Data Maturity Scan | Complete & Refine Solution

Naam: Mohamed Ajimi
Datum: 18/03/2024

1. Inhoud

2. Abstract 4
3. Introduction: 5
4. Objectives 5
5. Project Plan: 6
  5.1. Google Form Submission: 8
  5.2. Apps Script - Trigger Configuration: 9
  5.3. Google Cloud Integration: 13
    1. Pipeline Overview: 13
    2. Google Cloud technologies: 15
  5.4. Data Pipeline: 16
    1. Foundation Layer 16
    2. Conformed Layer: 19
    3. Presentation Layer 22
6. GDPR Compliance and Data Retention 25
7. Email Notification System 27
  7.1. Design and Workflow of the Notification System 27
    1. System Workflow 28
    2. Implementation of the Notification System 28
    Email Notification Infrastructure 28
    Google Cloud Functions 28
    3. Google Pub/Sub 31
    4. Google Apps Script 31
  7.2. Result: 35
8. The Looker Dashboard 35
  8.1. Customer Dashboard 35
    1. Overall view 35
    2. Interactive categorical view 36
  8.2. Admin Dashboard 37
    1. Admin overview / Category 38
    2. Admin – Info 38
9. Conclusion: 40

2. Abstract

The "Data Maturity" project pioneers an automated framework to evaluate and enhance organizational data management practices. Leveraging a comprehensive digital questionnaire, the project systematically assesses key dimensions of data management: journey, empowerment, culture, activation, and governance. This assessment is automated through a Google Cloud pipeline, facilitating real-time processing and visualization of data on a user-friendly dashboard. Our methodology innovates by employing self-service tools for data collection and advanced analytics for insight generation, significantly streamlining the evaluation process. Preliminary application demonstrates the framework's capacity to offer immediate, actionable insights, encouraging rapid improvements in data management strategies and fostering a culture of data-driven decision-making.

3. Introduction:

The Challenge of Data Management in Modern Organizations

In the digital age, effective data management is foundational to business success. Organizations are inundated with data that, if properly harnessed, has the potential to drive innovation, streamline operations, and foster competitive advantages. However, the complexity of data landscapes and rapid technological advancements pose significant challenges. Traditional methods of assessing data management practices are often labour-intensive, subjective, and struggle to keep pace with the demands of a data-driven business environment.

Introducing the "Data Maturity" Project

Recognizing these challenges, the "Data Maturity" project was developed to provide a systematic, automated solution to assess and improve an organization's data management capabilities. This initiative distinguishes itself by transforming the way businesses approach data management evaluation - moving from manual, episodic assessments to a dynamic, continuous improvement model.

Methodological Innovation and Real-time Insight Generation

At the heart of the project is a meticulously designed questionnaire that probes critical aspects of data management. The integration of this questionnaire with a Google Cloud pipeline exemplifies the project's innovative approach, automating data processing and analysis to deliver real-time insights. This capability not only reduces the resource burden traditionally associated with data management assessments but also provides organizations with the agility to respond to insights rapidly.

Towards a Data-empowered Future

The "Data Maturity" project's goal is to empower organizations to harness their data's full potential. By providing a clear, actionable roadmap for improvement, the project lays the groundwork for building robust data governance frameworks, fostering a culture of data-driven decision making, and achieving strategic business objectives. This paper will explore the project's development, from conceptualization to implementation, and discuss its broader implications for the future of organizational data management.

4. Objectives

Central to achieving this vision is the development and optimization of a data engineering pipeline that seamlessly integrates data collection, processing, and visualization. This pipeline is foundational to the project, facilitating the immediate transformation of raw questionnaire responses into actionable insights. The following objectives are designed to ensure the pipeline not only functions efficiently but also maximize value to the participating organizations:

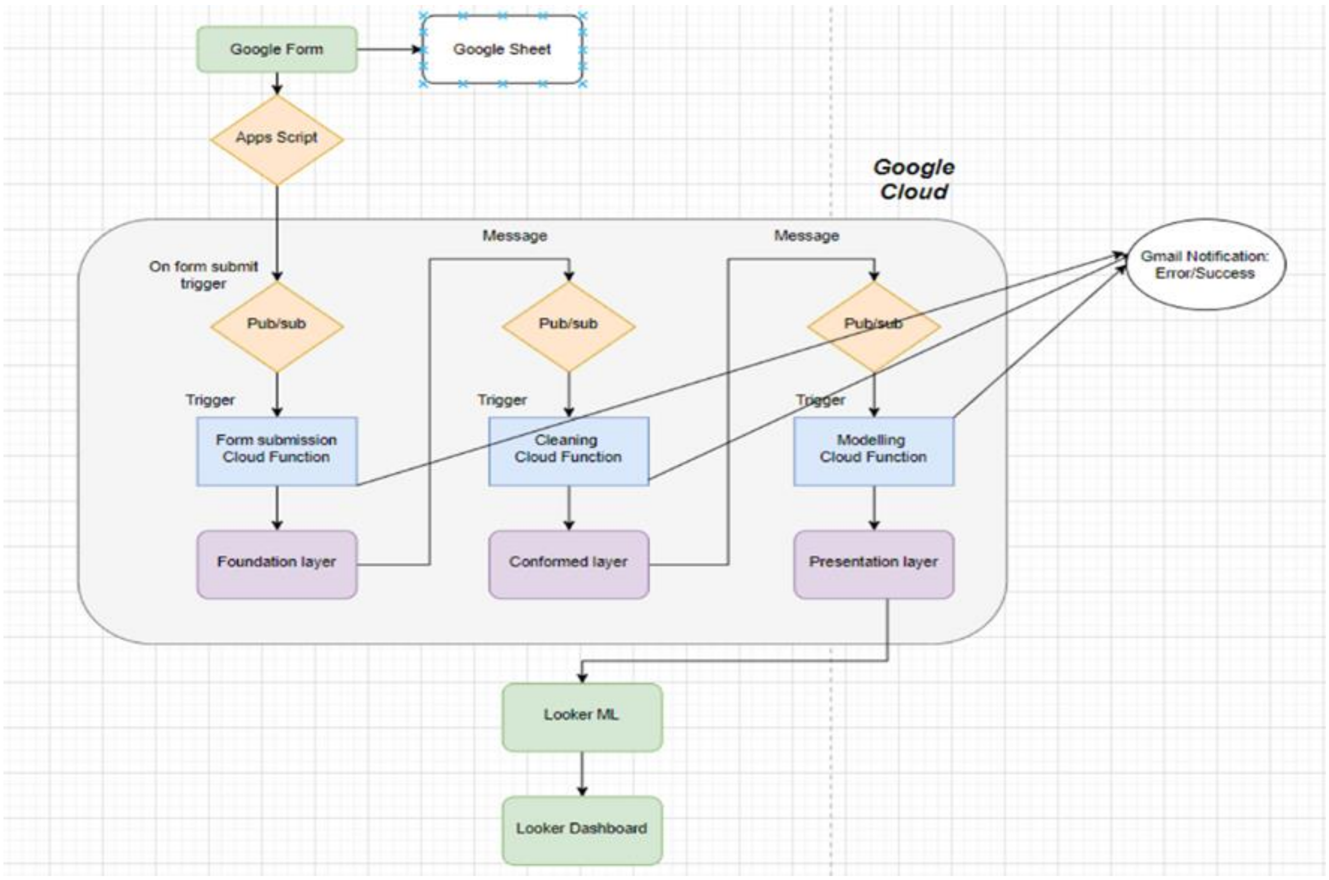
Real-time Data Processing:
Establish a system where questionnaire responses submitted via Google Forms are instantly processed. This includes setting up triggers in Google Cloud to initiate the data pipeline as soon as responses are received.

Data Cleaning and Preparation:
Automate the cleaning of incoming data to remove inconsistencies and format the data correctly for analysis. This involves setting rules for handling missing values, standardizing response formats, and ensuring data uniformity.

Dynamic Dashboard Creation:
Utilize Looker to develop dynamic dashboards that update in real time as new responses are submitted. These dashboards will visually represent the company's data maturity levels across different dimensions, highlighting areas of strength and those requiring improvement.

User Experience Optimization:
Ensure that the questionnaire is user-friendly, and that the dashboard interface is intuitive.

5. Project Plan:



Google Form Submission:

- Respondents submit their responses through a Google Form.
- The form is linked to a Google Sheet that records all responses.

Apps Script - Trigger Configuration:

- An Apps Script is set up to run whenever the Google Form is submitted.

- This script triggers a Google Cloud Pub/Sub message, signifying a new form submission.

Google Cloud Integration:

The Pub/Sub message is the start of the Google Cloud-based data processing pipeline. The message triggers a Cloud Function dedicated to handling form submissions.

- Data Processing Pipeline - Foundation Layer:
  - The Form Submission Cloud Function processes the initial raw data.
  - This layer is responsible for preparing the data for the cleaning process.
- Data Cleaning:
  - A second Pub/Sub message triggers another Cloud Function focused on data cleaning.
  - The Cleaning Cloud Function automates the data cleaning, handling missing values, standardizing formats, and ensuring data consistency.
  - Once cleaned, the data is moved to the Conformed Layer according to a Snowflake Model.
- Data Modelling and Analysis:
  - A third Pub/Sub message is used to trigger the Modelling Cloud Function.
  - This function applies predefined calculations to the cleaned data to derive insights.
  - The processed data is then pushed to the Presentation Layer according to a Star Schema Model.
- Error Handling and Notifications:
  - If any of the Cloud Functions encounter an error, an error notification is sent via Gmail to the administrators.
  - Upon successful completion of the data modelling, a success notification is also sent out.

Looker Integration:

- Looker ML connects to the Presentation Layer.
- It further processes the data.

Dashboard Creation:

- Looker generates dynamic, real-time dashboards based on the processed data.
- The dashboards are designed to be user-friendly and provide visual insights into the company's data maturity levels across various dimensions.

5.1. Google Form Submission:

The data collection phase of the "Data Maturity" project begins with participants submitting their responses through a Google Form. This method is chosen for its straightforward and professional interface, which allows for quick and organized data collection. The responses are automatically recorded in a connected Google Sheet, ensuring that the data is captured accurately and immediately.

5.2. Apps Script - Trigger Configuration:

Overview:

To ensure seamless real-time data processing, an Apps Script is configured to execute upon every submission of the Google Form. This script plays a pivotal role in automating the transition of data to the Google Cloud pipeline, marking the beginning of a data handling process that culminates in actionable insights displayed on dynamic dashboards.

Rationale:

The decision to use Apps Script is based on its robust integration with Google Forms and Google Cloud services, allowing for a straightforward yet powerful method to trigger downstream processes. Furthermore, the script's ability to manipulate and prepare the data for Pub/Sub messaging ensures that data integrity is maintained while catering to the specific requirements of subsequent processing stages.

Implementation:

The Apps Script is designed with two primary objectives: capturing form responses and preparing this data for Pub/Sub transmission. Here is a closer look at the code and the reasoning behind each decision.

