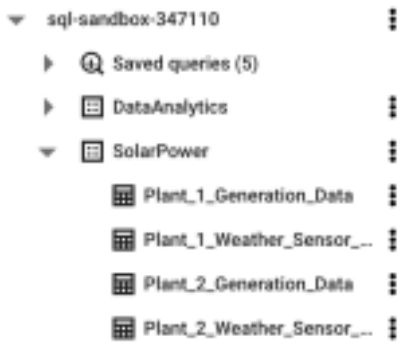
Solar_Power_Dataset

Go over to this Kaggle page, read carefully the description of the dataset as well as the description of each column in the data. There are two sets of data collected for 2 power plants: the “Generation Data” and the ”Weather Sensor Data”:

1. Download the four csv files using the Download button at the top of the page (it will download them into a zipped folder);

2. Move over to BigQuery, below your personal project create a new data set and name it “SolarPower”;

3. In the newly created data set, upload the four files into four separate tables (give to each table the corresponding file name):



4. Open a new editor window and write a query that combines the two pairs of

datasets to have Plant 1 and Plant 2 in the same table and a total of two resulting tables. CREATE those two TABLEs AS the result of the UNION between the above mentioned tables and call them “Generation_Data” and “Weather_Sensor_Data”.

```
CREATE TABLE SolarPower.Generation_Data AS
SELECT *
FROM `sql-sandbox-347110.SolarPower.Plant_1_Generation_Data`
UNION ALL
SELECT *
FROM `sql-sandbox-347110.SolarPower.Plant_2_Generation_Data`;

CREATE TABLE SolarPower.Weather_Sensor_Data AS
SELECT *
FROM `sql-sandbox-347110.SolarPower.Plant_1_Weather_Sensor_Data`
UNION ALL
SELECT *
FROM `sql-sandbox-347110.SolarPower.Plant_2_Weather_Sensor_Data`;
```

5. How many inverters (*hint*: *source_key*) are there in each plant?

```
SELECT plant_id, count(distinct source_key) as nr_inverters
FROM SolarPower.Generation_Data
GROUP BY plant_id;
```

6. How many days of observations do we have for each plant?

```
SELECT plant_id, count(distinct extract(date from date_time)) as
nr_days FROM SolarPower.Generation_Data
GROUP BY plant_id;
```

7. Which inverter generated the highest total yield? Which plant does it belong to? *Hint*: *careful with the aggregation function you use with the total_yield field, read carefully its description on the kaggle page.*

```
SELECT plant_id, source_key, max(total_yield) as total_yield
FROM SolarPower.Generation_Data
GROUP BY plant_id, source_key
ORDER BY total_yield desc;
```

DATA DEFINITION LANGUAGE

Using the w3school environment, create the following two tables (from the previous lesson on Joins) specifying the correct schemas (column names, data types, etc):

1. Create the “students” table:

students

Row	nmStudent	idCourse
1	Mark	1
2	Jack	2
3	Ivan	3
4	Beth	3
5	Sara	6

CREATE TABLE students(Row integer, nmStudent varchar(30), idCourse integer)

INSERT INTO students(Row,nmStudent,idCourse)
VALUES (1,'Mark',1),(2,'Jack',2),(3,'Ivan',3),(4,'Beth',3),(5,'Sara',6)

2. Create the “courses” table:

courses

Row	idCourse	nmCourse
1	1	Math
2	2	English
3	3	Physics
4	4	Business
5	5	History

CREATE TABLE courses(Row integer, idCourse integer, nmCourse varchar(30))

INSERT INTO courses(Row,idCourse, nmCourse)
VALUES (1,1,’Math’),(2,2,’English’),(3,3,’Physics’),(4,4,’Business’),(5,5,’History’) 3. Use all four types of JOINs as we saw them in lesson 2.7.17 and familiarise yourself with how JOINs work and when null values are generated and think about why that happens.

SELECT * FROM courses INNER/LEFT/RIGHT/FULL OUTER JOIN
students on courses.idCourse = students.idCourse

4. There is an error in the students table, change Sara's idCourse from 6 to 4.

```
UPDATE Students
SET idCourse = 4
WHERE nmStudent = 'Sara'
```

5. There is an error also in the courses table, change the name of idCourse = 5 from "History" to "Economics".

```
UPDATE Courses
SET nmCourse = 'Economics'
WHERE idCourse = 5
```

6. There is a new student in the class, his name is "George" and he is going to follow the Economics course; add a new row of data to the students table containing the new student's information.

```
INSERT INTO students(Row,nmStudent,idCourse)
VALUES (6,George,5)
```