# Left Join

Fill in the code shown to complete the inner join. Note how many records are in the result of the join in the **query result** tab.

-- get the city name (and alias it), the country code,

-- the country name (and alias it), the region,

-- and the city proper population

SELECT c1.name AS city, code, c2.name AS country,

region, city\_proper\_pop

-- specify left table

FROM cities AS c1

-- specify right table and type of join

INNER JOIN countries AS c2

-- how should the tables be matched?

ON c1.country\_code = c2.code

-- sort based on descending country code

ORDER BY code desc;

Change the code to perform a LEFT JOIN instead of an INNER JOIN. After executing this query, note how many records the query result contains.

-- get the city name (and alias it), the country code,

-- the country name (and alias it), the region,

-- and the city proper population

SELECT c1.name AS city, code, c2.name AS country,

region, city\_proper\_pop

-- specify left table

FROM cities AS c1

-- specify right table and type of join

left JOIN countries AS c2

-- how should the tables be matched?

ON c1.country\_code = c2.code

-- sort based on descending country code

ORDER BY code desc;

# Left join (2)

Perform an inner join. Alias the name of the country field as country and the name of the language field as language. Sort based on descending country name.

/\*

select country name AS country, the country's local name,

the language name AS language, and

the percent of the language spoken in the country

\*/

select c.name AS country, local\_name, l.name AS language, percent

-- countries on the left (alias as c)

FROM countries AS c

-- appropriate join with languages (as l) on the right

inner JOIN languages AS l

-- give fields to match on

ON c.code = l.code

-- sort by descending country name

ORDER BY country desc;

Perform a left join instead of an inner join. Observe the result, and also note the change in the number of records in the result. Carefully review which records appear in the left join result, but not in the inner join result.

/\*

select country name AS country, the country's local name,

the language name AS language, and

the percent of the language spoken in the country

\*/

select c.name AS country, local\_name, l.name AS language, percent

-- countries on the left (alias as c)

FROM countries AS c

-- appropriate join with languages (as l) on the right

left JOIN languages AS l

-- give fields to match on

ON c.code = l.code

-- sort by descending country name

ORDER BY country desc;

# Left join (3)

* Begin with a left join with the countries table on the left and the economies table on the right.
* Focus only on records with 2010 as the year.

-- select name, region, and gdp\_percapita

SELECT name, region, gdp\_percapita

-- from countries (alias c) on the left

FROM countries AS c

-- left join with economies (alias e)

LEFT JOIN economies AS e

-- match on code fields

ON c.code = e.code

-- focus on 2010 entries

WHERE e.year = 2010;

Modify your code to calculate the average GDP per capita AS avg\_gdp for **each region** in 2010. Select the region and avg\_gdp fields.

-- Select region, average gdp\_percapita (alias avg\_gdp)

SELECT region, avg(gdp\_percapita) as avg\_gdp

-- From countries (alias c) on the left

FROM countries AS c

-- Join with economies (alias e)

LEFT JOIN economies AS e

-- Match on code fields

ON c.code = e.code

-- Focus on 2010

WHERE e.year = 2010

-- Group by region

GROUP BY region;

Arrange this data on average GDP per capita for each region in 2010 from highest to lowest average GDP per capita.

-- Select region, average gdp\_percapita (alias avg\_gdp)

SELECT region, avg(gdp\_percapita) as avg\_gdp

-- From countries (alias c) on the left

FROM countries AS c

-- Join with economies (alias e)

LEFT JOIN economies AS e

-- Match on code fields

ON c.code = e.code

-- Focus on 2010

WHERE e.year = 2010

-- Group by region

GROUP BY region

-- Order by avg\_gdp, descending

ORDER BY avg\_gdp desc;

The left join code is commented out here. Your task is to write a new query using rights joins that produces the same result as what the query using left joins produces. Keep this left joins code commented as you write your own query just below it using right joins to solve the problem.

Note the order of the joins matters in your conversion to using right joins!

