

Entrega Final – Proyecto Final Ingeniería de Datos

Contenido

Generalidades.....	1
Caso de análisis	2
Reglas de negocio.....	2
Base de datos - Descripción de las columnas.....	3
Diagrama Entidad Relación	6
Diagrama Relacional (Sin normalizar)	7
Diagrama Relacional Normalizado	7
Limpieza de los Datos	10
Base de Datos COLLISION_BD en PostgreSQL	10
Conexión a Python	13
Preliminares a Dash.....	13
Implementación Dash.....	14
Discusión y análisis	15
Conclusiones	18

Generalidades

Las bases de datos trabajadas en nuestro proyecto se encuentran en los siguientes enlaces (1):

- [Motor Vehicle Collisions - Crashes | NYC Open Data \(cityofnewyork.us\)](https://data.cityofnewyork.us/Motor-Vehicle-Collisions-Crashes/mhwh-q3wa)
- [Motor Vehicle Collisions - Person | NYC Open Data \(cityofnewyork.us\)](https://data.cityofnewyork.us/Motor-Vehicle-Collisions-Crashes/person-q3wz-q5wz)
- [Motor Vehicle Collisions - Vehicles | NYC Open Data \(cityofnewyork.us\)](https://data.cityofnewyork.us/Motor-Vehicle-Collisions-Crashes/vehicles-q3wz-q5wz)

La base de datos consta de tres tablas donde se almacenan datos sobre las colisiones en Nueva York. La primera tabla, “Crashes” toma en cuenta la información de “Person” y “Vehicle” con tal de dar una visualización resumida de la base.

En nuestro caso, creamos una base de datos independiente en donde se consideraba la información relevante del caso de estudio de las tres tablas de esta base de datos. Esta base fue llamada COLLISION_DB y podemos encontrar todos los recursos usados en el GitHub. A lo largo del reporte describiremos y aclararemos la información contenida en el repositorio y los pasos para la creación de nuestra base de datos.

A continuación, podemos encontrar la dirección de nuestro repositorio en GitHub. En este repositorio tenemos contenidos los códigos implementados y los archivos correspondientes relacionados con nuestro proyecto. La URL es: <https://github.com/Dafne-Castellanos/Proyecto-Ingenier-a-de-Datos>

Caso de análisis

Hemos sido contratados por un investigador. Nuestro cliente está realizando un estudio de los accidentes vehiculares en la ciudad de Nueva York para implementar medidas que reduzcan la accidentalidad y crear programas de seguridad vial. Hoy en día la ciudad de Nueva York cuenta con ocho millones de habitantes y una superficie de 789 km², por dicha razón para realizar un mejor análisis de los datos el cliente desea sectorizar la ciudad por los cinco distritos (Brooklyn, Manhattan, Staten Island, Queens y El Bronx). Frente a esto, nos ha solicitado tener ciertas visualizaciones de la información a partir de los datos para analizar los siguientes casos:

1. La situación sanitaria mundial del COVID-19 impacto la movilidad en las distintas ciudades del mundo, por ello, el cliente desea saber las consecuencias de la pandemia en la accidentalidad de Nueva York, así que se busca identificar cuáles son las horas del día en que suceden más colisiones y el comportamiento de los siniestros en los distintos distritos, para ello es necesario revisar los registros de antes, durante y después de la cuarentena.
2. El cliente para la creación de un programa piloto de prevención necesita analizar cuál de los distritos ha tenido la mayor tasa de accidentalidad teniendo en cuenta la población y el área de cada uno.
3. El cliente también quiere saber cuáles son los sexos y los tipos de las personas que están involucradas en los accidentes, pues necesita guiar los programas de prevención a poblaciones específicas dentro de la ciudad de Nueva York.
4. El cliente desea conocer cuáles son los tipos de vehículos que ocasionan más accidentalidad y los daños más sufridos por los vehículos, para así implementar medidas que ayuden a discontinuar motorizados que sean un peligro potencial para los ciudadanos y que ayuden a proteger los automóviles de accidentes fatales.

En este sentido, el cliente provee las bases de datos de las colisiones vehiculares en la ciudad de Nueva York (1).

Reglas de negocio

De acuerdo con el contexto, a continuación, se muestra el planteamiento de nuestras reglas de negocio.

- Varias colisiones pueden involucrar muchas personas.
- En un distrito pueden ocurrir muchas colisiones.
- Muchos vehículos pueden chocar en varias colisiones.
- En varios vehículos pueden ir muchas personas.
- Las colisiones se deben identificar con un número.
- Al cliente le interesa saber la fecha y la hora en que ocurrió el accidente.

- Cada distrito se debe identificar con su nombre.
- De los distritos, al cliente le interesa saber la población y el área.
- De las colisiones se debe tener el zip code y la localización que está compuesta de longitud y latitud, dichos datos se ubican en un distrito.
- Si se tiene hasta n tipos de vehículos, entonces los tipos de vehículos que son antecesores deben tener registro.
- Los registros de personas deben tener un identificador único.
- Los números de personas fallecidas y heridas en las colisiones deben ser mayores o iguales a 0.
- Las personas deben tener un número que las identifique, por ejemplo, la cedula.
- Las personas pueden ser peatones, ciclistas, ocupantes del vehículo u otro motorizado.
- Las personas pueden estar heridas o haber muerto o no tener una especificación.
- Se debe conocer el género de las personas.
- De las personas se debe conocer su edad.
- Los registros de vehículos deben tener un identificador único.
- Los vehículos deben tener un número que los identifique, por ejemplo, la placa.
- Se deben contar con el tipo de los vehículos.
- Se debe tener los daños de los vehículos.
- Se pueden consignar hasta cuatro daños de un vehículo.

