

# **Real Time Traffic Project**

**Under Super Vision: DR. Amira Yousef**

**Ahmed Salama Soliman Hasan**

**Mohamed Nasr**

**Osama Ahmed Reda Hegazy**

**Sara Hisham Ahmed Mohamed**

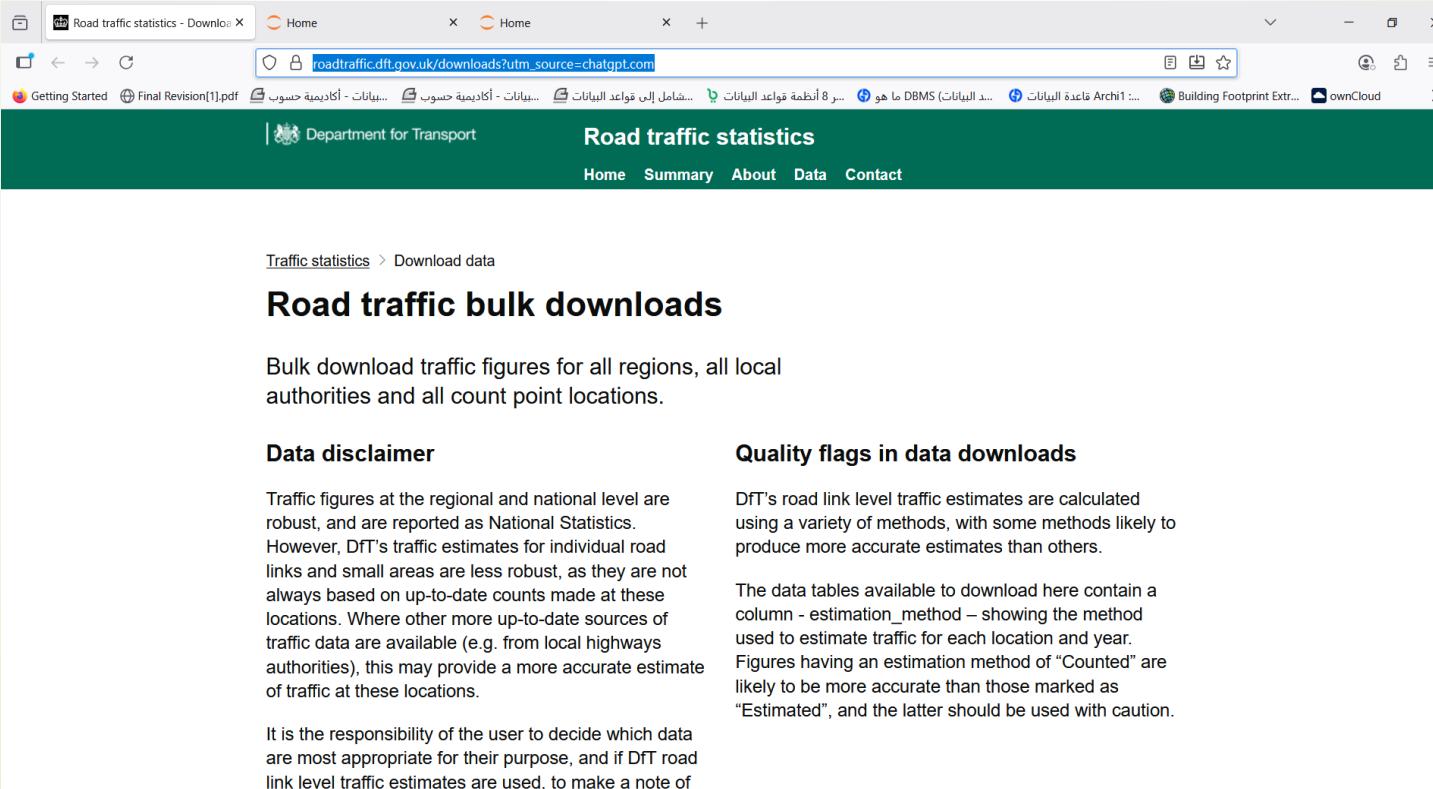
**Sherif Gamal Kamed AbdElwahed**

**Yousef Ahmed Mohamed Ibrahiem**

**Zakaria Yehia Ahmed**

# Data Source

[https://roadtraffic.dft.gov.uk/downloads?utm\\_source=chatgpt.com](https://roadtraffic.dft.gov.uk/downloads?utm_source=chatgpt.com)



The screenshot shows a web browser window with the URL [roadtraffic.dft.gov.uk/downloads?utm\\_source=chatgpt.com](https://roadtraffic.dft.gov.uk/downloads?utm_source=chatgpt.com) in the address bar. The page is titled "Road traffic statistics" and features a green header with links for Home, Summary, About, Data, and Contact. Below the header, a breadcrumb navigation shows "Traffic statistics > Download data". The main content is titled "Road traffic bulk downloads" and describes the service for downloading traffic figures for all regions, local authorities, and count point locations. Two columns of text provide additional information: "Data disclaimer" and "Quality flags in data downloads".

**Road traffic bulk downloads**

Bulk download traffic figures for all regions, all local authorities and all count point locations.

**Data disclaimer**

Traffic figures at the regional and national level are robust, and are reported as National Statistics. However, DfT's traffic estimates for individual road links and small areas are less robust, as they are not always based on up-to-date counts made at these locations. Where other more up-to-date sources of traffic data are available (e.g. from local highways authorities), this may provide a more accurate estimate of traffic at these locations.

It is the responsibility of the user to decide which data are most appropriate for their purpose, and if DfT road link level traffic estimates are used, to make a note of

**Quality flags in data downloads**

DfT's road link level traffic estimates are calculated using a variety of methods, with some methods likely to produce more accurate estimates than others.

The data tables available to download here contain a column - estimation\_method – showing the method used to estimate traffic for each location and year. Figures having an estimation method of "Counted" are likely to be more accurate than those marked as "Estimated", and the latter should be used with caution.

# Data Before

## (worked on 46476 out of 10,485,76records)



A screenshot of an Excel spreadsheet titled "dft\_traffic\_counts\_raw\_counts3.csv - Excel". The spreadsheet contains 22 rows of data, each representing a traffic count record. The columns are labeled A through R. Column A is "count\_point\_id", B is "direction\_of\_travel", C is "year", D is "count\_date", E is "hour", F is "region\_id", G is "region\_name", H is "region\_ons\_code", I is "local\_authority\_id", J is "local\_authority\_name", K is "local\_authority\_code", L is "road\_name", and M is "r". The data shows multiple entries for the same region and authority, indicating a high volume of traffic counts. The last row (row 22) has a green border around the "local\_authority\_id" cell.

count_point_id	direction_of_travel	year	count_date	hour	region_id	region_name	region_ons_code	local_authority_id	local_authority_name	local_authority_code	road_name	r
51	S	2004	5/21/2004	11	1	South West	E12000009	1	Isles of Scilly	E06000053	A3111	
51	S	2004	5/21/2004	15	1	South West	E12000009	1	Isles of Scilly	E06000053	A3111	
51	S	2004	5/21/2004	13	1	South West	E12000009	1	Isles of Scilly	E06000053	A3111	
51	N	2004	5/21/2004	7	1	South West	E12000009	1	Isles of Scilly	E06000053	A3111	
51	N	2004	5/21/2004	10	1	South West	E12000009	1	Isles of Scilly	E06000053	A3111	
51	S	2004	5/21/2004	9	1	South West	E12000009	1	Isles of Scilly	E06000053	A3111	
51	N	2004	5/21/2004	14	1	South West	E12000009	1	Isles of Scilly	E06000053	A3111	
51	N	2004	5/21/2004	8	1	South West	E12000009	1	Isles of Scilly	E06000053	A3111	
51	N	2004	5/21/2004	12	1	South West	E12000009	1	Isles of Scilly	E06000053	A3111	
51	N	2004	5/21/2004	18	1	South West	E12000009	1	Isles of Scilly	E06000053	A3111	
51	N	2004	5/21/2004	16	1	South West	E12000009	1	Isles of Scilly	E06000053	A3111	
51	N	2004	5/21/2004	17	1	South West	E12000009	1	Isles of Scilly	E06000053	A3111	
51	N	2004	5/21/2004	9	1	South West	E12000009	1	Isles of Scilly	E06000053	A3111	
51	N	2004	5/21/2004	11	1	South West	E12000009	1	Isles of Scilly	E06000053	A3111	
51	N	2004	5/21/2004	15	1	South West	E12000009	1	Isles of Scilly	E06000053	A3111	
51	S	2004	5/21/2004	7	1	South West	E12000009	1	Isles of Scilly	E06000053	A3111	
51	S	2004	5/21/2004	18	1	South West	E12000009	1	Isles of Scilly	E06000053	A3111	
51	S	2004	5/21/2004	17	1	South West	E12000009	1	Isles of Scilly	E06000053	A3111	
51	S	2004	5/21/2004	16	1	South West	E12000009	1	Isles of Scilly	E06000053	A3111	
51	S	2004	5/21/2004	8	1	South West	E12000009	1	Isles of Scilly	E06000053	A3111	
51	S	2004	5/21/2004	10	1	South West	E12000009	1	Isles of Scilly	E06000053	A3111	



dft\_traffic\_counts\_raw\_counts3.csv - Excel

File Home Insert Page Layout Formulas Data Review View Acrobat Tell me what you want to do... Sign in Share

Cut Copy Paste Format Painter Clipboard Font Alignment Number Styles Cells Editing Adobe Acrobat

Font: Calibri 24 A A Wrap Text General Conditional Format as Table Normal Bad Good Neutral

Number: \$ % , .00 .00

Formatting: Merge & Center Conditional Format as Table Insert Delete Format AutoSum Fill Sort & Find & Filter Clear Create a PDF

Cells: Insert Delete Format AutoSum Fill Sort & Find & Filter Select

Editing: Insert Delete Format AutoSum Fill Sort & Find & Filter Select

Adobe Acrobat

W20 : fx 11

	V	W	X	Y	Z	AA	AB	AC	AD	AE	AF
1	link_length_miles	pedal_cycles	two_wheeled_motor_vehicles	cars_and_taxis	buses_and_coaches	LGVs	HGVs_2_rigid_axle	HGVs_3_rigid_axle	HGVs_4_or_more_rigid_axle	HGVs_3_or_4_articulated_axle	HGVs_5_axle
2	0.19	12	2	27	2	16	2	0	0	0	0
3	0.19	10	1	29	1	13	2	0	0	0	0
4	0.19	7	0	21	2	23	5	0	0	0	0
5	0.19	5	2	3	1	13	0	0	0	0	0
6	0.19	5	1	28	4	4	4	0	0	0	0
7	0.19	12	0	17	2	19	7	0	0	0	0
8	0.19	9	3	19	1	8	5	0	0	0	1
9	0.19	19	2	13	2	13	7	7	0	0	0
10	0.19	9	1	15	2	13	1	0	0	0	0
11	0.19	12	2	23	1	3	2	0	0	0	0
12	0.19	3	3	19	1	18	5	0	0	0	0
13	0.19	16	1	16	1	4	2	0	0	0	0
14	0.19	14	1	17	5	17	5	0	0	0	0
15	0.19	12	1	17	3	14	6	0	0	0	0
16	0.19	12	3	23	2	16	1	0	0	0	0
17	0.19	4	0	4	1	9	1	0	0	0	0
18	0.19	7	2	25	1	9	2	0	0	0	0
19	0.19	14	4	21	1	11	3	0	0	0	0
20	0.19	11	5	25	1	19	5	0	0	0	0
21	0.19	17	1	13	1	14	4	0	0	0	0
22	0.19	5	2	36	4	9	6	0	0	0	0

dft\_traffic\_counts\_raw\_counts3.csv - Excel

File Home Insert Page Layout Formulas Data Review View Acrobat Tell me what you want to do... Sign in Share

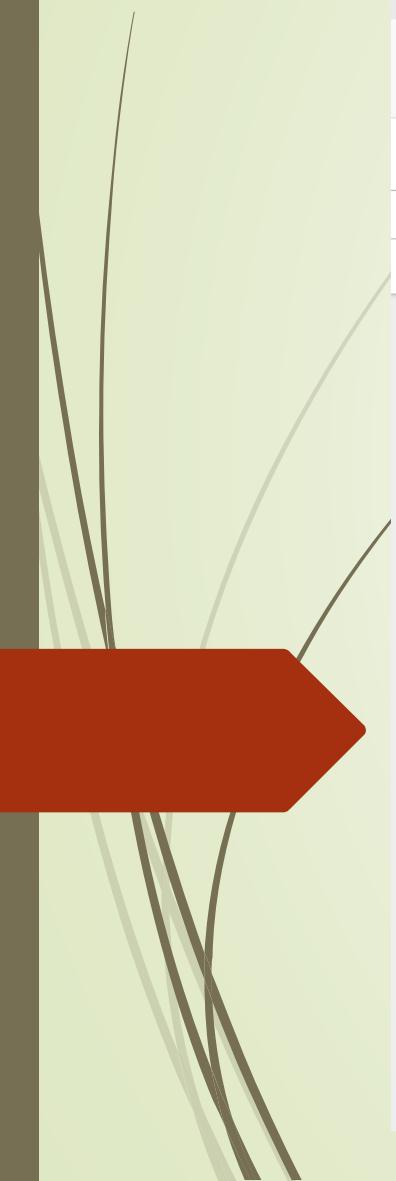
Cut Copy Format Painter Paste Calibri 24 A A Wrap Text General Conditional Format as Normal Bad Good Neutral Insert Delete Format AutoSum Fill Sort & Find & Filter Select Create a PDF Clipboard Font Alignment Number Styles Cells Editing Adobe Acrobat

AN21

	AE	AF	AG	AH	AI	AJ	AK	AL	AM	AN	AO	AP	AQ	AR	AS	AT	AU	AV	AW	AX	AY	AZ
1	HGVs_3_or_4_articulated_axle	HGVs_5_articulated_axle	HGVs_6_articulated_axle	all_HGVs	all_motor_vehicles																	
2	0	0	0	2	49																	
3	0	0	0	2	46																	
4	0	0	0	5	51																	
5	0	0	0	0	19																	
6	0	0	0	4	41																	
7	0	0	0	7	45																	
8	1	0	0	6	37																	
9	0	0	0	14	44																	
10	0	0	0	1	32																	
11	0	0	0	2	31																	
12	0	0	0	5	46																	
13	0	0	0	2	24																	
14	0	0	0	5	45																	
15	0	0	0	6	41																	
16	0	0	0	1	45																	
17	0	0	0	1	15																	
18	0	0	0	2	39																	
19	0	0	0	3	40																	
20	0	0	0	5	55																	
21	0	0	0	4	33																	
22	0	0	0	6	57																	

## شرح معاني الحقول (Columns):

اسم الحقل	المعنى
count_point_id	رقم تعرف نقطة العد (مكان تم فيه عد المركبات).
direction_of_travel	اتجاه السفر (إي شمال، جنوب، شرق، غرب).
year	سنة تسجيل البيانات.
count_date	التاريخ الفعلي للعد (يوم العد).
hour	الساعة (الوقت الذي تم فيه العد - غالباً من 0 إلى 23).
region_id	رقم تعرف المنطقة (رقم داخلي يستخدم لتحديد المنطقة).
region_name	اسم المنطقة (إي "London" أو "East Midlands").
region_ons_code	كود المنطقة حسب مكتب الإحصاء الوطني ONS للاستخدام الإحصائي.
local_authority_id	رقم تعرف السلطة المحلية (بلدية أو مجلس محلي).
local_authority_name	اسم السلطة المحلية (إي "Leeds City Council").
local_authority_code	كود السلطة المحلية (رمز مختصر).
road_name	اسم الطريق (إي A1 ، M25).
road_category	فئة الطريق (إي طريق سريع، طريق رئيسي، محلي، الخ).
road_type	نوع الطريق (قد يكون مزيد من التفصيل عن الفئة - مفرد، مزدوج، المزدوج، الخ).
start_junction_road_name	اسم الطريق عند بداية المفترق أو التقاطع.
end_junction_road_name	اسم الطريق عند نهاية المفترق أو التقاطع.
easting	الإحداثي الشرقي (نظام الإحداثيات البريطانية).
northing	الإحداثي الشمالي (نظام الإحداثيات البريطانية).
latitude	دائرة العرض (إحداثيات جغرافية).
longitude	خط الطول (إحداثيات جغرافية).
link_length_km	طول المقطع الطرقي بالكميلومترات.
link_length_miles	طول المقطع الطرقي بالأميال.
pedal_cycles	عدد الدراجات الهوائية.
two_wheeled_motor_vehicles	عدد الدراجات النارية (سكوتر، موتسيكل).
cars_and_taxis	عدد السيارات العادية والتاكسي.
buses_and_coaches	عدد الحافلات.
LGVs	سيارات النقل الخفيف - (Light Goods Vehicles) إي الفانات الصغيرة.
HGVs_2_rigid_axle	شاحنات ثقيلة بعدد 2 محاور صلب.
HGVs_3_rigid_axle	شاحنات ثقيلة بعدد 3 محاور صلبة.
HGVs_4_or_more_rigid_axle	شاحنات ثقيلة بعدد 4 أو أكثر من المحاور الصلبة.
HGVs_3_or_4_articulated_axle	شاحنات ثقيلة مفصلية بعدد 3 أو 4 محاور.
HGVs_5_articulated_axle	شاحنات ثقيلة مفصلية بعدد 5 محاور.
HGVs_6_articulated_axle	شاحنات ثقيلة مفصلية بعدد 6 محاور أو أكثر.
all_HGVs	إجمالي الشاحنات الثقيلة (HGVs).
all_motor_vehicles	إجمالي جميع المركبات الآلية (من موتسيكلات لحد الشاحنات).



