Google Cloud Platform

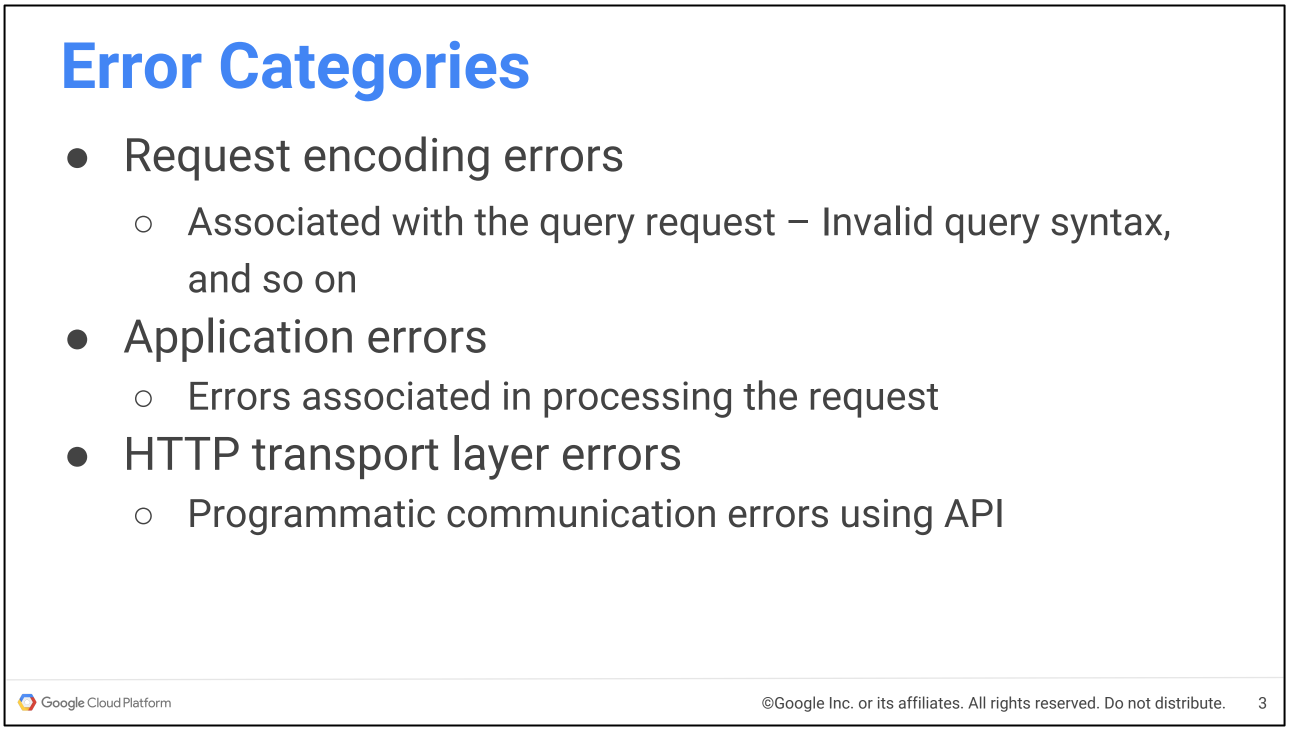
Troubleshooting Errors

BigQuery for Data Analysts
V1.2

Google Cloud Platform

Approximate timing: 30 minutes

Agenda



Error Categories

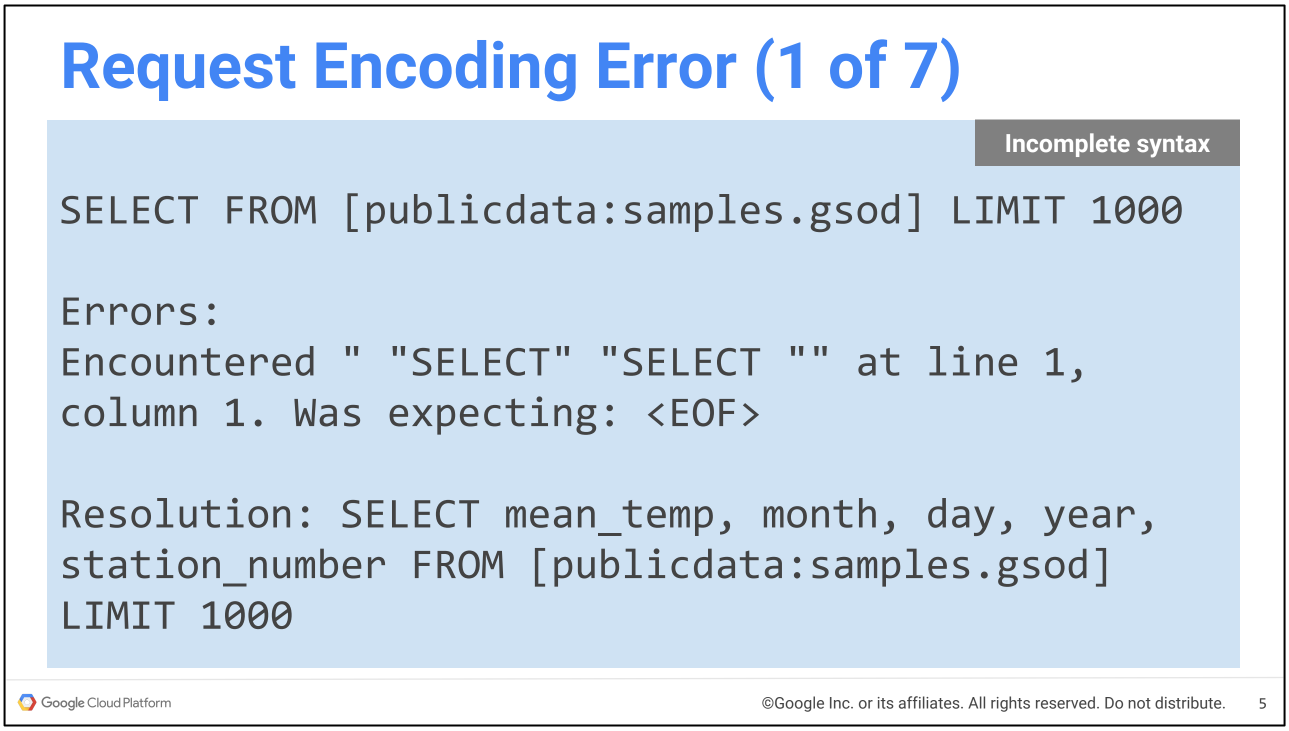
- Request encoding errors
  - Associated with the query request – Invalid query syntax, and so on
- Application errors
  - Errors associated in processing the request
- HTTP transport layer errors
  - Programmatic communication errors using API

Google Cloud Platform

Notes:
Request encoding error are the most common and the easiest to correct. These are syntactical error associated to the way that the query was written. Application errors are those returned from the BigQuery engine while processing the request. These errors are the most difficult to resolve. These usually are the result of some kind of quota threshold being exceeded. HTTP transport layer errors are associated to the network communication layer. These are programmatic errors that mostly seen when accessing BigQuery using the REST API from a external program.

Request Encoding Errors

- Incomplete syntax
- Missing or invalid objects
- Missing columns in GROUP BY (for aggregations)
- Incorrect or missing punctuation
- Misspellings
- Ambiguous field references



Request Encoding Error (1 of 7)

Incomplete syntax

```
SELECT FROM [publicdata:samples.gsod] LIMIT 1000

Errors:
Encountered " "SELECT" "SELECT "" at line 1,
column 1. Was expecting: <EOF>

Resolution: SELECT mean_temp, month, day, year,
station_number FROM [publicdata:samples.gsod]
LIMIT 1000
```

Google Cloud Platform

Notes:
Missing column name after SELECT.

