1. An introduction of your application, including why you want to develop the application and the main functions of your application.

Answer:
動機：一位同學某次趕著去搭公車，在路上只在意自己手機上的地圖，結果抬頭時正好目睹一場車禍，有驚無險的一個經驗，給了我們這個主意，如果能夠在出發前，對於自己路線更熟悉，甚至是可以選擇較安全的路線，就能夠在出門前先注意自己會經過的路口是不是有比較危險地點，那麼不管多趕時間，都會在那個路口特別注意，可以有效減少自己發生車禍的風險。

實作：我們使用了 R 語言進行上述實作，並且事先使用 google map 的 API 取得地圖資料，執行程式時，我們先輸入兩地點，API 會搜尋到正確地點，並且進行路線分析，決定地圖顯示範圍大小，計算兩條可以選擇的路徑，接著透過經緯度中心與地圖範圍得到地圖包含的經緯度範圍，去 AWS 取得對應資料並在圖上標示車禍事件發生的點。

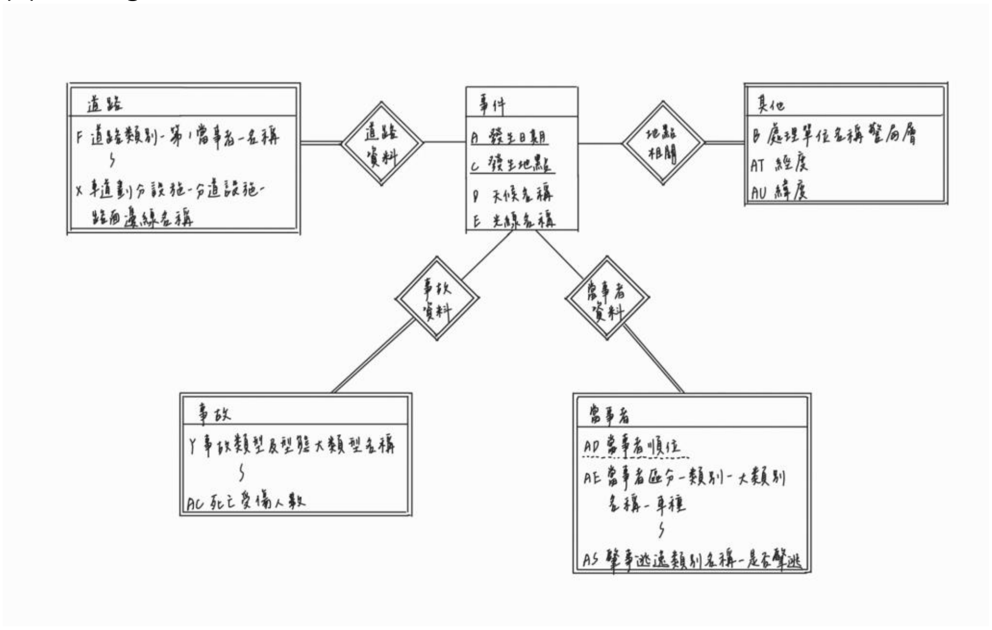
使用方法：執行程式後，輸入兩個地點（與 google map 相同，可以輸入地址或透過名稱搜尋，如輸入陽明交通大學、巨城等）。

輸出：兩條兩地之間的路線及該範圍的車禍事件分佈，紅點表示嚴重車禍，藍點則為其他車禍事件。（嚴重車禍表示傷者在車禍發生內 24 小時死亡）。

即可透過輸入位置，得到兩點之間的路線及車禍分佈。

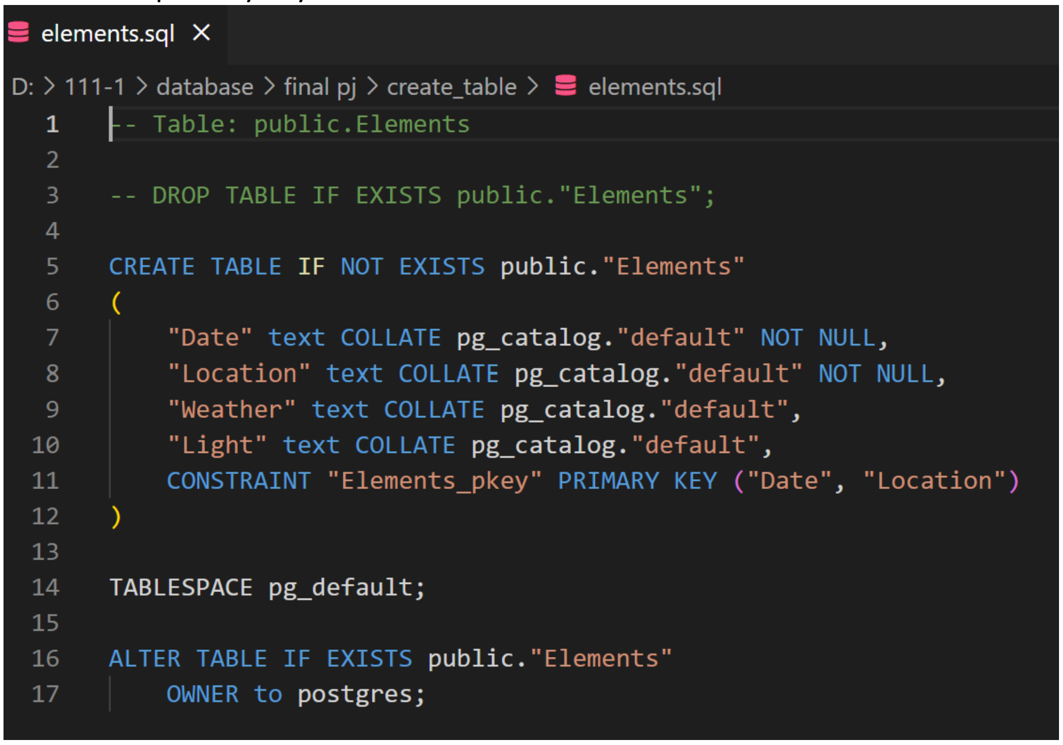
2. Database design - describe the schema of all your tables in the database, including keys and index, if applicable (why you need the keys, or why you think that adding an index is or is not helpful).

