Subqueries

DSC 301: Lecture 10

March 10, 2021

Abstract

Subqueries provide the means to build queries that are difficult or impossible otherwise. We have already seen several instances where subqueries would have "come to the rescue." In particular, "what percentage of flights arrive late?" and "how many destinations has each plane flown?"

Lecture Objectives

- What is a subquery
- Where to places subqueries
- When to use subqueries
- How to use:
  - with the `IN` operator
  - with comparison operators
  - with `ALL, ANY, SOME, EXISTS`
- How to code subqueries in:
  - the `SELECT` clause
  - the `WHERE` clause
  - the `FROM` clause
  - the `HAVING` clause

Subquery Basics

- A `SELECT` statement that's nested within another SQL statement
  - Can be used within: `SELECT,INSERT, UPDATE`, and `DELETE` statements

- Subqueries **cannot** contain `ORDER BY` clause.
- Subqueries must be contained within parentheses.
- Subqueries can return:
  - a single value,
  - a list of values (i.e., single column), or
  - a table of values.
- Subqueries can be nested
- Subqueries can pass aggregated values to the main query
- Subqueries can become complex, use the "beautifier" to split over several lines to increase readability
- Most subqueries can be coded as JOINS and most JOINS can be coded as subqueries

---

Using subqueries comes down to knowing: *when* to use and *where* to place them (depends on what the subquery returns (single value, list of values, or table of values).

Subquery Placement

- In a `WHERE` clause as a search condition
- In a `HAVING` clause as a search condition
- In a `FROM` clause as a "pseudo-table"
- In a `SELECT` clause as a "pseudo-column"

In a `WHERE` clause

Example: Which destination had the most flights in 2014?

A subquery in the `WHERE` clause is very common. For example, find all planes from carriers that fly to ORD (Chicago O'Hara). Note, this doesn't ask to find only planes from carriers that fly to ORD, that would be:

```
SELECT distinct(tailnum) FROM Flights WHERE dest='ORD';
```

Instead we want all planes of the carriers that fly to ORD. This may include planes that never land in Chicago.

All flights whose air time is greater than average of all carriers.

```
 select fid, carrier, dest, air_time from Flights where air_time>=
     all(select
avg(air_time) from Flights group by carrier);
```

```
 select * from Flights where air_time = (select max(air_time) from
     Flights);
```

```
SELECT DISTINCT
    (tailnum)
FROM
    Flights
WHERE
    carrier IN (SELECT
            carrier
        FROM
            Flights
        WHERE
            dest = 'ORD');
```

```
SELECT
    dest, distance
FROM
    Flights
WHERE
    air_time > (SELECT
            AVG(air_time)
        FROM
            Flights);
```

Multiple Table Queries

Although subqueries can (obviously) be used to query single tables, more commonly they are used for multiple table queries. For example, select all customers from West Virginia. Why is a subquery needed? Answer: The state is not stored in the Customers table, only the zip code. Therefore, we need to "look up" the zip codes from West Virginia using a subquery.

```
select * from Products where manufacturer_id in (select
    manufacturer_id from Manufacturers where url<>'');
```

```
SELECT C.id
FROM Customers as C
WHERE C.zip in
```

```
  (
    SELECT zip
    FROM Zipcodes
    WHERE state='WV'
  );
```

```
# Select 100 products that have not yet sold.
SELECT product_id
FROM Products as P
WHERE P.product_id NOT IN
  (
    SELECT OrderDetails.product
    FROM OrderDetails
  )
LIMIT 100;
```

In FROM clause

```
select manufacturer from (select * from Manufacturers where url<>''
    ) as M;
```

```
select round(avg(X.price),2),max(X.msrp) from (select price, msrp,
    inventory
from Products where country='China') as X where inventory>0;
```

The maximum average time of flights by carrier:

```
select max(avg_time) from (select avg(air_time) as avg_time from
    Flights
group by carrier) as X;
```

```
SELECT
    carrier, X.avg_time
FROM
    (SELECT
        carrier, AVG(air_time) AS avg_time
    FROM
        Flights
    GROUP BY carrier) AS X where X.avg_time>200;
```

In SELECT statement

SELECT subqueries are seldom used.

```
# Percentage of flights that arrive late
SELECT
    ROUND(100 * (SELECT
                    COUNT(fid)
                FROM
                    Flights
                WHERE
                    arr_delay > 0) / COUNT(fid),
            0) AS '% Arrive Late'
FROM
    Flights;
```