Request Encoding Error (2 of 7)

Missing object

```
SELECT * FROM [grand-object-704:demodata.power]
LIMIT 1000

Errors:
Not found: Table grand-object-704:demodata.power

Resolution:
Dataset must be shared (demodata is not in the
grand-object-704 project)
```

Request Encoding Error (3 of 7)

Invalid GROUP BY

```
SELECT ngram, first, second FROM [publicdata:samples.trigrams] AS
trigram
JOIN [publicdata:samples.shakespeare] AS shakespeare
ON trigram.first = shakespeare.word WHERE shakespeare.word = 'To'
GROUP BY ngram, first LIMIT 100

Errors:(L1:28):
Expression 'second' is not present in the GROUP BY list

Resolution:
Add second to GROUP BY - All non-aggregate fields in a SELECT must
appear in the GROUP BY
```

Request Encoding Error (4 of 7)

Missing punctuation

```
SELECT word SUM(word_count) AS total_count FROM
[publicdata:samples.shakespeare] GROUP BY word

Errors:
Encountered "" at line 1, column 17.

Resolution:
SELECT word, SUM(word_count) AS total_count FROM
[publicdata:samples.shakespeare] GROUP BY word
```

Request Encoding Error (5 of 7)

Misspelling

```
SELECT word, total_count/word_count AS perc
FROM (SELECT word, SUM(word_count) AS totaal_count,
word_count FROM [publicdata:samples.shakespeare]
GROUP BY word, word_count)

Errors:
Field 'total_count' not found; did you mean 'totaal_count'?

Resolution:
SELECT word, totaal_count/word_count…
```

Request Encoding Error (6 of 7)

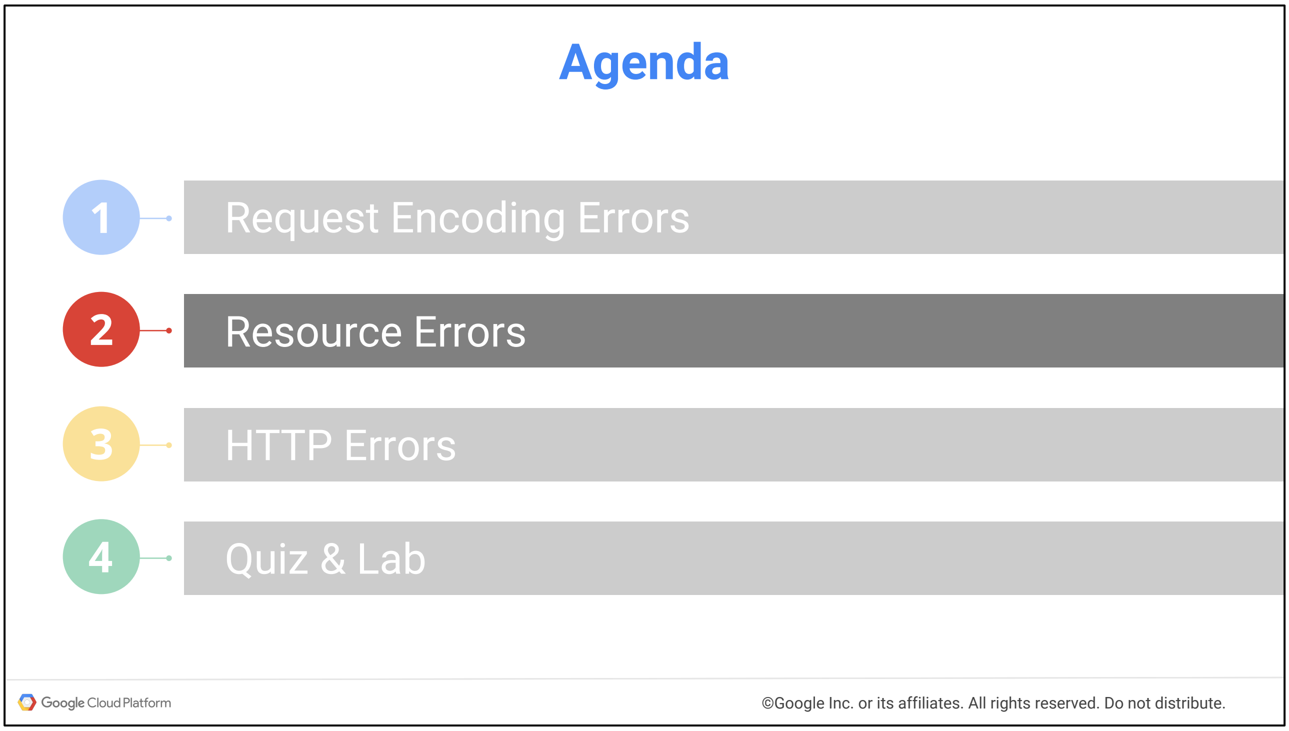
Misplaced statement

```
SELECT * WHERE repository_owner = 'facebook' AND
repository_name = 'react' FROM
[githubarchive:github.timeline] LIMIT 1000

Errors:
Encountered “ “WHERE” “WHERE “” at line 2, column 1. Was
expecting:

Resolution:
SELECT * FROM [githubarchive:github.timeline] WHERE...
```