Answer:
(1) ER diagram



(2)"Elements" table

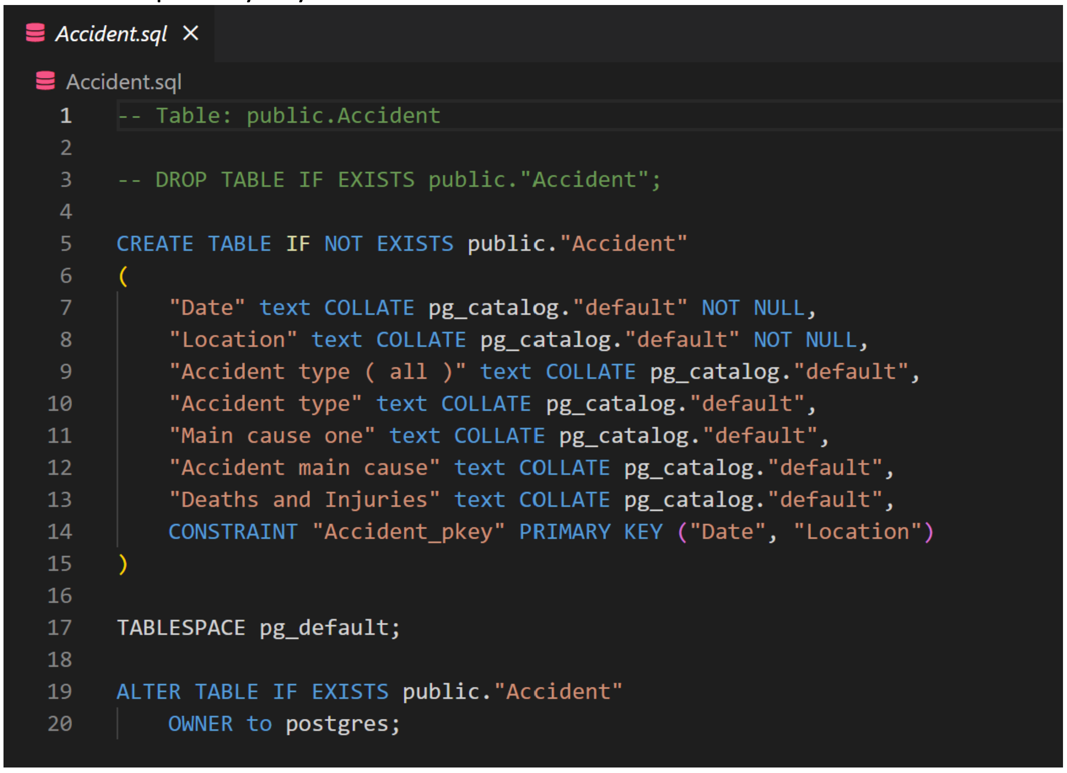
The primary keys are "Date" and "Location".


elements.sql
D: > 111-1 > database > final pj > create_table > elements.sql

```sql
-- Table: public.Elements

-- DROP TABLE IF EXISTS public."Elements";

CREATE TABLE IF NOT EXISTS public."Elements"
(
    "Date" text COLLATE pg_catalog."default" NOT NULL,
    "Location" text COLLATE pg_catalog."default" NOT NULL,
    "Weather" text COLLATE pg_catalog."default",
    "Light" text COLLATE pg_catalog."default",
    CONSTRAINT "Elements_pkey" PRIMARY KEY ("Date", "Location")
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public."Elements"
    OWNER to postgres;
```

(3) "Accident" table

The primary keys are "Date" and "Location".

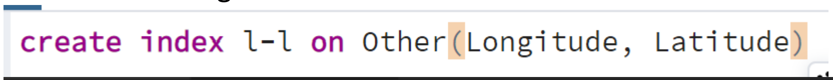


```sql
-- Table: public.Accident

-- DROP TABLE IF EXISTS public."Accident";

CREATE TABLE IF NOT EXISTS public."Accident"
(
    "Date" text COLLATE pg_catalog."default" NOT NULL,
    "Location" text COLLATE pg_catalog."default" NOT NULL,
    "Accident type ( all )" text COLLATE pg_catalog."default",
    "Accident type" text COLLATE pg_catalog."default",
    "Main cause one" text COLLATE pg_catalog."default",
    "Accident main cause" text COLLATE pg_catalog."default",
    "Deaths and Injuries" text COLLATE pg_catalog."default",
    CONSTRAINT "Accident_pkey" PRIMARY KEY ("Date", "Location")
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public."Accident"
    OWNER to postgres;
```

