

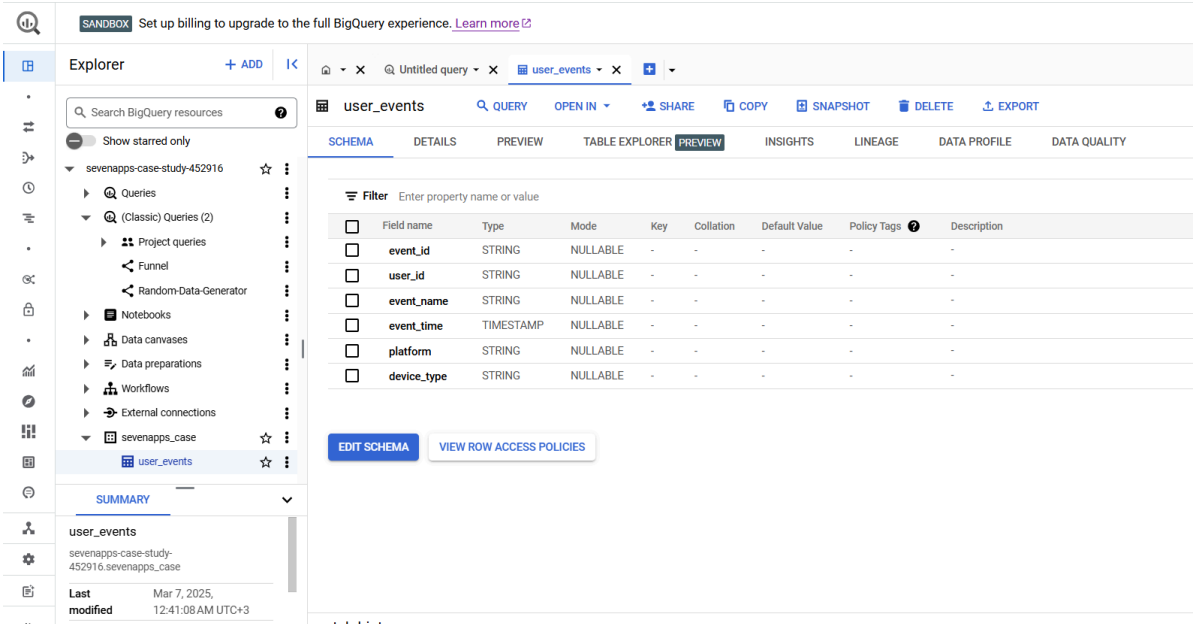


# SevenApps & Oğuzhan Gündüz

## Case Study Q1: SQL Implementation

### 1. Soruyu Hatırlayalım

- PageView → Download → Install dönüşüm oranlarını gösteren bir funnel analizi yapmamız isteniyor fakat elimizde örnek bir veri yok ve veri setini bizim random şekilde ürettirmemiz isteniyor.
- Veri setini oluştururken özgür olarak istediğimiz dili kullanabiliyoruz. fakat belirli bir kural dahilinde olması bekleniyor. Bu bağlamda oluşturacağımız `user_events` tablosunun aşağıdaki gibi olmasını istiyoruz.
  - `event_id` (string)
  - `user_id` (string)
  - `event_name` (string) - Olası değerler: 'PageView', 'Download', 'Install', 'Purchase'
  - `platform` (string) - Olası değerler: ios ve android
  - `device_type` (string)
  - `timestamp` (timestamp)
- Bu bağlamda oluşturacağım tablo için SQL dilini kullanmayı tercih ettim çünkü aktif iş hayatımda en çok SQL kullandım ve kullanıyorum.
  - Yazacağım sorgunun takibini yapabilmek ve hatalarımı anlamak için sorgumu BigQuery Sandbox versiyonunda geliştirdim. bu sayede adım adım yaptığım değişikliklerin `user_events` tablosuna etkilerini takip edebildim ve işin sonunda içime sinen tabloyu oluşturabildim.



- Oluşturduğum sorguya ilettiğim Zip dosyasındaki Q1 klasörü içindeki **“dataset\_creator.sql”** dan ulaşabilirsiniz.

## Veri Oluşturma Stratejim

Veri setini oluşturmak için aşağıdaki adımları izledim:

### 1. Kullanıcı Temeli Oluşturma:

```
CREATE OR REPLACE TABLE `sevenapps-case-study-452916.sevenapps_
case.user_events` AS
WITH users_base AS (
  SELECT
    user_seq,
    TO_HEX(MD5(CAST(user_seq AS STRING))) AS user_id,
    MOD(ABS(FARM_FINGERPRINT(CAST(user_seq AS STRING))), 30) A
S random_days,
    MOD(ABS(FARM_FINGERPRINT(CAST(user_seq AS STRING))), 1000)
/ 1000.0 AS random_val,
    TIMESTAMP_SUB(CURRENT_TIMESTAMP(),
```

```
INTERVAL MOD(ABS(FARM_FINGERPRINT(CAST(user_seq AS STRING))), 30) DAY) AS first_event_time
FROM UNNEST(GENERATE_ARRAY(1, 10000)) AS user_seq
),
```

- Toplam 10.000 kullanıcı için veri oluşturdum.
- Her kullanıcı için benzersiz bir ID oluşturmak için MD5 hash fonksiyonunu kullandım.

#### Platform ve Cihaz Dağılımı:

- Global pazar paylarını yansıtmak için Android'e %63, iOS'a %37 ağırlık verdim.
- Her kullanıcıya bir cihaz tipi atadım. (iPhone, iPad, Samsung, Pixel, vb.)