Code Snippet: Form Response Capture and Preparation

```
function onFormSubmit(e) {
  // Get the form object
  var form = FormApp.getActiveForm();
  var formItems = form.getItems();

  var responseItems = e.response.getItemResponses();
  var responseMap = {};

  // Create a map of responses keyed by item ID for easier lookup
  responseItems.forEach(function(itemResponse) {
    var itemId = itemResponse.getItem().getId();
    responseMap[itemId] = itemResponse.getResponse();
  });

  var allResponses = formItems.map(function(formItem) {
    var itemId = formItem.getId();
    var response = responseMap[itemId];
    var formattedResponse;

    // Determine the question type
    var questionType = formItem.getType().toString();

    // Process only specific types and exclude others like SECTION_HEADER,
IMAGE, etc.
    if (["TEXT", "MULTIPLE_CHOICE", "CHECKBOX", "SCALE"].indexOf(questionType)
=== -1) {
      return null; // Skip types you're not interested in
    }

    // Handle multiple responses for checkboxes
    if (Array.isArray(response)) {
      formattedResponse = JSON.stringify(response);
    } else {
      formattedResponse = response;
    }

    return {
      questionTitle: formItem.getTitle().replace(/\?/g, "'"),
      questionType: questionType,
      actualResponse: formattedResponse || "", // Use empty string for
unanswered questions


      brandDescr: "Solifaction"
    };
  }).filter(function(response) {
    return response !== null; // Filter out nulls from skipped types
  });

  if (allResponses.length > 0) {
    // Convert allResponses to a string and encode it for Pub/Sub
    var messageData =
Utilities.base64EncodeWebSafe(JSON.stringify(allResponses));

    // Send the detailed responses to Pub/Sub
    sendToPubSub(messageData);
  } else {
    console.log("No valid questions to send.");
  }
}
```

Explanation:

1. **Collect Responses**: It gathers the submitted responses and maps them by the form item ID, creating an easy reference system.
2. **Format and Filter**: It processes relevant form items (text, multiple choice, checkboxes, scale questions) and excludes non-actionable types (e.g., headers, images). Checkbox answers are converted into a JSON string for uniformity. This is done because the responses are naturally separated by commas, which could lead to confusion if any of the answers themselves contain commas.
3. **Prepare for Transmission:** It packages the formatted responses into an encoded message (base64) suitable for Google Cloud Pub/Sub, ensuring data integrity and security during transfer.
4. **Send to Pub/Sub**: The script sends the prepared data to Pub/Sub if there are valid responses. If there are no relevant responses to process, it records a log message.

Code Snippet: Data Transmission to Pub/Sub

```
function sendToPubSub(messageData) {
    var service = getOAuthService(); // Make sure this is properly implemented
    var pubsubUrl = 'https://pubsub.googleapis.com/v1/projects/datamaturity-
415814/topics/formSubmissionTopic:publish';
```

```
    if (service.hasAccess()) {
      var payload = {
        "messages": [
          { "data": messageData }
        ]
      };

      var options = {
        "method": "post",
        "contentType": "application/json",
        "headers": {
          "Authorization": "Bearer " + service.getAccessToken()
        },
        "payload": JSON.stringify(payload)
      };

      try {
        UrlFetchApp.fetch(pubsubUrl, options);
        Logger.log('Static message sent to Pub/Sub successfully.');
      } catch (error) {
        Logger.log('Error sending static message to Pub/Sub: ' +
error.toString());
      }
    } else {
      Logger.log('No access to the service. Please re-authorize.');
    }
  }
```

<u>Explanation:</u>

This function takes the prepared message data and sends it to a specific Google Cloud Pub/Sub topic for further processing.

1. **OAuth Service:** It initializes an OAuth service to handle authentication, ensuring secure API calls to Pub/Sub.

2. **Prepare Message:** The function wraps the encoded message data in a structure expected by Pub/Sub, readying it for transmission.
3. **API Call:** It then makes a POST request to the Pub/Sub API endpoint with the prepared message. The request includes necessary headers, like the authorization token obtained from the OAuth service.
4. **Logging:** Upon attempting to send the message, the function logs the success or failure of this operation. This is crucial for monitoring the system's health and debugging if needed.
5. **Access Verification:** If the function cannot access the OAuth service, it logs an error message prompting for re-authorization, ensuring that permission issues are flagged immediately.

Conclusion:

The Apps Script trigger is a key step in automating data flow from Google Forms to actionable insights through Google Cloud Pub/Sub, streamlining the data processing pipeline.

This sets the stage for detailed examination in subsequent documentation, which will cover the Google Cloud pipeline stages and Looker dashboard integration.

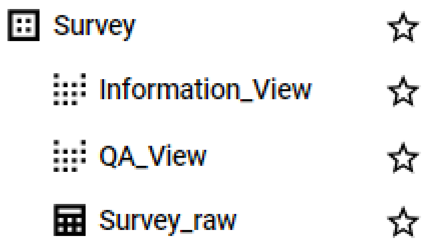
5.3. Google Cloud Integration:

1. Pipeline Overview:

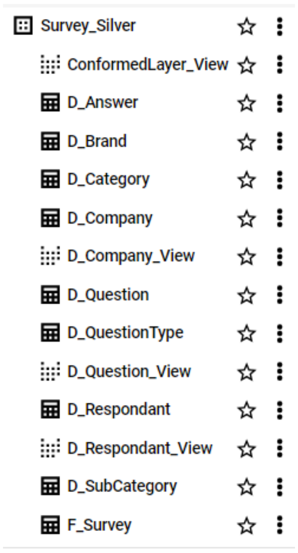
The Google Cloud Integration within the "Data Maturity" project adheres to the Medallion architecture, designed to handle data through three sequential and critical stages: the Foundation, Conformed, and Presentation layers. This structured approach ensures a disciplined and clear progression of data from its raw state to actionable insights.

- **Foundation Layer:** The journey begins when a Google Form submission triggers a Pub/Sub message, which activates the initial Cloud Function. This function represents the Foundation

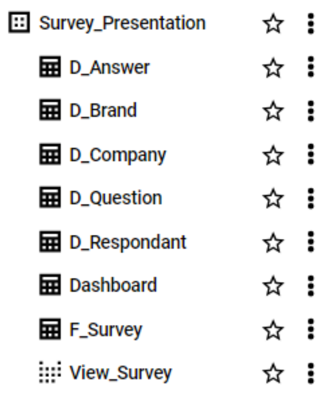
Layer, where raw data from form submissions is captured and preliminarily formatted. It is the raw staging area that prepares data for the subsequent cleansing process.



- **Conformed Layer:** Upon successful completion of the foundational processing, the first Cloud Function triggers the next one in the sequence, which operates within the Conformed Layer. Here, another Cloud Function takes over to perform comprehensive data cleaning and transformation. It ensures that the raw data is not only cleaned of inconsistencies but is also transformed according to the defined schema, employing a Snowflake model to facilitate complex queries and analytics.



- **Presentation Layer:** The clean and structured data is then ready for the third stage, triggered by another Pub/Sub message. The Modelling Cloud Function, representing the Presentation Layer, applies business logic and predefined calculations to the conformed data to derive insights. It organizes the data into a Star Schema, which is ideal for reporting and analysis, feeding directly into the visualization tools for end-user consumption.



Survey_Presentation
- D_Answer
- D_Brand
- D_Company
- D_Question
- D_Respondant
- Dashboard
- F_Survey
- View_Survey

The sequential triggering of Cloud Functions forms an integral part of the real-time, streamlined process. Each function is set to automatically trigger the next upon successful execution, ensuring a continuous, uninterrupted flow of data through the pipeline. This seamless automation enables the system to process data in real time and in sequence, with minimal latency and maximum efficiency.

This real-time, sequential processing is not only efficient but also minimizes the potential for bottlenecks, as each stage of the architecture is optimized to handle the specific tasks required at that point. It's a cascading workflow that ensures the integrity and availability of data at each step, culminating in a dynamic dashboard that provides real-time insights into the organization's data maturity.

2. Google Cloud technologies:

The Google Cloud technologies selected for the "Data Maturity" project are pivotal for creating an efficient and scalable data processing pipeline. Here's an explanation of the chosen technologies and their roles:

Pub/Sub:

→ - **Role:** Google Cloud Pub/Sub serves as the messaging middleware that enables event-driven and asynchronous communication between the different components of the pipeline. It acts as a highly scalable and flexible message queue that decouples services that produce events (like form submissions) from services that process events (such as Cloud Functions).

→ - **Why Chosen:** Pub/Sub was selected for its ability to provide real-time messaging. It ensures that data can be ingested from various sources and is immediately available for processing. Its robust delivery guarantees and ability to handle spikes in data make it ideal for dynamic data workflows, where the volume and frequency of data can vary greatly.

→ **- Cost:** The cost is free for up to 10GB of traffic per month, and above this, it's $40 per TB per month.

Cloud Functions:

→ **- Role:** Cloud Functions are stateless, auto-scaling, event-driven compute solutions that execute in response to Pub/Sub messages. In the context of this project, they process the data at each stage: initial form submission handling, data cleaning and transformation, and data modeling.

→ **- Why Chosen:** The choice of Cloud Functions is based on their ability to run backend code in response to HTTP requests or Pub/Sub events without server management. They provide a cost-effective solution that scales automatically with the number of requests and allows for a separation of concerns, with each function responsible for a discrete piece of the pipeline.

→ **- Cost:** Google Cloud Functions are priced based on the number of invocations, execution time, and the resources allocated to the function. Pricing includes a free tier, which provides a generous number of free invocations and execution time per month

  - 2 million invocations per month.
  - 400,000 GB-seconds of compute time per month.
  - 200,000 GHz-seconds of compute time per month
  - 5 GB of Internet egress traffic per month

Together, these technologies create a robust and scalable architecture that can handle the complexities of real-time data processing and analysis. They allow for an infrastructure that responds rapidly to new data, processes this data efficiently, and makes it available for querying and visualization, thus aligning perfectly with the real-time, data-driven objectives of the "Data Maturity" project.

5.4. Data Pipeline:

1. Foundation Layer

Purpose:
The Foundation Layer serves as the entry point for data within the "Data Maturity" project pipeline. Its primary role is to ingest raw data from Google Form submissions and prepare it for the subsequent stages of processing.

Workflow:

→ **Data Ingestion:** A Google Form submission triggers the publishing of a Pub/Sub message, signaling the availability of new data for the processing pipeline.

→ **Function Trigger:** The Pub/Sub message automatically triggers the execution of the designated Cloud Function, initiating the Foundation Layer's processing.

→ **Data Capture and Preliminary Formatting:**
-The Cloud Function decodes the message data and performs JSON parsing to structure the form responses.
- It applies a filter to isolate the responses to relevant question types: 'TEXT', 'MULTIPLE_CHOICE', 'CHECKBOX', and 'SCALE'.
- Metadata, such as submission timestamps and identifiers, are extracted to ensure comprehensive tracking through the pipeline.

→ **Response Transformation:**
- Responses are checked for list types indicative of checkbox selections and converted to a comma-separated string to accurately represent multiple answers.
- Textual responses are scanned for mid-sentence question marks and reformatted to maintain the integrity of the data.

