



UNIVERSITAT  
POLITÈCNICA  
DE VALÈNCIA



UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Informática

Visualización de datos telemétricos de vehículos  
ferroviarios en herramientas de visualización web.

Trabajo Fin de Grado

Grado en Ciencia de Datos

AUTOR/A: Grau Gil, José Vicente

Tutor/a: Escobar Román, Santiago

CURSO ACADÉMICO: 2022/2023

**Agradecimientos:**

Nieves Cordones, David; tutor en la empresa.



## Resumen

---

Actualmente la utilización de soluciones web dinámicas utilizando navegadores web para la presentación de información permite introducir sinergias con los avances técnicos en interacción con el usuario y componentes gráficos de visualización existentes en otros entornos. El objetivo del proyecto es implementar un panel de visualización de datos, que permita juntar la información de diferentes grupos de vehículos y presentarla de forma unificada y concisa, y en tiempo real, que confíe en la presentación de información utilizando un servidor web, que aloje el sitio web gráfico y la interacción con el sistema de monitorización del tren.

Consistirá en realizar un estudio de las diferentes plataformas web para la visualización de datos que sean tendencia en la actualidad. De ellas, analizaremos la variedad de controles, y adecuación de los mismos, a la hora de presentar datos de telemetría de vehículos ferroviarios; así como las posibilidades que ofrecen a la hora de customizar el comportamiento de los controles que se emplean para crear los dashboard. Una vez decidida la herramienta de visualización web, se comenzará el proceso de creación de los gráficos y paneles informativos para crear el dashboard, esto requerirá de los conocimientos adecuados para el acceso y análisis de los datos de la base de datos correspondiente. Posteriormente, pueden realizarse pruebas de despliegue en el servidor principal.

**Palabras clave:** visualización web; ferrocarriles; monitorización.

## Abstract

---

Currently, the use of dynamic web solutions through web browsers for information presentation allows for synergies with technical advancements in user interaction and graphic visualization components existing in other environments. The goal of the project is to implement a data visualization dashboard that combines information from different groups of vehicles and presents it in a unified and concise manner, in real-time, relying on web server-based information presentation that hosts the graphical website and interaction with the train monitoring system.

The project will involve studying different web platforms for data visualization that are currently trending. Among them, we will analyse the variety of controls and their suitability in presenting telemetry data from railway vehicles, as well as the possibilities they offer for customizing the behaviour of the controls used to create the dashboards. Once the web visualization tool is decided, the process of creating graphics and informative panels to build the dashboard will begin. This will require the appropriate knowledge for accessing and analysing data from the corresponding database. Subsequently, deployment tests can be performed on the main server.

**Keywords:** web visualization; railways; monitoring.





## Tabla de contenidos

---

1.	Introducción .....	9
1.1.	Motivación .....	9
1.2.	Objetivos .....	10
1.3.	Impacto Esperado.....	10
1.4.	Metodología .....	11
1.5.	Estructura .....	11
1.6.	Convenciones .....	12
2.	Contextualización y estado del arte.....	13
2.1.	La empresa .....	13
2.2.	Introducción .....	13
2.3.	Requisitos.....	14
2.4.	Herramientas estudiadas.....	15
2.5.	Reseña del capítulo.....	21
3.	Análisis del problema.....	22
3.1.	Análisis energético o de eficiencia algorítmica.....	22
3.2.	Análisis del marco legal y ético .....	23
3.3.	Análisis de riesgos e identificación de soluciones .....	24
3.4.	Plan de Trabajo.....	25
3.5.	Presupuesto .....	26
4.	Preparación y comprensión de los datos .....	27
5.	Configuración y Despliegue.....	29
5.1.	Introducción .....	29
5.2.	Uso y organización del dashboard .....	29
5.3.	Interfaz de usuario.....	30
5.4.	Variables del dashboard .....	30
5.5.	Consultas SQL y configuración de controles.....	31
5.5.1.	Flota.....	32
5.5.2.	Locomotoras.....	43
5.6.	Resultados .....	53
6.	Conclusiones .....	55
7.	Trabajos futuros.....	57
8.	Glosario .....	58
9.	Referencias .....	59
10.	Anexos.....	61



10.1.	Objetivos de Desarrollo Sostenible (ODS) .....	61
10.2.	Versiones de Grafana y PostgreSQL.....	62
10.3.	PostgreSQL .....	62
10.3.1.	Instalación .....	62
10.3.2.	Verificación de la instalación .....	66
10.3.3.	Cargar base de datos PostgreSQL .....	67
10.3.4.	Descargar la base de datos PostgreSQL .....	68
10.3.5.	Configuración del entorno de ejecución.....	68
10.4.	Grafana.....	70
10.4.1.	Instalación .....	70
10.4.2.	Primer inicio de sesión .....	73
10.4.3.	Creación de la conexión a la base de datos .....	73
10.4.4.	Instalar Radar Graph .....	74
10.4.5.	Importación del dashboard .....	74



## Tabla de figuras y tablas

---

Figura 1: herramienta SYSLOC.....	14
Figura 2: dashboard Tesla Model 3 [7].....	15
Figura 3: dashboard de series temporales.....	16
Figura 4: dashboard de conexiones red [11].....	17
Figura 5: dashboard sobre el censo de EE. UU. [12].....	17
Figura 6: dashboard de análisis de ventas [14].....	18
Figura 7: dashboard sobre una campaña de marketing [15].....	18
Figura 8: conteo de votos.....	19
Figura 9: monitoreo de un firewall.....	19
Figura 10: dashboard sanitario.....	20
Figura 11: eficiencia de los equipos.....	20
Figura 12: diagrama de clases.....	28
Tabla 1: variables.....	30
Figura 13: localización de las variables.....	30
Figura 14: distribución de paneles.....	31
Figura 15: paneles generales.....	32
Figura 16: mapa de la flota.....	33
Figura 17: configuración de controles.....	33
Figura 18: configuración de controles.....	33
Figura 19: configuración de controles.....	33
Figura 20: tabla de sistemas.....	34
Figura 21: configuración de la tabla.....	35
Figura 22: configuración de la tabla.....	35
Figura 23: distribución de eventos de la flota.....	36
Figura 24: configuración de controles.....	36
Figura 25: tabla de eventos recientes.....	37
Figura 26: tabla de eventos frecuentes.....	37
Figura 27: gráfico de eventos.....	38
Figura 28: mapa de localización de eventos.....	42
Figura 29: paneles generales.....	43
Figura 30: localización del vehículo.....	44
Figura 31: configuración de controles.....	44
Figura 32: configuración de controles.....	44
Figura 33: configuración de controles.....	44
Figura 34: gráfico de velocidad.....	45
Figura 35: gráfico de combustible.....	45
Figura 36: gráfico de distancia del vehículo vs flota.....	47
Figura 37: configuración de controles.....	47
Figura 38: distribución de eventos del vehículo.....	48
Figura 39: gráfico de eventos del vehículo.....	49
Figura 40: localización de eventos del vehículo.....	52
Figura 41: sección de las flotas.....	53
Figura 42: sección de los vehículos.....	54
Tabla 2: Objetivos Desarrollo Sostenible.....	61



Figura 43: versiones disponibles para instalar .....	62
Figura 44: primer paso del instalador.....	63
Figura 45: segundo paso del instalador.....	63
Figura 46: tercer paso del instalador .....	63
Figura 47: cuarto paso del instalador .....	64
Figura 48: quinto paso del instalador .....	64
Figura 49: sexto paso del instalador .....	64
Figura 50: séptimo paso del instalador.....	65
Figura 51: confirmar la instalación .....	65
Figura 52: instalación en curso.....	65
Figura 53: instalación finalizada .....	66
Figura 54: SQL Shell en el buscador .....	66
Figura 55: inicio del SQL Shell.....	66
Tabla 3: definición de parámetros.....	67
Figura 56: carga de datos .....	67
Figura 57: verificar la carga .....	68
Figura 58: extensión de PostgreSQL.....	68
Figura 59: primer paso para crear la conexión .....	69
Figura 60: segundo paso para crear la conexión .....	69
Figura 61: tercer paso para crear la conexión.....	69
Figura 62: cuarto paso para crear la conexión.....	69
Figura 63: quinto paso para crear la conexión .....	69
Figura 64: sexto paso para crear la conexión .....	69
Figura 65: séptimo paso para crear la conexión.....	70
Figura 66: menú desplegable.....	70
Figura 67: selector de la base de datos.....	70
Figura 68: Grafana versión 10.0.0.....	70
Figura 69: Grafana versión 9.3.2.....	71
Figura 70: inicio del instalador.....	71
Figura 71: términos y condiciones de Grafana.....	71
Figura 72: características a instalar .....	72
Figura 73: confirmar la instalación .....	72
Figura 74: instalación en progreso .....	72
Figura 75: instalación finalizada .....	73
Figura 76: menú de inicio de sesión.....	73





# 1. Introducción

---

Este Trabajo de Fin de Grado viene motivado por la realización de unas prácticas en la empresa STADLER RAIL VALENCIA S.A.U., desde donde se buscaba generar una aplicación mediante una herramienta de visualización web similar a la aplicación de escritorio que está en uso actualmente.

Actualmente, STADLER RAIL VALENCIA S.A.U. es una empresa que diseña y fabrica vehículos ferroviarios. También se encarga de gestionar en todas sus fases el sistema que controla y monitoriza los vehículos ferroviarios.

La función de este sistema es integrar un equipo que recolecte la información recibida y procesarla durante el funcionamiento del vehículo. Así, dispondremos de información totalmente actualizada sobre el estado del vehículo y sobre posibles eventos no esperados que hayan podido surgir. Gracias a esto, podremos tomar las decisiones adecuadas para realizar el correcto mantenimiento del vehículo.

Dado el avance de las diferentes formas de comunicar, la información obtenida durante los procesos de operación necesita ser precisa. Por ello, se requiere de presentaciones adecuadas, actualizadas y completas.

En este sentido, se busca utilizar herramientas web que aprovechan los navegadores web para mostrar la información. De esta manera, se puede intentar maximizar la implementación de los avances tecnológicos respecto a la interacción del usuario con los diferentes paneles de visualización que puedan ser utilizados en diferentes soportes de hardware.

Desde un inicio se realizó una intensa búsqueda de información relacionada con las posibles herramientas de visualización web que se podrían utilizar. Esto facilitó la posterior decisión por una de estas herramientas como la adecuada para desarrollar el contenido de este proyecto.

## 1.1. Motivación

Como se ha mencionado, este Trabajo de Fin de Grado se inició como unas prácticas en empresa en las que, tras unos meses de dedicación, se decidió que podrían ser adecuadas para un TFG acordado con la empresa. No se generaron diferentes alternativas puesto que el tema para la realización de las prácticas era generar una aplicación similar a la actual mediante una herramienta de visualización web, en cambio, sí se pudo decidir la herramienta de visualización web que se iba a utilizar mediante unos criterios que especificaremos más tarde.

Una de las motivaciones personales que más han influido en mi decisión de hacer este trabajo ha sido que se trataba de un trabajo relacionado con una empresa ferroviaria. Así, mediante este proyecto he podido estar en contacto con datos reales generados a partir de la interacción de las locomotoras con el maquinista y su entorno y he podido generar una visualización que servirá para entender lo que puede estar sucediendo. Adicionalmente, de esta forma he podido aprender del esquema de trabajo de la empresa así como tecnologías nuevas necesarias para el correcto desarrollo de las prácticas. Esto se ha volcado en un enriquecimiento personal a nivel de ciertos conocimientos técnicos en el entorno ferroviario y el funcionamiento de las herramientas utilizadas durante el desarrollo.

## 1.2. Objetivos

Este proyecto parte de la realización de unas prácticas lanzadas para estudiar la posibilidad de generar una visualización diferente a la que se tiene. En la actualidad la herramienta de consulta del estado de los vehículos ferroviarios es una aplicación de escritorio programada y personalizada para las necesidades de la empresa y se pretende obtener una aplicación que ofrezca unas visualizaciones similares pero utilizando un servidor web.

El objetivo definitivo del proyecto es implementar un panel de visualización de datos en el que se muestre de forma clara y conjunta la información de diferentes grupos de vehículos. Además, se pretende confiar la visualización de la información a una herramienta web que aloje el sitio web gráfico y facilite la consulta del sistema de monitorización de los trenes.

De esta forma, se mejora el sistema del cuadro de operaciones proporcionando un dashboard que pueda ser actualizado con un flujo de datos constante a tiempo real cuando la empresa pueda utilizarlo en su sistema, permita cierta interactividad para consultar las medidas y valores que se requieran y muestre alertas cuando se produzcan ciertos comportamientos.

Con todo esto, se pretende proporcionar una solución integral que permita acceder a las visualizaciones de forma intuitiva y eficiente, facilitando la supervisión y el mantenimiento de los vehículos ferroviarios

Para obtener este resultado, se deben cumplir los siguientes objetivos:

- Realizar una revisión de las herramientas de visualización disponibles y las tecnologías a utilizar.
- Analizar y definir de los paneles a implementar para la visualización. También se han de definir los requisitos que han de cumplir los paneles.
- Preparar las herramientas necesarias para transmitir la información a la visualización.
- Desarrollar los paneles adecuados e identificar su función dentro del dashboard.

Una vez cumplidos los diferentes objetivos, podremos documentar los resultados obtenidos para posteriormente presentarlos. También estaremos abiertos a recibir propuestas de mejoras para poder desarrollarlas en el futuro.

Tras una implantación hipotética en los sistemas de la empresa, algunas de las potenciales ventajas de este nuevo sistema serían la menor carga de trabajo de los ordenadores de la empresa puesto que esta herramienta de visualización web gestionaría el apartado gráfico de una forma más eficiente. Además, supondría un avance en el apartado gráfico ya que se dispondrán de los mismos gráficos que antes en cuanto a la información representada, pero con un aspecto totalmente renovado.

## 1.3. Impacto Esperado

La realización de este proyecto tiene un impacto esperado positivo en la empresa, ya que en cuanto se conecte con la base de datos, se podrá tener una visualización de los datos a tiempo real actualizada y con un aspecto totalmente renovado.

Por ello, se espera que con una correcta aplicación del proyecto, en un futuro la interfaz en la que se visualizan los datos que se obtienen de las locomotoras en funcionamiento pueda ser actualizada a esta versión. De esta forma la empresa evitaría tener una aplicación de escritorio y gestionarían los datos que les proporcionan los vehículos ferroviarios de una forma más intuitiva y visual.



Además, este trabajo se encuentra tiene cierto impacto respecto a los Objetivos de Desarrollo Sostenible (ODS). La forma en la que se puede relacionar con ellos se encuentra detallada en el apartado 10.1 del anexo.

#### 1.4. Metodología

Con la finalidad de cumplir los objetivos marcados, hemos seguido cierta metodología durante la realización del trabajo. En primer lugar, y acorde con los objetivos, hemos realizado una revisión de las herramientas de visualización disponibles y las tecnologías que se van a utilizar. Para ello, se recurrirá a sitios web corporativos de las propias herramientas de visualización para obtener ciertas especificaciones técnicas y sitios web que contengan opiniones y valoraciones personales de gente que haya hecho uso de ellas.

Una vez hemos cumplido el primer objetivo, procederemos a analizar y definir de los paneles a implementar para la visualización. Para ello recurriremos a información proporcionada por la empresa en la que se detalla qué tipo de gráficos y tablas serían útiles implementar, siguiendo la línea de su actual aplicación. También definiremos los requisitos que han de cumplir los paneles, es decir, prepararemos una lista con las características, tanto gráficas como de interacción que debería tener.

Una vez planteadas las visualizaciones que vamos a generar, prepararemos la base de datos que se utilizará. Ha de ser una base de datos relacional de la que obtendremos la información necesaria mediante consultas SQL adaptadas en cierta medida al lenguaje de la herramienta de visualización web. Esta base de datos será una captura de datos reales en un momento determinado del tiempo.

En cuanto dispongamos de nuestro repertorio de datos y de la información relacionada con el tipo de visualizaciones que queremos, podremos comenzar a desarrollar los paneles apropiados y a probar su funcionalidad dentro del dashboard. Además, se ha de verificar que la conexión de nuestra base de datos es firme y las consultas funcionan correctamente con el intérprete de la herramienta de visualización.

Estando ya desarrollados los diferentes objetivos, podremos documentar los resultados obtenidos. Como propuestas de mejoras futuras, se pueden recibir sugerencias para desarrollar en mayor medida el proyecto.

#### 1.5. Estructura

En cuanto a la distribución de esta memoria, primeramente tenemos el capítulo 1, la introducción, donde se da a conocer un pequeño contexto de la empresa con la que se ha trabajado; la motivación para realizar este trabajo; los objetivos que se han marcado para desarrollar el proyecto; el impacto que se espera que el trabajo tenga en la empresa; la metodología que se ha seguido para cumplir con los distintos objetivos; la estructura que se sigue a lo largo de la memoria y las convenciones a tener en cuenta.

A continuación, en el capítulo 2, se detalla el estudio del estado del arte que se ha realizado, en él se incluyen una presentación de la empresa; una breve presentación de la herramienta que se usa actualmente y que se pretende mejorar; los requisitos que habría de cumplir la herramienta con la que se decida trabajar; una presentación de las distintas herramientas con las valoraciones respecto a los requisitos y unas conclusiones en las que valoraremos toda la información recogida.



En el siguiente punto, el capítulo 3, se detalla un análisis del problema con aspectos como un análisis de eficiencia algorítmica, el marco legal y ético, riesgos e identificación de soluciones y un plan de trabajo y presupuesto.

En el capítulo 4 se relata cómo se han tratado los datos que se han utilizado para realizar el proyecto. Qué motor de base de datos se ha utilizado, cómo se ha realizado la carga y la estructura que presenta la base de datos utilizada.

Respecto al capítulo 5, contiene en detalle cómo se han hecho y lo que pretenden mostrar los diferentes gráficos y la vista general que se obtiene de la aplicación. Si se precisara de alguna configuración adicional que no se encuentra detallada en este apartado se encontrará una referencia a los anexos.

A continuación, encontramos el capítulo 6, las conclusiones donde se detallará el cumplimiento o no de los diferentes objetivos planteados.

El capítulo 7 son los posibles trabajos a futuro relacionados con este proyecto. Estos pueden ser desde mejoras del proyecto actual hasta una continuación futura del desarrollo de la aplicación.

Antes de finalizar tendremos un pequeño glosario en el capítulo 8, con una serie de palabras definidas para mejorar la comprensión de algunos detalles del trabajo.

Después tenemos el capítulo 9, la bibliografía, donde estarán detalladas las fuentes de información.

Por último, encontramos el capítulo 10 que recoge un anexo con una serie de indicaciones y explicaciones. En primer lugar, se detallan los Objetivos de Desarrollo Sostenible y posteriormente se recogen la instalación y correcta configuración tanto de PostgreSQL como de Grafana y ciertos *plug-in* y extensiones que se deben instalar antes de desplegar el dashboard.

## 1.6. Convenciones

En este apartado se detallan las diversas convenciones que se deben tener en cuenta a la hora de leer el documento:

- El código fuente se muestra en letra Courier New. Únicamente se empleará esta tipología para este tipo de contenido.
- Por norma general, las palabras extranjeras se remarcarán en cursiva.
- Se entrecomillarán con comillas dobles las citas textuales. Por norma general, solo el texto extraído de forma textual estará entrecomillado con comillas dobles, aunque pueden estar presentes en el código fuente.
- Respecto a la forma de señalar las referencias, se ha utilizado el estándar ISO 690 con referencias numéricas.
- Por norma general, el documento está escrito en español, exceptuando algunos tecnicismos y los títulos de los paneles gráficos en su sección correspondiente. Dichos títulos, no se encontrarán marcados en cursiva.
- Debido al carácter confidencial de los datos, a pesar de estar ya anonimizados en gran parte, se han censurado partes de las imágenes expuestas y partes del código utilizado en las consultas.



## 2. Contextualización y estado del arte

---

### 2.1. La empresa

Este TFG ha sido realizado en el ámbito de unas prácticas en la empresa STADLER RAIL VALENCIA S.A.U., empresa localizada en el polígono industrial del Mediterráneo de Albuixech, Valencia. Es una multinacional cuyo ámbito de trabajo es la fabricación de vehículos ferroviarios con sede en la ciudad de Bussnang, Suiza y tiene con diversas filiales en países como: Argelia, Hungría, Italia, Austria, Polonia, entre otros.

A nivel mundial la empresa cuenta con más de 13.000 empleados especializados en diversos ámbitos, desde el diseño de vehículos hasta técnicos especializados en montaje. También cuenta con ingenieros dedicados al desarrollo de nuevas tecnologías dedicadas al ámbito ferroviario.

Antes de ser adquirida, la fábrica de ferrocarriles en Valencia tuvo varios propietarios a lo largo de su historia. Todo empieza en 1897, momento en el que se estableció bajo el nombre de Talleres Devis. En 1929, tras adquirir cierta importancia en el ámbito ferroviario, empezó su expansión fuera de España. En 1947 se fusionó con otra empresa y pasó a ser MACOSA (Material y Construcciones SA). No sería hasta 1989 cuando Alstom, de origen francés, adquirió la empresa y en 1997 se trasladó a actual en Albuixech. Ya en el año 2005, Vossloh adquiriría la compañía y la mantendría hasta el 2016.

Desde el 1 de enero de 2016, Stadler ha estado establecido en Valencia, España. En esa fecha, Stadler adquirió el negocio de locomotoras de Vossloh en España y lo integró a su propia empresa. Esta adquisición ha sido una incorporación perfecta a la gama de productos de Stadler Rail, permitiéndole además acceder a nuevos mercados. El centro de Stadler en Valencia se dedica a la fabricación de diversos vehículos ferroviarios cuyo ámbito de uso es tanto urbano como interurbano. [1]

Stadler Rail Valencia S.A.U. ha cerrado acuerdos de producción de trenes para diferentes lugares y operadores tanto nacionales como pueden ser Alicante o Cataluña, como internacionales como el consorcio formado por seis operadores alemanes y austriacos a quienes se les atribuye el mayor contrato de la empresa [2] o la firma galesa Wales&Borders [3].

### 2.2. Introducción

Actualmente, la empresa dispone de una herramienta llamada SYSLOC, que se utiliza para la visualización de los datos de telemetría que envían los trenes. Ofrece la visualización para distintos perfiles de usuario y de forma que puedan acceder a conocer el estado en el que se encuentran los trenes y el análisis y la resolución de incidencias. En la figura 1 podemos ver una visión de algunos paneles que muestra SYSLOC.

Sin embargo, se pretende recurrir a una herramienta de *Business Intelligence* para servir de ayuda para la toma de decisiones enfocadas al entorno ferroviario. Las soluciones de *Business Intelligence* consisten en transformar los datos en información útil para así proporcionar el conocimiento adecuado para la toma de decisiones.

Para abordarlo, se comenzará por recabar información sobre aquellas herramientas que puedan resultar útiles para llevarlo a cabo. Recordemos que el objetivo es, en última instancia, mejorar el sistema del cuadro de operaciones proporcionando un dashboard que pueda ser actualizado con un flujo de datos constante a tiempo real cuando la empresa pueda utilizarlo en su sistema. Además también debe permitir cierta interactividad para consultar las medidas y



valores que se requieran y genere alertas cuando se produzcan ciertos comportamientos. Para ello, se deben tener en cuenta ciertos aspectos técnicos necesarios para mantener un nivel de calidad.

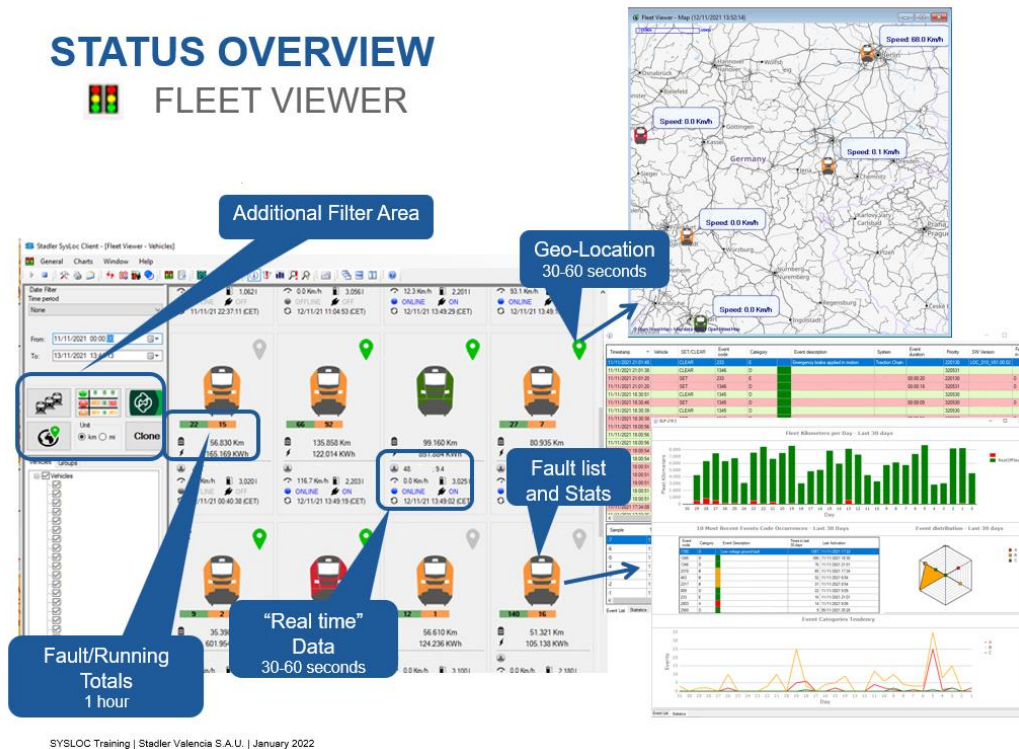


Figura 1: herramienta SYSLOC

### 2.3. Requisitos

Con el propósito de tener una lista acotada de candidatos, tener unos criterios de selección es fundamental. Hay requisitos que pueden cumplirse en cierto grado.

- **Gráficos:** la herramienta ha de contener una diversa paleta de gráficos por defecto. Además, estos deberían tener cierto dinamismo integrado y serán personalizables en cierto grado.
- **Comunidad de desarrolladores:** también se valorará que la plataforma desarrolladora ofrezca una amplia comunidad de desarrolladores en la que la participación sea abierta y esté disponible para los usuarios. Se valorará según la cantidad de seguidores que cada plataforma tiene en GitHub o en foros propios de la plataforma.
- **Geolocalización:** se tendrá en cuenta que la herramienta disponga de métodos para poder procesar coordenadas geográficas y así poder graficar mediante los mapas adecuados.
- **Desarrollo de controles:** además de los controles ya integrados, estudiar si se pueden generar nuevos controles o modificar ligeramente los ya existentes.
- **Buscador interno:** disponer de un buscador interno que permita consultar parámetros y medidas mediante filtros. Adicionalmente, se puede considerar que incluyan un buscador general que ayude a encontrar más fácilmente las medidas que se quieran consultar.



- Alertas: la posibilidad de poder programar ciertas alertas si se dan unas condiciones concretas o simplemente que actúen a modo de recordatorios programables. También se considerará si se pueden conectar a algún *bróker* de mensajería como Kafka.
- Licencia: se ofrecen diferentes tipos de licencias, desde herramientas de código abierto hasta otras que requieren un abono mensual. Se priorizará a aquellas que ofrezcan la mejor calidad-precio, es decir, que ofrecen diversas funcionalidades a un precio adecuado o sean de código abierto.
- Despliegue: puesto que se pretende realizar un despliegue *on-premise*, es principal tener en cuenta esta función.
- Conexión con fuentes de datos: se debe tener en cuenta que las herramientas puedan mantener una conexión a tiempo real con los datos almacenados en la base de datos.

## 2.4. Herramientas estudiadas

### Grafana

Se trata de una aplicación web de código abierto [4] desarrollada por un equipo independiente. Presenta una amplia gama de gráficos personalizables y dinámicos. Además, tiene una comunidad de desarrolladores activa con una gran cantidad de seguidores en el GitHub de Grafana Labs.

En cuanto a los gráficos que ofrecen localización, Grafana ofrece la opción de generar mapas con información geográfica e incluso modificar los iconos que representen los puntos. También da la posibilidad de implementar un buscador mediante filtros, un sistema de gestión de alertas [5] y permite la integración de funciones con Kafka [6].

Aquí se puede ver la figura 2, un dashboard creado a partir de datos de un Tesla Model 3, se puede observar la localización del coche a lo largo del tiempo, las actualizaciones que ha recibido, temperatura y otros datos que se encontraban disponibles. La visualización es muy atractiva y se ajusta a los requisitos planteados.

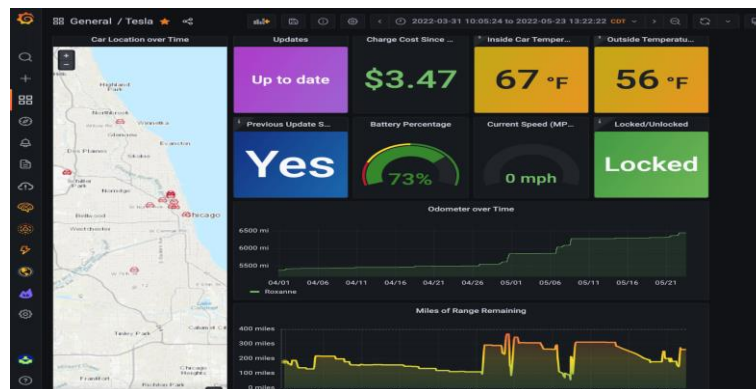


Figura 2: dashboard Tesla Model 3 [7]

Por último, Grafana se trata de una herramienta de código abierto, por lo que no necesita una suscripción al servicio. A pesar de ello, existen planes premium [8] con los que ampliar la capacidad del servicio.



## Chronograf

Pertenece al grupo InfluxData. Es una herramienta de código abierto, aunque se pueden ampliar las funcionalidades mediante una suscripción. Contiene variedad de gráficos por defecto pero estos son más orientados a datos estructurados a modo de series temporales.

Por otro lado, estos gráficos también se pueden implementar a modo de mapas y mostrar la información localizada según las coordenadas. Chronograf permite generar reglas [9] para mandar alertas si se dan unas condiciones. Además, estas se pueden integrar con el *bróker* Kafka. En cuanto a la comunidad de desarrolladores para tener mayor garantía de seguimiento, podemos tomar como referencia la plataforma de la empresa raíz, InfluxData.

La figura 3 muestra un ejemplo de cómo se podría configurar un dashboard con esta herramienta. Como se había mencionado, esta representación contiene gráficos que se enfocan más en la evolución de los datos en el tiempo.

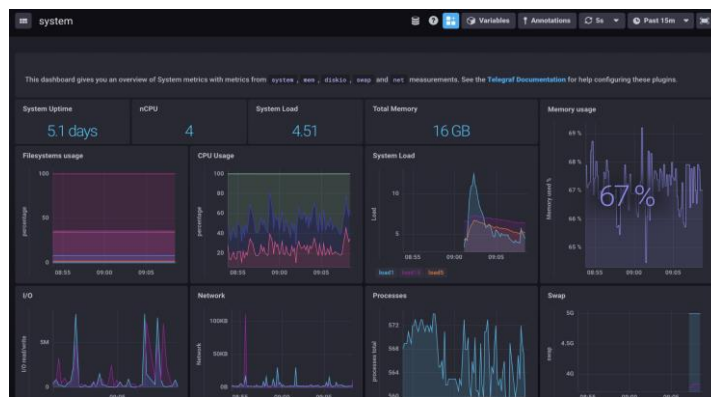


Figura 3: dashboard de series temporales

Respecto a la licencia necesaria, cabe decir que se trata de una plataforma gratuita, pero que, al igual que otras opciones valoradas, se pueden contratar planes [10] para aumentar las capacidades en el caso de que se necesite.

## Tableau

Herramienta desarrollada por el grupo Salesforce. Se trata de una opción mediante suscripción. Ofrece gran cantidad de interacciones con los gráficos, haciendo que estos sean muy dinámicos. Su comunidad de desarrollo se encuentra activa, tanto respecto al desarrollo de la aplicación como su repositorio GitHub.

En cuanto a las opciones que ofrece destaca la posibilidad de implementar gráficos utilizando la localización, la posibilidad de implementar un buscador mediante filtros, así como especificar ciertos parámetros para crear alertas. Además, permite su utilización con datos a tiempo real.

Respecto a la figura 4, se pretende observar las conexiones existentes con el objetivo de obtener información para realizar una futura ampliación de la red. Por ello podemos ver un mapa con las conexiones actuales, en el que si se selecciona una conexión concreta, se puede ver la información asociada.



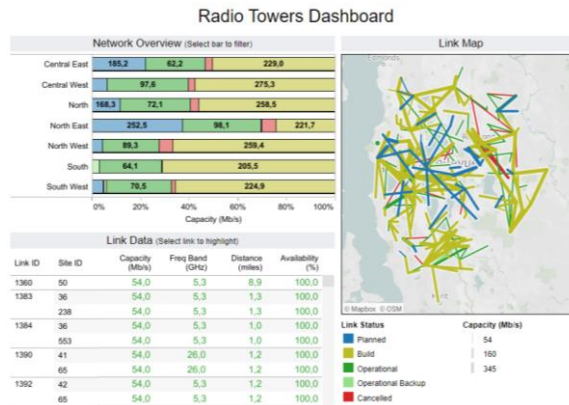


Figura 4: dashboard de conexiones red [11]

En este caso, podemos ver en la figura 5 como se trata de un dashboard interactivo en el que se puede observar la distribución de la población según los estados así como su población y el porcentaje de población según el rango de edad.

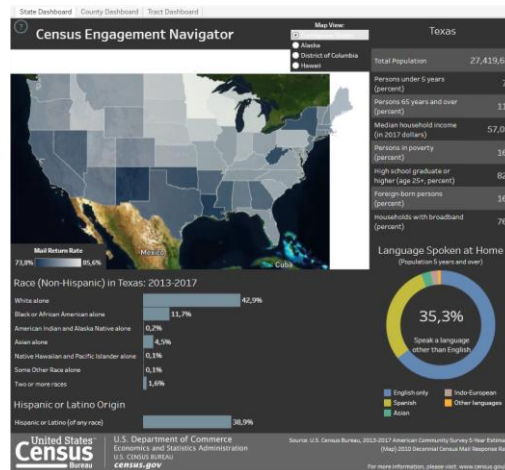


Figura 5: dashboard sobre el censo de EE. UU. [12]

Para trabajar con esta herramienta, se necesita una licencia que se puede obtener mediante la suscripción a uno de sus planes [13].

## Power BI

Desarrollada por Microsoft, ofrece gran variedad de gráficos. Estos también pueden ser geolocalizados, además posibilita la integración de filtros y un buscador dentro del propio panel de filtros para encontrar más fácilmente aquellos que nos interesen.

En cuanto a la comunidad de desarrollo, a pesar de tener una pequeña comunidad en GitHub, Microsoft ofrece un amplio foro en el que consultar información. Por otra parte, Power BI permite integrar el *broker* Kafka y visualizar datos recibidos en tiempo real.

En cuanto a algunos ejemplos, podemos ver en la figura 6, un dashboard que contiene diferentes gráficos de tarta que muestran descuentos aplicados a productos y los beneficios obtenidos de los productos. Además, también tiene una tabla resumen de cada continente y



destaca los valores más importantes en 6 cuadros en la parte superior. Por último, contiene un filtro en el que se pueden ajustar los diferentes requisitos.

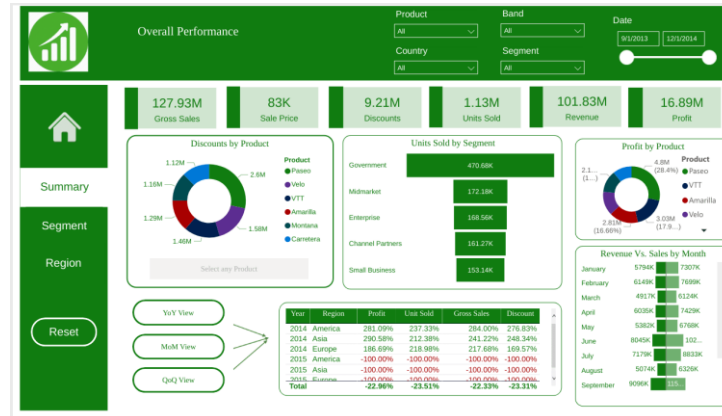


Figura 6: dashboard de análisis de ventas [14]

En cambio, en este caso la figura 7 muestra como habían funcionado diferentes campañas de marketing sobre seis determinados productos. Los resultados se muestran mediante diferentes gráficos de barras y un gráfico de dispersión que muestra la distribución de los clientes según la edad, el gasto realizado y la educación. Además, también se muestra información destacada en la parte superior del dashboard.



Figura 7: dashboard sobre una campaña de marketing [15]

Para disponer de esta herramienta, es necesaria una licencia que se obtiene mediante la suscripción a un plan mensual [16].

## Kibana

Pertenece al grupo Elastic, ofrecen diferentes pruebas gratuitas de su contenido. También ofrecen diversidad de gráficos y algunos integran opción de geolocalización. Se pueden incluir filtros y utilizarlos a modo de buscadores.

La empresa raíz, Elastic, ofrece un repositorio GitHub con más de mil doscientos seguidores. Además, existe un foro online en el que consultar información específica sobre Kibana. Por otro lado, Kibana ofrece un sistema de alertas [17] que pueda detectar cambios no



deseados en los datos que se reciben a tiempo real o activar avisos cuando se dan unas condiciones establecidas.

Algunos ejemplos de dashboard hechos con esta herramienta son los siguientes [18]. En primer lugar, la figura 8, utiliza un mapa de calor en el que se muestra la densidad de votos emitidos según la región del país.

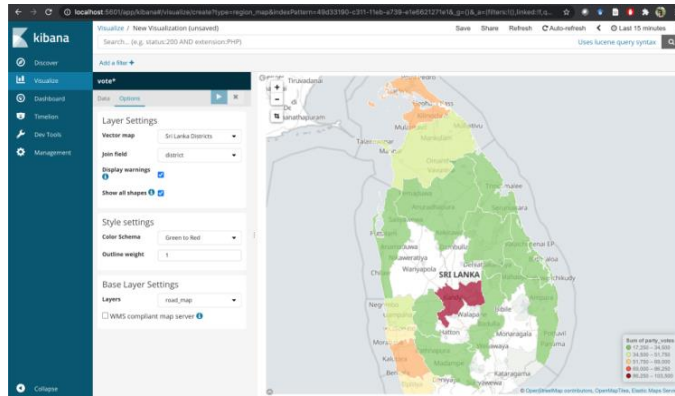


Figura 8: conteo de votos

Por otro lado, en la figura 9 vemos el control de un firewall y muestra los bloqueos por región, el total de bloqueos, los bloqueos por país, el top 15 de los puertos más bloqueados y una línea del tiempo en la que se muestran los bloqueos en el tiempo. Para esto hace uso de gráficos de barras, normales y apilados, mapas y gráficos de tarta.

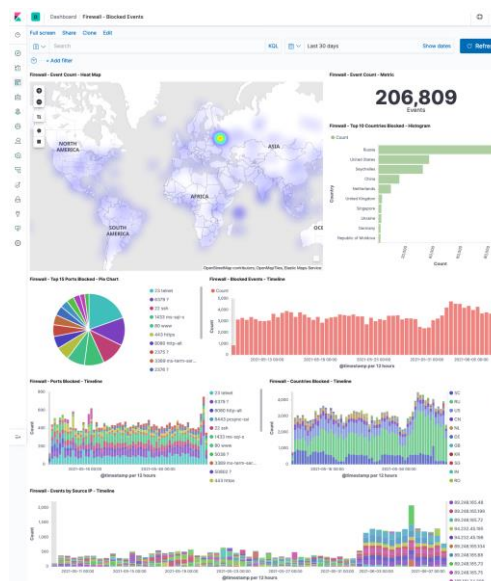


Figura 9: monitoreo de un firewall

Para obtener acceso a la plataforma, no se requiere una suscripción, ya que existe un plan gratuito. Pero al igual que sucede con otras plataformas, si se quieren ampliar las capacidades de la aplicación, es necesaria una suscripción [19].



## Qlik

Es del grupo Qlik y ofrece una herramienta elaborada a nivel gráfico. Proporciona la posibilidad de generar cuadros de mando interactivos que integren gráficos con datos de localización geográfica. En cuanto a la comunidad de desarrollo, Qlik no posee una extensa comunidad de GitHub, en cambio ofrece un foro en el que los desarrolladores pueden realizar preguntas y dar respuestas a cuestiones abiertas.

Permite la integración de un buscador en el que poner las palabras clave a buscar y generar unas consultas automatizadas sobre los datos. Adicionalmente, permite establecer alertas para que manden avisos si se dan ciertas condiciones. Para obtener acceso a Qlik, es necesaria la suscripción a uno de sus planes [20].

Algunos ejemplos de visualización, proporcionados por la propia plataforma [21] recogidos en la figura 10 y la figura 11, muestran cómo se pueden incluir gráficos de calor, de barras, de dispersión, de barras apiladas o de líneas entre otros. En el primer ejemplo podemos ver que se trata de un dashboard sobre datos clínicos mientras que el otro ejemplo trata sobre un resumen de la eficiencia de algunos equipos electrónicos.

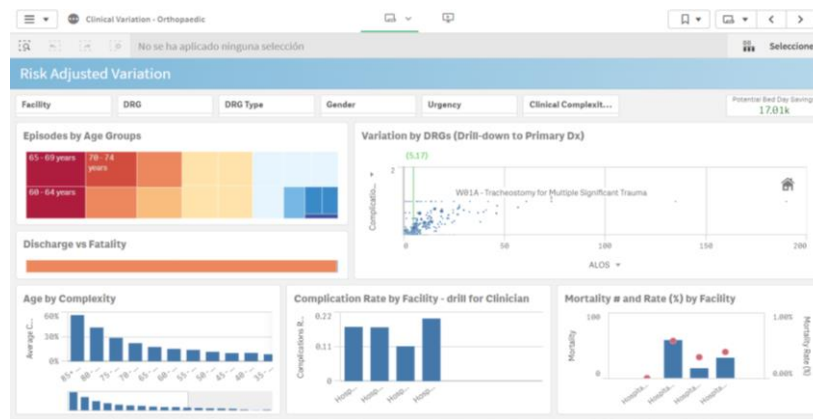


Figura 10: dashboard sanitario

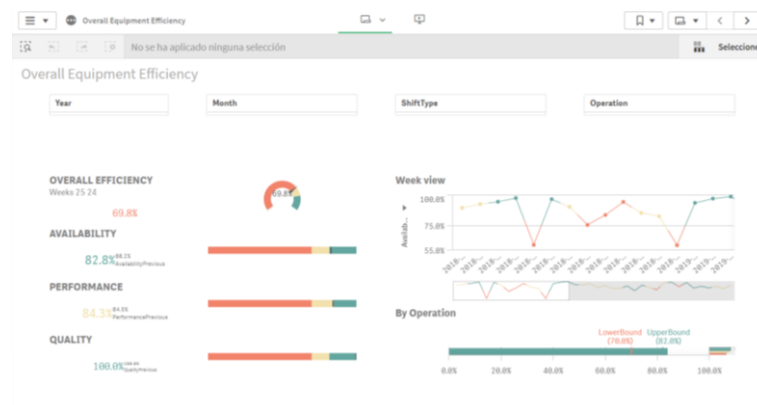


Figura 11: eficiencia de los equipos

## Otras herramientas

Otras herramientas que se consideraron en este análisis pero no cumplieron suficientes requisitos, fueron: Domo, Zoho Analytics, Freeboard, Redash, Cyclotron, DataDog, Cyfe y Dashthis.

Algunos de los motivos por los que fueron descartadas son: su poca popularidad y por tanto reducida comunidad de desarrollo, el hecho de que no serían suficientemente variadas como para cumplir con las expectativas requeridas, el tipo de instalación permitida no era el adecuado o que la herramienta estaba orientada a otro propósito como el análisis del flujo de datos de una web.

### 2.5. Reseña del capítulo

Analizando los resultados obtenidos, de las herramientas expuestas únicamente se han considerado las detalladas minuciosamente, ya que durante la recopilación de información fueron las más destacadas. De entre estas seis herramientas, se descartarán Tableau y Power BI debido a que no se ajustan a uno de los requisitos principales, tratarse de una herramienta web. Estas herramientas se tuvieron en consideración principalmente por el amplio abanico de opciones gráficas que ofrecían.

Por otro lado, de las herramientas restantes, se descartaron dos herramientas más, Chronograf y Qlik, puesto que la primera está más orientada a tratar con series temporales y la segunda porque al comparar las opciones que ofrecía con las opciones de las herramientas restantes, Kibana y Grafana, estas últimas parecían mejores opciones.

Por ello, decidimos que cabría la posibilidad de empezar a trabajar con Grafana y Kibana, aunque la principal herramienta que se va a tener en cuenta es Grafana ya que de entre todas las opciones consideradas, es la que mejores características ofrece.



### 3. Análisis del problema

---

En el entorno actual, la gestión y análisis de datos juegan un papel fundamental en la toma de decisiones estratégicas. Para ello, es crucial contar con herramientas eficientes que permitan visualizar de manera clara y concisa la información. En este campo, Grafana es una solución popular y potente para la visualización web, mientras que SQL se posiciona como el lenguaje de consulta asociado puesto que tratamos con datos almacenados en bases de datos relacionales.

Sin embargo, a pesar de las ventajas que ofrecen Grafana y SQL, su implementación efectiva puede presentar desafíos y problemas que deben ser abordados. Es fundamental realizar un análisis exhaustivo de los problemas que puedan surgir para comprender las dificultades potenciales y buscar soluciones adecuadas.

El objetivo de este análisis es examinar los obstáculos que pueden surgir al implementar Grafana con SQL en la visualización de datos, y proporcionar recomendaciones para superarlos. Se explorarán aspectos como la configuración de Grafana, la integración con bases de datos utilizando SQL, la optimización de consultas y la creación de paneles interactivos.

En resumen, este análisis explorará los desafíos y problemas asociados con la implementación de Grafana en la visualización de datos, con el objetivo de proporcionar recomendaciones para superarlos. Al identificar y abordar estos obstáculos, se podrá aprovechar mejor el potencial que ofrece para obtener la visualización de datos que se requiere.

#### 3.1. Análisis energético o de eficiencia algorítmica

En el desarrollo de un dashboard de visualización de datos es esencial considerar la eficiencia energética y computacional. La visualización de datos a través de Grafana requiere una gestión óptima de recursos para garantizar un rendimiento eficiente y minimizar el consumo energético y de memoria. Además, el uso adecuado del lenguaje SQL en la consulta puede influir en la eficiencia del procesamiento y en la utilización de los recursos del sistema.

El objetivo de este análisis es evaluar la eficiencia en la implementación de Grafana y las consultas SQL utilizadas en la visualización de datos, identificar posibles ineficiencias y proporcionar recomendaciones para optimizar su rendimiento. Se pueden explorar diferentes aspectos, como la configuración de Grafana para que los colores de las visualizaciones obtenidas sean más oscuros y así a la hora de desplegarlo, necesita menos brillo que se traduce en un ahorro energético.

En cuanto a SQL, se evaluará la eficiencia de las consultas utilizadas para acceder y manipular datos en la base de datos. Se pueden analizar y mejorar aspectos como la complejidad de las consultas y la optimización de la selección de datos para minimizar el consumo energético y mejorar la velocidad de ejecución.

El análisis de eficiencia algorítmica y energética permite identificar posibles áreas de mejora y ofrecer recomendaciones para optimizar el consumo de energía y mejorar el rendimiento tanto respecto a la velocidad como a la eficiencia en la visualización de datos.

En resumen, este análisis se ha enfocado en evaluar la eficiencia energética en la implementación de Grafana y SQL en la visualización de datos, con el objetivo de identificar ineficiencias y proporcionar sugerencias para optimizar el consumo de energía y mejorar el rendimiento. Al considerar la eficiencia algorítmica y energética en la implementación de estas herramientas, se logrará una visualización de datos más eficiente y sostenible e igualmente válida.



### 3.2. Análisis del marco legal y ético

Es fundamental tener en cuenta el marco legal y ético que rige la protección de datos, la propiedad intelectual y los principios éticos relacionados con el uso y tratamiento de la información. La correcta aplicación de estos aspectos legales y éticos no solo garantiza el cumplimiento de las normativas vigentes, sino que también promueve la confianza, la transparencia y la responsabilidad en la utilización de los datos.

#### Protección de Datos

En el ámbito de la protección de datos, es esencial cumplir con las leyes y regulaciones aplicables, como el Reglamento General de Protección de Datos (GDPR) en la Unión Europea. Estas regulaciones establecen los principios fundamentales para el manejo de datos personales, incluyendo la obtención de consentimiento informado, la seguridad de la información y el respeto de los derechos individuales.

En la implementación del dashboard con Grafana, se debe garantizar que se cumplan los requisitos de privacidad y seguridad de los datos. Además, se tratan de datos pertenecientes a casos reales de la empresa por lo que su acceso está muy limitado a personal exterior. Stadler tiene una red privada y esto implica que el acceso a los datos y servicios de la empresa únicamente se pueden realizar desde un equipo interno de la empresa. En el caso de realizar una conexión exterior, será necesaria una autenticación de doble factor. En el caso de este proyecto, el sistema de visualización de datos se instalará en un servidor propio de la empresa por lo que se encontrará accesible para uso interno de la empresa. Además, los datos utilizados para llevar a cabo este trabajo se anonimizaron previamente, de forma que no es posible la identificación de los diferentes trenes.

#### Propiedad Intelectual

En lo referente a la propiedad intelectual, se debe considerar la licencia de uso de Grafana, que es de código abierto por lo que la licencia básica es gratuita. También se debe considerar la licencia de uso del motor de la base de datos PostgreSQL, que al tratarse también de una herramienta de código abierto su licencia es gratuita. En cuanto a la propiedad de los datos utilizados, son propiedad de Stadler Rail Valencia S.A.U.

#### La ética en el uso de los datos

Además del marco legal, es esencial considerar los principios éticos en la implementación de la visualización de datos. Esto implica garantizar el buen uso de los datos para evitar la discriminación y el sesgo en la visualización de datos, asegurando que las representaciones gráficas sean claras, imparciales y comprensibles. A pesar de que en este caso se trata en su mayoría de datos mecánicos y de sensores, no hay que dejar de lado este enfoque.

#### Conclusión

En resumen, la implementación de Grafana y SQL en la visualización de datos requiere considerar el marco legal y ético en relación con la protección de datos, la propiedad intelectual y los principios éticos. Al abordar estos aspectos legales y éticos, se promueve una implementación responsable y ética del proyecto, lo que contribuye a un uso adecuado y beneficioso de los datos en la toma de decisiones.





### 3.3. Análisis de riesgos e identificación de soluciones

Al implementar Grafana y SQL en la visualización de datos, es importante considerar los posibles riesgos y desarrollar estrategias para solucionarlos. Estos riesgos pueden afectar a la seguridad de los datos, la integridad de la visualización y la privacidad de los usuarios. A continuación, se presenta un análisis de los riesgos más comunes y se proponen soluciones para abordarlos de manera efectiva.

#### 1. Riesgo de seguridad de datos.

Un posible riesgo es la exposición de datos sensibles o violación de la seguridad de la información debido a vulnerabilidades en Grafana.

Esto se podría solucionar mediante medidas de seguridad robustas, como el uso de conexiones cifradas (HTTPS) o requerir una autenticación al iniciar el dashboard. Además, hay que mantener las actualizaciones de seguridad al día.

#### 2. Riesgo de acceso no autorizado.

Otro riesgo posible es el acceso no autorizado a la plataforma Grafana o a los datos almacenados en la base de datos de PostgreSQL.

Para evitar este problema, se recomienda el uso de contraseñas seguras como medida de autenticación sólida. Además, restringir el acceso a las funciones y datos solo a los usuarios autorizados.

#### 3. Riesgo de falta de capacidad de procesamiento.

Puede darse el caso de que el dashboard resultante del proyecto presente cierta incapacidad para manejar grandes volúmenes de datos en el momento en el que se realice una prueba de conexión con la base de datos real.

Por ello, se propone que las consultas presenten la mayor optimización posible para mejorar el rendimiento. El proyecto se trata de una versión beta de uno a mayor escala que resultará en la creación de una nueva aplicación, por lo que este riesgo cabe estudiarlo a largo plazo.

#### 4. Riesgo de incompatibilidad.

Este riesgo consiste en que la versión desarrollada no sea compatible totalmente con la base de datos de la empresa al utilizar un motor de base de datos diferente.

En este caso, una solución adecuada es estudiar las diferencias significativas entre el motor de base de datos de la empresa y PostgreSQL, el utilizado durante el desarrollo del proyecto. De esta forma, puesto que ambos se basan en el lenguaje SQL pero con diferencias sutiles en la utilización de algunas funciones usadas en las consultas

### Conclusión

La implementación de Grafana y SQL en la visualización de datos conlleva riesgos potenciales que deben tenerse en cuenta. Hay que destacar que cabe la posibilidad de que existan más riesgos no identificados diferentes a los expuestos.

Al implementar las soluciones propuestas, como la mejora de la seguridad de los datos, la implementación de medidas de acceso autorizado, la optimización de las consultas y la identificación de incompatibilidades, se puede reducir significativamente la exposición a los riesgos.



### 3.4. Plan de Trabajo

1. Revisión del estado del arte.
  - Recopilación de herramientas
  - Identificación de fortalezas
2. Análisis de requisitos.
  - Identificar los objetivos y requerimientos específicos de la visualización a partir de la información ofrecida por la empresa.
  - Determinar las fuentes de datos y la estructura de la base de datos SQL.
  - Establecer las métricas a visualizar.
3. Diseño de la arquitectura.
  - Definir la infraestructura necesaria para alojar Grafana y la base de datos SQL.
  - Establecer la configuración de Grafana, incluyendo paneles, gráficos y alertas.
  - Diseñar el esquema de la base de datos SQL y definir el objetivo de las consultas necesarias para obtener los datos.
4. Implementación de Grafana y SQL.
  - Instalar y configurar Grafana en el entorno deseado.
  - Crear la base de datos SQL y definir las tablas y relaciones necesarias.
  - Desarrollar las consultas SQL para extraer y procesar los datos requeridos.
  - Verificar que los datos se encuentran en la base de datos.
5. Integración de datos.
  - Configurar las conexiones entre Grafana y la base de datos SQL.
  - Importar los datos existentes a la base de datos SQL, si es necesario.
  - Verificar que la conexión extrae los datos de la forma adecuada.
6. Desarrollo de paneles y visualizaciones.
  - Estudiar el panel a desarrollar
  - Extraer los datos necesarios para realizar el panel
  - Configurar las opciones de visualización del panel
  - Verificar que la información se muestra correctamente
7. Pruebas y optimización.
  - Probar el correcto funcionamiento de las variables
  - Optimizar consultas y configuraciones para mejorar el rendimiento y la eficiencia en el procesamiento de datos.
  - Validar la veracidad de las visualizaciones generadas.
8. Implementación.
  - Revisar que los nombres y etiquetas sean correctos
  - Realizar pruebas finales para asegurar que todo funcione correctamente

En este punto es en que el proyecto se encuentra, el próximo punto a abordar sería un hipotético despliegue en los equipos de la empresa. Este despliegue se podría desarrollar como un trabajo a futuro para tener en cuenta.



### 3.5. Presupuesto

El presupuesto para la implementación de este TFG viene influido por diversos factores, como los recursos necesarios tanto software como hardware o la duración. A continuación, se presentan algunos aspectos a considerar en el presupuesto.

1. Recursos humanos.
  - Tiempo dedicado a la realización del proyecto.
2. Licencias y herramientas.
  - Costo de las licencias de Grafana. En este caso se trata de una herramienta de código abierto por lo que el coste es nulo, pero si se desease explorar otras opciones, estas podrían ser de pago.
  - Costo de la licencia de PostgreSQL. Al igual que Grafana, esta es una herramienta de código abierto por lo que el coste es también nulo a no ser que se desee buscar otras opciones.
3. Infraestructura y servicios.
  - Para poder desarrollar y desplegar la visualización, se ha necesitado acceso a un ordenador con acceso a la red. Por ello, es conveniente tener disponer de un ordenador no necesariamente muy potente con acceso a la red.
4. Capacitación y formación.
  - Costos de capacitación y formación en Grafana y PostgreSQL. Estos costos se relacionan principalmente con el proceso de aprendizaje del uso de la herramienta. Además, si no se tienen nociones básicas en lenguaje SQL, subirá exponencialmente. No necesariamente se trata de un costo económico, pero sí en gran medida de un coste de tiempo de la persona o grupo implicado en el desarrollo.
5. Datos utilizados.
  - Aquí se detalla el coste de los datos utilizados. Para realizar este proyecto se ha utilizado un set de datos proporcionado por la empresa, por lo que el coste de adquisición es nulo.

Es importante evaluar cuidadosamente estos aspectos y estimar los costos asociados a cada uno de ellos. Además, es recomendable incluir un margen adicional para imprevistos y contingencias que puedan surgir durante la implementación del proyecto.

Por lo tanto, el coste de tiempo podrá variar según si lo realiza una persona con capacitación previa o no, siendo menor el tiempo necesario si se tiene conocimientos previos. Mientras que el coste económico de la realización del TFG es nulo a no ser que se utilice una herramienta que requiera una suscripción o se quiera ampliar las capacidades de la actual.



## 4. Preparación y comprensión de los datos

---

El desarrollo del proyecto es un proceso clave para obtener información significativa y valiosa a partir de conjuntos de datos contenidos en una base de datos relacional. Sin embargo, antes de poder realizar visualizaciones efectivas, es necesario preparar los datos y comprender su estructura y contenido.

La preparación de datos implica una serie de pasos que permiten limpiar, transformar y organizar los datos para su posterior análisis y visualización. De esta forma obtenemos resultados precisos y significativos en las visualizaciones.

En nuestro caso, esta preparación no fue tan intensa puesto que los datos vinieron proporcionados por la empresa en varios archivos de Excel a partir de los que se redactarían las consultas correspondientes para añadir dichos datos a la base de datos relacional creada.

La comprensión de la estructura de la base de datos también es necesaria para entenderlos adecuadamente. Gracias a esta comprensión profunda, se podrá tener conocimiento de las variables relevantes y así comprender la información que se mostrará en las visualizaciones generadas.

Además, es esencial comprender el contexto y el dominio de los datos. Esto permite obtener conocimientos adicionales sobre los datos, validar la interpretación de los resultados y garantizar que las visualizaciones sean relevantes y significativas.

Como se ha mencionado anteriormente, los datos utilizados provienen de un documento Excel con una serie de tablas. De esta forma se representaba la estructura que debía seguir la base de datos.

Además, respecto a la anonimización de los datos, se nos proporcionaron prácticamente anonimizados, es decir, se retiraron aquellos datos que permitían identificar los vehículos ferroviarios de los que se tenían datos. A pesar de esta primera fase realizada por la empresa, las imágenes y consultas SQL que se detallan más adelante se encontrarán censuradas puesto que hay nombres de ciertas variables que no se pueden exponer.

En primer lugar, necesitaremos descargar e instalar el motor de base de datos, véase el punto 10.3.1 en el anexo. Utilizaremos el motor de bases de datos PostgreSQL ya que es de código abierto y permite la conexión de sus bases de datos con Grafana.

Por otra parte, una vez redactadas las consultas que contienen los datos que deben estar en la base de datos, procedemos a ejecutarlas para que se carguen los datos a la base de datos y se generen las conexiones entre tablas establecidas. Para consultar cómo hacerlo, véase el punto 10.3.3 en el anexo.

Por último, configuraremos el entorno de ejecución que utilizaremos para probar el correcto funcionamiento de las consultas que pasaremos al intérprete de Grafana. En este caso utilizaremos Visual Studio Code como herramienta en la que realizar las pruebas. Para realizar la configuración, consultar el apartado 10.3.4 recogido en el anexo.

En cuanto a la distribución de las tablas en la base de datos, el diagrama de clases consta de 6 clases en forma de tablas, como vemos en la figura 12. La tabla 'LOCOMOTIVES', que recoge los identificadores de las locomotoras que se utilizarán para relacionarlos con el nombre de la locomotora; la tabla de 'DATAGPS', que recoge la localización geográfica y velocidad de las locomotoras en distintos momentos de tiempo; la tabla 'REALTIME', que recoge ciertos datos del estado de la locomotora así como la última posición recibida de la locomotora; la tabla de



‘TOTALIZERS’, que recoge los valores de distintas variables de las locomotoras; la tabla de ‘FUEL’, que recoge un histórico de la cantidad de combustible restante de la locomotora; y finalmente la tabla de ‘EVENTS’, que tiene registrados los eventos y sus características asociadas.

En todas las tablas se incluye el ID de la locomotora, que funcionará como clave para relacionar las distintas tablas. Además, la cardinalidad de las relaciones es 1 a muchos, esto quiere decir que una locomotora contenida en la tabla ‘LOCOMOTIVES’, puede tener desde 0 hasta N registros por cada identificador en las tablas asociadas con esta relación. Por otra parte, la relación 1..1 con la tabla ‘REALTIME’ pone de manifiesto que únicamente habrá un registro por cada locomotora y al menos habrá uno, puesto que en teoría, serían valores a tiempo real. También están detallados los tipos de datos correspondientes a cada uno de los atributos de las clases y su cardinalidad, siendo 0..1 el valor por defecto de estas cardinalidades.

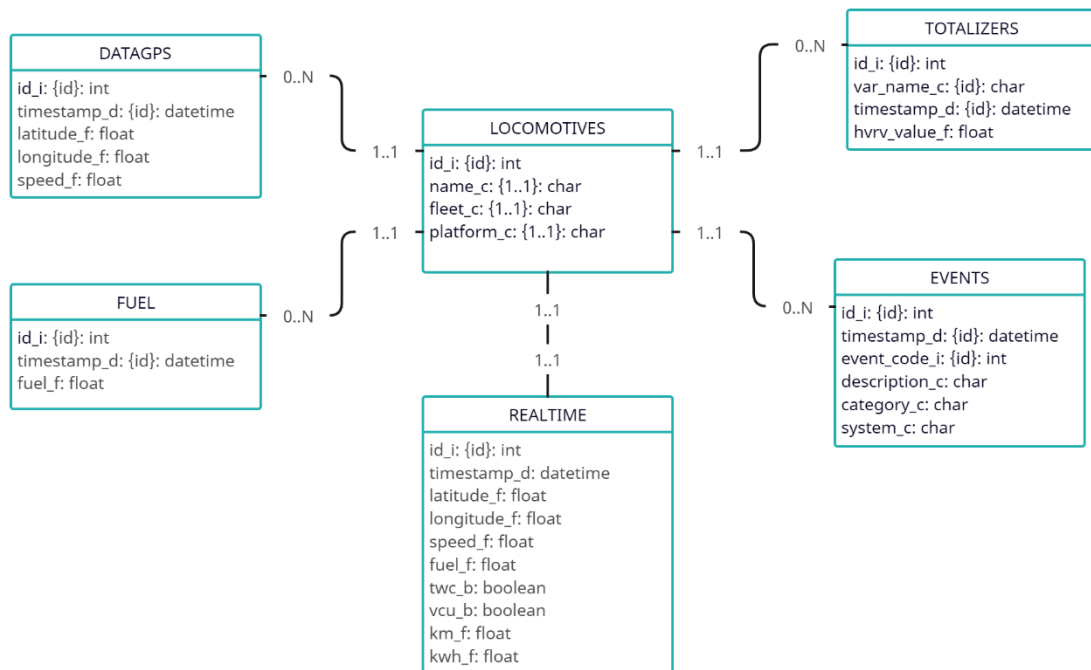


Figura 12: diagrama de clases

## 5. Configuración y Despliegue

---

Esta sección se inicia con una breve introducción en la que se recuerda el objetivo inicial del proyecto, la forma en la que se ha ido cumpliendo y qué herramientas hemos utilizado. Después, en la sección 5.2, 5.3 y 5.4 se detalla cómo hemos de utilizar el dashboard creado y cómo se distribuyen los paneles en las diferentes secciones. A continuación, a lo largo de la sección 5.5, realizamos un recorrido a través de cada uno de los paneles para conocer qué muestran y cómo se han configurado. En la sección 5.6, finalmente presentamos en los resultados la vista general de cada una de las secciones.

### 5.1. Introducción

Este proyecto tiene como objetivo crear un dashboard que ofrezca una visualización a tiempo real de forma que sirva de ayuda para el proceso de toma de decisiones. El proyecto se inició con un estudio del estado del arte en el que se analizarían las diversas opciones y nos decantaríamos por una de ellas, en este caso, Grafana. Una vez decidida la herramienta de visualización que se iba a utilizar, comenzó el proceso de aprendizaje de la herramienta. Durante este proceso se ampliarían las capacitaciones en lenguaje SQL para realizar consultas adecuadas para el intérprete de Grafana, también se estudiarían los diferentes gráficos que se ofrecen para realizar las visualizaciones y las opciones para modificarlos que ofrecen e incluso cierto uso de la terminal del dispositivo.

Para ello, se instaló Grafana y el motor de base de datos PostgreSQL en el equipo. En la sección 10.3 y 10.4 se detalla la instalación y configuración de las herramientas correspondientemente. Se utilizará este motor de base de datos porque Grafana dispone de un *plug-in* gratuito con el que poder conectar el dashboard con la base de datos de PostgreSQL. Además, como entorno de ejecución donde se realizarían pruebas de las consultas SQL se utilizaría Visual Studio Code con el que tendríamos acceso a los datos cargados en PostgreSQL a través de una extensión.

### 5.2. Uso y organización del dashboard

Este dashboard se ha organizado en dos grandes secciones, una para controles relacionados con los vehículos ferroviarios en particular y otra para controles relacionados con la flota en conjunto. Además, hay una serie de variables en la zona superior que utilizaremos para decidir qué flotas y qué vehículos de estas flotas mostraremos.

Por una parte, la sección relacionada con la flota muestra en primer lugar los gráficos relacionados con la información de la flota a nivel general. Respecto a la información de los eventos, la encontramos en la zona inferior de la sección, tras la información general de la flota.

Por otra parte, la sección relativa a la información de los vehículos en particular se encuentra organizada de forma muy similar, con la información general sobre ellos en la zona superior y los eventos ocurridos en la parte inferior de la sección.

Estas secciones están individualizadas y tendremos tantas réplicas como flotas o locomotoras hayamos seleccionado en la variable 'Fleet' o 'Locomotive' para las flotas o los vehículos ferroviarios correspondientemente.

Estas secciones se pueden reducir para facilitar la interacción con el dashboard simplemente apretando en el título de la sección.



### 5.3. Interfaz de usuario

Nos encontramos con un dashboard que nos proporciona dos grupos de información. Por un lado disponemos de la información relacionada con las flotas seleccionadas en la variable 'Fleet' y, por otro lado, se nos ofrece la información sobre los vehículos seleccionados que pertenezcan a la alguna de las flotas escogidas.

En cuanto a la sección relacionada con la flota, dispondremos de diversas tablas, gráficos y mapas que nos ayudarán a comprender el estado de la flota.

En la parte superior se encuentra una tabla con información actualizada de los vehículos correspondientes, un mapa con la posición de las locomotoras de la flota y también hay tres paneles que muestran el nombre, la energía y la distancia total recorrida de la flota.

A partir de la zona central se encuentra la información relacionada con los eventos sucedidos: la tabla de sistemas, una tabla descriptiva de los eventos, tanto los recientes como los más frecuentes, así como un contador del tipo de evento y un mapa con su localización. También hay un gráfico de radar en el que se muestra cuántos fallos ha habido de cada tipo.

La siguiente sección se corresponde con la información de los vehículos en particular.

En la zona superior están diversos paneles informativos y una tabla resumen con información ya mostrada en los paneles anteriores e información nueva.

En la parte central hay dos gráficos que muestran la velocidad y el combustible restante y un mapa con la localización del vehículo correspondiente.

Finalmente, se muestra un gráfico en el que se pueden observar los kilómetros recorridos por el vehículo respecto de su flota a lo largo del tiempo, un gráfico de radar en el que se visualiza la cantidad de ocurrencias de cada uno de los eventos registrados, un gráfico contabilizador de los eventos similar al descrito en la sección anterior y un mapa que registra la localización de los eventos.

### 5.4. Variables del dashboard

Las variables se encuentran situadas en la zona superior del dashboard como podemos ver en la figura 13. Su principal función es ser un filtro con el que ajustar los datos que mostrarán los gráficos según si pertenecen a la locomotora o a la flota seleccionada. Las diferentes opciones se muestran en un menú que se despliega al seleccionarla.

Variable	Función	Gráficos en los que se usa	Palabra clave en la consulta
<b>Locomotive</b>	Nombre del vehículo ferroviario.	En todos de la sección de los vehículos.	\$loco_name
<b>Fleet</b>	Nombre de la flota.	En todos de la sección de flota.	\$fleet

Tabla 1: variables

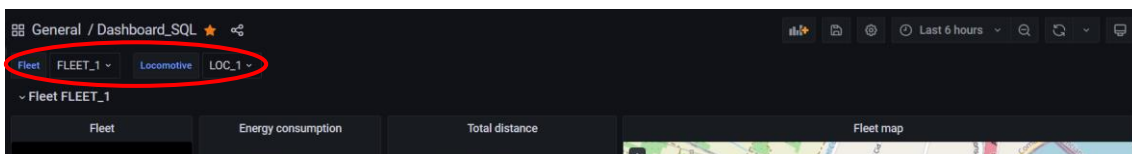


Figura 13: localización de las variables

## 5.5. Consultas SQL y configuración de controles

Con el objetivo de clarificar la utilidad de algunas consultas SQL que se han realizado, se agruparán de forma global las más simples y aquellas que requieran más enfoque se detallarán. En primer lugar explicaremos todo aquello relacionado con la flota y después lo relacionado con las locomotoras.

Además, dentro de cada uno de los dos grupos los paneles se dividen de forma que en primer lugar se muestra la información relacionada con las características de la flota o el vehículo en general. Como podemos ver en la figura 14 se muestran paneles como su nombre, la distancia recorrida, su situación actual, entre otros.

Por otro lado, una vez vista la información general, tenemos los eventos que han sucedido en la flota o los vehículos como vemos en la figura 14. Para ello disponemos de paneles como pueden ser un mapa con la distribución de los eventos, una tabla con los eventos más recientes y otra con los más frecuentes o un gráfico de radar en el que se muestra la cantidad de fallos correspondientes a cada uno de los sistemas. La figura 14 se corresponde con la sección relativa a la flota, algunos paneles difieren según la sección a la que pertenecen.

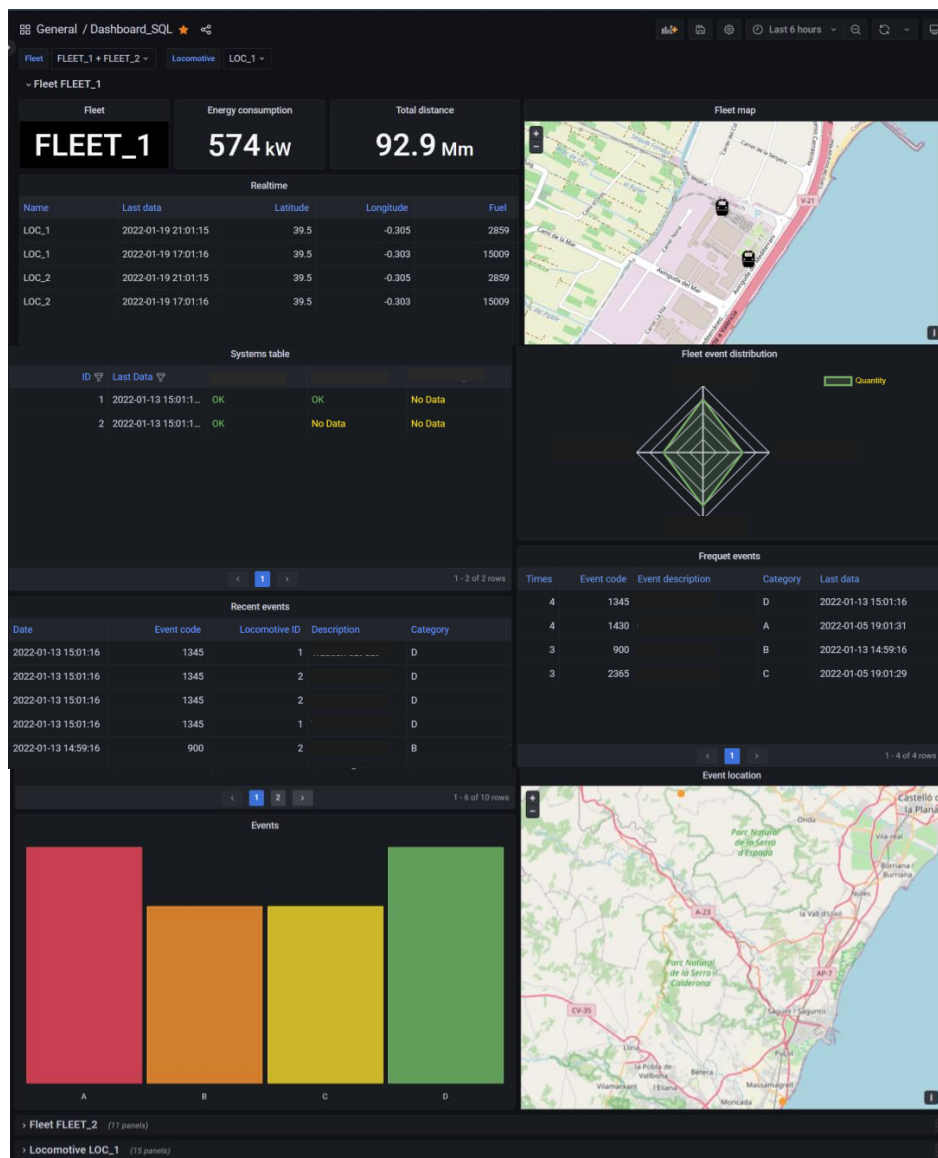


Figura 14: distribución de paneles





### 5.5.1. Flota

En primer lugar mostraremos la sección correspondiente con los paneles de la flota. Para ello, seleccionaremos una flota de entre las disponibles en la variable. A continuación se detallarán uno a uno todos los paneles implementados y sus configuraciones.

En la zona superior de la sección encontramos paneles y tablas que muestran información general de la flota como el valor del consumo de energía, la distancia total, la etiqueta de la flota o una tabla con valores. Estas consultas SQL son las más simples y constan de una cláusula 'SELECT' únicamente restringida mediante el 'WHERE' para que los datos pertenezcan a la flota correspondiente. Podemos ver en la figura 15 la forma en la que se distribuyen.

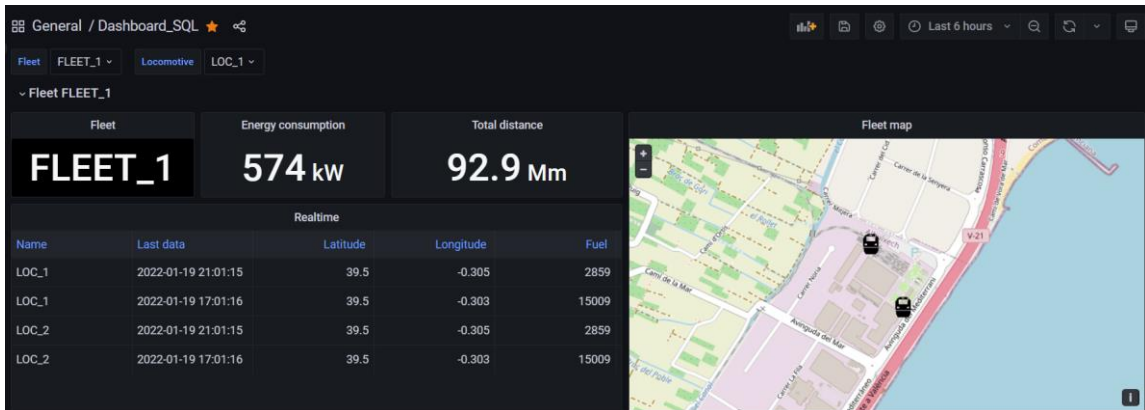


Figura 15: paneles generales

#### Fleet map

Este mapa pretende mostrar la ubicación de los vehículos de la flota correspondiente a tiempo real. Para ello, obtiene la latitud y longitud de la tabla 'REALTIME' únicamente de las locomotoras pertenecientes a la flota adecuada. Además, para cambiar el icono que viene por defecto, en el apartado *map layers*, se configuran los parámetros como se muestran en la imagen. Cabe destacar que el icono a introducir ha de estar en formato 'SVG', además para mejorar la apariencia, se aumenta la opacidad a 1. La locomotora utilizada como punto de referencia ha sido proporcionada por la empresa. Por último, el mapa seleccionado como base es el *Open Street Map*.

En la figura 16 se muestra la visualización obtenida con algunas de las configuraciones utilizadas. Además, las demás figuras de la sección muestran los parámetros que hay que configurar para obtener la visualización.

Consulta SQL:

```
1.     SELECT latitude_f AS "Latitude",
longitud_f AS "Longitude"
2.     FROM realtime
3.     WHERE
4.         realtime.ID_I IN
5.         ( SELECT LOCOMOTIVES.ID_I
6.           FROM LOCOMOTIVES
7.           WHERE $fleet = LOCOMOTIVES.FLEET_C );
```

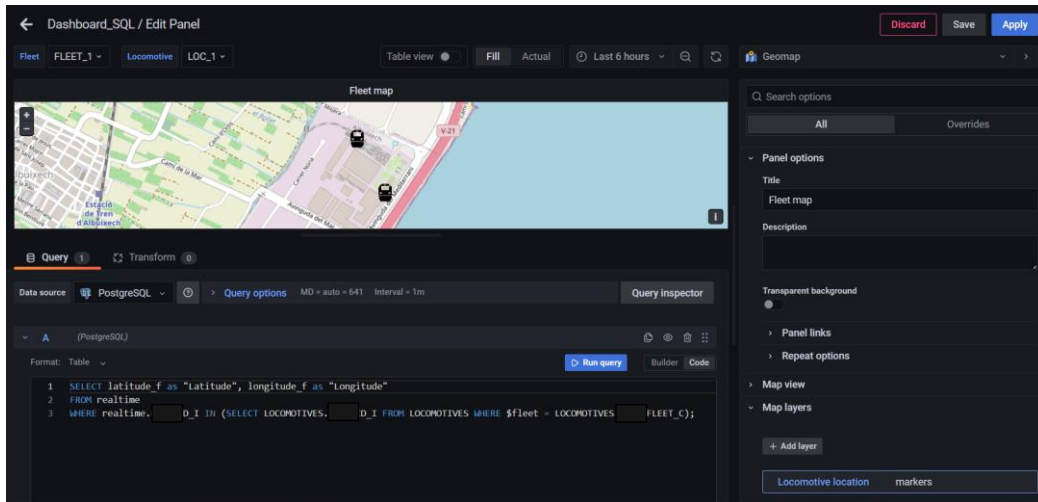


Figura 16: mapa de la flota

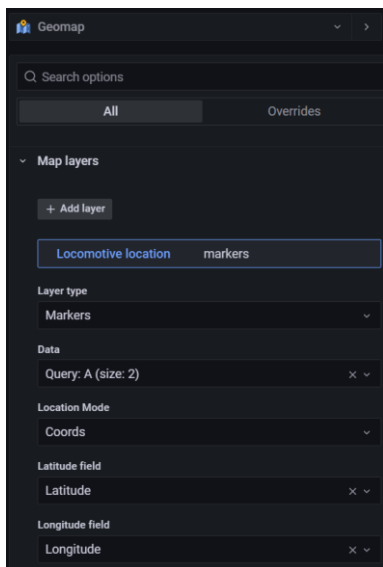


Figura 17: configuración de controles

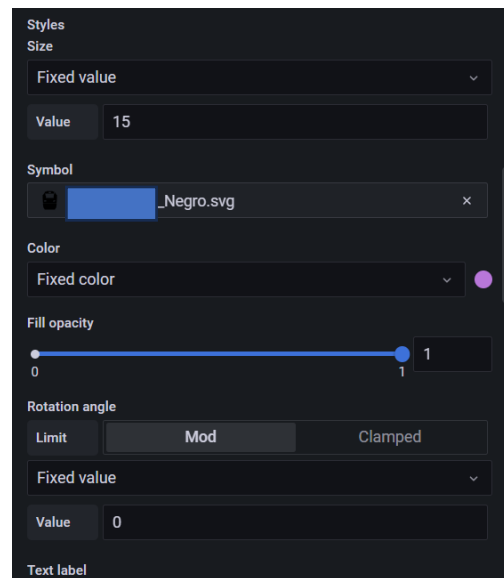


Figura 18: configuración de controles

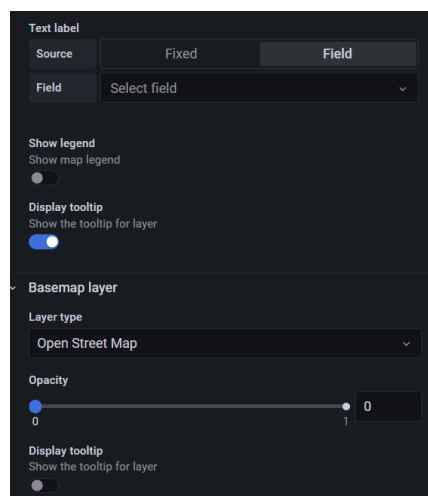


Figura 19: configuración de controles

## Systems table

Ahora pasamos a explicar la tabla de sistemas que podemos ver en la figura 20. Esta tabla muestra si cierto sistema del vehículo se encuentra en buen estado, no o no hay datos. La consulta que proporciona los datos de esta tabla hace uso de una función especial llamada 'crosstab', que sirve para pivotar y obtener los datos de forma horizontal. Antes de utilizarla, hay que activar el módulo que contiene esta función desde el SQL Shell:

```
$$ CREATE EXTENSION IF NOT EXISTS tablefunc;
```

Esta función se sitúa en el 'FROM' de forma que el 'SELECT' correspondiente puede seleccionar todo aquello que esta función devuelva. Dentro de la función 'crosstab', se redacta una consulta simple con las restricciones que queramos, la cual se situará como primer argumento. A continuación, se redacta otra consulta que devuelva los valores que se van a expandir horizontalmente. Por último, especificamos los campos que se van a devolver y el tipo de cada uno.

Una vez disponibles los datos con la consulta SQL, realizamos un postratamiento que se basa en redefinir los eventos según la categoría, es decir, si son de categoría A pasan a ser NOK (Not OK) y estará escrito en rojo, si son de tipo B pasan a ser NOK (Not OK) y escrito en naranja y si son de categoría C, D o E pasará a verse OK y su color será el verde. Por último, si no hay datos de una determinada columna, se mostrará No Data en color amarillo. Esta configuración la podemos ver en la figura 21 y la figura 22.

Consulta SQL:

```
1.  SELECT *
2.  FROM crosstab( 'select id_i, timestamp_d,
system_c, category_c
3.  from eventos
4.  WHERE eventos.ID_I in (SELECT LOCOMOTIVES.ID_I
FROM LOCOMOTIVES WHERE LOCOMOTIVES.FLEET_C = '${fleet}')
5.  ORDER BY ID_I, TIMESTAMP_D DESC,
SYSTEM_C', 'select distinct system_c from eventos Order by
system_c') AS rootTable
6.  ("ID" INT, "Last
Data" TIMESTAMP, "system1" VARCHAR, "system2" VARCHAR, "sy
stem3" VARCHAR, "system4" VARCHAR, "system5" VARCHAR, "sys
tem6" VARCHAR);
```

ID	Last Data	System1	System2	System3	System4	System5
1	2022-01-13 15:01:1...	OK	OK	No Data	NOK	NOK
2	2022-01-13 15:01:1...	OK	No Data	No Data	No Data	NOK

Figura 20: tabla de sistemas



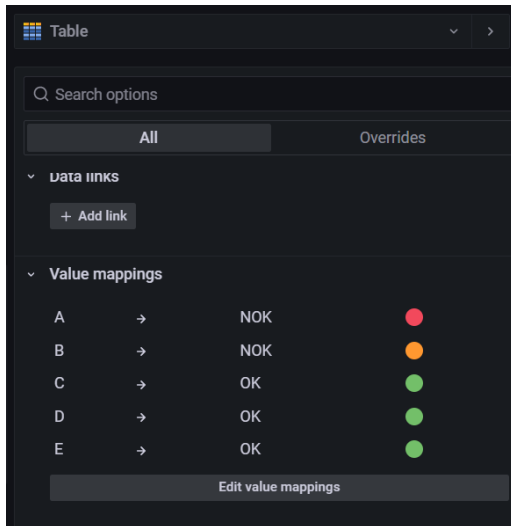


Figura 21: configuración de la tabla

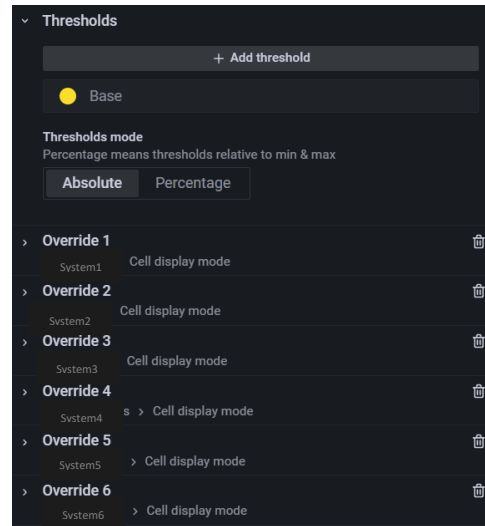


Figura 22: configuración de la tabla

### Fleet event distribution

Este gráfico muestra la distribución de los eventos según la cantidad de registros existentes de cada uno de sistemas. Para ello se utiliza un gráfico de radar en cuyos vértices se situarán los diferentes sistemas, como vemos en la figura 23.

Para realizar este gráfico primero hay que instalar el *plug-in* correspondiente, podemos consultar cómo hacerlo en el apartado 10.4.4 situado en el anexo.

Respecto a la configuración, tras elegir este modo de visualización, en la consulta correspondiente elegimos el formato de series temporales. En cuanto a la consulta, hemos provocado que la serie temporal sea únicamente de un único momento, el más reciente, y realice un *count* para sumar los eventos de distinto tipo que hay según el sistema. Todos estos valores los obtenemos mediante subconsultas en la cláusula 'FROM'. Por último, en la configuración, le damos el alias de 'Quantity' a la consulta A. Podemos ver esta configuración en la figura 24.

Consulta SQL:

```

1.  SELECT
2.    $__time(tiempo.intervalo),
3.    contador.cont,
4.    contador.tipo
5.  FROM
6.    ( SELECT MAX(timestamp_d) AS intervalo
7.      FROM eventos
8.      WHERE eventos.ID_I IN (
9.        SELECT LOCOMOTIVES.ID_I
10.       FROM LOCOMOTIVES
11.       WHERE $fleet=LOCOMOTIVES.FLEET_C)
12.    ) AS tiempo,
13.    ( SELECT COUNT(EVENTOS.CATEGORY_C) AS cont,
14.      system_c AS tipo
15.      FROM eventos
16.      WHERE eventos.ID_I IN (

```



```

17.      SELECT  LOCOMOTIVES.ID_I
18.      FROM    LOCOMOTIVES
19.      WHERE   $fleet=LOCOMOTIVES.FLEET_C)
20.      GROUP BY system_c) AS contador;

```

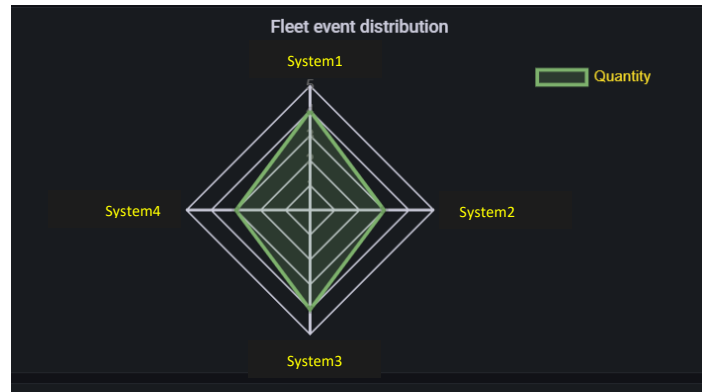


Figura 23: distribución de eventos de la flota

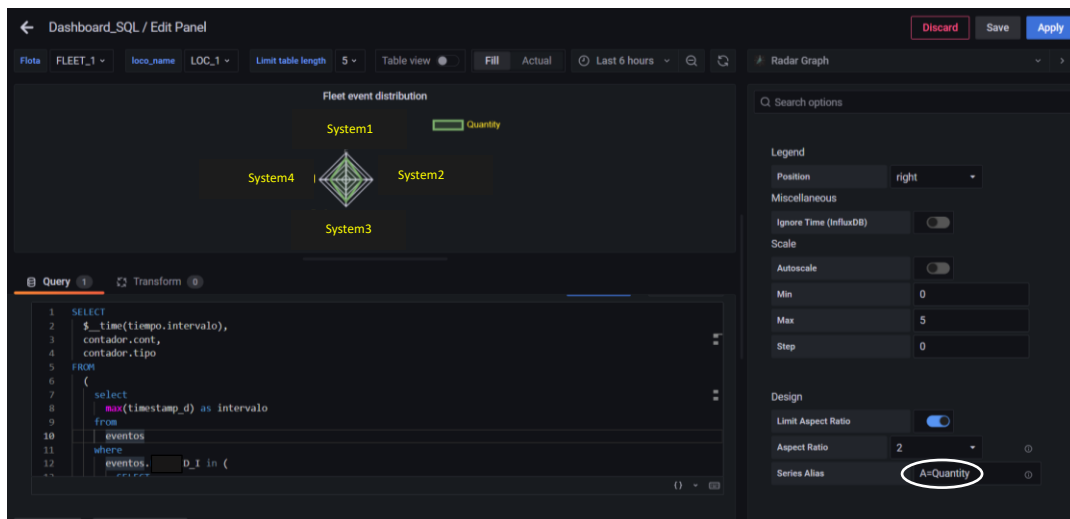


Figura 24: configuración de controles

### Recent events

En esta tabla mostramos los 10 eventos más recientes y cierta información sobre ellos, como podemos ver en la figura 25. Además, hemos habilitado la paginación para que aquellos que no quepan en la primera página, se muestren en las consecutivas y no mediante una barra deslizante en una sola página.

### Consulta SQL:

```

1. SELECT timestamp_d AS "Date", event_code_i AS "Event
   code", eventos.id_i AS "Locomotive ID",
   description_c AS "Description", category_c AS "Category",
   system_c AS "System"
2. FROM eventos, locomotives

```



```

3. WHERE eventos.ID_I IN (SELECT LOCOMOTIVES.ID_I FROM LOCOMO
   TIVES WHERE $fleet = LOCOMOTIVES.FLEET_C) AND LOCOMOTIVES.
   ID_I IN (SELECT LOCOMOTIVES.ID_I FROM LOCOMOTIVES WHERE $f
   leet = LOCOMOTIVES.FLEET_C)
4. ORDER BY timestamp_d DESC
5. LIMIT 10;

```

Date	Event code	Locomotive ID	Description	Category	System
2022-01-13 15:01:16	1345	1	Description1	D	System1
2022-01-13 15:01:16	1345	2	Description2	D	System2
2022-01-13 15:01:16	1345	2	Description3	D	System3

Figura 25: tabla de eventos recientes

### Frequent events

En la figura 26 vemos la tabla que muestra los eventos más frecuentes. Los eventos se mostrarán en diferentes páginas de la tabla si estos no caben en una sola. La consulta SQL es:

```

1. SELECT COUNT(event_code_i) AS "Times",
   event_code_i AS "Event code", description_c AS "Event
   description",
   category_c AS "Category", MAX(timestamp_d) AS "Last data"
2. FROM eventos
3. WHERE eventos.ID_I IN (SELECT LOCOMOTIVES.ID_I FROM LOCOMO
   TIVES WHERE $fleet = LOCOMOTIVES.FLEET_C)
4. GROUP BY event_code_i, description_c, category_c
5. ORDER BY "Times" DESC;

```

Times	Event code	Event description	Category	Last data
4	1345	Description1	D	2022-01-13 15:01:16
4	1430	Description2	A	2022-01-05 19:01:31
3	900	Description3	B	2022-01-13 14:59:16
3	2365	Description4	C	2022-01-05 19:01:29

Figura 26: tabla de eventos frecuentes



## Events

En este panel se genera un gráfico de barras como el de la figura 27 donde se clasificarán los diferentes eventos al mismo tiempo que se contarán. De esta forma cuantos más eventos de un tipo haya, mayor será la barra. Además, según el color de la barra se podrá saber de qué tipo es el evento.

Para obtener los datos de este gráfico recurrimos a una consulta SQL que devuelve la categoría del evento y un recuento de cada categoría. Para dar color a cada una de las categorías, especificamos que si es A, tenga el color rojo; si es B, sea naranja; si es C, sea amarillo y en cualquier otro caso, será verde.

Consulta SQL:

```
1. SELECT COUNT(EVENTOS.CATEGORY_C) AS "Quantity",  
EVENTOS.CATEGORY_C AS "Category"  
2. FROM EVENTOS  
3. WHERE EVENTOS.ID_I IN (SELECT LOCOMOTIVES.ID_I FROM  
LOCOMOTIVES WHERE $fleet = LOCOMOTIVES.FLEET_C)  
4. GROUP BY eventos.CATEGORY_C;
```

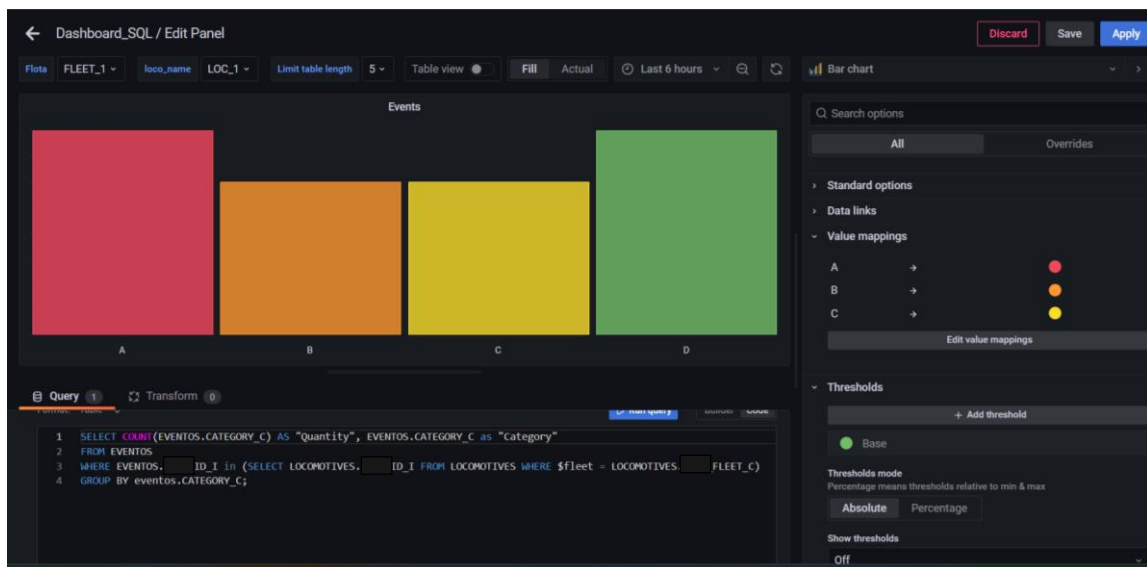


Figura 27: gráfico de eventos

## Event location

Este es el último gráfico de esta sección, se trata de un mapa en el que se mostrará la ubicación del último evento ocurrido de un sistema de cada vehículo. Es decir, por cada vehículo de la flota, se mostrará un evento por cada uno de los sistemas en el caso de que haya registrado un evento. Se puede ver en la figura 28.

Para realizar este mapa necesitamos las coordenadas de las locomotoras de la flota, la fecha del evento y los datos relacionados con el evento que queramos que se muestren al pasar el cursor sobre el punto. Además, se mostrarán los eventos que sean más recientes, es decir, de un mismo tipo de evento únicamente se mostrará la última incidencia de este tipo con las coordenadas más cercanas a la fecha de este evento. Para obtener la información necesaria, empezamos realizando



subconsultas a partir de las cuales vamos extrayendo información sin perder el identificador de los datos, para así después poder combinar y comparar la información de las tablas implicadas.

A la hora de preparar la configuración, separaremos cada consulta según la categoría de los eventos para mostrar puntos de distinto color en el mapa. El color se corresponderá con la categoría.

Además, para poder realizar el correcto cálculo de diferencia de fechas para obtener la localización más cercana del evento, necesitamos programar la función de 'DATEDIFF', que en PostgreSQL no viene implementada. Para ello, recurrimos a la función ofrecida en la documentación [22]. Para crearla, la ejecutaremos en nuestro entorno.

Para poder configurar los diferentes tipos de eventos con sus respectivos colores, hemos creado varias capas y a cada una de ellas se le ha asignado una de las consultas según si extrae eventos de categoría A, B... Así cada categoría tendrá su color correspondiente.

Función 'DATEDIFF':

```
1.      CREATE OR REPLACE FUNCTION DateDiff (units
VARCHAR(30), start_t TIMESTAMP, end_t TIMESTAMP)
2.          RETURNS INT AS $$
3.      DECLARE
4.          diff_interval INTERVAL;
5.          diff INT = 0;
6.          years_diff INT = 0;
7.      BEGIN
8.          IF units IN ('yy', 'yyyy', 'year', 'mm', 'm
', 'month') THEN
9.              years_diff = DATE_PART('year',
end_t) - DATE_PART('year', start_t);
10.
11.              IF units IN ('yy', 'yyyy', 'year') THEN
12.                  RETURN years_diff;
13.              ELSE
14.                  -- If end month is less than start
month it will subtracted
15.                  RETURN years_diff
* 12 + (DATE_PART('month', end_t) - DATE_PART('month',
start_t));
16.              END IF;
17.          END IF;
18.          -- Minus operator returns interval 'DDD
days HH:MI:SS'
19.          diff_interval = end_t - start_t;
20.
21.          diff = diff + DATE_PART('day',
diff_interval);
22.
23.          IF units IN ('wk', 'ww', 'week') THEN
24.              diff = diff/7;
25.              RETURN diff;
26.          END IF;
27.
28.          IF units IN ('dd', 'd', 'day') THEN
```





```

29.         RETURN diff;
30.     END IF;
31.
32.     diff = diff * 24 + DATE_PART('hour',
diff_interval);
33.
34.     IF units IN ('hh', 'hour') THEN
35.         RETURN diff;
36.     END IF;
37.
38.     diff = diff * 60 + DATE_PART('minute',
diff_interval);
39.
40.     IF units IN ('mi', 'n', 'minute') THEN
41.         RETURN diff;
42.     END IF;
43.
44.     diff = diff * 60 + DATE_PART('second',
diff_interval);
45.     RETURN diff;
46. END;
47. $$ LANGUAGE plpgsql;

```

Consulta SQL:

```

1.     SELECT
2.     DISTINCT ON (table1.sistema, table1.loco)
3.     table1.categoria AS "Category",
4.     table1.Loco AS "Locomotive",
5.     table1.fecha AS "GPS Date",
6.     table1.Latitude AS "Latitude",
7.     table1.Longitude AS "Longitude",
8.     table1.sistema AS "System",
9.     table1.tiempo AS "Event date"
10.    FROM
11.    (
12.        SELECT
13.            dataGPS.id_i AS Loco,
14.            DATAGPS.timestamp_d AS fecha,
15.            dataGPS.latitude_f AS Latitude,
16.            dataGPS.longitude_f AS Longitude,
17.            tiempo1.categoria,
18.            tiempo1.sistema,
19.            tiempo1.locomotiveIDEvent,
20.            tiempo1.tiempo,
21.            MIN(
22.                ABS (
23.                    DATEDIFF (
24.                        'second',
25.                        DATAGPS.timestamp_d :: TIMESTAMP,
26.                        tiempo1.tiempo :: TIMESTAMP
27.                    )

```



```

28.     )
29.     ) AS DIFMIN
30. FROM
31.     datagps,
32.     (
33.         SELECT
34.             eventos.timestamp_d AS tiempo,
35.             eventos.system_c AS sistema,
36.             eventos.category_c AS categoria,
37.             EVENTOS.id_i AS locomotiveIDEvent
38.         FROM
39.             eventos
40.         WHERE
41.             eventos.id_i IN (
42.                 SELECT
43.                     locomotives.id_i
44.                 FROM
45.                     locomotives
46.                 WHERE
47.                     locomotives.fleet_c = $fleet
48.             )
49.         GROUP BY
50.             categoria,
51.             sistema,
52.             locomotiveIDEvent,
53.             tiempo
54.     ) AS tiempo1
55. WHERE
56.     datagps.id_i IN (
57.         SELECT
58.             locomotives.id_i
59.         FROM
60.             locomotives
61.         WHERE
62.             locomotives.fleet_c = $fleet
63.     )
64. AND dataGPS.id_i = tiempo1.locomotiveIDEvent
nt
65. AND DATAGPS.timestamp_d BETWEEN tiempo1.ti
empo - INTERVAL '5 MINUTE'
66. AND tiempo1.tiempo + INTERVAL '5 MINUTE'
67. GROUP BY
68.     dataGPS.id_i,
69.     DATAGPS.timestamp_d,
70.     dataGPS.latitude_f,
71.     dataGPS.longitude_f,
72.     tiempo1.categoria,
73.     tiempo1.sistema,
74.     tiempo1.locomotiveIDEvent,
75.     tiempo1.tiempo
76. ORDER BY
77.     difmin ASC,
78.     dataGPS.id_i,

```



```

79.         dataGPS.latitude_f,
80.         dataGPS.longitude_f,
81.         tiempo1.categoria,
82.         tiempo1.sistema,
83.         tiempo1.locomotiveIDEvent
84.     ) AS table1
85. WHERE
86.     table1.categoria = 'A'
87. ORDER BY
88.     table1.sistema,
89.     table1.loco,
90.     difmin asc;

```

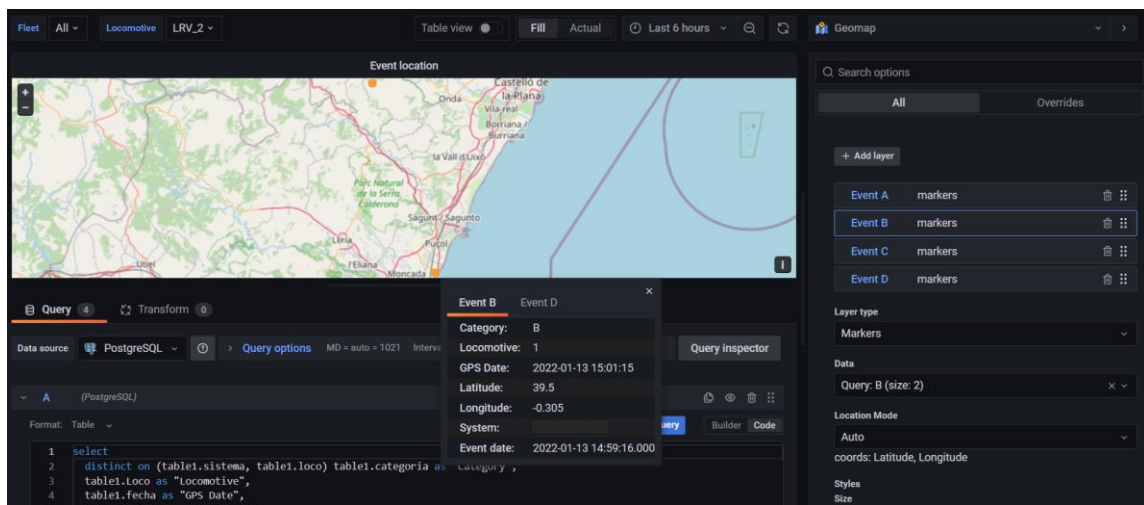


Figura 28: mapa de localización de eventos

### 5.5.2. Locomotoras

Ahora es el turno de la sección correspondiente con los paneles de los vehículos en particular. Para ello, seleccionaremos un vehículo de entre los disponibles en la variable. A continuación se detallarán uno a uno todos los paneles implementados y sus configuraciones.

En la zona superior, como vemos en la figura 29 y de forma similar a la distribución presentada en el apartado de flota, encontramos paneles y tablas que muestran información como las etiquetas del nombre de los vehículos, su flota algunos valores de variables, la distancia total recorrida y la energía total consumida. Además, también hay una tabla que muestra otros valores de forma compacta. En cuanto a las consultas SQL, son las más simples y constan de una cláusula 'SELECT' únicamente restringida mediante el 'WHERE' para que los datos pertenezcan al vehículo correspondiente.

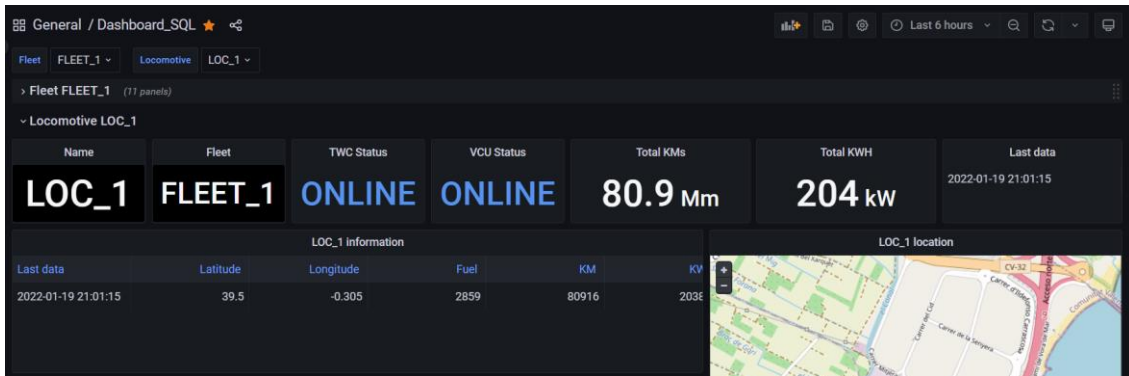


Figura 29: paneles generales

#### Locomotive location

Este mapa muestra la localización del vehículo seleccionado a tiempo real como vemos en la figura 30. Para ello, al igual que en el panel de localización de la flota, obtiene la latitud y longitud de la tabla 'REALTIME' únicamente de la locomotora adecuada. Además, para cambiar el icono que viene por defecto, en el apartado *map layers*, se configuran los parámetros como se muestran en la imagen. Cabe destacar que el icono a introducir ha de estar en formato 'SVG', además para mejorar la apariencia, se aumenta la opacidad a 1. La locomotora utilizada como punto de referencia ha sido proporcionada por la empresa. Por último, el mapa seleccionado como base es el *Open Street Map*.

Consulta SQL:

```
1. SELECT latitude_f AS "Latitude",
longitude_f AS "Longitude"
2. FROM realtime
3. WHERE (SELECT LOCOMOTIVES.ID_I FROM LOCOMOTIVES
WHERE $loco_name = LOCOMOTIVES.NAME_C)=realtime.ID_I;
```



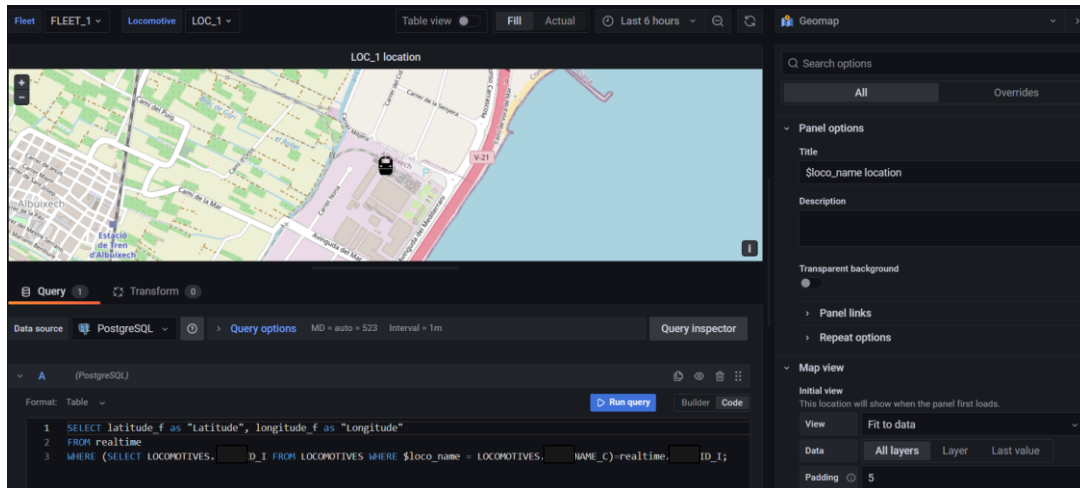


Figura 30: localización del vehículo

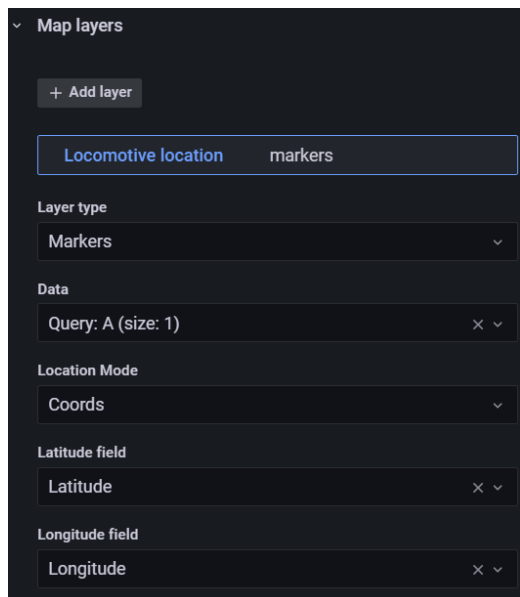


Figura 31: configuración de controles

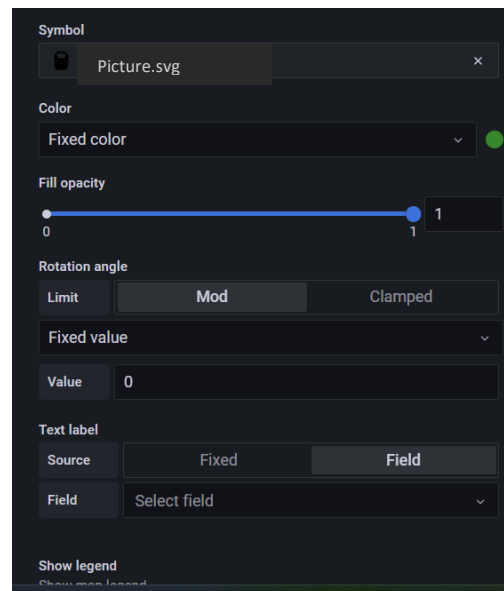


Figura 32: configuración de controles

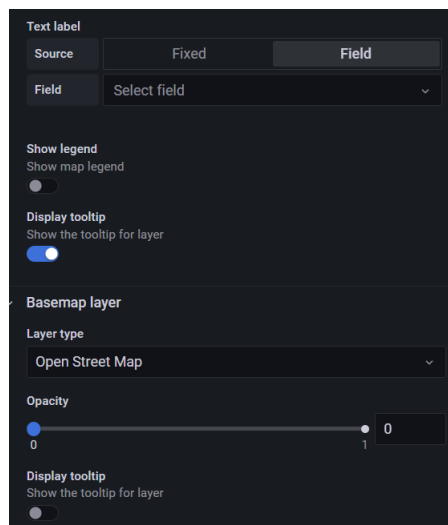


Figura 33: configuración de controles

## Fuel and Speed Gauge

Estos gráficos son similares, la figura 34 muestra la velocidad actual del vehículo y la figura 35 muestra el combustible restante del vehículo. En ambas consultas realizamos accesos a la tabla 'REALTIME' para recuperar la velocidad y el combustible respectivamente.

Respecto a la configuración de color, en el gráfico de la velocidad establecimos el blanco como color base y en el gráfico de combustible establecimos un *threshold* del 10% para que si se encuentra por debajo del ese porcentaje, se ponga de color rojo.

Consulta SQL para Speed:

```
1.     SELECT speed_f
2.     FROM realtime
3.     WHERE (SELECT LOCOMOTIVES.ID_I FROM LOCOMOTIVES
WHERE $loco_name = LOCOMOTIVES.NAME_C)=realtime.ID_I;
```

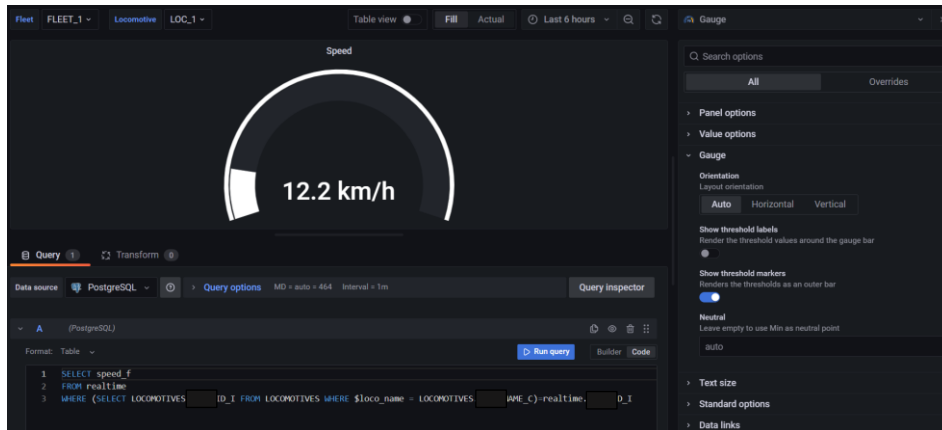


Figura 34: gráfico de velocidad

Consulta SQL para Fuel:

```
1.     SELECT fuel_f
2.     FROM realtime
3.     WHERE (SELECT LOCOMOTIVES.ID_I FROM LOCOMOTIVES
WHERE $loco_name = LOCOMOTIVES.NAME_C)=realtime.ID_I;
```

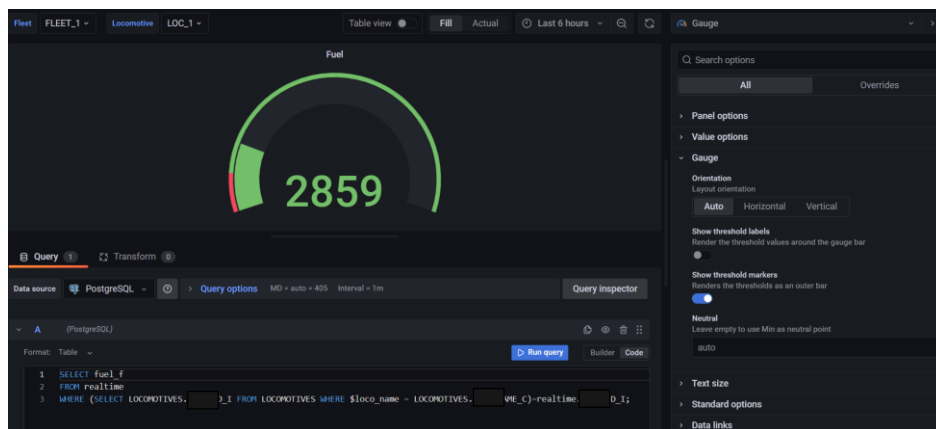


Figura 35: gráfico de combustible

## Totalizers barplot

En la figura 36 vemos como en este panel se quiere ver de forma destacada la distancia recorrida por la locomotora en contraposición a la distancia recorrida por el resto de la flota.

Para este gráfico hemos utilizado la función 'crosstab', que habremos activado en la página 34, de forma que obtenemos la fecha sin hora, un identificador que generemos y el valor máximo de la variable. Se escoge este valor ya que será el máximo diario, es decir, podemos ver la evolución y cómo aumentan los kilómetros día a día. Esta consulta se unirá a otra similar, pero que contendrá los datos de la flota sin la locomotora que hemos tenido en cuenta antes. Con los datos de ambas consultas, pasadas como un solo argumento a la función, no será necesario un segundo argumento que especifique las columnas a pivotar porque únicamente tratamos 3 variables: el tiempo, un identificador y el valor y para un caso con esta cantidad de campos no requiere más información. Para finalizar la consulta, especificamos el tipo de cada una de las variables que devolvemos.

Para configurar el color de las barras apiladas, hay que configurar ciertos campos en el apartado de *thresholds* como observamos en la figura 37.

Consulta SQL:

```
1.      SELECT *
2.      FROM
3.          crosstab(
4.              '
5.              ((with LOC as
6.                  (SELECT DATE(timestamp_d) as Date, 'Loco'
as ide, MAX(hrvv_value_f) as LOCO
7.                  FROM totalizers
8.                  WHERE hrvv_value_f>0 and (SELECT
LOCOMOTIVES.ID_I FROM LOCOMOTIVES WHERE
'${loco_name}'=LOCOMOTIVES.NAME_C AND LOCOMOTIVES.FLEET_C=
9.                  (SELECT fleet_c FROM locomotives WHERE name_c =
'${loco_name}'))=totalizers.ID_I and var_name_c =
'KM_COUNT')
10.              GROUP BY Date)
11.          select *
12.          from LOC
13.          union ALL
14.          select DATE(timestamp_d) as Date, 'Fleet' as
ide,max(hrvv_value_f) as FLEET
15.          from totalizers
16.          WHERE hrvv_value_f > 0 and totalizers.ID_I IN
(SELECT LOCOMOTIVES.ID_I FROM LOCOMOTIVES WHERE
'${loco_name}'!=LOCOMOTIVES.NAME_C AND
LOCOMOTIVES.FLEET_C=(SELECT fleet_c FROM locomotives WHERE
name_c = '${loco_name}') and var_name_c = 'KM_COUNT')
17.          GROUP BY Date
18.          order by date, ide))' AS rootTable2 (
19.              "Last Data" DATE,
20.              "Fleet" FLOAT,
                "Locomotive" FLOAT);
```



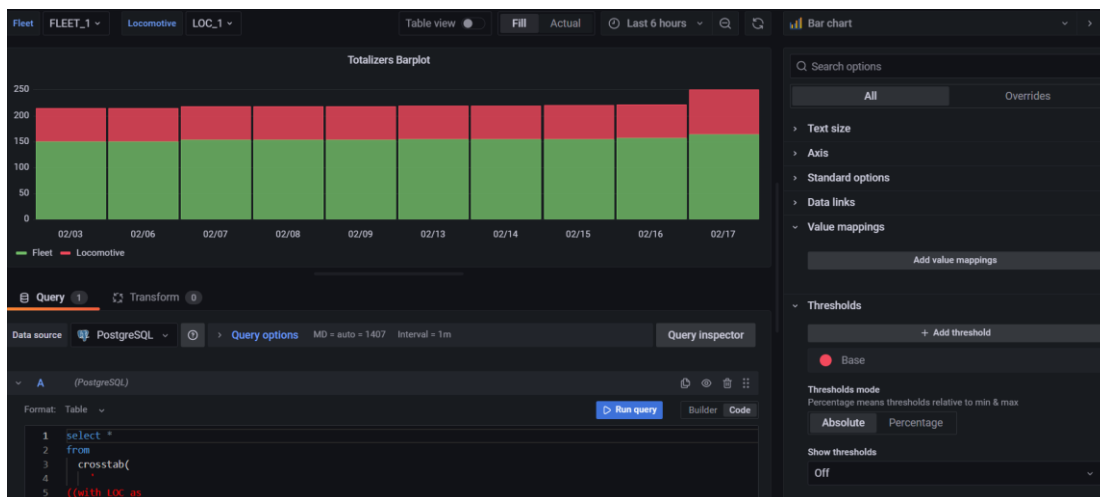


Figura 36: gráfico de distancia del vehículo vs flota

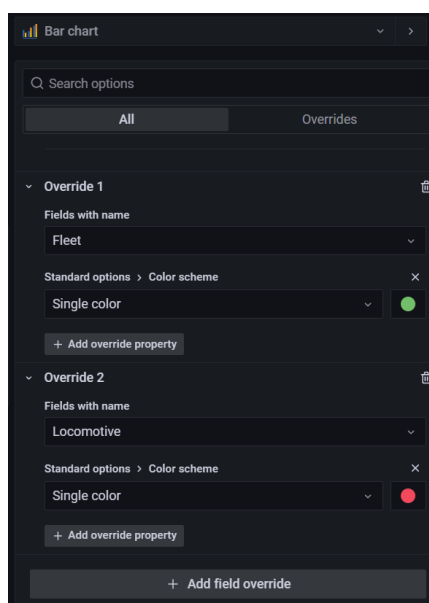


Figura 37: configuración de controles

### Locomotive event distribution

Para realizar este gráfico primero hay que instalar el *plug-in* correspondiente como se detalla en el anexo 10.4.4.

Esta visualización es análoga a la realizada para la flota, en los vértices se encontrarán los diferentes sistemas y el área interior se acercará más si hay mayor cantidad de eventos de este sistema, como vemos en la figura 38.

Tras elegir este modo de visualización, en la consulta correspondiente elegimos el formato de series temporales. Respecto a la consulta, hemos provocado que la serie temporal sea únicamente de un único momento, el más reciente, y realice un *count* para sumar los eventos de distinto tipo que hay según el sistema. Todos estos valores los obtenemos mediante subconsultas en la cláusula 'FROM'. Por último, le damos el alias de 'Quantity' a la consulta A.





Consulta SQL:

```
1.  SELECT
2.    $__time(tiempo.intervalo),
3.    contador.cont,
4.    contador.tipo
5.  FROM
6.    ( SELECT MAX(timestamp_d) AS intervalo
7.      FROM eventos
8.      WHERE
9.        eventos.ID_I IN (
10.         SELECT LÓCOMOTIVES.ID_I
11.         FROM LÓCOMOTIVES
12.         WHERE $loco_name = LÓCOMOTIVES.NAME_C)
13.    ) AS tiempo,
14.    ( SELECT COUNT(EVENTOS.CATEGORY_C) AS cont,
system_c AS tipo
15.      FROM eventos
16.      WHERE
17.        eventos.ID_I IN (
18.         SELECT LÓCOMOTIVES.ID_I
19.         FROM LÓCOMOTIVES
20.         WHERE $loco_name = LÓCOMOTIVES.NAME_C)
21.      GROUP BY system_c
22.    ) AS contador;
```

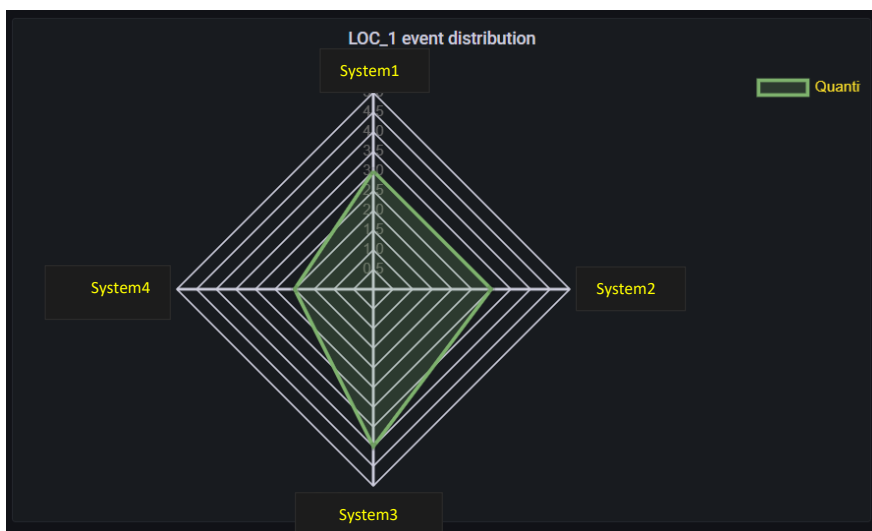


Figura 38: distribución de eventos del vehículo

## Events

En este panel se genera un gráfico de barras como vemos en la figura 39, que es similar al de la sección anterior, donde se clasificarán los diferentes eventos al mismo tiempo que se contarán. De esta forma cuantos más eventos de un tipo haya, mayor será la barra. Además, según el color de la barra se podrá saber de qué tipo es el evento.



Se trata de una visualización similar a la realizada para la flota, pero restringida únicamente a una locomotora. Para obtener los datos de este gráfico recurrimos a una consulta SQL que devuelve la categoría del evento y un recuento de cada categoría. Para dar color a cada una de las categorías, especificamos que si es A, tenga el color rojo; si es B, sea naranja; si es C, sea amarillo y en cualquier otro caso, será verde.

Consulta SQL:

```
1. SELECT COUNT (EVENTOS.CATEGORY_C) AS "Quantity",  
EVENTOS.CATEGORY_C AS "Category"  
2. FROM EVENTOS  
3. WHERE EVENTOS.ID_I IN (SELECT LOCOMOTIVES.ID_I F  
ROM LOCOMOTIVES WHERE $loco_name = LOCOMOTIVES.NAME_C)  
4. GROUP BY eventos.CATEGORY_C;
```

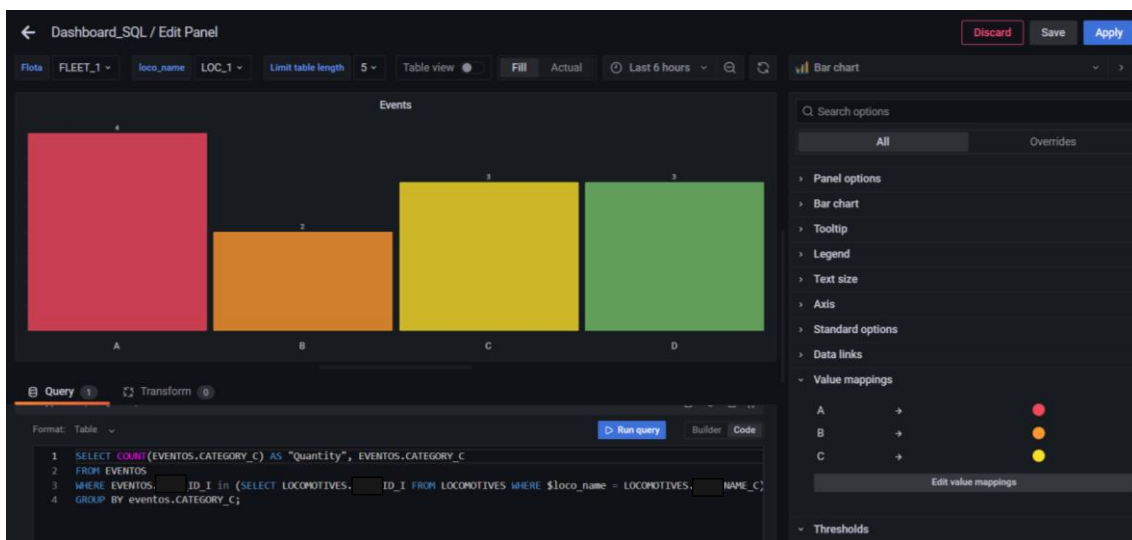


Figura 39: gráfico de eventos del vehículo

### Event location

Es el último gráfico de esta sección, es un mapa en el que se muestra la ubicación del último evento ocurrido de un sistema del vehículo. Es muy similar al mapa de localización de eventos de la flota, pero únicamente para un vehículo como se puede ver en la figura 40.

Para realizar este mapa necesitamos las coordenadas del vehículo, la fecha de los eventos y los datos relacionados con los eventos que queramos que se muestren al pasar el cursor sobre el punto. Además, se mostrarán los eventos que sean más recientes, es decir, de un mismo tipo de evento únicamente se mostrará la última incidencia de este tipo con las coordenadas más cercanas a la fecha de este evento. Para obtener la información necesaria, empezamos realizando subconsultas a partir de las cuales vamos extrayendo información sin perder el identificador de los datos, para así después poder combinar y comparar la información de las tablas implicadas.

Para poder configurar los diferentes tipos de eventos con sus respectivos colores, también hemos creado varias capas y a cada una de ellas se le ha asignado una de las consultas según si extrae eventos de categoría A, B... Así cada categoría tendrá su color correspondiente.



En este mapa también utilizaremos la función ‘DATEDIFF’ mencionada anteriormente en la página 39.

Consulta SQL:

```
1.  SELECT
2.      DISTINCT ON (table1.sistema, table1.loco)
3.      table1.categoria AS "Category",
4.      table1.Loco AS "Locomotive",
5.      table1.fecha AS "GPS Date",
6.      table1.Latitude AS "Latitude",
7.      table1.Longitude AS "Longitude",
8.      table1.sistema AS "System",
9.      table1.tiempo AS "Event date"
10. FROM
11.     (
12.         SELECT
13.             dataGPS.id_i AS Loco,
14.             DATAGPS.timestamp_d AS fecha,
15.             dataGPS.latitude_f AS Latitude,
16.             dataGPS.longitude_f AS Longitude,
17.             tiempo1.categoria,
18.             tiempo1.sistema,
19.             tiempo1.locomotiveIDEvent,
20.             tiempo1.tiempo,
21.             MIN(
22.                 ABS (
23.                     DATEDIFF (
24.                         'second',
25.                         DATAGPS.timestamp_d :: TIMESTAMP,
26.                         tiempo1.tiempo :: TIMESTAMP
27.                     )
28.                 )
29.             ) AS DIFMIN
30.         FROM
31.             datagps,
32.             (
33.                 SELECT
34.                     eventos.timestamp_d AS tiempo,
35.                     eventos.system_c AS sistema,
36.                     eventos.category_c AS categoria,
37.                     EVENTOS.id_i AS locomotiveIDEvent
38.                 FROM
39.                     eventos
40.                 WHERE
41.                     eventos.id_i = (
42.                         SELECT
43.                             locomotives.id_i
44.                         FROM
45.                             locomotives
46.                         WHERE
47.                             locomotives.name_c = $loco_name
48.                     )
49.                 GROUP BY
```



```

50.         categoria,
51.         sistema,
52.         locomotiveIDEvent,
53.         tiempo
54.     ) AS tiempol
55. WHERE
56.     datagps.id_i = (
57.         SELECT
58.             locomotives.id_i
59.         FROM
60.             locomotives
61.         WHERE
62.             locomotives.name_c = $loco_name
63.     )
64.     AND dataGPS.id_i = tiempol.locomotiveIDEve
nt
65.     AND DATAGPS.timestamp_d BETWEEN tiempol.ti
empo - INTERVAL '5 MINUTE'
66.     AND tiempol.tiempo + INTERVAL '5 MINUTE'
67. GROUP BY
68.     dataGPS.id_i,
69.     DATAGPS.timestamp_d,
70.     dataGPS.latitude_f,
71.     dataGPS.longitude_f,
72.     tiempol.categoria,
73.     tiempol.sistema,
74.     tiempol.locomotiveIDEvent,
75.     tiempol.tiempo
76. ORDER BY
77.     difmin ASC,
78.     dataGPS.id_i,
79.     dataGPS.latitude_f,
80.     dataGPS.longitude_f,
81.     tiempol.categoria,
82.     tiempol.sistema,
83.     tiempol.locomotiveIDEvent
84. ) AS table1
85. WHERE table1.categoria = 'A'
86. ORDER BY
87.     table1.sistema,
88.     table1.loco,
89.     difmin asc;

```



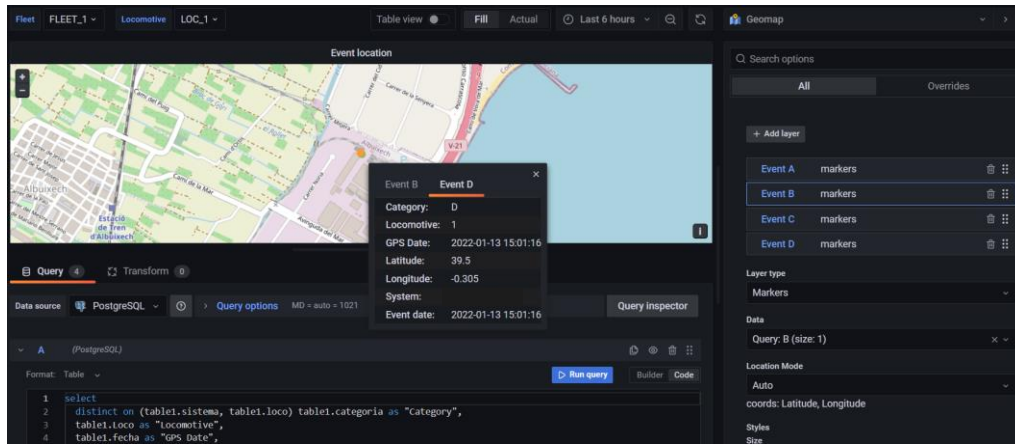


Figura 40: localización de eventos del vehículo

## 5.6. Resultados

Finalmente, tras aunar los diferentes paneles en una única visualización obtenemos el resultado que queríamos obtener desde un principio, un dashboard en el que se puede visualizar la información necesaria para ayudar en la toma de decisiones. Así primero tenemos información general de la sección y posteriormente tenemos información detallada de los diferentes eventos que han sucedido.

Aquí mostramos las vistas tanto de la sección de la flota en la figura 41 como de la sección de los vehículos ferroviarios en la figura 42. En la primera vista observamos que disponemos de las variables mencionadas anteriormente situadas en la zona superior.

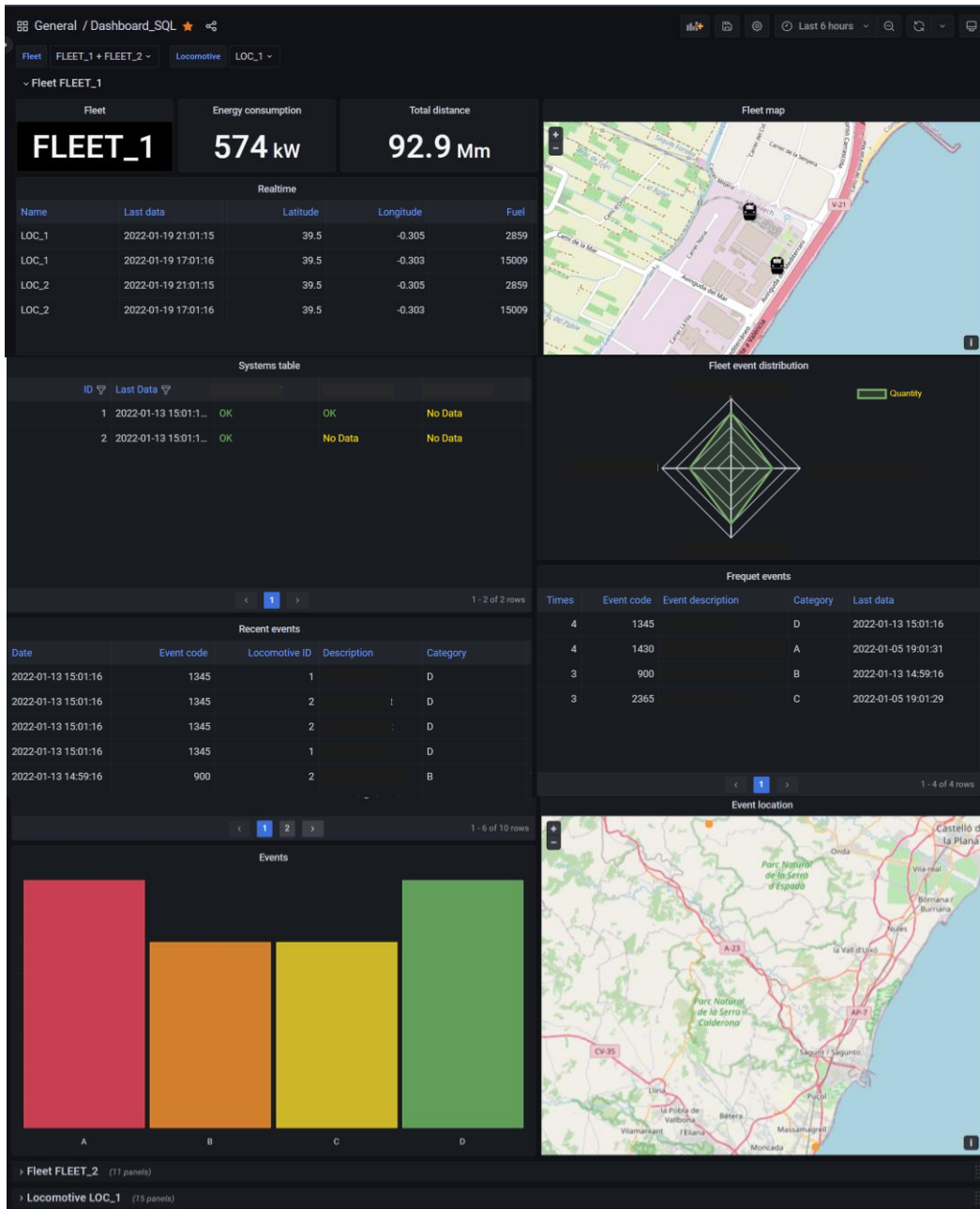


Figura 41: sección de las flotas

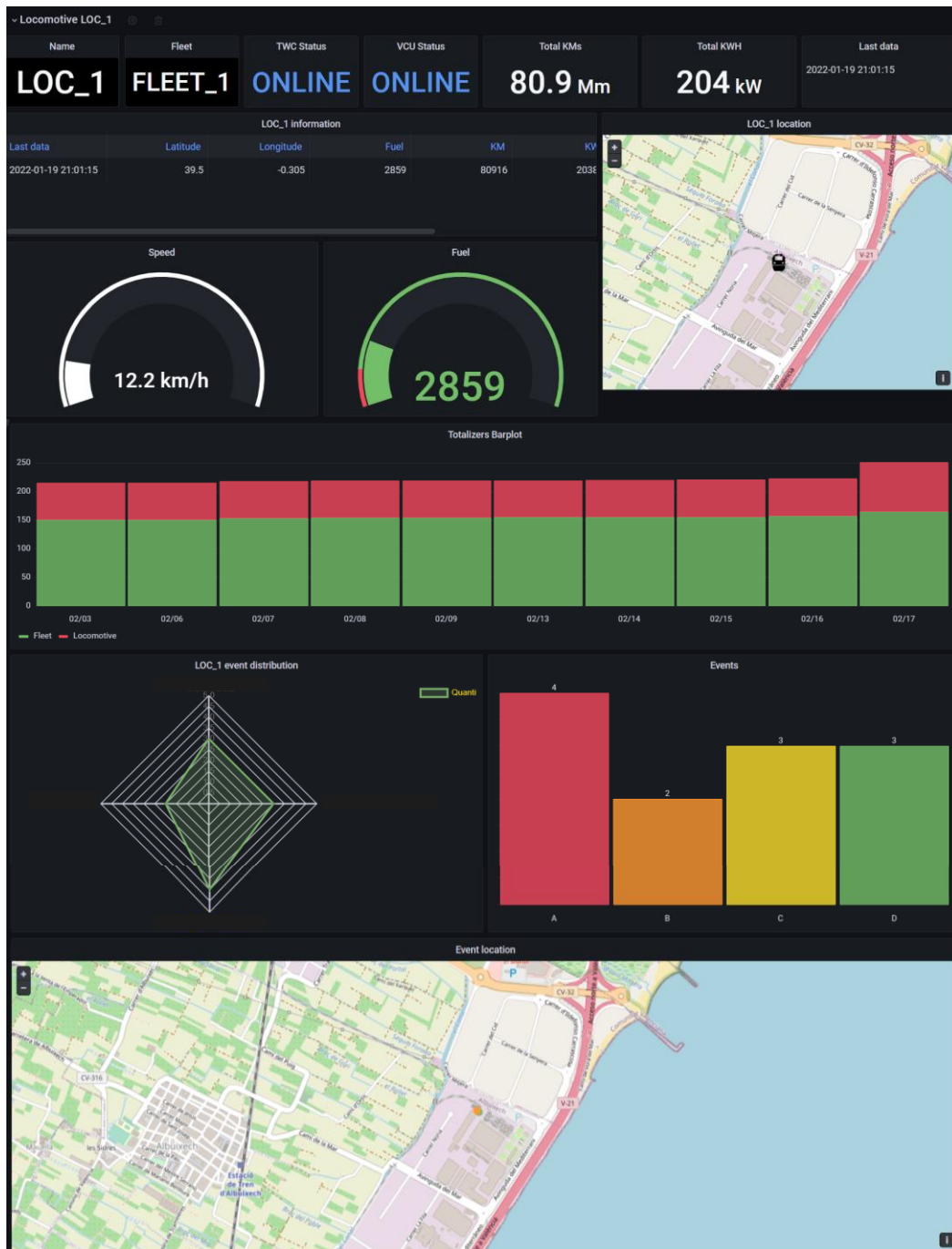


Figura 42: sección de los vehículos

## 6. Conclusiones

---

La visualización de datos se ha convertido en un aspecto fundamental en el mundo actual, permitiendo a las organizaciones y empresas aprovechar el poder de la información para tomar decisiones más informadas y eficientes. En este proyecto, nos propusimos cumplir una serie de objetivos clave relacionados con la revisión de herramientas de visualización, el análisis y definición de paneles, la preparación de herramientas de transmisión de información y el desarrollo de paneles adecuados para un dashboard eficaz.

En esta sección, veremos en detalle cómo cada uno de estos objetivos se ha cumplido. Desde la selección de las tecnologías hasta la creación de paneles que cumplen con los requisitos definidos, hemos trabajado para garantizar que el dashboard cumpla con las expectativas y necesidades que se requerían.

En primer lugar, se llevó a cabo una exhaustiva revisión de las herramientas de visualización disponibles y se seleccionaron las tecnologías más adecuadas para nuestro contexto. Esta etapa nos permitió contar con una sólida base de conocimiento y así asegurarnos de utilizar la herramienta que mejor se adaptaba para el correcto desarrollo del dashboard.

Además, se realizó un análisis detallado y una definición precisa de los paneles a implementar en la visualización. Mediante este proceso, pudimos identificar las necesidades y requisitos específicos que debían cumplir los paneles para ofrecer una experiencia visual efectiva y útil para las necesidades de la empresa.

También se prepararon todas las herramientas necesarias para transmitir la información de manera eficiente y coherente a través de la visualización. Se realizaron las instalaciones y configuraciones pertinentes para recopilar y transmitir los datos de forma clara y comprensible para Grafana, que se encargará de generar las visualizaciones con las configuraciones establecidas.

Finalmente, se desarrollaron los paneles de acuerdo con las especificaciones definidas, asegurando que cada uno cumpliera con su función dentro del dashboard. Se ha logrado crear una interfaz intuitiva y sencilla de usar de usar, que proporciona la información adecuada de una manera visual. De esta forma se ha obtenido una gran cantidad de conocimientos sobre esta nueva herramienta, Grafana, y se han puesto en práctica junto con conocimientos previos en lenguaje SQL. Cabe destacar que PostgreSQL utiliza el lenguaje SQL pero con ciertos matices diferentes al lenguaje visto durante la carrera puesto que no cuenta con algunas funciones, como DATEDIFF, que se obtuvo de la documentación revisada. También tuvimos que realizar activaciones de módulos que contenían funciones que se necesitaban, como es el caso de 'crosstab', para poder pivotar la tabla de datos.

El panel de visualización implementado tras completar los diferentes objetivos específicos ofrece cierta interactividad, permitiendo a los usuarios consultar las medidas y valores que necesiten en cada momento. Esto ofrece a la empresa la capacidad de acceder a la información relevante de manera ágil y eficiente, facilitando la toma de decisiones.

Mediante este TFG se sientan las bases de la creación de una nueva versión de una parte de la aplicación utilizada para la monitorización de los vehículos ferroviarios de la compañía.

Además, como se detallará en el apartado de posibles trabajos a futuro, el desarrollo de esta aplicación empieza con el trabajo detallado en este proyecto pero se continuará con la creación de las demás secciones contenidas en la aplicación.





En relación con los estudios cursados, a lo largo de la realización de este proyecto se han aplicado conocimientos en realización e interpretación de gráficos; instalación, creación y gestión de bases de datos relacionales, además de trabajar con consultas a la base de datos generada; se han obtenido conocimientos en instalación, configuración y gestión de la herramienta de visualización web Grafana.

En conclusión, para el cumplimiento de los objetivos y la realización del TFG se han requerido de conocimientos obtenidos durante la carrera, además de nuevos conocimientos que se han obtenido relacionados con herramientas de visualización web en mayor o menor medida, dado que sobre aquellas herramientas que no se han utilizado, se tiene un conocimiento superficial de ellas. En cambio de Grafana se tiene un profundo conocimientos de su funcionamiento.



## 7. Trabajos futuros

---

Como línea de trabajo a seguir, este proyecto es únicamente una pequeña parte de la aplicación que, en un futuro, se pretende renovar. Actualmente se continúa trabajando con una base de datos de mayor tamaño en PostgreSQL y se quiere generar un nuevo dashboard que contenga visualizaciones en formato de series temporales, a diferencia del presentado en este proyecto, que se trata de un dashboard que contiene visualizaciones para datos que se actualizarán a tiempo real.

En definitiva, se pretende continuar trabajando en el desarrollo de visualizaciones actualizadas basadas en la aplicación que se está utilizando actualmente. Para ello se necesitarán las mismas herramientas utilizadas hasta el momento siguiendo una metodología similar.

Por otra parte, la versión de la aplicación realizada a lo largo de este TFG está diseñada para obtener los datos de la base de datos a tiempo real, pero se ha desarrollado a partir de una captura en el tiempo de una serie de datos pertenecientes a distintas tablas, por lo que habría que valorar su desempeño a la hora de conectar el dashboard a la base de datos real de la empresa. Si se diese el caso, habría que revisar las consultas y configuraciones implementadas para que extraigan la información adecuada de la base de datos para transmitirla al dashboard.

En conclusión, estos son algunos aspectos con los que se podría continuar el desarrollo dashboard actual y también algunas líneas de trabajo que se pueden seguir en un futuro.



## 8. Glosario

---

**Bróker:** se refiere a una entidad o plataforma que se dedica a recopilar y distribuir alertas o notificaciones sobre eventos o situaciones específicas. Estas alertas suelen ser generadas por sistemas automatizados o por personas que monitorean diferentes fuentes de información en busca de eventos relevantes. Actúa como intermediario entre la fuente de la alerta y los destinatarios finales. Puede utilizar diferentes canales de comunicación, como mensajes de texto, correos electrónicos, aplicaciones móviles o sistemas de mensajería instantánea, para enviar las alertas a los destinatarios.

**Business Intelligence (BI):** se refiere al conjunto de herramientas, tecnologías y prácticas que permiten a las organizaciones recopilar, analizar y visualizar datos con el objetivo de obtener información significativa y tomar decisiones estratégicas informadas.

**Dashboard:** es una representación visual y resumida de datos, métricas e indicadores clave que proporciona una vista rápida y fácil de entender del rendimiento, la situación o el estado de aquello que se quiera visualizar. Se componen de gráficos, tablas y otros elementos visuales que presentan la información de manera clara y concisa.

**On-premise:** se refiere a una configuración o modelo de implementación de software o infraestructura tecnológica en el cual los recursos y sistemas están ubicados físicamente en las instalaciones de la organización o empresa. En un entorno de este tipo, las aplicaciones informáticas, servidores, bases de datos y otros componentes de infraestructura se instalan y se ejecutan en servidores y equipos locales dentro de la red de la organización.

**Threshold:** se refiere a un valor o condición específica que sirve como punto límite para tomar decisiones o realizar acciones. Se utiliza comúnmente en diversos contextos para establecer límites, referencias o disparadores para eventos o procesos específicos. En general, representa un punto en el cual se desencadena una acción, respuesta o resultado particular. Puede ser un valor numérico, un porcentaje, un rango o una condición específica que determina si un evento o condición determinada se considera significativo o notable.



## 9. Referencias

---

1. **Stadler Rail.** Sobre nosotros-Stadler. [En línea] <https://www.stadlerrail.com/es/sobre-nosotros/centros/stadler-valencia-su/179/>.
2. **Peralta, Luis Alberto.** Cinco Días. [En línea] 18 de Enero de 2022. [https://cincodias.elpais.com/cincodias/2022/01/17/companias/1642432445\\_092423.html](https://cincodias.elpais.com/cincodias/2022/01/17/companias/1642432445_092423.html).
3. **Valero, Dani.** Valencia Plaza. [En línea] 28 de Febrero de 2019. <https://valenciaplaza.com/stadler-confirma-la-construccion-en-valencia-de-36-trenes-para-gales-y-deja-otros-35-por-asignar>.
4. **Grafana.** Grafana corporation. [En línea] <https://grafana.com/>.
5. —. Alerting. *Grafana alerting*. [En línea] <https://grafana.com/docs/grafana/latest/alerting/>.
6. —. Kafka integration. *Grafana Kafka integration*. [En línea] <https://grafana.com/docs/grafana-cloud/data-configuration/integrations/integration-reference/integration-kafka/>.
7. **Bartolomeo, Joey.** Grafana dashboards in 2022: Memorable use cases of the year. *Grafana*. [En línea] 30 de Diciembre de 2022. <https://grafana.com/blog/2022/12/30/grafana-dashboards-in-2022/>.
8. **Grafana.** Pricing. *Grafana pricing*. [En línea] <https://grafana.com/pricing/>.
9. **Chronograf.** Create Chronograf alert rules | Chronograf 1.9 Documentation. *InfluxData Documentation*. [En línea] <https://docs.influxdata.com/chronograf/v1.10/guides/create-alert-rules/>.
10. —. Pricing. *Chronograf*. [En línea] <https://www.influxdata.com/influxdb-pricing/>.
11. **Andrulyte, Ieva Marija.** Telecommunications Analytics – Dashboard. *Tableau*. [En línea] <https://www.tableau.com/data-insights/dashboard-showcase/telecommunications-analytics>.
12. **Sutton, Will.** Explore nationwide US census engagement – Dashboard. *Tableau*. [En línea] <https://www.tableau.com/data-insights/dashboard-showcase/explore-nationwide-us-census-engagement>.
13. **Tableau.** Pricing. *Tableau*. [En línea] <https://www.tableau.com/es-es/pricing/teams-orgs>.
14. **Ansari, Tahreem.** Sales Analysis Dashboard - Microsoft Power BI Community. *Microsoft Power BI Community*. [En línea] 23 de Diciembre de 2020. <https://community.powerbi.com/t5/Data-Stories-Gallery/Sales-Analysis-Dashboard/m-p/1567208>.
15. **Ali, Mohammed.** Marketing Campaign Results Dashboard - Microsoft Power BI Community. *Microsoft Power BI Community*. [En línea] 21 de Febrero de 2021. <https://community.powerbi.com/t5/Data-Stories-Gallery/Marketing-Campaign-Results-Dashboard/m-p/1679227>.



16. **Power BI.** Pricing. *Power BI*. [En línea] <https://powerbi.microsoft.com/es-es/pricing/>.
17. **Kibana.** Alerting | Kibana Guide [8.5]. *Elastic*. [En línea] <https://www.elastic.co/guide/en/kibana/current/alerting-getting-started.html>.
18. **Bennett, Eleanor.** The Top 24 Kibana Dashboards & Visualisations. *Logit.io*. [En línea] 13 de Julio de 2021. <https://logit.io/blog/post/the-top-kibana-dashboards-and-visualisations/>.
19. **Kibana.** Subscriptions. *Kibana*. [En línea] <https://www.elastic.co/es/subscriptions>.
20. **Qlik.** Pricing. *Qlik*. [En línea] <https://www.qlik.com/es-es/pricing>.
21. —. Best Dashboard Examples: Over 100 by Industry & Role. *Qlik*. [En línea] <https://www.qlik.com/us/dashboard-examples>.
22. **SQLines.** PostgreSQL - DATEDIFF - Datetime Difference. *SQLines*. [En línea] 1 de Enero de 2012. <https://www.sqlines.com/postgresql/how-to/datediff>.
23. **Naciones Unidas.** Objetivos de Desarrollo Sostenible. [En línea] <https://www.un.org/sustainabledevelopment/es/objetivos-de-desarrollo-sostenible/>.



## 10. Anexos

### 10.1. Objetivos de Desarrollo Sostenible (ODS)

Además, este proyecto ayuda tanto de forma directa como indirecta a cumplir con los Objetivos de Desarrollo Sostenible (ODS) recogidos por las Naciones Unidas [23]. Los ODS fueron adoptados el 25 de septiembre de 2015 por una serie de mandatarios mundiales “como un llamamiento universal para poner fin a la pobreza, proteger el planeta y garantizar que para el 2030 todas las personas disfruten de paz y prosperidad”. Además están vinculados entre ellos, puesto que un avance en alguno de los objetivos supondrá un posible avance en otro. El grado de relación del trabajo con los objetivos se muestra en la tabla.

Objetivos de Desarrollo Sostenibles	Alto	Medio	Bajo	No Procede
1- Fin de la pobreza.				X
2- Hambre cero.				X
3- Salud y bienestar.				X
4- Educación de calidad.				X
5- Igualdad de género.				X
6- Agua limpia y saneamiento.				X
7- Energía asequible y no contaminante.				X
8- Trabajo decente y crecimiento económico.				X
9- Industria, innovación e infraestructuras.	X			
10- Reducción de las desigualdades.				X
11- Ciudades y comunidades sostenibles.		X		
12- Producción y consumo responsables.				X
13- Acción por el clima.		X		
14- Vida submarina.				X
15- Vida de ecosistemas terrestres.				X
16- Paz, justicia e instituciones sólidas.				X
17- Alianzas para lograr objetivos.				X

Tabla 2: Objetivos de Desarrollo Sostenible



De los objetivos de desarrollo sostenible mencionados anteriormente, el trabajo presentado está vinculado con tres de ellos.

- **9. Industria, innovación e infraestructura:** con este proyecto se pretende realizar un avance tecnológico respecto a la aplicación anterior. Puesto que los avances tecnológicos son esenciales para obtener soluciones, con esta nueva visualización se podrá tener una mejor perspectiva del impacto ambiental que tienen los vehículos ferroviarios en funcionamiento para así poder mejorar la eficiencia energética.
- **11. Ciudades y comunidades sostenibles:** fuertemente vinculado con el anterior objetivo. Para transformar las ciudades es necesario apostar por el transporte público. Este transporte público podrá ser mejor monitoreado gracias a esta herramienta y así influirá en ofrecer mejor calidad de transporte reduciendo las emisiones en la ciudad.
- **13. Acción por el clima:** altamente relacionado con ambos objetivos presentados previamente. Ambos promueven una reducción de la emisión de gases, ya sea de forma directa mediante mejoras de eficiencia en el vehículo, como de forma indirecta, haciendo que este vehículo sea más utilizado por la población y, por tanto, se reducen las emisiones provocadas por los desplazamientos urbanos e interurbanos.

## 10.2. Versiones de Grafana y PostgreSQL

Aquí detallamos las versiones que se instalaron y utilizaron tanto de la herramienta Grafana, como de PostgreSQL.

Grafana version: v9.3.2 (21c1d14e91)

PostgreSQL version: 15.1, compiled by Visual C++ build 1914, 64-bit

## 10.3. PostgreSQL

### 10.3.1. Instalación

1. Descargamos el instalador adecuado, durante el proyecto se ha utilizado la versión 15.1 para Windows, aunque actualmente únicamente se encuentra disponible la versión 15.3. Es necesario tener privilegios de administrador para realizar la instalación.

PostgreSQL Version	Linux x86-64	Linux x86-32	Mac OS X	Windows x86-64	Windows x86-32
15.3	<a href="https://www.postgresql.org">postgresql.org</a>	<a href="https://www.postgresql.org">postgresql.org</a>			Not supported
14.8	<a href="https://www.postgresql.org">postgresql.org</a>	<a href="https://www.postgresql.org">postgresql.org</a>			Not supported
13.11	<a href="https://www.postgresql.org">postgresql.org</a>	<a href="https://www.postgresql.org">postgresql.org</a>			Not supported
12.15	<a href="https://www.postgresql.org">postgresql.org</a>	<a href="https://www.postgresql.org">postgresql.org</a>			Not supported
11.20	<a href="https://www.postgresql.org">postgresql.org</a>	<a href="https://www.postgresql.org">postgresql.org</a>			Not supported
10.23*					
9.6.24*					
9.5.25*					
9.4.26*					

Figura 43: versiones disponibles para instalar



2. Se inicia el proceso abriendo el archivo descargado. Se abrirá el menú de instalación, hacemos clic en siguiente.



Figura 44: primer paso del instalador

3. Especificamos el directorio donde se instalará. Se puede utilizar uno cualquiera o el que viene especificado por defecto.

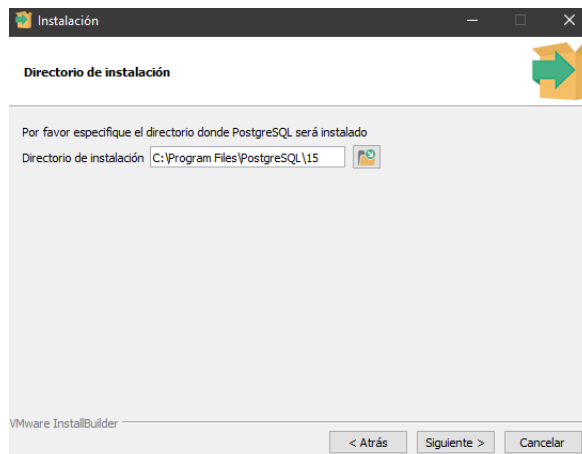


Figura 45: segundo paso del instalador

4. Seleccionamos los componentes, por defecto están todos seleccionados.

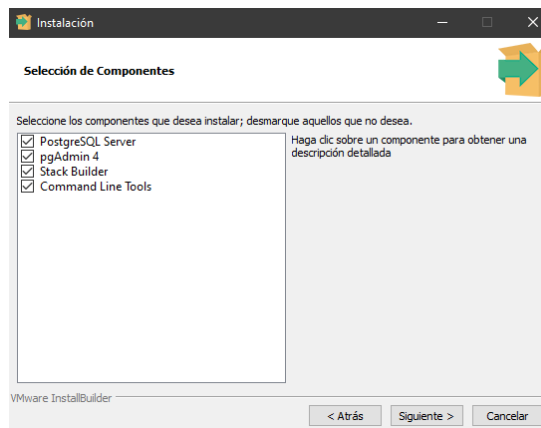


Figura 46: tercer paso del instalador



5. Seleccionamos el directorio donde se almacenarán los datos.



Figura 47: cuarto paso del instalador

6. Introducimos una contraseña para el superusuario 'postgres'.

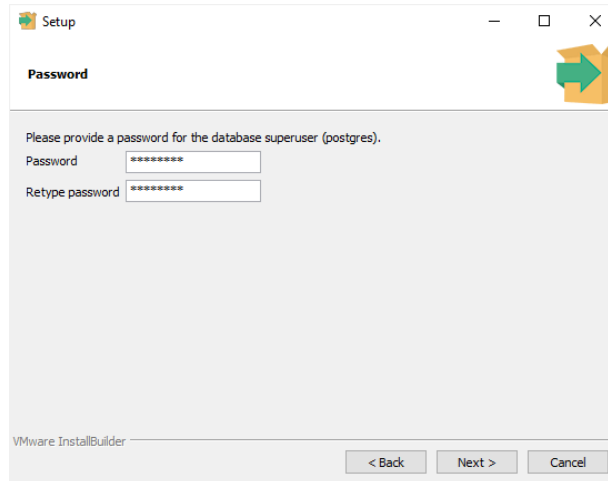


Figura 48: quinto paso del instalador

7. Introducimos el puerto que utilizará PostgreSQL. Por defecto viene seleccionado el 5432 pero si ya está en uso, introducimos otro diferente.

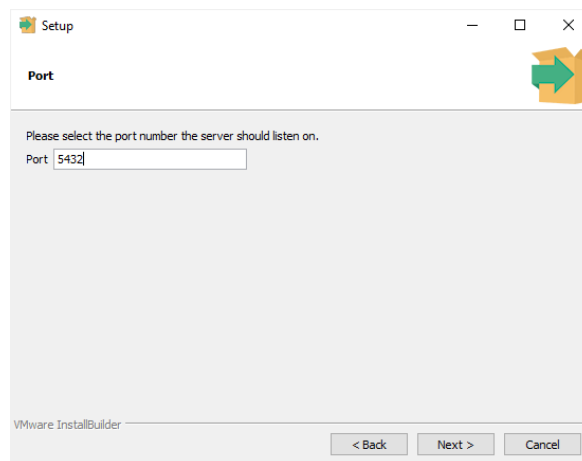


Figura 49: sexto paso del instalador



8. En el siguiente paso se selecciona la configuración regional que queremos. Por defecto se utilizará la misma que utilice nuestro equipo.

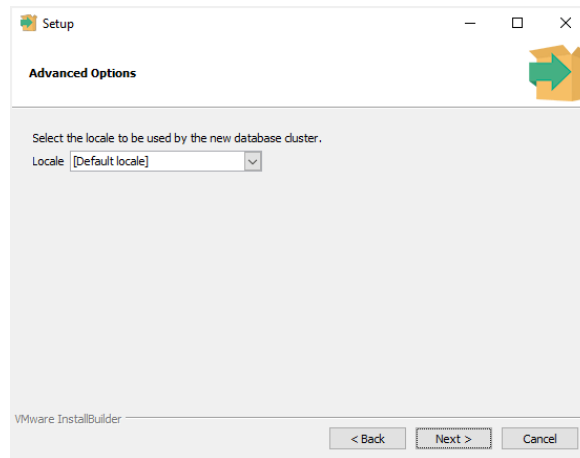


Figura 50: séptimo paso del instalador

9. A continuación tendremos una pantalla con un resumen de las diferentes configuraciones que hemos elegido. Debemos leerlo y verificar que toda la información es correcta. En caso contrario, volveremos al paso correspondiente y cambiaremos lo que sea necesario. Una vez verificado, estamos listos para comenzar la instalación. Tardará unos minutos en completarse.

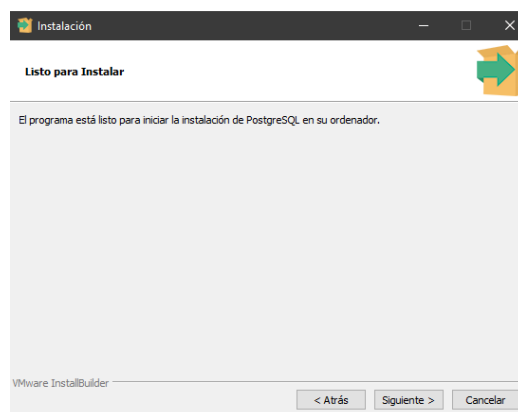


Figura 51: confirmar la instalación

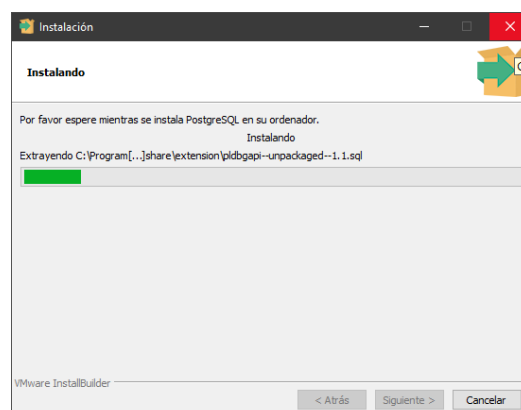


Figura 52: instalación en curso

10. Para finalizar la instalación, hacemos clic en terminar.



Figura 53: instalación finalizada

### 10.3.2. Verificación de la instalación

1. Busca 'SQL' en la barra de búsqueda del Sistema y abre la terminal 'SQL Shell'.

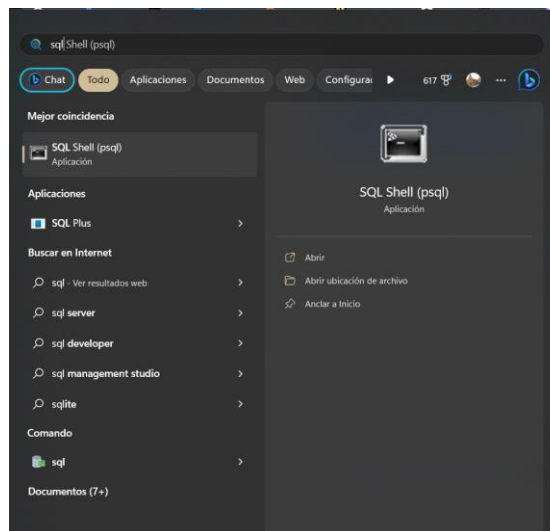


Figura 54: SQL Shell en el buscador

2. Presionamos la tecla intro e introducimos la contraseña que especificamos durante la configuración cuando la pida.

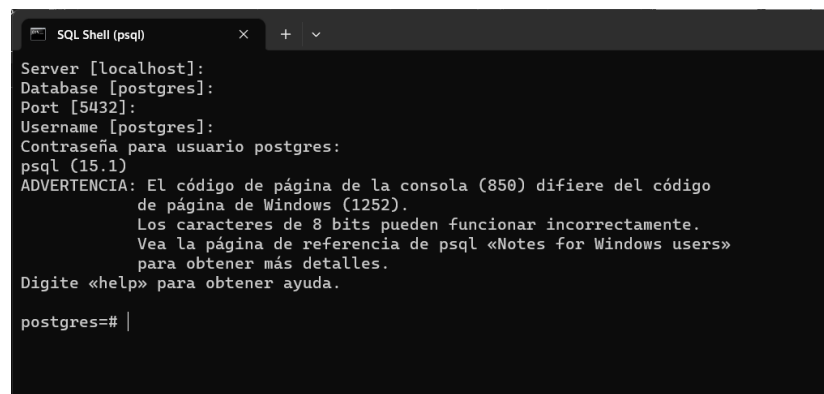


Figura 55: inicio del SQL Shell



### 10.3.3. Cargar base de datos PostgreSQL

Con el archivo que contiene la información de la base de datos utilizamos una serie de comandos para cargarlos. En primer lugar, desde la terminal del sistema abierta como administrador, nos situamos en la carpeta 'bin' de nuestra aplicación PostgreSQL.

```
$$ cd C:\Program Files\PostgreSQL\15\bin
```

A continuación, utilizaremos el comando 'psql' que se encuentra disponible en esta carpeta para cargar la información a la base de datos.

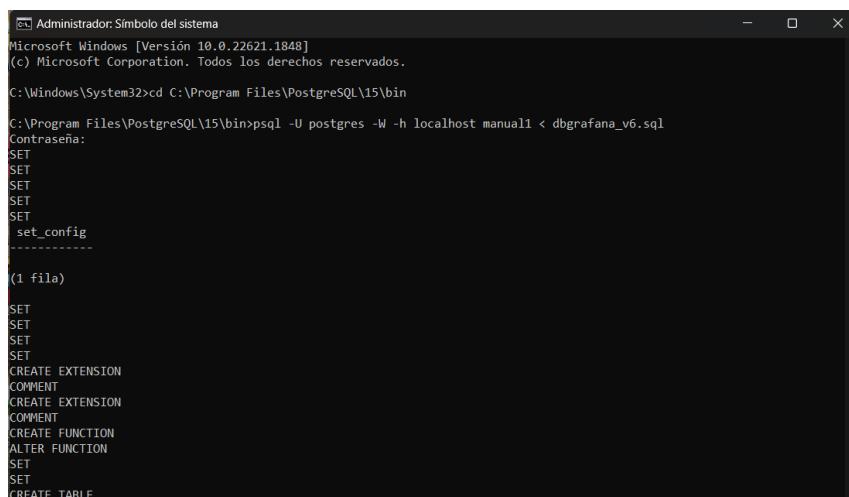
```
$$ psql -U postgres -W -h localhost nombre_base < base_a_cargar.sql
```

La definición de los diferentes parámetros es la siguiente.

Parámetro	Significado
-U	Se refiere al Usuario, en nuestro caso usamos el usuario: postgres
-W	Con este parámetro haremos que nos solicite la contraseña del usuario.
-h	Con este indicamos cuál es el servidor PostgreSQL al que nos conectaremos para importar nuestra información, si estamos en el mismo servidor podemos colocar localhost, si será un servidor remoto colocaremos la IP.
nombre_base	Este es el último parámetro en nuestra línea del comando y referencia el nombre de la base de datos a la que importaremos nuestros datos.
base_a_cargar.sql	Indicamos cual es el archivo que contiene los datos y queremos importar.

Tabla 3: definición de parámetros

Este es un ejemplo de cómo cargar los datos mediante la terminal:



```
Administrador: Símbolo del sistema
Microsoft Windows [Versión 10.0.22621.1848]
(c) Microsoft Corporation. Todos los derechos reservados.
C:\Windows\System32>cd C:\Program Files\PostgreSQL\15\bin
C:\Program Files\PostgreSQL\15\bin>psql -U postgres -W -h localhost manual1 < dbgrafana_v6.sql
Contraseña:
SET
SET
SET
SET
SET
set_config
-----
(1 fila)
SET
SET
SET
SET
CREATE EXTENSION
COMMENT
CREATE EXTENSION
COMMENT
CREATE FUNCTION
ALTER FUNCTION
SET
SET
CREATE TABLE
```

Figura 56: carga de datos

Además, para verificar que se ha cargado adecuadamente, podemos abrir el 'SQL Shell' y si utilizamos 'd' obtendremos las tablas e información sobre ellas.

```

SQL Shell (psql)
Server [localhost]:
Database [postgres]: manual1
Port [5432]:
Username [postgres]:
Contraseña para usuario postgres:
psql (15.1)
ADVERTENCIA: El código de página de la consola (850) difiere del código
de página de Windows (1252).
Los caracteres de 8 bits pueden funcionar incorrectamente.
Vea la página de referencia de psql «Notes for Windows users»
para obtener más detalles.
Dígame «help» para obtener ayuda.

manual1=# \d
Listado de relaciones
Esquema | Nombre | Tipo | Dueño
-----|-----|-----|-----
public | datagps | tabla | postgres
public | eventos | tabla | postgres
public | fuel | tabla | postgres
public | locomotives | tabla | postgres
public | realtime | tabla | postgres
public | totalizers | tabla | postgres
(6 filas)

```

Figura 57: verificar la carga

### 10.3.4. Descargar la base de datos PostgreSQL

Para descargar la base de datos PostgreSQL en Windows, se debe abrir la **terminal** como **administrador**. Una vez en la línea de comandos, accederemos a la carpeta bin contenida en la carpeta de la aplicación PostgreSQL. Una posible ruta puede ser:

```
$$ cd C:\Program Files\PostgreSQL\15\bin
```

A continuación, se utilizará el comando pg\_dump como se muestra:

```
$$ pg_dump -U user_name DB_name > dbgrafana_v*.sql
```

Donde v\* sirve para diferenciar la versión de la base de datos que se descarga. Puede especificarse como v1, v2, v3...

Nota: Se puede descargar tanto como archivo SQL como archivo PGSQL, simplemente hay que cambiar la denominación.

### 10.3.5. Configuración del entorno de ejecución

El entorno de ejecución que se ha utilizado es Visual Studio. Para poder utilizar la base de datos que está configurada en PostgreSQL en Visual Studio necesitamos instalar la extensión que se muestra en la imagen. Para ello, en el menú de extensiones buscamos 'PostgreSQL' y seleccionamos la primera opción y la instalamos. Puede que la aplicación necesite reiniciarse tras la instalación.

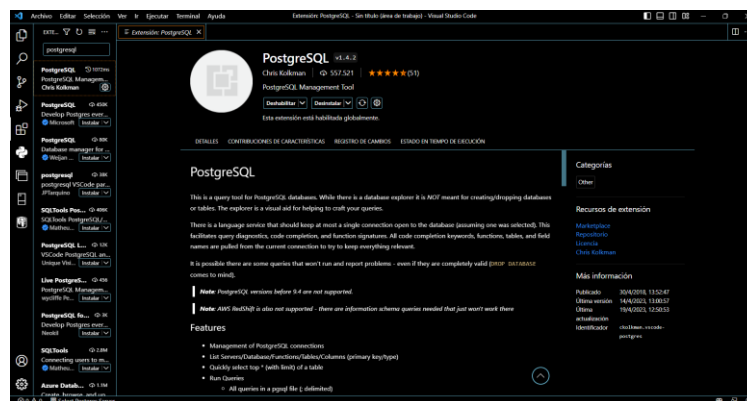


Figura 58: extensión de PostgreSQL



## Conexión a la base de datos

Para acceder a los datos que tenemos almacenados necesitamos crear una conexión. Para ello, seleccionaremos el icono de PostgreSQL que se habrá añadido en el menú de la izquierda y después clicaremos en el símbolo + para crear una conexión. En el primer paso pide el nombre del host de la base de datos, en nuestro caso es 'localhost'

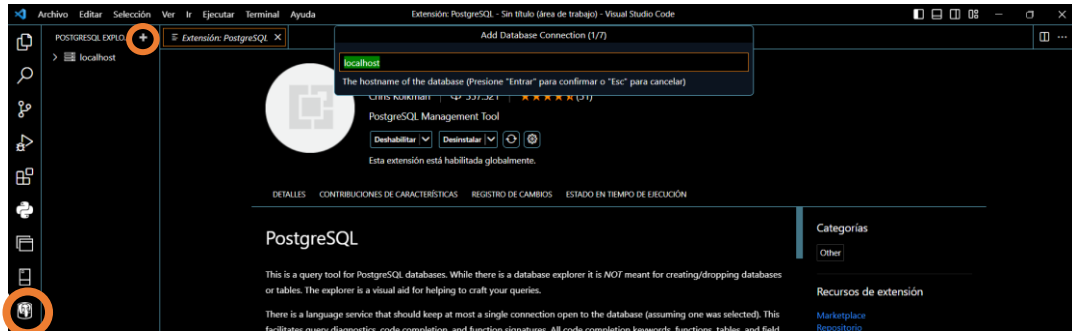


Figura 59: primer paso para crear la conexión

A continuación pide el nombre del usuario con el que acceder a los datos.

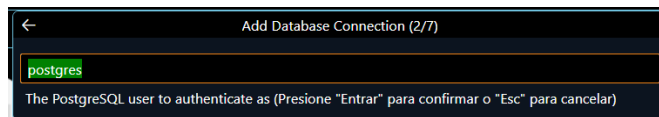


Figura 60: segundo paso para crear la conexión

Después necesita la contraseña

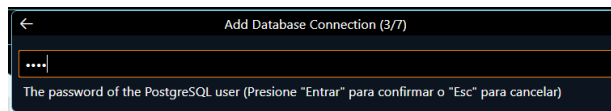


Figura 61: tercer paso para crear la conexión

También pide el puerto al que conectarse

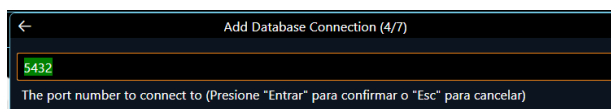


Figura 62: cuarto paso para crear la conexión

Y si se utiliza una conexión SSL. En este caso, la conexión es estándar.

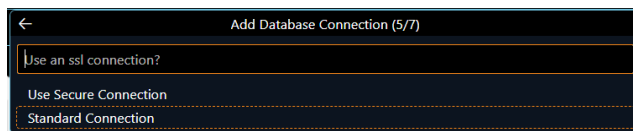


Figura 63: quinto paso para crear la conexión

Aquí podemos escoger la base de datos a la que accederá. Puede ser una de ellas o todas.

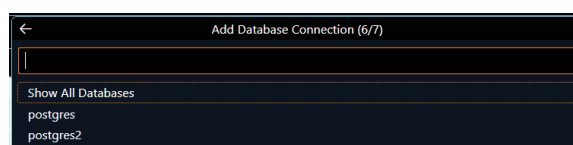


Figura 64: sexto paso para crear la conexión

Finalmente le damos un nombre a la conexión para diferenciarla.

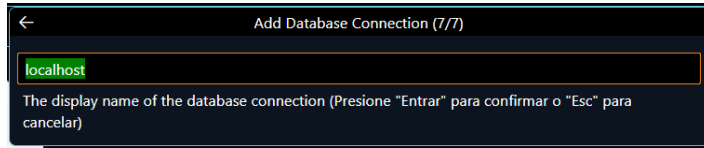


Figura 65: séptimo paso para crear la conexión

Para acceder a la conexión, tan solo tendremos que interactuar con el desplegable que se habrá generado en el menú de la izquierda como vemos en la figura 66. Además, si tenemos una consulta y queremos utilizar la conexión que hemos creado, tan solo tendremos que seleccionarla en el botón que hay en la parte inferior izquierda de la pantalla, como vemos en la figura 67.

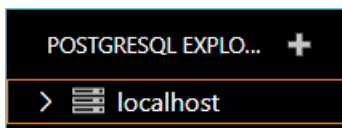


Figura 66: menú desplegable

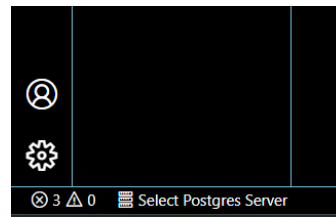


Figura 67: selector de la base de datos

## 10.4. Grafana

### 10.4.1. Instalación

1. La última versión de Grafana se selecciona por defecto, pero durante el proyecto hemos utilizado la versión 9.3.2
2. Seleccionamos una edición. La edición Enterprise es la utilizada en el proyecto.
3. Seleccionamos el archivo para el sistema adecuado. En el proyecto se usó Windows.
4. Descargamos el instalador del sitio oficial de Grafana.

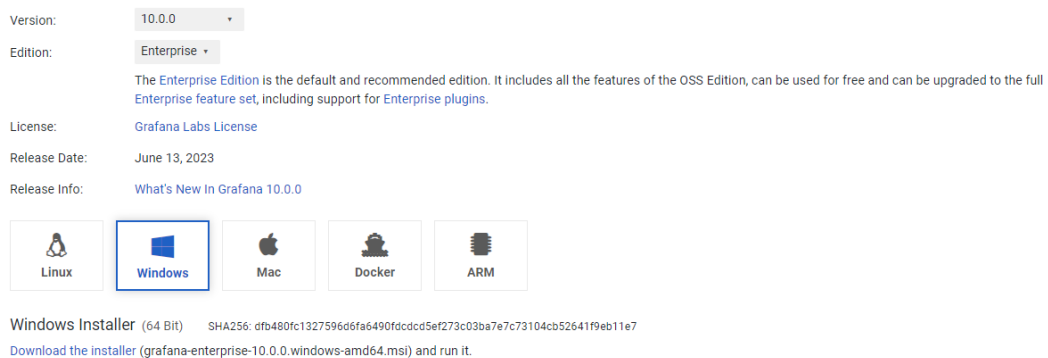


Figura 68: Grafana versión 10.0.0

Version: 9.3.2

Edition: Enterprise

The Enterprise Edition is the default and recommended edition. It includes all the features of the OSS Edition, can be used for free and can be upgraded to the full Enterprise feature set, including support for Enterprise plugins.

License: Grafana Labs License

Release Date: December 15, 2022

Release Info: What's New In Grafana 9.3.2

Linux Windows Mac Docker ARM

Windows Installer (64 Bit) SHA256: 761b29244556e861323c3a822773a80cfe80aae7009aafae6bf28283bf0e0df

Download the installer (grafana-enterprise-9.3.2.windows-amd64.msi) and run it.

Figura 69: Grafana versión 9.3.2

A continuación, abrimos el instalador descargado e iniciamos el proceso:

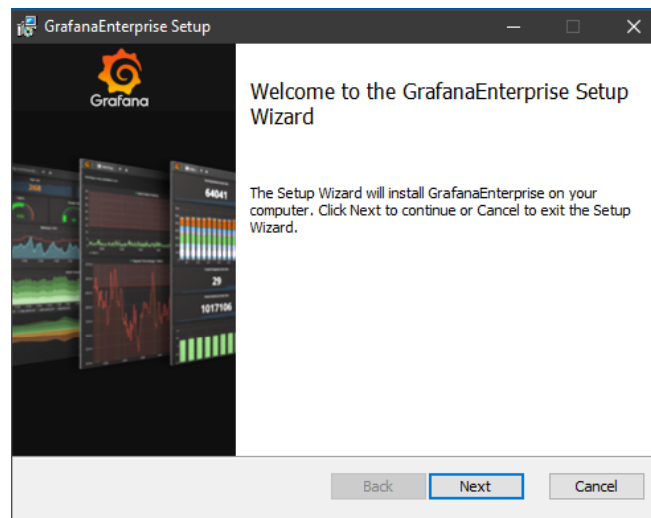


Figura 70: inicio del instalador

1. Primero debemos aceptar los términos y condiciones.

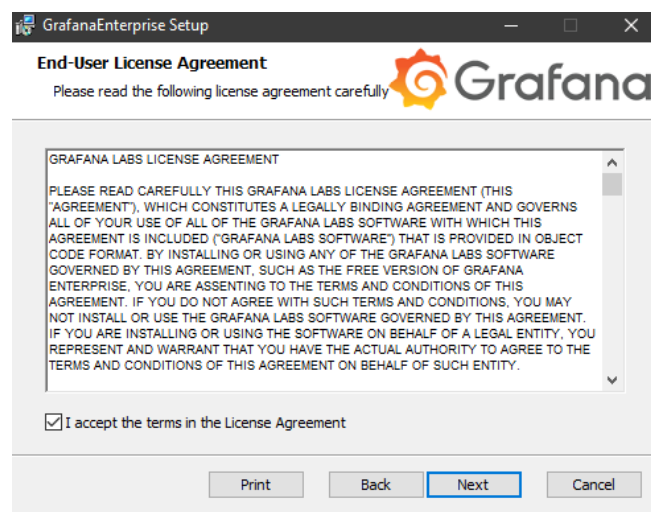


Figura 71: términos y condiciones de Grafana



- Después se muestran las características que se van a instalar y el espacio que ocupan.

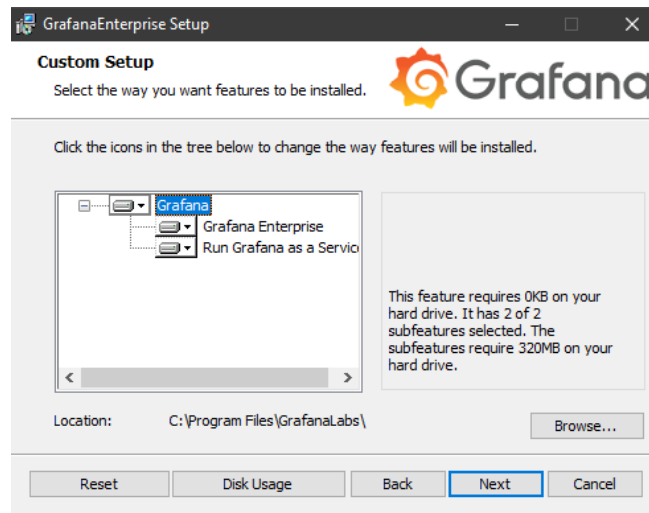


Figura 72: características a instalar

- Iniciamos la instalación cuando estemos listos.

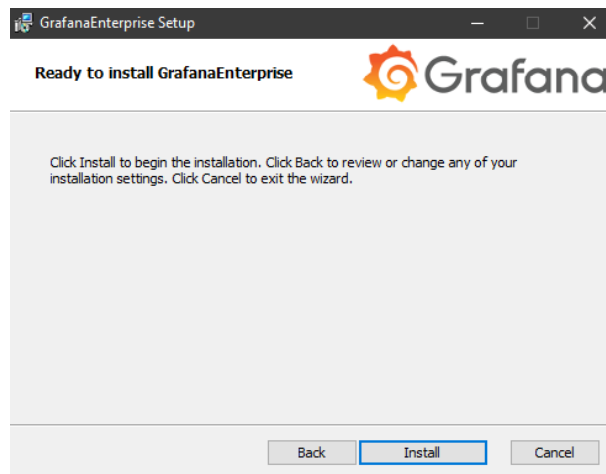


Figura 73: confirmar la instalación

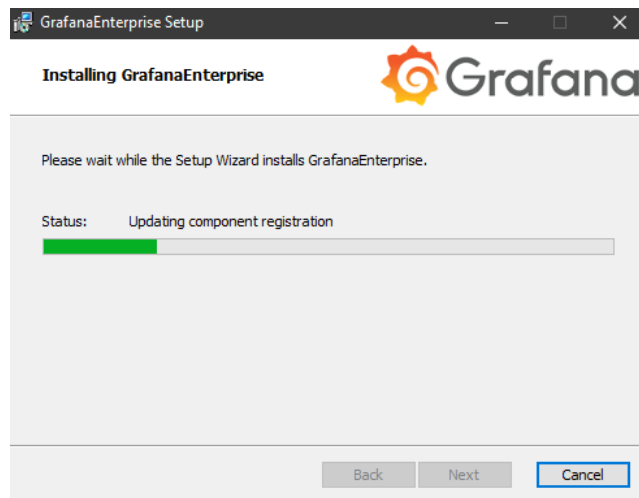


Figura 74: instalación en progreso

4. Si todo ha ido bien, hacemos clic en finalizar para cerrar el instalador.

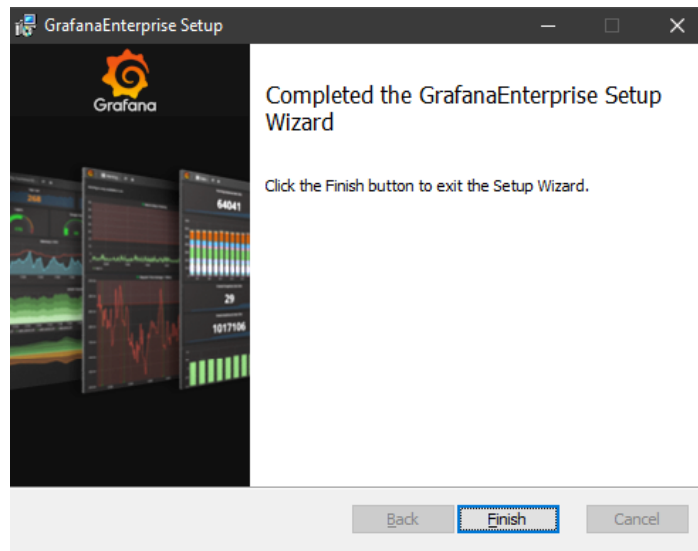


Figura 75: instalación finalizada

#### 10.4.2. Primer inicio de sesión

1. Abrimos el navegador y visitamos <http://localhost:3000/>. El puerto 3000 es el puerto HTTP por defecto en el que Grafana atiende si no se ha configurado uno distinto.

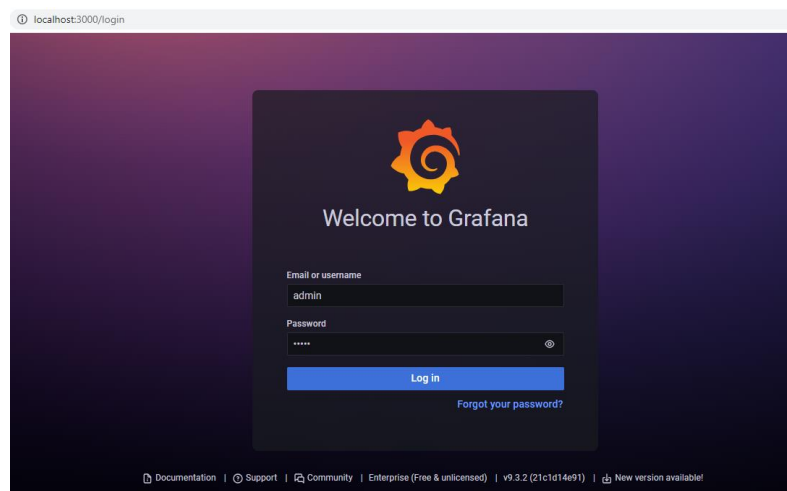


Figura 76: menú de inicio de sesión

2. En la página de inicio de sesión, escribimos 'admin' tanto como usuario como contraseña.
3. Automáticamente solicitará una nueva contraseña para que la modifiquemos.

#### 10.4.3. Creación de la conexión a la base de datos

Tras instalar Grafana y tener ya PostgreSQL, hay que crear la conexión entre ellos. Para ello, abrimos Grafana y verificamos que el plug-in de PostgreSQL se encuentre instalado. En caso

de que no esté instalado, se debe buscar en Configuración → Plugins → Select ALL filter → Search PostgreSQL → Install.

Una vez verificada su instalación, para añadir la fuente de datos hay que seguir los pasos:

Configuration → Data sources → Add data source → Rellenar con la información de inicio de PostgreSQL y desconecta el TSL/SSL Mode → Save and test

Si la información es correcta, marcará la conexión como OK y estará disponible para utilizar en los dashboard. No es necesario que esta base de datos tenga contenido para generar la conexión, este puede ser añadido después.

#### 10.4.4. Instalar Radar Graph

Antes de desplegar el dashboard, se va a utilizar un *polar plot* o *radar graph* que no se encuentra instalado por defecto. Para comenzar la instalación, es necesario abrir la terminal como administrador y acceder a la carpeta bin de la aplicación Grafana, ya que aquí se encuentra el programa necesario para la descarga. Una posible ruta es:

```
$$ cd C:\Program Files\GrafanaLabs\grafana\bin
```

Una vez se ha accedido, se puede utilizar el comando grafana-cli tanto para obtener la lista de los *plug-in* disponibles como para instalarlos. El comando se ha de utilizar de la siguiente forma:

1. Lista de *plug-in* disponibles: `$$ grafana-cli plugins list-remote`
2. Instalación del *radar graph*: `$$ grafana-cli plugins install snuids-radar-panel`

Tras la instalación será necesario reiniciar Grafana.

#### 10.4.5. Importación del dashboard

En cuanto a la forma de importar el dashboard, una vez instalado Grafana, hay que seguir la ruta: Import → Import json file. En el explorador seleccionamos el archivo en formato json que deseemos y, si se está cargando en una máquina que ya había contenido ese programa, deberemos cambiar el nombre del archivo y el identificador único (UID). Además, seleccionamos la base de datos correspondiente y si no muestra ningún error, importamos el dashboard.

Dashboards > Import > Import json file → Cambiar nombre, UID y seleccionar BBDD > Import

