WHERE are the subqueries?

DATA MANIPULATION IN SQL

Mona Khalil
Data Scientist, Greenhouse Software

datacamp

What is a subquery?

- A query *nested* inside another query

```
SELECT column
FROM (SELECT column
      FROM table) AS subquery;
```

- Useful for intermediary transformations

What do you do with subqueries?

- Can be in *any* part of a query
  - `SELECT`, `FROM`, `WHERE`, `GROUP BY`
- Can return a variety of information
  - Scalar quantities (`3.14159`, `-2`, `0.001`)
  - A list (`id = (12, 25, 392, 401, 939)`)
  - A table

Why subqueries?

- Comparing groups to summarized values
  - *How did Liverpool compare to the English Premier League's average performance for that year?*
- Reshaping data
  - *What is the highest monthly average of goals scored in the Bundesliga?*
- Combining data that cannot be joined
  - *How do you get both the home and away team names into a table of match results?*

Simple subqueries

- Can be evaluated independently from the outer query

```
SELECT home_goal
FROM match
WHERE home_goal > (
    SELECT AVG(home_goal)
    FROM match);
SELECT AVG(home_goal) FROM match;
```

```
1.560912914784 23
```

Simple subqueries

- Is only processed once in the entire statement

```
SELECT home_goal
FROM match
WHERE home_goal > (
    SELECT AVG(home_goal)
    FROM match);
```

Subqueries in the WHERE clause

- Which matches in the 2012/2013 season scored home goals *higher than overall average*?

```
SELECT AVG(home_goal) FROM match;
```

```
1.56091291478423
```

```
SELECT date, hometeam_id, awayteam_id, home_goal, away_goal
FROM match
WHERE season = '2012/2013'
      AND home_goal > 1.56091291478423;
```

Subqueries in the WHERE clause

- Which matches in the 2012/2013 season scored home goals *higher than overall average*?

```
SELECT date, hometeam_id, awayteam_id, home_goal, away_goal
FROM match
WHERE season = '2012/2013'
      AND home_goal > (SELECT AVG(home_goal)
                       FROM match);
```

date	hometeam_id	awayteam_id	home_goal	away_goal
2012-07-28	9998	1773	5	2
2012-07-29	9987	9984	3	3
2012-10-05	9993	9991	2	2

Subquery filtering list with IN

- Which teams are part of Poland's league?

```
SELECT
  team_long_name,
  team_short_name AS abbr
FROM team
WHERE
  team_api_id IN
  (SELECT hometeam_id
   FROM match
   WHERE country_id = 15722);
```

team_long_name	abbr
Ruch Chorzów	CHO
Jagiellonia	BIA
Lech Poznań	POZ
P. Warszawa	PWA
Cracovia	CKR
Górnik Łęczna	LEC
Polonia Bytom	GOR
Zagłębie Lubin	ZAG
Pogoń Szczecin	POG
Widzew Łódź	WID
Śląsk Wrocław	SLA

Practice time!

DATA MANIPULATION IN SQL

Subqueries in the FROM statement

DATA MANIPULATION IN SQL

Mona Khalil
Data Scientist, Greenhouse Software

datacamp

Subqueries in FROM

- Restructure and transform your data
  - Transforming data from long to wide before selecting
  - Prefiltering data
- Calculating *aggregates of aggregates*
  - *Which 3 teams has the highest average of home goals scored?*
    1. Calculate the `AVG` for each team
    2. Get the 3 highest of the `AVG` values

FROM subqueries...

```
SELECT
  t.team_long_name AS team,
  AVG(m.home_goal) AS home_avg
FROM match AS m
LEFT JOIN team AS t
ON m.hometeam_id = t.team_api_id
WHERE season = '2011/2012'
GROUP BY team;
```

```
| team                 | home_avg          |
|----------------------|-------------------|
| 1. FC Köln           | 1.13725490196078  |
| 1. FC Nürnberg       | 1.27058823529412  |
| 1. FSV Mainz 05      | 1.43697478991597  |
| AC Ajaccio           | 1.12280701754386  |
```

