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_poinÊs***.

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

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

id	customer_id	courier_id	acceptance_latitude	acceptance_longitude
70989363	16440619	10798260	41.4837787	2.0535802

The ***order_poinÊs*** table also has 40M+ rows. As FYI there are two types of point, 'DELIVERY' and 'PICKUP'. Here's an example:

order_id	point_type	latitude	longitude
70989363	PICKUP	41.4827549	2.0528974
70989363	DELIVERY	41.4873773	2.0650261

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;
```

---

2. 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 firstorder in Week N+1, N+2, N+3...

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

id	first_order_id	registration_date
5722	51189589	2015-09-08 09:40:29.000000
1708	65981902	2015-04-23 13:25:47.000000
3313	69182470	2015-06-16 08:20:46.000000

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

id	customer_id	activation_time
985	596	2015-03-15 21:10:34.000000

The output must be scalable for all weeks and does not require to be in a cohort format. The end usercould 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;
```

---

3. 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):

Num Users	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
Jan	276	166	152	137	139	145	138	140	139	136	137	135
Feb		2,242	1,218	1,055	1,028	965	948	929	845	868	822	812
Mar			8,031	4,423	3,956	3,794	3,529	3,479	3,216	3,121	3,007	2,828
Apr				12,133	6,344	5,790	5,350	5,044	4,751	4,480	4,403	4,007
May					14,278	6,782	5,918	5,546	5,196	4,913	4,856	4,441
Jun						19,030	8,677	7,582	6,869	6,334	6,273	5,651
Jul							31,427	13,935	11,781	10,448	10,110	9,223
Aug								43,726	17,300	14,160	13,412	12,236
Sep									46,559	16,190	14,087	12,664
Oct										46,959	13,810	11,643
Nov											43,117	13,192
Dec												38,936

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.

2. **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).

3. **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.

2. 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.

3. 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.

5. 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