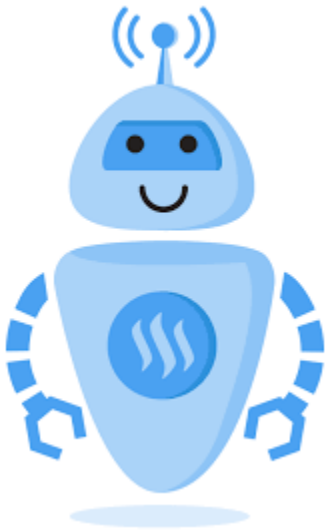
MarketLytics

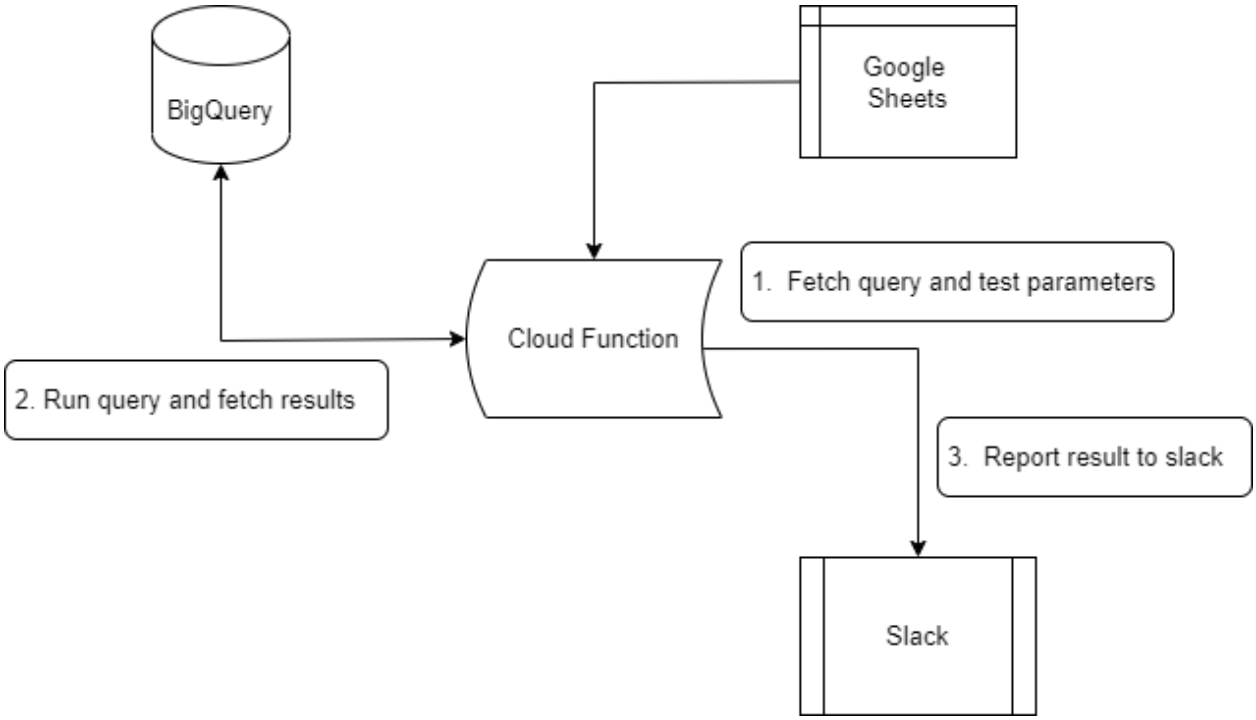


Anoma Bot - User Guide

Introduction

The aim of the Anoma bot is to notify data anomalies, the number of rows added in a day, and any data arrived in a day or not for a particular table of the database.
It is developed in such a way that it can be easily integrated with any project without changing much of the codebase.

Architecture



All the configuration, project ids, tests, and query parameters are configured in a google sheet or in JSON config file.. This sheet/file can hold parameters for multiple tests for multiple different projects.

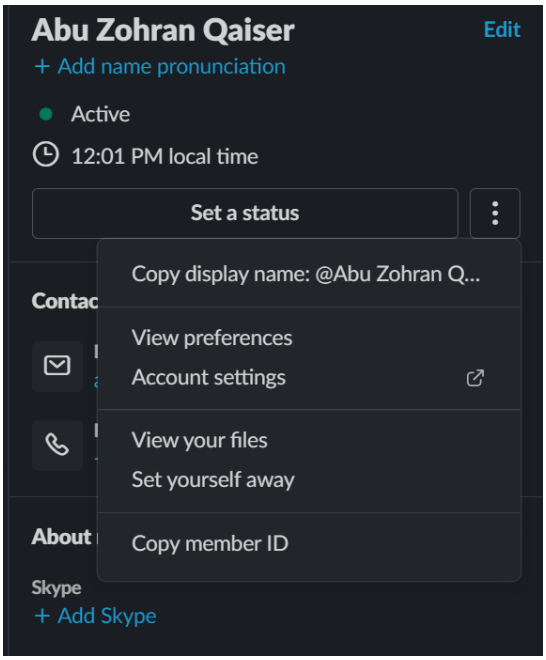
The code is hosted in a cloud function that on getting triggered reads the sheet or config file, picks the specified test parameters, runs it, and reports the results to slack.