→ **BigQuery Insertion:**
- A unique submission identifier is generated, and each response is compiled into a row formatted for BigQuery's schema.
- The Cloud Function then inserts these rows into the BigQuery 'Survey_raw' table, capturing the raw data for the initial layer of processing.

→ **Error Handling and Notification:** Any errors encountered during the process are logged to a BigQuery 'Error_Log' table and, if critical, are also published to an error notification Pub/Sub topic.

→ **Workflow Progression:**
- Following successful data insertion and logging, the Cloud Function publishes the submission identifier to the 'cleaning_trigger' Pub/Sub topic.
- This action cues the next Cloud Function in the pipeline, responsible for data cleaning and validation, to begin its process.

Components:

→ **Google Form:** Acts as the user interface for data collection.

→ **Google Cloud Pub/Sub:** Serves as the messaging queue that decouples data production from consumption.

→ **Cloud Function (Form Submission Handler):** The serverless component that executes in response to Pub/Sub messages and handles the raw data.

Challenges Addressed:

→ **Scalability:** Designed to scale automatically with the influx of data submissions, handling spikes and drops in form activity without manual intervention.

→ **Reliability:** Provides a reliable entry point for data that ensures no data loss occurs between the form submission and the next processing stage.

Output:

Primary Output:

The foundational processing layer's output is twofold.

- Firstly, it ensures that incoming data submissions are validated, formatted, and stored efficiently in a BigQuery table named `Survey_raw`. This dataset contains a comprehensive collection of the survey responses, prepared for the next stage of processing.
- Secondly, the processed data is also used to generate signals (via Pub/Sub messages) that trigger subsequent layers of the pipeline, specifically the Conformed Layer's Cloud Function, ensuring a seamless workflow.

BigQuery Data Views:

In addition to the primary storage and signaling functions, this layer facilitates further data analysis by creating two specific BigQuery views based on the `Survey_raw` table:

1. QA_VIEW (QUESTIONS AND ANSWERS):
   - This view organizes the survey data to emphasize the relationship between questions and their respective answers. It corrects question descriptions where necessary, categorizes question types for clarity, and structures responses appropriately. The view is particularly useful for analytical queries focusing on the survey content, enabling easy access to question-specific responses and metadata.
2. INFORMATION_VIEW (USER PERSONAL INFORMATION):
   - This view extracts and consolidates personal information submitted by survey participants. It organizes this data by submission ID, providing a clear summary of each respondent's personal details, including contact information and organizational affiliation. This view supports operations requiring direct engagement with participants or analyses centered around demographic or organizational characteristics.

Next Steps:
Upon successful processing, the Foundation Layer automatically triggers the next stage in the pipeline, initiating the Conformed Layer for further data cleaning and transformation.

2. Conformed Layer:

Purpose:
The Conformed Layer refines raw data ingested by the Foundation Layer, applying rigorous cleaning and transformation rules. Its objective is to standardize and enrich the data, making it consistent and ready for analytical processing and insight generation.

Workflow:

→ **Trigger:**
The workflow commences upon receiving a Pub/Sub message which is the **submission_id** . This message, triggered by the successful processing in the Foundation Layer, signifies that new data is ready for further refinement.

→ **Data Retrieval and Pre-Processing:**
- Information and QA Data Fetching: Utilizes two specialized BigQuery Views (`Information_View` and `QA_View`) to retrieve detailed respondent information and question-answer data corresponding to the submission_id. These views simplify the raw data, making it more accessible for the cleaning and transformation processes.
- Company and Respondent Identification: Checks for existing entries in the dataset for companies and respondents based on the retrieved information. If no match is found, it generates new unique identifiers for them, ensuring all data points are accurately linked and traceable.

→ **Data Cleaning and Transformation:**
- Question and Answer ID Mapping: For each question-answer pair from the `QA_View`, it maps text descriptions to their respective IDs using a predefined dictionary. This step is crucial for standardizing data representation and facilitating its analysis.
- Survey Data Compilation: Constructs a comprehensive dataset that includes survey IDs, company and respondent identifiers, question and answer IDs, and survey metadata. This dataset is prepared for insertion into a structured BigQuery table tailored for analytical processing.

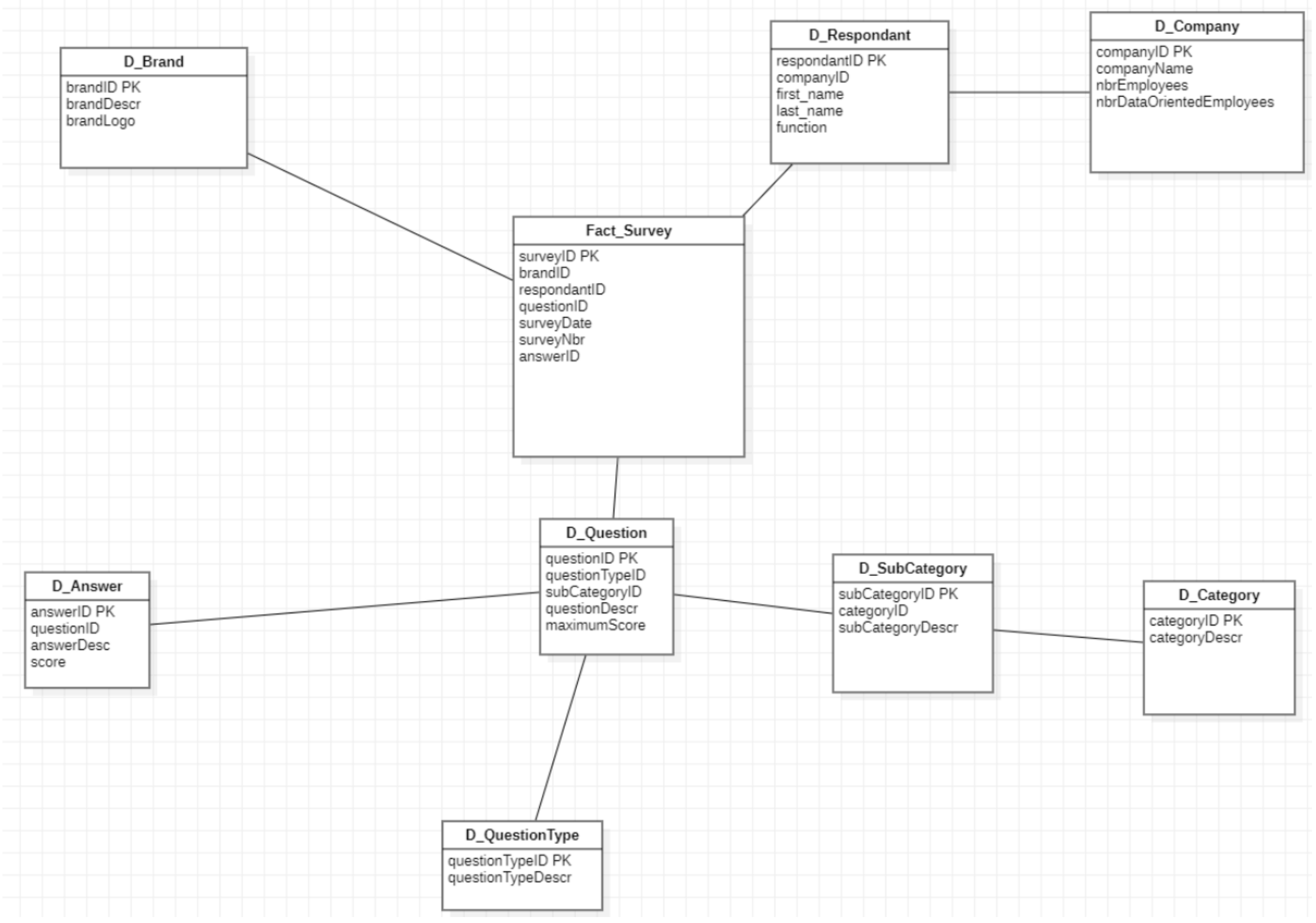
→ **Validation and Quality Assurance:**
- Data Integrity Checks: Performs validation on the transformed data to ensure it adheres to the required quality and completeness standards. This includes verifying the presence of all necessary identifiers and the accuracy of mappings.
- Error Detection and Notification: In cases of invalid question-answer mappings or missing data, the function logs detailed error messages and, if necessary, sends critical alerts through Pub/Sub.

→ **Workflow Continuation and Data Publishing:**
Upon successful cleaning, transformation, and validation, the Cloud Function publishes a new message to a Pub/Sub topic which is the surveyNbr (submission_id). This message indicates the readiness of the cleaned and conformed data for the next stage, driving the workflow into the Presentation Layer for advanced analytics and reporting.

Components:

→ **Pub/Sub:** Listens for messages signaling data readiness from the Foundation Layer.
→ **BigQuery:** Serves both as the source for raw data retrieval and the destination for the cleaned and conformed dataset.
→ **Cloud Functions:** Hosts the logic for data cleaning, transformation, and validation tasks.

Snowflake Schema Model:



Challenges Addressed:

→ **Data Quality:** Tackles inconsistencies, errors, and gaps in the raw data to improve overall data quality.

→ **Data Consistency:** Ensures uniformity across datasets, crucial for accurate analysis and reporting.

→ **Scalability:** Manages variable data volumes efficiently, maintaining performance as the data grows.

Output:

Upon completing the data cleaning and transformation processes, the Conformed Layer generates a cleaned consistent, and well-structured dataset. This data set, stored within BigQuery, is fully prepared for in-depth analysis and further modeling in the subsequent layers of the data processing pipeline. To augment the accessibility and utility of this dataset, especially in transitioning the data to the Presentation Layer, the following BigQuery Views are meticulously created and maintained:

D_COMPANY_VIEW: It encapsulates detailed information about companies identified or created during the Conformed Layer's processing. It includes company names, identifiers, and relevant metadata, facilitating a smoother linkage of company-specific data across the dataset.

D_RESPONDANT_VIEW: Similar to `D_Company_View`, this view focuses on respondent data, aggregating personal information. It ensures that individual respondent data is easily queryable, supporting analyses that require respondent-specific insights.

CONFORMEDLAYER_VIEW: As an overarching view, `ConformedLayer_View` synthesizes elements from the preceding views to present a comprehensive snapshot of the processed data. It is designed to streamline the transition of data to the Presentation Layer, ensuring that all necessary information is readily available for the final stages of analysis and reporting.

These views not only simplify the task of querying and analyzing the dataset but also play a pivotal role in ensuring a seamless flow of data towards the Presentation Layer. By providing structured access to both granular and aggregated data, they enable more sophisticated analyses and facilitate the generation of actionable insights in the later stages of the data processing pipeline.

Next Steps:
The cleaned and structured data, along with the specialized views, are now primed for the Presentation Layer. Here, data modeling and analysis are performed, leveraging the cleaned dataset to generate insights, support decision-making, and power dynamic dashboards and reports.

3. Presentation Layer

Purpose:
The Presentation Layer is designed to finalize the data preparation process by applying business logic and sophisticated calculations to the structured data obtained from the Conformed Layer. Its primary objective is to distill actionable insights from the data, making it readily consumable for decision-making and strategic analysis.