Road traffic statistics - Downloaded from JupyterLab

Untitled21

Untitled20

Code analysis and suggestions

Getting Started Final Revision[1].pdf ... آداب و أخلاق في قواعد البيانات ... دليل البيانات ... دليل DBMS ... دليل Arch1 ... Building Footprint Extr... ownCloud

jupyter Untitled21 Last Checkpoint: 6 minutes ago

File Edit View Run Kernel Settings Help Trusted

JupyterLab Python [conda env:base] \* Trusted

```
[3]: import pandas as pd

# قراءة الملف
df = pd.read_csv('C:/Users/HP/dft_traffic_counts_raw_counts3.csv', low_memory=False)

# عرض أول 5 صفوف
print(df.head())

# عدد الصفوف والأعمدة
print("Shape:", df.shape)

# معلومات عن الأعمدة
df.info()

# وصف إحصائي
print(df.describe())

# أسماء الأعمدة
print("Columns:", df.columns.tolist())
```

	count_point_id	direction_of_travel	year	count_date	hour	region_id	\
0	51	S	2004	5/21/2004	11	1	
1	51	S	2004	5/21/2004	15	1	
2	51	S	2004	5/21/2004	13	1	
3	51	N	2004	5/21/2004	7	1	



Road traffic statistics - Downloaded from jupyter Untitled21 Untitled20 Code analysis and suggestions

http://localhost:8888/notebooks/Untitled21.ipynb? 120% ownCloud

Getting Started Final Revision[1].pdf ... أكاديمية حسوب ... قواعد البيانات DBMS ما هو ... 8 أنظمة قواعد البيانات ... د.البيانات Archi1 ... Building Footprint Extr... ownCloud

jupyter Untitled21 Last Checkpoint: 7 minutes ago Trusted

File Edit View Run Kernel Settings Help JupyterLab Python [conda env:base] \*

```
region_name region_ons_code local_authority_id local_authority_name ... \
0 South West E1200009 1 Isles of Scilly ...
1 South West E1200009 1 Isles of Scilly ...
2 South West E1200009 1 Isles of Scilly ...
3 South West E1200009 1 Isles of Scilly ...
4 South West E1200009 1 Isles of Scilly ...

buses_and_coaches LGVs HGVs_2_rigid_axle HGVs_3_rigid_axle ...
0 2 16 2 0
1 1 13 2 0
2 2 23 5 0
3 1 13 0 0
4 4 4 4 0

HGVs_4_or_more_rigid_axle HGVs_3_or_4_articulated_axle ...
0 0 0
1 0 0
2 0 0
3 0 0
4 0 0

HGVs_5_articulated_axle HGVs_6_articulated_axle all_HGVs ...
0 0 0 2
1 0 0 2
2 0 0 5
3 0 0 0
4 0 0 4
```



Road traffic statistics - Downloaded from JupyterLab

Untitled21

Untitled20

Code analysis and suggestions

http://localhost:8888/notebooks/Untitled21.ipynb?

Getting Started Final Revision[1].pdf ... شامل إلى قواعد البيانات ...بيانات - أكاديمية حسوب ...بيانات - أكاديمية حسوب ...بيانات DBMS ما هو ...ر 8 أنظمة قواعد البيانات ...Archi1 ...Building Footprint Extr... ownCloud

jupyter Untitled21 Last Checkpoint: 7 minutes ago

File Edit View Run Kernel Settings Help Trusted

Code

[5 rows x 35 columns]  
Shape: (46475, 35)  
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 46475 entries, 0 to 46474  
Data columns (total 35 columns):

#	Column	Non-Null Count	Dtype
0	count_point_id	46475	non-null int64
1	direction_of_travel	46475	non-null object
2	year	46475	non-null int64
3	count_date	46475	non-null object
4	hour	46475	non-null int64
5	region_id	46475	non-null int64
6	region_name	46475	non-null object
7	region_ons_code	46475	non-null object
8	local_authority_id	46475	non-null int64
9	local_authority_name	46475	non-null object
10	local_authority_code	46475	non-null object
11	road_name	46475	non-null object
12	road_category	46475	non-null object
13	road_type	46475	non-null object
14	start_junction_road_name	46475	non-null object
15	end_junction_road_name	46475	non-null object
16	easting	46475	non-null int64
17	northing	46475	non-null int64
18	latitude	46475	non-null float64
19	longitude	46475	non-null float64
20	link_length_km	46475	non-null float64
21	link_length_miles	46475	non-null float64

Road traffic statistics - Downloaded from jupyter Untitled21 Untitled20 Code analysis and suggestions

http://localhost:8888/notebooks/Untitled21.ipynb? 120% 120% 120%

Getting Started Final Revision[1].pdf ...بيانات - أكاديمية حسوب ...بيانات - قاعدة البيانات ما هو ... أنظمة قواعد البيانات ... شاهد البياناتArch1 ... Building Footprint Extr... ownCloud

jupyter Untitled21 Last Checkpoint: 7 minutes ago Python [conda env:base] \* Trusted

	count_point_id	year	hour	region_id
count	46475.000000	46475.000000	46475.000000	46475.000000
mean	2628.435611	2009.463690	12.500118	4.371017
std	2513.007772	6.839983	3.451989	2.079526
min	51.000000	2000.000000	7.000000	1.000000
25%	605.000000	2004.000000	10.000000	3.000000
50%	1071.000000	2008.000000	13.000000	4.000000
75%	6015.000000	2014.000000	15.000000	5.000000
max	6050.000000	2024.000000	18.000000	10.000000

Road traffic statistics - Download X Home X Untitled21 X Untitled20 X Code analysis and suggestions X +

http://localhost:8888/notebooks/Untitled21.ipynb? 120% 120%

Getting Started Final Revision[1].pdf ... آدبيات - أكاديمية حسوب ... آدبيات - أكاديمية حسوب ... شامل إلى قواعد البيانات ... آدبيات - أكاديمية حسوب ... آدبيات - أكاديمية حسوب ... DBMS ما هو ... أنظمة قواعد البيانات ... قاعدة البيانات Archi1 ... Building Footprint Extr... ownCloud >

jupyter Untitled21 Last Checkpoint: 7 minutes ago Python [conda env:base] \* Trusted

File Edit View Run Kernel Settings Help JupyterLab Python [conda env:base] \* Trusted

Code

```
max      6059.000000  2024.000000  18.000000  10.000000\n\nlocal_authority_id      easting      northng      latitude  \\\ncount      46475.000000  46475.000000  4.647500e+04  46475.000000\nmean       42.692996  338684.577924  4.182568e+05  53.649515\nstd        33.740813  87046.630470  2.305291e+05  2.070376\nmin        1.000000  74900.000000  1.021700e+04  49.912233\n25%       15.000000  280000.000000  2.073800e+05  51.756333\n50%       35.000000  324887.000000  3.757400e+05  53.266830\n75%       65.000000  389200.000000  6.633200e+05  55.843729\nmax       213.000000  573500.000000  1.178850e+06  60.492229\n\nlongitude  link_length_km  ...  buses_and_coaches      LGVs  \\\ncount      46475.000000  46475.000000  ...  46475.000000  46475.000000\nmean       -2.951119  5.643836  ...  9.593158  230.150016\nstd        1.315516  4.979269  ...  16.567907  253.779161\nmin       -7.442666  0.100000  ...  0.000000  0.000000\n25%       -3.885847  2.300000  ...  2.000000  33.000000\n50%       -3.123265  4.100000  ...  6.000000  122.000000\n75%       -2.166362  7.400000  ...  13.000000  372.000000\nmax       0.490767  48.600000  ...  866.000000  2513.000000\n\nHGVs_2_rigid_axle  HGVs_3_rigid_axle  HGVs_4_or_more_rigid_axle  \\\ncount      46475.000000  46475.000000  46475.000000\nmean       51.915740  8.827047  9.482087\nstd        63.447956  11.027195  13.330071\nmin        0.000000  0.000000  0.000000\n25%       6.000000  1.000000  0.000000\n50%       24.000000  4.000000  4.000000
```



Road traffic statistics - Downloaded from Archi1... [Home](#) [Untitled21](#) [Untitled20](#) [Code analysis and suggestions](#)

<http://localhost:8888/notebooks/Untitled21.ipynb?> 120% [Star](#) [Edit](#) [Run](#) [Kernel](#) [Settings](#) [Help](#) Trusted

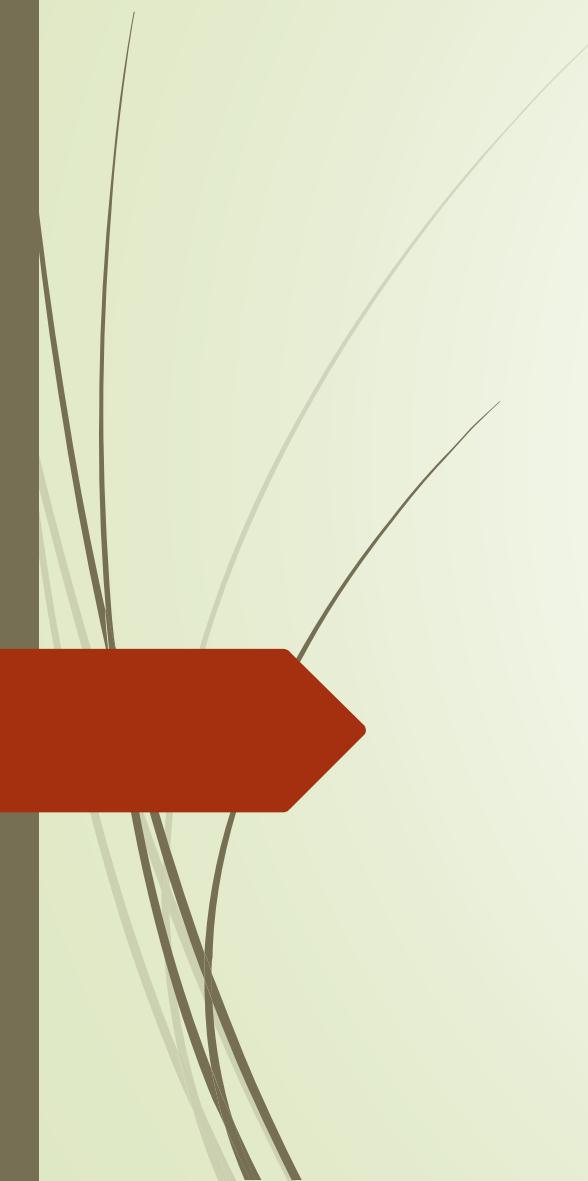
jupyter Untitled21 Last Checkpoint: 7 minutes ago 

**Code** JupyterLab Python [conda env:base]\*

	HGVs_3_or_4_articulated_axle	HGVs_5_articulated_axle	HGVs_6_articulated_axle	all_HGVs	all_motor_vehicles
max	745.000000	143.000000	244.000000	46475.000000	46475.000000
count	46475.000000	46475.000000	46475.000000	46475.000000	46475.000000
mean	9.786724	40.126025	165.909844	1606.654051	1606.654051
std	16.008458	65.132358	212.660027	212.660027	212.660027
min	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	1.000000	11.000000	11.000000	11.000000
50%	3.000000	9.000000	59.000000	59.000000	59.000000
75%	12.000000	52.000000	261.000000	261.000000	261.000000
max	244.000000	629.000000	1430.000000	1525.000000	9292.000000

[8 rows x 24 columns]

Columns: ['count\_point\_id', 'direction\_of\_travel', 'year', 'count\_date', 'hour', 'region\_id', 'region\_name', 'region\_ons\_code', 'local\_authority\_id', 'local\_authority\_name', 'local\_authority\_code', 'road\_name', 'road\_category', 'road\_type', 'start\_junction\_road\_name', 'end\_junction\_road\_name', 'easting', 'northing', 'latitude', 'longitude', 'link\_length\_km', 'link\_length\_miles', 'pedal\_cycles', 'two\_wheeled\_motor\_vehicles', 'cars\_and\_taxis', 'buses\_and\_coaches', 'LGVs', 'HGVs\_2\_rigid\_axle', 'HGVs\_3\_rigid\_axle', 'HGVs\_4\_or\_more\_rigid\_axle', 'HGVs\_3\_or\_4\_articulated\_axle', 'HGVs\_5\_articulated\_axle', 'HGVs\_6\_articulated\_axle', 'all\_HGVs', 'all\_motor\_vehicles']

- 
- قراءة البيانات وتنظيفها
  - تحويل الأعمدة الزمنية
  - التحقق من القيم الشاذة
  - هندسة خصائص جديدة (feature engineering)
  - تصدير البيانات إلى قاعدة بيانات SQL
  - إنشاء GeoDataFrame وتصدير البيانات إلى Shapefile وGeoJSON
  - تقرير جودة البيانات

```
# 1. Initial Setup and Data Loading
# -----
import pandas as pd
import numpy as np
import geopandas as gpd
from sqlalchemy import create_engine
from sqlalchemy.engine import URL

# Load the CSV data
df = pd.read_csv('C:/Users/HP/dft_traffic_counts_raw_counts3.csv',
low_memory=False)
print(f'Dataset shape: {df.shape}')
print(df.info())
```



```
Dataset shape: (46475, 35)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 46475 entries, 0 to 46474
Data columns (total 35 columns):
 #   Column           Non-Null Count Dtype  
 ---  -- 
 0   count_point_id    46475 non-null  int64  
 1   direction_of_travel 46475 non-null  object  
 2   year              46475 non-null  int64  
 3   count_date        46475 non-null  object  
 4   hour              46475 non-null  int64  
 5   region_id         46475 non-null  int64  
 6   region_name       46475 non-null  object  
 7   region_ons_code   46475 non-null  object  
 8   local_authority_id 46475 non-null  int64  
 9   local_authority_name 46475 non-null  object  
 10  local_authority_code 46475 non-null  object  
 11  road_name         46475 non-null  object  
 12  road_category     46475 non-null  object  
 13  road_type          46475 non-null  object  
 14  start_junction_road_name 46475 non-null  object  
 15  end_junction_road_name 46475 non-null  object  
 16  easting            46475 non-null  int64  
 17  northing           46475 non-null  int64  
 18  latitude            46475 non-null  float64 
 19  longitude           46475 non-null  float64 
 20  link_length_km     46475 non-null  float64 
 21  link_length_miles   46475 non-null  float64 
 22  pedal_cycles       46475 non-null  int64  
 23  two_wheeled_motor_vehicles 46475 non-null  int64  
 24  cars_and_taxis      46475 non-null  int64  
 25  buses_and_coaches    46475 non-null  int64  
 26  LGVs                46475 non-null  int64  
 27  HGVs_2_rigid_axle   46475 non-null  int64  
 28  HGVs_3_rigid_axle   46475 non-null  int64  
 29  HGVs_4_or_more_rigid_axle 46475 non-null  int64  
 30  HGVs_3_or_4_articulated_axle 46475 non-null  int64  
 31  HGVs_5_articulated_axle   46475 non-null  int64  
 32  HGVs_6_articulated_axle   46475 non-null  int64  
 33  all_HGVs            46475 non-null  int64  
 34  all_motor_vehicles    46475 non-null  int64  
dtypes: float64(4), int64(20), object(11)
memory usage: 12.4+ MB
None
```

```
# -----
```

## # 2. Data Exploration and Cleaning

```
# -----
```

```
# Check for missing values  
  
missing_values = df.isnull().sum()  
  
print("\nMissing values per column:")  
  
print(missing_values[missing_values > 0])  
  
Missing values per column:  
Series([], dtype: int64)
```

```
# Handle missing values  
  
df.fillna({  
    'link_length_km': 0,  
    'link_length_miles': 0,  
    'pedal_cycles': 0,  
}, inplace=True)  
  
Missing values per column:  
Series([], dtype: int64)  
  
# Convert count_date to datetime (auto format)  
  
df['count_date'] = pd.to_datetime(df['count_date'], errors='coerce', dayfirst=False)  
  
# Check data types  
  
print("\nData types after conversion:")  
print(df.dtypes)
```



```
Data types after conversion:
```

count_point_id	int64
direction_of_travel	object
year	int64
count_date	datetime64[ns]
hour	int64
region_id	int64
region_name	object
region_ons_code	object
local_authority_id	int64
local_authority_name	object
local_authority_code	object
road_name	object
road_category	object
road_type	object
start_junction_road_name	object
end_junction_road_name	object
easting	int64
northing	int64
latitude	float64
longitude	float64
link_length_km	float64
link_length_miles	float64
pedal_cycles	int64
two_wheeled_motor_vehicles	int64
cars_and_taxis	int64
buses_and_coaches	int64
LGVs	int64
HGVs_2_rigid_axle	int64
HGVs_3_rigid_axle	int64
HGVs_4_or_more_rigid_axle	int64
HGVs_3_or_4_articulated_axle	int64
HGVs_5_articulated_axle	int64
HGVs_6_articulated_axle	int64
all_HGVs	int64
all_motor_vehicles	int64
dtype: object	

```
# -----
```