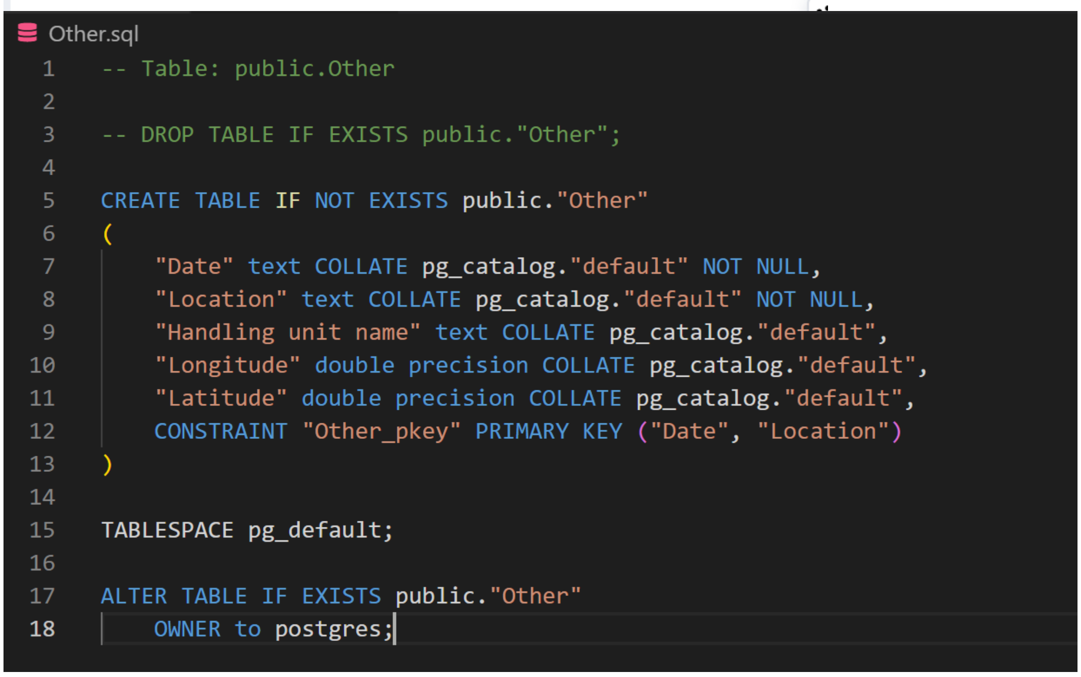
(4) “Other” table

The primary keys are “Date” and “Location”.

Use “Longitude” and “Latitude” to create an index in this table.



```
create index l-l on Other(Longitude, Latitude)
```

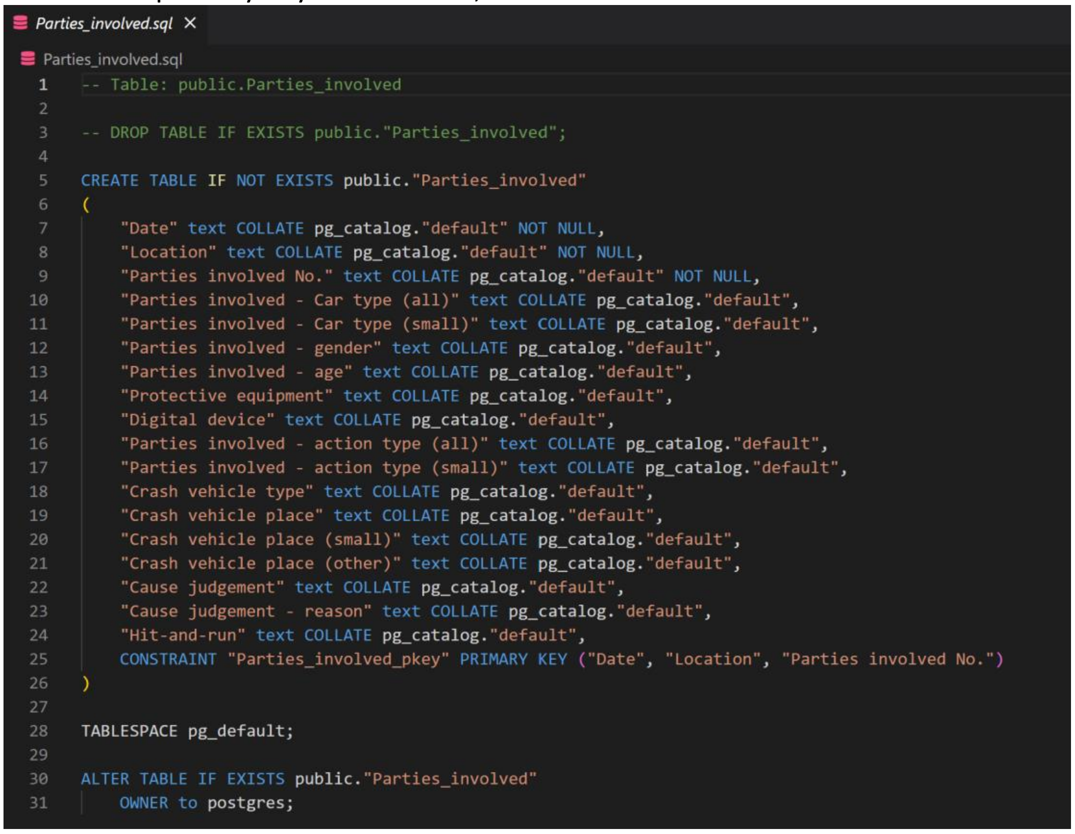


Other.sql

```sql
-- Table: public.Other

-- DROP TABLE IF EXISTS public."Other";

CREATE TABLE IF NOT EXISTS public."Other"
(
    "Date" text COLLATE pg_catalog."default" NOT NULL,
    "Location" text COLLATE pg_catalog."default" NOT NULL,
    "Handling unit name" text COLLATE pg_catalog."default",
    "Longitude" double precision COLLATE pg_catalog."default",
    "Latitude" double precision COLLATE pg_catalog."default",
    CONSTRAINT "Other_pkey" PRIMARY KEY ("Date", "Location")
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public."Other"
    OWNER to postgres;
```

