Lisbon School
of Economics
& Management
Universidade de Lisboa

Master
Data Analytics for Business

Master´s Final Work
Project

Solar Boats in the Amazon: A Cloud Data Warehouse Solution

Daniel Esteban Enriquez Eguiguren

Supervision:
Carlos Manuel Jorge Da Costa
Mário Fernando Maciel Caldeira
Flávio Alexandre Costa Romão

January - 2025

To my cat.

GLOSSARY

3NF – Third Normal Form

AWS – Amazon Web Services.

BI – Business Intelligence

DBT – Data Build Tool.

DW – Data Warehouse.

ELT – Extract Transform Load.

ETL – Extract Transform Load.

GCP – Google Cloud Platform.

KII – Key Impact Indicator

KPI – Key Performance Indicator

MS SSIS – Microsoft SQL Server Integration Services.

OLAP – Online Analytical Processing.

OLTP – Online Transactional Processing.

PKM – passenger-kilometre.

SQL – Structured Query Language.

.

ABSTRACT

Electric transportation has risen due to concerns about global warming and recent changes toward sustainable transportation. However, it is not easy to measure the actual sustainability of electric vehicle transportation.

Kara Solar is a non-profit organization that builds solar-powered boats in the Amazon rainforest, providing clean transportation for Indigenous communities. These solar-powered boats contribute to sustainable transportation in the Amazon rainforest and might prevent deforestation.

Typically, sustainability is represented by social, environmental, and economic. It is important for Kara Solar to be able to measure its impact on these three pillars. Therefore, there is an intrinsic need to make sense of its data by collecting, storing, and analyzing it.

The main objective of this project is to develop an artifact for the collection, storage, and analysis of Kara Solar's data to measure its impact via proxy variables to at least one of these sustainability pillars.

The artifact provides a cloud-native data warehousing solution that is scalable to process big data generated by IoT devices. The artifact features a Data Warehouse architecture based on the Kimball approach with modern components such as a Data Lake, ELT process, and transformations with a data build tool.

The artifact proved useful for collecting, storing, transforming, and analyzing Kara Solar's data from multiple sources, showcasing to stakeholders the impact within the Social Impact of Sustainability.

KEYWORDS: Data Lakehouse; Data Warehouse; Business Intelligence; Cloud Computing.

JEL CODES: G21; G31; M15; M54; C88; Y1.

Table of Contents

TABLE OF FIGURES

ACKNOWLEDGMENTS

I would like to thank my colleagues in Kara Solar, especially Billy Bowman-Shaw and Jordi, for their ideas, help and test of our boat monitoring system, “love letter.”

.

1. INTRODUCTION

1.1 Background

Electric transportation has risen due to concerns about global warming and recent changes toward sustainable transportation. According to the International Energy Agency (IEA, 2024), the market share of electric vehicles (EV) around the globe has risen from 4% in 2020 to around 18% in 2023. The share of electric buses is also growing worldwide. The recent factsheet released by the European Automobile Manufacturers' Association (European Automobile Manufacturers' Association, 2023) provides insights on the growth that electric buses have experienced in Europe. On 2018, only about 1.8% of new buses sold in the European Union were electric. This number has grown to about 12.7% in 2022.

Not only are terrestrial electric transportation modes rising, but also maritime transportation. According to (PR Newswire, 2024), electric boats are also on the rise. The electric boat market is forecasted to grow at a compounding annual growth rate of 12.9%, reaching a market value of 16.6 billion us dollars by 2031.

These numbers sound impressive, but how does it translate into sustainability? A big part of deforestation is linked to road construction; in fact, it is estimated that 95% of deforestation occurs within a 5.5 km radius of roads (Barber et al., 2014). Meanwhile, Vilela et al. (2020) found that 12,000 km of new roads in the Amazon rainforest would result in 2.4 million hectares of deforestation, meaning 200 hectares per km of new road built.

Kara Solar is a non-profit organization that builds solar-powered boats in the Amazon rainforest, providing clean transportation for Indigenous communities. Its goal is to defend the tropical rainforests and collaborate with indigenous guardians, building clean transport networks and technological independence (Kara Solar, n.d.). These solar-powered boats contribute to sustainable transportation in the Amazon rainforest and might prevent deforestation.

Typically, sustainability is represented by social, environmental, and economic (Purvis et al., 2019). It's important for Kara Solar to be able to measure its impact on these three pillars. Therefore, there is an intrinsic need to make sense of its data by collecting, storing, and analyzing it.

The main objective of this project is to provide a framework for collecting, storing, and analyzing Kara Solar's data to measure its impact via proxy variables on at least one of these sustainability pillars.

The framework proposed will focus on the social pillar, being able to measure the social impact by analyzing the routes taken by the solar-powered boats, the amount of energy consumed by the boats, the number of passengers that are traveling, and the purposes of such trips.

Kara Solar has been trying to measure this social impact in the past, but unfortunately, previous solutions have proven insufficient. During 2021-2023, the captains of the boats carried a Garmin GPS that would send its coordinates to the Garmin Server every 10 minutes, and the trip route could be seen within the Garmin Explore website. However, seeing everything within this website was inconvenient, and data access was problematic.

An ETL pipeline was developed to try to solve this problem of data accessibility. An ETL pipeline was developed on the Google Cloud Platform. This solution consisted of various serverless cloud functions to extract data from the Garmin Connect Restful API and process this data by removing entries that would not make sense, such as the boat being static or coordinates indicating a geographical position unfeasible, such as showing in the north pole or similar while keeping the records with good GPS signal. Finally, this cleaned data was inserted into various tables in a warehouse hosted on BigQuery.

A dashboard was also built to show the boats' routes, the number of trips, and the distance traveled. This solution proved the importance of incorporating data analysis to measure Kara Solar's impact. However, this solution quickly showed its limitations, given the sparse data points collected. Every 10 minutes, there was no way to measure the distance traveled accurately; there were other problems with this solution: the GPS could discharge and stop recording data and sending it to the satellite, and the data would not be collected if forgotten in the community. Finally, Garmin GPS could not collect information on the number of passengers traveling, the purpose of the trip, or telemetry data related to the boat's performance.

Given the limitations of the previous solutions, in 2023, Kara Solar started the development of its own onboard system to collect the data needed. This consists of a custom-made desktop GUI application for Raspberry Pi to collect boat performance telemetry data every second, as well as GPS signal, purpose of trip, and number of passengers traveling. While also helping the captain monitor important performance indicators of the boat's performance in real-time. This app solves the problems found with previous data collection devices and was deployed a year later, in September 2024.

This app, while useful, also comes with limitations. The biggest one is the impossibility of streaming the data collected to a private server or public Cloud due to connectivity limitations found in the Amazon rainforest. Without a cellular network signal or other types of wireless communication signals, it is unfeasible to use services such as Azure IoT or similar; therefore, the app needs to store all the data locally and upload in batches.

The following section will formally enumerate the objective this project will achieve.

1.2 Objective

As mentioned before, the objective of this project is to create a comprehensive data pipeline artifact that facilitates the collection, storage, transformation, and analysis of Kara Solar's data, enabling the measurement of its impact on the social sustainability pillar via proxy variables and KIIs.

1.3 Structure of the Master's Final Work

This Master's Final Work is structured as follows:

Chapter 2 contains the Methodological Approach, theoretical foundation for the steps to develop the artifact. Chapter 3 will focus on the literature review and different architectural choices for the Data Warehouse and Lake. Chapter 4 will describe empirical work, which is the implementation of the artifact. Finally, Chapter 5 will present the conclusions of the project.

2. Methodological Approach

The main objective of this project is to create a comprehensive data pipeline artifact that facilitates the collection, storage, transformation, and analysis of Kara Solar's data, enabling the measurement of its impact on the social sustainability pillar via proxy variables and KIIs. To develop such artifacts, this project will follow a design science methodology. This methodology contrasts natural sciences (which focuses on natural things, how they are, and how they work). DSR focuses on artificial things, such as how to design and make artifacts (Simon, 1988).

As Aparicio et al. (2023) mention, artifacts are outputs of design science. This can be in the form of constructs, models, methods, and instantiations. These artifacts must be an improvement of the current solution to a given problem or the first solution for a given problem.

The objective of this project is to create an artifact to solve a problem, which will be the first solution to the objective discussed in Section 1. This artifact is going to be an instantiation, an implementation of the artifact itself. A whole architecture for collecting, storing, and analyzing Kara Solar’s data to measure its social impact via proxy variables and KIIs.

DSR is the to-go methodology for these types of projects, as its focus is to answer the question, “How can we develop this?”. This methodology is especially popular in fields such as computer science, management science, information systems, and architecture (Aparicio et al., 2023). Since this project is mainly focused on data engineering, this is the best approach possible, as it is necessary to design and develop a robust, scalable, and reliable data architecture, which is one of its objectives.

This project adheres to the methodological approach of Design Science recommended by Aparicio et al. (2023) that consists of 6 clearly defined and outlined phases: Identify Problems and Motives, Define Objectives of a Solution, Artifact Design and development, Demonstration, Evaluation, Communication, diagramed in FIGURE 1.

The identity Problem and Motives phase refers to the problem understanding, what we want to develop, why, and who is going to use it. It is important to carefully outline the problem and justify why it needs a solution, why it is a problem, and why it needs to be solved.

Following the identification of the problem, it is important to define the objectives of the solution, which is phase 2. These are inferred from the problem definition and knowledge of feasible/unfeasible solutions (Aparicio et al., 2023).

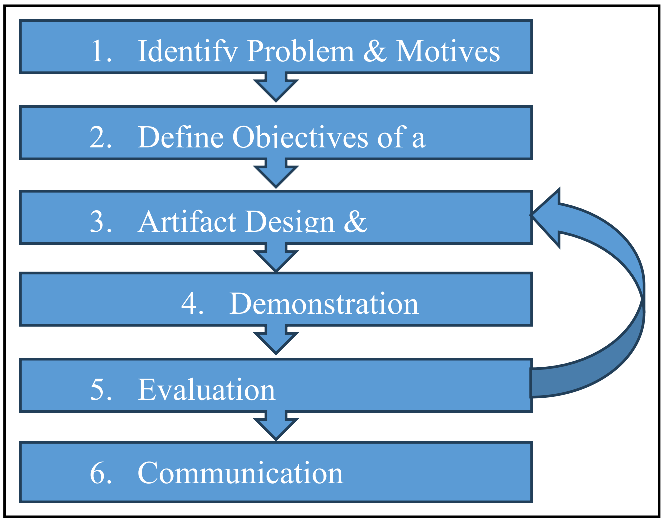


FIGURE 1 - Design Science Phases.

Next comes the third phase, in which an artifact is designed and developed. This can be one of the following: constructs, models, methods, and instantiations. To design an artifact, it is important to consider the problem to solve, the objectives, and the literature review. This is a critical step. How have other authors solved a similar problem, and what can you extract from those other solutions.

The fourth phase demonstrates how the artifact proposed in phase 3 is useful to solve an instance of the problem. It requires knowledge of how to use the proposed artifact (Aparicio et al., 2023).

Phase 5 involves the evaluation of the artifact, measuring how good or bad the artifact supports the solution to the problem. Aparicio et al., 2023 suggest evaluating with various metrics and quantitative analysis techniques. On the other hand, Prat et al. suggest a holistic approach to evaluating an artifact, a “holistic” approach that evaluates in the following dimensions: goal, environment, structure, activity, and evolution (2014). The first implementation of an artifact is usually not easy to evaluate, as it lacks a benchmark to compare the artifact. A method of evaluation could be of a type goal, which is whether the artifact solves the problem.

The goal evaluation encompasses three criteria: efficacy, validity, and generality. Efficacy refers to how well the artifact produces the expected outcome. Validity is the degree to which the artifact works correctly. Finally, generality is how generalized and broad the artifact is; efficacy is the most relevant criterion used (Prat et al., 2014).

If an artifact fails to accomplish the goals or, when evaluated, is not performing better than a previous solution, it's always a viable alternative to go back to Phase 3, as seen in FIGURE 1.

The final phase, number 6, involves the communication. In this final phase, the findings are communicated to a broader audience, detailing the problem, the artifact, and the evaluation.

3. Literature Review

As mentioned, Kara Solar built a desktop application to show and collect real-time sensor data, the number of passengers, and more. However, this data is stored in operational databases inside each boat. This data must first be consolidated and transformed into a more readable format to come up with relevant conclusions. This is where Data Warehouses play an important role.

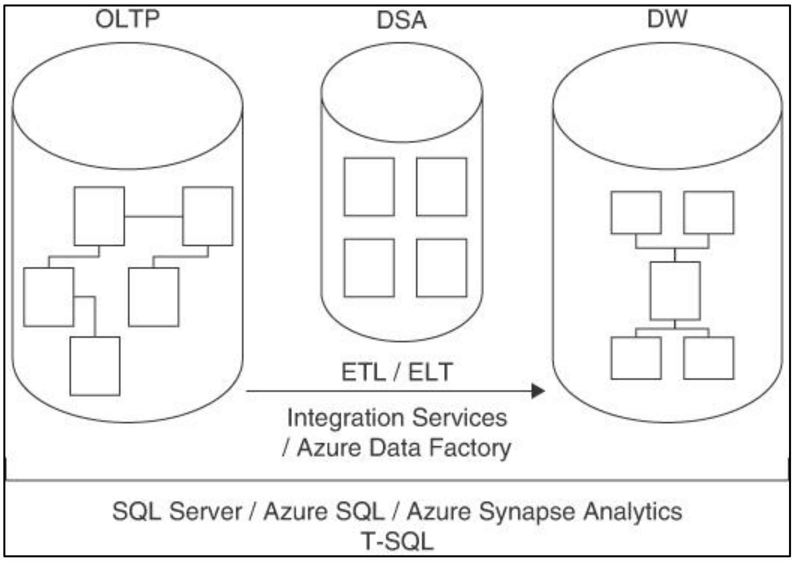


FIGURE 2 – Difference Between Operational Databases and Warehouse. (Ben-Gan, 2016)

FIGURE 2 displays the difference between operational or transactional databases and Data Warehouses. From the figure, it is seen that OLTP databases are in 3NF. This is because they are usually optimized for INSERT, UPDATE, and DELETE operations, as operational systems are needed to minimize redundancy for concurrent access and speed, and operational systems must be responsive to heavy loads.

