LABORATORIO

Diseño de una base de datos relacional

CURSO AVANZADO DE INGENIERÍA DE SOFTWARE
Alumno: Javier Nicolás Pérez Mesa

Índice

Diseño de la base de datos

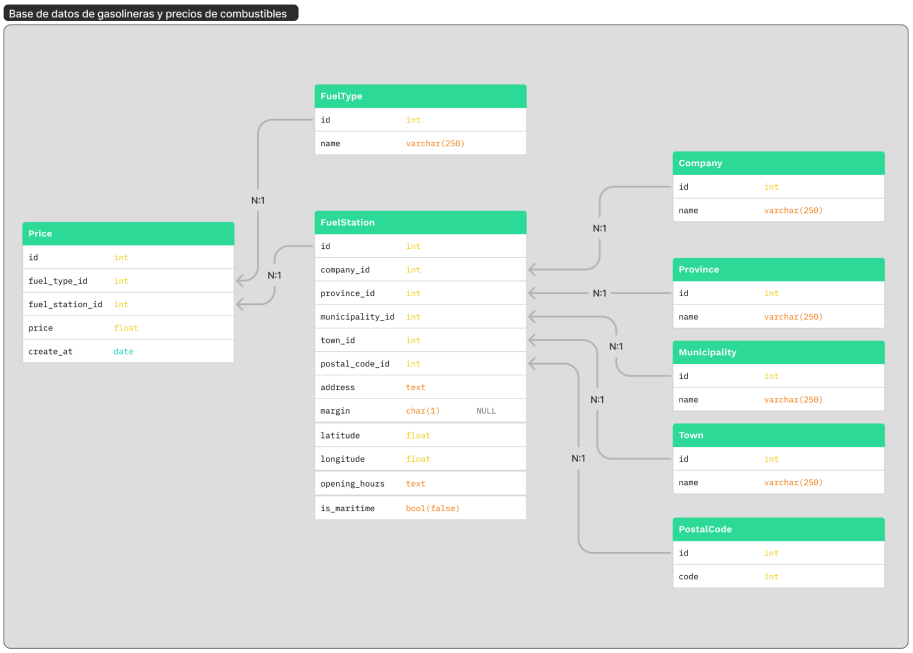
Análisis

Al analizar los datos se buscó en un primer momento encontrar las diferencias y los parecidos entre los dos documentos de los que vamos a obtener los datos, en esta primera revisión se vieron los siguientes puntos:

- Ambos documentos tenían los datos de localización (Provincia, Municipio, Localidad, Código postal y dirección) en las mismas ubicaciones y con el mismo tipo de datos.
- Los precios los tiene en diferentes ubicaciones y relacionados por columnas en las que se establece el precio por cada combustible, además los precios están formateados de forma diferente, pero revisando los datos veo que el máximo de caracteres enteros va a ser 4 y en el caso de decimales he limitado a 3 decimales, que me parece un nivel de precisión suficiente para este ejercicio, esto se tendrá que unificar.
- Las coordenadas GPS tienen longitudes diferentes entre latitud y longitud y cambia además entre los documentos, para evitar problemas he visto que el máximo de caracteres en este caso es de 3 enteros y 6 decimales.
- En los casos de las coordenadas GPS y de los precios están formateados con comas, lo que dificultará la conversión a CSV.
- En el caso de las estaciones de servicio marítimo no existe ni fecha de la toma de los datos ni el margen, en el caso de la fecha de la toma, usaremos la fecha de descarga del documento que se ve en la primera fila y se usará como por defecto si se encuentra algún registro sin este dato. En el caso del margen será suficiente establecerlo como N para indicar que no aplica, de esta forma será todo más uniforme, pero esto se hará durante la ingesta.
- Para unificar los tipos de combustible y teniendo en cuenta que son encabezados de columnas, el mejor camino será crear una tabla de excel donde poder mostrar todos los tipos de combustible disponibles, ya que son en gran medida compartidos entre ambos documentos.

