**Справочник по BigQuery**

<https://support.google.com/analytics/answer/4419694?hl=ru#zippy=%2C%D1%81%D0%BE%D0%B4%D0%B5%D1%80%D0%B6%D0%B0%D0%BD%D0%B8%D0%B5>

|  |  |
| --- | --- |
|  | *Эта функция доступна только в Аналитике 360, входящей в Google Платформу для маркетинга. Подробнее*[*о Google Платформе для маркетинга*](https://marketingplatform.google.com/)*…* |

Эта статья содержит примеры составления запросов данных Google Analytics, экспортируемым в BigQuery. Опробовать эти запросы можно на специально созданном [образце набора данных](https://support.google.com/analytics/answer/7586738).

*Содержание*

* [Оптимизация запросов](https://support.google.com/analytics/answer/4419694?hl=ru#queryOptimization)
* [Советы и рекомендации](https://support.google.com/analytics/answer/4419694?hl=ru#tips)
  + [Использование образца набора данных](https://support.google.com/analytics/answer/4419694?hl=ru#sample)
  + [Использование стандартного SQL](https://support.google.com/analytics/answer/4419694?hl=ru#standardsql)
  + [Запрос нескольких таблиц](https://support.google.com/analytics/answer/4419694?hl=ru#multiple)
    - [Данные за 3 дня](https://support.google.com/analytics/answer/4419694?hl=ru#3days)
    - [Данные за последние 1095 дней](https://support.google.com/analytics/answer/4419694?hl=ru#1095days)
    - [Данные за последние 36 месяцев](https://support.google.com/analytics/answer/4419694?hl=ru#36months)
    - [Данные за последние 3 года](https://support.google.com/analytics/answer/4419694?hl=ru#3years)
    - [Данные за указанный период](https://support.google.com/analytics/answer/4419694?hl=ru#specific)
    - [Данные за последние 3 года и текущий день](https://support.google.com/analytics/answer/4419694?hl=ru#3yearsplus)
* [Простые запросы: примеры](https://support.google.com/analytics/answer/4419694?hl=ru#basicQueryExamples)
  + [Суммарный *[показатель]* на *[параметр]*](https://support.google.com/analytics/answer/4419694?hl=ru#query1_TotalMetricPerDimension)
  + [Средний показатель отказов на *[параметр]*](https://support.google.com/analytics/answer/4419694?hl=ru#query2_AvgBounceRatePerDimension)
  + [Среднее количество просмотров страниц отдельных товаров по типу пользователей](https://support.google.com/analytics/answer/4419694?hl=ru#query3_AvgNumOfProductPageviewsForPurchasers)
  + [Среднее число транзакций на покупателя](https://support.google.com/analytics/answer/4419694?hl=ru#query4_AvgNumOfTransPerPurchaser)
  + [Средняя сумма расходов за сеанс](https://support.google.com/analytics/answer/4419694?hl=ru#query5_AvgAmountOfMoneySpentPerVisit)
  + [Последовательность обращений](https://support.google.com/analytics/answer/4419694?hl=ru#query6_SequenceOfHits)
  + [Несколько специальных параметров на уровне обращений или сеансов](https://support.google.com/analytics/answer/4419694?hl=ru#query7_MultipleCDs)
* [Сложные запросы: примеры](https://support.google.com/analytics/answer/4419694?hl=ru#advancedQueryExamples)
  + [Товары, приобретенные пользователями, также купившими товар А (обычная электронная торговля)](https://support.google.com/analytics/answer/4419694?hl=ru#query7_ProductsPurchasedByCustomersWhoPurchasedProductA)
  + [Товары, приобретенные пользователями, также купившими товар А (расширенная электронная торговля)](https://support.google.com/analytics/answer/4419694?hl=ru#query7A_ProductsPurchasedByCustomersWhoPurchasedProductA)
  + [Среднее количество взаимодействий пользователя перед покупкой](https://support.google.com/analytics/answer/4419694?hl=ru#query8_AvgNumOfUserInteractionsBeforePurchase)
  + [Процент проданных запасов товара](https://support.google.com/analytics/answer/4419694?hl=ru#query9_PctOfStockSoldPerProduct)
  + [Прибыльность каждого товара](https://support.google.com/analytics/answer/4419694?hl=ru#query10_ProfitabilityOfEachProduct)
  + [Фактическая прибыльность каждого товара](https://support.google.com/analytics/answer/4419694?hl=ru#query11_RealProfitabilityOfEachProduct)

Оптимизация запросов

Каждый запрос учитывается при расчете объема обработанных данных. Если выбрано слишком много полей, повышается количество обрабатываемых данных. В результате чрезмерно растет их суммарный объем. Оптимизируя запросы, вы можете эффективно использовать месячную норму обработки данных.

[Подробнее о ценах...](https://developers.google.com/bigquery/pricing)

Выбирайте только то, что вам нужно

Формулируя запрос, выбирайте только релевантные поля в операторе SELECT. Это позволит сократить объем данных и время на обработку запроса.