(5) “Parties involved“ table

The primary keys are “Date” , “Location” and “Parties involved No.“

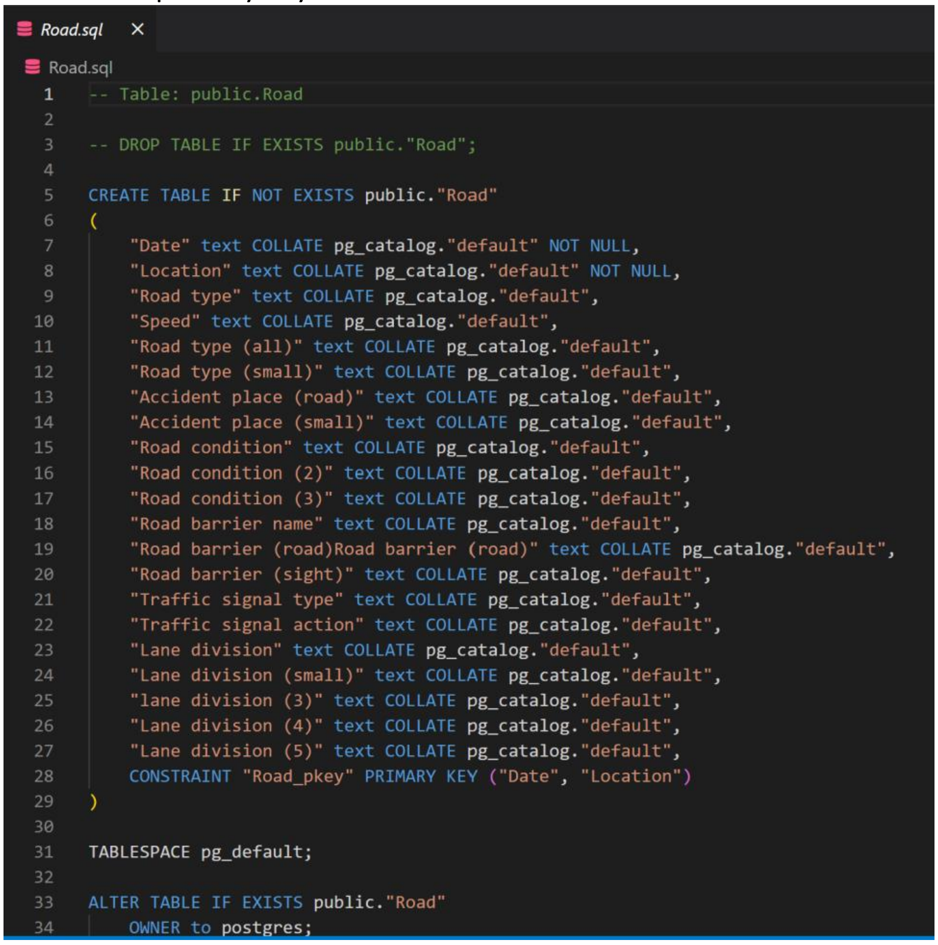


Parties_involved.sql

```sql
-- Table: public.Parties_involved

-- DROP TABLE IF EXISTS public."Parties_involved";

CREATE TABLE IF NOT EXISTS public."Parties_involved"
(
    "Date" text COLLATE pg_catalog."default" NOT NULL,
    "Location" text COLLATE pg_catalog."default" NOT NULL,
    "Parties involved No." text COLLATE pg_catalog."default" NOT NULL,
    "Parties involved - Car type (all)" text COLLATE pg_catalog."default",
    "Parties involved - Car type (small)" text COLLATE pg_catalog."default",
    "Parties involved - gender" text COLLATE pg_catalog."default",
    "Parties involved - age" text COLLATE pg_catalog."default",
    "Protective equipment" text COLLATE pg_catalog."default",
    "Digital device" text COLLATE pg_catalog."default",
    "Parties involved - action type (all)" text COLLATE pg_catalog."default",
    "Parties involved - action type (small)" text COLLATE pg_catalog."default",
    "Crash vehicle type" text COLLATE pg_catalog."default",
    "Crash vehicle place" text COLLATE pg_catalog."default",
    "Crash vehicle place (small)" text COLLATE pg_catalog."default",
    "Crash vehicle place (other)" text COLLATE pg_catalog."default",
    "Cause judgement" text COLLATE pg_catalog."default",
    "Cause judgement - reason" text COLLATE pg_catalog."default",
    "Hit-and-run" text COLLATE pg_catalog."default",
    CONSTRAINT "Parties_involved_pkey" PRIMARY KEY ("Date", "Location", "Parties involved No.")
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public."Parties_involved"
    OWNER to postgres;
```

(6) “Road” table

The primary keys are “Date” and “Location”



Road.sql

Road.sql

```sql
-- Table: public.Road

-- DROP TABLE IF EXISTS public."Road";

CREATE TABLE IF NOT EXISTS public."Road"
(
    "Date" text COLLATE pg_catalog."default" NOT NULL,
    "Location" text COLLATE pg_catalog."default" NOT NULL,
    "Road type" text COLLATE pg_catalog."default",
    "Speed" text COLLATE pg_catalog."default",
    "Road type (all)" text COLLATE pg_catalog."default",
    "Road type (small)" text COLLATE pg_catalog."default",
    "Accident place (road)" text COLLATE pg_catalog."default",
    "Accident place (small)" text COLLATE pg_catalog."default",
    "Road condition" text COLLATE pg_catalog."default",
    "Road condition (2)" text COLLATE pg_catalog."default",
    "Road condition (3)" text COLLATE pg_catalog."default",
    "Road barrier name" text COLLATE pg_catalog."default",
    "Road barrier (road)Road barrier (road)" text COLLATE pg_catalog."default",
    "Road barrier (sight)" text COLLATE pg_catalog."default",
    "Traffic signal type" text COLLATE pg_catalog."default",
    "Traffic signal action" text COLLATE pg_catalog."default",
    "Lane division" text COLLATE pg_catalog."default",
    "Lane division (small)" text COLLATE pg_catalog."default",
    "lane division (3)" text COLLATE pg_catalog."default",
    "Lane division (4)" text COLLATE pg_catalog."default",
    "Lane division (5)" text COLLATE pg_catalog."default",
    CONSTRAINT "Road_pkey" PRIMARY KEY ("Date", "Location")
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public."Road"
    OWNER to postgres;
```

3. Database design - describe the normal form of all your tables. If the tables are not in BCNF, please include the reason for it (performance trade-off, etc.).