...to main queries!

```
FROM (SELECT
          t.team_long_name AS team,
          AVG(m.home_goal) AS home_avg
      FROM match AS m
      LEFT JOIN team AS t
      ON m.hometeam_id = t.team_api_id
      WHERE season = '2011/2012'
      GROUP BY team)
```

...to main queries!

```
FROM (SELECT
          t.team_long_name AS team,
          AVG(m.home_goal) AS home_avg
      FROM match AS m
      LEFT JOIN team AS t
      ON m.hometeam_id = t.team_api_id
      WHERE season = '2011/2012'
      GROUP BY team) AS subquery
```

...to main queries!

```
SELECT team, home_avg
FROM (SELECT
          t.team_long_name AS team,
          AVG(m.home_goal) AS home_avg
      FROM match AS m
      LEFT JOIN team AS t
      ON m.hometeam_id = t.team_api_id
      WHERE season = '2011/2012'
      GROUP BY team) AS subquery
```

...to main queries!

```
SELECT team, home_avg
FROM (SELECT
          t.team_long_name AS team,
          AVG(m.home_goal) AS home_avg
      FROM match AS m
      LEFT JOIN team AS t
      ON m.hometeam_id = t.team_api_id
      WHERE season = '2011/2012'
      GROUP BY team) AS subquery
ORDER BY home_avg DESC
LIMIT 3;
```

```
| team           | home_avg |
|----------------|----------|
| FC Barcelona   | 3.8421   |
| Real Madrid CF | 3.6842   |
| PSV            | 3.3529   |
```

Things to remember

- You can create multiple subqueries in one `FROM` statement
  - Alias them!
  - Join them!
- You can join a subquery to a table in `FROM`
  - Include a joining columns in both tables!

Let's practice!

DATA MANIPULATION IN SQL

Subqueries in SELECT

DATA MANIPULATION IN SQL

Mona Khalil
Data Scientist, Greenhouse Software

datacamp

SELECTing what?

- Returns a **single value**
  - Include aggregate values to compare to individual values
- Used in mathematical calculations
  - Deviation from the average

Subqueries in SELECT

- Calculate the total matches across all seasons

```
SELECT COUNT(id) FROM match;
```

```
12837
```

Subqueries in SELECT

```
SELECT
  season,
  COUNT(id) AS matches,
  12837 as total_matches
FROM match
GROUP BY season;
```

```
| season    | matches | total_matches |
|-----------|---------|---------------|
| 2011/2012 | 3220    | 12837         |
| 2012/2013 | 3260    | 12837         |
| 2013/2014 | 3032    | 12837         |
| 2014/2015 | 3325    | 12837         |
```

Subqueries in SELECT

```
SELECT
  season,
  COUNT(id) AS matches,
  (SELECT COUNT(id) FROM match) as total_matches
FROM match
GROUP BY season;
```

```
| season    | matches | total_matches |
|-----------|---------|---------------|
| 2011/2012 | 3220    | 12837         |
| 2012/2013 | 3260    | 12837         |
| 2013/2014 | 3032    | 12837         |
| 2014/2015 | 3325    | 12837         |
```

SELECT subqueries for mathematical calculations

```
SELECT AVG(home_goal + away_goal)
FROM match
WHERE season = '2011/2012';
```

```
2.72
```

```
SELECT
  date,
  (home_goal + away_goal) AS goals,
  (home_goal + away_goal) - 2.72 AS diff
FROM match
WHERE season = '2011/2012';
```

Subqueries in SELECT

```
SELECT
  date,
  (home_goal + away_goal) AS goals,
  (home_goal + away_goal) -
     (SELECT AVG(home_goal + away_goal)
      FROM match
      WHERE season = '2011/2012') AS diff
FROM match
WHERE season = '2011/2012';
```

```
| date       | goals | diff               |
|------------|-------|--------------------|
| 2011-07-29 | 3     | 0.28354037267081   |
| 2011-07-30 | 2     | -0.71645962732919  |
| 2011-07-30 | 4     | 1.28354037267081   |
| 2011-07-30 | 1     | -1.71645962732919  |
```

