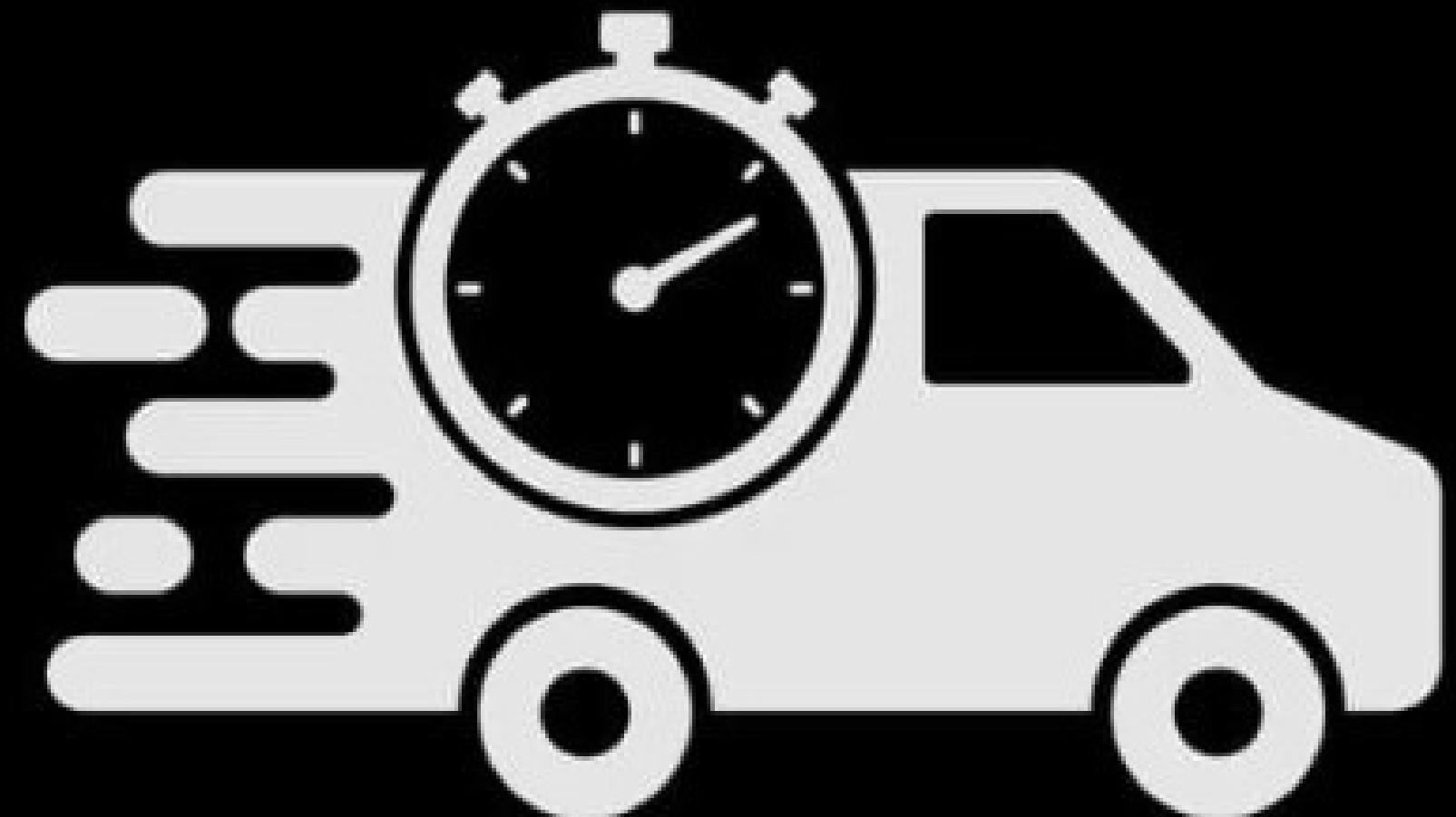
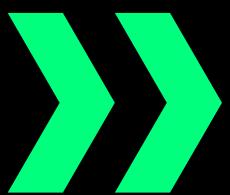
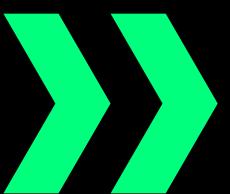


YOUR PACKAGE, OUR PRIORITY 

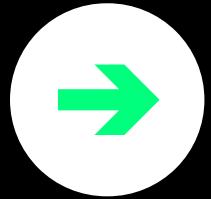
Express Delivery

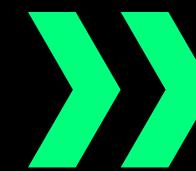


Download on the
App Store



GET IT ON
Google Play



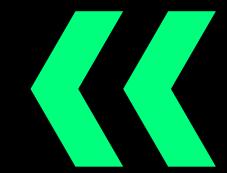


Express Delivery is a fast and reliable delivery company that connects buyers and sellers across the USA through an easy-to-use app. Catering to both individuals and businesses, it offers seamless, secure, and efficient delivery services, ensuring packages are handled quickly and with confidence.



Entering 2024, the company aims to analyze key data from 2023 to drive improvements and refine business strategies.

