
# Implementing a data pipeline to fuel real-time visualization of power consumption in a set of port equipment

Pedro Otoya

Ruben Martinez

# Abstract

The main goal of the project in place is related to the technical implementation of a Data pipeline that will interpret, stream, transform and store data in real-time for analytics in a Dashboard. This tool will be provided to maintenance people working with a set of special port equipment that will help them enhance their way of working by delivering key indicators that can provide meaningful insights in real-time, this will be the first step for being able to enter the Big Data world that could lead to help them in the future to do more advanced things, such as predictive maintenance or machine learning. The data will be collected using a long polling mechanism and its currently being held in a memory schema that comes from a Siemens PLC system that needs to be accessed through a ModbusTCP/IP communication protocol in order to be displayed on the Dashboard for real-time and historical analysis.

In the first section of this paper, a general context on the theoretical background will be discussed starting with a general introduction to Big Data, including its definition along with how it has been changing through the years all the way to the most recent applications, where we will discuss its current landscape, this will help us understand how the proposed work its related to these topics. After this, a brief summary regarding the user of the dashboard along with the goals and requirements will be described.

The second section of the paper will deliver more information regarding some of the technologies used to build up the project with some basic but useful information on what are they used for and how it will be used throughout the entire project. This will be mostly divided into 3 technologies, streaming, storage and visualization which will be discussed further on in the paper.

The third section will discuss more on how all of the technological components along with the information extracted come in place. This will be the part where the overall system architecture will be discussed starting with data used to build up the dashboard along with some of the transformations done and the overall architecture of the system.

To conclude some of the next steps will be discussed that will help to be the base on what might come in the future and how this can be further improved or leveraged for future projects or analysis and to finalize overall mention some of the learnings through the duration of the whole project.

# Introduction

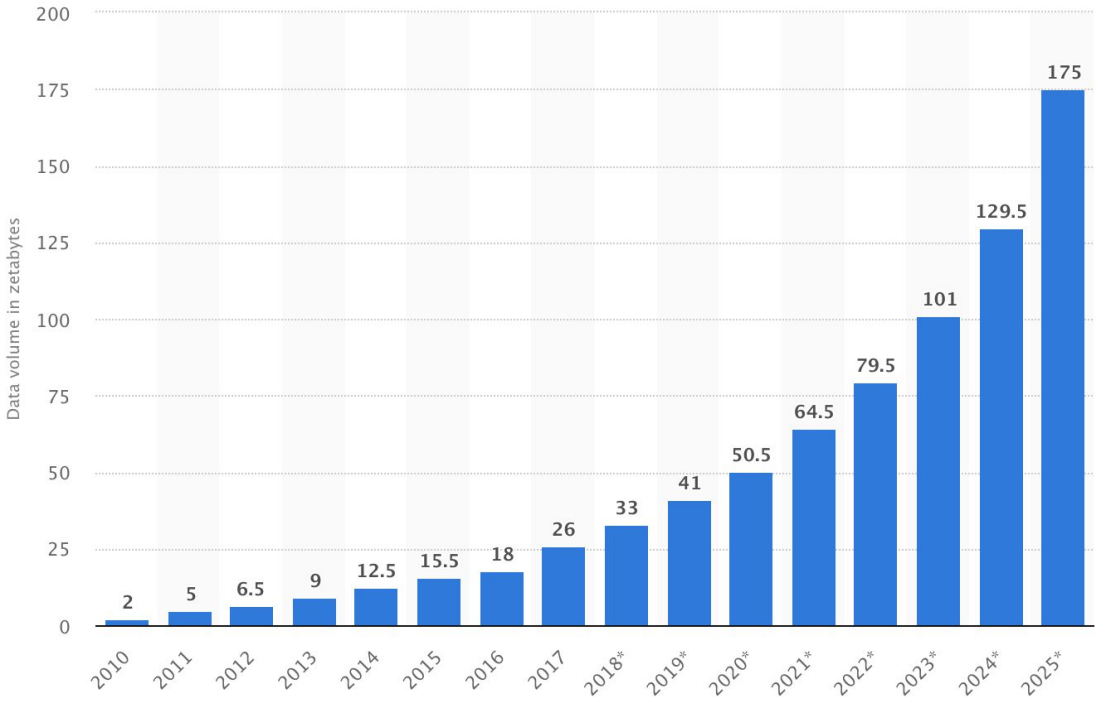
## What is Big Data?

Big Data has been defined in a very different way through the years as today it is still a very new concept for a lot of people. Due to its nature, it can be associated with many different things. Some people associated with information while others can associate it with the technologies that surround it just to name a few examples. Based on these two main points we can start by defining some of the characteristics surrounding Big Data as these are basic enablers of it. You need data to transform into information by leveraging specifics types of new technologies to derive value for a business.

As by today some of the most important institutions around the world try to give a proper definition but the term is so broad and it's being used along with other contexts that are hard to properly define it so we decided to choose one of the most simple yet describing definitions to this topic. Gartner, Inc, one of the leading research and advisory companies in the world tries to define Big Data along with its characteristics "Big data is high-volume, high-velocity and/or high-variety information assets that demand cost-effective, innovative forms of information processing that enable enhanced insight, decision making, and process automation"(Gartner) they mention what is considered to be the 3 V's of Big Data which basically refer to some of the most important characteristics associated with it. While several authors talk about more than 5 V's we would only focus on the 3 most mentioned and the original ones for the purpose of this definition.

Volume is related to the enormous size of the data as nowadays new information is generated each second by all of the different devices that surround us. It is reported that more than 41 zettabytes will be created by the end of the year according to sources like Statista which also provides a forecast that in the following years the data volume will grow to be as big as generating 175 zettabytes a year.

The chart below provides an overall view of how the data volume has been changing and will change throughout the years.




Variety refers to the different types of data that can be used it can be structured or unstructured or even a combination of both. Since now we have data generated from sources of all kinds from video to sensors or social media the variance of these sources can be very different hence it can become very complex to handle which makes it one of the most important characteristics of Big Data.

Velocity refers to the speed rate on which the data is created since a lot of information is now being generated in nearly real-time. Data being created has significantly increased its speed with the introduction of new technologies like smartphones and social networks which basically generate a new set of data in different forms in each second. This data generation can be very valuable to derive important insights into different industries.

The world of Big Data is mainly defined by these characteristics and it's important to say that these characteristics are pretty much dependent on each other as the likelihood of one changing basically can affect the others.

## Big Data in this project

Due to the overall nature of the project, this could be intrinsically related to Big Data as it contains some of the main characteristics of what normally Big Data represents.
Starting by the data producers who reside on the information that is created by the set of cranes. This data provides a structured format that can grow over time and it's being generated each second. Just by talking of this characteristic of the data source we can see that is related due to its nature with volume as it could exponentially growth overtime, variety as it can be later integrated with other types of information and velocity since its being generated in a short period of time because we are streaming it in real-time.

A data pipeline architecture pattern will help us to effectively be able to organize the data processing activities in order to move data in real-time from the data source to the data sinks which will serve as a layer that will power the visualization.

# Motivation

## Problem

The Port of Cartagena has a group of container cranes that produce data about their power consumption. Nonetheless, this data is not being collected and therefore not available for the port’s Maintenance department to monitor the power usage of their equipment. In this way, the port has no idea of how efficiently is energy being used and cannot take any measures to make further optimization in their electrical use.
As the port relies a lot on being constantly in operations, they currently own an agreement with their electric service provider in order to have a good quality of electricity at a reasonable price in order to deliver the best electricity to their machines so they have the need to be constantly in alert in case that the electricity being provided is lower than expected so that they can file a complaint about the service in order to improve costs and life of the machines. They have to constantly monitor reactive power losses and also any other bad operation condition of the crane by any of the energy-related measurements as this could indicate a malfunction related to the use of the equipment.

