TRABAJO FIN DE GRADO

Grado en Ingeniería Informática

UNIVERSIDAD DE ALMERIA

ESCUELA SUPERIOR DE INGENIERÍA

"Creación de un data warehouse sobre OpenStack-DI"

Curso 2020/2021

Alumno/a:

Francisco Javier López Soriano

Director/es:

Manuel Torres Gil

Trabajo Fin De Grado

Creación de un data warehouse sobre OpenStack-DI

Alumno/a: Francisco Javier López Soriano

Director: Manuel Torres Gil

Grado en Ingeniería Informática

Escuela Superior de Ingeniería

Universidad de Almería

Índice

Índice de figuras

Índice de tablas

Índice de fragmentos de código

Abreviaturas

BI	Business Intelligence
SaaS	Sofware as a service
IaaS	Infraestructure as a service
DW	Data Warehouse
REST	RESTful
ETL	Extract, Transform, Load
SSMA	SQL Server Migration Assistant
AS	Analysis Services
IS	Integration Services
SSDT	SQL Server Data Tools
OLAP	On-Line Analytical Processing
SSAS	SQL Server Analysis Server
SSIS	SQL Server Integration Server
3FN	Tercera Forma Normal
MDX	MultiDimensional eXpressions

1. Introducción

OpenStack-DI es el proveedor de Infraestructura como servicio (IaaS) de Cloud-DI, un cloud privado del Departamento de Informática de la UAL que da soporte a actividades de docencia e investigación. OpenStack-DI además, es el soporte para el resto de servicios de Cloud-DI, como son servidores virtuales para almacenamiento de archivos, gestión de proyectos, control de versiones, automatización de tareas y autenticación de usuarios.

Sin embargo, hasta la fecha nunca se ha realizado ningún tipo de análisis de los datos almacenados en dicha plataforma.

Para realizar este análisis no existe una plataforma, herramienta, aplicación… en la cual sea posible introducir los datos recogidos (ya que actualmente se recogen en las diferentes bases de datos de OpenStack-DI los recursos utilizados, las personas que participan en la plataforma, etc). Así, el propósito del conjunto de este Trabajo y Complemento de Fin de Grado es realizar dicho análisis utilizando técnicas de data warehousing para observar fácilmente el consumo de los recursos utilizados.

Este proyecto por tanto consiste en crear e implementar un Data Warehouse, entendido como un repositorio de datos que proporciona una visión global, común e integrada de los datos de la organización. La implementación de un Data Warehouse transfiere además una serie de propiedades a los datos de la organización: estabilidad, coherencia en los datos, fiabilidad y lo más importante, la posibilidad de obtener información a lo largo del tiempo.

Este conjunto de herramientas y procesos se encuentra dentro del área de Business Intelligence, o BI, como será referenciado a partir de ahora. Business Intelligence, o Inteligencia Empresarial en castellano, es el conjunto de procesos requeridos para ofrecer una solución informática que permita analizar cómo está funcionando una empresa [1]. Dentro del mundo del BI hay múltiples soluciones y maneras de realizar estos procesos, presentando algunas de ellas a lo largo de este proyecto.

Las soluciones más habituales están basadas en la creación de informes tanto predefinidos como a medida, junto a su distribución de forma automatizada. Algunas soluciones serían:

- La previsión de resultados (forecasting)
- Las herramientas de consultas para usuarios avanzados (query) incluyendo el acceso a cubos multidimensionales (OLAP)
- Los cuadros de mando (dashboards)
- Los almacenes de datos especiales (datawarehouse o datamarts)

A lo largo del proyecto se desarrollarán consultas para usuarios avanzados con el acceso al cubo multidimensional, que será creado previamente, la estructura datawarehouse, o DW, que contiene lo necesario para esta creación y una breve introducción a los cuadros de mando.

Como resumen, el objetivo del proyecto es realizar Data Warehousing sobre OpenStack-DI, que es el nombre que recibe el proceso que facilita la creación y explotación de un Data Warehouse.

1.1 Motivación

La motivación principal de este proyecto es adquirir nuevos conocimientos en el área de BI, ya que considero que el aprendizaje de la construcción de una base de datos multidimensional es más útil actualmente para una empresa que el aprendizaje de una base de datos relacional, ya que puedo aportar valor real a la organización, como se ha comentado en la introducción, pudiendo tanto lograr satisfacer los objetivos de negocio de una organización como plantear nuevas estrategias en procesos internos o externos.

Así pues, este proyecto se ve motivado por dos situaciones. La primera, de carácter personal, es el entusiasmo por la construcción de un Data Warehouse complejo y el aprendizaje del entorno de éste, incluyendo los procesos de extracción de datos de las bases de datos de OpenStack-DI, así como el estudio y utilización de una nueva aplicación para visualizar los datos de la infraestructura OpenStack-DI.

La segunda motivación, es la comentada también en el apartado de Introducción, donde descubrí que el departamento de informática no analiza los recursos utilizados por la plataforma OpenStack-DI y podría ser de utilidad la construcción de un entorno donde poder observar información como que usuarios usan más instancias, en qué circunstancias se usa más la plataforma... para poder mejorar en rendimiento o costes de dicha plataforma.

1.2 Objetivos

El objetivo principal del TFG es adquirir conocimientos en el área de BI que sean útiles en mi proyección profesional, ya que hasta el momento es cierto que he tratado con diferentes sistemas de información, tanto relacionales como transaccionales, pero no he realizado ningún proyecto desde cero, aprendiendo por tanto nuevos pasos y resolviendo nuevos problemas, estando así menos limitado en mis capacidades a la hora de afrontar un proyecto en un trabajo real.

Con la construcción de esta infraestructura se pretende que el departamento de informática de la Universidad de Almería tenga la información necesaria para poder importar esta infraestructura en una aplicación de análisis y llevar a cabo un estudio de la plataforma.

Para conseguir esto se plantean los siguientes objetivos:

- Análisis de las bases de datos de la plataforma OpenStack-DI, para decidir qué tablas (datos) podrían ser de utilidad para construir un modelo multidimensional.
- Estudiar, comprender y aplicar los procesos ETL para la construcción de un *Data Warehouse,* donde se tratarán las tablas comentadas anteriormente de la plataforma OpenStack-DI para realizar un proceso de limpieza y corrección de estas.
- Construcción de un modelo multidimensional para facilitar la construcción de un cubo OLAP, el cual debe ser de uso intuitivo para ofrecer valor al departamento.
- Ejemplificación con un par de casos básicos de informes de consulta habituales para facilitar al usuario el acceso a la información.

En resumen, el objetivo es conseguir aplicar varias tecnologías BI para demostrar así el potencial que se puede alcanzar y lo útil que puede llegar a ser.

1.3 Estructura de la memoria

La estructura del presente Trabajo de Fin de Grado se detalla a continuación:

- Capítulo 1: Introducción. En este capítulo se introduce el Trabajo Fin de Grado describiendo la motivación, objetivos y la planificación temporal.
- Capítulo 2: Herramientas y tecnologías. En este capítulo se realizará una pequeña introducción a las tecnologías y herramientas utilizadas en la construcción de la infraestructura y la consecución de los objetivos de este trabajo expuestos anteriormente.
- Capítulo 3: Desarrollo del proyecto. Una vez que sabemos qué queremos hacer y qué tecnologías y herramientas utilizar, llega el momento de utilizar estas herramientas para llevar a cabo las ideas planteadas. En dicho capitulo se presentará adecuadamente el problema y la explicación del desarrollo de las partes para conseguir la infraestructura final.
- Capítulo 4: Conclusiones y trabajo futuro: En este capítulo se citará lo aprendido al realizar este proyecto, así como las posibles mejores y ampliaciones de este proyecto.
- Bibliografía: Se enumeran las referencias y fuentes consultadas para la elaboración de este Trabajo Fin de Grado.

1.4 Planificación

El siguiente cronograma indica el periodo de duración de este Trabajo Fin de Grado dividido en cinco fases principales comentadas después del cronograma, correspondiendo a un total de 300 horas de trabajo mostrando estas horas utilizadas para cada etapa en el propio cronograma.

Etapas del proyecto												
Semanas	1ª	2ª	3ª	4ª	1ª	2ª	3ª	4ª	1ª	2ª	3ª	4ª
Cronograma TFG												
1. Análisis de la base de datos actual sobre OpenStack-DI	30											
2. Estudio de los procesos ETL y su funcionamiento.		35										
3. Transformación de los datos mediante procesos ETL.			75									
4. Creación del cubo mediante Analysis Services.						90						
5. Elaboración de la memoria del Trabajo Fin de Grado		70										

Tabla 1: Cronograma del proyecto

El desarrollo del proyecto ha sido planificado, como vemos en el cronograma, en las siguientes cinco fases principales:

I. **Análisis de la base de datos actual sobre OpenStack.** Análisis sobre la base de datos para comprender su estructura, qué datos analizar y cómo tratarlos

II. **Estudio de los procesos ETL y su funcionamiento.** Estudiar cómo se transforman los datos mediante procesos ETL.

III. **Transformación de los datos mediante procesos ETL.** Preparación de los datos para introducirlos en el *Data Waterhouse.*

IV. **Creación del cubo mediante Analysis Services.** Creación de la estructura multidimensional donde estarán almacenados los datos que posteriormente serán estudiados y visualización del mismo con ejemplos prácticos.

V. **Elaboración de la memoria del Trabajo Fin de Grado**. Realización de la documentación final que recoge toda la información del TFG.

2. Herramientas y tecnologías

En este capítulo se explican todas las herramientas y tecnologías usadas para el desarrollo del proyecto. No obstante, antes de comenzar, se muestra la figura siguiente para ilustrar cada una de ellas y el papel que juega en este proyecto.

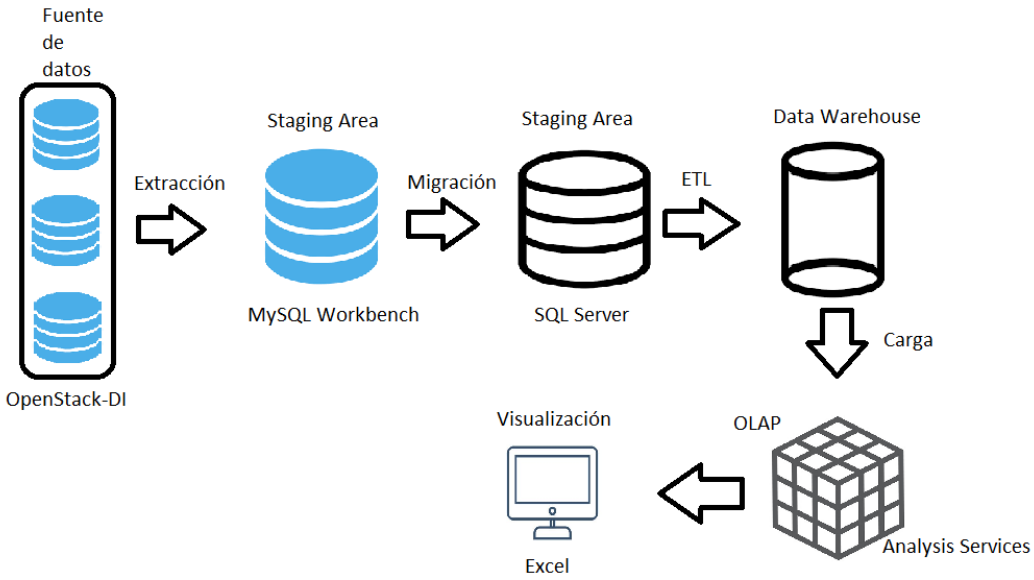


Figura 1: Proceso Data Warehousing proyecto

2.1 Herramientas

En este subapartado se va a comentar que herramientas vamos a usar durante el desarrollo del Data Warehouse, que es el objetivo de este Trabajo Fin de Grado.