Answer:

(1) ”Elements” table

The primary keys are “Date” and “Location”

“Date” and “Location” -> all other columns

So it is in BCNF

(2) “Accident” table

The primary keys are “Date” and “Location”

“Date” and “Location” -> all other columns

So it is in BCNF

(3) “Other” table

The primary keys are “Date” and “Location”

“Date” and “Location” -> all other columns

So it is in BCNF

(4) “Parties involved“ table

The primary keys are “Date” and “Location” and “Parties involved No.“

“Date” and “Location” and “Parties involved No.“ -> all other columns

So it is in BCNF

(5) “Road” table

The primary keys are “Date” and “Location”
“Date” and “Location” -> all other columns
So it is in BCNF

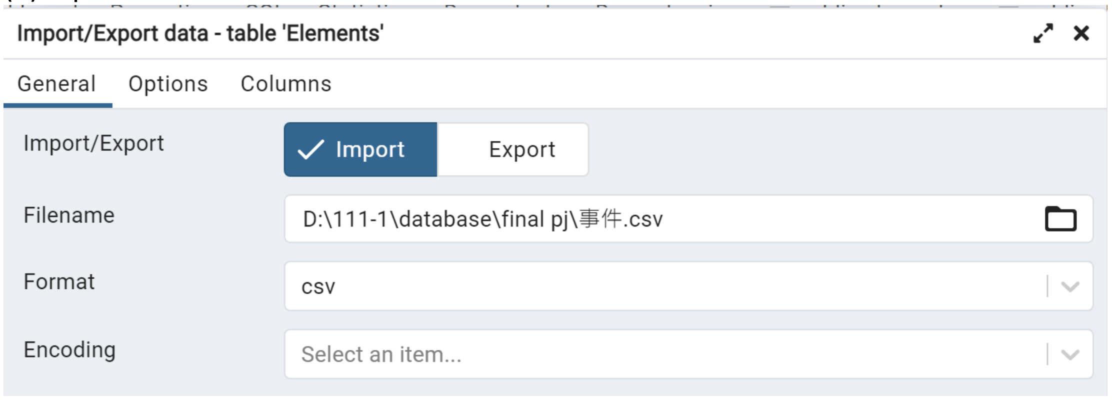
4. From the data sources to the database - describe the data source and the original format.

Answer:
在專題中所使用的資料是來自於政府資料開放平台 (https://data.gov.tw/dataset/12197) ，政府每年會公布前一年車禍事件統計，對於每件車禍案件，提供當事者性別、車種、肇事原因等相關資訊，並且資料中包含 A1 及 A2 車禍事件統計，A1 表示傷者在車禍發生內 24 小時死亡，其他則為 A2，此次專題使用 110 年車禍事件的資料，A1 及 A2 約共有 80 萬筆。在拆解資料時，在大部分 Tables 我們使用日期與地點當作 PK，在 Table”當事者"則使用日期、地點與順位當做 PK，並在過程中確認是 Lossless decomposition。

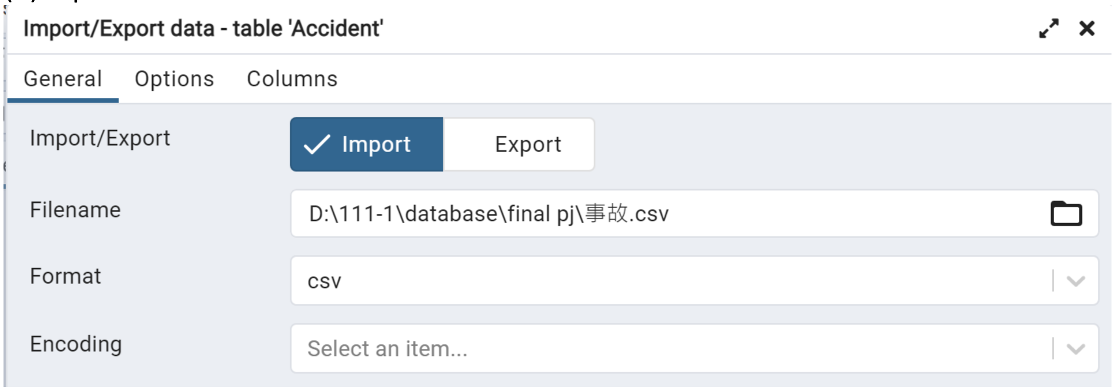
5. From the data sources to the database - describe the methods of importing the original data to your database and strategies for updating the data, if you have one.

Answer:
首先我們將資料如上述 Schema 分解，並在過程中注意到資料含有中文需要特別在創 database 的時候加上 UTF-8，並將 datatype 定義為"text” 。

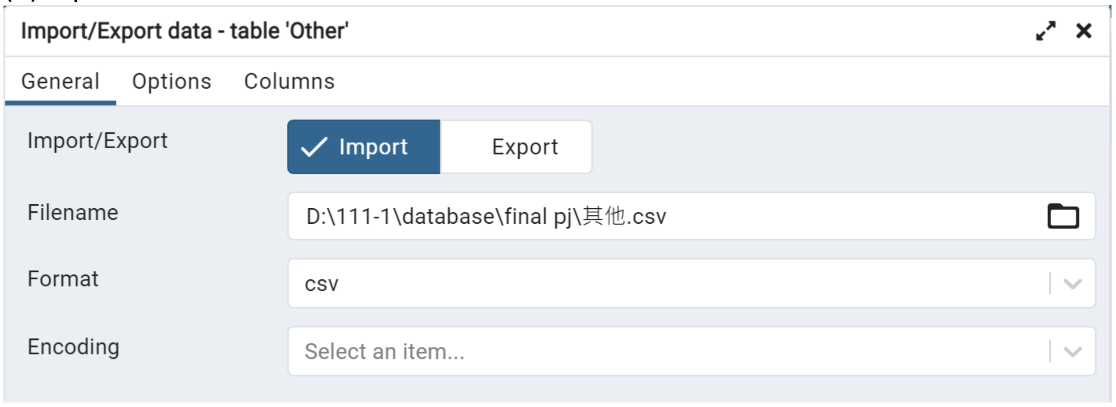
(1)import data in “Elements”


Import/Export data - table 'Elements'
General Options Columns
Import/Export: [✓ Import] Export
Filename: D:\111-1\database\final pj\事件.csv
Format: csv
Encoding: Select an item...

