BigQuery Code editor

For each activity in the following sections, this lab provides queries with common errors for you to troubleshoot. The lab directs you what to look at and suggests how to correct the syntax and return meaningful results.

To follow along with the troubleshooting and suggestions, copy and paste the query into the BigQuery Query editor. If there are errors, you see a red exclamation point at the line containing the error and in the query validator (bottom corner).

If you run the query with the errors, the query fails and the error is specified in the Job information.

When the query is error free, you see a green checkmark in the query validator. When you see the green checkmark, click **Run** to run the query to view what you get for results.

Standard SQL Query Syntax.

Find the total number of customers who went through checkout

Your goal in this section is to construct a query that gives you the number of unique visitors who successfully went through the checkout process for your website. The data is in the rev_transactions table which your data analyst team has provided. They have also given you example queries to help you get started in your analysis but you're not sure they're written correctly.

Troubleshoot queries that contain query validator, alias, and comma errors

Look at the below query and answer the following question:

```
#standardSQL
SELECT  FROM `data-to-inghts.ecommerce.rev_transactions` LIMIT 1000
content_copy
```

There is a typo in the dataset name

We have not specified any columns in the SELECT

What about this updated query?

```
#standardSQL
SELECT * FROM [data-to-insights:ecommerce.rev_transactions] LIMIT 1000
content_copy
```

We are using legacy SQL

What about this query that uses Standard SQL?

```
#standardSQL
SELECT FROM `data-to-insights.ecommerce.rev_transactions`
content_copy
```

ll no columns defined in SELECT

hat about now? This query has a column.

```
#standardSQL
SELECT
fullVisitorId
FROM `data-to-insights.ecommerce.rev_transactions`
content_copy
```

The page title is missing from the columns in SELECT

Without aggregations, limits, or sorting, this query is not insightful

What about now? The following query has a page title.

```
#standardSQL
SELECT fullVisitorId hits_page_pageTitle
FROM `data-to-insights.ecommerce.rev_transactions` LIMIT 1000
content_copy
```

~~0, the query will return an error~~

~~2, columns named fullVisitorId and hits_page_pageTitle~~

1, a column named hits_page_pageTitle

~~3 columns will be returned since we are missing a comma~~

What about now? The missing comma has been corrected.

```
#standardSQL
SELECT
  fullVisitorId
  , hits_page_pageTitle
FROM `data-to-insights.ecommerce.rev_transactions` LIMIT 1000
content_copy
```

Answer: This returns results, but are you sure visitors aren't counted twice? Also, returning only one row answers the question of how many unique visitors reached checkout. In the next section you find a way to aggregate your results.

Troubleshoot queries that contain logic errors, GROUP BY statements, and wildcard filters

Aggregate the following query to answer the question: How many unique visitors reached checkout?

```
#standardSQL
SELECT
   fullVisitorId
   , hits_page_pageTitle
FROM `data-to-insights.ecommerce.rev_transactions` LIMIT 1000
content_copy
```

What about this? An aggregation function, COUNT(), was added.

