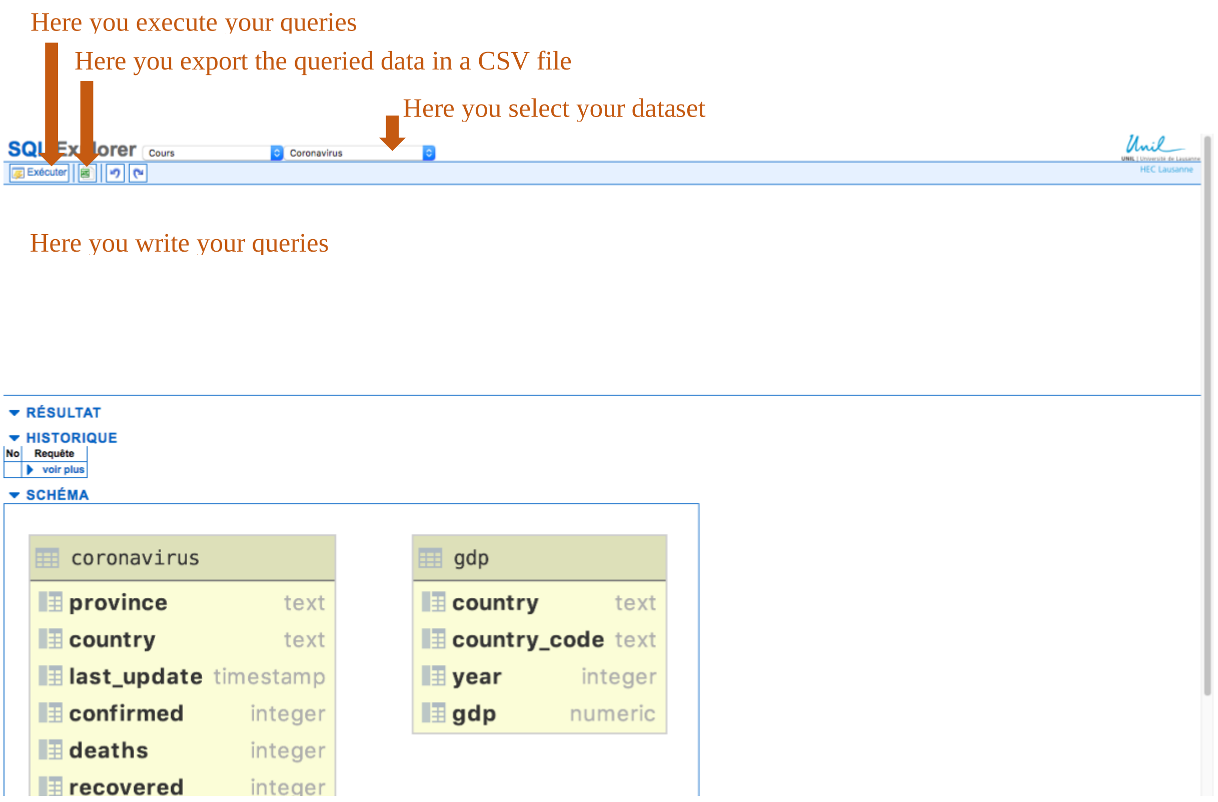
WEEK 5 - SQL Exercises

1) Walkthrough

<u>What is SQL Explorer?</u>
SQL Explorer is a reliable and effective tool used to execute and create SQL queries and manage multiple database connections and provides a simple and intuitive interface.

Access: https://hec.unil.ch/info1ere/sqlexplorer



Here you execute your queries
Here you export the queried data in a CSV file
Here you select your dataset
SQL Explorer
Cours
Coronavirus
Exécuter
Unil
HEC Lausanne
Here you write your queries
RÉSULTAT
HISTORIQUE
No
Requête
voir plus
SCHÉMA

coronavirus	
province	text
country	text
last_update	timestamp
confirmed	integer
deaths	integer
recovered	integer

gdp	
country	text
country_code	text
year	integer
gdp	numeric

Today we will use the **coronavirus** and **temperatures** datasets.

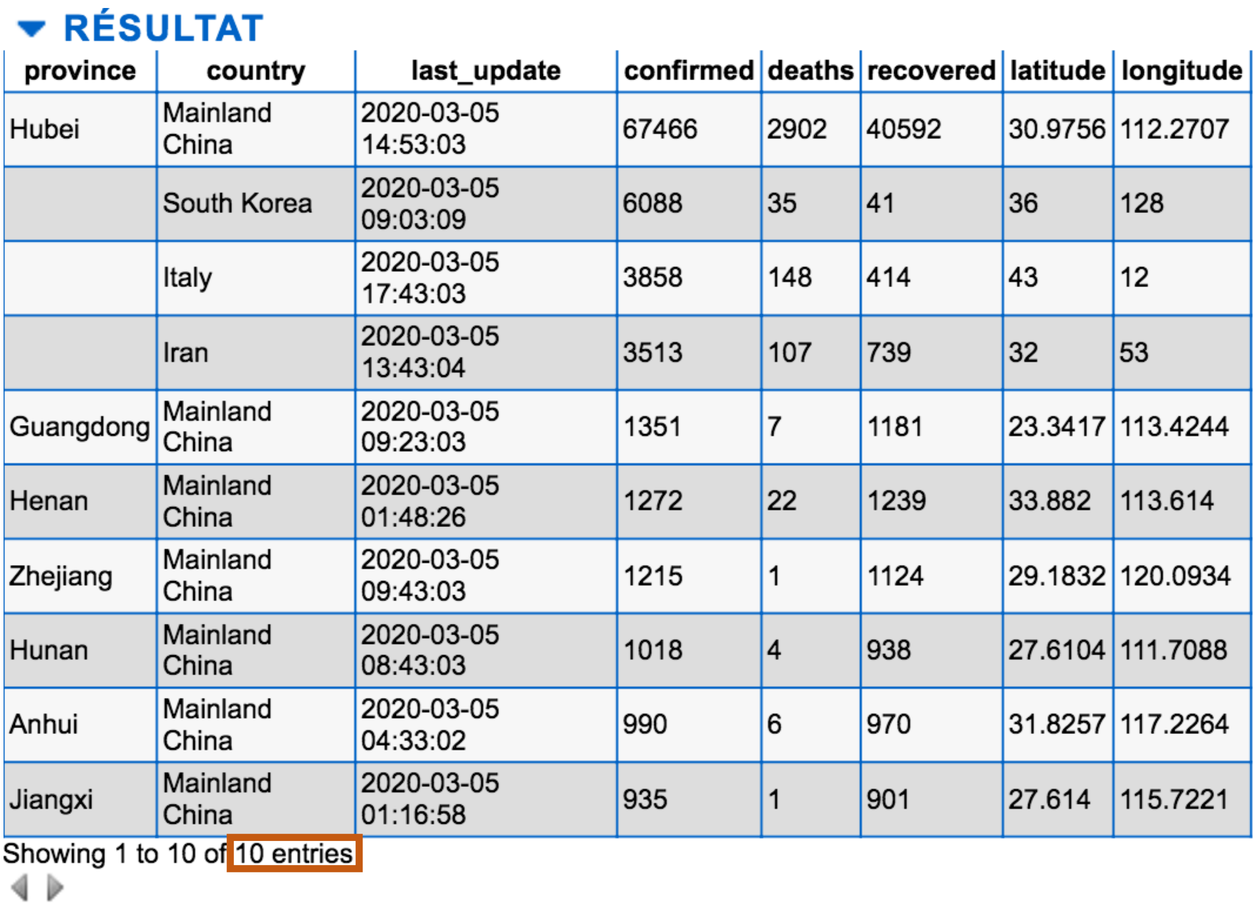
Let's start with **coronavirus**:

Have an overview of the **coronavirus** table:

```
select * from coronavirus limit 10
```

We only select the ten first ROWS the dataset

We select all COLUMNS from the dataset



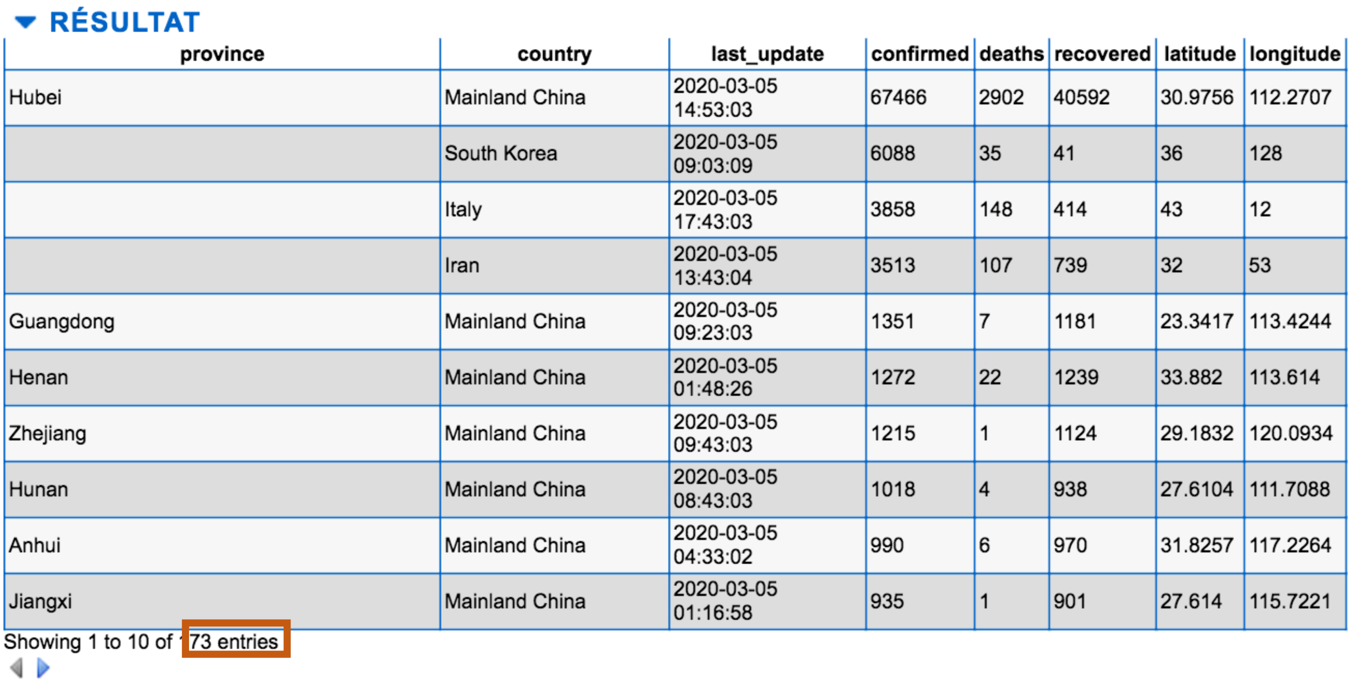
RÉSULTAT

province	country	last_update	confirmed	deaths	recovered	latitude	longitude
Hubei	Mainland China	2020-03-05 14:53:03	67466	2902	40592	30.9756	112.2707
	South Korea	2020-03-05 09:03:09	6088	35	41	36	128
	Italy	2020-03-05 17:43:03	3858	148	414	43	12
	Iran	2020-03-05 13:43:04	3513	107	739	32	53
Guangdong	Mainland China	2020-03-05 09:23:03	1351	7	1181	23.3417	113.4244
Henan	Mainland China	2020-03-05 01:48:26	1272	22	1239	33.882	113.614
Zhejiang	Mainland China	2020-03-05 09:43:03	1215	1	1124	29.1832	120.0934
Hunan	Mainland China	2020-03-05 08:43:03	1018	4	938	27.6104	111.7088
Anhui	Mainland China	2020-03-05 04:33:02	990	6	970	31.8257	117.2264
Jiangxi	Mainland China	2020-03-05 01:16:58	935	1	901	27.614	115.7221