On the other hand, we have DW which are optimized for analytical purposes. The architecture is quite different, as seen in the figure. Operational databases are not suitable for analytic purposes because of the number of tables, complex architecture, and relations. Less redundancy means that each entity has its own table (Ben-Gan, 2016). Thus making the analytics more complex. This is why DW has a different structure and architecture. Another big difference between the two is that usually, to keep the system responsive clean ups are performed on the OLTP. This can even include deleting unused data, losing historical data, and making it unfeasible for analytic purposes. The next section will describe in deeper detail the different architectures of DWs.

3.1 Data Warehouse

A Data Warehouse is a "subject-oriented, integrated, time-invariant, non-updateable collection of data used in support management decision-making process and business intelligence" (Hoffer et al., 2016, p. 430). Being subject-oriented implies that the DW is built around business entities, such as customers, stores, etc. Integrated means that the DW collects data from various sources and consolidates them into a single source of truth for the entire organization (Caetano & Costa, 2014). Time-invariant refers to a defined time frame for the data collection so business users can analyze time-series. Finally, non-updateable means that the final users cannot update the DW; it must match the origin (Hoffer et al., 2016).

Data Warehouses have three main architectures: Inmon, (2002), Kimball (2013), and Data Vault proposed by Linstedt & Olschimke, (2015). The first one revolves around building a single source of truth for the whole enterprise, normalized (3NF), not simply replicating the OLTP systems but integrating them into a single source of truth. The second approach involves building independent data marts, which are data collections of a specific business unit. All these independent data marts make up the entire Data Warehouse. Since every data mart is built independently, building the data marts can be quick for development; this approach is commonly referred to as the Kimball approach. The third architecture, Data Vault, uses a hybrid model to focus on flexibility and scalability. It separates data into three core components: hubs representing business keys, links between hubs and satellites, and contextual data such as attributes or history.

3.3.2. Kimball Approach

Kimball's DW/BI architecture consists of four distinct core elements: operational data sources, ETL system, data presentation area, and finally, the BI presentation layer. The operational data sources are the databases that operate the enterprises' applications the 'wheels' of the company; these are optimized for the application, and usually, the data professionals do not have control over them. The second part of the architecture is the ETL system. This is the connection between the DW and the operational data sources. Extraction is the process of reading and understanding the data from the source. Transformation is the process of putting the data in the necessary format for the data model of the DW, and finally, it concludes when this transformed data is physically loaded into the DW (Kimball & Ross, 2013).

The third component is the Presentation Area or the DW, which is a physical space where the data is organized, stored, and made available for the data users, stakeholders, and other BI purposes. Kimball recommends that this presentation area be made available to users in the format of OLAP cubes or star schema relationships. This is because the users are more comfortable with this type of format, and BI tools usually support these two types of formats well. Finally, the fourth component is the BI applications, which can be as simple as ad hoc queries for quick analysis or reporting or more complex applications such as data mining, machine learning models, or forecasting (Kimball & Ross, 2013).

FIGURE 3 shows Kimball's DW/BI architecture and the four core components. The transactional databases contain vital enterprise data. This data is then accessed by the ETL system, and the period is set by the needs of the enterprise. Then, it is transformed to match the DW data model and loaded into this DW using the same ETL process. The third component, the presentation layer, contains the DW in a format easily accessible by the end data users, normally a star schema model. Finally, BI Applications can range from simple SQL queries to data visualization tools such as Power BI, tableau, or complex machine learning applications (Kimball & Ross, 2013)

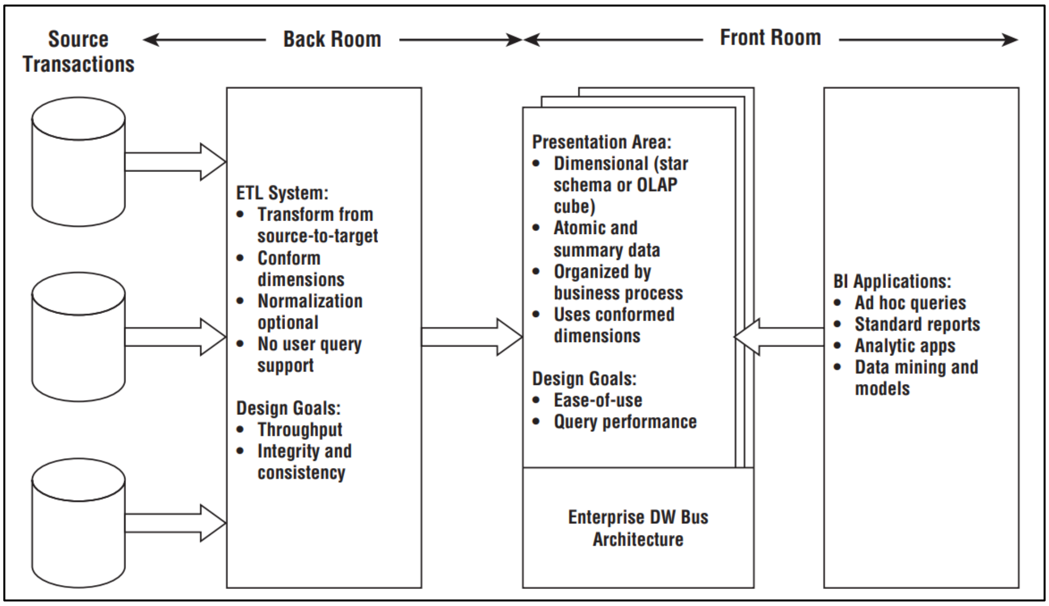


FIGURE 3 – Kimball Data Warehouse and Business Intelligence Architecture.(Kimball & Ross, 2013).

No matter the architectural choice for a DW, there is a consensus on how data should be delivered to BI and data users, and that is in the form of dimensional modeling, as this is the easiest for end users (Kimball & Ross, 2013). Dimensional modeling is a way of modeling the data into facts and dimensions. A fact is a table that stores measurements from a business process event, such as the number of products sold. These measurements by itself do not offer much value, but when examined in the context of dimensions, they can provide valuable insights. For example, the number of products sold on a specific date or a specific store. As Kimball and Ross (2013) mention, Dimensions describe a particular measure or event's who, what, where, when, how, and why.

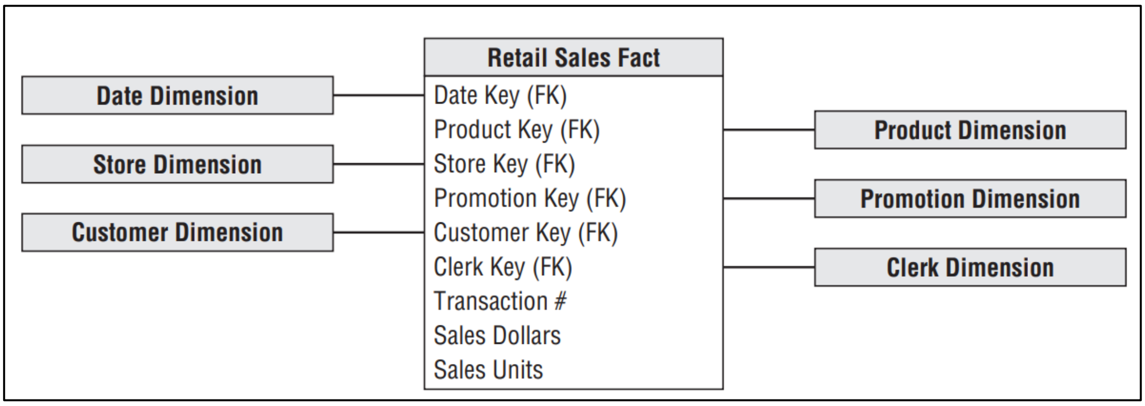


FIGURE 4 – Dimensional Modelling in Retail.(Kimball & Ross, 2013)

FIGURE 4 shows an example of dimensional modeling in action, a central fact table containing measures of a business process and sales in the case of this example. These metrics can provide insights when looked at in the context of the various dimensions. It can answer where the products of a particular brand sold the most, who the cleck is, who sells the most items, etc. This dimensional modeling is extremely easy to use for business intelligence and easy for data professionals to develop and maintain (Kimball & Ross, 2013).

3.3.1. Inmon Approach

In contrast to Kimball, Inmon proposed an operational data store, a single source of truth for an entire organization. This central repository is highly normalized, often resembling operational data models, and it is highly scalable. Instead of building independent data marts, with every independent Datamart going through its own ETL process, Inmon, (2002) proposed an integrated ETL process that consolidates every source into a single source of truth.

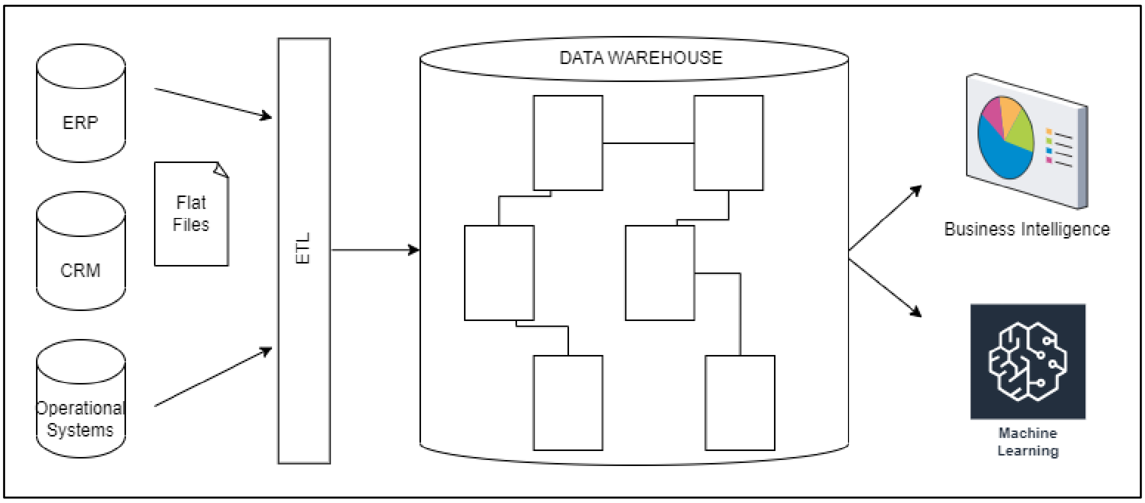


FIGURE 5 - Inmon DW Architecture.

FIGURE 5 shows a diagram of the Inmon Architecture approach for DW. Various sources are ingested into the ETL process, and various transformations occur. Then, this data is loaded into the DW in the fully normalized tables. As an illustrative example, operational systems may contain a table called customers, the CRM system contains a table client, and the ERP contains a table customer. This customer entity must be consolidated cleaned, duplicate rows eliminated, and undated information. Then, this table is loaded into a single fully integrated table customer within the DW. (Inmon, 2002)

This approach reduces redundancy, improves speed in updates, deletes, and inserts, allows for the most flexible design, improves data integrity adaptability to change, and supports the most complex queries. Every report can be built from normalized tables. However, this comes with a cost. It increases the number of joins required for common reports; inserts, updates, and deletes are fast, but queries are slow and more complex for end users. Another drawback of Inmon's approach is how hard it is to build a data warehouse that serves as a single source of truth for the entire enterprise, as every department should have a consensus on every definition, such as who a customer is. This makes the development of a fully normalized central repository slow and cumbersome. It is highly likely that the enterprise has completely changed by the time an Inmon DW is fully developed (Inmon, 2002).

3.3.1. Data Vault Approach

Data Vault was proposed to address problems when the data volume increases, but Kimball and Inmon usually do not handle data volume increases well (Yessad & Labiod, 2016). This approach is designed to be scalable, flexible, and auditable data warehouses. It clearly focuses on data lineage and the ability to trace data. Every row within the Data Vault must record the source and load data attributes. This approach is great for data lineage and compliance (Linstedt, 2002, 2003a, 2003b, 2004, 2005).

This approach does not clean data right away. Instead, all data, even if it is “wrong” or outdated, is treated equally. This approach makes it ideal for capturing changes that occur in an operational system or other sources (Linstedt, 2002).

Data Vault does not rely on the star schema nor fully normalized tables. Instead, it relies on three core structures: Hub, Link, and Satellites. The data model revolves around the Hubs, which are “business keys”. These are the most stable elements of the business and only change when the business changes. The most important part of the Data Vault is choosing the right business keys to build the data model around them and their relationships (Linstedt, 2002).

The hub represents a core business entity, usually entities such as products, customers, stores, etc. It’s important to note that hubs do not contain contextual data. Instead, it only contains a surrogate key, which is used to connect to other structures of the Data Vault, a business key, a unique identifier from the primary source of truth of the business entity, record source, comment, and load date. Important that every entity contains only one row within the hub (Linstedt, 2002).

The next structure is the Link, a table containing transactions between business keys, such as transactions or associations between customers and products, and a customer has a purchase history or transaction history of products. The Link is a table with many too many relationships between hubs. However, a link table does not contain contextual information about the entities. A link table might contain a surrogate key, a foreign key to the customer hub, or a foreign key to the orders hub without containing details on the order. Finally, it also contains the record source and timestamp (Linstedt, 2002).

The final structure is the satellite. As mentioned, hubs and links do not contain descriptive attributes. This is the function of the satellite tables. This adds context to the model to enrich the analysis. This table also captures changes within the data. Products might change size, color, or price; all this ever-changing information is stored in the satellite table of products, and this table is connected to the hub (Linstedt, 2002).

Data Vault is extremely useful for data lineage and auditing of the information systems within the organization, but the modeling is challenging. It takes specialized data architects to design an effective Data Vault. Another drawback is that it requires many joins for Business Intelligence to be effective. That is why Data Vault is often used in conjunction with the data marts layer that follows the star schema (Linstedt, 2002).

A summary of the advantages and disadvantages of the most popular architectures can be seen in Table I. It is important to note that every architecture has trade-offs. Some are fast to develop, but the data quality might not be as good as that of other slower-to-develop architecture. Some scale better than others.

Table I

DW Architectures Comparison.

	Inmon	Kimball	Data Vault
Data Modelling Complexity	High, requires tables to be in 3NF.	Simple, focus on star schema or snowflake.	High, focuses on hubs, links and satellites.