Diseño

En base a los datos obtenidos y el análisis realizado sobre los mismos se llegó a la conclusión de que se deberían crear al menos 8 tablas/entidades para almacenar de una forma correcta los datos mínimos requeridos por las especificaciones del ejercicio. Para poder realizar el diseño a falta de conocer una herramienta más óptima, se ha utilizado Figma que permite diseñar diagramas de forma relativamente sencilla.



El diagrama entidad-relación mostrado en la imagen representa la estructura de una base de datos diseñada para almacenar información sobre estaciones de servicio y precios de combustibles de los datos recogidos en el ministerio tanto de estaciones marítimas como terrestres. El diagrama se compone de varias entidades (tablas) que están interrelacionadas entre sí. A continuación se proporciona una explicación concisa de cada entidad y sus relaciones:

Entidades

FuelType

Es la tabla encargada de almacenar los datos sobre los tipos de combustible, solo almacena el nombre asociado con su identificador único. Aislar está información requiere la creación de un CSV especial para facilitar el proceso de ingesta, lo cual no es un problema al ser pocos datos. La relación N:1 con la tabla de precios (*Price)* indica que varios precios pueden corresponder a un mismo tipo de combustible.

Entidad	Descripción
id	Identificador único de cada registro de la tabla
name	Nombre del combustible

Province

Es la tabla encargada de almacenar el nombre de la provincia. Su objetivo es facilitarnos el filtrado por este campo en la tabla de estaciones de servicio (*FuelStation*). Tiene una relación N:1 con la tabla *FuelStation*, lo que refleja que pueden estar vinculadas a varias estaciones de servicio.

Entidad	Descripción
id	Identificador único de cada registro de la tabla
name	Nombre de la provincia

Municipality

Es la tabla encargada de almacenar el nombre del municipio. Su objetivo es facilitarnos el filtrado por este campo en la tabla de estaciones de servicio (*FuelStation*). Tiene una relación N:1 con la tabla *FuelStation*, lo que refleja que pueden estar vinculadas a varias estaciones de servicio.

Entidad	Descripción
id	Identificador único de cada registro de la tabla
name	Nombre del municipio

Town

Es la tabla encargada de almacenar el nombre de las localidades. Su objetivo es facilitarnos el filtrado por este campo en la tabla de estaciones de servicio (*FuelStation*). Tiene una relación N:1 con la tabla *FuelStation*, lo que refleja que pueden estar vinculadas a varias estaciones de servicio.

Entidad	Descripción
id	Identificador único de cada registro de la tabla
name	Nombre de la población

PostalCode

Es la tabla encargada de almacenar los códigos postales. Su objetivo es facilitarnos el filtrado por este campo en la tabla de estaciones de servicio (*FuelStation*). Tiene una relación N:1 con la tabla *FuelStation*, lo que refleja que pueden estar vinculadas a varias estaciones de servicio.

Entidad	Descripción
id	Identificador único de cada registro de la tabla
code	Código postal

Company

Es la tabla encargada de almacenar los nombres de las empresas. Su objetivo es facilitarnos el filtrado por este campo en la tabla de estaciones de servicio (*FuelStation*). Tiene una relación N:1 con la tabla *FuelStation*, lo que refleja que pueden estar vinculadas a varias estaciones de servicio.

Entidad	Descripción
id	Identificador único de cada registro de la tabla
code	Nombre de la empresa

FuelStation

Es la tabla encargada de almacenar los datos de las estaciones de servicio y es la tabla principal del esquema, aunque no será la primera en rellenarse de datos, ya que depende de mucha información previa. Para simplificar se ha buscado unificar en la medida de lo posible para poder albergar los datos de los dos documentos de excel. Además de los campos esperados en alguno de los dos documentos, se ha añadido el campo/atributo booleano *is_maritime* para especificar el tipo de estación.

Las relaciones N:1 con las entidades Company, Province, Municipality, Town, y PostalCode muestran que cada estación de servicio está vinculada a una sola compañía, provincia, municipio, ciudad y código postal, pero cada una de estas puede estar asociada a múltiples estaciones de servicio, lo que simplifica los filtrados y unifica los datos.

