B9DA111 Data Storage Solutions for Data Analytics: CA_ONE

Technical Report

Submitted by:
Anish Rao: 20066423

Lecturer: Bernie Lydon

Assignment Cover Sheet

Student Name(s) and Number(s) as per student card(s):
Anish Rao – 20066423

Programme: Master of Science in Data Analytics (BMS09DNL)

Lecturer Name: Bernie Lydon

Module/Subject Title: B9DA111 – Data Storage Solutions for Data Analytics

Assignment Title: Business Intelligence and ETL Solutions for NYC Taxi Operations.

No. of Words: 4530

By submitting this assignment, we are confirming that:

- **This assignment is all our own work;**
- **Any sources used have been referenced;**
- **We have followed the Generative AI instructions/ scale set out in the Assignment Brief;**
- **We have read the College rules regarding academic integrity in the QAH Part B Section 3, and the Generative AI Guidelines, and understand that penalties will be applied accordingly if work is found not to be our own.**
- **We understand that all work uploaded is submitted via Ouriginal, whereby a text-matching report will show any similarities with other texts.**

Table of Contents

1. Introduction

This technical report explains the complete thinking and process for creating business intelligence solution for New York City taxi operations. The complete architecture starts with defining the business vision and then followed by setting up a Datawarehouse in SQL Server Management Studio (SSMS). We then use raw data and perform ETL on it using SQL Server Integration Service (SSIS) to fill up the Datawarehouse. Finally we use SQL Server Reporting Services (SSRS) and Tableau to generate visualisations and reports. We also then use the Chinook sample database (Rocha, 2024) in Neo4j to compare differences between graph and relational databases.

The data source for this assignment is the NYC Taxi and Limousine Commission's Yellow Taxi Trip Records, specifically for January 2024 data which has more than 2.8 million trip records (NYC Taxi & Limousine Commission, 2024). The dataset provides a complex, real-world business situation which is great for showing data warehousing and analytical skills.

1.1 Reasons for Selecting Subject Area and Data

The NYC Taxi industry was chosen because it is a real-world dataset and the amount of data it provided would give us great analytical potential. There were several other factors also that supported our choice:

- Data Complexity: Multidimensional data like temporal, geographic, financial and other operational data for advanced analytics.
- Many different stakeholders: The taxi industry provides different values to many different stakeholders who need specific analytical information.
- Real-world impact: Provides actual insights that can be used to implement changes in real-world.
- Data quantity: The dataset has more than 2.8 million rows which is really good to show ETL and reporting skills.

1.2 Vision and Goals

Business Vision: To help improve NYC taxi operations using data driven analysis of 2.8+ million trips. This will help in optimizing fleet utilization and improve earnings of the drivers. This will also help in developing a strategy to allocate resources properly to meet demands in all of NYC's 266 service zones. This assignment will help to find best ways to operate by using evidence based decision and specific insights for different stakeholders.

Objectives:

- Operational Performance: Optimise fleet usage and driver productivity.
- Financial Performance: Analyse revenue flow and see profitability across different locations and time.
- Customer Experience: Analyse customer usage patterns to provide better service.

- Compliance Regulation: Regular reporting to make sure all everything is in compliance

Expected Business Transformation Outcomes:

- Increase in revenue by optimizing driver schedules and route efficiency
- Reduce operational cost by vendor management
- Improved customer satisfaction
- Compliance issue Risk mitigation

1.3 Key Stakeholders

The complete solution provides answers to stakeholders at different levels like operational, strategic, executive levels.

Operational Level Stakeholders:

- Operations Manager- performance of vendors and resource optimization
- Shift Supervisor- hourly/daily performance monitoring for scheduling optimization
- Customer Service Manager- customer satisfaction (number of tips), other service metrics
- Driver Relations Manager- driver earnings and driver deployment optimization

Strategic Level Stakeholders:

- Business Analyst- operational insights, geographic efficiency
- Technology Manager- Improving payment infrastructure
- Revenue Manager- Improving profits and financial assessment

Executive Level Stakeholders:

- **C-Level Executives-** Strategic trend analysis, daily performance KPI's
- **General Manager-** Full operational information for strategic planning

1.4 Business Requirements

Functional Requirements:

- Full data integration: We would need to combine data from multiple sources for complete operational visibility.
- Multi-dimensional analysis: The architecture should support analysing over time, geographic locations and finances.
- Reporting: Provide detailed operation summary reports and executive dashboards.
- Interactive analysis: Provide dynamic dashboards for enhanced data exploration.

Expected Business Insights:

- Identifying which locations and time generate most revenue and decide how to maximise profit.

- Improve how resources are used and optimize shifts to reduce operational costs and proper vendor management.
- Improving service levels for customer satisfaction.
- Identify and mitigate any revenue risks, any operational risks and violating compliance.

2. Schema

2.1 Data Warehouse Schema Design

The Datawarehouse implementation follows a star schema architecture for providing good querying performances. The schema design is centred around a single fact table which is then surrounded by multiple dimension tables all linked via foreign keys (Fig. 1).

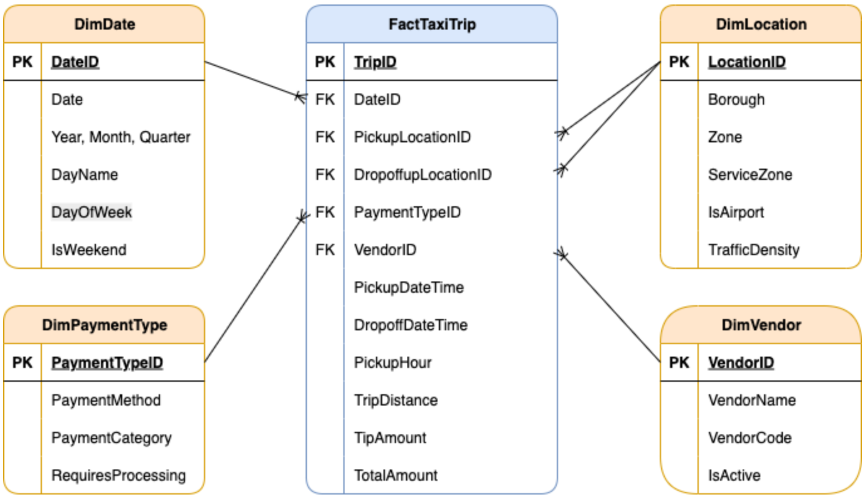


Figure 1: NYC Taxi Star Schema

Core Schema Components:

FactTaxiTrip- The central fact table has all the 2.8+ million records of individual taxi trips connected to other tables using foreign keys. The fact table stores both additive data like TotalAmount, TipAmount, TripDistance and non-additive data like PickupDateTime, DropoffDateTime for time analysis.

DimDate (33 rows)- Contains all date hierarchies for dates from 1 – 31 January, and 31st December and 1st Feb for handling edge cases

DimVendor (5 rows)- The Taxi vendor information including official codes

DimLocation (266 rows)- NYC zones with borough classifications

DimPaymentType (8 rows)- Payment method classifications as mentioned on NYCTLC website.

Key Design Features:

Role-Playing Dimensions: The DimLocation table is used for both pickup and drop-off location by separate foreign key relationships. This helps reducing storage, and we don't have much data duplication while also maintaining analytical flexibility.

Computed Columns: "PickupHour" is calculated and persisted in the fact table. This helps in removing any extra calculations we might need while creating reports. Also improves query performance for time-based analysis. "IsWeekend" and "IsAirport" also help in making it easy while providing information for weekend trips or trips to and from airports.

2.2 Reasons for Design

Dimensional Modeling Methodology

The star schema follows Ralph Kimball's dimensional modeling approach. It was specifically chosen because of how easy it is to use and also provides very good query performance. This methodology focuses on the Datawarehouse being understandable and fast but does impact the storage size.

Performance Optimization

One of the benefits of star schema is that it is denormalized which means less JOIN operations and faster query execution. This is important for fast reporting and dashboard performances. Since there are more than 2.8 million rows, we used indexing to make querying faster. We also used proper data types like DECIMAL for financial calculations and DATETIME2 for proper timestamping.

Role-Playing Dimensions

Using a single DimLocation table for both pickup and dropoff locations is done by almost all of the real-world Taxi schemas. This helps to reduce storage and data/table duplication. This also makes the ETL process more simple without sacrificing analytical capability.

Business Alignment

The star schema also aligns with the business requirements as it helps different stakeholders to understand the data without much technical database knowledge. The structured design also makes it easy to combine different business attributes to provide better reporting and analysis needed for different stakeholder groups. The structured also makes it easy to add more data in the future without needed many changes also reducing risks of any data loss.

3 ETL Process

3.1 ETL Procedure Explanation

Database Infrastructure Setup

For setting up the Datawarehouse we had to first define the schema and tables in SQL Server Management Studio (SSMS). We then created the "NYC_Taxi_DW" database and creating a schema "taxi" to save the tables.

The next step in the process was to define the table structures for all four dimension tables and the central fact table. This also included defining primary keys, foreign keys, data types for all columns, to make sure there is no issues in data loading. Since there would also be so many rows in the tables it was also necessary to do some indexing to improve query performance. All table creation scripts can be found in Appendix C.

ETL Architecture Overview

The Extract, Transform, and Load (ETL) process was done using SQL Server Integration Services (SSIS) to efficiently load the NYC Taxi dataset into the Datawarehouse. The ETL steps we used combines doing some basic Execute SQL tasks with Data Flow tasks for more complex transformations.

The SSIS package follows a proper sequence to keep data integrity and optimize efficiency:

- A temporary SQL task was added in the beginning for clearing existing table data to prevent any errors in inserting data as the whole process would be run again and again during development.
- Next we load up the dimension tables in the correct dependency order. First, inserting values in the Vendor and Payment dimension tables, followed by creating the date dimension table.
- This was followed by data flow tasks for loading raw data and doing required transformations and then writing to database tables for the location dimension and main fact table.
- Last step was to validate the data and ensure no errors in data loading by running some scripts to check all values, the row counts etc. are as expected.

Data Flow Process Description

Static Dimensions (vendor and payment type) just used simple insert statements using "Execute SQL tasks". The data for these tables was defined on the NYC taxi website. The date dimension table was generated programmatically to cover all dates from 1st to 31st January and 31st December (trips would start late at night and end on 1st Jan) and 1st February (trips would start late at 31st Jan and end on 1st Feb) as well.

The data flow task for loading location dimension was also pretty straightforward with only requiring easy derived columns generation for "IsAirport" and "TrafficDensity" columns.

The main fact table data flow was a bit complex with multiple derived column generations for handling of null values, conditional splits to only include any dates between 31st December and 1st February to make sure there are no extra data that might have been in the raw data for other timelines and finally data type conversion before mapping all the new columns to the final database table. All of this also helped in making sure data quality had no issues. We successfully setup null checks, data conversion for proper data storage and conditioning for only the data we actually need.

3.2 SSIS Workspace Screenshots

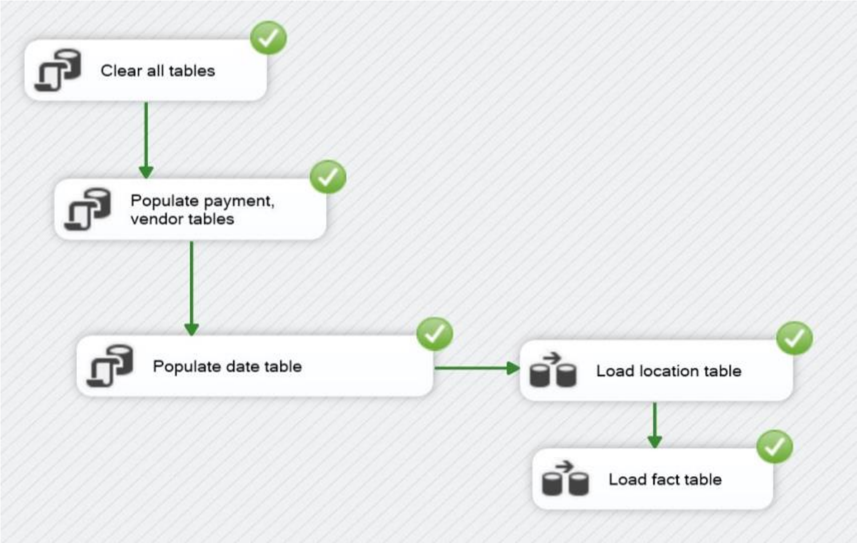
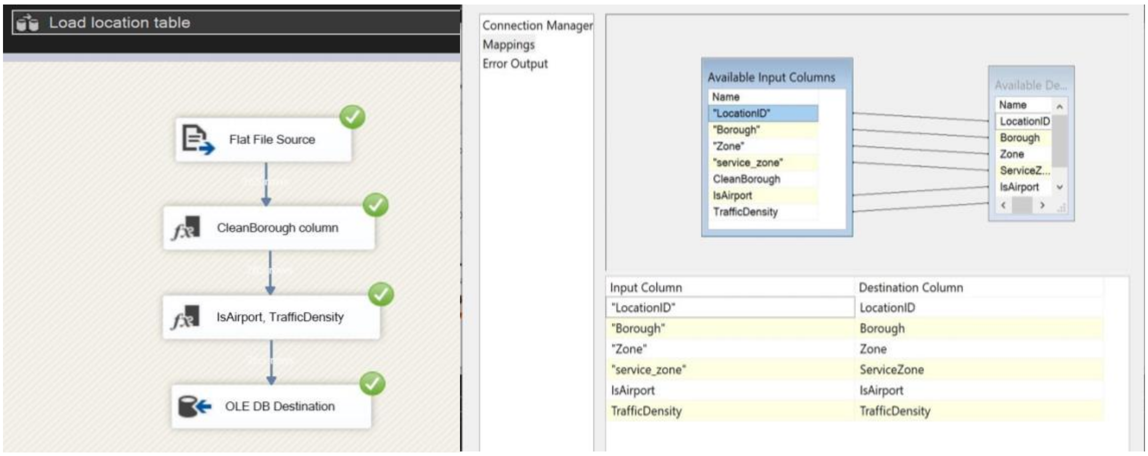
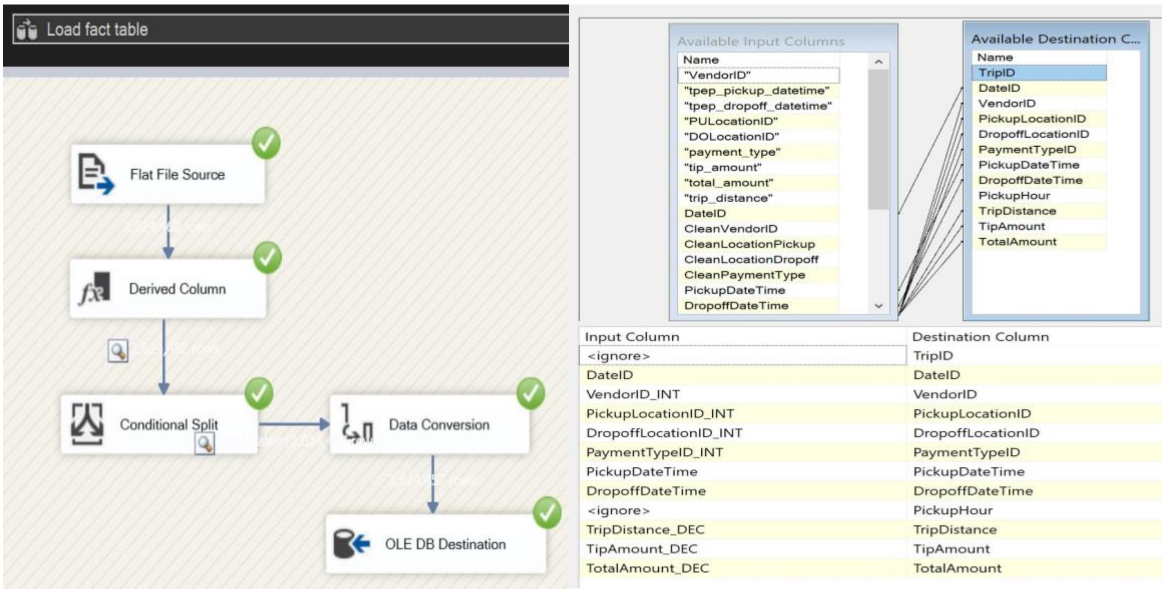


Figure 2: Complete ETL package control flow showing sequential execution of dimension loading followed by fact table processing



Input Column	Destination Column
"LocationID"	LocationID
"Borough"	Borough
"Zone"	Zone
"service_zone"	ServiceZone
IsAirport	IsAirport
TrafficDensity	TrafficDensity

Figure 3: Detailed data flow transformation for location dimension, including source file reading, derived column transformations, and destination mapping/ loading.



