

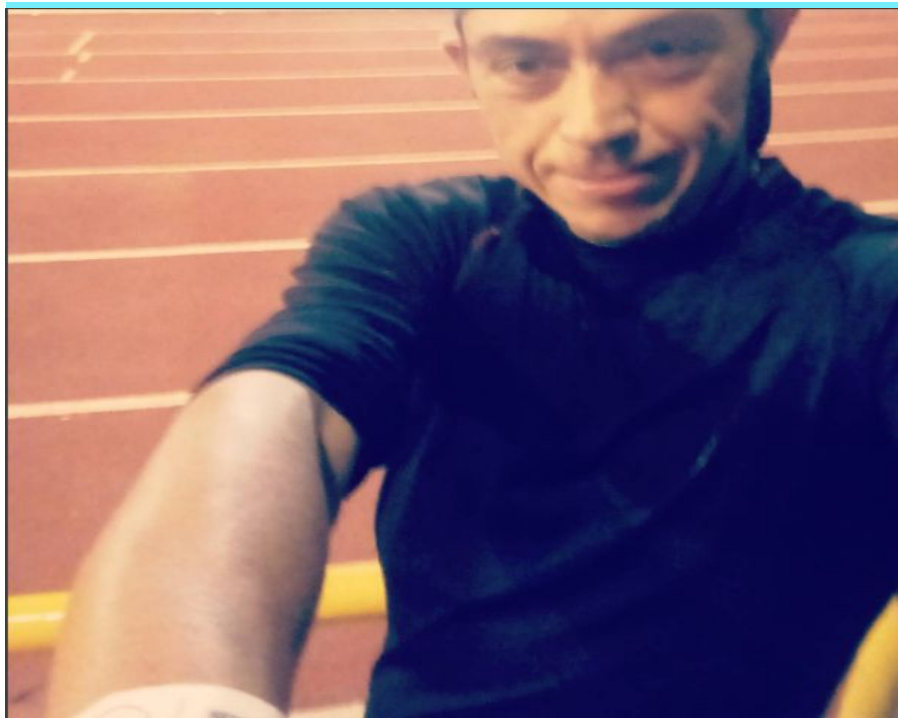


Universitat Oberta
de Catalunya

Análisis de Big Data usando SQL

uoc.edu

UOC Data Day
14 de Junio Barcelona
#DataScienceUOC

Bio

Data Scientist en Edreams
PhD. Inteligencia Artificial
Apasionado del lenguaje R

Cuando no estoy analizando
datos estoy corriendo, nadando
o sobre una bicicleta. 🏃‍♂️ 🏊‍♂️ 🚴‍♂️

Indice



Big Query

Un mundo repleto de datos

Solución de Google

¿Que es BigQuery?

¿Cuanto es útil BigQuery y cuando no?