Development Time	Long, due to the need for single source of truth and normalization.	Short, easy to implement and understand.	Short, as it uses a flexible, incremental approach on every structure.
Flexibility	Less flexible, focuses first on normalization.	Flexible, relies on business requirements with some structure.	Highly flexible, as it shines in handling raw data and lineage.
Ease of Use	Complex for analyst due to multiple joins.	Easy, designed for end users, reporting and BI.	Complex for end users as many joins are required.
Suitability for Business Intelligence	Low, might require query tuning or preaggregation for BI.	High, almost every BI tool is built upon the star schema.	Low, it’s good for capturing raw data and changes, but will need data marts for BI.
Adaptability to Changes	Difficult to adapt to changes as foreign keys need to change.	Easy, just need to add more facts and dimensions.	Very adaptable, handles changes in the structure with ease.
Focus	Centralized data hub, single source of truth normalized.	Mainly reporting and Business Intelligence, end users.	Flexible modelling, data lineage and auditing.
Data Quality	Extremely high, due to normalization and centralization.	Depends on the accuracy of dimensions and facts, might lead to	Extremely high as it focuses on auditing.

		differences across the organization.	
Performance	Fast for inserts, updates and deletions but slow for reporting.	High as it’s pre aggregated and does not require as much joins as 3NF.	Optimized for big data but very likely to require tuning for reporting.
Transformation Layer	ETL process is extremely complex and maintenance heavy.	Quite simple ETL or ELT as every data mart is built independently.	ELT process with transformations based on hubs, links and satellites.

(Linstedt, 2002, 2003a, 2003b, 2004, 2005; Kimball & Ross, 2013; Inmon, 2002).

One important thing to mention is that for end-users, the consensus is to present the data in a star schema, as this is the most convenient for Business Intelligence Tools and easier for data analysts and end users to interact with the data. Another important thing to mention is that modern data warehouses rely on a hybrid architecture that makes development fast, such as DBT. The architecture is usually built on top of a data lake and is structured in staging, intermediate models, and data marts. This allows for fast-paced development with the benefits of data lineage and auditing and relies on the lower prices in computing and storage that Cloud Computing has empowered (Yessad & Labiod, 2016).

3.2 Data Lake

In recent years, with the introduction of the public Cloud, the reduction in storage costs, and the increasing amount of data enterprises collect and handle, a new term has arisen, Data Lake. This is an integrated central repository of various data types, structured and non-structured in raw format, usually files in CSV format, parquet, audio, video, binary large objects, XML, JSON, etc. The characteristics of a data lake are the following: it stores everything, has flexible access, and can be dived anywhere (Hoffer et al., 2016).

A data lake stores everything: as mentioned above, a data lake is a central repository for any data across the organization in raw format, meaning it has not been processed by any ETL or ELT process and contains the information as is, without any transformation. Therefore, a data lake has no defined architecture and can have different file formats. A data lake is usually a collection of objects within a defined path in cloud storage (Stein & Morrison, 2014).

Dive in Anywhere refers to the property being available by various organizational stakeholders but is only limited by confidentiality constraints. This ensures that the stakeholders can enrich their analysis without relying on transformations made by other departments in ETL/ELT processes (Hoffer et al., 2016).

Flexible access refers to the ability of different users to access the Lake without a rigid or predefined schema, unlike relational databases. This is a schema on read, allowing flexibility when reading data from multiple sources from the Lake (Hoffer et al., 2016).

FIGURE 6 represents a diagram of a Data Lake: a collection of objects usually within a defined path or bucket in Google Cloud Storage or Amazon S3, with different data sources and formats such as txt, blob, CSV, audio, video, JSON, XML, Apache Avro, among others.

While having a raw central data repository might sound reasonable and advantageous, caution must be exerted, as this might not be well managed, lack metadata information, and become a “Data Swamp”. As mentioned by Stein and Morrison, a data lake can quickly become a data graveyard due to a lack of management. It is easy to forget everything that a Data Lake contains and hope to do something with it in the future (Stein & Morrison, 2014).

FIGURE 6 – Data Lake.

3.3 ETL and ELT Process

Data is often used to extract, transform, and load from sources to the data warehouse. This involves extracting data from various sources, transforming this data to the defined schema, and loading it into the DW. Apparently, it is an easy process, but this system consumes most of the time, effort, and resources needed for a DW project (Kimball & Ross, 2013).

FIGURE 7 shows graphically a diagram of the ETL process. An enterprise usually has multiple data sources, as they usually have specialized software for different tasks, such as an ERP for accounting and management necessities and another software for marketing or other departments. The data is extracted and loaded into a staging area, where the transformation process occurs. This process is usually performed with specialized tools such as Microsoft SQL Server Integration Services and scripts for more advanced data transformation needs. The staging area only contains the latest data. After the transformation, this data is then loaded into the DW in the predefined schema.

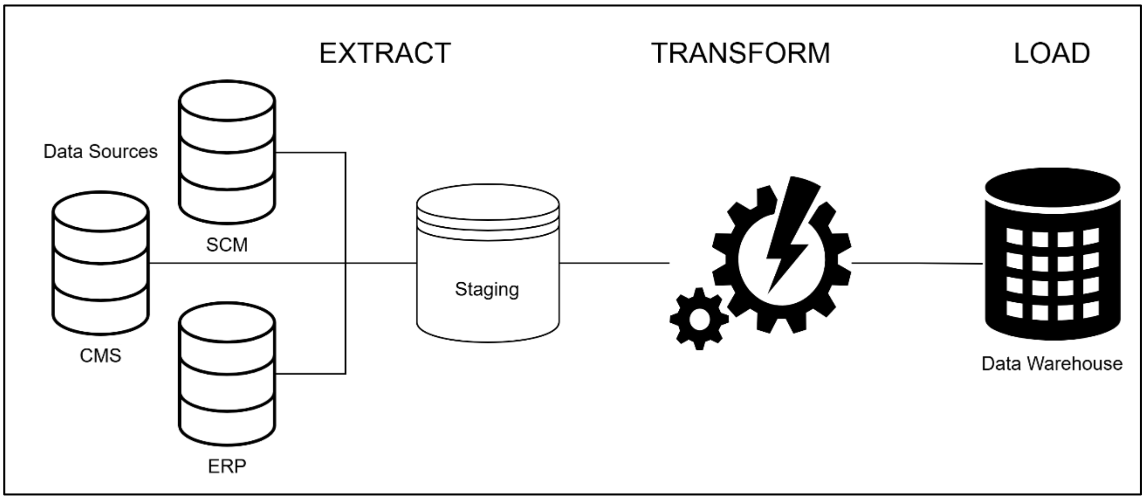


FIGURE 7 - ETL Extract, Transform, and Load Process.

There is another approach to this critical process to populate the DW, which has the same steps but in a different order: the ELT approach. This is seen as a more “modern” approach, as opposed to ETL, because it uses a combination of technologies for the different parts of the process instead of relying on a single ETL tool. It also relies on the reduction in storage and computing costs seen in recent years (Amazon Web Services, n.d.).

FIGURE 8 shows a diagram of the ELT process. The Extraction part is the same, however. However, this process usually uses different extraction technologies such as Fivetran, Airbyte, etc. The data is extracted from the source and then typically loaded into a bucket in AWS S3 or GCP Cloud Storage, usually to a data lake. Then, the transformation process begins, with tools such as DBT, materialized views or incremental tables being created on top of the sources. This makes up the data warehouse, which directly transforms the data from the Lake. This is possible due to the reduced computational power costs with tools like BigQuery or Amazon Redshift (Handy, 2016).

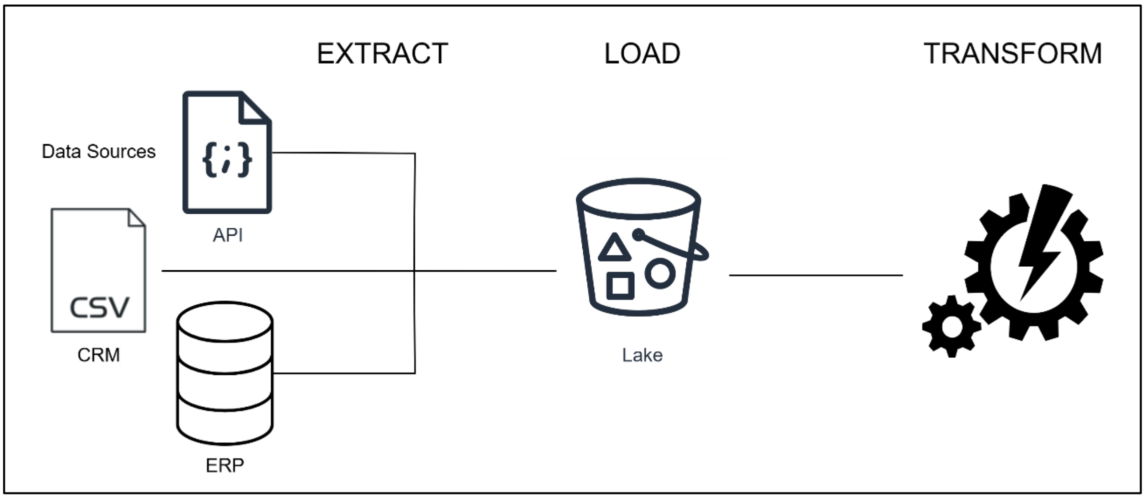


FIGURE 8 - ELT Extract, Load, and Transform Process.

There has been a recent shift towards ELT over ETL because of the speed of development. Since it has an additional staging process, it tends to be slower than ELT. Another disadvantage seen in the ETL process is the need to plan all the KPIs and reports that an organization will need. This might lead to overhead costs. A big improvement that ELT has made is its usage of distributed computing power, making it faster to develop, more cost-efficient, and more versatile, something that is much appreciated at this time. Due to high competition, enterprises must adapt quicker to changes and have faster analytics, ideally in real-time (Amazon Web Services, n.d.).

Overall, ELT processes are preferred nowadays over ETL, with certain exceptions, such as legacy systems. If only quick transformations are required, it might be easier and more cost-efficient to set up an ETL process (Amazon Web Services, n.d.).

4. Empirical Work

The empirical work will follow the phases discussed in the methodological approach, the 6 phases of Design Science Research.

4.1 Identify Problems and Motives

As mentioned in the introduction, Kara Solar wants to measure its sustainability impact, starting with the social pillar. The social pillar has many dimensions and includes things that are not easily measured, such as social equity and justice, as well as human rights, among others. However, there are things that can be measured, such as community development with the inclusion of these solar boats. This is our main problem:

developing a framework for the collection, storage, and analysis of Kara Solar's data to measure its social impact via proxy variables and KIIs.

There are various motives for developing our artifact: the main motive is to have a solid measurement of the impact of Kara Solar in the communities and how sustainable is this different method of transportation within the communities. In the first phase, which is covered by this project, the first social pillar is measured, and in the medium term, the other two pillars of sustainability are covered. That's why a scalable framework is needed to process data from multiple sources.

A second motive for developing the artifact, also not covered in this project, is to measure boat performance for the engineering team to find ways to improve the current designs and develop algorithms for effective boat energy spending on trips. This is another reason why a solid framework for data collection, storage, and analysis is needed.

The final reason is for funding purposes. As a non-profit organization, it is important to have KIIs to communicate to potential funding partners and stakeholders. These organizations want to know specifically what the actual impact of their donations is.

4.2 Define Objectives of a Solution

Now that the problem is clearly outlined, it's important to define the objectives to accomplish with our artifact.

In the motives, it is said that this project will need to be expanded soon, and in the introduction, it is mentioned that Kara Solar already has some data within Google Cloud's Platform infrastructure, so one of our objectives of the artifact is to design a scalable data architecture, by leveraging the existing infrastructure on Cloud. Migrations are costly in terms of manpower and financial resources, so it is important to leverage the existing infrastructure for our artifact. Also, it is said that an important aspect is to be able to scale this data architecture to accommodate more data sources in the Data Warehouse. This architecture will be useful not only for this project's goal: to measure the first pillar of sustainability within Kara Solar but also to expand it to measure all three pillars of sustainability and engineering purposes.

This data architecture must then also encompass a seamless integration of CI-CD pipelines, continuous development, and continuous integration. Data ingestion and data processing is not a single-time problem. This must be ingested and processed in a continuous way as priorities within the organization change, data sources change, and data schema changes.

The second objective of the artifact is to develop an ELT pipeline. Our data architecture must encompass ingesting or extracting the data from sources, loading it into the infrastructure on Cloud and within the DW, transforming this raw sources into usable data, to measure the first sustainability pillar, and in the medium term (not included in this project report) measure all three sustainability pillars and also to serve the engineering department demand for data.

The final objective for this artifact is to create a front-end dashboard, this is an important step. We not only need to design a scalable data engineering pipeline but also implement a front-end dashboard for end users. Most of the time, end users are not familiar with structured query language; therefore, it is hard for them to build their own reports, it's also the responsibility of the data department to deliver reports to end users and stakeholders.

4.3 Artifact Design and Development

In this phase of development, it's important to have a solid foundation of the theory that supports the design of the artifact. This project will follow best practices found in Google Cloud Platform, DBT and Looker documentation, and Kimball's approach to designing a DW.

The artifact design and development process will be broken into three distinct parts: dimensional modeling, data architecture, and development.

4.3.1. Dimensional Modelling

To come up with relevant measures or KIIs for the social impact of Kara Solar the first step is to gather the business requirements and data realities (Kimball & Ross, 2013). The business requirements are to show different metrics, which will be discussed shortly, in a dashboard, such as routes the boats are taking within the Amazonian rivers. These metrics can be examined across different dimensions, such as boat, day, and purpose of the trip.

It is important to note that non-profit organizations differ from for-profit organizations in terms of the metrics they want. The former wants to measure impact, while the latter wants to measure targets. For example, Kara Solar wants to measure how many members of the indigenous communities are using solar-powered boats, whereas a for-profit wants to measure how the year-on-year sales are going. The impact metrics are called Key Impact Indicators (KII), whereas enterprise progress towards objectives is called Key Performance Indicators (KPI).

As mentioned in the introduction, Kara Solar has a custom-made desktop application that collects telemetry data from the boats, as well as the captain's input, such as the number of passengers and the purpose of the trip. This desktop app collects some information that can be used to measure the social impact, such as when a trip begins and ends, the coordinates of the location of the boat every second, trip duration, the number of passengers a particular trip carries, and finally the purpose of the trip. The last two are inputs taken from the captain, while the rest of the data is collected automatically by the system, which takes this data from the sensors aboard.

