```
In [7]: import duckdb
        import pandas as pd
```

```
In [10]: con = duckdb.connect()
         customers = pd.read_csv("data/olist/olist_customers_dataset.csv")
         orders = pd.read_csv("data/olist/olist_orders_dataset.csv")
         order_items = pd.read_csv("data/olist/olist_order_items_dataset.csv")
         payments = pd.read_csv("data/olist/olist_order_payments_dataset.csv")
         reviews = pd.read_csv("data/olist/olist_order_reviews_dataset.csv")
         products = pd.read_csv("data/olist/olist_products_dataset.csv")
         sellers = pd.read_csv("data/olist/olist_sellers_dataset.csv")
         categories = pd.read_csv("data/olist/product_category_name_translation.csv")
```

Business Context

Olist is a Brazilian e-commerce platform that enables retailers to sell products across multiple online marketplaces through a single integrated system. Analysing data from the Olist store provides insight into customer behaviour, sales performance, delivery efficiency, and customer satisfaction. These insights support data-driven decision-making aimed at improving operational performance and identifying opportunities for growth within the Brazilian e-commerce market.

Dataset Overview

The Brazilian E-Commerce Public Dataset by Olist contains transactional data from around 100,000 orders placed between 2016 and 2018 across multiple online marketplaces in Brazil. The dataset is organised into several relational tables, allowing orders to be analysed across customers, products, sellers, payments, reviews, and geographic location.

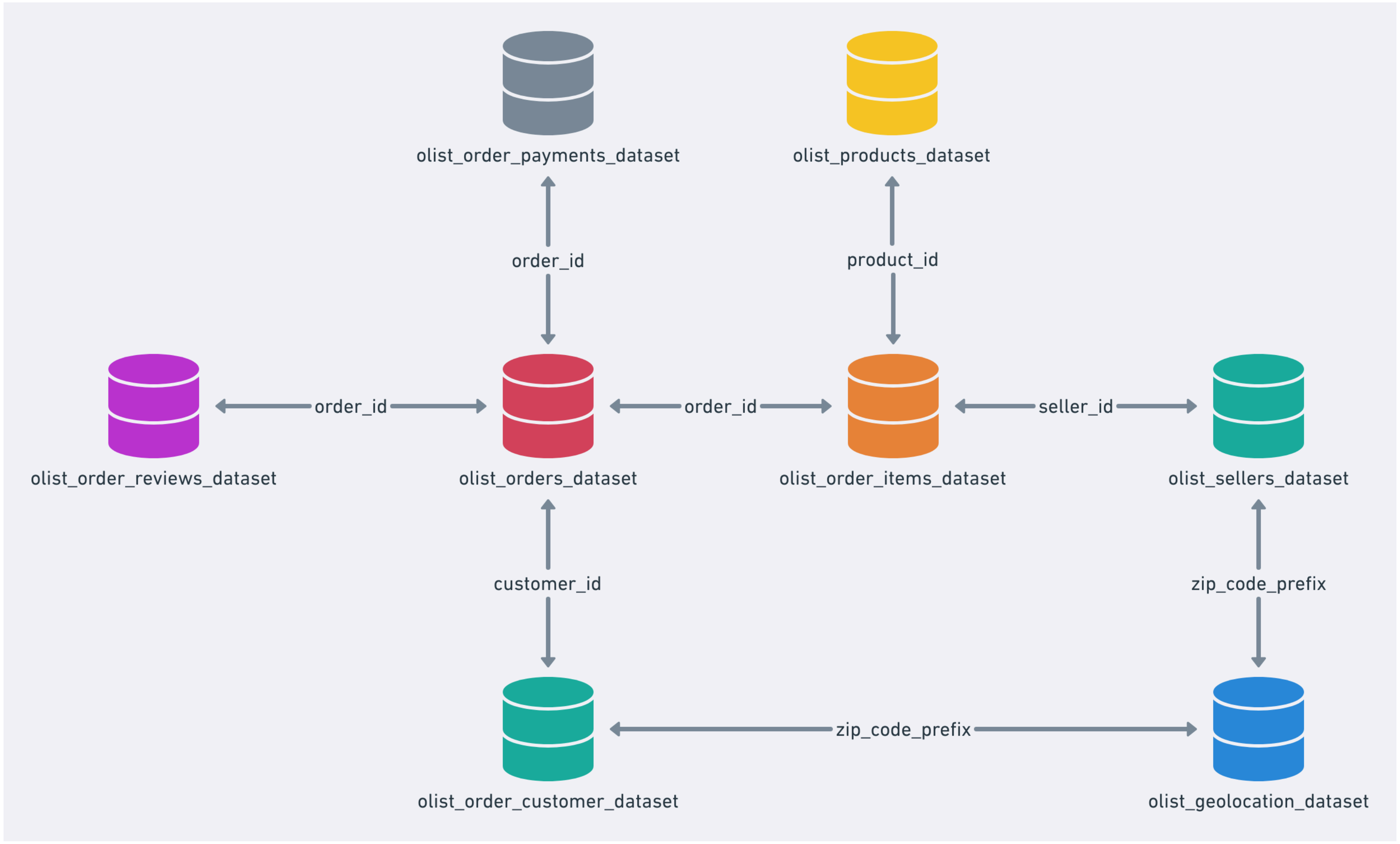
At the core of the dataset is the **olist_orders_dataset**, which represents individual customer orders and serves as the central table linking all other datasets. Each order is associated with a customer through a unique **customer_id**, enabling analysis of purchasing behaviour and delivery locations.