- Total sales and revenue

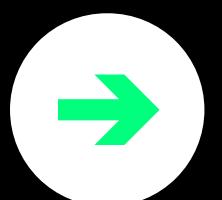


- Monthly delivery volume

- Customer growth and the most active states

- Most popular payment methods

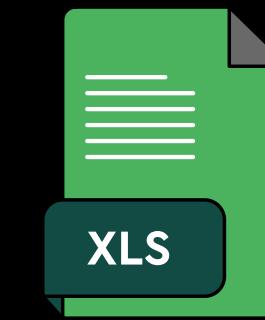
- Top drivers of the year



Company Dataset



DELIVERIES DATA



SHIPMENTS DATA



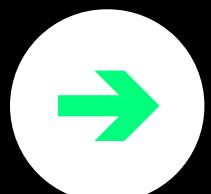
DRIVERS DATA



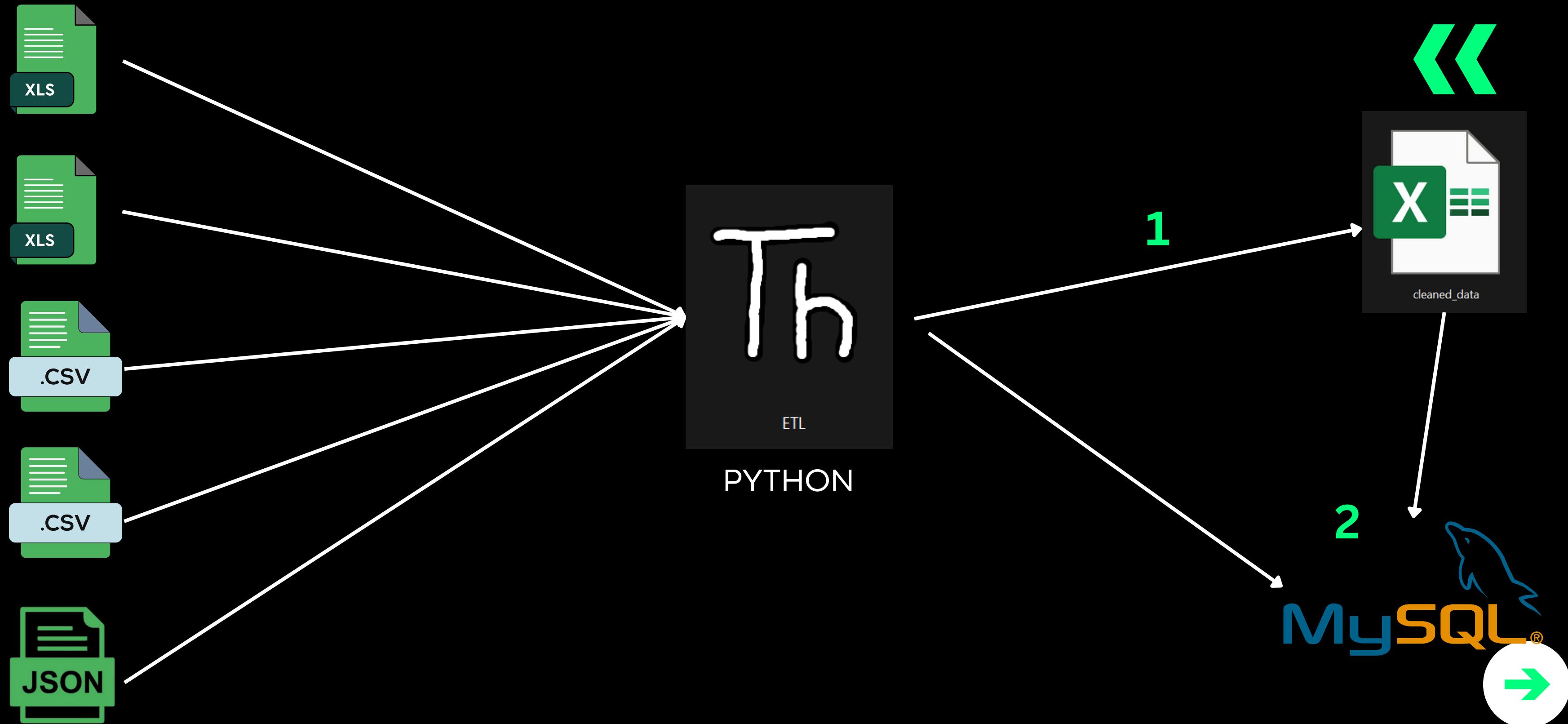
PAYMENTS DATA



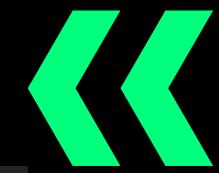
CUSTOMERS DATA



ETL Process



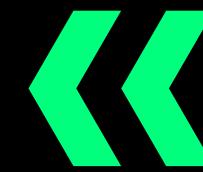
ETL Python



```
ETL.py x
1 import pandas as pd
2 import json
3
4 # Paths to input files
5 csv_drivers_path = "C:\\\\Users\\\\dell\\\\Desktop\\\\data cleaning\\\\drivers.csv"
6 csv_payments_path = "C:\\\\Users\\\\dell\\\\Desktop\\\\data cleaning\\\\payments.csv"
7 excel_deliveries_path = "C:\\\\Users\\\\dell\\\\Desktop\\\\data cleaning\\\\deliveries.xlsx"
8 excel_shipments_path = "C:\\\\Users\\\\dell\\\\Desktop\\\\data cleaning\\\\shipments.xlsx"
9 json_customers_path = "C:\\\\Users\\\\dell\\\\Desktop\\\\data cleaning\\\\customers.json"
10
11 # Path for the output Excel file
12 output_excel_path = "C:\\\\Users\\\\dell\\\\Desktop\\\\data cleaning\\\\etl data\\\\cleaned_data.xlsx"
13
14 # Load data from files
15 drivers_df = pd.read_csv(csv_drivers_path)
16 payments_df = pd.read_csv(csv_payments_path)
17 deliveries_df = pd.read_excel(excel_deliveries_path)
18 shipments_df = pd.read_excel(excel_shipments_path)
19
20 with open(json_customers_path, 'r') as f:
21     customers_data = json.load(f)
22 customers_df = pd.DataFrame(customers_data)
```



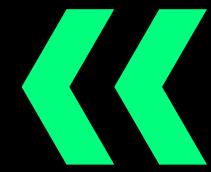
```
ETL.py x
23
24 # Cleaning Drivers Table
25 def clean_drivers(df):
26     def extract_name_from_email(row):
27         if pd.isna(row['first_name']) or pd.isna(row['last_name']):
28             first_name, last_name = row['email'].split('@')[0].split('.')
29             return pd.Series({'first_name': first_name, 'last_name': last_name})
30         return pd.Series({'first_name': row['first_name'], 'last_name': row['last_name']})
31
32 missing_names = df[df['first_name'].isna() | df['last_name'].isna()]
33 df.update(missing_names.apply(extract_name_from_email, axis=1))
34 return df
35
36 drivers_df = clean_drivers(drivers_df)
37
38 # Cleaning Payments Table
39 def clean_payments(payments_df, deliveries_df, shipments_df):
40     def determine_payment_method(row):
41         if pd.isna(row['payment_method']):
42             delivery_date = deliveries_df.loc[deliveries_df['delivery_id'] == row['delivery_id'], 'delivery_date']
43             if not delivery_date.empty and delivery_date.iloc[0] == row['payment_date']:
44                 return "cash"
45             return "paypal/credit card"
46         return row['payment_method']
47
48     def calculate_payment_amount(row):
49         if pd.isna(row['payment_amount']):
50             delivery_cost = deliveries_df.loc[deliveries_df['delivery_id'] == row['delivery_id'], 'delivery_cost'].fillna(0)
51             item_price = shipments_df.loc[shipments_df['shipment_id'] == row['shipment_id'], 'item_price'].fillna(0)
52             return delivery_cost.iloc[0] + item_price.iloc[0] if not delivery_cost.empty and not item_price.empty else
53             return row['payment_amount']
54
55     payments_df['payment_method'] = payments_df.apply(determine_payment_method, axis=1)
56     payments_df['payment_amount'] = payments_df.apply(calculate_payment_amount, axis=1)
57     return payments_df
58
59 payments_df = clean_payments(payments_df, deliveries_df, shipments_df)
```





```
58
59
60
61 # Cleaning Deliveries Table
62 def clean_deliveries(deliveries_df, payments_df, shipments_df):
63     def calculate_delivery_cost(row):
64         if pd.isna(row['delivery_cost']):
65             payment_amount = payments_df.loc[payments_df['delivery_id'] == row['delivery_id'], 'payment_amount'].fillna(0)
66             item_price = shipments_df.loc[shipments_df['delivery_id'] == row['delivery_id'], 'item_price'].fillna(0)
67             return payment_amount.iloc[0] - item_price.iloc[0] if not payment_amount.empty and not item_price.empty else 0
68         return row['delivery_cost']
69
70     deliveries_df['delivery_cost'] = deliveries_df.apply(calculate_delivery_cost, axis=1)
71     return deliveries_df
72
73 deliveries_df = clean_deliveries(deliveries_df, payments_df, shipments_df)
74
75 # Cleaning Shipments Table
76 def clean_shipments(shipments_df, payments_df):
77     def calculate_item_price(row):
78         if pd.isna(row['item_price']):
79             payment_amount = payments_df.loc[payments_df['shipment_id'] == row['shipment_id'], 'payment_amount'].fillna(0)
80             other_prices = shipments_df.loc[(shipments_df['delivery_id'] == row['delivery_id']) & (shipments_df['shipment_id'] != row['shipment_id']), 'item_price'].sum()
81             return payment_amount.iloc[0] - other_prices.sum() if not payment_amount.empty else 0
82         return row['item_price']
83
84     shipments_df['item_price'] = shipments_df.apply(calculate_item_price, axis=1)
85     return shipments_df
86
87 shipments_df = clean_shipments(shipments_df, payments_df)
88
89 # Cleaning Customers Table
```





```
91  
92 # Writing cleaned data to an Excel file  
93 with pd.ExcelWriter(output_excel_path, engine='openpyxl') as writer:  
94     customers_df.to_excel(writer, index=False, sheet_name='Customers')  
95     drivers_df.to_excel(writer, index=False, sheet_name='Drivers')  
96     payments_df.to_excel(writer, index=False, sheet_name='Payments')  
97     shipments_df.to_excel(writer, index=False, sheet_name='Shipments')  
98     deliveries_df.to_excel(writer, index=False, sheet_name='Deliveries')  
99  
100 print(f"Cleaned data has been saved to {output_excel_path}")  
101
```

