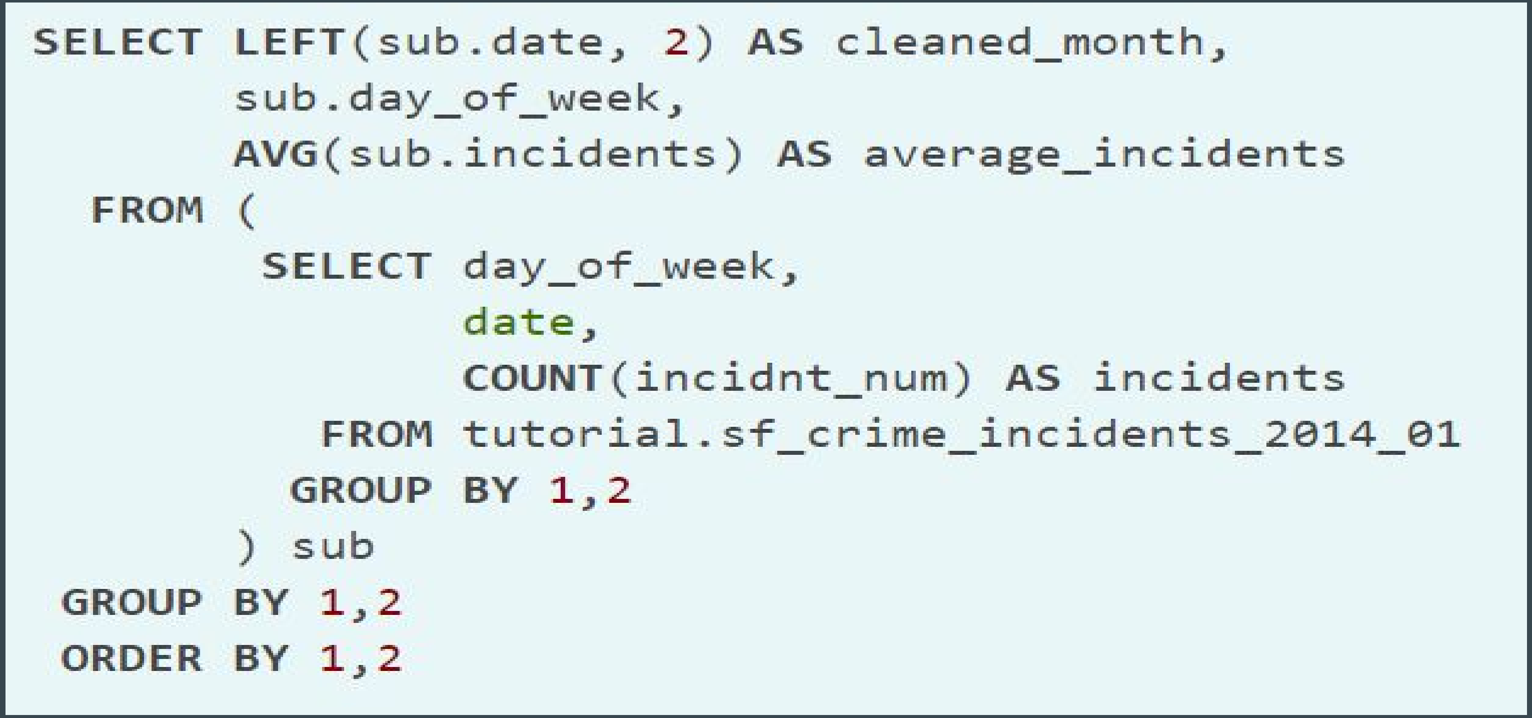
- Can be used in multiple places in a query but easiest to start with the FROM statement:



```
SELECT sub.*
  FROM (
        SELECT *
          FROM tutorial.sf_crime_incidents_2014_01
         WHERE day_of_week = 'Friday'
       ) sub
 WHERE sub.resolution = 'NONE'
```

