

Simples sobre una sola tabla

- 1) select population from country where name = 'Argentina'
- 2) select distinct continent from country
- 3) select name from country where region ='South America' and population >= 15000000
- 4) select name, gnp as PBI from country order by PBI desc limit 10
- 5) select governmentform as formaGobierno, count(governmentform) as cantidad from country  
group by formaGobierno  
order by cantidad desc
- 6) select continent, sum(surfacearea) as superficie from country group by continent order by  
superficie desc  
select continent, sum(surfacearea)/1000000 as superficieMilloneskm2 from country group by  
continent  
order by superficieMilloneskm2 desc
- 7) select continent, count(continent) as cantidadpaíses from country group by continent having  
count(continent) > 15  
order by cantidadpaíses desc
- 8)select continent, count(continent) as cantidadpaíses from country where population >  
20000000  
group by continent having count(continent) > 15 order by cantidadpaíses desc

Subqueries

- 1) Primero se ejecuta la sub-consulta, que nos devuelve la minima esperanza de vida de la tabla países. Luego se ejecuta la consulta principal que devuelve el nombre y la esperanza de vida de aquel/aquellos países que tengan la esperanza de vida devuelta por la subconsulta
- 2)select name, lifeexpentancy from country  
where lifeexpectancy = (select min(lifeexpentancy) from country)  
or lifeexpectancy = (select max(lifeexpentancy) from country)
- 3)select name, indepyear from country  
where continent = (select continent from country where indepyear = (select min(indepyear) from country))
- 4)select distinct continent from country  
where continent not in(select continent from country group by continent order by sum(gnp) asc limit 1)}

Joins

- 1)select c.name, cl.language from country c inner join countrylanguage cl  
on c.code = cl.countrycode where c.continent = 'Oceania'

```
select c.name, cl.language from country c, countrylanguage cl
where c.code = cl.countrycode c.continent = 'Oceania'
```

```
2)select c.name, count(cl.countrycode) as cantidadlenguas from country c inner join
countrylanguage cl
on c.code = cl.countrycode group by c.name, cl.countrycode having count(cl.countrycode) > 1
order by cantidadlenguas desc
```

```
3) select distinct cl.language from country c inner join countrylanguage cl
on c.code = cl.countrycode
where c.continent = (select continent from country where continent != 'Antarctica' group by
continent order by sum(gnp) asc limit 1)
```

4) Los nombres de los países y sus respectivas poblaciones calculada de acuerdo al campo de la tabla country:

```
select c.name, c.population as poblacion_Segun_Tabla_Country,
sum(ci.population) as poblacion_Segun_La_Suma_De_Ciudades ,
(sum (ci.population))*100/c.population as porcentajePoblacionUrbana
from country c inner join city ci on c.code = ci.countrycode
group by c.name,poblacion_Segun_Tabla_Country order by porcentajePoblacionUrbana desc
```

Estos países dan mal porque están mal los datos, tienen mas poblacion en las ciudades que la poblacion total

Singapore SGP sumaDeCiudades = 4017733 poblacionTotal = 3567000  
Cocos Islands CCK = 670 poblacionTotal = 600  
Gibraltar GIB = 27025 poblacionTotal = 25000

### EJERCICIO 3

```
SELECT C.CODE, COUNT(CL.LANGUAGE), sum(ci.population) as poblacionSumaCiudades
FROM COUNTRY C INNER JOIN countrylanguage CL ON C.CODE = CL.countrycode
inner JOIN CITY CI ON CI.countrycode = C.CODE
group by c.code
order by c.code
```

```
CREATE TABLE stats (
  countrycode character(3) NOT NULL,
  cant_lenguas integer NOT NULL,
  pop_urbana integer NOT NULL
);
```

```
INSERT INTO STATS
(select c.code, 0 , sum(ci.population) as poblacionUrbana
from country c, city ci
where c.code = ci.countrycode
group by c.code
order by c.code)
```

```
UPDATE STATS
SET CANT_LENGUAS = SUBQUERY.CANT_LENGUAS
FROM
(select count(cl.language) AS CANT_LENGUAS, C.CODE
from country c, countrylanguage cl
where c.code = cl.countrycode
group by c.code
order by c.code) AS SUBQUERY
WHERE STATS.countrycode = SUBQUERY.CODE
```

```
ALTER TABLE ONLY stats
  ADD CONSTRAINT stats_pkey PRIMARY KEY (countrycode);
```

```
ALTER TABLE ONLY stats
  ADD CONSTRAINT stats_fkey FOREIGN KEY (countrycode) REFERENCES country(code);
```

```
CREATE TABLE SITIO(
    ID INTEGER PRIMARY KEY,
    ENTIDAD VARCHAR NOT NULL,
    TIPO_ENTIDAD VARCHAR NOT NULL,
    PAIS VARCHAR NOT NULL,
    COUNTRYCODE character(3) NOT NULL
);
```


```
ALTER TABLE ONLY SITIO
    ADD CONSTRAINT sitio_fkey FOREIGN KEY (COUNTRYCODE) REFERENCES
country(code);
```

```
UPDATE country SET code2='UK' where code = 'GBR'
```

```
Ejercicio 5)1)
select *
from sitio s1 , sitio s2
where s1.countrycode = s2.countrycode
and s1.entidad like 'a%' and s2.entidad like 'b%'
limit 100
```

