Mastering Cohort Analysis with MySQL: A Comprehensive Guide

In the ever-evolving landscape of data-driven decision-making, understanding customer behavior is paramount for businesses striving to thrive. One of the most powerful tools in this arsenal is **cohort analysis**. This article delves deep into the concept of cohort analysis, illustrating how it can be implemented using MySQL queries. By the end, you'll not only grasp the theoretical underpinnings of cohort analysis but also gain practical insights into how it can address real-world business challenges.

Table of Contents

Introduction to Cohort Analysis

Cohort analysis is a subset of behavioral analytics that breaks down data into related groups, or cohorts, based on shared characteristics or experiences within a defined time-span. Instead of viewing all users as a single entity, cohort analysis allows businesses to track and compare groups over time, providing nuanced insights into trends, patterns, and behaviors.

For instance, a business might group customers based on the month they made their first purchase and then analyze how these groups behave over subsequent months. This approach can reveal critical information about customer retention, lifetime value, and the effectiveness of marketing strategies.

Importance and Real-World Applications

Understanding how different cohorts behave helps businesses:

- **Identify Trends:** Recognize patterns in customer behavior over time.
- **Measure Retention:** Determine how well customers are retained after their initial interaction.
- **Optimize Marketing:** Tailor marketing strategies based on the effectiveness seen across different cohorts.
- **Enhance Product Development:** Inform product improvements by understanding usage patterns.
- **Predict Revenue:** Forecast future revenue streams based on cohort performance.

Real-World Examples:

- **SaaS Companies:** Assessing how different sign-up cohorts adopt and utilize their software over time.
- **E-commerce Platforms:** Tracking purchasing behavior of customers who made their first purchase in a particular month.
- **Mobile Apps:** Analyzing user engagement and retention based on the week of app installation.

Setting Up the Data Schema

Before diving into cohort analysis, it's essential to set up the appropriate database schema. Below is the initial setup using MySQL:

```sql
-- Create a new schema named 'mysales'
CREATE SCHEMA mysales;

-- Switch to using the 'mysales' database
USE mysales;

-- Create a table named RETAIL with various columns to store retail data
CREATE TABLE IF NOT EXISTS RETAIL (
    InvoiceNo VARCHAR(10),
    StockCode VARCHAR(20),
    Description VARCHAR(100),
    Quantity DECIMAL(8,2),       -- Numbers with precision
    InvoiceDate VARCHAR(25),     -- Stored as string initially
    UnitPrice DECIMAL(8,2),      -- Price per unit with precision
    CustomerID BIGINT,           -- Large integer IDs for customers
    Country VARCHAR(25)
);
```

Explanation:

1. **Schema Creation (`mysales`):** In MySQL, a schema is synonymous with a database. Here, `mysales` serves as a container for our tables.
2. **Table `RETAIL`:** This table is designed to store transactional data, capturing essential details like invoice numbers, product codes, quantities, prices, customer IDs, and the country of purchase.

Loading Data into MySQL

Once the schema and table are set up, the next step is to populate the `RETAIL` table with data. This is achieved using the `LOAD DATA INFILE` command:

```
LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/Online Retail Data.csv'
INTO TABLE retail
FIELDS TERMINATED BY ','
OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY '\r\n'
IGNORE 1 ROWS
(InvoiceNo, StockCode, Description, Quantity, @date_str,
UnitPrice, CustomerID, Country)
SET InvoiceDate = STR_TO_DATE(@date_str, '%d/%m/%Y %H:%i');
```

Explanation:

- **`LOAD DATA INFILE`:** Imports data from a CSV file into the `RETAIL` table.
- **`FIELDS TERMINATED BY ','`:** Specifies that fields are separated by commas.
- **`OPTIONALLY ENCLOSED BY '"'`:** Handles fields that may be enclosed in double quotes.
- **`LINES TERMINATED BY '\r\n'`:** Denotes the end of a line in the CSV.
- **`IGNORE 1 ROWS`:** Skips the header row in the CSV file.
- **`@date_str`:** Temporarily holds the date string from the CSV.
- **`SET InvoiceDate = STR_TO_DATE(...)`:** Converts the string date into a `DATETIME` format for proper date handling in MySQL.