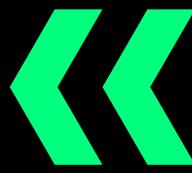
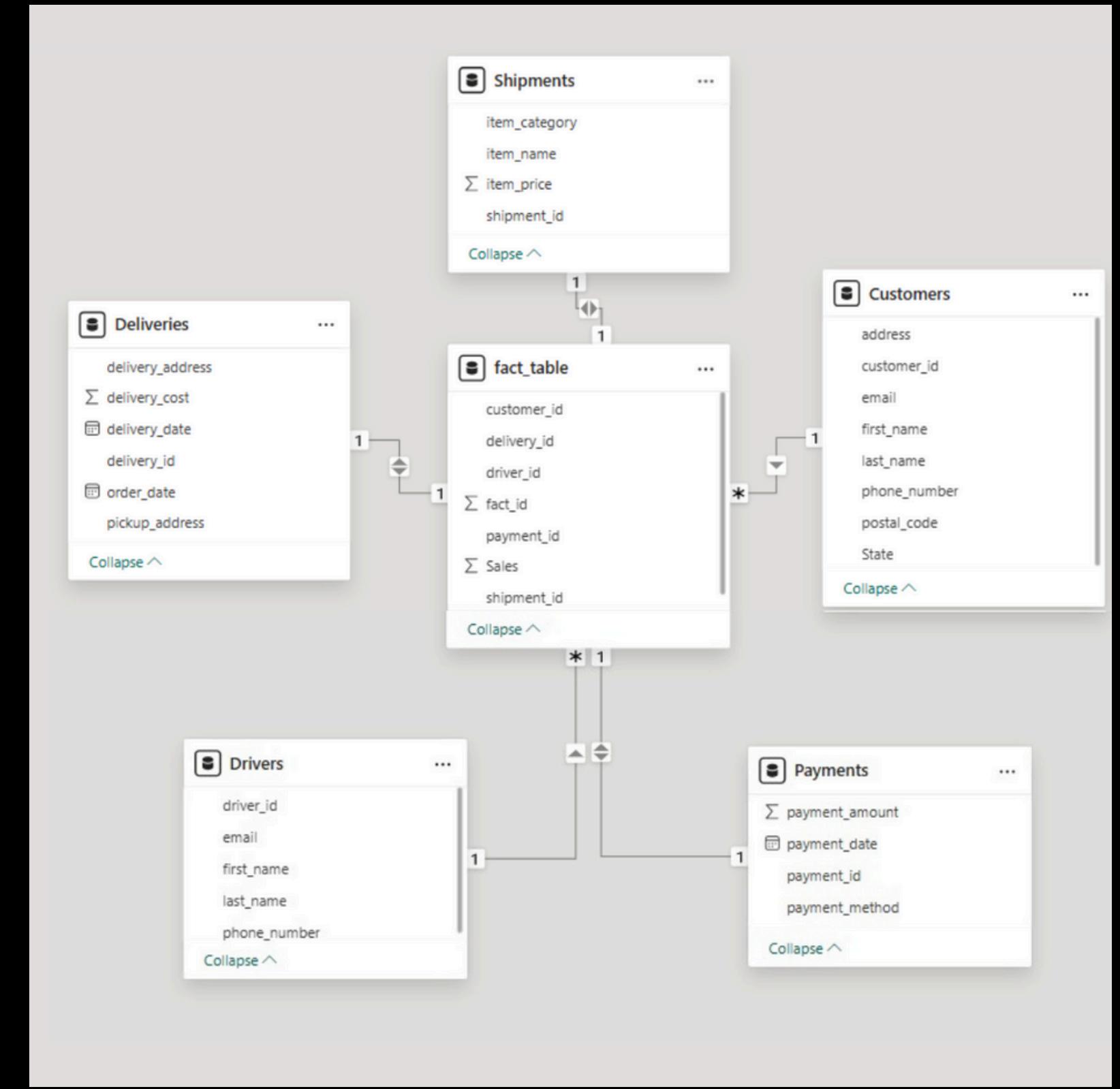
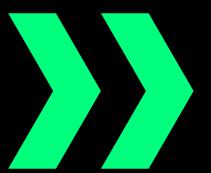
ETL 1 Output

14	Franklin	Smith	franklin.smith@example.com
15	Amy	Adkins	amy.adkins@example.com
16	Christine	Barnes	christine.barnes@example.com
17	Natasha	Wood	natasha.wood@example.com
18	Grace	Farrell	grace.farrell@example.com
19	Daniel	Landry	daniel.landry@example.com

Customers | Drivers | Payments | Shipments | Deliveries | +



Star Schema



Data Loading

SQL ROLAP

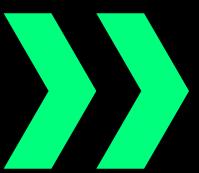
```
1 import pandas as pd
2 import mysql.connector
3
4 # Database connection setup
5 def create_connection(database=None):
6     if database:
7         return mysql.connector.connect(
8             host="localhost",
9             user="root",
10            password="islemesmiila*332-W",
11            database=database
12        )
13    else:
14        return mysql.connector.connect(
15            host="localhost",
16            user="root",
17            password="islemesmiila*332-W"
18        )
19
20 # Create database and tables
21 def setup_database():
22     conn = create_connection()
23     cursor = conn.cursor()
24
25     # Create database if it doesn't exist
26     cursor.execute("CREATE DATABASE IF NOT EXISTS deliveryexpress")
```



Data Loading

SQL ROLAP

```
loading data.py x
27     conn.commit() # Commit the creation of the database
28
29 # Now connect to the specific database
30 conn = create_connection("deliveryexpress")
31 cursor = conn.cursor()
32
33 # Create tables
34 tables = {
35     "Customers": """
36         CREATE TABLE IF NOT EXISTS Customers (
37             customer_id INT PRIMARY KEY,
38             first_name VARCHAR(255),
39             last_name VARCHAR(255),
40             email VARCHAR(255),
41             phone_number VARCHAR(20),
42             address VARCHAR(255),
43             postal_code VARCHAR(20),
44             state VARCHAR(50)
45         )
46     """,
47     "Payments": """
48         CREATE TABLE IF NOT EXISTS Payments (
49             payment_id INT PRIMARY KEY,
50             payment_amount DECIMAL(10, 2),
51             payment_method VARCHAR(50)
52         )
53     """
54 }
55
56 conn.close()
```