Order-level details are extended through the **olist_order_item_dataset**, which captures individual products within each order. This structure enables a single order to contain multiple items, each potentially sold by a different seller. Product attributes are stored in the **olist_products_dataset**, while seller information is provided in the **olist_sellers_dataset**, enabling seller-level and product-level performance analysis.

Payment information for each order is stored in the **olist_order_payments_dataset**, allowing analysis of payment methods, transaction values, and instalment behaviour. Customer satisfaction data is captured in the **olist_order_reviews_dataset**, which includes review scores and comments linked to orders after delivery.

Geographic analysis is supported through customer and seller location data. The **olist_customers_dataset** provides customer location details, which can be linked to the **olist_geolocation_dataset** using zip code prefixes. This enables spatial analysis of order distribution, delivery performance, and regional demand patterns.

Overall, the dataset's relational structure supports comprehensive analysis across the full e-commerce lifecycle, from order placement and payment through delivery and customer feedback.



Business Questions

```
In [11]: con.execute("""
         SELECT COUNT(*)
FROM orders
""").df()
```

Out[11]:

	count_star()
0	99441

1. What is the overall scale and timeframe of the Olist marketplace?

In [12]:

```
con.execute("""
SELECT
    COUNT(DISTINCT orders.order_id) AS total_orders,
    COUNT(DISTINCT orders.customer_id) AS total_customers,
    COUNT(DISTINCT order_items.seller_id) AS total_sellers,
    MIN(orders.order_purchase_timestamp) AS first_order_date,
    MAX(orders.order_purchase_timestamp) AS last_order_date,
    DATE_DIFF(
        'month',
        MIN(CAST(orders.order_purchase_timestamp AS TIMESTAMP)),
        MAX(CAST(orders.order_purchase_timestamp AS TIMESTAMP))
    ) AS active_months
FROM orders
LEFT JOIN order_items
    ON orders.order_id = order_items.order_id
WHERE orders.order_status NOT IN ('canceled', 'unavailable');
""").df()
```

Out[12]:

	total_orders	total_customers	total_sellers	first_order_date	last_order_date	active_months
0	98207	98207	3053	2016-09-04 21:15:19	2018-09-03 09:06:57	24

The results show that Olist is a large-scale e-commerce platform with broad participation across both the demand and supply sides of the marketplace. Over a two-year period, the platform processed more than 98,000 completed orders involving over 3,000 sellers. The relatively long and continuous timeframe suggests that the dataset captures stable marketplace operations, making it suitable for analysing growth patterns, customer engagement, seller dynamics, and operational performance over time.

