Introduction

Structured Query Language (Yapılandırılmış Sorgu Dili) veya **SQL**, veri tabanlarıyla kullanılan bir programlama dilidir ve herhangi bir veri bilimci için önemli bir beceridir. Bu derste, çok büyük veri setlerine SQL uygulamanızı sağlayan bir web hizmeti olan **BigQuery**'yi kullanarak SQL becerilerinizi geliştireceksiniz.

Bu derste, BigQuery veri setlerine erişmenin ve bunları incelemenin temellerini öğreneceksiniz. Bu temel bilgileri edindikten sonra, SQL becerilerinizi geliştirmek için tekrar buraya döneceğiz.

# Your first BigQuery commands

BigQuery'yi kullanmak için aşağıdaki Python paketini içe aktaracağız:

from google.cloud import bigquery

İş akışındaki ilk adım, bir **Client** nesnesi oluşturmaktır. Yakında göreceğiniz gibi, bu **Client** nesnesi BigQuery veri setlerinden bilgi alımında merkezi bir rol oynayacaktır.

*# Create a "Client" object*

client = bigquery.Client()

Using Kaggle's public dataset BigQuery integration

Bilgisayar bilimleri ve siber güvenlik haberlerine odaklanan bir web sitesi olan **Hacker News**'teki gönderilerin veri setiyle çalışacağız.

BigQuery'de her veri seti, ilgili bir projenin içinde yer alır. Bu durumda, **hacker\_news** veri setimiz, **bigquery-public-data** projesinin içindedir. Veri setine erişmek için,

* İlk olarak **dataset()** metodu ile veri setine bir referans oluştururuz.
* Ardından, az önce oluşturduğumuz referansı kullanarak **get\_dataset()** metodu ile veri setini çekeriz.

# Construct a reference to the "hacker\_news" dataset

dataset\_ref = client.dataset("hacker\_news", project="bigquery-public-data")

# API request - fetch the dataset

dataset = client.get\_dataset(dataset\_ref)

Her veri seti, yalnızca bir tablo koleksiyonudur. Bir veri setini, tamamı satır ve sütunlardan oluşan birden çok tablo içeren bir e-tablo dosyası gibi düşünebilirsiniz.

Veri setindeki tabloları listelemek için **list\_tables()** metodunu kullanırız.

*# List all the tables in the "hacker\_news" dataset*

tables = list(client.list\_tables(dataset))

*# Print names of all tables in the dataset (there are four!)*

for table **in** tables:

print(table.table\_id)

comments

full

full\_201510

stories

Bir veri kümesini nasıl getiriyorsak, bir tabloyu da getirebiliriz. Aşağıdaki kod hücresinde, hacker\_news veri kümesindeki tüm tabloyu getiriyoruz.

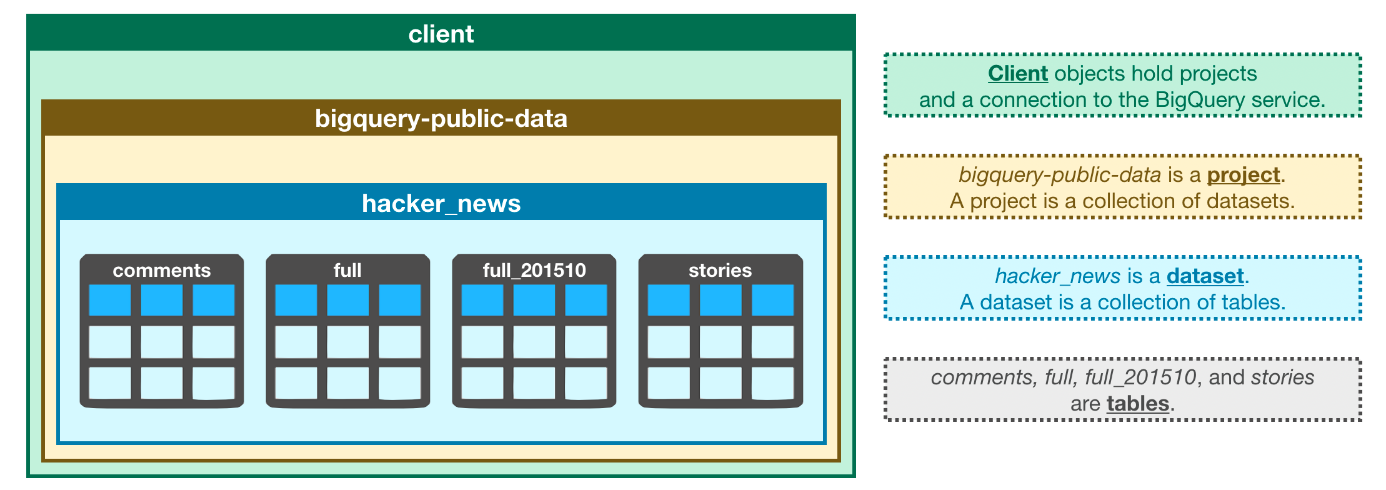
*# Construct a reference to the "full" table*