## Current process

In order for them to be able to resolve some of these issues today they are required to perform manual checks by going into the field and reviewing one by one each of the cranes and sensors including having routine checks of the sensors by time to time in order to track their performance then make sure that they are using the optimal amount of energy and power in order to have efficient work and enhance the cranes life.

The proposed solution will reduce significantly this process as it will deliver real-time information at the palm of their hand. This will enable them to substitute the current process and be able to save costs and improve the performance of the machines.

## Goals

1. Develop a data pipeline that moves the data produced by container cranes in real-time to databases for analytical use.
2. Provide a real-time dashboard that allows engineers to see how their cranes are using energy during the operation.
3. Allow historical analysis of power usage for each crane.

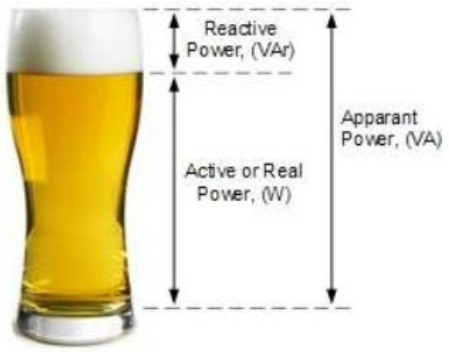
## User requirements

Users are maintenance engineers in need of a visualization tool to identify inefficiencies in the energy and power use of their container cranes.

## User domain concepts

For monitoring and power correction of electromechanical equipment, engineers use the concept of **Active, Apparent, and Reactive Power**.

The explanation of this lies in the fact that this type of equipment works with the AC power supply. Contrary to DC power, it is known that AC supplies current and voltage as waves traveling in a specific frequency. But, there is normally a delay between the current and voltage signals and this leads to the notion of power in its three components.




https://www.electronics-tutorials.ws/accircuits/reactive-power.html

The beer in the figure shows the logic of these power components and their role in electric energy supply. In very simple terms, **Reactive Power** is necessary for electricity to flow fast and maintain its quality while **Active Power** is the real energy being used by equipment.

Engineers in charge of electrical equipment have the goal of maintaining a healthy balance between active and reactive energy. In most cases, they tend to deliver for the maximum active power possible, which is the one that is effectively being utilized to move the loads on container cranes.

## Dashboard requirements

In order to achieve their objectives, engineers in the port of Cartagena have requested to be able to monitor and analyze variables in a dashboard containing the following:

1. Provide a view for energy analysis per crane displaying real-time and historical visualizations of **Active Energy**, **Apparent Energy**, **Reactive Energy.**
2. Provide a view for power analysis per crane displaying real-time and historical visualizations of **Active Power**, **Apparent Power**, **Reactive Power.**
3. Provide a decomposition of each **Power** into its three phases, namely: **Active Power (1,2,3)**, **Apparent Power (1,2,3)**, **Reactive Power (1,2,3).**
4. Provide a comparison of the different **Electrical Currents** per crane, displaying its composition by each of its different phases, namely: **Electrical Current(1,2,3,N)** along with its **three-phase average intensity.**
5. Provide a comparison between the different phases of **Electrical Potential** per crane, displaying its composition by each of its different phases, namely: **Electrical Potential(1_2,2_3,3_1).**

# Big Data Product Overview

To create a Dashboard for solving the Business issues, that were previously mentioned, it's important to define a Big Data software product with the following capabilities:

1. Extract data from a specific data source that is collecting sensor measured metrics per equipment and storing them in some kind of memory.
2. Transform datasource collected metrics into a schema that is flexible for easy adaptation to many different data sinks with different structures.
3. Load data from the flexible transformed schema into one or multiple data sinks that would be queried for analytics.
4. Visualize equipment data and metrics at a certain update frequency by a frontend application.

These phases are the essential responsibilities that the **data pipeline** has - to move data and transform it as it arrives and flows through it. The logic flow starts with the **extraction** stage, in which the system should pull data that is static from a deployed **PLC datasource** with a specific structure. After extracting this data, comes the **transformation** in which data flows through the pipeline initially in the structure that resembles it's source, but then a series of operations are progressively applied to this as it pases through the pipeline. At the end, this transformation has an objective in mind, which is prepare all data to be compatible with structure of the **datasinks** so that the load stage can quickly store them. Finally, the visualize part is responsible for constantly showing last updates of data as it is entering the the datasink.

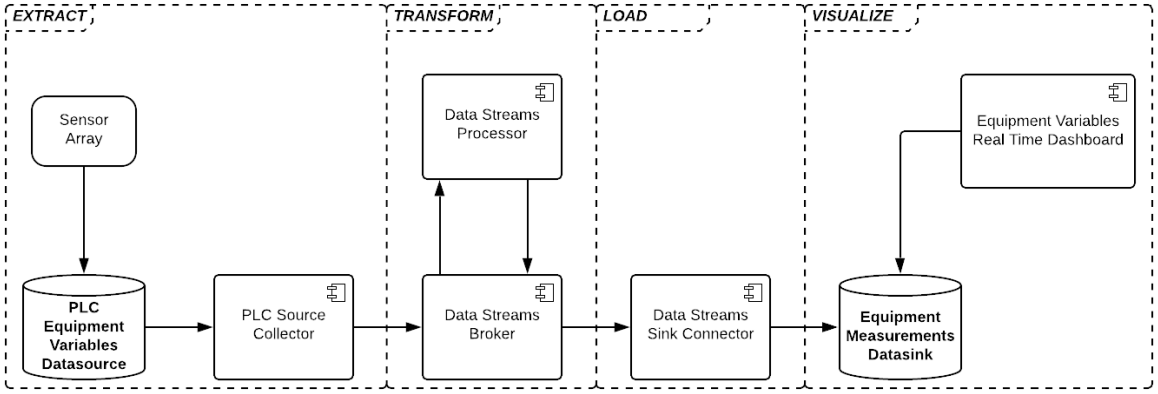
All in all, this process is the Extract Transform, Load and Visualize process, which is very common in real time big data applications.

## Design requirements and constraints

1. New equipment variable measurements should be updated every 5 seconds in the dashboard.
2. The data pipeline should take less than 1 second processing all stages for each batch of variable measurement for all equipments
3. Data collection frequency from PLC Siemens device should be at 1 second per each batch of measurements for all equipments.
4. Data collection needs to be resilient, fault-tolerant and scalable.
5. The network in which the plc data collector software would operate is a LAN with the PLC connected via ethernet.
6. Protocol for communicating with PLC for data extraction is Modbus TCP/IP

# System architecture and design

For solving the set of specifications and constraints mentioned above, an architecture was defined for this. The figure below shows all the set of components and the stage to which they belong. Each of them is responsible for a specific concern and wraps all the complexity necessary to solve problems at more granular levels. A thorough description will be given for each one of them and technologies used for implementing them would be briefly described.




## PLC Equipment Variables Datasource

The PLC equipment variables datasource is a repository holding data and making it available to be collected using the Modbus TCP protocol. This protocol follows a client server architecture in which the PLC's data is requested. For this to work, data collection in the PLC device is structured in a way that allows the client component to read this data by sending specific requests.

| Equipment | Variable | Units | Equipment | Modbus Address |
|---|---|---|---|---|
| G10 | Energia_Activa_Total | kWh | %DB11.DBD0 | 40001 |
| | Energia_Aparente_Total | kVAh | %DB11.DBD4 | 40003 |
| | Energia_Reactiva_Total | kVARh | %DB11.DBD8 | 40005 |
| | Potencia_Activa_Total | kW | %DB11.DBD12 | 40007 |
| | Potencia_Aparente_Total | kVA | %DB11.DBD16 | 40009 |
| | Potencia_Reactiva_Total | kVAR | %DB11.DBD20 | 40011 |

The table above shows how data is organized inside the PLC device. It also helps to understand how the Modbus TCP protocolo works. Each variable is given a specific **modbus address** so it can be queried. Then, the client that wants to read data composes a request specifying the first variable *address* from the equipment that wants to be read and the quantity of variables for that equipment. For example, equipment **G10** has **6** variables so a request for reading all of them would be like these "Hey Mr. PLC please send me **6** variables starting from address **40001**". To this the PLC would answer with the last values he has collected from sensors and stored in his memory. Something like a list of numbers with the

measurements for those variables would look like this: [2.57, 193.2, 5.5, 6.9, 8.3, 100]. Notice, there are exactly 6 values which is what was asked for.