2. How did marketplace activity evolve?

Understanding how order volume changes over time is critical for assessing whether the Olist marketplace experienced sustained growth, stagnation, or volatility during the observed period. Examining monthly order trends provides insight into demand dynamics and helps determine whether subsequent analyses reflect stable marketplace behaviour or are influenced by rapid expansion or contraction.

In [13]:
```
con.execute("""
SELECT
    DATE_TRUNC(
        'month',
        CAST(order_purchase_timestamp AS TIMESTAMP)
    ) AS order_month,
    COUNT(DISTINCT order_id) AS total_orders
FROM orders
WHERE order_status NOT IN ('canceled', 'unavailable')
GROUP BY order_month
ORDER BY order_month;
""").df()
```

Out[13]:

	order_month	total_orders
0	2016-09-01	2
1	2016-10-01	293
2	2016-12-01	1
3	2017-01-01	787
4	2017-02-01	1718
5	2017-03-01	2617
6	2017-04-01	2377
7	2017-05-01	3640
8	2017-06-01	3205
9	2017-07-01	3946
10	2017-08-01	4272
11	2017-09-01	4227
12	2017-10-01	4547
13	2017-11-01	7423
14	2017-12-01	5620
15	2018-01-01	7187
16	2018-02-01	6625
17	2018-03-01	7168
18	2018-04-01	6919
19	2018-05-01	6833
20	2018-06-01	6145
21	2018-07-01	6233
22	2018-08-01	6421

	order_month	total_orders
23	2018-09-01	1

Monthly order volumes show that the Olist marketplace experienced strong growth over time. Activity was minimal in late 2016, reflecting the early stage of the platform, before increasing steadily throughout 2017. By early 2018, order volumes stabilised at a consistently high level, suggesting a transition from rapid expansion to a more mature operational phase. The very low order counts at the beginning and end of the dataset likely reflect partial months rather than true declines in activity.

3. How is marketplace activity distributed across sellers?

In [14]:
```
con.execute("""
SELECT
    seller_id,
    COUNT(DISTINCT order_id) AS total_orders
FROM order_items
GROUP BY seller_id
ORDER BY total_orders DESC;
""").df()
```

Out[14]:

```
                              seller_id  total_orders
0      6560211a19b47992c3666cc44a7e94c0          1854
1      4a3ca9315b744ce9f8e9374361493884          1806
2      cc419e0650a3c5ba77189a1882b7556a          1706
3      1f50f920176fa81dab994f9023523100          1404
4      da8622b14eb17ae2831f4ac5b9dab84a          1314
...                                 ...           ...
3090   4cc4fd4fdd406a85bbdc1f824b731bd7             1
3091   f90f77ef2799a27f80d90c425ca944f7             1
3092   a3fa18b3f688ec0fca3eb8bfcbd2d5b3             1
3093   d9b00a97818674c7f8d8b1ef0e689679             1
3094   270572bb714b00531be85e16e1550f26             1

3095 rows × 2 columns
```