### # 3. Data Validation and Quality Checks

```
# -----
```

```
numeric_columns = ['hour', 'link_length_km', 'pedal_cycles', 'cars_and_taxis',  
'all_motor_vehicles']
```

```
for col in numeric_columns:
```

```
    if col in df.columns:
```

```
        print(f"\n{col} range: {df[col].min()} - {df[col].max()}\")
```

```
print("\nRoad categories:", df['road_category'].unique())
```

```
print("Road types:", df['road_type'].unique())
```

```
valid_coords = (df['latitude'].between(-90, 90)) & (df['longitude'].between(-180, 180))
```

```
print(f"\nInvalid coordinates: {len(df[~valid_coords])}\")
```

```
hour range: 7 - 18
```

```
link_length_km range: 0.1 - 48.6
```

```
pedal_cycles range: 0 - 219
```

```
cars_and_taxis range: 0 - 7642
```

```
all_motor_vehicles range: 0 - 9292
```

```
Road categories: ['PA' 'TM' 'TA']
```

```
Road types: ['Major']
```

```
Invalid coordinates: 0
```

```
# -----  
# 4. Data Transformation and Feature Engineering  
# -----  
  
df['total_vehicles'] = (  
    df['pedal_cycles'] +  
    df['two_wheeled_motor_vehicles'] +  
    df['cars_and_taxis'] +  
    df['buses_and_coaches'] +  
    df['LGVs'] +  
    df['all_HGVs'])  
  
df['vehicle_density'] = df['total_vehicles'] / df['link_length_km'].replace(0, 0.001)  
  
df['day_of_week'] = df['count_date'].dt.day_name()  
df['month'] = df['count_date'].dt.month  
df['season'] = df['month'].apply(lambda x: 'Winter' if x in [12, 1, 2] else  
                                'Spring' if x in [3, 4, 5] else  
                                'Summer' if x in [6, 7, 8] else  
                                'Autumn')  
  
df['hgv_percentage'] = df['all_HGVs'] / df['total_vehicles'].replace(0, 1)  
df['bicycle_percentage'] = df['pedal_cycles'] / df['total_vehicles'].replace(0, 1)
```

```
# -----  
  
# 5. Save to Parquet (optional)  
# -----  
  
df.to_parquet('traffic_data_processed.parquet', engine='pyarrow')  
  
# -----  
  
from sqlalchemy import create_engine  
from sqlalchemy.engine import URL  
  
connection_url = URL.create(  
    "mssql+pyodbc",  
    host="localhost",  
    database="traffic_db",  
    query={  
        "driver": "ODBC Driver 17 for SQL Server",  
        "Trusted_Connection": "yes"  
    }  
)  
  
engine = create_engine(connection_url)  
  
df.to_sql('traffic_counts', engine, if_exists='replace', index=False)
```

S File Edit View Query Git Project Tools Extensions Window Help Search Solution1 Sign in

traffic\_db Execute SQLQuery2.sql - not connected SQLQuery1.sql - not connected

Object Explorer

DESKTOP-KPL988A (SQL Server 16.0.1000.6 - DESKTOP-KPL988A)

- Databases
  - System Databases
  - Database Snapshots
  - files
  - files\_New
  - Morocco
  - NewDB
  - SalesDB
  - traffic\_db
    - Database Diagrams
    - Tables
      - System Tables
      - FileTables
      - External Tables
      - Graph Tables
      - dbo.traffic\_counts
    - Dropped Ledger Tables
  - Views
  - External Resources
  - Synonyms
  - Programmability
  - Query Store
  - Service Broker
  - Storage
  - Security
  - university
  - wasted\_management\_DW
  - Security
  - Server Objects
  - Replication
  - Always On High Availability
  - Management
    - SQL Server Agent (Agent XPs disabled)
    - XEvent Profiler
- Management

1   SELECT TOP (1000) [count\_point\_id]  
2       ,[direction\_of\_travel]  
3       ,[year]  
4       ,[count\_date]  
5       ,[hour]  
6       ,[region\_id]  
7       ,[region\_name]  
8       ,[region\_ons\_code]  
9       ,[local\_authority\_id]  
10      ,[local\_authority\_name]  
11      ,[local\_authority\_code]  
12      ,[road\_name]  
13      ,[road\_category]  
14      ,[road\_type]  
15      ,[start\_junction\_road\_name]  
16      ,[end\_junction\_road\_name]  
17      ,[easting]  
18      ,[northing]  
19      ,[latitude]

No issues found

Results Messages

	count_point_id	direction_of_travel	year	count_date	hour	region_id	region_name	region_ons_code	local_authority_id	local_authority_name	local_authority_code	road_name	road_cate
1	51	S	2004	2004-05-21 00:00:00.000	11	1	South West	E1200009	1	Isles of Scilly	E06000053	A3111	PA
2	51	S	2004	2004-05-21 00:00:00.000	15	1	South West	E1200009	1	Isles of Scilly	E06000053	A3111	PA
3	51	S	2004	2004-05-21 00:00:00.000	13	1	South West	E1200009	1	Isles of Scilly	E06000053	A3111	PA
4	51	N	2004	2004-05-21 00:00:00.000	7	1	South West	E1200009	1	Isles of Scilly	E06000053	A3111	PA
5	51	N	2004	2004-05-21 00:00:00.000	10	1	South West	E1200009	1	Isles of Scilly	E06000053	A3111	PA
6	51	S	2004	2004-05-21 00:00:00.000	9	1	South West	E1200009	1	Isles of Scilly	E06000053	A3111	PA
7	51	N	2004	2004-05-21 00:00:00.000	14	1	South West	E1200009	1	Isles of Scilly	E06000053	A3111	PA
8	51	N	2004	2004-05-21 00:00:00.000	8	1	South West	E1200009	1	Isles of Scilly	E06000053	A3111	PA
9	51	N	2004	2004-05-21 00:00:00.000	12	1	South West	E1200009	1	Isles of Scilly	E06000053	A3111	PA
10	51	N	2004	2004-05-21 00:00:00.000	18	1	South West	E1200009	1	Isles of Scilly	E06000053	A3111	PA
11	51	N	2004	2004-05-21 00:00:00.000	16	1	South West	E1200009	1	Isles of Scilly	E06000053	A3111	PA
12	51	N	2004	2004-05-21 00:00:00.000	17	1	South West	E1200009	1	Isles of Scilly	E06000053	A3111	PA
13	51	N	2004	2004-05-21 00:00:00.000	9	1	South West	E1200009	1	Isles of Scilly	E06000053	A3111	PA
14	51	N	2004	2004-05-21 00:00:00.000	11	1	South West	E1200009	1	Isles of Scilly	E06000053	A3111	PA

Disconnected.

Ready

```
# -----  
  
# 7. Aggregate Data for Reporting  
# -----  
  
daily_totals = df.groupby(['count_date', 'region_name', 'local_authority_name']).agg({  
    'total_vehicles': 'sum',  
    'pedal_cycles': 'sum',  
    'all_HGVs': 'sum'  
}).reset_index()  
  


|      | count_date | region_name   | local_authority_name | total_vehicles | \   |
|------|------------|---------------|----------------------|----------------|-----|
| 0    | 2000-03-17 | Scotland      | East Lothian         | 13320          |     |
| 1    | 2000-03-17 | Wales         | Cardiff              | 17916          |     |
| 2    | 2000-03-21 | North West    | Warrington           | 72633          |     |
| 3    | 2000-03-22 | East Midlands | Northamptonshire     | 9259           |     |
| 4    | 2000-03-22 | Scotland      | North Lanarkshire    | 7475           |     |
| ...  | ...        | ...           | ...                  | ...            | ... |
| 1936 | 2024-09-18 | Scotland      | Shetland Islands     | 3295           |     |
| 1937 | 2024-09-18 | Wales         | Gwynedd              | 10587          |     |
| 1938 | 2024-09-26 | North West    | Stockport            | 30383          |     |
| 1939 | 2024-10-01 | Wales         | Denbighshire         | 32774          |     |
| 1940 | 2024-10-08 | North West    | Cumberland           | 38656          |     |

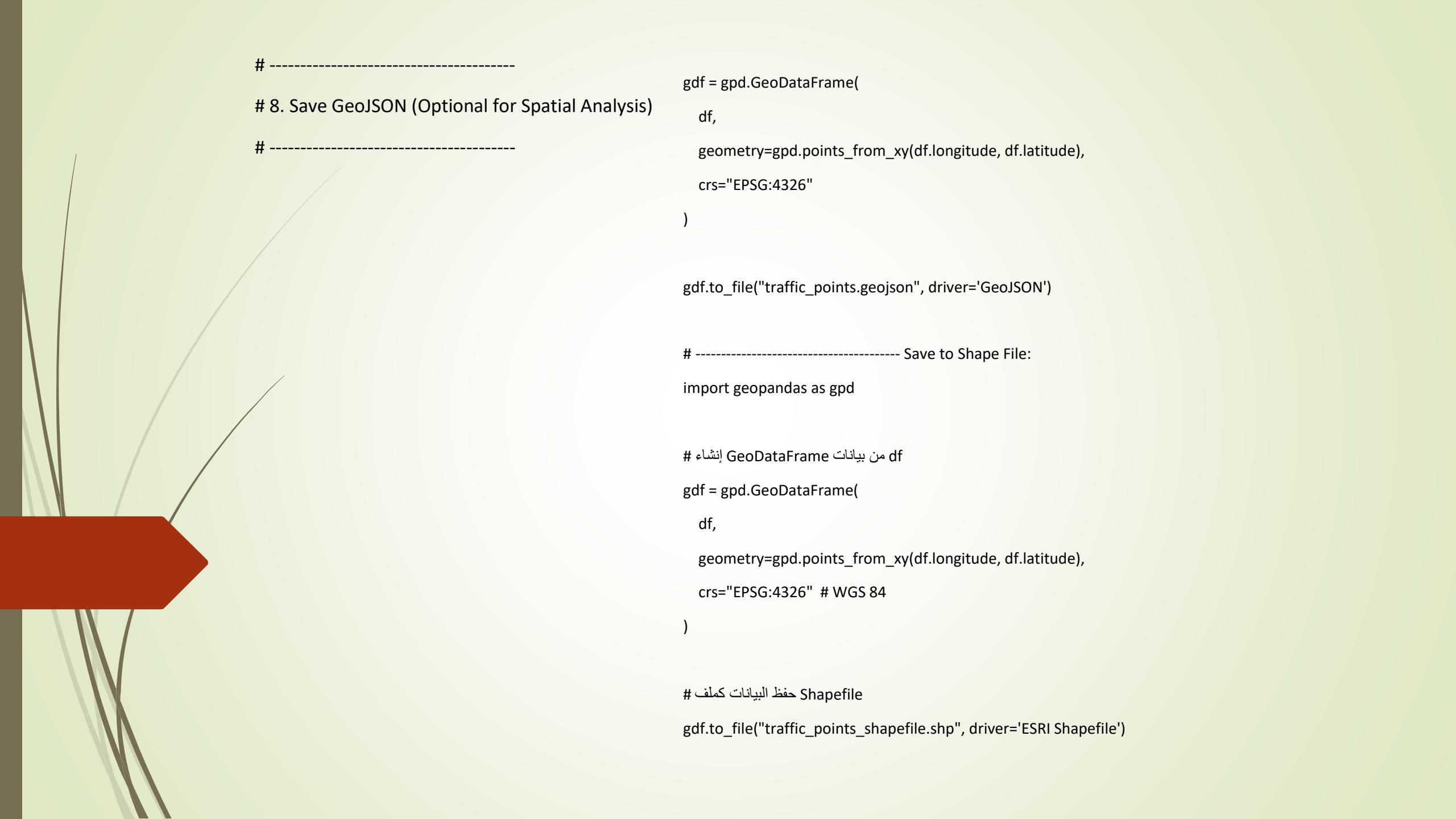
  


|      | pedal_cycles | all_HGVs |
|------|--------------|----------|
| 0    | 62           | 418      |
| 1    | 35           | 526      |
| 2    | 0            | 11003    |
| 3    | 1            | 1179     |
| 4    | 17           | 439      |
| ...  | ...          | ...      |
| 1936 | 13           | 43       |
| 1937 | 8            | 239      |
| 1938 | 143          | 627      |
| 1939 | 0            | 2998     |
| 1940 | 1            | 7344     |


[1941 rows x 6 columns]


```



```
# -----  
# 8. Save GeoJSON (Optional for Spatial Analysis)  
# -----  
  
gdf = gpd.GeoDataFrame(  
    df,  
    geometry=gpd.points_from_xy(df.longitude, df.latitude),  
    crs="EPSG:4326"  
)  
  
gdf.to_file("traffic_points.geojson", driver='GeoJSON')  
  
# ----- Save to Shape File:  
import geopandas as gpd  
  
# إنشاء DataFrame من البيانات  
gdf = gpd.GeoDataFrame(  
    df,  
    geometry=gpd.points_from_xy(df.longitude, df.latitude),  
    crs="EPSG:4326" # WGS 84  
)  
  
# حفظ البيانات كملف Shapefile  
gdf.to_file("traffic_points_shapefile.shp", driver='ESRI Shapefile')
```

```
# -----
```

## # 9. Data Quality Report

```
# -----
```

```
def check_data_quality(df):
    checks = {
        'total_rows': len(df),
        'missing_coordinates': df['latitude'].isnull().sum() + df['longitude'].isnull().sum(),
        'negative_traffic_counts': (df[['pedal_cycles', 'cars_and_taxis', 'all_HGVs']] < 0).any().any(),
        'duplicate_records': df.duplicated().sum()
    }
    return checks
```

```
quality_report = check_data_quality(df)
```

```
print("\nData Quality Report:")
for check, result in quality_report.items():
    print(f"{check}: {result}")
```



## قيم غير منطقية (Logical Errors)

العمود	التحقق
hour	هل القيم بين 0 و 23 فقط؟
total_vehicles	هل في قيم سالبة أو أكبر من مليون؟
link_length_km	هل يوجد قيم سالبة؟
vehicle_density	هل القيمة منطقية؟ مثلًا مش 100,000 مركبة/كم
bicycle_percentage و hgv_percentage	هل أكثر من 1 أو أقل من 0؟ (نسب غير منطقية)

```
print("Invalid hours:", df[~df['hour'].between(0, 23)].shape[0])  
  
print("Negative total_vehicles:", (df['total_vehicles'] < 0).sum())  
  
print("Suspiciously high density (> 10,000):", (df['vehicle_density'] > 10000).sum())  
  
print("Out-of-range percentages:", ((df['hgv_percentage'] > 1) | (df['hgv_percentage'] < 0)).sum())
```

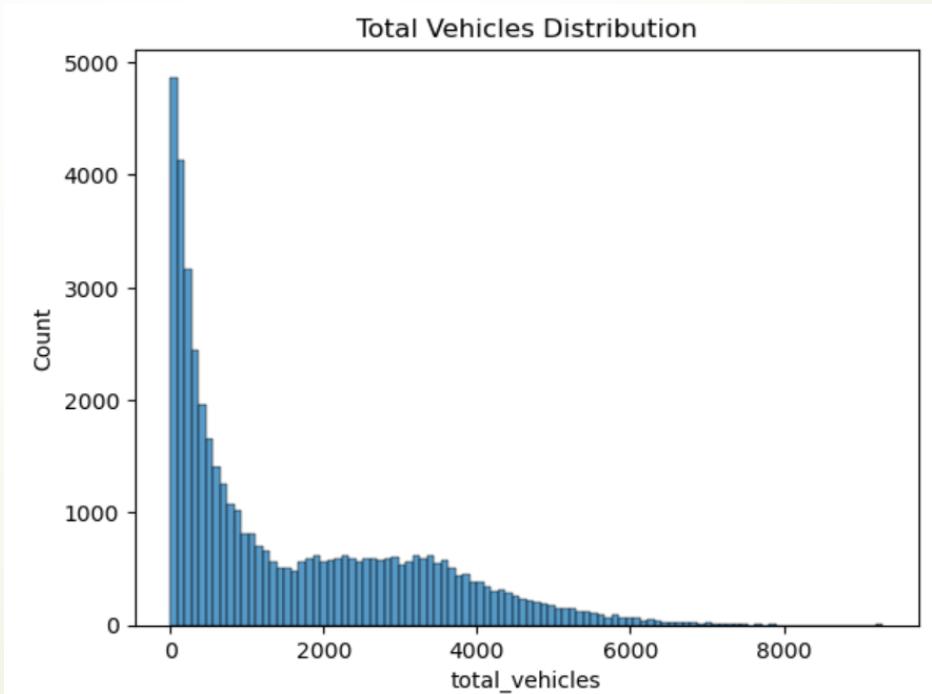
```
Invalid hours: 0  
Negative total_vehicles: 0  
Suspiciously high density (> 10,000): 0  
Out-of-range percentages: 0
```

## 2. توزيع البيانات

لو البيانات فيها skew ميل قوي أو outliers كثير، ممكן يؤثر على التحليل الإحصائي أو النمذجة.