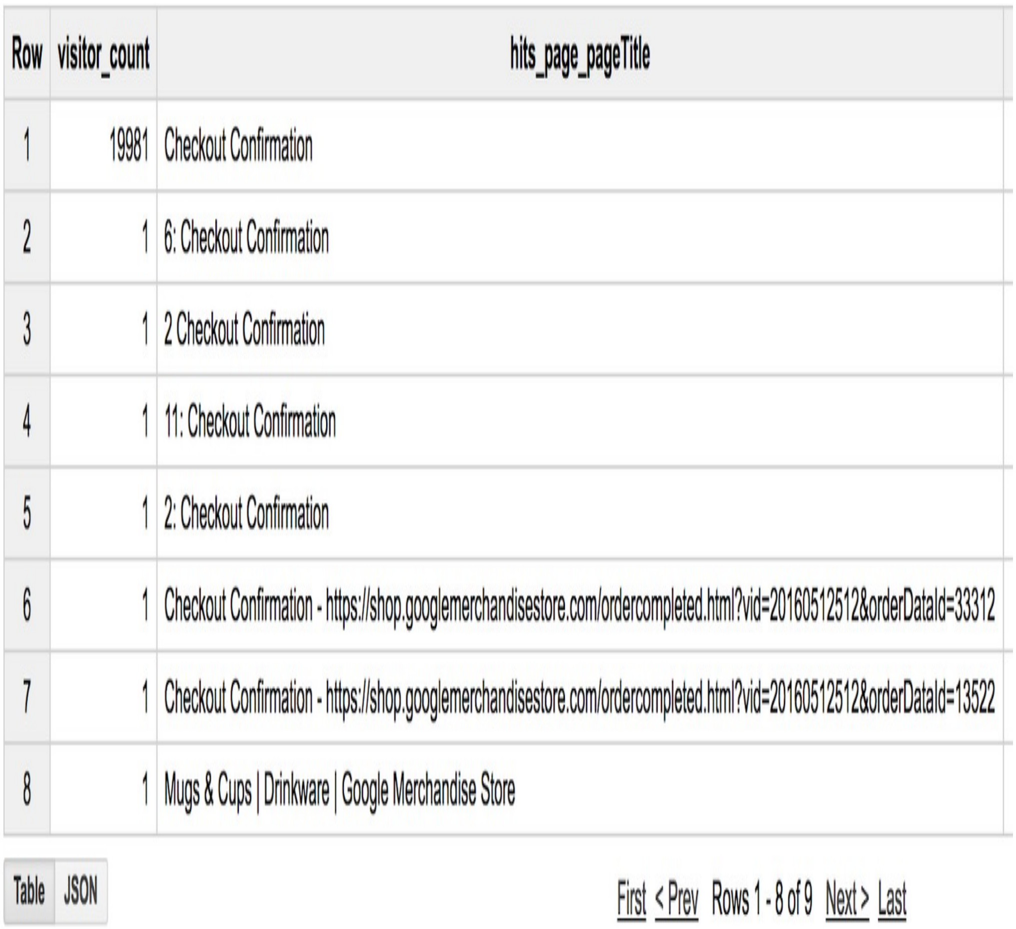
```
#standardSQL
SELECT
COUNT(fullVisitorId) AS visitor_count
, hits_page_pageTitle
FROM `data-to-insights.ecommerce.rev_transactions`
content_copy
```

The COUNT() function does not de-deduplicate the same fullVisitorId

It is missing a GROUP BY clause

In this next query, GROUP BY and DISTINCT statements were added.

```
#standardSQL
SELECT
COUNT(DISTINCT fullVisitorId) AS visitor_count
, hits_page_pageTitle
FROM `data-to-insights.ecommerce.rev_transactions`
GROUP BY hits_page_pageTitle
content_copy
```



Row	visitor_count	hits_page_pageTitle
1	19981	Checkout Confirmation
2	1	6: Checkout Confirmation
3	1	2 Checkout Confirmation
4	1	11: Checkout Confirmation
5	1	2: Checkout Confirmation
6	1	Checkout Confirmation - https://shop.googlemerchandisestore.com/ordercompleted.html?vid=20160512512&orderDataId=33312
7	1	Checkout Confirmation - https://shop.googlemerchandisestore.com/ordercompleted.html?vid=20160512512&orderDataId=13522
8	1	Mugs & Cups | Drinkware | Google Merchandise Store

Results

Great! The results are good, but they look strange. Filter to just "Checkout Confirmation" in the results.

```
#standardSQL
SELECT
COUNT(DISTINCT fullVisitorId) AS visitor_count
, hits_page_pageTitle
FROM `data-to-insights.ecommerce.rev_transactions`
WHERE hits_page_pageTitle = "Checkout Confirmation"
GROUP BY hits_page_pageTitle
content_copy
```

Click Check my progress to verify the objective.

List the cities with the most transactions with your ecommerce site

Troubleshoot ordering, calculated fields, and filtering after aggregating errors

Complete the partially written query:

```
SELECT
geoNetwork_city,
totals_transactions,
COUNT( DISTINCT fullVisitorId) AS distinct_visitors
FROM
`data-to-insights.ecommerce.rev_transactions`
GROUP BY
content_copy
```

Possible solution

```
#standardSQL
SELECT
geoNetwork_city,
SUM(totals_transactions) AS totals_transactions,
COUNT( DISTINCT fullVisitorId) AS distinct_visitors
FROM
`data-to-insights.ecommerce.rev_transactions`
GROUP BY geoNetwork_city
content_copy
```

Update your previous query to order the top cities first.