2.1.1 MySQL Workbench

MySQL Workbench es una herramienta que integra diseño y administración de bases de datos con facilidades para la gestión y mantenimiento de bases de datos MYSQL [2].

Esta herramienta permite diseñar visualmente, modelar, generar y administrar bases de datos incluyendo todo lo necesario para modelar datos, realizar ingeniería directa e inversa y un sinfín de utilidades para desarrollar la primera parte de este proyecto.

Figura 2: Logo MySQL Workbench

MySQL Workbench ha sido utilizado para analizar la base de datos de OpenStack-DI, ya que el motor de datos en el que está basado esta plataforma es *MySQL.*

Para poder utilizar la base de datos MySQL tenemos que instalar el motor *MYSQL* [3], que en este caso se trata de un gestor de base de datos al adaptar el estándar SQL (también conocido como MySQL Server) y durante esta instalación es posible añadir la instalación de un motor gráfico como el comentado MySQL Workbench donde se ha resuelto la primera parte del desarrollo de este proyecto.

MySQL Workbench presenta el diseño que se puede ver en la siguiente figura, siendo esta un ejemplo de abrir un esquema de datos almacenado en MySQL.

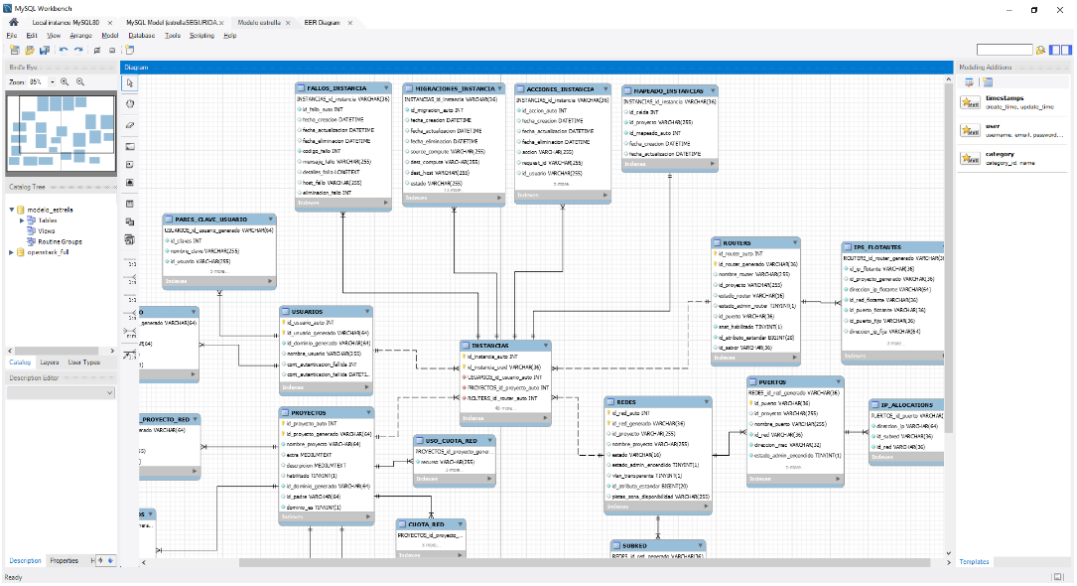


Figura 3: Captura de pantalla de la herramienta MYSQL Workbench

2.1.2 Microsoft SQL Server Migration Assistant

Microsoft SQL Server Migration Assistant (SSMA) es una herramienta diseñada para automatizar la migración de bases de datos a SQL Server desde Microsoft Access, DB2, MySQL, Oracle y SAP ASE.

Esta herramienta es utilizada para realizar la migración de una base de datos MySQL construida en MySQL Workbench a una base de datos SQL Server. Para su utilización hace falta tener instalado los motores de origen y destino de la migración, en este caso, el motor MySQL y el motor SQL Server. Además, para la conexión con la base de datos de MySQL hace falta el controlador MyODBC que se trata de un conector que proporciona acceso a una base de datos MySQL.

La siguiente imagen muestra un ejemplo de visualización de la herramienta.

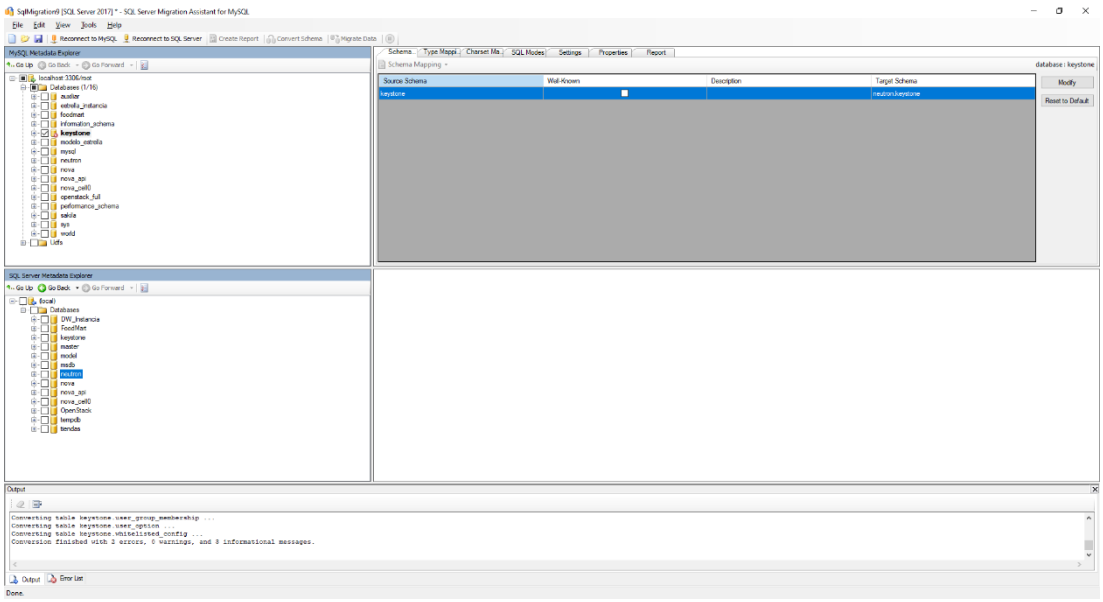


Figura 4: Captura de pantalla de la herramienta SSMA

2.1.3 Microsoft SQL Server Management Studio

SQL Server Management Studio es un entorno integrado para administrar cualquier infraestructura de SQL Server.

La herramienta permite acceder a varios tipos de servidor:

- Database Engine: Sistema de Gestión de Bases de Datos.
- Analysis Services: Herramienta de procesamiento analítico y minería de datos.
- Reporting Services: Software de generación de informes.
- Integration Services: Componente utilizado para migración de datos.

Esta herramienta ha sido utilizada como sistema de gestión de las bases de datos utilizadas para la realización de este proyecto y como conexión a Analysis Services para visualizar y consultar el resultado final del Data Warehouse.

Para poder gestionar la base de datos SQL Server de una manera más sencilla se ha realizado la instalación de esta herramienta comentada previamente, SQL Server Management Studio que será instalada durante el proceso de instalación del motor SQL Server [4].

La siguiente figura muestra una imagen de visualización de la herramienta cuando se realiza una consulta sobre una de las bases de datos que se pueden almacenar en el motor SQL Server.

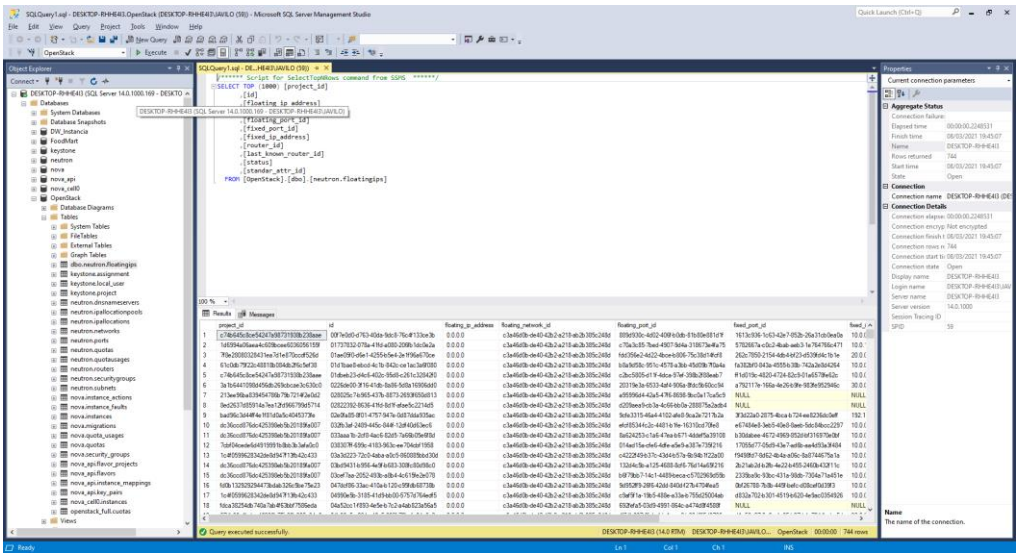


Figura 5: Captura de pantalla de SQL Server Management Studio

2.1.4 Azure DevOps

Azure DevOps hospeda los repositorios creados en Visual Studio conectados mediante Team Explorer [5].

Figura 6: Logo Azure DevOps

Con esta herramienta es posible almacenar los repositorios creados en Visual Studio donde se realiza este proyecto en una instancia del servidor Azure.

Mediante la página web de la instancia del servidor Azure es posible explorar “commits, pushes, branches...”, navegar por los archivos subidos y ver el historial de cambios para comparar versiones. La principal ventaja frente a otros controladores de versiones es que está integrado en Visual Studio con Team Explorer y esto ofrecer la posibilidad de trabajar directamente desde Visual Studio con el proyecto/repositorio.

La figura 7 muestra la conexión realizada en un proyecto además de poder visualizar el panel que ofrece al integrarse en Visual Studio.