❖ استخدم هذا الكود لرؤية التوزيع:

```
import matplotlib.pyplot as plt  
  
import seaborn as sns  
  
sns.histplot(df['total_vehicles'], bins=100)  
plt.title("Total Vehicles Distribution")  
plt.show()
```



### 3. تواریخ غیر منطقية

- هل عندك تواریخ في المستقبل؟
- هل بعض السجلات تواریخها ناقصة (NaT)؟

```
from datetime import datetime

print("Future dates:", df[df['count_date'] > datetime.today()].shape[0])
print("Missing dates:", df['count_date'].isnull().sum())

Future dates: 0
Missing dates: 0
```

4. نقاط جغرافية خارج بريطانيا؟ إذا البيانات من UK مثل DfT

• تأكد إن longitude و latitude داخل نطاق المملكة المتحدة

✖ تقريرياً:

• 61 إلى Lat: 49

• 2 إلى Lon: -9

```
uk_bounds = df[  
    (df['latitude'].between(49, 61)) &  
    (df['longitude'].between(-9, 2))  
]  
  
print("Outside UK bounds:", df.shape[0] - uk_bounds.shape[0])
```

Outside UK bounds: 0

5. تكرار غير متوقع في الـ ID أو المواقع.

هل عندك تكرار في count\_point\_id + count\_date + hour ؟ المفترض تكون فريدة.

```
duplicate_keys = df.duplicated(subset=['count_point_id', 'count_date', 'hour']).sum()  
print("Duplicated count_point_id + count_date + hour:", duplicate_keys)
```

Duplicated count\_point\_id + count\_date + hour: 23078

6. التناسق بين الأعمدة

مثلاً total\_vehicles : هل يساوي فعلياً مجموع مكوناته؟

```
df['calc_total'] = (  
    df['pedal_cycles'] +  
    df['two_wheeled_motor_vehicles'] +  
    df['cars_and_taxis'] +  
    df['buses_and_coaches'] +  
    df['LGVs'] +  
    df['all_HGVs'])
```

```
print("Mismatch in total_vehicles:", (df['total_vehicles'] != df['calc_total']).sum())  
Mismatch in total_vehicles: 0
```

## 7. التغطية الزمنية للبيانات

```
date_range = df['count_date'].dropna().sort_values().unique()  
  
print(f"Total distinct dates: {len(date_range)}")  
  
print(f"From {date_range.min()} to {date_range.max()}")
```

```
Total distinct dates: 1269  
From 2000-03-17 00:00:00 to 2024-10-08 00:00:00
```

كم عدد الأيام المختلفة في البيانات؟

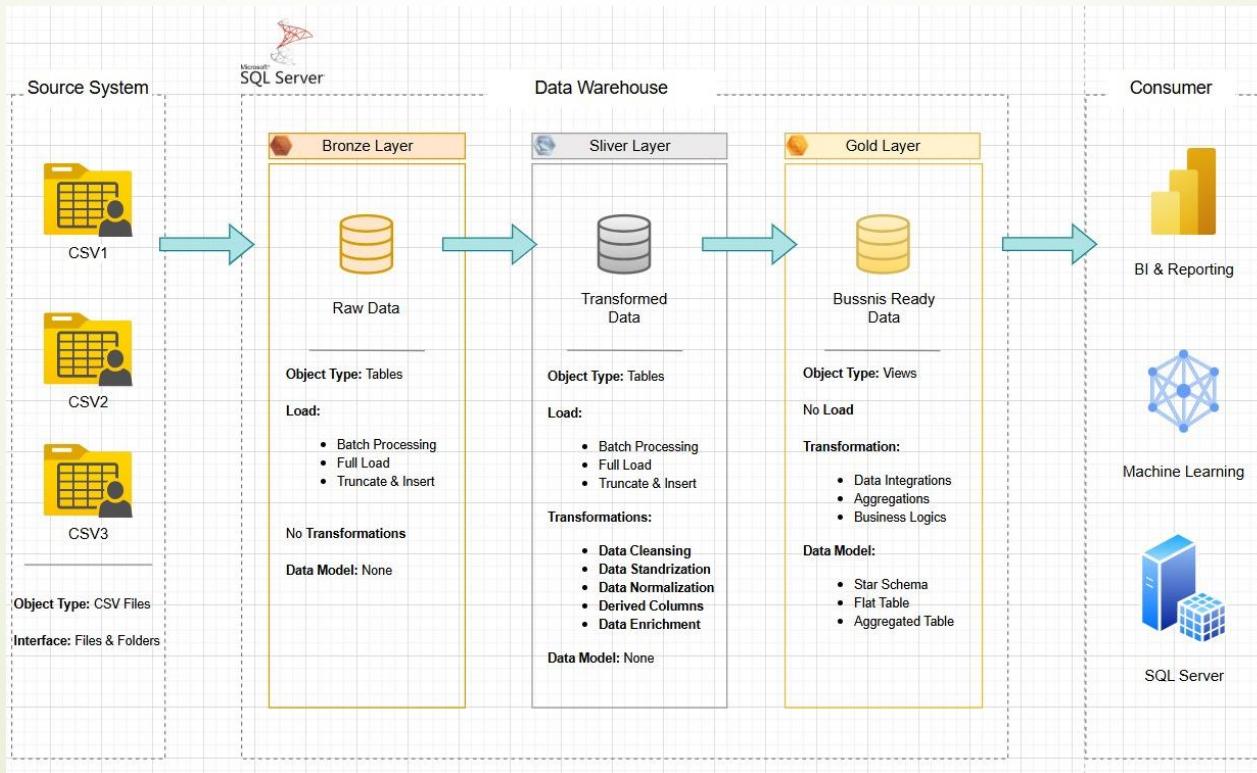
هل التواریخ متصلة؟ هل عندك فجوات؟

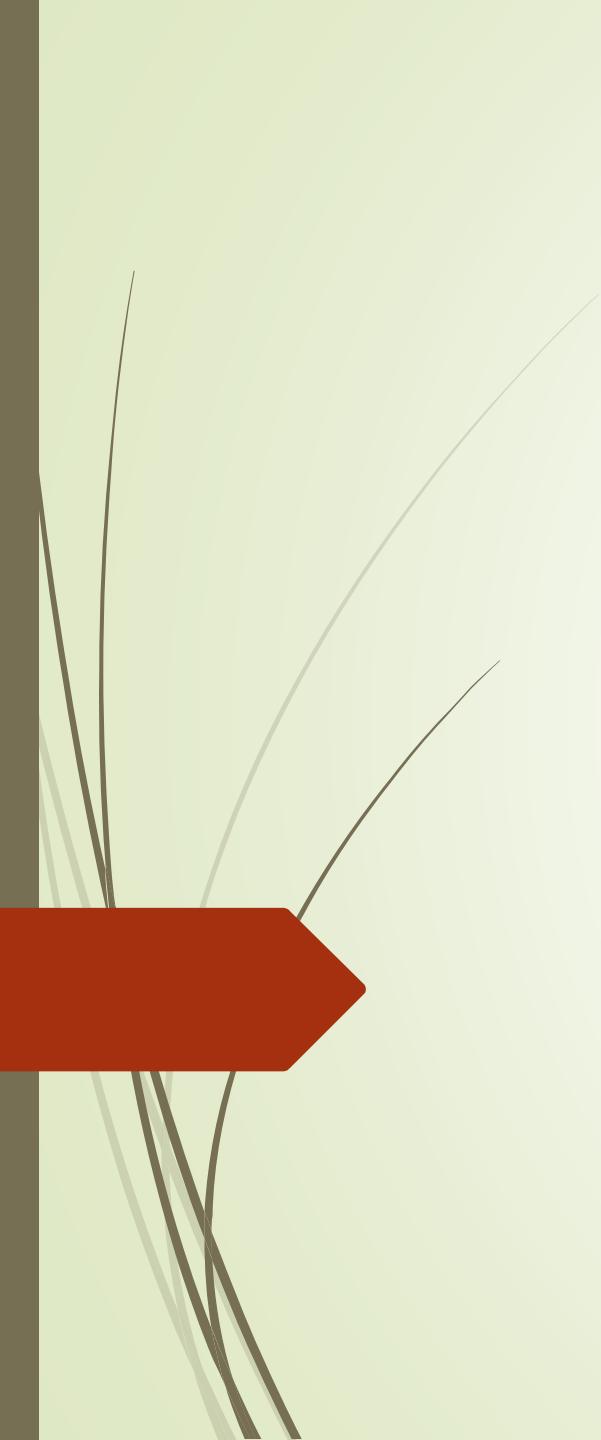
## 8. تحليل المناطق الفارغة(spatial gaps)

```
print("Unique count locations:", df[['latitude', 'longitude']].drop_duplicates().shape[0])  
  
Unique count locations: 467
```

هل هناك مناطق بلا بيانات إطلاقاً؟

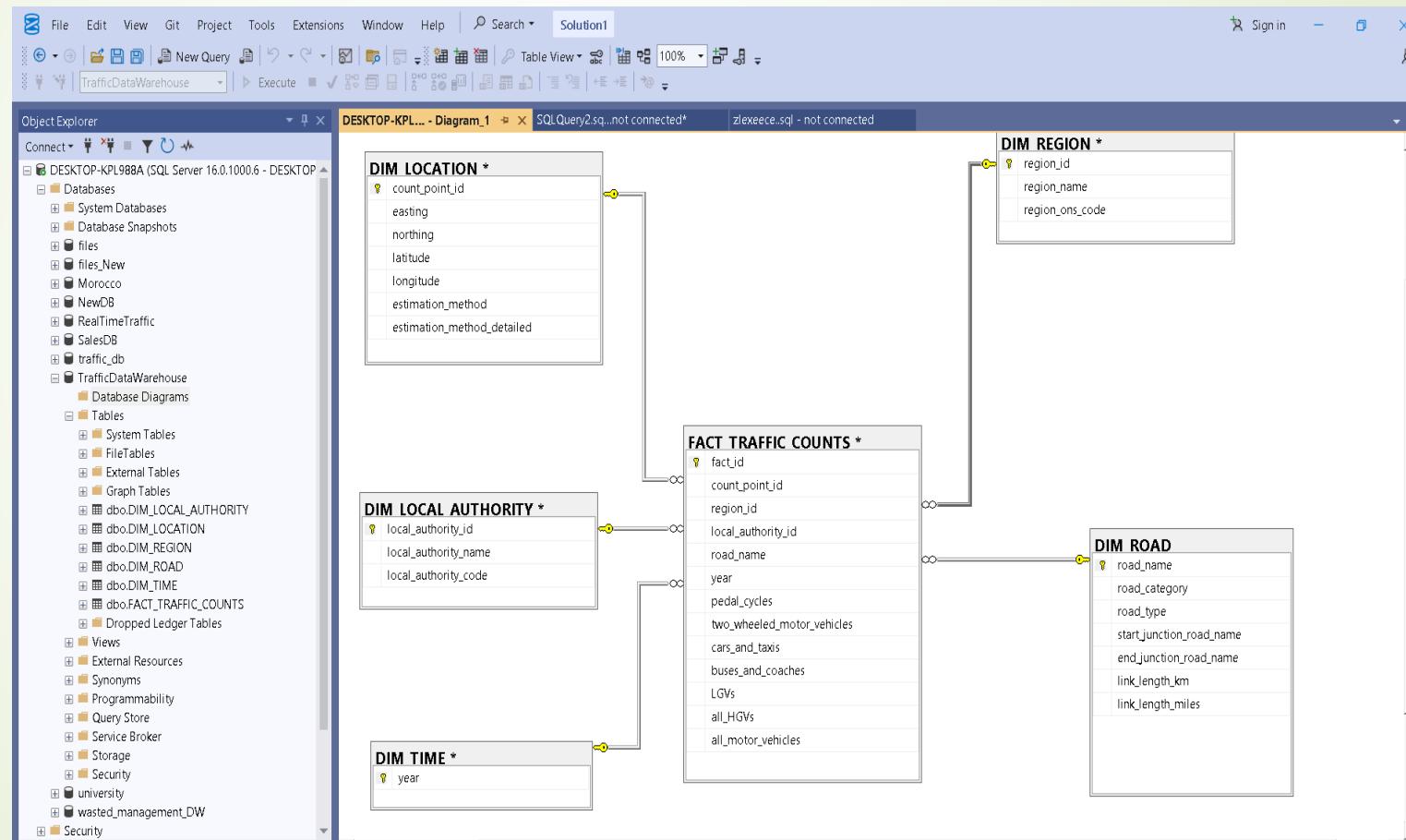
كم عدد النقاط الجغرافية الفريدة؟





# **SQL Server**

# Data After Restructure (Star Schema)



## Cleaning Data:

```
import pandas as pd
```

# تحميل الملف الأصلي 1.

```
file_path = "dft_traffic_counts_raw_counts3.csv" #
```

```
df = pd.read_csv(file_path)
```

# إزالة الصفوف المكررة 2.

```
df = df.drop_duplicates()
```

# إنشاء جداول الأبعاد 3.

```
# DIM_REGION
```

```
dim_region = df[['region_id', 'region_name',  
'region_ons_code']].drop_duplicates().reset_index(drop=True)
```



```
# DIM_LOCAL_AUTHORITY
```

```
dim_local_authority = df[['local_authority_id', 'local_authority_name',  
'local_authority_code']].drop_duplicates().reset_index(drop=True)
```

```
# DIM_LOCATION
```

```
dim_location = df[['count_point_id', 'easting', 'northing', 'latitude',  
'longitude']].drop_duplicates().reset_index(drop=True)
```

```
dim_location['estimation_method'] = 'N/A'
```

```
dim_location['estimation_method_detailed'] = 'N/A'
```

```
# DIM_ROAD
```

```
dim_road = df[['road_name', 'road_category', 'road_type', 'start_junction_road_name',  
'end_junction_road_name', 'link_length_km',  
'link_length_miles']].drop_duplicates().reset_index(drop=True)
```

```
# DIM_TIME
```

```
dim_time = df[['year']].drop_duplicates().reset_index(drop=True)
```

# ↗ إنشاء جدول الواقع .4 (Fact)

```
fact_traffic_counts = df[['count_point_id', 'region_id', 'local_authority_id', 'road_name', 'year',  
                           'pedal_cycles', 'two_wheeled_motor_vehicles', 'cars_and_taxis',  
                           'buses_and_coaches', 'LGVs', 'all_HGVs', 'all_motor_vehicles']].copy()
```

# ↗ إضافة مفتاح اساسي (fact\_id)

```
fact_traffic_counts.insert(0, 'fact_id', range(1, len(fact_traffic_counts) + 1))
```

# ↗ حفظ الجداول النظيفة .5

```
dim_region.to_csv("DIM_REGION.csv", index=False)  
dim_local_authority.to_csv("DIM_LOCAL_AUTHORITY.csv", index=False)  
dim_location.to_csv("DIM_LOCATION.csv", index=False)  
dim_road.to_csv("DIM_ROAD.csv", index=False)  
dim_time.to_csv("DIM_TIME.csv", index=False)  
fact_traffic_counts.to_csv("FACT_TRAFFIC_COUNTS.csv", index=False)
```

print("تم إنشاء الملفات بنجاح ↗")

```
print("- DIM_REGION.csv")  
print("- DIM_LOCAL_AUTHORITY.csv")  
print("- DIM_LOCATION.csv")  
print("- DIM_ROAD.csv")  
print("- DIM_TIME.csv")  
print("- FACT_TRAFFIC_COUNTS.csv")
```



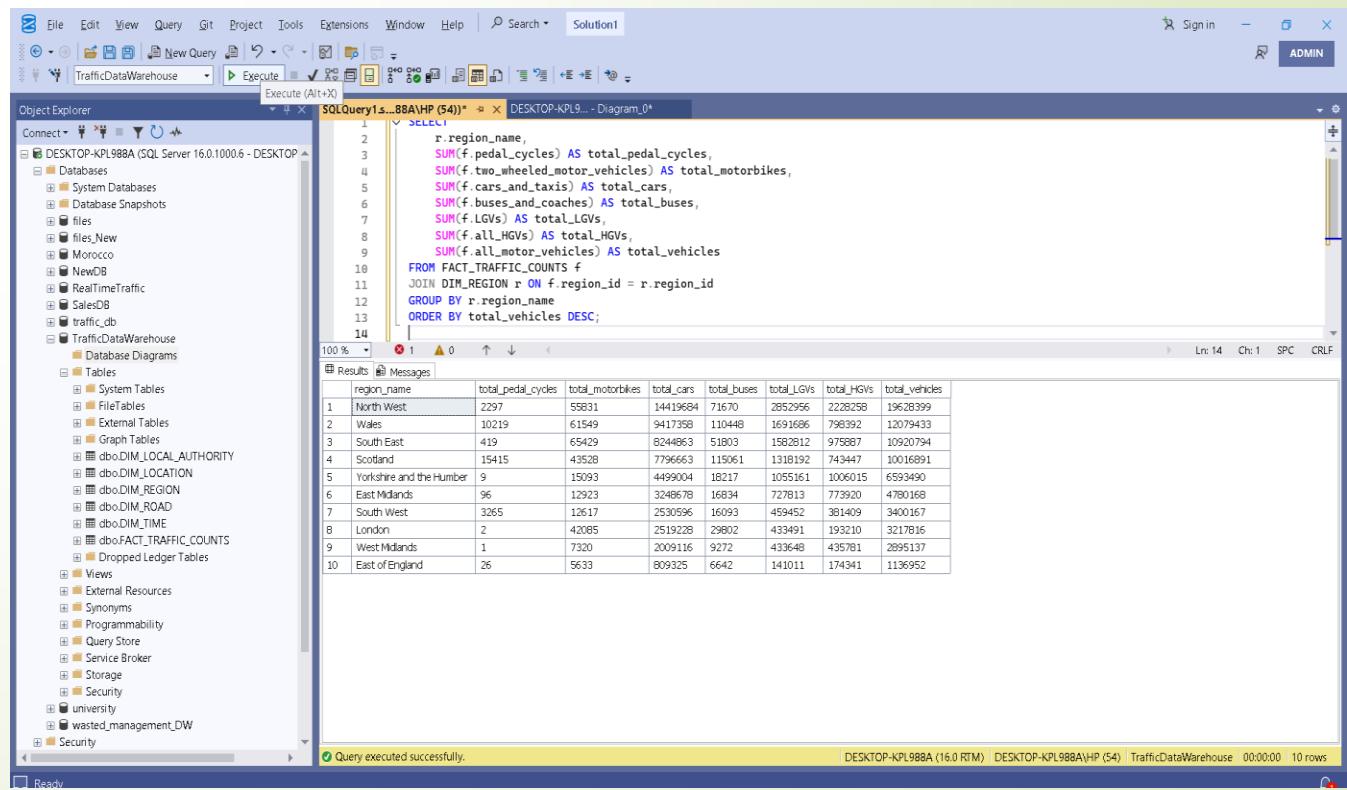
# Data Analysis:

## Traffic by Region

(تحليل إجمالي عدد المركبات لكل إقليم) Region):

```
USE TrafficDataWarehouse;  
GO
```

```
SELECT  
    r.region_name,  
    SUM(f.pedal_cycles) AS total_pedal_cycles,  
    SUM(f.two_wheeled_motor_vehicles) AS total_motorbikes,  
    SUM(f.cars_and_taxis) AS total_cars,  
    SUM(f.buses_and_coaches) AS total_buses,  
    SUM(f.LGVs) AS total_LGVs,  
    SUM(f.all_HGVs) AS total_HGVs,  
    SUM(f.all_motor_vehicles) AS total_vehicles  
FROM FACT_TRAFFIC_COUNTS f  
JOIN DIM_REGION r ON f.region_id = r.region_id  
GROUP BY r.region_name  
ORDER BY total_vehicles DESC;
```



The screenshot shows the SQL Server Management Studio interface with the following details:

- Object Explorer:** Shows the database structure, including the TrafficDataWarehouse database and its tables like FACT\_TRAFFIC\_COUNTS and DIM\_REGION.
- SQL Query Editor:** Contains the T-SQL query provided in the slide.
- Results Grid:** Displays the output of the query, listing regions and their traffic counts. The columns are: region\_name, total\_pedal\_cycles, total\_motorbikes, total\_cars, total\_buses, total\_LGVs, total\_HGVs, and total\_vehicles.
- Status Bar:** Shows "Query executed successfully." and the session details: DESKTOP-KPL988A (16.0 RTM) | DESKTOP-KPL988A\HP (54) | TrafficDataWarehouse | 00:00:00 | 10 rows.

region_name	total_pedal_cycles	total_motorbikes	total_cars	total_buses	total_LGVs	total_HGVs	total_vehicles
North West	2297	55831	14419684	71670	2852956	2228258	19628399
Wales	10219	61549	9417358	110448	1691686	796392	12079433
South East	419	65429	8244863	51803	1582812	975687	10920794
Scotland	15415	43528	7796663	115061	1318192	743447	10016991
Yorkshire and the Humber	9	15093	4499004	18217	1055161	1006015	6593490
East Midlands	96	12923	3248678	16834	727813	773920	4780168
South West	3265	12617	2530596	16093	459452	381409	3400167
London	2	42085	2519228	29802	433491	193210	3217816
West Midlands	1	7320	2009116	9272	433648	435781	2995137
East of England	26	5633	809325	6642	141011	174341	1136952

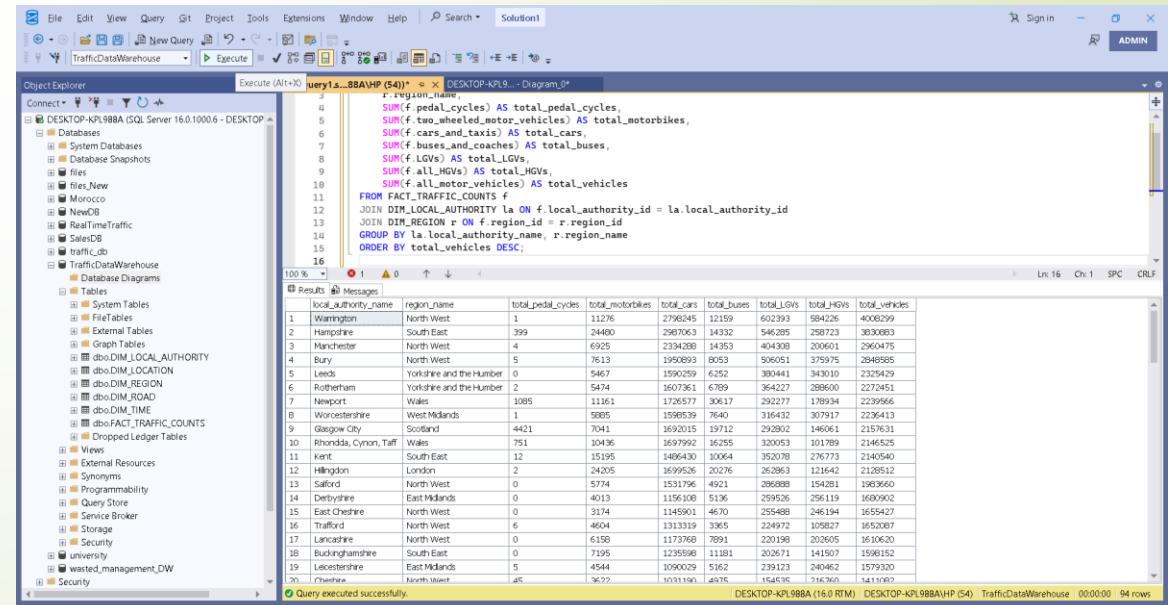
## Traffic by Local Authority

(تحليل إجمالي المركبات لكل سلطة محلية) (Local Authority):

```

SELECT
    la.local_authority_name,
    r.region_name,
    SUM(f.pedal_cycles) AS total_pedal_cycles,
    SUM(f.two_wheeled_motor_vehicles) AS total_motorbikes,
    SUM(f.cars_and_taxis) AS total_cars,
    SUM(f.buses_and_coaches) AS total_buses,
    SUM(f.LGVs) AS total_LGVs,
    SUM(f.all_HGVs) AS total_HGVs,
    SUM(f.all_motor_vehicles) AS total_vehicles
FROM FACT_TRAFFIC_COUNTS f
JOIN DIM_LOCAL_AUTHORITY la ON f.local_authority_id = la.local_authority_id
JOIN DIM_REGION r ON f.region_id = r.region_id
GROUP BY la.local_authority_name, r.region_name
ORDER BY total_vehicles DESC;

```



```

-- Query 1: SELECT * FROM FACT_TRAFFIC_COUNTS
-- Results:
local_authority_name region_name total_pedal_cycles total_motorbikes total_cars total_buses total_LGVs total_HGVs total_vehicles
1 Warrington North West 1 11276 2796245 12159 602393 58426 4006299
2 Hampshire South East 399 24460 2967063 14332 546285 258723 3800863
3 Manchester North West 4 6925 2394208 14353 404306 200601 290475
4 Bury North West 5 7613 1950891 8053 506051 375975 2848985
5 Leeds Yorkshire and the Humber 0 5467 1590259 6252 380441 343010 2335429
6 Rotherham Yorkshire and the Humber 2 5474 160761 679 364227 286600 2272451
7 Newport Wales 1085 11161 1786577 30617 292277 178934 2239966
8 Worcestershire West Midlands 1 5965 1596539 7640 316432 307917 2228413
9 Glasgow Scotland 4421 7041 1692015 19712 292658 26161 2317631
10 Rhondda, Cynon, Taff Wales 751 10436 1657992 16255 320053 161789 2446525
11 Oldham North East 10 15195 1405454 10554 30308 2272451 2446525
12 Hillingdon London 2 24205 1699526 20276 262863 121642 21298512
13 Salford North West 0 5774 1531796 4921 286889 154281 1983660
14 Derbyshire East Midlands 0 4013 1156108 5136 259526 256119 1690092
15 East Cheshire North West 0 3174 1145901 4670 255489 246194 1655427
16 Trafford North West 6 4604 1313319 3365 224972 105827 1652087
17 Lancashire North West 0 6158 1173768 7691 220198 202605 1610620
18 Buckinghamshire South East 0 7195 1285850 11181 202671 141907 1596152
19 Leicestershire East Midlands 5 4544 1090029 5162 239123 240462 1579320
20 Cheshire North West 45 9622 10311590 4976 154035 216260 1411192

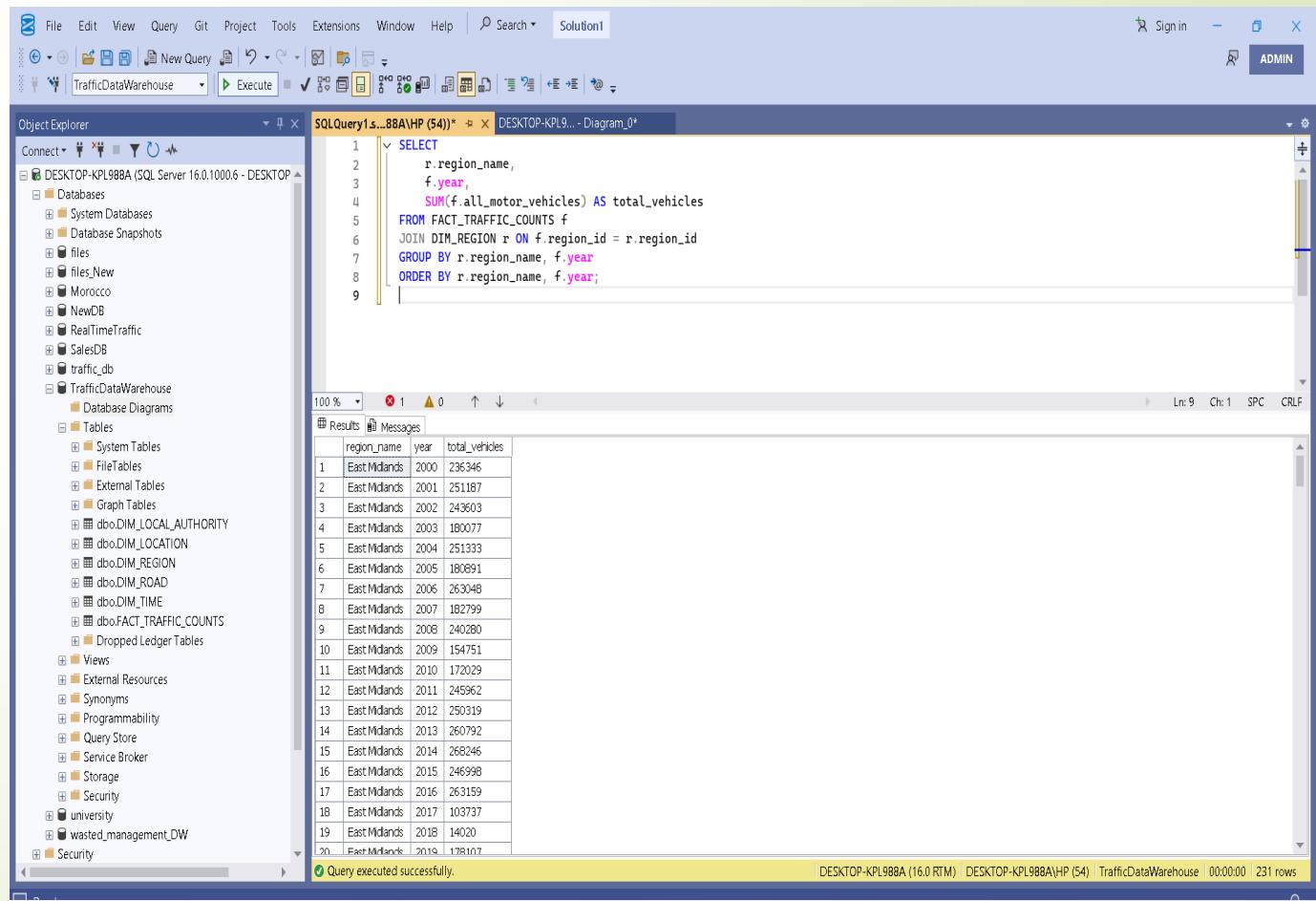
```

Query executed successfully.

## Traffic Trend by Region (Over Time)

إذا تري التحليل عبر السنوات:

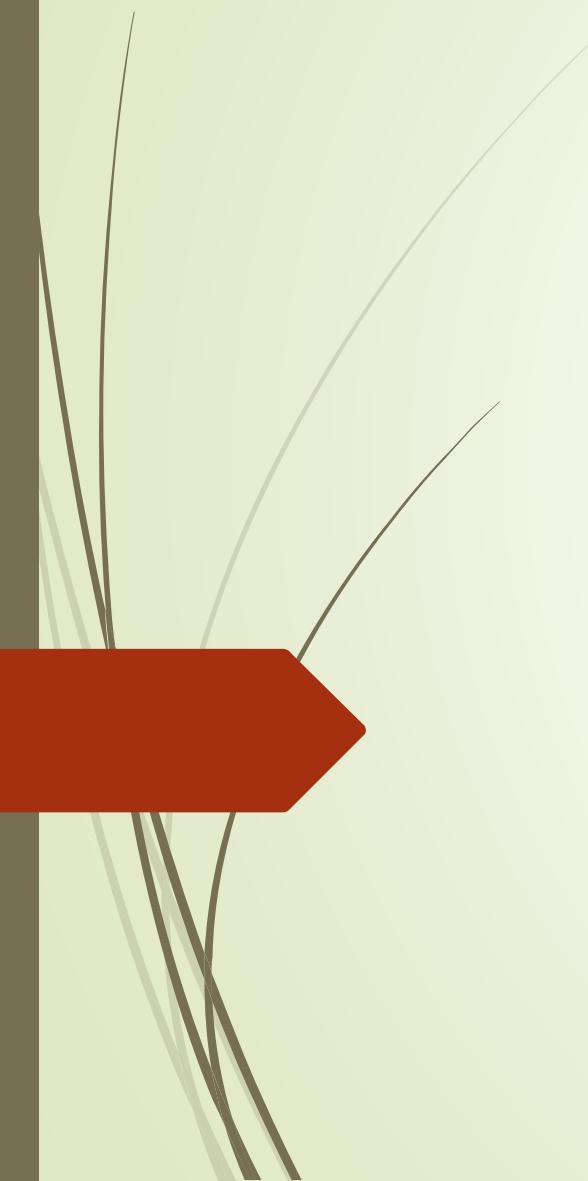
```
SELECT
    r.region_name,
    f.year,
    SUM(f.all_motor_vehicles) AS total_vehicles
FROM FACT_TRAFFIC_COUNTS f
JOIN DIM_REGION r ON f.region_id = r.region_id
GROUP BY r.region_name, f.year
ORDER BY r.region_name, f.year;
```



The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. The top bar includes File, Edit, View, Query, Git, Project, Tools, Extensions, Window, Help, and a Search dropdown. On the right, there's a "Sign in" button and an "ADMIN" user indicator. The main window has tabs for "TrafficDataWarehouse" and "DESKTOP-KPL98A\HP (54)\*". The "Object Explorer" pane on the left lists databases like DESKTOP-KPL98A, System Databases, and traffic\_db. The "SQLQuery1.sql" tab contains the provided T-SQL code. The "Results" pane at the bottom displays the query results as a table:

region_name	year	total_vehicles
East Midlands	2000	236346
East Midlands	2001	251187
East Midlands	2002	243603
East Midlands	2003	180077
East Midlands	2004	251333
East Midlands	2005	180891
East Midlands	2006	263048
East Midlands	2007	182799
East Midlands	2008	240280
East Midlands	2009	154751
East Midlands	2010	172029
East Midlands	2011	245962
East Midlands	2012	250319
East Midlands	2013	260792
East Midlands	2014	268246
East Midlands	2015	246598
East Midlands	2016	263159
East Midlands	2017	103737
East Midlands	2018	14020
East Midlands	2019	178107

At the bottom of the results pane, a message says "Query executed successfully." The status bar at the bottom right shows "DESKTOP-KPL98A (16.0 RTM) DESKTOP-KPL98A\HP (54) TrafficDataWarehouse 00:00:00 231 rows".



يمكنك بعدها رسماها بيانياً في BI أو Python Matplotlib أو Plotly.