ountain View

ossible solution

```
#standardSQL
SELECT
```

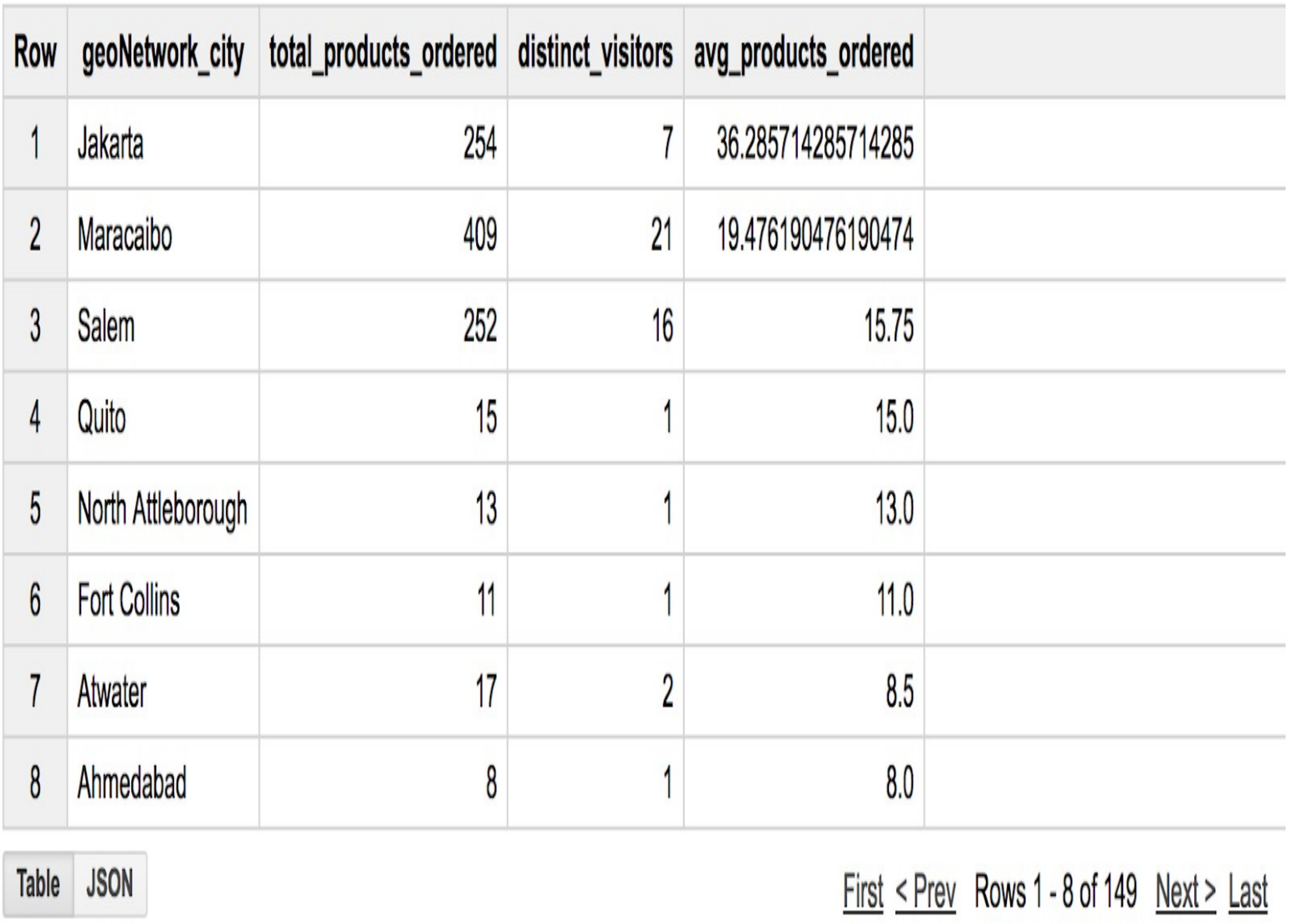
```
geoNetwork_city,
SUM(totals_transactions) AS totals_transactions,
COUNT( DISTINCT fullVisitorId) AS distinct_visitors
FROM
`data-to-insights.ecommerce.rev_transactions`
GROUP BY geoNetwork_city
ORDER BY distinct_visitors DESC
content_copy
```

Update your query and create a new calculated field to return the average number of products per order by city.

Possible solution

```
#standardSQL
SELECT
geoNetwork_city,
SUM(totals_transactions) AS total_products_ordered,
COUNT( DISTINCT fullVisitorId) AS distinct_visitors,
SUM(totals_transactions) / COUNT( DISTINCT fullVisitorId) AS avg_products_ordered
FROM
`data-to-insights.ecommerce.rev_transactions`
GROUP BY geoNetwork_city
ORDER BY avg_products_ordered DESC
content_copy
```

Results



Row	geoNetwork_city	total_products_ordered	distinct_visitors	avg_products_ordered
1	Jakarta	254	7	36.285714285714285
2	Maracaibo	409	21	19.476190476190474
3	Salem	252	16	15.75
4	Quito	15	1	15.0
5	North Attleborough	13	1	13.0
6	Fort Collins	11	1	11.0
7	Atwater	17	2	8.5
8	Ahmedabad	8	1	8.0

Filter your aggregated results to only return cities with more than 20 avg_products_ordered.