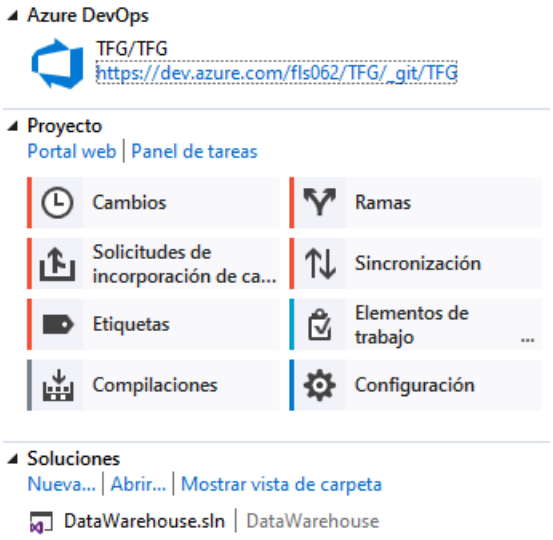


Figura 7: Vista Azure DevOps desde Visual Studio

2.1.5 Microsoft SQL Server Data Tools

Microsoft SQL Server Data Tools, o SSDT abreviado, es una herramienta de desarrollado integrada en Visual Studio para crear bases de datos relacionales, bases de datos de Azure SQL, modelos de datos de Analysis Services (AS), paquetes de Integración Services (IS) e informes de Reporting Services. Gracias a SSDT, es posible diseñar e implementar cualquier tipo de contenido de SQL Server de la misma manera y facilidad con la que se desarrollaría una aplicación en Visual Studio, gracias a su diseño gráfico.

Figura 8: Logo Visual Studio

Serán usados dos tipos de proyectos incluidos en esta herramienta, Analysis Services, o SSAS, e Integration Services, o SSIS.

Con SSAS es posible crear un modelo tabular y multidimensional y administrar bases de datos. Estos modelos son creados mediante plantillas de proyecto en Visual Studio con la extensión Analysis Services

Con SSIS es posible crear un contenedor de paquetes Integration Services con los cuales extraer, cargar o transformar datos.

2.1.6. Microsoft Excel

Microsoft Excel es una herramienta del conjunto Microsoft Office 365 que permite crear hojas de cálculo.

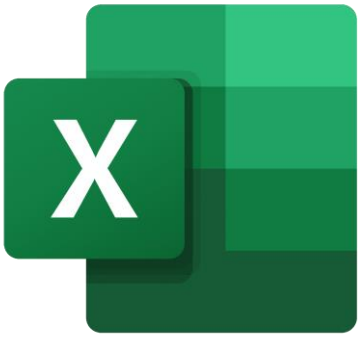


Figura 9: Logo Excel

Esta herramienta será utilizada para realizar consultas sobre el cubo desarrollado en el proyecto, ya que permite realizar conexiones a orígenes de datos Analysis Services. La herramienta es capaz de crear una hoja de cálculo dinámica donde visualizar los campos de la tabla en función de la utilidad del propio campo, así como crear gráficos dinámicos de estos datos.

2.2 Tecnologías

En este subapartado vamos a comentar que tecnologías vamos a usar durante el desarrollo del Data Warehouse, que es el objetivo de este Trabajo Fin de Grado.

2.2.1 MySQL

Figura 10: Logo MySQL

MySQL es un sistema de gestión de base de datos relacional que ofrece la posibilidad de ejecutarse en casi todos los sistemas operativos y es el sistema utilizado por la plataforma OpenStack-DI para almacenar su información, debido a que es un sistema de código abierto y permite tener relaciones entre tablas al tratarse de un sistema basado en bases de datos relacionales.

Entre las principales características del MySQL que puede indicarle al servidor, están [6]:

- Consultar los datos del modelo almacenado.
- Realizar operaciones para manipular o modificar los datos.
- Definir relaciones entre datos.
- Definir los tipos de datos del modelo.

Esta tecnología será usada en primera instancia en el desarrollo del proyecto debido a que es la utilizada por la plataforma como ha sido comentado anteriormente y al tener un numero de relaciones entre tablas significativo optar por un motor MySQL era la mejor opción.

2.2.2 SQL Server

Figura 11: Logo SQL Server

Es un sistema de gestión de base de datos utilizado para la construcción del modelo del Data Warehouse y para la construcción de los procesos ETL y el cubo OLAP apoyando esta tecnología con SSDT.

Las principales características de Microsoft SQL Server [7]:

- Soporte de transacciones.
- Escalabilidad, estabilidad y seguridad.
- Permite trabajar en modo cliente-servidor.

3. Desarrollo del proyecto

A lo largo de este capítulo se va a desarrollar en varias secciones el proyecto, cada sección comprende una de las principales tareas en las que se ha dividido el proyecto, explicando detalladamente los pasos seguidos para completar cada desarrollo.

3.1 Presentación del problema

Este proyecto surge de la necesidad de analizar los datos de uso de OpenStack-DI, proveedor de Infraestructura como servicio (IaaS) de Cloud-DI, un cloud privado del Departamento de Informática, pero hasta la fecha nunca se ha realizado ningún tipo de análisis de los datos almacenados en dicha plataforma debido a que actualmente el Departamento de Informática la manera que tiene de guardar los datos es consolidar esta información de las instancias, proyectos y usuarios en el almacenamiento de datos relacionales.

Y es que, el almacenamiento de datos relacional presenta los siguientes problemas [8]:

- Los informes presentados son estáticos y los usuarios que realicen una exploración de estos informes no lo podrán hacer de una manera interactiva con la que obtener más detalle.
- El rendimiento de las consultas es muy variable, debido a que las bases de datos relacionales presentan muchas uniones entre tablas y esto puede ralentizar las consultas.
- Es difícil combinar los conjuntos de información relacionados.
- Es difícil auditar la información.

Debido a estos inconvenientes se ha de buscar una solución que no incluya el almacenamiento de datos relacional y ahí es donde aparece el concepto de Data Warehouse.

Por tanto, con este proyecto se pretende proporcionar una herramienta para la toma de decisiones o el estudio de los recursos de cualquier área mediante un modelo de datos multidimensional que facilitará la navegación y explotación de datos que nos proporcionarán las siguientes ventajas frente al almacenamiento de datos relacional:

- Proporcionar acceso a datos unificados.
- Simplificar la vista de datos a los usuarios, acelerando el desarrollo de consultas interactivas y predefinidas.
- Creación de consultas con la posibilidad de unir diferentes áreas temáticas
- Crear y configurar agregados.
- Almacenar y reutilizar cálculos complejos.
- Presentar una versión traducida en cualquier idioma según la necesidad del usuario final.

El núcleo del proyecto es por consecuente un Data Warehouse, y antes de continuar detallando el problema, hay que explicar en qué consiste este concepto tan nombrado como es el Data Warehouse o también traducido como Almacén de Datos en español. Un Data Warehouse supone una base de datos corporativa que se caracteriza por integrar y depurar información de una o más fuentes distintas, para luego procesarla, proporcionando una visión global, común e íntegra de los datos de la organización con independencia de cómo se vayan a tratar por parte del usuario final.

Cuando hablamos de Data Warehouse hay que tener en cuenta otros conceptos relacionados en su contexto:

- Data Mart: es un subconjunto o varios subconjuntos de los datos almacenados en el Data Warehouse que ofrecen la posibilidad de responder a diferentes funciones o necesidades de los usuarios o departamentos.
- Data Warehousing: es el proceso de extracción y filtrado de los datos recogidos por las bases de datos que recogen las operaciones de la organización para realizar una transformación, integración y almacenamiento en un almacén de datos cuya utilidad final es acceder directamente al almacén para dar soporte en la toma de decisiones de la organización.
- ODS (Operational Data Store): es un tipo de almacenamiento de datos que proporciona únicamente los últimos valores de los datos y no su historial.
- Procesos ETL: es la tecnología que permite integrar los datos de origen en los ODS, Data Marts y Data Warehouse.
- Metadatos: estos datos hacen referencia a los propios datos hablando sobre otros datos.

La ventaja principal que ofrece este tipo de base de datos consiste en la posibilidad de estructurar la información que se almacena de diferentes maneras. La información está presente de manera homogénea y fiable, haciendo posible la consulta y el tratamiento jerarquizado de ésta.

Bill Inmon fue uno de los pioneros en tratar el tema relativo al Almacén de datos, y destacó las como características del Data Warehouse la orientación a temas ya que solo se seleccionan los datos necesarios para generar el modelo, la posibilidad de variar los datos en el tiempo, la no volatilidad de los datos debido a que la información no se puede modificar y por tanto no se puede eliminar una vez se han almacenado los datos en la estructura, la integración de los datos en una estructura consistente y la existencia de metadatos [9].

Recopilando toda esta información y traduciéndola a este proyecto, consistirá en una gran base de datos cuyas fuentes son las distintas bases de datos proporcionadas por OpenStack-DI. La idea principal es que la información de nuestra base de datos sea presentada desnormalizada para optimizar las consultas.

Así pues, este problema se va a resolver desarrollando una serie de apartados que finalizarán con la construcción de una estructura multidimensional presentada en diferentes herramientas de visualización.

El primero paso del proyecto consiste en estudiar las bases de datos recibidas de OpenStack para poder descartar aquellas tablas que no son necesarias y elegir aquellas que son de interés para el modelo.

En segundo lugar, se construirá el modelo del Data Warehouse basado en las tablas elegidas previamente. Este modelo es la principal diferencia entre una base de datos relacional y una base de datos multidimensional y será más detallado en su desarrollo.

En tercer lugar, se desarrollarán unos procesos denominados ETL que se encargan de extraer la información de las tablas seleccionadas, transformar los datos de las tablas y cargarlos en la Data Warehouse.

En cuarto lugar, se realizará la construcción del BackEnd donde se encuentra el desarrollo de la estructura multidimensional para el uso del usuario final. Este BackEnd es también conocido como cubo OLAP, y es el método más utilizado para analizar y evaluar los datos que tiene la Data Warehouse.

Por último, se presentarán las herramientas para visualizar los datos contenidos en el cubo OLAP.

Estos pasos entrarían en lo conocido como Data Warehousing, concepto comentado previamente y se pueden visualizar gráficamente en la siguiente figura referenciada [10] donde se muestra la arquitectura habitual de este Data Warehouse.