Available Input Columns

Name
"VendorID"
"tpep_pickup_datetime"
"tpep_dropoff_datetime"
"PULocationID"
"DOLocationID"
"payment_type"
"tip_amount"
"total_amount"
"trip_distance"
DateID
CleanVendorID
CleanLocationPickup
CleanLocationDropoff
CleanPaymentType
PickupDateTime
DropoffDateTime

Available Destination C...

Name
TripID
DateID
VendorID
PickupLocationID
DropoffLocationID
PaymentTypeID
PickupDateTime
DropoffDateTime
PickupHour
TripDistance
TipAmount
TotalAmount

Input Column	Destination Column
<ignore>	TripID
DateID	DateID
VendorID_INT	VendorID
PickupLocationID_INT	PickupLocationID
DropoffLocationID_INT	DropoffLocationID
PaymentTypeID_INT	PaymentTypeID
PickupDateTime	PickupDateTime
DropoffDateTime	DropoffDateTime
<ignore>	PickupHour
TripDistance_DEC	TripDistance
TipAmount_DEC	TipAmount
TotalAmount_DEC	TotalAmount

Figure 4: Detailed data flow transformation for main fact table, including source file reading, derived column conditional splits, data conversion, and destination mapping/ loading

3.3 ETL Challenges and Solutions

During initial implementation, a column mapping issue was noticed where TotalAmount and TipAmount were mistakenly swapped in the OLE DB Destination task. This resulted in very strange average fare calculations ($3.42 instead of expected $15-30). To find this issue out we created SQL scripts that we ran in SSMS and compared them with what the raw data should be and then saw the mapping issue. To prevent any issues like this further we added in more data validation scripts which we ran in the end.

We also noticed that often the OLE DB Destination task would end up failing which was due to improper null value handling which we then used derived columns to handle in the data flow task.
Example Null value handling transformation

CleanVendorID => ["VendorID"] == "\\N" ? "99" : ["VendorID"]

We also did Batch processing with 10,000-row batches to balance memory usage and processing speed to handle any computational issues with our systems. Fast Load options was also enabled for bulk data insertion operations.

This whole ETL process was able to successfully process 100% of the raw source data and also making sure data quality standards are maintained and all data loads as expected. See Appendix D for complete ETL implementation details including SSIS package configurations and transformation details.

4. Visualizations and Reports

4.1 Tableau Visualizations

4.1.1 Business Requirements

Trip Volume Analysis

Fleet managers need to understand properly the demand trends in a full day or week. This helps them identify peak times, quite times and any weekly variations to help decide how many drivers should be out and when.

Payment Method Preferences for Infrastructure Planning

The technology managers needs to understand which types of payment method is the most famous and in which area to help them decide how and what to invest in to make sure customers are able to pay with their preferred method

Trip Duration Analysis for Service Level Management

The customer service teams need to understand the duration of a trip which would help them improve operational efficiency and set achievable expectations.

Tip Analysis for Driver Optimization

Driver relations managers can find out customer from which areas tip better so they can allocate drivers In particular locations accordingly to maximise their earnings.

4.1.2 Visualizations

Visualization 1: Trip Volume Patterns Analysis

This is a dual axis chart (Fig. 5). On top is the multi-line chart that shows number of trip by hour of day on each day of the week. The bottom chart is a filled line chart that just shows number of trips at each hour of the day (irrespective of day).

Key Insights: We can see clear patterns that most of the trips are during weekday evenings. But on the weekends it is much different. This data can be used to strategically deploy vehicles and adjust shift schedules accordingly to match the demand during the week.

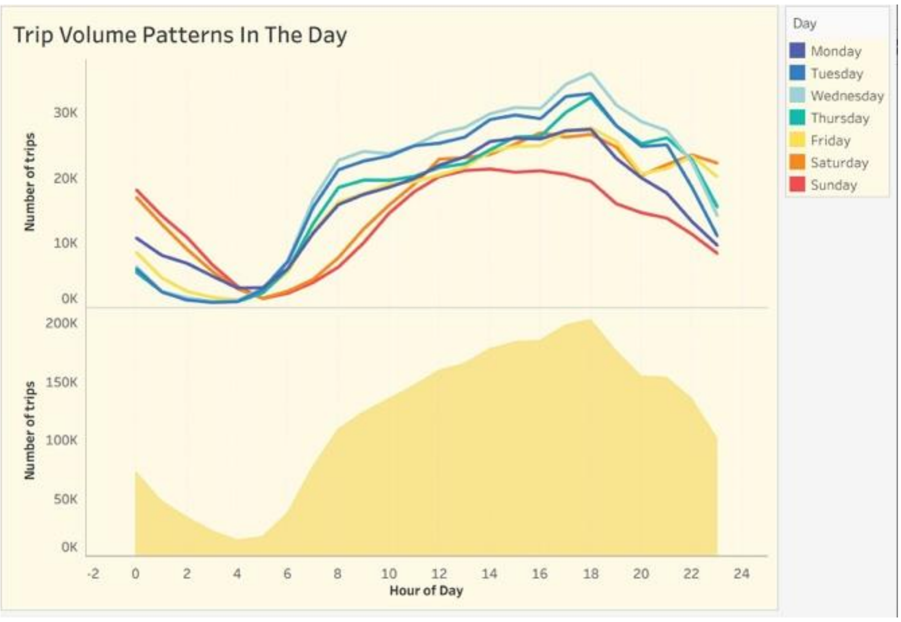


Figure 5: Trip volume patterns by Hour and day

Visualization 2: Payment Method Preferences by Borough

The stacked vertical bar chart shows the different payment methods distribution by the different boroughs of NYC (Fig. 6).

Key Insights: Staten Island shows much higher cash usage than Bronx, Manhattan, Brooklyn and EWR where customers prefer using card much more. This can be used to making targeted investments and how to deploy payment systems for revenue optimization.

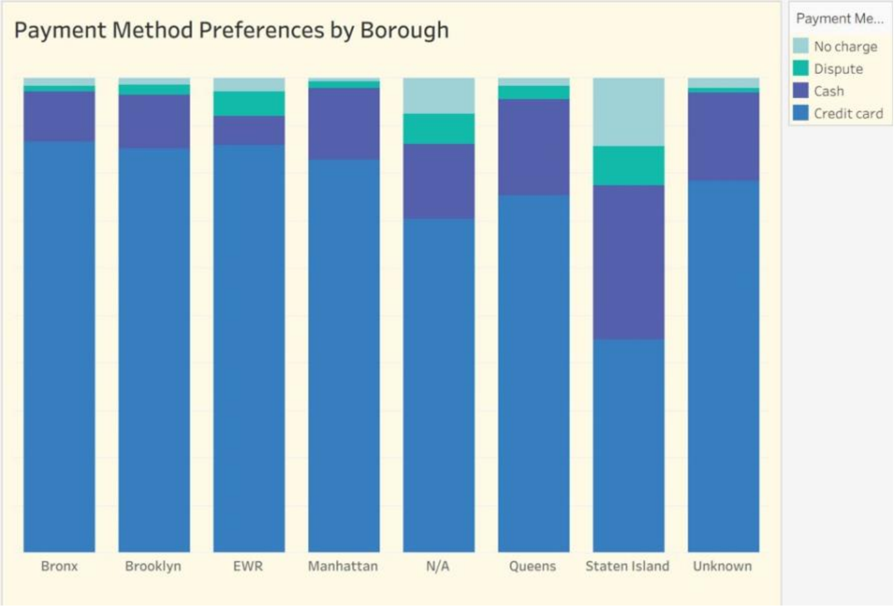


Figure 6: Payment Method Distribution by Borough

Visualization 3: Trip Duration Distribution Analysis

The donut chart provides information trip durations which are divided into four categories- Quick: 0-15min, Standard: 15-30min, Long: 30-60min and Extended: 60+min (Fig. 7).

Key Insights: The trip duration analysis clearly shows that 66% of trips are very quick and 25% are standard durations in NYC. The chart can be filtered by borough and times of the day and this will give specific duration metrics. Stakeholders can then plan accordingly to maximise their fleet productivity in each area by deciding the types of vehicles to deploy and route optimizations.

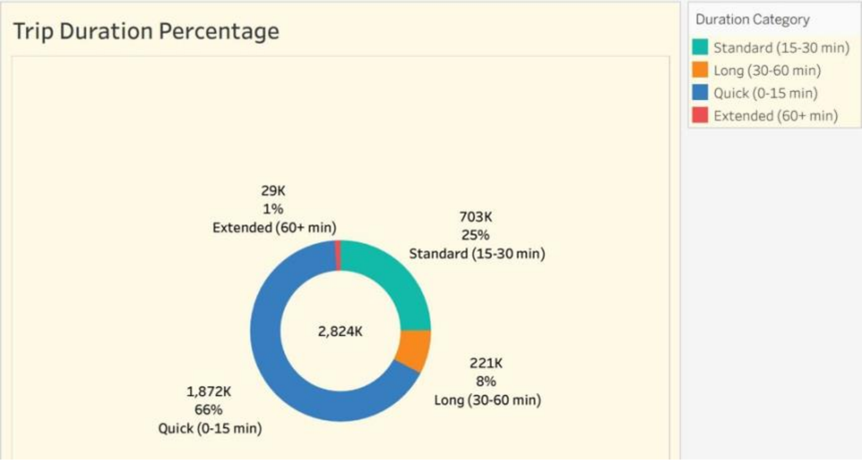


Figure 7: Trip Duration Category Distribution

Visualization 4: Average Tip Analysis by Borough

The Horizontal bar chart shows the average tip amounts by borough. Also shows the trip volumes by the colour of the bar - darker the colour, more the trips (Fig. 8).

Key Insights: We can clearly see a very big geographic variation. Customers going to EWR (Newark Airport) tip the most and in Bronx show minimal tipping. This helps in strategic driver deployment to improve driver retention by improving their earnings.

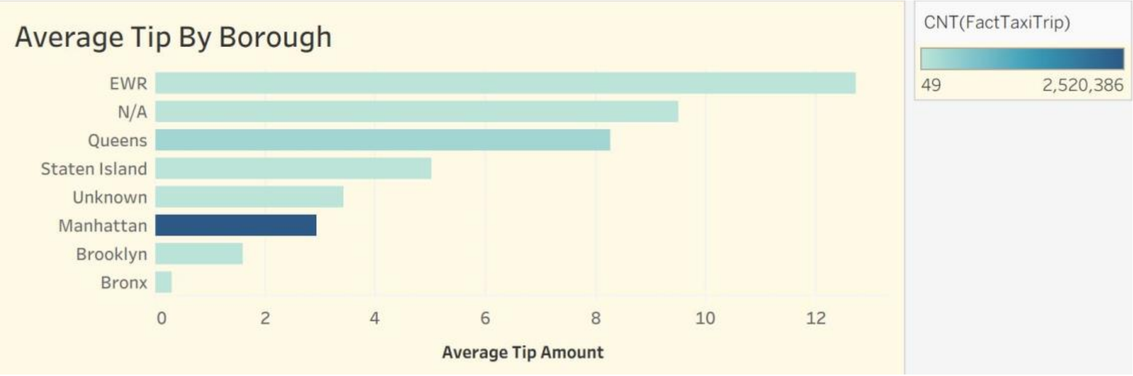


Figure 8: Average Tip Analysis by Geographic Area

Interactive Dashboard Implementation:

The dashboard combines all four visualizations with cross-filtering enabled. Hovering on a chart also highlights data which helps users to see any relationships between the information from other visualisations (Fig. 9).

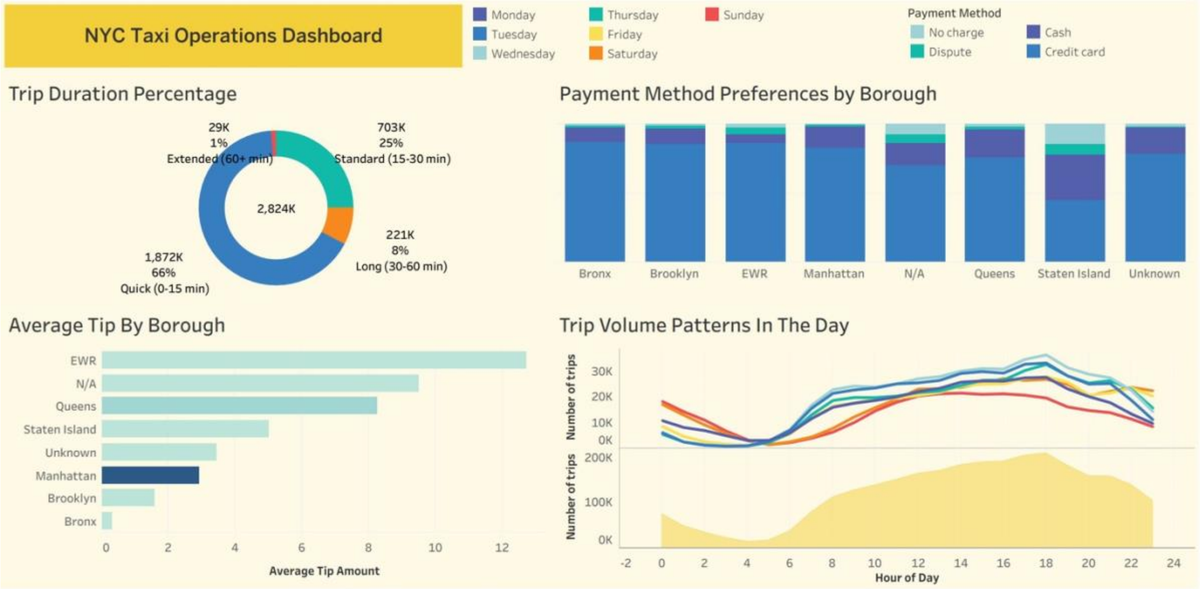


Figure 9: Tableau Dashboard

See Appendix A for complete Tableau calculated fields and interactive feature specifications.

4.2 SSRS Reports

4.2.1 Business Requirements

Vendor Performance Monitoring

Operations manager need to be able to compare vendor performances to decide which vendor contracts to renew or change and make sure good service quality at optimal costs.

Revenue Analysis and Pattern Identification

Revenue managers need to see how much revenue is generated at what time and day of the week so that they can maximise the profits and optimize driver schedules as well. This would also help in reducing operational costs by reducing vehicles at quite times.

Location Efficiency Analysis

Business analysts need geographic performance to find out trends and optimize routes and strategy planning to make sure they are ahead of the competition.

Executive Dashboard Requirements

Higher executives usually want to see a summary of the performances and KPI’s to be able to analyse data quickly and make informed decisions.

Route Performance Tracking

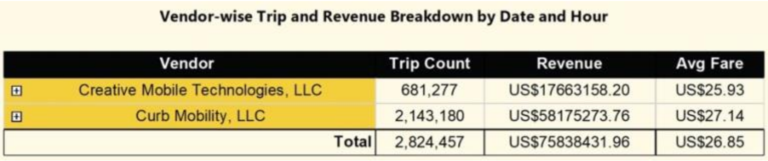
Financial analysts need detailed information about the top revenue generating routes to assess any risks and prioritize accordingly to ensure long-term business availability.

