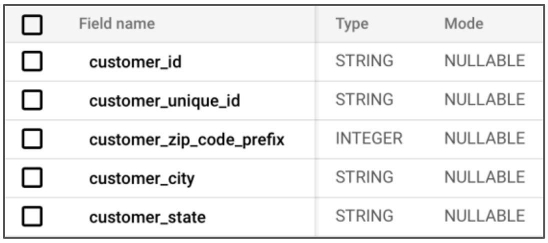
Q1. Business Case: Target SQL

1. (1)

RESULT SCREENSHOT:



□	Field name	Type	Mode
□	customer_id	STRING	NULLABLE
□	customer_unique_id	STRING	NULLABLE
□	customer_zip_code_prefix	INTEGER	NULLABLE
□	customer_city	STRING	NULLABLE
□	customer_state	STRING	NULLABLE

INFERENCE:

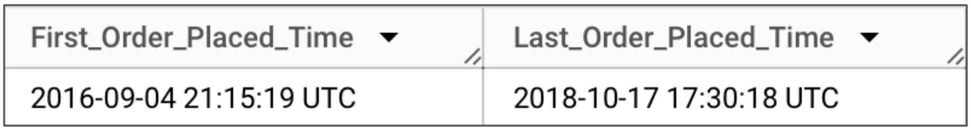
The 'Customers' table consists of columns with data types as *STRING* and *INTEGER.*

1. (2)

QUERY:

```
SELECT
MIN(order_purchase_timestamp) AS First_Order_Placed_Time,
MAX(order_purchase_timestamp) AS Last_Order_Placed_Time
FROM `target_business_case1.orders`
```

RESULT SCREENSHOT:



First_Order_Placed_Time	Last_Order_Placed_Time
2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC

INFERENCE:

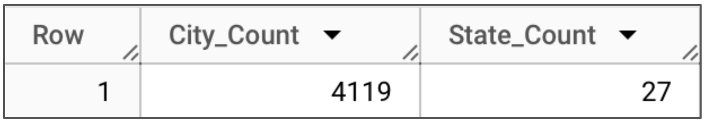
The Target orders were placed between the time range ***2016-09-04 21:15:19 UTC*** *and* ***2018-10-17 17:30:18 UTC***

1. (3)

QUERY:

```
SELECT
COUNT(DISTINCT customer_city) AS City_Count,
COUNT(DISTINCT customer_state) AS State_Count
FROM `target_business_case1.customers`
WHERE customer_id IN
(SELECT DISTINCT(customer_id) FROM `target_business_case1.orders`)
```

RESULT SCREENSHOT:



Row	City_Count	State_Count
1	4119	27

INFERENCE:

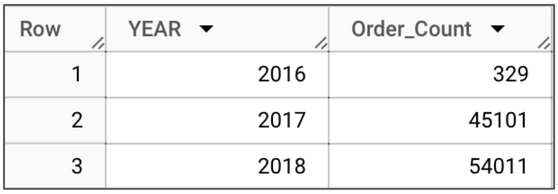
Customers placed orders from a total of 4119 Cities and 27 States during the period ranging from 2016 to 2018.

2. (1)

QUERY:

```
SELECT
EXTRACT(YEAR FROM order_purchase_timestamp) AS YEAR,
COUNT(order_id) AS Order_Count
FROM `target_business_case1.orders`
GROUP BY 1
ORDER BY 1
```

RESULT SCREENSHOT:



Row	YEAR	Order_Count
1	2016	329
2	2017	45101
3	2018	54011

INFERENCE:

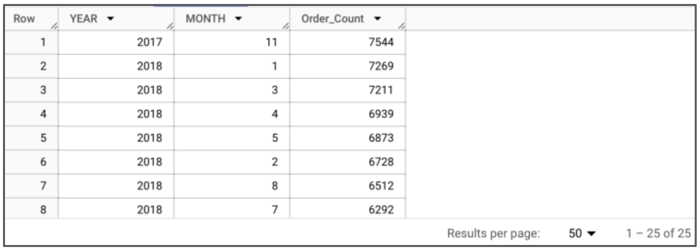
Based on the data analysis, there is a clear growing trend observed in the Target business. Initially when it was setup in Brazil in the year 2016, it was not consistent and efficient. However, starting from 2017 to 2018, the business grew drastically.

2. (2)

QUERY:

```
SELECT
EXTRACT(YEAR FROM order_purchase_timestamp) AS YEAR,
EXTRACT(MONTH FROM order_purchase_timestamp) AS MONTH,
COUNT(order_id) AS Order_Count
FROM `target_business_case1.orders`
GROUP BY 1, 2 ORDER BY 3 desc
```

RESULT SCREENSHOT:



Row	YEAR ▾	MONTH ▾	Order_Count ▾
1	2017	11	7544
2	2018	1	7269
3	2018	3	7211
4	2018	4	6939
5	2018	5	6873
6	2018	2	6728
7	2018	8	6512
8	2018	7	6292

Results per page: 50 ▾ 1 – 25 of 25

INFERENCE:

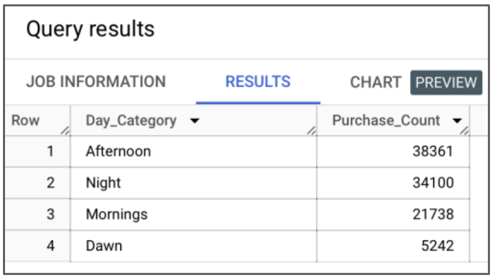
With reference to the data assessed, maximum orders were placed in the month of November 2017 and January 2018. The increase in business in these months could be due to festive days like Thanksgiving Day in Nov'17 and New Year in Jan'2018.