Showing 1 to 10 of 10 entries

Select all columns and all rows:

```
select * from coronavirus
```



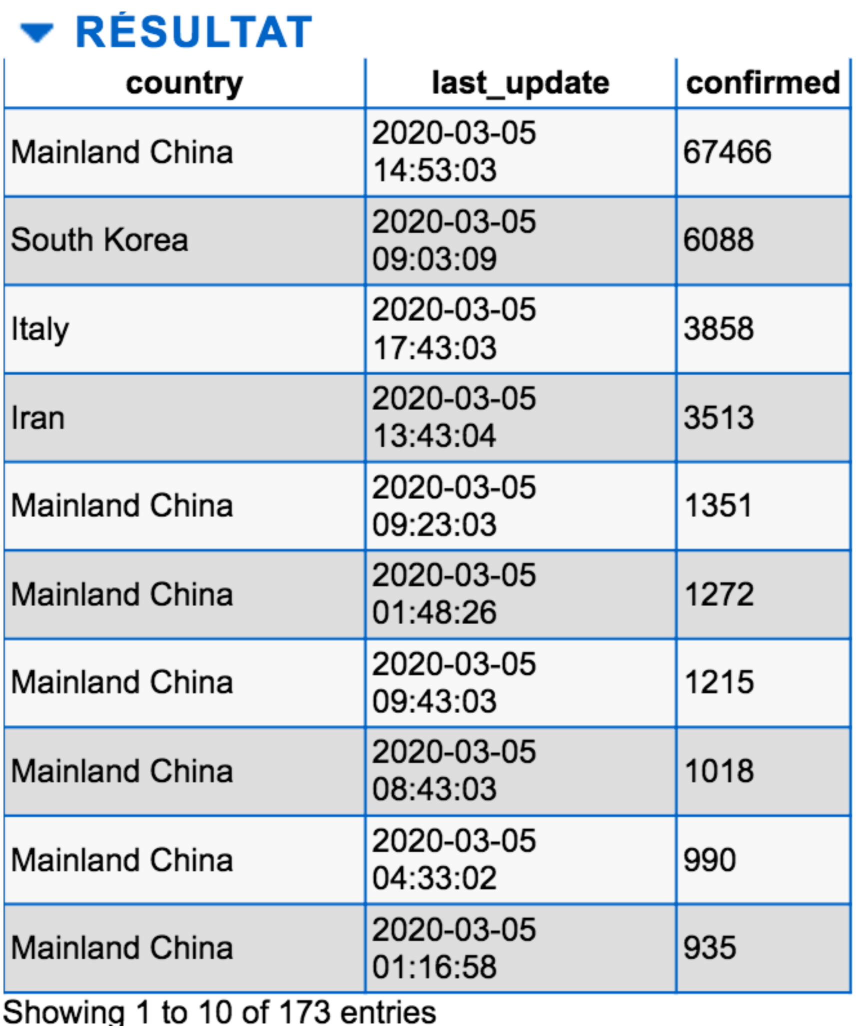
RÉSULTAT

province	country	last_update	confirmed	deaths	recovered	latitude	longitude
Hubei	Mainland China	2020-03-05 14:53:03	67466	2902	40592	30.9756	112.2707
	South Korea	2020-03-05 09:03:09	6088	35	41	36	128
	Italy	2020-03-05 17:43:03	3858	148	414	43	12
	Iran	2020-03-05 13:43:04	3513	107	739	32	53
Guangdong	Mainland China	2020-03-05 09:23:03	1351	7	1181	23.3417	113.4244
Henan	Mainland China	2020-03-05 01:48:26	1272	22	1239	33.882	113.614
Zhejiang	Mainland China	2020-03-05 09:43:03	1215	1	1124	29.1832	120.0934
Hunan	Mainland China	2020-03-05 08:43:03	1018	4	938	27.6104	111.7088
Anhui	Mainland China	2020-03-05 04:33:02	990	6	970	31.8257	117.2264
Jiangxi	Mainland China	2020-03-05 01:16:58	935	1	901	27.614	115.7221

Showing 1 to 10 of 173 entries

Select specific columns, for example **country, last_update, confirmed**:

```
Select country, last_update, confirmed from
coronavirus
```

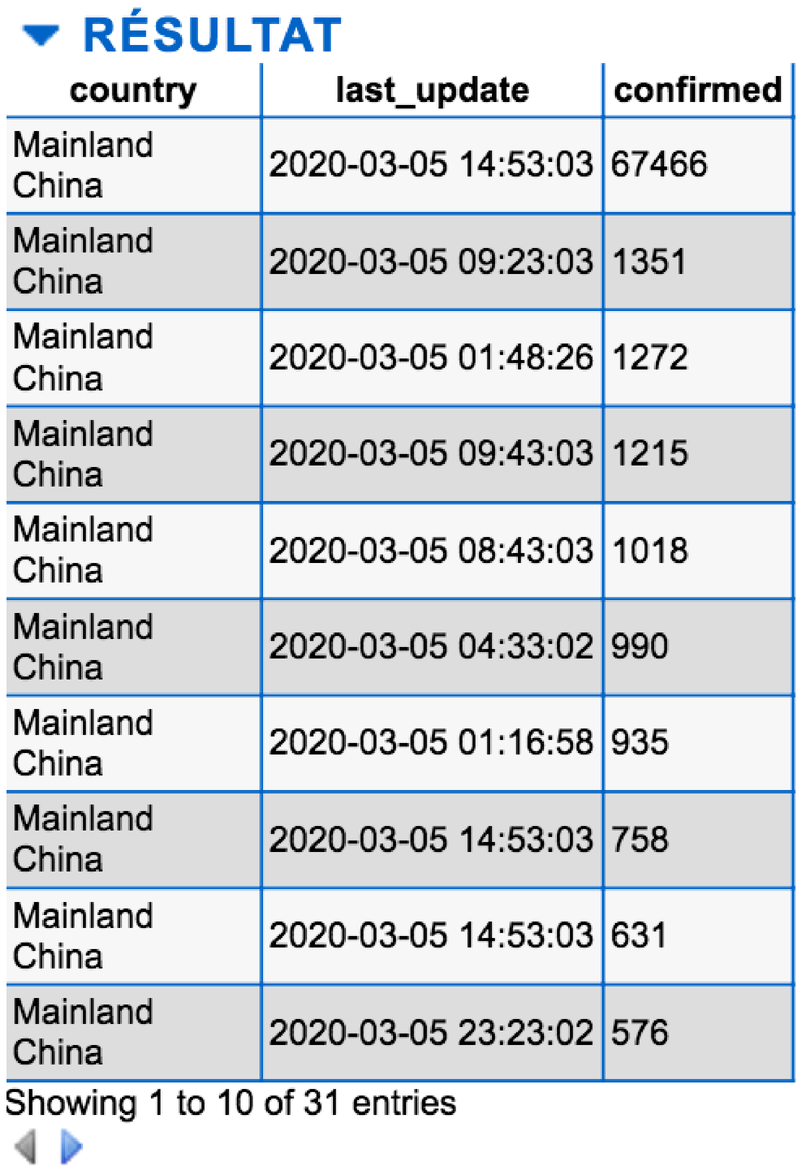


country	last_update	confirmed
Mainland China	2020-03-05 14:53:03	67466
South Korea	2020-03-05 09:03:09	6088
Italy	2020-03-05 17:43:03	3858
Iran	2020-03-05 13:43:04	3513
Mainland China	2020-03-05 09:23:03	1351
Mainland China	2020-03-05 01:48:26	1272
Mainland China	2020-03-05 09:43:03	1215
Mainland China	2020-03-05 08:43:03	1018
Mainland China	2020-03-05 04:33:02	990
Mainland China	2020-03-05 01:16:58	935

Showing 1 to 10 of 173 entries

Key word **WHERE** is used to add a condition to your request.
For example:

```
select country, last_update, confirmed
from coronavirus
where country = 'Mainland China'
```



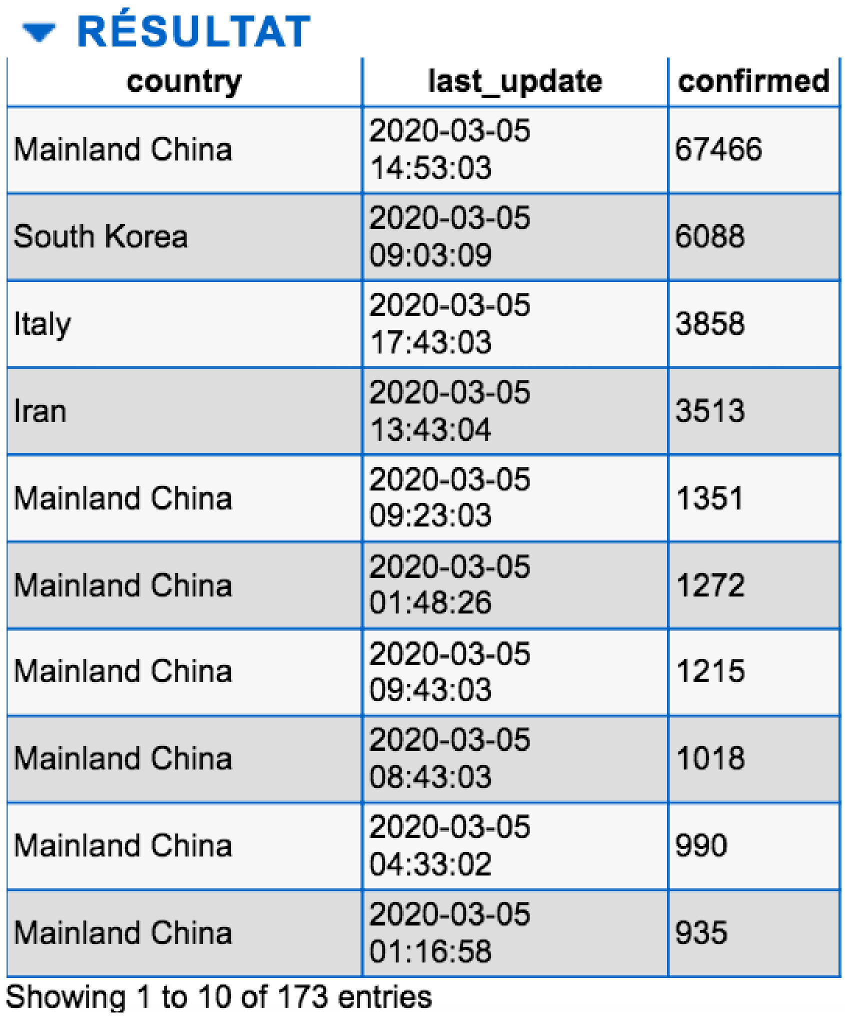
country	last_update	confirmed
Mainland China	2020-03-05 14:53:03	67466
Mainland China	2020-03-05 09:23:03	1351
Mainland China	2020-03-05 01:48:26	1272
Mainland China	2020-03-05 09:43:03	1215
Mainland China	2020-03-05 08:43:03	1018
Mainland China	2020-03-05 04:33:02	990
Mainland China	2020-03-05 01:16:58	935
Mainland China	2020-03-05 14:53:03	758
Mainland China	2020-03-05 14:53:03	631
Mainland China	2020-03-05 23:23:02	576