In [21]:

```
con.execute("""
WITH seller_orders AS (
    SELECT
        oi.seller_id,
        COUNT(DISTINCT oi.order_id) AS total_orders
    FROM order_items oi
    JOIN orders o
        ON oi.order_id = o.order_id
    WHERE o.order_status NOT IN ('canceled', 'unavailable')
    GROUP BY oi.seller_id

),
total_sellers AS (
    SELECT COUNT(*) AS total_count
    FROM seller_orders
)
SELECT
    CASE
        WHEN total_orders = 1 THEN '1 order'
        WHEN total_orders BETWEEN 2 AND 5 THEN '2–5 orders'
        WHEN total_orders BETWEEN 6 AND 20 THEN '6–20 orders'
        WHEN total_orders BETWEEN 21 AND 100 THEN '21–100 orders'
        ELSE '100+ orders'
    END AS order_bucket,
    COUNT(*) AS seller_count,
    ROUND(
        COUNT(*) * 100.0 / total_sellers.total_count,
        2
    ) AS seller_percentage

FROM seller_orders
CROSS JOIN total_sellers
GROUP BY order_bucket, total_sellers.total_count
ORDER BY
    CASE
        WHEN order_bucket = '1 order' THEN 1
        WHEN order_bucket = '2–5 orders' THEN 2
        WHEN order_bucket = '6–20 orders' THEN 3
        WHEN order_bucket = '21–100 orders' THEN 4
        ELSE 5
    END;

""").df()
```

Out[21]:

	order_bucket	seller_count	seller_percentage
0	1 order	560	18.34
1	2–5 orders	851	27.87
2	6–20 orders	847	27.74
3	21–100 orders	585	19.16
4	100+ orders	210	6.88

Marketplace activity on Olist is unevenly distributed across sellers, reflecting a clear long-tail structure. While a substantial proportion of sellers complete only a small number of orders, most commonly between one and five, a much smaller group of sellers accounts for very high order volumes. Specifically, fewer than 7 percent of sellers process more than 100 orders, while the majority operate at low to moderate transaction levels. This indicates that overall marketplace performance is disproportionately influenced by a relatively small subset of high-activity sellers, a pattern typical of large e-commerce platforms.

4. How is demand distributed across customers?

To better understand demand dynamics within the Olist marketplace, it is important to examine how purchasing activity is distributed across customers and whether orders are driven by repeat buyers or one-time purchasers.

In [26]:
```
con.execute("""
WITH customer_orders AS (
    SELECT
        c.customer_unique_id,
        COUNT(DISTINCT o.order_id) AS total_orders
    FROM orders o
    JOIN customers c
        ON o.customer_id = c.customer_id
    WHERE o.order_status NOT IN ('canceled', 'unavailable')
    GROUP BY c.customer_unique_id
),
total_customers AS (
    SELECT COUNT(*) AS total_count
    FROM customer_orders
)
SELECT
    CASE
        WHEN total_orders = 1 THEN '1 order'
        WHEN total_orders BETWEEN 2 AND 5 THEN '2–5 orders'
        WHEN total_orders BETWEEN 6 AND 20 THEN '6–20 orders'
        WHEN total_orders BETWEEN 21 AND 100 THEN '21–100 orders'
        ELSE '100+ orders'
    END AS order_bucket,
    COUNT(*) AS customer_count,
    ROUND(
        COUNT(*) * 100.0 / total_customers.total_count,
        2
    ) AS customer_percentage
FROM customer_orders
CROSS JOIN total_customers
GROUP BY order_bucket, total_customers.total_count
ORDER BY
    CASE
        WHEN order_bucket = '1 order' THEN 1
        WHEN order_bucket = '2–5 orders' THEN 2
        WHEN order_bucket = '6–20 orders' THEN 3
        WHEN order_bucket = '21–100 orders' THEN 4
        ELSE 5
    END;
""").df()
```

Out[26]:

	order_bucket	customer_count	customer_percentage
0	1 order	92102	96.96
1	2–5 orders	2878	3.03
2	6–20 orders	10	0.01

Customer purchasing behaviour on the Olist marketplace is highly concentrated among one-time buyers. Nearly all customers place a single order, while only a small fraction return for multiple purchases. This indicates that marketplace demand is driven primarily by new customer acquisition rather than repeat purchasing, suggesting limited customer retention over the observed period.

5. Comparing sellers vs customers directly

While seller activity follows a long-tail distribution with a meaningful group of high-volume sellers, customer demand is far more concentrated among one-time buyers. This imbalance suggests that marketplace growth is driven primarily by seller participation and new customer acquisition rather than repeat purchasing behaviour.