4.2.2 Reports

Report 1: Vendor Performance Drill-Down Analysis

Target Stakeholder: Operations Manager

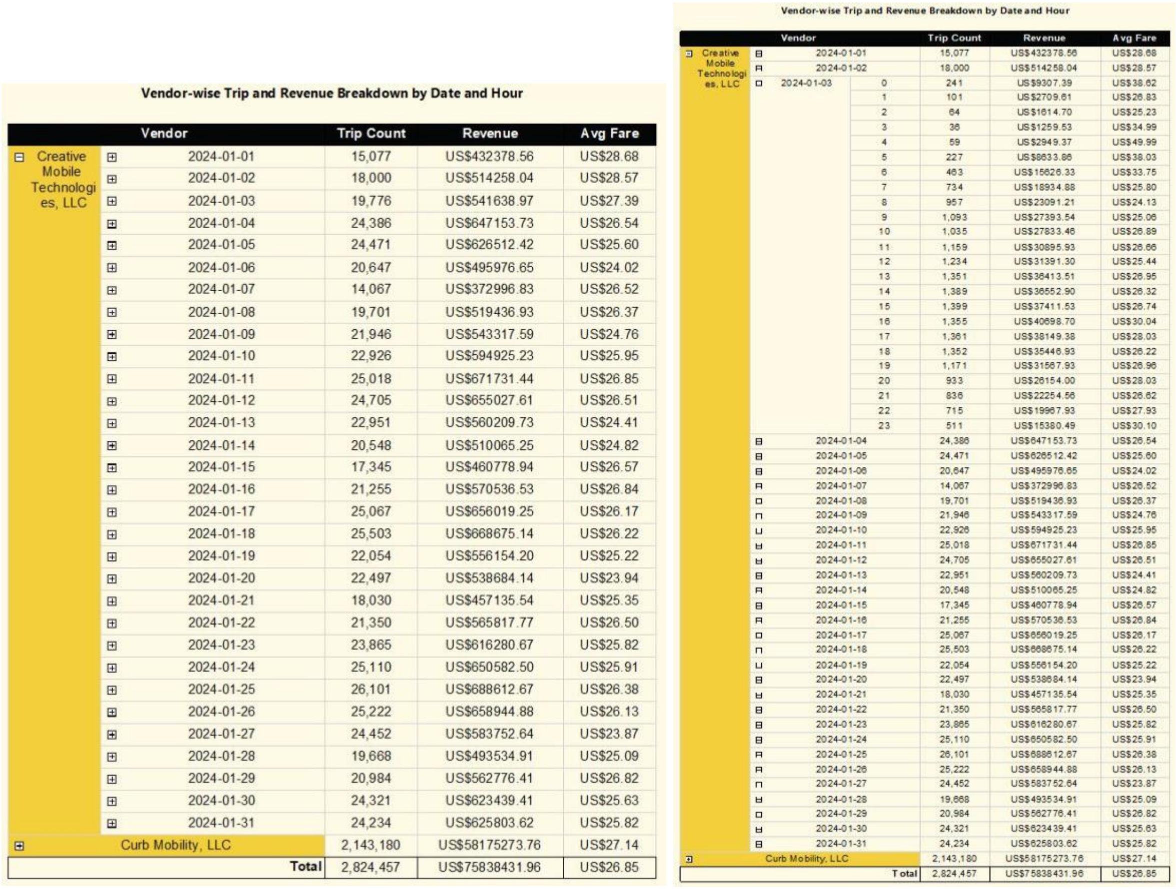
Business Value: This is a Three-level (Vendor -> Date -> Hour) drill-down report (Fig. 10). It has visibility toggles that lets users expand data to date level and hour level (Fig. 11). Stakeholders can compare vendor performance to support data driven contract negotiations, vendor management and service optimization as per the trips and revenue they generate.



Vendor-wise Trip and Revenue Breakdown by Date and Hour

Vendor	Trip Count	Revenue	Avg Fare
⊞ Creative Mobile Technologies, LLC	681,277	US$17663158.20	US$25.93
⊞ Curb Mobility, LLC	2,143,180	US$58175273.76	US$27.14
Total	2,824,457	US$75838431.96	US$26.85

Figure 10: Vendor Performance Drill-Down report



Vendor-wise Trip and Revenue Breakdown by Date and Hour

Vendor		Trip Count	Revenue	Avg Fare
⊟ Creative Mobile Technologies, LLC	⊞ 2024-01-01	15,077	US$432378.56	US$28.68
	⊞ 2024-01-02	18,000	US$514258.04	US$28.57
	⊞ 2024-01-03	19,776	US$541638.97	US$27.39
	⊞ 2024-01-04	24,386	US$647153.73	US$26.54
	⊞ 2024-01-05	24,471	US$626512.42	US$25.60
	⊞ 2024-01-06	20,647	US$495976.65	US$24.02
	⊞ 2024-01-07	14,067	US$372996.83	US$26.52
	⊞ 2024-01-08	19,701	US$519436.93	US$26.37
	⊞ 2024-01-09	21,946	US$543317.59	US$24.76
	⊞ 2024-01-10	22,926	US$594925.23	US$25.95
	⊞ 2024-01-11	25,018	US$671731.44	US$26.85
	⊞ 2024-01-12	24,705	US$655027.61	US$26.51
	⊞ 2024-01-13	22,951	US$560209.73	US$24.41
	⊞ 2024-01-14	20,548	US$510065.25	US$24.82
	⊞ 2024-01-15	17,345	US$460778.94	US$26.57
	⊞ 2024-01-16	21,255	US$570536.53	US$26.84
	⊞ 2024-01-17	25,067	US$656019.25	US$26.17
	⊞ 2024-01-18	25,503	US$668675.14	US$26.22
	⊞ 2024-01-19	22,054	US$556154.20	US$25.22
	⊞ 2024-01-20	22,497	US$538684.14	US$23.94
	⊞ 2024-01-21	18,030	US$457135.54	US$25.35
	⊞ 2024-01-22	21,350	US$565817.77	US$26.50
	⊞ 2024-01-23	23,865	US$616280.67	US$25.82
	⊞ 2024-01-24	25,110	US$650582.50	US$25.91
	⊞ 2024-01-25	26,101	US$688612.67	US$26.38
	⊞ 2024-01-26	25,222	US$658944.88	US$26.13
	⊞ 2024-01-27	24,452	US$583752.64	US$23.87
	⊞ 2024-01-28	19,668	US$493534.91	US$25.09
	⊞ 2024-01-29	20,984	US$562776.41	US$26.82
	⊞ 2024-01-30	24,321	US$623439.41	US$25.63
	⊞ 2024-01-31	24,234	US$625803.62	US$25.82
⊞ Curb Mobility, LLC		2,143,180	US$58175273.76	US$27.14
	Total	2,824,457	US$75838431.96	US$26.85

Vendor-wise Trip and Revenue Breakdown by Date and Hour

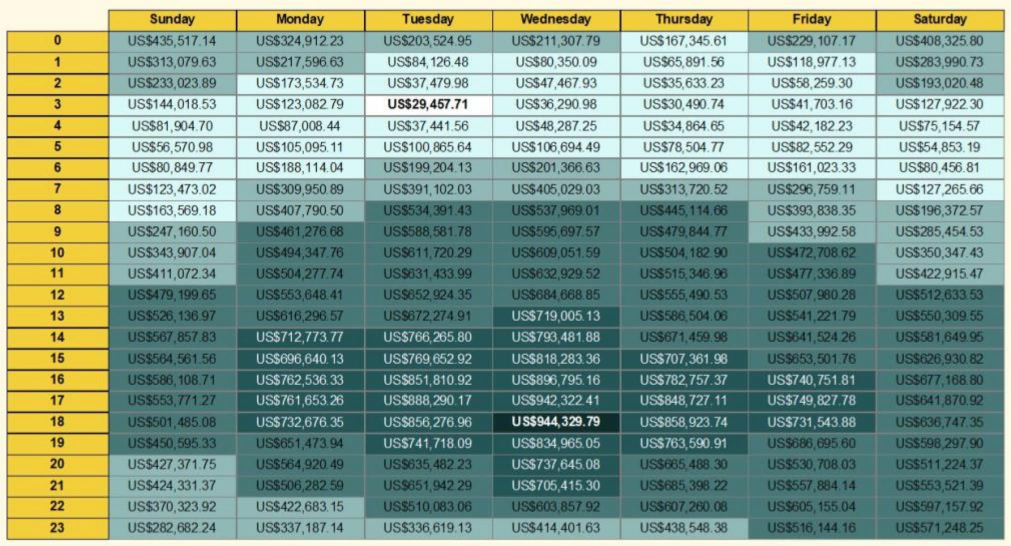
Vendor			Trip Count	Revenue	Avg Fare
⊟ Creative Mobile Technologies, LLC	⊞ 2024-01-01		15,077	US$432378.56	US$28.68
	⊞ 2024-01-02		18,000	US$514258.04	US$28.57
	⊟ 2024-01-03	0	241	US$9307.39	US$38.62
		1	101	US$2709.61	US$26.83
		2	64	US$1614.70	US$25.23
		3	36	US$1259.53	US$34.99
		4	59	US$2949.37	US$49.99
		5	227	US$8633.86	US$38.03
		6	463	US$15626.33	US$33.75
		7	734	US$18934.88	US$25.80
		8	957	US$23091.21	US$24.13
		9	1,093	US$27393.54	US$25.06
		10	1,035	US$27833.46	US$26.89
		11	1,159	US$30895.93	US$26.66
		12	1,234	US$31391.30	US$25.44
		13	1,351	US$36413.51	US$26.95
		14	1,389	US$36552.90	US$26.32
		15	1,399	US$37411.53	US$26.74
		16	1,355	US$40698.70	US$30.04
		17	1,361	US$38149.38	US$28.03
		18	1,352	US$35446.93	US$26.22
		19	1,171	US$31567.93	US$26.96
		20	933	US$26154.00	US$28.03
		21	836	US$22254.56	US$26.62
		22	715	US$19967.93	US$27.93
		23	511	US$15380.49	US$30.10
	⊞ 2024-01-04		24,386	US$647153.73	US$26.54
	⊞ 2024-01-05		24,471	US$626512.42	US$25.60
	⊞ 2024-01-06		20,647	US$495976.65	US$24.02
	⊞ 2024-01-07		14,067	US$372996.83	US$26.52
	⊞ 2024-01-08		19,701	US$519436.93	US$26.37
	⊞ 2024-01-09		21,946	US$543317.59	US$24.76
	⊞ 2024-01-10		22,926	US$594925.23	US$25.95
	⊞ 2024-01-11		25,018	US$671731.44	US$26.85
	⊞ 2024-01-12		24,705	US$655027.61	US$26.51
	⊞ 2024-01-13		22,951	US$560209.73	US$24.41
	⊞ 2024-01-14		20,548	US$510065.25	US$24.82
	⊞ 2024-01-15		17,345	US$460778.94	US$26.57
	⊞ 2024-01-16		21,255	US$570536.53	US$26.84
	⊞ 2024-01-17		25,067	US$656019.25	US$26.17
	⊞ 2024-01-18		25,503	US$668675.14	US$26.22
	⊞ 2024-01-19		22,054	US$556154.20	US$25.22
	⊞ 2024-01-20		22,497	US$538684.14	US$23.94
	⊞ 2024-01-21		18,030	US$457135.54	US$25.35
	⊞ 2024-01-22		21,350	US$565817.77	US$26.50
	⊞ 2024-01-23		23,865	US$616280.67	US$25.82
	⊞ 2024-01-24		25,110	US$650582.50	US$25.91
	⊞ 2024-01-25		26,101	US$688612.67	US$26.38
	⊞ 2024-01-26		25,222	US$658944.88	US$26.13
	⊞ 2024-01-27		24,452	US$583752.64	US$23.87
	⊞ 2024-01-28		19,668	US$493534.91	US$25.09
	⊞ 2024-01-29		20,984	US$562776.41	US$26.82
	⊞ 2024-01-30		24,321	US$623439.41	US$25.63
	⊞ 2024-01-31		24,234	US$625803.62	US$25.82
⊞ Curb Mobility, LLC			2,143,180	US$58175273.76	US$27.14
		Total	2,824,457	US$75838431.96	US$26.85

Figure 11: Drill-Down Report Level 2 & 3

Report 2: Revenue Matrix Analysis

Target Stakeholder: Shift Supervisor

Business Value: This matrix report in Fig. 12 (Day × Hour) helps highlighting peak revenue times that can help managers/supervisors allocate drivers accordingly. The darker cells indicate higher revenue totals.



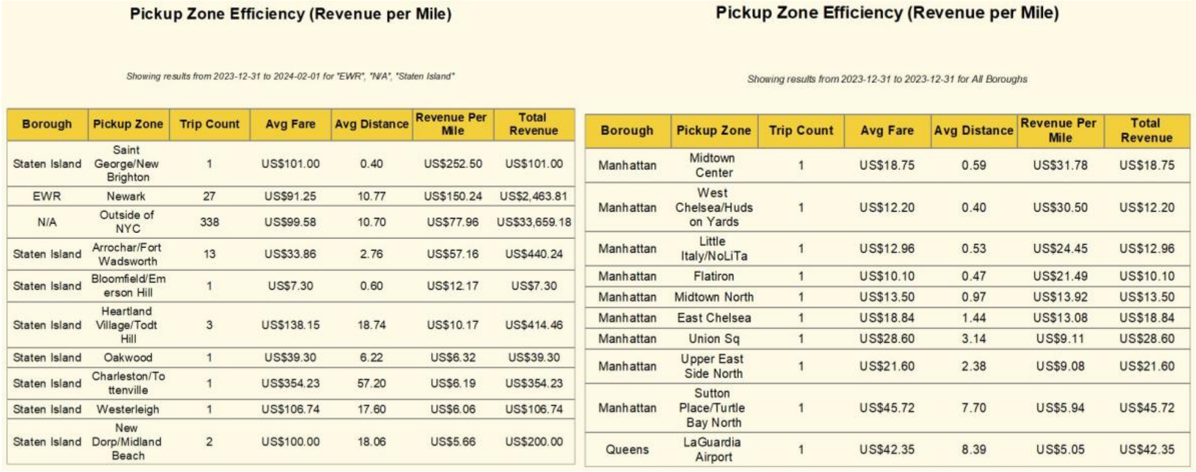
	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
0	US$435,517.14	US$324,912.23	US$203,524.95	US$211,307.79	US$167,345.61	US$229,107.17	US$408,325.80
1	US$313,079.63	US$217,596.63	US$84,126.48	US$80,350.09	US$65,891.56	US$118,977.13	US$283,990.73
2	US$233,023.89	US$173,534.73	US$37,479.98	US$47,467.93	US$35,633.23	US$58,259.30	US$193,020.48
3	US$144,018.53	US$123,082.79	**US$29,457.71**	US$36,290.98	US$30,490.74	US$41,703.16	US$127,922.30
4	US$81,904.70	US$87,008.44	US$37,441.56	US$48,287.25	US$34,864.65	US$42,182.23	US$75,154.57
5	US$56,570.98	US$105,095.11	US$100,865.64	US$106,694.49	US$78,504.77	US$82,552.29	US$54,853.19
6	US$80,849.77	US$188,114.04	US$199,204.13	US$201,366.63	US$162,969.06	US$161,023.33	US$80,456.81
7	US$123,473.02	US$309,950.89	US$391,102.03	US$405,029.03	US$313,720.52	US$296,759.11	US$127,265.66
8	US$163,569.18	US$407,790.50	US$534,391.43	US$537,969.01	US$445,114.66	US$393,838.35	US$196,372.57
9	US$247,160.50	US$461,276.68	US$588,581.78	US$595,697.57	US$479,844.77	US$433,992.58	US$285,454.53
10	US$343,907.04	US$494,347.76	US$611,720.29	US$609,051.59	US$504,182.90	US$472,708.62	US$350,347.43
11	US$411,072.34	US$504,277.74	US$631,433.99	US$632,929.52	US$515,346.96	US$477,336.89	US$422,915.47
12	US$479,199.65	US$553,648.41	US$652,924.35	US$684,668.85	US$555,490.53	US$507,980.28	US$512,633.53
13	US$526,136.97	US$616,296.57	US$672,274.91	US$719,005.13	US$586,504.06	US$541,221.79	US$550,309.55
14	US$567,857.83	US$712,773.77	US$766,265.80	US$793,481.88	US$671,459.98	US$641,524.26	US$581,649.95
15	US$564,561.56	US$696,640.13	US$769,652.92	US$818,283.36	US$707,361.98	US$653,501.76	US$626,930.82
16	US$586,108.71	US$762,536.33	US$851,810.92	US$896,795.16	US$782,757.37	US$740,751.81	US$677,168.80
17	US$553,771.27	US$761,653.26	US$888,290.17	US$942,322.41	US$848,727.11	US$749,827.78	US$641,870.92
18	US$501,485.08	US$732,676.35	US$856,276.96	**US$944,329.79**	US$858,923.74	US$731,543.88	US$636,747.35
19	US$450,595.33	US$651,473.94	US$741,718.09	US$834,965.05	US$763,590.91	US$686,695.60	US$598,297.90
20	US$427,371.75	US$564,920.49	US$635,482.23	US$737,645.08	US$665,488.30	US$530,708.03	US$511,224.37
21	US$424,331.37	US$506,282.59	US$651,942.29	US$705,415.30	US$685,398.22	US$557,884.14	US$553,521.39
22	US$370,323.92	US$422,683.15	US$510,083.06	US$603,857.92	US$607,260.08	US$605,155.04	US$597,157.92
23	US$282,682.24	US$337,187.14	US$336,619.13	US$414,401.63	US$438,548.38	US$516,144.16	US$571,248.25