Showing 1 to 10 of 31 entries

Key word **ORDER BY** is used to order the results of a request by a column in an ascending (ASC) or descending (**DESC**) manner.

For example:

```
select country, last_update, confirmed
from coronavirus
order by confirmed DESC
```



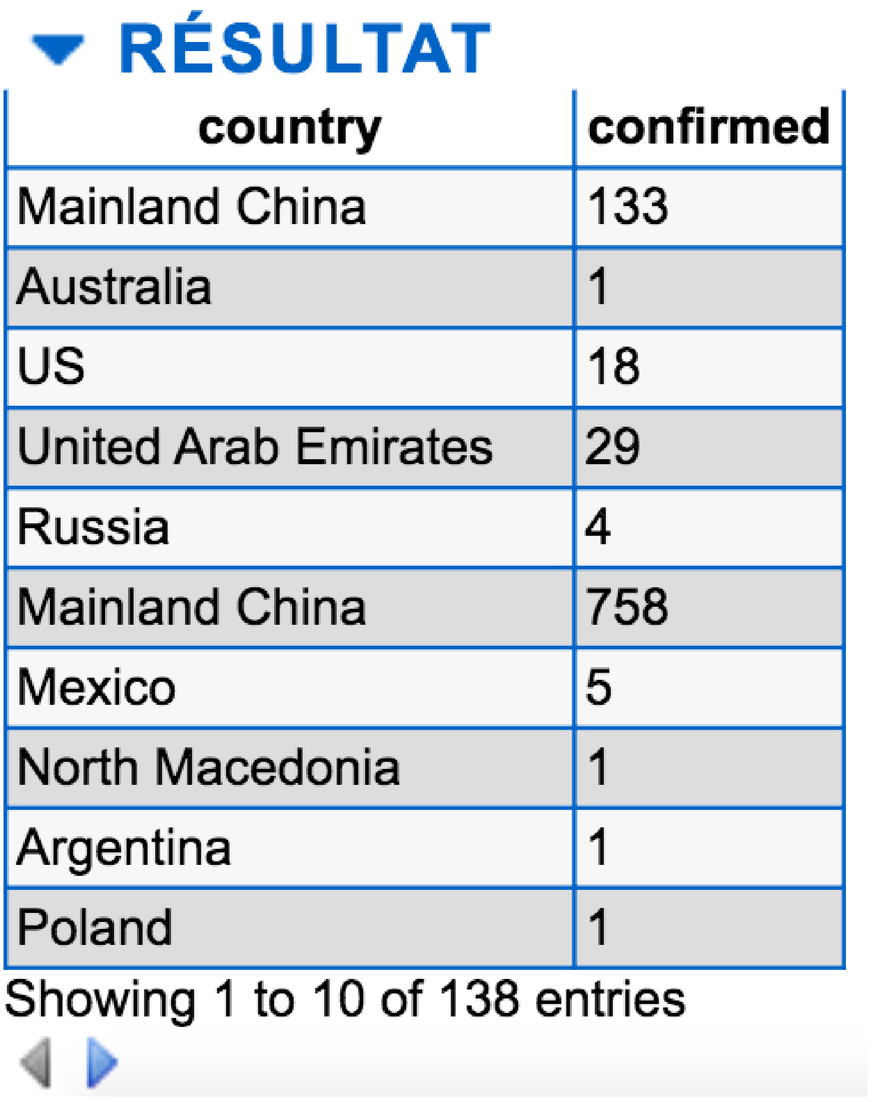
RÉSULTAT

country	last_update	confirmed
Mainland China	2020-03-05 14:53:03	67466
South Korea	2020-03-05 09:03:09	6088
Italy	2020-03-05 17:43:03	3858
Iran	2020-03-05 13:43:04	3513
Mainland China	2020-03-05 09:23:03	1351
Mainland China	2020-03-05 01:48:26	1272
Mainland China	2020-03-05 09:43:03	1215
Mainland China	2020-03-05 08:43:03	1018
Mainland China	2020-03-05 04:33:02	990
Mainland China	2020-03-05 01:16:58	935

Showing 1 to 10 of 173 entries

Key word **GROUP BY** is used to group the results of a request by a column.
For example:

```
select country, confirmed
from coronavirus
group by country, confirmed
```

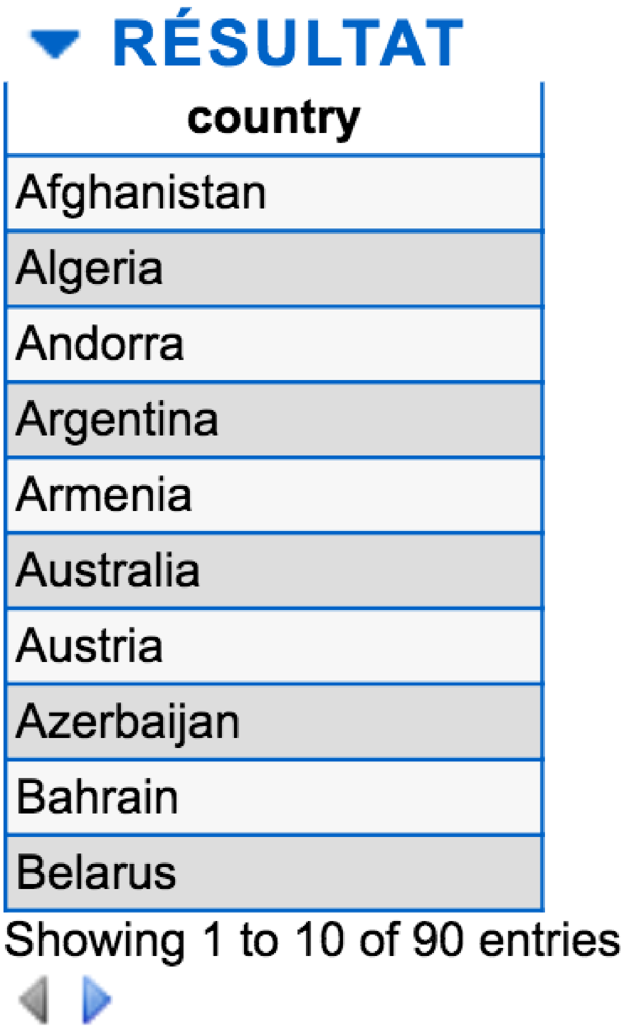


country	confirmed
Mainland China	133
Australia	1
US	18
United Arab Emirates	29
Russia	4
Mainland China	758
Mexico	5
North Macedonia	1
Argentina	1
Poland	1