6. What product categories drive marketplace demand?

To better understand the drivers of marketplace activity, the next step is to examine which product categories account for the highest share of completed orders.

```
In [27]: con.execute("""
SELECT
    t.product_category_name_english AS category,
    COUNT(DISTINCT oi.order_id) AS total_orders
FROM order_items oi
JOIN products p
    ON oi.product_id = p.product_id
JOIN categories t
    ON p.product_category_name = t.product_category_name
JOIN orders o
    ON oi.order_id = o.order_id
WHERE o.order_status NOT IN ('canceled', 'unavailable')
GROUP BY category
ORDER BY total_orders DESC;
""").df()
```

Out[27]:

	category	total_orders
0	bed_bath_table	9399
1	health_beauty	8800
2	sports_leisure	7673
3	computers_accessories	6654
4	furniture_decor	6425
...	...	...
66	arts_and_craftmanship	23
67	la_cuisine	13
68	cds_dvds_musicals	12
69	fashion_childrens_clothes	8
70	security_and_services	2

71 rows × 2 columns

Marketplace demand on Olist is driven primarily by everyday consumer and household categories rather than specialised or infrequently purchased products. The highest-volume categories include bed, bath and table items, health and beauty products, sports and leisure goods, computer accessories, and furniture and home décor. These categories typically reflect functional or lifestyle purchases rather than high-involvement or recurring subscriptions.

At the lower end of the distribution, a large number of niche categories account for very few orders, indicating a broad but uneven product offering. This pattern suggests that while the marketplace supports a wide variety of sellers and products, overall transaction volume is concentrated in a relatively small set of popular categories. The dominance of categories associated with infrequent or replacement-based purchases helps explain the low level of customer repeat purchasing observed earlier.

7. How do delivery outcomes relate to customer satisfaction?

Delivery performance is a critical driver of customer satisfaction in e-commerce. Delays or failed deliveries can negatively affect customer experience and discourage repeat purchasing, particularly in marketplaces dominated by one-time buyers.

In [28]:
```
con.execute("""
SELECT
    CASE
        WHEN o.order_delivered_customer_date <= o.order_estimated_delivery_date
            THEN 'On-time delivery'
        ELSE 'Late delivery'
    END AS delivery_status,
    COUNT(DISTINCT o.order_id) AS total_orders,
    ROUND(AVG(r.review_score), 2) AS avg_review_score
FROM orders o
JOIN reviews r
    ON o.order_id = r.order_id
WHERE o.order_status = 'delivered'
  AND o.order_delivered_customer_date IS NOT NULL
GROUP BY delivery_status;
""").df()
```

Out[28]:

	delivery_status	total_orders	avg_review_score
0	On-time delivery	88163	4.29
1	Late delivery	7661	2.57

Delivery performance has a strong relationship with customer satisfaction on the Olist marketplace. Orders delivered on time receive an average review score of 4.29, indicating generally high satisfaction. In contrast, late deliveries receive a substantially lower average score of 2.57, reflecting negative customer experiences. While most orders are delivered on time, the sharp drop in satisfaction associated with late deliveries suggests that delivery performance is a key factor influencing customer perceptions. This relationship likely contributes to the high proportion of one-time buyers observed earlier, as poor delivery experiences reduce the likelihood of repeat purchasing.

8. Conclusion

The analysis shows that the Olist marketplace operates at significant scale and experienced strong growth over the observed two-year period. Marketplace activity is characterised by a long-tail seller structure, where a small group of high-volume sellers accounts for a disproportionate share of orders, while customer demand is overwhelmingly driven by one-time buyers. Demand is concentrated in household and lifestyle product categories that are typically infrequently purchased, which helps explain low repeat purchasing behaviour. In addition, delivery performance plays a critical role in shaping customer satisfaction, with late deliveries associated with substantially lower review scores. Taken together, these findings suggest that Olist’s growth relies heavily on seller participation and new customer acquisition, while improvements in delivery reliability may be necessary to strengthen customer retention.