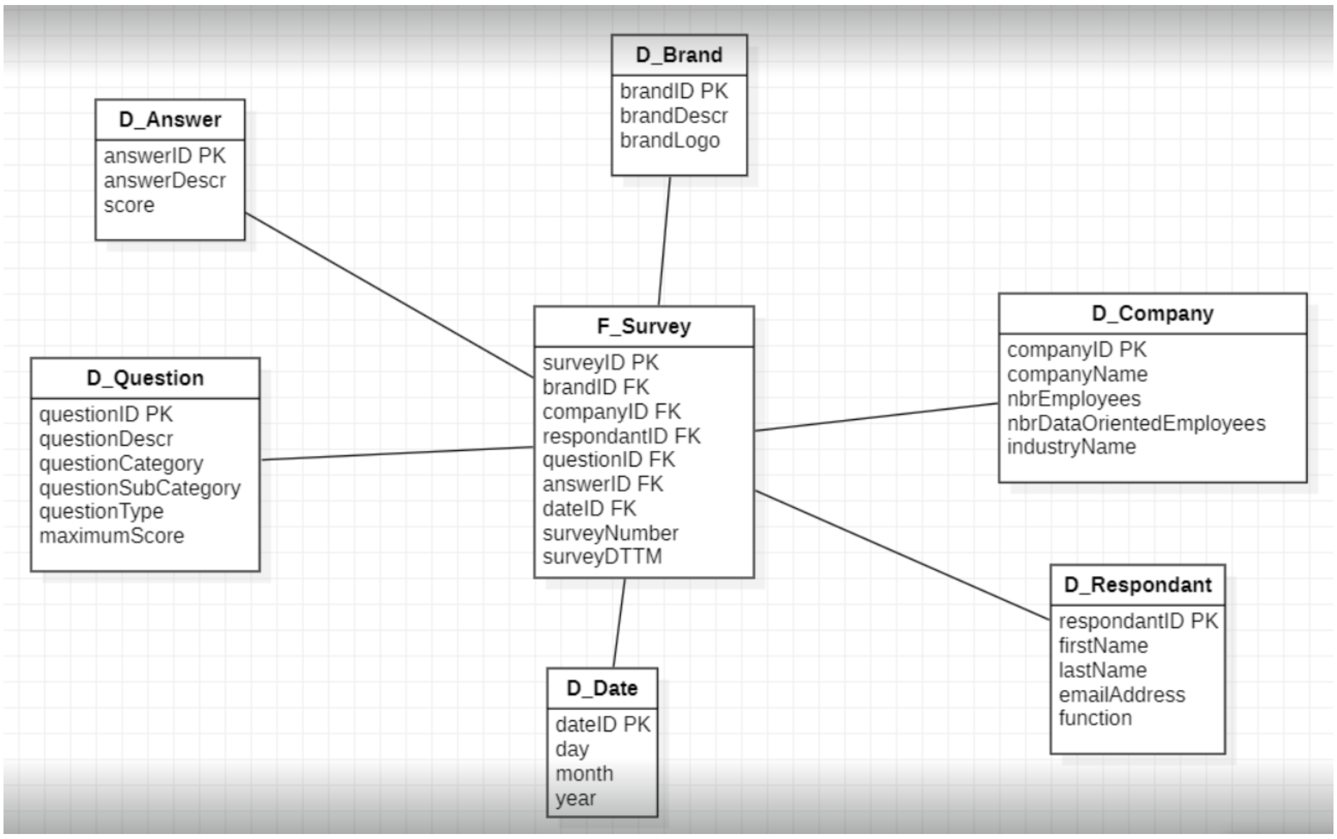
Workflow:

- → **Trigger Activation:** Begins with a Pub/Sub message **surveyNbr**, indicating that the data from the Conformed Layer is processed and ready for further analysis.
- → Data Modeling: Utilizes a Cloud Function to apply business logic. This stage transforms the data into a format that is optimized for analysis, adhering to a Star Schema for enhanced querying efficiency.
- → **Insight Derivation:** Focuses on extracting meaningful insights from the modeled data, identifying trends, patterns, and opportunities that can inform business strategies.
- → **Data Organization**: Structures the data into a Star Schema, strategically organizing it into fact and dimension tables. This schema facilitates efficient querying and analysis, serving as a foundation for the generation of reports and dashboards.

Components:

- → **Cloud Functions:** Execute the business logic and calculations on the data, transforming it into an analytics-ready state.
- → **Pub/Sub:** Signals the readiness of data for the Presentation Layer and can also be used to trigger subsequent actions based on the analysis outcomes.
- → **BigQuery:** Hosts the data organized in a Star Schema, providing a powerful and scalable platform for running complex queries and analysis.

Star Schema Model:



Challenges Addressed:

→ **Complex Data Transformation:** Automates the application of complex business logic and calculations, streamlining the process of transforming raw data into actionable insights.

→ **MaximumScores of Checkboxes:**
Context:
In surveys with checkbox questions, each selected option is typically stored as a separate row in the database, leading to multiple rows for a single question, each with the same maximum score. This redundancy can complicate calculations, such as computing the total score for a respondent, because simply summing the maximum scores of all rows would inaccurately inflate the total possible score for questions with multiple answers.

The Solution:
- The SQL snippet employs a window function (`ROW_NUMBER() OVER (...)`) in combination with a `CASE` statement to ensure that the maximum score for a given question (with multiple answers due to checkboxes) is counted only once in calculations, while still maintaining the granularity of having each answer on a separate row.

- `ROW_NUMBER() OVER (PARTITION BY surveyNbr, questionID ORDER BY answerID)`: This part assigns a unique row number to each answer for a given question within a specific survey, ordered by the answer ID. Each question's first answer (per survey) is assigned the number 1, with subsequent answers receiving increasing numbers.

- `CASE` Statement: Determines the value of `actualMaximumScore` based on the row number assigned in the previous step.

 - If the row number is 1 (meaning it is the first answer for this question within the survey), the `actualMaximumScore` for this row is set to the question's maximum score. This ensures that the maximum score for the question is counted once.

 - For all subsequent answers to the same question (row numbers greater than 1), the `actualMaximumScore` is set to 0, effectively excluding these from the total maximum score calculation.

Implications:
This approach allows for accurate scoring by ensuring that the maximum possible score for checkbox questions is not overstated in analyses. It enables detailed analytics on a per-answer basis while correctly handling the aggregate scoring for questions with multiple selectable options. This method is particularly useful when calculating metrics like average scores or percentages, where the denominator (the total maximum score) needs to be accurate to yield meaningful results.

→ **Efficient Data Querying:** The use of a Star Schema significantly improves the efficiency of data querying, enabling faster and more flexible analyses.

→ **Scalable Insight Generation:** Ensures that the infrastructure supports scalable analyses, capable of accommodating growing data volumes and complexity without compromising performance.

Output:
The final output of the Presentation Layer includes a comprehensively modeled dataset that is strategically organized for easy analysis. This dataset is then used to power reporting and visualization tools.

6. GDPR Compliance and Data Retention

Overview

In adherence to the General Data Protection Regulation (GDPR), our project includes stringent measures for data retention and the privacy of personal information. To comply with these regulations, we have implemented a series of scheduled queries within our Google Cloud Platform

(GCP) infrastructure that automatically remove personal data that has exceeded the retention period of one year. This ensures that we do not retain personal data beyond the duration necessary for the purposes for which the data was collected.

Scheduled Queries for Data Deletion

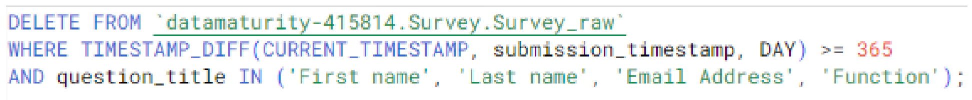
We have established three scheduled queries that run at predefined intervals, targeting data stored in different layers of our pipeline:

- **gdpr_foundation_query:** Cleanses the `Survey_raw` table in the Foundation Layer, removing personal identifiers like 'First name', 'Last name', 'Email Address', and 'Function' that are older than 365 days.
- **gdpr_conformed_query:** Purges entries from the `Survey_Silver.D_Respondant` table in the Conformed Layer that are older than 365 days.
- **gdpr_presentation_query:** Deletes data from the `Survey_Presentation.D_Respondant` table in the Presentation Layer that has surpassed the one-year mark.

Deletion Logic

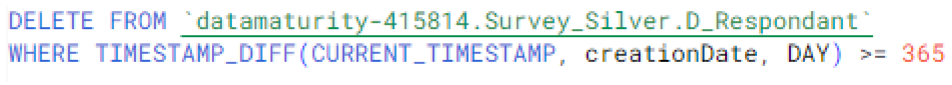
Each query utilizes the `TIMESTAMP_DIFF` function to calculate the difference between the current timestamp and the data creation or submission timestamp. If this difference equals or exceeds 365 days, the corresponding records are deleted. Here are the details of the queries executed:

- Foundation Layer:



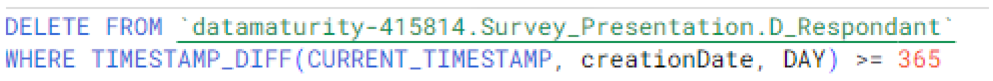
```
DELETE FROM `datamaturity-415814.Survey.Survey_raw`
WHERE TIMESTAMP_DIFF(CURRENT_TIMESTAMP, submission_timestamp, DAY) >= 365
AND question_title IN ('First name', 'Last name', 'Email Address', 'Function');
```

- Conformed Layer:



```
DELETE FROM `datamaturity-415814.Survey_Silver.D_Respondant`
WHERE TIMESTAMP_DIFF(CURRENT_TIMESTAMP, creationDate, DAY) >= 365
```

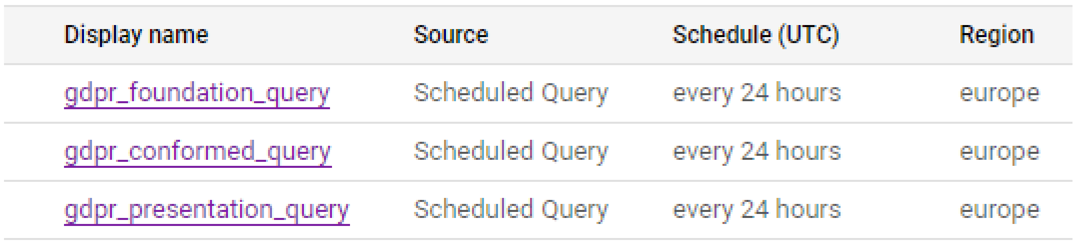
- Presentation Layer:



```sql
DELETE FROM `datamaturity-415814.Survey_Presentation.D_Respondant`
WHERE TIMESTAMP_DIFF(CURRENT_TIMESTAMP, creationDate, DAY) >= 365
```

Execution Schedule

The queries are scheduled to execute every 24 hours, ensuring continuous compliance with GDPR data retention policies. The following table summarizes the scheduled execution times for each query:



Display name	Source	Schedule (UTC)	Region
gdpr_foundation_query	Scheduled Query	every 24 hours	europe
gdpr_conformed_query	Scheduled Query	every 24 hours	europe
gdpr_presentation_query	Scheduled Query	every 24 hours	europe

7. Email Notification System

The notification system functions as a dual-state monitor, providing alerts for both error events and successful executions, thereby enabling a proactive approach to data management. Upon the detection of any anomalies or interruptions in the pipeline, it promptly dispatches detailed alerts to the responsible teams, enabling swift action to be taken. Conversely, a successful data throughput triggers a confirmation message, signaling operational continuity and data readiness.