Base de datos - Descripción de las columnas

En el proyecto estamos trabajando con cuatro bases de datos; tres de ellas proveen los registros de las colisiones de vehículos motorizados, las personas y los vehículos involucrados, dichos datos son recopilados por el Departamento de Policía de Nueva York con el objetivo de realizar análisis detallados de seguridad de tráfico. La base de datos restante proporciona las cifras de la población y el área de cada distrito, se incluyó pues era necesaria para sectorizar la información. Esta última se realizó con base en el siguiente enlace:

- <https://www.citypopulation.de/en/usa/newyorkcity/>

De esta forma, tenemos las siguientes bases de datos:

- Motor Vehicle Collisions – Crashes
- Motor Vehicle Collisions – Person
- Motor Vehicle Collisions – Vehicles
- Borough

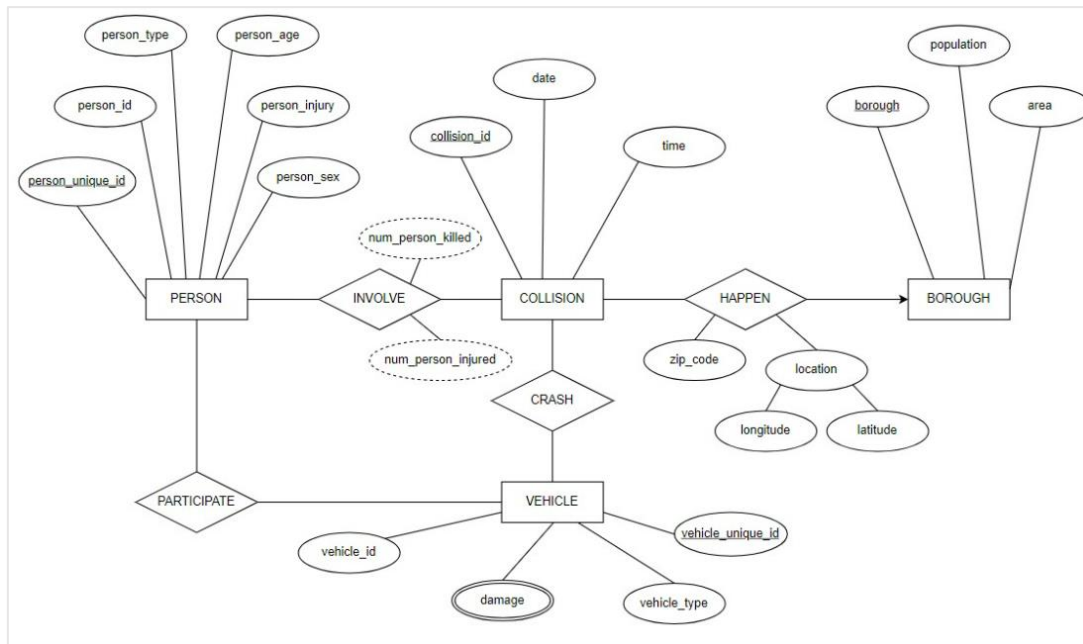
Las tres primeras bases tienen en total 75 columnas, en las cuales se destacan `collision_id`, `person_unique_id` y `vehicle_unique_id`, pues son las llaves primarias. No obstante, por los casos de análisis planteados anteriormente decidimos tomar las columnas necesarias que provean datos relevantes para las consultas que vamos a realizar. A continuación, enunciamos dichas columnas, la base de datos a la que pertenecen, su descripción y las razones por las que consideramos tomarla:

Base de datos	Nombre columna	Descripción	Razones
Motor Vehicle Collisions – Crashes	collision_id	Código de registro único generado por el sistema.	Se incluyo pues es la llave primaria de COLLISION.
	accident_date	Fecha de ocurrencia de la colisión.	Se necesita para saber cuándo ocurrieron las colisiones.
	accident_time	Hora de ocurrencia de la colisión.	Se necesita para saber las horas de las colisiones.
	zip code	Código postal de ocurrencia del incidente.	El zip code ayuda a saber la ubicación del accidente en caso de no tener el distrito.
	latitude	Coordenada de latitud para el Sistema Global de Coordenadas, WGS 1984, grados decimales (EPSG 4326).	La latitud ayuda a ubicar el accidente en caso de no tener el distrito.
	longitude	Coordenada de longitud para el Sistema Global de Coordenadas, WGS 1984, grados decimales (EPSG 4326).	La longitud ayuda a ubicar el accidente en caso de no tener el distrito.
	location	Par de longitud y latitud.	Es el par formado por longitud y longitud.
	number of persons injured	Número de personas lesionadas.	Se necesita para el número de víctimas heridas.
	number of persons killed	Número de personas fallecidas.	Se necesita para saber el número víctimas fatales.
Motor Vehicle Collisions – Person	unique_id	Código de registro único generado por el sistema.	Se incluyo pues es la llave primaria de la tabla PERSON.
	person_id	Código de identificación de la víctima asignado por el sistema.	Se necesita pues el id que identifica a cada persona (ejemplo: cédula).
	person_type	Ciclista, Ocupante, Peatón, etc.	Se necesita para saber que tipos de personas están involucradas en accidentes.
	person_injury	Heridos, fallecidos, no especificados.	El estado de las víctimas ayuda a

			generar nuevos análisis respecto a las personas involucradas.
	person_age	Edad de la víctima	Las edades ayudan a generar nuevos análisis respecto a las personas involucradas.
	person_sex	Género de la víctima	Los sexos ayudan a generar nuevos análisis respecto a las personas involucradas.
Motor Vehicle Collisions – Vehicles	unique_id	Código de registro único generado por el sistema	Se incluyó pues el a llave primaria de VEHICLE.
	vehicle_id	Código de identificación del vehículo asignado por el sistema	Se necesita pues la forma en cómo se identifica cada vehículo (ejemplo: la placa).
	vehicle_type	Tipo de vehículo basado en la categoría de vehículo seleccionada (sedan, SUV, taxi, moto, etc)	Se necesita para saber los tipos de vehículos involucrados en los accidentes.
	vehicle_damage	Ubicación en el vehículo donde ocurrió la mayor parte del daño	Se incluyó pues ayuda a identificar las partes más dañadas para buscar como protegerlas.

Diagrama Entidad Relación

Podemos ver el planteamiento de nuestro modelo entidad relación:



Como mencionamos, los atributos y entidades creadas, se relacionan con nuestro caso de estudio y con el objetivo planteado por el cliente. Por tanto, no se encuentra la totalidad de las columnas de la base de datos (1). Hicimos uso de las necesarias para cumplir con las visualizaciones de los casos de análisis.

Teniendo en cuenta un diagrama Entidad-Relación se procede a su transformación al Diagrama Relacional de la siguiente manera. Toda entidad fuerte pasa a ser una tabla con su misma llave primaria. Al haber entidades débiles se creó otra tabla cuya llave primaria está conformada por la misma y el discriminador que la conforma. Todos los atributos univaluados, simples o base que se encuentren se convierten en columnas de la tabla formada por su entidad.