Пример: избегайте использования подстановочных знаков.

| **Неудачный вариант: использование подстановочного знака** |
| --- |
| SELECT \* FROM [table name]; |

| **Удачный вариант: использование названий полей во избежание ненужной обработки** |
| --- |
| SELECT field1, field2 FROM [table name]; |

Разрешите кеширование

По возможности избегайте использования функций в качестве полей. Функции (такие как NOW() или TODAY()) возвращают переменные результаты, что препятствует кешированию запросов, ускоряющему их обработку. Вместо этого указывайте точные значения времени и даты.

В настоящее время кешированные результаты не поддерживаются для запросов к нескольким таблицам с использованием подстановочного знака, даже если установлен флажок "Использовать кешированные результаты". Если вы выполняете один и тот же запрос с подстановочными знаками несколько раз, оплата взимается за каждый запрос. [Подробнее…](https://cloud.google.com/bigquery/docs/querying-wildcard-tables)

Используйте промежуточные таблицы для часто используемых подзапросов

Если определенный запрос часто используется в качестве подзапроса, его можно сохранить в качестве промежуточной таблицы, нажав Save as Table (Сохранить в таблицу) над результатами запроса. Затем на эту таблицу можно ссылаться в разделе FROM запроса, что позволит уменьшить как объем данных, так и время на их обработку.

| **Использование промежуточной таблицы** |
| --- |
| SELECT field1, field2 FROM [Dataset name.table name]; |

Отладка запросов

BigQuery отлаживает код по мере его составления. В окне кода данные отладки выводятся непосредственно под запросом. Отладку можно также осуществлять в API при использовании флага dryRun.

Допустимым запросам соответствует зеленый индикатор, нажав на который можно узнать объем обработанных данных. Это позволяет оптимизировать данные до выполнения запроса, так что вы можете избежать ненужной обработки данных.



Недопустимым запросам соответствует красный индикатор, нажав на который можно просмотреть информацию об ошибке, а также найти строку и столбец, где она возникла. Приведенный ниже пример содержит пустой оператор GROUP BY, который вызвал ошибку.



Советы и рекомендации

Использование образца набора данных

В приведенных ниже примерах используется [образец набора данных Google Analytics](https://bigquery.cloud.google.com/table/bigquery-public-data:google_analytics_sample.ga_sessions_20170801).

Чтобы использовать запрос для собственных данных, просто замените названия проекта и набора данных в примерах.

Использование стандартного или устаревшего SQL

BigQuery поддерживает два диалекта SQL:

* [Стандартный SQL](https://cloud.google.com/bigquery/sql-reference/index)
* [Устаревший SQL](https://cloud.google.com/bigquery/query-reference)

О различиях между ними читайте в [*этой статье*](https://cloud.google.com/bigquery/docs/reference/standard-sql/migrating-from-legacy-sql).

Сейчас для запросов данных, передаваемых на хранение в BigQuery, рекомендуется использовать стандартный SQL.

О том, как включить стандартный SQL в BigQuery, ИКС, API или другом интерфейсе читайте в этой [статье](https://cloud.google.com/bigquery/docs/reference/standard-sql/enabling-standard-sql).

Самый простой способ начать работу – добавить комментарий "standardSQL" в верхней части запроса на стандартном SQL, как показано в примерах ниже.

Если вы используете устаревший SQL, то данные Google Аналитики 360 будут каждый день сохраняться в новой таблице. Чтобы запросить сразу несколько таблиц, можно перечислить их названия через запятую, воспользоваться [подстановочной функцией](https://cloud.google.com/bigquery/query-reference#tablewildcardfunctions) TABLE\_DATE\_RANGE или даже несколькими функциями TABLE\_DATE\_RANGE через запятую, как в примерах ниже.

Запрос нескольких таблиц

Ниже приведены примеры запросов на стандартном и устаревшем SQL для одних и тех же данных.

Данные за 3 дня

**Стандартный SQL**

**Данные за 3 дня, UNION ALL**

#standardSQL  
WITH ga\_tables AS (  
SELECT  
date,  
SUM(totals.visits) AS visits,  
SUM(totals.pageviews) AS pageviews,  
SUM(totals.transactions) AS transactions,  
SUM(totals.transactionRevenue)/1000000 AS revenue  
FROM `bigquery-public-data.google\_analytics\_sample.ga\_sessions\_20160801`  
GROUP BY date  
  
UNION ALL  
  
SELECT  
date,  
SUM(totals.visits) AS visits,  
SUM(totals.pageviews) AS pageviews,  
SUM(totals.transactions) AS transactions,  
SUM(totals.transactionRevenue)/1000000 AS revenue  
FROM `bigquery-public-data.google\_analytics\_sample.ga\_sessions\_20160802`  
GROUP BY date  
  
UNION ALL  
  
SELECT  
date,  
SUM(totals.visits) AS visits,  
SUM(totals.pageviews) AS pageviews,  
SUM(totals.transactions) AS transactions,  
SUM(totals.transactionRevenue)/1000000 AS revenue  
FROM `bigquery-public-data.google\_analytics\_sample.ga\_sessions\_20160803`  
GROUP BY date  
  
)  
SELECT  
date,  
visits,  
pageviews,  
transactions,  
revenue,  
FROM ga\_tables  
ORDER BY date ASC