HOW IT MIGHT BE USED

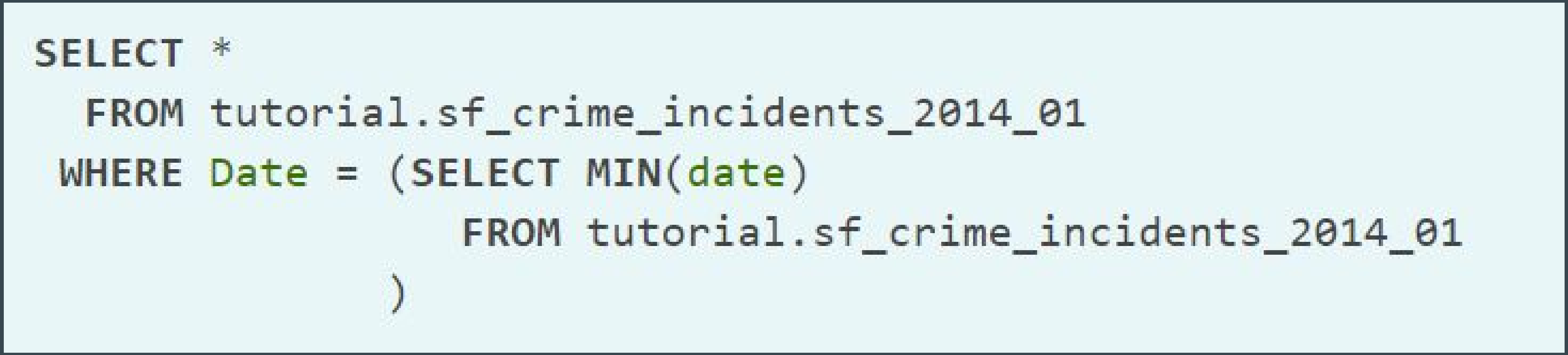
- Can be used to aggregate in multiple stages:



```
SELECT LEFT(sub.date, 2) AS cleaned_month,
       sub.day_of_week,
       AVG(sub.incidents) AS average_incidents
  FROM (
        SELECT day_of_week,
               date,
               COUNT(incidnt_num) AS incidents
          FROM tutorial.sf_crime_incidents_2014_01
         GROUP BY 1,2
       ) sub
 GROUP BY 1,2
 ORDER BY 1,2
```

HOW IT MIGHT BE USED

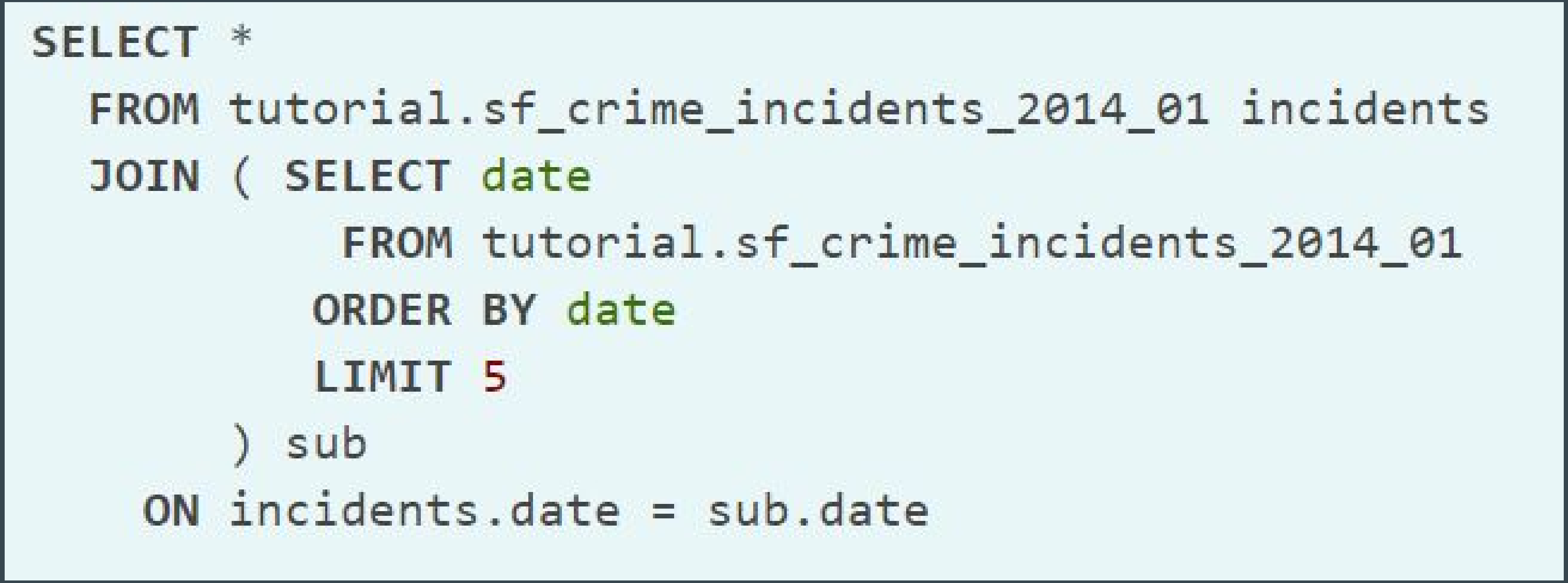
- Can be used in conditional logic:



```
SELECT *
  FROM tutorial.sf_crime_incidents_2014_01
 WHERE Date = (SELECT MIN(date)
                 FROM tutorial.sf_crime_incidents_2014_01
              )
```

HOW IT MIGHT BE USED

- Can be used to join the same table with itself:



```
SELECT *
  FROM tutorial.sf_crime_incidents_2014_01 incidents
  JOIN ( SELECT date
           FROM tutorial.sf_crime_incidents_2014_01
          ORDER BY date
          LIMIT 5
       ) sub
    ON incidents.date = sub.date
```

References

https://community.modeanalytics.com/sql/tutorial/sql-subqueries/

http://searchoracle.techtarget.com/feature/Define-SQL-subqueries