Data Loading

SQL ROLAP

loading data.py ×

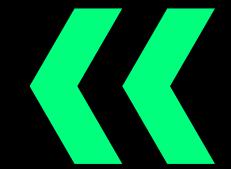
```
53     """
54     "Drivers": """
55         CREATE TABLE IF NOT EXISTS Drivers (
56             driver_id INT PRIMARY KEY,
57             first_name VARCHAR(255),
58             last_name VARCHAR(255),
59             email VARCHAR(255),
60             phone_number VARCHAR(20)
61         )
62     """,
63     "Shipments": """
64         CREATE TABLE IF NOT EXISTS Shipments (
65             shipment_id INT PRIMARY KEY,
66             item_category VARCHAR(50),
67             item_name VARCHAR(255),
68             item_price DECIMAL(10, 2)
69         )
70     """,
71     "Deliveries": """
72         CREATE TABLE IF NOT EXISTS Deliveries (
73             delivery_id INT PRIMARY KEY,
74             delivery_address VARCHAR(255),
75             pickup_address VARCHAR(255),
76             delivery_cost DECIMAL(10, 2)
77         )
78     """,
```



Data Loading

SQL ROLAP

```
loading data.py x
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103 # Insert data from Excel to MySQL
```



Data Loading



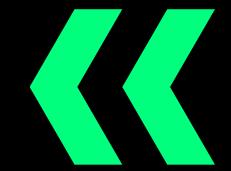
SQL ROLAP

```
loading data.py x
103 # Insert data from Excel to MySQL
104 def insert_data_from_excel(file_path):
105     conn = create_connection("deliveryexpress") # Connect to the 'deliveryexpress' database
106     cursor = conn.cursor()
107
108     # Read each sheet and insert data
109     sheets = ["Customers", "Payments", "Drivers", "Shipments", "Deliveries"]
110     for sheet in sheets:
111         df = pd.read_excel(file_path, sheet_name=sheet)
112
113         # Filter columns that match table structure
114         cursor.execute(f"DESCRIBE {sheet}")
115         table_columns = [row[0] for row in cursor.fetchall()]
116         df = df[[col for col in df.columns if col in table_columns]]
117
118         # Insert data into the table
119         for _, row in df.iterrows():
120             placeholders = ", ".join(["%s"] * len(row))
121             columns = ", ".join(row.index)
122             sql = f"INSERT INTO {sheet} ({columns}) VALUES ({placeholders})"
123             cursor.execute(sql, tuple(row))
124
125         conn.commit()
126         conn.close()
127
128 # Populate fact table
```



Data Loading

SQL ROLAP



```
127  
128 # Populate fact table  
129 def populate_fact_table():  
130     conn = create_connection("deliveryexpress") # Connect to the 'deliveryexpress' database  
131     cursor = conn.cursor()  
132  
133     # Get the maximum number of rows across dimensions  
134     cursor.execute("SELECT COUNT(*) FROM Customers")  
135     customer_count = cursor.fetchone()[0]  
136  
137     cursor.execute("SELECT COUNT(*) FROM Payments")  
138     payment_count = cursor.fetchone()[0]  
139  
140     cursor.execute("SELECT COUNT(*) FROM Drivers")  
141     driver_count = cursor.fetchone()[0]  
142  
143     cursor.execute("SELECT COUNT(*) FROM Shipments")  
144     shipment_count = cursor.fetchone()[0]  
145  
146     cursor.execute("SELECT COUNT(*) FROM Deliveries")  
147     delivery_count = cursor.fetchone()[0]  
148  
149     max_rows = max(customer_count, payment_count, driver_count, shipment_count, delivery_count)  
150
```

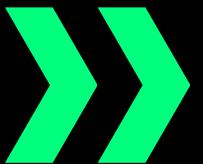


Data Loading

SQL ROLAP



```
loading data.py * x
151     # Insert rows into fact_table
152     for i in range(max_rows):
153         cursor.execute("""
154             INSERT INTO fact_table (customer_id, payment_id, driver_id, shipment_id, delivery_id, Sales)
155             VALUES (
156                 IFNULL((SELECT customer_id FROM Customers LIMIT %s, 1), 0),
157                 IFNULL((SELECT payment_id FROM Payments LIMIT %s, 1), 0),
158                 IFNULL((SELECT driver_id FROM Drivers LIMIT %s, 1), 0),
159                 IFNULL((SELECT shipment_id FROM Shipments LIMIT %s, 1), 0),
160                 IFNULL((SELECT delivery_id FROM Deliveries LIMIT %s, 1), 0),
161                 IFNULL((SELECT delivery_cost FROM Deliveries LIMIT %s, 1), 0)
162             )
163             """ , (i, i, i, i, i, i))
164
165     conn.commit()
166     conn.close()
167
168 # Main function
169 def main():
170     file_path = "C:\\\\Users\\\\dell\\\\Desktop\\\\data cleaning\\\\etl data\\\\cleaned_data.xlsx"
171
172     setup_database()
173     insert_data_from_excel(file_path)
174     populate_fact_table()
175
176 if __name__ == "__main__":
177     main()
```



Data Loading

OUTPUT

The screenshot illustrates the results of data loading into a database named **deliveryexpress**. It shows two distinct data sets:

- Customer Data (Top Grid):** This grid displays 22 rows of customer information, including customer ID, first name, last name, email, phone number, address, state, and postal code.
- Delivery Data (Bottom Grid):** This grid displays 20 rows of delivery details, including delivery ID, delivery address, pickup address, and delivery cost.

Customer Data (Top Grid):

customer_id	first_name	last_name	email	phone_number	address	state	postal_code
97	David	Powell	david.powell@example.com	+1664087343	578 Kimberly Turn...	Iowa	76659
193	Rebecca	Fields	rebecca.fields@example.com	+1912906014	8469 Gonzalez Spri...	Virginia	45624
436	Denise	Curry	denise.curry@example.com	+1229700073	67291 Montgomer...	Wyoming	51921
511	Adam	Lane	adam.lane@example.com	+1362735441	656 Simpson Well ...	Indiana	03476
528	Sara	Duncan	sara.duncan@example.com	+1155020303	78502 Nathan Driv...	Utah	82646
569	Keith	Stanton	keith.stanton@example.com	+1164612445	USCGC Nelson, FP...	Missouri	46562
589	Dalton	Osborn	dalton.osborn@example.com	+1690012837	970 Natalie Shoals ...	Arkansas	01518
933	Christina	Lopez	christina.lopez@example.com	+1953802075	3088 Mark Brooks ...	Arkansas	45807
1019	Jason	Allen	jason.allen@example.com	+1484105725	11154 Harvey Stre...	Rhode Island	96892
1149	William	Price	william.price@example.com	+1937768549	1210 Jennifer Har...	Indiana	89342
1363	Andrew	Morgan	andrew.morgan@example.com	+1588092694	504 Allen Well Apt...	Georgia	93058
1504	Albert	Clark	albert.dark@example.com	+1372987211	06166 Carpenter ...	Tennessee	12950
2031	Jeffery	Rhodes	jeffery.rhodes@example.com	+1110423086	1871 Mann Turnpike...	North Carolina	01781
2151	Gina	Sanchez	gina.sanchez@example.com	+1565608900	844 Willis Ridge Ap...	Louisiana	93221
2187	Isaac	Singh	isaac.singh@example.com	+1231348586	67949 McCullough ...	North Carolina	83923
2235	Travis	Davis	travis.davis@example.com	+1224206458	82341 Smith Spur, ...	Ohio	46841