Данные за последние 1095 дней

**Стандартный SQL**

**Данные за последние 1095 дней, \_TABLE\_SUFFIX**

#standardSQL  
SELECT  
date,  
SUM(totals.visits) AS visits,  
SUM(totals.pageviews) AS pageviews,  
SUM(totals.transactions) AS transactions,  
SUM(totals.transactionRevenue)/1000000 AS revenue  
FROM `bigquery-public-data.google\_analytics\_sample.ga\_sessions\_\*`  
WHERE  
\_TABLE\_SUFFIX BETWEEN FORMAT\_DATE('%Y%m%d',DATE\_SUB(CURRENT\_DATE(), INTERVAL 1095 DAY))  
AND  
FORMAT\_DATE('%Y%m%d',DATE\_SUB(CURRENT\_DATE(), INTERVAL 1 DAY))  
GROUP BY date  
ORDER BY date ASC

Данные за последние 36 месяцев

**Стандартный SQL**

**Данные за последние 36 месяцев, \_TABLE\_SUFFIX**

#standardSQL  
SELECT  
date,  
SUM(totals.visits) AS visits,  
SUM(totals.pageviews) AS pageviews,  
SUM(totals.transactions) AS transactions,  
SUM(totals.transactionRevenue)/1000000 AS revenue  
FROM `bigquery-public-data.google\_analytics\_sample.ga\_sessions\_\*`  
WHERE  
\_TABLE\_SUFFIX BETWEEN FORMAT\_DATE('%Y%m%d',DATE\_SUB(CURRENT\_DATE(), INTERVAL 36 MONTH))  
AND  
FORMAT\_DATE('%Y%m%d',DATE\_SUB(CURRENT\_DATE(), INTERVAL 1 DAY))  
GROUP BY date  
ORDER BY date ASC

Данные за последние 3 года

**Стандартный SQL**

**Данные за последние 3 года, \_TABLE\_SUFFIX**

#standardSQL  
SELECT  
date,  
SUM(totals.visits) AS visits,  
SUM(totals.pageviews) AS pageviews,  
SUM(totals.transactions) AS transactions,  
SUM(totals.transactionRevenue)/1000000 AS revenue  
FROM `bigquery-public-data.google\_analytics\_sample.ga\_sessions\_\*`  
WHERE  
\_TABLE\_SUFFIX BETWEEN FORMAT\_DATE('%Y%m%d',DATE\_SUB(CURRENT\_DATE(), INTERVAL 3 YEAR))  
AND  
FORMAT\_DATE('%Y%m%d',DATE\_SUB(CURRENT\_DATE(), INTERVAL 1 DAY))  
GROUP BY date  
ORDER BY date ASC

Данные за указанный период

**Стандартный SQL**

**Данные за указанный период, \_TABLE\_SUFFIX**

#standardSQL  
SELECT  
date,  
SUM(totals.visits) AS visits,  
SUM(totals.pageviews) AS pageviews,  
SUM(totals.transactions) AS transactions,  
SUM(totals.transactionRevenue)/1000000 AS revenue  
FROM `bigquery-public-data.google\_analytics\_sample.ga\_sessions\_\*`  
WHERE  
\_TABLE\_SUFFIX BETWEEN '20160801' AND '20170731'  
GROUP BY date  
ORDER BY date ASC

Данные за последние 3 года и текущий день

**Стандартный SQL**

**Данные за последние 3 года и текущий день, UNION ALL, \_TABLE\_SUFFIX**

Примечание. Этот пример запроса не подходит для общедоступного набора данных Google Analytics, поскольку там нет таблицы за текущий день.

#standardSQL  
WITH ga\_tables AS ( SELECT  
date,  
SUM(totals.visits) AS visits,  
SUM(totals.pageviews) AS pageviews,  
SUM(totals.transactions) AS transactions,  
SUM(totals.transactionRevenue)/1000000 AS revenue  
FROM `bigquery-public-data.google\_analytics\_sample.ga\_sessions\_\*`  
WHERE  
\_TABLE\_SUFFIX BETWEEN FORMAT\_DATE('%Y%m%d',DATE\_SUB(CURRENT\_DATE(), INTERVAL 3 YEAR))  
AND  
FORMAT\_DATE('%Y%m%d',DATE\_SUB(CURRENT\_DATE(), INTERVAL 1 DAY))  
GROUP BY date  
UNION ALL  
  
SELECT  
date,  
SUM(totals.visits) AS visits,  
SUM(totals.pageviews) AS pageviews,  
SUM(totals.transactions) AS transactions,  
SUM(totals.transactionRevenue)/1000000 AS revenue  
FROM `bigquery-public-data.google\_analytics\_sample.ga\_sessions\_\*`  
WHERE  
\_TABLE\_SUFFIX BETWEEN FORMAT\_DATE('%Y%m%d',DATE\_SUB(CURRENT\_DATE(), INTERVAL 1 DAY))  
AND  
FORMAT\_DATE('%Y%m%d',DATE\_SUB(CURRENT\_DATE(), INTERVAL 0 DAY))  
GROUP BY date  
)  
SELECT  
date,  
visits,  
pageviews,  
transactions,  
revenue,  
FROM ga\_tables  
ORDER BY date ASC

Простые запросы: примеры

В этом разделе описывается составление простых запросов с использованием показателей и параметров из образца данных Google Analytics.