## PLC Collector - Java application

This component is a Java Application that is responsible for extracting the data from the PLC and send it directly through the pipeline for transformation. His logic flow can be resumed in the following steps:

1. Connect to the PLC
2. Load the PLC tables to know what addresses correspond to specific equipment variables
3. For each equipment send a Modbus request to read all variables belonging to him
4. As each measurement of equipment variables arrive, push them to the data streams broker.
5. Wait for 1 second
6. Restart the whole process again

Java was chosen for the implementation of this extract and send logic because it's a robust programming language that offers stable battle-tested libraries for connecting with PLC's and also for connecting with the streams broker in charge of receiving data and execute the pipeline.

## Data Streams Broker - Kafka broker cluster

This component is a very robust and scalable cluster that acts as a data streams broker between the extract part, the transform part and load part. Apache Kafka is the technology behind it and enables Big Data Streams processing. This is because it works in a distributed fashion and runs on many nodes at different times. Due to this, there is a lot of redundancy so that if some nodes fail then they can be covered by others who are fit to receive messages. This is very important for enabling high velocity processing because tasks are distributed among nodes and also there is low probability that the pipeline stops moving data at a single point in time.

Data Streams Broker uses something called topic, which is basically a fancier name for message queue. In a very superficial level a topic allows other systems to send and read messages from and to it enabling communication in an ordered fashion. Topics are the element of communication between all stages of ETL process.

## Data Streams Processor (Kafka sql processors)

This component is basically what is called data pipeline, since it's the core of the transformation logic. It uses Kafka SQL processors to shape data as it is arriving to the input topics.

## Data Streams Connector (Kafka Connect)

This component is in charge of writing to a data sink using kafka connect and is in charge of simply loading data from a topic. Kafka connect technology enables very simple writes to many different databases. In this case it loads data into influxdb using a connector and it's very robust and fault tolerant.

## Equipment Measurements Datasink - Influxdb

For the success of the project, it is important to have a place where the actual data gets stored and since the amount of data being generated is very big and rapidly increasing a no traditional approach was taken. Storage represents a critical part of the system as this will allow to effectively read and write into the history data and later on could enable engineers to derive more meaningful information out of their data, so that's the main reason a solution that was highly scalable and highly available was needed.

Before going more deeply into our storage technology, its important to discuss a few important concepts that will help to provide a better explanation on this topic. Firstly it is important to mention what NoSQL database is. A NoSQL database is very known for as "Not only SQL database" which refers to the fact that these types of solutions are well known for being an alternative to the more traditional and standard relational databases like Oracle, MySQL or Microsoft SQL Server in other words this products or services rely mostly on approaches that are different from the well known tabular/table data models which can most of the times make operations be more faster as well as providing more flexibility on this type of solutions. Needless to say, this type of approach does carry some constraints as in order to gain the scalability and availability that these resources provide some things have to be given up as there is no perfect solution. The disadvantages that some of this can carry are mostly related to:

- Having a **narrow focus**, which is mostly related to functionality but also that some of the use cases that these databases were created for could mean that the solution is very much specialized although this could also represent an advantage if well applied.
- **Lack of standard:** As these types of solution can be very much different from one another even though they are designed in the same way.
- **GUI:** Most of these solutions could lack having a Graphic User Interface that could increase the learning curve for some persons.

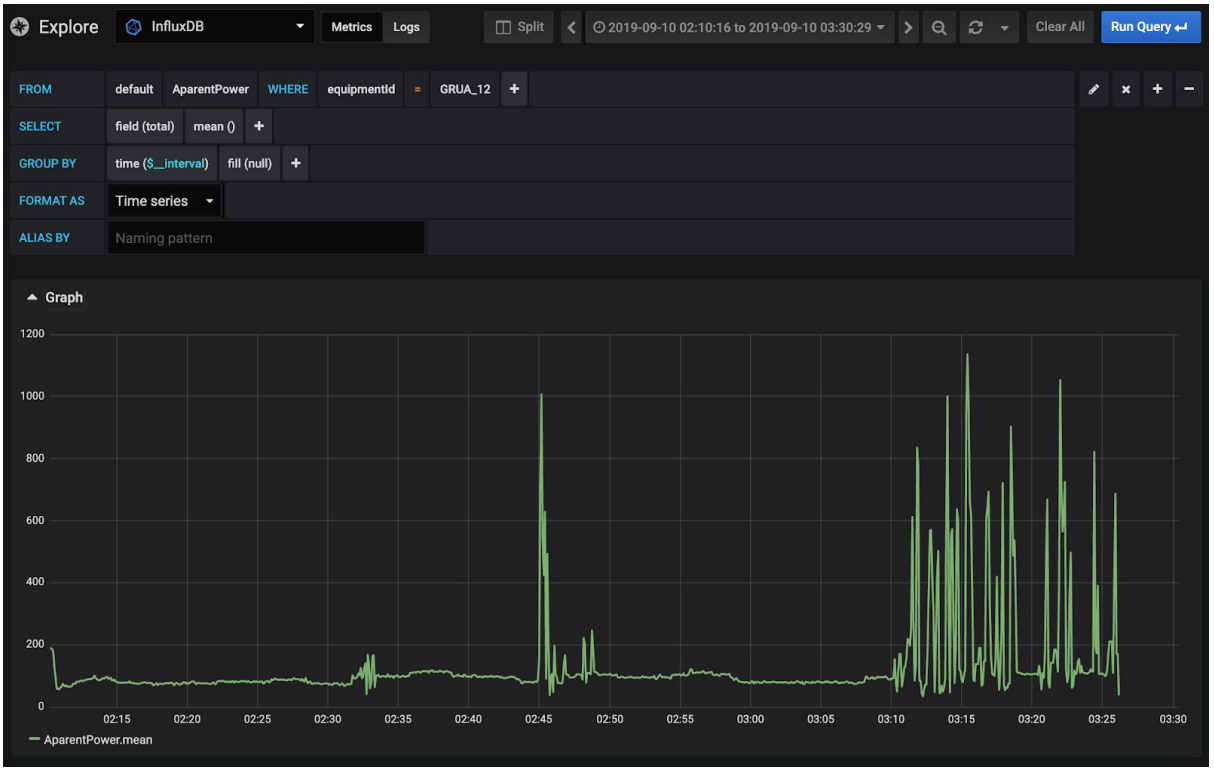
- **SQL Support**: As mentioned some of these solutions also lack the well now SQL language which could represent an important factor in adopting this type of technology as it could increase the possible learning curve that users could have.

Nevertheless, if correctly evaluated according to each of the needs of the project, this type of solution could signify an efficient solution where the benefits would be more than the disadvantages.

After this brief introduction to the NoSQL world, we will proceed to discuss our storage solution, InfluxDB.”Real-time visibility into stacks, sensors, and systems” is what the official page of InfluxDB states and is a very accurate summary of what this storage technology has to offer, InfluxDB is an open-source time-series database solution which has as the main focus to provide a solution that enables writing and reading large amounts of data that involves timestamped information. One of the most popular use cases for this database is related to the monitoring and tracking of sensors as this enables businesses to be able to effectively gain insights of the current situation of machines that they own for them to take action and control of what is happening at the moment due to the real-time streaming that InfluxDB is able to provide to companies which made it a perfect fit for this particular use case as it would provide a highly scalable, highly available and highly extensible solution that later on could enable a major competitive advantage as they will enhance their analytics capacity. Later on, we will discuss a bit more about how InfluxDB works and how it was applied to this specific case.