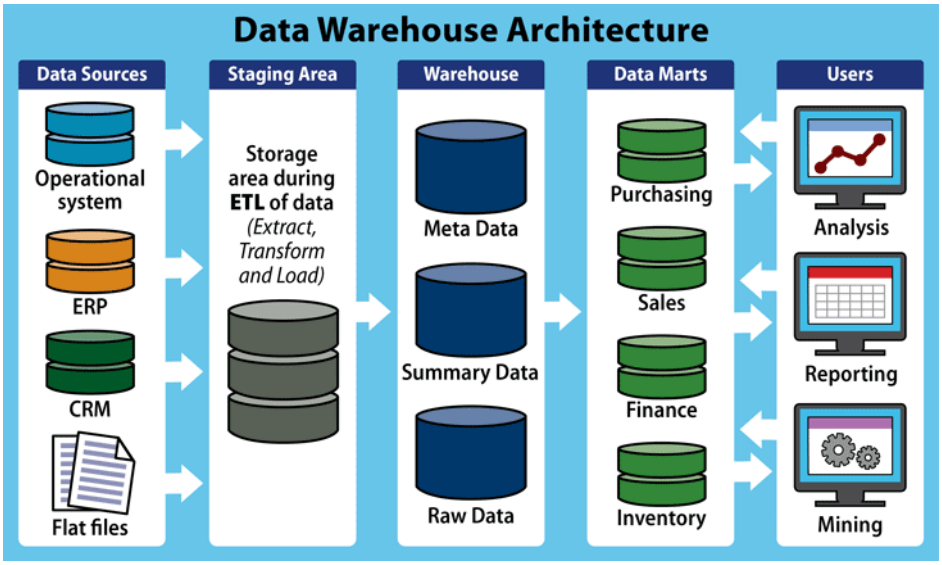


Figura 12: Arquitectura Data Warehouse

A partir de ahora, se encuentran los apartados que contienen estos pasos comentados del desarrollo del proyecto.

3.2 Descripción de la Base de Datos OpenStack

La base de datos con la que se realiza este proyecto tiene su procedencia en la infraestructura OpenStack ofrecida por Departamento de Informática de la UAL como se ha explicado en el apartado 3.1 *"Presentación del problema"* de este documento.

OpenStack-DI es una plataforma cloud que ofrece una herramienta web a los alumnos y docentes de la Universidad de Almería para crear Saas (Software as a service) e Iaas (Infraestructure as a service).

OpenStack-DI es una implantación propia del departamento de informática de la plataforma OpenStack, y por eso, es posible acceder a las bases de datos que albergan toda la información que respaldan los servicios.

Estas bases de datos suministradas solamente incluyen la parte de la plataforma que ofrece servicios IaaS. En esta sección la plataforma ofrece la posibilidad de crear proyectos, donde a su vez se pueden crear instancias con las características deseadas. Estas instancias están respaldadas por una red y grupo de seguridad el cual también puede modificar el usuario.

Por tanto, las bases de datos proporcionadas incluirán información sobre estas características, y esta información será a su vez el objeto de estudio una vez desarrollado el proyecto.

En primera instancia, comenzando con el desarrollo de esta sección, cabe mencionar, que, aunque el resultado final sea un modelo o base de datos, este procede de la unión de varias bases de datos, que se van a cargar en MYSQL Workbench mediante scripts SQL. El fragmento 1 muestra un trozo de script SQL de la carga de una base de datos.

```
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8 */;
/*!40103 SET @OLD_TIME_ZONE=@@TIME_ZONE */;
/*!40103 SET TIME_ZONE='+00:00' */;
/*!40014 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0 */;
/*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0 */;
/*!40101 SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;
/*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;

--
-- Table structure for table `access_token`
--

DROP TABLE IF EXISTS `access_token`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `access_token` (
  `id` varchar(64) NOT NULL,
  `access_secret` varchar(64) NOT NULL,
  `authorizing_user_id` varchar(64) NOT NULL,
  `project_id` varchar(64) NOT NULL,
  `role_ids` text NOT NULL,
  `consumer_id` varchar(64) NOT NULL,
  `expires_at` varchar(64) DEFAULT NULL,
  PRIMARY KEY (`id`),
  KEY `ix_access_token_authorizing_user_id` (`authorizing_user_id`),
  KEY `ix_access_token_consumer_id` (`consumer_id`),
  CONSTRAINT `access_token_ibfk_1` FOREIGN KEY (`consumer_id`) REFERENCES `consumer` (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;
```

Fragmento 1: Código SQL para la creación de una base de datos

En total hemos cargado 5 bases de datos mediante 5 scripts proporcionados por el administrador de la infraestructura OpenStack, estas bases de datos son las siguientes:

- Base de datos keystone: Contiene los datos de los proyectos y usuarios y se puede ver su esquema en la siguiente figura.

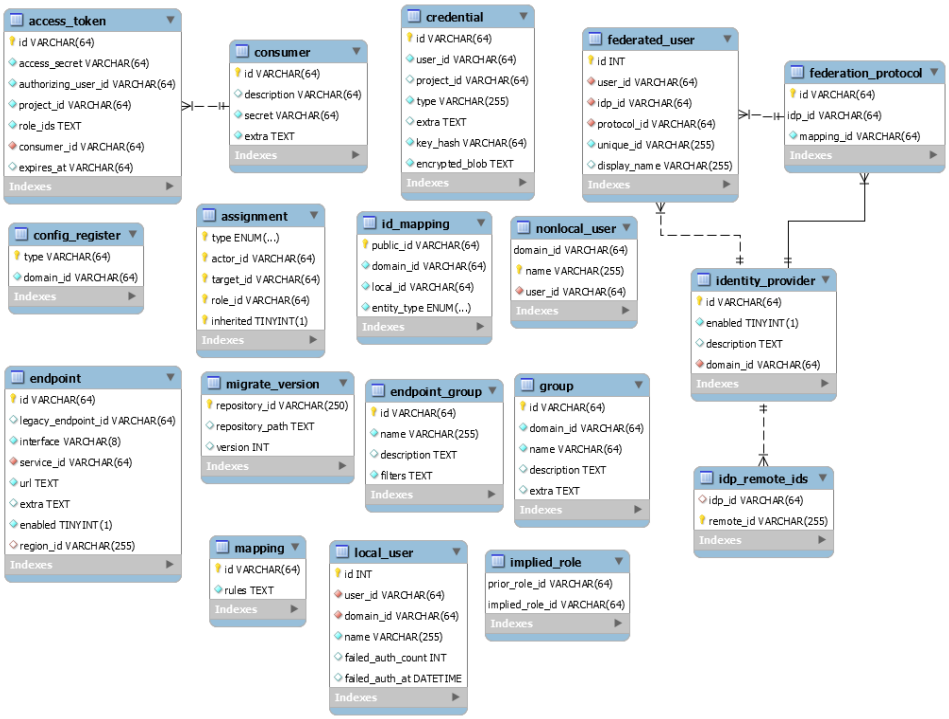


Figura 13: Ejemplo del contenido de la base de datos keystone

- Base de datos neutron: Contiene datos de las redes y vemos el contenido de algunas tablas en la siguiente figura.

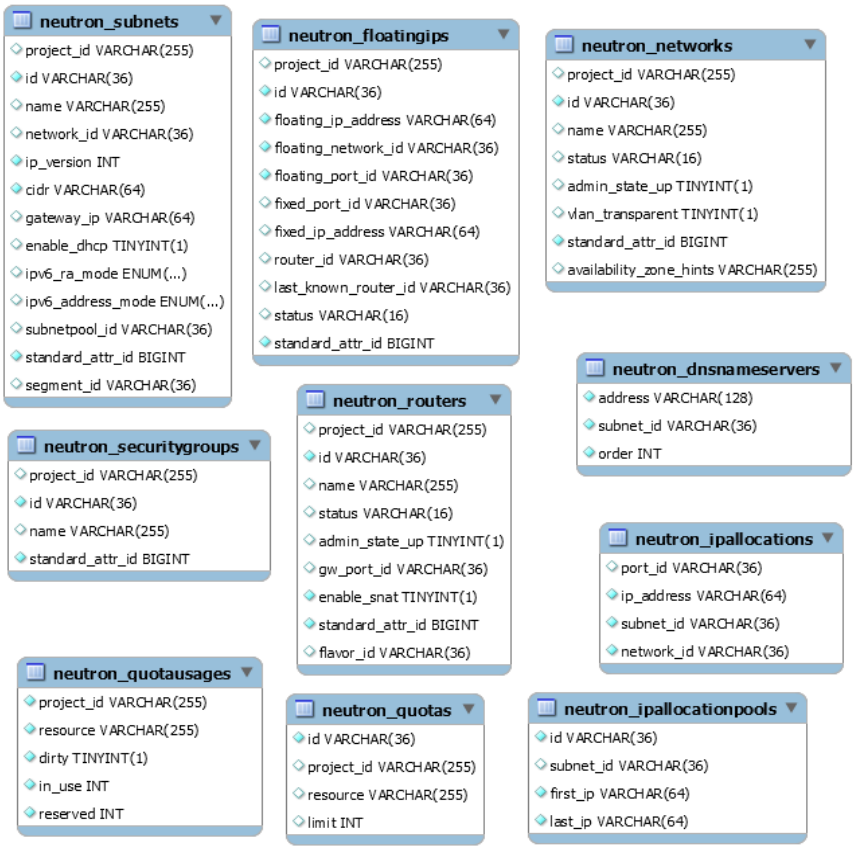


Figura 14: Ejemplo del contenido de la base de datos neutron

- Base de datos nova, nova_api y nova_cell0: Contiene datos sobre las instancias. La figura 15 contiene algunas de las tablas almacenadas en esta base de datos.

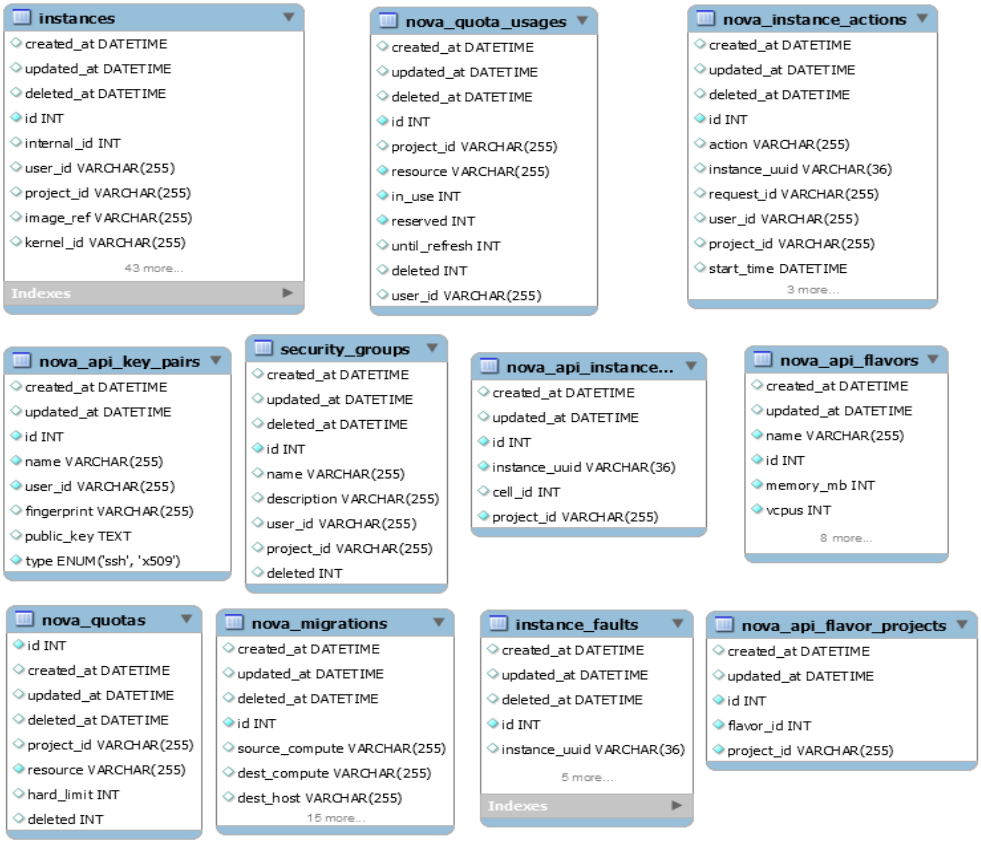


Figura 15: Ejemplo del contenido de las bases de datos nova

Una vez estudiado cada esquema proporcionado por la infraestructura OpenStack el siguiente paso consiste en sintetizar todas las bases de datos en una única base de datos sobre la que construir el modelo estrella además de eliminar tablas que carecen de importancia para el análisis de datos o que se encuentren vacías.

