Universidad de Chile
Facultad de Ciencias Físicas y Matemáticas
Departamento de Ciencias de la Computación

Reporte "Análisis del COVID-19"

Bases de Datos CC3201-1

Integrantes Grupo 6: Felipe Escárate Fernandez
Nahuel Gómez Raguileo
Miguel Monzó Díaz
Profesor: Claudio Gutiérrez
Auxiliar: Benjamín Correa Karstulovic
Cristóbal Miranda
Florencia Yáñez Gutiérrez
Raúl Alexander Silva A.
Fecha de entrega: 13 de Diciembre de 2021
Santiago de Chile

Índice de Contenidos

Índice de Figuras

Índice de Códigos

Resumen

El COVID-19 ha sido uno de los virus más desafiantes para la medicina moderna, y si bien es cierto ha sido una catástrofe a nivel mundial, los países han tenido distintas repercusiones debido a la pandemia. Creemos fehacientemente que la manera en que ha sido tratada la pandemia va directamente ligada al nivel socioeconómico previo del país, ya sea por la credibilidad política que tengan los habitantes a las autoridades locales, como el hecho de tener suficientes ingresos para poder comprar una cantidad de vacunas suficientes para vacunar a toda la población.

En el presente proyecto tomaremos datos económicos de la mayoría de los países del mundo, los cuales no siendo totalmente recientes (2017) creemos son suficientes para darnos una idea extrapolada de cual era la condición inicial del país previa pandemia, gracias a esto y A un dataset que contenga el seguimiento de casos (también por país) podremos hacer un cruce de información para lograr una conclusión acertada.

Búsqueda de datos

Para la búsqueda de datos no basamos principalmente en las tablas que posee la página https://www.kaggle.com/, en esta pudimos encontrar información acerca de los casos activos, progreso de vacunación, casos recuperados, PIB, entre otros. Si bien es cierto habían datasets con una mayor cantidad de información (por ejemplo el indice de contagio), esta era solo para países particulares, por lo que fueron rechazados a la brevedad.

A continuación se presentan algunos de los datasets más interesantes que cumplían con nuestros requisitos de investigación:

- Muertes
  https://www.kaggle.com/dhruvildave/covid19-deaths-dataset?select=us-counties.csv
- Datos vacunación 1
  https://www.kaggle.com/gpreda/covid-world-vaccination-progress
- Datos vacunación 2
  https://www.kaggle.com/fedesoriano/coronavirus-covid19-vaccinations-data
- Datos Socioeconómicos
  https://www.kaggle.com/nishanthsalian/socioeconomic-country-profiles
- Población por país
  https://www.kaggle.com/tanuprabhu/population-by-country-2020
- Datos varios OMS
  https://covid19.who.int/info/
- Datos muertes y recuperados Covid 19
  https://www.kaggle.com/imdevskp/corona-virus-report?select=covid_19_clean_complete.csv

Selección de datos

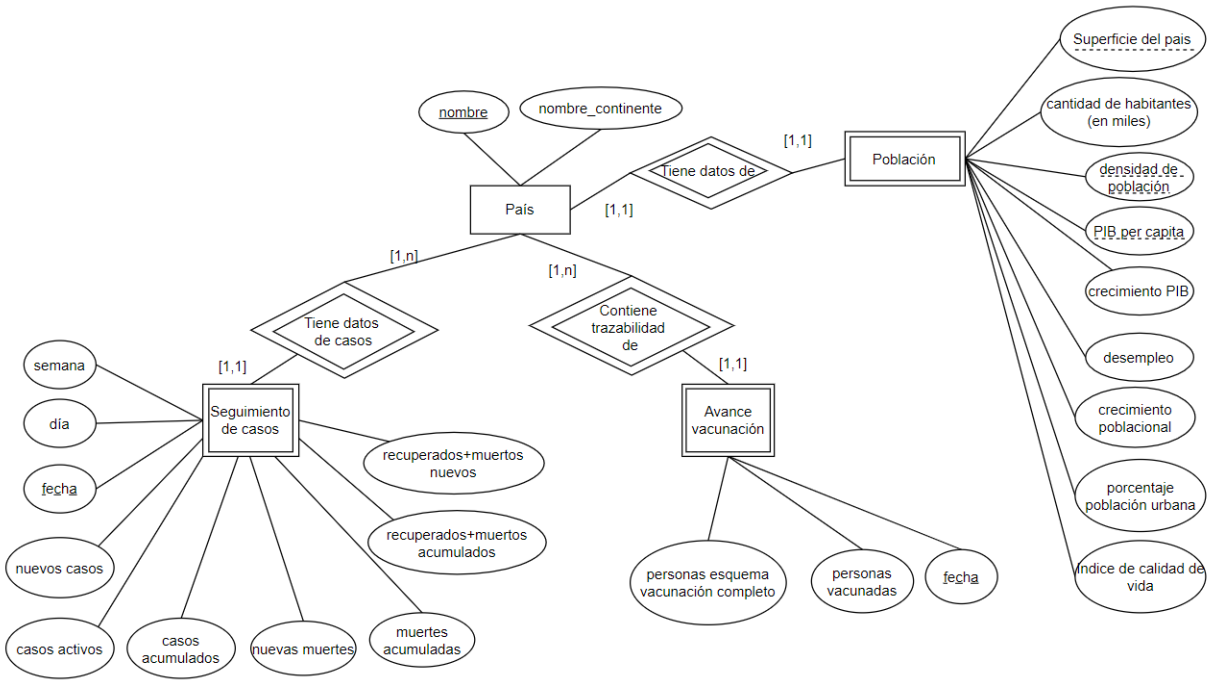
Dado que no nos sirve tener datos repetidos, debimos elegir entre los links antes mostrados para no entrar en conflictos de información. Debido al prestigio (y la continua actualización que recibe el dataset), nos quedamos con las tablas de la OMS con los nombres: “Daily cases and deaths by date reported to WHO”, “Vaccination data”, “Latest reported counts of cases and deaths, and transmission classification”, “Vaccination metadata”; junto a estas, dejamos los datos socioeconómicos para hacer la trazabilidad de información.