## Dashboard Visualization Server - Grafana

In order to be able to display the proper results in a friendly way, a visualization needed to be created. This would enable people to quickly understand in real-time what is happening. A couple of technologies were considered but, in the end, we decided to go with an out of the box solution which enabled us to integrate the current stack and build the necessary dashboard along with bringing some other functionality and that could handle the scalability and speed of the data that was being generated. The decision came to use an Open source product that is called Grafana, what makes this so special is that it is a specialized solution built around the use case of time series analytics and monitoring that integrated perfectly with our stack. What is really important about this platform is that it does not only packs the solution to build dynamics dashboard but it also lets you directly create alerts for when an item surpasses a certain value and even directly from the platform to create queries on the fly in order to really be able to explore and understand your data.

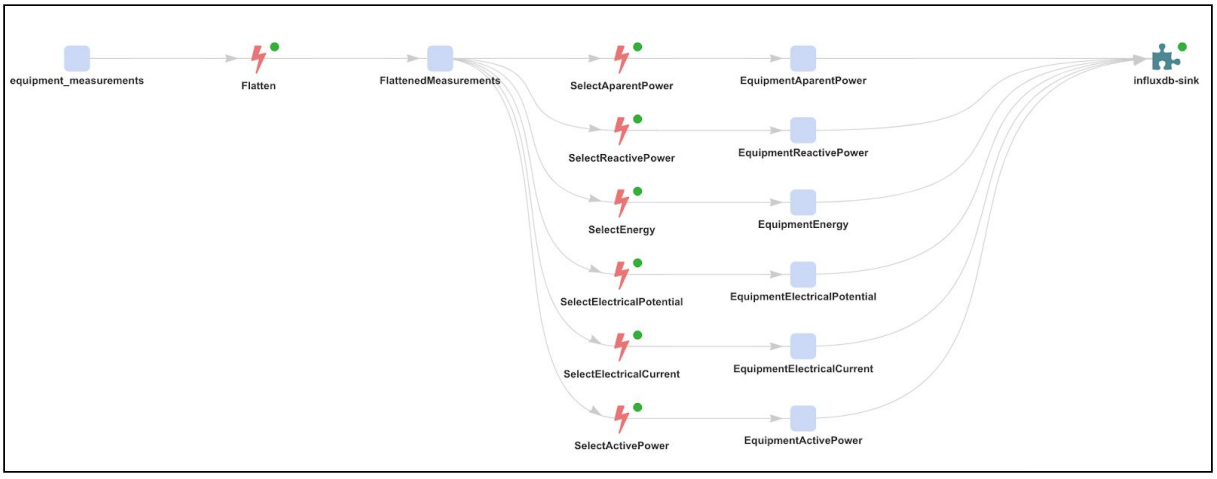




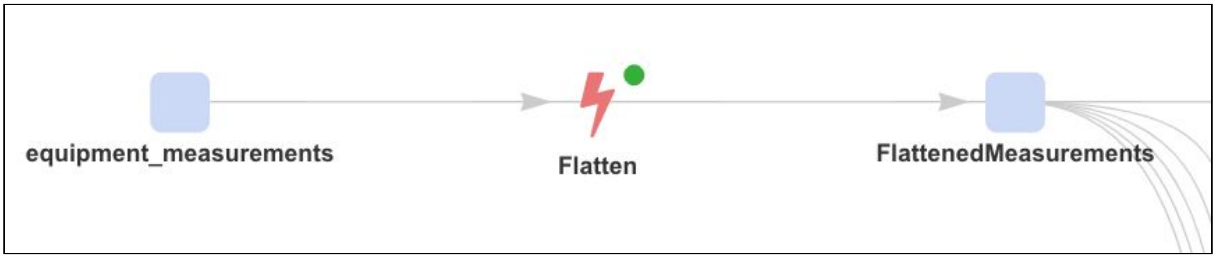
# Data Pipeline Implementation

In order to be able to correctly adapt the schema that was coming all the way from the cranes into the measurement schema that it was needed for InfluxDB some transformations where needed. For this task, we used a Data streaming platform that is called Lenses which enable us to use our existing data infrastructure that was relying on Kafka in order to build the necessary dataflows to deliver the data into InfluxDB.

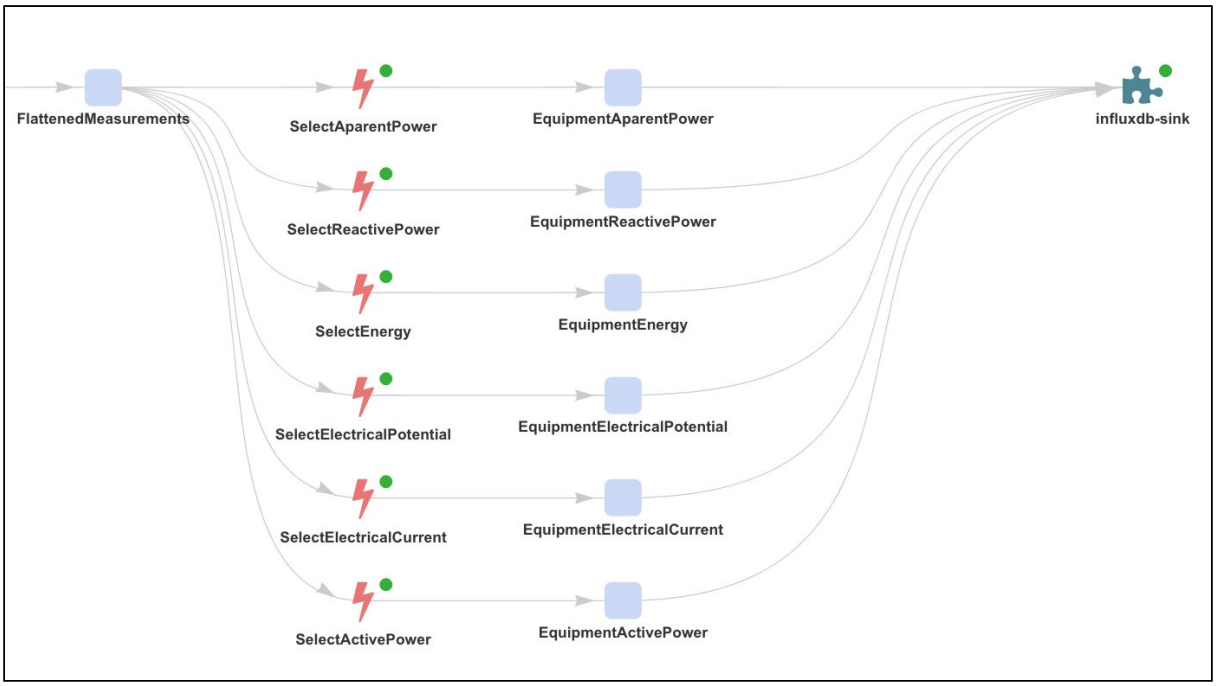
Below you will see the overall flow of the data starting from the Topic that directly receives the data from the equipment until the end when the data is dumped into InfluxDB. We will provide a more detailed explanation of the parts of this data flow next.




The data flow can be mainly subdivided into 2 main parts the Flatten and the Selects.
The first part of data flow is composed of 2 Kafka topics and 1 SQL Processor or Kafka stream.