Tras este proceso, se dispone de 12 tablas para realizar el esquema estrella. Cada tabla seleccionada contiene información importante ya sea para el futuro análisis de la base de datos o para proporcionar información entre tablas. Las tablas seleccionadas y su contenido vienen desarrollado a continuación:

- **Instances**: Contiene toda la información sobre la creación, modificación y eliminación de una instancia y se trata de la unión de dos tablas de instancias encontrados en dos bases de datos para aumentar los registros.
- **Instance_faults:** Son instancias que han fallado en alguna de sus acciones y en esta tabla se recogen algunos datos como el código y mensaje de dicho error, a parte de las correspondientes fechas (desde que se lanza hasta que falla la instancia).
- **Keystone_local_user**: Contiene a todos los usuarios y características de estos como el dominio, nombre, conexiones fallidas y cuando se realizó la última conexión.
- **Keystone_project**: Contiene todos los proyectos creados, con el nombre y descripción de estos, si se encuentra habilitado, y su dominio.

- **Neutron_dnsnamesservers:** Contiene las direcciones dns y a que subred se encuentra asociada esa dirección dns.
- **Neutron_networks:** Contiene información de las redes de los proyectos, así como el nombre identificativo de la red, información de su estado, si se encuentran levantadas, etc.
- **Neutron_routers:** Contiene la información sobre los routers, así como sus asociaciones.
- **Neutron_securitygroups**: Contiene los grupos de seguridad de las redes, asociados al proyecto.
- **Neutron_subnets:** Contiene las subredes, asociadas a un proyecto y una red.
- **Nova_api_flavors:** Contiene los sabores(tamaños) y las características de estos que se le pueden poner a las instancias.
- **Nova_api_key_pairs:** Contiene los pares de claves que han creado los usuarios para sus proyectos.
- **Nova_migrations:** Contiene las migraciones que se han realizado en una instancia, incluye cuando se realizó la acción, computadoras que intervienen, estado, tipo de instancia antiguo y nuevo, tipo de migración, etc.

Una vez seleccionadas estas tablas serán extraídas de sus respectivas bases de datos para transferirlas a un nuevo esquema donde se empezará la construcción del modelo estrella desarrollado en el apartado 3.3 *"Construcción del modelo".*

3.3 Construcción del modelo

En este apartado se va a desarrollar que modelo se ha seleccionado para servir como estructura del Data Warehouse y como se ha construido.

En primer lugar, hay que pensar en por qué necesitamos un modelo dimensional y no un modelo relacional como tienen la mayoría de base de datos, así como el origen de nuestra fuente de datos.

Un modelo relacional (OLTP) es un modelo cuyo propósito es optimizar el almacenamiento de la información ya sea mediante una representación orientada a los datos o a los objetos. Estos modelos se encuentran normalizados para que funcionen de manera óptima en un sistema transaccional, es decir, poder realizar de una manera óptima todas las operaciones transaccionales (eliminar, insertar, actualizar y seleccionar) ya que se encuentran equilibradas en la estructura ofreciendo la misma importancia a todas ellas en cualquier momento. Pero este modelo presenta unas desventajas respecto a otros tipos de modelos orientados al análisis y es que al estar orientado a transacciones si el propósito del uso del modelo es realizar consultas sobre grandes cantidades de datos y multitud de tablas, la consulta será lenta o imposible de realizar, además de ser un sistema optimizado para recuperar o modificar un número reducido de tuplas además de tener por tanto un número reducido de dimensiones.

Por otro lado, están los modelos multidimensionales orientados al análisis y a las consultas de información. Estos modelos dan mayor importancia a las operaciones de selección de datos, ya que la inserción y actualización de datos solo se produce de manera periódica o única, y al ser por tanto un costo único, estos modelos se encuentran optimizados para recuperar grandes cantidades de datos.

Es por ello, que estos modelos multidimensionales son una de las bases de un Data Warehouse, ya que nos proporcionan la estructura para poder implementar un cubo OLAP, el cual se va a detallar en capítulos posteriores.

A continuación, se presenta una tabla a modo de resumen de las diferencias entre ambos modelos:

OLTP	Data Warehouse
Orientados a aplicaciones	Orientados a temas
Utilizados para mover el negocio	Utilizados para analizar el negocio
Usado por empleados comunes	Usado por ejecutivos y analistas
Contiene datos detallados	Contiene datos resumidos y refinados
Contiene datos actuales	Contiene datos históricos
Contiene datos aislados	Contiene datos integrados
Acceso repetitivo y transacciones pequeñas	Acceso a medida con consultas complejas
Acceso de lectura y escritura	Acceso básicamente de lectura Actualización por lotes
Ciclo de vida clásico	Ciclo de vida diferente
No hay redundancia	La redundancia es algo habitual

Tabla 2: Comparación OLTP vs DW

Ahora que sabemos que los Data Warehouse contienen modelos multidimensionales, es hora de elegir qué modelo contendrá el Data Warehouse de nuestro proyecto.

Y es que, dentro de los modelos multidimensionales, existen dos tipos de modelos principalmente: modelo en estrella y modelo en copo de nieve.

Un esquema en estrella como el mostrado en la siguiente figura, es una representación relacional de las tablas que estamos tratando, donde tenemos una tabla de central, también llamada tabla de hechos, rodeada de tablas de dimensión desnormalizadas.

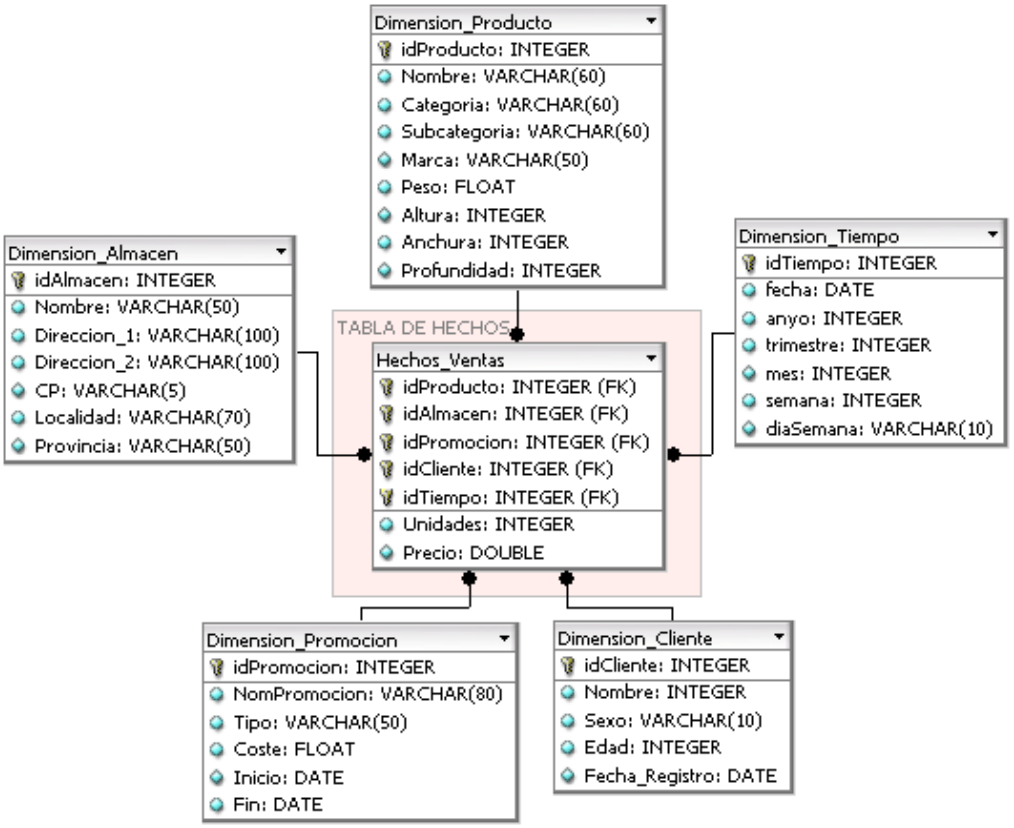


Figura 16: Ejemplo de esquema en estrella

Por otro lado, un modelo en copo de nieve como el de la figura 17 es un modelo en estrella normalizado en la 3FN [11].