Entidad	Descripción

id	Identificador único de cada registro de la tabla
company_id	**Clave foránea** de la tabla de compañías.
province_id	**Clave foránea** de la tabla de provincias.
municipality_id	**Clave foránea** de la tabla de los municipios.
town_id	**Clave foránea** de la tabla de las localidades.
postal_code_id	**Clave foránea** de la tabla de los códigos postales.
address	Dirección de la estación de servicio
margin	Ubicación de la estación de servicio, solo acepta un caracter y tenemos tres opciones posibles, N: no aplica, D: derecho, I: Izquierdo. No se controlan que sean esos campos a nivel de base de datos, lo cual es un punto a mejorar, ya sea desde el programa de ingesta o desde la propia base de datos.
latitude	Número decimal que nos indica la latitud GPS, se ha establecido con un máximo de 9 carácteres de los cuales 6 son decimales
longitude	Número decimal que nos indica la longitud GPS, se ha establecido con un máximo de 9 carácteres de los cuales 6 son decimales
opening_hours	Texto que muestra el horario de apertura, se ha hecho como texto porque esta información es muy heterogénea.
is_maritime	Es el booleano con el que controlamos si la estación es marítima o terrestre, con un único campo para eso es suficiente en base al principio de exclusión, si no es esto es aquello.

Price

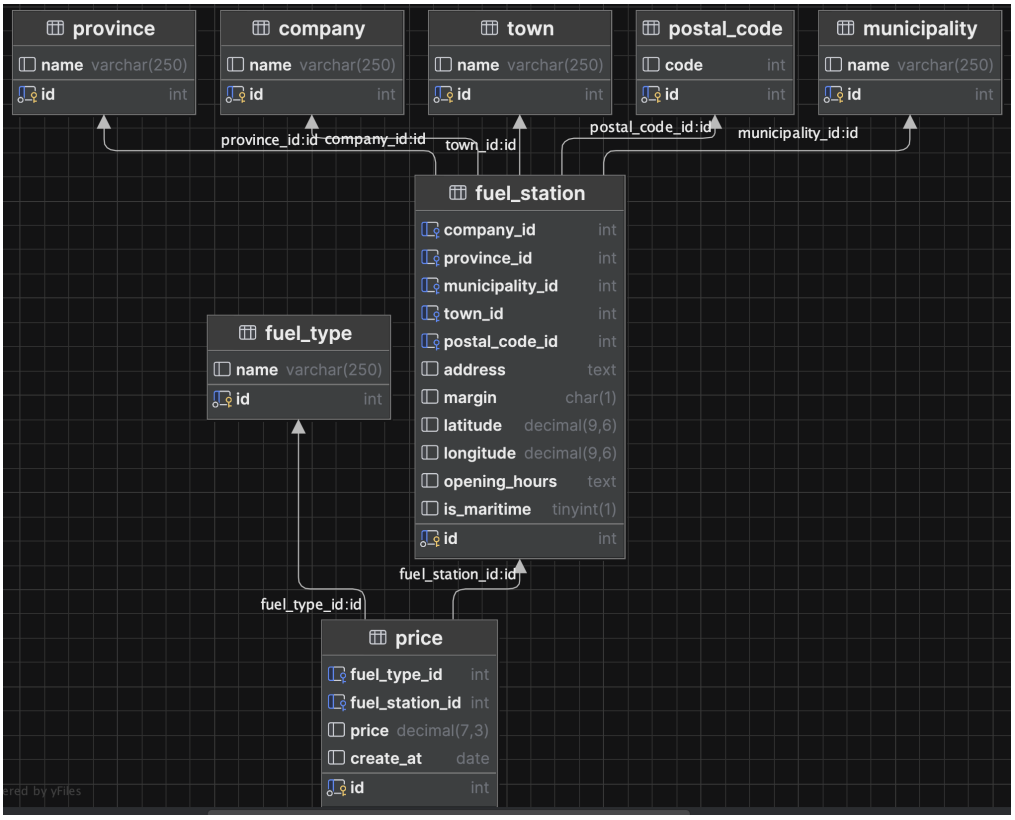
Es la tabla encargada de almacenar los datos sobre los precios de los diferentes tipos de combustible separando este dato de lo que sería la estación de servicio. Ya que si crece la base de datos sería lógico que se crearán nuevos precios para los mismos tipos de combustible, o para otros nuevos, según avanza el tiempo, esto nos permite hacer búsquedas donde poder extraer valores con los que hacer estadísticas de precios por combustible y rango de fechas.

