# Subquery inside where

Begin by calculating the average life expectancy across all countries for 2015.

select avg(life\_expectancy)

from populations

where year = 2015;

Recall that you can use SQL to do calculations for you. Suppose we wanted only records that were above 1.15 \* 100 in terms of life expectancy for 2015:

SELECT \*FROM populationsWHERE life\_expectancy > 1.15 \* 100 AND year = 2015;

Select all fields from populations with records corresponding to larger than 1.15 times the average you calculated in the first task for 2015. In other words, change the 100 in the example above with a subquery.

select \*

from populations

where life\_expectancy >

1.15 \* (select avg(life\_expectancy)

from populations

where year = 2015) and

year = 2015;

# Subquery inside where (2)

* Make use of the capital field in the countries table in your subquery.
* Select the city name, country code, and urban area population fields.

-- select the appropriate fields

select name, country\_code, urbanarea\_pop

-- from the cities table

from cities

-- with city name in the field of capital cities

where name IN

(select capital

from countries)

ORDER BY urbanarea\_pop DESC;

# Subquery inside select

Just **Submit Answer** here!

SELECT countries.name AS country, COUNT(\*) AS cities\_num

FROM cities

INNER JOIN countries

ON countries.code = cities.country\_code

GROUP BY country

ORDER BY cities\_num DESC, country

LIMIT 9;

* Remove the comments around the second query and comment out the first query instead.
* Convert the GROUP BY code to use a subquery inside of SELECT, i.e. fill in the blanks to get a result that matches the one given using the GROUP BY code in the first query.
* Again, sort the result by cities\_num descending and then by country ascending.

SELECT countries.name AS country,

(SELECT count(\*)

FROM cities

WHERE countries.code = cities.country\_code) AS cities\_num

FROM countries

ORDER BY cities\_num desc, country

LIMIT 9;

# Subquery inside from

* Begin by determining for each country code how many languages are listed in the languages table using SELECT, FROM, and GROUP BY.
* Alias the aggregated field as lang\_num.

select code, count(name) as lang\_num

from languages

group by code;

* Include the previous query (aliased as subquery) as a subquery in the FROM clause of a new query.
* Select the local name of the country from countries.
* Also, select lang\_num from subquery.
* Make sure to use WHERE appropriately to match code in countries and in subquery.
* Sort by lang\_num in descending order.

select local\_name, subquery.lang\_num

from countries,

(select code, count(\*) as lang\_num

from languages

group by code)as subquery

where countries.code = subquery.code

order by lang\_num desc;

# Advanced subquery

* Create an inner join with countries on the left and economies on the right with USING. Do not alias your tables or columns.
* Retrieve the country name, continent, and inflation rate for 2015.

select countries.name, countries.continent, economies.inflation\_rate

from countries

inner join economies

using(code)

where economies.year = 2015;

* Determine the maximum inflation rate for each continent in 2015 using the previous query as a subquery called subquery in the FROM clause.
* Select the maximum inflation rate AS max\_inf grouped by continent.

This will result in the six maximum inflation rates in 2015 for the six continents as one field table. (Don't include continent in the outer SELECT statement.)

SELECT MAX(inflation\_rate) AS max\_inf

FROM (

SELECT name, continent, inflation\_rate

FROM countries

INNER JOIN economies

USING (code)

WHERE year = 2015) AS subquery

GROUP BY continent;

* Append the second part's query to the first part's query using WHERE, AND, and IN to obtain the name of the country, its continent, and the maximum inflation rate for each continent in 2015. Revisit the sample output in the assignment text at the beginning of the exercise to see how this matches up.
* For the sake of practice, change all joining conditions to use ONinstead of USING.

This code works since each of the six maximum inflation rate values occur only once in the 2015 data. Think about whether this particular code involving subqueries would work in cases where there are ties for the maximum inflation rate values.

SELECT name, continent, inflation\_rate

FROM countries

INNER JOIN economies

ON countries.code = economies.code

WHERE year = 2015

AND inflation\_rate IN (

SELECT MAX(inflation\_rate) AS max\_inf

FROM (

SELECT name, continent, inflation\_rate

FROM countries

INNER JOIN economies

ON countries.code = economies.code

WHERE year = 2015) AS subquery

GROUP BY continent);

# Subquery challenge

* Select the country code, inflation rate, and unemployment rate.
* Order by inflation rate ascending.
* Do not use table aliasing in this exercise.

SELECT code, inflation\_rate, unemployment\_rate

FROM economies

WHERE year = 2015 AND code not in

(SELECT code

FROM countries

WHERE (gov\_form = 'Constitutional Monarchy' OR gov\_form LIKE '%Republic%'))

ORDER BY inflation\_rate;

# Subquery review

Ans ==> WHERE

# Final challenge

* Select unique country names. Also select the total investment and imports fields.
* Use a left join with countries on the left. (An inner join would also work, but please use a left join here.)
* Match on code in the two tables AND use a subquery inside of ONto choose the appropriate languages records.
* Order by country name ascending.
* Use table aliasing but **not** field aliasing in this exercise.

SELECT DISTINCT name, total\_investment, imports

FROM countries AS c

LEFT JOIN economies AS e

ON (c.code = e.code

AND c.code IN (

SELECT l.code

FROM languages AS l

WHERE official = 'true'

) )

WHERE region = 'Central America' AND year = 2015

ORDER BY name;

# Final challenge (2)

* Include the name of region, its continent, and average fertility rate aliased as avg\_fert\_rate.
* Sort based on avg\_fert\_rate ascending.
* Remember that you'll need to GROUP BY all fields that aren't included in the aggregate function of SELECT.

-- choose fields

SELECT region, continent, AVG(fertility\_rate) AS avg\_fert\_rate

-- left table

FROM countries AS c

-- right table

INNER JOIN populations AS p

-- join conditions

ON c.code = p.country\_code

-- specific records matching a condition

WHERE year = 2015

-- aggregated for each what?

GROUP BY region, continent

-- how should we sort?

ORDER BY avg\_fert\_rate;

# Final challenge (3)

* Select the city name, country code, city proper population, and metro area population.
* Calculate the percentage of metro area population composed of city proper population for each city in cities, aliased as city\_perc.
* Focus only on capital cities in Europe and the Americas in a subquery.
* Make sure to exclude records with missing data on metro area population.
* Order the result by city\_perc descending.
* Then determine the top 10 capital cities in Europe and the Americas in terms of this city\_perc percentage.
* Do not use table aliasing in this exercise.

SELECT name, country\_code, city\_proper\_pop, metroarea\_pop,

city\_proper\_pop / metroarea\_pop \* 100 AS city\_perc

FROM cities

WHERE name IN

(SELECT capital

FROM countries

WHERE (continent = 'Europe'

OR continent LIKE '%America'))

AND metroarea\_pop IS NOT NULL

ORDER BY city\_perc DESC

LIMIT 10;