Performing Cohort Analysis: Queries Explained

The heart of this guide lies in performing cohort analysis using MySQL queries. We'll explore three distinct analyses:

1. **Invoice Counts by Cohort**
2. **Customer Counts by Cohort**
3. **Revenue by Cohort**

Each analysis builds upon the previous, layering insights to provide a comprehensive understanding of customer behavior.

1. Cohort Analysis Based on Invoice Counts

Objective: Determine how the number of invoices (transactions) evolves for each customer cohort over time.

Query Breakdown:

```sql
-- Step 1: Create a Common Table Expression (CTE) named CTE1 to prepare data
WITH CTE1 AS (
    SELECT
        InvoiceNo, CUSTOMERID,
        INVOICEDATE,
        ROUND(QUANTITY * UNITPRICE, 2) AS REVENUE
    FROM RETAIL
    WHERE CUSTOMERID IS NOT NULL
),

-- Step 2: Create CTE2 to calculate purchase and first purchase months
CTE2 AS (
    SELECT InvoiceNo, CUSTOMERID, INVOICEDATE,
        DATE_FORMAT(INVOICEDATE, '%Y-%m-01') AS PURCHASE_MONTH,
        DATE_FORMAT(MIN(INVOICEDATE) OVER (PARTITION BY CUSTOMERID ORDER BY INVOICEDATE), '%Y-%m-01') AS FIRST_PURCHASE_MONTH,
        REVENUE
    FROM CTE1
),

-- Step 3: Create CTE3 to determine cohort months
CTE3 AS (
    SELECT InvoiceNo, FIRST_PURCHASE_MONTH,
        CONCAT('Month_', TIMESTAMPDIFF(MONTH, FIRST_PURCHASE_MONTH, PURCHASE_MONTH)) AS COHORT_MONTH
    FROM CTE2
)

-- Step 4: Perform the final query to pivot and count invoices by cohort months
SELECT
    FIRST_PURCHASE_MONTH,
    SUM(CASE WHEN COHORT_MONTH = 'Month_0' THEN 1 ELSE 0 END) AS Month_0,
    SUM(CASE WHEN COHORT_MONTH = 'Month_1' THEN 1 ELSE 0 END) AS Month_1,
    -- Repeat SUM(CASE ...) for each subsequent month up to Month_12
```

```
    SUM(CASE WHEN COHORT_MONTH = 'Month_12' THEN 1 ELSE 0 END) AS Month_12
FROM CTE3
GROUP BY FIRST_PURCHASE_MONTH
ORDER BY FIRST_PURCHASE_MONTH;
```

Step-by-Step Explanation:

1. **CTE1 – Data Preparation:**
   - **Selecting Relevant Columns:** Extracts `InvoiceNo`, `CUSTOMERID`, `INVOICEDATE`, and calculates `REVENUE` as `QUANTITY * UNITPRICE`.
   - **Filtering:** Excludes records where `CUSTOMERID` is `NULL`, ensuring analysis only on valid customers.
2. **CTE2 – Determining Purchase Context:**
   - **`PURCHASE_MONTH`:** Formats `INVOICEDATE` to the first day of the purchase month (e.g., '2023-07-01').
   - **`FIRST_PURCHASE_MONTH`:** Identifies the first purchase month for each customer using a window function (`MIN(INVOICEDATE) OVER (...)`), ensuring all transactions are linked to the customer's initial cohort.
3. **CTE3 – Calculating Cohort Months:**
   - **`COHORT_MONTH`:** Calculates the difference in months between the `FIRST_PURCHASE_MONTH` and the `PURCHASE_MONTH`. This indicates how many months have passed since the customer's first purchase, labeled as 'Month_0' (initial month), 'Month_1' (one month after), and so on.
4. **Final Query – Pivoting Data:**
   - **Aggregation:** Groups data by `FIRST_PURCHASE_MONTH` (the cohort) and counts the number of invoices in each subsequent `COHORT_MONTH`.
   - **Conditional Summation:** Uses `SUM(CASE WHEN ...)` to pivot rows into columns, representing each month in the cohort's lifecycle.