- Datos Socioeconómicos
  https://www.kaggle.com/nishanthsalian/socioeconomic-country-profiles
- Datos varios OMS
  https://covid19.who.int/info/

Análisis de datos

Creación Modelo ER

Teniendo los datos que se utilizarían en el proyecto, se procedió a desarrollar el Modelo Entidad-Relación. Tal como se llevó a cabo en el Laboratorio 1, se seleccionó la entidad principal para nuestro trabajo, en este caso resultó ser "País", de esta deriva información relevante como el "Continente" al que pertenecen (y así abarcar información general continental), además se le incorporó información de "Población" y "Seguimiento de casos":



Traspaso al Modelo Relacional

Entidades
País(nombre:string,nombre_continente:string)
Población(País.nombre:string, superficie_del_país:float, cantidad_de_habitantes:int, densidad_de_población:float, PIB_per_capita:float, crecimiento_PIB:float, desempleo, crecimiento_poblacional:float, porcentaje_población_urbana:float, índice_de_calidad_de_vida:float)
Seguimiento_de_casos(País.nombre:string, fecha:date, día:int, semana:int, muertes:int, recuperados:int, casos_actuales:int)
Avance_vacunación(País.nombre:string, fecha:date, personas_esquema_vacunación_completo:int, personas_vacunadas:int)

Procesamiento de datos

Pre-procesamiento de datos

Una vez seleccionados los data-frames a utilizar, se trabajan con el fin de obtener una tabla funcional que pierda la menor cantidad de datos posibles. Para ello se comienza verificando que los nombres de los países coincidan con la tabla país (que contiene a las llaves). Una vez concluido el paso anterior, se procede a realizar un OUTER JOIN entre las tablas que se van a utilizar de modo que no se pierda ningún dato al momento de unirlas. Por último se realiza la limpieza de los datos, eliminando los datos NA y las columnas que no se van a utilizar, obteniendo así las tablas que se cargarán a PostgreSQL.

Implementación relacional

Una vez obtenidos los archivos csv "finales", se cargan a PostgreSQL para así poder trabajarlos. Para ello se deben comenzar creando el esquema y las tablas que contendran los datos que nos interesa cargar. Para ello se siguieron los pasos para la creación de Entidades vista en el laboratorio 5.

Código 1: Creación del esquema

```
1 CREATE schema hito23;
```

Luego de tener listo el esquema en el que se adjuntarán las entidades, se procede a crear las reflejadas en el diagrama del modelo ER.

Código 2: Creación entidad “País”

```
1 CREATE TABLE hito23.pais(
2   nombre VARCHAR(255),
3    nombre_continente VARCHAR(255),
4    PRIMARY KEY (nombre)
5 )
```

Código 3: Creación entidad “Pobalción”

```
CREATE TABLE hito23.poblacion(
  nombre_pais VARCHAR(255),
   superficie_del_pais FLOAT,
   cantidad_de_habitantes BIGINT,
   densidad_de_poblacion FLOAT,
   PIB_per_capita FLOAT,
   crecimiento_PIB FLOAT,
   desempleo FLOAT,
   crecimiento_poblacional FLOAT,
   porcentaje_población_urbana FLOAT,
   indice_de_calidad_de_vida FLOAT,
   FOREIGN KEY (nombre_pais) REFERENCES hito23.pais(nombre),
   PRIMARY KEY (nombre_pais)
)
```

Código 4: Creación entidad “Seguimiento de casos”

```
CREATE TABLE hito23.seguimiento_de_casos(
  nombre_pais VARCHAR(255),
   fecha DATE,
   dia INT,
   semana SMALLINT,
  nuevos_casos BIGINT,
  casos_actuales BIGINT,
  casos_acumulados BIGINT,
  nuevas_muertes BIGINT,
  muertes_acumuladas BIGINT,
  nuevos_recuperados_y_muertes BIGINT,
  recuperados_acumulados BIGINT,
   FOREIGN KEY (nombre_pais) REFERENCES hito23.pais(nombre),
   PRIMARY KEY (nombre_pais, fecha)
)
```

Código 5: Creación entidad “Avance de vacunación”

```
CREATE TABLE hito23.avance_de_vacunacion(
  nombre_pais VARCHAR(255),
   fecha DATE,
  personas_esquema_vacunación_completo BIGINT,
  personas_vacunadas BIGINT,
   FOREIGN KEY (nombre_pais) REFERENCES hito23.pais(nombre),
   PRIMARY KEY (nombre_pais, fecha)
)
```

Carga de datos

Previa a la carga de datos de Postgres se realizó un limpieza con “Pandas”. Esto debido a que habían tablas con más de 80 columnas, de las cuales no todas serían utilizadas. Además, el impor-

tarlas completas a Postgres habría sido tedioso en exceso y habría involucrado un gasto de recursos computacionales innecesario .