La siguiente query consulta sobre dos instancias de la tabla sitios. Como hay un asterisco mostrará los resultados de las instancias de forma apaisada, en primer lugar busca todas las entidades que arranquen con 'a' y en la segunda instancia todas aquellas que arrancan con 'b'. Limitando en 100 los resultados, pero no van a ser solamente 100 resultados ya que al mostrar la información de forma apaisada en total serían 200 resultados.


5)2):

	QUERY PLAN	
	text	
1	Limit (cost=1000.00..1003.50 rows=100 width=50) (actual time=0.360..11.669 rows=100 loops=1)	
2	-> Nested Loop (cost=1000.00..55077007.31 rows=1571633259 width=50) (actual time=0.359..11.651 rows=100 loops=1)	
3	Join Filter: (s1.countrycode = s2.countrycode)	
4	Rows Removed by Join Filter: 3614	
5	-> Seq Scan on sitio s1 (cost=0.00..19474.00 rows=60574 width=25) (actual time=0.016..0.016 rows=1 loops=1)	
6	Filter: ((entidad)::text ~~ 'a%':text)	
7	Rows Removed by Filter: 3	
8	-> Materialize (cost=1000.00..19542.60 rows=60574 width=25) (actual time=0.237..11.032 rows=3714 loops=1)	
9	-> Gather (cost=1000.00..19239.73 rows=60574 width=25) (actual time=0.235..9.806 rows=3714 loops=1)	
10	Workers Planned: 2	
11	Workers Launched: 2	
12	-> Parallel Seq Scan on sitio s2 (cost=0.00..12182.33 rows=25239 width=25) (actual time=0.031..4.899 rows=12...)	
13	Filter: ((entidad)::text ~~ 'b%':text)	
14	Rows Removed by Filter: 22293	
15	Planning Time: 0.283 ms	
16	Execution Time: 11.860 ms	

5)3)

```
CREATE INDEX countrycode_idx ON sitio(countrycode);
```

5)4):

	Data Output	Explain	Messages	Notifications
	<b>QUERY PLAN</b> text 			
1	Limit (cost=0.42..0.88 rows=100 width=50) (actual time=0.072..0.937 rows=100 loops=1)			
2	-> Nested Loop (cost=0.42..7136135.03 rows=1571633259 width=50) (actual time=0.069..0.926 rows=100 loops=1)			
3	-> Seq Scan on sitio s1 (cost=0.00..19474.00 rows=60574 width=25) (actual time=0.029..0.030 rows=1 loops=1)			
4	Filter: ((entidad)::text ~~ 'a%':text)			
5	Rows Removed by Filter: 15			
6	-> Index Scan using countrycode_idx on sitio s2 (cost=0.42..114.07 rows=342 width=25) (actual time=0.036..0.872 row...			
7	Index Cond: (countrycode = s1.countrycode)			
8	Filter: ((entidad)::text ~~ 'b%':text)			
9	Rows Removed by Filter: 1399			
10	Planning Time: 0.403 ms			
11	Execution Time: 0.973 ms			

Al crear el índice se logra obtener los datos de forma más rápida sin necesidad de recorrer todas las filas. Tal como podemos ver en las imágenes, en la primera de ellas se puede ver que recorre 60.000 filas aproximadamente, ya que realiza una búsqueda secuencial, mientras que después de agregar el índice recorre 342 filas ya que realiza la recorrida de forma indexada. Por otro lado, antes del índice podemos ver que el “Planning time” es menor que después de agregar el índice (la mitad aproximadamente), esto se debe a que le cuesta menos planificar la consulta.

Por último, la gran diferencia se puede observar en el “Execution time” que después de agregar el índice es casi 12 veces menor que antes de agregarlo, esto se debe a que al recorrer la tabla a través del índice encuentran los resultados mucho más rápido bajando notablemente el tiempo de ejecución.

## Ejercicio 6

```
select name,population from country order by name
```

```
select name,gnp from country order by name
```

```
select count(*),c.name from sitio s inner join country c on s.countrycode = c.code group by c.name
```