(2)import data in “Accidents”


Import/Export data - table 'Accident'
General Options Columns
Import/Export: [✓ Import] Export
Filename: D:\111-1\database\final pj\事故.csv
Format: csv
Encoding: Select an item...

(3)import data in “Others”

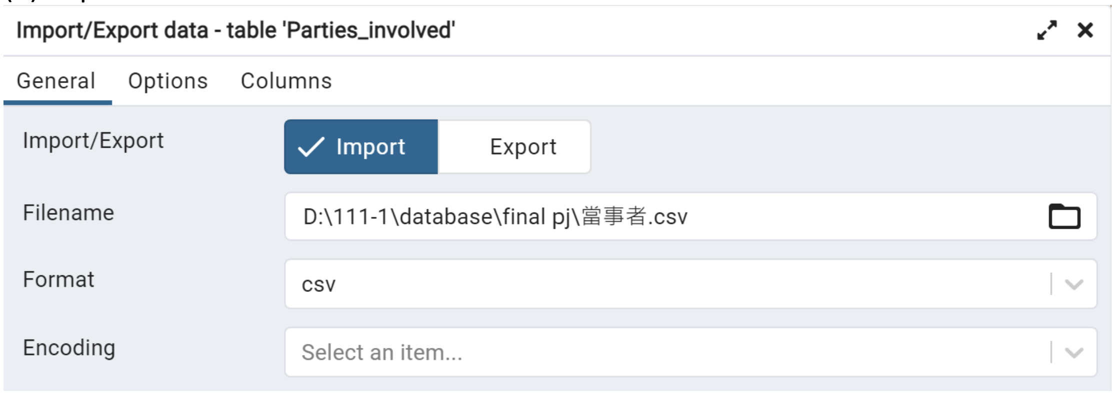


Import/Export data - table 'Other'

General | Options | Columns

Import/Export	[✓ Import] Export
Filename	D:\111-1\database\final pj\其他.csv
Format	csv
Encoding	Select an item...

(4) import data in “Parties involved“ table

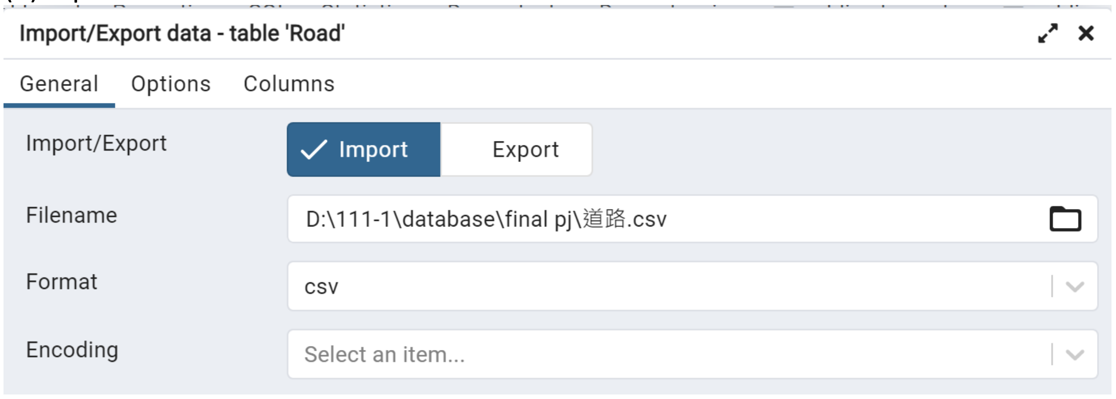


Import/Export data - table 'Parties_involved'

General | Options | Columns

Import/Export	[✓ Import] Export
Filename	D:\111-1\database\final pj\當事者.csv
Format	csv
Encoding	Select an item...

(5)import data in “Road” table



Import/Export data - table 'Road'

General | Options | Columns

Import/Export	[✓ Import] Export
Filename	D:\111-1\database\final pj\道路.csv
Format	csv
Encoding	Select an item...

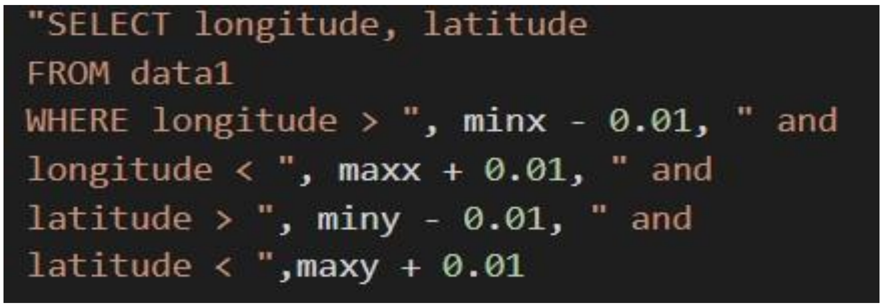
因為這個資料一年只會更新一次，所以在這個應用中只需要每年更新一次資料庫，進行新一輪的分析，update 的部分需求度並不高，因此我們沒有進行此部份的設計。

6. Application with database - explain why your application needs a database.

Answer:
這些車禍事件即使在應用程序未運行時也需要持續保存，而資料庫能用來儲存和管理大量資料。台灣一年車禍事件的資料高達八十多萬筆，儲存在資料庫可以比儲存在平面文件更有效率的處理大量數據。而且資料庫能快速取得我們想要的經緯度以及能將應用程式常用的資料存在內存中，進而提高應用程式的性能。