#### Funnel Aşamaları Oluşturma:

```
user_funnel AS (
  SELECT
    ...
    -- Her aşama için geçiş olasılıkları
    TRUE AS will_pageview, -- Herkes PageView görür
    MOD(ABS(FARM_FINGERPRINT(CONCAT(user_id, '1'))), 100) < 70 AS
will_download
    ...
)
```

- Her kullanıcının her bir aşamadan geçme olasılığını belirledim:
  - PageView → Download: %70
  - Download → Install: %80
  - Install → SignUp: %75
  - SignUp → Subscription: %30

#### Zaman Tutarlılığı Sağlama:

```
event_times AS (
  ...
  -- Her kullanıcı için tutarlı zaman hesaplamaları
  TIMESTAMP_ADD(first_event_time, INTERVAL CAST(...) HOUR) AS download_time
  ...
)
```

- 72 saat koşulunu test edebilmek için event'ler arası zaman farkını kontrollü şekilde oluşturdum.
- Bir event'in zamanı, her zaman önceki event'in zamanından sonra olacak şekilde tasarladım.

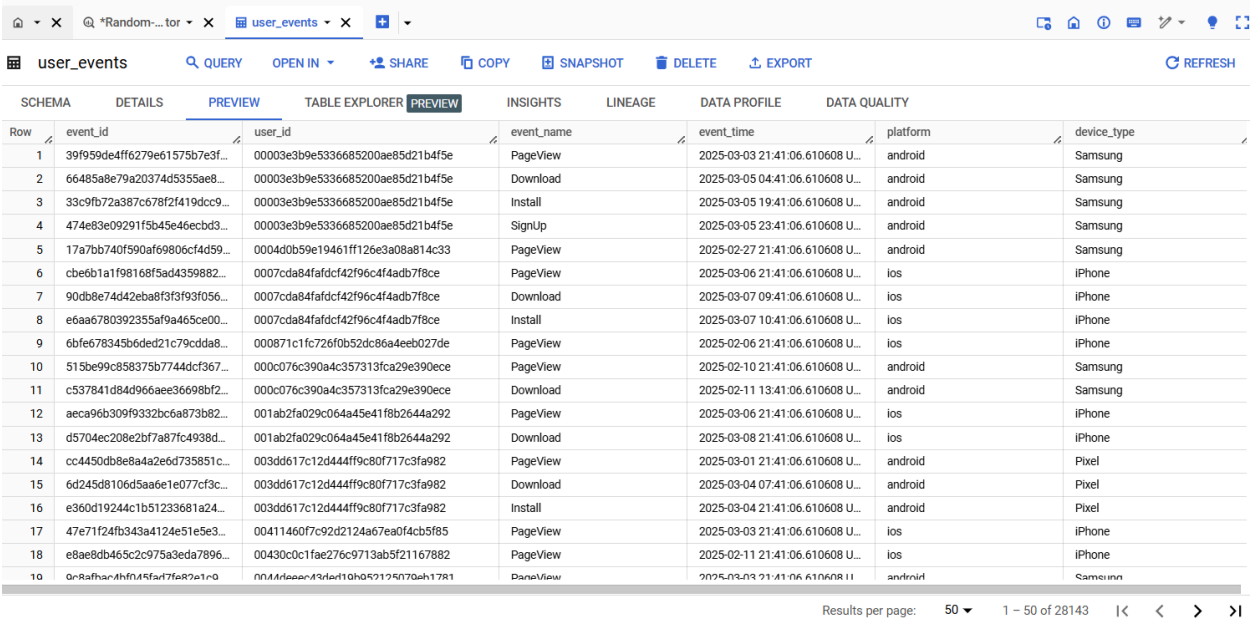
Olayları Birleştirme:

```
events AS (  
  -- Beş farklı event tipini birleştir  
  SELECT ... FROM ... WHERE will_pageview  
  UNION ALL  
  SELECT ... FROM ... WHERE will_download  
)
```

- Her kullanıcının ilgili funnel aşamalarını tekil olaylar halinde oluşturdum.
- Her olay için benzersiz bir ID ve gerçekçi zaman damgaları ekledim.

Sonuçta

`user_events` tablosunu istenen şekilde oluşturdum.