2. (3)

QUERY:

```
SELECT Day_Category, COUNT(b.purchase_time) AS Purchase_Count FROM
(SELECT purchase_time, CASE
        WHEN purchase_time BETWEEN '00:00:00' AND '06:59:59' THEN
        'Dawn'
        WHEN purchase_time BETWEEN '07:00:00' AND '11:59:59' THEN
        'Mornings'
        WHEN purchase_time BETWEEN '12:00:00' AND '17:59:59' THEN
        'Afternoon'
        WHEN purchase_time BETWEEN '18:00:00' AND '23:59:59' THEN
        'Night'
        END AS Day_Category
FROM (SELECT EXTRACT(TIME FROM order_purchase_timestamp) AS
purchase_time FROM `target_business_case1.orders`) AS a) b
GROUP BY 1
ORDER BY 2 DESC
```

RESULT SCREENSHOT:



Row	Day_Category	Purchase_Count
1	Afternoon	38361
2	Night	34100
3	Mornings	21738
4	Dawn	5242

INFERENCE:

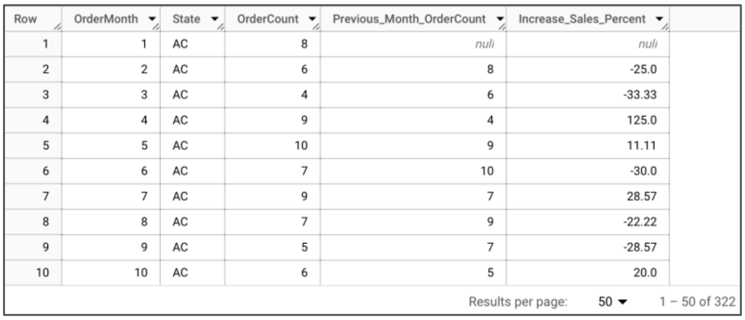
Considering the examined data, most of the orders were placed during the ***afternoon*** *day category with maximum purchase count as 38361. Near to it, the second highest order count is seen during the night with count as 34100.*

3. (1)

QUERY:

```
SELECT *,
ROUND(((OrderCount -
Previous_Month_OrderCount)/Previous_Month_OrderCount) * 100,2) AS
Increase_Sales_Percent
FROM
(SELECT *,
LAG(OrderCount,1)OVER(PARTITION BY State ORDER BY OrderMonth) AS
Previous_Month_OrderCount
FROM
(SELECT EXTRACT(MONTH FROM order_purchase_timestamp) AS
OrderMonth,
c.customer_state AS State,
COUNT(o.order_id) AS OrderCount
FROM `target_business_case1.orders` o
INNER JOIN `target_business_case1.customers` c
ON c.customer_id = o.customer_id
GROUP BY 1,2) a ) b
ORDER BY 2,1
```

RESULT SCREENSHOT:



Row	OrderMonth	State	OrderCount	Previous_Month_OrderCount	Increase_Sales_Percent
1	1	AC	8	*null*	*null*
2	2	AC	6	8	-25.0
3	3	AC	4	6	-33.33
4	4	AC	9	4	125.0
5	5	AC	10	9	11.11
6	6	AC	7	10	-30.0
7	7	AC	9	7	28.57
8	8	AC	7	9	-22.22
9	9	AC	5	7	-28.57
10	10	AC	6	5	20.0

INFERENCE:

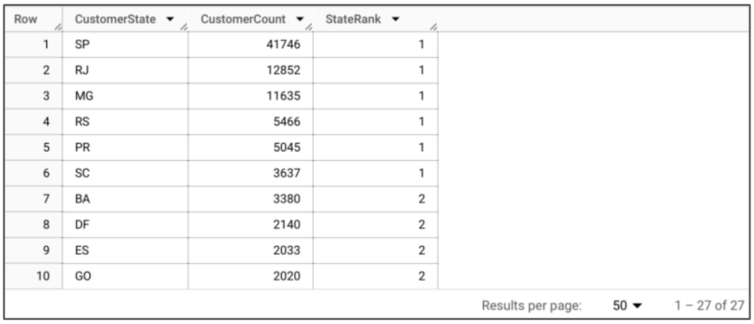
Based on month-on-month sales analysis, overall, there is an increase in sales observed for many states. The positive values in Increase_Sales_Percent column indicate an increase in sales percent for that month and negative values indicate a decrease in sales percent.