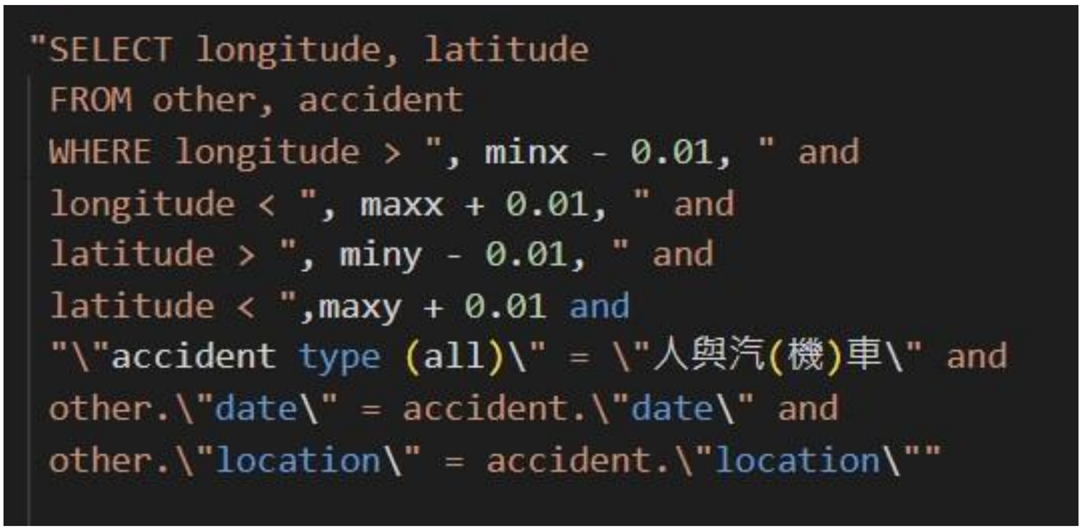
7. Application with database - includes the queries that are performed by your application, how your application performed these queries (connections between application and database), and what are the cooperating functions for your application.

Answer:
當執行程式時，API 會將使用者輸入的地點轉為經緯度，透過以下的程式碼到 database 取得範圍內的車禍資料



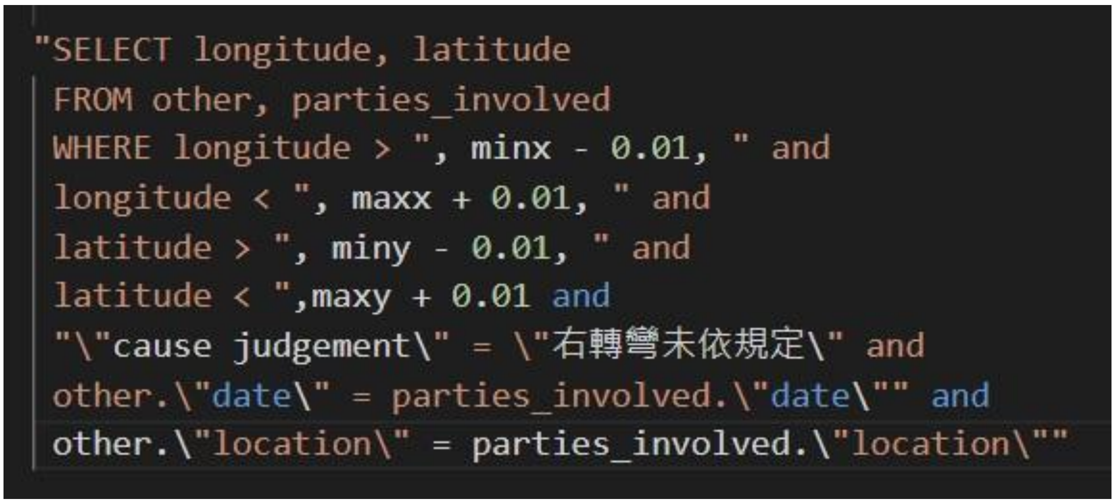
```
"SELECT longitude, latitude
FROM data1
WHERE longitude > ", minx - 0.01, " and
longitude < ", maxx + 0.01, " and
latitude > ", miny - 0.01, " and
latitude < ",maxy + 0.01
```

一般的使用者若想知到車禍事件的分布，得知車禍的種類是甚麼並沒有太大的意義。但對政府而言，若想針對車禍原因去規劃路口，需要取得特定部分的資料。舉例來說，我們可以透過以下的程式碼，不但得知範圍內的車禍事件，還把事件限定在車子與人相撞，而非車子與車子相撞。這樣一來，若這路口的點相當密集，就代表可能這路口的斑馬線設置不當，進而重新規劃這區域的人行道設計。



```
"SELECT longitude, latitude
 FROM other, accident
 WHERE longitude > ", minx - 0.01, " and
 longitude < ", maxx + 0.01, " and
 latitude > ", miny - 0.01, " and
 latitude < ",maxy + 0.01 and
 "\"accident type (all)\" = \"人與汽(機)車\" and
 other.\"date\" = accident.\"date\" and
 other.\"location\" = accident.\"location\""
```

再舉個例子來說，透過以下的程式碼，可以得知指定範圍內"右轉彎未依規定"的車禍事件，若這個路口德點相當密集，則代表可能這個路口的紅綠燈燈號設置不當，需要重新設計紅綠燈或是多派人力去該路口指揮交通，進而減少車禍事件的發生。