Delivery Data (Bottom Grid):

delivery_id	delivery_address	pickup_address	delivery_cost
261	0920 Daniel Plains, Matthewfurt, CO 64705	186 Thompson Way Suite 756, New Anita, PW ...	39.26
659	37820 Hannah Manor, North Douglas, OR 03087	43840 Katie Mission Apt. 573, Ricetown, SD 58...	42.96
710	3365 Torres Gardens Apt. 249, North Rachelshi...	31867 Wendy Neck, Bethview, KY 97732	36.38
784	21977 Chan Port, Lake Whitney, SD 07110	Unit 1692 Box 7592, DPO AA 41071	35.54
817	58454 Traci Trail, South Kayla, KY 24852	793 Morgan Trail, Lake Shawn, DC 63181	50.66
826	511 James Mount Suite 497, Benjaminport, NC ...	575 Schmidt Lodge, New David, GU 04780	51.22
840	PSC 3737, Box 6989, APO AA 56478	60670 Benitez Crossroad, Anthonyberg, OH 28...	45.13
1268	273 Rivera Spur Apt. 090, Lloydland, IA 85625	2731 Kathy Mill, Bellchester, NM 24839	46.67
1287	05726 Linda Village, East Lori, WI 18052	078 May Hollow Suite 364, Joneston, GA 58120	53.01
1742	53900 Calhoun Station, Oconnellfort, VT 93254	883 Smith Parks Suite 581, Walkertown, MO 30...	65.84
1813	40430 King Vista, Bennettberg, IL 95296	817 Anthony Gardens, Theodorefurt, MI 55799	50.75
2117	799 Emily Tunnel Apt. 707, South Amandamout...	Unit 8137 Box 6742, DPO AP 56053	54.51
2780	6278 Chad Crossroad, Port Joshua, RI 89986	32741 Scott Glen Apt. 255, Port Derek, CA 82251	66.16
2880	397 Rose Skyway Suite 900, South Emilyton, S...	95145 Rachel Valleys, West Ericahaven, FM 84...	44.40
3508	1182 Shari Port, North Luisberg, SD 39544	72444 Cummings Crest Apt. 269, Shariport, SD ...	55.03



Data Loading

OUTPUT



	driver_id	first_name	last_name	email	phone_number
▶	1256	Candace	Andersen	candace.andersen@example.com	1728353036
	1612	Joseph	Harrison	joseph.harrison@example.com	1746465246
	2734	Christina	Larsen	christina.larsen@example.com	1934005483
	3125	Courtney	Haas	courtney.haas@example.com	1486514103
	3576	Meredith	Harris	meredith.harris@example.com	1957019839
	4150	Zachary	Roy	zachary.roy@example.com	1227156752
	5188	Jessica	Hall	jessica.hall@example.com	1981693562
	6023	Howard	Cummings	howard.cummings@example.com	1807170618
	6028	Shawn	Mata	shawn.mata@example.com	1265193450
	6128	Margaret	Best	margaret.best@example.com	1591512275
	6999	Julie	Moore	julie.moore@example.com	1634171577
	7555	Amanda	Garrett	amanda.garrett@example.com	1539587176
	7683	Jacob	Miller	jacob.miller@example.com	1250862121
	7831	Pam	Gonzalez	pam.gonzalez@example.com	1888763590
	8045	Justin	Ballard	justin.ballard@example.com	1793231627

	payment_id	payment_amount	payment_method
▶	490	115.28	paypal
	491	287.04	paypal
	737	163.66	credit card
	785	248.86	paypal
	856	552.08	cash
	1107	74.62	cash
	1451	158.79	paypal
	1471	960.19	cash
	1995	75.44	paypal
	2093	137.69	paypal
	2620	192.94	cash
	2663	247.94	credit card
	2979	852.36	cash
	2992	574.28	credit card
	3362	318.90	cash



Data Loading

OUTPUT

The screenshot shows a data loading interface with two tables displayed side-by-side.

Left Table (dim_table):

	shipment_id	item_category	item_name	item_price
▶	131	electronics	Tablet	623.61
	173	clothes	Sweater	98.86
	350	furniture	Wardrobe	343.55
	550	book and education	E-Reader	68.78
	600	sports and outdoors	Backpack	117.86
	1150	book and education	Textbook	166.48
	1214	electronics	Laptop	391.91
	1350	book and education	E-Reader	46.86
	1731	beauty and personal care	Perfume	160.52
	2078	book and education	Novel	70.72
	2118	clothes	Jacket	174.51
	2444	furniture	Sofa	488.74
	2847	beauty and personal care	Perfume	71.79
	3120	beauty and personal care	Makeup Kit	193.98

Right Table (fact_table):

fact_id	customer_id	payment_id	driver_id	shipment_id	delivery_id	sales
1	288389	261600	509476	757779	64672	64.42
2	231148	557068	387622	890889	961228	60.50
3	396922	802730	814571	571196	702431	63.20
4	716751	656274	114898	510013	966528	60.84
5	69403	257584	182317	605270	944022	62.27
6	449245	324561	740042	669079	141173	34.24
7	988210	395517	189907	765584	592216	43.76
8	669987	368008	316408	861013	408096	41.54
9	509597	331567	408950	101551	774301	52.27
10	259947	468930	803463	996335	705078	63.75
11	224643	327717	214757	985755	514922	36.63
12	292075	169822	374635	176091	436644	64.11
13	409386	294523	487925	168514	638267	53.15
14	59942	706358	639694	875588	986744	31.74
15	612024	167751	855937	533904	152663	67.21
16	165409	890425	98754	87756	574910	32.61
17	529959	730697	443581	785606	912475	62.87



Dashboard



DELIVERY EXPRESS OVERVIEW DASHBOARD

4000

Total Number of customers

249.36K

Total Revenue in \$

2600

Number of Drivers

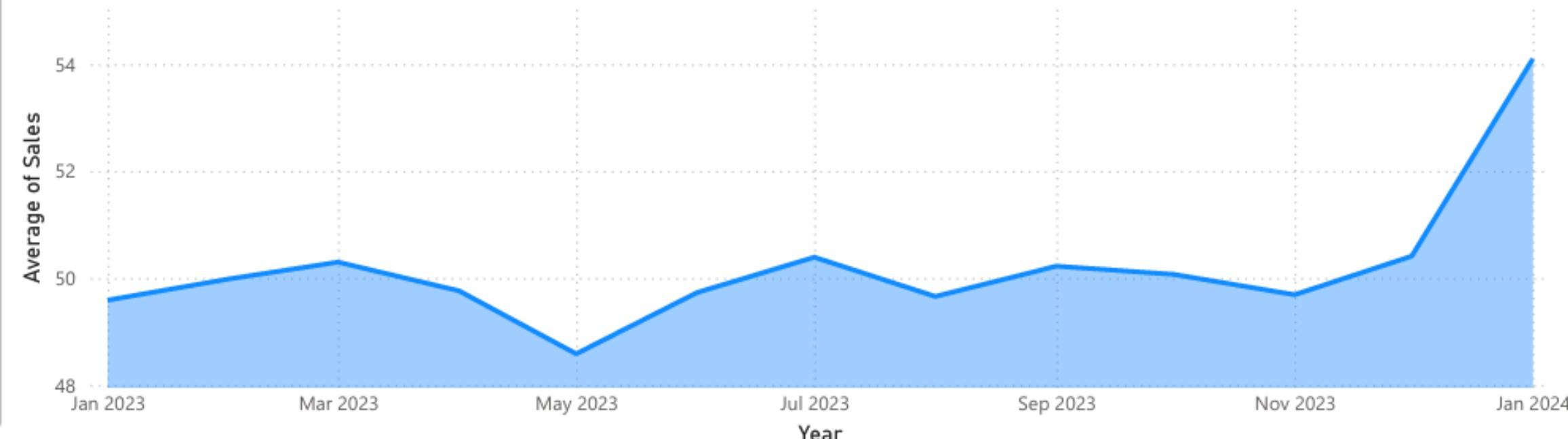
5000

Total Number of Deliveries

1.58MTotal Amount of Payments
made by customers

Year	Month	Number of Deliveries
2023	January	368
2023	February	378
2023	March	464
2023	April	402
2023	May	457
2023	June	390
2023	July	435
2023	August	407
2023	September	414
2023	October	430
2023	November	415
2023	December	434
Total		5000