Una posible configuración con BigQuery

ETL's y visualización con BigQuery

Costes

Demo

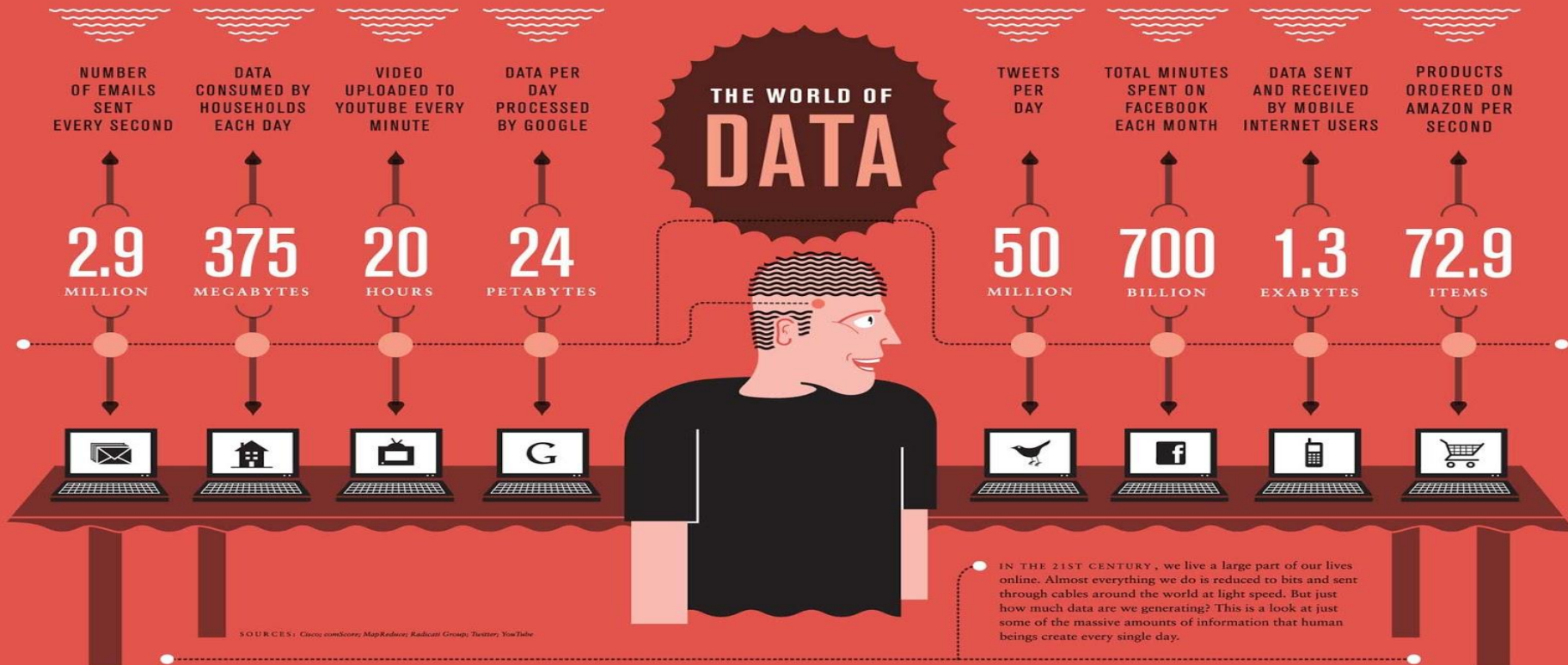
Arquitectura BigQuery

Estructura de Proyectos





Sintaxis

Demo

Un mundo repleto de datos



Solución de Google

| INGESTA | ALMACENAMIENTO | PROCESO Y ANÁLISIS | EXPLORACIÓN Y VISUALIZACIÓN |
|---|---|---|--|
|  App Engine |  Cloud Storage |  Cloud Dataflow |  Cloud Datalab |
|  Compute Engine |  Cloud SQL |  Cloud Dataproc |  Google Data Studio |
|  Container Engine |  Cloud Datastore |  BigQuery |  Google Sheets |
|  Cloud Pub/Sub |  Cloud Bigtable |  Cloud ML | |
|  Stackdriver Logging |  BigQuery |  Cloud Vision API | |
|  Cloud Transfer Service | |  Cloud Speech API | |
| | |  Translate API | |
| | |  Cloud Natural Lang API | |

¿Que es BigQuery?

Es un servicio web que permite realizar consultas **SQL** de un conjunto de datos a gran escala que funciona de forma **super rápida** con Google Storage mientras explotan la potente infraestructura de Google.

Tiene el objetivo de facilitar el análisis de datos y permitir a las empresas adoptar decisiones de negocio de una forma más **rápida y precisa**, ya que se ejecuta en la nube y permite análisis en tiempo real.



¿Cuándo es útil BigQuery y cuando no?

