SQL Nested Queries: Comprehensive Review

DSC 208R - Data Management for Analytics

June 2025

Introduction to Nested Queries

A nested query (or subquery) is a query within another SQL query. Subqueries can be used in the 'SELECT', 'FROM', or 'WHERE' clauses.

Schemas Used in Examples

- **Movie** (name, year, genre)
- **ActedIN** (actorname, moviename)
- **Actor** (name, age)

Data for Examples

Movie Table:

Name	Year	Genre
Apocalypse Now	1979	War
The God Father	1972	Crime
Planet Earth II	2016	Nature Documentary

ActedIN Table:

Actorname	Moviename
Marlon Brando	Apocalypse Now
Al Pacino	The God Father
Marlon Brando	The God Father

Actor Table:

Name	Age
Marlon Brando	80
Al Pacino	82
De Niro	79

Types of Nested Queries

'IN' / 'NOT IN'

These operators check if a value is present (or not present) within the result set of a subquery.

Query: Find all actors who acted in at least one movie.

```
SELECT name
FROM Actor
WHERE name IN (SELECT actorname FROM ActedIN);
```

Result:

name
Marlon Brando
Al Pacino

Explanation: The subquery 'SELECT actorname FROM ActedIN' returns a list of actor names who have acted in movies. The outer query then selects names from the 'Actor' table that are present in this list.

‘EXISTS‘ / ‘NOT EXISTS‘

These operators check for the existence of rows returned by a subquery. They return true if the subquery returns any rows, and false otherwise. The subquery typically refers to attributes from the outer query (correlated subquery).

Query: Find all actors who acted in at least one movie. (Equivalent to ‘IN‘ example)

```
SELECT name
FROM Actor AS T1
WHERE EXISTS (SELECT * FROM ActedIN AS T2 WHERE T1.name = T2.actorname);
```

Result:

name
Marlon Brando
Al Pacino

Explanation: For each actor in ‘Actor‘ (aliased as ‘T1‘), the subquery checks if there exists any record in ‘ActedIN‘ (aliased as ‘T2‘) where the actor’s name matches. If such a record exists, the outer query includes the actor’s name.

‘UNIQUE‘

The ‘UNIQUE‘ operator returns true if a subquery produces no duplicate rows.

Query: Find movies that have only one actor listed.

```
SELECT name
FROM Movie
WHERE UNIQUE (SELECT moviename FROM ActedIN WHERE moviename = Movie.name);
```

Explanation: This query would look for movies where the subquery ‘SELECT moviename FROM ActedIN WHERE moviename = Movie.name‘ (for each movie) returns only one instance of that ‘moviename‘, implying only one actor acted in it. Based on the provided ‘ActedIN‘ table, ’Apocalypse Now’ has one entry, and ’The God Father’ has two (Marlon Brando, Al Pacino).

‘ALL‘

The ‘ALL‘ operator is used with comparison operators (‘¿‘, ‘¡‘, ‘=‘, etc.) and returns true if the comparison is true for all values in the result set of the subquery.

Query: Find actors who are older than all actors who acted in ’The God Father’.

```
SELECT name
FROM Actor
WHERE age > ALL (SELECT T1.age FROM Actor AS T1, ActedIN AS T2 WHERE T1.name = T2.actorname AND
    T2.moviename = 'The God Father');
```

Explanation: The subquery finds the ages of all actors in ’The God Father’ (Al Pacino: 82, Marlon Brando: 80). The outer query then selects actors whose age is greater than *all* of these ages (i.e., greater than 82). In this dataset, no actor is older than 82.

‘ANY‘ / ‘SOME‘

The ‘ANY‘ (or ‘SOME‘) operator is used with comparison operators (‘¿‘, ‘¡‘, ‘=‘, etc.) and returns true if the comparison is true for at least one value in the result set of the subquery.

Query: Find actors who are older than at least one actor who acted in ’The God Father’.

```
SELECT name
FROM Actor
WHERE age > ANY (SELECT T1.age FROM Actor AS T1, ActedIN AS T2 WHERE T1.name = T2.actorname AND
    T2.moviename = 'The God Father');
```

Explanation: The subquery returns ages 82 (Al Pacino) and 80 (Marlon Brando). The outer query selects actors whose age is greater than *any* of these ages.

- De Niro (79) is not ‘¿ ANY‘ (82, 80).
- Marlon Brando (80) is not ‘¿ ANY‘ (82, 80).
- Al Pacino (82) is not ‘¿ ANY‘ (82, 80).

In this specific dataset, no actor is strictly ‘¿ ANY‘ of (82, 80) if they are themselves one of those actors. For example, Al Pacino (82) is not greater than 82, and 80 is not greater than 82. If there was an actor aged 85, they would be included.

Correlated vs. Non-Correlated Subqueries

- **Non-correlated subquery:** A subquery that can be executed independently of the outer query. It typically runs once and provides its result set to the outer query. Examples include the ‘IN‘ clause used previously.
- **Correlated subquery:** A subquery that depends on the outer query for its values and therefore executes once for each row processed by the outer query. ‘EXISTS‘, ‘NOT EXISTS‘, ‘UNIQUE‘, ‘ALL‘, and ‘ANY‘ typically involve correlated subqueries.