Figure 12: Hourly Revenue Distribution Matrix

Report 3: Location Efficiency Analysis (Parameterized)

Target Stakeholder: Strategic Planning Manager

Business Value: The tabular reports (Fig. 13) have multiple parameters like borough, start date and end date. This helps provide only specific information from a particular time and location. It also has dynamic titles which makes it more easier for user to understand which dates and locations the data is being shown for.



Pickup Zone Efficiency (Revenue per Mile)

Showing results from 2023-12-31 to 2024-02-01 for "EWR", "N/A", "Staten Island"

Borough	Pickup Zone	Trip Count	Avg Fare	Avg Distance	Revenue Per Mile	Total Revenue
Staten Island	Saint George/New Brighton	1	US$101.00	0.40	US$252.50	US$101.00
EWR	Newark	27	US$91.25	10.77	US$150.24	US$2,463.81
N/A	Outside of NYC	338	US$99.58	10.70	US$77.96	US$33,659.18
Staten Island	Arrochar/Fort Wadsworth	13	US$33.86	2.76	US$57.16	US$440.24
Staten Island	Bloomfield/Emerson Hill	1	US$7.30	0.60	US$12.17	US$7.30
Staten Island	Heartland Village/Todt Hill	3	US$138.15	18.74	US$10.17	US$414.46
Staten Island	Oakwood	1	US$39.30	6.22	US$6.32	US$39.30
Staten Island	Charleston/Tottenville	1	US$354.23	57.20	US$6.19	US$354.23
Staten Island	Westerleigh	1	US$106.74	17.60	US$6.06	US$106.74
Staten Island	New Dorp/Midland Beach	2	US$100.00	18.06	US$5.66	US$200.00

Pickup Zone Efficiency (Revenue per Mile)

Showing results from 2023-12-31 to 2023-12-31 for All Boroughs

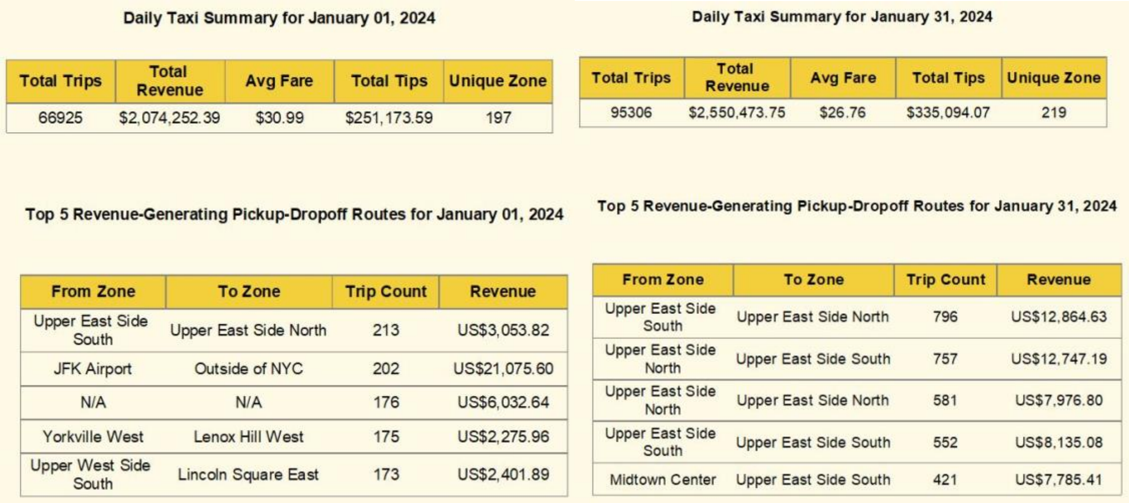
Borough	Pickup Zone	Trip Count	Avg Fare	Avg Distance	Revenue Per Mile	Total Revenue
Manhattan	Midtown Center	1	US$18.75	0.59	US$31.78	US$18.75
Manhattan	West Chelsea/Hudson Yards	1	US$12.20	0.40	US$30.50	US$12.20
Manhattan	Little Italy/NoLiTa	1	US$12.96	0.53	US$24.45	US$12.96
Manhattan	Flatiron	1	US$10.10	0.47	US$21.49	US$10.10
Manhattan	Midtown North	1	US$13.50	0.97	US$13.92	US$13.50
Manhattan	East Chelsea	1	US$18.84	1.44	US$13.08	US$18.84
Manhattan	Union Sq	1	US$28.60	3.14	US$9.11	US$28.60
Manhattan	Upper East Side North	1	US$21.60	2.38	US$9.08	US$21.60
Manhattan	Sutton Place/Turtle Bay North	1	US$45.72	7.70	US$5.94	US$45.72
Queens	LaGuardia Airport	1	US$42.35	8.39	US$5.05	US$42.35

Figure 13: Parameterized Location Efficiency Report

Report 4: Executive Daily Dashboard with SubReport

Target Stakeholder: C-Level Executive / General Manager

Business Value: The tabular report (Fig. 14) provides daily performance summary with the first table showing the days summary and SubReport showing top 5 profitable locations. The reports are controlled by a date parameter to filter that particular day's summary. These help executives monitor daily results and make any quick decisions based on the data.



Daily Taxi Summary for January 01, 2024

Total Trips	Total Revenue	Avg Fare	Total Tips	Unique Zone
66925	$2,074,252.39	$30.99	$251,173.59	197

Top 5 Revenue-Generating Pickup-Dropoff Routes for January 01, 2024

From Zone	To Zone	Trip Count	Revenue
Upper East Side South	Upper East Side North	213	US$3,053.82
JFK Airport	Outside of NYC	202	US$21,075.60
N/A	N/A	176	US$6,032.64
Yorkville West	Lenox Hill West	175	US$2,275.96
Upper West Side South	Lincoln Square East	173	US$2,401.89

Daily Taxi Summary for January 31, 2024

Total Trips	Total Revenue	Avg Fare	Total Tips	Unique Zone
95306	$2,550,473.75	$26.76	$335,094.07	219

Top 5 Revenue-Generating Pickup-Dropoff Routes for January 31, 2024

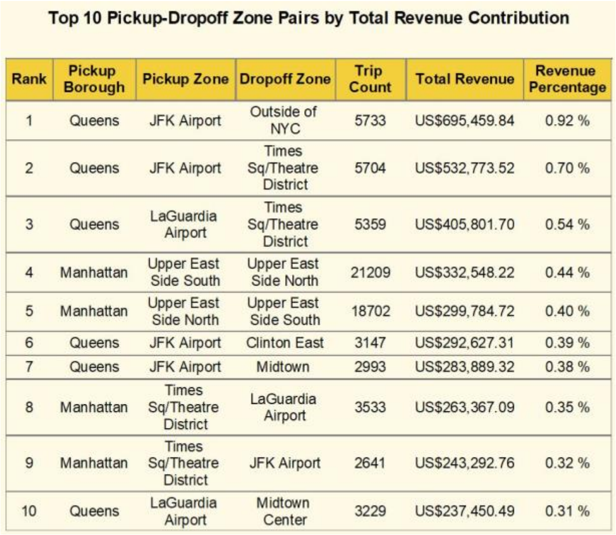
From Zone	To Zone	Trip Count	Revenue
Upper East Side South	Upper East Side North	796	US$12,864.63
Upper East Side North	Upper East Side South	757	US$12,747.19
Upper East Side North	Upper East Side North	581	US$7,976.80
Upper East Side South	Upper East Side South	552	US$8,135.08
Midtown Center	Upper East Side South	421	US$7,785.41

Figure 14: Executive Summary Dashboard

Report 5: Revenue Analysis (additional report)

Target Stakeholder: Revenue Manager / Financial Analyst

Business Value: The tabular report (Fig. 15) shows the top 10 revenue routes to help deciding what routes to prioritize and how best to maximise the revenue.



Top 10 Pickup-Dropoff Zone Pairs by Total Revenue Contribution

Rank	Pickup Borough	Pickup Zone	Dropoff Zone	Trip Count	Total Revenue	Revenue Percentage
1	Queens	JFK Airport	Outside of NYC	5733	US$695,459.84	0.92 %
2	Queens	JFK Airport	Times Sq/Theatre District	5704	US$532,773.52	0.70 %
3	Queens	LaGuardia Airport	Times Sq/Theatre District	5359	US$405,801.70	0.54 %
4	Manhattan	Upper East Side South	Upper East Side North	21209	US$332,548.22	0.44 %
5	Manhattan	Upper East Side North	Upper East Side South	18702	US$299,784.72	0.40 %
6	Queens	JFK Airport	Clinton East	3147	US$292,627.31	0.39 %
7	Queens	JFK Airport	Midtown	2993	US$283,889.32	0.38 %
8	Manhattan	Times Sq/Theatre District	LaGuardia Airport	3533	US$263,367.09	0.35 %
9	Manhattan	Times Sq/Theatre District	JFK Airport	2641	US$243,292.76	0.32 %
10	Queens	LaGuardia Airport	Midtown Center	3229	US$237,450.49	0.31 %

Figure 15: Top Routes Revenue Analysis Report

All SSRS reports have been properly formatted and include relevant visualisations that help give specific answers to different stakeholders. Full details of the report designs and parameter configurations are available in Appendix E.

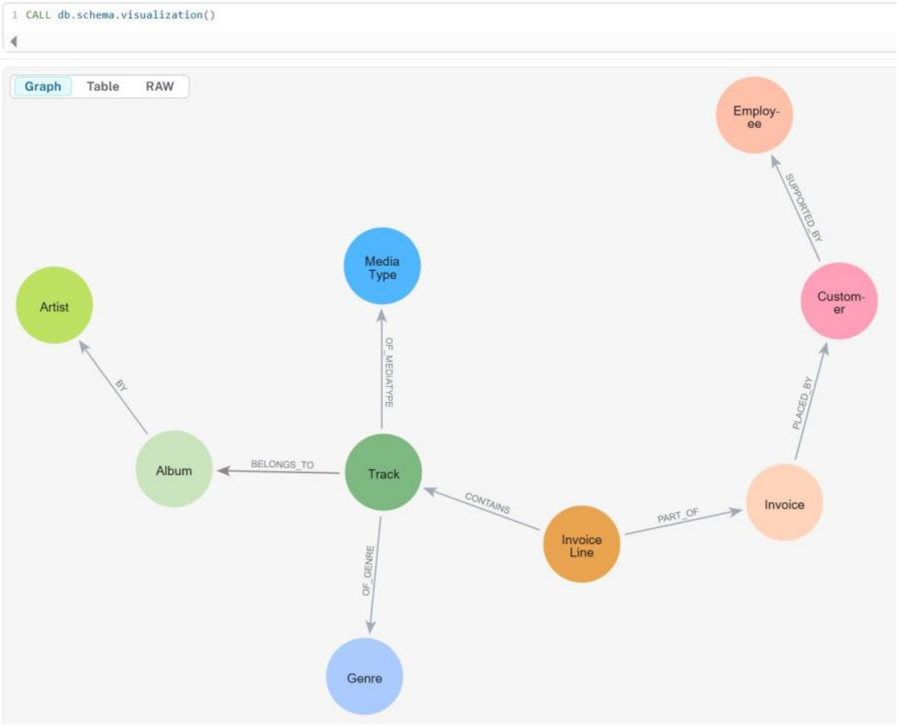
5. Graph and Relational Database Comparison

For graph and relational database comparison we used the Chinook music store dataset (Rocha, 2024). It gives us complete relationships between customers, artists, albums, tracks, invoices and employees. For comparing the dataset in SQL and CQL, it was loaded into SSMS using a script that is available on the GitHub page for the dataset. Individual CSV files were then exported from SSMS to be imported in Neo4j Aura Cloud Instance.

5.1 Neo4j Implementation Details

The graph schema has 9 nodes (Fig. 16), Track, Genre, Album, MediaType, Artist, Invoice, InvoiceLine, Customer, Employee. Unlike relational models which need foreign keys and JOINS, graph models have relationships that link all the nodes with each other which makes it very flexible. All nodes for our dataset have the following relationships:

- Track BELONGS_TO Album
- Track OF_GENRE Genre
- Track OF_MEDIATYPE MediaType
- InvoiceLine PART_OF Invoice
- InvoiceLine CONTAINS Track
- Invoice PLACED_BY Customer
- Customer SUPPORTED_BY Employee
- Album BY Artist



```
CALL db.schema.visualization()
```

Figure 16: Graph Schema

5.2 Comparison to Relational Databases

For our comparison we used the below seven analytical questions to write the queries and then compared their performances:

- Top customer revenue analysis – See which customers spend the most for targeted marketing.
- Artist revenue contribution – Figure out which artists generate most revenue to deciding who to invest in more.
- Geographic genre preferences – To identify most popular genres in each country for regional marketing decisions.
- Customer purchase diversity – Check different music tastes of customers for cross-selling opportunities.
- Employee performance analysis – Ranking top employees who are performing the best and decide how much bonus to give.
- Artist co-purchase patterns – To identify which artists tracks are bought together for bundle strategies.
- Track recommendations – To identify which tracks are purchased together most for recommending more tracks to customers ("Customers also bought...")

Importing data into Neo4j took only four seconds while SSMS took double the time to import around eight seconds.

Performance Metrics Summary: Below are the query speed comparisons for both databases

Query Type	Neo4j (CQL)	SSMS (SQL Server)
Top Customers	51ms	2ms
Artist Revenue	103ms	17ms
Genre by Country	129ms	13ms
Customer Diversity	72ms	19ms
Employee Performance	38ms	6ms
Artist Co-purchases	403ms	36ms
Track Recommendations	115ms	102ms

From the results it is clear that SQL querying was much faster than CQL querying. This is because relational databases are better for aggregations (SUM, COUNT, AVG) and multi table joins. All the SQL and CQL queries can be found in Appendix B.