Request Encoding Error (7 of 7)

Ambiguous field

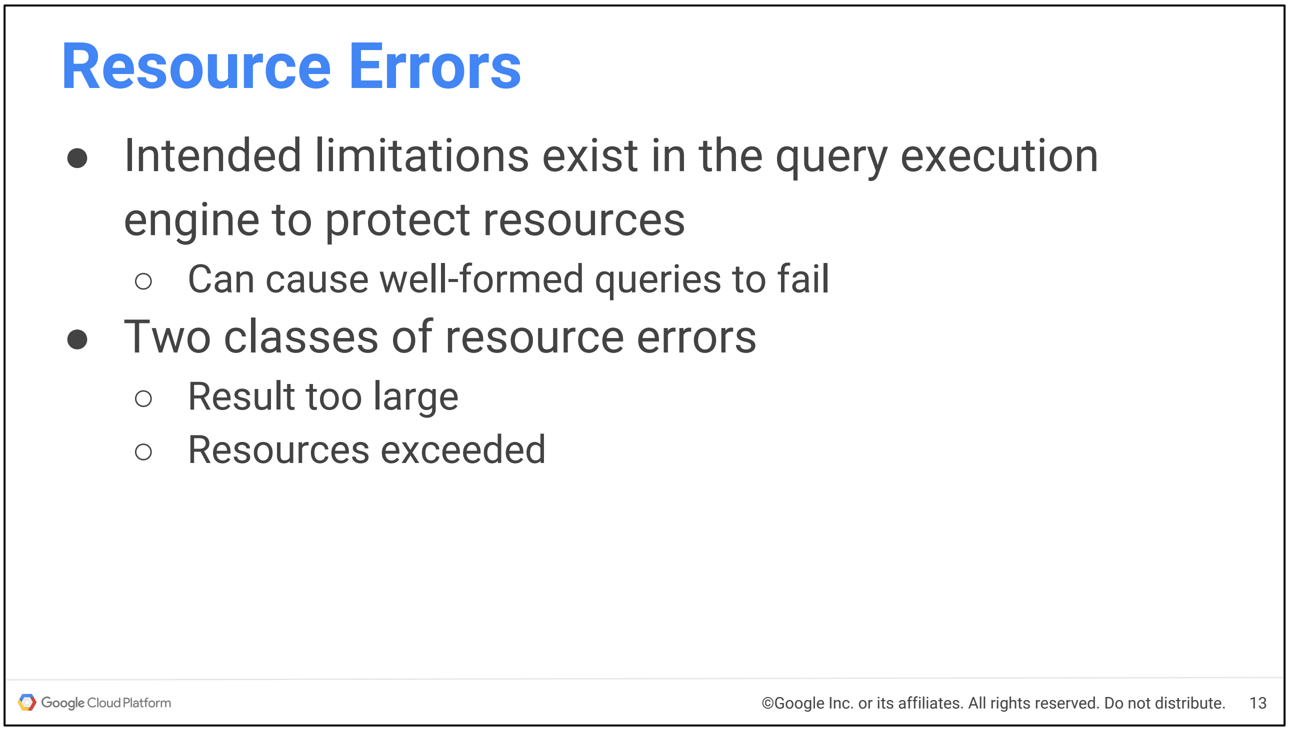
```
SELECT carrier, sum(carrier_delay) as delay FROM
(select * from TABLE_QUERY(flightPerformance,'table_id
contains "flights201"')) A join flightPerformance.carriers
b on a.carrier=b.carrier_code group by 1 order by 2 desc
limit 5

Errors:
2.8 - 2.14: Ambiguous field reference carrier.

Resolution:
Both tables contain a carrier field - You must clarify
SELECT carrier…
```