Based on these requirements and the data realities, the following Key Impact Indicators are proposed:

1. Passenger-kilometer, pkm, is a popular unit of measurement in transportation, defined as the transport of one passenger by solar-powered boat over one kilometer.
2. Number of passengers traveling: This is the number of passengers using a particular boat in a particular time frame.
3. Total distance traveled (km) is just the total distance traveled by a particular boat in a particular time frame.
4. Number of trips: this is just the number of trips a particular boat made in a particular time frame. This can be examined by the purpose of the trip dimension.
5. Average Trip Duration: this is simply the average time to complete a trip, expressed in minutes.
6. Percentage of trips for essential purposes: This is the number of trips made for medical and school purposes over the total number of trips.

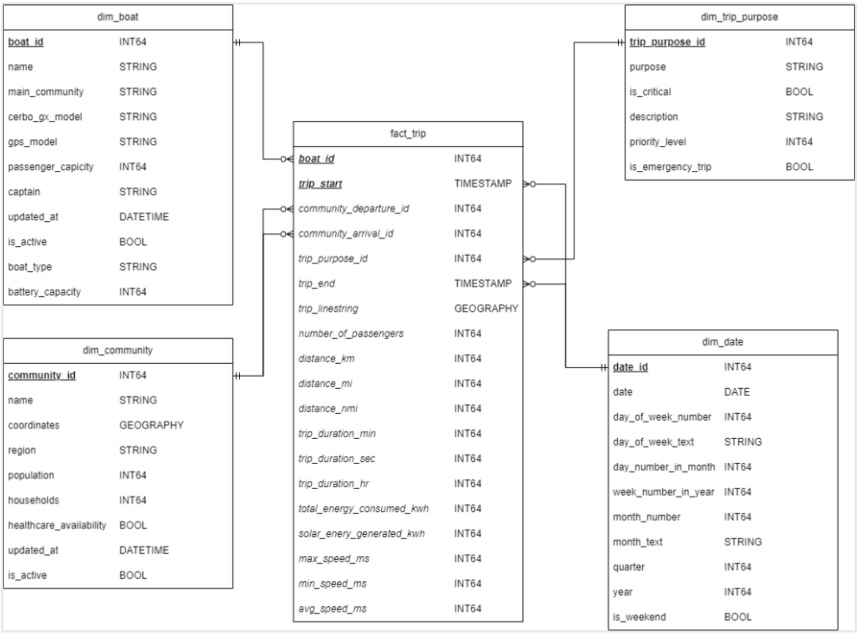
After defining the KIIs, Kimball and Ross (2013) recommend a four-step dimensional design process for dimensional modeling to set up the DW to accommodate the metrics Kara Solar wants to measure. This approach will be followed.

The first step is to select a business process. A business process is usually an action verb, such as invoicing, receiving payments and so on. For Kara Solar, the business process to measure is Trips. Moreover, trip logging expresses it as an actionable verb or an action. This business process is accompanied by the core system onboard the boats, as discussed previously.

The second step is to declare the grain. This is what a row within the fact table represents. It is always a critical step to declare the lowest granularity possible, as it’s impossible to get details below the granularity selected. The granularity for the data mart that this project describes is one row per trip.

The third step involves identifying the dimensions, which are decorators of the fact tables, to enrich the analysis. As Kimball and Ross (2013). mentions this represents the “who, what, where, when, why, and how” of our facts. For this data mart, the dimensions are boat, community, trip purpose, and date. This answers the questions of what boat is traveling, where it is going when it is traveling, and why it is going.

Finally, the last step is identifying the facts that this process is measuring. The KIIs are defined at the beginning of this section. Now that the steps to define the dimensional model for the business process Trip are concluded, the Logical Data Model Diagram of the Trip Data Mart is shown in FIGURE 9.



dim_boat	
boat_id	INT64
name	STRING
main_community	STRING
cerbo_gx_model	STRING
gps_model	STRING
passenger_capicity	INT64
captain	STRING
updated_at	DATETIME
is_active	BOOL
boat_type	STRING
battery_capacity	INT64

dim_community	
community_id	INT64
name	STRING
coordinates	GEOGRAPHY
region	STRING
population	INT64
households	INT64
healthcare_availability	BOOL
updated_at	DATETIME
is_active	BOOL

fact_trip	
boat_id	INT64
trip_start	TIMESTAMP
community_departure_id	INT64
community_arrival_id	INT64
trip_purpose_id	INT64
trip_end	TIMESTAMP
trip_linestring	GEOGRAPHY
number_of_passengers	INT64
distance_km	INT64
distance_mi	INT64
distance_nmi	INT64
trip_duration_min	INT64
trip_duration_sec	INT64
trip_duration_hr	INT64
total_energy_consumed_kwh	INT64
solar_enery_generated_kwh	INT64
max_speed_ms	INT64
min_speed_ms	INT64
avg_speed_ms	INT64

dim_trip_purpose	
trip_purpose_id	INT64
purpose	STRING
is_critical	BOOL
description	STRING
priority_level	INT64
is_emergency_trip	BOOL

dim_date	
date_id	INT64
date	DATE
day_of_week_number	INT64
day_of_week_text	STRING
day_number_in_month	INT64
week_number_in_year	INT64
month_number	INT64
month_text	STRING
quarter	INT64
year	INT64
is_weekend	BOOL

FIGURE 9 – Social Impact Datamart.

4.3.2. Architecture

As mentioned, Kara Solar already has an account in the Google Cloud Platform and has been using it; however, it does not have a clearly defined architecture. The main objective is to develop a scalable framework for collecting, processing, and analyzing Kara Solar's data.

This project will follow 's (n.d.) documentation and architecture recommendations for building a data warehouse within its infrastructure, with some modifications to improve the custom pipeline and reduce costs. The architecture used is simple, using Cloud Storage as a Data Lake for storing data from multiple sources and in different formats, step 1 in FIGURE 10. From this step, Google recommends the microservice Workflows as a fully managed orchestration platform. This is to trigger different data cleansing steps or other types of run functions.

The cleaned data goes directly to BigQuery, a fully managed, serverless, and scalable data warehouse solution. Here, the data is stored and can be queried to serve different purposes, such as visualization with Data Studio (number 5), Vertex AI, a managed platform for training machine learning models, and cloud functions to deploy machine learning models.

The core architecture relies on Cloud Storage as a Data Lake, Workflows for data movement and orchestration, and finally, BigQuery as the Data Warehouse.

This project, as mentioned, will follow a similar architecture, with some differences, to save costs and manage a custom pipeline. The data sources will be loaded into GCS as in the recommended architecture, BigQuery will be used as the main DW, and Looker (previously known as Data Studio) will be used for data visualization. For this project, there is no machine learning involved, so there is no need to use vertex AI or deploy any ML models with cloud functions.

The main difference is in the data cleansing and composer. This project will not use Workflows as an orchestrator but rather rely on the DBT cloud for this purpose and for data transformations.

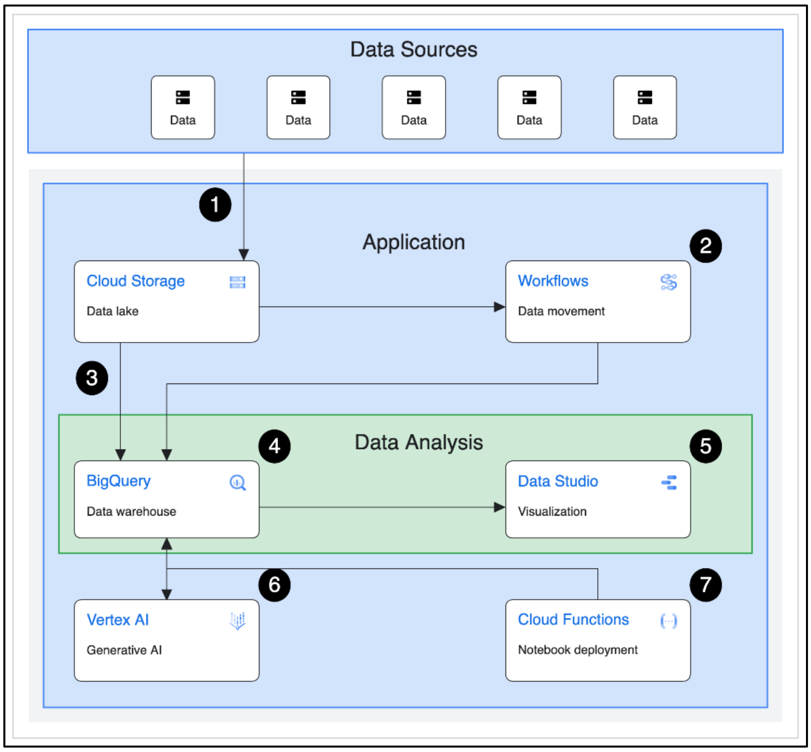


FIGURE 10 – Data warehouse with BigQuery. (Retrieved from Google Cloud, n.d.)

FIGURE 11 shows the architecture plan for this project and the ELT pipeline. The sources will be loaded into GCS with a Python script. This process is scalable as no matter how many boats are incorporated into the float, all of them will be able to upload their individual data in batches. The data Lake is composed of files of different natures within

GCS. BigQuery can access these files natively or with the help of a cloud function. DBT cloud will orchestrate the whole pipeline, and all the data transformations will be done with the DBT core. Finally, Looker will be used for data visualization and business intelligence.

Note how FIGURE 8 and FIGURE 11 resemble each other, the only difference is the technologies applied, the former is an ELT process skeleton while the latter is an implementation of the ELT process in the artifact.

One thing to note is that architecture is always evolving. For this project, a Python script combined with Unix cron jobs might be enough, but for ingesting more sources to the Data Lake, the architecture could evolve to include specialized tools for ingestion, such as Airbyte or Fivetran.

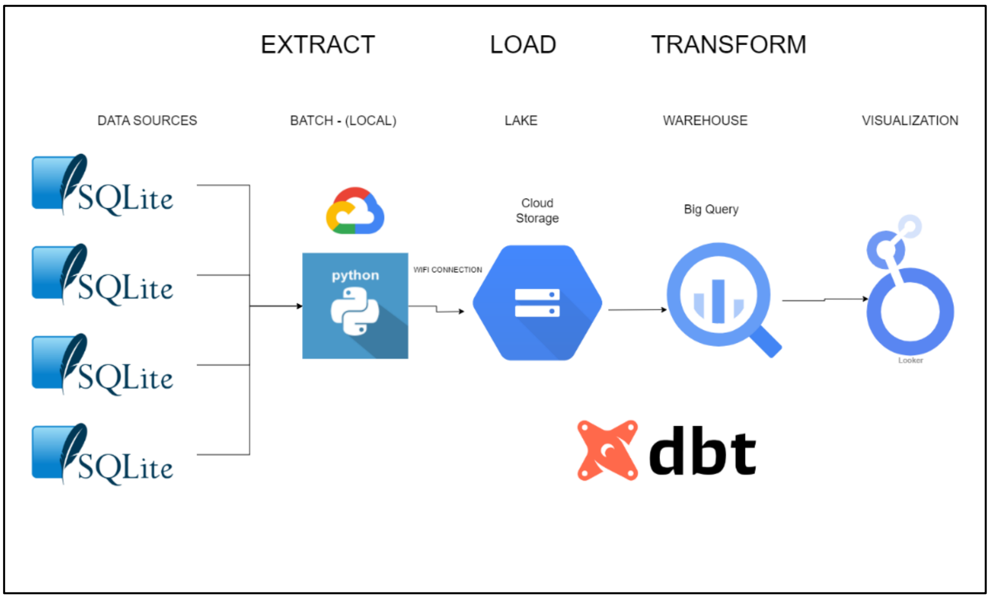


FIGURE 11 – ELT Pipeline in GCP.

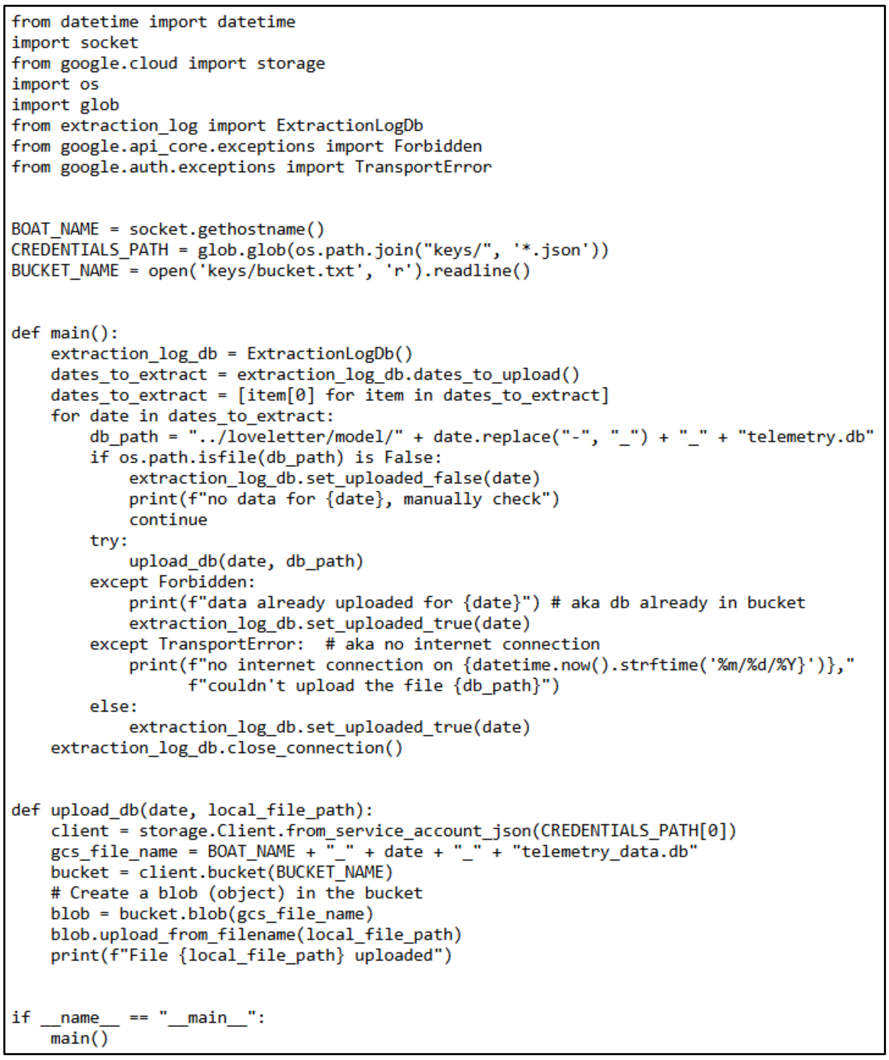
The data architecture has been defined; the next section covers the development of the artifact.

