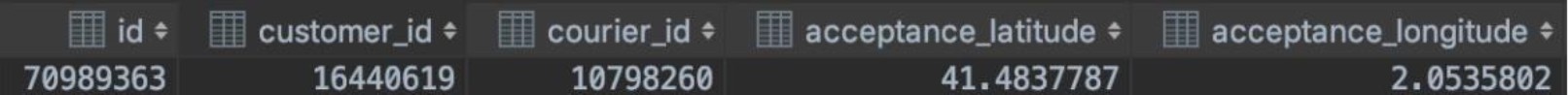
# Exercise 1: SQL

**Note:** The code must be written on Amazon Redshift SQL and must be scalable.

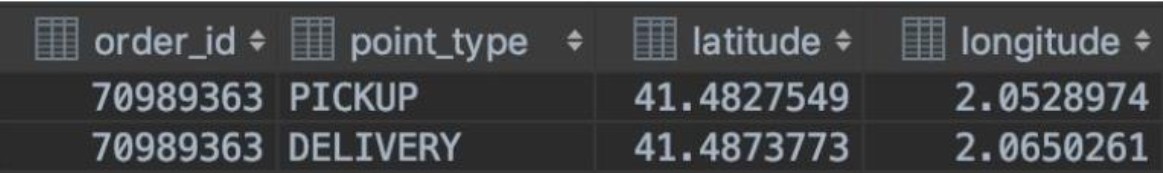
1. Let’s say you have two tables: ***orders*** and ***order\_poinĒs***.

Create an SQL query that shows the distance between the courier starting position and the pickup point, as well as the distance between the pickup point and the delivery point.

The ***orders*** table has 20M+ rows; here’s the ﬁrst row:



The ***order\_poinĒs*** table also has 40M+ rows. As FYI there are two types of point, ‘DELIVERY’ and ‘PICKUP’. Here’s an example:



Solution:

-- COURIER to PICKUP

WITH

    combined AS (

        SELECT

            t1.id,

            t1.customer\_id,

            t1.courier\_id,

            t1.acceptance\_latitude AS lat1,

            t1.acceptance\_longitude AS lon1,

            t2.point\_type,

            t2.latitude AS lat2,

            t2.longitude AS lon2

        FROM

            orders t1

            INNER JOIN order\_points t2 ON t1.id = t2.order\_id

    )

SELECT

    id,

    customer\_id,

    courier\_id,

    point\_type,

    lat1,

    lon1,

    lat2,

    lon2,

    (

        6371 \* acos(

            cos(radians (lat1)) \* cos(radians (lat2)) \* cos(radians (lon2) - radians (lon1)) + sin(radians (lat1)) \* sin(radians (lat2))

        )

    ) AS distance\_km

FROM

    Combined

-- PICKUP to DELIVERY

where

    point\_type = 'PICKUP';

WITH

    pickupdelivery AS (

        SELECT

            p.order\_id,

            p.latitude AS pickup\_lat,

            p.longitude AS pickup\_lon,

            d.latitude AS delivery\_lat,

            d.longitude AS delivery\_lon

        FROM

            order\_points p

            INNER JOIN order\_points d ON p.order\_id = d.order\_id

        WHERE

            p.point\_type = 'PICKUP'

            AND d.point\_type = 'DELIVERY'

    )

SELECT

    order\_id,

    pickup\_lat,

    pickup\_lon,

    delivery\_lat,

    delivery\_lon,

    (

        6371 \* acos(

            cos(radians (pickup\_lat)) \* cos(radians (delivery\_lat)) \* cos(radians (delivery\_lon) - radians (pickup\_lon)) + sin(radians (pickup\_lat)) \* sin(radians (delivery\_lat))

        )