Agenda

1. Request Encoding Errors
2. Resource Errors
3. HTTP Errors
4. Quiz & Lab

Google Cloud Platform



Resource Errors

- Intended limitations exist in the query execution engine to protect resources
  - Can cause well-formed queries to fail
- Two classes of resource errors
  - Result too large
  - Resources exceeded

Notes:
There are technical limitations due to things like memory of individual nodes. Intended limitations protect the tenant resource pool from users taking advantage of resources.

Result Too Large

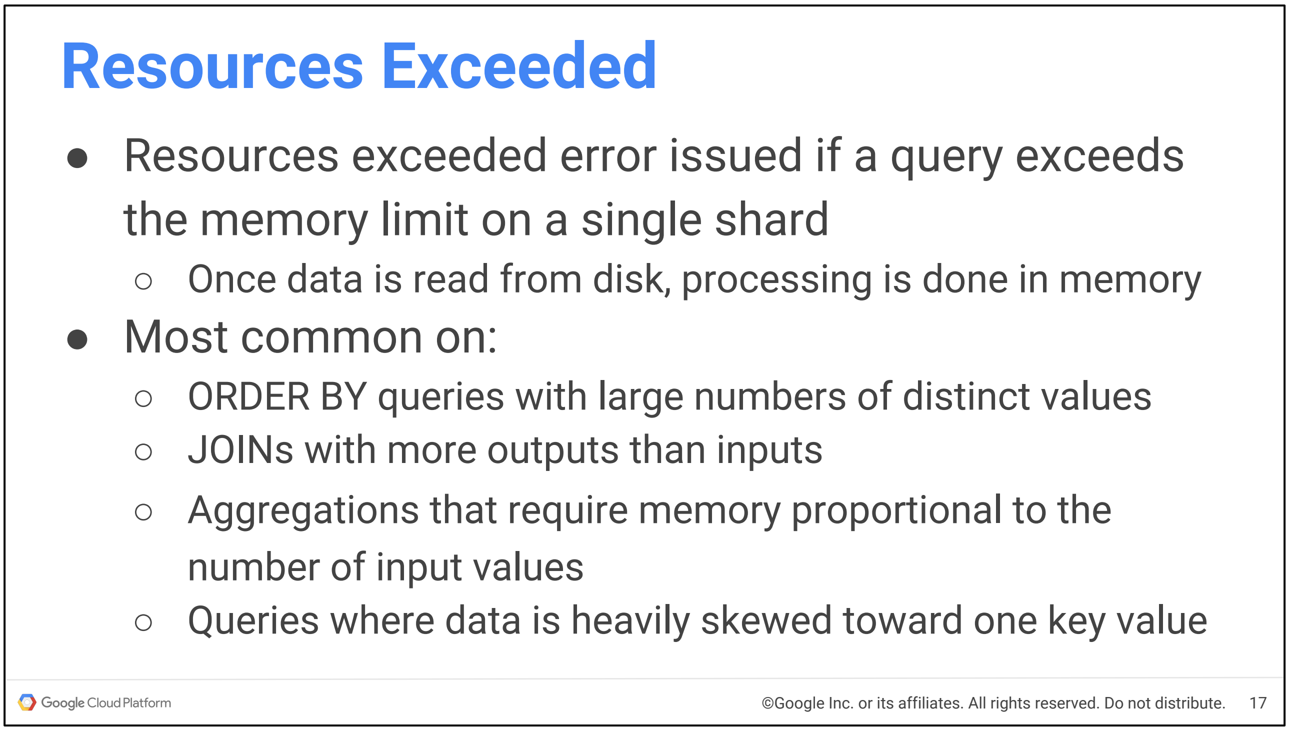
- BigQuery limits result sets to approximately 128MB compressed
  - Queries returning larger results cannot fit into response
- Result too large:
  - Commonly thrown on queries that use an ORDER BY with large cardinality
  - Can happen in multiple stages of the serving tree