The main goal of this part of the flow is to basically be able to ‘flatten’ the schema or in other words to provide another view of the schema by putting it all in one single table in a more user-friendly version of the data in order to perform more easily the necessary operations to bring the data over InfluxDB. The flow starts with the equipment measurement topic which basically receives all the messages coming directly from the PLC. After that, the Flatten SQL Processor receives the messages from the topic and initiates the Flattened process which will be done to finalize in the Flattened Measurements topic. Once the data has been dumped into the Flattened Measurements topic in the needed way we can proceed with the second part of the flow which basically refers to the Selects. The main goal of this part is to be able to take the already flattened schema directly from the Flattened Measurements topic process it using the SQL Processors in order to insert the data in the necessary topic that will then deliver the message into InfluxDB.




This flow basically consists of 7 topics, 6 SQL Processors and 1 Sink Connector. The flow starts with Flattened Measurements by using the SQL Processors to read the already flattened data and dump it into their respective topics which after that they will deliver the data to the sink which is a connector that has access to InfluxDB.

For deeper information regarding regarding the queries that were executed for the transformations and the schema of each of the topics please see the Annex pages.

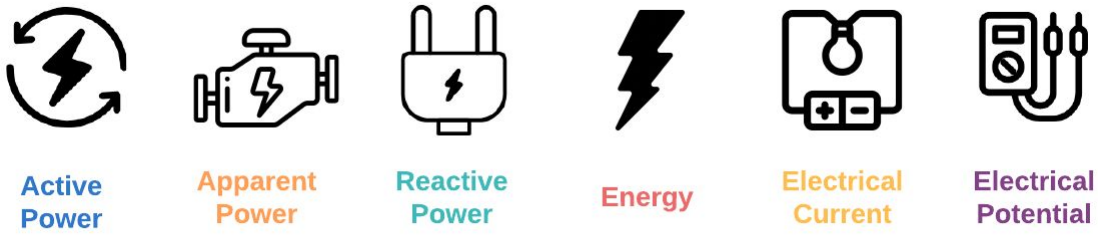
## InfluxDB Measurements Schema

In order to fully understand the implementation is important to define a few key concepts on how InfluxDB basically works as this was the way, the schema was defined.

InfluxDB handles a concept that is called a ‘Measurement’ which basically is a description or name of the data stored, this so-called ‘Measurement’ handles all the tags, fields and time which are basically the columns in a traditional relational database with the measurement being the table this concept is aimed to drive the design in a way that each table or measurement represents a real measurement in real life such as temperature or price. You can probably associate the tags, fields and time with just a normal column like in a traditional relational database and in concept, it might be pretty similar but the way that InfluxDB process all the queries makes both the time and tags really important as it can deeply affect the performance of the database.The tags in InfluxDB, are optional but is very important to pay attention to them because they have the characteristic that they are indexed which

makes performance change drastically so the design should be focus to tag your most filtered value in order to increase the overall performance of the database when you query it. In our particular case the tag was set to the particular equipment since it was gonna be the most filtered value of our design. To conclude the fields are just normal columns that are not indexed so overall we could say that they are normal columns.

For our approach we divided the measurements in the following way.




Trying to really decompose as much as we could each of the different possible measures so that they could fit within.

At the end the schema was as show below. Having the Equipment as a Tag and the remaining values as fields.

| Active Power |
|---|
| Phase 1 |
| Phase 2 |
| Phase 3 |
| Total |
| Units |
| EquipmentId |
| MeasuredAt |

| Apparent Power |
|---|
| Phase 1 |
| Phase 2 |
| Phase 3 |
| Total |
| Units |
| EquipmentId |
| MeasuredAt |

| Reactive Power |
|---|
| Phase 1 |
| Phase 2 |
| Phase 3 |
| Total |
| Units |
| EquipmentId |
| MeasuredAt |

| Energy |
|---|
| Total Active Power |
| Total Apparent Power |
| Total Reactive Power |
| EquipmentId |
| MeasuredAt |

| Electrical Current |
|---|
| Phase 1 |
| Phase 2 |
| Phase 3 |
| Phase N |
| Intensidad Media Trifasica |
| EquipmentId |
| MeasuredAt |

| Electrical Potential |
|---|
| Phase 1_2 |
| Phase 2_3 |
| Phase 3_1 |
| Phase 1_N |
| Phase 2_N |
| Phase 3_N |
| EquipmentId |
| MeasuredAt |

# Results

## Final Grafana Dashboard

For the final solution, 2 main dashboards were developed, in order to correctly achieve the objectives proposed at the beginning of the project.

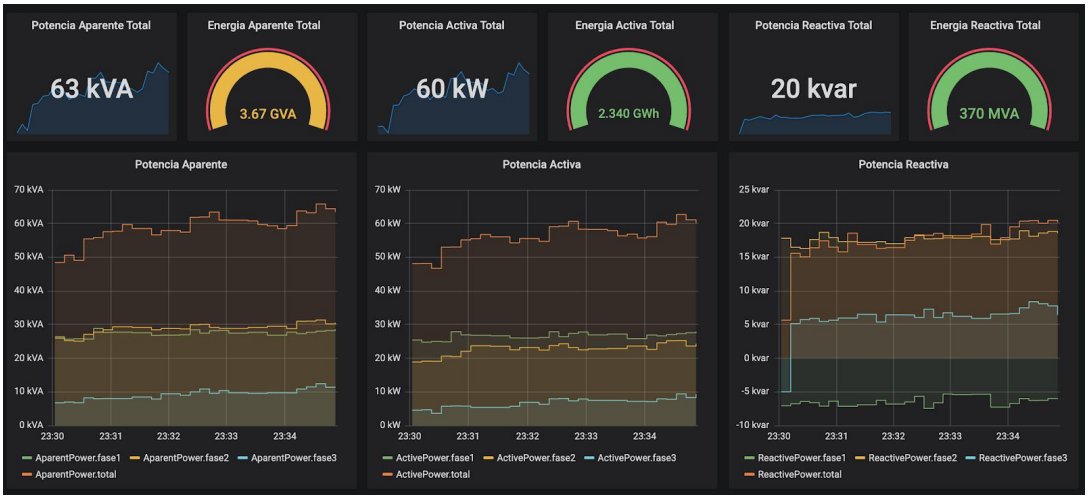
1. The first dashboard was aimed at providing quick information into some of the most important metrics of all of the cranes **Reactive Power** and **Reactive Energy.** This will let the maintenance engineers quickly glance at which set of equipment is using more energy that is supposed to and they would be able to deep dive into the other

metrics by quickly going into the second dashboard by accessing into directly clicking into the set of equipment they are more interested in.

The dashboard is mainly divided into 3 panels. The first section displaying the information regarding the current total **Reactive Energy** by each equipment.