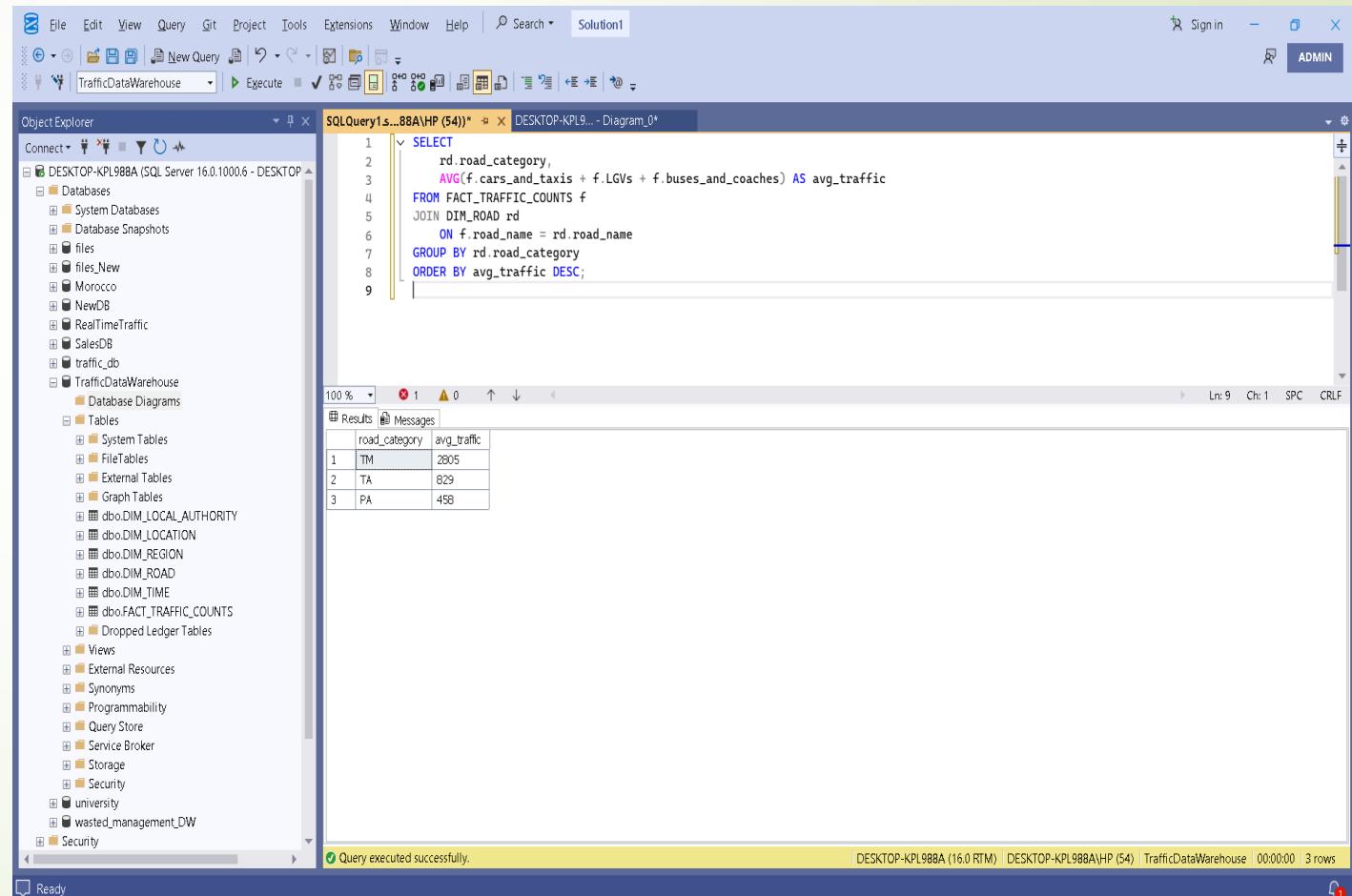
**الميكل لديك حالياً**

- جدول الواقع: FACT\_TRAFFIC\_COUNTS
- جدول الطرق: DIM\_ROAD
- العلاقة:

FACT\_TRAFFIC\_COUNTS.road\_name → •  
DIM\_ROAD.road\_name

↙ الاستعلام الصحيح في SQL Server

```
SELECT
    rd.road_category,
    AVG(f.cars_and_taxis + f.LGVs + f.buses_and_coaches) AS avg_traffic
FROM FACT_TRAFFIC_COUNTS f
JOIN DIM_ROAD rd
    ON f.road_name = rd.road_name
GROUP BY rd.road_category
ORDER BY avg_traffic DESC;
```



The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. The query window displays the following SQL code:

```
SELECT
    rd.road_category,
    AVG(f.cars_and_taxis + f.LGVs + f.buses_and_coaches) AS avg_traffic
FROM FACT_TRAFFIC_COUNTS f
JOIN DIM_ROAD rd
    ON f.road_name = rd.road_name
GROUP BY rd.road_category
ORDER BY avg_traffic DESC;
```

The results pane shows the following data:

road_category	avg_traffic
TM	2805
TA	829
PA	458

At the bottom of the results pane, a message indicates: "Query executed successfully."

## توضيح الفكرة

هذا التحليل يحسب متوسط حجم الحركة المرورية لكل فئة طريق.

- تم جمع ثلاثة فئات من المركبات:

cars\_and\_taxis + LGVs + buses\_and\_coaches

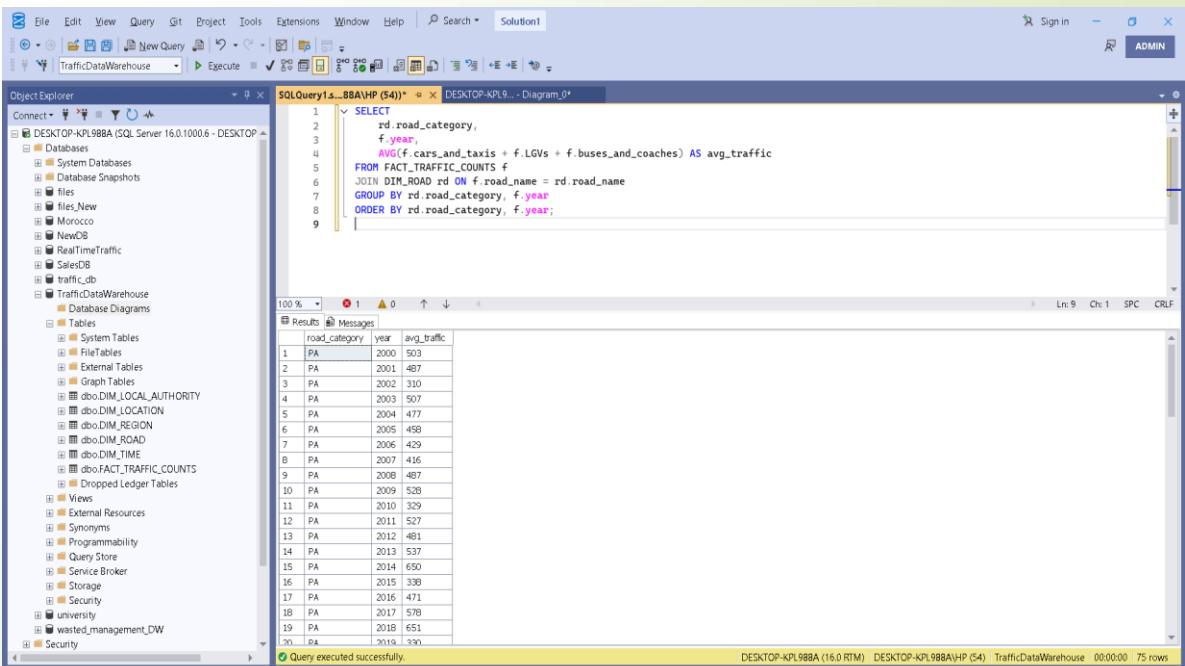
- يمكن تعديل المعادلة لتشمل فئات إضافية مثل HGVs أو pedal\_cycles.

## تحليل أكثر تقدماً

يمكنك توسيع التحليل بإضافة سنة أو منطقة لفهم التغير الزمني أو الجغرافي:

تحليل فئة الطريق عبر السنوات:

```
SELECT
    rd.road_category,
    f.year,
    AVG(f.cars_and_taxis + f.LGVs + f.buses_and_coaches) AS avg_traffic
FROM FACT_TRAFFIC_COUNTS f
JOIN DIM_ROAD rd ON f.road_name = rd.road_name
GROUP BY rd.road_category, f.year
ORDER BY rd.road_category, f.year;
```



The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. In the center, there is a query window titled 'SQLQuery1...'. The query itself is:

```
SELECT
    rd.road_category,
    f.year,
    AVG(f.cars_and_taxis + f.LGVs + f.buses_and_coaches) AS avg_traffic
FROM FACT_TRAFFIC_COUNTS f
JOIN DIM_ROAD rd ON f.road_name = rd.road_name
GROUP BY rd.road_category, f.year
ORDER BY rd.road_category, f.year;
```

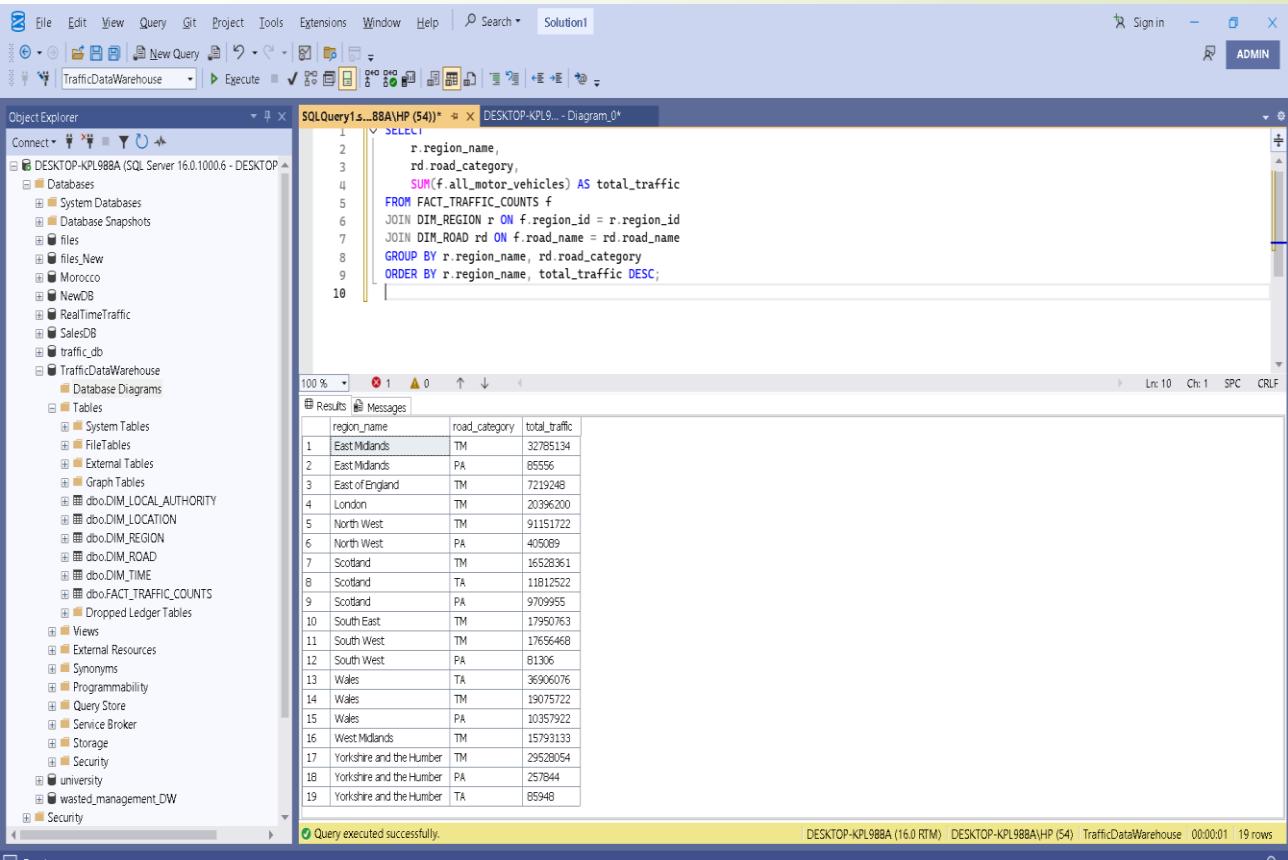
Below the query window is a results grid titled 'Results'. The data returned by the query is:

road_category	year	avg_traffic
PA	2000	503
PA	2001	487
PA	2002	310
PA	2003	507
PA	2004	477
PA	2005	458
PA	2006	429
PA	2007	416
PA	2008	487
PA	2009	528
PA	2010	329
PA	2011	527
PA	2012	481
PA	2013	537
PA	2014	650
PA	2015	338
PA	2016	471
PA	2017	578
PA	2018	651
PA	2019	390

At the bottom of the screen, a status bar indicates 'Query executed successfully.' and shows the connection details: DESKTOP-KPL98A (16.0 RTM) | DESKTOP-KPL98A\HP (54) | TrafficDataWarehouse | 00:00:00 | 75 rows.

## □ تحليل فئة الطريق حسب الأقاليم:

```
SELECT
    r.region_name,
    rd.road_category,
    SUM(f.all_motor_vehicles) AS total_traffic
FROM FACT_TRAFFIC_COUNTS f
JOIN DIM_REGION r ON f.region_id = r.region_id
JOIN DIM_ROAD rd ON f.road_name = rd.road_name
GROUP BY r.region_name, rd.road_category
ORDER BY r.region_name, total_traffic DESC;
```



```
SELECT
    r.region_name,
    rd.road_category,
    SUM(f.all_motor_vehicles) AS total_traffic
FROM FACT_TRAFFIC_COUNTS f
JOIN DIM_REGION r ON f.region_id = r.region_id
JOIN DIM_ROAD rd ON f.road_name = rd.road_name
GROUP BY r.region_name, rd.road_category
ORDER BY r.region_name, total_traffic DESC;
```

region_name	road_category	total_traffic
East Midlands	TM	32785134
East Midlands	PA	65556
East of England	TM	7219248
London	TM	20396200
North West	TM	91151722
North West	PA	405089
Scotland	TM	16528361
Scotland	TA	11812522
Scotland	PA	9709955
South East	TM	17950763
South West	TM	17656468
South West	PA	81306
Wales	TA	36906076
Wales	TM	19075722
Wales	PA	10357922
West Midlands	TM	15793133
Yorkshire and the Humber	TM	29528054
Yorkshire and the Humber	PA	257844
Yorkshire and the Humber	TA	65948

# Plot:

```
import geopandas as gpd
import pandas as pd
import matplotlib.pyplot as plt
from sqlalchemy import create_engine

#إعداد معلومات الاتصال
server = 'localhost' #اسم السيرفر المحلي
database = 'traffic_db' #اسم قاعدة البيانات

#الاتصال بدون مصادقة (No username/password)
#if server == 'localhost':
#    conn_str = f"mssql+pyodbc://sa@{server}/{database}?driver=ODBC+Driver+17+for+SQL+Server"

#إنشاء المحرك (engine)
engine = create_engine(conn_str)

#الاستعلام: جمع عدد السيارات حسب الإحداثيات
query = """
SELECT
    cp.latitude,
    cp.longitude,
    SUM(tc.cars_and_taxis) AS total_cars
FROM
    [TrafficDataWarehouse].[dbo].[DIM_LOCATION] cp
JOIN
    [TrafficDataWarehouse].[dbo].[FACT_TRAFFIC_COUNTS] tc ON cp.count_point_id =
        tc.count_point_id
GROUP BY
    cp.latitude, cp.longitude
"""

#الاتصال بدون مصادقة (No username/password)
#إذا كان السيرفر لا يتطلب تسجيل دخول (يسمح بالوصول العام)
#conn_str = f"mssql+pyodbc://sa@{server}/{database}?driver=ODBC+Driver+17+for+SQL+Server"
```

The screenshot shows a Jupyter Notebook interface running on a local host. The notebook has a single cell containing Python code for reading data from a SQL database and creating a map plot.

```
# تحويل النتائج إلى DataFrame
df = pd.read_sql(query, engine)

# التحقق من أول صفوف النتائج
print(df.head())

# تحويل البيانات إلى GeoDataFrame (نقطة مكتبة)
gdf = gpd.GeoDataFrame(df, geometry=gpd.points_from_xy(df.longitude, df.latitude), crs="EPSG:4326")

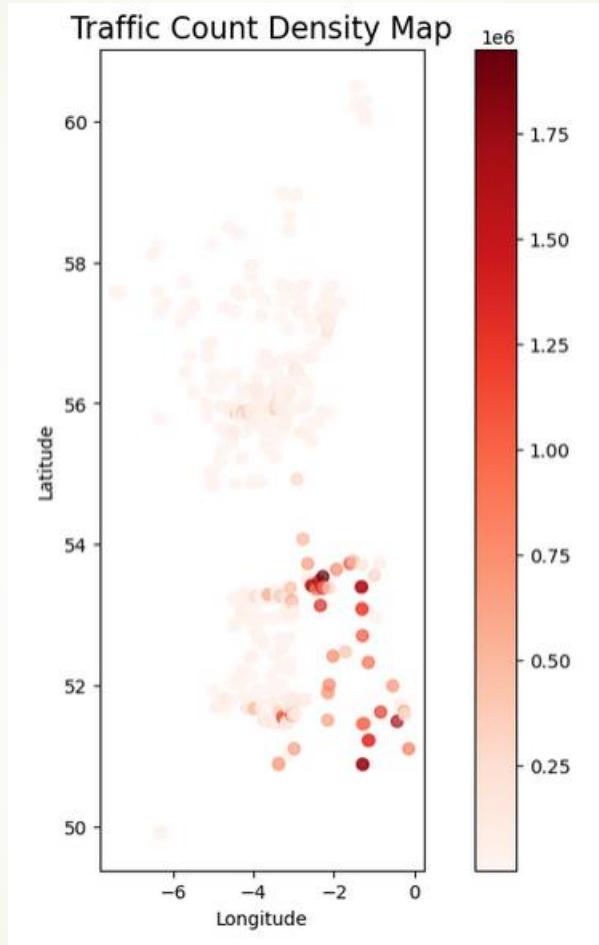
# رسم خريطة الكثافة المرورية
fig, ax = plt.subplots(figsize=(10, 8))
gdf.plot(column='total_cars', legend=True, cmap='Reds', ax=ax, markersize=40, alpha=0.7)

ax.set_title('Traffic Count Density Map', fontsize=16)
ax.set_xlabel('Longitude')
ax.set_ylabel('Latitude')
plt.show()
```