SCHEMA	DETAILS	PREVIEW	TABLE EXPLORER	PREVIEW	INSIGHTS	LINEAGE	DATA PROFILE	DATA QUALITY
Row	event_id	user_id	event_name	event_time	platform	device_type		
1	39f959de4ff6279e61575b7e3f...	00003e3b9e5336685200ae85d21b4f5e	PageView	2025-03-03 21:41:06.610608 U...	android	Samsung		
2	66485a8e79a20374d5355ae8...	00003e3b9e5336685200ae85d21b4f5e	Download	2025-03-05 04:41:06.610608 U...	android	Samsung		
3	33c9fb72a387c678f2f419d0c9...	00003e3b9e5336685200ae85d21b4f5e	Install	2025-03-05 19:41:06.610608 U...	android	Samsung		
4	474e83e09291f5b45e46ecbd3...	00003e3b9e5336685200ae85d21b4f5e	SignUp	2025-03-05 23:41:06.610608 U...	android	Samsung		
5	17a7bb740f590af69806cf4d59...	0004d0b59e19461ff126e3a08a814c33	PageView	2025-02-27 21:41:06.610608 U...	android	Samsung		
6	cbeeb1a1f98168f5ad4359882...	0007cda84fafdcf42f96c4f4adb7f8ce	PageView	2025-03-06 21:41:06.610608 U...	ios	iPhone		
7	90db8e74d42eba8f3f3f93f056...	0007cda84fafdcf42f96c4f4adb7f8ce	Download	2025-03-07 09:41:06.610608 U...	ios	iPhone		
8	e6aa6780392355af9a465ce00...	0007cda84fafdcf42f96c4f4adb7f8ce	Install	2025-03-07 10:41:06.610608 U...	ios	iPhone		
9	6bfe678345b6ded21c79cdda8...	000871c1fc726f0b52dc86a4eeb027de	PageView	2025-02-06 21:41:06.610608 U...	ios	iPhone		
10	515be99c858375b7744dcf367...	000c076c390a4c357313fca29e390ece	PageView	2025-02-10 21:41:06.610608 U...	android	Samsung		
11	c537841d84d966aee36698bf2...	000c076c390a4c357313fca29e390ece	Download	2025-02-11 13:41:06.610608 U...	android	Samsung		
12	aeca96b309f9332bc6a873b82...	001ab2fa029c064a45e41f8b2644a292	PageView	2025-03-06 21:41:06.610608 U...	ios	iPhone		
13	d5704ec208e2bf7a87fc4938d...	001ab2fa029c064a45e41f8b2644a292	Download	2025-03-08 21:41:06.610608 U...	ios	iPhone		
14	cc4450db8e8a4a2e6d735851c...	003dd617c12d444ff9c80f717c3fa982	PageView	2025-03-01 21:41:06.610608 U...	android	Pixel		
15	6d245d8106d5aa6e1e077cf3c...	003dd617c12d444ff9c80f717c3fa982	Download	2025-03-04 07:41:06.610608 U...	android	Pixel		
16	e36dd19244c1b51233681a24...	003dd617c12d444ff9c80f717c3fa982	Install	2025-03-04 21:41:06.610608 U...	android	Pixel		
17	47e71f24fb343a4124e51e5e3...	00411460f7c92d2124a67ea0f4cb5f85	PageView	2025-03-03 21:41:06.610608 U...	ios	iPhone		
18	e8ae8db465c2c975a3eda7896...	00430c0c1fae276c9713ab5f21167882	PageView	2025-02-11 21:41:06.610608 U...	ios	iPhone		
19	0a8afba44bf8f45fa7f5a72e1e0...	004AAd5eeef3fde410b0952157570aeh17R1	PageView	2025-03-03 21:41:06.610608 U...	android	Sameinn		

- Tablom 10.000 unique kullanıcının 28.143 adet event'ini içermektedir.
- Oluşturduğum tabloya yine Q1 klasörü içindeki **"user\_events.csv"** üzerinden göz atabilirsiniz.

Funnel Analizi

Veri setini oluşturduktan sonra, istenen funnel analizini gerçekleştirmek için şu yaklaşımı kullandım:

Sıralamalı Event Analizi:

```
WITH user_events_with_next AS (  
  SELECT  
    user_id,  
    platform,  
    event_type,  
    event_time,  
    LEAD(event_type) OVER (PARTITION BY user_id ORDER BY event_tim
```

```
e) AS next_event_type,  
    LEAD(event_time) OVER (PARTITION BY user_id ORDER BY event_time  
e) AS next_event_time  
FROM `sevenapps-case-study-452916.sevenapps_case.user_events`  
ORDER BY user_id, event_time  
)
```

- LEAD() pencere fonksiyonunu kullanarak her kullanıcının bir sonraki eventini belirledim
- Bu yaklaşım, olaylar arasındaki zamansal ilişkiyi korumamı sağladı

## 72 Saat Koşulu Uygulaması:

```
TIMESTAMP_DIFF(next_event_time, event_time, HOUR) <= 72
```

- Her dönüşüm için 72 saat içinde gerçekleşme koşulunu ekledim
- TIMESTAMP\_DIFF() fonksiyonunu kullanarak saat cinsinden zaman farkını hesapladım

## Dönüşüm Oranları Hesaplaması:

```
ROUND(SAFE_DIVIDE(download_to_install_count, pageview_to_download_  
count) * 100, 2) AS download_to_install_rate
```

- SAFE\_DIVIDE() kullanarak olası sıfıra bölme hatalarını önledim
- Hem adımlar arası (örn. Download → Install) hem de toplam (örn. PageView → Install) dönüşüm oranlarını hesapladım

## Platform Bazlı Segmentasyon:

```
GROUP BY platform  
ORDER BY platform
```

- Tüm metrikleri platform bazında (iOS/Android) grupladım

## Sonuç Olarak

- Platform bazında toplam unique user sayısını,
- pageview sayısını,
- 72 saat içindeki pageview'den download'a dönme sayısını ve diğer aşamaların, aşamalar arası dönüşüm sayılarını,
- 72 saat'de aşamalar arası dönüşüm oranları ve pageview'den itibaren toplam dönüşüm oranlarını hesaplatmış oldum.