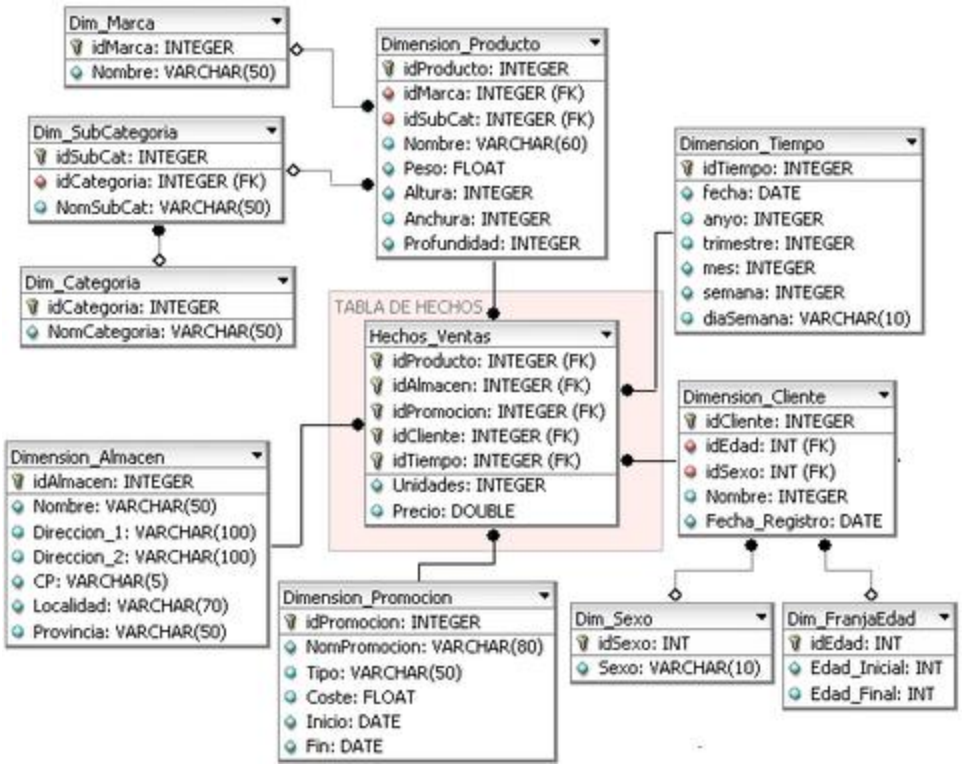


Figura 17: Ejemplo modelo copo de nieve

Como se puede apreciar en las figuras ambos modelos son extremadamente parecidos, ambos contienen una tabla de hechos central rodeada de dimensiones, siendo la diferencia que el modelo copo de nieve tiene las tablas de las dimensiones en 3FN.

La elección del modelo multidimensional para este proyecto será el modelo en estrella ya que es un modelo más sencillo donde la extracción de datos será más rápida y como se ha visto en el estudio de las bases de datos OpenStack no disponemos de tantas tablas donde sea posible aplicar la 3FN y por tanto conllevaremos la redundancia necesaria.

Una vez seleccionado el modelo hay que construirlo teniendo en cuenta las partes que tiene este modelo.

En el centro del modelo se encuentra la tabla de hechos que contiene en su interior los hechos que representan el objeto de análisis. Estos hechos pueden ser de evento (su ocurrencia es impredecible) o de estado (modelan el estado de un proceso en un momento determinado) siendo en el caso de este proyecto hechos de estado, ya que se modela el estado de un proceso en un momento determinado. Al ser el objeto de análisis es por ello por lo que será la tabla central.

Estos hechos tienen medidas asociadas, las cuales representan las propiedades de los hechos que quieren ser estudiados. Estas medidas son propiedades numéricas asociadas a los hechos, representando así las propiedades de este hecho. Las medidas tienen dos componentes: la propiedad numérica del hecho y la fórmula para combinar varios valores en uno (suma, agregación, recuento...)

Las medidas pueden ser de tres tipos: aditivas, semi aditivas o no aditivas. El tipo de medida de los hechos de nuestro proyecto será desarrollado más adelante cuando sean seleccionadas dichas medidas.

Por otro lado, rodeando la tabla de hechos, se encuentran las tablas de dimensión, que contienen atributos que ofrecen información característica de nuestro objeto de análisis. Estas dimensiones son usadas para seleccionar los datos y agrupar los datos al nivel de detalle deseado.

Las dimensiones se encargan de describir el "Quién, Qué, Dónde, Cuándo, Cómo y Por qué" de los eventos. Suelen tener muchas menos filas que las tablas de hechos, ya que contienen la descripción de los hechos, pero tienen gran cantidad de columnas para detallar esta información.

Las dimensiones se organizan en jerarquías, formadas por niveles. Cada uno de estos niveles representa un nivel de detalle para el análisis de datos. Un ejemplo de una jerarquía con dos niveles es la jerarquía realizada para una dimensión tiempo mostrada en la siguiente figura.

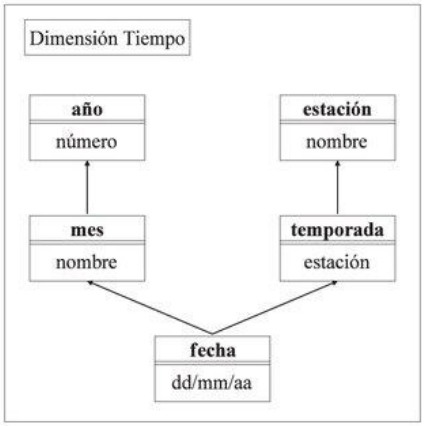


Figura 18: Ejemplo dimensión con dos niveles de detalle

Las instancias de estas dimensiones son conocidas como miembros.

Una vez se ha visto el tipo de modelo y sus componentes, a continuación, se va a desarrollar como se ha realizado este modelo para nuestro propósito.

En primera instancia, se ha seleccionado el proceso a modelar, ya que este será nuestra tabla de hechos, y como ya se ha comentado, el eje central del modelo.

Se podrían definir los procesos de la plataforma OpenStack como "Proyectos"," Usuarios"," Redes" e "Instancias".

Como el fin de este trabajo es proporcionar una herramienta donde poder analizar el uso de la plataforma y el principal uso de esta son la creación y utilización de instancias dentro de proyectos, se ha descartado en primer lugar que "Usuarios" y "Redes" sea el proceso para modelar.

Por tanto, se tiene dos propuestas interesantes para elegir como proceso que son "Proyectos" e "Instancias".

Se podría considerar realizar dos esquemas en estrella para dos Data Warehouse diferentes, pero tras un estudio de estos procesos se ha considerado que el proceso a modelar es "Instancias" ya que

su estudio es más interesante al tener más implicación en la plataforma debido a que un usuario constantemente crea y modifica instancias, mientras que el proyecto únicamente es el lugar donde se albergan estas instancias.

Lo siguiente que debemos establecer es la granularidad del proceso, es decir, que representa cada fila de la tabla de hechos y en nuestro caso es muy sencillo, ya que lo llevamos al máximo nivel de detalle posible gracias a la previa elección del proceso, teniendo una fila por cada nueva instancia generada.

Para definir este esquema necesitamos las dimensiones, que son las tablas que rodearán nuestra tabla de hechos y complementarán la información de nuestro proceso principal.

Han sido definidas 8 dimensiones para rodear la tabla de hechos:

- Dimensión usuario: Se trata de la información de cada usuario encontrado en la base de datos.
- Dimensión proyecto: Se trata de la unión de las tablas proyecto y grupos de seguridad de la red, encontramos aquí cada proyecto unido a los grupos de seguridad que han creado los usuarios en sus proyectos.
- Dimensión redes: Comprende toda la información de las redes, donde se une la propia red, con sus subredes, los routers creados en estas redes y por ultimo las direcciones dns de las subredes.
- Dimensión sabores: Contiene únicamente la tabla sabor donde se almacenan los sabores predefinidos por el sistema que puede utilizar cada usuario para su instancia.
- Dimensión instancia: Contiene únicamente la tabla instancia donde se encuentra toda la información de la creación de la instancia.
- Dimensión migraciones instancia: Esta dimensión almacena todas las migraciones realizadas por los usuarios a sus instancias.
- Dimensión fallos instancia: Esta dimensión almacena todos los fallos que han sufrido las instancias.
- Dimensión fecha: Contiene todas las fechas y el detalle de estas para el periodo de tiempo de creación, actualización y eliminación de las instancias.

La dimensión fecha es la única dimensión que no contiene tablas de las bases de datos OpenStack, y por tanto su diseño y su contenido debe ser realizado manualmente y será comentado en apartados posteriores.

Esta dimensión ha sido creada para poder tener una perspectiva temporal a la hora de analizar la fecha de las instancias además de ahorrar espacio en la tabla de hechos, ya que, a partir de ahora, las fechas encontradas en la tabla de hechos (Fecha de creación de la instancia, fecha de modificación de la instancia y fecha de eliminación de la instancia) pasarán a ser un numero entero que ha sido generado mediante un código propuesto para las fechas.

Por último, se encuentra la tabla de hechos, que contiene como columnas los campos que se unirán con cada dimensión, es decir, existe una columna por cada dimensión y estos campos son unidos mediante claves externas a las claves principales de dichas dimensiones y si hubiera, que no es el caso, las medidas asociadas a estos hechos, que como se ha explicado anteriormente son las propiedades numéricas que aportan un extra de información a la hora de analizar los hechos. Estas propiedades numéricas no son añadidas ahora, ya que serán creadas y modificadas a la hora de realizar el cubo OLAP en SSAS. Los hechos encontrados en la tabla de hechos representan el objeto de análisis

del modelo y pueden encontrarse hechos de evento cuya ocurrencia no se puede predecir como las ventas que se producen en una organización o hechos de estado que muestran cómo se encuentra un proceso en un momento determinado como los niveles de inventario de un almacén [12].

En resumen, en la figura 19 se puede ver la unión de estas dimensiones y la tabla hechos.

Cabe destacar que se han creado tres claves externas o tres relaciones entre la tabla de hechos y la dimensión fecha debido a que en la tabla de hechos han sido agregadas las fechas de creación, actualización y eliminación de la instancia.

Figura 19: Modelo estrella

Por tanto, tras la finalización de este apartado, se dispone de una base de datos que alberga un esquema estrella con el proceso de "Instancias" como hecho, y ochos dimensiones, que hacen un total de nueve tablas las cuales se encuentran vacías, es decir, sin datos en sus filas, y por tanto se debe rellenar mediante los procesos 3.4 ETL (Extraer, Transformar y Cargar) para empezar a construir el Data Warehouse.

3.4 Descripción del proceso ETL

En el apartado 3.3 fue construido el modelo estrella y con esto ahora tenemos dos bases de datos.

Por un lado, se tiene una base de datos transaccional que recoge las tablas extraídas de las bases de datos de OpenStack y por otro lado una base de datos que contiene el modelo estrella y es donde se encuentran las tablas vacías que rellenaremos en este apartado.

Por tanto, la siguiente gran etapa del proceso de construcción del Data Warehouse es el proceso de extracción, transformación y carga (ETL) que permite mover datos desde múltiples fuentes, reformatearlos y limpiarlos, para cargarlos en otra base de datos, o almacén de datos como sucederá en este proyecto.

A continuación, se presenta una explicación de la función de estos procesos.

El primer proceso de este conjunto de procesos es la extracción y consiste en conectar las fuentes de origen en las cuales se extraerán los datos que posteriormente serán usados en el Data Warehouse.

Para facilitar este proceso, se divide el mismo en tres fases. La primera fase consiste en analizar las necesidades del modelo para comprender los movimientos y transformaciones necesarias y así facilitar el trabajo. En la segunda fase se realiza una identificación de los archivos de los cuales se va a extraer los datos, donde se averiguará el tipo y formato de estos archivos. Por último, se realizará la extracción de los datos en función de estas necesidades previamente analizadas [13].