3. (2)

QUERY:

```
SELECT *, NTILE(5)OVER(ORDER BY CustomerCount desc) AS StateRank
FROM
(SELECT customer_state AS CustomerState, COUNT(customer_id) AS
CustomerCount
FROM `target_business_case1.customers`
GROUP BY 1) a
ORDER BY 3,2 desc
```

RESULT SCREENSHOT:



Row	CustomerState	CustomerCount	StateRank
1	SP	41746	1
2	RJ	12852	1
3	MG	11635	1
4	RS	5466	1
5	PR	5045	1
6	SC	3637	1
7	BA	3380	2
8	DF	2140	2
9	ES	2033	2
10	GO	2020	2

INFERENCE:

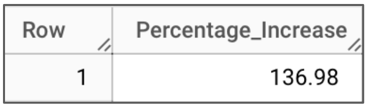
*In analyzing the customer distribution across all states, there is significant variability in the number of customers in each state. Most business is brought by 20% of states with **StateRank** value 1.*

4. (1)

QUERY:

```
SELECT
ROUND(((SUM(CASE WHEN year=2018 THEN cost_of_orders ELSE 0 END) -
SUM(CASE WHEN year=2017 THEN cost_of_orders ELSE 0 END)) /
SUM(CASE WHEN year=2017 THEN cost_of_orders ELSE 0 END)) * 100,2) AS
PercentageIncrease FROM
(SELECT * FROM
          (SELECT
          EXTRACT(YEAR FROM o.order_purchase_timestamp) AS year,
          EXTRACT(MONTH FROM o.order_purchase_timestamp) AS month,
          ROUND((SUM(payment_value)),2) AS cost_of_orders
          FROM `target_business_case1.payments` p
          INNER JOIN `target_business_case1.orders` o ON p.order_id =
          o.order_id
          GROUP BY 1,2) a
WHERE a.year IN (2017,2018) AND (a.MONTH BETWEEN 1 AND 8)
ORDER BY 2,1) b
```

RESULT SCREENSHOT:



Row	Percentage_Increase
1	136.98

INFERENCE:

Considering the examined data, there is an increase of 136.98% sales revenue from 2017 to 2018 for months between January and August indicating increasing sales trend.

NOTE: *For questions 4. (2) & 4. (3), query retrieves data from view created named as* **`target_business_case1.price_freight_view`**

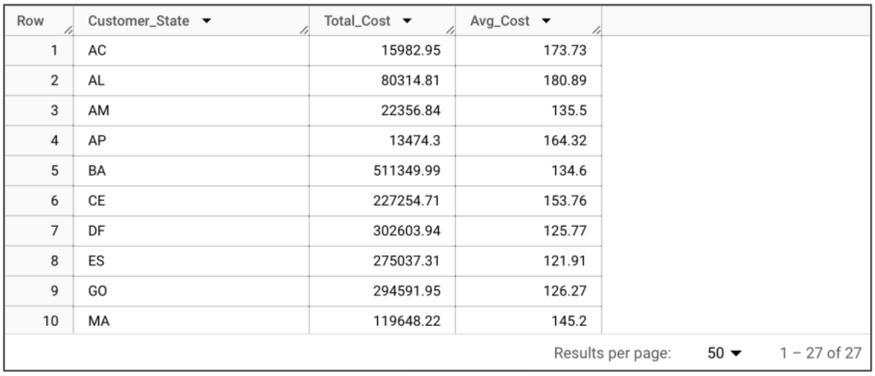
```
CREATE VIEW `target_business_case1.price_freight_view` AS
SELECT
        oi.order_id,
        oi.price,
        oi.freight_value,
        o.customer_id,
        c.customer_state
FROM `target_business_case1.order_items` oi
INNER JOIN `target_business_case1.orders` o ON o.order_id = oi.order_id
INNER JOIN `target_business_case1.customers` c ON c.customer_id =
o.customer_id;
```

4. (2)

QUERY:

```
SELECT
    customer_state AS Customer_State,
    ROUND(SUM(price),2) AS Total_Cost,
    ROUND(AVG(price),2) AS Avg_Cost,
FROM `target_business_case1.price_freight_view`
GROUP BY 1
ORDER BY 1
```

RESULT SCREENSHOT:



Row	Customer_State	Total_Cost	Avg_Cost
1	AC	15982.95	173.73
2	AL	80314.81	180.89
3	AM	22356.84	135.5
4	AP	13474.3	164.32
5	BA	511349.99	134.6
6	CE	227254.71	153.76
7	DF	302603.94	125.77
8	ES	275037.31	121.91
9	GO	294591.95	126.27
10	MA	119648.22	145.2

Results per page: 50 1 – 27 of 27

INFERENCE:

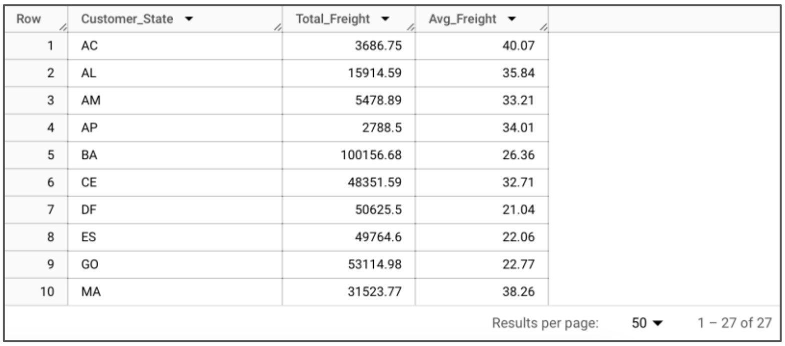
Based on the analysis, total order costs differ significantly for each state. There are some high performing states with higher total and average costs. These states can be focused on new product promotion. In contrast, states with lower total and average cost need to identify the challenges to improve business sales.