Entidad	Descripción
id	Identificador único de cada registro de la tabla

fuel_type_id	**Clave foránea** de la tabla de tipos de combustible.
fuel_station_id	**Clave foránea** de la tabla de estaciones de servicio
price	Almacena el precio del combustible de la estación de servicio y del combustible indicados en las claves foráneas.
create_at	Almacena la fecha en la que se registró, esto nos permite hacer un histórico de los precios de cada combustible por cada estación de servicio o de manera general.

DDL

La definición de las instrucciones SQL se han ajustado en la medida de los posible a los requerimientos del diseño de la base de datos. La estructura justifica la normalización y el diseño relacional al evitar la redundancia de datos y permitir la escalabilidad. Junto con este documento se adjunta una captura del esquema resultante.



Preparación de los datos para la ingesta

Para facilitar la carga de los datos se ha hecho un formateo previo de los datos, en el que se han modificado caracteres y eliminado columnas innecesarias. A continuación se detallan las intervenciones más importantes:

- **Eliminación de Comas en Números y Coordenadas GPS**
  En el Excel original, para cumplir con el formato CSV y evitar conflictos con el delimitador (que suele ser la coma), se han eliminado las comas de los números, incluidas las coordenadas GPS. Esto es crucial porque las comas en los valores numéricos pueden ser interpretadas incorrectamente como separadores de campo en el formato CSV.

- **Eliminación de Comas en Nombres**
  Al igual que con los números, se han eliminado las comas de los nombres en el Excel para evitar que se interpreten como separadores de campo en el CSV. Esto ayuda a mantener la integridad de los datos, especialmente en campos de texto que pueden incluir comas como parte de la información (por ejemplo, en nombres de compañías o direcciones).

- **Mantenimiento de un Único Archivo por Origen**
  Se mantiene un solo archivo CSV por fuente de datos (estaciones terrestres y marítimas), lo que implica que la responsabilidad de organizar y estructurar los datos recae en el programa de ingesta.

- **Creación del CSV de Tipos de Combustible**
  Se ha creado un archivo CSV a mano para los tipos de combustible, eliminando la palabra "precio" de los nombres de los combustibles y consolidando la información de ambos documentos (terrestres y marítimos). Esto estandariza los datos de tipos de combustible en un solo archivo para facilitar la ingesta, lo cual es una práctica recomendable para normalizar datos y simplificar las operaciones de carga.

Creación del programa de ingesta

Antes de entrar a este punto, tengo que avisar de que no tengo conocimientos muy amplios de Java, mi especialidad previa a entrar en este curso ha estado centrada en el desarrollo con Python. Con esto solo quiero indicar que he hecho lo que mejor he podido en lo que respecta a la escritura del programa y si hay usos extraños o cualquier otra cosa nace del desconocimiento de quien está peleando con un lenguaje por primera vez. Dicho esto continuamos.

El sistema de ingesta se ha construido en base al proyecto de ejemplo que se vio en clase, se ha buscado la forma más simple y redundante en muchos casos para resolver las dificultades que se han ido encontrado para poder procesar los tres archivos CSV.

- **Todos los precios estaban en columnas separadas:**
  Se ha tenido que crear una función que se encarga de recoger el dato indicando la posición de la columna en el CSV y el nombre del tipo de combustible para poder buscar en la base de datos el identificador.
- **Riesgo de campos duplicados:**
  Por ejemplo en de localidad, municipio, provincia, código postal y compañía o incluso las propias estaciones de servicio y precios. Para solventar esto en los campos de las tablas destinadas a categorizar he utilizado Set para evitar duplicados durante la elaboración de las listas de cada uno de estos campos, ahorra tiempo y evita complejidad innecesaria. En el caso de la tabla de precios y estaciones de servicio se ha introducido un filtro para buscar por los campos que no deberían cambiar en cada ingesta, de esta forma si existe no lo introducimos en la base de datos.
- **Solo podemos buscar por nombre**:
  En la mayoría de los casos buscamos por nombre para establecer las relaciones con los registros existentes en las tablas que ya se han cargado de datos y que se tiene que relacionar entre sí. Para que funcione correctamente se ha seguido un orden similar al que se ve en la sección de entidades de este documento para que la ingesta tenga sentido y siempre estén todos los datos presentes.