Las llaves primarias se reconocen por la palabra subrayada. Un atributo compuesto pasa a ser columna en el relacional. Mientras que un multivaluado se convierte en una tabla parte para formar atomicidad por medio de una llave primaria compuesta por la misma y el atributo multivaluado. Cualquier atributo derivado se omite en el Diagrama Relacional.

Para relaciones 1:1 se busca la posibilidad de una función. Cuando está es de 1:n se pasa igual solo que si hay atributos de la relación, estos se mueven a la tabla de :n; y si es m:n se crea una tercera tabla entre las entidades con las llaves de los dos lados relación y se decide como van las relaciones entre ellas, ya sea de 1 a muchos entre la entidad y la nueva tabla creada en medio o viceversa.

Nota: Se deben usar los mismos nombres en ambos modelos.

Diagrama Relacional (Sin normalizar)

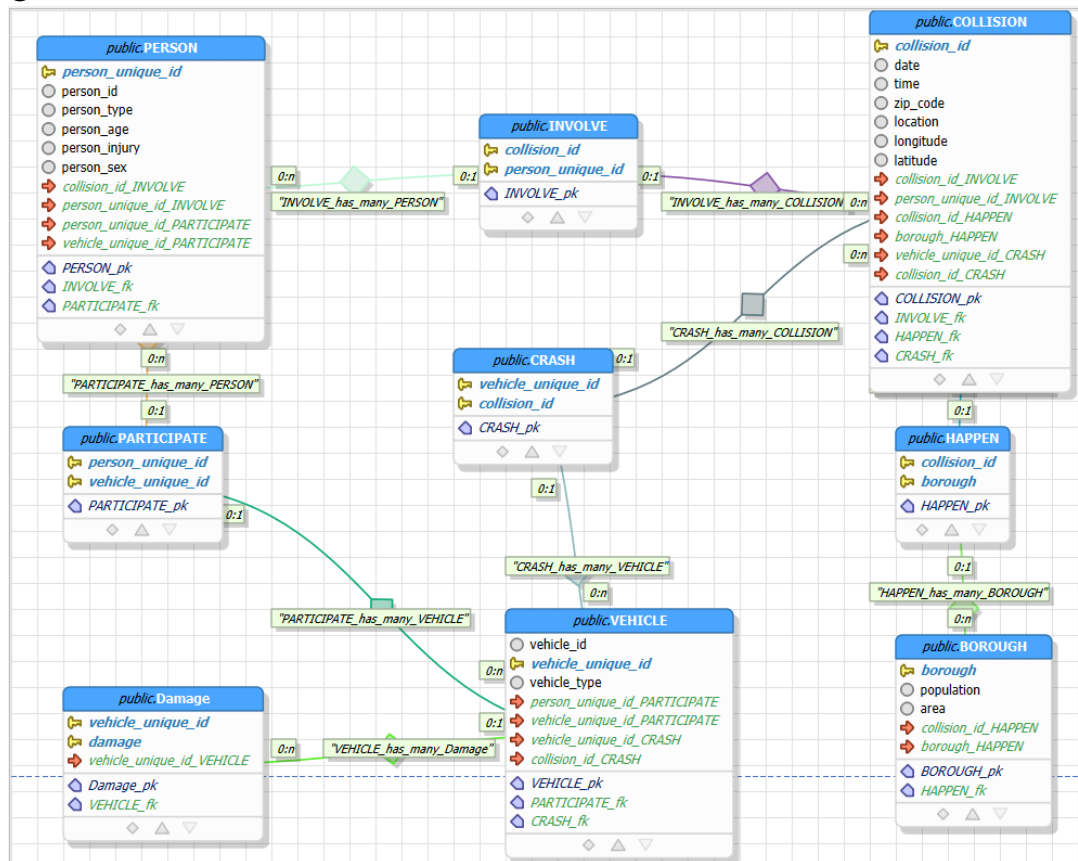
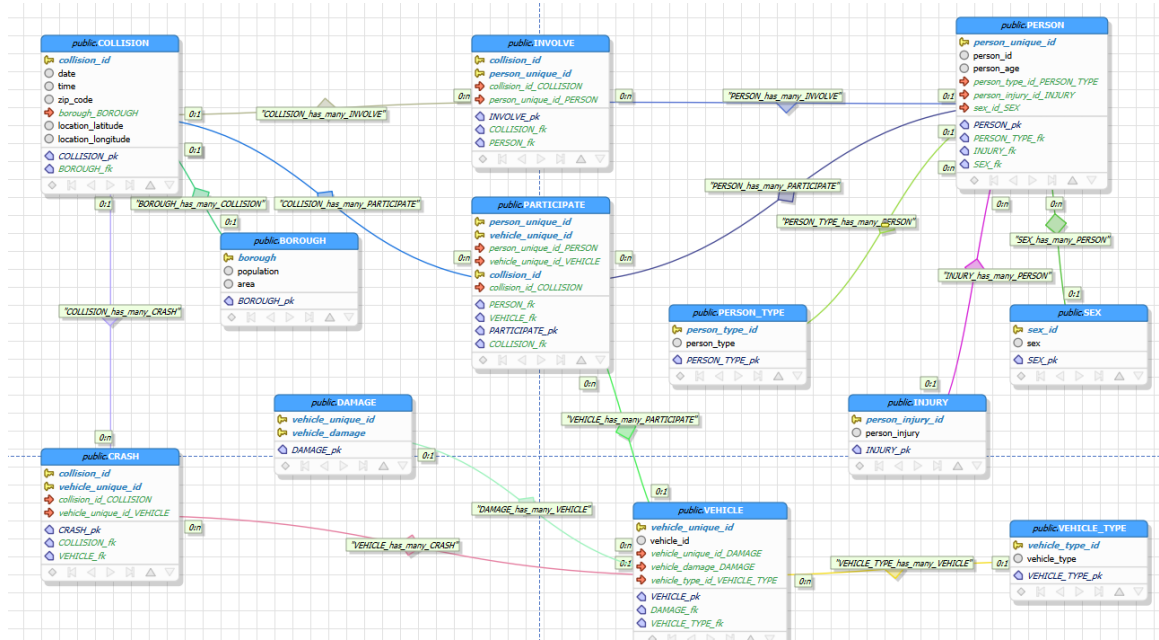


Diagrama Relacional Normalizado



Teniendo en cuenta nuestro diagrama ER, y diagrama relacional realizamos nuestro diagrama relacional normalizado:

En este caso, tuvimos en cuenta los requisitos para llegar a la tercera forma normal. Por lo tanto, normalizamos la siguiente información:

1. De la tabla PERSON, derivamos las tablas: SEX, INJURY y PERSON_TYPE.
2. De la tabla VEHICLE, derivamos las tablas: VEHICLE_TYPE y DAMAGE.