The notebook also displays a table of data and a resulting map plot titled "Traffic Count Density Map".

	latitude	longitude	total_cars
0	51.302578	NaN	1486430
1	51.362068	NaN	721333
2	57.588097	-7.442666	1428
3	57.559101	-7.320952	495
4	58.110603	-6.509713	942

Traffic Count Density Map  $1e6$



## Data Analysis – Date time:

### ⇨ الهدف من التحليل

تحليل بيانات الحركة المرورية Traffic Counts حسب الوقت، لفهم الأنماط الزمنية مثل:

- توزيع عدد المركبات حسب السنة / الشهر / اليوم / الساعة
- تحديد ساعات الذروة Peak Hours
- حساب إجمالي وعدد متوسط المركبات في أوقات مختلفة
- معرفة الاتجاهات الزمنية (يومي، أسبوعي، سنوي)

---

### 1. □ تحويل التاريخ والساعة إلى حقل تاريخ/وقت موحد

إذا كان لديك عمود count\_date تاريخ فقط وعمود hour رقم الساعة، يمكنك تكوين عمود جديد يمثل الوقت الكامل:

```
ALTER TABLE FACT_TRAFFIC_COUNTS1  
ADD count_datetime AS  
DATEADD(HOUR, hour, CAST(count_date AS DATETIME));
```

الآن أصبح لديك عمود count\_datetime يمثل التاريخ والساعة معًا، يمكن استخدامه للتحليل الزمني.

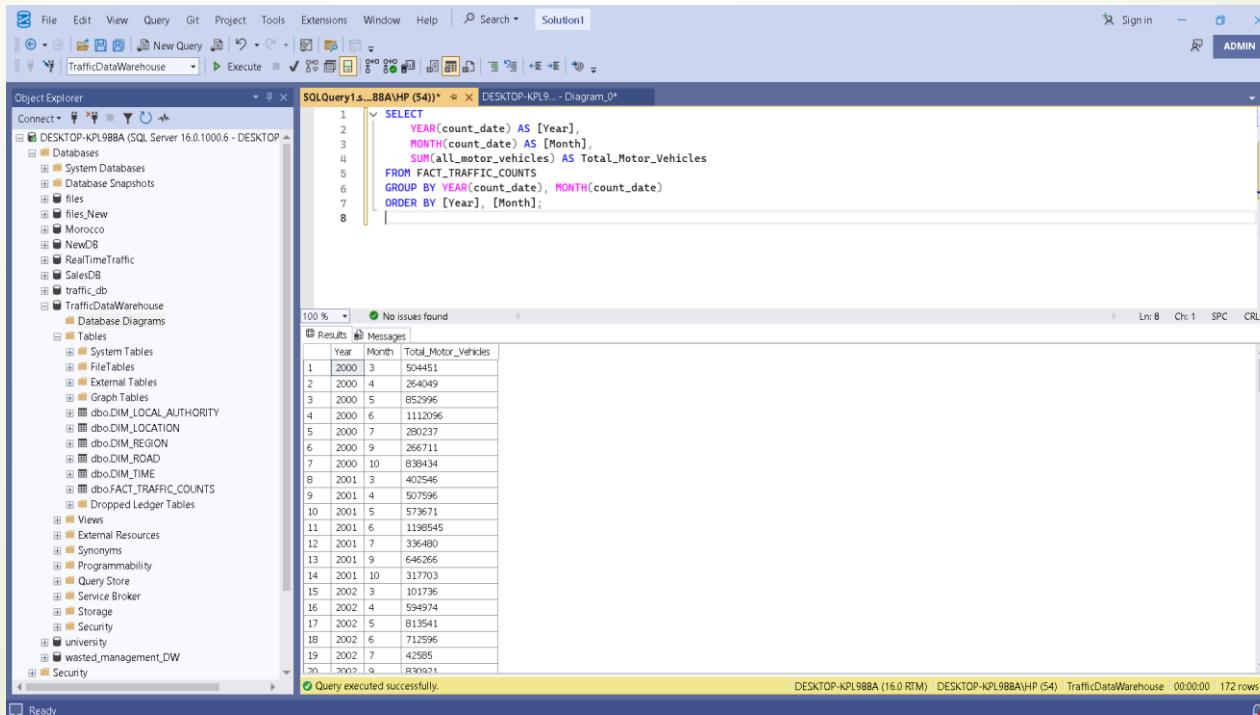
---

## 2. تحليل إجمالي المركبات حسب السنة والشهر

SELECT

```
YEAR(count_date) AS [Year],  
MONTH(count_date) AS [Month],  
SUM(all_motor_vehicles) AS Total_Motor_Vehicles  
FROM FACT_TRAFFIC_COUNTS  
GROUP BY YEAR(count_date), MONTH(count_date)  
ORDER BY [Year], [Month];
```

هذا التحليل يوضح الاتجاه العام لحركة المرور على مدار السنة.



The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. The top menu bar includes File, Edit, View, Query, Git, Project, Tools, Extensions, Window, Help, and a Search field. The title bar indicates the connection is to 'TrafficDataWarehouse' on 'DESKTOP-KPL98A (SQL Server 16.0.1000.6 - DESKTOP-KPL98A)' with a session ID of '54'. The status bar at the bottom right shows '00:00:00 172 rows'.

The left pane is the Object Explorer, showing the database structure. Under 'TrafficDataWarehouse', there are several tables listed under the 'Tables' node, including 'dbo.DIM\_LOCAL\_AUTHORITY', 'dbo.DIM\_LOCATION', 'dbo.DIM\_REGION', 'dbo.DIM\_ROAD', 'dbo.DIM\_TIME', 'dbo.FACT\_TRAFFIC\_COUNTS', and 'Dropped Ledger Tables'.

The main pane displays the T-SQL query:

```
SELECT  
    YEAR(count_date) AS [Year],  
    MONTH(count_date) AS [Month],  
    SUM(all_motor_vehicles) AS Total_Motor_Vehicles  
FROM FACT_TRAFFIC_COUNTS  
GROUP BY YEAR(count_date), MONTH(count_date)  
ORDER BY [Year], [Month];
```

Below the query, the results pane shows a table with 172 rows of data:

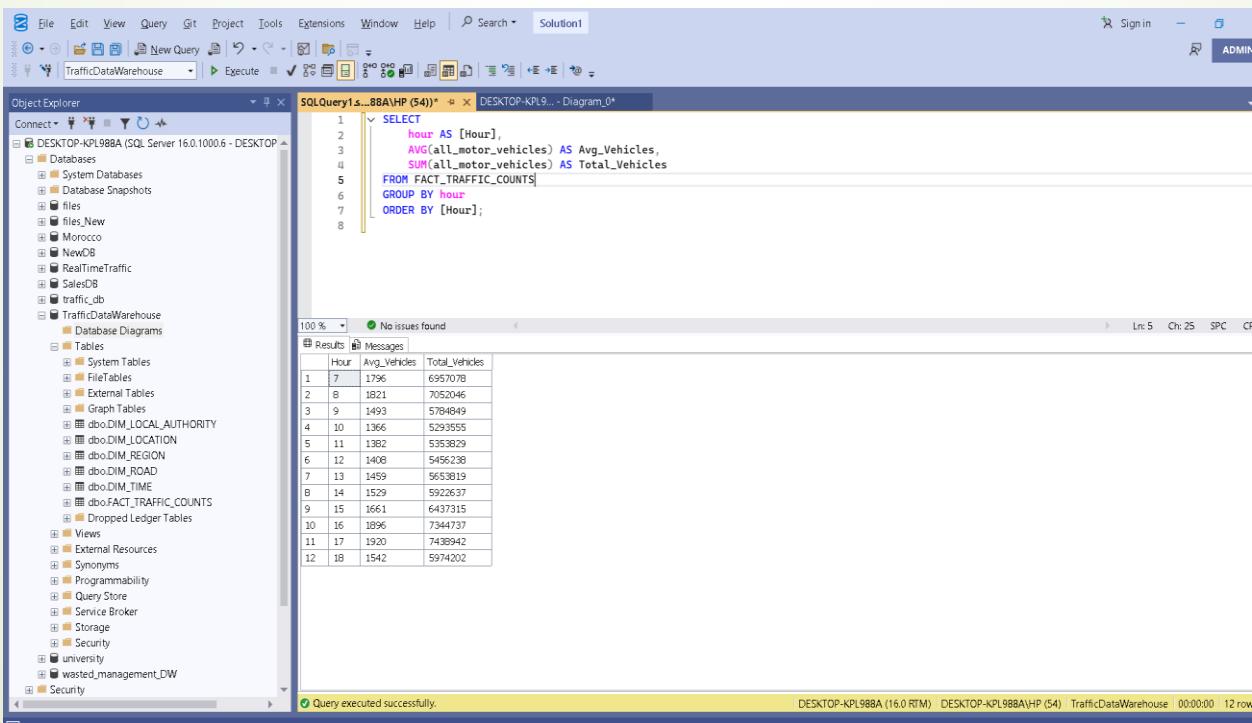
	Year	Month	Total_Motor_Vehicles
1	2000	3	504451
2	2000	4	264049
3	2000	5	852996
4	2000	6	1112096
5	2000	7	280237
6	2000	9	266711
7	2000	10	838434
8	2001	3	402546
9	2001	4	507596
10	2001	5	573671
11	2001	6	1198545
12	2001	7	336480
13	2001	9	646266
14	2001	10	317703
15	2002	3	101736
16	2002	4	594974
17	2002	5	813541
18	2002	6	712596
19	2002	7	42585
20	2002	9	839921

At the bottom of the results pane, a message states 'Query executed successfully.'

### ① تحليل عدد المركبات حسب الساعة

```
SELECT
    hour AS [Hour],
    AVG(all_motor_vehicles) AS Avg_Vehicles,
    SUM(all_motor_vehicles) AS Total_Vehicles
FROM FACT_TRAFFIC_COUNTS
GROUP BY hour
ORDER BY [Hour];
```

هذا الاستعلام يظهر متوسط وعدد المركبات في كل ساعة لتحديد أوقات الذروة.



The screenshot shows the SQL Server Management Studio interface. On the left, the Object Explorer displays the database structure, including the TrafficDataWarehouse database and its tables like FACT\_TRAFFIC\_COUNTS. In the center, the SQL Query Editor window contains the T-SQL code provided above. Below the editor, the Results pane shows the output of the query, which is a table with two columns: Hour and Total\_Vehicles. The data consists of 12 rows, each representing an hour from 1 to 12, showing the total number of vehicles counted. At the bottom of the Results pane, it says "Query executed successfully."

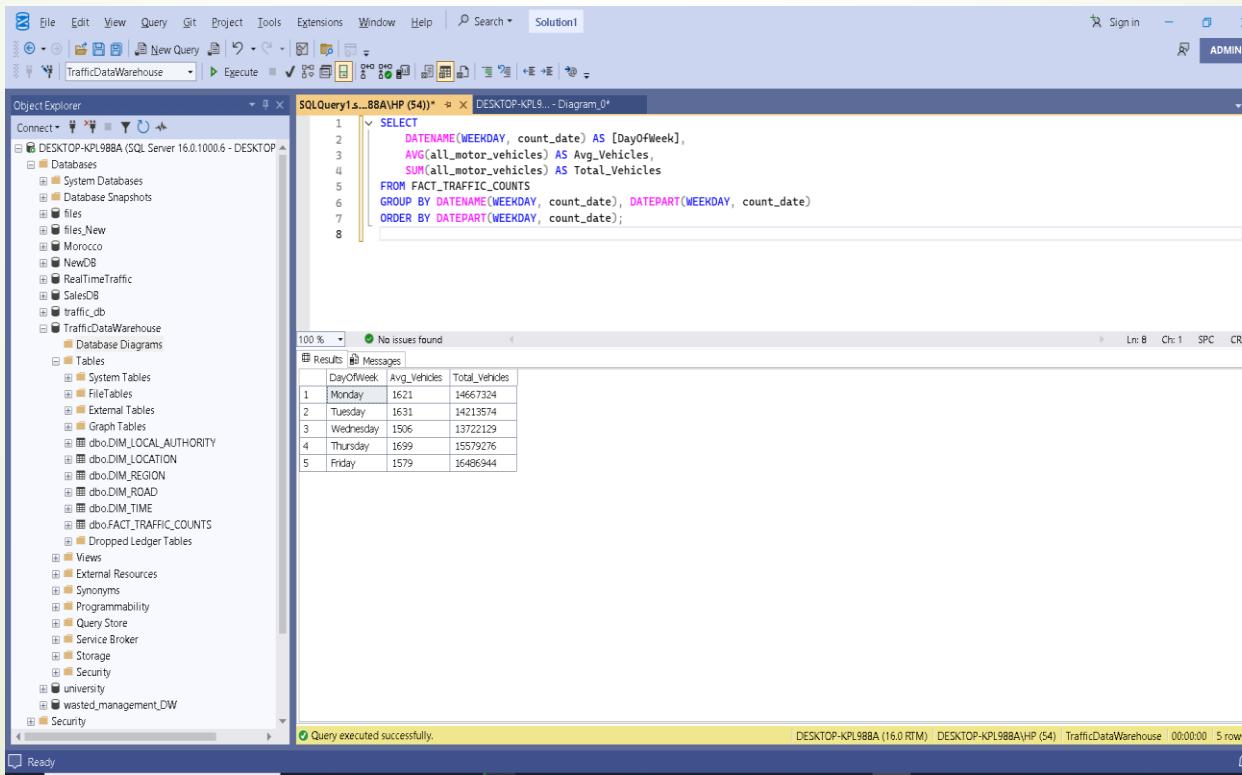
Hour	Total_Vehicles
1	1796
2	1821
3	1493
4	1366
5	1382
6	1406
7	1459
8	1529
9	1661
10	1896
11	1920
12	1542

## ٤. تحليل حسب اليوم في الأسبوع

SELECT

```
DATENAME(WEEKDAY, count_date) AS [DayOfWeek],  
AVG(all_motor_vehicles) AS Avg_Vehicles,  
SUM(all_motor_vehicles) AS Total_Vehicles  
FROM FACT_TRAFFIC_COUNTS  
GROUP BY DATENAME(WEEKDAY, count_date),  
DATEPART(WEEKDAY, count_date)  
ORDER BY DATEPART(WEEKDAY, count_date);
```

يساعد على معرفة أكثر الأيام ازدحاماً في الأسبوع.