4. (3)

QUERY:

```
SELECT
    customer_state AS Customer_State,
    ROUND(SUM(freight_value),2) AS Total_Freight,
    ROUND(AVG(freight_value),2) AS Avg_Freight,
FROM `target_business_case1.price_freight_view`
GROUP BY 1
ORDER BY 1
```

RESULT SCREENSHOT:



Row	Customer_State	Total_Freight	Avg_Freight
1	AC	3686.75	40.07
2	AL	15914.59	35.84
3	AM	5478.89	33.21
4	AP	2788.5	34.01
5	BA	100156.68	26.36
6	CE	48351.59	32.71
7	DF	50625.5	21.04
8	ES	49764.6	22.06
9	GO	53114.98	22.77
10	MA	31523.77	38.26

Results per page: 50 ▾ 1 – 27 of 27

INFERENCE:

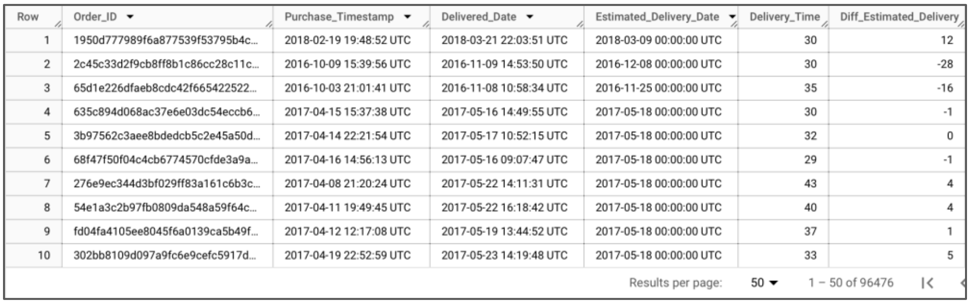
With reference to freight price analysis, both the total and average freight pricing varies from state to state. There are states with high total and high average freight that could mainly be focusing on premium shipping services or the shipping distance to cover could be large.
Some states have low total and low average freight pricing. They are cost efficient, but delivery services may be compromised.
States with high total and low average freight indicate less per unit cost with a greater number of units being transported. This is an optimal choice considering the cost factor and effectiveness.

5. (1)

QUERY:

```
SELECT
order_id AS Order_ID,
order_purchase_timestamp AS Purchase_Timestamp,
order_delivered_customer_date AS Delivered_Date,
order_estimated_delivery_date AS Estimated_Delivery_Date,
DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp,
day) AS Delivery_Time,
DATE_DIFF(order_delivered_customer_date,
order_estimated_delivery_date, day) AS Diff_Estimated_Delivery
FROM `target_business_case1.orders`
WHERE order_delivered_customer_date IS NOT NULL
```

RESULT SCREENSHOT:



Row	Order_ID	Purchase_Timestamp	Delivered_Date	Estimated_Delivery_Date	Delivery_Time	Diff_Estimated_Delivery
1	1950d777989f6a877539f53795b4c...	2018-02-19 19:48:52 UTC	2018-03-21 22:03:51 UTC	2018-03-09 00:00:00 UTC	30	12
2	2c45c33d2f9cb8ff8b1c86cc28c11c...	2016-10-09 15:39:56 UTC	2016-11-09 14:53:50 UTC	2016-12-08 00:00:00 UTC	30	-28
3	65d1e226dfaeb8cdc42f665422522...	2016-10-03 21:01:41 UTC	2016-11-08 10:58:34 UTC	2016-11-25 00:00:00 UTC	35	-16
4	635c894d068ac37e6e03dc54eccb6...	2017-04-15 15:37:38 UTC	2017-05-16 14:49:55 UTC	2017-05-18 00:00:00 UTC	30	-1
5	3b97562c3aee8bdedcb5c2e45a50d...	2017-04-14 22:21:54 UTC	2017-05-17 10:52:15 UTC	2017-05-18 00:00:00 UTC	32	0
6	68f47f50f04c4cb6774570cfde3a9a...	2017-04-16 14:56:13 UTC	2017-05-16 09:07:47 UTC	2017-05-18 00:00:00 UTC	29	-1
7	276e9ec344d3bf029ff83a161c6b3c...	2017-04-08 21:20:24 UTC	2017-05-22 14:11:31 UTC	2017-05-18 00:00:00 UTC	43	4
8	54e1a3c2b97fb0809da548a59f64c...	2017-04-11 19:49:45 UTC	2017-05-22 16:18:42 UTC	2017-05-18 00:00:00 UTC	40	4
9	fd04fa4105ee8045f6a0139ca5b49f...	2017-04-12 12:17:08 UTC	2017-05-19 13:44:52 UTC	2017-05-18 00:00:00 UTC	37	1
10	302bb8109d097a9fc6e9cefc5917d...	2017-04-19 22:52:59 UTC	2017-05-23 14:19:48 UTC	2017-05-18 00:00:00 UTC	33	5

