

Best Practice mit BigQuery

- E-Commerce Einstieg 2019 mit eigenem Shop
- CRO Experte für Startups und Mittelständler
- Einblicke in >200 Shops



V//RIFY.iO® - A/B Testing Platform

- ∞ Unlimited Traffic
- 🍪 No Cookies
- 🎯 Single Source of Truth
- 🔍 Shopify & Big Query Integration



Darum geht es heute:

- **Einführung: Was ist BigQuery überhaupt?**
- **Praxisbeispiel: A/B-Tests**
- **Wie ich mit BigQuery arbeite**
- **Tipps & Tools aus der Praxis**
- **Verbindung GA4 ↔ BigQuery**



Google
Big Query

Einführung

Was ist BigQuery?

- **BigQuery ist ein Cloud-basiertes Data Warehouse von Google**
→ speichert und analysiert sehr große Datenmengen (z. B. GA4-Rohdaten)
- **Ermöglicht es, große Datenmengen (z. B. GA4-Daten) blitzschnell per SQL auszuwerten**
→ eigene Fragen an die Daten stellen – ohne Einschränkungen durch vordefinierte Reports
- **Ideal für eigene Analysen, A/B-Testing und datengetriebenes Arbeiten**
→ Nutzung für individuelle Analysen, Segmentierungen, Testauswertungen und Reporting

Wie kommen Daten in BigQuery?

- **Automatisch aus GA4 exportiert**
→ per aktivierter BigQuery-Verknüpfung in den GA4-Property-Einstellungen
- **GA4-Rohdaten-Export (automatisch)**
→ die in GA4 erfassten Events werden 1:1 als Rohdaten in BigQuery gespeichert
→ Beispiel: ein Klick auf „In den Warenkorb“ wird als einzelnes Event mit Zeitstempel, User ID usw. gespeichert
- **Zusätzlich eigene Datenquellen möglich**
→ z. B. CRM, Shop, Logfiles, APIs, CSV/Excel
- **Eigene Daten flexibel importieren**
→ per API, Datei-Upload oder Zeitplan (Scheduled Jobs)

Events in BigQuery

	events_20250622	06-22 ▼	 Abfrag
Schema	Details	Vorschau	Tabellen-Exp
☐	user_pseudo_id		
☐	▶ privacy_info		
☐	▶ user_properties		
☐	user_first_touch_timestamp		
☐	▶ user_ltv		
☐	▶ device		
☐	▶ geo		
☐	▶ app_info		
☐	▶ traffic_source		
☐	stream_id		
☐	platform		
☐	▶ event_dimensions		
☐	▼ ecommerce		
☐	total_item_quantity		
☐	purchase_revenue_in_usd		
☐	purchase_revenue		
☐	refund_value_in_usd		
☐	refund_value		
☐	shipping_value_in_usd		
☐	shipping_value		
☐	tax_value_in_usd		
☐	tax_value		
☐	unique_items		
☐	transaction_id		
☐	▶ items		

Vorteile und Unterschiede zu GA4 Daten

- **Spezielle Tests und Kombinationen möglich**
→ z. B. Nutzer in Verbindung mit bestimmten Events oder Verhaltensmustern analysieren
- **Beliebig lange Zeiträume analysierbar**
→ in BigQuery speicherst du die Daten selbst — kein 14-Monats-Limit wie in GA4
- **Keine Sampling-Probleme**
→ auch bei langen Testläufen oder kleinen Segmenten sind alle Daten verfügbar. Keine Hochrechnungen.
- **Daten gesichert & exportierbar**
→ Rohdaten dauerhaft in BigQuery gespeichert, jederzeit downloadbar



Vorteile und Unterschiede zu GA4 Daten

Unterschiede zu GA4 direkt:

- GA4 Standard-Speicherung: 2 bis 14 Monate je nach Einstellung
→ Funnel- und Exploration-Daten (z. B. Segmente, Verhaltensflows) sind nach Ablauf gelöscht
- BigQuery Sandbox: 60 Tage Tabellenlaufzeit — Upgrade für dauerhaftes Speichern möglich (geringe Kosten)
- BigQuery bietet volle Datenhoheit
→ keine Abhängigkeit von der Retention-Einstellung in GA4. Daten können exportiert werden.



Das kostet Big Query



- **Kostenloses Kontingent inklusive**
→ bis 10 GB Speicher und 1 TB Abfragen pro Monat gratis
- **Benachrichtigungen bei Limits möglich**
→ automatische E-Mail-Warnungen bei Speicher- oder Abfragegrenzen einstellbar
- **Abrechnung nach tatsächlicher Nutzung**
→ nur was gespeichert oder abgefragt wird, wird berechnet (kein Fixpreis)
- **Keine Angst vor Kostenfallen**
→ man sieht im Google Cloud-Konto jederzeit die aktuelle Nutzung

Abfrage A/B Test & Datenmenge

- Test lief 45 Tage
- Abfrage 6,7 GB
- Diese Abfrage könnte ich 150x machen um 1 TB zu verbrauchen

Abfrageergebnisse		
Jobinformationen		
Ergebnisse		
Dia		
ZEILE	total_users	users_with_trans...
1	10093	2414

Abfrageergebnisse

Ergebnisse speichern

Öffnen in

Jobinformationen

Ergebnisse

Diagramm

JSON

Ausführungsdetails

Ausführungsgrafik

Job-ID

Nutzer

nikospies1@gmail.com

Ort

EU

Erstellungszeit

23.06.2025, 15:36:56 UTC+2

Beginn

23.06.2025, 15:36:56 UTC+2

Ende

23.06.2025, 15:36:59 UTC+2

Dauer

2 Sek

Verarbeitete Byte

6,69 GB

In Rechnung gestellte Byte

6,69 GB

Beispielrechnung Kontingent

- Annahme: Shop mit 100.000 Nutzern & 1.500 Käufen pro Monat
→ ca. 10–15 Events pro Nutzer → ca. 1,5 Mio. Events / Monat



x 100.000



x 1.500

- Datengröße: ca. 1 KB pro Event (GA4-Events mit Standard-Daten + Custom Dimensions)
→ 1,5 Mio. Events × 1 KB = ca. 1,5 GB pro Monat

Beispielrechnung Kontingent



x 100.000



x 1.500

- **Kostenloses Kontingent: 10 GB Speicher**
- Ergebnis: nach ca. 6–7 Monaten wäre das Free-Tier (10 GB) erreicht
- Danach: jeder weitere GB kostet nur ca. 0,02 USD / Monat
→ Beispiel: bei 15 GB Speicher = ca. 0,10 € / Monat

Beispielrechnung Kontingent



x 100.000



x 1.500

- **Abfrage-Kontingent: 1 TB pro Monat kostenlos**
- Beispiel: Abfrage mit 6,7 GB (siehe vorherige Folie)
→ ca. 150 Abfragen pro Monat kostenlos
- In der Praxis: bei typischen A/B-Tests und Analysen bleibt man fast immer im Free-Tier
→ bei 5 - 10 Testauswertungen pro Monat → meist deutlich unter dem 1 TB Limit

Praxisbeispiele

Was ist SQL?