Showing 1 to 10 of 138 entries

Key word **DISTINCT** is used to display distinct (different) values.
For example:

```
select distinct country
from coronavirus
order by country ASC
```


RÉSULTAT

country
Afghanistan
Algeria
Andorra
Argentina
Armenia
Australia
Austria
Azerbaijan
Bahrain
Belarus

Showing 1 to 10 of 90 entries

<u>Other conditions :</u>

Comparison of values : **=, >, <, >=, <=, <>**
Ex :

```
Select country, last_update, confirmed
from coronavirus
where confirmed > 1000
```

Interval : **[NOT] BETWEEN … AND …**
EX :

```
Select country, last_update, confirmed
from coronavirus
where confirmed between 2000 and 10000
```

List of values : **[NOT] IN** (list of values)
EX :

```sql
Select country, last_update, confirmed
from coronavirus
where confirmed not in (1,2,3,6088,67466)
```

List of values: **[NOT] LIKE** (partial value chaine)
EX :

```sql
Select country, last_update, confirmed
from coronavirus
where country like 'I%'
```

/!\ ‘I%’ means all the words that start with ‘I’

Undetermination: **IS [NOT] NULL**
EX :

```sql
Select country, last_update, confirmed
from coronavirus
where confirmed is not null
```

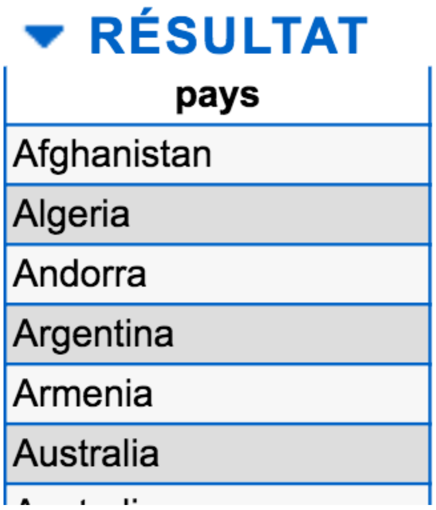
/!\ NULL is different from zéro

Arithmetic expressions:
To multiply: *
To divide: /
To add: +
To subtract: -

/!\ only usable on variables of type date or time

Key word **AS** is used to give a new name to a column when extracting.
For example:

```sql
select country AS pays
from coronavirus
order by country ASC
```



pays
Afghanistan
Algeria
Andorra
Argentina
Armenia
Australia

Logical operators:

```
AND: TRUE if the two conditions are true
OR: TRUE if at least one of the conditions is true
NOT: if none of the conditions are true
```

2) Exercises to do

A)Using Covid dataset

The latest data you can find here:
https://github.com/CSSEGISandData/COVID-19/tree/master/csse_covid_19_data/csse_covid_19_daily_reports

1.Find the total number of confirmed cases worldwide.

select sum(confirmed) from coronavirus

⇨ 197'168

2.Find the total number of deaths in China.

select sum(deaths) from coronavirus
where country = 'China'

⇨ 3230

3.How many confirmed cases in USA in California, Colorado and Conneticut?
[look at the data, to see how you can formulate this query]
(see how the data are)

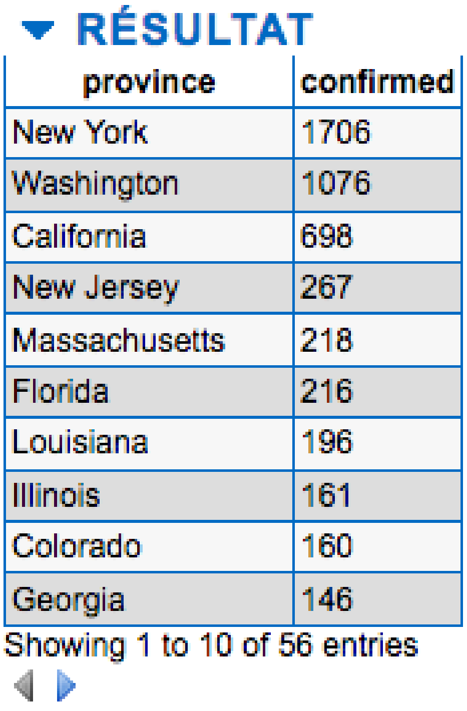
select * from coronavirus where country = ‘US’

select sum(confirmed) from coronavirus
where country = 'US'
and province like 'C%'

⇨ 926

4.Show all the provinces and confirmed cases in the USA ordered by number of confirmed cases (descending)

select province, confirmed from coronavirus
where country = 'US'
order by confirmed desc



RÉSULTAT

province	confirmed
New York	1706
Washington	1076
California	698
New Jersey	267
Massachusetts	218
Florida	216
Louisiana	196
Illinois	161
Colorado	160
Georgia	146

Showing 1 to 10 of 56 entries

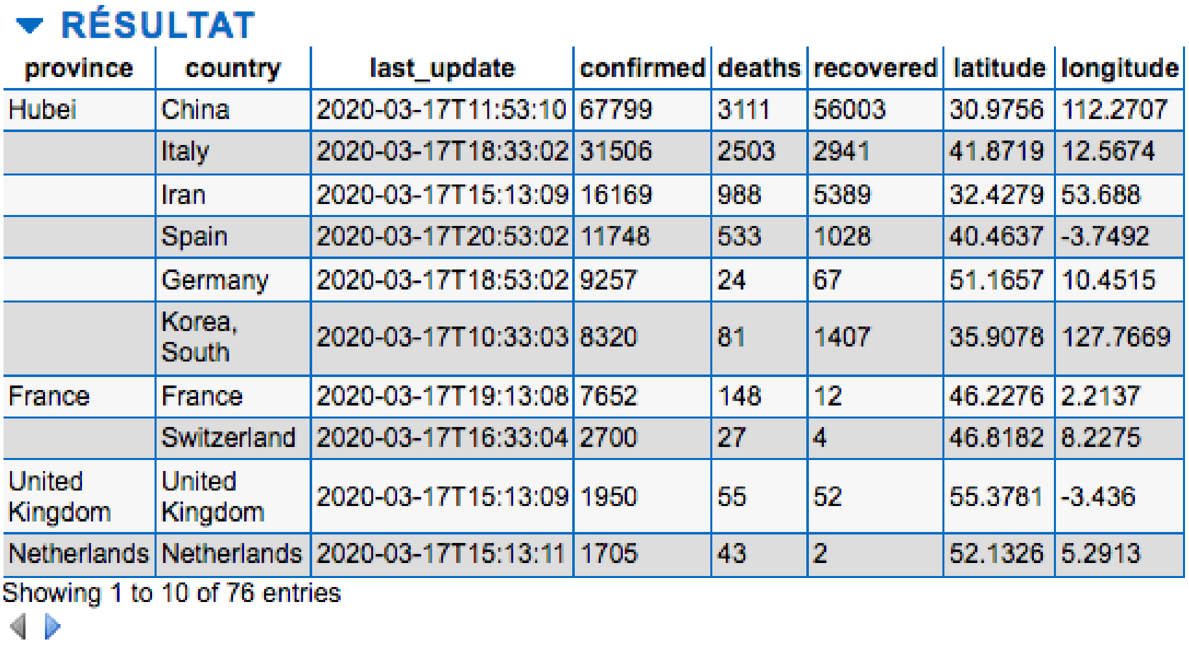
5.How many will be the estimated cases next week for Switzerland if each week the cases multiply by 4?

```
select province, country, confirmed,
round(confirmed*4) as estimated_next_week
from coronavirus
where country = 'Switzerland'
```