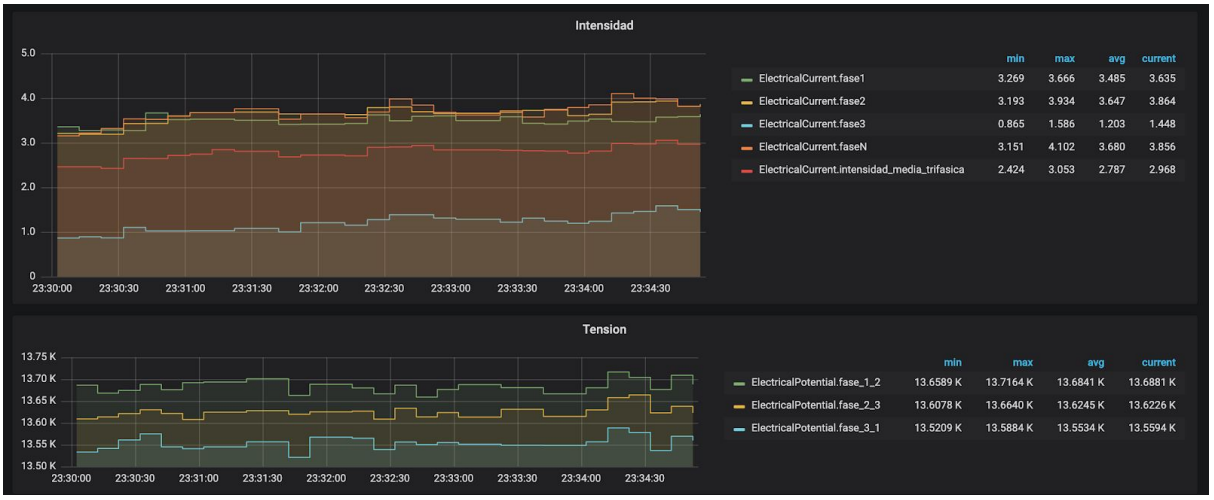
The second panel basically displays the current **Reactive Power** by each of the equipment and lastly, the panel below provides an overall comparison of the **Reactive Power** through time, enable engineers to quickly see peaks of energy that are being generated at certain times or moments throughout the day


2. The second dashboard was aimed at providing a full decomposition of all of the **energy** and **power** consumption variables related to single equipment. Each of the dashboards contains the same panels in the same order, the only thing changing by dashboard is the specific equipment that is being displayed.



As seen above the variables related to **Electrical Current, Electrical Potential, Active Energy, Reactive Energy, Apparent Energy, Apparent Power, Active Power, and Reactive Power** are all being displayed inside the dashboard.

The first part of the dashboard is aimed into displaying data regarding the Current Totals of each of the types of Energy and Power and just below them, we provide a decomposition of each of the phases overtime for all of the different Power measurements.




Finally but not less important a decomposition of the **Electrical Current and Electrical Potential** by displaying the variables decomposed by all of their phases, proving a view over time of how these variables are progressing throughout the day.

## Processing capacity

In order to correctly measure the performance of the whole pipeline and be available to identify how efficient we were able to deliver the data some metrics were taken and have been decomposed in the table below.

| Processor Name | Messages In/s | Messages Out/s |
|---|---|---|
| Flatten | 10.1 | 4.33 |
| SelectApparentPower | 4.12 | 4.21 |
| SelectReactivePower | 4.12 | 4.21 |
| SelectActiveEnergy | 4.12 | 4.21 |
| SelectEnergy | 4.12 | 4.21 |
| SelectElectricalPotential | 4.12 | 4.21 |
| SelectElectricalCurrent | 4.12 | 4.21 |

These metrics can give us good insight on how the pipeline is performing overall when in optimal conditions. It is important to mention that all of this was made using limited resources and because of this there could be room for improvement and that these numbers are not definite as this could improve or decrease depending on the computing resources available.
By looking at the metrics above we can simply see that we are basically able to read 4 messages per second and to write a little more than 4 messages per second which tells us that we are working at 100% of capacity in the whole system.

# Conclusions and future work

By being able to work in a solution of this caliber we were able to learn and apply different concepts learned through the duration of our Master were both business and technology came together. From being able to properly identify the solution based on the business value and requirements that were expected to be delivered, to work with cutting edge technologies that enabled us to provide the best solution possible. With this, we can recognize the true value that this service can deliver to the people in the port and be able to completely

revolutionize a sector that might not have as much innovation as other sectors in the industry.

This solution will be just the start of furthermore possible innovations with this sector and particularly with this company as with the infrastructure that was delivered they are pretty much set to be able to go no the next stage which could include being able to completely analyze their data and explore it and enable machine learning and predictive analytics capabilities.

# Appendix

Equipment Measurement Topic Schema

```
{
  "type": "record",
  "name": "EquipmentMeasurement",
  "namespace": "com.pedro.plc.collector.kafka",
  "fields": [
    {
      "name": "equipmentId",
      "type": "string"
    },
    {
      "name": "measuredAt",
      "type": "string"
    },
    {
      "name": "measuredAtEpoch",
      "type": "long"
    },
    {
      "name": "measurements",
      "type": {
        "type": "map",
        "values": {
          "type": "record",
```

```
          "name": "Variable",
          "fields": [
            {
              "name": "name",
              "type": "string"
            },
            {
              "name": "units",
              "type": "string"
            },
            {
              "name": "val",
              "type": "float"
            }
          ]
        }
      }
    }
  ]
}
```

Flattened Measurement Topic Schema

```
{
  "type": "record",
  "name": "EquipmentMeasurement",
  "namespace": "com.pedro.plc.collector.kafka",
  "fields": [
    {
      "name": "equipmentId",
      "type": "string"
    },
    {
      "name": "measuredAt",
      "type": "string"
    },
    {
      "name": "measuredAtEpoch",
      "type": "long"
    },
    {
      "name": "energia_activa_total",
      "type": [
        "null",
        {
```

```
        "type": "record",
        "name": "Variable",
        "fields": [
          {
            "name": "name",
            "type": "string"
          },
          {
            "name": "units",
            "type": "string"
          },
          {
            "name": "val",
            "type": "float"
          }
        ]
      }
    ],
    "doc": ""
  },
  {
    "name": "energia_aparente_total",
    "type": [
      "null",
      "Variable"
    ],
    "doc": ""
  },
  {
    "name": "energia_reactiva_total",
    "type": [
      "null",
      "Variable"
    ],
    "doc": ""
  },
  {
    "name": "frecuencia",
    "type": [
      "null",
      "Variable"
    ],
    "doc": ""
  },
  {
```

```
      "name": "intensidad_1",
      "type": [
        "null",
        "Variable"
      ],
      "doc": ""
    },
    {
      "name": "intensidad_2",
      "type": [
        "null",
        "Variable"
      ],
      "doc": ""
    },
    {
      "name": "intensidad_3",
      "type": [
        "null",
        "Variable"
      ],
      "doc": ""
    },
    {
      "name": "intensidad_n",
      "type": [
        "null",
        "Variable"
      ],
      "doc": ""
    },
    {
      "name": "intensidad_media_trifasica",
      "type": [
        "null",
        "Variable"
      ],
      "doc": ""
    },
    {
      "name": "intensidad_1_thd",
      "type": [
        "null",
        "Variable"
      ],
```

```
      "doc": ""
    },
    {
      "name": "intensidad_2_thd",
      "type": [
        "null",
        "Variable"
      ],
      "doc": ""
    },
    {
      "name": "intensidad_3_thd",
      "type": [
        "null",
        "Variable"
      ],
      "doc": ""
    },
    {
      "name": "potencia_aparente_1",
      "type": [
        "null",
        "Variable"
      ],
      "doc": ""
    },
    {
      "name": "potencia_aparente_2",
      "type": [
        "null",
        "Variable"
      ],
      "doc": ""
    },
    {
      "name": "potencia_aparente_3",
      "type": [
        "null",
        "Variable"
      ],
      "doc": ""
    },
    {
      "name": "potencia_aparente_total",
      "type": [
```

```
        "null",
        "Variable"
      ],
      "doc": ""
    },
    {
      "name": "potencia_activa_1",
      "type": [
        "null",
        "Variable"
      ],
      "doc": ""
    },
    {
      "name": "potencia_activa_2",
      "type": [
        "null",
        "Variable"
      ],
      "doc": ""
    },
    {
      "name": "potencia_activa_3",
      "type": [
        "null",
        "Variable"
      ],
      "doc": ""
    },
    {
      "name": "potencia_activa_total",
      "type": [
        "null",
        "Variable"
      ],
      "doc": ""
    },
    {
      "name": "potencia_reactiva_1",
      "type": [
        "null",
        "Variable"
      ],
      "doc": ""
    },
```

```
    {
      "name": "potencia_reactiva_2",
      "type": [
        "null",
        "Variable"
      ],
      "doc": ""
    },
    {
      "name": "potencia_reactiva_3",
      "type": [
        "null",
        "Variable"
      ],
      "doc": ""
    },
    {
      "name": "potencia_reactiva_total",
      "type": [
        "null",
        "Variable"
      ],
      "doc": ""
    },
    {
      "name": "tension_1_2",
      "type": [
        "null",
        "Variable"
      ],
      "doc": ""
    },
    {
      "name": "tension_2_3",
      "type": [
        "null",
        "Variable"
      ],
      "doc": ""
    },
    {
      "name": "tension_3_1",
      "type": [
        "null",
        "Variable"
```

```
      ],
      "doc": ""
    },
    {
      "name": "tension_1_N",
      "type": [
        "null",
        "Variable"
      ],
      "doc": ""
    },
    {
      "name": "tension_2_N",
      "type": [
        "null",
        "Variable"
      ],
      "doc": ""
    },
    {
      "name": "tension_3_N",
      "type": [
        "null",
        "Variable"
      ],
      "doc": ""
    }
  ]
}
```