INFERENCE:

In accordance with the data studied, 2 columns are derived.
Delivery_Time *is the difference in days between actual delivery date and purchase date.*
Diff_Estimated_Delivery *is the difference in days between actual delivery date and estimated delivery date.*
All the NULL values from column 'order_delivered_customer_date' are excluded as the null values might be due to missing data, customer canceling the order, or the order got returned due to some unavoidable situation.
The negative values in this column indicate delivery completion prior to the estimated delivery date and the positive values indicate late delivery post the estimated date.

5. (2)

QUERY:

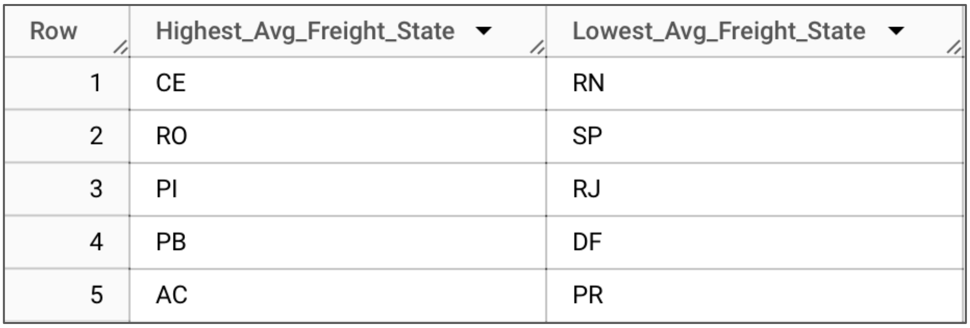
```
WITH freight AS
(SELECT
        g.geolocation_state,
        ROUND(AVG(oi.freight_value),2) AS avg_freight,
FROM `target_business_case1.order_items` oi
INNER JOIN `target_business_case1.sellers` s ON s.seller_id = oi.seller_id
INNER JOIN `target_business_case1.geolocation` g
ON g.geolocation_zip_code_prefix = s.seller_zip_code_prefix
GROUP BY 1),

highest AS
(SELECT
        geolocation_state AS Highest_Avg_Freight_State,
        MAX(avg_freight) AS highest_avg_freight FROM freight
GROUP BY 1 ORDER BY 2 DESC LIMIT 5),

lowest AS
(SELECT
        geolocation_state AS Lowest_Avg_Freight_State,
        MIN(avg_freight) AS lowest_avg_freight FROM freight
GROUP BY 1 ORDER BY 2 LIMIT 5)

SELECT Highest_Avg_Freight_State, Lowest_Avg_Freight_State FROM
(SELECT
        Highest_Avg_Freight_State,
        ROW_NUMBER()OVER(ORDER BY highest_avg_freight desc) AS rnk
FROM highest) H
INNER JOIN
(SELECT
        Lowest_Avg_Freight_State,
        ROW_NUMBER()OVER(ORDER BY lowest_avg_freight) AS rnk FROM
lowest) L
ON L.rnk = H.rnk
```

RESULT SCREENSHOT:



Row	Highest_Avg_Freight_State	Lowest_Avg_Freight_State
1	CE	RN
2	RO	SP
3	PI	RJ
4	PB	DF
5	AC	PR

INFERENCE:

*According to the data assessed, calculated the average freight details for each state. The lowest average freight indicates the cost of per unit is less and higher average freight indicates the cost of per unit is more. Obtained the details of the top 5 states with the highest and lowest average freight in columns ‘*Highest_Avg_Freight_State*’ and ‘*Lowest_Avg_Freight_State*’ respectively.*

NOTE: *For questions 5. (3) & 5. (4), query retrieves data from view created named as* ***`target_business_case1. order_state_view`***

```
CREATE VIEW `target_business_case1.order_state_view` AS
SELECT
    c.customer_state,
    o.order_purchase_timestamp,
    o.order_delivered_customer_date,
    o.order_estimated_delivery_date,
    timestamp_diff(order_delivered_customer_date,
order_purchase_timestamp, hour)
    AS DP_time_diff,
    timestamp_diff(order_delivered_customer_date,
order_estimated_delivery_date, hour)
    AS DE_time_diff
FROM
(SELECT
    customer_id,
    order_purchase_timestamp,
    order_estimated_delivery_date,
    order_delivered_customer_date
FROM `target_business_case1.orders`
WHERE order_delivered_customer_date IS NOT NULL) o
INNER JOIN `target_business_case1.customers` c ON o.customer_id =
c.customer_id;
```

5. (3)

QUERY:

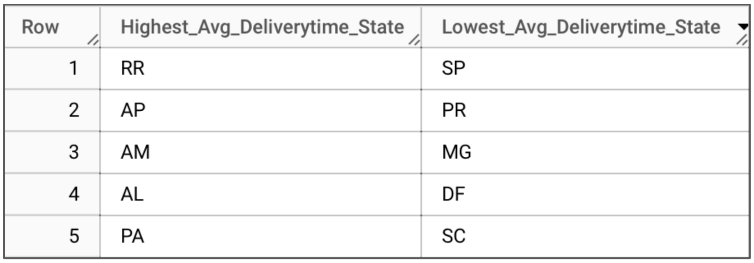
```
WITH CTE1 AS
(SELECT
        DISTINCT customer_state,
        ROUND(AVG(DP_time_diff/24)OVER(PARTITION BY
customer_state),2) AS AVG_Delivery_time
FROM `target_business_case1.order_state_view`),

highest AS
(SELECT customer_state,
ROUND(MAX(AVG_Delivery_time),2) AS highest_avg_dt
FROM CTE1 GROUP BY 1 ORDER BY 2 DESC LIMIT 5),

lowest AS
(SELECT customer_state,
ROUND(MIN(AVG_Delivery_time),2) AS lowest_avg_dt
FROM CTE1 GROUP BY 1 ORDER BY 2 LIMIT 5)

SELECT Highest_Avg_Deliverytime_State, Lowest_Avg_Deliverytime_State
FROM
        (SELECT customer_state AS Highest_Avg_Deliverytime_State,
        ROW_NUMBER()OVER(ORDER BY highest_avg_dt desc) AS rnk
FROM highest) H
INNER JOIN
        (SELECT customer_state AS Lowest_Avg_Deliverytime_State,
        ROW_NUMBER()OVER(ORDER BY lowest_avg_dt) AS rnk FROM
lowest) L
ON L.rnk = H.rnk
```

RESULT SCREENSHOT:



Row	Highest_Avg_Deliverytime_State	Lowest_Avg_Deliverytime_State
1	RR	SP
2	AP	PR
3	AM	MG
4	AL	DF
5	PA	SC

INFERENCE:

Calculated the average delivery time details for each state and obtained the details of the top 5 states with highest and lowest average delivery time taken.

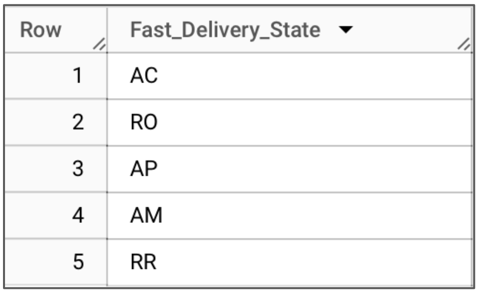
5. (4)

QUERY:

```
WITH CTE1 AS
(SELECT
DISTINCT customer_state AS Fast_Delivery_State,
ROUND(AVG(DE_time_diff/24)OVER(PARTITION BY customer_state),2) AS
AVG_Quick_DT
FROM `target_business_case1.view1`
ORDER BY 2)

SELECT Fast_Delivery_State FROM CTE1
LIMIT 5
```

RESULT SCREENSHOT:



Row	Fast_Delivery_State
1	AC
2	RO
3	AP
4	AM
5	RR

INFERENCE:

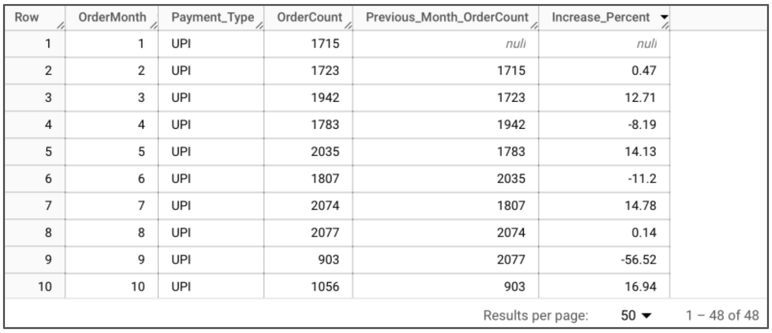
After analyzing the data, the top 5 states with fast order delivery as compared to the estimated date of delivery are extracted.

6. (1)

QUERY:

```
SELECT *, ROUND(((OrderCount -
Previous_Month_OrderCount)/Previous_Month_OrderCount) * 100, 2) AS
Increase_Percent
FROM
(SELECT *,
     LAG(OrderCount,1)OVER(PARTITION BY Payment_Type ORDER BY
OrderMonth) AS Previous_Month_OrderCount
FROM
(SELECT
     EXTRACT(MONTH FROM order_purchase_timestamp) AS OrderMonth,
     p.payment_type AS Payment_Type,
     COUNT(o.order_id) AS OrderCount
FROM `target_business_case1.orders` o
INNER JOIN (SELECT order_id, payment_type FROM
`target_business_case1.payments`
WHERE payment_type != 'not_defined') p ON p.order_id = o.order_id
GROUP BY 1,2) a ) b
ORDER BY 2,1
```

RESULT SCREENSHOT:



Row	OrderMonth	Payment_Type	OrderCount	Previous_Month_OrderCount	Increase_Percent
1	1	UPI	1715	null	null
2	2	UPI	1723	1715	0.47
3	3	UPI	1942	1723	12.71
4	4	UPI	1783	1942	-8.19
5	5	UPI	2035	1783	14.13
6	6	UPI	1807	2035	-11.2
7	7	UPI	2074	1807	14.78
8	8	UPI	2077	2074	0.14
9	9	UPI	903	2077	-56.52
10	10	UPI	1056	903	16.94

Results per page: 50 ▾ 1 – 48 of 48

INFERENCE:

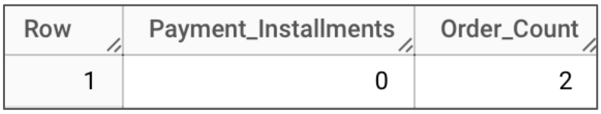
*With reference to payment type analysis, retrieved is the date on month-on-month analysis based on payment type for each month. The column **Increase_Percent** with positive value indicates the percentage increase in number of orders placed when compared to previous months order count using same payment method and negative value indicates downfall for that month.*

6. (2)

QUERY:

```
SELECT Payment_Installments, Order_Count
FROM
(SELECT
payment_installments AS Payment_Installments,
COUNT(order_id) AS Order_Count
FROM `target_business_case1.payments`
GROUP BY 1) a
WHERE a.Payment_Installments = 0
ORDER BY 1
```

RESULT SCREENSHOT:



Row	Payment_Installments	Order_Count
1	0	2

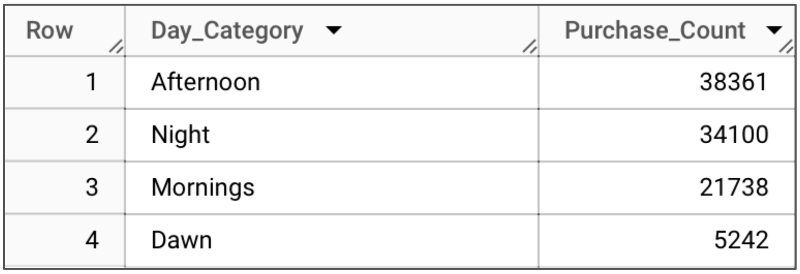
INFERENCE:

Based on the data analyzed, there are only 2 orders placed based on the payment installments that have been paid.

7. Actionable Insights & Recommendations

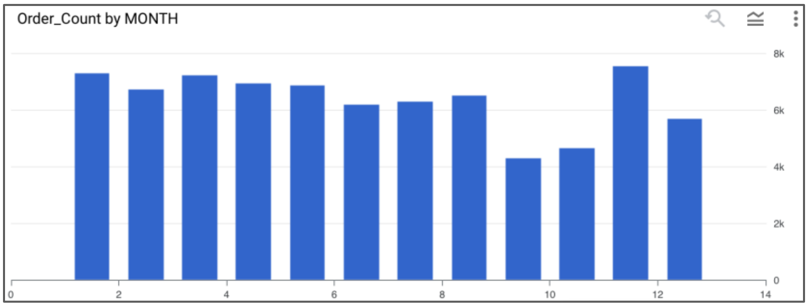
This report is about comprehensive analysis of Target's data from 2016 to 2018, aiming to understand the business trends and patterns. The analysis is done on various attributes, mainly aimed at purchase trend, customer distribution, sales percentage, freight costing, delivery patterns, etc. Here are certain insights and recommendations based on the data analyzed.

1. Insight: Most of the purchases were made during the afternoon and night.
   Recommendation:
   - The launch of any new product can be done during this peak purchase time of the day to seek more customers' attention and interest.
   - Introduce 'Happy Hours' discount during off sales daytime in Morning and Dawn



Row	Day_Category ▼	Purchase_Count ▼
1	Afternoon	38361
2	Night	34100
3	Mornings	21738
4	Dawn	5242

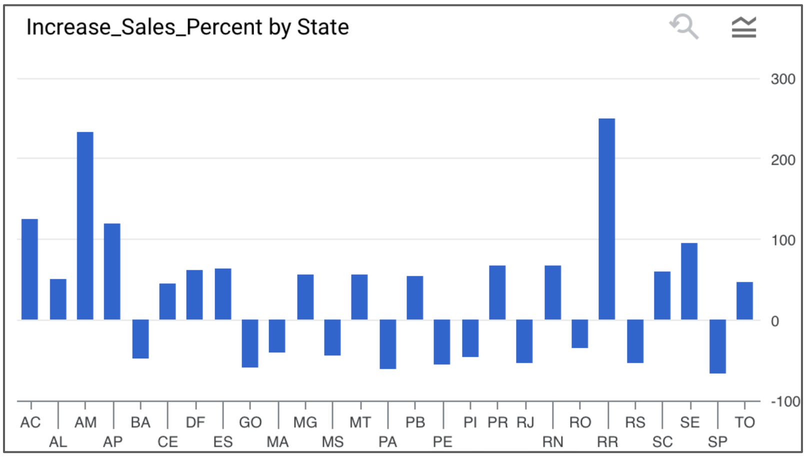
2. Insight: Order purchase counts dropped a lot from the month of June to October.
   Recommendation:
   - Launch sales like Winter Sales, Clearance Sales, Weekend Sale for the month, etc. to improve the business in these months.