Суммарный [*показатель*] на [*параметр*]

Ниже приведены примеры скриптов, позволяющих узнать, *какое суммарное количество транзакций на браузер устройства зарегистрировано в июле 2017 г.*

Стандартный SQL

**Суммарное количество транзакций на браузер устройства в июле 2017 г.**

#standardSQL  
SELECT  
device.browser,  
SUM ( totals.transactions ) AS total\_transactions  
FROM `bigquery-public-data.google\_analytics\_sample.ga\_sessions\_\*`  
WHERE  
\_TABLE\_SUFFIX BETWEEN '20170701' AND '20170731'  
GROUP BY  
device.browser  
ORDER BY  
total\_transactions DESC

Средний показатель отказов на [параметр]

Показатель отказов определяется как *процент посещений с одним просмотром страницы*. Ниже приведены примеры скриптов, позволяющих узнать *фактический показатель отказов на источник трафика*.

Стандартный SQL

**Показатель отказов на источник трафика в июле 2017 г.**

#standardSQL  
SELECT  
source,  
total\_visits,  
total\_no\_of\_bounces,  
( ( total\_no\_of\_bounces / total\_visits ) \* 100 ) AS bounce\_rate  
FROM (  
SELECT  
trafficSource.source AS source,  
COUNT ( trafficSource.source ) AS total\_visits,  
SUM ( totals.bounces ) AS total\_no\_of\_bounces  
FROM `bigquery-public-data.google\_analytics\_sample.ga\_sessions\_\*`  
WHERE  
\_TABLE\_SUFFIX BETWEEN '20170701' AND '20170731'  
GROUP BY  
source )  
ORDER BY  
total\_visits DESC

Среднее количество просмотров страниц товаров по типу пользователей (покупатели и остальные посетители)

Ниже приведены примеры скриптов, позволяющие узнать *среднее количество просмотров страниц товаров пользователями, совершившими покупку, в июле 2017 г.*

Стандартный SQL

**Среднее количество просмотров страниц товаров пользователями, совершившими покупку, в июле 2017 г.**

#standardSQL  
SELECT  
( SUM(total\_pagesviews\_per\_user) / COUNT(users) ) AS avg\_pageviews\_per\_user  
FROM  
SELECT  
fullVisitorId AS users,  
SUM(totals.pageviews) AS total\_pagesviews\_per\_user  
FROM`bigquery-public-data.google\_analytics\_sample.ga\_sessions\_\*`  
WHERE  
\_TABLE\_SUFFIX BETWEEN '20170701' AND '20170731'  
AND  
totals.transactions >=1  
GROUP BY  
users )

Ниже приведены примеры скриптов, позволяющих узнать *среднее количество просмотров страниц товаров пользователями, которые не совершили покупку, в июле 2017 г.*

Стандартный SQL

**Среднее количество просмотров страниц товаров пользователями, не совершившими покупку, в июле 2017 г.**

#standardSQL  
SELECT  
( SUM(total\_pagesviews\_per\_user) / COUNT(users) ) AS avg\_pageviews\_per\_user  
FROM (  
SELECT  
fullVisitorId AS users,  
SUM(totals.pageviews) AS total\_pagesviews\_per\_user  
FROM`bigquery-public-data.google\_analytics\_sample.ga\_sessions\_\*`  
WHERE  
\_TABLE\_SUFFIX BETWEEN '20170701' AND '20170731'  
AND  
totals.transactions IS NULL  
GROUP BY  
users )

Среднее число транзакций на покупателя

Ниже приведены примеры скриптов, позволяющих узнать *среднее суммарное количество транзакций на пользователя, совершившего покупку, в июле 2017 г.*

Стандартный SQL

**Среднее количество транзакций на пользователя, совершившего покупку, в июле 2017 г.**

#standardSQL  
SELECT  
(SUM (total\_transactions\_per\_user) / COUNT(fullVisitorId) ) AS avg\_total\_transactions\_per\_user  
FROM (  
SELECT  
fullVisitorId,  
SUM (totals.transactions) AS total\_transactions\_per\_user  
FROM  
`bigquery-public-data.google\_analytics\_sample.ga\_sessions\_\*`  
WHERE  
\_TABLE\_SUFFIX BETWEEN '20170701' AND '20170731'  
AND totals.transactions IS NOT NULL  
GROUP BY  
fullVisitorId )

Средняя сумма расходов за сеанс

Ниже приведены примеры скриптов, позволяющих узнать *среднюю сумму расходов за сеанс в июле 2017 г.*

Стандартный SQL

**Средняя сумма расходов за сеанс в июле 2017 г.**

#standardSQL  
SELECT  
( SUM(total\_transactionrevenue\_per\_user) / SUM(total\_visits\_per\_user) ) AS  
avg\_revenue\_by\_user\_per\_visit  
FROM (  
SELECT  
fullVisitorId,  
SUM( totals.visits ) AS total\_visits\_per\_user,  
SUM( totals.transactionRevenue ) AS total\_transactionrevenue\_per\_user  
FROM  
`bigquery-public-data.google\_analytics\_sample.ga\_sessions\_\*`  
WHERE  
\_TABLE\_SUFFIX BETWEEN '20170701' AND '20170731'  
AND  
totals.visits > 0  
AND totals.transactions >= 1  
AND totals.transactionRevenue IS NOT NULL  
GROUP BY  
fullVisitorId )

Последовательность обращений

Ниже приведены примеры скриптов, позволяющих узнать *последовательность просмотра страниц*.