4.3.3. Development

As mentioned in the architecture, the idea is to load different datasets directly into GCS, as this serves the purpose of a Data Lake.

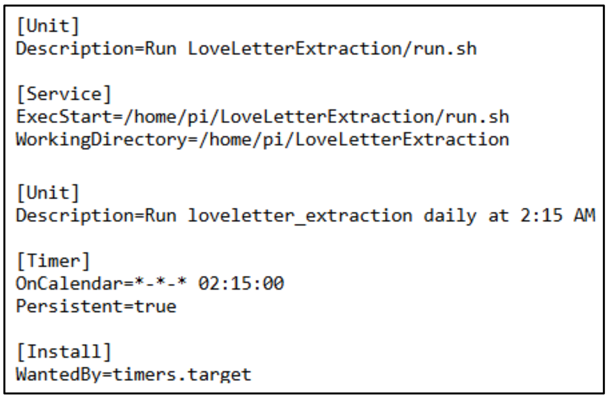
For this project, the only data sources are ephemeral SQLite databases generated daily on each boat. These ephemeral databases must be extracted from every boat and loaded directly into a bucket within GCS.

FIGURE 12 shows the script used to extract the ephemeral databases used for the main application that logs the telemetry data on every boat. This whole ephemeral database is then loaded directly into a bucket previously defined. This is done with a Service Account. This process is independent in every boat, but all of them share a copy of the same logging application and the same service account. This process of extraction and loading is done every day at 2:15 am, controlled automatically by a service file and a timer, as seen in FIGURE 13.



```
from datetime import datetime
import socket
from google.cloud import storage
import os
import glob
from extraction_log import ExtractionLogDb
from google.api_core.exceptions import Forbidden
from google.auth.exceptions import TransportError


BOAT_NAME = socket.gethostname()
CREDENTIALS_PATH = glob.glob(os.path.join("keys/", '*.json'))
BUCKET_NAME = open('keys/bucket.txt', 'r').readline()


def main():
    extraction_log_db = ExtractionLogDb()
    dates_to_extract = extraction_log_db.dates_to_upload()
    dates_to_extract = [item[0] for item in dates_to_extract]
    for date in dates_to_extract:
        db_path = "../loveletter/model/" + date.replace("-", "_") + "_" + "telemetry.db"
        if os.path.isfile(db_path) is False:
            extraction_log_db.set_uploaded_false(date)
            print(f"no data for {date}, manually check")
            continue
        try:
            upload_db(date, db_path)
        except Forbidden:
            print(f"data already uploaded for {date}") # aka db already in bucket
            extraction_log_db.set_uploaded_true(date)
        except TransportError:  # aka no internet connection
            print(f"no internet connection on {datetime.now().strftime('%m/%d/%Y')},"
                  f"couldn't upload the file {db_path}")
        else:
            extraction_log_db.set_uploaded_true(date)
    extraction_log_db.close_connection()


def upload_db(date, local_file_path):
    client = storage.Client.from_service_account_json(CREDENTIALS_PATH[0])
    gcs_file_name = BOAT_NAME + "_" + date + "_" + "telemetry_data.db"
    bucket = client.bucket(BUCKET_NAME)
    # Create a blob (object) in the bucket
    blob = bucket.blob(gcs_file_name)
    blob.upload_from_filename(local_file_path)
    print(f"File {local_file_path} uploaded")


if __name__ == "__main__":
    main()
```

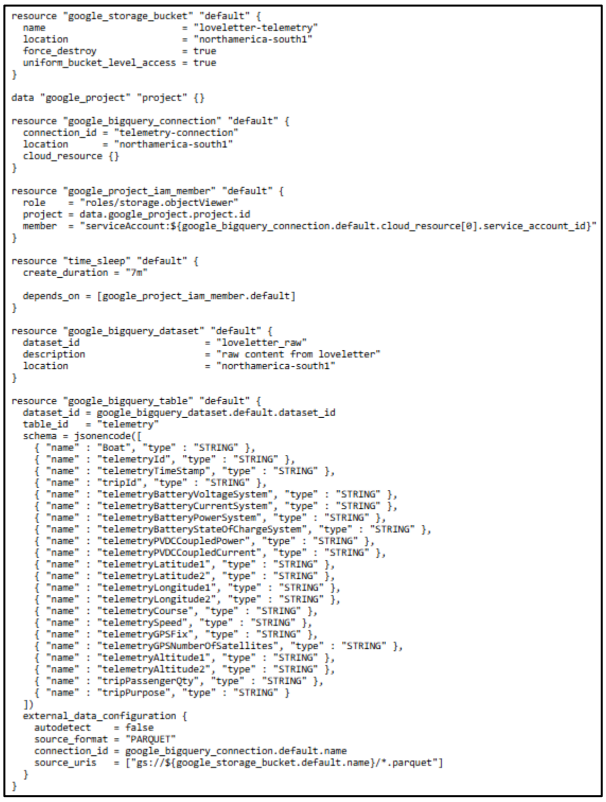
FIGURE 12 – Extract and Load Script.



```
[Unit]
Description=Run LoveLetterExtraction/run.sh

[Service]
ExecStart=/home/pi/LoveLetterExtraction/run.sh
WorkingDirectory=/home/pi/LoveLetterExtraction

[Unit]
Description=Run loveletter_extraction daily at 2:15 AM

[Timer]
OnCalendar=*-*-* 02:15:00
Persistent=true

[Install]
WantedBy=timers.target
```

FIGURE 13 – Service File and Timer.

Once a new object is loaded into this bucket, a cloud function is triggered to convert the ephemeral dataset into an Apache Parquet format that is appropriate for BigQuery.



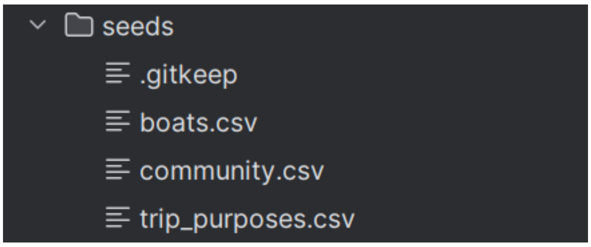
```
resource "google_storage_bucket" "default" {
  name                        = "loveletter-telemetry"
  location                    = "northamerica-south1"
  force_destroy               = true
  uniform_bucket_level_access = true
}

data "google_project" "project" {}

resource "google_bigquery_connection" "default" {
  connection_id = "telemetry-connection"
  location      = "northamerica-south1"
  cloud_resource {}
}

resource "google_project_iam_member" "default" {
  role    = "roles/storage.objectViewer"
  project = data.google_project.project.id
  member  = "serviceAccount:${google_bigquery_connection.default.cloud_resource[0].service_account_id}"
}

resource "time_sleep" "default" {
  create_duration = "7m"

  depends_on = [google_project_iam_member.default]
}

resource "google_bigquery_dataset" "default" {
  dataset_id                      = "loveletter_raw"
  description                     = "raw content from loveletter"
  location                        = "northamerica-south1"
}

resource "google_bigquery_table" "default" {
  dataset_id = google_bigquery_dataset.default.dataset_id
  table_id   = "telemetry"
  schema = jsonencode([
    { "name" : "Boat", "type" : "STRING" },
    { "name" : "telemetryId", "type" : "STRING" },
    { "name" : "telemetryTimeStamp", "type" : "STRING" },
    { "name" : "tripId", "type" : "STRING" },
    { "name" : "telemetryBatteryVoltageSystem", "type" : "STRING" },
    { "name" : "telemetryBatteryCurrentSystem", "type" : "STRING" },
    { "name" : "telemetryBatteryPowerSystem", "type" : "STRING" },
    { "name" : "telemetryBatteryStateOfChargeSystem", "type" : "STRING" },
    { "name" : "telemetryPVDCCoupledPower", "type" : "STRING" },
    { "name" : "telemetryPVDCCoupledCurrent", "type" : "STRING" },
    { "name" : "telemetryLatitude1", "type" : "STRING" },
    { "name" : "telemetryLatitude2", "type" : "STRING" },
    { "name" : "telemetryLongitude1", "type" : "STRING" },
    { "name" : "telemetryLongitude2", "type" : "STRING" },
    { "name" : "telemetryCourse", "type" : "STRING" },
    { "name" : "telemetrySpeed", "type" : "STRING" },
    { "name" : "telemetryGPSFix", "type" : "STRING" },
    { "name" : "telemetryGPSNumberOfSatellites", "type" : "STRING" },
    { "name" : "telemetryAltitude1", "type" : "STRING" },
    { "name" : "telemetryAltitude2", "type" : "STRING" },
    { "name" : "tripPassengerQty", "type" : "STRING" },
    { "name" : "tripPurpose", "type" : "STRING" }
  ])
  external_data_configuration {
    autodetect    = false
    source_format = "PARQUET"
    connection_id = google_bigquery_connection.default.name
    source_uris   = ["gs://${google_storage_bucket.default.name}/*.parquet"]
  }
}
```

FIGURE 14 - Big Lake Table.

The Data Lake can be used directly in BigQuery as a raw source with a Big Lake External Table, FIGURE 14 shows the creation of the external table called telemetry under the “loveletter_raw” schema. Everything is created with Terraform to allow for version control. Now that the Data Lake is appropriately configured, the extract and load process is concluded, and we can proceed to the data transformation in the DW.

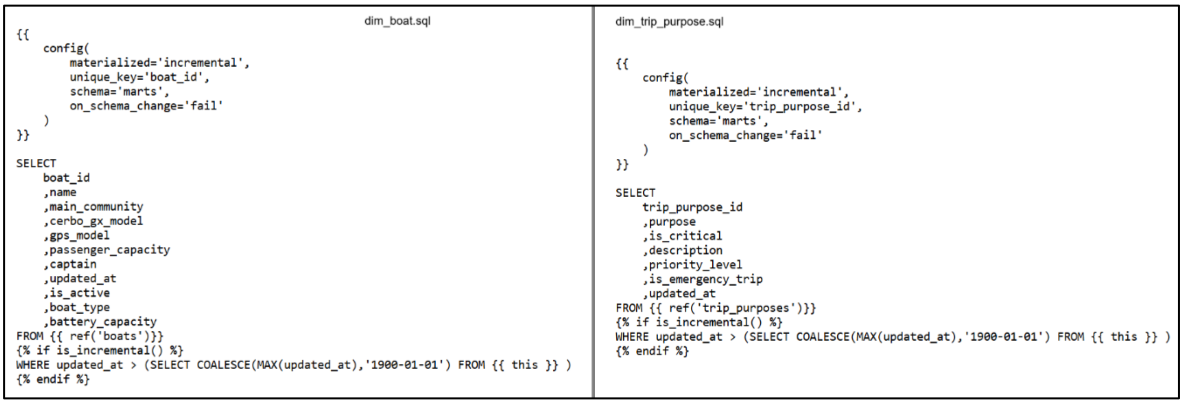
The transformation process will be carried out with DBT (data build tool), which is an industry gold standard tool for data transformation. The main advantage of this tool is that it seamlessly transforms raw data into marts while allowing for versioning control, reusability, testing, and scalability. Every sql file is treated as a model that can be referenced.

The data transformation begins with the dimensions, as this must be built first, before any fact. The raw source of the dimensions is DBT seeds. This is static information. In this project, the main system is not an integrated one; therefore, it does not manage information about boats or communities. For this project, the boats, communities, and trip purposes are all static information. DBT offers a way to handle this with seeds and CSV files that are versioned and controlled by git. When creating a seed, the DBT will upload this as tables within the DW, as seen in FIGURE 15.



```
seeds
    .gitkeep
    boats.csv
    community.csv
    trip_purposes.csv
```

FIGURE 15 - Seeds within Kara Solar's DBT repository.



dim_boat.sql

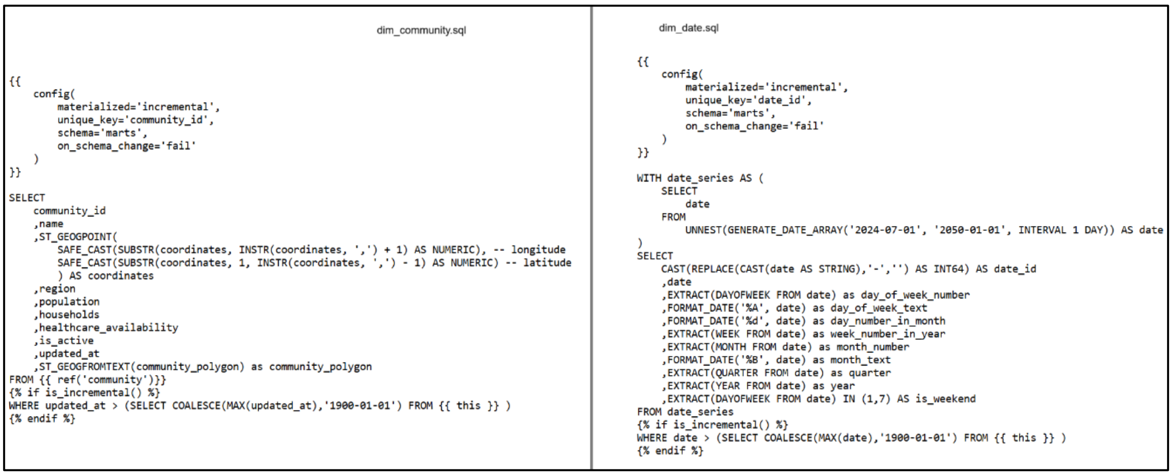
```
{{
    config(
        materialized='incremental',
        unique_key='boat_id',
        schema='marts',
        on_schema_change='fail'
    )
}}

SELECT
    boat_id
    ,name
    ,main_community
    ,cerbo_gx_model
    ,gps_model
    ,passenger_capacity
    ,captain
    ,updated_at
    ,is_active
    ,boat_type
    ,battery_capacity
FROM {{ ref('boats')}}
{% if is_incremental() %}
WHERE updated_at > (SELECT COALESCE(MAX(updated_at),'1900-01-01') FROM {{ this }} )
{% endif %}
```

dim_trip_purpose.sql

```
{{
    config(
        materialized='incremental',
        unique_key='trip_purpose_id',
        schema='marts',
        on_schema_change='fail'
    )
}}

SELECT
    trip_purpose_id
    ,purpose
    ,is_critical
    ,description
    ,priority_level
    ,is_emergency_trip
    ,updated_at
FROM {{ ref('trip_purposes')}}
{% if is_incremental() %}
WHERE updated_at > (SELECT COALESCE(MAX(updated_at),'1900-01-01') FROM {{ this }} )
{% endif %}
```

FIGURE 16 – Dimension boat and trip purpose transformation.