- **Sprache für Datenabfragen**
→ mit SQL kann man aus einer Datenbank gezielt Informationen herausholen
- **In BigQuery nutze ich SQL, um GA4-Daten auszuwerten**
→ z. B. wie oft wurde etwas gekauft, welche Nutzergruppe hat besser performt
- **Kleine Befehle – große Wirkung**
→ mit wenigen Zeilen SQL kann ich eigene Reports und Analysen erstellen

Grundlagen um Tests auswerten zu können

- **GA4-Tracking vollständig und sauber eingerichtet**
 - z. B. Purchase-Event, alle wichtigen Events kommen in GA4 an
 - Consent-Banner korrekt eingebunden und eingestellt
- **Testing-Tool integriert und Events im GTM erfasst**
 - das Testing-Tool sendet relevante Events (z. B. „Testing Tool Event“) an GA4
- **Testing-Events enthalten eindeutige Informationen**
 - z. B. „experiment_id“ und „variant_name“ im Event → damit die Testgruppen später sauber ausgewertet werden können

Grundlagen um Tests auswerten zu können

16 cookie_consent_update

15 Einwilligungsbefehl „upd...

14 Set

13 Set

12 Trigger-Gruppe

11 **verify**

10 Fenster geladen

9 DOM ist bereit.

8 Container geladen

7 template_load

6 Initialisierung

Ausgabe von ⓘ

TagsVariablenDatenschicht

Datenschichtwerte nach dieser Nachricht:

```
1 {
2   developer_id: {dN2ZKMj: true, dY2Q2ZW: true},
3   gtm: {uniqueEventId: 877, priorityId: 2, start: 1750760063328},
4   event: "verify",
5   ads_data_redaction: false,
6   url_passthrough: false,
7   templateType: "index",
8   verify_experimentId: 21952,
9   verify_experimentName: "sidemenu-0011",
10  verify_variationId: null,
11  verify_variationName: "Original",
12  verify_abTest: "21952_sidemenu-0011:Original",
13  verify_abTestLong: "21952_sidemenu-0011:Original",
14  verify_abTestShort: "21952:Original",
15  verify_success: 1
16 }
```

Wo können wir abfragen?

Google Cloud

Nach Ressourcen, Dokumenten, Produkten und mehr suchen (/)

Suche

Willkommen

Sie arbeiten in

Projektnummer:

Projekt-ID:

Dashboard

Cloud Hub

Neu

+ VM erstellen

+ Abfrage in BigQuery ausführen

+ Anwendung bereitstellen

+ Storage-Bucket erstellen

Schnellzugriff

API APIs und Dienste

IAM und Verwaltung

Abrechnung

Compute Engine

Cloud Storage

BigQuery

VPC-Netzwerk

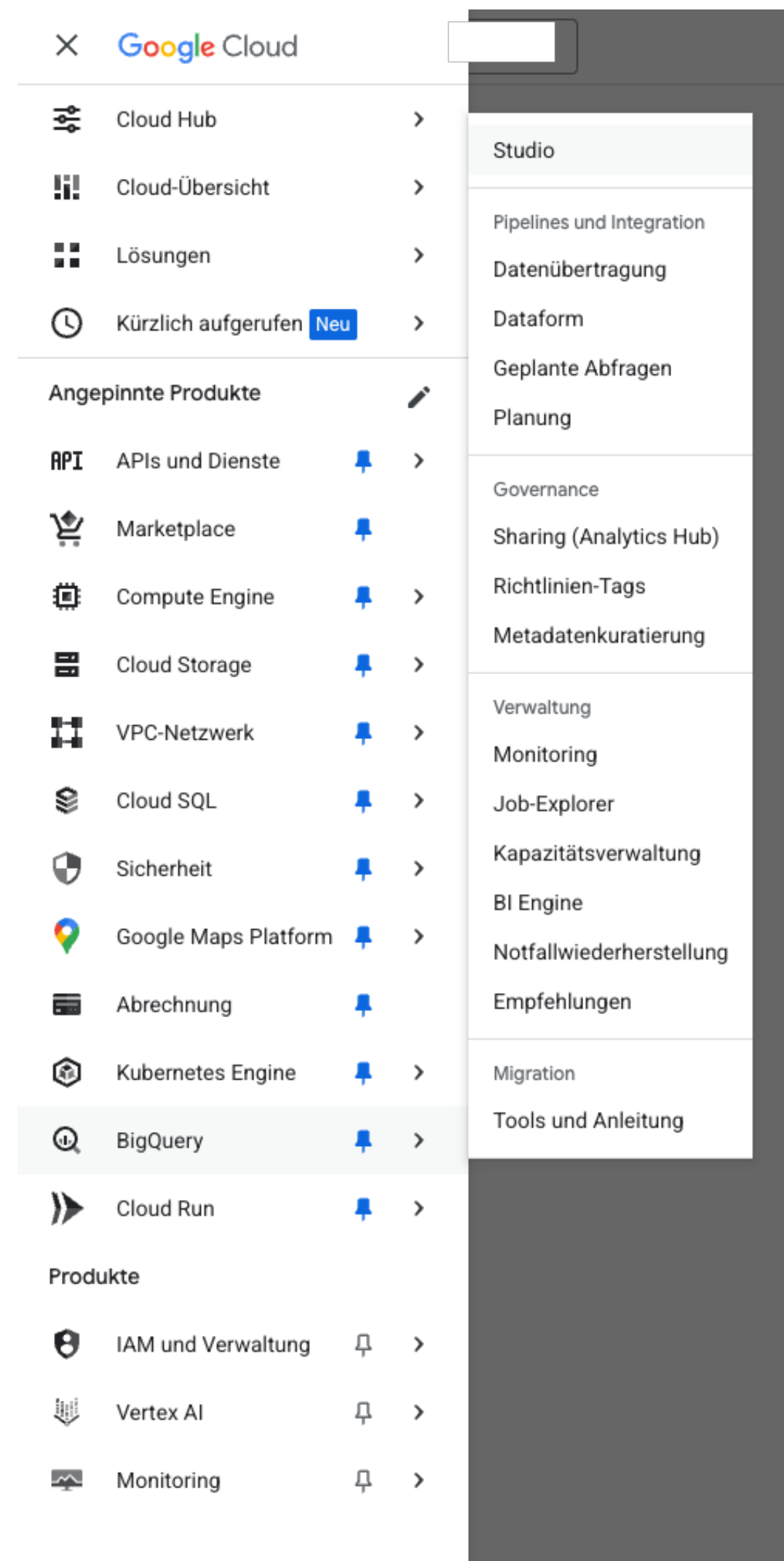
Kubernetes Engine

Alle Produkte ansehen

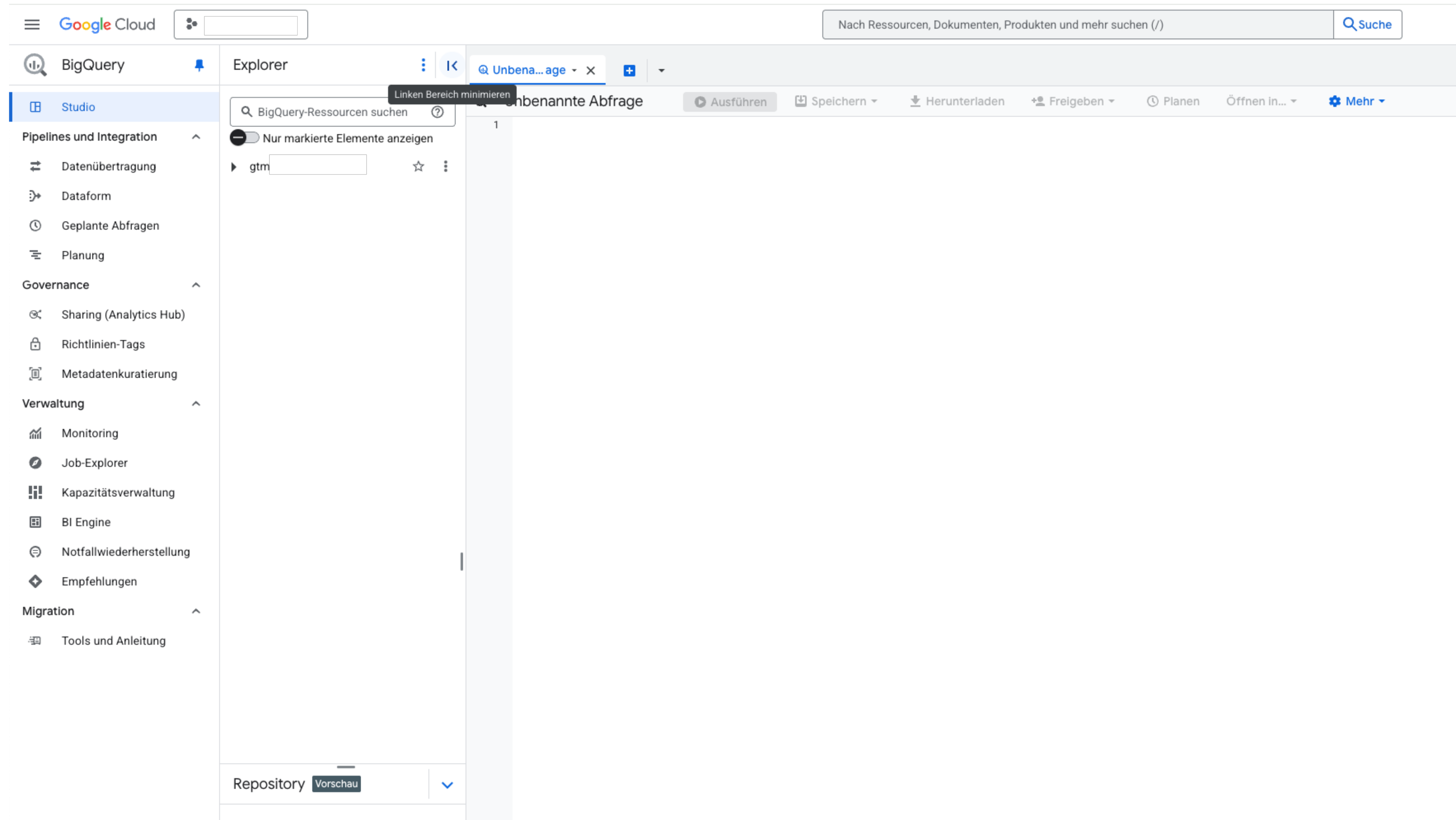
Gemini 2.0 Flash testen

Gemini testen

Wo können wir abfragen?



Wo können wir abfragen?



Wo können wir abfragen?

The screenshot displays the Google Cloud BigQuery Studio interface. At the top, the Google Cloud logo is visible on the left, and a search bar on the right contains the text "Nach Ressourcen, Dokumenten, Produkten und mehr suchen (/)" with a "Suche" button. Below the header, the interface is divided into three main sections:

- Left Sidebar (Navigation):** Contains a "BigQuery" header and a "Studio" tab. Below these are several categories with expandable arrows:
 - Pipelines und Integration:** Includes "Datenübertragung", "Dataform", "Geplante Abfragen", and "Planung".
 - Governance:** Includes "Sharing (Analytics Hub)", "Richtlinien-Tags", and "Metadatenkuratierung".
 - Verwaltung:** Includes "Monitoring", "Job-Explorer", "Kapazitätsverwaltung", "BI Engine", "Notfallwiederherstellung", and "Empfehlungen".
 - Migration:** Includes "Tools und Anleitung".
- Explorer (Middle):** Features a search bar "BigQuery-Ressourcen suchen" and a toggle "Nur markierte Elemente anzeigen". It shows a tree structure with two main folders:
 - gtm:** Contains "Repositories", "Abfragen", "(Klassische) Abfragen (32)", "Notebooks", "Daten-Canvase", "Datenvorbereitungen", "Pipelines", and "Externe Verbindungen".
 - analytics:** Contains "events_ (243)", "pseudonymous_...", and "users_ (245)".
- Main Editor (Right):** Displays a query editor for "Unbenannte Abfrage". It includes a toolbar with buttons for "Ausführen", "Speichern", "Herunterladen", "Freigeben", "Planen", "Öffnen in...", and "Mehr". The query text area shows a single line with the number "1".

Wo können wir abfragen?

BigQuery-Ressourcen suchen

Nur markierte Elemente anzeigen

gtm

Repositories

Abfragen

(Klassische) Abfragen (32)

Projektanfragen

2024-12-04 18:15:15

AA Test - Original

AA Test - Variante

BFCM Original

BFCM Variante

PUR Events

collection_001.1 - Kohorten

collection_001.1 - Original

collection_001.1 - Variante

collection_001.1 - Original

Ausführen

(Klassische) Abfrage speichern

Freigeben

Planen

```
1 SELECT
2   (SELECT COUNT(DISTINCT user_pseudo_id)
3   FROM `gtm`
4   WHERE event_name = 'Abtesting'
5         AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
6         AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325') AS total_users,
7   COUNT(DISTINCT user_pseudo_id) AS users_with_transactions,
8   ROUND(SUM(ecommerce.purchase_revenue), 2) AS Revenue,
9   ROUND((COUNT(DISTINCT user_pseudo_id) /
10  (SELECT COUNT(DISTINCT user_pseudo_id)
11  FROM `gtm`
12  WHERE event_name = 'Abtesting'
13        AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
14        AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325')) * 100, 2) AS CR,
15  ROUND(SUM(ecommerce.purchase_revenue) / COUNT(DISTINCT user_pseudo_id), 2) AS AOV,
16  ROUND(SUM(ecommerce.purchase_revenue) /
17  (SELECT COUNT(DISTINCT user_pseudo_id)
18  FROM `gtm`
19  WHERE event_name = 'Abtesting'
20        AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
21        AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325'), 2) AS RPU
22 FROM `gtm`
23 WHERE _TABLE_SUFFIX BETWEEN '20250210' AND '20250325'
24 AND ecommerce.purchase_revenue != 0
25 AND user_pseudo_id IN (
26   SELECT user_pseudo_id
27   FROM `gtm`
28   WHERE event_name = 'Abtesting'
29         AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
30         AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325');
```

Abfrage abgeschlossen

Abfrageergebnisse

Jobinformationen

Ergebnisse

Diagramm

JSON

Ausführungsdetails

Ausführungsgrafik

ZEILE	total_users	users_with_trans...	Revenue	CR	AOV	RPU
1	22819	745	67191.17	3.26	90.19	2.94

So werte ich Tests aus

```
SELECT
  (SELECT COUNT(DISTINCT user_pseudo_id)
   FROM `gtm-123456789.analytics_123456789.events_*`
   WHERE event_name = 'Abtesting'
        AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
        AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325') AS total_users,
  COUNT(DISTINCT user_pseudo_id) AS users_with_transactions,
  ROUND(SUM(ecommerce.purchase_revenue), 2) AS Revenue,
  ROUND((COUNT(DISTINCT user_pseudo_id) /
    (SELECT COUNT(DISTINCT user_pseudo_id)
     FROM `gtm-123456789.analytics_123456789.events_*`
     WHERE event_name = 'Abtesting'
          AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
          AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325')) * 100, 2) AS CR,
  ROUND(SUM(ecommerce.purchase_revenue) / COUNT(DISTINCT user_pseudo_id), 2) AS AOV,
  ROUND(SUM(ecommerce.purchase_revenue) /
    (SELECT COUNT(DISTINCT user_pseudo_id)
     FROM `TABELLEN ID`
     WHERE event_name = 'Abtesting'
          AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
          AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325'), 2) AS RPU
FROM `gtm-123456789.analytics_123456789.events_*`
WHERE _TABLE_SUFFIX BETWEEN '20250210' AND '20250325'
      AND ecommerce.purchase_revenue != 0
      AND user_pseudo_id IN (
  SELECT user_pseudo_id
  FROM `gtm-123456789.analytics_123456789.events_*`
  WHERE event_name = 'Abtesting'
        AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
        AND _TABLE_SUFFIX BETWEEN '20250210' AND ,20250325');
```