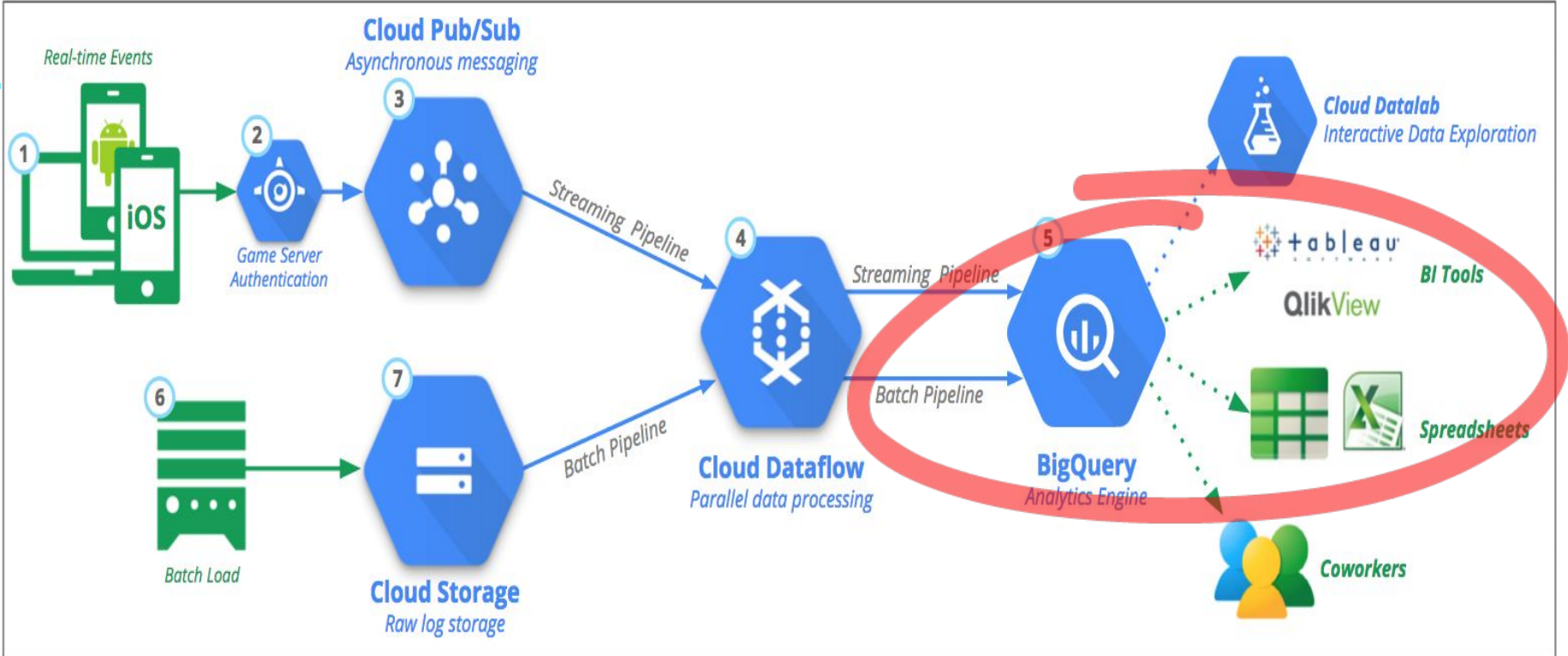


¿Como podemos acceder a los datos en BigQuery?

- BigQuery Web
- BQ línea de comandos
- Servicio API

C#, Go, Java, Node.js, PHP, Python, Ruby

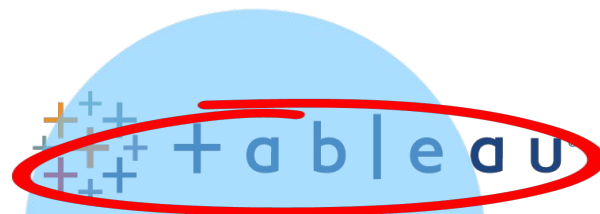
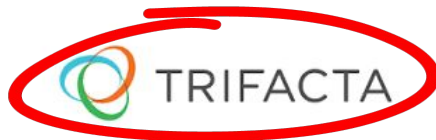
Una posible configuración con BigQuery



ETL's y visualización con BigQuery

Herramientas ETL's

Visualización y BI



Google
Sheets



Costes

BIGQUERY PRICING

BigQuery charges for data storage, streaming inserts, and for querying data, but loading and exporting data are free of charge. For detailed pricing information, please view the [pricing guide](#).

| ITEM | PRICE |
|--|--|
| Storage | \$0.02 per GB, per month \$0.01 per GB, per month for long term storage |
| Streaming Inserts | \$0.01 per 200 MB |
| Loading, Copying, or Exporting Data Metadata Operations | Free |

When querying data you can choose from two different pricing options:

| FEATURE | PRICE |
|-------------------|---|
| Pay-as-you-go | \$5 per TB First terabyte (1 TB) per month is free* |
| Flat-rate pricing | Starting at \$40,000/month for a dedicated reservation of 2,000 slots. For more information, see Flat-rate pricing . |

* The first terabyte (1 TB) of data processed with Google BigQuery each month is free.

Note: BigQuery's [quota policy](#) applies for these operations.

Indice



Big Query

Un mundo repleto de datos

Solución de Google

¿Que es BigQuery?

¿Cuándo es útil BigQuery y cuándo no?