Average of Sales by Year, Quarter and Month



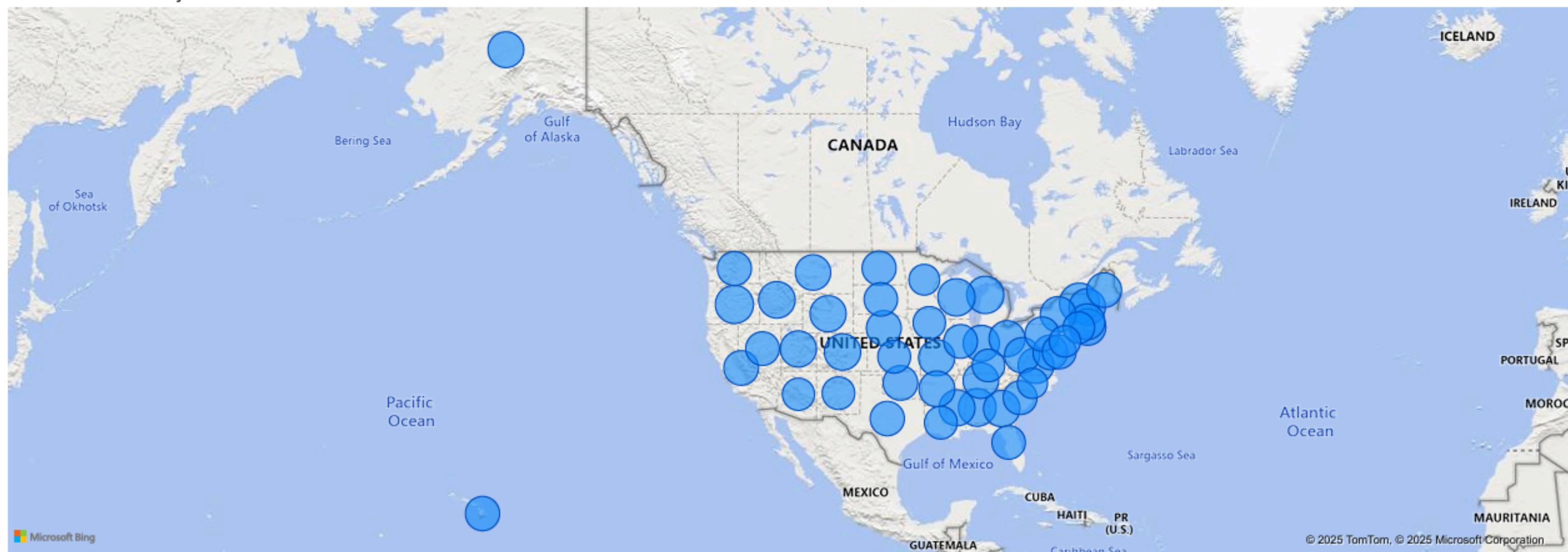
4000

Total Number of customers

TOTAL NUMBER OF CUSTOMERS PER STATE

Alabama
State selected

% of Customers by State



5000

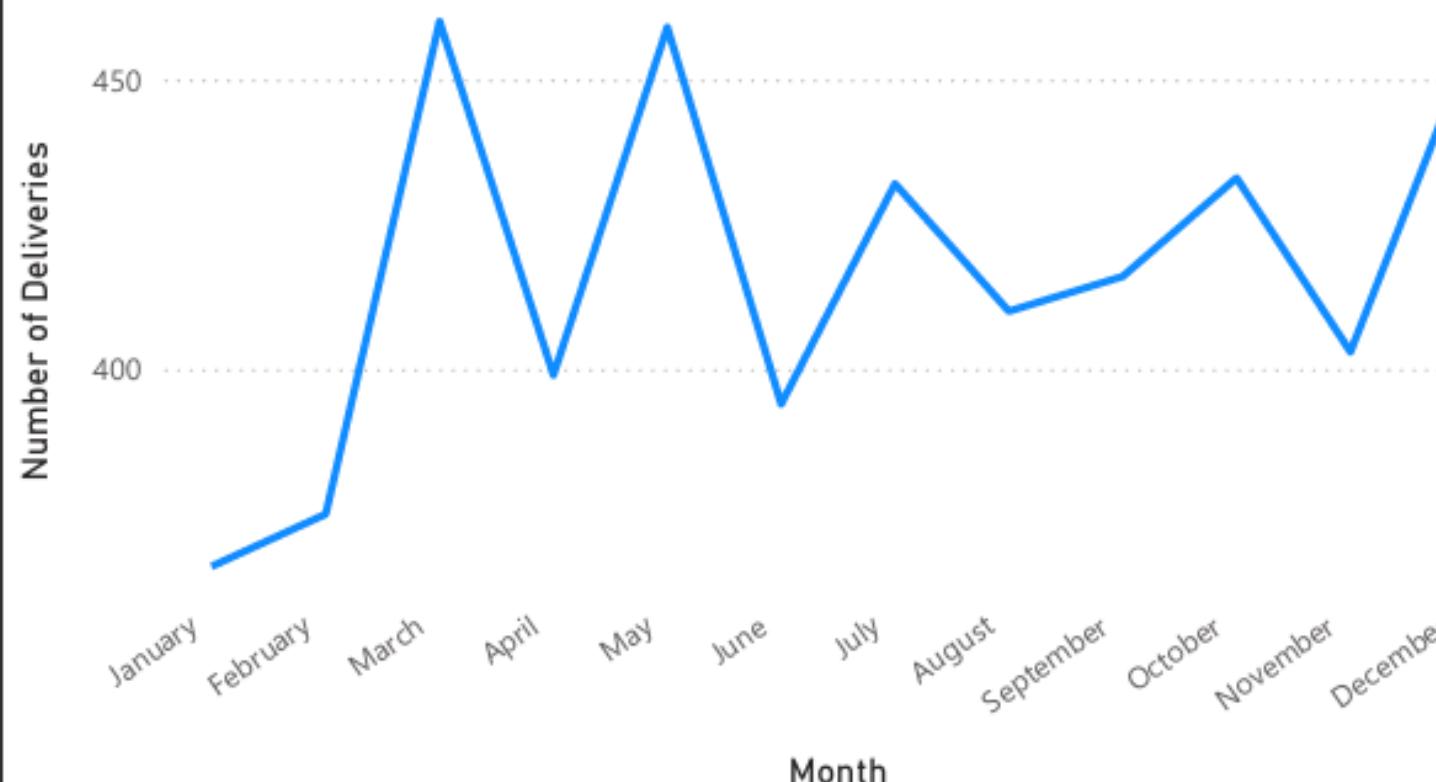
Total Number of deliveries

DELIVERIES ANALYSIS:

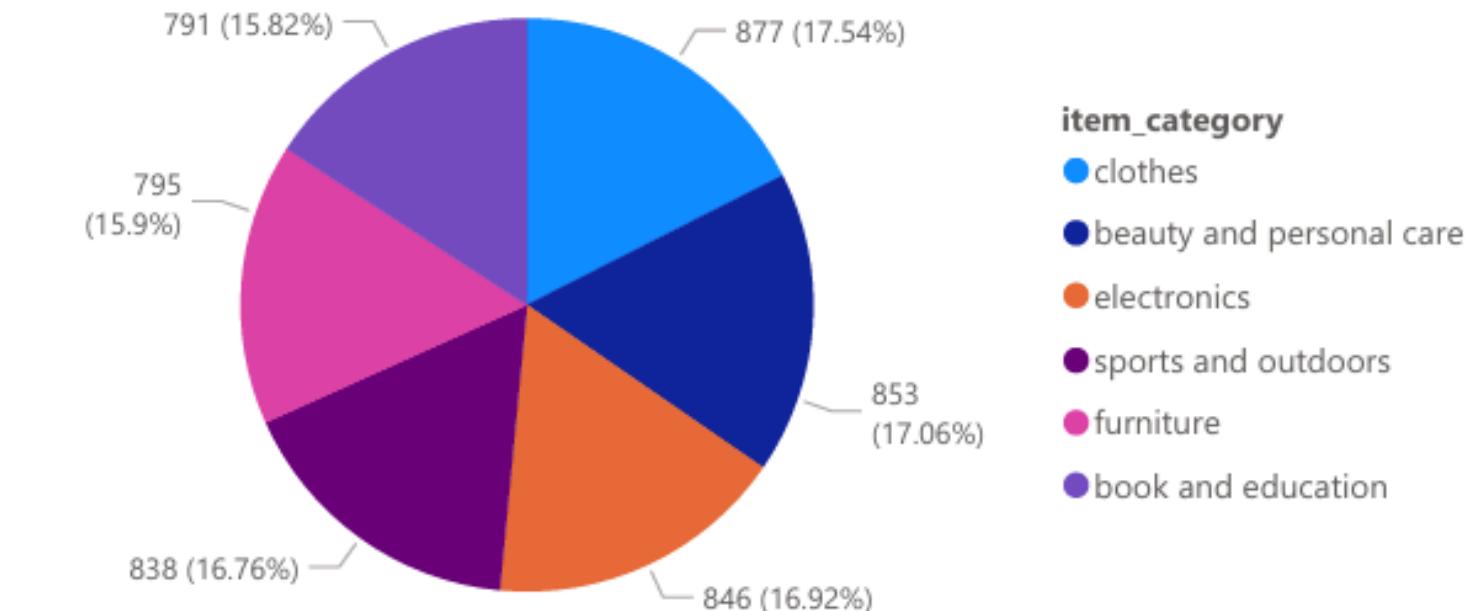
- **March** is the month with the highest number of deliveries performed by **DELIVERY EXPRESS**.
- The item category with the highest number of deliveries performed is "clothes", they represent **17.54%** of the total number of Deliveries.
- This Analysis is conducted to measure the performance of the business from **JANUARY 2023 to JANUARY 2024**.

Year	Month	Number of Deliveries
2023	January	345
2023	February	375
2023	March	460
2023	April	399
2023	May	459
2023	June	394
2023	July	432
2023	August	410
2023	September	416
2023	October	433
2023	November	403
2023	December	453
2024	January	21
Total		5000

Number of Deliveries by Month



Total Number of Deliveries by item_category

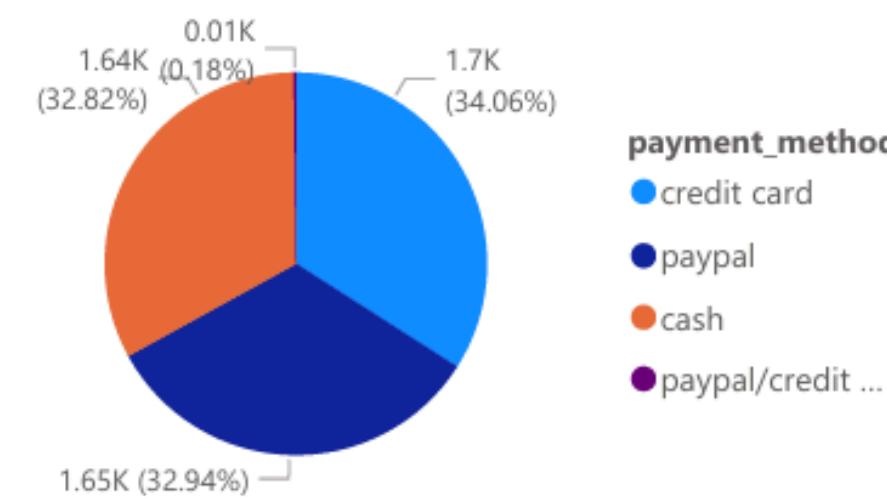


1.58M

Total Amount of Money paid by customers

payment_method	Total Number of payments per each method
cash	1641
credit card	1703
paypal	1647
paypal/credit card	9
Total	5000

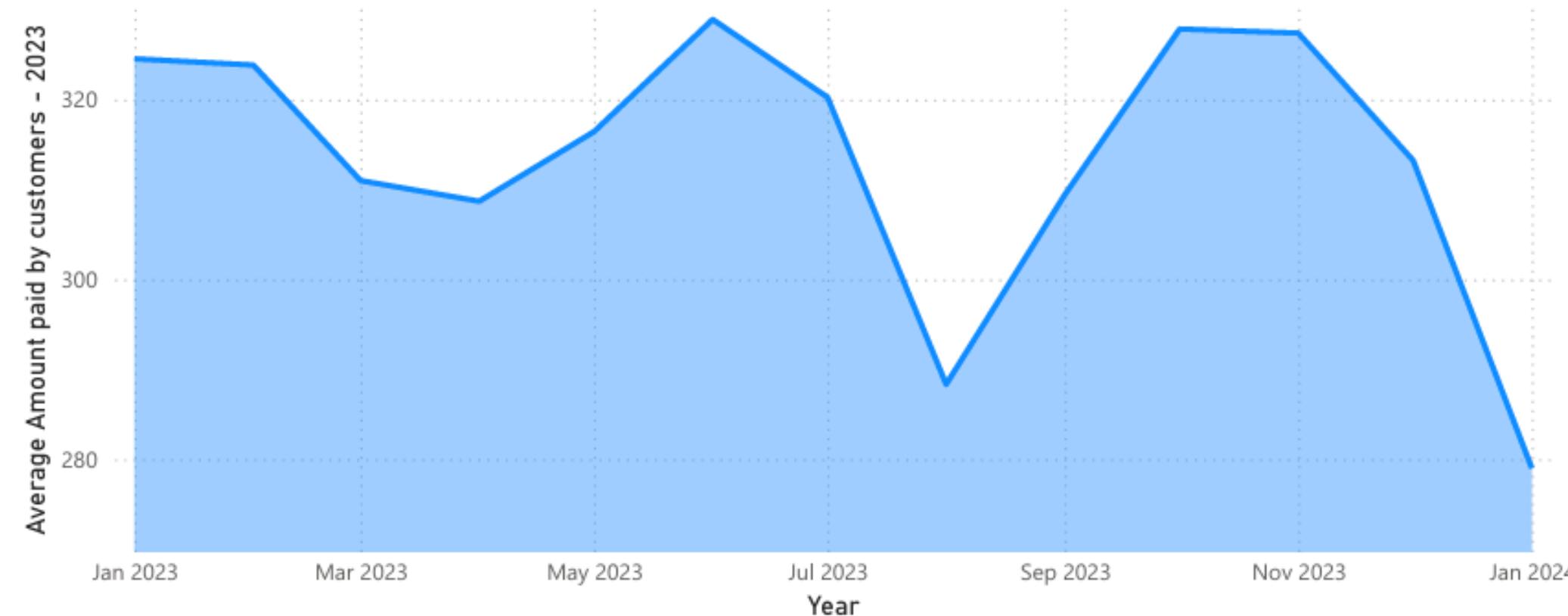
Total Number of payments by payment_method