Стандартный SQL

**Последовательность просмотра страниц пользователями в июле 2017 г.**

#standardSQL  
SELECT  
fullVisitorId,  
visitId,  
visitNumber,  
hits.hitNumber AS hitNumber,  
hits.page.pagePath AS pagePath  
FROM  
`bigquery-public-data.google\_analytics\_sample.ga\_sessions\_\*`,  
UNNEST(hits) as hits  
WHERE  
\_TABLE\_SUFFIX BETWEEN '20170701' AND '20170731'  
AND  
hits.type="PAGE"  
ORDER BY  
fullVisitorId,  
visitId,  
visitNumber,  
hitNumber

В этом запросе обращения ограничиваются типом PAGES, чтобы исключить события и транзакции. Каждая строка выходных данных представляет просмотр страницы. Строки выводятся в порядке перечисления полей в операторе SELECT.

Несколько специальных параметров на уровне обращений или сеансов

**специальный параметр на уровне обращений**

SELECT fullVisitorId, visitId, hits.hitNumber, hits.time,  
MAX(IF(hits.customDimensions.index=1,  
hits.customDimensions.value,  
NULL)) WITHIN hits AS customDimension1,  
FROM [tableID.ga\_sessions\_20150305]  
LIMIT 100

**специальный параметр на уровне сеансов**

SELECT fullVisitorId, visitId,  
MAX(IF(customDimensions.index=2,  
customDimensions.value,  
NULL)) WITHIN RECORD AS customDimension2,  
FROM [tableID.ga\_sessions\_20150305]  
LIMIT 100

В каждом запросе:

Оператор SELECT запрашивает релевантные поля параметров и показателей.

Функция MAX:

* Возвращает специальный параметр в новом столбце. Чтобы вернуть несколько специальных параметров в новых столбцах, повторите функцию.
* Операторы WITHIN hits и WITHIN RECORDS оценивают условие в повторяющихся полях в BigQuery.
* Условие, заданное функцией MAX, оценивается для каждого специального параметра. Если не выполняется условие index=1 для обращений или index=2 для сеансов, возвращается значение NULL.
* Возвращается максимальное значение, которое соответствует значению специального параметра 1 для обращений или значению специального параметра 2 для сеансов, поскольку все остальные значения равны NULL.

Сложные запросы: примеры

Теперь перейдем к составлению запросов с использованием сложных функций, доступных в BigQuery.

Товары, приобретенные пользователями, также купившими товар А (обычная электронная торговля)

Ниже представлена структура скрипта, позволяющего узнать, *какие товары приобретали пользователи, которые также купили товар А*.

**товары, приобретенные пользователями, также купившими товар А (обычная электронная торговля)**

SELECT hits.item.productName AS other\_purchased\_products, COUNT(hits.item.productName) AS quantity  
FROM [‘Dataset Name’ ]  
WHERE fullVisitorId IN (  
  SELECT fullVisitorId  
  FROM [‘Dataset Name’ ]  
  WHERE hits.item.productName CONTAINS 'Product Item Name A'  
   AND totals.transactions>=1  
  GROUP BY fullVisitorId )  
 AND hits.item.productName IS NOT NULL  
 AND hits.item.productName != 'Product Item Name A'  
GROUP BY other\_purchased\_products  
ORDER BY quantity DESC;

1. В первой строке выбираются все остальные товары, приобретенные пользователем. Для расчета количества каждого купленного товара используется функция COUNT(). Результат отражается в поле quantity, а соответствующий товар – в поле other\_purchased\_products.
2. В подзапросе, выделенном серым цветом, выбираются только уникальные пользователи (fullVisitorId), совершившие транзакции (totals.transactions>=1) и купившие в ходе транзакции товар А (WHERE hits.item.productName CONTAINS 'Product Item Name A').

Правила, заданные операторами WHERE и AND в запросе верхнего уровня, который выделен зеленым цветом, игнорируют значения hits.item.productName, равные NULL и содержащие товар А.

Ниже приведен пример запроса, который позволяет узнать, *какие ещё товары приобрел покупатель, купивший Brighton Metallic Pens - Set of 4*.