Algunos de los comandos utilizados en pandas para la limpieza de datos fueron:

Código 6: Filtrado de datos para la entidad País

```
#Ejemplo para entidad País
import pandas as pd

from google.colab import drive
drive.mount('/content/drive')

df = pd.read_csv('/content/drive/MyDrive/Trabajos en grupo/Bases de Datos/Proyecto/Tablas/
    ↪ socioEconEditado.csv')

df = df.loc[:,['country',
            'Region']]

df.to_csv('/content/drive/MyDrive/Trabajos en grupo/Bases de Datos/Proyecto/Tablas/
    ↪ paisConIndices.csv')

df2 = pd.read_csv('/content/drive/MyDrive/Trabajos en grupo/Bases de Datos/Proyecto/Tablas/
    ↪ paisConIndices.csv')
```

Posterior al filtrado de datos, se importaron los datos a las tablas indicadas en la parte anterior.

Código 7: Import de datos para “País”

```
COPY hito23.pais
   FROM 'H:\Bibliotecas\Escritorio\Tablas\paisSinIndices.csv'
   CSV HEADER;
```

Código 8: Import de datos para “Socieconómico”

```
COPY hito23.poblacion
   FROM 'H:\Bibliotecas\Escritorio\Tablas\socioEcon.csv'
   CSV HEADER;
```

Código 9: Import de datos para “Seguimiento de casos OJOJO”

```
COPY hito23.seguimiento_de_casos
   FROM 'H:\Bibliotecas\Escritorio\Tablas\seguimientoCasos.csv'
  WITH DELIMITER ';'
   CSV HEADER;
```

Código 10: Import de datos para “Avance de vacunación”

```
COPY hito23.avance_de_vacunacion
   FROM 'H:\Bibliotecas\Escritorio\Tablas\avanceVacunacion.csv'
   CSV HEADER;
```

Tablas resultantes

A continuación se presentan los pantallazos de las tablas resultantes en PostgreSQL:

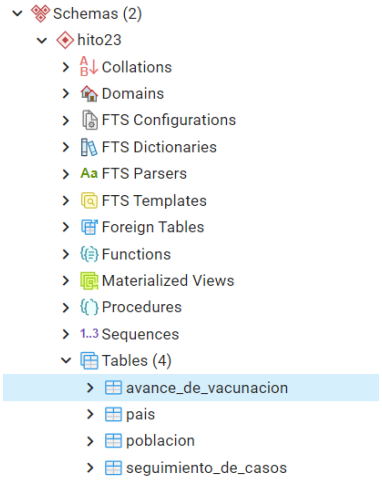


Figura 1: Interfaz de PostgreSQL en la que se pueden apreciar los nombres de las tablas que componen el Schema

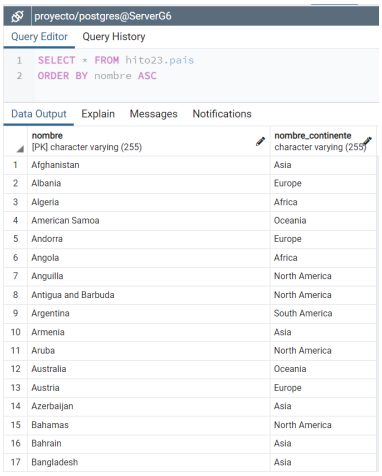
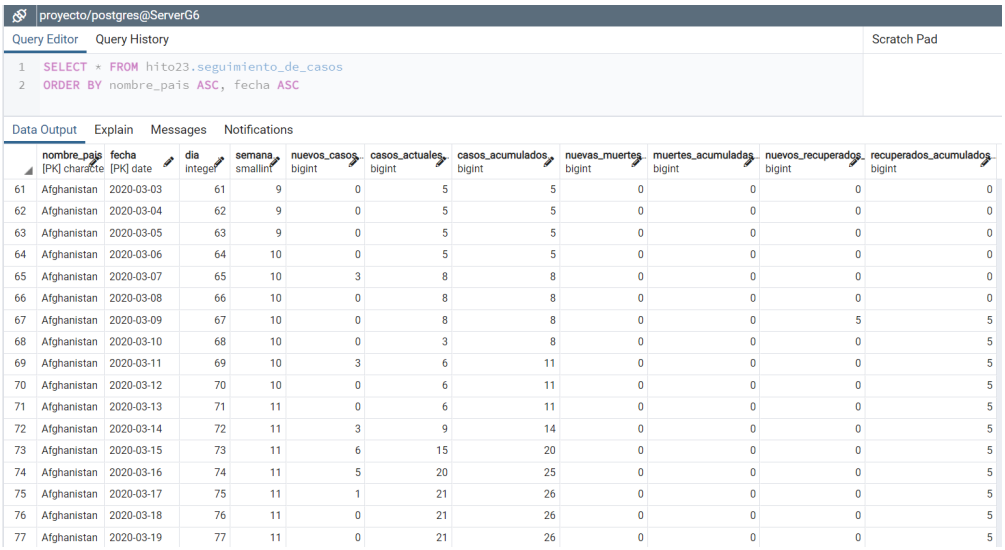