3. Insight: For some states, overall sales went down when compared with the previous month's purchase sales.

   Recommendation:

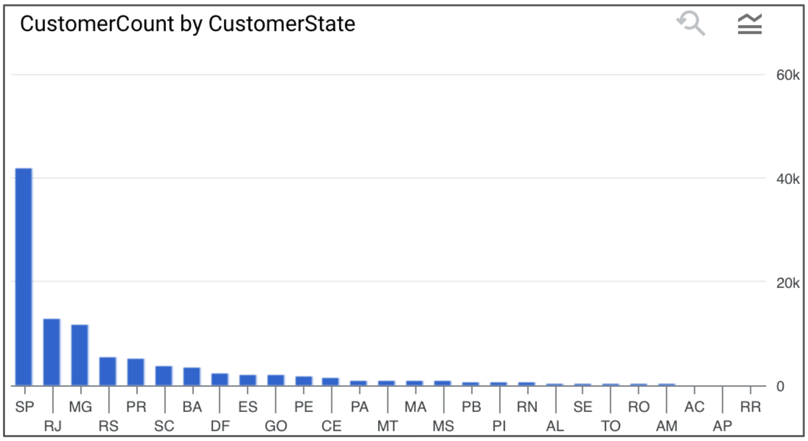
   - Suggest more products that meet the preferences of the customers.
   - Identify the challenge and focus more on marketing

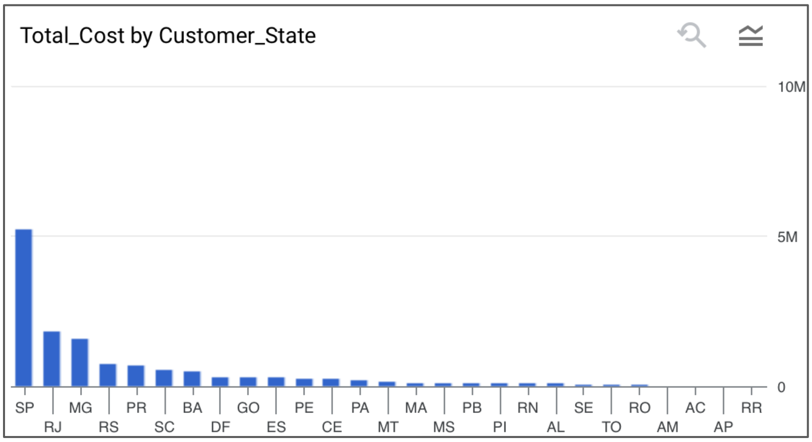


4. Insight: Count of customer each state is proportional to the total revenue generated by that state.

   Recommendations:

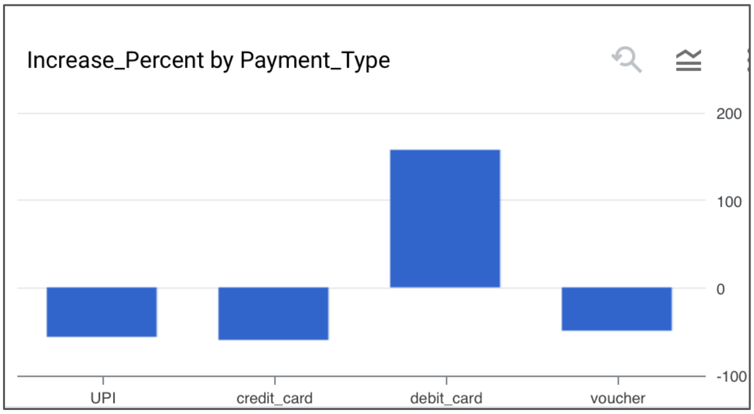
   - Promotion of new products in higher customer count states will have more reach and attention.
   - Introduce Membership program depending on each state revenue, with benefits like additional discounts, free shipping, etc. to have more repetitive business from loyal customers.
   - Review customer feedback from states with lower revenue and focus on areas for improvement.





5. Insight: Payment mode debit card has increased payment percent while other payment modes are down scaling.
   Recommendations:
   - Introduce minimum to maximum cash back offers on UPI payment mode
   - Card based discounts and additional offers when payment done through Credit Cards
   - Offer a voucher with date of expiry or wallet points as a first-purchase voucher or first-order-wallet points. Setting date of voucher expiry will set an urgency to made order soon.
   - When a new customer completes their first purchase transaction, they should be eligible to redeem the voucher or 10% of wallet points during their next checkout process.

   These strategies can encourage new customers to make their first purchase to avail benefits along with product purchase.


Increase_Percent by Payment_Type
UPI
credit_card
debit_card
voucher