⇨ 10'800

6. Find all the records not in USA where confirmed cases are more than 100 (order in descending manner).

```
select * from coronavirus
where country NOT in ('US') and confirmed>100
order by confirmed desc
```



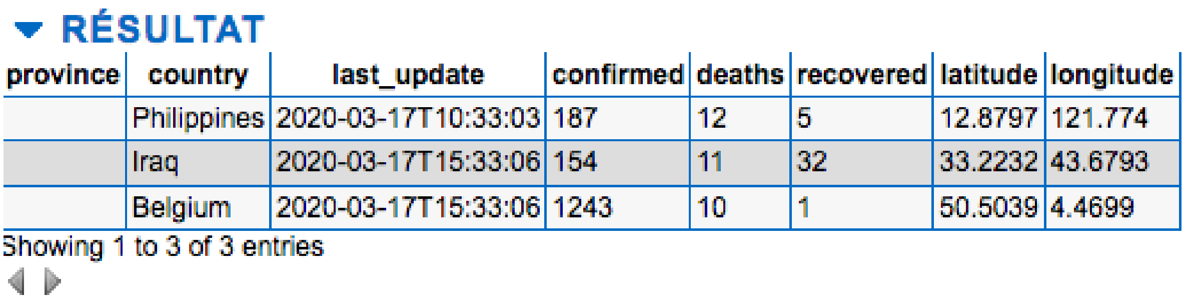
▼ RÉSULTAT

province	country	last_update	confirmed	deaths	recovered	latitude	longitude
Hubei	China	2020-03-17T11:53:10	67799	3111	56003	30.9756	112.2707
	Italy	2020-03-17T18:33:02	31506	2503	2941	41.8719	12.5674
	Iran	2020-03-17T15:13:09	16169	988	5389	32.4279	53.688
	Spain	2020-03-17T20:53:02	11748	533	1028	40.4637	-3.7492
	Germany	2020-03-17T18:53:02	9257	24	67	51.1657	10.4515
	Korea, South	2020-03-17T10:33:03	8320	81	1407	35.9078	127.7669
France	France	2020-03-17T19:13:08	7652	148	12	46.2276	2.2137
	Switzerland	2020-03-17T16:33:04	2700	27	4	46.8182	8.2275
United Kingdom	United Kingdom	2020-03-17T15:13:09	1950	55	52	55.3781	-3.436
Netherlands	Netherlands	2020-03-17T15:13:11	1705	43	2	52.1326	5.2913

Showing 1 to 10 of 76 entries

7. Find all the records not about USA and China that have between 10 and 20 deaths

```
select * from coronavirus
where country not in ('US' , 'China')
and deaths between 10 and 20
order by deaths desc
```


RÉSULTAT

province	country	last_update	confirmed	deaths	recovered	latitude	longitude
	Philippines	2020-03-17T10:33:03	187	12	5	12.8797	121.774
	Iraq	2020-03-17T15:33:06	154	11	32	33.2232	43.6793
	Belgium	2020-03-17T15:33:06	1243	10	1	50.5039	4.4699

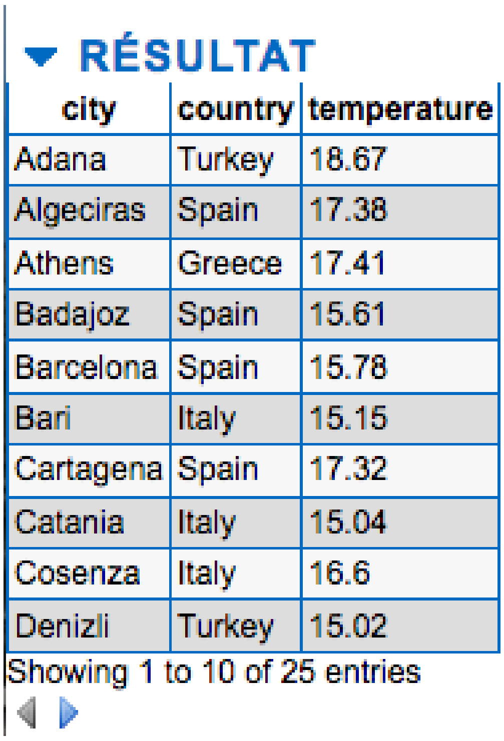
Showing 1 to 3 of 3 entries

B)Using Temperatures dataset

Now, let's use the **temperatures** datasets :
https://hec.unil.ch/info1ere/sqlexplorer/Temperatures

1. Find all cities with temperature between 15 and 25, return city, country, temperature

select city, country, temperature
from cities where temperature between 15 and 25

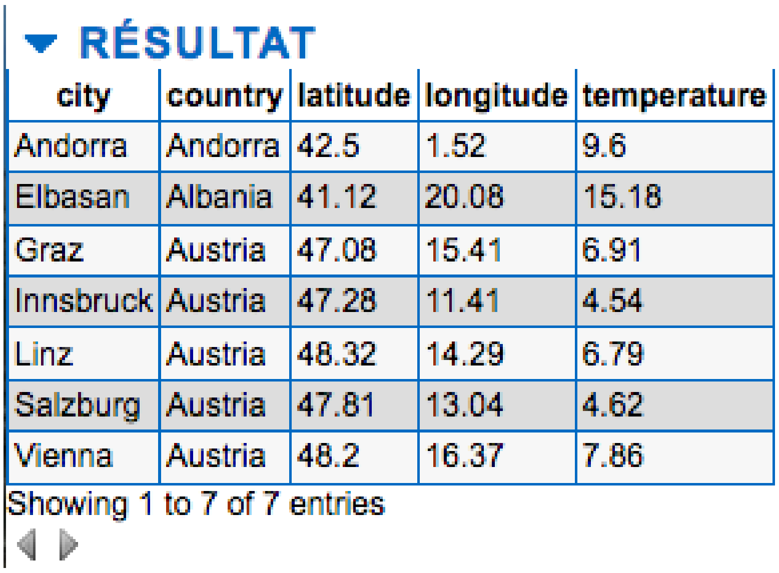

RÉSULTAT

city	country	temperature
Adana	Turkey	18.67
Algeciras	Spain	17.38
Athens	Greece	17.41
Badajoz	Spain	15.61
Barcelona	Spain	15.78
Bari	Italy	15.15
Cartagena	Spain	17.32
Catania	Italy	15.04
Cosenza	Italy	16.6
Denizli	Turkey	15.02

Showing 1 to 10 of 25 entries

2. Find all temperature records that contain a country whose name starts from A or a

select * from cities where upper(country) LIKE 'A%'


RÉSULTAT

city	country	latitude	longitude	temperature
Andorra	Andorra	42.5	1.52	9.6
Elbasan	Albania	41.12	20.08	15.18
Graz	Austria	47.08	15.41	6.91
Innsbruck	Austria	47.28	11.41	4.54
Linz	Austria	48.32	14.29	6.79
Salzburg	Austria	47.81	13.04	4.62
Vienna	Austria	48.2	16.37	7.86

Showing 1 to 7 of 7 entries

3. What is the maximum temperature in Austria?

select max(temperature) from cities where country='Austria'

⇨ 7.86

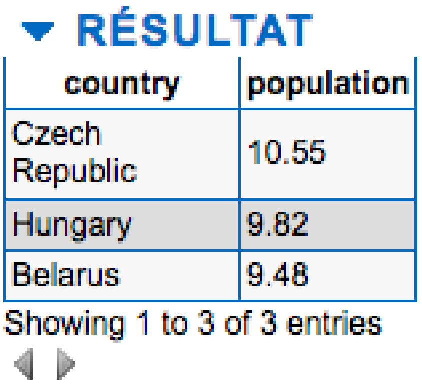
4. What is the average temperature in records that are from Italy or Greece?

select avg(temperature) from cities where country='Italy' or country='Greece'
or
select avg(temperature) from cities where country in ('Italy', 'Greece')

⇨ 14.1963157894736842

5. Find all the countries and their population without coastline and with population more than 9 million (population column is in millions).

select country, population
from Countries
where coastline = 'f' and population > 9
order by population desc



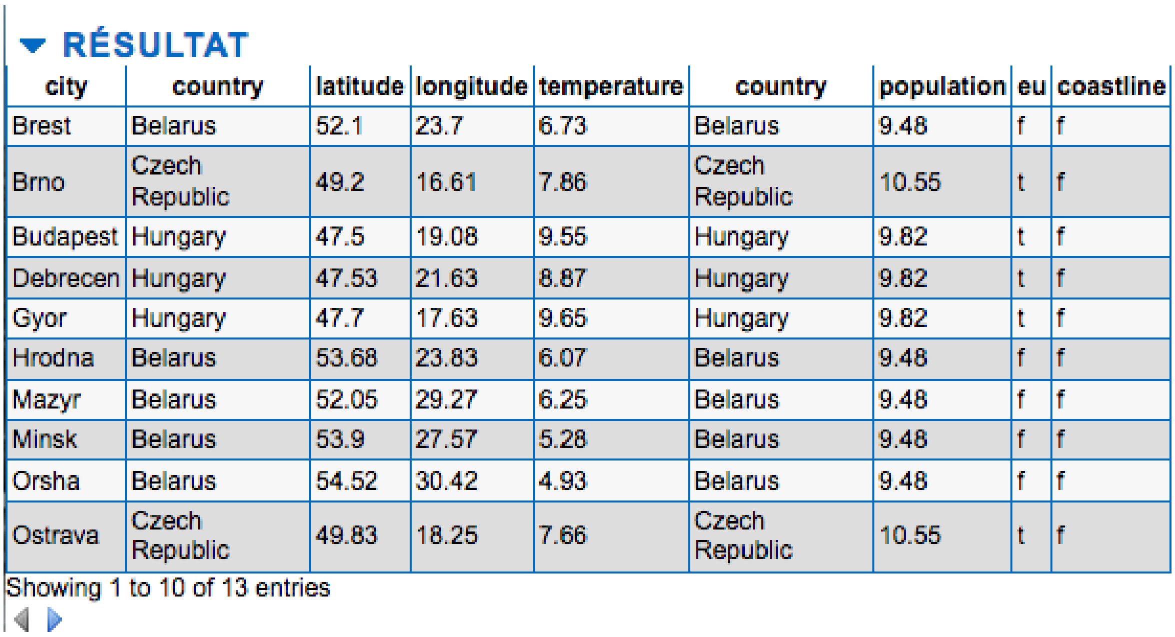
RÉSULTAT

country	population
Czech Republic	10.55
Hungary	9.82
Belarus	9.48

Showing 1 to 3 of 3 entries

6. Find all cities with latitude more than 45, no coastline and population more than 9 million.

```
select *
from Cities, Countries
where Cities.country = Countries.country
and latitude > 45 and coastline = 'f' and population > 9
```



RÉSULTAT

city	country	latitude	longitude	temperature	country	population	eu	coastline
Brest	Belarus	52.1	23.7	6.73	Belarus	9.48	f	f
Brno	Czech Republic	49.2	16.61	7.86	Czech Republic	10.55	t	f
Budapest	Hungary	47.5	19.08	9.55	Hungary	9.82	t	f
Debrecen	Hungary	47.53	21.63	8.87	Hungary	9.82	t	f
Gyor	Hungary	47.7	17.63	9.65	Hungary	9.82	t	f
Hrodna	Belarus	53.68	23.83	6.07	Belarus	9.48	f	f
Mazyr	Belarus	52.05	29.27	6.25	Belarus	9.48	f	f
Minsk	Belarus	53.9	27.57	5.28	Belarus	9.48	f	f
Orsha	Belarus	54.52	30.42	4.93	Belarus	9.48	f	f
Ostrava	Czech Republic	49.83	18.25	7.66	Czech Republic	10.55	t	f

Showing 1 to 10 of 13 entries

7. **How many** countries have latitude more than 45, no coastline and population more than 9 million.

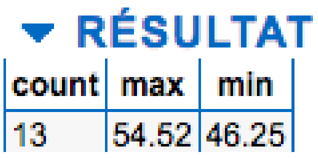
```
select count(*)
from Cities, Countries
```

```
where Cities.country = Countries.country
and latitude > 45 and coastline = 'f' and population > 9
```

⇨ 13

8. **How many** cities have latitude more than 45, no coastline and population more than 9 million, AND what is the maximum and minimum latitude of those cities ?

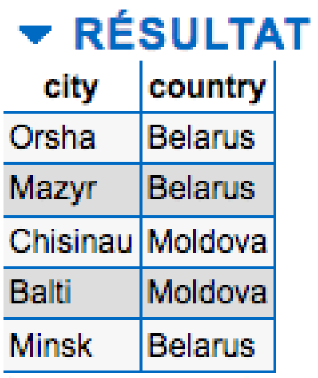
```
select count(*), min(latitude), max(latitude)
from Cities, Countries
where Cities.country = Countries.country
and latitude > 45 and coastline = 'f' and population > 9
```


RÉSULTAT

count	max	min
13	54.52	46.25

9. Find the cities and countries without coastline. Sort them by descending longitude and return just 5 results.

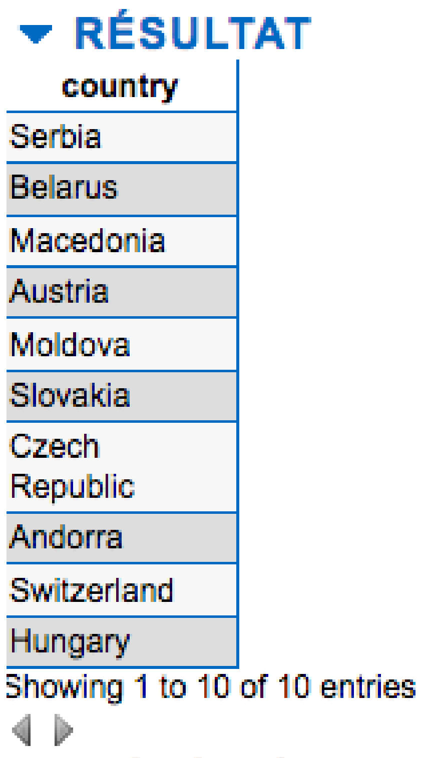
```
select city, Cities.country
from Cities, Countries
where Cities.country = Countries.country
and coastline = 'no'
order by longitude desc
limit 5
```


RÉSULTAT

city	country
Orsha	Belarus
Mazyr	Belarus
Chisinau	Moldova
Balti	Moldova
Minsk	Belarus

10. What are the countries without coastline?

```
select distinct(Cities.country)
from Cities, Countries
where Cities.country = Countries.country
and coastline = 'no'
```



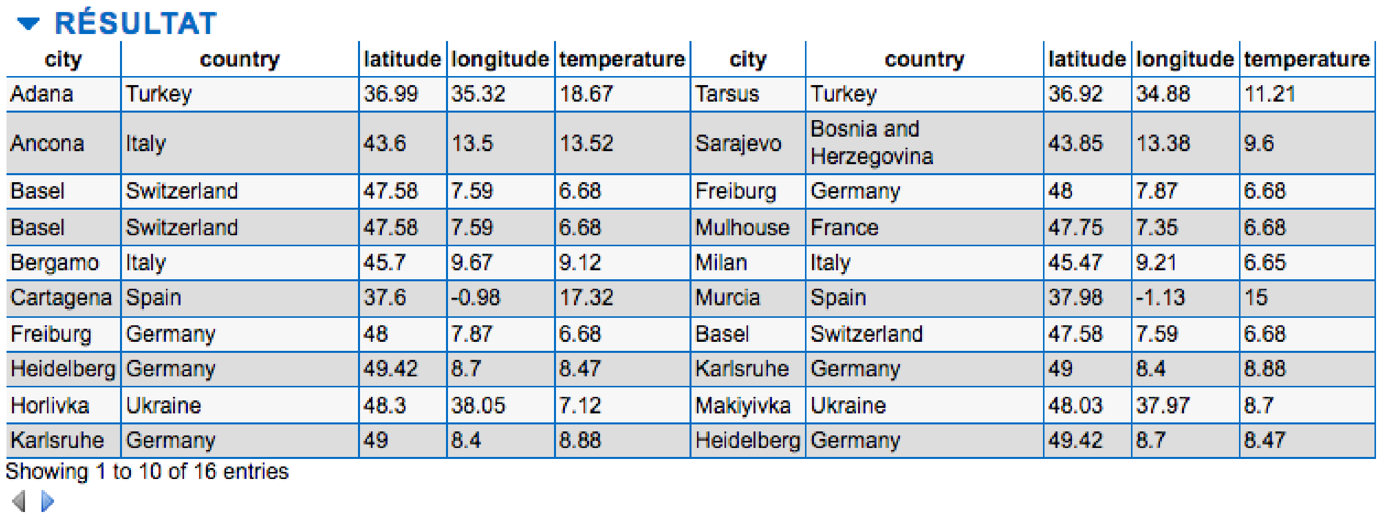
country
Serbia
Belarus
Macedonia
Austria
Moldova
Slovakia
Czech Republic
Andorra
Switzerland
Hungary

Showing 1 to 10 of 10 entries

11. Find all pairs of cities that are close together, i.e., longitude and latitude are less than 0.5 apart (self-join: join a table with itself), do not include a city with itself!

[results should be 16]

```
select *
from cities A, cities B
where
abs(A.latitude - B.latitude) < 0.5 and
abs(A.longitude - B.longitude) < 0.5 and
A.city != B.city
```

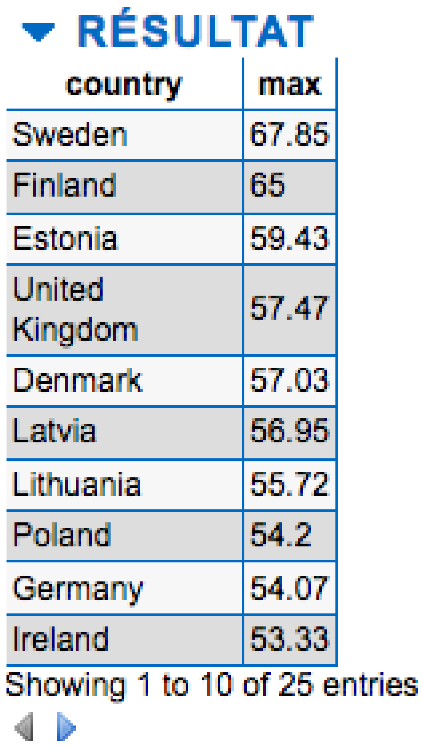

RÉSULTAT

city	country	latitude	longitude	temperature	city	country	latitude	longitude	temperature
Adana	Turkey	36.99	35.32	18.67	Tarsus	Turkey	36.92	34.88	11.21
Ancona	Italy	43.6	13.5	13.52	Sarajevo	Bosnia and Herzegovina	43.85	13.38	9.6
Basel	Switzerland	47.58	7.59	6.68	Freiburg	Germany	48	7.87	6.68
Basel	Switzerland	47.58	7.59	6.68	Mulhouse	France	47.75	7.35	6.68
Bergamo	Italy	45.7	9.67	9.12	Milan	Italy	45.47	9.21	6.65
Cartagena	Spain	37.6	-0.98	17.32	Murcia	Spain	37.98	-1.13	15
Freiburg	Germany	48	7.87	6.68	Basel	Switzerland	47.58	7.59	6.68
Heidelberg	Germany	49.42	8.7	8.47	Karlsruhe	Germany	49	8.4	8.88
Horlivka	Ukraine	48.3	38.05	7.12	Makiyivka	Ukraine	48.03	37.97	8.7
Karlsruhe	Germany	49	8.4	8.88	Heidelberg	Germany	49.42	8.7	8.47

Showing 1 to 10 of 16 entries

12. What is the maximum latitude for all countries in EU? (we have maximum, which hints it may be a "group by" if it is for all entities, which it is. If it was for one entity, then we wouldn't need a group by.)

```
select Cities.country, max(latitude)
from Cities, Countries
where Cities.country = Countries.country
and EU = 'yes'
group by Cities.country
order by max(latitude) desc
```


RÉSULTAT

country	max
Sweden	67.85
Finland	65
Estonia	59.43
United Kingdom	57.47
Denmark	57.03
Latvia	56.95
Lithuania	55.72
Poland	54.2
Germany	54.07
Ireland	53.33

Showing 1 to 10 of 25 entries