The models used to create the dimensions of the boat and the purpose of the trip can be seen in FIGURE 16. Both models are being materialized as incremental tables, meaning that only when a change occurs in the underlying seed in the “updated_at” column will these tables be updated. This type of materialization is useful for saving computing resources and allowing the BI tool to process them quickly. These models represent the DataMart defined before within the DW. FIGURE 17 shows the transformations carried out to populate and create the table's dimension, community, and date. As can be seen, the transformations are simple due to the seeds used as source.



dim_community.sql

```
{{
    config(
        materialized='incremental',
        unique_key='community_id',
        schema='marts',
        on_schema_change='fail'
    )
}}

SELECT
    community_id
    ,name
    ,ST_GEOGPOINT(
        SAFE_CAST(SUBSTR(coordinates, INSTR(coordinates, ',') + 1) AS NUMERIC), -- longitude
        SAFE_CAST(SUBSTR(coordinates, 1, INSTR(coordinates, ',') - 1) AS NUMERIC) -- latitude
        ) AS coordinates
    ,region
    ,population
    ,households
    ,healthcare_availability
    ,is_active
    ,updated_at
    ,ST_GEOGFROMTEXT(community_polygon) as community_polygon
FROM {{ ref('community')}}
{% if is_incremental() %}
WHERE updated_at > (SELECT COALESCE(MAX(updated_at),'1900-01-01') FROM {{ this }} )
{% endif %}
```

dim_date.sql

```
{{
    config(
        materialized='incremental',
        unique_key='date_id',
        schema='marts',
        on_schema_change='fail'
    )
}}

WITH date_series AS (
    SELECT
        date
    FROM
        UNNEST(GENERATE_DATE_ARRAY('2024-07-01', '2050-01-01', INTERVAL 1 DAY)) AS date
)
SELECT
    CAST(REPLACE(CAST(date AS STRING),'-','') AS INT64) AS date_id
    ,date
    ,EXTRACT(DAYOFWEEK FROM date) as day_of_week_number
    ,FORMAT_DATE('%A', date) as day_of_week_text
    ,FORMAT_DATE('%d', date) as day_number_in_month
    ,EXTRACT(WEEK FROM date) as week_number_in_year
    ,EXTRACT(MONTH FROM date) as month_number
    ,FORMAT_DATE('%B', date) as month_text
    ,EXTRACT(QUARTER FROM date) as quarter
    ,EXTRACT(YEAR FROM date) as year
    ,EXTRACT(DAYOFWEEK FROM date) IN (1,7) AS is_weekend
FROM date_series
{% if is_incremental() %}
WHERE date > (SELECT COALESCE(MAX(date),'1900-01-01') FROM {{ this }} )
{% endif %}
```

FIGURE 17 - Dimension community and data transformation.

The data transformation from the raw table loaded from the system requires a deeper explanation; the raw source table “telemetry” needs to be cast and transformed. This is done with a staging model called stg_telemetry, which contains the same granularity as the raw table. After transformation and casting, this table is filtered to contain only the trips taken, not the entire telemetry, and is grouped by trip and boat. In this step, most of the metrics are calculated. This is done in the model int_trip. Finally, within the Marts, the fact_trip table is created, which contains the foreign keys to every dimension in the mart. This transformation process is seen in FIGURE 18.

The code to make such transformations takes lots of space, so it’s not included in this written report but is available to everyone in the following repository: “github.com/KaraSolar/love_letter_analytics.git”. This concludes the DW creation and the physical implementation of the Social Impact Data Mart defined in FIGURE 9.

As mentioned before, to orchestrate the whole pipeline, this project is using the DBT cloud, as there is no need for a more advanced orchestration tool such as Apache Airflow right now. The final component involved in this artifact is the dashboard. This will be discussed in the following section, the artifact demonstration, as this constitutes just a front-end or presentation layer of the whole artifact.

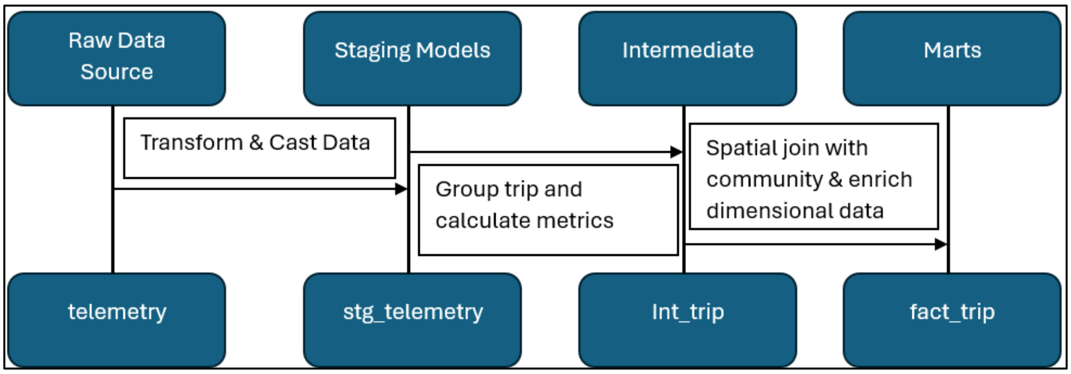
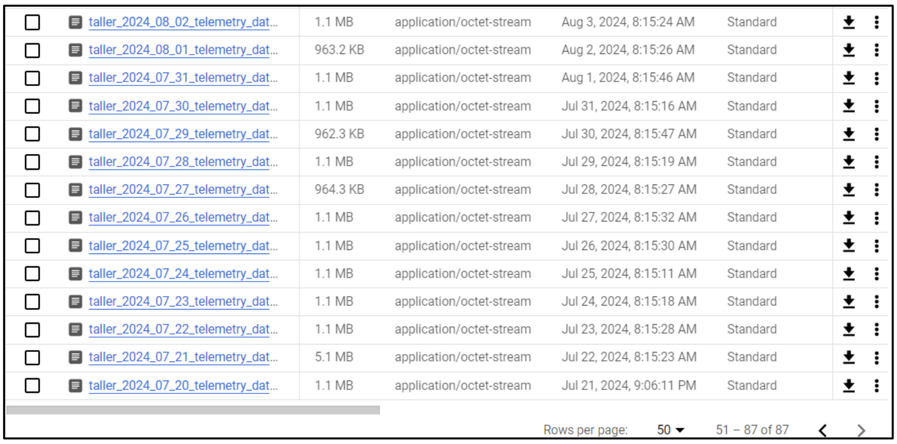


FIGURE 18 – Fact trip transformation process.

4.4 Demonstration

This section will discuss and demonstrate the artifact. The first thing discussed within the artifact is the extraction and loading of the telemetry data collected by the boats. It is important to mention that the demonstration was taken from the quality assurance (QA) environment of Kara Solar. All the figures shown of the artifact are taken from this QA environment, not from production data, as this has not yet been released to the public, only the source code.



□	taller_2024_08_02_telemetry_dat...	1.1 MB	application/octet-stream	Aug 3, 2024, 8:15:24 AM	Standard
□	taller_2024_08_01_telemetry_dat...	963.2 KB	application/octet-stream	Aug 2, 2024, 8:15:26 AM	Standard
□	taller_2024_07_31_telemetry_dat...	1.1 MB	application/octet-stream	Aug 1, 2024, 8:15:46 AM	Standard
□	taller_2024_07_30_telemetry_dat...	1.1 MB	application/octet-stream	Jul 31, 2024, 8:15:16 AM	Standard
□	taller_2024_07_29_telemetry_dat...	962.3 KB	application/octet-stream	Jul 30, 2024, 8:15:47 AM	Standard
□	taller_2024_07_28_telemetry_dat...	1.1 MB	application/octet-stream	Jul 29, 2024, 8:15:19 AM	Standard
□	taller_2024_07_27_telemetry_dat...	964.3 KB	application/octet-stream	Jul 28, 2024, 8:15:27 AM	Standard
□	taller_2024_07_26_telemetry_dat...	1.1 MB	application/octet-stream	Jul 27, 2024, 8:15:32 AM	Standard
□	taller_2024_07_25_telemetry_dat...	1.1 MB	application/octet-stream	Jul 26, 2024, 8:15:30 AM	Standard
□	taller_2024_07_24_telemetry_dat...	1.1 MB	application/octet-stream	Jul 25, 2024, 8:15:11 AM	Standard
□	taller_2024_07_23_telemetry_dat...	1.1 MB	application/octet-stream	Jul 24, 2024, 8:15:18 AM	Standard
□	taller_2024_07_22_telemetry_dat...	1.1 MB	application/octet-stream	Jul 23, 2024, 8:15:28 AM	Standard
□	taller_2024_07_21_telemetry_dat...	5.1 MB	application/octet-stream	Jul 22, 2024, 8:15:23 AM	Standard
□	taller_2024_07_20_telemetry_dat...	1.1 MB	application/octet-stream	Jul 21, 2024, 9:06:11 PM	Standard

Rows per page: 50 ▾ 51 – 87 of 87 ‹ ›

FIGURE 19 – Extract & Load Demonstration.

The uploading process is shown in FIGURE 19. This is the GCP bucket that stores the telemetry data. This snapshot was taken from +5 UTC time, but when translating to Ecuador, the time zone was exactly 2:15 am. The extraction process is working properly, all the data is being uploaded correctly, and the uploading process registers are being stored in the appropriate logs.

The second part of the loading process is the creation of a Big Lake table with terraform within the Kara Solar infrastructure in GCP. As seen in FIGURE 20, the configuration made with Terraform is working properly, and a Big Lake table pointing to the appropriate bucket was created with no errors within the qa project and loveletter_raw schema.

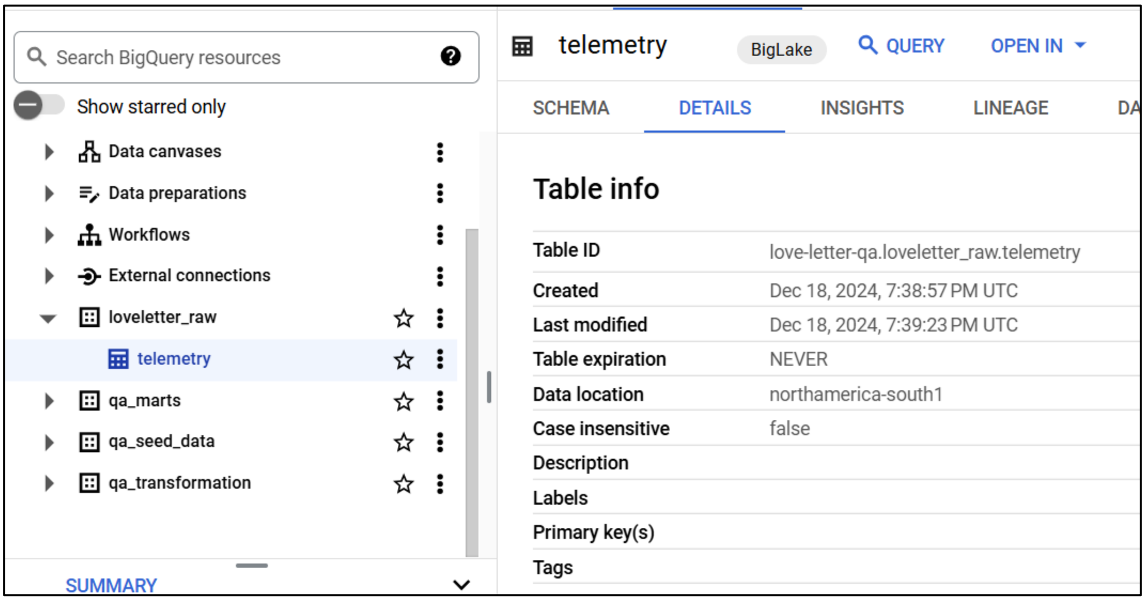
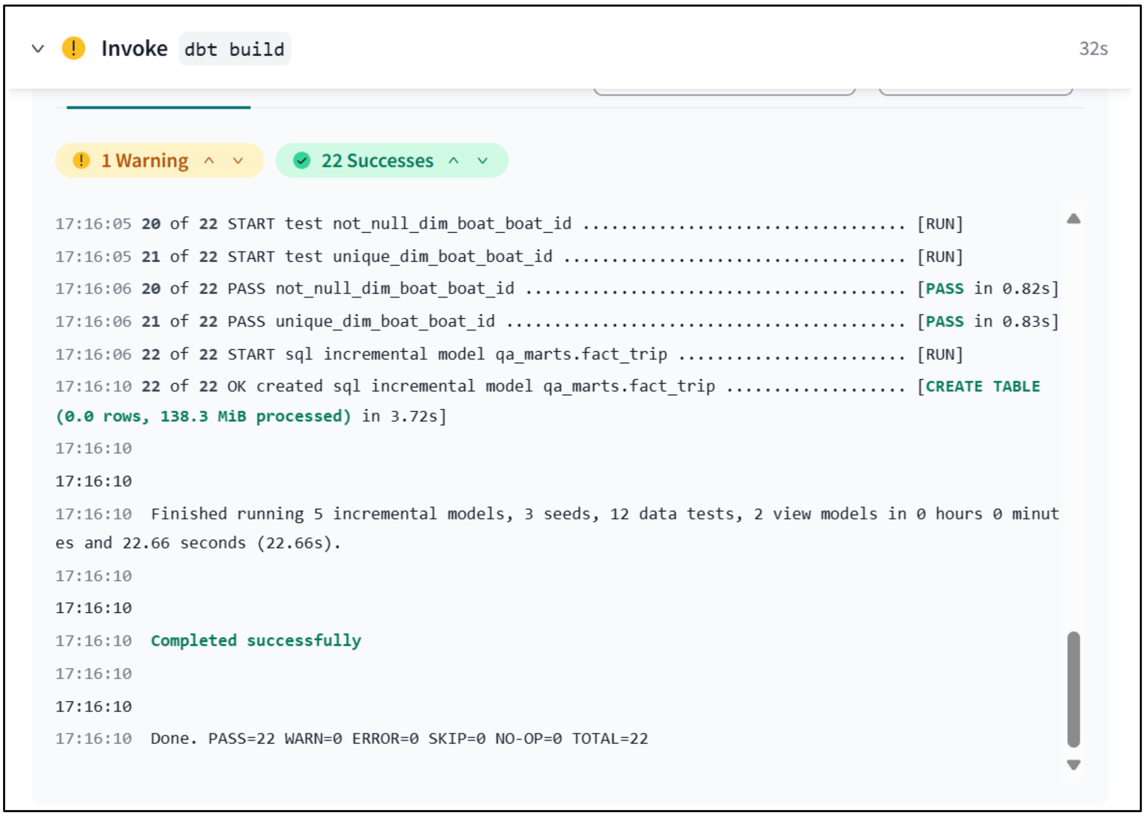


Table info

Table ID	love-letter-qa.loveletter_raw.telemetry
Created	Dec 18, 2024, 7:38:57 PM UTC
Last modified	Dec 18, 2024, 7:39:23 PM UTC
Table expiration	NEVER
Data location	northamerica-south1
Case insensitive	false
Description	
Labels	
Primary key(s)	
Tags	

FIGURE 20 – Big Lake table.

