Marketing Funnel Analytics (SQLite)

Author: Hatem Elgenedy

README

Project Overview

This project analyzes a marketing funnel using SQLite to understand how users move through the key stages Visit → Signup → Purchase. The objective is to measure funnel performance, conversion efficiency, and revenue contribution by marketing channel using clean, reusable SQL logic.

Business Questions

- How many users visit, sign up, and purchase?
- What are the conversion rates between funnel stages?
- Which marketing channels generate the most revenue?
- Which KPIs are ready for dashboards?

Dataset

Table: marketing_events

Each row represents a user interaction within the marketing funnel.

Key Columns

- user_id – unique user identifier
- event_date – date of the event
- event_type – visit, signup, or purchase
- channel – acquisition channel
- revenue – purchase revenue (0 for non-purchase events)

Data Model (Schema)

- Table: marketing_events
- Views: v_funnel_kpis, v_revenue_by_channel
- KPI Table: kpi_summary

Key KPIs

- Visitors
- Signups
- Purchasers
- Visit → Signup Conversion Rate
- Visit → Purchase Conversion Rate
- Total Revenue
- Revenue by Channel
- Average Order Value (AOV)

SQL Logic

- CTEs used for funnel calculations
- Aggregations using COUNT, SUM, and AVG
- NULLIF used to prevent divide-by-zero errors
- Views created for reuse in BI tools
- KPI summary table designed for dashboards

Tools Used

SQLite, DB Browser for SQLite

How to Run

- Open the database in DB Browser for SQLite
- Execute marketing_funnel_project.sql
- Run: SELECT * FROM kpi_summary;

Key Takeaway

This project demonstrates how SQL can be used to design a full analytics workflow—from raw event data to dashboard-ready KPIs—using clean schema design and production-style SQL.

SQL Code

```
CREATE TABLE marketing_events (
  event_id   INTEGER PRIMARY KEY AUTOINCREMENT,
  user_id    INTEGER NOT NULL,
  event_date TEXT    NOT NULL,
  event_type TEXT    NOT NULL,
  channel    TEXT    NOT NULL,
  revenue    REAL    NOT NULL DEFAULT 0
);

INSERT INTO marketing_events (user_id, event_date, event_type, channel,
revenue) VALUES
(1, '2025-01-01', 'visit',    'Organic',  0),
(1, '2025-01-02', 'signup',   'Organic',  0),
(1, '2025-01-05', 'purchase', 'Organic', 120),
(2, '2025-01-03', 'visit',    'Paid',     0),
(2, '2025-01-10', 'purchase', 'Paid',     80);

CREATE VIEW v_funnel_kpis AS
WITH funnel AS (
  SELECT
    COUNT(DISTINCT CASE WHEN event_type = 'visit' THEN user_id END)    AS
visitors,
    COUNT(DISTINCT CASE WHEN event_type = 'signup' THEN user_id END)   AS
signups,
    COUNT(DISTINCT CASE WHEN event_type = 'purchase' THEN user_id END) AS
purchasers
  FROM marketing_events
)
SELECT
  visitors,
  signups,
  purchasers,
  ROUND(1.0 * signups / NULLIF(visitors, 0), 2)    AS visit_to_signup,
  ROUND(1.0 * purchasers / NULLIF(visitors, 0), 2) AS visit_to_purchase
FROM funnel;

CREATE VIEW v_revenue_by_channel AS
SELECT
  channel,
  COUNT(*)     AS purchases,
  SUM(revenue) AS total_revenue,
  AVG(revenue) AS avg_order_value
FROM marketing_events
WHERE event_type = 'purchase'
GROUP BY channel;

CREATE TABLE kpi_summary (
  metric TEXT PRIMARY KEY,
  value  REAL
);

INSERT INTO kpi_summary (metric, value)
SELECT 'Visitors', visitors FROM v_funnel_kpis;
```

```sql
INSERT INTO kpi_summary (metric, value)
SELECT 'Signups', signups FROM v_funnel_kpis;

INSERT INTO kpi_summary (metric, value)
SELECT 'Purchasers', purchasers FROM v_funnel_kpis;

INSERT INTO kpi_summary (metric, value)
SELECT 'Visit_to_Signup', visit_to_signup FROM v_funnel_kpis;

INSERT INTO kpi_summary (metric, value)
SELECT 'Visit_to_Purchase', visit_to_purchase FROM v_funnel_kpis;

INSERT INTO kpi_summary (metric, value)
SELECT 'Total_Revenue', SUM(total_revenue) FROM v_revenue_by_channel;

SELECT * FROM kpi_summary;
```