These notifications are not merely informational; they are actionable. Error alerts are embedded with direct links to logs and tools, equipping the recipients with immediate access to diagnostic resources. Successful data processing messages confirm the availability of data for downstream consumption, ensuring that business continuity is maintained.

7.1. Design and Workflow of the Notification System

1. System Workflow

The workflow of the email notification system is a meticulously architected sequence of triggers and actions designed to capture and communicate the status of data processing through the pipeline.

1. **Event Trigger:** Each stage of the data pipeline is monitored for completion and errors. On the occurrence of either event, a message is published to the appropriate Google Pub/Sub topic.
2. **Message Relay:** The Pub/Sub service acts as a highly reliable intermediary that relays messages from the Cloud Functions to the Google Apps Script. It ensures message delivery even in high-throughput scenarios.
3. **Notification Trigger:** On receipt of a message from the Pub/Sub, the Google Apps Script is triggered. It parses the content to determine the nature of the event (error or success).
4. **Email Composition:** Depending on the event type, the script dynamically composes an email. The email for an error includes details such as the function name, submission ID, and steps for troubleshooting. For a successful event, it confirms the successful completion of data processing.
5. **Dispatch:** The composed email is dispatched to the predefined list of recipients, which typically includes system administrators, data engineers, and relevant project stakeholders.
6. **Logging:** Concurrently with the email dispatch, error events are logged in a BigQuery table specifically designated for error tracking, ensuring a permanent record for audit and analysis purposes.

2. Implementation of the Notification System

Email Notification Infrastructure

The notification system's infrastructure is built on Google Cloud Functions and Apps Script, orchestrated through Google Cloud Pub/Sub. This setup is designed to capture both error and success events in the data pipeline and translate these into email communications.

Google Cloud Functions

Trigger Mechanism:

Each Cloud Function is associated with a distinct part of the data pipeline and is triggered by certain conditions:

- **Error Events**: A Cloud Function is instrumented to detect processing errors. Upon encountering an error, the function captures the error context, which includes relevant details such as the error message, the function name, the time of the occurrence, and a unique identifier for the data being processed. This information is then packaged into a JSON object and published to a dedicated Pub/Sub topic for error notifications.

- **Success Events:** Similarly, when a Cloud Function completes its task successfully, it publishes a message to a separate Pub/Sub topic designated for success events. This message confirms the successful processing and includes details such as the timestamp and unique identifiers, which may be referenced in the success notification.

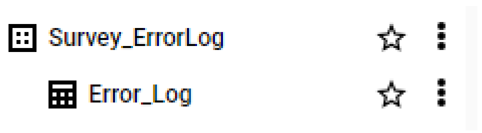
Error Handling and Notification:

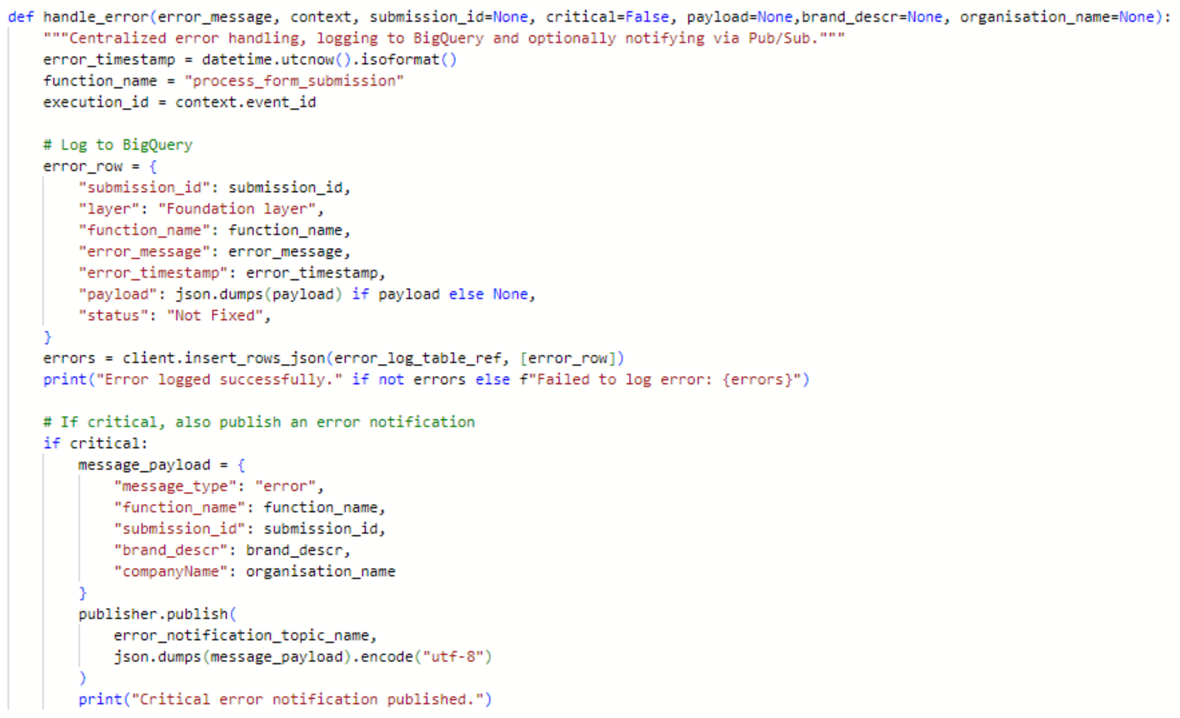
Error handling within Cloud Functions is critical to maintaining data integrity and operational continuity. Here's how it's addressed:

- **Try-Catch Blocks:** Each function includes try-catch blocks to gracefully handle exceptions and prevent unexpected termination of the function.

- **Error Publishing:** In the event of an error, the catch block constructs a message that encapsulates the error details and uses the Pub/Sub client library to publish it to the error notification topic.

- **Log Entries:** In addition to publishing to Pub/Sub, the error details are logged in the Cloud Function's logs and in a BigQuery Table for immediate access and later review.





```python
def handle_error(error_message, context, submission_id=None, critical=False, payload=None,brand_descr=None, organisation_name=None):
    """Centralized error handling, logging to BigQuery and optionally notifying via Pub/Sub."""
    error_timestamp = datetime.utcnow().isoformat()
    function_name = "process_form_submission"
    execution_id = context.event_id

    # Log to BigQuery
    error_row = {
        "submission_id": submission_id,
        "layer": "Foundation layer",
        "function_name": function_name,
        "error_message": error_message,
        "error_timestamp": error_timestamp,
        "payload": json.dumps(payload) if payload else None,
        "status": "Not Fixed",
    }
    errors = client.insert_rows_json(error_log_table_ref, [error_row])
    print("Error logged successfully." if not errors else f"Failed to log error: {errors}")

    # If critical, also publish an error notification
    if critical:
        message_payload = {
            "message_type": "error",
            "function_name": function_name,
            "submission_id": submission_id,
            "brand_descr": brand_descr,
            "companyName": organisation_name
        }
        publisher.publish(
            error_notification_topic_name,
            json.dumps(message_payload).encode("utf-8")
        )
        print("Critical error notification published.")
```

Success Notification Publishing:

Upon successful execution of the last cloud function "presentation-layer-processing", it will perform the following:

- **Message Construction:** A success message is created, containing metadata about the completed task.

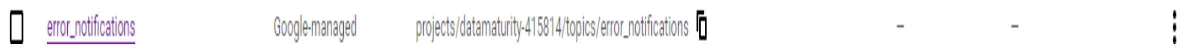
- **Success Publishing:** This message is then published to the success notification topic using the Pub/Sub client library.

```python
def publish_success_message(context, submission_id, brand_descr=None, company_name=None):
    function_name = "presentation-layer-processing"

    # Prepare the success message payload
    message_payload = {
        "message_type": "success",
        "function_name": function_name,
        "submission_id": submission_id,
        "brand_descr": brand_descr,
        "companyName": company_name
    }

    # Publish the success message to the Pub/Sub topic
    try:
        publisher.publish(notification_topic_name, json.dumps(message_payload).encode('utf-8'))
        print("Success message published.")
    except Exception as e:
        print(f"Failed to publish success message: {e}")
```

3. Google Pub/Sub

Pub/Sub topics are configured for both error and success scenarios:

- **The error notification topic** receives messages when a Cloud Function fails and upon the successful completion of the pipeline.



error_notifications   Google-managed   projects/datamaturity-415814/topics/error_notifications   -   -

4. Google Apps Script

Script Triggers and Execution:

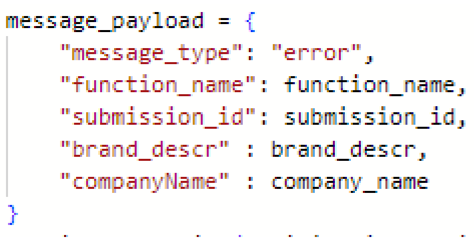
The Apps Script is triggered via HTTP POST requests sent by Google Cloud Functions. These requests are initiated whenever a Cloud Function publishes a message to a Pub/Sub notification topic, indicating either an error or a successful event. Upon receiving the message, the Apps Script executes predefined functions to send out the corresponding email notifications.

```javascript
// Conditionally send email notification based on message type
if (messageType === 'error') {
  sendEmailNotificationError(functionName, submissionId, brandDescr, companyName);
} else if (messageType === 'success') {
  sendEmailNotificationSuccess(functionName, submissionId, brandDescr, companyName);
} else {
  console.log('Unknown message type');
  // Handle unknown message type if necessary
}
```

Implementation Details:

1. Error Notification Flow:

The payload received from the pub/sub:



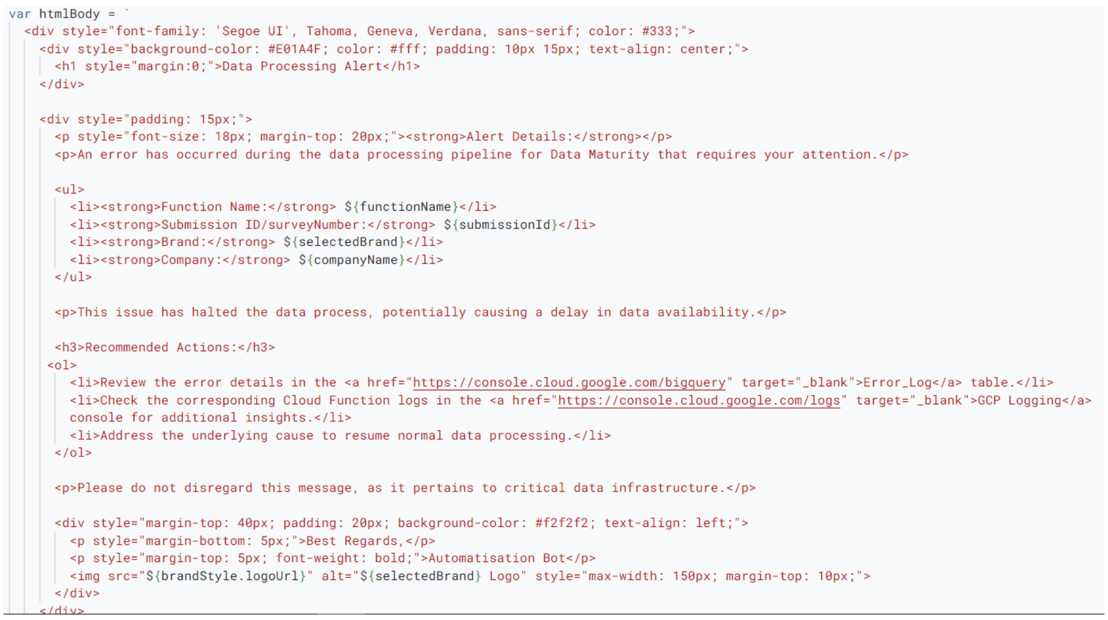
```
message_payload = {
    "message_type": "error",
    "function_name": function_name,
    "submission_id": submission_id,
    "brand_descr" : brand_descr,
    "companyName" : company_name
}
```

The `**sendEmailNotificationError**` function is designed to alert the responsible parties of any issues in the data processing pipeline:

- - Recipient: The `email` variable is defined with the administrator's or team's email address, ensuring that the right personnel are alerted.