table\_ref = dataset\_ref.table("full")

*# API request - fetch the table*

table = client.get\_table(table\_ref)

Bir sonraki bölümde, bu tablonun içeriğini daha detaylı inceleyeceksiniz. Şimdilik, öğrendiklerinizi pekiştirmek için aşağıdaki görseli kullanın.



# Table schema

Bir tablonun yapısına **şema** adı verilir. İstediğimiz veriyi etkili bir şekilde çekebilmek için bir tablonun şemasını anlamamız gerekir.

Bu örnekte, yukarıda çektiğimiz **full** tablosunu inceleyeceğiz.

*# Print information on all the columns in the "full" table in the "hacker\_news" dataset*

table.schema

[SchemaField('title', 'STRING', 'NULLABLE', 'Story title', (), None),

SchemaField('url', 'STRING', 'NULLABLE', 'Story url', (), None),

SchemaField('text', 'STRING', 'NULLABLE', 'Story or comment text', (), None),

SchemaField('dead', 'BOOLEAN', 'NULLABLE', 'Is dead?', (), None),

SchemaField('by', 'STRING', 'NULLABLE', "The username of the item's author.", (), None),

SchemaField('score', 'INTEGER', 'NULLABLE', 'Story score', (), None),

SchemaField('time', 'INTEGER', 'NULLABLE', 'Unix time', (), None),

SchemaField('timestamp', 'TIMESTAMP', 'NULLABLE', 'Timestamp for the unix time', (), None),

SchemaField('type', 'STRING', 'NULLABLE', 'Type of details (comment, comment\_ranking, poll, story, job, pollopt)', (), None),

SchemaField('id', 'INTEGER', 'NULLABLE', "The item's unique id.", (), None),

SchemaField('parent', 'INTEGER', 'NULLABLE', 'Parent comment ID', (), None),

SchemaField('descendants', 'INTEGER', 'NULLABLE', 'Number of story or poll descendants', (), None),

SchemaField('ranking', 'INTEGER', 'NULLABLE', 'Comment ranking', (), None),

SchemaField('deleted', 'BOOLEAN', 'NULLABLE', 'Is deleted?', (), None)]

Her **SchemaField**, bize belirli bir sütun (alan olarak da adlandırılır) hakkında bilgi verir. Bu bilgiler sırasıyla şunlardır:

* Sütunun **adı**
* Sütundaki **alan türü** (veya veri türü)
* Sütunun **modu** (**'NULLABLE'** bir sütunun NULL değerlere izin verdiği anlamına gelir ve varsayılan ayardır)
* O sütundaki verinin **açıklaması**

İlk alanın SchemaField'ı şöyledir:

SchemaField('by', 'string', 'NULLABLE', "The username of the item's author.",())

Bu bize şunu söyler:

* Alan (veya sütun) **by** olarak adlandırılmıştır.
* Bu alandaki veriler **dizedir (string)**.
* **NULL** değerlere izin verilir.
* Her bir öğenin yazarına karşılık gelen kullanıcı adlarını içerir.