The next step is to look at the orchestrator within the DBT cloud to see if the models are being created correctly and the tests are being passed. FIGURE 21 shows the orchestrator logs. All the models and tests were passed correctly without any errors. Five incremental models were built, four dimensions and one fact table, three seeds as mentioned above, 12 tests to evaluate the correctness of the transformation process, and two views corresponding to the staging and intermediate models.



```
17:16:05 20 of 22 START test not_null_dim_boat_boat_id .............................. [RUN]
17:16:05 21 of 22 START test unique_dim_boat_boat_id ................................ [RUN]
17:16:06 20 of 22 PASS not_null_dim_boat_boat_id .................................... [PASS in 0.82s]
17:16:06 21 of 22 PASS unique_dim_boat_boat_id ...................................... [PASS in 0.83s]
17:16:06 22 of 22 START sql incremental model qa_marts.fact_trip ..................... [RUN]
17:16:10 22 of 22 OK created sql incremental model qa_marts.fact_trip ................ [CREATE TABLE
(0.0 rows, 138.3 MiB processed) in 3.72s]
17:16:10
17:16:10
17:16:10  Finished running 5 incremental models, 3 seeds, 12 data tests, 2 view models in 0 hours 0 minut
es and 22.66 seconds (22.66s).
17:16:10
17:16:10
17:16:10  Completed successfully
17:16:10
17:16:10
17:16:10  Done. PASS=22 WARN=0 ERROR=0 SKIP=0 NO-OP=0 TOTAL=22
```

FIGURE 21 – Orchestrator Logs.

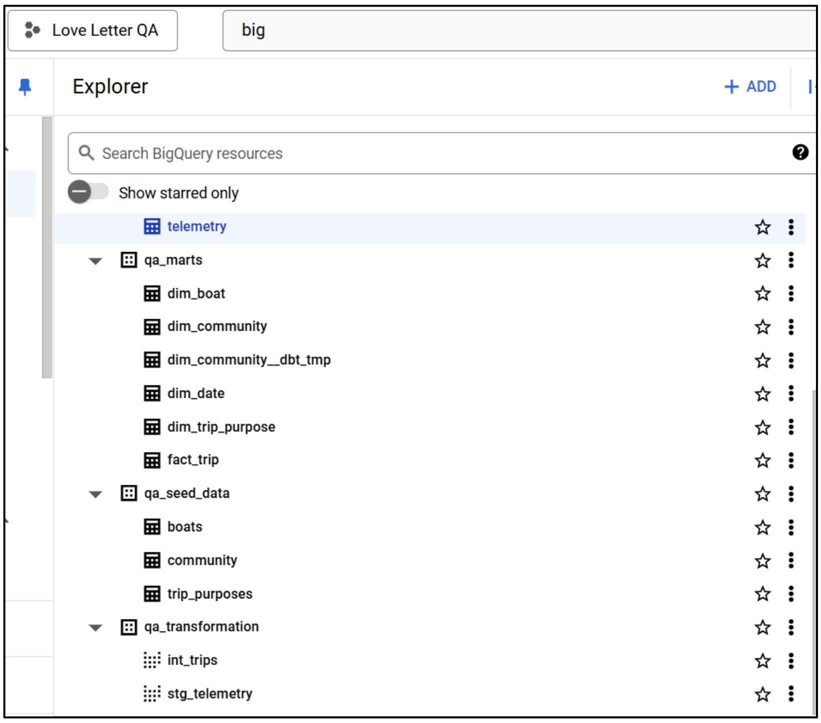
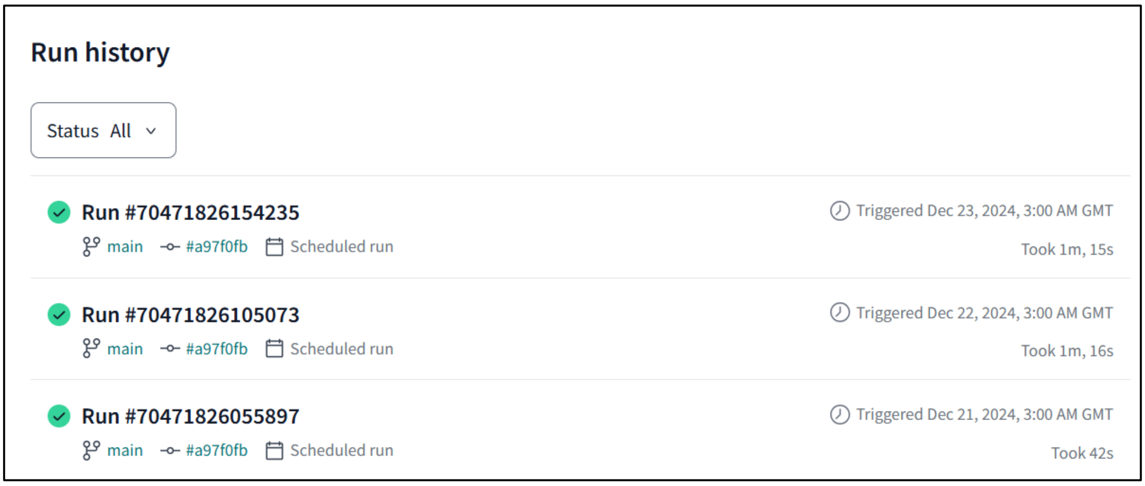


FIGURE 22 – Kara Solar Data Warehouse

A visual inspection can be seen in FIGURE 22; the tables, views, and seeds were created successfully, and this ELT process will continue every day. The orchestrator logs can be inspected to check for errors within the pipeline. FIGURE 23 shows the orchestrator's recent runs within DBT cloud. The job is running every day without any issues.



Run history

Status All

✓ Run #70471826154235 Triggered Dec 23, 2024, 3:00 AM GMT
main #a97f0fb Scheduled run Took 1m, 15s

✓ Run #70471826105073 Triggered Dec 22, 2024, 3:00 AM GMT
main #a97f0fb Scheduled run Took 1m, 16s

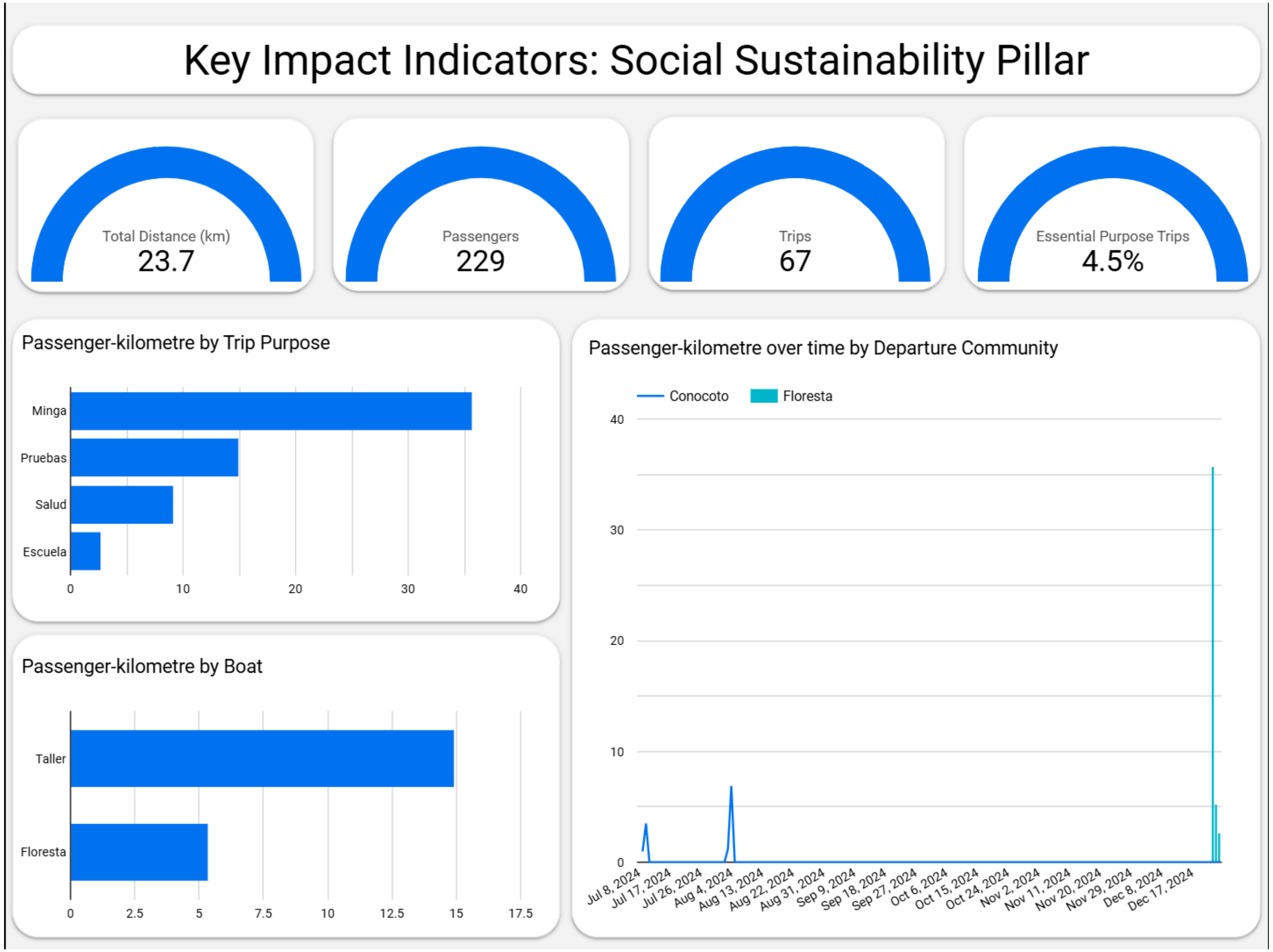
✓ Run #70471826055897 Triggered Dec 21, 2024, 3:00 AM GMT
main #a97f0fb Scheduled run Took 42s

FIGURE 23 – Orchestrator jobs run.

The final step is to construct a dashboard to show the KIIs that Kara Solar has defined to measure its social sustainability impact. It is important to note that the images shown on the dashboard correspond to test data and do not reflect the data contained in the production environment.

FIGURE 24 and FIGURE 25 show the first and second pages of the Key Impact Indicators dashboard. As seen in the figures, the dashboard shows all the important KIIs defined. The total distance traveled by boats in km, the total number of passengers, the total number of trips, and the number of essential purpose trips as a percentage of the total number of trips. The most important metric is the passenger-kilometer. This metric is shown by trip purpose, by boat, and the evolution over time by departing the community.

In the second page of the dashboard, there are three KIIs: the total energy consumed in kWh, the average trip duration, and the energy generated by the solar panels on the roof of the boat during a particular trip. Finally, the last portion of the dashboard shows an interactive map of the routes taken by the boats, color-coded by different trips. This dashboard, if fully interactive, can show these metrics by any dimension shown in the dimensional modeling section. This concludes the demonstration of the artifact.



Key Impact Indicators: Social Sustainability Pillar

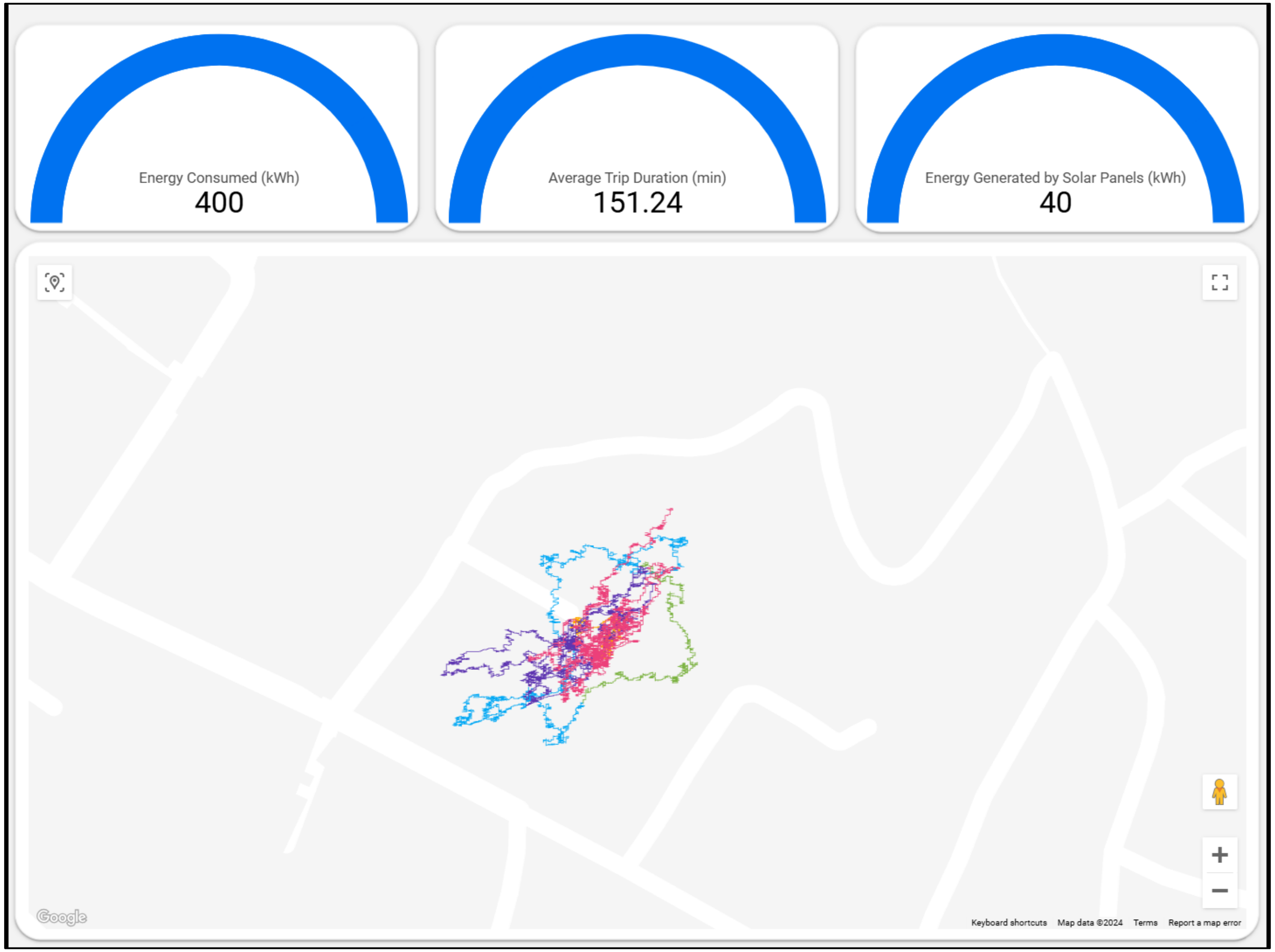
Total Distance (km) 23.7

Passengers 229

Trips 67

Essential Purpose Trips 4.5%

FIGURE 24 - Dashboard: Key Impact Indicators of Social Sustainability Pillar, page 1.