The screenshot shows the SSMS interface with a query window titled 'SOQuery1...88A\HP (54)\*'. The query is:

```
1 SELECT  
2     DATENAME(WEEKDAY, count_date) AS [DayOfWeek],  
3     AVG(all_motor_vehicles) AS Avg_Vehicles,  
4     SUM(all_motor_vehicles) AS Total_Vehicles  
5 FROM FACT_TRAFFIC_COUNTS  
6 GROUP BY DATENAME(WEEKDAY, count_date), DATEPART(WEEKDAY, count_date)  
7 ORDER BY DATEPART(WEEKDAY, count_date);
```

The results window displays the following data:

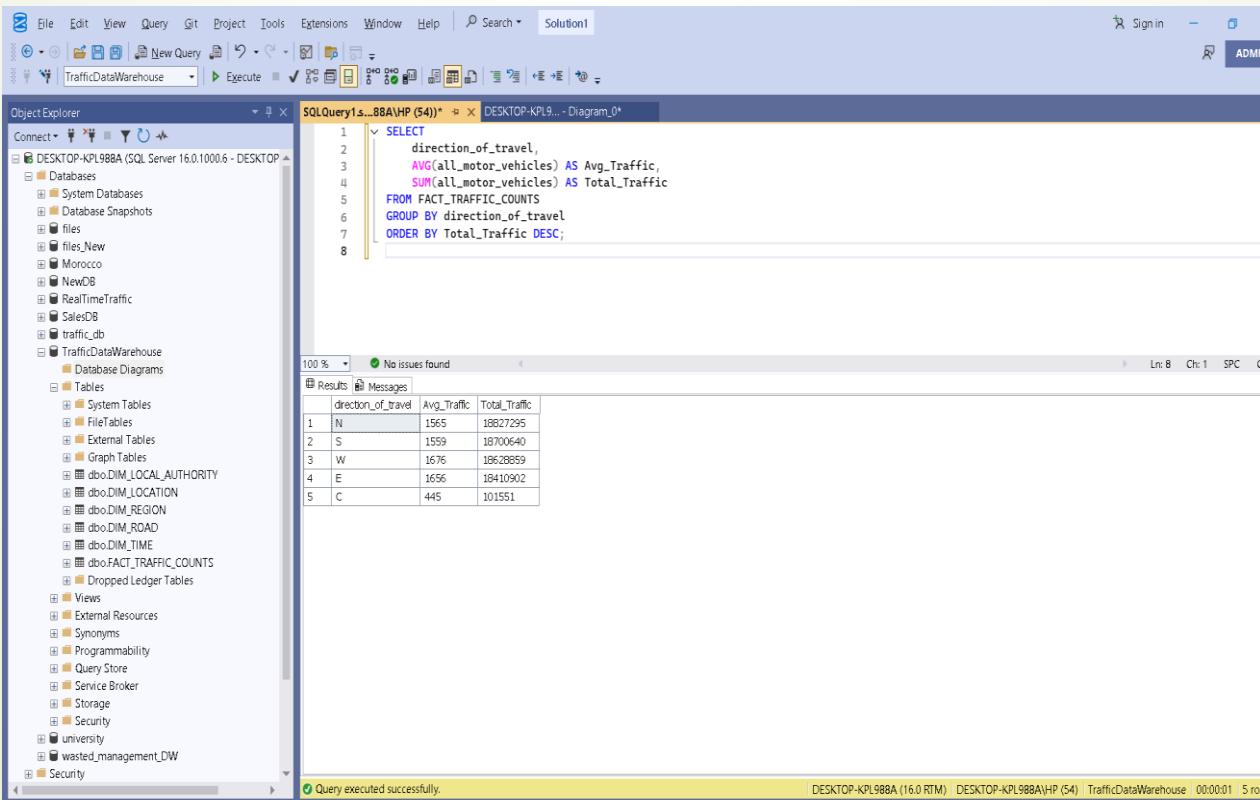
DayOfWeek	Avg_Vehicles	Total_Vehicles
Monday	1621	14667324
Tuesday	1631	14213574
Wednesday	1506	13722129
Thursday	1699	15579276
Friday	1579	16486944

## 5. تحليل الاتجاه (اتجاه السير)

SELECT

```
    direction_of_travel,  
    AVG(all_motor_vehicles) AS Avg_Traffic,  
    SUM(all_motor_vehicles) AS Total_Traffic  
FROM FACT_TRAFFIC_COUNTS  
GROUP BY direction_of_travel  
ORDER BY Total_Traffic DESC;
```

مفيد لتحديد الاتجاهات ذات الكثافة المرورية الأعلى.

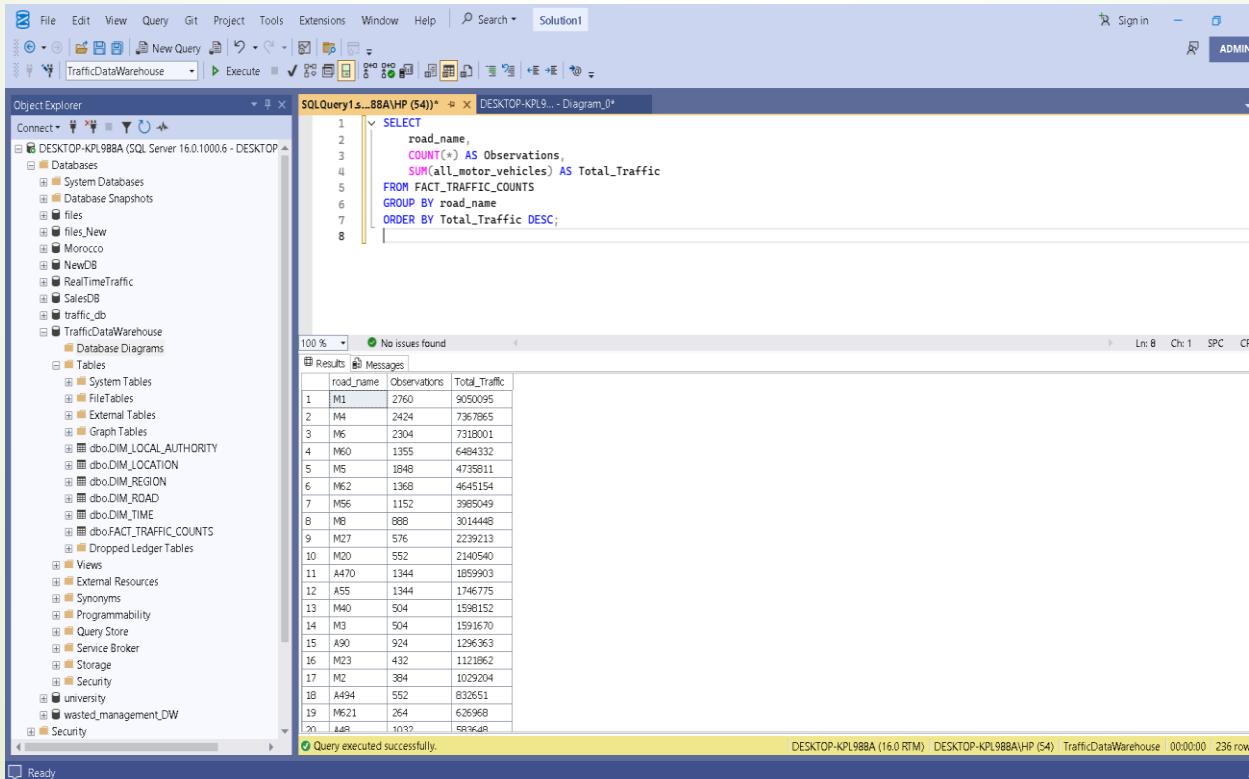


```
File Edit View Query Git Project Tools Extensions Window Help Search Solution1  
TrafficDataWarehouse Execute Sign in ADMIN  
Object Explorer SQLQuery1...8BA\HP (54)* DESKTOP-KPL98A - Diagram0*  
Connect Databases System Databases Database Snapshots files files_New Morocco NewDB RealTimeTraffic SalesDB traffic_db TrafficDataWarehouse Database Diagrams Tables System Tables FileTables External Tables Graph Tables dbo.DIM_LOCAL_AUTHORITY dbo.DIM_LOCATION dbo.DIM_REGION dbo.DIM_ROAD dbo.DIM_TIME dbo.FACT_TRAFFIC_COUNTS Dropped Ledger Tables Views External Resources Synonyms Programmability Query Store Service Broker Storage Security university wasted_management_DW Security  
SQLQuery1...8BA\HP (54)* DESKTOP-KPL98A - Diagram0*  
1 SELECT  
2     direction_of_travel,  
3     AVG(all_motor_vehicles) AS Avg_Traffic,  
4     SUM(all_motor_vehicles) AS Total_Traffic  
5 FROM FACT_TRAFFIC_COUNTS  
6 GROUP BY direction_of_travel  
7 ORDER BY Total_Traffic DESC;  
8  
Results Messages  
direction_of_travel Avg_Traffic Total_Traffic  
1 N 1565 18827295  
2 S 1559 18700640  
3 W 1676 18628859  
4 E 1656 18410902  
5 C 445 101551  
100 % No issues found Ln: 8 Ch: 1 SPC CRLF  
Query executed successfully. DESKTOP-KPL98A (16.0 RTM) DESKTOP-KPL98A\HP (54) TrafficDataWarehouse 00:00:01 5 rows
```

## 6. تحليل الطريق Road Name

```
SELECT
    road_name,
    COUNT(*) AS Observations,
    SUM(all_motor_vehicles) AS Total_Traffic
FROM FACT_TRAFFIC_COUNTS
GROUP BY road_name
ORDER BY Total_Traffic DESC;
```

□ هذا التحليل يظهر أكثر الطرق ازدحاماً بالمرور.

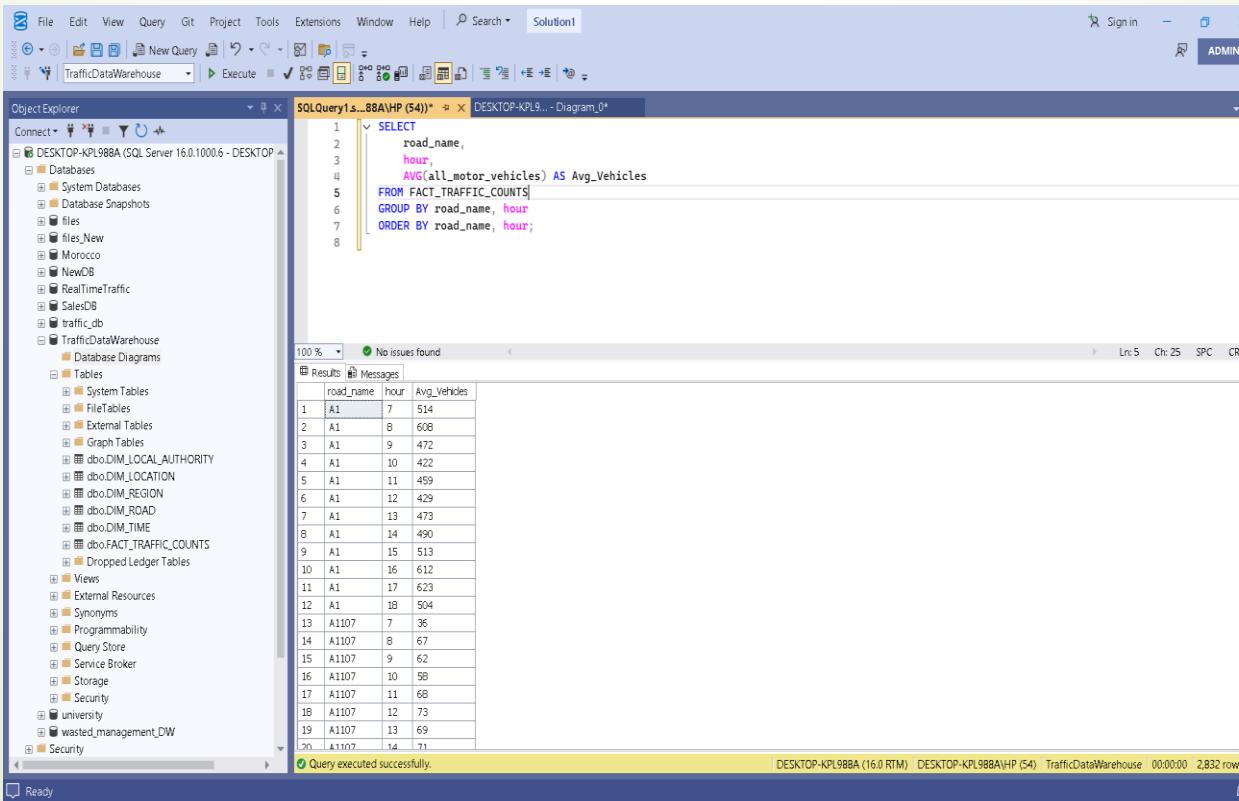


The screenshot shows the SQL Server Management Studio interface. The query window displays the T-SQL code for selecting road names, observation counts, and total traffic from the FACT\_TRAFFIC\_COUNTS fact table. The results window shows a table with columns: road\_name, Observations, and Total\_Traffic. The data is ordered by Total\_Traffic in descending order. A large orange arrow points from the left towards the results grid.

road_name	Observations	Total_Traffic
M1	2760	9050095
M4	2424	7367865
M6	2304	7318001
M60	1355	6484332
M5	1848	4739911
M2	1368	4645154
M66	1152	3965049
M8	888	3014448
M27	576	2239213
M20	552	2140540
A470	1344	1859903
A55	1344	1746775
M40	504	1598152
M3	504	1591670
A90	924	1296363
M23	432	1121862
M2	384	1029204
A494	552	832651
M621	264	626968
A48	1632	681648

## 7. دمج التحليل الزمني والمكاني مثلاً — حركة المرور لكل طريق حسب الساعة:

```
SELECT
    road_name,
    hour,
    AVG(all_motor_vehicles) AS Avg_Vehicles
FROM FACT_TRAFFIC_COUNTS
GROUP BY road_name, hour
ORDER BY road_name, hour;
```



The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. The top menu bar includes File, Edit, View, Query, Git, Project, Tools, Extensions, Window, Help, and a Search dropdown. Below the menu is a toolbar with various icons. The main window has tabs for Object Explorer, SQLQuery1, and DESKTOP-KPL98A\HP (54)\*. The Object Explorer sidebar shows the database structure, including the TrafficDataWarehouse database with its tables (e.g., FACT\_TRAFFIC\_COUNTS, DIM\_ROAD, DIM\_TIME). The central pane displays the T-SQL query provided in the question. The results pane shows a table with 2,832 rows, each containing road\_name, hour, and Avg\_Vehicles values. A status bar at the bottom indicates the query was executed successfully.

road_name	hour	Avg_Vehicles
A1	7	514
A1	8	608
A1	9	472
A1	10	422
A1	11	459
A1	12	429
A1	13	473
A1	14	490
A1	15	513
A1	16	612
A1	17	623
A1	18	504
A1107	7	36
A1107	8	67
A1107	9	62
A1107	10	58
A1107	11	68
A1107	12	73
A1107	13	69
A1107	14	71

Query executed successfully.

```

SELECT
    year,
    SUM(all_motor_vehicles) AS Total_Traffic,
    LAG(SUM(all_motor_vehicles)) OVER (ORDER BY
year) AS Prev_Year_Traffic,
    ( (SUM(all_motor_vehicles) -
LAG(SUM(all_motor_vehicles)) OVER (ORDER BY year)) * 100.0
    / LAG(SUM(all_motor_vehicles)) OVER (ORDER BY
year) ) AS Growth_Rate_Percent
FROM FACT_TRAFFIC_COUNTS
GROUP BY year
ORDER BY year;

```

**بيانات** يوضح معدل النمو أو التغير السنوي في عدد المركبات.

The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. In the top-left corner, the Object Explorer shows the database structure, including the 'TrafficDataWarehouse' database. In the center, a query window displays the following T-SQL code:

```

1  SELECT
2      year,
3      SUM(all_motor_vehicles) AS Total_Traffic,
4      LAG(SUM(all_motor_vehicles)) OVER (ORDER BY year) AS Prev_Year_Traffic,
5      ( (SUM(all_motor_vehicles) -
6          LAG(SUM(all_motor_vehicles)) OVER (ORDER BY year)) * 100.0
7          / LAG(SUM(all_motor_vehicles)) OVER (ORDER BY year) ) AS Growth_Rate_Percent
8  FROM FACT_TRAFFIC_COUNTS
9  GROUP BY year
10 ORDER BY year;

```

Below the code, the results pane shows a table with four columns: 'year', 'Total\_Traffic', 'Prev\_Year\_Traffic', and 'Growth\_Rate\_Percent'. The table contains 21 rows of data, starting from 2001 and ending at 2020. The 'Growth\_Rate\_Percent' column shows the percentage change in traffic volume from the previous year.

year	Total_Traffic	Prev_Year_Traffic	Growth_Rate_Percent
2001	3982807	4118974	-3.305847524164
2002	3968550	3962807	-10.903290065228
2003	3417423	3548550	-3.69522628196
2004	3616571	3417423	5.62741996769
2005	3202366	3616571	-11.452975760741
2006	4088717	3202366	27.679004325951
2007	3873546	4088717	-5.285300886960
2008	3786454	3873546	-2.452386649204
2009	3817802	3786454	3.438463833024
2010	2994595	3817802	-17.70336874529
2011	3171659	2994595	5.912793633142
2012	3071319	3171659	-3.16344610910
2013	3041196	3071319	-0.912618782028
2014	3116489	3041196	2.410441071115
2015	2779241	3116489	-26.86510547084
2016	2981765	2779241	30.82716980568
2017	2552890	2981765	-14.369870243255
2018	1052090	2552890	-58.794715363162
2019	2860697	1052090	175.905160209925
2020	1634351	2860697	-46.1911394910

At the bottom of the SSMS interface, a status bar indicates "Query executed successfully." and "DESKTOP-KPL988A (16.0 RTM) DESKTOP-kpl988a\HP (S4) TrafficDataWarehouse 00:00:02 25 rows".



# The beneficiaries of the "Real Time Traffic Project" include:

1. **Commuters and Drivers** – who can access up-to-date traffic information, avoid congestion, and reduce travel time.
2. **Public Transportation Authorities** – who can optimize routes, schedules, and fleet management using real-time data.
3. **Emergency Services** – such as police, ambulance, and fire departments, enabling faster response times and route planning.
4. **Urban Planners and Decision Makers** – who can use traffic data to improve infrastructure planning and traffic flow management.
5. **Logistics and Delivery Companies** – who can enhance operational efficiency and reduce fuel consumption through route optimization.
6. **Environmental Agencies** – benefiting from reduced vehicle emissions due to improved traffic flow.
7. **Citizens and Local Communities** – who experience safer roads, less congestion, and better air quality.



Thank  
you! :)