Una posible configuración con BigQuery

ETL's y visualización con BigQuery

Costes

Demo

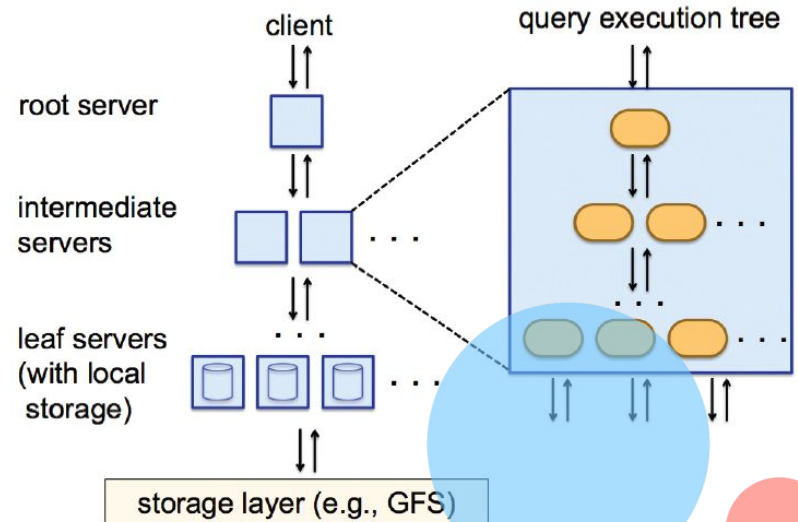
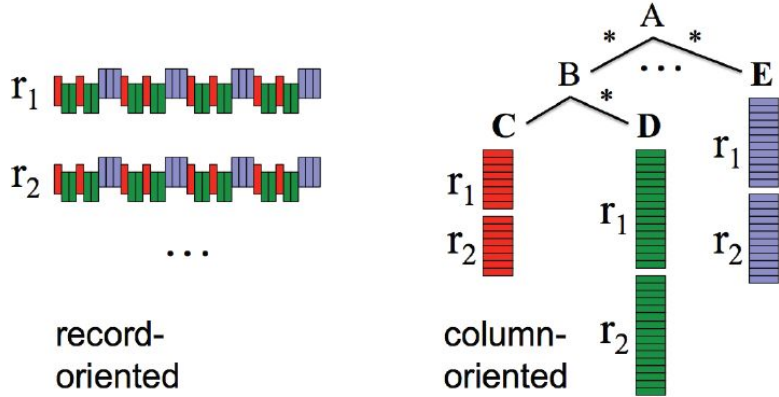
Arquitectura BigQuery

Estructura de Proyectos

Sintaxis

Demo

Arquitectura BigQuery



Denormalización



Estructura de Proyectos

Proyectos
Datasets
Tablas

Jobs (Trabajos)

The image consists of three screenshots illustrating the structure of a project in ArcRecord Cloud:

- Top Screenshot:** Shows the main project view with a tree structure. The root node is "ArcRecord Cloud", which contains "dataset_name", "earthquake_dataset", "lab", and "test". A context menu is open over "dataset_name" with options: "Create new dataset", "Switch to project", and "Refresh".
- Middle Screenshot:** Shows the "dataset_name" node expanded. It contains "table_name", "table_name1", and "earthquake_dataset". A context menu is open over "table_name" with options: "Load", "Create new table", "Share dataset", and "Delete dataset".
- Bottom Screenshot:** Shows the "table_name" node expanded. It contains "table_name1". A context menu is open over "table_name" with options: "Copy table", "Export table", and "Delete table".

Overlaid on the bottom screenshot is a "COMPOSE QUERY" window showing a "Job History" table:

| Recent Jobs | |
|-------------|--|
| Load | uploaded file to arcrecord-cloud-project/test/table1 |
| Load | uploaded file to arcrecord-cloud-project/test/table1 |
| Load | uploaded file to arcrecord-cloud-project/test/table1 |