**товары, приобретенные покупателем, который 24 июня 2013 г. купил Brighton Metallic Pens - Set of 4**

SELECT hits.item.productName AS other\_purchased\_products, COUNT(hits.item.productName) AS quantity  
FROM [GoogleStore.ga\_sessions\_20130624]  
WHERE fullVisitorId IN (  
  SELECT fullVisitorId  
  FROM [GoogleStore.ga\_sessions\_20130624]  
  WHERE hits.item.productName CONTAINS 'Brighton Metallic Pens - Set of 4'  
   AND totals.transactions>=1  
  GROUP BY fullVisitorId )  
 AND hits.item.productName IS NOT NULL  
 AND hits.item.productName !='Brighton Metallic Pens - Set of 4'  
GROUP BY other\_purchased\_products  
ORDER BY quantity DESC;

В Dremel/BigQuery оператор WHERE expr IN вызывает команду *JOIN*. При этом действуют ограничения размера. В частности, размер правой части (в данном случае это количество посетителей) не должен превышать 8 МБ. В Dremel это называется *Broadcast JOIN*. Если размер превышает 8 МБ, необходимо использовать *Shuffled JOIN* с применением синтаксиса *JOIN EACH*. Для этого нельзя использовать оператор *IN*, но тот же запрос можно создать с помощью команды *JOIN*.

Товары, приобретенные пользователями, также купившими товар А (расширенная электронная торговля)

Этот запрос похож на предыдущий, но предназначен для расширенной электронной торговли. В нем для получения данных за несколько дней также используются диапазоны TABLE\_DATE\_RANGE.

**товары, приобретенные пользователями, также купившими товар А (расширенная электронная торговля)**

SELECT hits.product.productSKU AS other\_purchased\_products, COUNT(hits.product.productSKU) AS quantity  
FROM SELECT fullVisitorId, hits.product.productSKU, hits.eCommerceAction.action\_type FROM TABLE\_DATE\_RANGE([73156703.ga\_sessions\_], TIMESTAMP('2015-07-01'), TIMESTAMP('2015-07-14')))  
WHERE fullVisitorId IN (  
  SELECT fullVisitorId  
FROM TABLE\_DATE\_RANGE([73156703.ga\_sessions\_], TIMESTAMP('2015-07-01'), TIMESTAMP('2015-07-14'))  
WHERE hits.product.productSKU CONTAINS '10 15103'  
AND hits.eCommerceAction.action\_type = '6'  
GROUP BY fullVisitorId )  
AND hits.product.productSKU IS NOT NULL  
AND hits.product.productSKU !='10 15103'  
AND hits.eCommerceAction.action\_type = '6'  
GROUP BY other\_purchased\_products  
ORDER BY quantity DESC;

Среднее количество взаимодействий пользователя перед покупкой

Рассмотрим пример запроса, выполняемого командой JOIN() [...] ON, которая зависит только от данных Google Аналитики.

Ниже представлена структура скрипта, позволяющего узнать *среднее количество взаимодействий пользователя перед покупкой*.

**количество взаимодействий пользователя перед покупкой**

SELECT one.hits.item.productSku AS ProductSku, ( sum\_of\_hit\_number / total\_hits ) AS avg\_hit\_number  
FROM (  
 SELECT hits.item.productSku, SUM(hits.hitNumber) AS sum\_of\_hit\_number  
 FROM [‘GA Dataset Name’ ]  
 WHERE hits.item.productSku IS NOT NULL  
  AND totals.transactions>=1  
 GROUP BY hits.item.productSku ) AS ‘Alias\_Name\_1’  
JOIN (  
 SELECT hits.item.productSku, COUNT( fullVisitorId ) AS total\_hits  
 FROM [‘GA Dataset Name’ ]  
 WHERE hits.item.productSku IS NOT NULL  
  AND totals.transactions>=1  
 GROUP BY hits.item.productSku ) AS ‘Alias\_Name\_2’  
ON Alias\_Name\_1.hits.item.productSku = Alias\_Name\_2.hits.item.productSku;

1. В первой строке выполняется основная математическая операция, определяющая среднее количество взаимодействий в расчете на продукт. Этот запрос создает связь между подзапросами ‘*Alias\_Name\_1*’ и ‘*Alias\_Name\_2*’.
2. Подзапрос ‘*Alias\_Name\_1*’ возвращает поле, в котором с помощью агрегатной функции SUM() суммируются все обращения, зафиксированные для товара.
3. Подзапрос ‘*Alias\_Name\_2*’ определяет количество обращений пользователей в расчете на товар с помощью функции COUNT().
4. В последней строке указано общее поле (hits.item.productSku), используемое в двух объединенных наборах данных.

Ниже приведен пример запроса, который позволяет узнать *среднее количество взаимодействий пользователя перед покупкой 10 сентября 2013 г.*

| **количество взаимодействий пользователя перед покупкой 10 сентября 2013 г.** |
| --- |
| SELECT one.hits.item.productSku AS ProductSku, ( sum\_of\_hit\_number / total\_hits ) AS avg\_hit\_number FROM (  SELECT hits.item.productSku, SUM(hits.hitNumber) AS sum\_of\_hit\_number  FROM [GoogleStore.ga\_sessions\_20130728]  WHERE hits.item.productSku IS NOT NULL   AND totals.transactions>=1  GROUP BY hits.item.productSku  ) AS one JOIN (  SELECT hits.item.productSku, COUNT( fullVisitorId ) AS total\_hits  FROM [GoogleStore.ga\_sessions\_20130728]  WHERE hits.item.productSku IS NOT NULL   AND totals.transactions>=1  GROUP BY hits.item.productSku ) AS two ON one.hits.item.productSku = two.hits.item.productSku; |

Процент проданных запасов товара

Рассмотрим пример запроса, в котором используются данные не только из Google Analytics, но и из других систем. Объединив два набора данных, вы сможете более детально анализировать поведение пользователей. Импортируя данные из других систем в BigQuery, помните, что это повлияет на размер месячного платежа за хранение данных.