6. Conclusion

6.1 Key Achievements

This assignment successfully delivers a complete data management and analytics solution for NYC taxi operations. The implementation helps demonstrate usage of multiple tools like SSMS for database setup, SSIS for ETL process, SSRS for quick report generation and Tableau for interactive visualisations.

Technical Deliverables completed:

- Star schema Datawarehouse with four dimension tables and one fact table.
- ETL pipeline able to process 100% of raw data with proper validations.
- Five SSRS reports with drill-down and parameter capabilities.
- Four interactive tableau visualisations with cross filtering in dashboard.
- Comparison of Graph and Relational database for Chinook database performances using Neo4j and SSMS.

6.2 Business Impact and Value

Technical Deliverables completed: The reports and visualisations provide actionable insights for transportation operations (NYC taxi) and music retail analytics (Chinook).

Technical Deliverables completed: The geographic analysis helps identifying investment opportunities. The demand pattern analysis provides better fleet optimization strategies and evaluating vendor performances help in data driven contract negotiations.

Database Performance: The comparison of relational and graph database reveals that SQL databases are much better for analytical operations. But graph databases are much easier to understand and provide better relationship modelling.

6.3 Future Scalability Recommendations

The dimensional model and the ETL framework are designed so that they can support importing of large amounts of data and any other additional relations without needing many changes. This also make it easier to support advanced analytics.

Future enhancements could also include real time data streaming to allow predictive analytics for demand forecasting.

7. Bibliography

Microsoft Corporation (2024) SQL Server Integration Services (SSIS) developer documentation. Available at: https://docs.microsoft.com/en-us/sql/integration-services/ (Accessed: 10 July 2025).

Microsoft Corporation (2025) SQL Server Reporting Services (SSRS) technical reference. Available at: https://docs.microsoft.com/en-us/sql/reporting-services/ (Accessed: 15 July 2025).

Neo4j, Inc. (2025) Neo4j graph database documentation. Available at: https://neo4j.com/docs/ (Accessed: 23 July 2025).

NYC Taxi & Limousine Commission (2025) TLC trip record data. Available at: https://www.nyc.gov/site/tlc/about/tlc-trip-record-data.page (Accessed: 1 July 2025).

Neo4j in 100 Seconds. YouTube, 24 January 2023. Available at: https://www.youtube.com/watch?v=T6L9EoBy8Zk (Accessed: 24 July 2025).

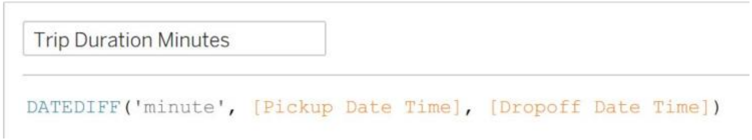
Rocha, L., 2024. Chinook database. Available at: https://github.com/lerocha/chinook-database [Accessed 26 July 2025]

Tableau Software (2024) Tableau desktop user guide. Available at: https://www.tableau.com/blog/beginners-guide-tableau-public (Accessed: 18 July 2025).

Appendix A: Visualization Code

Tableau Calculated Fields:

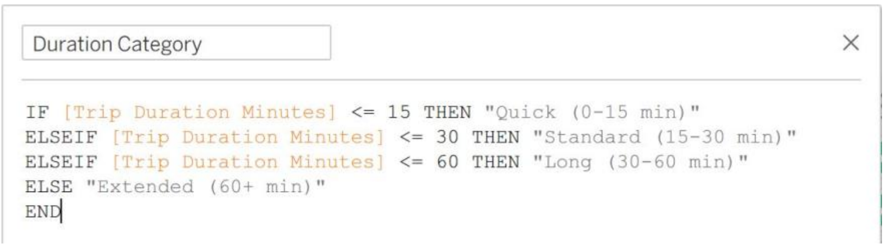
Trip Duration- This is a support field for getting Trip duration categories



Trip Duration Minutes

```
DATEDIFF('minute', [Pickup Date Time], [Dropoff Date Time])
```

Trip Duration Categories-

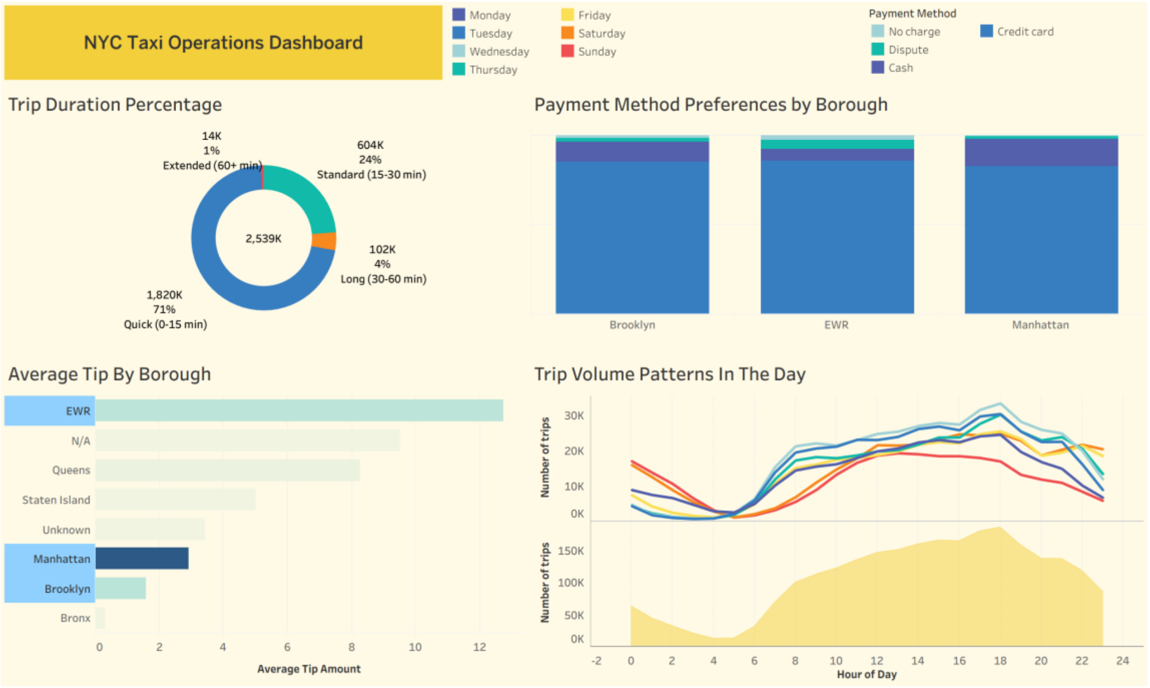


Duration Category

```
IF [Trip Duration Minutes] <= 15 THEN "Quick (0-15 min)"
ELSEIF [Trip Duration Minutes] <= 30 THEN "Standard (15-30 min)"
ELSEIF [Trip Duration Minutes] <= 60 THEN "Long (30-60 min)"
ELSE "Extended (60+ min)"
END
```

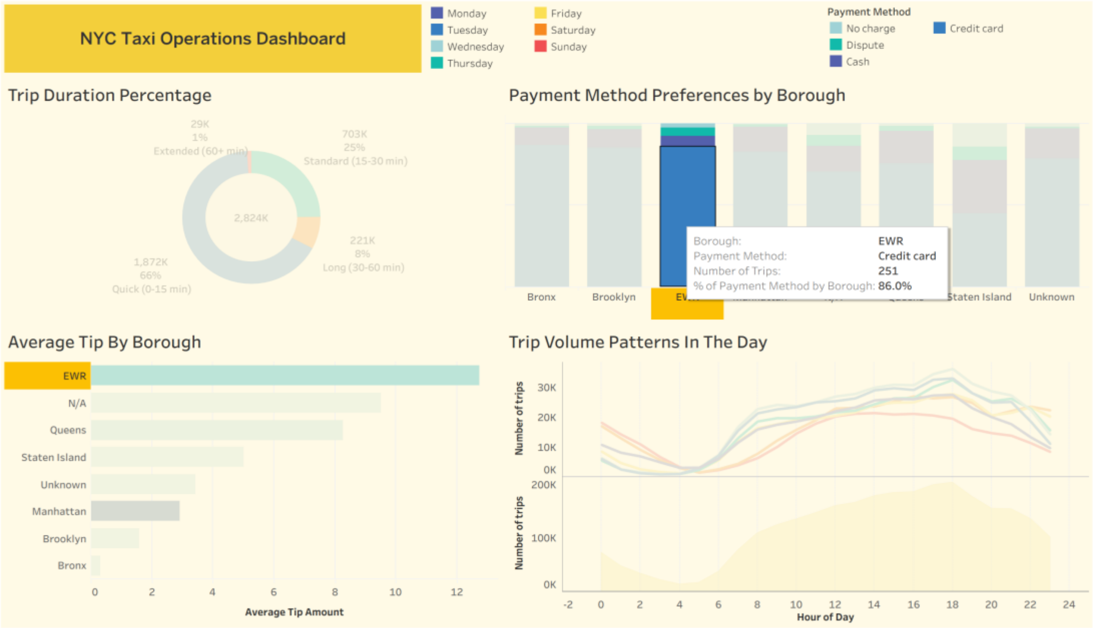
Cross-filtering-

Clicking on any single (or multiple boroughs) will result in all 4 visualisations to show data only for the selected boroughs.



Highlighting-

Hovering over any of the boroughs in either the stacked bar chart or horizontal bar chart highlights the specific boroughs in both charts.

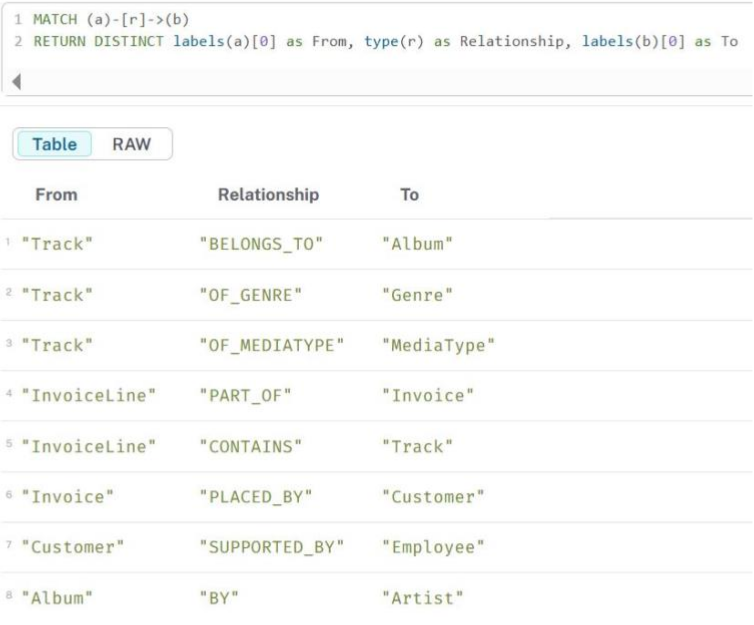


Borough: EWR
Payment Method: Credit card
Number of Trips: 251
% of Payment Method by Borough: 86.0%

Appendix B: Neo4j Code

Data loading

For importing data in SSMS, the script available on Chinook datasets GitHub page was used. The tables were then exported as CSV's and then used to import them in Neo4j Aura Cloud. The data model was then defined manually by creating individual nodes and defining relationships accordingly.

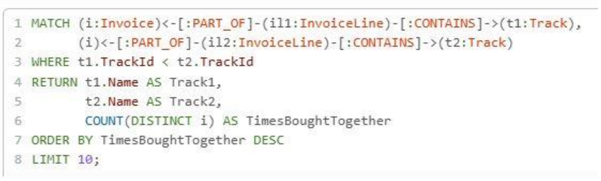


```
MATCH (a)-[r]->(b)
RETURN DISTINCT labels(a)[0] as From, type(r) as Relationship, labels(b)[0] as To
```

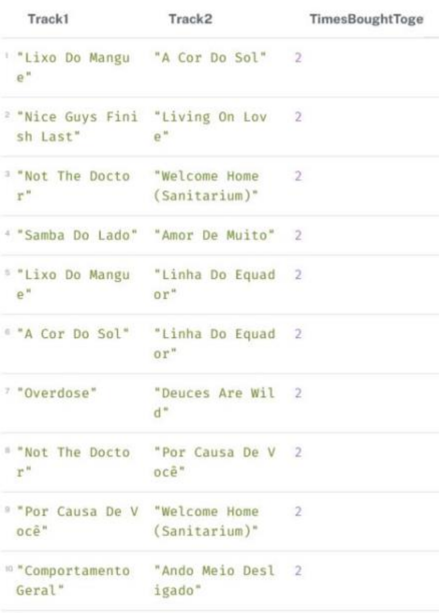
From	Relationship	To
"Track"	"BELONGS_TO"	"Album"
"Track"	"OF_GENRE"	"Genre"
"Track"	"OF_MEDIATYPE"	"MediaType"
"InvoiceLine"	"PART_OF"	"Invoice"
"InvoiceLine"	"CONTAINS"	"Track"
"Invoice"	"PLACED_BY"	"Customer"
"Customer"	"SUPPORTED_BY"	"Employee"
"Album"	"BY"	"Artist"

CQL Scripts with results

Query 1

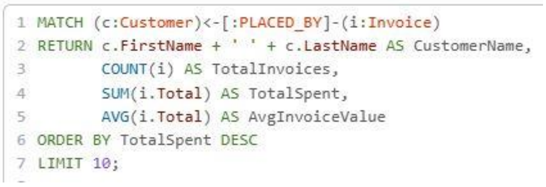


```
MATCH (i:Invoice)<-[:PART_OF]-(il1:InvoiceLine)-[:CONTAINS]->(t1:Track),
      (i)<-[:PART_OF]-(il2:InvoiceLine)-[:CONTAINS]->(t2:Track)
WHERE t1.TrackId < t2.TrackId
RETURN t1.Name AS Track1,
       t2.Name AS Track2,
       COUNT(DISTINCT i) AS TimesBoughtTogether
ORDER BY TimesBoughtTogether DESC
LIMIT 10;
```

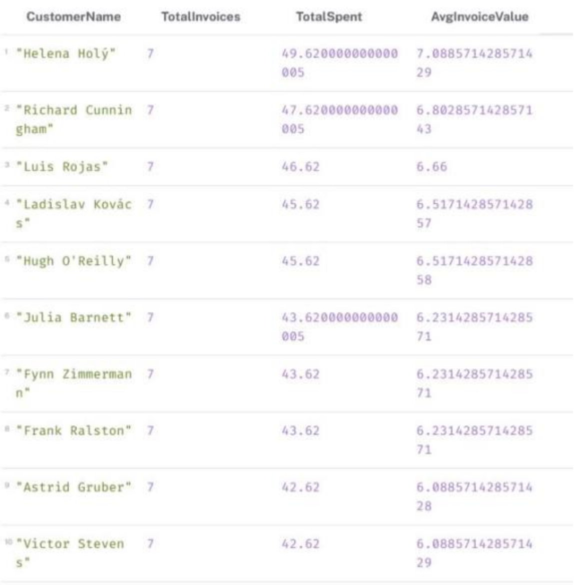


Track1	Track2	TimesBoughtToge
"Lixo Do Mangue"	"A Cor Do Sol"	2
"Nice Guys Finish Last"	"Living On Love"	2
"Not The Doctor"	"Welcome Home (Sanitarium)"	2
"Samba Do Lado"	"Amor De Muito"	2
"Lixo Do Mangue"	"Linha Do Equador"	2
"A Cor Do Sol"	"Linha Do Equador"	2
"Overdose"	"Deuces Are Wild"	2
"Not The Doctor"	"Por Causa De Você"	2
"Por Causa De Você"	"Welcome Home (Sanitarium)"	2
"Comportamento Geral"	"Ando Meio Desligado"	2