Respecto a la extracción de datos se puede realizar de tres maneras diferentes:

- Extracción total. Mediante esta extracción se consigue la totalidad de los datos ya que se barre por completo las tablas de las cuales se desea extraer la información.
- Extracción incremental. Se trata de una extracción similar a la total, ya que también se barren las tablas seleccionadas, pero se diferencian en que la extracción se va procesando por lotes de lo que ha sido modificado o agregado en la fuente de origen.
- Notificación de actualizaciones. Con esta extracción solo se extraen los datos cuando se produce una actualización.

La segunda etapa de los procesos ETL consiste en la transformación de datos, donde se seleccionan y modifican los datos que se cargarán en el Data Warehouse

Las acciones o procesos más habituales son “Reformateo de datos, conversión de unidades, selección de columnas, agregación de columnas, dividir una columna en varias, traducción de códigos, obtener nuevos valores calculados, unir datos de varias fuentes, etc”.

La última etapa del proceso es la carga de los datos extraídos y transformados en el Data Warehouse.

Es posible realizar la carga de dos maneras diferentes. La primera manera de realizar la carga es mediante “*Inserts*”, un sistema donde se mueven grandes bloques de datos de la manera más sencilla y común en la cual se puede realizar el proceso de carga, presentando el inconveniente de que si se trata de cargar una gran fuente de datos y se produce un corte de luz o un problema similar que pause la carga tendría que repetirse toda la carga de datos. La segunda manera de realizar la carga es mediante “*Loads*” dónde la carga se realiza de una manera escalonada y segura, donde el sistema se encarga de agrupar de forma automática la información según unas variables previamente

establecidas por el usuario. La ventaja que tiene esta manera de realizar la carga es que, si se produce un fallo durante el proceso, se puede retomar desde un punto en concreto, sin necesidad de repetir todo el proceso de nuevo como tendría que hacerse si se aplica el método de "Inserts" [14].

3.4.1 Migración base de datos

Debido a que se van a usar herramientas para los procesos ETL basadas en un motor SQL como SQL Server Data Tools (SSDT) para Visual Studio debido a la experiencia previa en esta herramienta y la preferencia personal en comparación a las herramientas ofrecidas para motores de MYSQL, es necesario migrar previamente la base de dato almacenada en MYSQL que contiene las tablas deseadas para realizar las dimensiones a SQL Server.

Para la migración de MySQL a SQL se usará la herramienta SSMA (Microsoft SQL Server Migration Assistant) versionado para MySQL.

El proyecto se encargará de migrar el esquema solicitado mediante una conexión a MySQL a la base de datos creada en Sql Server denominada "OpenStackTR" como representa la figura 20.

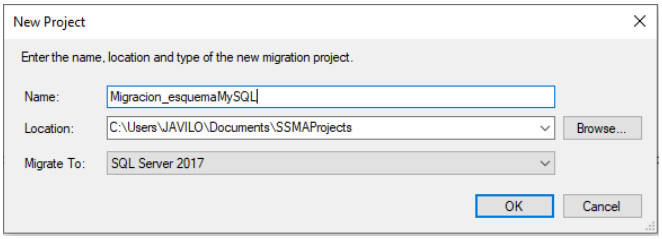


Figura 20: Creación proyecto SSMA

Para recoger todos los esquemas creados en MySQL se debe establecer e iniciar la conexión mediante el controlador ODBC para MySQL que ha sido instalado previamente.

Esta conexión es realizada también mediante un asistente de SSDT que solicita unos parámetros necesarios para realizar la conexión como se puede ver en la siguiente figura.

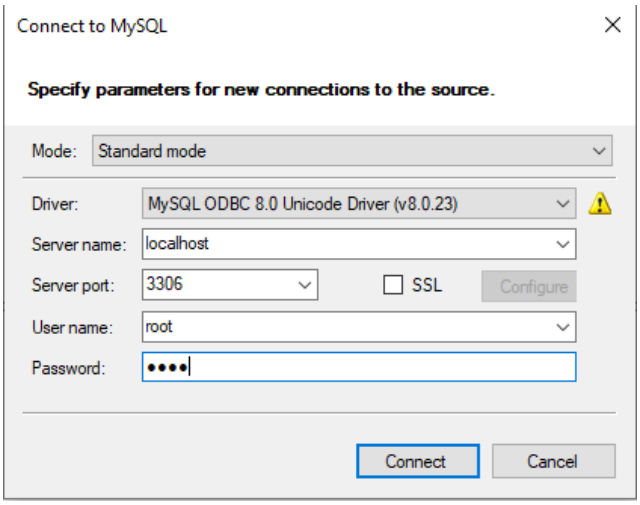


Figura 21: Conexión a MySQL

Esta conexión ofrece todos los esquemas que tenemos en el motor de MySQL, donde seleccionaremos los modelos de donde queremos extraer las tablas. Los modelos que presenta la

figura 22 son algunos de los modelos encontrados en nuestro motor, y donde encontramos los necesarios.

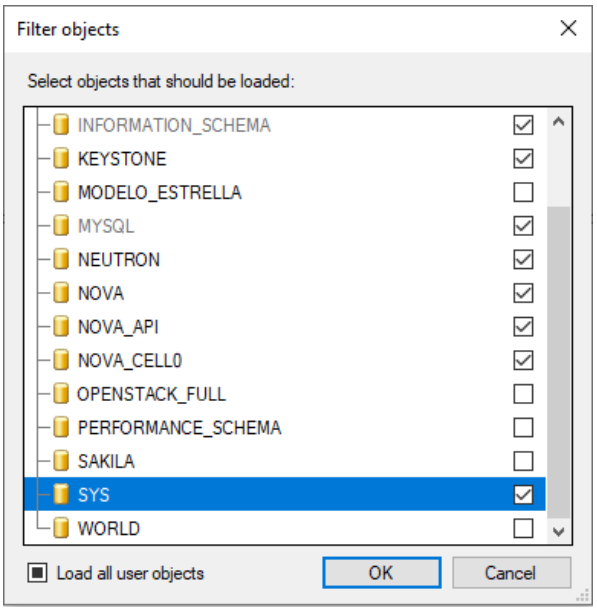


Figura 22: Modelos seleccionados de MySQL

Por otra parte, hay que realizar la conexión al destino, en este caso SQL Server y en esta conexión se debe elegir también la base de datos destino "OpenStackTR". Esta conexión se puede ver en la siguiente figura.

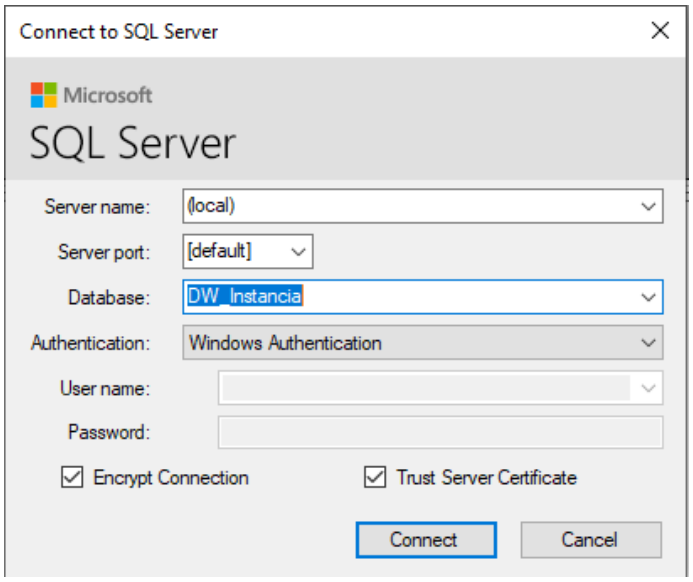


Figura 23: Conexión a SQL Server

Una vez se ha conectado origen y destino, se convertirá el esquema MySQL elegido y se sincronizará con la base de datos creada en SQL Server, como se puede ver en la figura 24, que a su vez es con la que hemos iniciado la conexión "OpenStackTR".

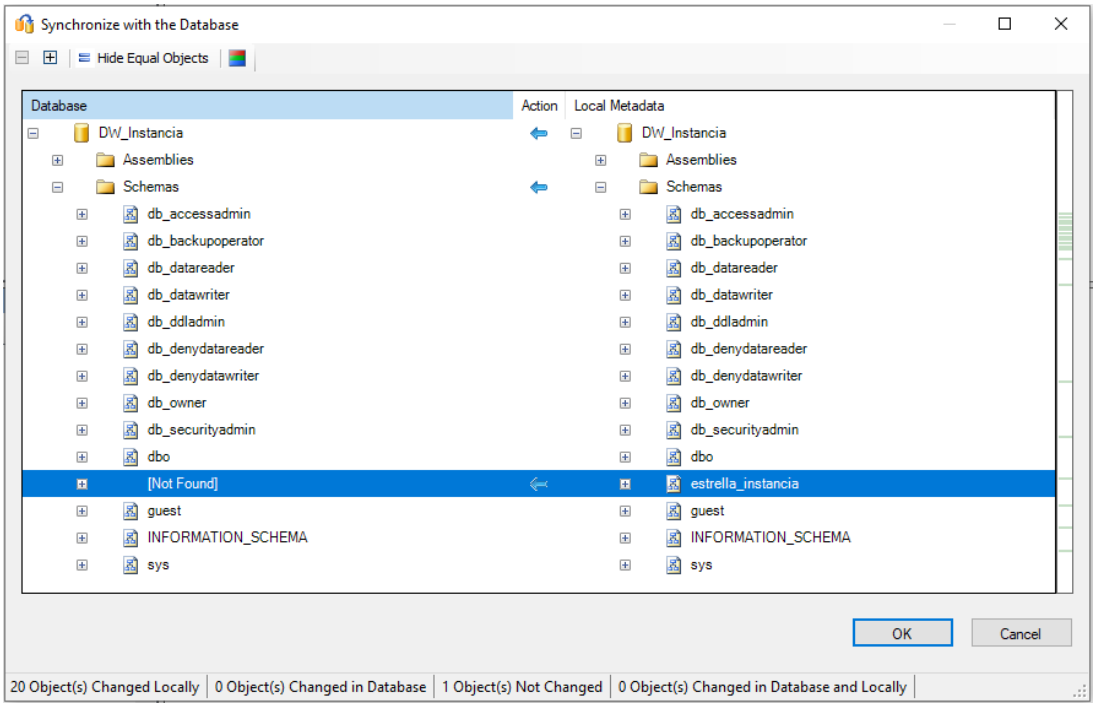


Figura 24: Sincronización entre bases de datos

Esta sincronización sirve para traspasar el esquema o esqueleto que tenemos en la base de datos de MySQL a un esquema vacío de una base de datos SQL.

Para realizar la carga de datos la herramienta SSMA nos ofrece una opción "Migrate Data" donde se vuelve a iniciar ambas conexiones y automáticamente migrará el contenido de las tablas del esquema MySQL a las nuevas tablas de la base de datos SQL.