Handling Result Too Large Errors (1 of 2)

- Use filters to limit the result set
- Use LIMIT clause
- Remove ORDER BY for large datasets (order by without limit is meaningless)
- Specify destination table and use `allowLargeResults` flag
  - Impacts query performance

Handling Result Too Large Errors (2 of 2)

- Limitations of `allowLargeResults` flag
  - Cannot specify top-level ORDER BY, TOP or LIMIT clause
    - Negates benefit of `allowLargeResults` because query output is no longer computed in parallel
  - Using `allowLargeResults` flag with ORDER BY can cause resources exceeded errors
    - Master applies final sorting
  - Using `allowLargeResults` with window functions requires PARTITION BY clause



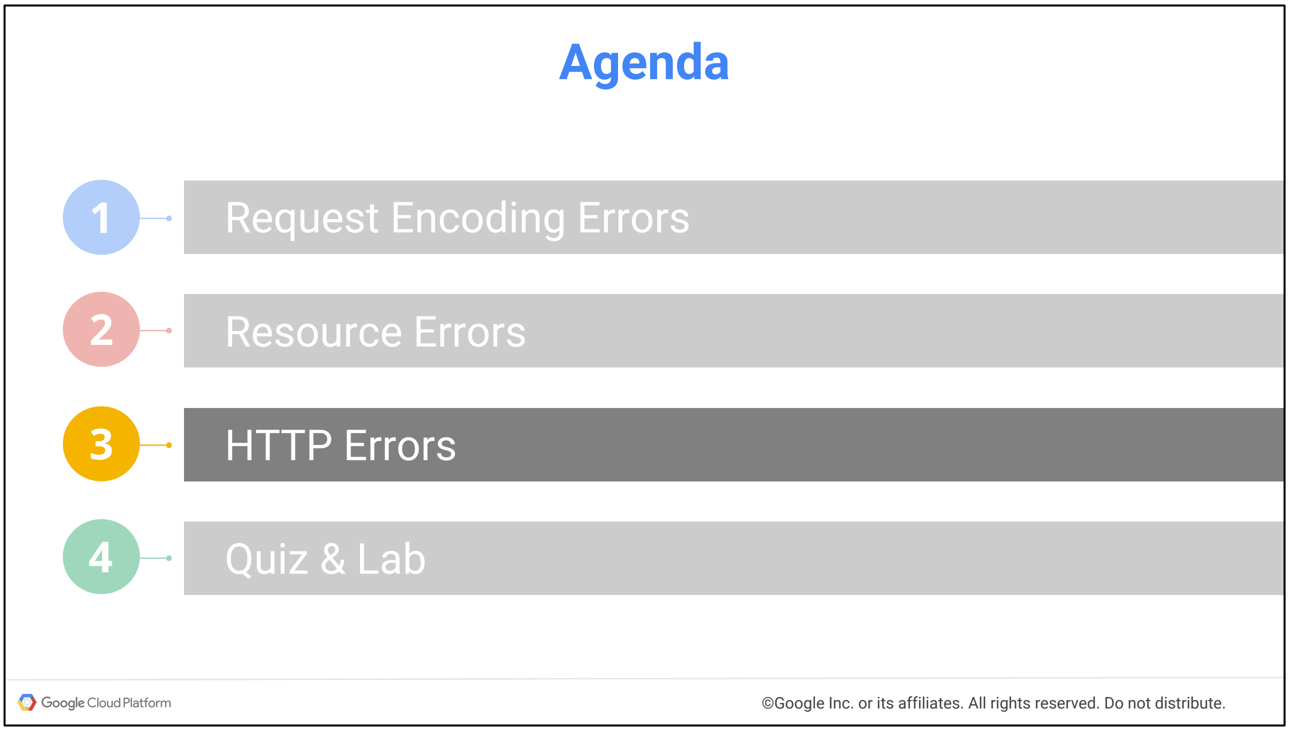
Resources Exceeded

- Resources exceeded error issued if a query exceeds the memory limit on a single shard
  - Once data is read from disk, processing is done in memory
- Most common on:
  - ORDER BY queries with large numbers of distinct values
  - JOINs with more outputs than inputs
  - Aggregations that require memory proportional to the number of input values
  - Queries where data is heavily skewed toward one key value

Notes:
ORDER BY queries of large cardinality required data to be ordered on a single node. In cases of "data skew", where data is heavily skewed towards one key value (for example, a "guest" ID or a null key), one worker receives all the records for that key, and is overloaded.

Handling Resources Exceeded Errors

- If possible, limit use of ORDER BY
- Use aggregation functions that generate fewer output results than input rows