Sintaxis

```
query_statement:
  [ WITH with_query_name AS ( query_expr ) [, ...] ]
  query_expr

query_expr:
  { select | ( query_expr ) | query_expr set_op query_expr }
  [ ORDER BY expression [{ ASC | DESC }] [, ...] ]
  [ LIMIT count [ OFFSET skip_rows ] ]

select:
  SELECT [{ ALL | DISTINCT }]
    { [ expression ] * [ EXCEPT ( column_name [, ...] ) ]
      [ REPLACE ( expression [ AS ] column_name [, ...] ) ]
      [ expression [ [ AS ] alias ] } [, ...]
  [ FROM from_item [, ...] ]
  [ WHERE bool_expression ]
  [ GROUP BY expression [, ...] ]
  [ HAVING bool_expression ]
  [ WINDOW window_name AS ( window_definition ) [, ...] ]

set_op:
  UNION { ALL | DISTINCT }

from_item: {
  table_name [ [ AS ] alias ] |
  join |
  ( query_expr ) [ [ AS ] alias ] |
  field_path |
  { UNNEST( array_expression ) | UNNEST( array_path ) | array_path }
  [ [ AS ] alias ] [ WITH OFFSET [ [ AS ] alias ] ] |
  with_query_name [ [ AS ] alias ]
}

join:
  from_item join_type JOIN from_item
  [ { ON bool_expression | USING ( join_column [, ...] ) } ]

join_type:
  { [INNER] | CROSS | FULL [OUTER] | LEFT [OUTER] | RIGHT [OUTER] }
```


Sintaxis

Agregados simples

```
SELECT COUNT(foo), MAX(foo), STDDEV(foo) FROM ...
```

Consultas más complejas

```
SELECT ... FROM ... WHERE REGEXP_MATCH(ulr, "\.com$") AND user CONTAINS 'test'
```

JOINS

```
SELECT COUNT(*) FROM foo1 (JOIN foo2) ON foo1.id = foo2.id...
```

Demo

Proyecto Stackoverflow

Contar usuarios

```
SELECT count(*) FROM [bigquery-public-data:stackoverflow.users] LIMIT 1000
```

Contar usuarios con filtros

```
SELECT display_name,location FROM [bigquery-public-data:stackoverflow.users] where  
reputation>100 and REGEXP_MATCH(location,"London") LIMIT 1000
```

Diferentes tablas mediante JOINS

```
SELECT  
  u.id, u.display_name  
FROM  
  [bigquery-public-data:stackoverflow.users] as u  
JOIN  
  (SELECT * FROM [bigquery-public-data:stackoverflow.stackoverflow_posts] )  
AS  
  s  
ON  
  u.id = s.owner_user_id limit 1000
```

Demo

Diferentes tablas mediante JOINS GROUP_BY

```
SELECT
  u.id, u.display_name
FROM
  [bigquery-public-data:stackoverflow.users] as u
JOIN
  (SELECT * FROM [bigquery-public-data:stackoverflow.stackoverflow_posts] )
AS
  s
ON
  u.id = s.owner_user_id GROUP BY u.id, u.display_name limit 1000
```

Diferentes tablas mediante JOINS GROUP_BY + filtros

```
SELECT
  u.id, u.display_name
FROM
  [bigquery-public-data:stackoverflow.users] as u
JOIN
  (SELECT * FROM [bigquery-public-data:stackoverflow.stackoverflow_posts] )
AS
  s
ON
  u.id = s.owner_user_id where u.reputation>100 and REGEXP_MATCH(u.location,"London") GROUP BY u.id, u.display_name limit 1000
```

Demo

Proyecto noaa_gsod

Múltiples tablas

```
#standardSQL
```

```
SELECT
  max,
  mo,
  da,
  year
FROM
  `bigquery-public-data.noaa_gsod.gsod19*`
WHERE
  _TABLE_SUFFIX BETWEEN '29'
  AND '40'
```

Proyecto github_timeline versus github_nested - Flat vs JSON

```
SELECT repository.url, actor_attributes.blog FROM
[bigquery-public-data:samples.github_nested] limit 30000
```

Demo

Diferentes tablas mediante JOINS GROUP_BY

```
SELECT
  u.id, u.display_name
FROM
  [bigquery-public-data:stackoverflow.users] as u
JOIN
  (SELECT * FROM [bigquery-public-data:stackoverflow.stackoverflow_posts] )
AS
  s
ON
  u.id = s.owner_user_id GROUP BY u.id, u.display_name limit 1000
```


Demo


INTEGRACIÓN

GOOGLE SPREADSHEET [LINK](#)

GOOGLE STUDIO [LINK](#)

Universitat Oberta
de Catalunya

 davidcabanillas75

 davidcabanillasbarbacil



<https://bigquery.cloud.google.com>