Funnel															
<pre>1 WITH user_events_with_next AS ( 2   SELECT 3     user_id, 4     platform, 5     event_name, 6     event_time, 7 8     LEAD(event_name) OVER (PARTITION BY user_id ORDER BY event_time) AS next_event_name, 9     LEAD(event_time) OVER (PARTITION BY user_id ORDER BY event_time) AS next_event_time 10  FROM `sevenapps-case-study-452916.sevenapps_case.user_events` 11  ORDER BY user_id, event_time</pre>															
Query results															
JOB INFORMATION		RESULTS	CHART	JSON	EXECUTION DETAILS		EXECUTION GRAPH								
Row	platform	unique_users	pageview_count	pageview_to_download	download_to_install	install_to_signup_pct	signup_to_subscription	pageview_to_download	download_to_install	install_to_signup_pct	signup_to_subscription	pageview_to_install	pageview_to_signup	pageview_to_subscription	
1	android	6247	6247	4354	3489	2676	792	69.7	80.13	76.7	29.6	55.85	42.84	12.68	
2	ios	3753	3753	2638	2109	1581	504	70.29	79.95	74.96	31.88	56.2	42.13	13.43	

- Oluşturduğum sorguya Q1 klasörünün içindeki **"funnel.sql"** den ulaşabilirsiniz.

## Sonuçlar ve Bulgular

Analizim sonucunda aşağıdaki bulguları elde ettim:

### 1. Kullanıcı Dağılımı:

- Android: 6,247 kullanıcı (%62.5)
- iOS: 3,753 kullanıcı (%37.5)

### 2. Aşamalar Arası Dönüşüm Oranları:

- PageView → Download: Android %69.7, iOS %70.29
- Download → Install: Android %80.13, iOS %79.95
- Install → SignUp: Android %76.7, iOS %74.96
- SignUp → Subscription: Android %29.6, iOS %31.88

### 3. Bütünden Dönüşüm Oranları:

- PageView → Download: Android %69.7, iOS %70.29
- PageView → Install: Android %55.85, iOS %56.2
- PageView → SignUp: Android %42.84, iOS %42.13
- PageView → Subscription: Android %12.68, iOS %13.43

### 4. Önemli Bulgu: iOS platformundaki kullanıcılar, PageView → Download ve SignUp → Subscription aşamalarında Android kullanıcılarına göre daha yüksek dönüşüm oranına sahip.

### 5. Darboğaz Analizi: En büyük dönüşüm kaybı, tüm platformlarda SignUp → Subscription aşamasında gerçekleşiyor (%30 civarı dönüşüm oranı).

## Case Study Q2: Mobil Uygulama Verilerinin Optimum Modellenmesi

### A. Veri Modelleme Yaklaşımı

#### Denormalizasyon-Odaklı Hibrit Yaklaşım

Mobil uygulama verileri inanılmaz büyük hacimlere ulaşabiliyor ve bu verileri verimli şekilde modellemek, hem performans hem de maliyet açısından kritik bir konu. Tüm verileri tek bir tabloya yazmak sorguları hızlandırabilir ama veri tekrarını artırır, tamamen normalize etmek ise performansı düşürebilir. Bu yüzden, hibrit bir model kullanmanın en mantıklı çözüm olduğunu düşünüyorum.

Bu sistemde **denormalize edilmiş bir fact tablosu** ve destekleyici **dimension (boyut) tabloları** kullanacağız.

- **Denormalize Edilmiş Fact Table** (Analizleri hızlandırır, okuma optimizasyonu sağlar)
- **Dimension Tables** (Lookup verileri için kullanılır, veri tutarlılığını artırır)
- **Aggregation Tables** (Ön hesaplanmış dönüşüm oranları ve cohort analizleri için)
- **Materialized Views** (Gerçek zamanlı funnel analizlerini hızlandırır)

Bu modelde, **sık sorgulanan bilgiler fact tablosunda denormalize edilirken, az değişen veya daha az sorgulanan detaylar dimension tablolarında saklanacaktır.**

```
-- Ana tablo: Denormalize edilmiş event verileri
CREATE TABLE fact_user_events (
  -- Temel bilgiler
  event_id VARCHAR(255) PRIMARY KEY,
  event_timestamp TIMESTAMP,
  event_date DATE, -- Bölümleme için

  -- Olay detayları
  event_type VARCHAR(50), -- 'PageView', 'Download', 'Install'

  -- Denormalize edilmiş kullanıcı ve cihaz bilgileri (sık sorgulanan)
  user_id VARCHAR(255),
  session_id VARCHAR(255),
  platform VARCHAR(50), -- 'ios', 'android'
  device_id VARCHAR(100), -- device tablosuna referans
  device_name VARCHAR(100), -- Denormalize edilmiş
  device_type VARCHAR(100), -- Denormalize edilmiş
  chipset_name VARCHAR(100), -- Denormalize edilmiş (sık sorgulanan)
  country_code VARCHAR(2), -- Denormalize edilmiş

  -- Esnek özellikler
  event_properties JSON -- Olaya özgü ekstra bilgiler
)
PARTITION BY RANGE (event_date), LIST (platform);
CREATE INDEX idx_platform ON fact_user_events (platform);
```

## Bölümleme (Partitioning) Stratejisi



Büyük veri setleriyle çalışırken, **partitioning ve indexing doğru şekilde kullanıldığında sorgu hızını artırır ve sistemi daha verimli hale getirir.** Ama burada önemli olan, **veriyi nasıl işlediğimizi ve hangi analizlerin sık yapıldığını göz önünde bulundurmak.**

Benim yaklaşımımda, **event\_date için partition yaparken, platform için index kullanmak daha mantıklı**. Çünkü:

- **Tarih bazlı analizler çok sık yapıldığı için,** `event_date` üzerinden partition oluşturmak sorgu performansını ciddi şekilde artırıyor. **Eski verileri temizlemek ve yönetmek de çok daha kolay hale geliyor.**
- **Platform sayısı az (iOS ve Android) olduğu için,** platform bazlı partition yapmak gereksiz küçük bölümler oluşturabilir. Bunun yerine **platform için bir index eklemek, sorguların hızlı çalışmasını sağlamak için yeterli.**
- **Bu yapı, büyük veri setlerinde gereksiz partition oluşmasını engellerken,** aynı zamanda hem **tarih bazlı** hem de **platform bazlı sorguların hızlı çalışmasını sağlıyor.**

Yani **platform için partition yapmak yerine index kullanarak esnekliği koruyor, event\_date üzerinden partition yaparak sorgu performansını artırıyorum.**

## Dimension ve Lookup Tabloları

Denormalize veri modelini zenginleştirmek için aşağıdaki boyut ve lookup tablolarını öneriyorum:

-- 1. Kullanıcı Boyut Tablosu (Detaylı kullanıcı profili)

```
CREATE TABLE dim_users (
    user_id VARCHAR(255) PRIMARY KEY,
    first_name VARCHAR(100),
    last_name VARCHAR(100),
    email VARCHAR(255),
    registration_date DATE,
    registration_platform VARCHAR(50),
    registration_device_id VARCHAR(100),
    country_code VARCHAR(2),
    city VARCHAR(100),
    region VARCHAR(100),
    postal_code VARCHAR(20),
    language VARCHAR(50),
    age_group VARCHAR(20),
    gender VARCHAR(20),
    user_segment VARCHAR(50),
    acquisition_source VARCHAR(100),
    acquisition_campaign VARCHAR(100)
);
```

-- 2. Cihaz Boyut Tablosu (Detaylı donanım bilgileri)

```
CREATE TABLE dim_devices (
```



```

device_id VARCHAR(100) PRIMARY KEY,
device_name VARCHAR(100),
device_type VARCHAR(100), -- 'mobile', 'tablet', 'desktop'
device_brand VARCHAR(100), -- 'Apple', 'Samsung', 'Xiaomi'
device_model VARCHAR(100), -- 'iPhone 13', 'Galaxy S22'
chipset_id VARCHAR(50),
chipset_name VARCHAR(100), -- 'A15 Bionic', 'Snapdragon 8 Gen 1'
chipset_manufacturer VARCHAR(100), -- 'Apple', 'Qualcomm'
processor_cores INTEGER,
processor_speed DECIMAL(4,2),
ram_gb INTEGER,
storage_gb INTEGER,
battery_capacity INTEGER,
screen_size_inch DECIMAL(3,1),
screen_resolution VARCHAR(50),
platform VARCHAR(50), -- 'iOS', 'Android'
os_version VARCHAR(50), -- '15.4.1', '12.0'
);

```

-- 3. Chipset Lookup Tablosu (Detaylı işlemci bilgileri)

```

CREATE TABLE lookup_chipsets (
    chipset_id VARCHAR(50) PRIMARY KEY,
    chipset_name VARCHAR(100),
    manufacturer VARCHAR(100),
    architecture VARCHAR(50), -- 'ARM', 'x86'
    manufacturing_process VARCHAR(20), -- '5nm', '7nm'
    release_date DATE,
    cpu_benchmark_score INTEGER,
    gpu_benchmark_score INTEGER,
    ai_benchmark_score INTEGER,
    energy_efficiency_score INTEGER,
    cpu_cores INTEGER,
    cpu_max_clock_speed DECIMAL(4,2),
    gpu_cores INTEGER,
    gpu_max_clock_speed DECIMAL(4,2),
    ai_dedicated_cores INTEGER,
    memory_bandwidth_gbps INTEGER,
    max_memory_support_gb INTEGER
);

```

-- 4. Coğrafi Konum Lookup Tablosu (Coğrafi analiz için)

```

CREATE TABLE lookup_geo (
    country_code VARCHAR(2) PRIMARY KEY,
    country_name VARCHAR(100),
    continent VARCHAR(50),
    region VARCHAR(100),

```

```
timezone VARCHAR(50),
currency_code VARCHAR(3),
currency_name VARCHAR(50),
avg_internet_speed DECIMAL(6,2)
);
```

-- 5. Platform Detayları Lookup Tablosu

```
CREATE TABLE lookup_platforms (
  platform_id INTEGER PRIMARY KEY,
  platform_name VARCHAR(50), -- 'iOS', 'Android'
  platform_version VARCHAR(50), -- '15', '12'
  vendor VARCHAR(100), -- 'Apple', 'Google'
  release_date DATE,
  api_level INTEGER, -- Android için
  supported_architectures JSON, -- ['arm64-v8a', 'armeabi-v7a']
  kernel_version VARCHAR(50),
  is_current BOOLEAN,
  end_of_support_date DATE
);
```

-- 6. Event Tipleri Lookup Tablosu (Olay taksonomi yönetimi)

```
CREATE TABLE lookup_event_types (
  event_type VARCHAR(50) PRIMARY KEY,
  event_category VARCHAR(50), -- 'Navigation', 'Conversion', 'Engagement'
  event_description TEXT,
  funnel_stage VARCHAR(50), -- 'Awareness', 'Consideration', 'Decision'
  user_intent VARCHAR(100), -- 'Research', 'Purchase', 'Support'
  required_properties JSON, -- Gerekli özellikler
  optional_properties JSON, -- Opsiyonel özellikler
  validation_rules JSON, -- Doğrulama kuralları
  -- Meta veriler
  created_date DATE,
  is_active BOOLEAN,
  last_updated TIMESTAMP
);
```