Query 2

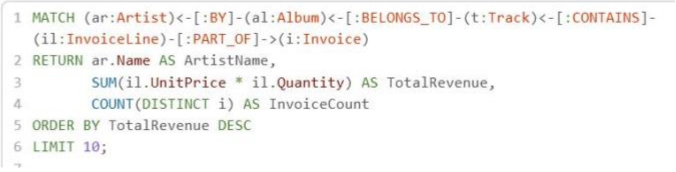


```
MATCH (c:Customer)<-[:PLACED_BY]-(i:Invoice)
RETURN c.FirstName + ' ' + c.LastName AS CustomerName,
       COUNT(i) AS TotalInvoices,
       SUM(i.Total) AS TotalSpent,
       AVG(i.Total) AS AvgInvoiceValue
ORDER BY TotalSpent DESC
LIMIT 10;
```

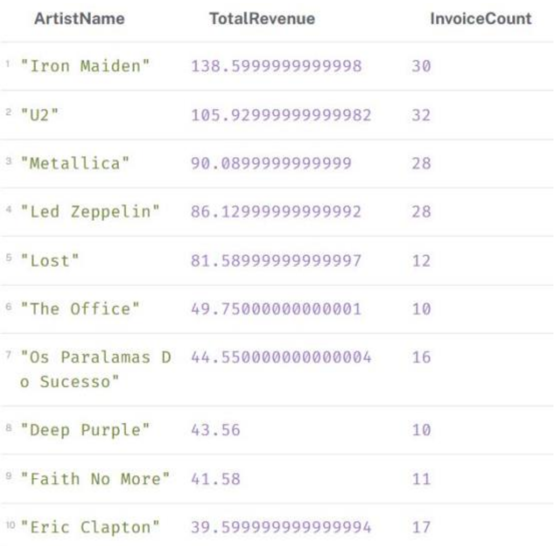


CustomerName	TotalInvoices	TotalSpent	AvgInvoiceValue
"Helena Holý"	7	49.620000000000005	7.088571428571429
"Richard Cunningham"	7	47.620000000000005	6.802857142857143
"Luis Rojas"	7	46.62	6.66
"Ladislav Kovács"	7	45.62	6.517142857142857
"Hugh O'Reilly"	7	45.62	6.517142857142858
"Julia Barnett"	7	43.620000000000005	6.231428571428571
"Fynn Zimmermann"	7	43.62	6.231428571428571
"Frank Ralston"	7	43.62	6.231428571428571
"Astrid Gruber"	7	42.62	6.088571428571428
"Victor Stevens"	7	42.62	6.088571428571429

Query 3

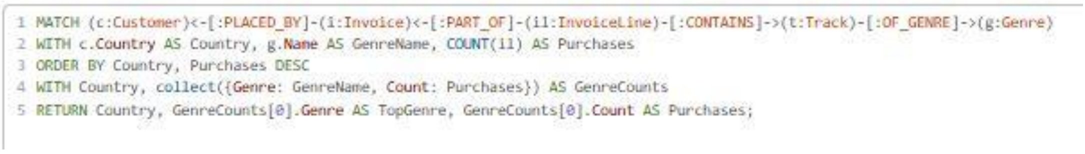


```
MATCH (ar:Artist)<-[:BY]-(al:Album)<-[:BELONGS_TO]-(t:Track)<-[:CONTAINS]-
  (il:InvoiceLine)-[:PART_OF]->(i:Invoice)
RETURN ar.Name AS ArtistName,
       SUM(il.UnitPrice * il.Quantity) AS TotalRevenue,
       COUNT(DISTINCT i) AS InvoiceCount
ORDER BY TotalRevenue DESC
LIMIT 10;
```

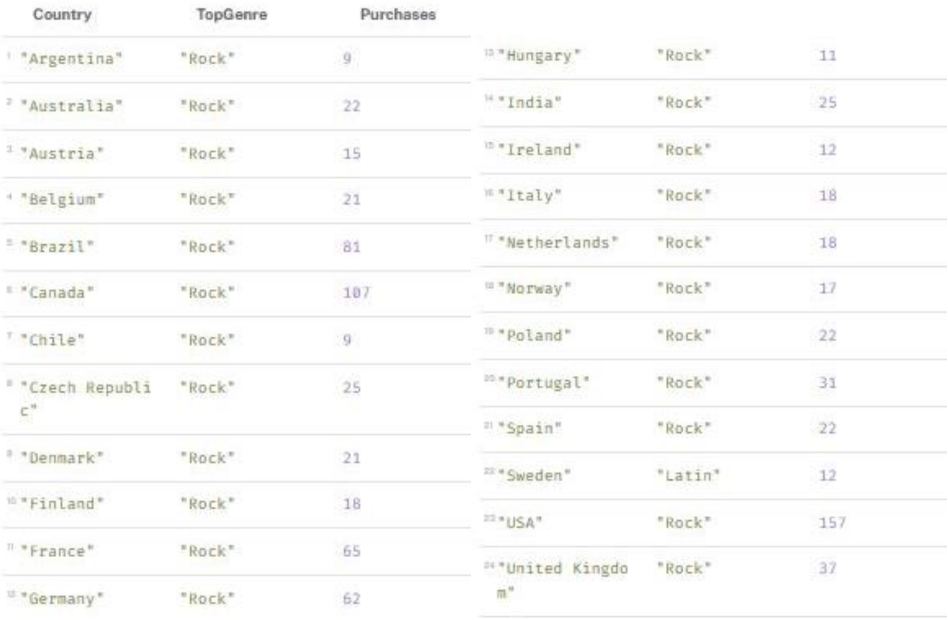


ArtistName	TotalRevenue	InvoiceCount
"Iron Maiden"	138.5999999999998	30
"U2"	105.92999999999982	32
"Metallica"	90.0899999999999	28
"Led Zeppelin"	86.12999999999992	28
"Lost"	81.58999999999997	12
"The Office"	49.75000000000001	10
"Os Paralamas Do Sucesso"	44.550000000000004	16
"Deep Purple"	43.56	10
"Faith No More"	41.58	11
"Eric Clapton"	39.599999999999994	17

Query 4

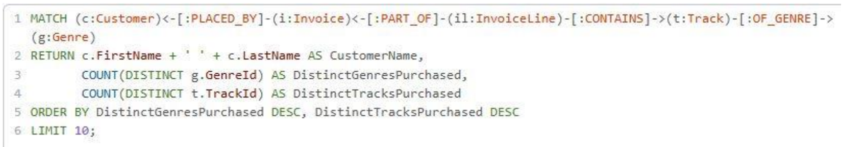


```
MATCH (c:Customer)<-[:PLACED_BY]-(i:Invoice)<-[:PART_OF]-(il:InvoiceLine)-[:CONTAINS]->(t:Track)-[:OF_GENRE]->(g:Genre)
WITH c.Country AS Country, g.Name AS GenreName, COUNT(il) AS Purchases
ORDER BY Country, Purchases DESC
WITH Country, collect({Genre: GenreName, Count: Purchases}) AS GenreCounts
RETURN Country, GenreCounts[0].Genre AS TopGenre, GenreCounts[0].Count AS Purchases;
```

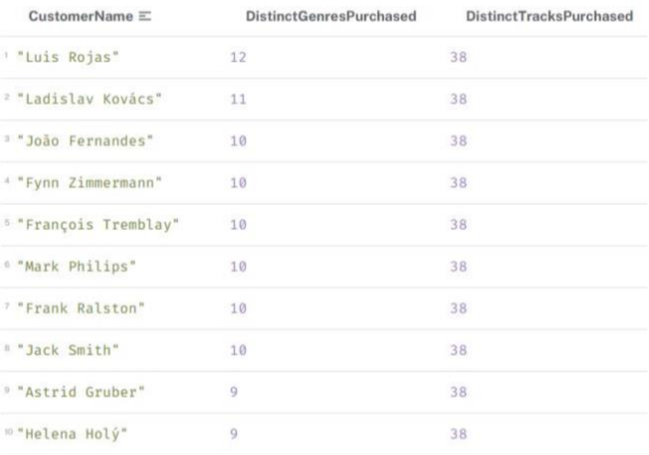


Country	TopGenre	Purchases
"Argentina"	"Rock"	9
"Australia"	"Rock"	22
"Austria"	"Rock"	15
"Belgium"	"Rock"	21
"Brazil"	"Rock"	81
"Canada"	"Rock"	107
"Chile"	"Rock"	9
"Czech Republic"	"Rock"	25
"Denmark"	"Rock"	21
"Finland"	"Rock"	18
"France"	"Rock"	65
"Germany"	"Rock"	62
"Hungary"	"Rock"	11
"India"	"Rock"	25
"Ireland"	"Rock"	12
"Italy"	"Rock"	18
"Netherlands"	"Rock"	18
"Norway"	"Rock"	17
"Poland"	"Rock"	22
"Portugal"	"Rock"	31
"Spain"	"Rock"	22
"Sweden"	"Latin"	12
"USA"	"Rock"	157
"United Kingdom"	"Rock"	37

Query 5

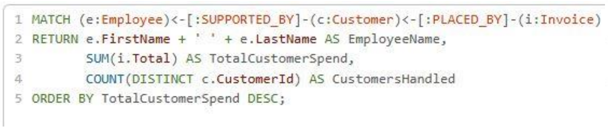


```
MATCH (c:Customer)<-[:PLACED_BY]-(i:Invoice)<-[:PART_OF]-(il:InvoiceLine)-[:CONTAINS]->(t:Track)-[:OF_GENRE]->
(g:Genre)
RETURN c.FirstName + ' ' + c.LastName AS CustomerName,
       COUNT(DISTINCT g.GenreId) AS DistinctGenresPurchased,
       COUNT(DISTINCT t.TrackId) AS DistinctTracksPurchased
ORDER BY DistinctGenresPurchased DESC, DistinctTracksPurchased DESC
LIMIT 10;
```

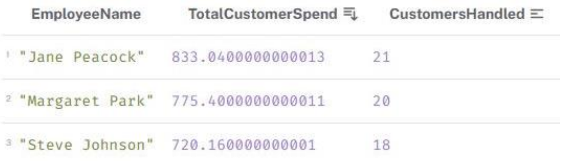


	CustomerName	DistinctGenresPurchased	DistinctTracksPurchased
1	"Luis Rojas"	12	38
2	"Ladislav Kovács"	11	38
3	"João Fernandes"	10	38
4	"Fynn Zimmermann"	10	38
5	"François Tremblay"	10	38
6	"Mark Philips"	10	38
7	"Frank Ralston"	10	38
8	"Jack Smith"	10	38
9	"Astrid Gruber"	9	38
10	"Helena Holý"	9	38

Query 6

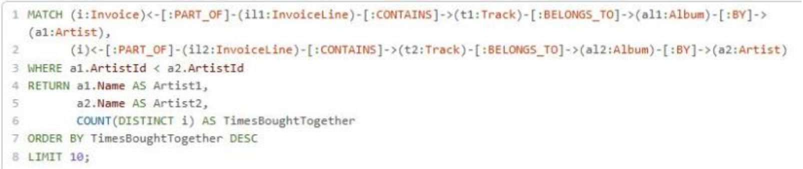


```
MATCH (e:Employee)<-[:SUPPORTED_BY]-(c:Customer)<-[:PLACED_BY]-(i:Invoice)
RETURN e.FirstName + ' ' + e.LastName AS EmployeeName,
       SUM(i.Total) AS TotalCustomerSpend,
       COUNT(DISTINCT c.CustomerId) AS CustomersHandled
ORDER BY TotalCustomerSpend DESC;
```

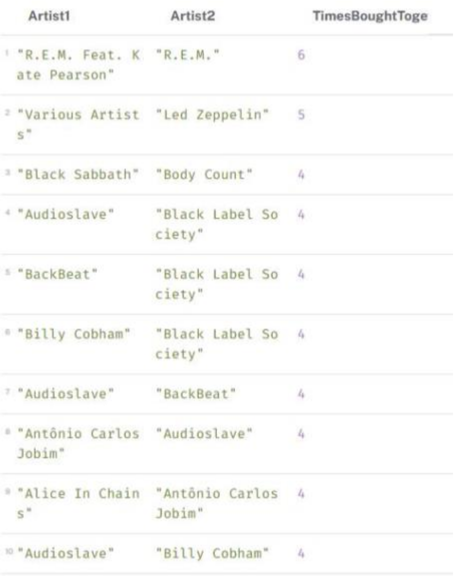


	EmployeeName	TotalCustomerSpend	CustomersHandled
1	"Jane Peacock"	833.0400000000013	21
2	"Margaret Park"	775.4000000000011	20
3	"Steve Johnson"	720.160000000001	18

Query 7



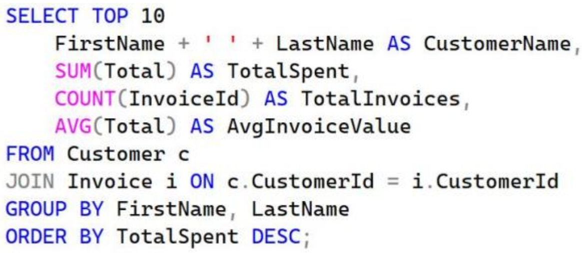
```
MATCH (i:Invoice)<-[:PART_OF]-(il1:InvoiceLine)-[:CONTAINS]->(t1:Track)-[:BELONGS_TO]->(al1:Album)-[:BY]->
(a1:Artist),
      (i)<-[:PART_OF]-(il2:InvoiceLine)-[:CONTAINS]->(t2:Track)-[:BELONGS_TO]->(al2:Album)-[:BY]->(a2:Artist)
WHERE a1.ArtistId < a2.ArtistId
RETURN a1.Name AS Artist1,
       a2.Name AS Artist2,
       COUNT(DISTINCT i) AS TimesBoughtTogether
ORDER BY TimesBoughtTogether DESC
LIMIT 10;
```



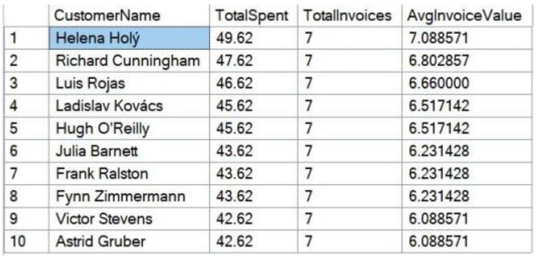
	Artist1	Artist2	TimesBoughtToge
1	"R.E.M. Feat. Kate Pearson"	"R.E.M."	6
2	"Various Artists"	"Led Zeppelin"	5
3	"Black Sabbath"	"Body Count"	4
4	"Audioslave"	"Black Label Society"	4
5	"BackBeat"	"Black Label Society"	4
6	"Billy Cobham"	"Black Label Society"	4
7	"Audioslave"	"BackBeat"	4
8	"Antônio Carlos Jobim"	"Audioslave"	4
9	"Alice In Chains"	"Antônio Carlos Jobim"	4
10	"Audioslave"	"Billy Cobham"	4

SQL Scripts with results

Query 1

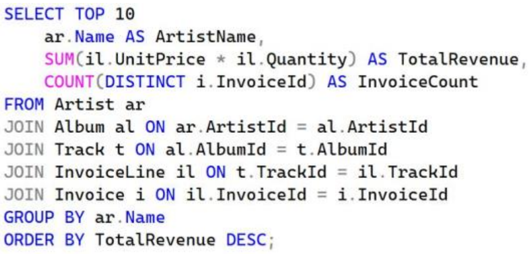


```
SELECT TOP 10
    FirstName + ' ' + LastName AS CustomerName,
    SUM(Total) AS TotalSpent,
    COUNT(InvoiceId) AS TotalInvoices,
    AVG(Total) AS AvgInvoiceValue
FROM Customer c
JOIN Invoice i ON c.CustomerId = i.CustomerId
GROUP BY FirstName, LastName
ORDER BY TotalSpent DESC;
```