Abfrageergebnisse							Ergebnisse speichern ▾	
Jobinformationen		Ergebnisse	Diagramm	JSON	Ausführungsdetails		Ausführungsgrafik	
ZEILE	total_users ▾	users_with_trans...	Revenue ▾	CR ▾	AOV ▾	RPU ▾		
1	22819	745	67191.17	3.26	90.19	2.94		

So werte ich Tests aus

```
SELECT
  (SELECT COUNT(DISTINCT user_pseudo_id)
   FROM `TABELLEN ID`)
```

BigQuery-Ressourcen suchen

Nur markierte Elemente anzeigen

▼

Repositories

Abfragen

(Klassische) Abfragen (32)

Notebooks

Daten-Canvasse

Datenvorbereitungen

Pipelines

Externe Verbindungen

▼

events_ (242)

pseudonymous_users_ (243)

users_ (244)

events_20250...

2025-06-22

Abfrage

Öffnen in

Freigeben

Schema

Details

Vorschau

Tabellen-Explorer

Vorschau

Statistiken

LI

Tabelleninformationen

Tabellen-ID	events_20250622
Erstellt	
Zuletzt geändert	
Tabellenablauf	NIE
Speicherort der Daten	
Standardsortierung	
Standard-Rundungsmodus	ROUNDING_MODE_UNSPECIFIED
Groß- und Kleinschreibung wird nicht berücksichtigt	false
Beschreibung	
Label	
Primärschlüssel	
Tags	

- Wir zählen alle Nutzer im Test
- Wir filtern die Eventquelle
- Wir filtern den Zeitraum
- Wir filtern den Test
- Wir filtern das Event

```

SELECT
  (SELECT COUNT(DISTINCT user_pseudo_id)
   FROM `gtm-123456789.analytics_123456789.events_*`
   WHERE event_name = 'Abtesting'
        AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
        AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325') AS total_users,

  COUNT(DISTINCT user_pseudo_id) AS users_with_transactions,

  ROUND(SUM(ecommerce.purchase_revenue), 2) AS Revenue,

  ROUND((COUNT(DISTINCT user_pseudo_id) /
        (SELECT COUNT(DISTINCT user_pseudo_id)
         FROM `gtm-123456789.analytics_123456789.events_*`
         WHERE event_name = 'Abtesting'
              AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
              AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325')) * 100, 2) AS CR,

  ROUND(SUM(ecommerce.purchase_revenue) / COUNT(DISTINCT user_pseudo_id), 2) AS AOV,

  ROUND(SUM(ecommerce.purchase_revenue) /
        (SELECT COUNT(DISTINCT user_pseudo_id)
         FROM `gtm-123456789.analytics_123456789.events_*`
         WHERE event_name = 'Abtesting'
              AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
              AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325'), 2) AS RPU

FROM `gtm-123456789.analytics_123456789.events_*`

WHERE _TABLE_SUFFIX BETWEEN '20250210' AND '20250325'
      AND ecommerce.purchase_revenue != 0
      AND user_pseudo_id IN (
        SELECT user_pseudo_id
        FROM `gtm-123456789.analytics_123456789.events_*`
        WHERE event_name = 'Abtesting'
              AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
              AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325');

```

total_users → Anzahl aller User im Test

```
SELECT
  (SELECT COUNT(DISTINCT user_pseudo_id)
   FROM `gtm-123456789.analytics_123456789.events_*`
   WHERE event_name = 'Abtesting'
   AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
   AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325') AS total_users,
```

```
COUNT(DISTINCT user_pseudo_id) AS users_with_transactions,
```

```
ROUND(SUM(ecommerce.purchase_revenue), 2) AS Revenue,
```

```
ROUND((COUNT(DISTINCT user_pseudo_id) /
  (SELECT COUNT(DISTINCT user_pseudo_id)
   FROM `gtm-123456789.analytics_123456789.events_*`
   WHERE event_name = 'Abtesting'
   AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
   AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325')) * 100, 2) AS CR,
```

```
ROUND(SUM(ecommerce.purchase_revenue) / COUNT(DISTINCT user_pseudo_id), 2) AS AOV,
```

```
ROUND(SUM(ecommerce.purchase_revenue) /
  (SELECT COUNT(DISTINCT user_pseudo_id)
   FROM `gtm-123456789.analytics_123456789.events_*`
   WHERE event_name = 'Abtesting'
   AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
   AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325'), 2) AS RPU
```

```
FROM `gtm-123456789.analytics_123456789.events_*`
```

```
WHERE _TABLE_SUFFIX BETWEEN '20250210' AND '20250325'
```

```
AND ecommerce.purchase_revenue != 0
```

```
AND user_pseudo_id IN (
```

```
  SELECT user_pseudo_id
```

```
  FROM `gtm-123456789.analytics_123456789.events_*`
```

```
  WHERE event_name = 'Abtesting'
```

```
  AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
```

```
  AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325');
```

- Wir zählen Käufer
- Wir summieren Umsatz

users_with_transactions → Anzahl der Käufer

purchase_revenue → Gesamtumsatz im Testzeitraum

```

SELECT
  (SELECT COUNT(DISTINCT user_pseudo_id)
   FROM `gtm-123456789.analytics_123456789.events_*`
   WHERE event_name = 'Abtesting'
   AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
   AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325') AS total_users,

  COUNT(DISTINCT user_pseudo_id) AS users_with_transactions,

  ROUND(SUM(ecommerce.purchase_revenue), 2) AS Revenue,

  ROUND((COUNT(DISTINCT user_pseudo_id) /
   (SELECT COUNT(DISTINCT user_pseudo_id)
    FROM `gtm-123456789.analytics_123456789.events_*`
    WHERE event_name = 'Abtesting'
    AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
    AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325')) * 100, 2) AS CR,

  ROUND(SUM(ecommerce.purchase_revenue) / COUNT(DISTINCT user_pseudo_id), 2) AS AOV,

  ROUND(SUM(ecommerce.purchase_revenue) /
   (SELECT COUNT(DISTINCT user_pseudo_id)
    FROM `gtm-123456789.analytics_123456789.events_*`
    WHERE event_name = 'Abtesting'
    AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
    AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325'), 2) AS RPU

FROM `gtm-123456789.analytics_123456789.events_*`

WHERE _TABLE_SUFFIX BETWEEN '20250210' AND '20250325'
AND ecommerce.purchase_revenue != 0
AND user_pseudo_id IN (
  SELECT user_pseudo_id
  FROM `gtm-123456789.analytics_123456789.events_*`
  WHERE event_name = 'Abtesting'
  AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
  AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325');