Energy Consumed (kWh)
400

Average Trip Duration (min)
151.24

Energy Generated by Solar Panels (kWh)
40

FIGURE 25 - Dashboard: Key Impact Indicators of Social Sustainability Pillar, page 2.

4.5 Evaluation

The evaluation of the artifact is given by the Goal dimension discussed in the methodological approach section. This dimension has three defined criteria: efficacy, validity, and generality, with efficacy being the main criterion. The criteria can only be evaluated as pass or fail, as it is not possible to assign an objective scoring system.

The first criterion is efficacy. The question to be asked is: Does the data pipeline artifact achieve its goal of collecting, storing, transforming, and analyzing data to measure Kara Solar's social impact? From the previous section, the demonstration of the artifact is clear that the pipeline has high efficacy as it collects, stores, transforms, and analyses data to measure social impact.

The pipeline is pulling data from various boats, each boat with multiple sensors in a continuous matter. After the collection, the raw sources are in a Data Lake. Then, the pipeline transforms the data from raw sources with DBT into usable formats for the business. All the transformations are being done in a DW. Finally, the pipeline enables analytics with the dashboards shown in FIGURE 24 and FIGURE 25.

The pipeline is working as planned, and this can be seen in the run history of the jobs in the orchestrator in FIGURE 25. Given the demonstration of the artifact and the definition of these criteria, a Passing score is assigned to the efficacy criteria.

The next criterion is validity. The question is: Does the data pipeline artifact perform its tasks correctly and consistently? The answer to this question again relies on the demonstration of the artifact, especially in FIGURE 23, which shows the completion of the orchestrator jobs. The pipeline correctly does all its tasks consistently; therefore, a passing score is assigned to validity criteria.

Finally, the last criterion is generality. The question to be asked is: Can the data pipeline artifact handle different types of data and adapt to changes in Kara Solar's operations? One of the main goals when designing the artifact was to keep scalability in mind. It is important to note that the Data Lake and the pipeline, in general, can handle not only the telemetry data that is being collected, stored, transformed, and analyzed but in the medium term, Kara Solar will introduce different data sources such as acoustic sensors (non-structed data), the pipeline is more than capable of handling this different data types.

Another important aspect is to keep in mind adaptability. This is where DBT and an ELT design play a pivotal role, as this allows continuous development and continuous integration, schema changes, data realities, and necessities changes. DBT and ELT design allows for faster development and adaptation to the necessities of Kara Solar.

Another important aspect is scalability. Since this project deals with telemetry data from IoT devices, it must handle big data. This is why the services of Google Cloud Platform were selected. BigQuery allows the processing of massive amounts of data at a low cost. This is part of scalability.

Finally, an important consideration in generality is vendor lock-in. When using public cloud services, there is always a risk of vendor lock-in, making migration to a private cloud or another public cloud costly. To reduce this vendor lock-in risk, this pipeline artifact uses DBT, which allows for dynamical referencing and other features, making it less prone to vendor lock-in. For these reasons, a Passing grade is assigned to generality criteria.

All three criteria are assigned a Passing score; therefore, the goal dimension is also given a Passing score. Since this is the first instantiation of the problem and no benchmark exists for this specific problem, the evaluation process finalizes with a Passing Score for this data pipeline artifact, as it solves the problem with efficacy, validity, and generality.

Not only is a goal dimension and its evaluation criteria important, but a deep analysis of the artifact's strengths, weaknesses, opportunities, and threats can also be found in Table II.

Table II

SWOT Analysis of the Artifact.

SWOT Matrix	
Strengths	Weaknesses

- Scalability 5/5: the usage of BigQuery and DBT allows for scalability. The incremental strategy and partitioning avoid full table scans, making the ELT process fast and scalable.
- Modularity 4/5: the separation between staging, intermediate models, and DataMart allows the encapsulation of the logic into separate modules, making it easier to maintain.
- Flexible Architecture 5/5: The usage of modern tools and Data Lake makes the auditing process easier as every record and table has lineage. The Data Lake makes it easy to adapt the schema to changes in the organization analysis requirements and change.
- Cloud Agnostic Artifact 4/5: The reduced dependency on Google Cloud Platform's microservices allows for easy migration to another public cloud provider or own on-premise infrastructure. BigQuery can be replaced by Amazon Redshift, Snowflake, or PostgreSQL, whereas GCS can be replaced by an SFTP or similar.
- Complexity 3/5: the artifact requires a deep understanding of the underlying data and Modbus communication protocol to maintain the pipeline, as well as a deep understanding of DBT, data warehousing, cloud technologies, and geospatial data.
- Dependencies 3/5: the artifact relies heavily on reliable internet connection in the Amazon rainforest, which might not always be the case. It also relies heavily on tools such as DBT and cloud computing power.
- Data Quality 2/5: given the reliance on sensors and GPS signals, it is difficult to detect incorrect records and misreads from the sensors. Therefore, the data quality, despite the best cleaning efforts, might not be enough.
- Limited Testing 3/5: Some corner cases might not have been covered due to data availability, especially in the data cleaning process. There can be incorrect sensor readings that have not been covered yet in the testing process within DBT models.

Opportunities	Threats
• Data Sources Integration 5/5: the artifact is designed to handle multiple data source ingestion. Kara Solar will soon include acoustic sensors to measure wildlife. The artifact can handle the ingestion and processing of virtually any data source. • Advanced Analytics 5/5: the artifact enables advanced analytics, such as business intelligence, but can easily accommodate more advanced analytics, such as prediction and machine learning. • Community Support 4/5: Using open-source tools such as DBT and the open-source nature of Kara Solar's codebase allows for community support and possibly community collaboration on the code base. • Funding Opportunities 4/5: the possibility of measuring Kara Solar's social impact opens the possibility for further funding of Kara Solar's operations and research projects with the data collected.	• Budget Constraints 3/5: limited costs to cloud computing can present a threat if the data volume increases to the petabytes of data. • Tool dependencies 2/5: DBT is an open-source tool, but in the future, it might become a paid tool for development. • Changing requirements 4/5: Enterprises evolve and change over time; the artifact needs to keep evolving and adapting to changing requirements.

5. Conclusion

This Master's final work aimed to create a data pipeline artifact that facilitates the collection, storage, transformation, and analysis of Kara Solar's data, enabling the measurement of its impact on the social sustainability pillar via proxy variables and KIIs.

The artifact achieves its goal because it solves the problem with efficacy, validity, and generality. The artifact is useful for collecting, storing, transforming, and analyzing data to measure Kara Solar's social impact; it's important to note that the artifact is performing its tasks correctly and consistently daily. Finally, the artifact is generalist enough to accommodate different types of data, expand with more data sources, and be scalable.

Kara Solar will continue to expand its data sources; therefore, it needs flexibility in the Data Warehouse approach. That's the reason why a Kimball approach to DW is more beneficial than an Inmon approach. The development is faster, it adapts quicker to business changes, and it is more flexible overall.

Following this need for flexibility, an ELT rather than an ETL approach for the pipeline was followed, allowing for more flexibility and faster development. It allows for non-structured data processing, which will be proven to be useful in the future. Another advantage is to keep track of data lineage as the transformations are centralized.

Finally, it was observed that the evaluation process of the artifact lacks a benchmark. Since this is the first implementation of the artifact and the first solution to the problem, it is not possible to set up a better evaluation process for the artifact.

In the future, when another artifact is developed to solve this problem, it is important to evaluate the new artifact by a benchmark if the performance is better than the implementation of this Master's final work.

REFERENCES

Amazon Web Services. (n.d.). *What's the Difference Between ETL and ELT?* Retrieved October 13, 2024, from https://aws.amazon.com/compare/the-difference-between-etl-and-elt/

Aparicio, J. T., Aparicio, M., & Costa, C. J. (2023). Design Science in Information Systems and Computing. *Proceedings of International Conference on Information Technology and Applications: ICITA 2022*, 409–419. https://doi.org/10.1007/978-981-19-9331-2_35

Barber, C. P., Cochrane, M. A., Souza, C. M., & Laurance, W. F. (2014). Roads, deforestation, and the mitigating effect of protected areas in the Amazon. *Biological Conservation*, *177*, 203–209. https://doi.org/10.1016/j.biocon.2014.07.004

Ben-Gan, I. (2016). *T-SQL Fundamentals* (4th ed.). Microsoft Press.

Caetano, T. V., & Costa, C. J. (2014). Data Warehousing num contexto de Sistemas Integrados. *Atas Da Conferência Da Associação Portuguesa de Sistemas de Informação*, *12*, 186–199.

European Automobile Manufacturers' Association. (2023, May 18). *New EU bus sales by power source*. https://www.acea.auto/figure/buses-eu-fuel-type/

Google Cloud. (n.d.). *Data Warehouse Architecture on Google Cloud*. Google Cloud. Retrieved December 18, 2024, from https://cloud.google.com/architecture/big-data-analytics/data-warehouse?_gl=1*gjfvbo*_ga*NjAyOTEwMjA2LjE3MDkyMjczMTE.*_ga_W

H2QY8WWF5*MTczNDUzMzExMi4xNzguMS4xNzM0NTMzMTY2LjE5LjAuMA..#architecture

Handy, T. (2016, April 5). Building a Mature Analytics Workflow. *Dbt Labs*. https://www.getdbt.com/blog/building-a-mature-analytics-workflow

Hoffer, J. A., Ramesh, V., & Topi, H. (2016). *Modern database management* (13th ed.). Pearson.

IEA. (2024). Global EV Outlook 2024. *IEA, Paris*. https://www.iea.org/reports/global-ev-outlook-2024

Inmon, W. H. (2002). *Building the Data Warehouse* (3e ed.). John Wiley & Sons, Inc.

Kara Solar. (n.d.). About Us. *Kara Solar*. Retrieved October 5, 2024, from https://karasolar.com/who-we-are

Kimball, R., & Ross, M. (2013). *The data warehouse toolkit: The definitive guide to dimensional modeling* (3rd ed.). John Wiley & Sons.

Linstedt, D. (2002, July 1). Data Vault Series 1 – Data Vault Overview. *The Data Administration Newsletter*. https://tdan.com/data-vault-series-1-data-vault-overview/5054

Linstedt, D. (2003a, January 1). Data Vault Series 2 – Data Vault Components. *The Data Administration Newsletter*. https://tdan.com/data-vault-series-2-data-vault-components/5155

Linstedt, D. (2003b, January 4). Data Vault Series 3 – End Dates and Basic Joins. *The Data Administration Newsletter*. https://tdan.com/data-vault-series-3-end-dates-and-basic-joins/5067

Linstedt, D. (2004, January 1). Data Vault Series 4 – Link Tables. *The Data Administration Newsletter*. https://tdan.com/data-vault-series-4-link-tables/5172

Linstedt, D. (2005, January 1). Data Vault Series 5 – Loading Practices. *The Data Administration Newsletter*. https://tdan.com/data-vault-series-5-loading-practices/5285

Linstedt, D., & Olschimke, M. (2015). *Building a Scalable Data Warehouse with Data Vault 2.0* (p. 661).

PR Newswire. (2024, February 14). The Electrifying Rise of Electric Boats. *NetworkNewsWire Editorial Coverage*. https://www.prnewswire.com/news-releases/the-electrifying-rise-of-electric-boats-302061218.html#:~:text=The%20global%20electric%20boat%20market,to%20%2416.6%20billion%20by%202031.

Prat, N., Comyn-Wattiau, I., & Akoka, J. (2014). *Artifact evaluation in information systems design-science research–a holistic view*.

Purvis, B., Mao, Y., & Robinson, D. (2019). Three pillars of sustainability: In search of conceptual origins. *Sustainability Science*, *14*(3), 681–695. https://doi.org/10.1007/s11625-018-0627-5

Simon, H. A. (1988). The Science of Design: Creating the Artificial. *Design Issues*, *4*(1/2), 67–82. JSTOR. https://doi.org/10.2307/1511391

Stein, B., & Morrison, A. (2014). The enterprise data lake: Better integration and deeper analytics. *PwC Technology Forecast: Rethinking Integration*, *1*(1–9), 18. https://doi.org/10.12691/acis-3-1-3

Vilela, T., Malky Harb, A., Bruner, A., Laísa da Silva Arruda, V., Ribeiro, V., Auxiliadora Costa Alencar, A., Julissa Escobedo Grandez, A., Rojas, A., Laina, A., & Botero, R. (2020). A better Amazon road network for people and the environment. *Proceedings of the National Academy of Sciences*, *117*(13), 7095–7102. https://doi.org/10.1073/pnas.1910853117

Yessad, L., & Labiod, A. (2016). Comparative study of data warehouses modeling approaches: Inmon, Kimball and Data Vault. *2016 International Conference on System Reliability and Science (ICSRS)*, 95–99. https://doi.org/10.1109/ICSRS.2016.7815845