Seleccionamos estas tablas debido a que recaían en una redundancia horizontal. Además, se tenían anomalías en el caso de querer actualizar, borrar o insertar información, pues por alto volumen de datos disponibles en las tablas la modificación manual no es óptima y de igual forma, no se soportaba la integridad de los datos en caso realizar alguna modificación en alguno de los campos.

Ahora, para la creación de estas tablas, en cada una de ellas, generamos un identificador único que tenía una referencia a la tabla de principal (PERSON o VEHICLE), una referencia su nombre (SEX, INJURY, PERSON_TYPE, VEHICLE_TYPE o DAMAGE) y el numero de la categoría. Por ejemplo, en el caso de VEHICLE_TYPE sus identificadores se realizaron de la siguiente manera. Tomamos en cuenta V como referencia a la tabla VEHICLE, T como el de VEHICLE_TYPE y el numero correspondiente a la categoría. Así los identificadores son de la forma VT01, VT02, etc.

De esta forma los identificadores de estas tablas tienen la siguiente forma:

Tabla principal	Tabla creada	Forma del identificador
PERSON	SEX	PS##
	INJURY	PI##
	PERSON_TYPE	PT##
VEHICLE	VEHICLE_TYPE	VT##
	DAMAGE	VD##

Las categorías tomadas en cuenta se realizaron posteriormente de hacer una limpieza de los datos. Así pues, las tablas de resultantes de la normalización son las siguientes:



- SEX

	sex_id [PK] character varying (6)	sex character varying
1	PS01	F
2	PS02	M
3	PS03	U
4	PS04	nd



- INJURY

	person_injury_id [PK] character varying (6)	person_injury character varying
1	PI01	Unspecified
2	PI02	Injured
3	PI03	Killed



- PERSON_TYPE

	person_type_id [PK] character varying (6) 	person_type character varying 
1	PT01	Occupant
2	PT02	Pedestrian
3	PT03	Bicyclist
4	PT04	Other Motorized

- VEHICLE_TYPE

	vehicle_type_id [PK] character varying (6) 	vehicle_type character varying 
1	VT01	Sedan
2	VT02	SUV
3	VT03	nd
4	VT04	Taxi
5	VT05	Pickup
6	VT06	Van
7	VT07	Bus
8	VT08	Truck
9	VT09	Bike
10	VT10	Moto
11	VT11	Ambulance
12	VT12	Otros

- DAMAGE

	vehicle_damage_id [PK] character varying (6) 	vehicle_damage character varying 
1	VD01	Center Front End
2	VD02	Left Front Bumper
3	VD03	Center Back End
4	VD04	Right Front Bumper
5	VD05	No Damage
6	VD06	Left Front Quarter Panel
7	VD07	Right Front Quarter Panel
8	VD08	Left Rear Quarter Panel
9	VD09	Left Side Doors
10	VD10	Left Rear Bumper
11	VD11	Right Side Doors
12	VD12	Right Rear Quarter Panel
13	VD13	Right Rear Bumper
14	VD14	Other
15	VD15	Roof
16	VD16	Demolished
17	VD17	Trailer
18	VD18	Overtured
19	VD19	Undercarriage
20	VD20	nd

Limpieza de los Datos

De los conjuntos de datos originales, que cuentan con información diaria, la idea era contar con un horizonte de análisis que abarcara más de un (1) año, y para poder hacer algunas comparaciones entre años se decidió tomar información para tres (3) años.

Dado que la información de “vehículos” incluía información hasta los primeros días de dic-21, se determinó que el horizonte total de análisis estaba comprendido entre el 1-dic-2018 y el 30-nov-2021.

Adicionalmente, se realizaron actividades de depuración de información, principalmente asociados a datos faltantes de varios registros tanto de accidentes, como de personas y vehículos involucrados. Esto, con el fin de poder hacer la normalización de la información de manera efectiva, y poder consolidar un modelo de base de datos eficiente.

Teniendo en cuenta lo anterior, la información a ser trabajada se puede resumir en las siguientes cifras de las principales entidades:

- 282.135 accidentes analizados
- 956.437 personas relacionadas a dichos accidentes
- 508.502 vehículos involucrados

Base de Datos COLLISION_BD en PostgreSQL

Con el diagrama normalizado, realizamos la creación de las tablas de manera que se respetara la información consignada en el modelo. De esta manera, en el archivo tables o “tables ddl.txt” en el GitHub se puede observar la creación de la base de datos junto con las tablas.

Nota: Si se llega a tener alguna duda en el proceso de cargado de los datos en PostgreSQL o la conexión con Python o con alguno de los pasos que se describirán a continuación, se sugiere consultar el siguiente video: [video instructivo](#)

En el video va a encontrar como descargar los archivos del repositorio GitHub, como exportarlos, como pasarlos a la unidad C:, como correr en PostgreSQL la creación de las tablas y como insertar los datos en ellas. Además, de cómo realizar la conexión con Python y como ejecutar las consultas respectivas.

El acceso a esta información se obtiene con los siguientes pasos:

1. Ingresar al enlace de GitHub: <https://github.com/Dafne-Castellanos/Proyecto-Ingenieria-de-Datos>
2. Descargar o abrir el archivo de texto “tables ddl.txt”

Para continuar con la creación del DDL de la base debemos:

3. Ingresar a PostgreSQL
4. Correr el código contenido de “tables ddl.txt” en un script de PostgreSQL

Con esto vamos a tener la creación de la base de datos COLLISION_DB con sus respectivas tablas.