-- 7. Oturum (Session) Özet Tablosu

```
CREATE TABLE fact_sessions (
  session_id VARCHAR(255) PRIMARY KEY,
  user_id VARCHAR(255),
  device_id VARCHAR(100),
  platform VARCHAR(50),
  start_time TIMESTAMP,
  end_time TIMESTAMP,
  duration_seconds INTEGER,
```

```

-- Oturum metrikleri
event_count INTEGER,
unique_screens_viewed INTEGER,
total_scrolled_distance INTEGER,
engagement_score DECIMAL(5,2),
-- Teknik bilgiler
network_type VARCHAR(50), -- 'wifi', 'cellular', '4G', '5G'
app_version VARCHAR(50),
-- Meta veriler
session_date DATE, -- Bölümleme için
-- Yabancı anahtar
FOREIGN KEY (user_id) REFERENCES dim_users(user_id),
FOREIGN KEY (device_id) REFERENCES dim_devices(device_id)
);

```

## Aggregation Tables & Materialized Views

```

-- Önceden hesaplanmış funnel metrikleri
CREATE TABLE agg_daily_funnel (
  date DATE,
  platform VARCHAR(50),
  country_code VARCHAR(2),

  -- 72 saat içindeki dönüşümler
  pageview_count INTEGER,
  pageview_unique_users INTEGER,
  download_count INTEGER,
  download_unique_users INTEGER,
  install_count INTEGER,
  pageview_to_download_72h_count INTEGER,
  download_to_install_72h_count INTEGER,

  -- Dönüşüm oranları
  pageview_to_download_rate DECIMAL(5,2),
  download_to_install_rate DECIMAL(5,2),

  PRIMARY KEY (date, platform, country_code)
);

-- Gerçek zamanlı kullanıcı yolculuğu analizi
CREATE MATERIALIZED VIEW mv_user_journey AS
SELECT
  user_id,
  platform,
  country_code,
  MIN(CASE WHEN event_type = 'PageView' THEN event_timestamp END) AS
  first_pageview_time,

```

```

        MIN(CASE WHEN event_type = 'Download' THEN event_timestamp END) AS first_download_time,
        MIN(CASE WHEN event_type = 'Install' THEN event_timestamp END) AS first_install_time,
        /* ... diğer event'ler ... */

-- 72 saat içinde dönüşüm bayrakları
CASE WHEN
    MIN(CASE WHEN event_type = 'Download' THEN event_timestamp END)
IS NOT NULL AND
    TIMESTAMP_DIFF(
        MIN(CASE WHEN event_type = 'Download' THEN event_timestamp END),
        MIN(CASE WHEN event_type = 'PageView' THEN event_timestamp END),
        HOUR) <= 72
    THEN 1 ELSE 0 END AS converted_pageview_to_download_72h
FROM fact_user_events
GROUP BY user_id, platform, country_code;

```

## Veri İlişkileri ve Lookup Tabloları Kullanımı

```

-- Device ve Chipset bilgilerini birleştirme sorgusu örneği
SELECT
    d.device_name,
    d.device_brand,
    d.device_model,
    c.chipset_name,
    c.manufacturer AS chipset_manufacturer,
    c.architecture,
    c.manufacturing_process,
    c.cpu_cores,
    c.cpu_max_clock_speed,
    c.performance_tier
FROM dim_devices d
JOIN lookup_chipsets c ON d.chipset_id = c.chipset_id
WHERE d.platform = 'iOS'
ORDER BY c.cpu_benchmark_score DESC;

-- Kullanıcı ve cihaz bilgilerini birleştirme
SELECT
    u.user_id,
    u.country_code,
    g.country_name,
    u.registration_date,
    u.user_segment,
    d.device_name,

```

```
d.device_brand,  
d.device_model,  
c.chipset_name,  
c.performance_tier  
FROM dim_users u  
JOIN dim_devices d ON u.registration_device_id = d.device_id  
JOIN lookup_chipsets c ON d.chipset_id = c.chipset_id  
JOIN lookup_geo g ON u.country_code = g.country_code  
WHERE u.registration_date >= CURRENT_DATE - INTERVAL '90' DAY;
```

## B. Kullanıcı Özniteliklerinin Eklenmesi (Ülke, Cihaz Tipi vb.)

Kullanıcı özniteliklerini eklemek için denormalizasyon prensiplerine dayalı şu stratejiyi öneriyorum:

### 1. Sık Kullanılan Özniteliklerin Denormalize Edilmesi

```
-- Fact tablosuna ek öznitelikler  
ALTER TABLE fact_user_events  
ADD COLUMN city VARCHAR(100),  
ADD COLUMN language VARCHAR(50),  
ADD COLUMN os_version VARCHAR(50),  
ADD COLUMN connection_type VARCHAR(50);
```

### 2. Az Değişen ve Az Kullanılan Öznitelikler için Dimension Tablosu

```
-- Kullanıcı boyut tablosuna ek öznitelikler  
ALTER TABLE dim_users  
ADD COLUMN gender VARCHAR(20),  
ADD COLUMN age_group VARCHAR(20),  
ADD COLUMN subscription_status VARCHAR(50),  
ADD COLUMN user_preferences JSON;
```