```

- Wir berechnen die CR
- Wir berechnen den AOV

AOV (Average Order Value) → Umsatz geteilt durch Anzahl der Käufer (nur User mit Käufen)

```

SELECT
  (SELECT COUNT(DISTINCT user_pseudo_id)
   FROM `gtm-123456789.analytics_123456789.events_*`
   WHERE event_name = 'Abtesting'
   AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
   AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325') AS total_users,

  COUNT(DISTINCT user_pseudo_id) AS users_with_transactions,

  ROUND(SUM(ecommerce.purchase_revenue), 2) AS Revenue,

  ROUND((COUNT(DISTINCT user_pseudo_id) /
   (SELECT COUNT(DISTINCT user_pseudo_id)
    FROM `gtm-123456789.analytics_123456789.events_*`
    WHERE event_name = 'Abtesting'
    AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
    AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325')) * 100, 2) AS CR,

  ROUND(SUM(ecommerce.purchase_revenue) / COUNT(DISTINCT user_pseudo_id), 2) AS AOV,

```

```

ROUND(SUM(ecommerce.purchase_revenue) /
 (SELECT COUNT(DISTINCT user_pseudo_id)
  FROM `gtm-123456789.analytics_123456789.events_*`
  WHERE event_name = 'Abtesting'
  AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
  AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325'), 2) AS RPU

```

```

FROM `gtm-123456789.analytics_123456789.events_*`

WHERE _TABLE_SUFFIX BETWEEN '20250210' AND '20250325'
AND ecommerce.purchase_revenue != 0
AND user_pseudo_id IN (
  SELECT user_pseudo_id
  FROM `gtm-123456789.analytics_123456789.events_*`
  WHERE event_name = 'Abtesting'
  AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
  AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325');

```

RPU (Revenue per User) → Umsatz geteilt durch alle User im Test (auch ohne Kauf)

```

SELECT
  (SELECT COUNT(DISTINCT user_pseudo_id)
   FROM `gtm-123456789.analytics_123456789.events_*`
   WHERE event_name = 'Abtesting'
   AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
   AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325') AS total_users,

  COUNT(DISTINCT user_pseudo_id) AS users_with_transactions,

  ROUND(SUM(ecommerce.purchase_revenue), 2) AS Revenue,

  ROUND((COUNT(DISTINCT user_pseudo_id) /
   (SELECT COUNT(DISTINCT user_pseudo_id)
    FROM `gtm-123456789.analytics_123456789.events_*`
    WHERE event_name = 'Abtesting'
    AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
    AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325')) * 100, 2) AS CR,

  ROUND(SUM(ecommerce.purchase_revenue) / COUNT(DISTINCT user_pseudo_id), 2) AS AOV,

  ROUND(SUM(ecommerce.purchase_revenue) /
   (SELECT COUNT(DISTINCT user_pseudo_id)
    FROM `gtm-123456789.analytics_123456789.events_*`
    WHERE event_name = 'Abtesting'
    AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
    AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325'), 2) AS RPU

```

```
FROM `gtm-123456789.analytics_123456789.events_*`
```

```

WHERE _TABLE_SUFFIX BETWEEN '20250210' AND '20250325'
  AND ecommerce.purchase_revenue != 0
  AND user_pseudo_id IN (
    SELECT user_pseudo_id
    FROM `gtm-123456789.analytics_123456789.events_*`
    WHERE event_name = 'Abtesting'
      AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
      AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325');

```

- Wir filtern 0€ Käufe raus

Skripte zum Anpassen

```
SELECT
  (SELECT COUNT(DISTINCT user_pseudo_id)
   FROM `gtm-123456789.analytics_123456789.events_*`
   WHERE event_name = 'Abtesting'
        AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
        AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325') AS total_users,

  COUNT(DISTINCT user_pseudo_id) AS users_with_transactions,

  ROUND(SUM(ecommerce.purchase_revenue), 2) AS Revenue,

  ROUND((COUNT(DISTINCT user_pseudo_id) /
   (SELECT COUNT(DISTINCT user_pseudo_id)
    FROM `gtm-123456789.analytics_123456789.events_*`
    WHERE event_name = 'Abtesting'
         AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
         AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325')) * 100, 2) AS CR,

  ROUND(SUM(ecommerce.purchase_revenue) / COUNT(DISTINCT user_pseudo_id), 2) AS AOV,

  ROUND(SUM(ecommerce.purchase_revenue) /
   (SELECT COUNT(DISTINCT user_pseudo_id)
    FROM `gtm-123456789.analytics_123456789.events_*`
    WHERE event_name = 'Abtesting'
         AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
         AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325'), 2) AS RPU

FROM `gtm-123456789.analytics_123456789.events_*`

WHERE _TABLE_SUFFIX BETWEEN '20250210' AND '20250325'
      AND ecommerce.purchase_revenue != 0
      AND user_pseudo_id IN (
        SELECT user_pseudo_id
        FROM `gtm-123456789.analytics_123456789.events_*`
        WHERE event_name = 'Abtesting'
             AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
             AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325');
```

Beispiel: Safe Space

 Plan: Pro Account

 0

DASHBOARD / Test Results

Demo A/B Test

Test Results

Experiment-ID

21305

Visitors

0

Start date

23.5.2025 

Days Running

20

Status

active

Variation Overview

1. Original

Traffic: 50%

Variation-ID:
-

2. Variation 1

Traffic: 50%

Variation-ID:
31367

Space for your notes or a hypothesis





23.5.2025 - 12.6.2025

Update Results

Add GA4 goal

Add BigQuery goal

33

Es braucht kein IT Studium!

ChatGPT 4o 

 Gemein:

Bitte ergänze die SQL Abfrage und füge ganz unten eine weitere Bedingung ein, dass nutzer, die über 300€ bestellt haben nicht berücksichtigt werden

```
SELECT
(SELECT COUNT(DISTINCT user_pseudo_id)
FROM gtm-w8s7h6x2-
WHERE event_name = 'Abtesting'
AND (SELECT value.string_value FROM UNNEST(event_params)
WHERE key = 'varify_abTestShort') = '17145:Original'
AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325')
AS total_users,
COUNT(DISTINCT user_pseudo_id) AS users_with_transactions,
ROUND(SUM(ecommerce.purchase_revenue), 2) AS Revenue,
ROUND((COUNT(DISTINCT user_pseudo_id) /
(SELECT COUNT(DISTINCT user_pseudo_id)
FROM gtm-w8s7h6x2-
WHERE event_name = 'Abtesting'
AND (SELECT value.string_value FROM
UNNEST(event_params) WHERE key = 'varify_abTestShort') =
```

Stelle irgendeine Frage

+  Tools

ChatGPT kann Fehler machen. Überprüfe wichtige Informationen. Siehe [Cookie-Voreinstellungen](#).

Es braucht kein IT Studium!

Hier ist deine ergänzte Abfrage.