    ) AS distance\_km

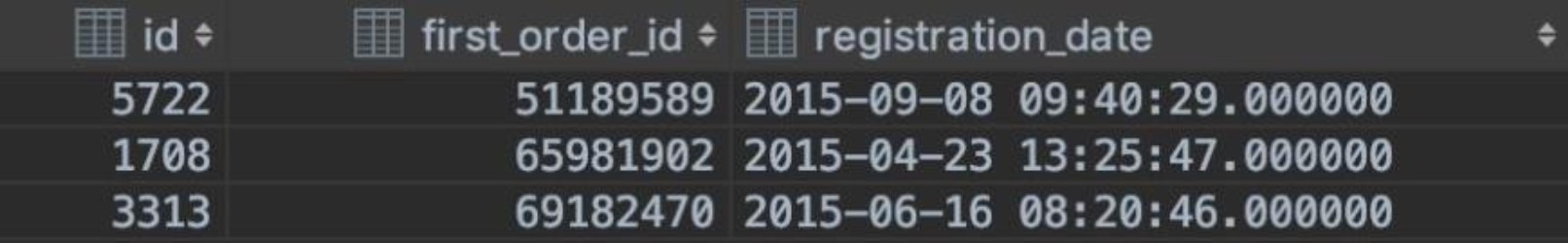
FROM

    PickupDelivery;

1. Build one SQL query to create a cohort of Signup to First Order and show the result.

The objective of this cohort is to see, out of the users that signed up in Week N, how many did their ﬁrst order in Week N+1, N+2, N+3...

The ***users*** table has 5M+ rows; here’s the ﬁrst three rows:



The ***orders*** table has 20M+ rows; here’s the ﬁrst row:



The output must be scalable for all weeks and does not require to be in a cohort format. The end user could potentially use the pivot function from Excel or Google sheets to do so.

Solution:

WITH SignupWeek AS (

    SELECT

        id AS customer\_id,

        registration\_date,

        DATE\_PART('week', registration\_date) AS signup\_week,

        DATE\_PART('year', registration\_date) AS signup\_year

    FROM

        Customer

),

FirstOrderWeek AS (

    SELECT

        c.id AS customer\_id,

        c.registration\_date,

        o.activation\_time,

        DATE\_PART('week', c.registration\_date) AS signup\_week,

        DATE\_PART('year', c.registration\_date) AS signup\_year,

        DATE\_PART('week', o.activation\_time) AS order\_week,

        DATE\_PART('year', o.activation\_time) AS order\_year

    FROM

        Customer c

    INNER JOIN

        Orders o ON c.first\_order\_id = o.id

)

SELECT

    sw.signup\_year,

    COUNT(CASE WHEN fw.order\_week = sw.signup\_week + 1 THEN fw.customer\_id END) AS week\_1,

    COUNT(CASE WHEN fw.order\_week = sw.signup\_week + 2 THEN fw.customer\_id END) AS week\_2,

    COUNT(CASE WHEN fw.order\_week = sw.signup\_week + 3 THEN fw.customer\_id END) AS week\_3,

    (week\_1+week\_2+week\_3) as  users\_ordered\_in\_3weeks

FROM

    SignupWeek sw

LEFT JOIN

    FirstOrderWeek fw ON sw.customer\_id = fw.customer\_id

GROUP BY

    sw.signup\_year

ORDER BY

    sw.signup\_year;

1. Build a SQL query that returns a table with the following ﬁelds:

## City Group

Is a construction from the city ﬁeld. We want the following groups:

* + - Group1 (contains Barcelona)
    - Group2 (contains Madrid)
    - Group3 (contains Valencia and Murcia)
    - Group4 (contains the rest of cities, but no Gen1 cities. A Gen1 city is a city where ALL its orders are Gen1)
* **Last Week Number of Orders** (closed week)
* **Week over Week Number of Orders** (The increase or decrease of Last week Number of Orders vs the previous)
* **Last Week Number of Registrations** (The number of user registrations in the app)

## Average Number of Food Orders by User Last Month

* **Last Month Number of Old Active Users** (number of old users that ordered last month. Old = user that did its ﬁrst order the previous month or before)

There are 2 table given:

* Orders:
  + id (one unique ID for row)
  + city
  + user\_id
  + Gen1 (1 if it’s a Gen1 order, 0 if it’s Gen2. A Gen1 order is an order that is delivered by the partner itself)
  + category (FOOD or GROCERIES)
  + order\_date (the date of the order)
* users:
  + id (one unique ID for row)
  + city
  + registration\_date (date of registration in the app)
  + ﬁrst\_order\_date (date of their ﬁrst order in the app)

Solution:

WITH

    CityGroup AS (

        SELECT DISTINCT

            city,

            CASE

                WHEN city = 'Barcelona' THEN 'Group1'

                WHEN city = 'Madrid' THEN 'Group2'

                WHEN city IN ('Valencia', 'Murcia') THEN 'Group3'

                ELSE 'Group4'

            END AS city\_group

        FROM

            Orders

    ),

    LastWeekOrders AS (

        SELECT

            c.city\_group,

            COUNT(\*) AS last\_week\_orders

        FROM

            Orders o

            JOIN CityGroup c ON o.city = c.city

        WHERE

            DATE\_PART ('week', o.order\_date) = DATE\_PART ('week', CURRENT\_DATE - INTERVAL '1 week')

            AND DATE\_PART ('year', o.order\_date) = DATE\_PART ('year', CURRENT\_DATE)

        GROUP BY

            c.city\_group

    ),

    WeekOverWeekOrders AS (

        SELECT

            c.city\_group,

            COUNT(\*) AS last\_week\_orders,

            LAG (COUNT(\*)) OVER (

                PARTITION BY

                    c.city\_group

                ORDER BY

                    DATE\_PART ('week', o.order\_date)

            ) AS prev\_week\_orders

        FROM

            Orders o

            JOIN CityGroup c ON o.city = c.city

        WHERE

            DATE\_PART ('week', o.order\_date) IN (

                DATE\_PART ('week', CURRENT\_DATE),

                DATE\_PART ('week', CURRENT\_DATE - INTERVAL '1 week')

            )

            AND DATE\_PART ('year', o.order\_date) = DATE\_PART ('year', CURRENT\_DATE)

        GROUP BY

            c.city\_group,

            DATE\_PART ('week', o.order\_date)

    ),

    LastWeekRegistrations AS (

        SELECT

            c.city\_group,

            COUNT(\*) AS last\_week\_registrations

        FROM

            Users u

            JOIN CityGroup c ON u.city = c.city

        WHERE

            DATE\_PART ('week', u.registration\_date) = DATE\_PART ('week', CURRENT\_DATE - INTERVAL '1 week')

            AND DATE\_PART ('year', u.registration\_date) = DATE\_PART ('year', CURRENT\_DATE)

        GROUP BY

            c.city\_group

    ),

    AvgFoodOrdersPerUserLastMonth AS (

        SELECT

            c.city\_group,

            AVG(

                CASE

                    WHEN o.category = 'FOOD' THEN 1

                    ELSE 0

                END

            ) AS avg\_food\_orders\_per\_user\_last\_month

        FROM

            Orders o

            JOIN CityGroup c ON o.city = c.city

        WHERE

            o.order\_date >= DATE\_TRUNC ('month', CURRENT\_DATE - INTERVAL '1 month')

            AND o.order\_date < DATE\_TRUNC ('month', CURRENT\_DATE)

        GROUP BY

            c.city\_group

    ),

    LastMonthOldActiveUsers AS (

        SELECT

            c.city\_group,

            COUNT(DISTINCT u.id) AS last\_month\_old\_active\_users

        FROM

            Users u

            JOIN Orders o ON u.id = o.user\_id

            JOIN CityGroup c ON u.city = c.city

        WHERE

            o.order\_date >= DATE\_TRUNC ('month', CURRENT\_DATE - INTERVAL '1 month')

            AND o.order\_date < DATE\_TRUNC ('month', CURRENT\_DATE)

            AND u.first\_order\_date < DATE\_TRUNC ('month', CURRENT\_DATE - INTERVAL '1 month')

        GROUP BY

            c.city\_group

    )

SELECT

    COALESCE(LastWeekOrders.city\_group, 'Group1') AS city\_group,

    COALESCE(LastWeekOrders.last\_week\_orders, 0) AS last\_week\_orders,

    COALESCE(WeekOverWeekOrders.prev\_week\_orders, 0) AS prev\_week\_orders,

    COALESCE(LastWeekRegistrations.last\_week\_registrations, 0) AS last\_week\_registrations,

    COALESCE(

        AvgFoodOrdersPerUserLastMonth.avg\_food\_orders\_per\_user\_last\_month,

        0

    ) AS avg\_food\_orders\_per\_user\_last\_month,

    COALESCE(

        LastMonthOldActiveUsers.last\_month\_old\_active\_users,

        0

    ) AS last\_month\_old\_active\_users

FROM

    LastWeekOrders

    FULL JOIN WeekOverWeekOrders ON LastWeekOrders.city\_group = WeekOverWeekOrders.city\_group

    FULL JOIN LastWeekRegistrations ON LastWeekOrders.city\_group = LastWeekRegistrations.city\_group

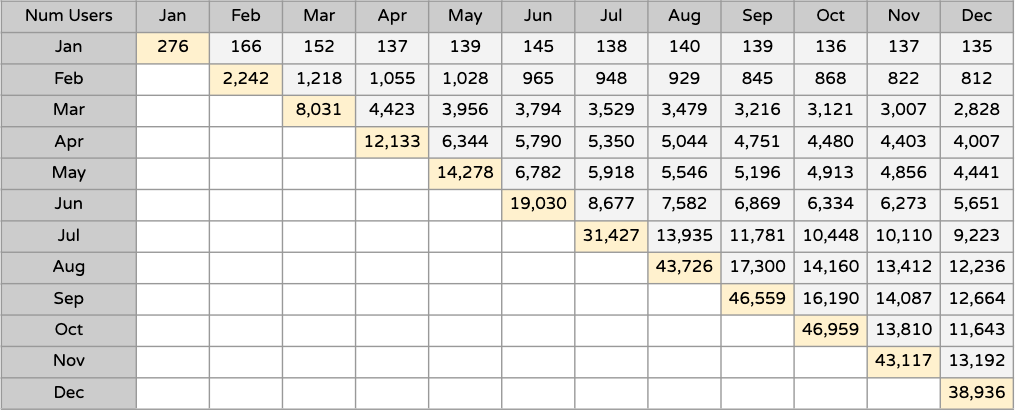
    FULL JOIN AvgFoodOrdersPerUserLastMonth ON LastWeekOrders.city\_group = AvgFoodOrdersPerUserLastMonth.city\_group

    FULL JOIN LastMonthOldActiveUsers ON LastWeekOrders.city\_group = LastMonthOldActiveUsers.city\_group

ORDER BY

    LastWeekOrders.city\_group;

# EXERCISE 2: New Customer cohorts

This chart contains raw information about the evolution of users over a year (user cohorts):

## Task:

1. List 3 metrics or ratios you would build with this data that you consider key to understand and manage the business
2. Deﬁne 2-3 actions based on this data that could help you improve your current performance

Solution:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Num Users | Jan | Feb | Mar | Apr | May | Jun | Jul | Aug | Sep | Oct | Nov | Dec |
| Jan | 276 | -39.9% | -8.4% | -9.9% | 1.5% | 4.3% | -4.8% | 1.4% | -0.7% | -2.2% | 0.7% | -1.5% |
| Feb |  | 2,242 | -45.7% | -13.4% | -2.6% | -6.1% | -1.8% | -2.0% | -9.0% | 2.7% | -5.3% | -1.2% |
| Mar |  |  | 8,031 | -44.9% | -10.6% | -4.1% | -7.0% | -1.4% | -7.6% | -3.0% | -3.7% | -6.0% |
| Apr |  |  |  | 12,133 | -47.7% | -8.7% | -7.6% | -5.7% | -5.8% | -5.7% | -1.7% | -9.0% |
| May |  |  |  |  | 14,278 | -52.5% | -12.7% | -6.3% | -6.3% | -5.4% | -1.2% | -8.5% |
| Jun |  |  |  |  |  | 19,030 | -54.4% | -12.6% | -9.4% | -7.8% | -1.0% | -9.9% |
| Jul |  |  |  |  |  |  | 31,427 | -55.7% | -15.5% | -11.3% | -3.2% | -8.8% |
| Aug |  |  |  |  |  |  |  | 43,726 | -60.4% | -18.2% | -5.3% | -8.8% |
| Sep |  |  |  |  |  |  |  |  | 46,559 | -65.2% | -13.0% | -10.1% |
| Oct |  |  |  |  |  |  |  |  |  | 46,959 | -70.6% | -15.7% |
| Nov |  |  |  |  |  |  |  |  |  |  | 43,117 | -69.4% |
| Dec |  |  |  |  |  |  |  |  |  |  |  | 38,936 |

Task 1

1. **Monthly Active Users (MAU)**: MAU provides insight into the overall reach and popularity of the business over time. Calculating the total number of unique users who engaged with the platform or service within a given month can help to track user growth and engagement trends.

1. **Retention Rate**: The retention rate helps assess the effectiveness of the business in retaining customers and can indicate user satisfaction, product stickiness, and long-term viability. Measuring the percentage of users who continue to use the platform or service over time will show the outliers and trends that will help to shift focus to problematic areas (periods).

1. **Churn Rate**:  Churn rate provides insights into customer attrition and the need for retention efforts. Calculating the percentage of users who stop using the platform or service within a specific period, typically on a monthly or quarterly basis. Identifying reasons for churn can help to develop strategies to improve user experience and reduce customer loss.

These three metrics can provide valuable insights into user behavior, engagement patterns, and the overall health of the business, enabling informed decision-making and strategic planning.

Task 2:

1. Enhance Retention Strategies: Given the variations in user numbers across months, it's essential to implement effective retention strategies to maintain and increase user engagement over time. Following are recommended actions to perform.

* **Personalized Communication**: Implement personalized communication strategies to engage users based on their preferences, behavior, and lifecycle stage. This could include targeted email campaigns, in-app messages, or push notifications tailored to individual user needs and interests.
* **Feature Enhancements**: Continuously improve the platform by adding new features or functionalities based on user feedback and behavior analysis. Enhancing the user experience can increase satisfaction and loyalty, leading to higher retention rates.
* **Reward Programs**: Introduce loyalty programs or incentives to reward and incentivize loyal users. Offer discounts, exclusive access to content, or loyalty points for regular engagement, referrals, or purchases to encourage continued usage.

1. Optimise Acquisition Channels: Analyse the performance of acquisition channels to identify the most effective channels for acquiring and retaining users. Actions to consider include:

* Channel Analysis: Evaluate the effectiveness of different acquisition channels (e.g., organic search, paid advertising, referral programs) in driving user growth and engagement. Allocate resources to channels with the highest return on investment (ROI) and optimize underperforming channels.
* Targeted Marketing Campaigns: Develop targeted marketing campaigns tailored to specific user segments and acquisition channels. Utilize data analytics to identify high-value user segments and create personalized messaging and offers to attract and retain these users.
* A/B Testing: Experiment with different messaging, creatives, and offers across acquisition channels to identify the most effective strategies for user acquisition and retention. Conduct A/B tests to optimize conversion rates and maximize ROI on marketing spend.

Implementing these actions can help improve user retention, drive sustainable growth, and optimize marketing efforts to maximize ROI and long-term profitability.

# Exercise 3: Partner OOH & promotion analysis

You need to assess the results of an OOH & promotion action with one of our partners. The product on promotion is KFC’s Streetwise 2. Here are the 4 questions for you to answer:

## Questions

1. Is the Streetwise 2 cannibalizing from other orders from KFC?
2. Does the campaign have positive ROI for Glovo and should Glovo repeat the campaign? Support the conclusion with numbers.
3. Should KFC repeat the campaign?
4. Assuming Glovo wanted to launch a new OOH campaign with KFC: what changes in media investment / promo funding would you recommend Glovo to make?

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | **Promo Period** | | | |  |  |  |  |
|  | **w1** | **w2** | **w3** | **w4** | **w5** | **w6** | **w7** | **w8** | **w9** | **w10** | **w11** | **w12** |
| Glovo New Customers | 6 215 | 5 530 | 5 408 | 5 453 | 5 728 | 5 630 | 6 765 | 5 651 | 5 463 | 5 724 | 5 761 | 5 858 |
| Glovo orders | 101 423 | 99 495 | 98 058 | 98 020 | 101 383 | 101 915 | 102 473 | 99 249 | 98 524 | 98 886 | 98 705 | 98 796 |
| Non-KFC  orders | 89 258 | 88 408 | 87 503 | 88 248 | 88 130 | 88 397 | 88 414 | 88 354 | 88 053 | 87 883 | 87 645 | 87 543 |
| KFC orders | 12 165 | 11 088 | 10 555 | 9 773 | 13 253 | 13 518 | 14 058 | 10 895 | 10 472 | 11 004 | 11 060 | 11 252 |
| KFC SW2  orders | 2 433 | 2 550 | 1 900 | 1 466 | 3 313 | 4 731 | 4 920 | 2 087 | 1 972 | 2 090 | 2 092 | 2 217 |

## Additional information for the case

1. KFC's objective is to increase sales by promoting their Streetwise 2 menu (SW2). 100% of the SW2 orders sold during the campaign period have a discount.
2. Glovo user LTV is €15.
3. KFC's AOV is of €15, and has a margin on its products of 13%. Assume the AOV is the same for all KFC products (including the SW2).
4. The discount is 7% of KFC's AOV. 30% of the discount is paid by Glovo, while KFC assumes the remaining 70%.
5. Additionally, Glovo supports the campaign with 10 billboards / week, with a weekly cost of €500/board. This is fully paid by Glovo. There is no OOH investment outside of promo weeks.

8. Glovo has an average investment of €2.5K / week in Performance Marketing for non-promo weeks. For promo weeks that increases to €3.0K, €3.25K, €3.5K, and €3.75K respectively.

7. Glovo's goal is to drive NC, while KFC's goal is to drive more KFC orders within Glovo.

Solution:

**1. Is the Streetwise 2 cannibalizing from other orders from KFC?**

To determine if the Streetwise 2 (SW2) promotion is cannibalizing other KFC orders, we can compare the number of KFC orders excluding SW2 before, during, and after the promotion period.

* **Total KFC Orders:**
  + Weeks 1-4 (before promotion): 12,165, 11,088, 10,555, 9,773
  + Weeks 5-8 (promotion period): 13,253, 13,518, 14,058, 10,895
  + Weeks 9-12 (after promotion): 10,472, 11,004, 11,060, 11,252
* **SW2 Orders:**
  + Weeks 1-4: 2,433, 2,550, 1,900, 1,466
  + Weeks 5-8: 3,313, 4,731, 4,920, 2,087
  + Weeks 9-12: 1,972, 2,090, 2,092, 2,217
* **Non-SW2 KFC Orders:**
  + Weeks 1-4: 12,165 - 2,433 = 9,732; 11,088 - 2,550 = 8,538; 10,555 - 1,900 = 8,655; 9,773 - 1,466 = 8,307
  + Weeks 5-8: 13,253 - 3,313 = 9,940; 13,518 - 4,731 = 8,787; 14,058 - 4,920 = 9,138; 10,895 - 2,087 = 8,808
  + Weeks 9-12: 10,472 - 1,972 = 8,500; 11,004 - 2,090 = 8,914; 11,060 - 2,092 = 8,968; 11,252 - 2,217 = 9,035

The observation show that the non-SW2 KFC orders remained relatively stable throughout the promotion period when compared to the periods before and after. Therefore, the data does not strongly suggest that SW2 is cannibalizing other KFC orders significantly.

1. **Does the campaign have a positive ROI for Glovo and should Glovo repeat the campaign?**

To evaluate the ROI, we need to consider the incremental New Customers (NCs) and the costs associated with the campaign.

**New Customers:**

* Incremental NCs during promotion weeks:
  + Weeks 1-4: 6,215, 5,530, 5,408, 5,453 (average = 5,652)
  + Weeks 5-8: 5,728, 5,630, 6,765, 5,651 (average = 5,944; increase = 5,944 - 5,652 = 292)
  + Weeks 9-12: 5,463, 5,724, 5,761, 5,858 (average = 5,702; increase = 5,702 - 5,652 = 50)

**Incremental Orders and Revenue:**

* Additional orders during promotion weeks:
  + Increase in SW2 orders: (3,313 + 4,731 + 4,920 + 2,087) - (2,433 + 2,550 + 1,900 + 1,466) = 12,051 - 8,349 = 3,702
* Revenue from additional SW2 orders:
  + Revenue: 3,702 \* €15 = €55,530

**Campaign Costs:**

* Discounts on SW2 orders (7% of AOV, 30% covered by Glovo):
  + Discount per order: €15 \* 7% = €1.05
  + Glovo's share: €1.05 \* 30% = €0.315
  + Total discount cost to Glovo: 3,702 \* €0.315 = €1,167.63
* Performance marketing cost increase: (€3,000 + €3,250 + €3,500 + €3,750) - 4 \* €2,500 = €13,500 - €10,000 = €3,500
* Billboard costs: 10 billboards/week \* €500/board \* 4 weeks = €20,000
* Total campaign cost for Glovo: €1,167.63 + €3,500 + €20,000 = €24,667.63

**ROI Calculation:**

* Additional NC LTV revenue: 292 \* €15 = €4,380
* Total revenue from additional SW2 orders: €55,530
* Total revenue generated: €55,530 + €4,380 = €59,910
* ROI = (Revenue - Cost) / Cost = (€59,910 - €24,667.63) / €24,667.63 ≈ 1.43 (143%)

The ROI is positive, suggesting Glovo should consider repeating the campaign.

**3. Should KFC repeat the campaign?**

KFC’s goal is to increase sales so we should examine the additional revenue and costs for KFC.

* Additional, SW2 orders: 3,702
* Revenue from additional SW2 orders: 3,702 \* €15 = €55,530
* Cost of discounts (70% of €1.05 per order): 3,702 \* €1.05 \* 70% = €2,693.49
* Margin on additional orders (13%): 3,702 \* €15 \* 13% = €7,214.10

**Profit Calculation:**

* Net profit: Revenue - Discount cost + Margin = €55,530 - €2,693.49 + €7,214.10 = €60,050.61

The campaign resulted in a substantial profit for KFC, indicating they should consider repeating it.

1. **Recommendations for Future Campaigns**

If Glovo and KFC were to launch a new OOH campaign, here are some suggestions:

* **Media Investment:**
  + Given the positive ROI, Glovo can maintain or slightly increase billboard investments if it drives sufficient awareness and NCs. Consider testing digital marketing channels alongside OOH to potentially lower costs and track effectiveness better.
* **Promo Funding:**
  + Evaluate the discount rate; while the current rate has worked, testing different discount levels may optimize the cost-benefit ratio. Consider a lower discount if the campaign remains effective.
* **Performance Marketing:**
  + Maintain the tiered approach but optimize spending based on performance metrics from the current campaign. If the highest spend level yields diminishing returns, reallocate those funds to more efficient channels.

Note: All corresponding graphs are in the presentation