-- convert this code to use RIGHT JOINs instead of LEFT JOINs

/\*

SELECT cities.name AS city, urbanarea\_pop, countries.name AS country,

indep\_year, languages.name AS language, percent

FROM cities

LEFT JOIN countries

ON cities.country\_code = countries.code

LEFT JOIN languages

ON countries.code = languages.code

ORDER BY city, language;

\*/

SELECT cities.name AS city, urbanarea\_pop, countries.name AS country,

indep\_year, languages.name AS language, percent

FROM languages

RIGHT JOIN countries

ON languages.code = countries.code

RIGHT JOIN cities

ON countries.code = cities.country\_code

ORDER BY city, language;

# Full join

Choose records in which region corresponds to North America or is NULL.

SELECT name AS country, code, region, basic\_unit

FROM countries

FULL JOIN currencies

USING (code)

WHERE region = 'North America' OR region IS NULL

ORDER BY region;

Repeat the same query as above but use a LEFT JOIN instead of a FULL JOIN. Note what has changed compared to the FULL JOINresult!

SELECT name AS country, code, region, basic\_unit

FROM countries

left JOIN currencies

USING (code)

WHERE region = 'North America' OR region IS NULL

ORDER BY region;

Repeat the same query as above but use an INNER JOIN instead of a FULL JOIN. Note what has changed compared to the FULL JOINand LEFT JOIN results!

SELECT name AS country, code, region, basic\_unit

FROM countries

inner JOIN currencies

USING (code)

WHERE region = 'North America' OR region IS NULL

ORDER BY region;

# Full join (2)

* Choose records in which countries.name starts with the capital letter 'V' or is NULL.
* Arrange by countries.name in ascending order to more clearly see the results.

SELECT countries.name, code, languages.name AS language

FROM languages

FULL JOIN countries

USING (code)

WHERE countries.name LIKE 'V%' OR countries.name IS NULL

ORDER BY countries.name;

Repeat the same query as above but use a left join instead of a full join. Note what has changed compared to the full join result!

SELECT countries.name, code, languages.name AS language

FROM languages

LEFT JOIN countries

USING (code)

WHERE countries.name LIKE 'V%' OR countries.name IS NULL

ORDER BY countries.name;

Repeat once more, but use an inner join instead of a left join. Note what has changed compared to the full join and left join results.

SELECT countries.name, code, languages.name AS language

FROM languages

INNER JOIN countries

USING (code)

WHERE countries.name LIKE 'V%' OR countries.name IS NULL

ORDER BY countries.name;

# Full join (3)

* Complete a full join with countries on the left and languages on the right.
* Next, full join this result with currencies on the right.
* Select the fields corresponding to the country name AS country, region, language name AS language, and basic and fractional units of currency.
* Use LIKE to choose the Melanesia and Micronesia regions (Hint:'M%esia').

SELECT c.name AS country, c.region, l.name AS language,

cu.basic\_unit, cu.frac\_unit

FROM countries AS c

FULL JOIN languages AS l

USING (code)

FULL JOIN currencies AS cu

USING (code)

WHERE c.region LIKE 'M%esia';

# A table of two cities

* Create the cross join above and select only the city name AS city and language name AS language. (Recall that cross joins do **not** use ON or USING.)
* Make use of LIKE and Hyder% to choose Hyderabad in both countries.

SELECT c.name AS city, l.name AS language

FROM cities AS c

cross JOIN languages AS l

WHERE c.name LIKE 'Hyder%';

Use an inner join instead of a cross join. Think about what the difference will be in the results for this inner join result and the one for the cross join.

SELECT c.name AS city, l.name AS language

FROM cities AS c

inner JOIN languages AS l

on c.country\_code = l.code

WHERE c.name LIKE 'Hyder%';

# Outer challenge

* Select country name AS country, region, and life expectancy AS life\_exp.
* Make sure to use LEFT JOIN, WHERE, ORDER BY, and LIMIT.

select c.name as country, c.region, p.life\_expectancy as life\_exp

from countries as c

left join populations as p

on c.code = p.country\_code

where year = 2010

order by life\_exp

limit 5;