Equipment Apparent Power Topic Schema

```
{
  "type": "record",
  "name": "EquipmentMeasurement",
  "namespace": "com.pedro.plc.collector.kafka",
  "fields": [
    {
      "name": "equipmentId",
      "type": "string"
    },
    {
      "name": "measuredAt",
      "type": "string"
    },
    {
```

```
      "name": "measuredAtEpoch",
      "type": "long"
    },
    {
      "name": "fase1",
      "type": [
        "null",
        "float"
      ]
    },
    {
      "name": "fase2",
      "type": [
        "null",
        "float"
      ]
    },
    {
      "name": "fase3",
      "type": [
        "null",
        "float"
      ]
    },
    {
      "name": "total",
      "type": [
        "null",
        "float"
      ]
    },
    {
      "name": "unidades",
      "type": [
        "null",
        "string"
      ]
    }
  ]
}
```

Equipment Reactive Power Topic Schema

```
{
  "type": "record",
  "name": "EquipmentMeasurement",
  "namespace": "com.pedro.plc.collector.kafka",
```

```
  "fields": [
    {
      "name": "equipmentId",
      "type": "string"
    },
    {
      "name": "measuredAt",
      "type": "string"
    },
    {
      "name": "measuredAtEpoch",
      "type": "long"
    },
    {
      "name": "fase1",
      "type": [
        "null",
        "float"
      ]
    },
    {
      "name": "fase2",
      "type": [
        "null",
        "float"
      ]
    },
    {
      "name": "fase3",
      "type": [
        "null",
        "float"
      ]
    },
    {
      "name": "total",
      "type": [
        "null",
        "float"
      ]
    },
    {
      "name": "unidades",
      "type": [
        "null",
```

```
        "string"
      ]
    }
  ]
}
```

Equipment Energy Topic Schema

```
{
  "type": "record",
  "name": "EquipmentMeasurement",
  "namespace": "com.pedro.plc.collector.kafka",
  "fields": [
    {
      "name": "equipmentId",
      "type": "string"
    },
    {
      "name": "measuredAt",
      "type": "string"
    },
    {
      "name": "measuredAtEpoch",
      "type": "long"
    },
    {
      "name": "energia_activa_total",
      "type": [
        "null",
        "float"
      ]
    },
    {
      "name": "energia_aparente_total",
      "type": [
        "null",
        "float"
      ]
    },
    {
      "name": "energia_reactiva_total",
      "type": [
        "null",
        "float"
      ]
    }
  ]
```

```
}
```

Equipment Electrical Potential Topic Schema

```
{
  "type": "record",
  "name": "EquipmentMeasurement",
  "namespace": "com.pedro.plc.collector.kafka",
  "fields": [
    {
      "name": "equipmentId",
      "type": "string"
    },
    {
      "name": "measuredAt",
      "type": "string"
    },
    {
      "name": "measuredAtEpoch",
      "type": "long"
    },
    {
      "name": "fase_1_2",
      "type": [
        "null",
        "float"
      ]
    },
    {
      "name": "fase_2_3",
      "type": [
        "null",
        "float"
      ]
    },
    {
      "name": "fase_3_1",
      "type": [
        "null",
        "float"
      ]
    },
    {
      "name": "fase_1_N",
      "type": [
```

```
        "null",
        "float"
      ]
    },
    {
      "name": "fase_2_N",
      "type": [
        "null",
        "float"
      ]
    },
    {
      "name": "fase_3_N",
      "type": [
        "null",
        "float"
      ]
    }
  ]
}
```

Equipment Electrical Current Topic Schema

```
{
  "type": "record",
  "name": "EquipmentMeasurement",
  "namespace": "com.pedro.plc.collector.kafka",
  "fields": [
    {
      "name": "equipmentId",
      "type": "string"
    },
    {
      "name": "measuredAt",
      "type": "string"
    },
    {
      "name": "measuredAtEpoch",
      "type": "long"
    },
    {
      "name": "fase1",
      "type": [
        "null",
        "float"
      ]
    },
```

```
    {
      "name": "fase2",
      "type": [
        "null",
        "float"
      ]
    },
    {
      "name": "fase3",
      "type": [
        "null",
        "float"
      ]
    },
    {
      "name": "faseN",
      "type": [
        "null",
        "float"
      ]
    },
    {
      "name": "intensidad_media_trifasica",
      "type": [
        "null",
        "float"
      ]
    }
  ]
}
```

Equipment Active Power Topic Schema

```
{
  "type": "record",
  "name": "EquipmentMeasurement",
  "namespace": "com.pedro.plc.collector.kafka",
  "fields": [
    {
      "name": "equipmentId",
      "type": "string"
    },
    {
      "name": "measuredAt",
      "type": "string"
    },
    {
```

```
      "name": "measuredAtEpoch",
      "type": "long"
    },
    {
      "name": "fase1",
      "type": [
        "null",
        "float"
      ]
    },
    {
      "name": "fase2",
      "type": [
        "null",
        "float"
      ]
    },
    {
      "name": "fase3",
      "type": [
        "null",
        "float"
      ]
    },
    {
      "name": "faseN",
      "type": [
        "null",
        "float"
      ]
    },
    {
      "name": "intensidad_media_trifasica",
      "type": [
        "null",
        "float"
      ]
    }
  ]
}
```

Flatten SQL Processor

```
set autocreate=true;
INSERT INTO FlattenedMeasurements
SELECT equipmentId, measuredAt, measuredAtEpoch,
      measurements.ENERGIA_ACTIVA_TOTAL[0] as energia_activa_total,
```

```
      measurements.ENERGIA_APARENTE_TOTAL[0] as energia_aparente_total,
      measurements.ENERGIA_REACTIVA_TOTAL[0] as energia_reactiva_total,
      measurements.FRECUENCIA[0] as frecuencia,
      measurements.INTENSIDAD_1[0] as intensidad_1,
      measurements.INTENSIDAD_2[0] as intensidad_2,
      measurements.INTENSIDAD_3[0] as intensidad_3,
      measurements.INTENSIDAD_N[0] as intensidad_n,
      measurements.INTENSIDAD_MEDIA_TRIFASICA[0] as
intensidad_media_trifasica,
      measurements.INTENSIDAD_1_THD[0] as intensidad_1_thd,
      measurements.INTENSIDAD_2_THD[0] as intensidad_2_thd,
      measurements.INTENSIDAD_3_THD[0] as intensidad_3_thd,
      measurements.POTENCIA_APARENTE_1[0] as potencia_aparente_1,
      measurements.POTENCIA_APARENTE_2[0] as potencia_aparente_2,
      measurements.POTENCIA_APARENTE_3[0] as potencia_aparente_3,
      measurements.POTENCIA_APARENTE_TOTAL[0] as
potencia_aparente_total,
      measurements.POTENCIA_ACTIVA_1[0] as potencia_activa_1,
      measurements.POTENCIA_ACTIVA_2[0] as potencia_activa_2,
      measurements.POTENCIA_ACTIVA_3[0] as potencia_activa_3,
      measurements.POTENCIA_ACTIVA_TOTAL[0] as potencia_activa_total,
      measurements.POTENCIA_REACTIVA_1[0] as potencia_reactiva_1,
      measurements.POTENCIA_REACTIVA_2[0] as potencia_reactiva_2,
      measurements.POTENCIA_REACTIVA_3[0] as potencia_reactiva_3,
      measurements.POTENCIA_REACTIVA_TOTAL[0] as
potencia_reactiva_total,
      measurements.TENSION_1_2[0] as tension_1_2,
      measurements.TENSION_2_3[0] as tension_2_3,
      measurements.TENSION_3_1[0] as tension_3_1,
      measurements.TENSION_1_N[0] as tension_1_N,
      measurements.TENSION_2_N[0] as tension_2_N,
      measurements.TENSION_3_N[0] as tension_3_N
      FROM equipment_measurements
      WHERE equipmentId = 'GRUA_10' or
              equipmentId = 'GRUA_12' or
            equipmentId = 'GRUA_6'  or
            equipmentId = 'GRUA_9'
```