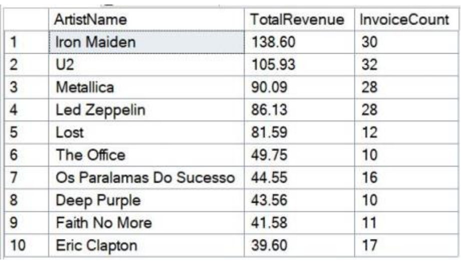


	CustomerName	TotalSpent	TotalInvoices	AvgInvoiceValue
1	Helena Holý	49.62	7	7.088571
2	Richard Cunningham	47.62	7	6.802857
3	Luis Rojas	46.62	7	6.660000
4	Ladislav Kovács	45.62	7	6.517142
5	Hugh O'Reilly	45.62	7	6.517142
6	Julia Barnett	43.62	7	6.231428
7	Frank Ralston	43.62	7	6.231428
8	Fynn Zimmermann	43.62	7	6.231428
9	Victor Stevens	42.62	7	6.088571
10	Astrid Gruber	42.62	7	6.088571

Query 2

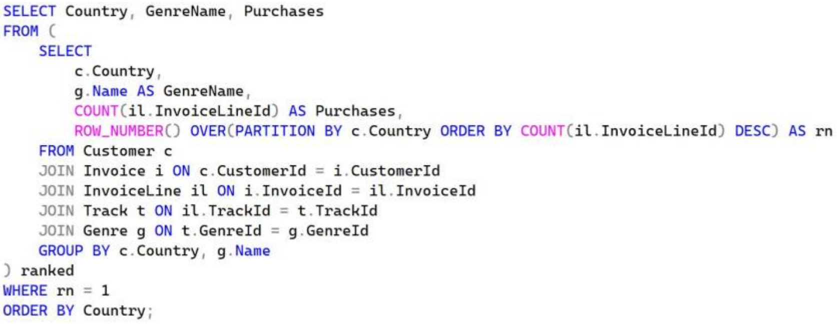


```
SELECT TOP 10
    ar.Name AS ArtistName,
    SUM(il.UnitPrice * il.Quantity) AS TotalRevenue,
    COUNT(DISTINCT i.InvoiceId) AS InvoiceCount
FROM Artist ar
JOIN Album al ON ar.ArtistId = al.ArtistId
JOIN Track t ON al.AlbumId = t.AlbumId
JOIN InvoiceLine il ON t.TrackId = il.TrackId
JOIN Invoice i ON il.InvoiceId = i.InvoiceId
GROUP BY ar.Name
ORDER BY TotalRevenue DESC;
```

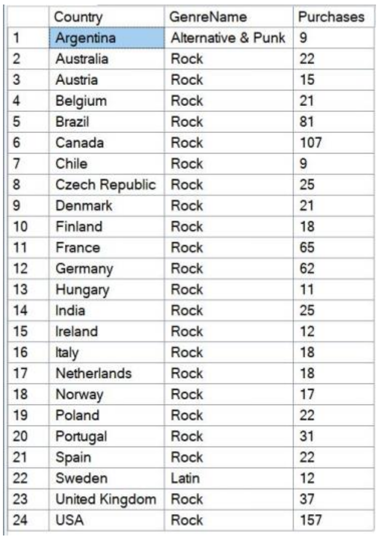


	ArtistName	TotalRevenue	InvoiceCount
1	Iron Maiden	138.60	30
2	U2	105.93	32
3	Metallica	90.09	28
4	Led Zeppelin	86.13	28
5	Lost	81.59	12
6	The Office	49.75	10
7	Os Paralamas Do Sucesso	44.55	16
8	Deep Purple	43.56	10
9	Faith No More	41.58	11
10	Eric Clapton	39.60	17

Query 3

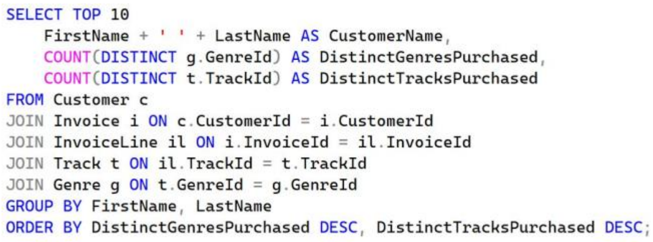


```
SELECT Country, GenreName, Purchases
FROM (
    SELECT
        c.Country,
        g.Name AS GenreName,
        COUNT(il.InvoiceLineId) AS Purchases,
        ROW_NUMBER() OVER(PARTITION BY c.Country ORDER BY COUNT(il.InvoiceLineId) DESC) AS rn
    FROM Customer c
    JOIN Invoice i ON c.CustomerId = i.CustomerId
    JOIN InvoiceLine il ON i.InvoiceId = il.InvoiceId
    JOIN Track t ON il.TrackId = t.TrackId
    JOIN Genre g ON t.GenreId = g.GenreId
    GROUP BY c.Country, g.Name
) ranked
WHERE rn = 1
ORDER BY Country;
```

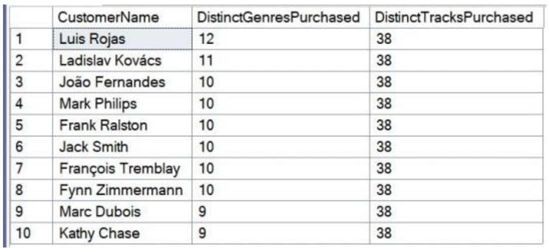


	Country	GenreName	Purchases
1	Argentina	Alternative & Punk	9
2	Australia	Rock	22
3	Austria	Rock	15
4	Belgium	Rock	21
5	Brazil	Rock	81
6	Canada	Rock	107
7	Chile	Rock	9
8	Czech Republic	Rock	25
9	Denmark	Rock	21
10	Finland	Rock	18
11	France	Rock	65
12	Germany	Rock	62
13	Hungary	Rock	11
14	India	Rock	25
15	Ireland	Rock	12
16	Italy	Rock	18
17	Netherlands	Rock	18
18	Norway	Rock	17
19	Poland	Rock	22
20	Portugal	Rock	31
21	Spain	Rock	22
22	Sweden	Latin	12
23	United Kingdom	Rock	37
24	USA	Rock	157

Query 4

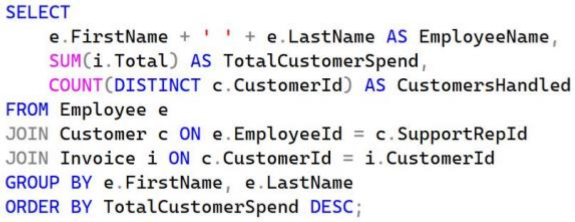


```
SELECT TOP 10
    FirstName + ' ' + LastName AS CustomerName,
    COUNT(DISTINCT g.GenreId) AS DistinctGenresPurchased,
    COUNT(DISTINCT t.TrackId) AS DistinctTracksPurchased
FROM Customer c
JOIN Invoice i ON c.CustomerId = i.CustomerId
JOIN InvoiceLine il ON i.InvoiceId = il.InvoiceId
JOIN Track t ON il.TrackId = t.TrackId
JOIN Genre g ON t.GenreId = g.GenreId
GROUP BY FirstName, LastName
ORDER BY DistinctGenresPurchased DESC, DistinctTracksPurchased DESC;
```

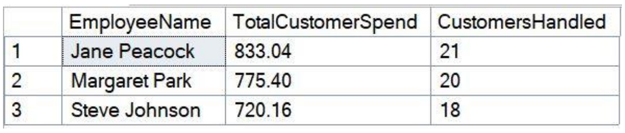


	CustomerName	DistinctGenresPurchased	DistinctTracksPurchased
1	Luis Rojas	12	38
2	Ladislav Kovács	11	38
3	João Fernandes	10	38
4	Mark Philips	10	38
5	Frank Ralston	10	38
6	Jack Smith	10	38
7	François Tremblay	10	38
8	Fynn Zimmermann	10	38
9	Marc Dubois	9	38
10	Kathy Chase	9	38

Query 5

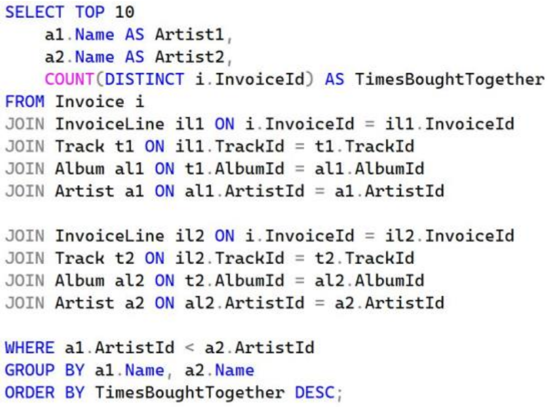


```
SELECT
    e.FirstName + ' ' + e.LastName AS EmployeeName,
    SUM(i.Total) AS TotalCustomerSpend,
    COUNT(DISTINCT c.CustomerId) AS CustomersHandled
FROM Employee e
JOIN Customer c ON e.EmployeeId = c.SupportRepId
JOIN Invoice i ON c.CustomerId = i.CustomerId
GROUP BY e.FirstName, e.LastName
ORDER BY TotalCustomerSpend DESC;
```

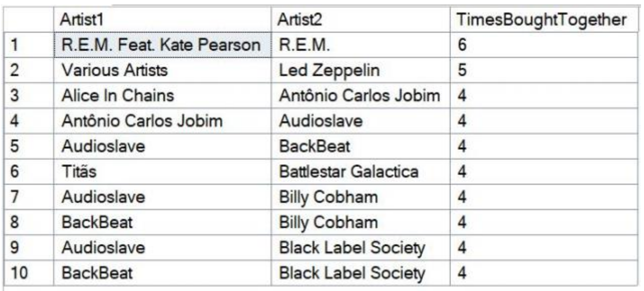


	EmployeeName	TotalCustomerSpend	CustomersHandled
1	Jane Peacock	833.04	21
2	Margaret Park	775.40	20
3	Steve Johnson	720.16	18

Query 6

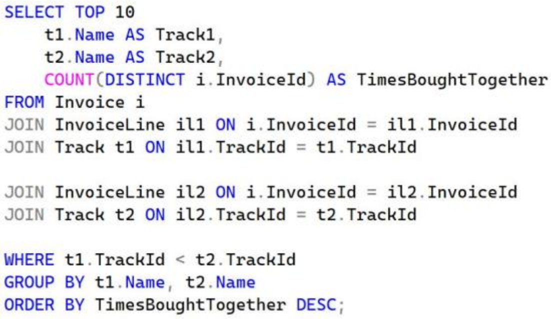


```
SELECT TOP 10
    a1.Name AS Artist1,
    a2.Name AS Artist2,
    COUNT(DISTINCT i.InvoiceId) AS TimesBoughtTogether
FROM Invoice i
JOIN InvoiceLine il1 ON i.InvoiceId = il1.InvoiceId
JOIN Track t1 ON il1.TrackId = t1.TrackId
JOIN Album al1 ON t1.AlbumId = al1.AlbumId
JOIN Artist a1 ON al1.ArtistId = a1.ArtistId

JOIN InvoiceLine il2 ON i.InvoiceId = il2.InvoiceId
JOIN Track t2 ON il2.TrackId = t2.TrackId
JOIN Album al2 ON t2.AlbumId = al2.AlbumId
JOIN Artist a2 ON al2.ArtistId = a2.ArtistId

WHERE a1.ArtistId < a2.ArtistId
GROUP BY a1.Name, a2.Name
ORDER BY TimesBoughtTogether DESC;
```

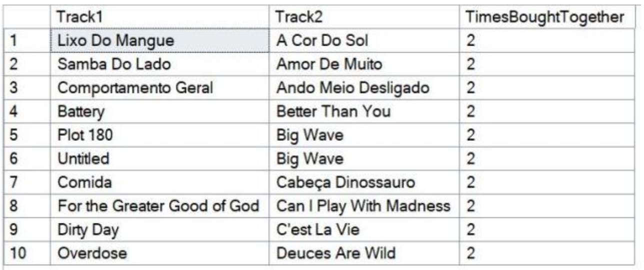


	Artist1	Artist2	TimesBoughtTogether
1	R.E.M. Feat. Kate Pearson	R.E.M.	6
2	Various Artists	Led Zeppelin	5
3	Alice In Chains	Antônio Carlos Jobim	4
4	Antônio Carlos Jobim	Audioslave	4
5	Audioslave	BackBeat	4
6	Titãs	Battlestar Galactica	4
7	Audioslave	Billy Cobham	4
8	BackBeat	Billy Cobham	4
9	Audioslave	Black Label Society	4
10	BackBeat	Black Label Society	4

Query 7



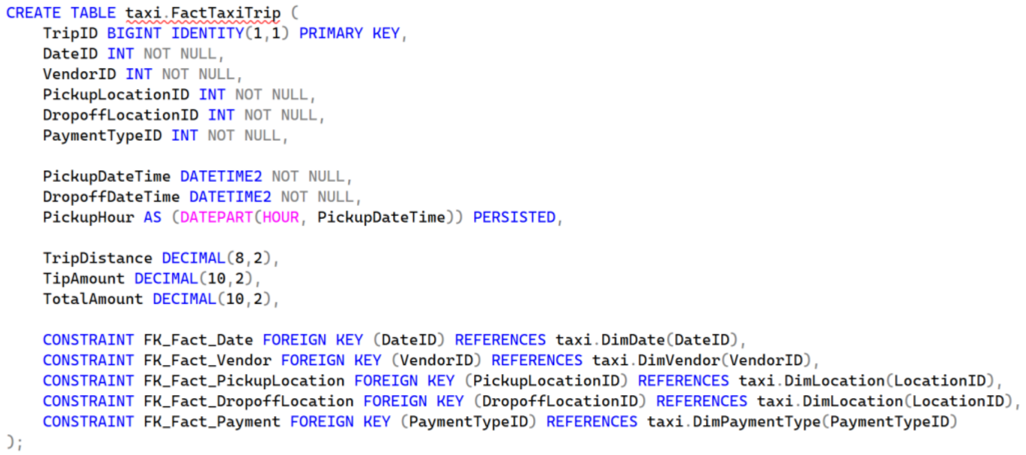
```
SELECT TOP 10
    t1.Name AS Track1,
    t2.Name AS Track2,
    COUNT(DISTINCT i.InvoiceId) AS TimesBoughtTogether
FROM Invoice i
JOIN InvoiceLine il1 ON i.InvoiceId = il1.InvoiceId
JOIN Track t1 ON il1.TrackId = t1.TrackId

JOIN InvoiceLine il2 ON i.InvoiceId = il2.InvoiceId
JOIN Track t2 ON il2.TrackId = t2.TrackId

WHERE t1.TrackId < t2.TrackId
GROUP BY t1.Name, t2.Name
ORDER BY TimesBoughtTogether DESC;
```



	Track1	Track2	TimesBoughtTogether
1	Lixo Do Mangue	A Cor Do Sol	2
2	Samba Do Lado	Amor De Muito	2
3	Comportamento Geral	Ando Meio Desligado	2
4	Battery	Better Than You	2
5	Plot 180	Big Wave	2
6	Untitled	Big Wave	2
7	Comida	Cabeça Dinossauro	2
8	For the Greater Good of God	Can I Play With Madness	2
9	Dirty Day	C'est La Vie	2
10	Overdose	Deuces Are Wild	2

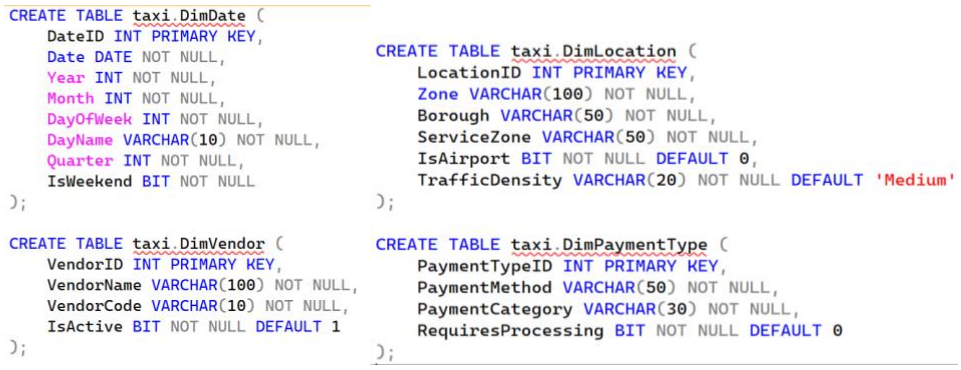
Appendix C: Data Warehouse Development Scripts