- - Subject Line: The email subject explicitly states that there is a "Pipeline Error," which helps in quick identification and prioritization of the message.

- - Email Content: The body of the email is composed in HTML, providing a rich-text format that enhances readability and allows for brand-specific styling.



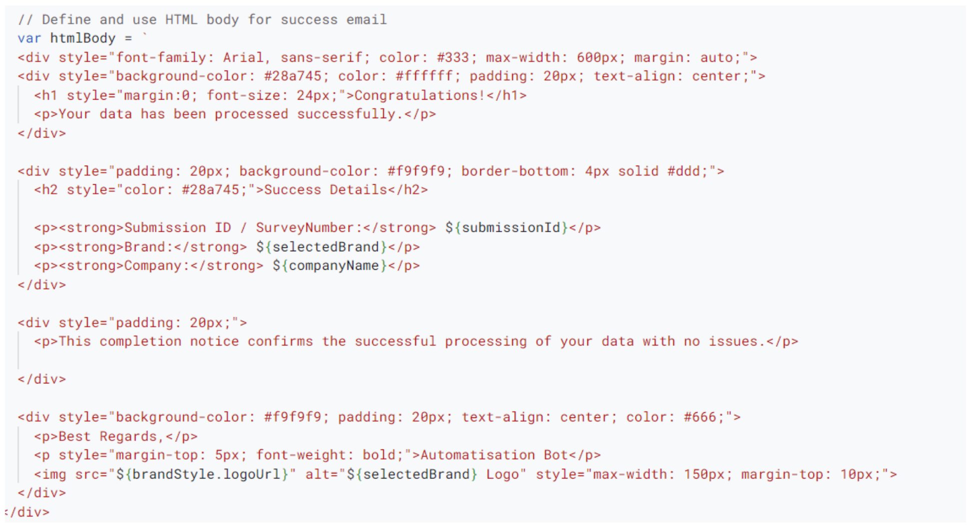
```
var htmlBody = `
  <div style="font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif; color: #333;">
    <div style="background-color: #E01A4F; color: #fff; padding: 10px 15px; text-align: center;">
      <h1 style="margin:0;">Data Processing Alert</h1>
    </div>

    <div style="padding: 15px;">
      <p style="font-size: 18px; margin-top: 20px;"><strong>Alert Details:</strong></p>
      <p>An error has occurred during the data processing pipeline for Data Maturity that requires your attention.</p>

      <ul>
        <li><strong>Function Name:</strong> ${functionName}</li>
        <li><strong>Submission ID/surveyNumber:</strong> ${submissionId}</li>
        <li><strong>Brand:</strong> ${selectedBrand}</li>
        <li><strong>Company:</strong> ${companyName}</li>
      </ul>

      <p>This issue has halted the data process, potentially causing a delay in data availability.</p>

      <h3>Recommended Actions:</h3>
     <ol>
        <li>Review the error details in the <a href="https://console.cloud.google.com/bigquery" target="_blank">Error_Log</a> table.</li>
        <li>Check the corresponding Cloud Function logs in the <a href="https://console.cloud.google.com/logs" target="_blank">GCP Logging</a>
        console for additional insights.</li>
        <li>Address the underlying cause to resume normal data processing.</li>
      </ol>

      <p>Please do not disregard this message, as it pertains to critical data infrastructure.</p>

      <div style="margin-top: 40px; padding: 20px; background-color: #f2f2f2; text-align: left;">
        <p style="margin-bottom: 5px;">Best Regards,</p>
        <p style="margin-top: 5px; font-weight: bold;">Automatisation Bot</p>
        <img src="${brandStyle.logoUrl}" alt="${selectedBrand} Logo" style="max-width: 150px; margin-top: 10px;">
      </div>
    </div>
```

2. Success Notification Flow:

The `**sendEmailNotificationSuccess**` function generates an email confirming the successful execution of data processing tasks.

The message contains details like the submission ID and brand information to provide context to the recipient.

- - Recipient: Directed to the same email address to maintain consistency in communication.
- - Subject Line: The subject line conveys a positive message, indicating that a form has been successfully submitted.
- - Email Content: The HTML body of the success email adopts a congratulatory tone and provides details of the processed data, including submission ID and company information.



```
// Define and use HTML body for success email
var htmlBody = `
<div style="font-family: Arial, sans-serif; color: #333; max-width: 600px; margin: auto;">
<div style="background-color: #28a745; color: #ffffff; padding: 20px; text-align: center;">
  <h1 style="margin:0; font-size: 24px;">Congratulations!</h1>
  <p>Your data has been processed successfully.</p>
</div>

<div style="padding: 20px; background-color: #f9f9f9; border-bottom: 4px solid #ddd;">
  <h2 style="color: #28a745;">Success Details</h2>

  <p><strong>Submission ID / SurveyNumber:</strong> ${submissionId}</p>
  <p><strong>Brand:</strong> ${selectedBrand}</p>
  <p><strong>Company:</strong> ${companyName}</p>
</div>

<div style="padding: 20px;">
  <p>This completion notice confirms the successful processing of your data with no issues.</p>

</div>

<div style="background-color: #f9f9f9; padding: 20px; text-align: center; color: #666;">
  <p>Best Regards,</p>
  <p style="margin-top: 5px; font-weight: bold;">Automatisation Bot</p>
  <img src="${brandStyle.logoUrl}" alt="${selectedBrand} Logo" style="max-width: 150px; margin-top: 10px;">
</div>
/div>
```

3. Branding and Styling

The script incorporates a styling guide that aligns with the company's branding strategy. It uses conditionally applied styles based on the `brand` variable to personalize the emails for different brands under the company umbrella. This attention to detail in the email's appearance reinforces the company's professional image.

```
// Define brand-specific styles and logo URLs
var styles = {
  'Agiliz': {
    color: '#E01A4F', // Agiliz brand color
    logoUrl: 'https://storage.googleapis.com/data_maturity_brandlogos/agiliz%20logo.png' // Agiliz logo
  },
  'Solifaction': {
    color: '#009688', // Solifaction brand color
    logoUrl: 'https://storage.googleapis.com/data_maturity_brandlogos/solifaction%20logo.png' // Solifaction logo
  }
};

// Set default brand if not provided
var selectedBrand = styles[brand] ? brand : 'Agiliz';
var brandStyle = styles[selectedBrand];
```

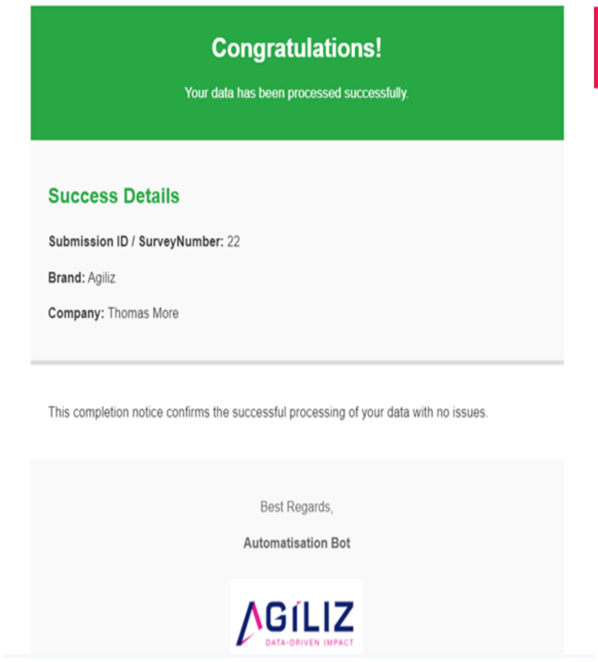
4. Email Dispatch:

The MailApp service in Apps Script is responsible for sending the email. It accepts parameters such as the recipient's email address, the subject line, and the HTML body content. Once the `sendEmail` function is called with the appropriate parameters, the email is queued for delivery through Google's mail servers.

```
MailApp.sendEmail({
  to: email,
  subject: subject,
  htmlBody: htmlBody
});
```

For deployment, ensure that the email addresses, URLs, and any sensitive data are correctly configured and secured. This script should be thoroughly tested to ensure that it handles all potential message types correctly and sends notifications as expected.

7.2. Result:


Congratulations!

Your data has been processed successfully.

Success Details

Submission ID / SurveyNumber: 22

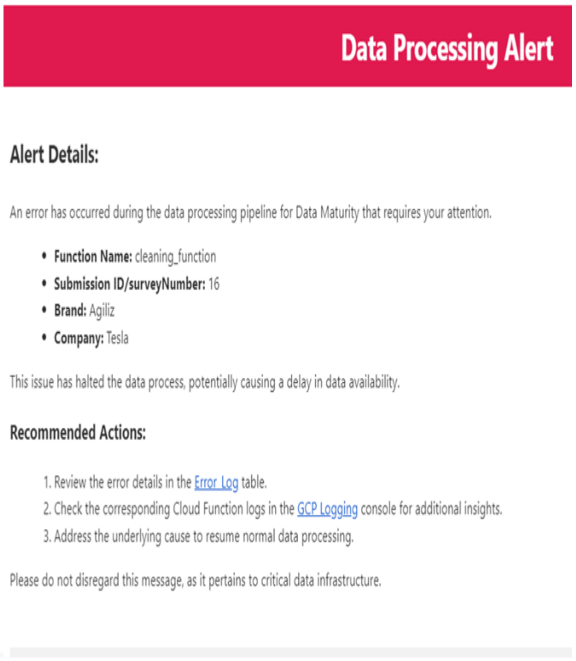
Brand: Agiliz

Company: Thomas More

This completion notice confirms the successful processing of your data with no issues.

Best Regards,

Automatisation Bot


Data Processing Alert

Alert Details:

An error has occurred during the data processing pipeline for Data Maturity that requires your attention.