### 3. Esnek Öznitelik Yönetimi

Sürekli değişen veya yeni eklenen öznitelikler için esnek bir yaklaşım:

```
-- Esnek kullanıcı öznitelikleri tablosu  
CREATE TABLE user_attributes (  
    user_id VARCHAR(255),  
    attribute_name VARCHAR(100),  
    attribute_value VARCHAR(255),  
    updated_at TIMESTAMP,  
    PRIMARY KEY (user_id, attribute_name)  
);
```

## 4. Denormalizasyon Güncelleme Süreci

Yeni eklenen özniteliklerin fact tablosuyla tutarlı olması için ETL süreci:

```
-- ETL süreci örneği (günlük çalıştırılabilir)
UPDATE fact_user_events e
SET
    country_code = u.country_code,
    city = u.city,
    language = u.language
FROM dim_users u
WHERE e.user_id = u.user_id
AND e.event_date = CURRENT_DATE;
```

## C. Mevcut Sistem Potansiyel Sorunları ve Çözümleri

### Veri Büyüklüğü ve Performans Sorunları

**Sorun:** Denormalizasyon veri hacmini artırabilir ve belirli sorgu tipleri için performans sorunları oluşturabilir.

#### Çözüm:

- Etkin bölümlleme stratejisi (zaman, platform bazlı)
- Sürekli kullanılan metrikler için aggregate tabloları
- Sorgular için indeksleme optimizasyonu.

**bkz. fact\_user\_events table.**

### Veri Tutarlılığı Zorlukları

**Sorun:** Denormalize edilmiş modelde veri tekrarı olduğundan, boyut bilgileri değiştiğinde tutarsızlıklar oluşabilir.

#### Çözüm:

- Otomatik günlük tutarlılık kontrolleri bkz.

```
-- 1. Günlük veri kalitesi kontrolü
CREATE VIEW vw_data_quality_daily AS
SELECT
    event_date,
    COUNT(*) AS total_events,
    COUNT(DISTINCT user_id) AS unique_users,
    SUM(CASE WHEN event_type NOT IN ('PageView', 'Download', 'Install', 'SignUp', 'Subscription')
        THEN 1 ELSE 0 END) AS invalid_event_types,
    SUM(CASE WHEN platform NOT IN ('ios', 'android')
        THEN 1 ELSE 0 END) AS invalid_platforms,
    SUM(CASE WHEN user_id IS NULL THEN 1 ELSE 0 END) AS missing_user_id
```

```

FROM fact_user_events
GROUP BY event_date
ORDER BY event_date DESC;

-- 2. Anomali tespiti
WITH daily_counts AS (
    SELECT
        event_date,
        platform,
        COUNT(*) AS event_count
    FROM fact_user_events
    WHERE event_date BETWEEN CURRENT_DATE - INTERVAL '14' DAY AND C
    URRENT_DATE
    GROUP BY event_date, platform
),
baseline AS (
    SELECT
        platform,
        AVG(event_count) AS avg_events,
        STDDEV(event_count) AS stddev_events
    FROM daily_counts
    WHERE event_date < CURRENT_DATE
    GROUP BY platform
)
SELECT
    d.event_date,
    d.platform,
    d.event_count,
    b.avg_events,
    ABS(d.event_count - b.avg_events) / NULLIF(b.stddev_events, 0) AS zscore,
    CASE WHEN ABS(d.event_count - b.avg_events) / NULLIF(b.stddev_events,
0) > 3
        THEN 'ANOMALİ' ELSE 'NORMAL' END AS status
FROM daily_counts d
JOIN baseline b ON d.platform = b.platform
WHERE d.event_date = CURRENT_DATE;

```

## Veri Bütünlüğü İzleme Eksiklikleri

**Sorun:** Mevcut sistemde veri bütünlüğü ve kalitesini izleyen otomatik sistemler eksik olabilir.

### Çözüm:

- Günlük veri kalitesi raporu
- Otomatik anomali tespiti
- SLA (Hizmet Düzeyi Anlaşması) bazlı izleme ve uyarılar



```
-- Günlük event sayılarında keskin düşüşler için uyarı sistemi
WITH daily_trends AS (
  SELECT
    event_date,
    COUNT(*) AS event_count,
    LAG(COUNT(*), 1) OVER (ORDER BY event_date) AS prev_day_count
  FROM fact_user_events
  GROUP BY event_date
  ORDER BY event_date DESC
  LIMIT 2
)
SELECT
  event_date,
  event_count,
  prev_day_count,
  100 * (event_count - prev_day_count) / prev_day_count AS change_percent,
  CASE WHEN 100 * (event_count - prev_day_count) / prev_day_count < -25
    THEN 'ALERT' ELSE 'NORMAL' END AS status
FROM daily_trends
WHERE event_date = CURRENT_DATE;
```

## Özet

Mobil uygulama funnel verilerini modellerken denormalizasyon tekniklerini akıllıca kullanarak sorgu performansını maksimize ediyorum. Bu yaklaşım, kullanıcı davranışlarını anlamak için dönüşüm hızlarını ölçmede kritik öneme sahip.

- **Denormalizasyon:** Sık sorgulanan öznitelikleri ana tabloya ekleyerek JOIN'leri minimize ettim
- **Bölümlendirme:** Tarih bazlı bölümlendirme ile zamana dayalı analiz hızını artırdım
- **Önceden Hesaplanmış Tablolar:** Karmaşık hesaplamalar için aggregation tabloları ve materialized view'lar
- **Veri Kalitesi:** Tutarlılık kontrolü, anomali tespiti ve değişiklik izleme mekanizmaları

## Case Study Q3: Visualization

### Hangi Araçları Kullanmalıyız?

Görselleştirme için farklı araçlar mevcut. Seçim yaparken **kullanım kolaylığı, esneklik ve veri kaynaklarına entegrasyon yeteneği** gibi faktörleri göz önünde bulundurmalıyız.