Ahora bien, para el cargue de la información de la base de datos vamos a hacer lo siguiente:

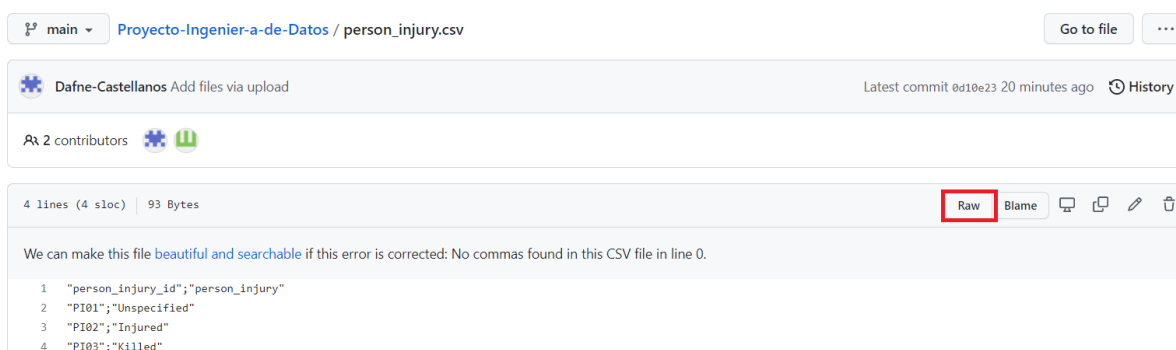
1. Crear una carpeta llamada "dataDB" en la unidad C: de su equipo.
2. Descargar los siguientes archivos csv y rar

De esta descarga estos archivos ya están listos para insertar en PostgreSQL:

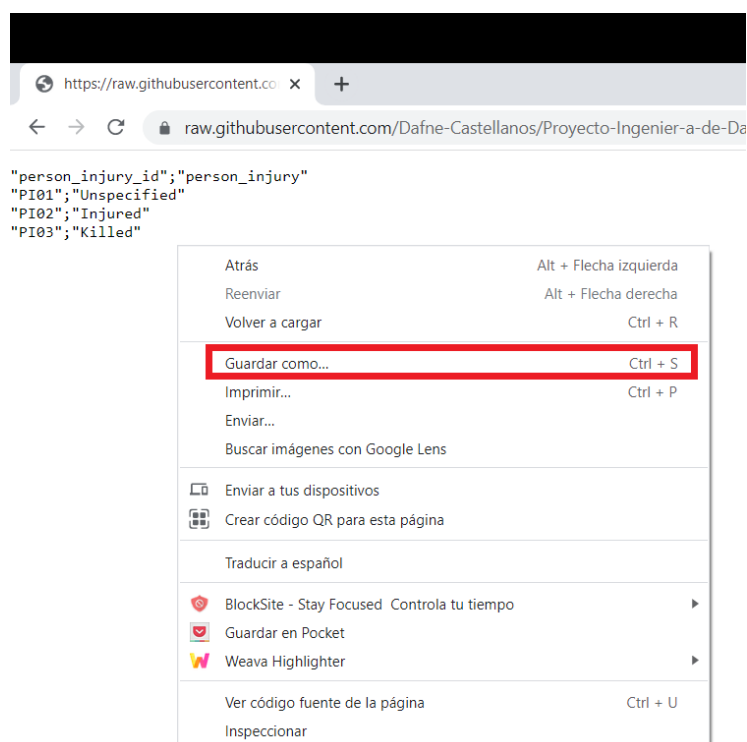
- a. person_injury.csv
- b. person_sex.csv
- c. person_type.csv
- d. vehicle_damage.csv
- e. vehicle_type.csv

La descarga de estos archivos se debe hacer de la siguiente forma:

1. Oprimir el boton raw



2. Clic derecho y Guardar como, luego selecciona la unidad C:

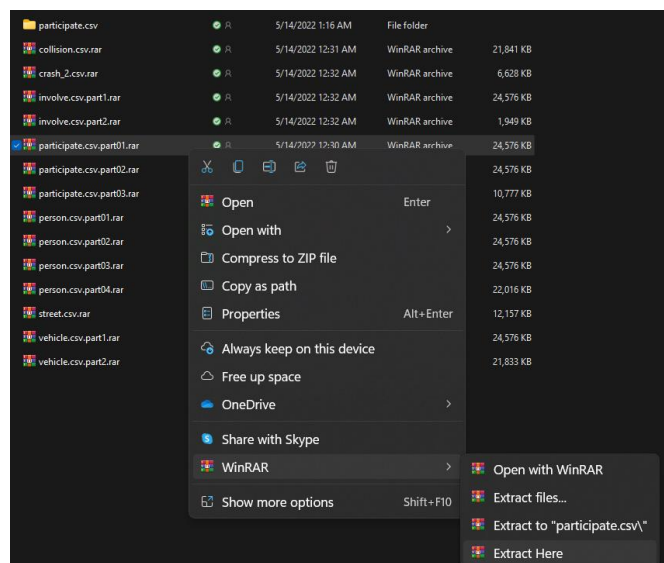


Los demás, son RAR por tanto debemos tomar en cuenta el siguiente procedimiento:

1. Clic derecho a una carpeta rar

2. Seguir la ruta WinRAR\ExtractHere

Un ejemplo de lo anterior se realizó con participate.csv.rar

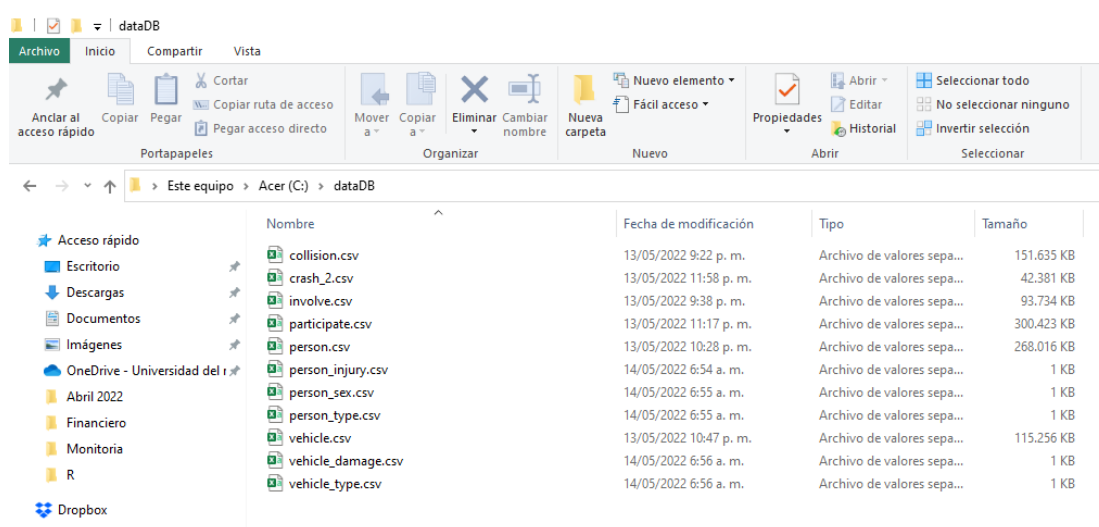


Esto lo haremos para:

- collision.csv.rar
- crash2.csv.rar
- involve.csv.rar
- participate.csv.rar
- person.csv.rar
- vehicle.csv.rar

Nota: Se deben descargar todos los archivos rar, no obstante, solo es necesario extraer la parte 1 de cada uno de ellos.

De esta manera, tenemos en nuestra carpeta C:\dataDB todos los archivos csv necesarios.



Ahora, continuamos con

3. Descargar o abrir el archivo de texto “insertion.txt”

4. Ingresar a PostgreSQL
5. Correr el código contenido de “insertion.txt” en un script de PostgreSQL

Al ejecutar este último código, se deben realizar las inserciones de los datos a las tablas y es así como pasamos nuestra información y base de datos a PostgreSQL.

Conexión a Python

Antes de realizar la conexión con Python, es necesario tener instalado el paquete psycopg2, si no se tiene instalado recomendamos ver la documentación que se encuentra en el siguiente enlace: <https://pypi.org/project/psycopg2/>

Luego de tener instalado psycopg2 podemos realizar los siguientes pasos para realizar y visualizar las consultas:

1. Descargar el archivo Conexion_consulta.py que se encuentra en el GitHub.
2. Enviar el archivo Conexion_consulta.py a la carpeta dataDB que está en la unidad C: creada en los anteriores pasos.
3. Abrir el símbolo del sistema y correr las siguientes líneas (una por una):

```
py -m venv prueba
cd prueba
cd Scripts
activate.bat
```

4. Correr esta línea: `py C:\dataDB\Conexion_Consulta.py`

De esta forma, tendremos la conexión de nuestra base de datos con Python.

Preliminares a Dash

Continuando con la línea de comandos anterior y antes de realizar la página en dash, es necesario tener instalado el mismo dash, si no se tiene instalado recomendamos ver la documentación que se encuentra en el siguiente enlace: <https://dash.plotly.com/installation>

Luego de tener instalado dash hay que instalar los siguientes paquetes de la misma forma:

- dash-renderer
- dash-html-components
- dash-core-components
- plotly

Y rectificar que si está las librerías:

- pandas

Por ejemplo:

```
>\[RUTA]>pip3 install dash dash-renderer dash-html-components dash-core-components plotly pandas
```

Nota: es recomendable hacerlo uno por uno

Luego de tener instalado instalados todos los paquetes procedemos con los siguientes pasos

5. Descargar los archivos Conexión_consulta.py, project.py y projectSQL.py que se encuentran en la carpeta “implementacionDASH” del GitHub.
6. Enviar los archivos a la carpeta dataDB que está en la unidad C: creada anteriormente.
7. Abrir el símbolo del sistema y correr la siguiente línea:

```
py C:\dataBD\project.py
```

5. Entre varias líneas de código, debería aparecer lo siguiente, allí copias el enlace <http://127.0.0.1:8050/>:

```
Dash is running on http://127.0.0.1:8050/

* Serving Flask app 'project' (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: on
```

6. Esta línea se pega en el navegador y de esta forma se puede ver nuestro dashboard.

Implementación Dash

Teniendo en cuenta cada uno de los escenarios de análisis, se diferentes gráficas como: diagramas de barras, diagramas circulares y diagramas de línea para la visualización de la información que nuestro cliente ha solicitado.

Primero, era necesario realizar las funciones necesarias para que los datos a visualizar correspondieran a lo solicitado por nuestro cliente. Para esto, necesitábamos plantear las consultas SQL que nos dieran la información deseada (ver archivo projectSQL.py), con títulos adecuados y con las agrupaciones pertinentes para que, a la hora de realizar los gráficos en el código, no se presentaran inconvenientes. Por ejemplo, en el primer escenario de análisis, nos es solicitado la visualización del número de accidentes por distrito, seccionando a su vez por los periodos de la pandemia. En este sentido, para facilidad en los posteriores procedimientos, las consultas se realizaron teniendo en cuenta cada uno de los distritos y los respectivos periodos de la pandemia.

Una vez terminadas las consultas SQL, desarrolladas a partir de los escenarios de análisis, se dio paso a la realización de las gráficas para las consultas realizadas. Reiteramos que fue clave confirmar previamente que el planteamiento de las consultas haya sido el indicado para evitar errores. Para tener las gráficas, inicialmente fue necesaria la conexión entre la base de datos en el PostgreSQL y nuestro documento de projectSQL.py donde se dieron las consultas, en el script project.py. Para esto, se creó un dataframe con el uso de la librería de pandas donde se almacenan los datos retornados de la consulta SQL.

Así pues, el código usado a nivel general fue:

```
con = Connection()
con.openConnection()
query = pd.read_sql_query(sql.consultaSQL(), con.connection)
con.closeConnection()
```

Ahora con el dataframe de nuestra consulta es posible hablar sobre la realización de los gráficos. En el caso de los diagramas de barras y de líneas, era importante tener en cuenta, la categoría usada desde la consulta SQL para poder insertarla en el eje x, y considerar el número o la información dependiente de estas categorías para ingresarlos en el eje y.

Por su parte, en los diagramas circulares, era importante considerar de donde se obtenían los datos y cuáles eran los nombres de las categorías para que al implementar el código adecuadamente.