Fact Table Creation Script



```sql
CREATE TABLE taxi.FactTaxiTrip (
    TripID BIGINT IDENTITY(1,1) PRIMARY KEY,
    DateID INT NOT NULL,
    VendorID INT NOT NULL,
    PickupLocationID INT NOT NULL,
    DropoffLocationID INT NOT NULL,
    PaymentTypeID INT NOT NULL,

    PickupDateTime DATETIME2 NOT NULL,
    DropoffDateTime DATETIME2 NOT NULL,
    PickupHour AS (DATEPART(HOUR, PickupDateTime)) PERSISTED,

    TripDistance DECIMAL(8,2),
    TipAmount DECIMAL(10,2),
    TotalAmount DECIMAL(10,2),

    CONSTRAINT FK_Fact_Date FOREIGN KEY (DateID) REFERENCES taxi.DimDate(DateID),
    CONSTRAINT FK_Fact_Vendor FOREIGN KEY (VendorID) REFERENCES taxi.DimVendor(VendorID),
    CONSTRAINT FK_Fact_PickupLocation FOREIGN KEY (PickupLocationID) REFERENCES taxi.DimLocation(LocationID),
    CONSTRAINT FK_Fact_DropoffLocation FOREIGN KEY (DropoffLocationID) REFERENCES taxi.DimLocation(LocationID),
    CONSTRAINT FK_Fact_Payment FOREIGN KEY (PaymentTypeID) REFERENCES taxi.DimPaymentType(PaymentTypeID)
);
```

Dimension Table Creation Scripts



```sql
CREATE TABLE taxi.DimDate (
    DateID INT PRIMARY KEY,
    Date DATE NOT NULL,
    Year INT NOT NULL,
    Month INT NOT NULL,
    DayOfWeek INT NOT NULL,
    DayName VARCHAR(10) NOT NULL,
    Quarter INT NOT NULL,
    IsWeekend BIT NOT NULL
);
```

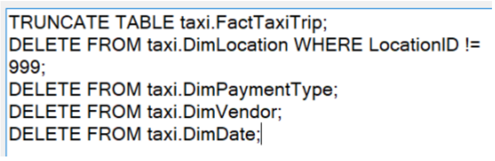
```sql
CREATE TABLE taxi.DimLocation (
    LocationID INT PRIMARY KEY,
    Zone VARCHAR(100) NOT NULL,
    Borough VARCHAR(50) NOT NULL,
    ServiceZone VARCHAR(50) NOT NULL,
    IsAirport BIT NOT NULL DEFAULT 0,
    TrafficDensity VARCHAR(20) NOT NULL DEFAULT 'Medium'
);
```

```sql
CREATE TABLE taxi.DimVendor (
    VendorID INT PRIMARY KEY,
    VendorName VARCHAR(100) NOT NULL,
    VendorCode VARCHAR(10) NOT NULL,
    IsActive BIT NOT NULL DEFAULT 1
);
```

```sql
CREATE TABLE taxi.DimPaymentType (
    PaymentTypeID INT PRIMARY KEY,
    PaymentMethod VARCHAR(50) NOT NULL,
    PaymentCategory VARCHAR(30) NOT NULL,
    RequiresProcessing BIT NOT NULL DEFAULT 0
);
```

Appendix D: SSIS ETL Implementation Details

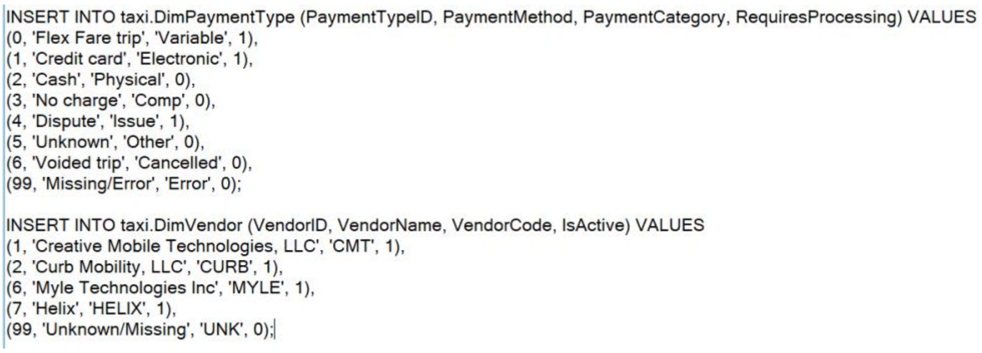
Execute SQL task (Clear all tables)



```sql
TRUNCATE TABLE taxi.FactTaxiTrip;
DELETE FROM taxi.DimLocation WHERE LocationID !=
999;
DELETE FROM taxi.DimPaymentType;
DELETE FROM taxi.DimVendor;
DELETE FROM taxi.DimDate;
```

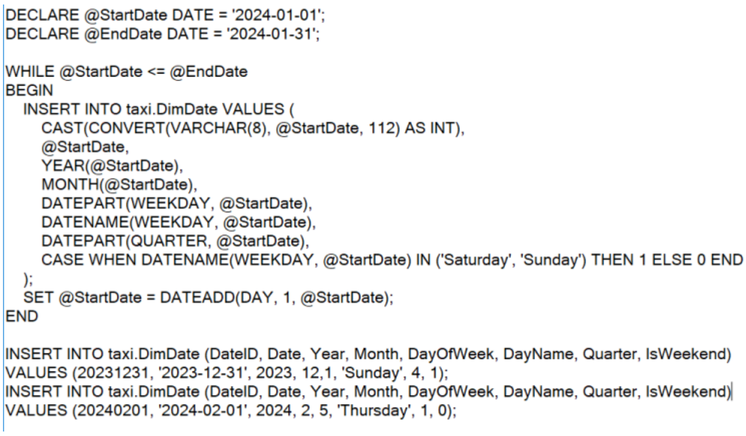
Execute SQL task (Populate payment and vendor tables)

All values are as provided by NYCTLC website



```sql
INSERT INTO taxi.DimPaymentType (PaymentTypeID, PaymentMethod, PaymentCategory, RequiresProcessing) VALUES
(0, 'Flex Fare trip', 'Variable', 1),
(1, 'Credit card', 'Electronic', 1),
(2, 'Cash', 'Physical', 0),
(3, 'No charge', 'Comp', 0),
(4, 'Dispute', 'Issue', 1),
(5, 'Unknown', 'Other', 0),
(6, 'Voided trip', 'Cancelled', 0),
(99, 'Missing/Error', 'Error', 0);

INSERT INTO taxi.DimVendor (VendorID, VendorName, VendorCode, IsActive) VALUES
(1, 'Creative Mobile Technologies, LLC', 'CMT', 1),
(2, 'Curb Mobility, LLC', 'CURB', 1),
(6, 'Myle Technologies Inc', 'MYLE', 1),
(7, 'Helix', 'HELIX', 1),
(99, 'Unknown/Missing', 'UNK', 0);
```

Execute SQL task (Populate Date table)



```
DECLARE @StartDate DATE = '2024-01-01';
DECLARE @EndDate DATE = '2024-01-31';

WHILE @StartDate <= @EndDate
BEGIN
   INSERT INTO taxi.DimDate VALUES (
      CAST(CONVERT(VARCHAR(8), @StartDate, 112) AS INT),
      @StartDate,
      YEAR(@StartDate),
      MONTH(@StartDate),
      DATEPART(WEEKDAY, @StartDate),
      DATENAME(WEEKDAY, @StartDate),
      DATEPART(QUARTER, @StartDate),
      CASE WHEN DATENAME(WEEKDAY, @StartDate) IN ('Saturday', 'Sunday') THEN 1 ELSE 0 END
   );
   SET @StartDate = DATEADD(DAY, 1, @StartDate);
END

INSERT INTO taxi.DimDate (DateID, Date, Year, Month, DayOfWeek, DayName, Quarter, IsWeekend)
VALUES (20231231, '2023-12-31', 2023, 12,1, 'Sunday', 4, 1);
INSERT INTO taxi.DimDate (DateID, Date, Year, Month, DayOfWeek, DayName, Quarter, IsWeekend)
VALUES (20240201, '2024-02-01', 2024, 2, 5, 'Thursday', 1, 0);
```

Load location Data Flow task

Flat File source – *taxi_zone_lookup.csv (Lookup_zone_csv)*

Derived Column (CleanBorough new column) - *REPLACE(["Borough"],"\"","")*

Derived Column (IsAirport, TrafficDensity)

- IsAirport (new column) => *FINDSTRING(["Zone"],"Airport",1) > 0 ? (DT_BOOL)1 : (DT_BOOL)0*
- Traffic Density (new column) => *(DT_STR,20,1252)(CleanBorough == "Manhattan" ? "High" : (CleanBorough == "Brooklyn" || CleanBorough == "Queens" || CleanBorough == "Bronx" ? "Medium" : "Low"))*

Load FactTable Data Flow task

Flat File source – *yellow-tripdata-2024-01_CLEANED.csv (trip_data_cleaned_csv)*

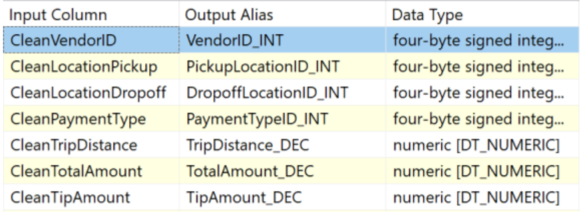
Derived Column

- DateID (new column) => *(DT_I4)REPLACE(SUBSTRING(["tpep_pickup_datetime"],1,10),"-","")*
- PickupDateTime (new column) => *(DT_DBTIMESTAMP2,0)["tpep_pickup_datetime"]*
- DropOffDateTime (new column) => *(DT_DBTIMESTAMP2,0)["tpep_dropoff_datetime"]*
- CleanVendorID(new column) => *["VendorID"] == "\N" ? "99" : ["VendorID"]*
- CleanPaymentType (new column) => *["payment_type"] == "\N" ? "99" : ["payment_type"]*
- CleanLocationPickup (new column) => *["PULocationID"] == "\N" ? "999" : ["PULocationID"]*
- CleanLocationDropoff (new column) => *["DOLocationID"] == "\N" ? "999" : ["DOLocationID"]*
- CleanTripDistance (new column) => *["trip_distance"] == "\N" ? "0" : ["trip_distance"]*
- CleanTipAmount (new column) => *["tip_distance"] == "\N" ? "0" : ["tip_distance"]*
- CleanTotalAmount (new column) => *["total_distance"] == "\N" ? "0" : ["total_distance"]*

Conditional Split – this makes sure that only data between 31st December and 1st Feb is loaded

DateID >= 20231231 && DateID <= 20240201

Data Conversion - this step is to make sure all data types match what was defined earlier while setting up tables in SSMS

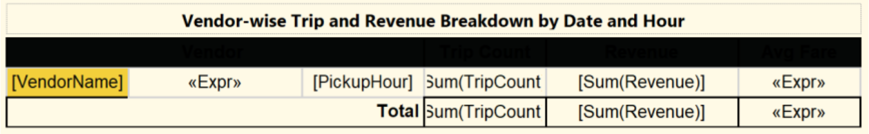


Input Column	Output Alias	Data Type
CleanVendorID	VendorID_INT	four-byte signed integ...
CleanLocationPickup	PickupLocationID_INT	four-byte signed integ...
CleanLocationDropoff	DropoffLocationID_INT	four-byte signed integ...
CleanPaymentType	PaymentTypeID_INT	four-byte signed integ...
CleanTripDistance	TripDistance_DEC	numeric [DT_NUMERIC]
CleanTotalAmount	TotalAmount_DEC	numeric [DT_NUMERIC]
CleanTipAmount	TipAmount_DEC	numeric [DT_NUMERIC]

Appendix E: SSRS Reports

This section will contain all “design view” screenshots for all generated SSRS reports.

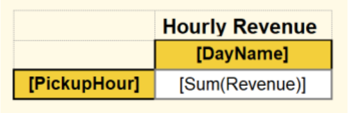
Drill-Down Report – Grouping done by Vendor name -> Date -> Hour of day



Vendor-wise Trip and Revenue Breakdown by Date and Hour

Vendor			Trip Count	Revenue	Avg Fare
[VendorName]	«Expr»	[PickupHour]	Sum(TripCount	[Sum(Revenue)]	«Expr»
		Total	Sum(TripCount	[Sum(Revenue)]	«Expr»

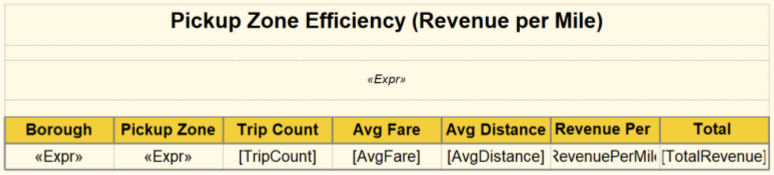
Matrix Report



	Hourly Revenue
	[DayName]
[PickupHour]	[Sum(Revenue)]

Parameterized Report –

3 parameters available, start date, end date and borough which also dynamically changes the “subtitle”

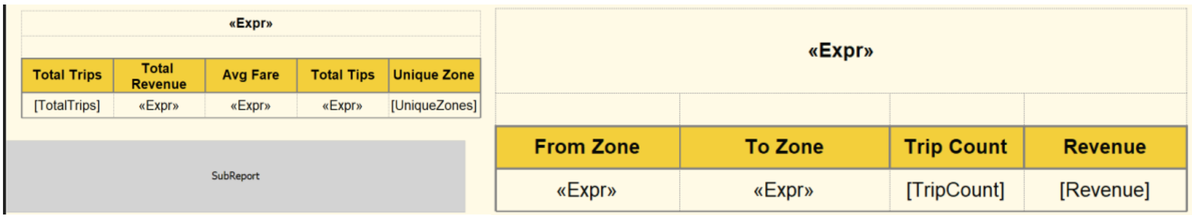


Pickup Zone Efficiency (Revenue per Mile)

«Expr»

Borough	Pickup Zone	Trip Count	Avg Fare	Avg Distance	Revenue Per	Total
«Expr»	«Expr»	[TripCount]	[AvgFare]	[AvgDistance]	RevenuePerMil	[TotalRevenue]

Report with Embedded SubReport – both reports have a date parameter available



«Expr»

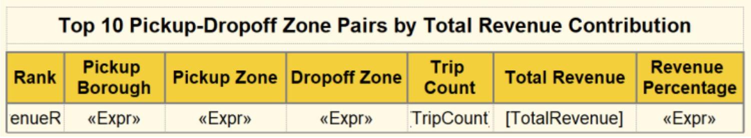
Total Trips	Total Revenue	Avg Fare	Total Tips	Unique Zone
[TotalTrips]	«Expr»	«Expr»	«Expr»	[UniqueZones]

SubReport

«Expr»

From Zone	To Zone	Trip Count	Revenue
«Expr»	«Expr»	[TripCount]	[Revenue]

Basic Table



Top 10 Pickup-Dropoff Zone Pairs by Total Revenue Contribution

Rank	Pickup Borough	Pickup Zone	Dropoff Zone	Trip Count	Total Revenue	Revenue Percentage
enueR	«Expr»	«Expr»	«Expr»	TripCount	[TotalRevenue]	«Expr»