PAYMENT ANALYSIS:

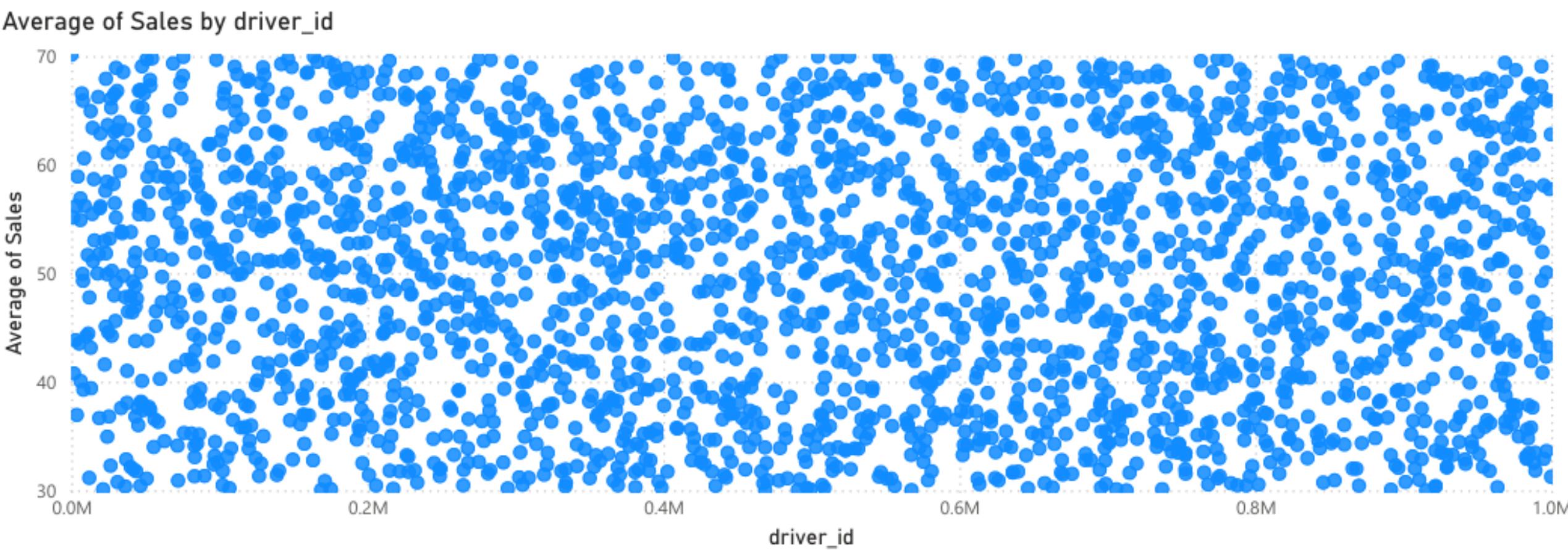
- Most of the payments occurring they were performed using credit cards, which gives us insights on the customers' preferred payment method.
- Customers tend to pay with credit cards more than cash or PayPal.
- In June 2023, payments were at maximum amount.

Average Amount paid by customers - 2023 by Year and Month



2600

Total Number of Drivers

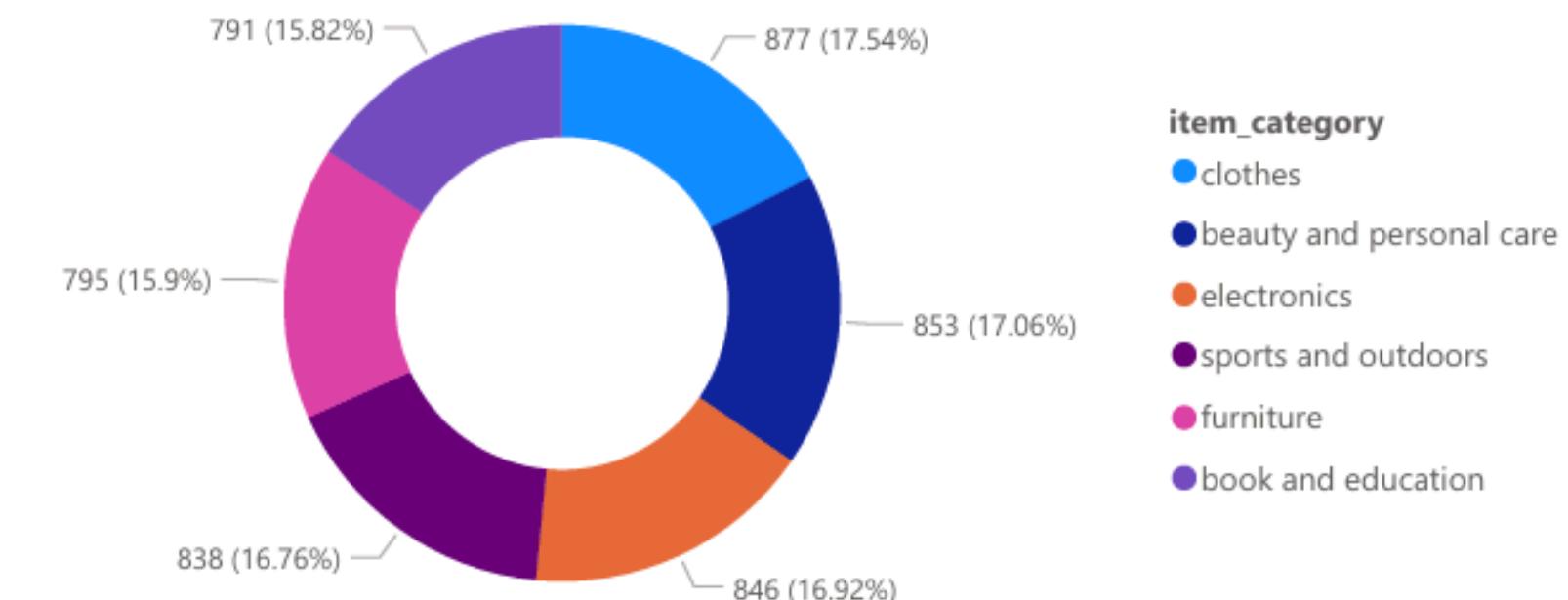


Top Drivers : TOP DRIVER: DARIN GORDON

number of deliveries	driver_id
9	366762
7	24581
7	190804
7	422500
7	496438
7	589389
7	813050
7	912280
7	929211
6	46272
5000	

first_name	last_name	driver_id
Aaron	Avila	968903
Aaron	Campbell	225750
Aaron	Harris	205781
Aaron	Hickman	198267
Aaron	King	172101
Aaron	Mitchell	595349
Aaron	Smith	391156
Aaron	Turner	117153
Abigail	Brady	791499
Abigail	Knapp	822880
Abigail	Meyers	706957

Number of Drivers by item_category





THANK YOU!