What's wrong with the following query?

```
#standardSQL
SELECT
geoNetwork_city,
SUM(totals_transactions) AS total_products_ordered,
COUNT( DISTINCT fullVisitorId) AS distinct_visitors,
SUM(totals_transactions) / COUNT( DISTINCT fullVisitorId) AS avg_products_ordered
FROM
`data-to-insights.ecommerce.rev_transactions`
WHERE avg_products_ordered > 20
GROUP BY geoNetwork_city
ORDER BY avg_products_ordered DESC
```

content_copy

You cannot filter aggregated fields in the `WHERE` clause (use `HAVING` instead)

You cannot filter on aliased fields within the `WHERE` clause

Possible solution

```
#standardSQL
SELECT
geoNetwork_city,
SUM(totals_transactions) AS total_products_ordered,
COUNT( DISTINCT fullVisitorId) AS distinct_visitors,
SUM(totals_transactions) / COUNT( DISTINCT fullVisitorId) AS avg_products_ordered
FROM
`data-to-insights.ecommerce.rev_transactions`
GROUP BY geoNetwork_city
HAVING avg_products_ordered > 20
ORDER BY avg_products_ordered DESC
```

content_copy

Click Check my progress to verify the objective.

Find the total number of products in each product category

Find the top selling products by filtering with NULL values

What's wrong with the following query? How can you fix it?

```
#standardSQL
SELECT hits_product_v2ProductName, hits_product_v2ProductCategory
FROM `data-to-insights.ecommerce.rev_transactions`
GROUP BY 1,2
content_copy
```

No aggregate functions are used

Large GROUP BYs really hurt performance (consider filtering first and/or using aggregation functions)

What is wrong with the following query?

```
#standardSQL
SELECT
COUNT(hits_product_v2ProductName) as number_of_products,
hits_product_v2ProductCategory
FROM `data-to-insights.ecommerce.rev_transactions`
WHERE hits_product_v2ProductName IS NOT NULL
GROUP BY hits_product_v2ProductCategory
ORDER BY number_of_products DESC
content_copy
```

e COUNT() function is not the distinct number of products in each category

odate the previous query to only count distinct products in each product category.

Possible solution

```
#standardSQL
SELECT
COUNT(DISTINCT hits_product_v2ProductName) as number_of_products,
hits_product_v2ProductCategory
FROM `data-to-insights.ecommerce.rev_transactions`
WHERE hits_product_v2ProductName IS NOT NULL
GROUP BY hits_product_v2ProductCategory
ORDER BY number_of_products DESC
LIMIT 5
content_copy
```

ot set)

otes:

•(not set) could indicate the product has no category

•${productitem.product.origCatName} is front-end code to render the category which may indicate the Google Analytics tracking script is firing before the page is fully-rendered

Click Check my progress to verify the objective.

Congratulations!

You troubleshot and fixed broken queries in BigQuery standard SQL. Remember to use the Query Validator for incorrect query syntax but also to be critical of your query results even if your query executes successfully.

Finish your Quest

This self-paced lab is part of the Qwiklabs BigQuery Basics for Data Analysts Quest. A Quest is a series of related labs that form a learning path. Completing this Quest earns you the badge above, to recognize your achievement. You can make your badge (or badges) public and link to them in your online resume or social media account. Enroll in a Quest and get immediate completion credit if you've taken this lab. See other available See other available Qwiklabs Quests.

Looking for a hands-on challenge lab to demonstrate your BigQuery skills and validate your knowledge? On completing this quest, finish this additional challenge lab to receive an exclusive Google Cloud digital badge.

Take your next lab

Continue your Quest with the next lab, Explore and Create Reports with Data Studio.

Check out other labs to learn more about BigQuery:

•Exploring Your Ecommerce Dataset with SQL in BigQuery

•Weather Data in BigQuery

•Creating Date-Partitioned Tables in BigQuery

Next steps/learn more

•Explore BigQuery Public Datasets.
•Already have a Google Analytics account and want to query your own datasets in BigQuery? Follow this export guide.
•Check out 15 Awesome things you probably didn’t know about BigQuery.

Google Cloud Training & Certification

...helps you make the most of Google Cloud technologies. Our classes include technical skills and best practices to help you get up to speed quickly and continue your learning journey. We offer fundamental to advanced level training, with on-demand, live, and virtual options to suit your busy schedule. Certifications help you validate and prove your skill and expertise in Google Cloud technologies.

Manual Last Updated: April 8, 2021

Lab Last Tested: January 30, 2021