SELECT subqueries -- things to keep in mind

- Need to return a **SINGLE** value
  - Will generate an error otherwise
- Make sure you have all filters in the right places
  - Properly filter **both** the main and the subquery!

```
SELECT
    date,
    (home_goal + away_goal) AS goals,
    (home_goal + away_goal) -
       (SELECT AVG(home_goal + away_goal)
        FROM match
        WHERE season = '2011/2012') AS diff
FROM match
WHERE season = '2011/2012';
```

Let's practice!

DATA MANIPULATION IN SQL

Subqueries everywhere! And best practices!

DATA MANIPULATION IN SQL

Mona Khalil
Data Scientist, Greenhouse Software

datacamp

As many subqueries as you want...

- Can include multiple subqueries in `SELECT`, `FROM`, `WHERE`

```
SELECT
    s.stage,
    ROUND(s.avg_goals,2) AS avg_goal,
    (SELECT AVG(home_goal + away_goal)
     FROM match WHERE season = '2013/2014') AS overall_avg
FROM
    (SELECT
         stage,
         AVG(home_goal + away_goal) AS avg_goals
     FROM match
     WHERE season = '2013/2014'
     GROUP BY stage) AS s
WHERE
    s.avg_goals > (SELECT AVG(home_goal + away_goal)
                   FROM match WHERE season = '2013/2014');
```

Format your queries

- Line up `SELECT`, `FROM`, `WHERE`, and `GROUP BY`

```
SELECT
    col1,
    col2,
    col3
FROM table1
WHERE col1 = 2;
```

Annotate your queries

```
/* This query filters for col1 = 2
and only selects data from table1 */
SELECT
    col1,
    col2,
    col3
FROM table1
WHERE col1 = 2;
```

Annotate your queries

```
SELECT
    col1,
    col2,
    col3
FROM table1 -- this table has 10,000 rows
WHERE col1 = 2; -- Filter WHERE value 2
```

Indent your queries

- Indent your subqueries!

```
SELECT
    col1,
    col2,
    col3
FROM table1
WHERE col1 IN
        (SELECT id
         FROM table2
         WHERE year = 1991);
```

Indent your queries

```
SELECT
  date,
  hometeam_id,
  awayteam_id,
  CASE WHEN hometeam_id = 8455 AND home_goal > away_goal
            THEN 'Chelsea home win'
       WHEN awayteam_id = 8455 AND home_goal < away_goal
            THEN 'Chelsea away win'
       WHEN hometeam_id = 8455 AND home_goal < away_goal
            THEN 'Chelsea home loss'
       WHEN awayteam_id = 8455 AND home_goal > away_goal
            THEN 'Chelsea away loss'
       WHEN (hometeam_id = 8455 OR awayteam_id = 8455)
            AND home_goal = away_goal THEN 'Chelsea Tie'
       END AS outcome
FROM match
WHERE hometeam_id = 8455 OR awayteam_id = 8455;
```

Holywell's SQL Style Guide

Is that subquery necessary?

- Subqueries require computing power
  - How big is your database?
  - How big is the table you're querying from?
- Is the subquery *actually* necessary?

Properly filter each subquery!

- Watch your filters!

```
SELECT
    s.stage,
    ROUND(s.avg_goals,2) AS avg_goal,
    (SELECT AVG(home_goal + away_goal)
     FROM match WHERE season = '2013/2014') AS overall_avg
FROM
    (SELECT
         stage,
         AVG(home_goal + away_goal) AS avg_goals
     FROM match
     WHERE season = '2013/2014'
     GROUP BY stage) AS s
WHERE
    s.avg_goals > (SELECT AVG(home_goal + away_goal)
                   FROM match WHERE season = '2013/2014');
```

Let's practice!

DATA MANIPULATION IN SQL