En este sentido, de manera genérica usamos estas líneas de código a lo largo del dashboard:

- `figBar = px.bar(df.head(), x=" ", y=" ")`
- `figPie = px.pie(df.head(), names=" ", values=" ")`
- `figLine = px.line(df, x=' ', y=' ', markers = True)`

Discusión y análisis

1. Número de accidentes antes, durante y después de la pandemia por distrito

Para este caso de estudio se decidió sectorizar los diagramas por distritos, de esta forma se cuenta con un gráfico de líneas y tres diagramas de barras y tortas para evidenciar los tres periodos de tiempo (Pre-Pandemia, En Pandemia y Postpandemia) por cada borough.

- Interpretaciones de los datos a partir de las gráficas

Con los tres tipos de diagramas podemos visualizar para el Bronx que antes de la pandemia la hora que presentaba más accidentalidad era las 16:00 con un récord de 8165 registros. Durante la pandemia vemos que el número de accidentes cayo significativamente para cada hora, sin embargo, se presenta el mismo comportamiento que para antes de la pandemia; las 16:00 siguen siendo uno de los momentos del día con más colisiones y se presenta un mínimo a las 4:00 con 1135. Los registros de después de la pandemia se encuentran por debajo que los de durante, estos presentan el mismo comportamiento observándose el pico más alto a las 16:00 con 3455 colisiones.

El distrito de Brooklyn presenta un comportamiento similar que el Bronx, pues el número más alto de accidentes antes de la pandemia se ubica a las 16:00 con un valor de 15.500 y el registro más bajo ocurre a las 3:00 con 1975. Durante la pandemia ocurre el mismo fenómeno, las 16:00 presentan el récord más alto con 7870, y en el periodo post-pandemia se presenta el pico a las 17:00 con un valor de 6480.

En Manhattan evidenciamos que antes de la pandemia los accidentes ocurrían con una frecuencia más alta, pues existe una diferencia de 5530 registros entre el punto máximo de pre-pandemia y el valor más alto en post-pandemia. Cabe anotar que el comportamiento de la accidentalidad en las distintas horas en pandemia y después de ella es muy similar.

En Queens observamos el mismo comportamiento que para Manhattan, la frecuencia de accidentes es más alta en el periodo pre-pandemia y los picos más altos se encuentran para los tres periodos a las 16:00.

Por último, en Staten Island se presenta el mismo comportamiento que para el resto de los distritos, antes de la pandemia se registraban más accidentes que durante y después. Y se presenta un pico a las 17 con 1195 registros. De esta forma, podemos concluir que para antes de la pandemia en la mayoría de las horas se registraban más accidentes y luego de la aparición del Covid-19 el número de accidentes se redujo considerablemente para los cinco distritos. Además, que en todos los boroughs se presenta un máximo de registros a las 16:00 o 17:00.

- Ventajas y desventajas de cada diagrama

Ahora, en cuanto las ventajas de cada visualización, podemos concluir que para el diagrama de barras podemos evidenciar dichos máximos y mínimos en el número de accidentes de forma más fácil. Además, de contar con la posibilidad de visualizar con mayor simpleza si se presenta algún sesgo en los datos. Lo que ayuda a generar interpretaciones estadísticas para futuros análisis.

En el diagrama de torta notaremos que no es el más apropiado para mostrar la información, pues el alto número de categorías hace difícil diferenciar los valores y las horas, es por esto que algunos académicos recomiendan utilizar los diagramas circulares para máximo seis categorías. Al igual, no se evidencia tan fácilmente los máximos y mínimos.

Por otro lado, el diagrama de líneas presenta una ventaja y es que resume la información de los tres periodos de tiempo en un solo gráfico, lo que facilita las comparaciones entre las variables. Sin embargo, no es tan fácil identificar a simple vista los máximos y los mínimos, pues algunos registros se encuentran en una altura muy similar.

2. Tasas de accidentalidad por Distritos

a. Por población:

Se puede observar que Manhattan tiene la mayor tasa de accidentalidad por habitante, siendo este un indicio que se debería intervenir en el distrito con un proyecto vial de tamaño macro. Su despliegue y prueba puede hacerse en Staten Island ya que allí es en donde hay menos accidentalidad con respecto a la población. También se observa que Brooklyn y Queens manejan estadísticos no muy diferentes, tienen una diferencia del 0.7%.

En el gráfico de torta se podría decir que los datos están divididos de manera semejante, más en el gráfico de líneas aparenta tener una variabilidad más grande.

En este caso manejamos números entre cero y uno, por lo que la gráfica de barras nos muestra un poco más la realidad de los datos, sin embargo, estos mismos datos se despliegan de una manera más amigable para la vista en el gráfico circular, por sus porcentajes. Por último, en el gráfico de líneas hay una desventaja, y es que, si bien buscamos la misma información, éstas no están correlacionadas, por lo que la unión de puntos puede ser engañosa. La ventaja de la herramienta dash en sí es que, al acercar el cursor a una columna o parte de la gráfica, este te mostrara su valor exacto y de donde provino.

b. Por área:

El distrito con mayor accidentalidad por área es Manhattan, este lugar por sí solo duplica a cada otro distrito por aparte, y dado el tamaño de Staten Island, está vez si se ve su gran diferencia significativa en comparación a los demás.

Con una tasa de accidentalidad de casi 5.0 la moda de este gráfico se acerca al 50%, luego le siguen Brooklyn, Bronx, y Queens en ese orden.

Trabajamos con unidades de miles, por lo que es mucho más sencillo el análisis a gran escala. Una ventaja del gráfico de líneas es que gracias al volumen de datos esté sale similar al de barras. Y una pequeña desventaja de la gráfica circular es que cuando un sector es muy grande, tiende a confundir a las personas, dando pie a especulaciones exageradas que no toman los demás distritos.

3. Tipos y sexos de las personas involucradas en accidentes

a. Tipos de persona:

Casi todas las personas involucradas son ocupantes, esto puede darse por la inclusión de la tabla vehículo que prioriza estos tipos de accidentes en la base de datos, si bien aparece un 1% de peatones, estos son muy pocos. Y puede darse el caso que estén involucrados para que el seguro del vehículo les cubra un valor monetario, con un 0.95% los ciclistas ocupan el tercer lugar, aquí puede haber un error más de registro, pues no es muy común que una persona que transite en bicicleta se vincule formalmente a un accidente, normalmente solucionan de manera informal o ponen quejas verbales, no aptas en nuestros registros.