Ich habe ganz unten eine weitere Bedingung eingefügt, die nur Nutzer mit einer **Gesamtbestellsumme ≤ 300€** berücksichtigt:

sql

Kopieren Bearbeiten

```
SELECT
  (SELECT COUNT(DISTINCT user_pseudo_id)
   FROM `ecommerce.events_*`
   WHERE event_name = 'Abtesting'
        AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestS'
        AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325') AS total_users,
  COUNT(DISTINCT user_pseudo_id) AS users_with_transactions,
  ROUND(SUM(ecommerce.purchase_revenue), 2) AS Revenue,
  ROUND((COUNT(DISTINCT user_pseudo_id) /
        (SELECT COUNT(DISTINCT user_pseudo_id)
         FROM `ecommerce.events_*`
         WHERE event_name = 'Abtesting'
              AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTe'
              AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325')) * 100, 2) AS CR,
  ROUND(SUM(ecommerce.purchase_revenue) / COUNT(DISTINCT user_pseudo_id), 2) AS AOV,
  ROUND(SUM(ecommerce.purchase_revenue) /
        (SELECT COUNT(DISTINCT user_pseudo_id,
```

Es braucht kein IT Studium!

collection_001.1 - Original

Ausführen

(Klassische) Abfrage speichern

Freigeben

Planen

```
1 SELECT
2   (SELECT COUNT(DISTINCT user_pseudo_id)
3    FROM `gtm-`
4    WHERE event_name = 'Abtesting'
5          AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
6          AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325') AS total_users,
7   COUNT(DISTINCT user_pseudo_id) AS users_with_transactions,
8   ROUND(SUM(ecommerce.purchase_revenue), 2) AS Revenue,
9   ROUND((COUNT(DISTINCT user_pseudo_id) /
10    (SELECT COUNT(DISTINCT user_pseudo_id)
11     FROM `g`
12     WHERE event_name = 'Abtesting'
13           AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
14           AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325')) * 100, 2) AS CR,
15   ROUND(SUM(ecommerce.purchase_revenue) / COUNT(DISTINCT user_pseudo_id), 2) AS AOV,
16   ROUND(SUM(ecommerce.purchase_revenue) /
17    (SELECT COUNT(DISTINCT user_pseudo_id)
18     FROM `g`
19     WHERE event_name = 'Abtesting'
20           AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
21           AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325'), 2) AS RPU
22 FROM `gtm-`
23 WHERE _TABLE_SUFFIX BETWEEN '20250210' AND '20250325'
24 AND ecommerce.purchase_revenue != 0
25 AND user_pseudo_id IN (
26   SELECT user_pseudo_id
27   FROM `g`
28   WHERE event_name = 'Abtesting'
29         AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
30         AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325');
31
```

✓

Diese Abfrage verarbeitet bei der Ausführung 2,02 GB.

Abfrageergebnisse

Jobinformationen		Ergebnisse	Diagramm	JSON	Ausführungsdetails		Ausführungsgrafik	
ZEILE	total_users	users_with_trans...	Revenue	CR	AOV	RPU		
1	22819	745	67191.17	3.26	90.19	2.94		

Unbenannte Abfrage

Ausführen

Speichern

Herunterladen

Freigeben

Planen

```
16 ROUND(SUM(ecommerce.purchase_revenue) /
17   (SELECT COUNT(DISTINCT user_pseudo_id)
18    FROM `g`
19    WHERE event_name = 'Abtesting'
20          AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
21          AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325'), 2) AS RPU
22 FROM `gtm-`
23 WHERE _TABLE_SUFFIX BETWEEN '20250210' AND '20250325'
24 AND ecommerce.purchase_revenue != 0
25 AND user_pseudo_id IN (
26   SELECT user_pseudo_id
27   FROM `g`
28   WHERE event_name = 'Abtesting'
29         AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '17145:Original'
30         AND _TABLE_SUFFIX BETWEEN '20250210' AND '20250325'
31 )
32 AND user_pseudo_id IN (
33   SELECT user_pseudo_id
34   FROM (
35     SELECT user_pseudo_id, SUM(ecommerce.purchase_revenue) AS total_revenue
36     FROM `gtm-`
37     WHERE _TABLE_SUFFIX BETWEEN '20250210' AND '20250325'
38     AND ecommerce.purchase_revenue != 0
39     GROUP BY user_pseudo_id
40   )
41   WHERE total_revenue <= 300
42 );
43
```

✓

Abfrage abgeschlossen

Abfrageergebnisse

Jobinformationen

Ergebnisse

Diagramm

JSON

Ausführungsdetails

Ausführungsgrafik

ZEILE	total_users	users_with_trans...	Revenue	CR	AOV	RPU
1	22819	723	57764.1	3.17	79.9	2.53

Nutzung von Events aus dem GTM

- Bei A/B-Tests gibt es oft wichtige Events, die in der Auswertung aber vergessen werden
- Beispiel: wenn ich im Sidecart teste, muss ich wissen → wurde der Sidecart überhaupt geöffnet?
- Solche Events können über den Google Tag Manager erstellt und an GA4 gesendet werden
 - z. B. Event „sidecart geöffnet“
 - z. B. Event „all search open“
- In BigQuery kann ich diese Events später in der Abfrage nutzen
 - als Filter oder Segment → z. B. nur Nutzer auswerten, die den Sidecart geöffnet haben
- Jedes Event kann als Zusatz-Filter genutzt werden
 - wichtig: es muss im GTM sauber angelegt und korrekt gefeuert werden

Nutzung von Events aus dem GTM

×

21 Sichtbarkeit von Elementen >

GA4 - Event - Sidecart geöffnet

✓

Ausgelöst

Tag-Details

Properties

Name	Wert
Typ	Google Analytics: GA4-Ereignis
Auslösestatus	Erfolgreich
E-Commerce-Daten senden	false
Include user-provided data from your website	false
Ereignisname	"sidecart_geöffnet"
Mess-ID	"ALT - GA4 - ID"

Nutzung von Events aus dem GTM

```
SELECT
  (SELECT COUNT(DISTINCT user_pseudo_id)
   FROM `gtm-123456789.analytics_123456789.events_*`
   WHERE event_Name = 'Abtesting'
   AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'verify_abTestShort') = '14767:Original'
   AND _TABLE_SUFFIX BETWEEN '20241212' AND '20250121'
   AND user_pseudo_id IN (
     SELECT user_pseudo_id
     FROM `gtm-123456789.analytics_123456789.events_*`
     WHERE event_name = 'sidecart_geöffnet'
     AND _TABLE_SUFFIX BETWEEN '20241212' AND '20250121')) AS total_users,
```

Mein Workflow

Mein Workflow

1. Testplanung 
2. Testvorbereitung 
3. Test-Implementierung 
4. Kontrolle / BigQuery Abfragen aufsetzen 
5. Auswertung Kohorten, Segmente usw. 

Spezielle Abfragen

```
SELECT
  user_pseudo_id,
  ROUND(SUM(ecommerce.purchase_revenue), 2) AS Purchase_Revenue,
  MAX(ecommerce.transaction_id) AS Transaction_ID,
  FORMAT_DATETIME('%T', DATETIME(TIMESTAMP_MICROS(event_timestamp), 'Europe/Berlin')) AS Berlin_Time
FROM `cg-sky-12345.123456.events_*`
WHERE _TABLE_SUFFIX BETWEEN '20231128' AND '20231128'
  AND ecommerce.purchase_revenue != 0
  AND ecommerce.purchase_revenue > 60
  AND user_pseudo_id IN (
    SELECT user_pseudo_id
    FROM `cg-sky-12345.123456.events_*`
    WHERE event_name = 'ablyft_abtest'
      AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'experiment_name') = 'test_cart_002'
      AND (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'variation_name') = 'Variation #1'
      AND _TABLE_SUFFIX BETWEEN '20231128' AND '20231128'
      AND user_pseudo_id IN (
        SELECT user_pseudo_id
        FROM `cg-sky-12345.123456.events_*`
        WHERE event_name = 'sidecart_geöffnet'
          AND _TABLE_SUFFIX BETWEEN '20231128' AND '20231128'
      )
  )
GROUP BY user_pseudo_id, event_timestamp
ORDER BY Berlin_Time;
```

Spezielle Abfragen

- Andere Software: ablyft
- Andere Parameter zum Abfragen
- Kürzere Abfrage weil es keine Berechnungen gibt

Tipps & Tools aus der Praxis

ChatGPT als SQL-Assistent

- Ich nutze ChatGPT oft als Hilfe beim Schreiben von SQL-Abfragen
- ChatGPT hilft bei:
 - Syntax und Aufbau von Abfragen
 - Umwandeln von Ideen in SQL
 - Optimieren und Erklären von bestehenden Queries
- Besonders praktisch für komplexere Fragen und Filterlogik
 - spart viel Zeit beim Erstellen und Anpassen von Queries
- Ich habe ein BigQuery Projekt und arbeite dort mit einem GBT das SQL Profi ist

Looker Studio

- Wird oft genutzt, um BigQuery-Daten in Dashboards zu visualisieren
- Für A/B-Tests kann man z. B. eigene Test-Dashboards bauen
→ z. B. Käufe pro Variante im Zeitverlauf anzeigen
- Ich selbst nutze es aktuell nicht — weil ich die Auswertung direkt in BigQuery / SQL mache.
- Muss eingestellt werden, dass Daten nicht die ganze Zeit synchronisiert!



Looker Studio

Looker Studio

Experiment ID: 11110 (1) ▾

Event Name: page_view (1) ▾

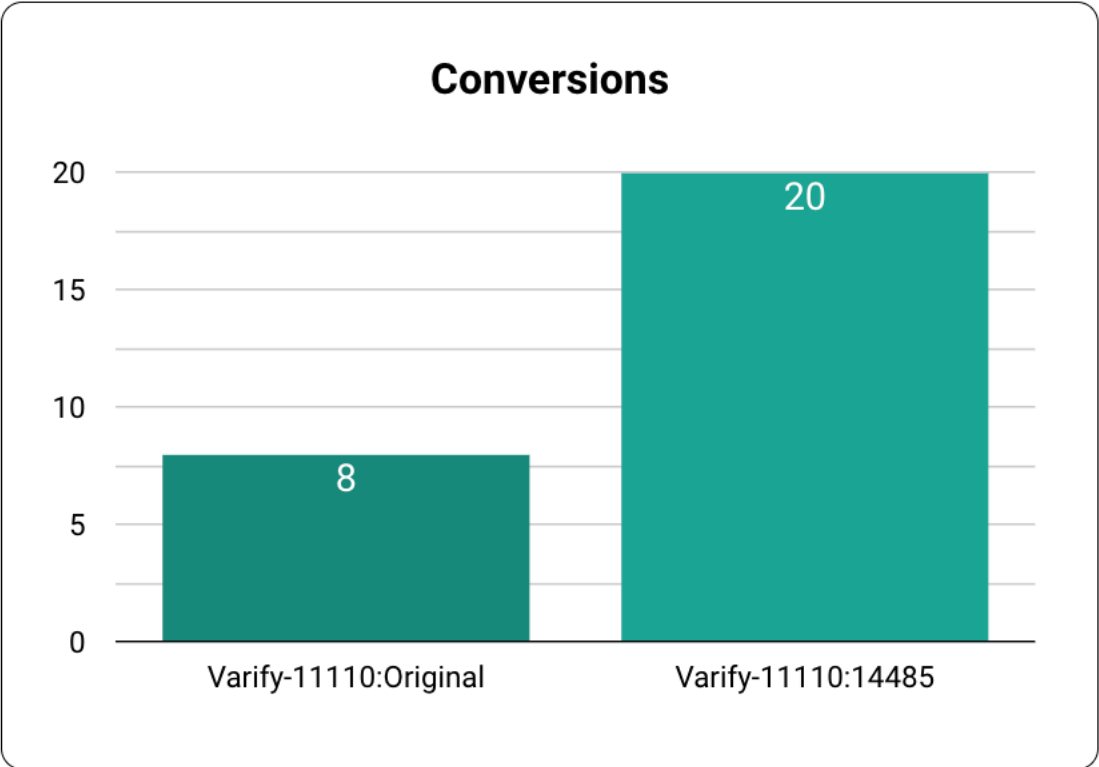
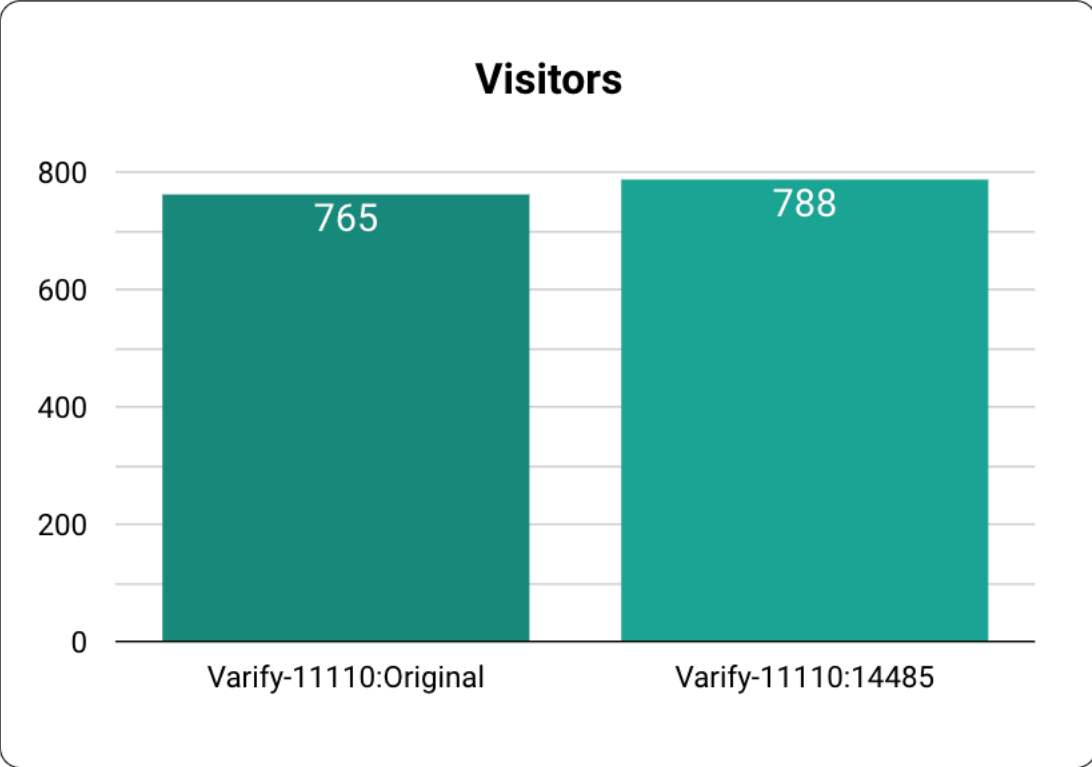
01.01.2024 - 31.12.2024 ▾

Varify.io BigQuery Looker Studio Dashboard



Varify experiment data is based on BigQuery data. The statistical calculation of significance is conducted using a two-sided Chi-Square test. The significance calculation is only valid if the number of conversions is less than the number of visitors. It is not applicable for revenue values. Improvements and significance calculations are always made in comparison to the original.

		Visitors	Conversions	Conversion Rate	Improvement	Confidence	Significance
1.	Varify-11110:Original	765	8	1.05%	-	-	Nicht signifikant
2.	Varify-11110:14485	788	20	2.54%	142.70%	97%	Signifikant (p < 0....



Safe spaces

	Visitors	Conversions	Conversion Rate	Improvement	Confidence	Significant
Original	84,895	1,730	2.04%	-	-	-
Variation 1	84,897	1,790	2.11%	4.46	83.8%	No
 No statistically significant differences were found. Please continue testing to ensure a sufficient sample size.						

Goal 3: Add to cartsGA4

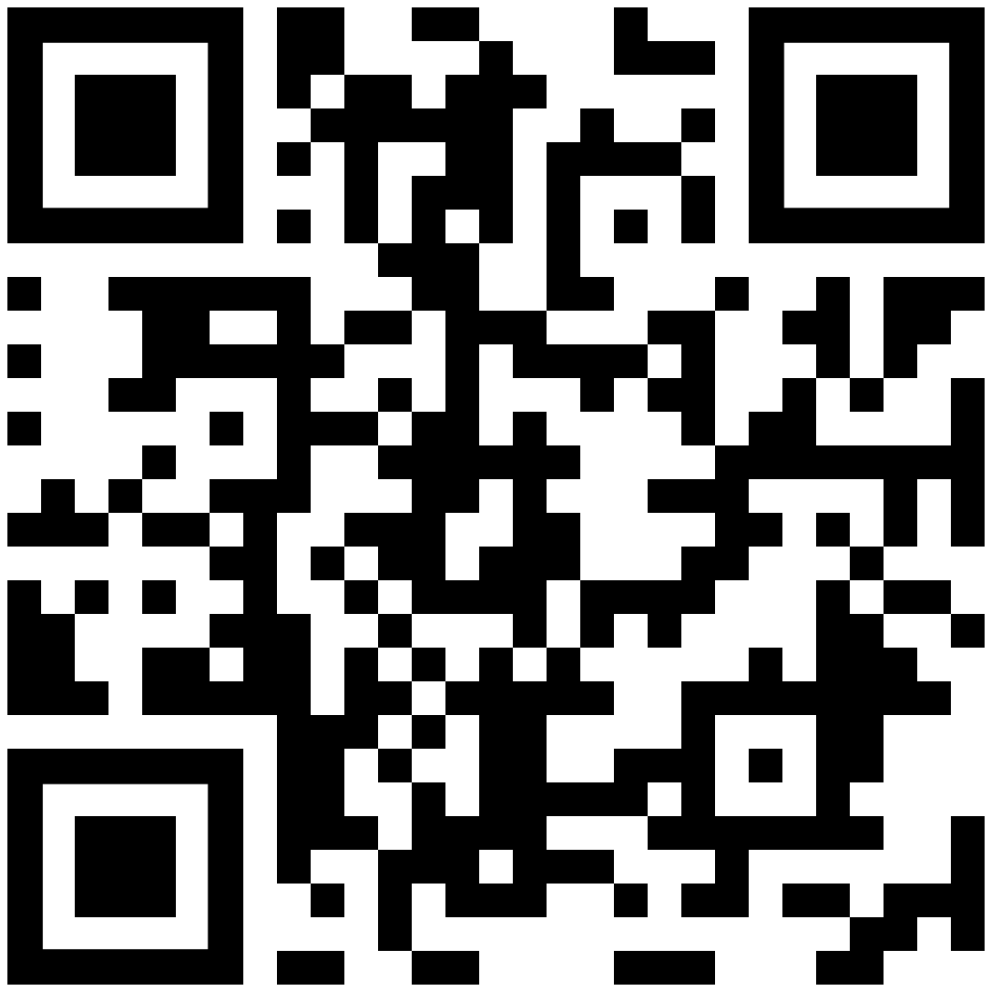
	Visitors	Conversions	Conversion Rate	Improvement	Confidence	Significant
Original	84,698	6,314	7.45%	-	-	-
Variation 1	84,717	6,754	7.97%	6.94%	100.0%	Yes

Goal 4: Add to carts (count)Big Query

	Visitors	Conversions	Conversion Rate	Improvement	Confidence	Significant
Original	84,895	6,340	7.47%	-	-	-
Variation 1	84,897	6,790	7.99%	7.27%	100.0%	Yes

Goal 5: AbtestingGA4

	Visitors	Event Count	Conversion Rate	Improvement	Confidence	Significant
Original	84,698	84,662	99.96%	-	-	-
Variation 1	84,717	84,687	99.96%	0.01%	77.0%	No
 No statistically significant differences were found. Please continue testing to ensure a sufficient sample size.						



BigQuery Access

Verbindung GA4 & BigQuery

GA4 & BigQuery verknüpfen

Produktverknüpfungen

Mit diesen Einstellungen können Sie festlegen, welche Produkte mit dieser Property verknüpft werden



Google AdSense-Verknüpfungen



Google Ads-Verknüpfungen



Ad Manager-Verknüpfungen



BigQuery-Verknüpfungen



Display & Video 360 – Verknüpfungen



Floodlight-Verknüpfungen



Merchant Center-Verknüpfungen



Google Play-Verknüpfungen



Search Ads 360-Verknüpfungen



Search Console-Verknüpfungen

 BigQuery-Verknüpfungen

Suchen

Verknüpfen

Projekt-ID	Projektname	Projektnummer
------------	-------------	---------------

GA4 & BigQuery verknüpfen

Details zur erstellten Verknüpfung

Projekt-ID

Projektname

Projektnummer

Standardort für Datenpoolerstellung

Europäische Union (eu)

Erstellt von

Erstellungsdatum

07.05.2025

Datenkonfigurationen

Ereignisdaten

Datenstreams und Ereignisse

Legen Sie fest, welche Datenstreams und Ereignisse exportiert werden sollen. Alle Ereignismengen sind Schätzwerte. Ob das Tageslimit durchgesetzt wird, hängt von der tatsächlich exportierten Menge ab. [Weitere Informationen](#)

GESCHÄTZTE GESAMTZAHL DER TÄGLICHEN ereignisse, die zu exportieren sind

0,02 / 1 Million(en) (Tageslimit)

2 von 2 Streams ausgewähltKeine Ereignisse ausgeschlossen

[Datenstreams und Ereignisse konfigurieren](#)

☐ Werbe-IDs für App-Streams einbeziehen

Exporttyp

☒ Täglich

Die Daten werden einmal täglich komplett exportiert

☐ Streaming (bestmöglich)

Fortlaufender Export, innerhalb von Sekunden nach Ereigniseintritt. Daten werden bestmöglich exportiert, ohne Garantie für Vollständigkeit. [Weitere Informationen](#)

Nutzerdaten

Alle Nutzer mit Aktivitäten für den aktuellen Tag werden exportiert, wenn sich eines der [Attribute des Nutzers](#) geändert hat. Der Export von Nutzerdaten wird pausiert, wenn mehr Ereignisdaten exportiert werden, als das Limit erlaubt. [Weitere Informationen zum Exportieren von BigQuery-Nutzerdaten](#)

Exporttyp

☒ Täglich

Die Daten werden einmal täglich komplett exportiert

Täglich ausgewählt, damit nicht die ganze Zeit große Datenmengen gestreamt werden!

51

Oder via varify.io per auto setup

Analytics

Google Analytics 4

Data Transfer

Manual

Test Evaluation

GA4 and Varify

Consent Mode

Per Consent

Google Authenticator Status: ● connected

Disconnect

Account

Property

BJJ Grappling

BJJ-Grappling.de - GA4

Use Bigquery Data ● activated

Off

Save

Privacy

Privacy Settings

Nächste Schritte

- Daten vergleichen für die Single Source of Truth
- Simple SQL Skripte ausprobieren
- Austauschen mit Experten
- BigQuery über varify.io kostenlos ausprobieren (Stichtag: 31.07)
- Mitglied unserer (wirklich) exklusiven Slack Community werden

Mehr Umsatz für deinen Online-Shop
Fragen zum Webinar? Lass uns sprechen!



Niko Spies
CRO Experte

Steffen Schulz
CEO Varify.io