El objetivo en todo el proceso ha sido crear funciones que sean reutilizables para montar, por decirlo así, la ingesta con cada uno de los documentos que se tienen que leer. Según se ha ido avanzando se han ido añadiendo aquellos elementos necesarios para facilitar, en la medida de lo posible, el proceso. Un ejemplo que resume este proceso es que al principio el uso de Set no daba los resultados esperados, hasta que analizando el funcionamiento del Set en Java necesitaba dos funciones en la clase de la entidad que va a comparar para que pueda hacer esa comprobación de duplicados que estaba buscando con su uso.

Consultas solicitadas

Sobre las consultas todas están disponibles en el documento que acompaña al proyecto dentro de la carpeta “queries”, acompañadas de comentarios en un único archivo SQL. Del resto sólo cabe destacar el caso de las coordenadas GPS, el cuales el más complejo que me he encontrado al desconocer esa funcionalidad dentro de MySQL bajo el nombre “ST_Distance_Sphere”.

Los resultados de cada una de ellas son los siguientes:

- Nombre de la empresa con más estaciones de servicio terrestres:
  Repsol con 2755
- Nombre de la empresa con más estaciones de servicio marítimas:
  Repsol con 70
- Localización, nombre de empresa, y margen de la estación con el precio más bajo para el combustible “Gasolina 95 E5” en la Comunidad de Madrid:
  Cl crta. de Torrelodones 10, Ballenoil, D
- Localización, nombre de empresa, y margen de la estación con el precio más bajo para el combustible “Gasóleo A” si resido en el centro de Albacete y no quiero desplazarme más de 10 KM:
  Avenida primera s/n, Gmoil, D
- Provincia en la que se encuentre la estación de servicio marítima con el combustible “Gasolina 95 E5” más caro:
  Pontevedra

Conclusiones

En resumen, todo el proceso ha sido un aprendizaje continuo, teniendo en cuenta el punto que señalaba anteriormente sobre Java además de las particularidades del tipo de base de datos que estamos manejando.

Diseño:
He buscado organizar los datos en base a los casos de uso planteados por las consultas que se necesitaban realizar. Esto me ha permitido centrar mejor el diseño desde el borrador inicial. Una mejora que podría haber hecho en este punto, podría haber sido añadir las posiciones de los precios en la lista de tipos de combustibles, lo que facilitaría el proceso de ingesta.

Desarrollo:
Ha sido complejo, teniendo en cuenta lo que resaltaba en el punto de la creación del sistema de ingesta. He tenido que ir aprendiendo sobre la marcha las soluciones que ya conocía en Python, aunque ante la duda he optado por las soluciones más sencillas aunque sean repetitivas. Podría haber mejorado la ingesta automatizando en base los tipos de combustible disponibles en la base de datos, además de implementar la inserción durante el proceso de lectura. Pero no he visto esta solución hasta el final y no he podido dedicar tiempo para refactorizar el código.

Anexo: Instrucciones de uso y contenido

Junto con este documento tenemos una carpeta con el proyecto, en su interior tenemos la siguiente estructura:

- proyecto:
  Además de las carpetas indicadas están los archivos de las imágenes incluidas en este documento.
  - csv: Contiene los archivos CSV que se han creado para la ingesta de los datos.
  - database: Contiene el archivo SQL para la creación de la base de datos.
  - excel: Contiene los archivos originales en los que se ha basado todo el proceso.
  - queries: Contiene el archivo SQL con todas las consultas que se solicitaron con el ejercicio.
  - src: Ficheros del proyecto de Java.

Configuración de la ejecución utilizando la base de datos que creamos con MySQL: