Universidad Zaragoza
1542

Trabajo Fin de Grado

Capa semántica global para la gobernanza de datos previa a su análisis

Global semantic layer for data governance prior to analysis

Autor

Alejandro Adell Pina

Director

Alejandro Moreno Budría

Ponente

Francisco Javier Zarazaga Soria

Escuela de Ingeniería y Arquitectura

2020/2021

Capa semántica global para la gobernanza de datos previa a su análisis

RESUMEN

Una capa semántica es una capa de abstracción que proporciona una forma coherente de interpretar los datos. Asigna datos complejos a términos comerciales familiares para que los usuarios de toda la empresa puedan acceder a la misma fuente de verdad, con total confianza en su integridad. El propósito básico de la capa semántica es hacer que los datos sean más útiles para la empresa y simplificar las consultas para los usuarios [1].

En un primer paso, en este trabajo de fin de grado se ha llevado a cabo el análisis y la experimentación con diferentes alternativas de mercado que proporcionan una solución de capa semántica. Se identificaron una serie de limitaciones en su disponibilidad y características . A partir de aquí se ha diseñado e implementado una solución propia. Un prototipo de aplicación web, la cual, a través de una interfaz gráfica, permite conectarse a una fuente de datos SQL, modelar su capa semántica y posteriormente proporcionar el acceso a los datos modelados a diferentes herramientas de business intelligence o machine learning. Esta aplicación web facilita el modelado de una capa semántica global e intermedia de una fuente de datos, ofreciendo al usuario final una única versión consistente de los datos, sin bloqueos de proveedor que no permiten el uso de la capa semántica en productos de otro propietario. Además de un sistema de usuarios y de roles que aporta gobernabilidad, limitando el acceso a los datos tanto a nivel de fila como de columna. La aplicación únicamente almacena lo necesario para realizar la consulta del modelo construido a la fuente de datos SQL. De modo que tras conectar los datos se puede elegir la opción de escribir y almacenar directamente una consulta SQL o modelar gráficamente .

El modelo gráfico está orientado a fuentes de datos SQL basadas en diagramas en estrella. Tras seleccionar la tabla de hechos y sus dimensiones se pueden agregar y renombrar diferentes campos y nuevas medidas al modelo. Posteriormente a través de un traductor SQL implementado se genera y almacena la consulta capaz de extraer los datos modelados de la fuente de datos.

DECLARACIÓN DE AUTORÍA

TRABAJOS DE FIN DE GRADO / FIN DE MÁSTER

Escuela de Ingeniería y Arquitectura Universidad Zaragoza

DECLARACIÓN DE AUTORÍA Y ORIGINALIDAD

(Este documento debe remitirse a secelna@unizar.es dentro del plazo de depósito)

D./Dª. Alejandro Adell Pina ,

en aplicación de lo dispuesto en el art. 14 (Derechos de autor) del Acuerdo de 11 de septiembre de 2014, del Consejo de Gobierno, por el que se aprueba el Reglamento de los TFG y TFM de la Universidad de Zaragoza,

Declaro que el presente Trabajo de Fin de Estudios de la titulación de Grado en Ingeniería Informática (Título del Trabajo)

Capa semántica global para la gobernanza de datos previa a su análisis

es de mi autoría y es original, no habiéndose utilizado fuente sin ser citada debidamente.

Zaragoza, 10/10/2021

Fdo: Alejandro Adell Pina

ÍNDICE

AGRADECIMIENTOS

En primer lugar, a mi tutor Alejandro Moreno, por ofrecerme la posibilidad de realizar este Trabajo de Fin de Grado en HIBERUS TECNOLOGÍAS DIFERENCIALES, S.L, sin olvidarme de todas las personas del departamento de Data & Analytics que me formaron, ayudaron y aconsejaron durante todo el trabajo. Además a mi mentor de la universidad F. Javier Zarazaga por su ayuda en la planificación y organización de este trabajo.

Por último a mi familia y amigos, que han estado a lo largo de toda mi carrera apoyándome en todo momento y animándome a seguir adelante.

1 INTRODUCCIÓN

1.1 Contexto del Trabajo

A medida que el volumen de los datos crece, también lo hace su complejidad para explotarlos y analizarlos. Lo que dificulta el acceso a usuarios comerciales, usuarios normalmente no cualificados para la extracción de los datos, cuyo único objetivo es el análisis de datos claramente predefinidos.

Una capa semántica permite el análisis de autoservicio para una amplia audiencia al eliminar la necesidad de saber cómo se estructuran los datos, y al describir los datos en términos comerciales familiares. Esencialmente, una capa semántica funciona como middleware entre sus fuentes de datos y sus plataformas de Business Intelligence y machine learning, proporcionando conectividad virtualizada, modelado y otras capacidades de manipulación de datos. Debido a que todos los datos se filtran a través de esta capa semántica, todos los usuarios ven una sola versión de la verdad. Es decir, obtienen una versión consistente y coherente de los datos, usando las mismas medidas y dimensiones para informar sobre la misma realidad [2].

Una capa semántica proporciona definiciones compartidas de:

- **Modelo de datos:** conjunto de los datos, tablas, columnas, hechos, dimensiones, relaciones…
- **Métricas:** Conocidas también como KPIS, indicadores o valores, son representadas por números generados en una o varias operaciones o transacciones. Sirven para responder las preguntas referidas a cantidades o importes. Por ejemplo: ventas totales, media ingresos mensuales, total de costes...
- **Reglas de gobernanza:** Incluyen usuarios, grupos, roles, datos y permisos de aplicaciones.

El presente Trabajo Final de Grado se ha desarrollado en la empresa Hiberus Tecnología Diferenciales, S.L. Esta compañía está en un proceso de expansión de sus líneas de negocio, siendo todos los aspectos vinculados al análisis de datos uno de sus principales pilares.

1.2 Contexto Tecnológico

La naturaleza competitiva y dinámica del entorno empresarial de hoy está impulsando la demanda de sistemas de información que puedan proporcionar respuestas rápidas a consultas empresariales complejas. La industria de los sistemas de información respondió a estas demandas con desarrollos como las bases de datos analíticas, los

almacenes de datos, las técnicas de explotación de datos, las estructuras de bases de datos multidimensionales y sistemas de procesamiento analítico en líneas (OLAP). En un modelo de datos OLAP (Online Analytical Processing), la información es vista como cubos, los cuales consisten de categorías descriptivas (dimensiones) y valores cualitativos (medidas). El modelo de datos multidimensional simplifica a los usuarios formular consultas complejas, arreglar datos en un reporte, cambiar de datos detallados y filtrar o rebanar los datos en subconjuntos significativos [3].

Una forma de implementar este tipo de modelo de datos es mediante esquemas en estrella, un tipo de esquema de base de datos relacional que consta de una sola tabla de hechos central rodeada de tablas de dimensiones [4].

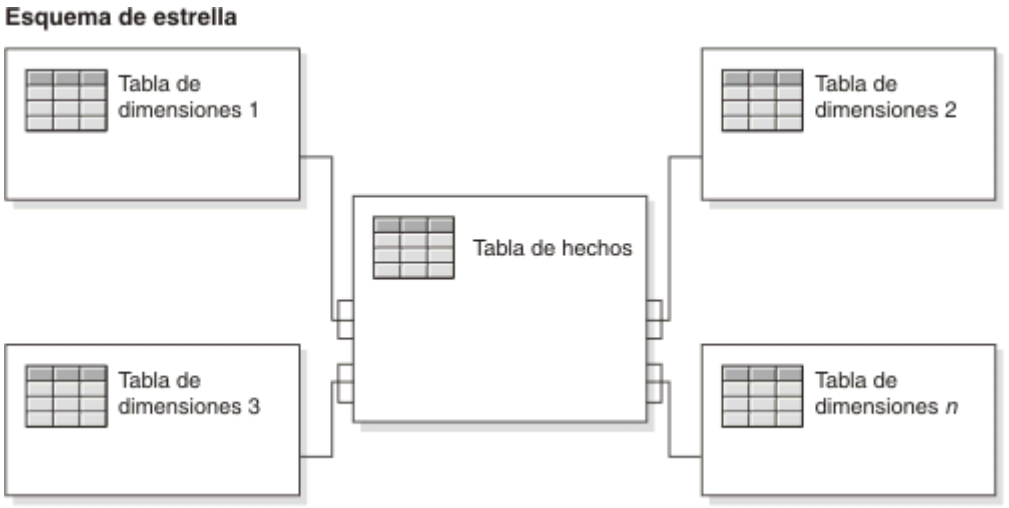


Figura 1: Esquema de estrella con una sola tabla de hechos con enlaces a varias tablas de dimensiones [4].

Los almacenes de datos son bases de datos especialmente diseñadas para ser consultadas y analizadas de forma óptima, en lugar de para el procesamiento de transacciones. Separa e independiza el proceso analítico del procesamiento transaccional [5].

1.3 Motivación y problema que se aborda

El departamento Data & Analytics de Hiberus Tecnología Diferenciales, S.L ofreció la posibilidad de realizar un Trabajo Final De Grado centrado en la investigación y experimentación con diferentes soluciones de capa semántica.

Se ha clasificado las diferentes posibles soluciones que disponen la posibilidad de modelar una capa semántica por el tipo de herramienta:

- **Plataformas Business Intelligence:** Plataformas tradicionales para modelar datos, realizar consultas y realizar diferentes informes y visualizaciones. Estas plataformas pueden tener incorporada su propia herramienta para modelar su capa semántica. Sus principales ventajas al modelar la capa semántica son una integración a medida para el usuario que utiliza la plataforma, su fácil uso al enfocarse al usuario final y está exenta de utilizar una tecnología adicional para el modelado. Por el contrario sus principales contras son la posibilidad de compartir y reusar el modelo ya que están ligados principalmente a aplicaciones de escritorio, donde cada usuario modela su propia capa semántica, como consecuencia se obtiene más de una versión de la verdad, lo que puede llevar a obtener conclusiones inconsistentes. Además generalmente están ligadas a "vendor lock-in", donde únicamente es posible compartir el modelo a herramientas del mismo proveedor, obligando a utilizar sus productos, esto puede afectar a grandes empresas donde sus análisis se realizan con más de una herramienta. Las principales plataformas de business intelligence del mercado son PowerBI, Tableau, Qlik y Looker.
- **Plataformas de virtualización de datos:** La virtualización de datos es una tecnología que permite combinar información procedente de fuentes de datos diversas y transformarlas en una única fuente virtual de datos a la que pueden acceder diferentes aplicaciones [6]. Alguna de estas plataformas abstrae la fuente y ubicación de los datos en un formato tabular, de modo que podría considerarse como un modelo semántico. Sus principales ventajas son la posibilidad de abstraer donde y como los datos están almacenados, además de poder combinar fuentes de datos de distinta procedencia. Al contrario que las plataformas Business Intelligence, se añade una capa de abstracción anterior al análisis de los datos, de modo que se consigue una capa semántica global, capaz de ser utilizada en diferentes herramientas de distintas plataformas. Por otra parte también tiene desventajas, estas plataformas no están diseñadas con el único fin de implementar una capa semántica, de modo que puede ser un proceso complejo y tedioso no enfocado a usuarios comerciales, esto también afecta en términos de rendimiento a la hora de realizar las consultas. Dos de las principales plataformas de virtualización de datos que permiten de cierto modo implementar una capa semántica son Dremio y Denodo.
- **Almacenes de datos:** Los almacenes de datos de los que se ha hablado en el anterior apartado, al igual que las plataformas de virtualización podrían usarse como capa semántica, consiguiendo una única versión de la verdad, un alto rendimiento a la hora de realizar consultas y fácil gobernabilidad. Pero estas plataformas en general están diseñadas para el procesamiento de los datos no para servirlos, integrar nuevos datos al modelo puede ser un proceso lento y complicado, además acceder a estos depende totalmente de la tecnología y no

está enfocado a un usuario comercial. Los principales almacenes de datos son Snowflake, Google BigQuery, Amazon Redshift , Azure Synapse SQL Analytics…

- **Soluciones de capa semántica:** Soluciones especializadas en ayudar a los usuarios a servir los datos de forma simple y en la terminología del negocio. Ofrecen una única visión de la verdad, flexibilidad para saber cómo y dónde están los datos almacenados, además de seguridad y gobernanza, capaz de admitir un inicio de sesión único, y seguridad en el acceso a los datos a nivel de fila y columna. Permiten dar soporte a más de una herramienta de visualización y machine learning, al proporcionar una capa más de abstracción en los datos. Las principales soluciones de capa semántica analizadas del mercado son AtScale, Microsoft Analysis Services y Kyvos.

En el anexo 1 se presenta un análisis más detallado de cada una de estas plataformas.

Tras un periodo de análisis y experimentación se llegó a la conclusión de que AtScale [7] y Kyvos [8], son las dos plataformas cloud dedicadas al modelado de capas semánticas con mejores prestaciones encontradas. Ya que además de tener todas las características ya mencionadas para este tipos de soluciones, están pensadas para entornos con un procesado masivo de los datos, reduciendo costes respecto a otras soluciones como Microsoft Analysis Services. Pero no ha sido posible experimentar con estas soluciones. AtScale ofrece la posibilidad de solicitar una versión de prueba, pero tras realizar varias solicitudes no se obtuvo respuesta. Por otra parte Kyvos ofrece instrucciones para desplegar su stack en Amazon de forma gratuita, pero la capa gratuita que ofrece Amazon no es suficiente para desplegarlo.

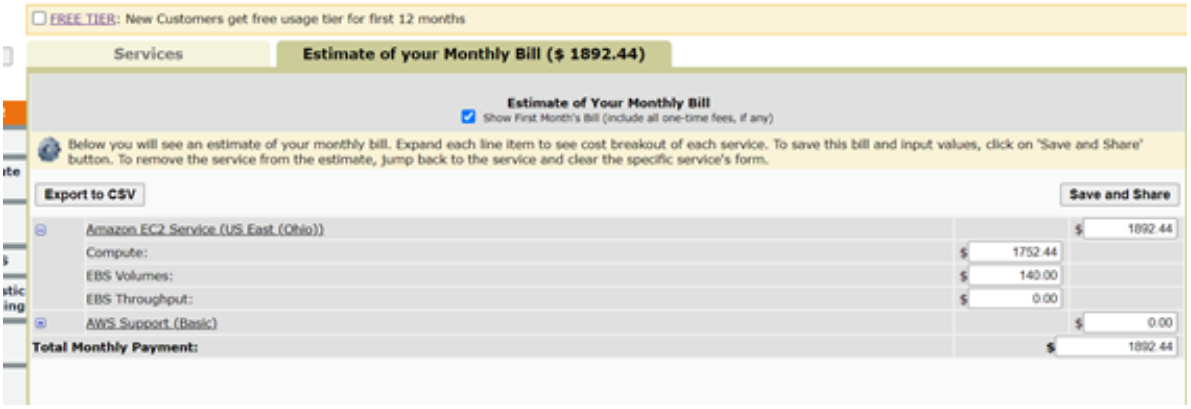


Figura 2: Estimación del coste mensual requerido en Amazon para desplegar el stack de Kyvos.

Tras estos resultados, se planteó la posibilidad de desarrollar de forma experimental una solución propia, una solución capaz de realizar el modelado de una capa semántica para una fuente de datos y que además posea las diferentes características descritas anteriormente.

1.4 Alcance, objetivos y limitaciones

El objetivo final planteado ha sido el desarrollo de una aplicación web, un prototipo de funcionalidad básica, que permita conectarse a fuentes de datos y guardar en datasets el modelo semántico creado.

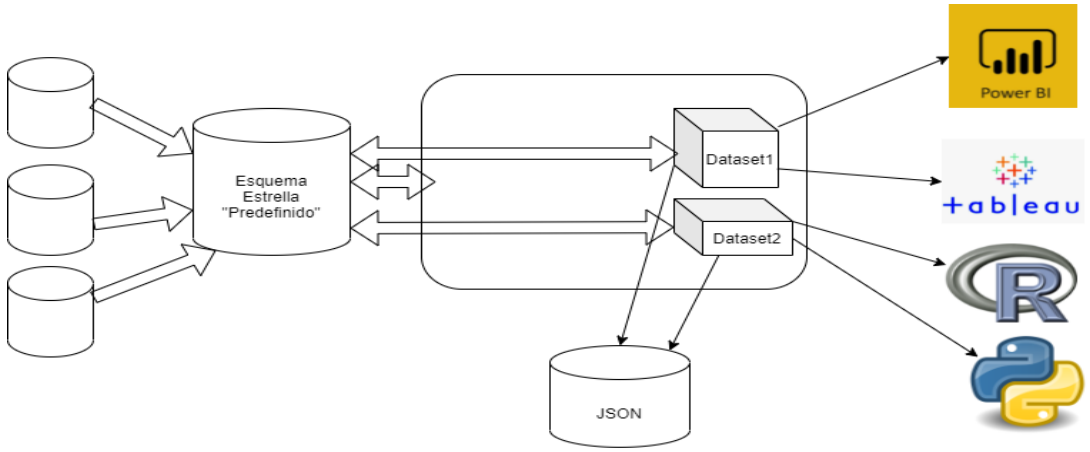


Figura 3: Idea planteada para el desarrollo de la aplicación.

Además esta aplicación web debe ofrecer una API para que los usuarios finales accedan directamente desde diferentes herramientas de Business Intelligence o Machine Learning a los datasets predefinidos. Esta API se basará en un traductor JSON-SQL, donde se almacenará en formato JSON únicamente la estructura de los datasets creados y lo necesario para realizar la consulta al origen de los datos.

Los objetivos principales buscados en el desarrollo de la aplicación web son:

- **Finalidad de capa semántica**: No se debe almacenar los datos modelados de la consulta, únicamente se almacena la estructura modelada en formato JSON para traducirla y realizar la consulta al recibir una petición.
- **Capa semántica global**: Ofrecer al usuario final una única versión consistente de los datos modelados, sin vendor lock-in, disponible para múltiples herramientas.
- **Modelado Gráfico**: Modelado de los datos de forma gráfica, sin tener que escribir tediosas consultas SQL.
- **Modelado SQL**: Aparte del modelado gráfico, que exista la posibilidad de modelar la consulta empleando código SQL, útil para consultas muy específicas.
- **Sistema de usuarios que aporta gobernabilidad**: poder limitar la accesibilidad de los datos por usuario. Tanto a nivel de fila como de columna.

Al ser un desarrollo experimental, una de las principales limitaciones para el desarrollo de la aplicación es a nivel económico. Por eso se ha decidido realizar toda la aplicación bajo un entorno totalmente gratuito. Igualmente otro de los grandes problemas es la dimensión de la aplicación, por ello se ha decidido centrarse únicamente en un tipo de fuente de datos, bases de datos MySQL. Diferentes fuentes de datos pueden utilizar diferente formato de acceso a los datos, incluso diferente dialecto SQL, lo que supondría un gran esfuerzo a la hora de implementar un traductor JSON-SQL y la forma en que cada fuente de datos interactúa con la aplicación. Del mismo modo se ha decidido limitar el modelado gráfico a únicamente fuentes de datos con un diagrama en estrella.

1.5 Herramientas de trabajo

Para el diseño de la aplicación se ha empleado Balsamiq Wireframes [9], una interfaz de usuario para el diseño de mockups que permite generar un primer diseño de la aplicación de forma inmediata.

La implementación de la aplicación web se ha realizado mediante el "MEAN stack" . MEAN Stack (acrónimo para MongoDB, Express.js, AngularJS, Node.js) es un framework o conjunto de subsistemas de software para el desarrollo de aplicaciones y páginas web dinámicas que están basadas, cada una de estas, en el lenguaje de programación JavaScript. Gracias a esta característica el conjunto se integra exitosamente en una plataforma autosuficiente [10].

MongoDB es una base de datos multiplataforma orientada a documentos que proporciona alto rendimiento, alta disponibilidad y fácil escalabilidad. MongoDB evita la estructura basada en tablas de la base de datos relacional para adaptar documentos similares a JSON [11].

NodeJs es un run-time de JavaScript construido sobre el motor de JavaScript de Chrome . Node.js usa un modelo de E/S sin bloqueo controlado por eventos que lo hace liviano y eficiente [12].

Express es un marco de aplicación web Node.js mínimo y flexible que proporciona un conjunto sólido de funciones para aplicaciones web y móviles [11]. Es utilizado para crear de forma rápida y fácil una API robusta de llamadas HTTP.

AngularJS, es un framework de JavaScript de código abierto, mantenido por Google, que se utiliza para crear y mantener aplicaciones web de una sola página. Su objetivo es aumentar las aplicaciones basadas en navegador con capacidad de Modelo Vista

Controlador (MVC), en un esfuerzo para hacer que el desarrollo y las pruebas sean más fáciles [11].

Tanto para el servidor como para la aplicación web se ha utilizado Visual Studio Code [13] como editor de código fuente.

Para el control de versiones se ha usado GitHub, una plataforma de desarrollo que emplea el sistema de control de versiones git.

Se ha utilizado KNIME, una plataforma de minería de datos que permite el desarrollo de modelos en un entorno visual, para generar correctamente datasets de prueba y almacenarlos en una base de datos MySQL.

Se ha usado Postman, una aplicación que permite realizar pruebas API. Para probar las diferentes peticiones HTTP de la API del backend de la aplicación.

Por último se han utilizado diferentes herramientas de business intelligence como PowerBI para probar el funcionamiento final de la aplicación. Donde a través del API de la aplicación se conectaba a los datos previamente modelados en la aplicación web, para generar distintos informes.

1.6 Esquema general de la memoria del proyecto

La memoria de este trabajo comienza con una introducción a la línea de investigación, a las tecnologías con las que se ha trabajado, a los problemas que se abordan, los objetivos y limitaciones del trabajo. Posteriormente se explica de forma detallada los requisitos del sistema, el diseño del sistema planteado y su implementación. Esta información se ve complementada con cinco anexos:

El anexo 1 complementa el apartado de la memoria Motivación y problema que se aborda, aportando un análisis detallado de diferentes productos del mercado con capa semántica propia y la problemática a la que se enfrentan.

El anexo 2 complementa el apartado de la memoria Diseño del sistema, concretamente al subapartado Diseño de la interfaz de usuario , donde se presenta y comenta el prototipo de la interfaz gráfica de la aplicación web, realizado con Balsamiq Wireframes.

El anexo 3 complementa el apartado de la memoria Diseño del sistema concretamente el subapartado diseño de la base de datos, explica en detalle el funcionamiento del objeto “ConsultaSQL” utilizado para el modelado gráfico de datasets y se muestra un ejemplo del funcionamiento del traductor JSON-SQL.

En el anexo 4 se muestra de forma detallada la navegación completa de la aplicación web.

En el anexo 5 se muestra un ejemplo real de acceso a los datos modelados de un dataset modelado en la aplicación web, desde la herramienta externa Power BI.

2 TRABAJO DESARROLLADO

2.1 Requisitos del sistema

Anteriormente se ha descrito los requisitos básicos que debe tener una aplicación basada en el modelado de una capa semántica global, a continuación, basándose en estos, se van a nombrar los requisitos funcionales y no funcionales que debe tener la aplicación a implementar.

Como punto de partida, en el sistema a desarrollar, cualquier persona que acceda a la aplicación puede registrarse como nuevo usuario (autorregistro), con el fin de poder crear y gestionar sus propios datasets. No se ha establecido un sistema de control de usuarios por parte de un administrador dado que el alcance es el de un prototipo. Por otro lado en cada dataset se pueden crear diferentes perfiles de usuario con un nombre y correo electrónico (no tienen porque estar registrados en la aplicación), usados para asegurarse que desde herramientas ajenas a la aplicación, únicamente realicen consultas a los datos modelados del dataset, usuarios con permiso definido en el propio dataset. Además a estos accesos de usuario definidos en los datasets se les asigna reglas de seguridad de acceso a los datos a nivel de fila y columna, que se les aplicarán al acceder a los datos modelados por el dataset desde herramientas externas a la aplicación web.

REQUSISTOS FUNCIONALES	
RF-1	La aplicación permitirá iniciar sesión a usuarios registrados y crear nuevos usuarios.
RF-2	Los usuarios que hayan iniciado sesión podrán ver su perfil (nombre usuario y correo electrónico) y cambiar su contraseña.
RF-3	Los usuarios que hayan iniciado sesión podrán ver, editar y borrar, sus datasets previamente creados.

RF-4	Los usuarios que hayan iniciado sesión podrán crear nuevos datasets, conectándose a un nuevo origen de datos. El nivel de prototipo del sistema a desarrollar debe ser diseñado para dar cabida a fuentes de datos heterogéneas. Sin embargo, a nivel de prototipo se va a requerir únicamente la implementación en MySQL.
RF-5	Al crear un nuevo dataset se permitirá elegir el tipo de modelado, gráfico o código SQL.
RF-6	Un dataset con modelado gráfico seleccionará cuál es la tabla de hechos de la fuente de datos y posteriormente podrá agregar nuevas tablas dimensión, seleccionando qué columna de estas es la clave referenciada en la tabla de hechos.
RF-7	En un dataset con modelado gráfico, tanto en la tabla de hechos como las dimensiones se podrán agregar y renombrar las columnas que el dataset quiera mostrar, además de agregar nuevas métricas, filtros o agrupaciones.
RF-8	Un dataset con modelado gráfico podrá visualizar y eliminar las tablas dimensión, columnas, métricas, condiciones y agrupaciones que tiene agregadas.
RF-9	Un dataset con modelado tipo código SQL, podrá visualizar, escribir y guardar directamente la sentencia SQL que define el dataset, así como testear el número de filas que devuelve la sentencia.
RF-10	Los usuarios que hayan iniciado sesión podrán ver la información de todos sus datasets creados, consulta SQL que genera y el número de filas que devuelve.
RF-11	Los usuarios que hayan iniciado sesión podrán ver, agregar y eliminar quién tiene acceso a cada uno de sus datasets, así como añadirles o editar su seguridad a nivel de fila y columna.

Tabla 1: Requisitos funcionales de la aplicación.

REQUISITOS NO FUNCIONALES	
RNF-1	La aplicación tendrá una interfaz clara e intuitiva basada en cajas, que guíe al usuario en el proceso de crear nuevos datasets.
RNF-2	Cada dataset tendrá definido un conjunto de usuarios (con un correo electrónico asociado, no tienen porqué estar registrados en la aplicación),a los que se le asignará un token, con el que podrán acceder a los datos generados por el dataset (respetando la seguridad a nivel de fila y columna que se les haya asignado), desde herramientas ajenas a la aplicación, mediante una API REST HTTP.
RNF-3	La aplicación mostrará en todo momento un feedback al usuario que le informe

	de que sus operaciones se han realizado correctamente o que problemas han ocurrido.
RNF-4	La aplicación deberá tener un sistema de seguridad que garantice la integridad de la fuente de datos a consultar, impidiendo sentencias SQL que modifiquen los datos.

Tabla 2: Requisitos no funcionales de la aplicación.

2.2 Arquitectura software del sistema

A partir de la idea inicial mostrada en la figura 3, se ha diseñado un nuevo diagrama general del sistema, incluyendo las tecnologías a utilizar.

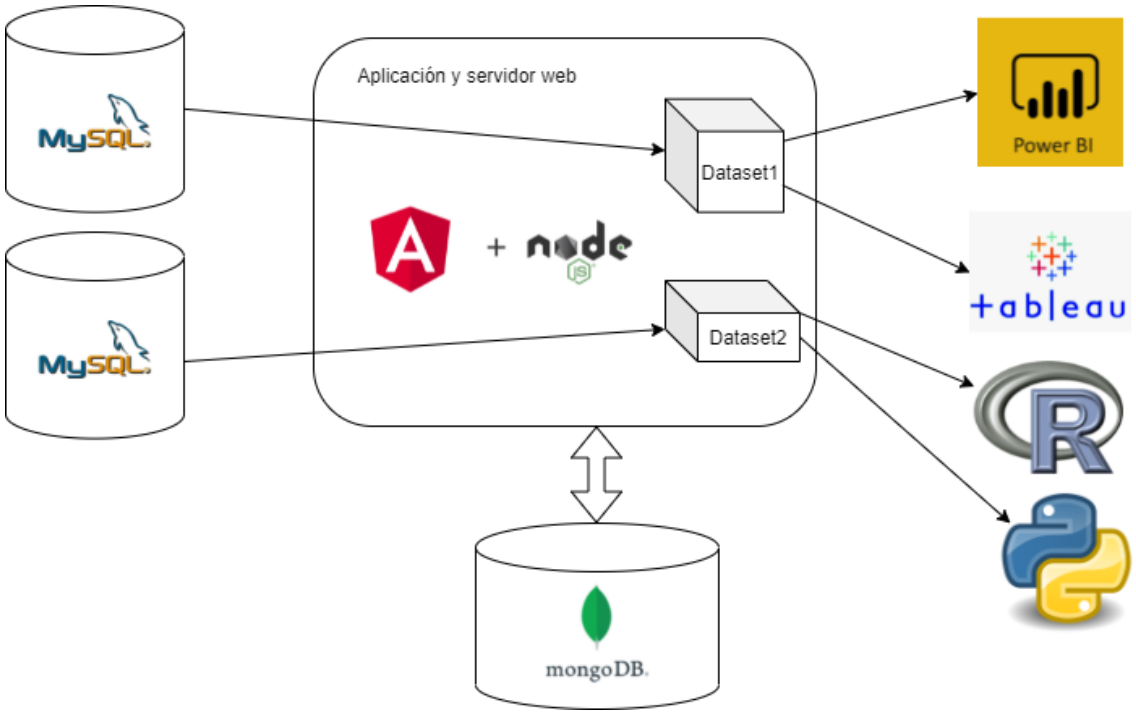


Figura 4: Diagrama general del sistema.

Por un lado la aplicación web utiliza el framework de Angular que se encarga de toda la parte visible de la aplicación, la gestión de usuarios, creación y configuración de datasets. A su vez el servidor web utiliza NodeJS y Express [14] , que se encarga de gestionar las diferentes peticiones de la aplicación, comunicarse con las fuentes de datos externas MySQL a la que acceden los diferentes datasets y ofrecer un servicio API REST HTTP, desde el que herramientas externas como PowerBI, Tableau, Python y R, puedan conectarse a los datos modelados. El servidor web se comunica con una base

de datos mongoDB alojada en un servidor cloud de MongoDB Atlas [15], almacenando toda la información de usuarios y datasets de la aplicación.

Se ha diseñado un diagrama de despliegue que muestra la arquitectura de ejecución del sistema.

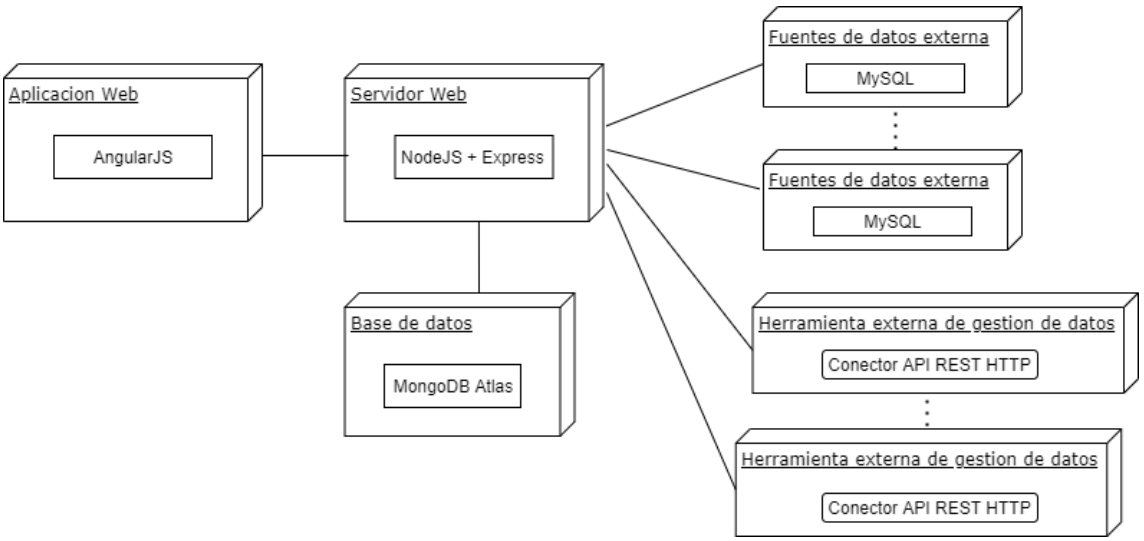


Figura 5: Diagrama de despliegue.

La aplicación web implementada sobre Angular se conecta a la API del servidor web por medio de peticiones HTTP. El servidor web se conectará a la base de datos MongoDB ubicada en un cluster de MongoDB Atlas a través del paquete de mongoose [16] utilizando la URL del clúster de MongoDB Atlas. El servidor web extraerá la información de los datos modelados en los datasets, conectando a fuentes de datos externas MySQL, mediante el paquete mysql disponible en NPM [17] y utilizando los datos de conexión definidos en el dataset. Herramientas externas se podrán conectar a los datos modelados si disponen de un conector API REST HTTP [18].

2.3 Diseño del sistema

Diseño de la interfaz de usuario

Primero, se diseñó un prototipo de la interfaz de usuario con la herramienta Balsamiq Wireframes, centrándose en un diseño minimalista e intuitivo basado en cajas.

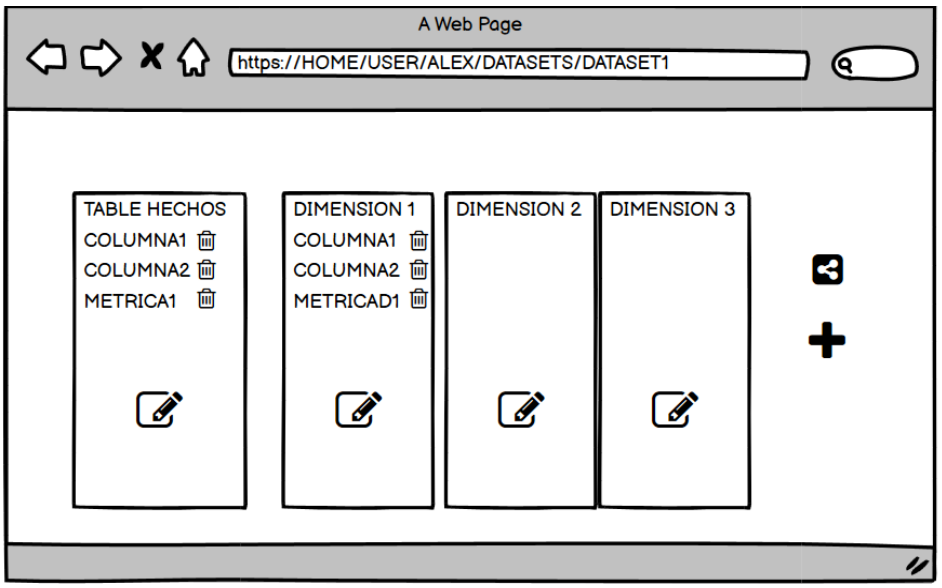


Figura 6: Ejemplo del prototipo diseñado con Balsamiq Wireframes.

En el anexo 2 se presenta el diseño completo del prototipo con información más detallada.

Diseño del software

Hay que separar el diseño del software del sistema en la aplicación web y el diseño del servidor web.

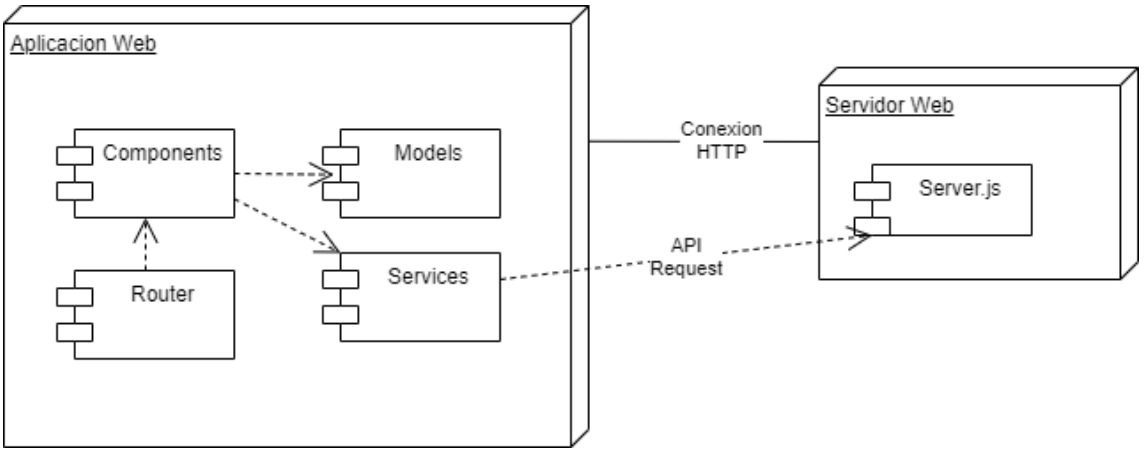


Figura 7: Diagrama de componentes de la aplicación web.

Por un lado, la arquitectura de la aplicación web implementada sobre Angular se puede simplificar en cuatro componentes. Router es el encargado de gestionar la navegación de la web, enrutando todas sus páginas y redirecciones, en Models están definidas las

diferentes estructuras de datos con las que la aplicación va a trabajar, en este caso, usuarios y datasets, que se explican de forma más detallada en el apartado dedicado para el diseño de la base de datos. En Services están implementados en ficheros typescript [19], todo lo necesario para interactuar con la API del servidor web, es decir, tanto para la gestión de los usuarios, como para los datasets estarán definidos sus servicios que permitan realizar la petición correspondiente al servidor web, por ejemplo, crear, borrar, editar datasets y usuarios. Y por último en los componentes están implementadas todas las páginas de la aplicación web.

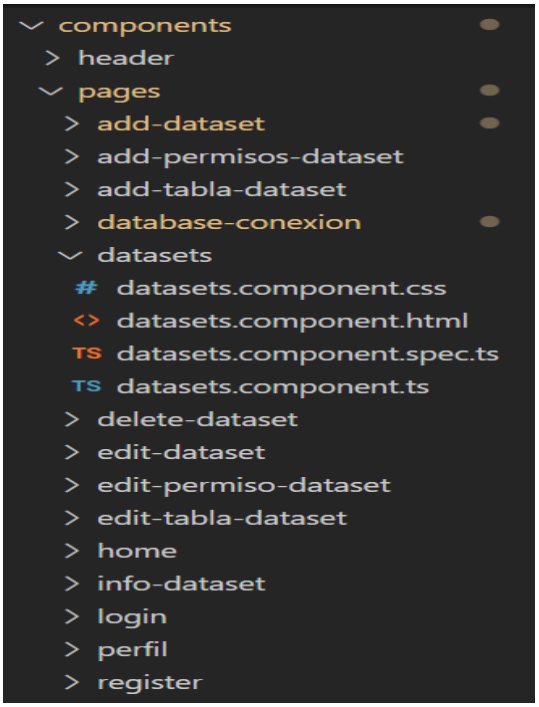


Figura 8: Estructura de los diferentes componentes de la aplicación web.

Como se observa en la anterior imagen, los componentes están clasificados por páginas de la aplicación web. Cada una de estas páginas tendrá un fichero ".ts" que corresponde al lenguaje de programación typescript [19], donde estará definida toda la lógica de la página, desde validar los campos de un formulario, a cómo accede a diferentes servicios para interactuar con el servidor web. Además de un fichero html [20] donde está definido el contenido de la página, un fichero css [21] donde están definido los estilos de la página y un archivo autogenerado ".spec.ts" para pruebas unitarias.

Por otro lado, el servidor web está desarrollado sobre la plataforma NodeJS y ExpressJS, estas herramientas, a través de declaraciones de eventos, permiten

administrar el hilo de ejecución principal del programa para atender diferentes peticiones HTTP.

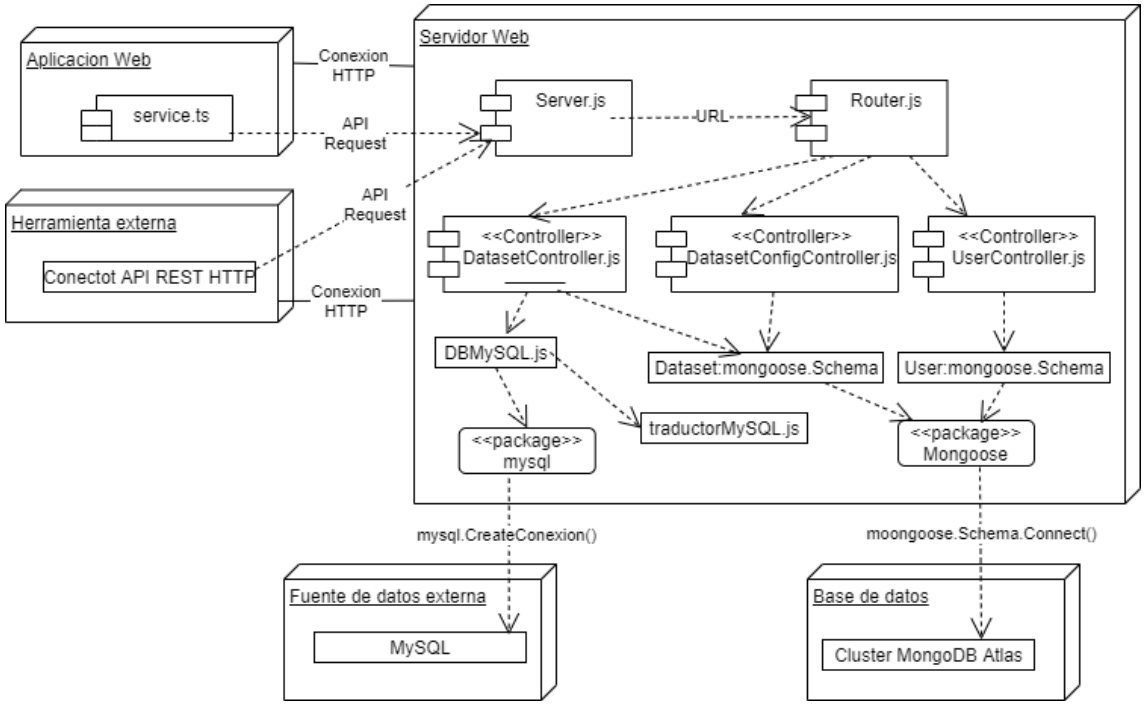


Figura 9: Diagrama de componentes del servidor web.

Llegan llamadas HTTP a la API del servidor tanto de la aplicación web, como de herramientas externas que quieran utilizar los datos modelados en los datasets. Router.js se encarga de atender dichas peticiones HTTP, definiendo con qué recursos debe responder a cada petición. Se ha clasificado en tres tipos de peticiones que llegan al servidor, cada tipo de petición es gestionada por uno de los controladores del servidor.

"DatasetController.js" es el controlador principal, encargado de gestionar las peticiones que necesitan conectar con la fuente de datos externa definida en cada dataset. Trata peticiones de la aplicación web como obtener los nombres de las tablas y sus columnas de la fuente de datos, para poder mostrar estos nombres a la hora de realizar el modelado gráfico o peticiones como realizar un test al dataset para obtener el número de filas que devuelve su consulta a la fuente de datos. También es el encargado de gestionar las peticiones de herramientas externas para obtener los datos modelados de cada dataset. Esta última es la petición principal sobre la que se basa la aplicación, esta petición llega con los parámetros dataset (nombre del dataset), usuario (nombre del acceso a usuario definido en el propio dataset, con reglas de seguridad a nivel de fila y columna definidas) y token (token asignado al anterior usuario para permitirle el acceso al dataset), a través de mongoose se conectara al cluster de MongoDB, si existe un dataset, con ese nombre y permisos definidos para el usuario con ese token, se

encargará de traer la información del dataset al servidor web, según su tipo de fuente de datos (al ser un prototipo únicamente está implementado para MySQL, pero está pensado para ser escalable a otras fuentes) acedera al archivo ".js" donde está definido como interactuar con ese tipo de fuente, si el dataset utiliza un modelado gráfico, usará "traductorMysql.js" para traducir la estructura del dataset definida en un objeto JSON (explicado más detalladamente en el apartado de diseño de la base de datos) al código de su consulta MySQL. Si no utiliza el modelado gráfico, utilizará directamente la consulta SQL definida. Antes de realizar la consulta a la fuente de datos, se añadirán las reglas de seguridad a nivel de fila o columna definidas para ese usuario, para que únicamente acceda a los datos que no se le han restringido. Por último utilizando el paquete "mysql" y con los datos de conexión del dataset se conectará a la fuente de datos para realizar la consulta construida y devolverá estos datos a la herramienta externa que ha realizado la petición HTTP.

"DatasetConfigController.js" es el encargado de gestionar las peticiones de la aplicación web relacionadas con la administración de los dataset, añadir un nuevo dataset a la base de datos, actualizar o borrar un dataset existente, obtener la información de todos datasets pertenecientes a un usuario y de obtener la información de un dataset determinado. Se conecta a la base de datos de MongoDB alojada en el cluster de MongoDB Atlas, utilizando el esquema del dataset definido en mongoose y utilizando la URL de conexión al clúster.

"UserController.js" es el encargado de gestionar las peticiones de la aplicación web relacionadas con la administración de los usuarios, validar los datos de inicio de sesión de un usuario en la aplicación, registrar nuevos usuarios en la aplicación, cambiar la contraseña a uno de los usuarios y mostrar la información de uno de los usuarios. Se conecta a la base de datos de MongoDB alojada en el cluster de MongoDB Atlas, utilizando el esquema de la información de los usuarios definido en mongoose y utilizando la URL de conexión al clúster.

Diseño de la base de datos

Se ha usado una base de datos MongoDB alojada en un cluster de MongoDB Atlas para almacenar los datos de la aplicación web. Además se ha utilizado mongoose una herramienta de mapeo objeto-relacional (ORM), que permite escribir consultas y realizar esquemas de los datos en una estructura JSON. En la base de datos hay definidas dos colecciones de datos usuarios y datasets, estas colecciones de datos también estarán definidas en la aplicación web y en el servidor web utilizando mongoose Schema.

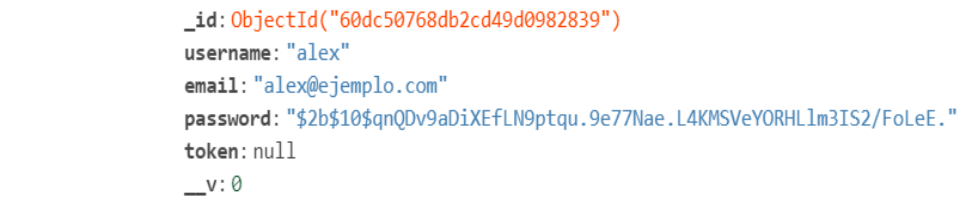


Figura 10: Ejemplo de usuario almacenado en la base de datos MongoDB.

Los usuarios son almacenados en la base de datos con los siguientes campos, “username”, “email”, “password” y “token”. Password no almacena directamente la contraseña proporcionada por el usuario, si no que guarda la contraseña encriptada como una clave hash, utilizando el paquete ‘bcrypt’ [22], consiguiendo una mayor privacidad a la hora de almacenar las contraseñas. En el campo token, se almacena un JSON Web Token [23], utilizando el paquete ‘jsonwebtoken’ [24], al iniciar sesión en la aplicación web se le genera al usuario un token de sesión que le permite hacer uso de las diferentes peticiones del servidor web. Por lo que si le llega una petición definida para la aplicación al servidor sin un token de sesión válido, denegará la petición.

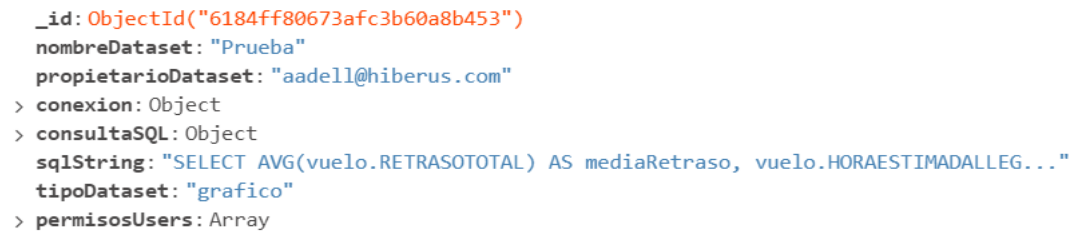


Figura 11: Ejemplo de dataset almacenado en la base de datos MongoDB.

La estructura de almacenamiento de los datasets, es una estructura compleja con los campos “nombreDataset”, “propietarioDataset” (corresponde al correo electrónico del usuario que lo ha creado), “tipoDataset” (corresponde al tipo de modelado que usa el dataset, gráfico o código SQL) y “sqlString” (corresponde a la consulta SQL, si el tipo de dataset es código SQL será directamente la consulta que ha escrito el usuario para modelar el dataset, si el tipo de dataset es gráfico, corresponderá al resultado de utilizar el traductor JSON-SQL implementado para traducir el objeto JSON “consultaSQL”). El objeto “conexion”, “consultaSQL” y el array “permisosUser” son estructuras más complejas que se explican a continuación.

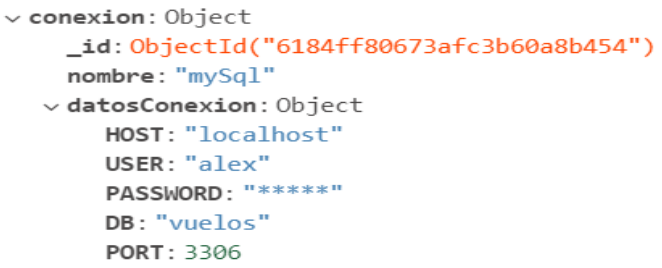


Figura 12: Ejemplo del objeto "conexion" de un dataset almacenado en la base de datos MongoDB.

En el objeto "conexion" perteneciente a la estructura de dataset, se encuentra el campo nombre (corresponde al tipo de conexión, en este caso una conexión a una base de datos MySQL), el objeto "datosConexion" dependiendo del tipo de conexión, tendrá la información necesaria para conectarse a esa fuente de datos (host, usuario, contraseña, base de datos , puerto...).

Figura 13: Ejemplo del objeto "permisosUsers" de un dataset almacenado en la base de datos MongoDB.

En el objeto "permisosUsers" se define para cada dataset, permisos de acceso a los datos modelados, se definen accesos a usuarios (no tienen que corresponder con usuarios registrados) con un nombre, correo (utilizado para enviarle un mensaje en caso de que el token sea actualizado) y se le asigna un token de acceso único generado de forma aleatorio usado para acceder desde herramientas ajenas a la aplicación a los datos modelados. También para cada uno de los accesos a usuario definidos, se puede configurar la seguridad de acceso a nivel de columna, definida en el objeto "columnasRestringidas" y la seguridad de acceso a nivel de fila, definida en el array "filtrosUser", que cada objeto contiene los campos "nombreColumna" (corresponde a la columna sobre la que se va a aplicar el filtro) y "filtro" (corresponde a el filtro que se

le va a aplicar en la columna). Como ejemplo, en la figura 13 se ha definido un acceso con nombre de usuario “Alex”, que a nivel de columna se le ha restringido la columna “NUMVUELO” y a nivel de fila se le ha aplicado un filtro “ =California” sobre la columna “ESTADO” que provocará que únicamente acceda a los datos modelados en los que la columna “ESTADO” sea igual a “California”.

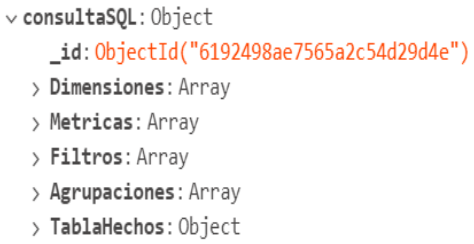


Figura 14: Ejemplo del objeto “consultaSQL” de un dataset almacenado en la base de datos MongoDB.

El objeto “consultaSQL” se utiliza en los datasets con modelado gráfico, facilitando a la aplicación web tener una estructura de datos donde gráficamente sea sencillo modificar el dataset. Además es la estructura utilizada por el traductor JSON-SQL implementado, donde a partir de los datos definidos en este objeto, obtiene la consulta SQL equivalente. En el anexo 3 se presenta la estructura de “consultaSQL” y el funcionamiento del traductor JSON-SQL de forma más detallada.

2.4 Dimensión del trabajo realizado

Inicialmente la propuesta del trabajo era de un ámbito más teórico, basado en la investigación, análisis y experimentación con diferentes soluciones del mercado que proporcionan una solución de capa semántica. Tras un periodo de estudio y análisis, debido a la poca disponibilidad de las mismas (por coste, en opciones de evaluación, y soluciones open), se decidió (de manera negociada entre empresa y alumno) cambiar el enfoque del trabajo. De este modo, se optó por diseñar e implementar una solución propia, un prototipo de aplicación web, que permita el modelado semántico de datos, basándose en las características que se consideran deseables a partir de las obtenidas al analizar este tipo de herramientas. De este modo, el prototipo servía tanto como base de un hipotético futuro desarrollo propio de la empresa, como de base para comparar soluciones comerciales o de software libre que puedan ir considerándose.

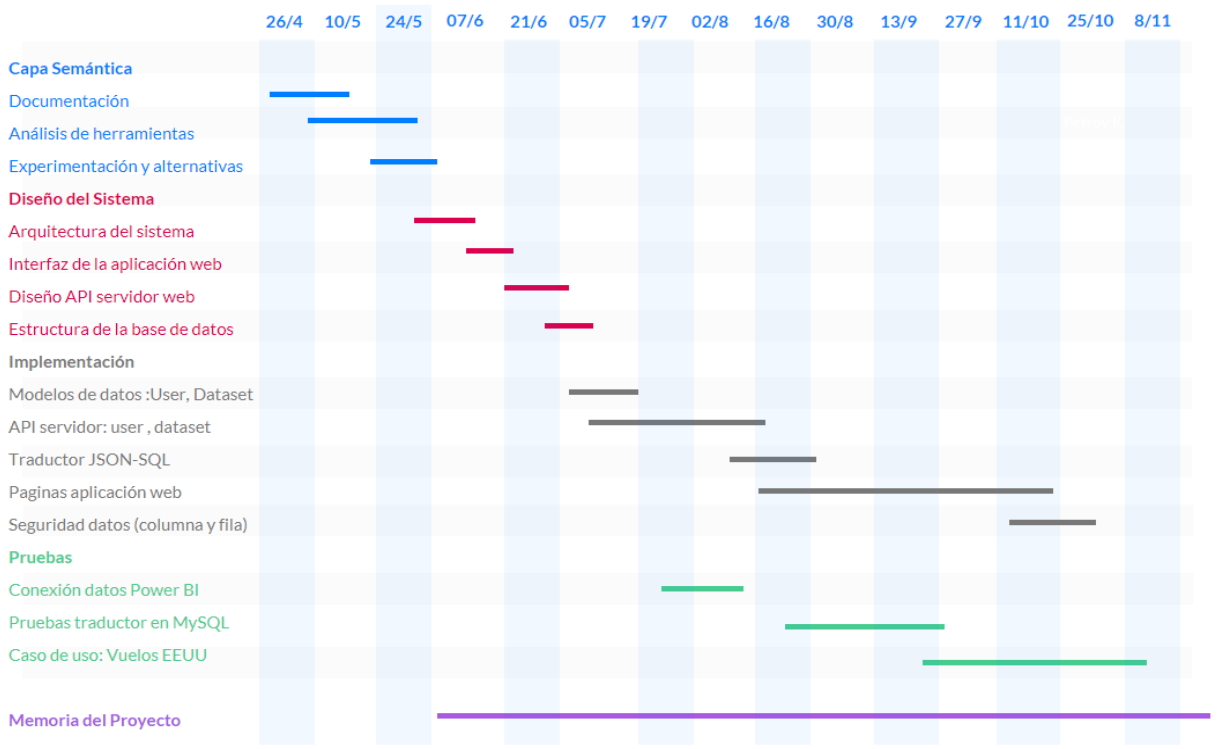


Figura 16: Diagrama de Gantt del proyecto.

Una vez claro el enfoque del proyecto, se empezó a diseñar y construir todo el sistema de acuerdo al conjunto de actividades y calendario que refleja el diagrama de Gantt adjunto. En el mismo, hay que destacar que, aunque la estructura de Dataset y el traductor JSON-SQL están marcados como un único periodo, ha sido modificado a lo largo del proyecto de forma incremental, al añadir nuevas funcionalidad en el modelo gráfico de la aplicación web o al depurar errores al realizar pruebas más específicas con el caso de uso de los vuelos de Estados Unidos.

Aunque el número de líneas de código no es una de las mejores métricas para determinar el volumen de trabajo, puede resultar útil para estimar su dimensión.

```
aadell@HL-012198 MINGW64 ~/Desktop/proyecto/frontend/frontTFG/src (master)
$ git ls-files | xargs cat | wc -l
7570
```

Figura 17: Número de líneas de código de la aplicación web, excluyendo librerías y archivos de configuración.

```
aadell@HL-012198 MINGW64 ~/Desktop/proyecto/backend/TFG-Alejandro (backend)
$ git ls-files | xargs cat | wc -l
4161
```

Figura 18: Número de líneas de código del servidor web, excluyendo librerías.

Sumando el número de líneas de código de la aplicación y el servidor web resulta de un total de 11731 líneas de código.

2.5 Problemas encontrados

Uno de los principales problemas a la hora de idear una solución que permita el modelado de una capa semántica global fue determinar cómo conectarse con múltiples herramientas externas. Se optó por solucionarlo a través de un servicio API REST HTTP desde el servidor web, ya que la gran mayoría de herramientas de gestión de datos permiten utilizar este tipo de conectores y no requieren de tecnología muy específica.

A parte de crear modelos semánticos guardando directamente consultas SQL, era interesante que la aplicación permitiera realizar modelos gráficamente, posibilitando así que usuarios comerciales no tan especializados puedan realizar sus propios modelos. Trabajar directamente desde un código SQL resultaba imposible, por ello era necesario crear una estructura intermedia que permita dinámicamente agregar y quitar elementos al modelo. La idea de crear esta estructura junto a su posterior traductor a SQL, surge tras analizar la herramienta Looker y su lenguaje propio LookML, un lenguaje para describir dimensiones, agregados, cálculos y relaciones de datos en una base de datos SQL [25]. Además tener un traductor universal a SQL que permita consultar fuentes de distintos dialectos SQL enseguida resultó una tarea imposible, por ellos se optó por centrarse únicamente en un tipo de fuente de datos Mysql y estructurar el servidor de manera que sea escalable a otras fuentes de datos, siendo únicamente necesario crear su propio traductor y la lógica de cómo conectarse a la nueva fuente de datos.

3 Lecciones aprendidas y conclusiones

3.1 Conocimientos adquiridos

En primer lugar he aprendido que es una capa semántica, sus ventajas y la necesidad de tener un buen modelo de datos antes de analizarlos. Se ha reforzado conocimientos en lenguajes como JavaScript , Typescript , HTML, CSS, JSON,SQL , en plataformas como Angular, entornos de ejecución como NodeJS y diseño de bases de datos NoSQL como MongoDB. Además he conseguido una amplia visión del panorama actual relacionado con el análisis de datos y big data, aprendiendo y analizando el funcionamiento de

distintos tipos de plataformas de gestión de datos, como plataformas de virtualización de datos, de business intelligence, almacenes de datos y de modelado semántico de datos.

Por último a nivel personal he conseguido la madurez necesaria para realizar un proyecto de cierta complejidad, solucionando los problemas que iban surgiendo y tomando mis propias decisiones.

3.2 Ideas Futuras

Para llevar la aplicación a un entorno de producción se deberían realizar ciertas mejoras que hicieran la solución mucho más atractiva. La mayoría de las mejoras están relacionadas con el modelado gráfico de datasets de la aplicación, en primer lugar sería necesario añadir funcionalidades al modelado gráfico como permitir añadir dimensiones con clave compuesta, ya que actualmente no está diseñado para comparar claves compuestas entre la tabla de hechos y una tabla dimensión, además permitir modelar estructuras de datos con dimensiones anidadas o otros tipos de estructura de datos como el esquema en copo de nieve [26]. Otra de las principales mejoras a realizar es ampliar la cantidad de fuentes de datos a la que se puede conectar la aplicación para modelar sus datos, añadir otros gestores de bases de datos aparte de MySQL o incluso añadir servicios de bases en la nube como Google BigQuery o SnowFlake. Conseguir que funcione el modelado gráfico con otras fuentes de datos no es una tarea sencilla ya que para cada una de las nuevas fuentes habría que adaptar el traductor JSON-SQL a su sintaxis SQL propia, una de las soluciones más óptimas sería combinar esta aplicación con una plataforma de virtualización de datos que le proporcionará conectividad con una gran cantidad de fuentes de datos, teniendo únicamente que adaptar el traductor JSON-SQL para poder acceder a los datos de la plataforma de virtualización.

3.3 Conclusiones

Conclusiones sobre el trabajo técnico desarrollado

Tras analizar los diferentes tipos de plataformas que permiten el modelado de capas semánticas de datos, se ha llegado a la conclusión de que no existe una solución perfecta que sirva para cubrir todos los casos. Dependiendo de la complejidad, volumen y disponibilidad de los datos, de la dimensión del proyecto y presupuesto económico se puede elegir que tipo de plataforma se adapta mejor a esas necesidades. Finalmente tras plantear realizar una solución propia, un prototipo de aplicación web

que permita el modelado semántico de los datos, basándose en características obtenidas del análisis previo de este tipo de soluciones se ha conseguido llegar a los siguientes objetivos:

- **La solución permite el modelado de capas semánticas globales** que aportan consistencia y gobernanza a los datos antes de su análisis, además sin vendor-lock in, permitiendo analizar estos datos desde herramientas externas. Actuando como una capa mas de abstracción al no almacenar todos los datos modelados, almacenando únicamente la estructura y los datos necesarios para realizar la consulta a la fuente de datos
- **La aplicación web permite diferentes opciones de modelado semántico**, permitiendo realizar un modelo gráfico, sin tener que escribir tediosas consultas SQL, útil para usuarios comerciales con poca experiencia con SQL y permitiendo además modelar directamente la consulta SQL, recibiendo un feedback de los errores y el número de filas obtenidos en consulta, útil para consultas muy específicas que no puedan ser realizadas con el modelado gráfico.
- **Sistema de control de acceso a usuarios a los datos que aporta gobernabilidad**, permitiendo definir reglas de acceso a los datos tanto a nivel de fila como de columna.

El resultado final de la aplicación puede comprobarse en el anexo 4, donde se muestra la navegación completa de la aplicación web y en anexo 5 donde se realiza un ejemplo de acceso a los datos modelados de un dataset desde la herramienta externa Power BI.

Conclusión personal

A nivel personal, este proyecto me ha hecho madurar y darme cuenta de lo capaz que puedo llegar a ser. Además me ha permitido incorporarse en el mundo laboral, estar en un ambiente de trabajo real y conocer muchos profesionales del ámbito de la ingeniería y el análisis de datos que me han formado y ayudado a ampliar mis conocimientos en diversas tecnologías que pueden ser claves en mi futuro laboral.

4 BIBLIOGRAFÍA

[1] Anand, Ajay. “What is a Semantic Layer? How to Build it for Future Data Workloads?” *Kyvos Insights*, 13 January 2020, https://www.kyvosinsights.com/blog/what-is-a-semantic-layer-how-to-build-one-that-can-handle-future-data-workloads/.

[2] “Semantic Layer – What is it?” *APOS Systems*, https://www.apos.com/content/semantic-layer-org.

[3] “Tratamiento de los datos: OLTP, OLAP, Data Warehouse.” *Evaluando Software*, 21 May 2021, https://www.evaluandosoftware.com/tratamiento-los-datos-oltp-olap-data-warehouse/.

[4] “Modelado dimensional: Esquemas de estrella.” *IBM*, https://www.ibm.com/docs/es/ida/9.1.2?topic=schemas-star.

[5] Ilarri Artigas, Sergio. “Recurso Moodle de la asignatura Almacenes y Minería de Datos.” *Almacenes de datos*.

[6] Rodriguez, Enrique. “¿Que es la “Virtualización de Datos”? | Ayse Lucus.” *Ayse Lucus |*, 9 May 2021, https://www.ayselucus.es/noticia/%C2%BFque-es-la-%E2%80%9Cvirtualizacion-de-datos%E2%80%9D.

[7] *AtScale | Semantic Layer Solution for Data & Analytics*, https://www.atscale.com/.

[8] *Kyvos | BI Acceleration Platform Powered by Smart OLAP™ | A Cloud & Big Data Analytics Company*, https://www.kyvosinsights.com/.

[9] “Balsamiq Wireframes - Industry Standard Low-Fidelity Wireframing Software.” *Balsamiq*, https://balsamiq.com/wireframes/.

[10] “MEAN.” *Wikipedia*, https://es.wikipedia.org/wiki/MEAN.

[11] “Mean Stack Development[For Developers].” *Hacker Noon*, 22 March 2017, https://hackernoon.com/mean-stack-development-for-developers-4d88c40c4103.

[12] “AngularJS.” *Wikipedia*, https://es.wikipedia.org/wiki/AngularJS.

[13] *Visual Studio Code - Code Editing. Redefined*, https://code.visualstudio.com/.

[14] “Introducción a Express/Node - Aprende sobre desarrollo web | MDN.” *MDN Web Docs*, 20 November 2021,

https://developer.mozilla.org/es/docs/Learn/Server-side/Express_Nodejs/Introduction.

[15] “MongoDB Atlas Database | Multi-Cloud Database Service.” *MongoDB*, https://www.mongodb.com/atlas/database.

[16] *Mongoose ODM v6.0.13*, https://mongoosejs.com/.

[17] “mysql.” *npm*, 23 January 2020, https://www.npmjs.com/package/mysql.

[18] Civantos, Maria. “¿Qué es una API REST? Características y usos de las APIs REST.” *Tribalyte Technologies*, 24 May 2021, https://tech.tribalyte.eu/blog-que-es-una-api-rest.

[19] “TypeScript.” *Wikipedia*, https://es.wikipedia.org/wiki/TypeScript.

[20] “HTML: Lenguaje de etiquetas de hipertexto | MDN.” *MDN Web Docs*, 19 November 2021, https://developer.mozilla.org/es/docs/Web/HTML.

[21] “CSS | MDN.” *MDN Web Docs*, https://developer.mozilla.org/es/docs/Web/CSS.

[22] “bcrypt.” *npm*, 26 February 2021, https://www.npmjs.com/package/bcrypt.

[23] “JSON Web Token.” *Wikipedia*, https://es.wikipedia.org/wiki/JSON_Web_Token.

[24] “jsonwebtoken.” *npm*, 18 March 2019, https://www.npmjs.com/package/jsonwebtoken.

[25] “What is LookML?” *Looker documentation*, https://docs.looker.com/data-modeling/learning-lookml/what-is-lookml.

[26] “Esquema en copo de nieve.” *Wikipedia*, https://es.wikipedia.org/wiki/Esquema_en_copo_de_nieve.

[27] *“Cuadrante mágico de Gartner 2021 para plataformas de Analítica y Business Intelligence.” Inforges, 5 March 2021, https://www.inforges.es/post/cuadrante-magico-de-gartner-2021-para-analitica-business-intelligence.*

[28] “Power BI.” Wikipedia, https://es.wikipedia.org/wiki/Power_BI.

[29] “Analysis Services vs Power BI Premium Model Feature Matrix.” *Insight Quest*, 10 November 2019, https://insightsquest.com/2019/11/10/analysis-services-vs-power-bi-premium-model-feature-matrix/.

[30] “What Is Data Modeling? Data Models Explained.” *Qlik*, https://www.qlik.com/us/data-modeling.

[31] “What is LookML?” *Looker documentation*, https://docs.looker.com/data-modeling/learning-lookml/what-is-lookml.

[32] “Looker, la herramienta de analítica avanzada.” *Intelligence Partner*, https://www.intelligencepartner.com/looker-analitica-avanzada/.

[33] *Dremio | SQL Lakehouse Platform for High-Performance BI*, https://www.dremio.com/.

[34] “¿Qué es Snowflake? - Paradigma.” *Paradigma Digital*, https://www.paradigmadigital.com/dev/que-es-snowflake/.

[35] *BigQuery - Ayuda de Google Cloud Platform Console*, https://support.google.com/cloud/answer/9113366?hl=es.

[36] “AWS Redshift | Herramientas de Marketing Digital | MarTech FORUM ®.” *MarTech Forum*, https://www.martechforum.com/herramienta/amazon-redshift/.

[37] “¿Qué es Azure Synapse Analytics? - Azure Synapse Analytics.” *Microsoft Docs*, 5 November 2021, https://docs.microsoft.com/es-es/azure/synapse-analytics/overview-what-is.

[38] “Analysis Services introducción a la documentación.” *Microsoft Docs*, 13 September 2021, https://docs.microsoft.com/es-es/analysis-services/analysis-services-docs?view=asallproducts-allversions.

[39] “Kyvos - Wikipedia.” - *Wikipedia*, https://en.wikipedia.org/wiki/Kyvos.

5 ANEXOS

ANEXO 1: Análisis de productos del mercado con capa semántica propia

En el apartado de la memoria motivación y problemas que se abordan se ha clasificado los diferentes productos del mercado que pueden disponer de una capa semántica propia, plataformas de business intelligence, almacenes de datos, plataformas de virtualización de datos y soluciones de capa semántica. En este anexo se van a analizar de forma más detallada dichos productos.

Plataformas de business intelligence

Estas herramientas pueden tener incorporada su propia herramienta para modelar su capa semántica. En general estas plataformas crean su propio modelo a medida, creando más de una versión de la verdad, normalmente están ligadas a "vendor lock-in", provocando que únicamente se pueda compartir el modelo entre productos de la misma firma o disponiendo una versión premium con un alto coste económico. Las principales plataformas business intelligence del mercado son Power BI (Microsoft), Tableau, Qlik y Looker (Google).

Figura 1: Cuadrante mágico de Gartner 2021 para analitica business intelligence, Imagen obtenida de [27].

Power BI es un servicio de análisis de datos de Microsoft orientado a proporcionar visualizaciones interactivas y capacidades de inteligencia empresarial con una interfaz lo suficientemente simple como para que los usuarios finales puedan crear por sí mismos sus propios informes y paneles Power BI proporciona servicios de BI basados en la nube, conocidos como "Power BI Services", junto con una interfaz basada en escritorio, denominada "Power BI Desktop" [28].

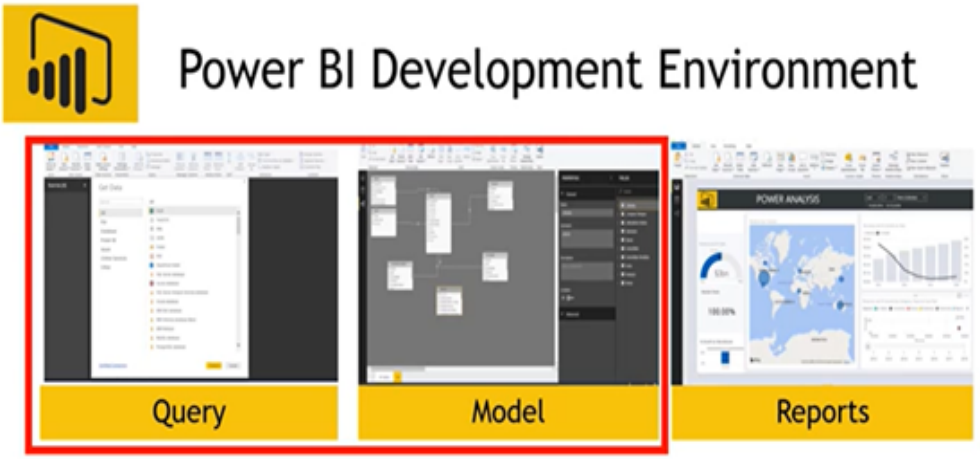


Figura 2: Representación del modelo semántico en Power BI, imagen cedida de presentación interna Hiberus- Capa semántica Microsoft.

Power BI utiliza un modelo tabular (bases de datos que se ejecutan en memoria o se conectan a datos de orígenes de datos relacionales) para almacenar su modelo de datos. "Query" y "Model" son las dos capas de Power Bi consideradas como capa semántica:

- Query: Capa encargada de la conexión, extracción y transformación de los datos en tablas.
- Model: Capa encargada de realizar el modelo de datos, definiendo las relaciones entre las diferentes tablas.

En la actualidad, los conjuntos de datos de Power BI Premium están en proceso de convertirse en la herramienta de modelado empresarial insignia de Microsoft, un superconjunto de Analysis Services, al cerrar las brechas existentes con el modelo tabular de Analysis Services e introducir nuevas funciones de modelado exclusivas de Power BI, como modelos compuestos y agregaciones [29].

Power BI es una de las mejores herramientas para el modelado y la visualización de datos. Con las mejoras de los últimos años en cuanto a capacidad de procesamiento, almacenamiento y la incorporación de funcionalidades exclusivas de modelado, lleva

camino incluso de acabar sustituyendo a Microsoft Analysis Services. Esta herramienta es una de la más utilizada en Hiberus para este tipo de soluciones y fue la referencia junto a Microsoft Analysis Services para entender el modelado semántico, aun así fue descartada desde el primer momento ya que Power BI únicamente deja compartir modelos de datos en su versión premium y además el integrar estos modelos con herramientas ajenas a Microsoft puede llegar a ser un proceso complejo.

Tableau: Por detrás de Power BI, es la segunda herramienta de modelado y análisis de datos líder del mercado, es apropiada para grandes empresas, que manejan una gran cantidad de datos (Power BI es conocido históricamente por administrar volúmenes más pequeños. Sin embargo, desde su lanzamiento, ha recorrido un largo camino y puede administrar un gran volumen de datos). Su modelo de datos tiene dos capas:

-Capa lógica: En la capa lógica se pueden combinar relaciones entre los datos de diferentes tablas usando relaciones o hilos.

-Capa física: En la capa física se puede combinar los datos entre tablas, usando uniones de filas o de columna. Cada tabla lógica contendrá al menos una tabla física.

La principal desventaja de Tableau es su coste. Además para compartir modelos de datos y publicar informes en línea, se requiere una versión de pago generalmente más costosa que la de Power BI.

Qlik: Qlik Sense es una plataforma de análisis y BI innovadora y robusta que permite a los usuarios de todos los niveles explorar datos libremente mediante selecciones interactivas y búsquedas globales. Qlik indexa automáticamente todas las relaciones en sus datos, por lo que no es necesario limpiar completamente o modelar los datos por adelantado. Con la preparación de datos inteligente y de autoservicio, los usuarios no técnicos pueden combinar, transformar y cargar visualmente datos de múltiples fuentes mediante la funcionalidad de arrastrar y soltar. La creación de perfiles de datos inteligente muestra a los usuarios las relaciones entre las tablas y cómo se asociaría, permitiendo que cada usuario cree su propio modelo de datos o use las asociaciones sugeridas para construir uno más rápido [30].

Destaca por su motor cognitivo, su arquitectura abierta y las capacidades multicloud. Como contras, la complejidad de precios, su bajo impulso de mercado y la falta de cohesión en la integración de otros productos adquiridos por la compañía [27], además compartir su modelo de datos con otras herramientas resulta casi irrealizable.

Google (Looker): Looker es una de las plataformas de visualización de datos más potentes del mercado, tiene su propio lenguaje LookML, utilizado para describir dimensiones, agregados, cálculos y relaciones de datos en una base de datos SQL [32]. Esto le hace un producto diferenciador permitiendo modelar y reutilizar su capa

semántica en diferentes informes. Además de BigQuery (Google), Looker se conecta con Redshift, Snowflake, así como con más de 50 dialectos SQL compatibles [33]. Los precios de Looker no están disponibles públicamente y ofrecen un enfoque de precios personalizados al negocio. La principal desventaja de LookML es su curva de aprendizaje, llegar a realizar modelos de datos complejos para usuarios con poca experiencia puede llegar a ser una tarea costosa.

Plataformas de virtualización de datos

Estas plataformas extraen la fuente y ubicación de los datos combinándolos y transformándolos en una única fuente de verdad virtual. Al añadir esta capa de abstracción se puede considerar como una capa semántica global desde la que se puede acceder desde diferentes herramientas y plataformas. Su principal desventaja es que no están diseñadas con el único fin de modelar una capa semántica, lo que puede ser un proceso complejo y tedioso no enfocado para usuarios comerciales, además esto puede también afectar en términos de rendimiento. Aun así cabe mencionar dos de las principales plataformas de virtualización de datos Dremio y Denodo.

Dremio: Es una plataforma de datos como servicio (DAAS) de código abierto, almacena lagos de datos SQL, que permite alto rendimiento en BI y analíticas directamente del lago de datos [34]. Posee una alta conectividad tanto en orígenes de datos como a la hora de conectarlo con herramientas externas de BI utilizando JDBC, ODBC y REST. Uno de sus principales inconvenientes ha sido su poca comunidad, ya que ha resultado difícil encontrar información que no provenga de la propia compañía. Aun así ha sido de las herramientas que más posibilidades de experimentación ha ofrecido, ya que hay disponible una versión demo, disponible unos días, con la que se ha experimentado conectándose a diferentes fuentes de datos. como MySQL, posteriormente crear métricas y diferentes agregaciones en tablas virtuales y por último conectar estos datos modelados a herramientas como Power BI, para su análisis. Dremio consta de una versión comunitaria y una versión empresarial con características más específicas como control de cargas trabajo, linaje de datos y seguridad a nivel de roles. La versión comunitaria está integrada en los servicios de Amazon Web Services y permite su uso de forma gratuita (pagando una instancia EC2 de Amazon de unos 314$ al mes), por otro lado la versión empresarial tiene un coste más elevado que no es fácil de estimar.

Denodo: Denodo proporciona una plataforma de software de integración y virtualización de datos llamada Denodo Platform , así como servicios de soporte, capacitación y consultoría. Denodo es otra de las herramientas de virtualización de datos líderes del mercado, a través de la virtualización de datos, Denodo vende su plataforma como una capa semántica global de datos que permite la integración de múltiples fuentes de datos en una sola, posteriormente modelar, transformar y dar valor semántico a sus datos y por último ofrecer estos datos modelados a múltiples

herramientas de análisis de datos. Está más enfocado a dar valor semántico a los datos que Dremio ya que permite crear relaciones entre tablas. Aun así Dremio está mejor valorado por expertos por una mayor facilidad de uso, configuración y administración. A diferencia de Dremio, no ha sido posible experimentar con Denodo, ya que aunque ofrezca una versión de prueba de 30 días hace falta desplegarlo en alguna infraestructura como Amazon Web Services, Google Cloud o Azure con sus costes añadidos.

Las plataformas de virtualización de datos están en pleno crecimiento y cada vez son más las empresas que deciden añadir esa capa de abstracción a sus datos, consiguiendo su centralización y una conectividad total. Aun así, tras compartir este tipo de plataformas en Hiberus, se decidió no centrar el análisis en estas plataformas y buscar plataformas más especializadas en ofrecer capas semánticas de datos como solución.

Almacenes de datos

Desde un primer momento este tipo de soluciones han sido descartadas como posibles soluciones para administrar capas semánticas de datos. Los modelos de negocio que necesitan una capa semántica en sus datos, generalmente conectan almacenes de datos a otros proveedores que les proporcionen estos servicios, ya que los almacenes de datos están pensados para el procesamiento de datos no para servirlos. Crear modelos de datos dentro de estos almacenes es un proceso complejo y poco dinámico ya que cualquier cambio en el modelo de datos supondría una tediosa integración. Aun así cabe mencionar los principales proveedores de almacenes de datos del mercado por su directa relación con plataformas de modelado de capas semánticas. Algunos de los principales proveedores de almacenes de datos del mercado son Snowflake, Google BigQuery, Amazon Redshift y Azure Synapse SQL Analytics.

Snowflake: Es un almacén de datos analítico en la nube (SaaS) que permite elegir con qué proveedor de servicios trabajar por debajo. La diferencia es que no está basado en las soluciones comunes de Big Data, como podría ser hadoop, sino que han generado su propio motor SQL específicamente pensando en la nube [35].

Google BigQuery: BigQuery es un almacén de datos de Google de bajo coste y totalmente administrado que permite extraer analíticas de petabytes de datos. Es autónomo, por lo que no es necesario gestionar ninguna infraestructura ni contar con un administrador de bases de datos. BigQuery te permite centrar tus esfuerzos en analizar datos para obtener información valiosa a través del conocido lenguaje SQL y del aprendizaje automático integrado [36].

Amazon Redshift: AWS Redshift es un almacén de datos rápido y completamente administrado que permite analizar todos los datos empleando de forma sencilla y

rentable SQL estándar y las herramientas de inteligencia empresarial (BI) existentes [37].

Azure Synapse SQL Analytics: Azure Synapse es un servicio de análisis empresarial que acelera el tiempo necesario para obtener información de los sistemas de almacenamientos de datos y de macrodatos. Azure Synapse reúne lo mejor de las tecnologías SQL que se usan en el almacenamiento de datos empresariales, las tecnologías de Spark que se utilizan para macrodatos, Data Explorer para análisis de serie temporal y de registro, Pipelines para la integración de datos y ETL/ELT, y la integración profunda con otros servicios de Azure, como Power BI, CosmosDB y AzureML [38].

Soluciones de capa semántica

Son soluciones especializadas en ayudar a los usuarios a servir los datos de forma simple y en terminología de negocio . Ofreciendo una versión unificada del modelo de datos, con una única versión de la verdad aportando gobernanza, seguridad y permitiendo dar soporte a más de una herramienta de visualización. Las principales soluciones del mercado son Microsoft Analysis Services, AtScale, y Kyvos.

Microsoft Analysis Services: Analysis Services proporciona modelado semántico de nivel empresarial, gobernanza, ciclo de vida y administración de datos en tres plataformas diferentes: la nube en Azure, local con SQL Server y además potencia Power BI premium [39].

La manera de cómo implementar un proyecto de modelo tabular en SQL Server y en Azure es muy similar. La principal diferencia entre estas plataformas es la forma de aprovisionamiento de los recursos. Además SQL Server permite realizar modelos multidimensionales. El modelo multidimensional es anterior al modelo tabular, este a diferencia del tabular, trabaja en disco en vez de en memoria y utilizaba MDX un lenguaje de consultas bastante tedioso. El avance de las tecnologías y la disminución del precio del almacenamiento en memoria permitió el nacimiento del modelo tabular, que utiliza índices columnares y un lenguaje de consulta menos complejo DAX.

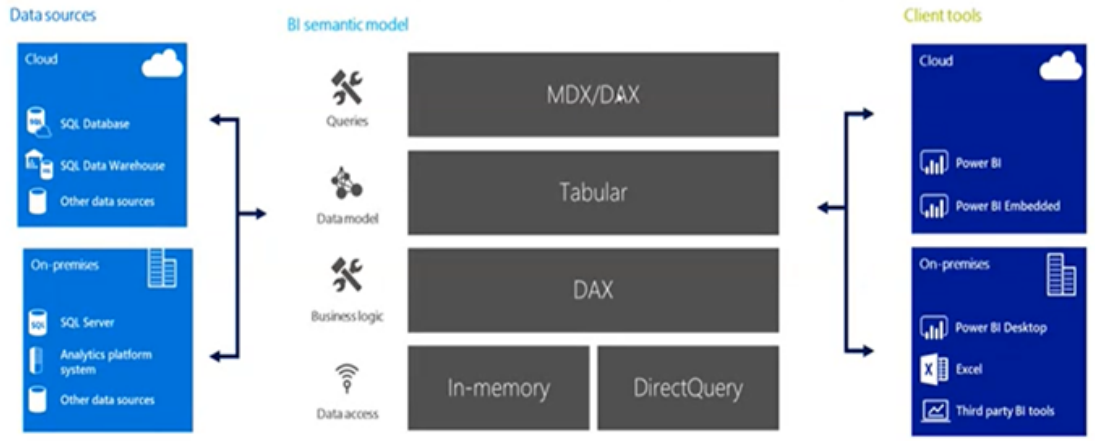


Figura 3: Representación del modelo semántico en Azure Analysis Service, imagen cedida de presentación interna Hiberus- Capa semántica Microsoft.

En la figura 3 se muestran todos los elementos que constituyen el modelo semántico tabular de Azure Analysis Services. En primer lugar las consultas a los diferentes orígenes de datos se realizan con MDX o DAX. Posteriormente se crea un modelo tabular, aplicando toda la lógica de negocio con DAX y por último se decide cómo acceder a los datos si cargarlos directamente en memoria o utilizando DirecQurey (consultando directamente a las fuentes de datos).

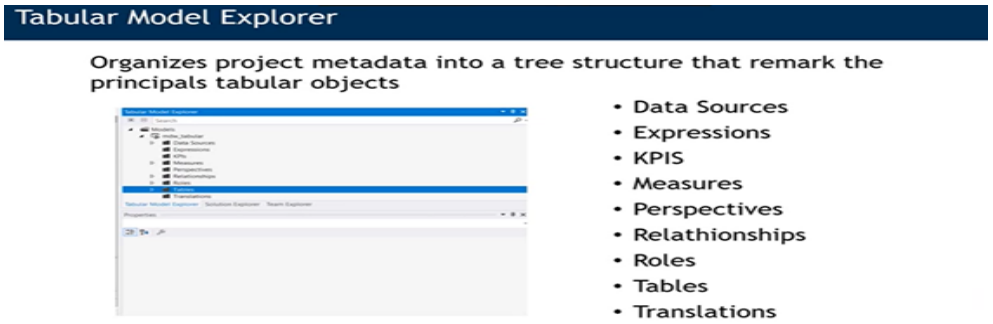


Figura 4: Organización de un modelo tabular en Microsoft Analysis Service, imagen cedida de presentación interna Hiberus- Capa semántica Microsoft.

En la figura 4 se muestra la organización de un modelo tabular en Microsoft Analysis Service:

- "Data Sources" define el origen de los datos.
- "Expressions" define las consultas con las que se accedera a esos datos.
- "KPIS" define colores y rangos de los indicadores.
- "Measures" define las medidas creadas aplicando lógica de negocio, utilizando el lenguaje DAX.

- “Perspectives” define para modelos grandes de datos, la posibilidad de ver los datos desde diferentes puntos de vista (dirección, ventas, compras...). Es la característica diferenciadora con el actual modelo de datos de Power BI, mientras que el modelo de datos de Power BI está pensado para un único punto de vista, con un único diagrama en estrella, Analysis Service puede estar pensado para todo el modelado de la empresa, con más de un diagrama en estrella.
- “Relationships” utilizado para establecer las relaciones entre tablas.
- “Roles” define roles de seguridad de acceso a los datos a nivel de fila y columna.
- “Tables” define como están estructuradas las tablas en la que se encuentran los datos.
- “Translations”: Permite definir los datos en terminología de negocio (por ejemplo cambio de nombre de la columna sales por ventas).

El compuesto de productos de Analysis Services de Microsoft es líder del mercado en plataformas dedicadas al modelado semántico de datos. Su modelo tabular ha sido referencia para comprender todas las características que debe poseer un modelo semántico. Sus principales limitaciones actuales se deben al procesamiento masivo de datos. Power BI Premium actualmente permite cargar modelos de datos de hasta 10GB, Azure Analysis Services tiene diferentes niveles de licencia, las licencias “standard” que permiten el escalado horizontal van de S0 con 10 GB memoria y $0.81/hora a S(400 GB memoria, $20.76/hora y una capacidad de procesamiento mucho mayor.

Debido a estas limitaciones de rendimiento, almacenamiento y el coste económico que conlleva se optó por analizar otras herramientas con tecnología especializada para el diseño de modelos semánticos en datos masivos como Kyvos y AtScale.

Kyvos: Es una plataforma de aceleración de inteligencia empresarial para plataformas en la nube y big data. El software proporciona análisis multidimensionales basados en OLAP en Big Data y plataformas en la nube, permite el análisis en Hadoop basado en esquemas OLAP, agregaciones y rutas de desglose predefinidas. Precalcula agregados en múltiples niveles de jerarquías dimensionales para mejorar los tiempos de respuesta a las consultas en comparación con las plataformas SQL-on-Hadoop. Los usuarios pueden analizar datos a través de la herramienta de visualización de Kyvos o utilizando otras plataformas de BI [40].

Kyvos ofrece instrucciones para desplegar su stack en amazon de forma gratuita, pero la capa gratuita que ofrece Amazon no es suficiente para desplegarlo.

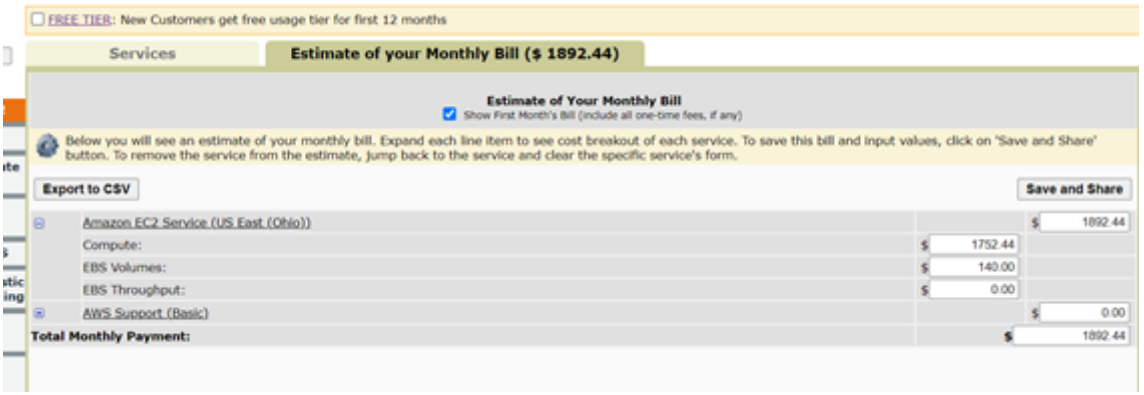


Figura 5: Estimación del coste mensual requerido en Amazon para desplegar el stack de Kyvos.

AtScale: Es una solución de capa semántica basada en una virtualización de datos inteligente que puede ayudar a tomar decisiones basadas en datos a escala y construir una cultura de autoservicio basada en datos. AtScale transforma la forma en que los consumidores comerciales acceden e interactúan con los datos empresariales a través de conexiones en vivo y una capa semántica unificada independiente del tamaño, formato o ubicación de los datos. Los analistas de negocio pueden dividir y cortar datos a escala y profundizar en los datos en segundos sin tener que depender de una tecnología específica. AtScale permite análisis “hipotéticos” multidimensionales a través de Cloud OLAP. Sus principales características son conseguir una única fuente de verdad unificada, BI de autoservicio sin vendor lock-in, ocultar las complejidades de las plataformas y estructuras de datos subyacentes y todo esto con rápido tiempo de respuesta [7] .

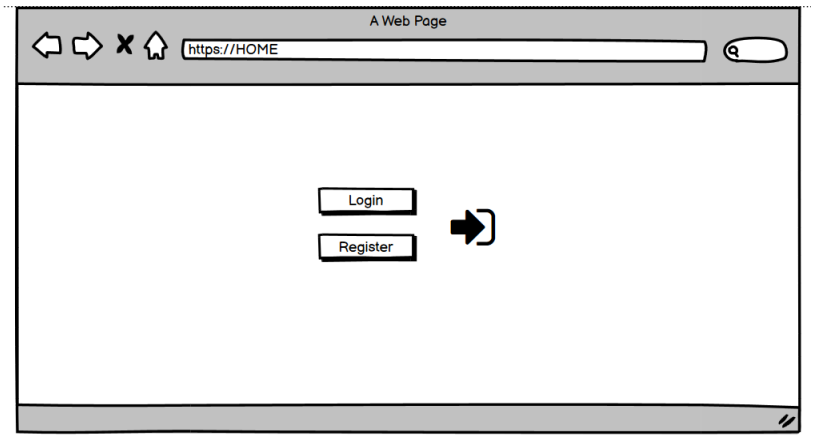
Todas estas características han sido obtenidas de su propia documentación, no se ha encontrado otra fuente que lo contraste. Además, AtScale permite solicitar una versión demo, tras varias peticiones sin respuesta se optó por no seguir analizando esta herramienta.

Como conclusión no existe una plataforma perfecta que solucione todos los problemas que pueden llegar a encontrarse a la hora de realizar un modelo de datos. Elegir plataforma va a depender de muchas características como complejidad del modelo, volumen de datos, volumen de consultas, herramientas BI que accedan al modelo, fuente de datos, presupuesto del proyecto...

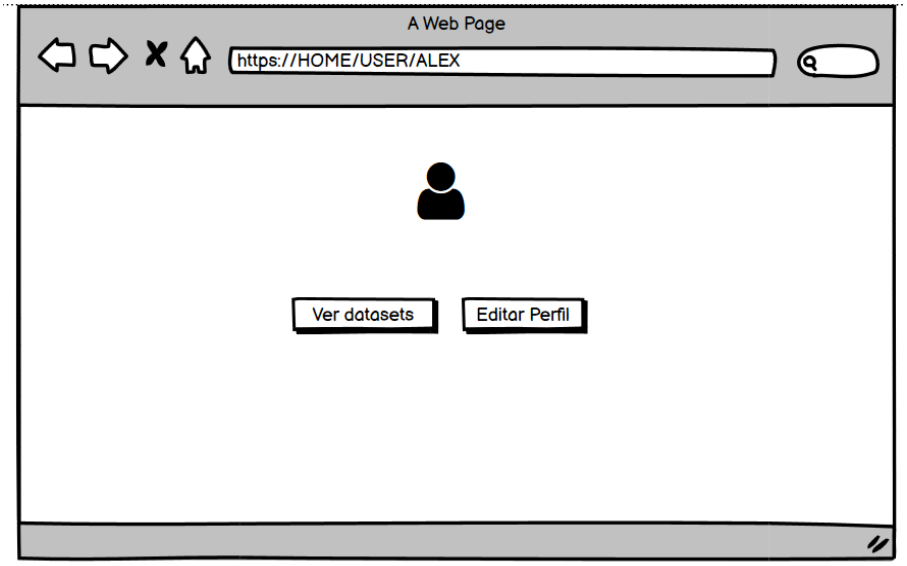
ANEXO 2: Diseño de la interfaz de usuario con Balsamiq Wireframes

En este anexo se van a presentar las diferentes pantallas del diseño de la interfaz de usuario, utilizando Balsamiq Wireframes. No se han diseñado todas las pantallas de la aplicación, solo las necesarias para mostrar el funcionamiento de la aplicación.

Pantalla de Inicio de la aplicación : El usuario puede acceder a la aplicación si ya tiene un usuario registrado o registrar uno nuevo

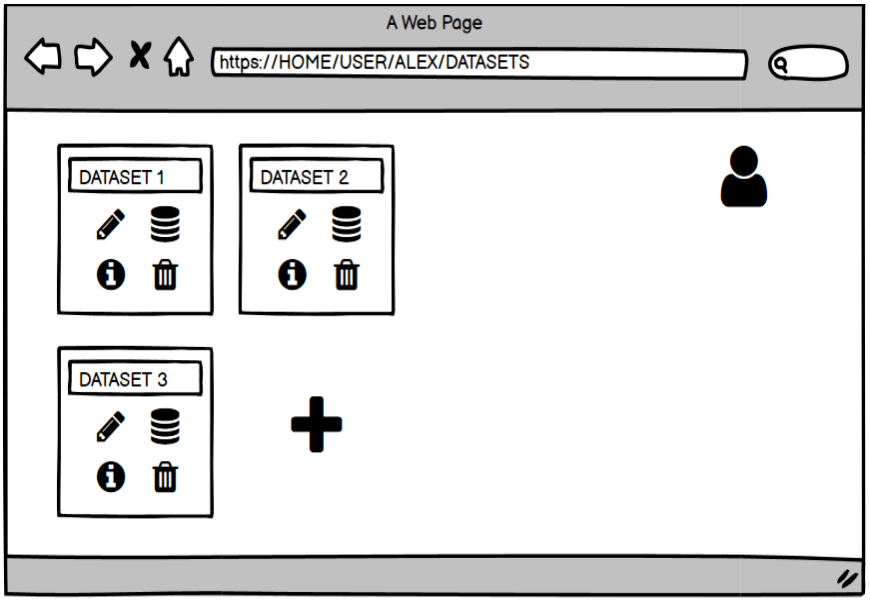


Pantalla principal: Una vez el usuario haya iniciado sesión podrá elegir entre ver sus datasets o editar su perfil.

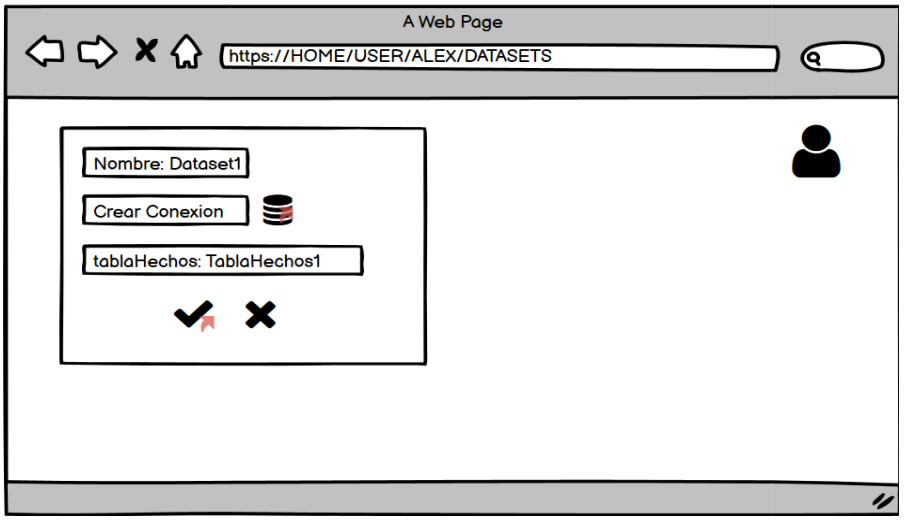


Pantalla datasets: El usuario que haya iniciado sesión podrá ver todos sus datasets ya creados. Podrá editarlos, ver la información del dataset (consulta SQL, número de filas

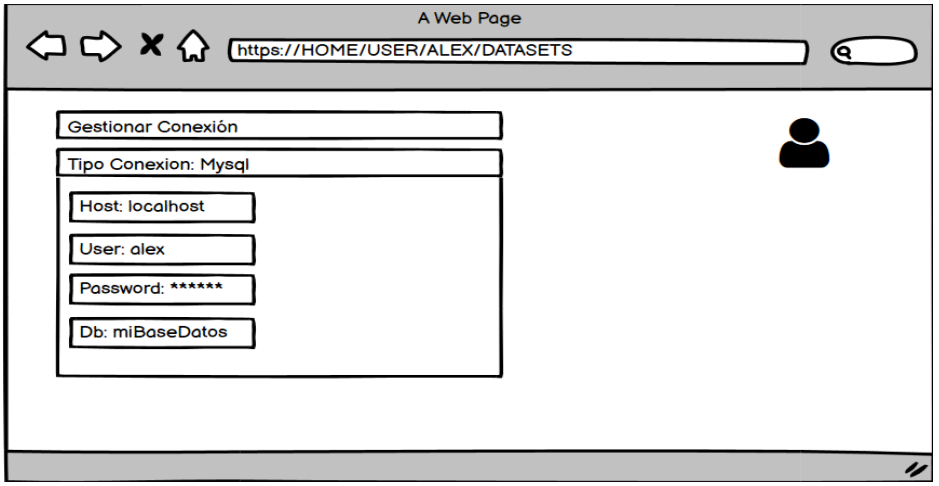
que devuelve), editar los permisos de conexión del dataset y eliminarlo. También podrá crear nuevos datasets.



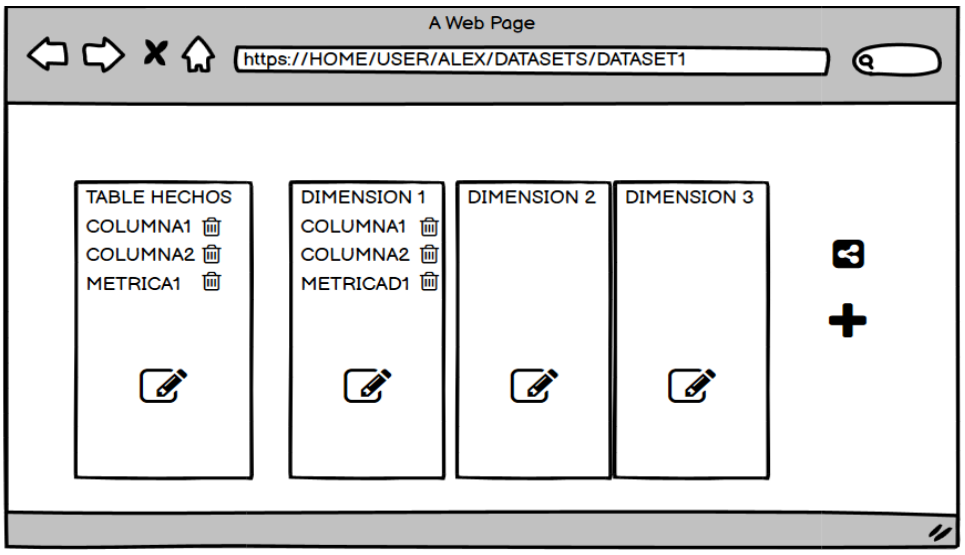
Pantalla para crear nuevos datasets: Esta pantalla es para la creación de nuevos datasets, se elegirá el nombre del dataset, se creará una nueva conexión a una base de datos y por último se selecciona cual es la tabla de hecho de la base de datos.



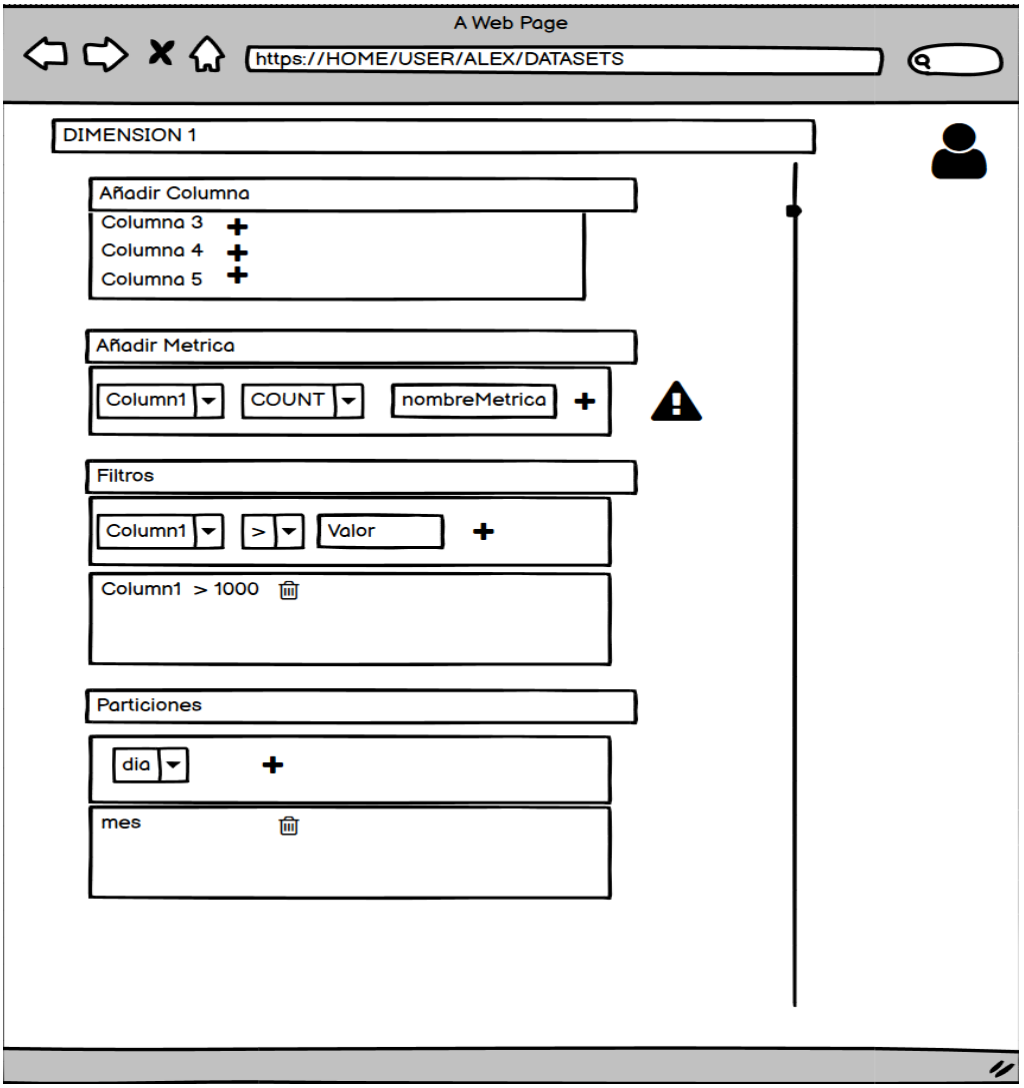
Pantalla para gestionar una nueva conexión a una base de datos: Se elegirá el tipo de conexión, en este caso Mysql y se completarán los parámetros de conexión.



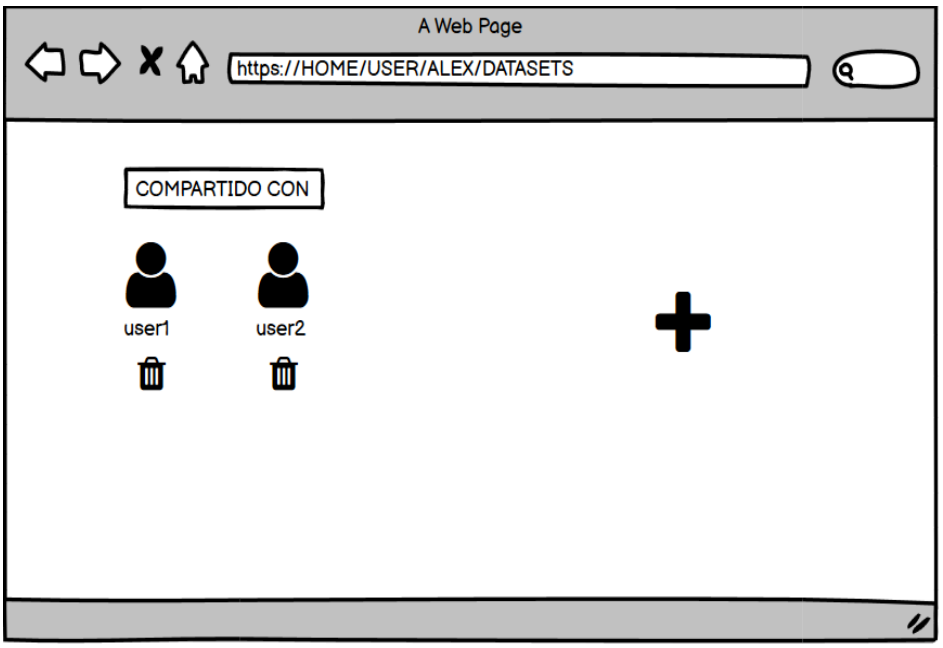
Pantalla para editar un dataset: En esta pantalla se muestra la tabla de hechos y las dimensiones que tiene definidas un dataset junto a sus columnas y métricas. Incluyendo un botón para poder editarlas, otro para agregar nuevas dimensiones y otro para editar los datos de conexión.



Pantalla para editar tabla de hechos o tablas dimensión: En esta pantalla se puede editar los datos concretos que se quieren modelar, desde agregar columnas, crear métricas, añadir filtros o agrupaciones (particiones).



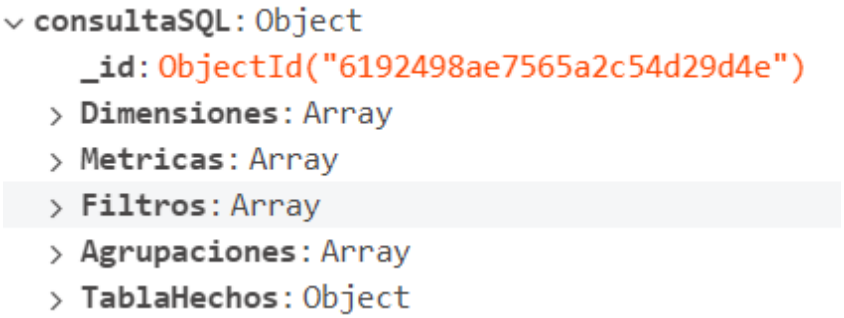
Pantalla gestionar permisos conexión de los datasets: Esta pantalla gestiona los permisos de conexión de un dataset determinado. Donde se podrá añadir y quitar usuarios (no tienen por que estar registrados) que dispongan de acceso a los datos modelados por el dataset, desde herramientas ajenas a la aplicación. Hay que destacar que en este diseño no está contemplado el añadir seguridad a nivel de fila y columna a los usuarios.



Aunque a lo largo del proyecto se ha modificado y añadido nuevas páginas, el diseño final de la aplicación y el diseño de las anteriores páginas sigue siendo bastante similar.

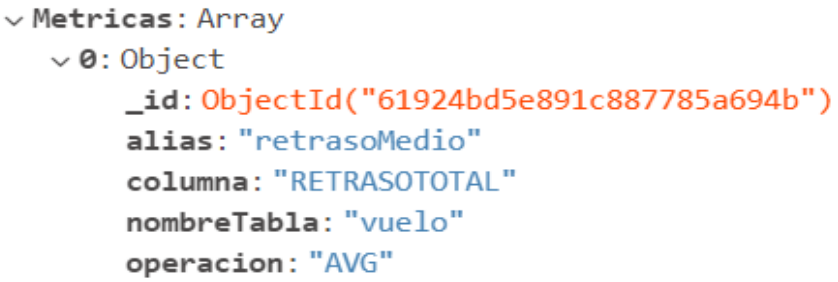
ANEXO 3: Estructura de "consultaSQL" y ejemplo de funcionamiento del traductor JSON-SQL.

Este anexo complementa la información del apartado diseño de la base de datos de la memoria del trabajo, explicando en detalle cómo funciona la estructura del objeto "consultaSQL" utilizado en los datasets con modelado gráfico, facilitando a la aplicación web tener una estructura de datos donde gráficamente sea sencillo modificar el dataset. Además es la estructura utilizada por el traductor JSON-SQL implementado, donde a partir de los datos definidos en este objeto, obtiene la consulta SQL equivalente que define el dataset.

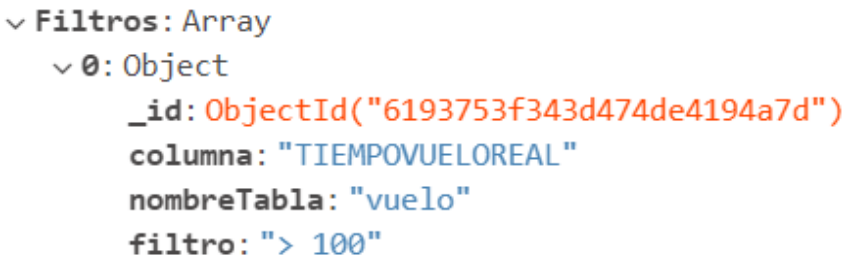


En la anterior imagen se observan los diferentes objetos que definen a “consultaSQL”.

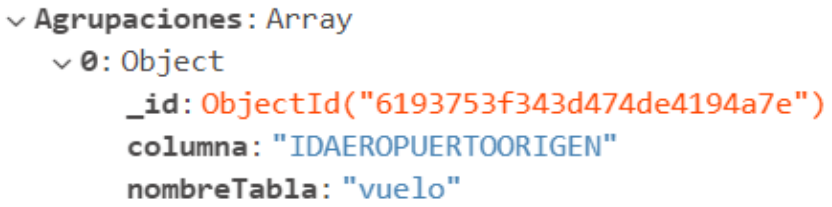
“Dimensiones” es un array de objetos que definen los campos necesarios para cada una de las tablas dimensión. Para cada tabla dimensión se almacena el nombre de la tabla, su columna que actúa como clave, la columna qué lo referencia en la tabla de hechos y un array de objetos columna que almacena las columnas de esa dimensión que serán visibles en el modelo de datos del datasets y si se ha renombrado esa columna con un “alias”.



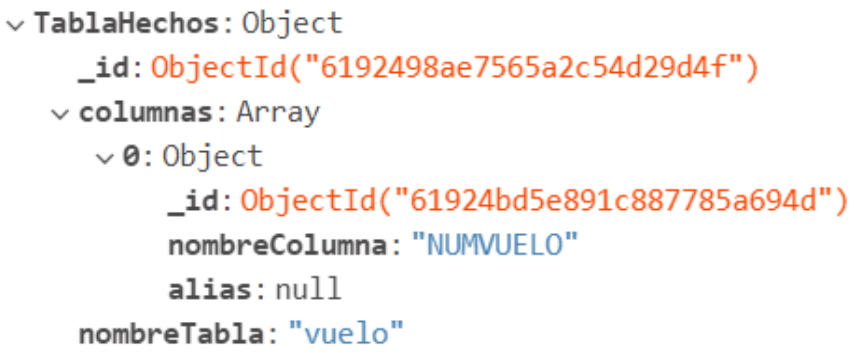
“Metricas” es un array de objetos que define cada una de las métricas incorporadas en el modelo de datos del dataset. En cada métrica se define un “alias” que corresponde al nombre con el que se define la métrica, el nombre de la tabla, su columna sobre la que se aplica y el tipo de operación que se aplica sobre esa columna y tabla (las operaciones permitidas corresponden con las operaciones permitidas por SQL a la hora de hacer agrupamientos, media, conteos, máximo ,mínimo...).



"Filtros" es un array de objetos que define cada uno de los filtros que se aplican sobre el modelo de datos del dataset. En cada filtro se define la tabla, su columna sobre la que se aplica y el filtro a aplicar que corresponde a un filtro introducido manualmente.



"Agrupaciones" es un array de objetos que define cada una de las agrupaciones que se aplicará al modelo de datos del dataset. En cada agrupación se define la tabla y su columna sobre la que aplica la agrupación.



"TablaHechos" es un objeto que define a la tabla de hechos del dataset. Se define el nombre de la tabla y un array de columnas igual al de las dimensiones donde se defina las columnas de esta tabla que se van a agregar al modelo de datos del dataset y si se ha renombrado con un "alias".

Sin el diseño de esta estructura de datos, modelar un dataset gráficamente resultaría inviable, ya que facilita mostrar y editar los componentes del modelo de datos del dataset. Para poder realizar consultas a la fuente de datos de un dataset de modelado gráfico era necesario implementar un traductor JSON-SQL para la estructura de "consultaSQL". Al ser un prototipo de aplicación web únicamente se ha implementado un traductor JSON-SQL para fuentes de datos Mysql ("traductorMysql.js").

A continuación se va a mostrar la estructura completa de un objeto “consultaSQL” de un dataset.

Este objeto “consultaSQL” definido en uno de los datasets de la aplicación, tras aplicarle el traductor JSON-SQL implementado para mySQL “traductorMysql.js”, daría como resultado el siguiente código de consulta SQL.

```
SELECT
  AVG(vuelo.RETRASOTOTAL) AS retrasoMedio, vuelo.NUMVUELO
  ,aeropuerto.ESTADO ,aeropuerto.CIUDAD AS ciudad
FROM
  vuelo INNER JOIN aeropuerto ON aeropuerto.IDAEROPUERTO =
  vuelo.IDAEROPUERTOORIGEN
WHERE
  vuelo.TIEMPOVUELOREAL > 100
GROUP BY
  vuelo.IDAEROPUERTOORIGEN ;
```

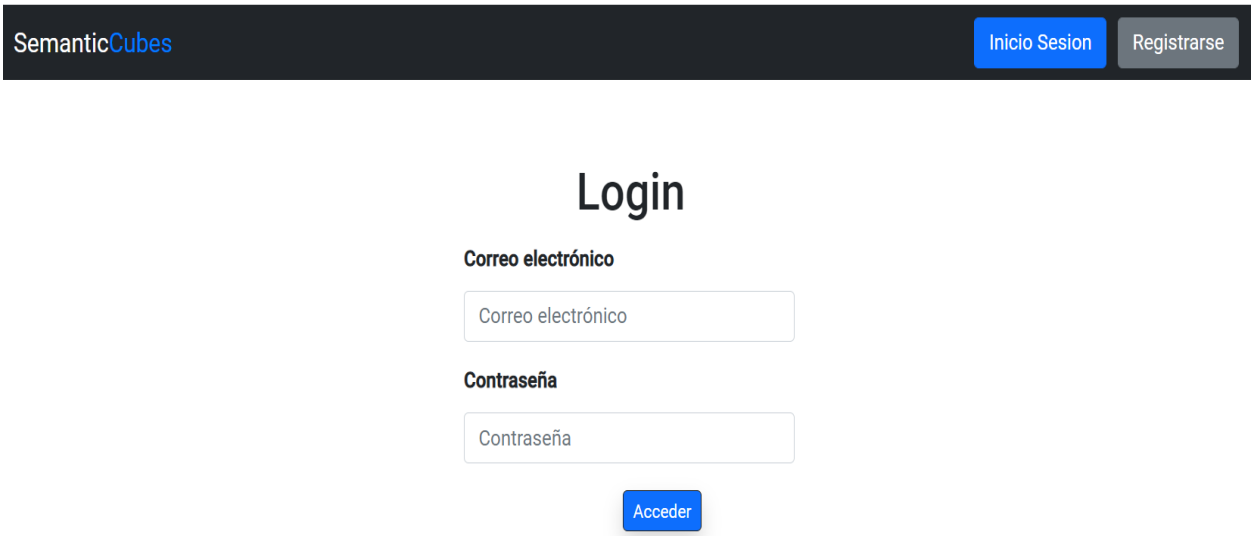
Este código SQL es el que define el modelo del dataset y es la consulta SQL que se realizará a en la fuente de datos para obtener los datos modelados.

ANEXO 4: Navegación de la aplicación web.

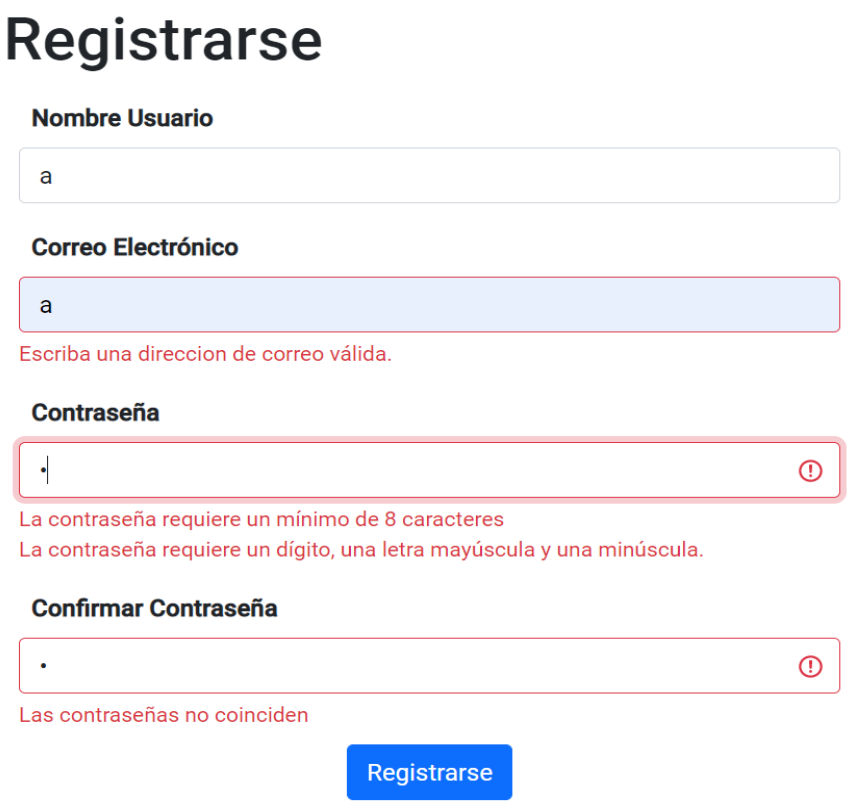
En este anexo se va a presentar una navegación completa de la aplicación web a través de sus diferentes páginas.

Se ha llamado a la aplicación web “SemanticCubes” ya que la función principal es el modelado de capas semánticas y “Cubes” por que se modela los datos en una estructura de cubos OLAP.

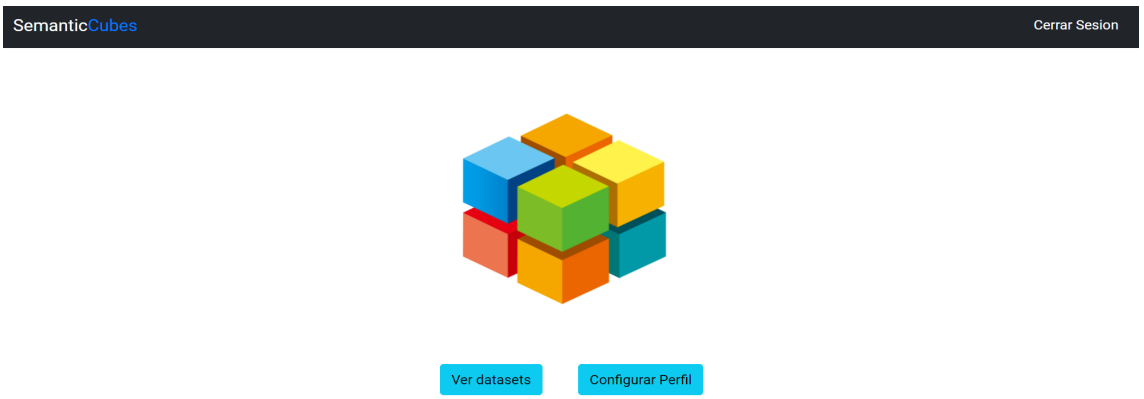
Al acceder a la aplicación web en primer lugar aparece la página de inicio de sesión.



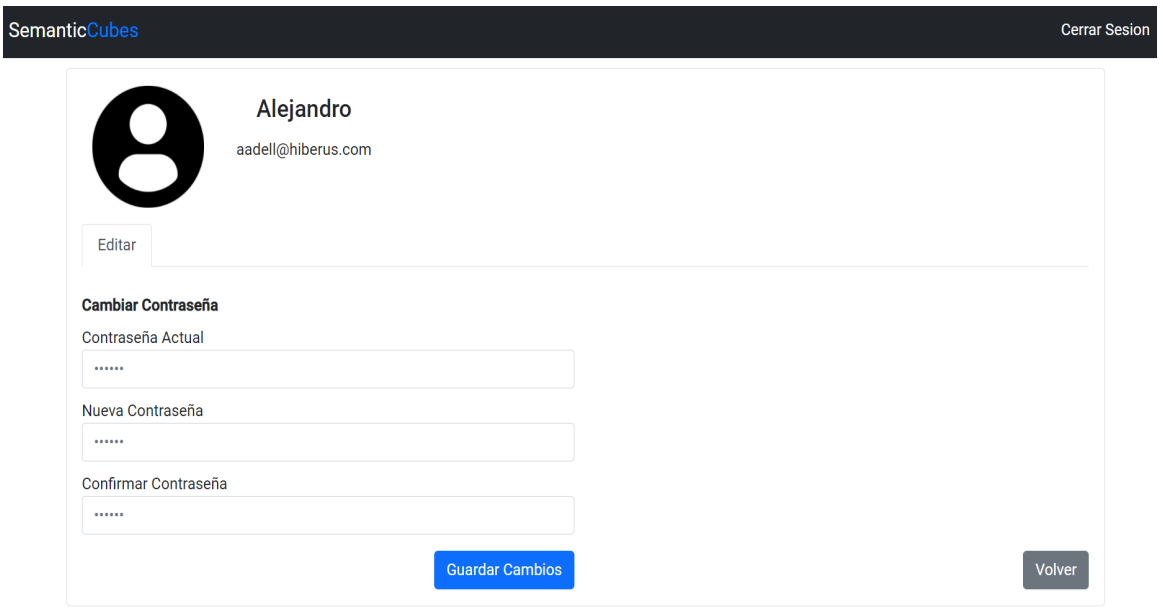
Si se pulsa el botón de arriba a la derecha que pone “Registrarse” da la opción de crear un nuevo usuario para la aplicación, rellenando un formulario.



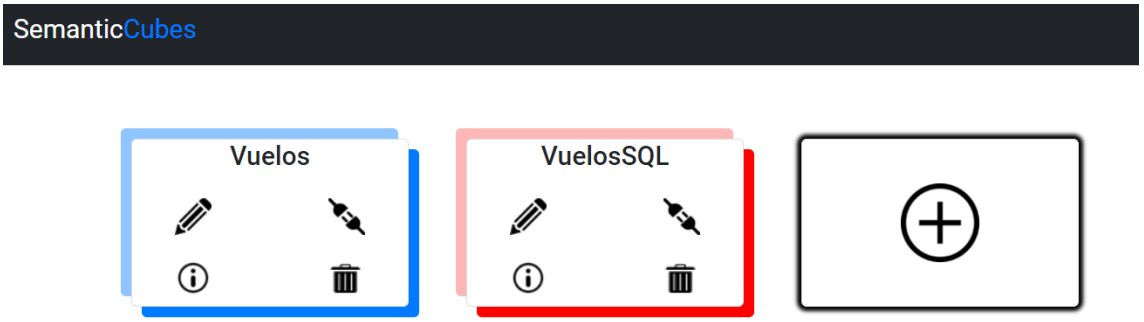
Al iniciar sesión con un correo y contraseña válido, se redirige a la página principal de la aplicación que da la opción de ver los datasets pertenecientes a un usuario o configurar su perfil.



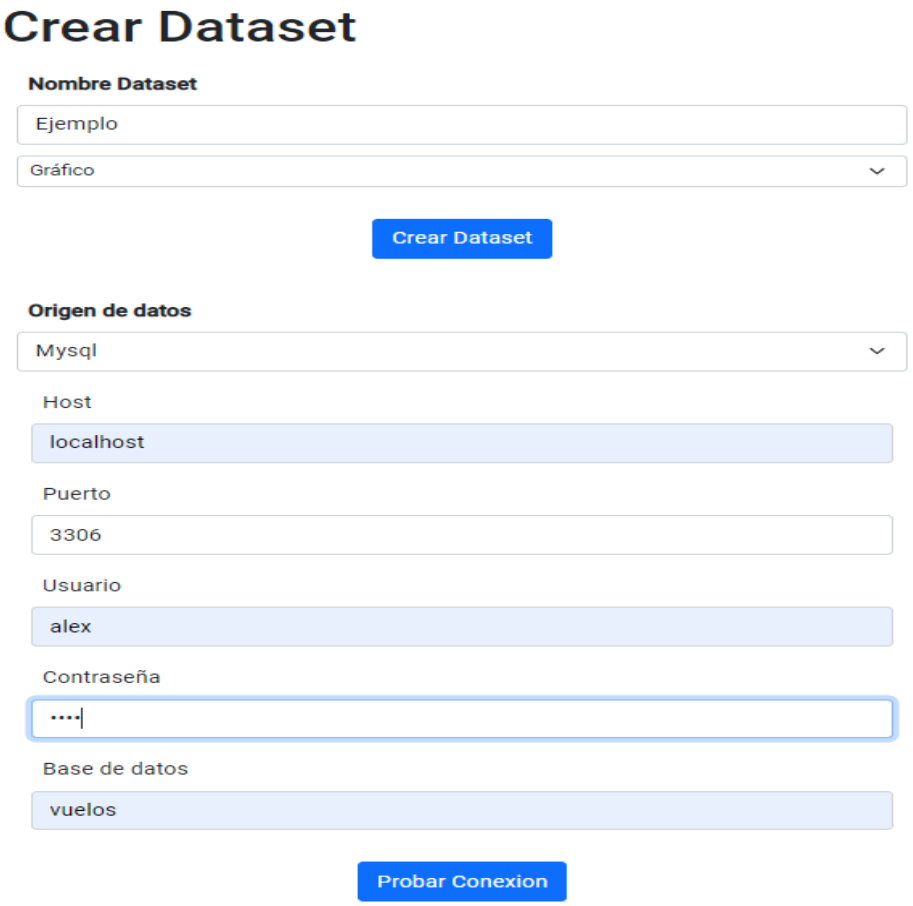
La siguiente página corresponde a la de configurar el perfil del usuario, al ser un prototipo únicamente ofrece la opción de cambiar la contraseña.



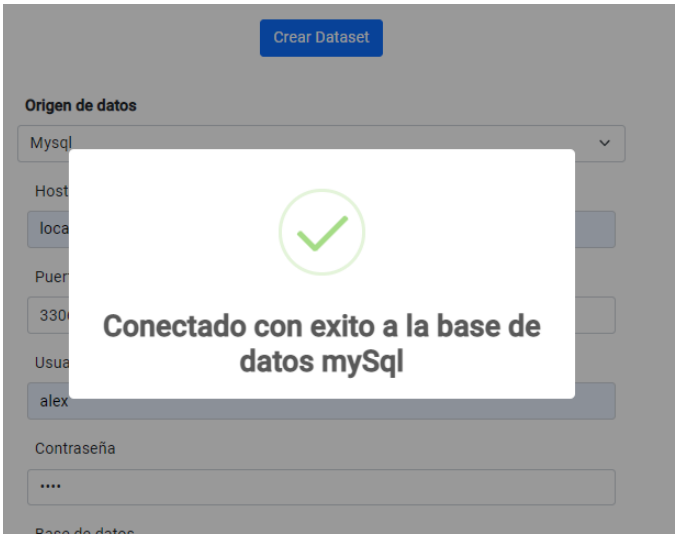
Si en la página principal ha pulsado ver datasets, el usuario podrá ver los diferentes datasets que ha modelado. Además para cada dataset podrá editarlo (símbolo lápiz), gestionar las conexiones y permisos de conexión (símbolo enchufe), ver su información (símbolo información) o eliminar completamente el dataset (símbolo basura). También existe la opción de crear un nuevo datasets (símbolo “+”).



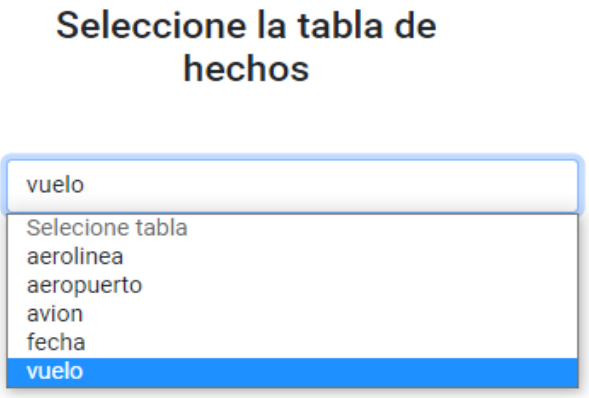
Si ha pulsado la opción de crear un nuevo dataset, tendrá que rellenar un formulario con el nombre del dataset, tipo de modelado (gráfico o código SQL), posteriormente seleccionar el origen de datos (únicamente implementado para Mysql) y rellenar sus parametros de conexion.



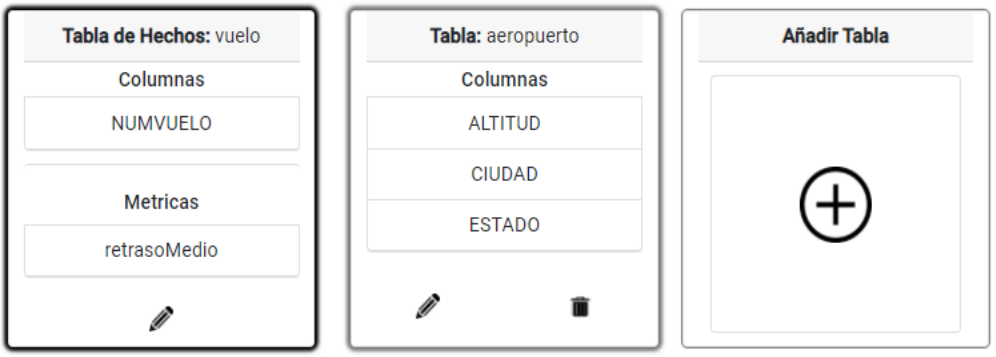
Tras probar la conexión, si se ha realizado con éxito podrá seleccionar crear dataset.



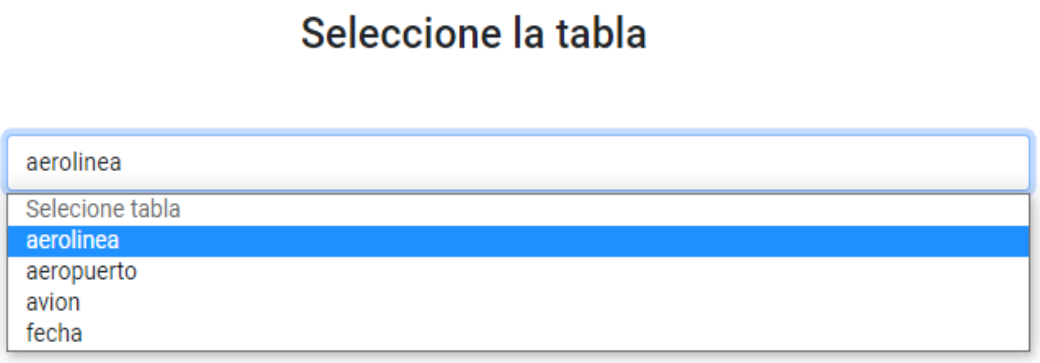
Tras crear un dataset (modelado gráfico), si selecciona la opción de editarlo al ser la primera vez que accede tendrá que decidir cuál es la tabla de hechos (a través de la API del servidor web, con los datos de conexión del dataset, se conectara a la fuente de datos para mostrar las tablas disponibles).



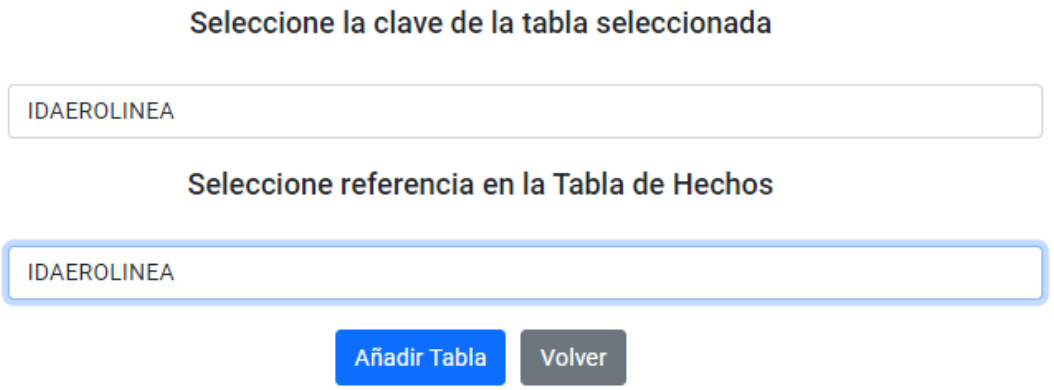
Una vez seleccionada la tabla de hechos ya aparecerá la página por defecto al pulsar el botón de editar. Donde se muestra la tabla de hechos y dimensiones, con sus columnas y métricas seleccionadas para el dataset. Se podrá editar la información del modelo (columnas, métricas, filtros, agregaciones) de la tabla de hechos y las tablas dimensión. Además se podrá agregar y eliminar nuevas tablas al dataset.



Si selecciona añadir tabla (al igual que en seleccionar la tabla de hechos, realizará la petición al servidor web para ver obtener la tablas disponibles de la fuente de datos) permitirá seleccionar las opciones disponibles .



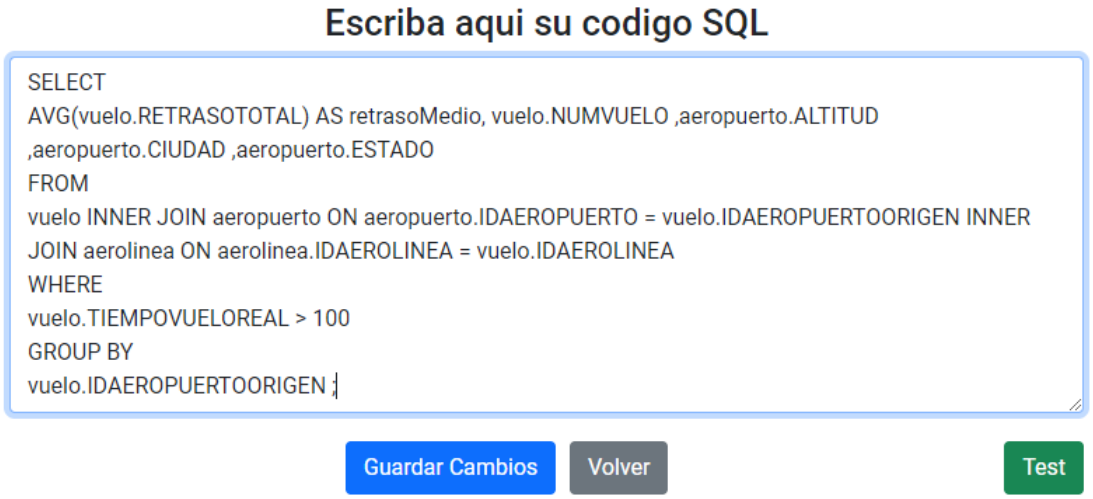
Tras seleccionar la tabla, habrá que seleccionar con qué campo de la tabla de hechos se relaciona la nueva tabla (realizará otra petición al servidor para obtener los campos de la tabla de hechos y de la tabla a añadir).



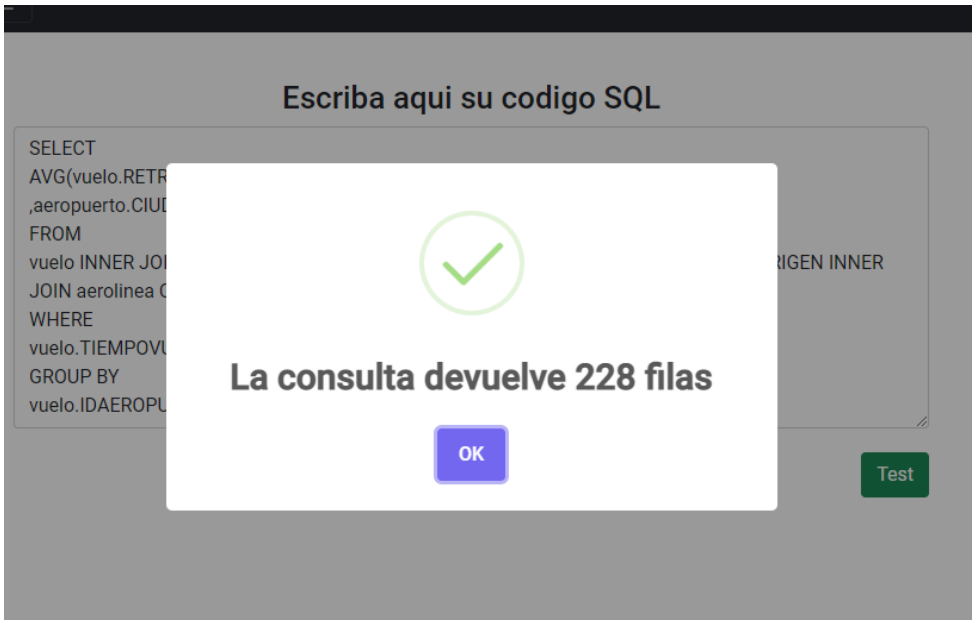
Al editar la tabla de hechos o cualquiera de las tablas dimensión, el editor es el mismo. Se podrá eliminar o añadir nuevas columnas de la tabla al dataset (con la posibilidad de

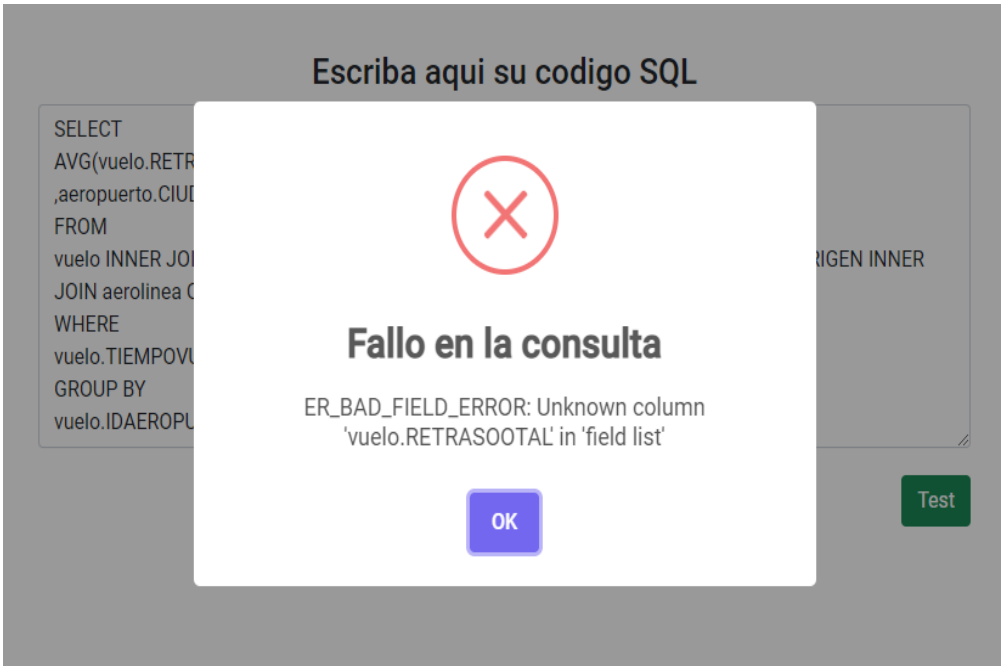
renombrarlas), agregar nuevas métricas (por ejemplo obtener la media del tiempo de vuelo), agregar o quitar filtros a columnas de la tabla y agregar o quitar agrupaciones.

Si el dataset no es del tipo modelado gráfico sino que es del tipo código SQL, al seleccionar la opción de editar el dataset únicamente le saldrá la opción de escribir directamente el código SQL.



Además tendrá la opción de testear la consulta antes de guardarla, recibiendo feedback del número de filas o el error que devuelve.





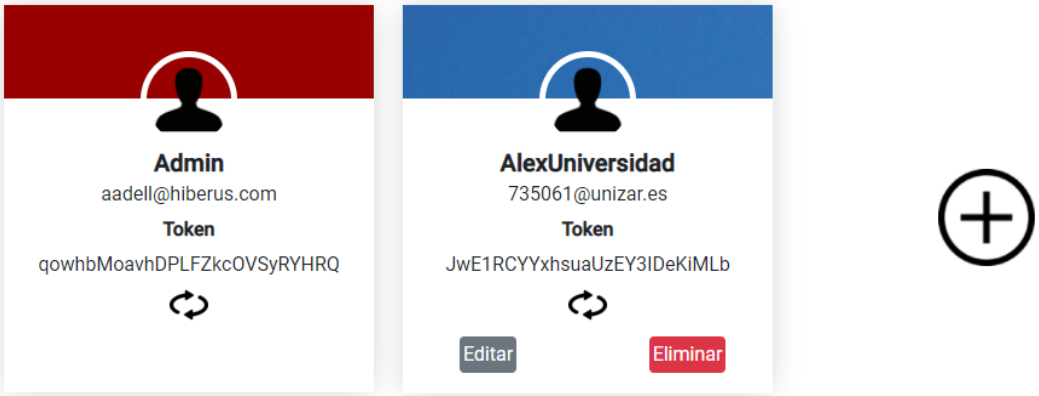
Si se selecciona ver la información (icono i) de uno de los datasets entre otra información, podrá ver la consulta SQL que utiliza el dataset (si es el caso de un dataset con modelado gráfico verá la consulta generada por el traductor JSON-SQL) y el número de filas que devuelve el modelo del dataset al realizar la consulta a la fuente de datos.

Nombre Dataset	Propietario	Base de datos	nº filas
Vuelos	aadell@hiberus.com	mySql	228

Consulta SQL:

```
SELECT
  AVG(vuelo.RETRASOTOTAL) AS retrasoMedio, vuelo.NUMVUELO ,aeropuerto.ALTITUD ,aeropuerto.CIUDAD ,aeropuerto.ESTADO
FROM
  vuelo INNER JOIN aeropuerto ON aeropuerto.IDAEROPUERTO = vuelo.IDAEROPUERTOORIGEN INNER JOIN aerolinea ON
  aerolinea.IDAEROLINEA = vuelo.IDAEROLINEA
WHERE
  vuelo.TIEMPOVUELOREAL > 100
GROUP BY
  vuelo.IDAEROPUERTOORIGEN ;
```

Si selecciona el icono de conexión de uno de los datasets podrá ver los perfiles de permisos de conexión definidos para ese dataset (nombre, correo y token). Además podrá crear nuevos perfiles, generar otro token único aleatorio para ese perfil o borrar perfiles existentes.



Si se genera un nuevo token de acceso a los datos automáticamente se enviará un mensaje al correo de referencia como que el token ha sido cambiado.


Cambio de token

A aadell@hiberus.com(aadell@hiberus.com vía hiberus.com)
Para Alejandro Adell Pina

El remitente real de este mensaje es diferente que el remitente normal. Haga clic aquí para obtener más información.
Haga clic aquí para descargar imágenes. Para ayudarle a proteger su confidencialidad, Outlook ha impedido la descarga automática de algunas imágenes en este mensaje.

Buenas, este correo es para informarle que ha sido modificado el token de uno de los dataset que tiene acceso.

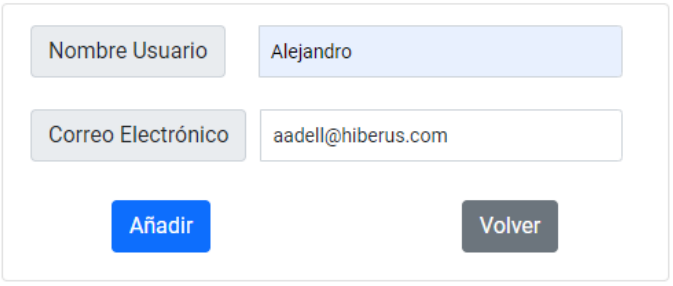
Dataset :

Ejemplo

Nuevo token:

Xh8FEUVoo7bU5rOESOiKLk34kK

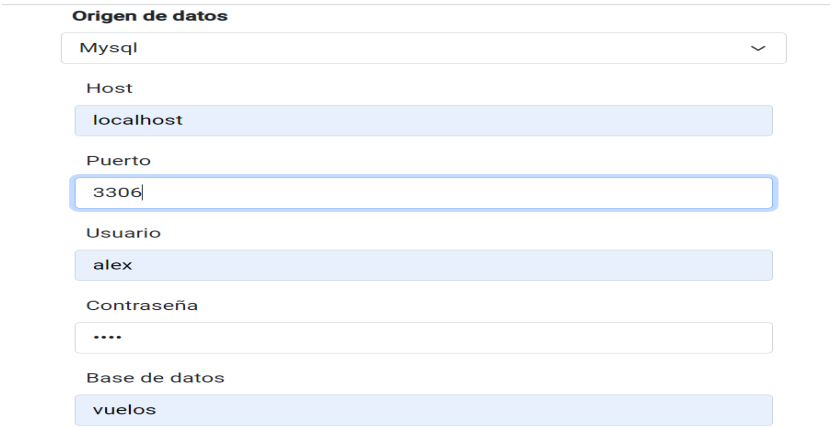
Si se selecciona la opción de añadir nuevos permisos al dataset únicamente tendrá que rellenar un formulario con un correo electrónico que no corresponda a otro perfil de acceso definido en ese dataset y un nombre de usuario que no tiene por que corresponder a un usuario registrado en la aplicación.



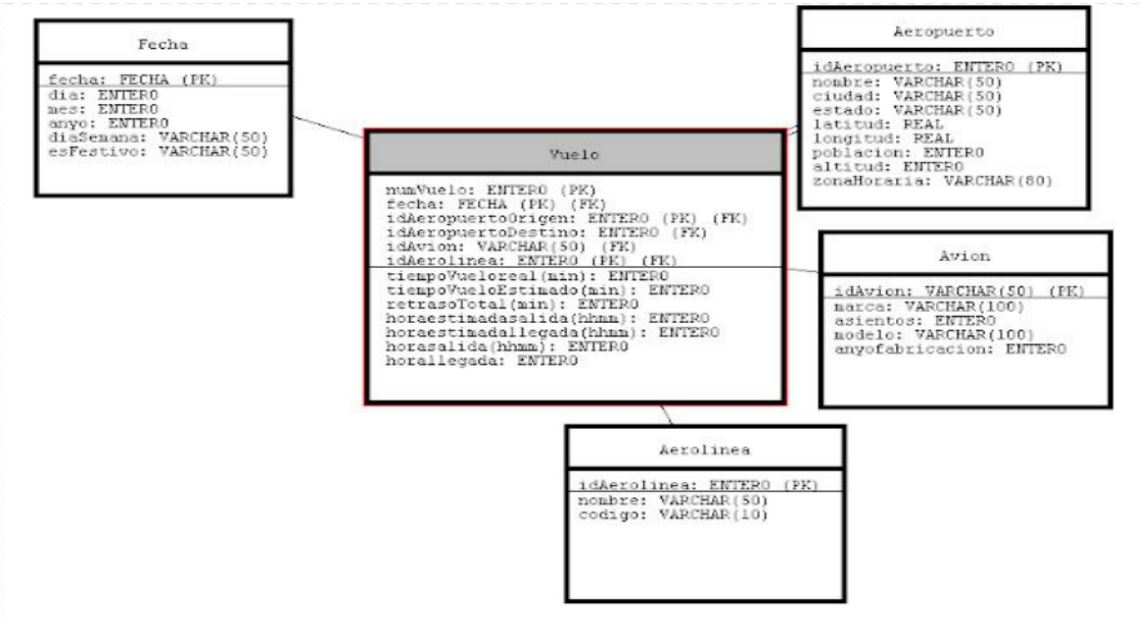
Si se selecciona la opción de editar los permisos de un perfil de acceso del dataset podrá definir la seguridad a nivel de acceso de fila o columna al dataset para ese perfil.

ANEXO 5: Ejemplo de Acceso a los datos modelados de un dataset desde la herramienta externa Power BI

En este anexo se va a mostrar como acceder a los datos modelados por un dataset desde una herramienta externa al servidor web como Power BI. En primer lugar se ha creado un dataset con modelado gráfico y conectado con una base de datos Mysql.



Se ha creado y poblado una base de datos de prueba en Mysql llamada “vuelos”. Esta base de datos almacena información de los vuelos de Estados Unidos de enero de 2020. Se ha reutilizado un proceso ETL para estos datos creado en Knime en una práctica de la asignatura “Almacenes y Minería de datos”. En la siguiente imagen se muestra el diagrama en estrella que define cómo están almacenados los datos.



Desde la aplicación web se ha modelado gráficamente el dataset, seleccionando como tabla de hechos “vuelo” y agregando como tablas dimension “aeropuerto” y “avion”. Además se han añadido agrupaciones por idAeropuerto, marca y modelo de avión.

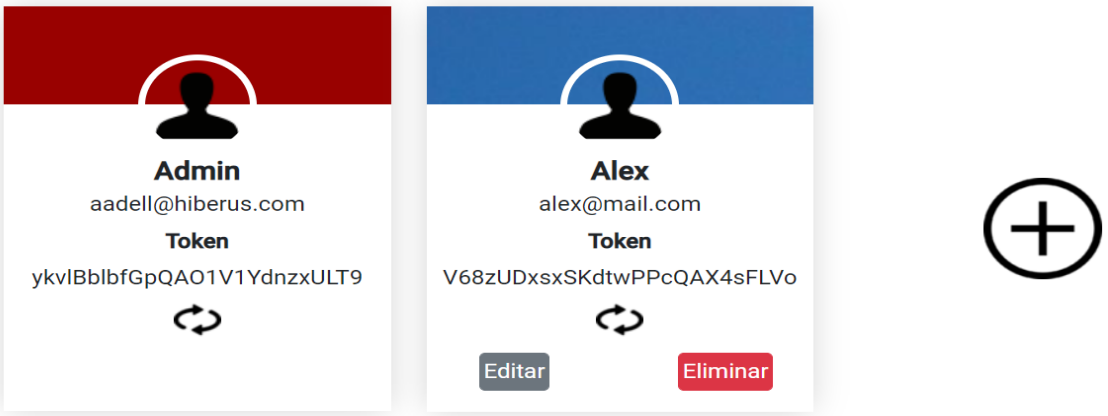
Desde la página de información del dataset, se puede ver la consulta SQL generada y el número de filas que devuelve al realizar la consulta en la base de datos Mysql.

Nombre Dataset	Propietario	Base de datos	nº filas
Vuelos	aadell@hiberus.com	mySql	5893

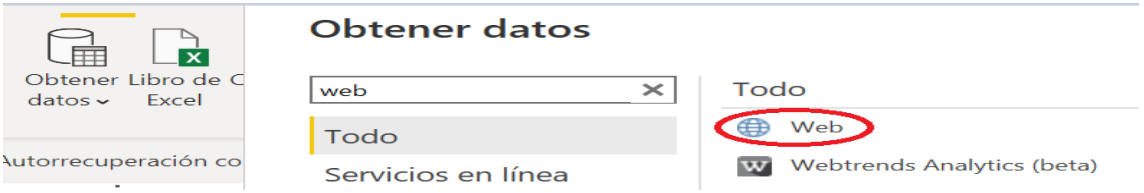
Consulta SQL:

```
SELECT
  AVG(vuelo.RETRASOTOTAL) AS retrasoMedio, vuelo.NUMVUELO
  ,aeropuerto.ESTADO ,aeropuerto.CIUDAD AS ciudad ,avion.MARCA ,avion.MODELO
FROM
  vuelo INNER JOIN aeropuerto ON aeropuerto.IDAEROPUERTO =
  vuelo.IDAEROPUERTOORIGEN INNER JOIN avion ON avion.matriculaAvion =
  vuelo.matriculaAvion
GROUP BY
  aeropuerto.IDAEROPUERTO , avion.MARCA , avion.MODELO ;
```

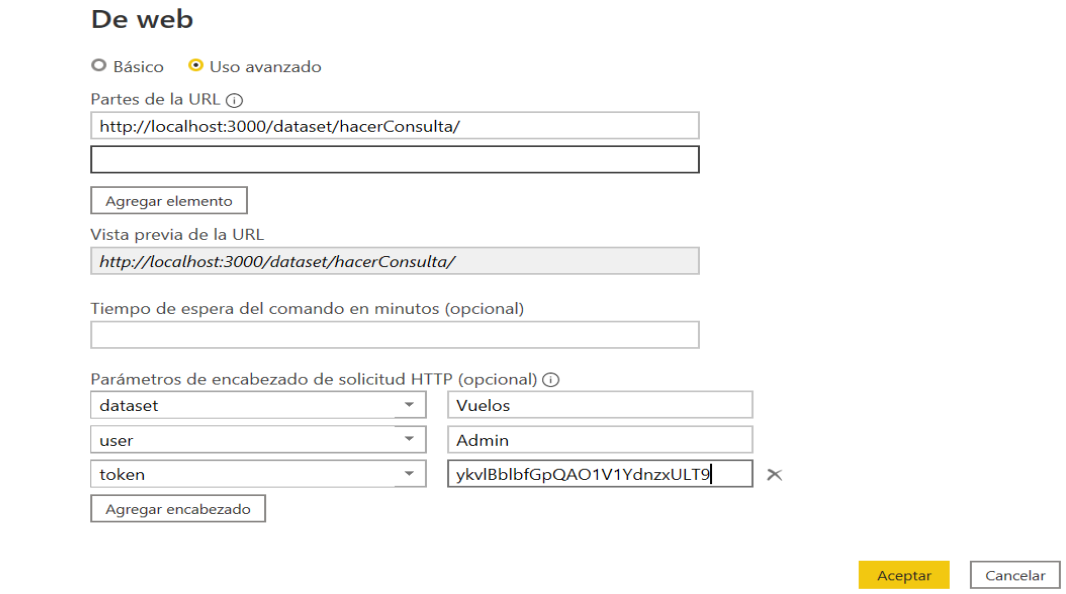
En la página de conexiones de ese dataset se ha definido un nuevo acceso a usuario “Alex”, se le ha definido reglas de seguridad a nivel de columna para restringir la columna “NUMVUELO” y a nivel de fila se le ha restringido que solo pueda ver las filas con “ESTADO”=’California’.



A continuación vamos a utilizar la herramienta Power BI Desktop para acceder a los datos modelados por el dataset “Vuelos”. Para ello en “Obtener datos'' se selecciona el tipo Web para realizar una petición HTTP REST al API del servidor web.



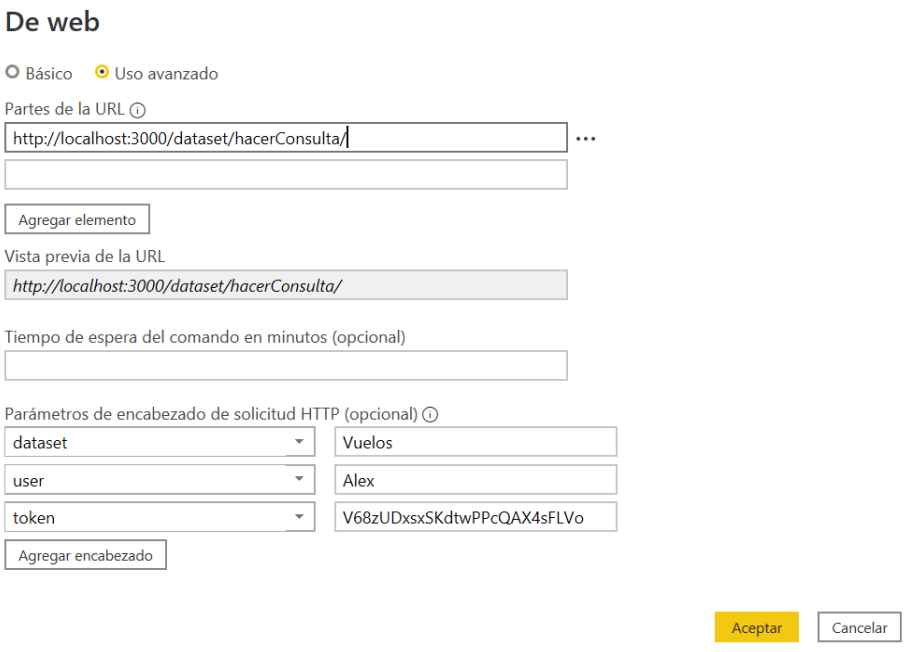
Para conectarse a los datos se utiliza la URL de la API del servidor web *“http://localhost:3000/dataset/hacerConsulta/”* para realizar la consulta. Y los parametros “dataset” (corresponde con el nombre del dataset a consultar, Vuelos), “user” (corresponde con el acceso a usuario definido en el dataset, en esta caso va a ser desde “Admin”) y token (corresponde con el token asignado para ese acceso a usuario).



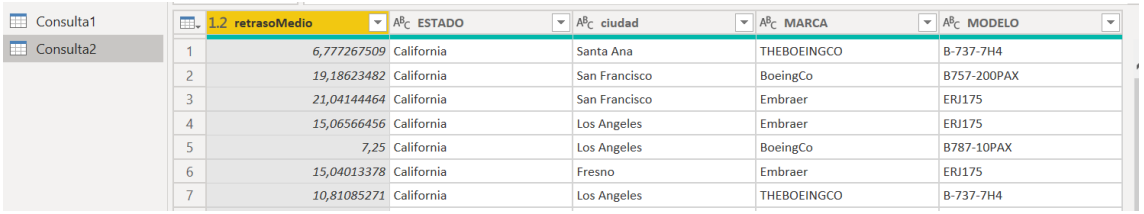
Al aceptar se obtienen los datos modelados por el dataset de la fuente de datos.

	1.2 retrasoMedio	123 NUMVUELO	ABC ESTADO	ABC ciudad	ABC MARCA	ABC MODELO
1	5,378787879	4815	Kentucky	Louisville	THEBOEINGCO	B-737-7H4
2	6,777267509	6423	California	Santa Ana	THEBOEINGCO	B-737-7H4
3	10,9719888	3391	Oregon	Portland	Embraer	ERJ175
4	19,1875	3481	Washington	Spokane	Embraer	ERJ175
5	21,44363929	3492	Washington	Seattle	Embraer	ERJ175
6	21,11569507	5528	Colorado	Denver	Embraer	ERJ175

Se va a realizar otra consulta al mismo dataset desde el acceso definido para el usuario “Alex”.



Se observa en los datos obtenidos que para el mismo dataset, conectándose con otro usuario se obtienen datos diferentes. Esto se debe a las reglas de seguridad a nivel de fila y columna definidas para este usuario. El acceso del usuario “Alex” a nivel de columna tiene restringida la columna “NUMVUELO” y a nivel de fila tiene restringido para únicamente acceder a las filas que “ESTADO” = ‘California’. Como se observa la columna “NUMVUELO” ha sido eliminada del resultado de la consulta y únicamente se muestran las filas con “ESTADO” = ‘California’.


Consulta1
Consulta2

	1.2 retrasoMedio	ABC ESTADO	ABC ciudad	ABC MARCA	ABC MODELO
1	6,777267509	California	Santa Ana	THEBOEINGCO	B-737-7H4
2	19,18623482	California	San Francisco	BoeingCo	B757-200PAX
3	21,04144464	California	San Francisco	Embraer	ERJ175
4	15,06566456	California	Los Angeles	Embraer	ERJ175
5	7,25	California	Los Angeles	BoeingCo	B787-10PAX
6	15,04013378	California	Fresno	Embraer	ERJ175
7	10,81085271	California	Los Angeles	THEBOEINGCO	B-737-7H4