Ниже представлена структура скрипта, позволяющего узнать *процент проданных запасов товара*.

**процент проданных запасов товара**

SELECT AnalyticsImport.product\_data\_20130728.productId, ((( one.quantity\_sold ) / ( AnalyticsImport.product\_data\_20130728.productstock + one.quantity\_sold )) \* 100 ) AS percentage\_of\_stock\_sold  
FROM [ ‘Imported\_DataSet’ ]  
JOIN (  
  SELECT hits.item.productSku, SUM( hits.item.itemQuantity ) AS quantity\_sold  
  FROM [‘GA Dataset’ ]  
  WHERE hits.item.productSku IS NOT NULL  
   AND totals.transactions>=1  
  GROUP BY hits.item.productSku ) AS ‘Alias\_Name’  
ON Imported\_DataSet.’productId\_field’ = Alias\_Name.hits.item.productSku;

1. В первой строке возвращается два поля: одно содержит все идентификаторы товаров, в другом выполняется математическая операция, определяющая процент проданных запасов товара по его идентификатору.
2. Поскольку этот запрос зависит от двух наборов данных, необходимо использовать функцию JOIN() ... ON. Она объединяет строки из двух наборов данных по общему полю. В данном случае это наборы данных [ ‘Imported\_DataSet’ ] и ‘Alias\_Name’.
3. [ ‘Imported\_DataSet’ ] содержит внешние данные. Этот набор данных включает показатель количества оставшегося товара (Imported DataSet.’stock\_left\_field’) и параметр идентификатора товара (Imported\_DataSet.’productId\_field’).
4. Название ‘Alias\_Name’ присваивается данным, которые возвращает подзапрос, выделенный серым цветом. В этом подзапросе используются данные Google Analytics для определения общего количества проданных единиц товара.
5. Последняя строка содержит оператор ON, отражающий общее для двух наборов данных поле, по которому они объединяются.

Многие переменные этого запроса содержат название набора данных в виде префикса (например, Imported\_DataSet.’productId\_field’, Alias\_Name.quantity\_sold). Это указывает, какое поле в каком именно наборе данных выбрано.

Ниже приведен пример запроса, который позволяет узнать *процент проданных запасов товара 28 июля 2013 г.*

**процент проданных запасов товара 28 июля 2013 г.**

SELECT AnalyticsImport.product\_data\_20130728.productId, ( ( ( one.quantity\_sold ) / ( AnalyticsImport.product\_data\_20130728.productstock + one.quantity\_sold ) ) \* 100 ) AS percentage\_of\_stock\_sold  
FROM AnalyticsImport.product\_data\_20130728  
JOIN (  
  SELECT hits.item.productSku, SUM( hits.item.itemQuantity ) AS quantity\_sold  
  FROM GoogleStore.ga\_sessions\_20130728  
  WHERE hits.item.productSku IS NOT NULL  
   AND totals.transactions>=1  
  GROUP BY hits.item.productSku ) AS one  
ON AnalyticsImport.product\_data\_20130728.productId = one.hits.item.productSku  
ORDER BY percentage\_of\_stock\_sold DESC;

Прибыльность каждого товара

Ниже представлена структура скрипта, позволяющего узнать *прибыльность каждого товара*

**прибыльность товаров**

SELECT Alias\_Name.hits.item.productSku, ( Imported\_DataSet.’product profit field’ \* Alias\_Name.quantity ) AS profit  
FROM (  
  SELECT Alias\_Name.hits.item.productSku, Imported\_DataSet.’product profit field’  
  FROM [ ‘Imported Data Set’ ]  
  JOIN (  
    SELECT hits.item.productSku, SUM( hits.item.itemQuantity ) AS quantity  
    FROM [ ‘GA Dataset Name’ ]  
    WHERE hits.item.productSku IS NOT NULL  
    GROUP BY hits.item.productSku ) AS ‘Alias\_Name’  
  ON Imported\_DataSet.productId = Alias\_Name.hits.item.productSku );

1. Первая строка содержит математическую операцию для расчета общей прибыли по каждому товару.
2. Подзапрос, выделенный серым цветом, использует внешние данные по размеру прибыли от продажи товара.
3. Подзапрос, выделенный красным цветом, использует данные Google Analytics, которые объединяются с внешними данными. Он рассчитывает количество проданного товара.
4. Оператор ON в последней строке указывает общее для двух наборов данных поле. В данном случае это идентификатор товара.