Outcome:

The resulting table displays each customer cohort (based on their first purchase month) alongside the count of invoices they generated in each subsequent month up to twelve months. This provides a clear view of purchasing activity trends over time.

2. Cohort Analysis Based on Customer Counts

Objective: Assess customer retention by counting distinct customers in each cohort across subsequent months.

Query Breakdown:

```sql
-- Step 1: Create a Common Table Expression (CTE) named CTE1 to prepare data
WITH CTE1 AS (
    SELECT
        InvoiceNo, CUSTOMERID,
        INVOICEDATE,
        ROUND(QUANTITY * UNITPRICE, 2) AS REVENUE
    FROM RETAIL
    WHERE CUSTOMERID IS NOT NULL
),

-- Step 2: Create CTE2 to calculate purchase and first purchase months
CTE2 AS (
    SELECT InvoiceNo, CUSTOMERID, INVOICEDATE,
        DATE_FORMAT(INVOICEDATE, '%Y-%m-01') AS PURCHASE_MONTH,
        DATE_FORMAT(MIN(INVOICEDATE) OVER (PARTITION BY CUSTOMERID ORDER BY INVOICEDATE), '%Y-%m-01') AS FIRST_PURCHASE_MONTH,
        REVENUE
    FROM CTE1
),

-- Step 3: Create CTE3 to determine cohort months
CTE3 AS (
    SELECT CUSTOMERID, FIRST_PURCHASE_MONTH,
        CONCAT('Month_', TIMESTAMPDIFF(MONTH, FIRST_PURCHASE_MONTH, PURCHASE_MONTH)) AS COHORT_MONTH
    FROM CTE2
)

-- Final Query: Count distinct customers in each cohort for subsequent months
SELECT FIRST_PURCHASE_MONTH AS Cohort,
    COUNT(DISTINCT IF(COHORT_MONTH = 'Month_0', CUSTOMERID, NULL)) AS Month_0,
    COUNT(DISTINCT IF(COHORT_MONTH = 'Month_1', CUSTOMERID, NULL)) AS Month_1,
    -- Repeat COUNT(DISTINCT IF ...) for each subsequent month up to Month_12
    COUNT(DISTINCT IF(COHORT_MONTH = 'Month_12', CUSTOMERID, NULL)) AS Month_12
```

```
FROM CTE3
GROUP BY FIRST_PURCHASE_MONTH
ORDER BY FIRST_PURCHASE_MONTH;
```

Step-by-Step Explanation:

1. **CTE1 & CTE2 – Data Preparation:**
   - Similar to the first analysis, these CTEs prepare the data by selecting relevant columns and determining purchase contexts.
2. **CTE3 – Calculating Cohort Months:**
   - **Focus Shift:** Unlike the first analysis, this CTE focuses on `CUSTOMERID` instead of `InvoiceNo`, setting the stage for distinct customer counts.
3. **Final Query – Counting Distinct Customers:**
   - **Distinct Counts:** Utilizes `COUNT(DISTINCT IF(...))` to ensure each customer is counted only once per cohort month.
   - **Conditional Counting:** Similar to the first query, it conditionally counts customers based on their `COHORT_MONTH`, providing a month-by-month retention snapshot.

Outcome:

The resulting table showcases each cohort's size and how many distinct customers remain active in each subsequent month. This is pivotal for understanding customer retention rates and identifying potential drop-off points.

3. Cohort Analysis Based on Revenue

Objective: Analyze revenue trends by cohort, observing how revenue contribution evolves over time.