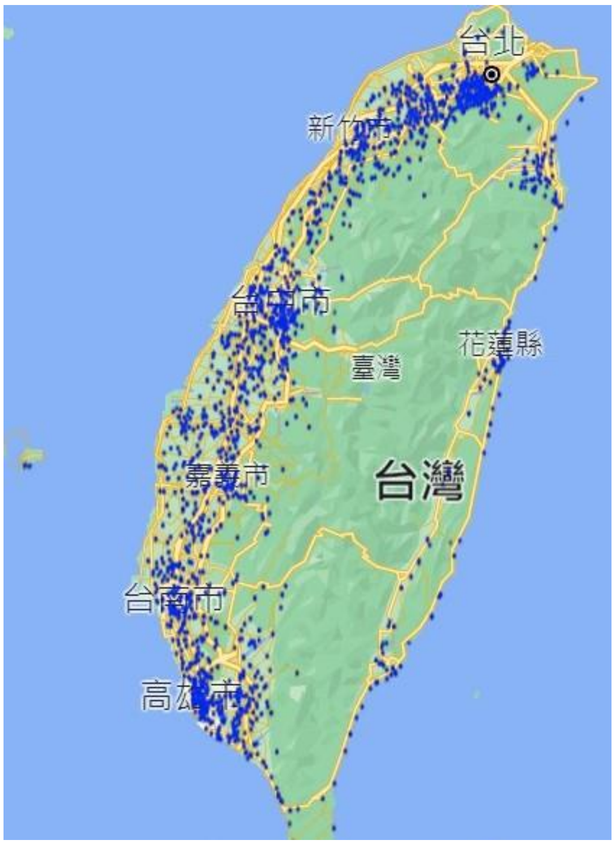


```
"SELECT longitude, latitude
 FROM other, parties_involved
 WHERE longitude > ", minx - 0.01, " and
 longitude < ", maxx + 0.01, " and
 latitude > ", miny - 0.01, " and
 latitude < ",maxy + 0.01 and
 "\"cause judgement\" = \"右轉彎未依規定\" and
 other.\"date\" = parties_involved.\"date\"" and
 other.\"location\" = parties_involved.\"location\""
```

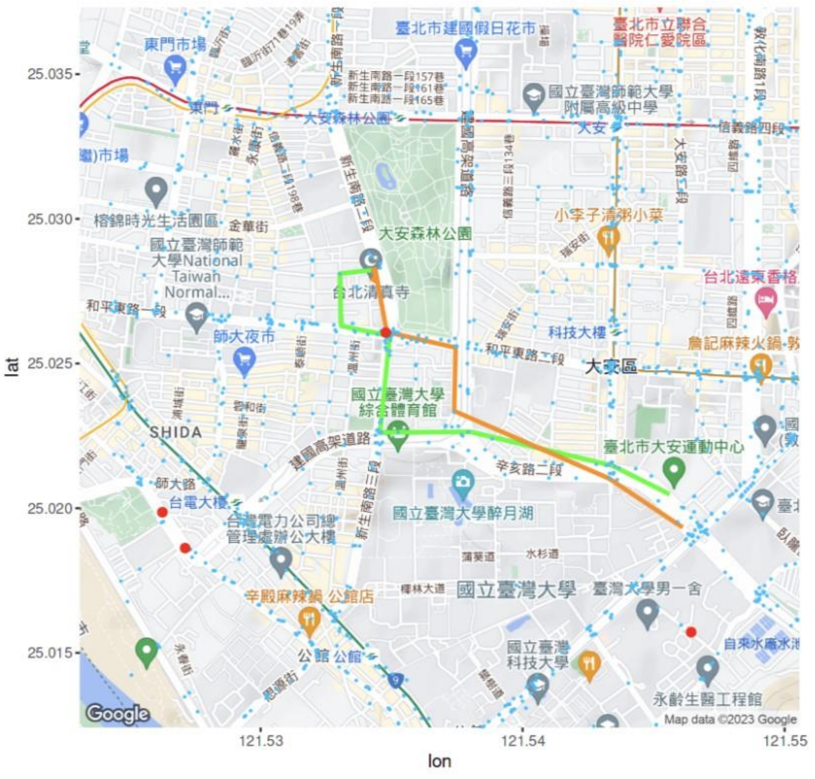
8. All the other details of your application that you want us to know.

Answer:

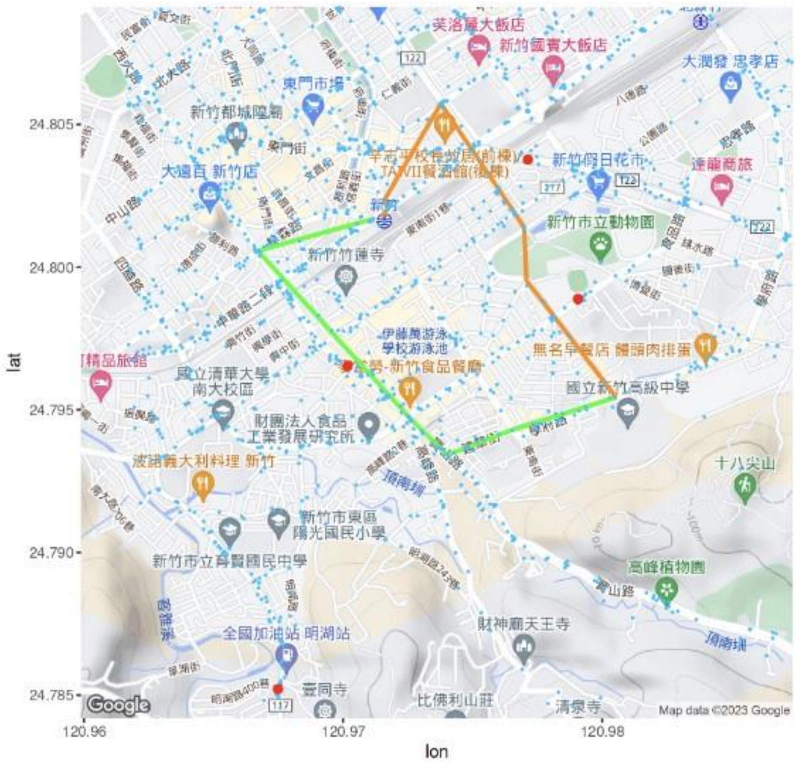
全台灣 A1(嚴重車禍)的分布圖



輸出範例 1(台北清真寺，大安運動中心)



輸出範例 2(新竹高中，新竹火車站)


lat
lon
24.805
24.800
24.795
24.790
24.785
120.96
120.97
120.98
新竹
國立新竹高級中學
新竹市立動物園
Map data ©2023 Google