- **Looker Studio (Google Data Studio) → BigQuery ve Firebase entegrasyonu kolay** olduğu için tercih edilebilir.

- **Tableau / Power BI** → Daha güçlü **özelleştirme ve drill-down özellikleri** için tercih edilebilir.
- **Apache Superset** → Ücretsiz ve büyük verilere optimize iyi bir seçenektir.

## Dashboard’da Hangi Metrikleri Dahil Etmeliyiz?

Bir funnel analiz dashboard'u için **kritik metrikleri** belirlememiz gerekiyor.

- **Ana Funnel Adımları:**

Metrik	Açıklama
Toplam PageViews	Uygulama sayfasını görüntüleyen kullanıcı sayısı
Toplam Downloads	Uygulamayı indiren kullanıcı sayısı
Toplam Installs	Uygulamayı yükleyen kullanıcı sayısı
Toplam Signups	Uygulamaya kaydolan kullanıcı sayısı
Toplam Subscriptions	Ücretli kullanıcı olan kişi sayısı

- **Dönüşüm Oranları (Conversion Rates):**

Metrik	Açıklama
PageView → Download (%)	Sayfa görüntüleme yapanların indirme oranı
Download → Install (%)	İndirenlerin yükleme oranı
Install → Signup (%)	Yükleyenlerin kayıt olma oranı
Signup → Subscription (%)	Kayıt olanların ücretli kullanıcıya dönüşme oranı

- **Ek Segmentasyon ve Kırılımlar:**

Kırılım	Neden Önemli?
Platform (iOS vs. Android)	Hangi işletim sisteminde dönüşüm daha yüksek?
Ülke Bazlı Analiz	Hangi ülkelerde dönüşüm daha iyi?
Cihaz Türü	Hangi cihaz modellerinde dönüşüm daha iyi?
Kullanıcı Segmentleri	VIP kullanıcılar vs. normal kullanıcılar arasındaki farklar

- **Ek Metrikler (Opsiyonel):**

Metrik	Açıklama
Ortalama Dönüşüm Süresi (h)	PageView → Download arasındaki ortalama süre
72 Saat İçinde Dönüşüm Oranı	Kullanıcının 72 saat içinde funnel'ı tamamlayıp tamamlamadığı

## Dashboard Tasarımı

Bir dashboard tasarlarken en önemli hedefimiz, dönüşüm oranlarını kolayca takip edebilecek bir yapı oluşturmaktır. İdeal bir dashboard şunları içermelidir:

## Dashboard Bölümleri

## Genel Bakış (Overview)

- Toplam Kullanıcı Sayısı
- Günlük/Aylık Aktif Kullanıcılar (DAU/MAU)
- Tüm funnel aşamalarında toplam sayı ve dönüşüm oranları
- En iyi dönüşüm oranına sahip platform / ülke / segment bilgileri

## Funnel Görselleştirmesi

- Funnel grafiği (PageView → Download → Install → Signup → Subscription)
- Funnel aşamalarındaki mutlak değerler ve yüzdesel değişimler
- Zaman serisi olarak funnel aşamalarının trendi

## Platform ve Cihaz Kırılımı

- iOS vs Android için dönüşüm oranları
- Cihaz türlerine göre dönüşüm analizi (Samsung, iPhone, Xiaomi vb.)

## Coğrafi Kırılım

- Ülkeler bazında dönüşüm oranları
- Harita görselleştirmesi (daha yüksek dönüşüm oranı olan bölgeler için)

## Kullanıcı Segmentleri

- VIP vs Normal Kullanıcıların funnel dönüşüm oranları
- Kullanıcı kohort analizi (hangi kullanıcı grubu daha iyi dönüşüyor?)

---

## Hangi Grafik Türleri Kullanılmalı?

### Funnel Analizi için:

- **Step Chart (Adım Grafiği)** → Funnel adımlarındaki kullanıcı kayıplarını gösterir.
- **Sankey Diagramı** → Kullanıcı akışını ve hangi aşamada en büyük kayıpların olduğunu anlamamızı sağlar.
- **Waterfall Chart (Şelale Grafiği)** → Her funnel adımında **ne kadar kullanıcı kaybedildiğini ve toplam etkiyi görselleştirmek** için idealdir. **Özellikle sayfa görüntüleme → indirme → yükleme → kayıt → abonelik gibi süreçlerde kullanıcı kayıplarını net bir şekilde gösterir.**

### Zaman Serisi Analizi için:

- **Line Chart (Çizgi Grafiği)** → Funnel aşamalarında zaman içindeki değişimi gösterir.
- **Bar Chart (Çubuk Grafiği)** → Günlük/haftalık bazda dönüşüm oranlarını kıyaslamak için kullanılır.

### Segment ve Kırılım Analizleri için:

- **Heatmap (Isı Haritası)** → Farklı ülkeler için dönüşüm oranlarını analiz etmek için kullanılır.
- **Pie Chart (Pasta Grafiği)** → iOS vs. Android gibi platform analizlerinde kullanılır.

---

*Farklı zamanlarda hazırladığım örnek dashboardları Q3 klasörü içinde bulabilirsiniz. (Hiç biri gerçek veri içermemektedir. güncel olmasalar da en azından estetik anlayışımı anlayabilirsiniz. :) )*

---

**Teşekkürler.**