Steps to configure Anoma bot by using Google Sheet

1. Configure Google Sheet

Link: Anomaly Bot

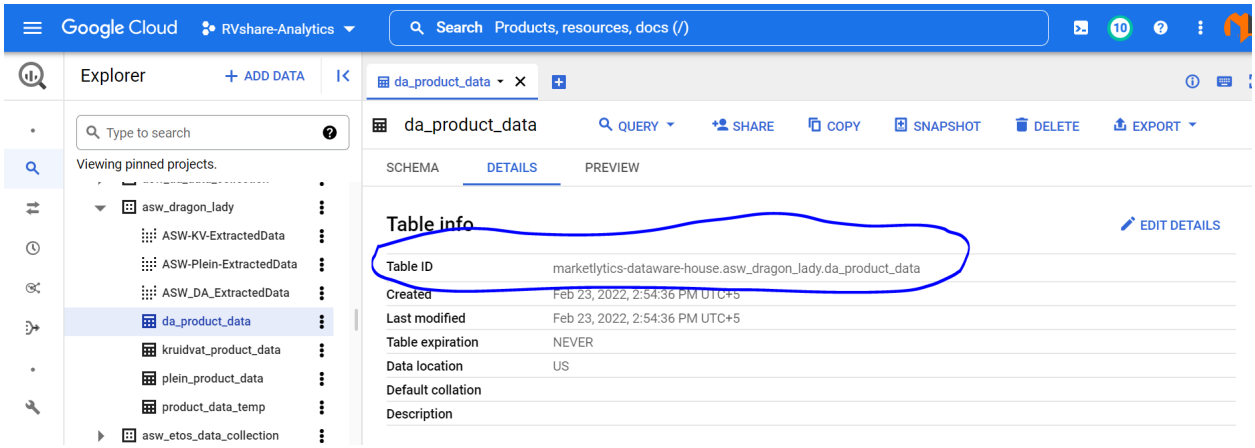
Slack_member_id: an individual who configures the bot so that the individual gets tagged by the bot on the slack channel



It is the unique slack ID of the bot so that it can be tagged by the bot on the slack channel

temp_for_testing_and_errors.

To find your member ID on slack: **Click on profile > Click on three dot dropdown menu > copy member ID**



Project_name: It is the exact project name that contains the required table on google cloud.
test_id: It is the unique integer ID that must be assigned by the one who is configuring the bot.
Test_name: It is the name of the test/task that an individual will assign.
test_type: There are three tests that the bot can perform;

1. Anomaly
2. Data_arrived_or_not
3. No_of_rows

threshold: This parameter will only be passed if the test type is set as “Anomaly” otherwise leave it blank. It is the integer number that identifies the threshold for marking anomalies in the dataset.
timezone: It is the timezone that will be provided by an individual. Since we are sitting in Karachi so it will be “Asia/Karachi”.

cron_schedule: It is the time of the scheduled test in cron format.
main_table_name: It is the name of the table for which the bot will run the test.
To find the main table name: **Go on Google Cloud > Click on the project > Click on the dataset > Click on the table for which bot will run > Click on the details tab > Copy Table ID**

date_column_name: It is the name of the date column of the selected table.

2. Test the bot

- First, join the slack channel: **temp_for_testing_and_errors**
- Put the below-mentioned URL on the browser and replace “<insert test id here>” with the test id that you provided in Google Sheet and run.

https://us-central1-marketlytics-dataware-house.cloudfunctions.net/generalized-anomaly-bot?test_id=<insert test id here>

- The bot will respond with the result on the slack channel

3. Steps to Deploy Cloud function

- If you haven't already, sign up for a GCP account at https://console.cloud.google.com/ and create a new project.

- Install Google Cloud SDK:
  Download and install the Google Cloud SDK from https://cloud.google.com/sdk/docs/install. This will enable you to interact with GCP services from your local machine.

- Authenticate your account:
  Open your terminal/command prompt and run gcloud auth login. Follow the instructions to log in to your Google Cloud account.

- Set the active project:
  Run gcloud config set project <your-project-id> to set the active project, replacing <your-project-id> with your GCP project ID.

- Deploy the function:
  Deploy your function using the gcloud command. Replace <your-function-name> with a name for your function:
  Example: gcloud functions deploy <your-function-name> --runtime python310 --trigger-http --allow-unauthenticated --entry-point hello_world

  In this command, we specify the runtime as Python 3.10, trigger as HTTP, allow unauthenticated requests, and set the entry point to the hello_world function.

- Test the function:
  After the deployment is successful, you'll see the URL for your Cloud Function in the terminal output. Access the URL in your browser or with a tool like curl to test your function.

  That's it! You've deployed a Python program as a Google Cloud Function. Remember to check the official documentation for more details and options.