- **Function Name:** cleaning_function
- **Submission ID/surveyNumber:** 16
- **Brand:** Agiliz
- **Company:** Tesla

This issue has halted the data process, potentially causing a delay in data availability.

Recommended Actions:

1. Review the error details in the Error Log table.
2. Check the corresponding Cloud Function logs in the GCP Logging console for additional insights.
3. Address the underlying cause to resume normal data processing.

Please do not disregard this message, as it pertains to critical data infrastructure.

Our "Data Maturity" project uses Google Cloud’s tools to quickly inform our team about any issues or successes in our data processing. This system of automated alerts helps us respond fast, keeping our work smooth and efficient. Each alert helps us make better decisions and keeps our project on track.

8. The Looker Dashboard

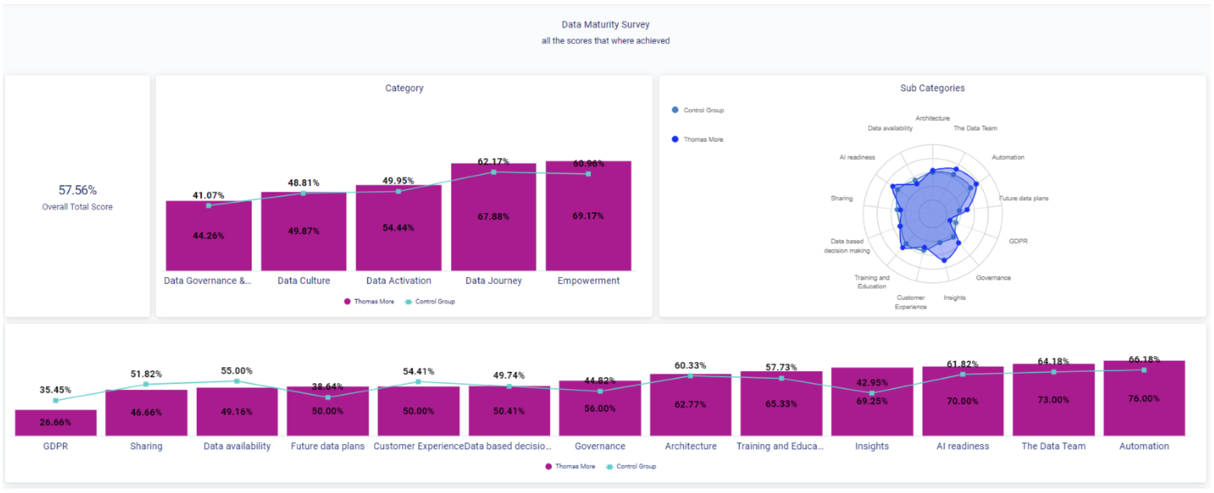
8.1. Customer Dashboard

1. Overall view

For the overall view page, we have a summary of the Total score, Category scores and sub-category scores visualized as shown below (all categories are visualized the same way so only one example of them is shown)

This overview page is mainly used to get a quick look of the scores and or to make a pdf of the scoring

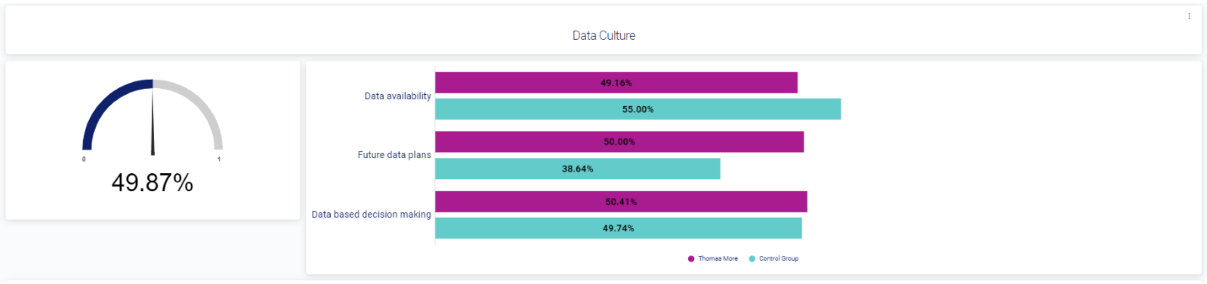
There is also a monitoring group in the radar plot so we can make it visible how and where they should look to improve.



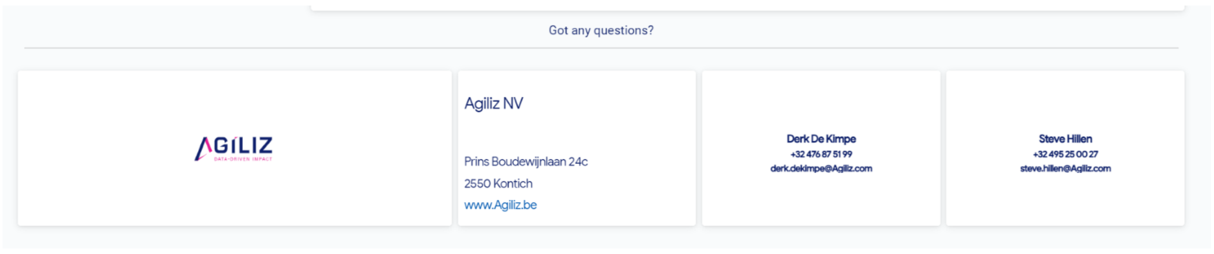
The filters that can be used here are:

- Function
  - To show and choose what company roles the personnel had that filled in the surveys.
- Survey Number
  - To specifically look at one or more surveys instead of all of them at ones.

We view also each category and their sub categories



At the bottom of the page there is a footer as well



Got any questions?

Agiliz NV

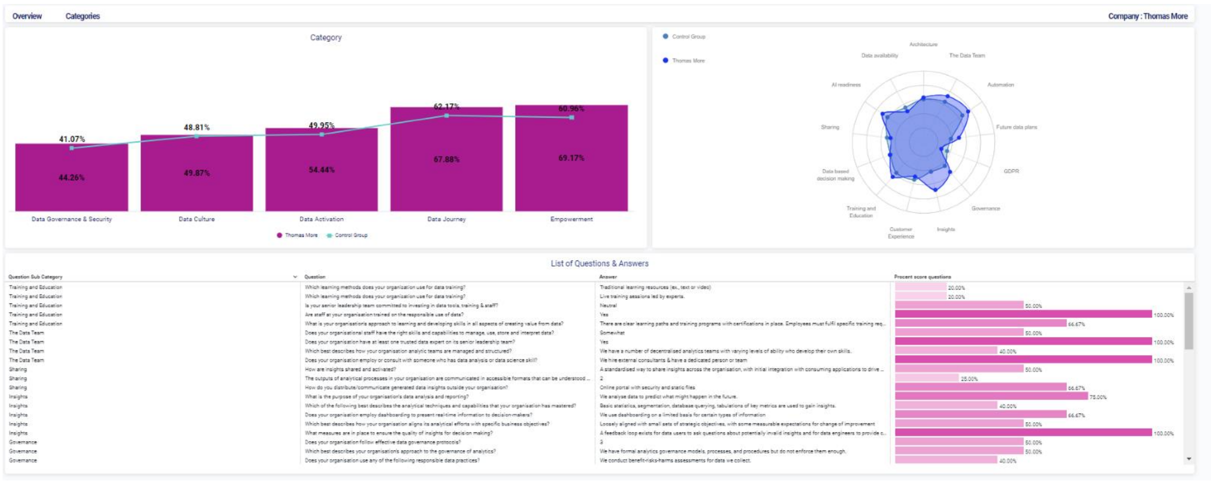
Prins Boudewijnlaan 24c
2550 Kontich
www.Agiliz.be

Derk De Kimpe
+32 476 87 51 99
derk.dekimpe@Agiliz.com

Steve Hillen
+32 495 25 00 27
steve.hillen@Agiliz.com

2. Interactive categorical view

For the interactive part of the dashboard, we look closer at the individual questions of each category / sub category



Question Sub Category	Question	Answer	Procent score questions
Training and Education	Which learning methods does your organization use for data training?	Traditional learning resources (ex., text or video)	20.00%
Training and Education	Which learning methods does your organization use for data training?	Live training sessions led by experts.	20.00%
Training and Education	Is your senior leadership team committed to investing in data tools, training & staff?	Neutral	50.00%
Training and Education	Are staff at your organisation trained on the responsible use of data?	Yes	100.00%
Training and Education	What is your organisation's approach to learning and developing skills in all aspects of creating value from data?	There are clear learning paths and training programs with certifications in place. Employees must fulfil specific training req...	66.67%
The Data Team	Does your organisational staff have the right skills and capabilities to manage, use, store and interpret data?	Somewhat	50.00%
The Data Team	Does your organisation have at least one trusted data expert on its senior leadership team?	Yes	100.00%
The Data Team	Which best describes how your organisation analytic teams are managed and structured?	We have a number of decentralised analytics teams with varying levels of ability who develop their own skills.	40.00%
The Data Team	Does your organisation employ or consult with someone who has data analysis or data science skill?	We hire external consultants & have a dedicated person or team	100.00%
Sharing	How are insights shared and activated?	A standardised way to share insights across the organisation, with initial integration with consuming applications to drive ...	50.00%
Sharing	The outputs of analytical processes in your organisation are communicated in accessible formats that can be understood ...	2	25.00%
Sharing	How do you distribute/communicate generated data insights outside your organisation?	Online portal with security and static files	66.67%
Insights	What is the purpose of your organisation's data analysis and reporting?	We analyse data to predict what might happen in the future.	75.00%
Insights	Which of the following best describes the analytical techniques and capabilities that your organisation has mastered?	Basic statistics, segmentation, database querying, tabulations of key metrics are used to gain insights.	40.00%
Insights	Does your organisation employ dashboarding to present real-time information to decision-makers?	We use dashboarding on a limited basis for certain types of information	66.67%
Insights	Which best describes how your organisation aligns its analytical efforts with specific business objectives?	Loosely aligned with small sets of strategic objectives, with some measurable expectations for change of improvement	50.00%
Insights	What measures are in place to ensure the quality of insights for decision making?	A feedback loop exists for data users to ask questions about potentially invalid insights and for data engineers to provide c...	100.00%
Governance	Does your organisation follow effective data governance protocols?	3	50.00%
Governance	Which best describes your organisation's approach to the governance of analytics?	We have formal analytics governance models, processes, and procedures but do not enforce them enough.	50.00%
Governance	Does your organisation use any of the following responsible data practices?	We conduct benefit-risks-harms assessments for data we collect.	40.00%

The filters that can be used here are:

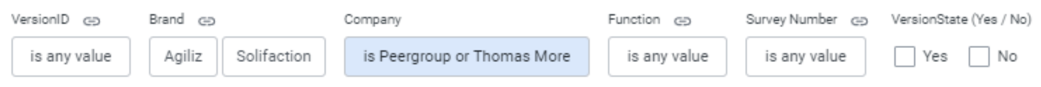
- Category
    - To choose and specify the categories we want to dive in.
- Sub-Category
    - To choose and specify the subcategories we want to dive in.
- Function
    - To show and choose what company roles the personnel had that filled in the surveys.
- Survey Number
    - To specifically look at one or more surveys instead of all of them at ones.