Doğru olduğundan emin olmak için **full** tablosunun ilk beş satırını kontrol etmek amacıyla **list\_rows()** metodunu kullanabiliriz. (Bazen veri tabanlarının güncel olmayan açıklamaları olabilir, bu yüzden kontrol etmek iyi bir fikirdir.) Bu metod, **to\_dataframe()** metoduyla hızlı bir şekilde bir pandas DataFrame'e dönüştürülebilen bir BigQuery **RowIterator** nesnesi döndürür.

*# Preview the first five lines of the "full" table*

client.list\_rows(table, max\_results=5).to\_dataframe()

/opt/conda/lib/python3.7/site-packages/ipykernel\_launcher.py:2: UserWarning: Cannot use bqstorage\_client if max\_results is set, reverting to fetching data with the tabledata.list endpoint.

| title | url | text | dead | by | score | time | timestamp | type | id | parent | descendants | ranking | deleted |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | None | None | I would rather just have wired earbuds, period... | None | zeveb | NaN | 1591717736 | 2020-06-09 15:48:56+00:00 | comment | 23467666 | 23456782 | NaN | NaN | None |
| 1 | None | None | DNS? | None | nly | NaN | 1572810465 | 2019-11-03 19:47:45+00:00 | comment | 21436112 | 21435130 | NaN | NaN | None |
| 2 | None | None | These benchmarks seem pretty good. Filterable... | None | mrkeen | NaN | 1591717727 | 2020-06-09 15:48:47+00:00 | comment | 23467665 | 23467426 | NaN | NaN | None |
| 3 | None | None | Oh really?<p>\* Excel alone uses 86.1MB of priv... | None | oceanswave | NaN | 1462987532 | 2016-05-11 17:25:32+00:00 | comment | 11677248 | 11676886 | NaN | NaN | None |
| 4 | None | None | These systems are useless. Of the many flaws:... | None | nyxxie | NaN | 1572810473 | 2019-11-03 19:47:53+00:00 | comment | 21436113 | 21435025 | NaN | NaN | None |

list\_rows() metodu, yalnızca belirli bir sütundaki bilgilere bakmamızı da sağlar. Örneğin, by sütunundaki ilk beş girdiyi görmek istiyorsak, bunu yapabiliriz!

*# Preview the first five entries in the "by" column of the "full" table*

client.list\_rows(table, selected\_fields=table.schema[:1], max\_results=5).to\_dataframe()

/opt/conda/lib/python3.7/site-packages/ipykernel\_launcher.py:2: UserWarning: Cannot use bqstorage\_client if max\_results is set, reverting to fetching data with the tabledata.list endpoint.

| title |
| --- |
| 0 | None |
| 1 | None |
| 2 | None |
| 3 | None |
| 4 | None |

# Disclaimer

Kodlama alıştırmasına geçmeden önce, hâlihazırda biraz SQL bilgisi olanlar için hızlı bir uyarı:

Her Kaggle kullanıcısı, 30 gün içinde 5 TB veriyi ücretsiz olarak tarayabilir. Bu sınıra ulaştığınızda, sıfırlanması için beklemeniz gerekir.

Şimdiye kadar gördüğünüz komutlar, bu limitin anlamlı bir kısmını talep etmez. Ancak bazı BiqQuery veri setleri çok büyüktür. Bu nedenle, eğer zaten SQL biliyorsanız, kotanızı etkili bir şekilde nasıl kullanacağınızı görene kadar **SELECT** sorgularını çalıştırmayı bekleyin. Eğer bunu okuyan çoğu insan gibiyseniz ve bu sorguları nasıl yazacağınızı henüz bilmiyorsanız, bu uyarı hakkında endişelenmenize gerek yok.