Figura 2: Tabla país en PostgreSQL


proyecto/postgres@ServerG6

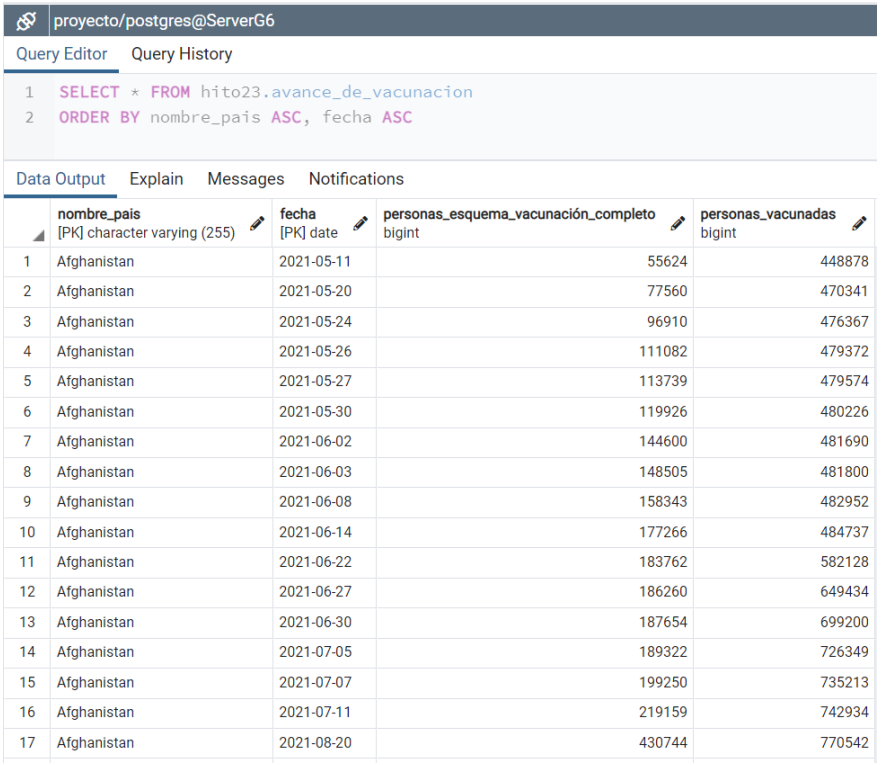
Query Editor | Query History | Scratch Pad

```
SELECT * FROM hito23.seguimiento_de_casos
ORDER BY nombre_pais ASC, fecha ASC
```

Data Output | Explain | Messages | Notifications

	nombre_pais [PK] characte	fecha [PK] date	dia integer	semana smallint	nuevos_casos bigint	casos_actuales bigint	casos_acumulados bigint	nuevas_muertes bigint	muertes_acumuladas bigint	nuevos_recuperados bigint	recuperados_acumulados bigint
61	Afghanistan	2020-03-03	61	9	0	5	5	0	0	0	0
62	Afghanistan	2020-03-04	62	9	0	5	5	0	0	0	0
63	Afghanistan	2020-03-05	63	9	0	5	5	0	0	0	0
64	Afghanistan	2020-03-06	64	10	0	5	5	0	0	0	0
65	Afghanistan	2020-03-07	65	10	3	8	8	0	0	0	0
66	Afghanistan	2020-03-08	66	10	0	8	8	0	0	0	0
67	Afghanistan	2020-03-09	67	10	0	8	8	0	0	5	5
68	Afghanistan	2020-03-10	68	10	0	3	8	0	0	0	5
69	Afghanistan	2020-03-11	69	10	3	6	11	0	0	0	5
70	Afghanistan	2020-03-12	70	10	0	6	11	0	0	0	5
71	Afghanistan	2020-03-13	71	11	0	6	11	0	0	0	5
72	Afghanistan	2020-03-14	72	11	3	9	14	0	0	0	5
73	Afghanistan	2020-03-15	73	11	6	15	20	0	0	0	5
74	Afghanistan	2020-03-16	74	11	5	20	25	0	0	0	5
75	Afghanistan	2020-03-17	75	11	1	21	26	0	0	0	5
76	Afghanistan	2020-03-18	76	11	0	21	26	0	0	0	5
77	Afghanistan	2020-03-19	77	11	0	21	26	0	0	0	5

Figura 3: Tabla poblacion en PostgreSQL


proyecto/postgres@ServerG6

Query Editor | Query History

```
SELECT * FROM hito23.avance_de_vacunacion
ORDER BY nombre_pais ASC, fecha ASC
```