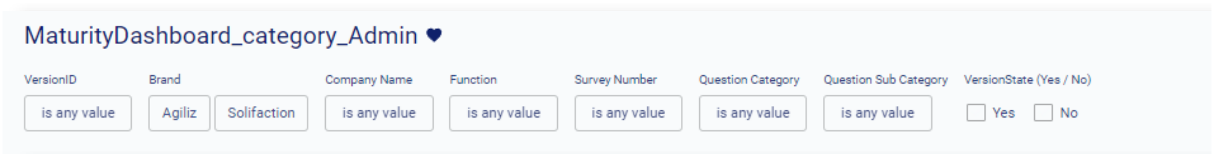
For the question-and-answer list we get a listing of each question and given answer, meaning if a question was a checkbox question (multiple answer question), then each individual answer will be shown and how much that specific answer contributed to the score of the that question on 100%.

8.2. Admin Dashboard

Overall, the admin panel has the exact same functionalities as the customer dashboard. But the big difference is that we can see all the companies' and all of their surveys so there are a couple more filters.

1. Admin overview / Category


VersionID: is any value
Brand: Agiliz | Solifaction
Company: is Peergroup or Thomas More
Function: is any value
Survey Number: is any value
VersionState (Yes / No): Yes | No

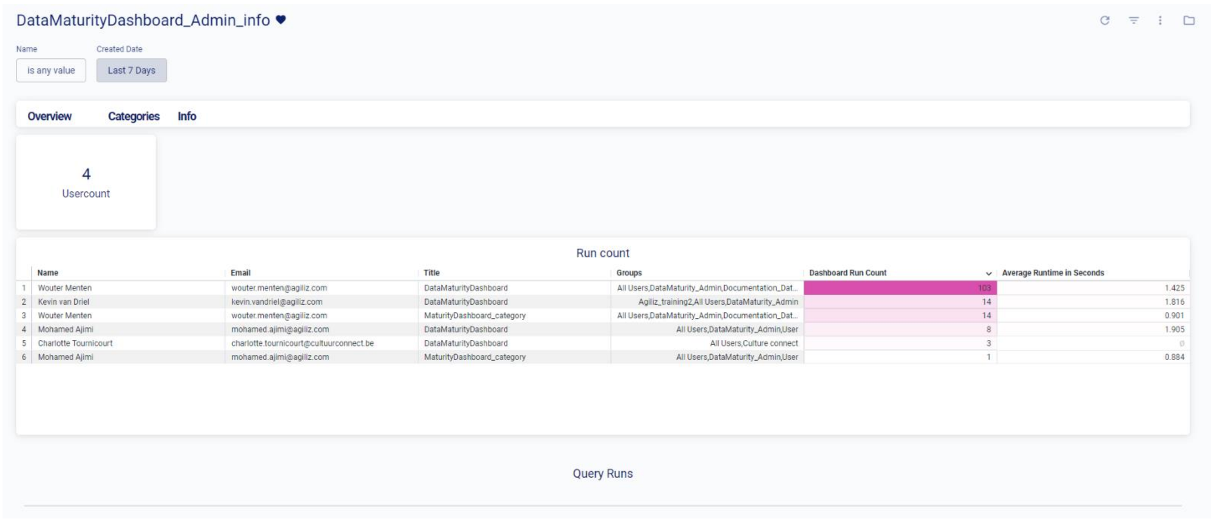

MaturityDashboard_category_Admin
VersionID: is any value
Brand: Agiliz | Solifaction
Company Name: is any value
Function: is any value
Survey Number: is any value
Question Category: is any value
Question Sub Category: is any value
VersionState (Yes / No): Yes | No

The extra filters here are:

- Brand
  - Since the dashboard is made to host both Agiliz and Solifaction data
- Company name
  - To specifically be able to look at a chosen company
- Version ID
  - To select the specific versions of the form
- Version State
  - To look for only the active or non-active versions

2. Admin – Info

The info panel is split up into a user count with their total run counts


DataMaturityDashboard_Admin_info

Name: is any value
Created Date: Last 7 Days

Overview | Categories | Info

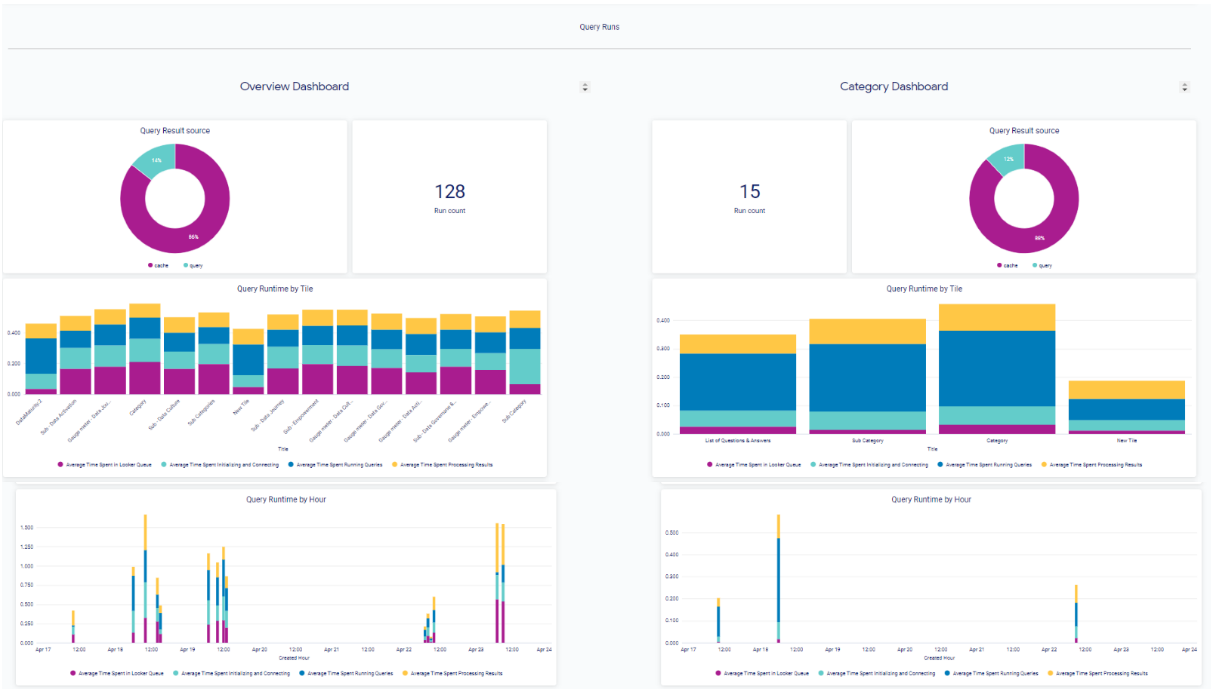
4
Usercount

Run count

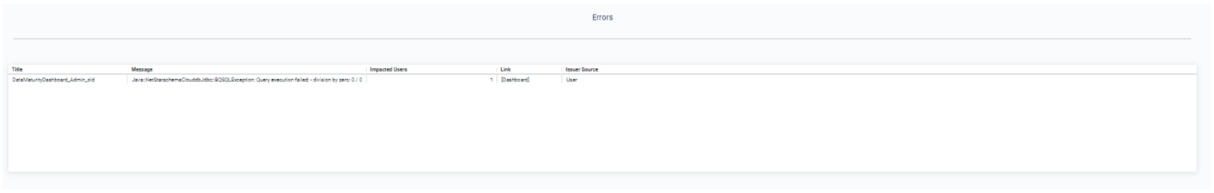
	Name	Email	Title	Groups	Dashboard Run Count	Average Runtime in Seconds
1	Wouter Menten	wouter.menten@agiliz.com	DataMaturityDashboard	All Users,DataMaturity_Admin,Documentation_Dat...	103	1.425
2	Kevin van Driel	kevin.vandriel@agiliz.com	DataMaturityDashboard	Agiliz_training2,All Users,DataMaturity_Admin	14	1.816
3	Wouter Menten	wouter.menten@agiliz.com	MaturityDashboard_category	All Users,DataMaturity_Admin,Documentation_Dat...	14	0.901
4	Mohamed Ajimi	mohamed.ajimi@agiliz.com	DataMaturityDashboard	All Users,DataMaturity_Admin,User	8	1.905
5	Charlotte Tournicourt	charlotte.tournicourt@cultuurconnect.be	DataMaturityDashboard	All Users,Culture connect	3	0
6	Mohamed Ajimi	mohamed.ajimi@agiliz.com	MaturityDashboard_category	All Users,DataMaturity_Admin,User	1	0.884

Query Runs

The Query runs of each dashboard of the customer

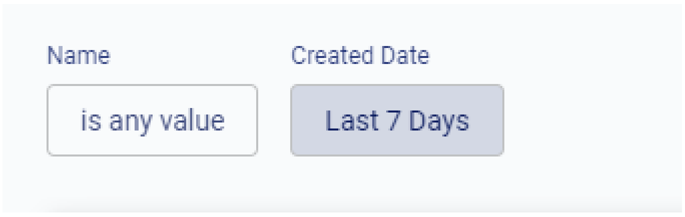

Query Runs
Overview Dashboard
Category Dashboard
Query Result source
128
Run count
15
Run count
cache
query
Query Runtime by Tile
Query Runtime by Hour
Average Time Spent in Looker Queue
Average Time Spent Initializing and Connecting
Average Time Spent Running Queries
Average Time Spent Processing Results

And the encountered Errors of each dashboard


Errors

Title	Message	Impacted Users	Link	Issuer Source
DataMaturityDashboard_Admin_old	Java::NetStarschemaClouddbJdbc::BQSQLException: Query execution failed: - division by zero: 0 / 0	1	[Dashboard]	User

Filters


Name

is any value

Created Date

Last 7 Days

Name: is dependent on the group name of the person that is connected to the dashboards

So, we could select a specific company group and only see there runs

Created Date: To select and filter on a specific time span

9. Conclusion:

Key Achievements:

- **Automated Data Processing:** By automating the entire data flow from collection to insight generation, the project significantly reduces the manual effort required, minimizes errors, and accelerates the time to insight.

- **Advanced Data Analysis:** Utilizing BigQuery and Cloud Functions, among other tools, the project applies complex transformations and business logic to raw data, enabling sophisticated analysis that can drive strategic decisions.

- Dynamic Reporting and Visualization: Through the creation of interactive dashboards and reports, stakeholders are provided with intuitive access to data insights, facilitating a deeper understanding of their organization's data maturity.

Challenges Overcome:

- Scalability and Efficiency: The project successfully addresses challenges related to handling large volumes of data, ensuring that the pipeline scales efficiently while maintaining high performance.

- Data Quality and Consistency: By implementing rigorous cleaning and validation processes, the project ensures that the data used for analysis is of high quality and consistency.

- Complex Data Transformation: Through innovative SQL scripting and Cloud Function logic, the project navigates the complexities of transforming multi-dimensional and multi-type data into a format suitable for analysis.