Tanto grafica de barras como grafica circular tienen la desventaja que no permite visualizar la cantidad de “Otros motorizados” gracias a la abrupta suma de casi 98% que tiene los ocupantes en este tipo.

b. Sexos de las personas:

Los hombres son tres quintos de la población involucrada en los accidentes, sin ánimo de discriminación o inequidad, se ve más el caso donde los conductores son hombres, por eso puede darse está sutil diferencia con las mujeres, que ocupan la no despreciable sima del 20% de involucrados.

Podemos discutir que el 10% de las personas involucradas en accidente son de sexo indefinido. Hoy en día no existen investigaciones si esto afecta o no la conducción o movilidad de una persona por lo cual si se quiere discernir más en el tema se requiere de otro tipo de investigación y una más profunda.

Usando millones de unidades ambos gráficos tienen la ventaja de mostrar de manera correcta la distribución de los datos. La desventaja del gráfico circular es que se puede prestar para aseveraciones exageradas frente al sexo y la accidentalidad en Nueva York.

4. Tipos y daños de los vehículos involucrados en los accidentes

a. Tipos de vehículos:

Por alguna razón el tipo de vehículo sedan es el más involucrado en los accidentes de Nueva York. Esto da pie a una investigación enfocada en saber si esto puede ser por un desperfecto de fábrica, o por el tipo de persona que compra el vehículo, o la relación modelos con ciudad, entre otros.

A lo que nos concierne es que son casi la mitad de la población de vehículos, dejando el sesgo que puede presentar que sea el más vendido del estado o el que más se registre en nuestra base de datos, le sigue las SUV y los camiones, más se pueden enfocar en prestar atención a los dos primeros y así evitar futuras colisiones.

La ventaja del gráfico de barras es la facilidad con la que se puede reconocer el tipo de vehículo con más o menos accidentes, sin embargo, es necesario pasar el mouse por cada barra para

evidenciar los datos de manera precisa. La desventaja del gráfico circular es que, la cantidad de categorías excede a las seis que son recomendadas por los expertos, por lo que no se evidencia de forma simple los porcentajes por cada tipo de vehículo.

b. Daños en los vehículos

La gran mayoría de accidentes perjudican la parte frontal de los vehículos, se puede deducir que son colisiones contra otro vehículo o muro lo que causa este común denominador.

Por lo general un choque de gran magnitud por delante puede llegar a perjudicar esquinas y laterales como lo muestra la distribución en el gráfico de barras.

Hay una desventaja con estas gráficas y la cantidad de datos, al visualizar el gráfico de barras algunas columnas en el eje x se pierden y en el gráfico circular se ve muy organizado todo hasta que se llega a los porcentajes del 1% o menos donde ya no se entiende la información por que estas se trasponen.

Conclusiones

A partir del trabajo realizado a lo largo del proyecto podemos concluir:

- La selección de la base de datos es una de las tareas más complejas, pues es el recurso principal que nutre todo el proyecto. En primera instancia, para realizar una correcta construcción de los diagramas y reglas de negocio es necesario escoger una base de datos con múltiples tablas que se puedan relacionar entre sí. En segundo lugar, el número de registros tiene un alto impacto en la complejidad del proyecto, pues una fuente con muy pocas tuplas no arrojará resultados significativos y una fuente con un valor muy grande, puede ocasionar problemas como registros nulos o inconsistencias en las llaves primarias y foráneas. Es por esto, que consideramos un paso fundamental la buena elección de una base de datos y aunque nuestra elección presentó un reto muy extenso, somos conscientes que, en la realidad laboral los datos tendrán más imperfecciones y llegarán en volúmenes más altos.
- El planteamiento de las reglas de negocio fue uno de los pilares para la construcción del modelo Entidad Relación, con la redacción de estas aprendimos la importancia de escuchar al cliente y saber cómo transformar las ideas a recursos tangibles, como los diagramas. Asimismo, aprendimos a reconocer conceptos de suma importancia como la cardinalidad en las relaciones o las entidades, en las reglas de negocio.
- El diseño de la base de datos presentó un ejercicio de gran significancia para nuestro aprendizaje, pues las fuentes que escogimos contenían registros muy variados, lo que afianzó nuestra habilidad para la construcción de diagramas, en especial el Entidad Relación, pues tuvimos que considerar los dominios y atributos de cada variable. Por otro lado, el paso del diagrama ER al relacional también nos ayudó a afirmar los conceptos aprendidos durante el semestre y a ampliar nuestros conocimientos, en otro tipo de herramientas como PgModeler. Por último, el proceso de normalización fue retador por la cantidad de tablas que tenía nuestro diagrama, no obstante, se realizó de forma correcta lo que brindó una mayor facilidad al momento de cargar los datos, pues no se visualizaba redundancia horizontal o dependencias funcionales transitivas que ocasionaran anomalías.

- La carga de datos no presento mayor inconveniente, pues los comandos implementados en PostgreSQL para la carga masiva funcionan de manera sencilla y eficaz. Sin embargo, las bases de datos al tener un volumen muy alto de información generaban problemas como inconsistencias entre las llaves foráneas, por lo cual, el proceso de inserción se tardó más de lo esperado, adicionalmente, los tiempos de ejecución en algunas ocasiones superaban los cinco minutos. Estos detalles fueron solucionados con un limpiado y procesamiento de los datos.
- La conexión entre Python y SQL se realizó de forma exitosa y la construcción de las consultas fue uno de los procesos más sencillos a lo largo del proyecto, gracias a todo el conocimiento adquirido en SQL a lo largo del curso. El único detalle que nos gustaría resaltar es la importancia de colocar los datos correspondientes, como el usuario, el puerto, la base de datos, en el archivo de conexión para la correcta ejecución de las consultas.
- La implementación del dash consideramos fue una de las partes más entretenidas del proyecto, además que afianza otros tipos de habilidades como las blandas, pues se requiere de un mínimo conocimiento en diseño y un pensamiento estético. Al igual, las gráficas son una herramienta excelente para visualizar los datos y poder interpretar lo que estos nos dicen. Sin la existencia de diagramas no sería posible llevar la información a las personas que no poseen los conocimientos técnicos. Asimismo, a partir de estas visualizaciones podemos extrapolar los datos a otras áreas del conocimiento como la estadística o la probabilidad, para así construir análisis más detallados.