**SES\_Survey\_2\_DataPush**

**Objective**

The objective of this report is to provide a detailed explanation of the code chunk that cleans and transforms the **"Student\_Survey\_2.csv"** data for later use in BigQuery. This report will explain the specific steps taken in the code, as well as the functions and libraries used.

**Methodology**

The code chunk uses Python programming language and the following libraries:

**pandas:** for creating and manipulating DataFrames

**google-auth:** for authenticating the user in Google Colab

**google-cloud-bigquery:** for pushing data to BigQuery

The code follows the following steps:

**Step 1: Importing and Authenticating**

The code first authenticates the user in Google Colab using the google-auth library. The necessary libraries are then imported, including pandas.

**Step 2: Creating a DataFrame(df2)**

Next, the code creates a Pandas DataFrame by reading the CSV file uploaded in the previous step. It then renames some of the columns to a specific format.

**Step 3: Data Cleaning**

The code performs some data cleaning by applying the following transformations:

**Dropping unwanted columns:** The columns **'Timestamp', 'Username', 'Unnamed: 12', 'Unnamed: 13', and 'Unnamed: 14'** were dropped from the dataframe using the **df2.drop()** function. These columns were not required for the analysis and contained no useful information.

**Checking for null values:** The **df2.isnull(). sum()** function was used to check the number of null values in each column. This step is important to ensure that there are no missing values in the data, which could cause errors in analysis or modeling.

**Handling missing values:** In this case, the **'Events\_suggestions'** column had some missing values. To handle this, the **df2['Events\_suggestions']. fillna('None')** function was used to fill all the missing values in that column with the string 'None'. This ensures that there are no null values in the data and makes it easier to work with the data for further analysis.

**Step 4: Creating a BigQuery Table and Inserting Data into it using Python**

In this step, an API connection is made to the Google BigQuery database for the project **'surveyproject-378222'**. The code drops the existing table **'Surevy2\_Base\_Table'** if it already exists. Then it defines a schema for the new table that includes the required columns along with their datatypes.

The code tries to get the existing table and prints a message stating that the table already exists in BigQuery if it exists. Otherwise, the code creates a new table with the defined schema and prints a message that the table has been created in BigQuery.

After creating the table, the code loads the data from the Pandas DataFrame **'df2'** to the BigQuery table using the 'load\_table\_from\_dataframe' function. The function takes the DataFrame, table ID, and job configuration as inputs. The job configuration specifies the source format, write disposition, and other options for the job.

Finally, the code reads the data from the created table 'Surevy2\_Base\_Table' using the **'read\_gbq'** function from the **'pandas\_gbq'** module. The function takes a SQL query string, project ID, and dialect as inputs. The SQL query selects all the columns from the table, and the resulting DataFrame **'sb2**' is described using the 'describe' function.