Data Output | Explain | Messages | Notifications

	nombre_pais [PK] character varying (255)	fecha [PK] date	personas_esquema_vacunación_completo bigint	personas_vacunadas bigint
1	Afghanistan	2021-05-11	55624	448878
2	Afghanistan	2021-05-20	77560	470341
3	Afghanistan	2021-05-24	96910	476367
4	Afghanistan	2021-05-26	111082	479372
5	Afghanistan	2021-05-27	113739	479574
6	Afghanistan	2021-05-30	119926	480226
7	Afghanistan	2021-06-02	144600	481690
8	Afghanistan	2021-06-03	148505	481800
9	Afghanistan	2021-06-08	158343	482952
10	Afghanistan	2021-06-14	177266	484737
11	Afghanistan	2021-06-22	183762	582128
12	Afghanistan	2021-06-27	186260	649434
13	Afghanistan	2021-06-30	187654	699200
14	Afghanistan	2021-07-05	189322	726349
15	Afghanistan	2021-07-07	199250	735213
16	Afghanistan	2021-07-11	219159	742934
17	Afghanistan	2021-08-20	430744	770542

Figura 4: Tabla avance de vacunación en PostgreSQL

proyecto/postgres@ServerG6

Query Editor | Query History | Scratc

```
1  SELECT * FROM hito23.poblacion
2  ORDER BY nombre_pais ASC
```

Data Output | Explain | Messages | Notifications

	nombre_pais [PK] character varying (255)	superficie_del_pais double precision	cantidad_de_habitantes bigint	densidad_d double prec	pib_per_capita double precisic	crecimiento_pib double precisio	desempleo double prec	crecimiento_poblaci double precision	porcentaje_población_urbana double precision	indice_de_calidad_de_vida double precision
1	Argentina	2780400	44271	16.2	14564.5	2.4	6.5	1	91.8	139.59
2	Australia	7692060	24451	3.2	51352.2	2.4	5.5	1.5	89.4	176.54
3	Austria	83871	8736	106	44117.7	1	6.2	0.6	66	190.37
4	Belarus	207600	9468	46.7	5750.8	-3.9	0.5	0	76.7	119.23
5	Belgium	30528	11429	377.5	40277.8	1.5	8.3	0.6	97.9	160.52
6	Bosnia and Herzegovina	51209	3507	68.8	4265	3.1	25.4	-1	39.8	139.14
7	Brazil	8515767	209288	25	8528.3	-3.8	12.4	0.9	85.7	96.15
8	Bulgaria	111002	7085	65.3	6846.8	3	8.1	-0.6	73.9	138.2
9	Canada	9984670	36624	4	43205.6	0.9	7.1	1	81.8	167.18
10	Chile	756102	18055	24.3	13416.2	2.3	6.8	0.9	89.5	136.2
11	China	9600000	1409517	150.1	8109.1	6.9	4.6	0.5	55.6	90.95
12	ChinaHK	1106	7365	7014.2	42431	2.4	3.5	0.6	100	94.45
13	Colombia	1141748	49066	44.2	6056.1	3.1	10.5	1	76.4	115.38
14	Croatia	56594	4189	74.9	11479.4	1.6	11.7	-0.4	59	170.63
15	Cyprus	9251	1180	127.7	21941.9	1.7	10.3	0.9	66.9	166.64
16	Czechia	78868	10618	137.5	17561.7	4.5	3.9	0.1	73	165.41
17	Denmark	42921	5734	135.1	53149.3	1.6	6	0.5	87.7	184.92
18	Egypt	1002000	97553	98	3452.3	4.2	11.5	2.2	43.1	91.81

Figura 5: Tabla de datos Socioeconómicos en PostgreSQL

Índices ocupados

Para mejorar la eficiencia de las consultas, se utilizó un arbol B+ que permite consultar de manera eficientes los trios de datos. Logrando los siguientes resultados de eficiencia para cada consulta:

	QUERY PLAN text
1	Sort (cost=10638.08..10638.67 rows=236 width=23) (actual time=55.222..58.566 rows=175 loops=1)
2	[...] Sort Key: (min(fecha))
3	[...] Sort Method: quicksort Memory: 38kB
4	[...] -> Finalize GroupAggregate (cost=10567.81..10628.78 rows=236 width=23) (actual time=54.873..58.510 rows=175 loops=1)
5	[...] Group Key: nombre_pais
6	[...] -> Gather Merge (cost=10567.81..10622.88 rows=472 width=23) (actual time=54.868..58.412 rows=312 loops=1)
7	[...] Workers Planned: 2
8	[...] Workers Launched: 2
9	[...] -> Sort (cost=9567.79..9568.38 rows=236 width=23) (actual time=23.666..23.671 rows=104 loops=3)
10	[...] Sort Key: nombre_pais
11	[...] Sort Method: quicksort Memory: 37kB
12	[...] Worker 0: Sort Method: quicksort Memory: 31kB
13	[...] Worker 1: Sort Method: quicksort Memory: 31kB
14	[...] -> Partial HashAggregate (cost=9556.12..9558.48 rows=236 width=23) (actual time=23.320..23.339 rows=104 loops=3)
15	[...] Group Key: nombre_pais
16	[...] Batches: 1 Memory Usage: 45kB
17	[...] Worker 0: Batches: 1 Memory Usage: 45kB
18	[...] Worker 1: Batches: 1 Memory Usage: 45kB
19	[...] -> Parallel Seq Scan on seguimiento_de_casos (cost=0.00..9343.79 rows=28311 width=23) (actual time=9.952..18.194 rows=22585 loops=3)
20	[...] Filter: (casos_acumulados > 10000)
21	[...] Rows Removed by Filter: 29178
22	Planning Time: 0.146 ms
23	Execution Time: 58.631 ms