Final Query for Revenue Cohort Analysis:

```sql
-- Step 1: Create a Common Table Expression (CTE) named CTE1 to calculate revenue from valid transactions
WITH CTE1 AS (
    SELECT
        CUSTOMERID,
        INVOICEDATE,
        ROUND(QUANTITY * UNITPRICE, 0) AS REVENUE
    FROM RETAIL
    WHERE CUSTOMERID IS NOT NULL
),

-- Step 2: Create CTE2 to calculate purchase and first purchase months
CTE2 AS (
    SELECT
        CUSTOMERID,
        INVOICEDATE,
        DATE_FORMAT(INVOICEDATE, '%Y-%m-01') AS PURCHASE_MONTH,
        DATE_FORMAT(MIN(INVOICEDATE) OVER (PARTITION BY CUSTOMERID ORDER BY INVOICEDATE), '%Y-%m-01') AS FIRST_PURCHASE_MONTH,
        REVENUE
    FROM CTE1
),

-- Step 3: Create CTE3 to calculate cohort months
CTE3 AS (
    SELECT
        FIRST_PURCHASE_MONTH AS Cohort,
        CONCAT('Month_', TIMESTAMPDIFF(MONTH, FIRST_PURCHASE_MONTH, PURCHASE_MONTH)) AS COHORT_MONTH,
        REVENUE
    FROM CTE2
)

-- Final Query: Calculate the total revenue for each cohort month using conditional aggregation
SELECT
```

```
    Cohort,
    SUM(CASE WHEN COHORT_MONTH = 'Month_0' THEN REVENUE ELSE 0 END) AS Month_0,
    SUM(CASE WHEN COHORT_MONTH = 'Month_1' THEN REVENUE ELSE 0 END) AS Month_1,
    -- Repeat SUM(CASE ...) for each subsequent month up to Month_12
    SUM(CASE WHEN COHORT_MONTH = 'Month_12' THEN REVENUE ELSE 0 END) AS Month_12
FROM CTE3
GROUP BY Cohort
ORDER BY Cohort;
```

Step-by-Step Explanation:

1. **CTE1 – Revenue Calculation:**
   - **Precision Adjustment:** Rounds `REVENUE` to zero decimal places for simplicity.
2. **CTE2 – Determining Purchase Context:**
   - Mirrors previous analyses, setting up `PURCHASE_MONTH` and `FIRST_PURCHASE_MONTH`.
3. **CTE3 – Calculating Cohort Months:**
   - **Renaming:** `FIRST_PURCHASE_MONTH` is aliased as `Cohort` for clarity.
   - **Cohort Month Calculation:** Determines the number of months elapsed since the cohort's first purchase.
4. **Final Query – Aggregating Revenue:**
   - **Conditional Summation:** Uses `SUM(CASE WHEN ...)` to total revenue for each `COHORT_MONTH`.
   - **Pivoting:** Converts rows into columns representing each month in the cohort lifecycle.

Outcome:

This query produces a table that details the revenue generated by each cohort in each subsequent month. It offers invaluable insights into revenue retention and growth, helping businesses understand which cohorts are most profitable over time.

Insights Derived from Cohort Analysis

Applying cohort analysis through these queries can unveil several critical business insights:

1. **Customer Retention Rates:**
   - By analyzing how many customers remain active over time, businesses can assess the effectiveness of their retention strategies.
2. **Revenue Growth Patterns:**
   - Understanding how revenue evolves across cohorts helps in forecasting and identifying high-value customer segments.
3. **Effectiveness of Marketing Campaigns:**
   - Comparing cohorts from different time periods can reveal the impact of specific marketing efforts on customer acquisition and retention.
4. **Product Adoption Trends:**
   - Tracking how different cohorts adopt products can guide product development and feature prioritization.
5. **Churn Identification:**
   - Identifying when and why customers stop interacting can inform strategies to reduce churn.

Practical Application Scenario:

Imagine an e-commerce company launching a new product in January. By performing cohort analysis, they observe that January's cohort maintains high purchasing activity even six months later. In contrast, cohorts from other months show a rapid decline in activity. This insight indicates that the January launch was particularly successful, possibly due to effective marketing or favorable market conditions. The company can then replicate the strategies used in January for future product launches.

Conclusion

Cohort analysis is a potent method for dissecting and understanding customer behavior over time. By leveraging MySQL's powerful querying capabilities, businesses can perform detailed cohort analyses to uncover trends, measure retention, and drive strategic decisions. Whether it's counting invoices, tracking distinct customers, or analyzing revenue streams, cohort analysis provides a nuanced view that raw aggregated data often obscures.

Implementing the queries outlined in this guide equips you with the tools to transform transactional data into actionable insights. As businesses continue to navigate competitive landscapes, the ability to understand and respond to customer behavior through cohort analysis becomes increasingly invaluable.

Harness the power of cohort analysis to not only observe but also anticipate customer needs, fostering sustained business growth and enduring customer relationships.