- Avoid JOINs that generate more outputs than inputs
- Avoid queries that create data skew
    - Queries on “guest” IDs or null values
- No rule that works for every case



Agenda

1. Request Encoding Errors
2. Resource Errors
3. **HTTP Errors**
4. Quiz & Lab

HTTP Errors and Responses

- All BigQuery API calls are HTTP requests
- All HTTP requests return status codes
- Codes between 200 and 299 are success codes
- HTTP error codes are between 400 and 599
- BigQuery returns a standard JSON response on error

Sample error response

```
"error": {
"errors": [
{
  "domain": "global",
  "reason": "notFound",
  "message": "Not Found: Dataset
myproject:foo"
}],
"code": 404,
"message": "Not Found: Dataset
myproject:foo"
}
```

Agenda

1. Request Encoding Errors
2. Resource Errors
3. HTTP Errors
4. Quiz & Lab

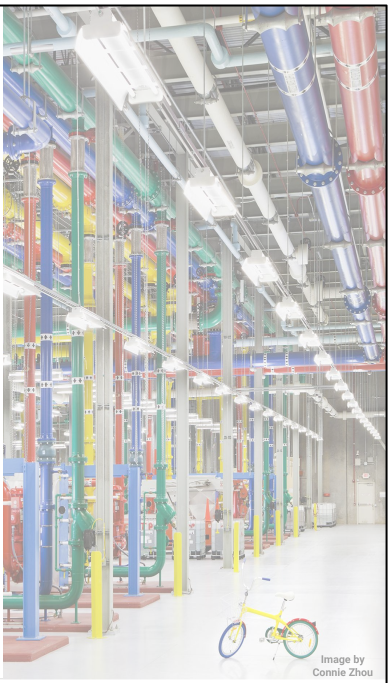
Module Review

How do you allow result sets larger than 128 MB?
*(select **2** of the available options)*

- ❑ Use interactive query priority
- ❑ Use a FLATTEN operator
- ❑ Use the “Allow Large Results” option
- ❑ Enable query caching
- ❑ Set a destination table

Lab

Resolving query and resource errors



Resources

- Troubleshooting
  https://cloud.google.com/bigquery/troubleshooting-errors
- Stack Overflow
  http://stackoverflow.com/questions/tagged/google-bigquery

Module Review

How do you allow result sets larger than 128 MB?
*(select **2** of the available options)*

- ❑ Use interactive query priority
- ❑ Use a FLATTEN operator
- ✓ Use the “Allow Large Results” option
- ❑ Enable query caching
- ✓ Set a destination table

cloud.google.com