Ниже приведен пример запроса, который позволяет узнать *прибыльность каждого товара на 28 июля 2013 г.*

**прибыльность товаров на 28 июля 2013 г.**

SELECT two.hits.item.productSku, ((AnalyticsImport.product\_data\_20130728.productprice-AnalyticsImport.product\_data\_20130728.productcost)\*two.quantity) AS profit  
FROM (  
  SELECT two.hits.item.productSku, AnalyticsImport.product\_data\_20130728.productcost, two.quantity, AnalyticsImport.product\_data\_20130728.productprice  
  FROM AnalyticsImport.product\_data\_20130728  
  JOIN (  
    SELECT hits.item.productSku, SUM( hits.item.itemQuantity ) AS quantity  
    FROM GoogleStore.ga\_sessions\_20130728  
    WHERE hits.item.productSku IS NOT NULL  
    GROUP BY hits.item.productSku ) AS two  
  ON AnalyticsImport.product\_data\_20130728.productId = two.hits.item.productSku );

Прибыль рассчитывается путем определения разницы между ценой товара и его себестоимостью. Эта информация хранится во внешнем наборе данных.

Фактическая прибыльность каждого товара (*с учетом возвратов*)

Ниже представлена структура скрипта, позволяющего узнать *фактическую прибыльность каждого товара*.

**фактическая прибыльность товаров**

SELECT Alias\_Name.hits.item.productSku, (gross\_profit-total\_refund\_revenue) AS real\_profit  
FROM (  
  SELECT Alias\_Name.hits.item.productSku, ( ( Imported\_DataSet.productprice - Imported\_DataSet.productcost ) \* Alias\_Name.quantity ) AS gross\_profit, ( ( Imported\_DataSet.refunddeliveryprice + Imported\_DataSet.productprice ) \* Imported\_DataSet.refundquantity ) AS total\_refund\_revenue  
  FROM (  
    SELECT Alias\_Name.hits.item.productSku, Imported\_DataSet.productcost, Alias\_Name.quantity, Imported\_DataSet.productprice, Imported\_DataSet.refunddeliveryprice, Imported\_DataSet.refundquantity  
    FROM [ ‘Imported DataSet Name’ ] AS 'Imported\_DataSet'  
    JOIN (  
      SELECT hits.item.productSku, SUM( hits.item.itemQuantity ) AS quantity  
      FROM [‘GA Dataset Name’ ]  
      WHERE hits.item.productSku IS NOT NULL  
       AND totals.transactions >=1  
      GROUP BY hits.item.productSku ) AS 'Alias\_Name'  
    ON Imported\_DataSet.productId = Alias\_Name.hits.item.productSku ) );

1. Этот запрос аналогичен [*предыдущему*](https://support.google.com/analytics/answer/4419694?hl=ru#query10_ProfitabilityOfEachProduct). Отличия относятся к внешнему набору данных в подзапросе, выделенном серым цветом, и к математической операции, рассчитывающей фактическую прибыль.
2. Во внешнем наборе данных также рассчитывается общая сумма возвратов (в операторе SELECT в подзапросе, выделенном красным цветом).
3. Затем выполняется математическая операция в первой строке, определяющая фактическую прибыль путем вычитания дохода, потраченного на возвраты, из валовой прибыли.

Подробнее о запросе читайте в разделе [*Прибыльность каждого товара*](https://support.google.com/analytics/answer/4419694?hl=ru#query10_ProfitabilityOfEachProduct).

Ниже приведен пример запроса, который позволяет узнать *фактическую прибыльность каждого товара на 28 июля 2013 г.*

**фактическая прибыльность товаров на 28 июля 2013 г.**

SELECT two.hits.item.productSku, (gross\_profit-total\_refund\_revenue) AS real\_profit  
FROM (  
  SELECT two.hits.item.productSku, ( ( AnalyticsImport.product\_data\_20130728.productprice - AnalyticsImport.product\_data\_20130728.productcost ) \* two.quantity ) AS gross\_profit, ( ( AnalyticsImport.product\_data\_20130728.refunddeliveryprice + AnalyticsImport.product\_data\_20130728.productprice ) \* AnalyticsImport.product\_data\_20130728.refundquantity ) AS total\_refund\_revenue  
  FROM (  
    SELECT two.hits.item.productSku, Analytics.product\_data\_20130728.productcost, two.quantity, AnalyticsImport.product\_data\_20130728.productprice, AnalyticsImport.product\_data\_20130728.refunddeliveryprice, AnalyticsImport.product\_data\_20130728.refundquantity  
    FROM AnalyticsImport.product\_data\_20130728  
    JOIN (  
      SELECT hits.item.productSku, SUM( hits.item.itemQuantity ) AS quantity  
      FROM GoogleStore.ga\_sessions\_20130728  
      WHERE hits.item.productSku IS NOT NULL  
       AND totals.transactions >=1  
      GROUP BY hits.item.productSku ) AS two  
    ON AnalyticsImport.product\_data\_20130728.productId = two.hits.item.productSku ) );

При расчете фактической прибыли принимается во внимание прибыльность товара с учетом возвратов. Общий доход, потраченный на возвраты, для товара рассчитывается по следующей формуле:

*Общий возвращенный доход для товара = (цена товара + стоимость доставки товара) \* количество возвращенных единиц товара.*