Select Apparent Power SQL Processor

```
SET autocreate=true;
INSERT INTO EquipmentAparentPower
      SELECT equipmentId,
               measuredAt,
               measuredAtEpoch,
               potencia_aparente_1.val as fase1,
```

```
                potencia_aparente_2.val as fase2,
                potencia_aparente_3.val as fase3,
                potencia_aparente_total.val as total,
                potencia_aparente_total.units as unidades
      FROM FlattenedMeasurements
```

Select Reactive Power SQL Processor

```
SET autocreate=true;
INSERT INTO EquipmentAparentPower
      SELECT equipmentId,
                measuredAt,
                measuredAtEpoch,
                potencia_aparente_1.val as fase1,
                potencia_aparente_2.val as fase2,
                potencia_aparente_3.val as fase3,
                potencia_aparente_total.val as total,
                potencia_aparente_total.units as unidades
      FROM FlattenedMeasurements
```

Select Equipment Energy SQL Processor

```
SET autocreate=true;
INSERT INTO EquipmentEnergy
      SELECT equipmentId,
                measuredAt,
                measuredAtEpoch,
                energia_activa_total.val as energia_activa_total,
                energia_aparente_total.val as energia_aparente_total,
                energia_reactiva_total.val as energia_reactiva_total
      FROM FlattenedMeasurements
```

Select Electrical Potential SQL Processor

```
SET autocreate=true;
INSERT INTO EquipmentElectricalPotential
      SELECT equipmentId,
                measuredAt,
                measuredAtEpoch,
                tension_1_2.val as fase_1_2,
                tension_2_3.val as fase_2_3,
                tension_3_1.val as fase_3_1,
                tension_1_N.val as fase_1_N,
```

```
                tension_2_N.val as fase_2_N,
                tension_3_N.val as fase_3_N
      FROM FlattenedMeasurements
```

Select Electrical Current SQL Processor

```
SET autocreate=true;
INSERT INTO EquipmentElectricalCurrent
      SELECT equipmentId,
                measuredAt,
                measuredAtEpoch,
                intensidad_1.val as fase1,
                intensidad_2.val as fase2,
                intensidad_3.val as fase3,
                intensidad_n.val as faseN,
                intensidad_media_trifasica.val as
intensidad_media_trifasica
      FROM FlattenedMeasurements
```

Select Active Power SQL Processor

```
SET autocreate=true;
INSERT INTO EquipmentActivePower
      SELECT equipmentId,
                measuredAt,
                measuredAtEpoch,
                potencia_activa_1.val as fase1,
                potencia_activa_2.val as fase2,
                potencia_activa_3.val as fase3,
                potencia_activa_total.val as total,
                potencia_activa_total.units as unidades
      FROM FlattenedMeasurements
```

InfluxDB Sink Connect Insert statements

```
INSERT INTO ReactivePower SELECT fase1, fase2, fase3, total,
unidades,measuredAt FROM EquipmentReactivePower WITHTIMESTAMP
measuredAtEpoch WITHTAG (equipmentId);
INSERT INTO ActivePower SELECT fase1, fase2, fase3, total,
unidades,measuredAt FROM EquipmentActivePower WITHTIMESTAMP
measuredAtEpoch WITHTAG(equipmentId);
INSERT INTO AparentPower SELECT fase1, fase2, fase3, total,
unidades,measuredAt FROM EquipmentAparentPower WITHTIMESTAMP
measuredAtEpoch WITHTAG(equipmentId);
```

```
INSERT INTO Energy SELECT
energia_reactiva_total,energia_aparente_total,energia_activa_total,measu
redAt FROM EquipmentEnergy WITHTIMESTAMP measuredAtEpoch
WITHTAG(equipmentId);
INSERT INTO ElectricalCurrent SELECT
fase1,fase2,fase3,faseN,intensidad_media_trifasica,measuredAt FROM
EquipmentElectricalCurrent WITHTIMESTAMP measuredAtEpoch
WITHTAG(equipmentId);
INSERT INTO ElectricalPotential SELECT
fase_1_2,fase_2_3,fase_3_1,fase_1_N,fase_2_N,fase_3_N,measuredAt FROM
EquipmentElectricalPotential WITHTIMESTAMP measuredAtEpoch
WITHTAG(equipmentId);
```

# References


Lee, J., Ardakani, H. D., Yang, S., & Bagheri, B. (2015). Industrial big data analytics and cyber-physical systems for future maintenance & service innovation. Procedia Cirp, 38, 3-7.

Khan, M., Wu, X., Xu, X., & Dou, W. (2017, May). Big data challenges and opportunities in the hype of Industry 4.0. In 2017 IEEE International Conference on Communications (ICC) (pp. 1-6). IEEE.

Kambatla, K., Kollias, G., Kumar, V., & Grama, A. (2014). Trends in big data analytics. Journal of Parallel and Distributed Computing, 74(7), 2561-2573.

Demchenko, Y., Grosso, P., De Laat, C., & Membrey, P. (2013, May). Addressing big data issues in scientific data infrastructure. In 2013 International Conference on Collaboration Technologies and Systems (CTS) (pp. 48-55). IEEE.

Katal, A., Wazid, M., & Goudar, R. H. (2013, August). Big data: issues, challenges, tools and good practices. In 2013 Sixth international conference on contemporary computing (IC3) (pp. 404-409). IEEE.

Narkhede, N., Shapira, G., & Palino, T. (2017). Kafka: The Definitive Guide: Real-time Data and Stream Processing at Scale. " O'Reilly Media, Inc.".

Bhattacharya, D., & Mitra, M. (2013). Analytics on big fast data using real time stream data processing architecture (p. 34). EMC Corporation

Amini, S., Gerostathopoulos, I., & Prehofer, C. (2017, June). Big data analytics architecture for real-time traffic control. In 2017 5th IEEE International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS) (pp. 710-715). IEEE.

Goodhope, K., Koshy, J., Kreps, J., Narkhede, N., Park, R., Rao, J., & Ye, V. Y. (2012). Building LinkedIn's Real-time Activity Data Pipeline. IEEE Data Eng. Bull., 35(2), 33-45.

Gandomi, A., & Haider, M. (2015). Beyond the hype: Big data concepts, methods, and analytics. International journal of information management, 35(2), 137-144.

Meier, A., & Kaufmann, M. (2019). Nosql databases. In *SQL & NoSQL Databases* (pp. 201-218). Springer Vieweg, Wiesbaden.