Figura 6: Consulta sin los indices

	QUERY PLAN text
1	Sort (cost=5737.81..5738.40 rows=236 width=23) (actual time=16.779..16.790 rows=175 loops=1)
2	[...] Sort Key: (min(fecha))
3	[...] Sort Method: quicksort Memory: 38kB
4	[...] -> GroupAggregate (cost=0.42..5728.51 rows=236 width=23) (actual time=0.142..16.734 rows=175 loops=1)
5	[...] Group Key: nombre_pais
6	[...] -> Index Only Scan using indice_seguimiento on seguimiento_de_casos (cost=0.42..5216.55 rows=67947 width=23) (actual time=0.019..10.2...
7	[...] Index Cond: (casos_acumulados > 10000)
8	[...] Heap Fetches: 0
9	Planning Time: 0.105 ms
10	Execution Time: 16.825 ms

Figura 7: Consulta con los indices segun B+

Consultas realizadas

A continuación se presentan los códigos de las consultas realizadas:

Código 11: Consulta Muertes/PIB

```
SELECT c1.pais, pais.nombre_continente, c1.muertes, c1.densidad, c1.porcentaje_muertes, c1.pib
FROM hito23.pais AS pais JOIN
```

```
(SELECT p.nombre_pais AS pais, s.muertes_acumuladas AS muertes, p.densidad_de_poblacion AS densidad,
 (((s.muertes_acumuladas*100000/p.cantidad_de_habitantes)::FLOAT)/1000000) AS porcentaje_muertes, p.pib_per_capita AS pib
FROM hito23.poblacion AS p JOIN hito23.seguimiento_de_casos AS s
   ON p.nombre_pais = s.nombre_pais AND s.fecha = TO_DATE('2021-10-21', 'YYYY-MM-DD')) AS c1 ON pais.nombre = c1.pais
ORDER BY c1.porcentaje_muertes DESC
```

Código 12: Consulta Vacunación/PIB

```
SELECT c1.pais, pais.nombre_continente, c1.PIB, c1.porcentaje_vacunados
FROM hito23.pais AS pais JOIN
(SELECT p.nombre_pais AS pais, p.pib_per_capita AS PIB,
 ((a.personas_esquema_vacunación_completo/p.cantidad_de_habitantes)/10) AS porcentaje_vacunados
FROM hito23.poblacion AS p JOIN hito23.avance_de_vacunacion AS a
   ON p.nombre_pais = a.nombre_pais AND a.fecha = TO_DATE('2021-10-21', 'YYYY-MM-DD')) AS c1
   ON pais.nombre = c1.pais
ORDER BY c1.porcentaje_vacunados DESC
```

Código 13: Consulta vacunado, recuperados e índices socioeconómicos

```
SELECT vr.nombre, (((vr.vacunados*100000/pob.cantidad_de_habitantes)::FLOAT)/1000000) AS porcentaje_vacunados,
                (((vr.recuperados::FLOAT/vr.casos::FLOAT))*100) AS porcentaje_recuperados,
                pob.porcentaje_población_urbana, pob.indice_de_calidad_de_vida
FROM hito23.poblacion as pob JOIN
(SELECT s.nombre_pais AS nombre, a.personas_esquema_vacunación_completo AS vacunados,
 (s.recuperados_muertes_acumulados-s.muertes_acumuladas) AS recuperados, s.casos_acumulados AS casos
FROM hito23.seguimiento_de_casos AS s JOIN hito23.avance_de_vacunacion AS a
   ON s.nombre_pais = a.nombre_pais AND  s.fecha = TO_DATE('2021-09-21', 'YYYY-MM-DD')
   AND  a.fecha = s.fecha) as vr
   ON vr.nombre = pob.nombre_pais
ORDER BY porcentaje_vacunados DESC
```

Código 14: Consulta fecha de los 10.000 casos

```
SELECT nombre_pais, MIN(fecha) AS fechas, MIN(casos_acumulados) AS casos_acumulados
FROM hito23.seguimiento_de_casos
WHERE casos_acumulados > 10000
GROUP BY nombre_pais
ORDER BY fechas
```

Resultados obtenidos

Luego de obtener los datos, se realizó un análisis enfocado en las relaciones entre datos que podrían tener una mayor relevancia. De este modo, se obtuvieron los siguientes gráficos :

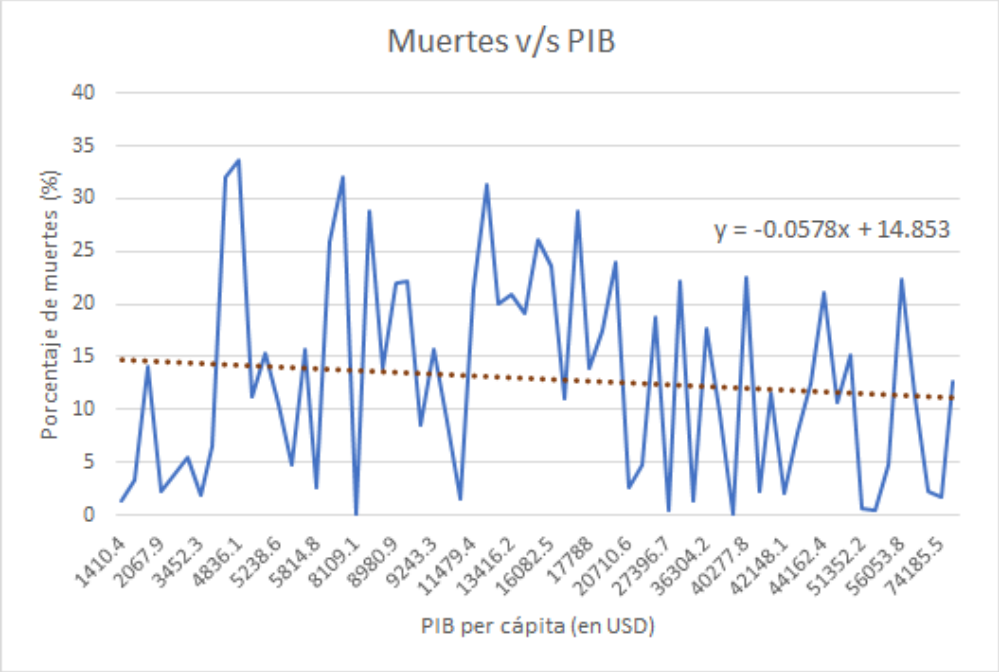


Figura 8: Gráfico Muertes v/s PIB

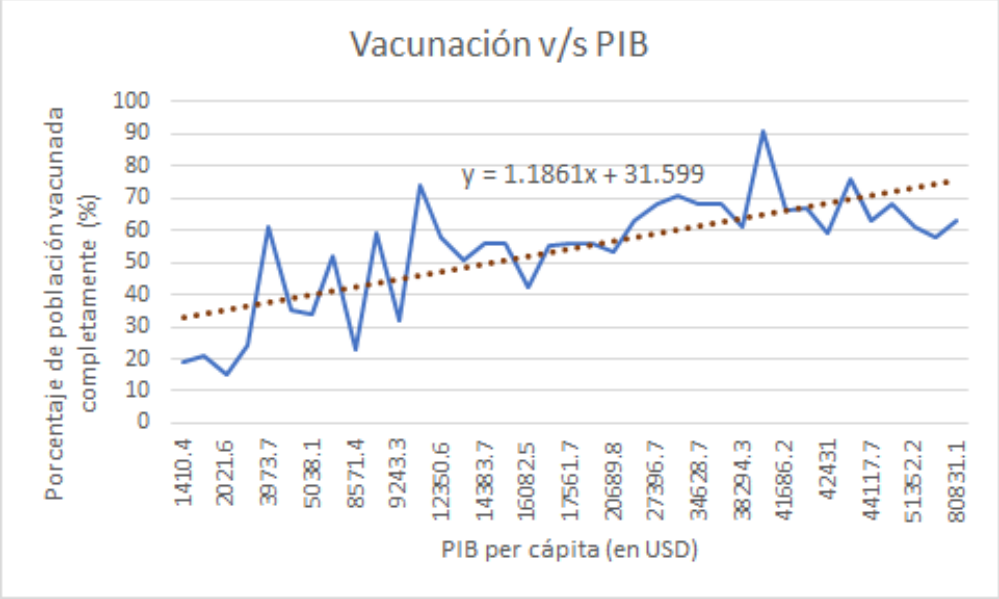


Figura 9: Gráfico Vacunación v/s PIB

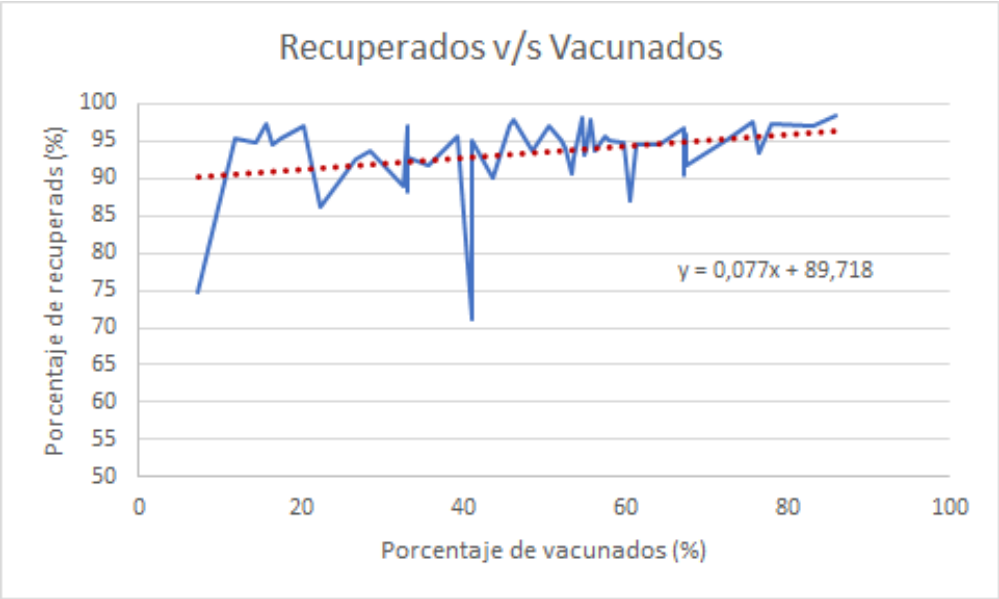


Figura 10: Gráfico Recuperados v/s Vacunados

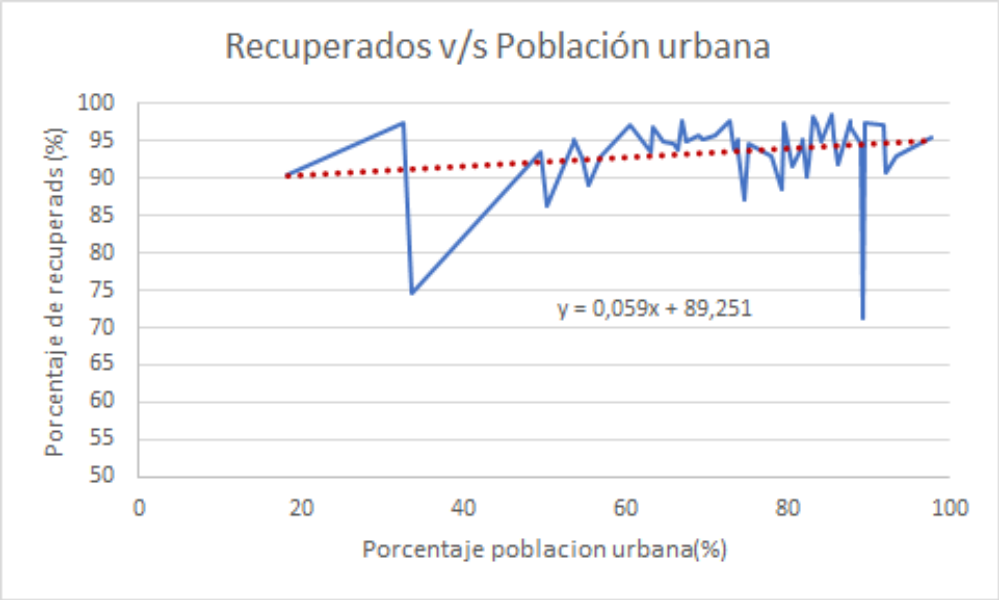


Figura 11: Gráfico v/s Población urbana

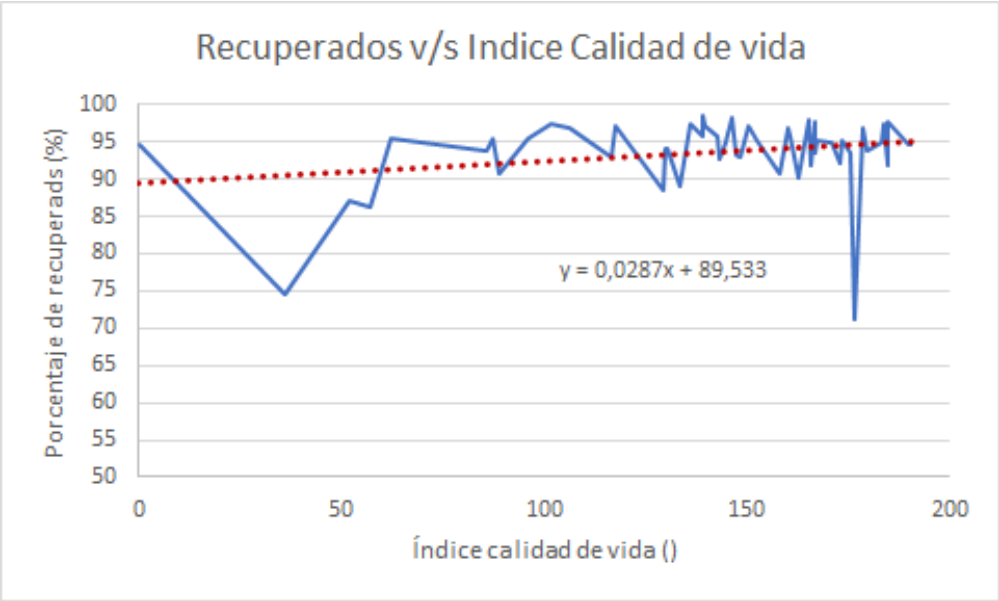


Figura 12: Gráfico Recuperados v/s Índice calidad de vida

Conclusiones

Tras el análisis y trabajo de los datos que conforman los data-sets, se ha logrado demostrar que existe relación entre el PIB de un país y los resultados que tiene este al enfrentarse al COVID-19, hecho que se ve reflejado en los gráficos presentados en la sección anterior. De este modo, se tiene que:

- A medida que aumenta el PIB per cápita de un país, disminuye el porcentaje de muertes por COVID-19.
- A medida que aumenta el PIB per cápita de un país, aumenta también el porcentaje de población vacunada.
- En el último gráfico se puede apreciar que no existe una correlación entre el porcentaje de recuperados y el porcentaje de población urbana. De este modo es posible concluir que la densidad demográfica no afecta en el porcentaje de recuperados de COVID-19, como se pensaba en un inicio.

Finalmente, se puede concluir que al trabajar con datos actualizados, con un buen manejo de datos y un estudio de ellos bien enfocado, se llega a comprobar/negar ideas que pueden resultar relevantes para tomar decisiones claves, en este caso nos permitió saber que factores eran determinantes al momento de manejar la pandemia del COVID-19.