Una vez migrados el esquema a SQL Server se pueden usar herramientas de tratamiento o visualización basado en este motor como SQL Server Management Studio o SSDT previamente comentado donde se continuará el proceso ETL contenido en este apartado.

3.4.2 Recapitulación de la situación de las bases de datos

Antes de desarrollar el proceso ETL hay que poner en situación las bases de datos que disponemos y de qué manera se pueden encontrar en SQL Server.

En primer lugar, se dispone de la base de datos "InstanciasDW" la cual alberga el modelo estrella y por tanto será la base de datos que haga la conexión en "Analysis Services" para la creación del Data Warehouse.

Esta base de datos solo contiene el esqueleto del modelo, es decir, las tablas y sus relaciones, no es una base de datos que posea datos ya que estos serán insertados en el desarrollo del siguiente apartado. Las tablas de esta base de datos se muestran en la figura 25.

Por otro lado, se dispone de la base de datos "OpenStackTR" que contiene las tablas necesarias para rellenar el modelo estrella, las cuales fueron citadas en el apartado 3.3 y migradas de MySQL en el apartado 3.4.1.

Esta base de datos necesita un tratamiento antes de poder ser usadas en el proceso ETL para rellenar el modelo estrella previamente comentado.

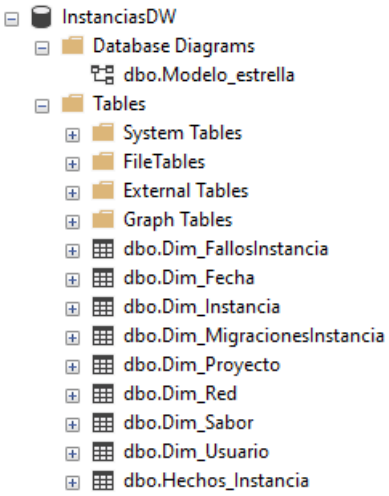


Figura 25: Base de datos que alberga el modelo estrella

El primer tratamiento es la creación de una tabla fecha, ya que esta no se encontraba en las bases de datos recibidas de la plataforma OpenStack, y es necesaria para extraer su contenido e insertarlo en la dimensión fecha.

El primer paso en la creación de una tabla de fechas es elegir la granularidad que en este caso será a nivel de días, ya que no existe la necesidad de profundizar en el máximo nivel de detalle el cual sería segundos ya que no se considera necesario para este esquema.

Los campos que tendrá esta tabla serán los campos necesarios para establecer una clave para la fecha y ofrecer información adicional sobre la misma y se pueden ver en la siguiente figura.

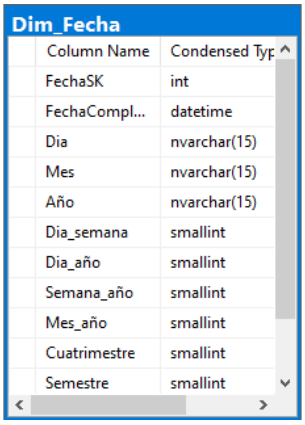


Figura 26: Diseño de la tabla fecha

Esta tabla no procede de la extracción de las bases de datos de la plataforma OpenStack, si no que su diseño ha sido creado manualmente como se ha mencionado anteriormente y por tanto su contenido es rellenado mediante un procedimiento.

Dicho procedimiento recoge las columnas de entrada creadas para la tabla “Dimensión Fecha” y establece como su salida un valor para cada columna. El valor más importante es la clave de la tabla,

el campo “FechaSK”, ya que este se relaciona con la tabla de hechos donde se encuentran varias fechas las cuales queremos convertir a números enteros para ganar espacio y rapidez en la base de datos.

Este campo es generado seleccionando cada parte de la fecha que está procesando el procedimiento y lo convierte en el formato “aaaammdd”. Los demás campos son generados mediante funciones sql que permiten coger el día, mes, año, etc de la fecha y con esto se le da información adicional y precisa a la fecha que se presenta.

Además, para no malgastar espacio en la base de datos, se ha establecido una fecha inicial y final con la que rellenar esta tabla cubriendo solamente las fechas de las cuales tenemos datos en las bases de datos OpenStack.

En el siguiente fragmento de código se puede encontrar el script SQL para realizar el procedimiento de crear y guardar las fechas en la tabla “Dim_Fecha” de la base de datos “OpenStackTR”

```
create procedure InserDate
   @CurrentDate datetime
as
insert into OpenStackTR.dbo.dim_fecha([FechaSK]
      ,[FechaCompleta]
      ,[Dia]
      ,[Mes]
      ,[Año]
      ,[Dia_semana]
      ,[Dia_año]
      ,[Semana_año]
      ,[Mes_año]
      ,[Cuatrimestre]
      ,[Semestre])
   values(
      (DATEPART(year , @CurrentDate) * 10000) + (DATEPART(month , @CurrentDate)*100)
       + DATEPART(day , @CurrentDate)
      , @CurrentDate
      , DATENAME(day , @CurrentDate)
      , DATENAME(mm, @CurrentDate)
      , DATENAME(yy, @CurrentDate)
      , DATEPART(dw , @CurrentDate)
      , DATEPART(DAYOFYEAR , @CurrentDate)
      , DATEPART(WW , @CurrentDate)
      , DATEPART(month , @CurrentDate)
      , DATEPART(quarter , @CurrentDate)
      ,  CASE WHEN DATEPART(quarter , @CurrentDate) < 3 THEN 1
           ELSE 2
         END
   )
GO
declare @StartDate datetime, @EndDate datetime

set @StartDate = '20170101'
--Cargamos el periodo de 3 años de existencia de datos
set @EndDate = '20201231'

while @StartDate <= @EndDate begin
   exec InserDate @StartDate

   set @StartDate = dateadd(day,1,@StartDate)

end
```

Fragmento 2: Procedimiento para insertar las fechas

El segundo tratamiento es la creación de tablas auxiliares para poder unir correctamente los identificadores de la tabla de hechos evitando además claves duplicadas y conocer la información adecuada de cada instancia.

Estas tablas han de ser creadas para las dimensiones que reciben una o más tablas, ya que, debido a esto, aumentaran las filas y por tanto el número máximo del identificador, y por lo que hay que crear

en una tabla la unión de las tablas necesarias para la correspondiente dimensión. El script SQL que realiza esta creación se encuentra en el fragmento 3.3

Esto es necesario de hacer para las tablas que forman la dimensión proyecto (formada por las tablas proyecto y grupos de seguridad), la dimensión usuario (formada por las tablas usuario y pares de claves) y la dimensión redes (formada por las tablas red, subred, router y dns)

```
USE OpenStackTR
SELECT pro.id as id_proyecto,pro.name as name_proyecto,
pro.description,pro.enabled,pro.domain_id,pro.parent_id,
pro.is_domain,sec.id as id_grupo,sec.name,sec.standard_attr_id
into proyectoaux
from keystone.project pro
inner join
neutron.securitygroups sec ON (pro.id=sec.project_id)
```

Fragmento 3: Ejemplo creación tabla auxiliar

El tercer tratamiento consiste en añadir un campo auto incremental en aquellas tablas cuyo identificador no es incremental, tiene valores repetidos o no lo posee como las tablas auxiliares recién creadas.

Esta acción se realiza manualmente modificando las propiedades de la tabla donde se desea añadir este valor, que para nuestra base de datos “OpenStackTR” será en todas las tablas, habiendo creado previamente un campo de tipo entero para establecerlo como “identity column” como muestra la siguiente figura que asigna el campo “id_aux_instancia” como “idendity”.

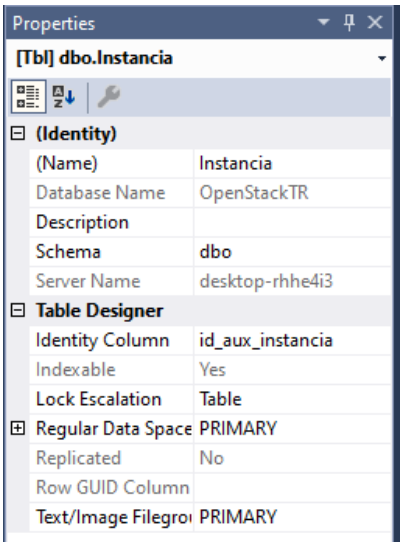


Figura 27: Creación campo identity

El último tratamiento es la inserción de filas cuyo id sea -1 ya que posteriormente se realizará una transformación donde los registros “NULL” de la tabla de hechos serán convertidos en -1, y por tanto es necesario tener una coincidencia de ese identificador en la dimensión consultada.

Estos registros han de ser insertados manualmente mediante una actualización de la tabla y solo serán insertados en aquellas tablas que lo necesiten, y el contenido de sus columnas dará a entender que el valor correspondiente es nulo, ya sea por ser desconocido o por no existir correspondencia.

La siguiente figura muestra los registros de una de las dimensiones donde se ha insertado una fila con id -1.

id	created_at	name	memory_mb	vcpus	swap	flavorid	rxtx_factor	root_gb	ephemeral_gb	disabled	is_public
-1	9999-12-31 00:00:00	Nulo	0	0	0	0	0	0	0	1	0
1	2017-09-27 22:35:47	tiny	512	1	0	6270f052-12d1-43ad-8548-f9b38eed17f3	1	1	0	0	1
2	2017-09-27 22:35:52	small	2048	1	0	a891976a-e76a-4e39-810b-5de22a8513e2	1	20	0	0	1
3	2017-09-27 22:35:56	medium	4096	2	0	1b06c408-0ab0-467c-ab92-d46d15db591d	1	40	0	0	1
4	2017-09-27 22:35:59	large	8192	4	0	30423ae0-c0a1-43ec-9ca8-e1894724c188	1	80	0	0	1
5	2017-09-27 22:36:04	xlarge	16384	8	0	dbbe6b04-ef49-4dad-b0df-4ad52de2c6ed	1	160	0	0	1
6	9999-12-31 00:00:00	N/A	0	0	0	0	0	0	0	0	0
7	2017-09-30 07:34:14	di.02core02RAM80HD	2048	2	0	39831a6b-2fb8-4f68-a4a5-aa376f2d51e9	1	80	0	0	1
8	2017-10-02 19:38:39	di.02core04RAM60HD	4096	2	0	0bfed370-bb3b-4533-b923-d810b1da35a9	1	60	0	0	1
9	2017-11-29 11:30:23	BigData	16384	4	0	6beff4f2-3065-43fe-8e98-22b37d5e5ea1	1	60	0	0	0
11	2018-07-25 09:54:09	p2psp	1024	1	0	7b3a2fcd-212b-4b27-844d-8acca2b09277	1	4	0	0	1
12	9999-12-31 00:00:00	N/A	0	0	0	0	0	0	0	0	0
13	9999-12-31 00:00:00	N/A	0	0	0	0	0	0	0	0	0
14	9999-12-31 00:00:00	N/A	0	0	0	0	0	0	0	0	0
15	2018-12-29 13:19:31	ocp.master	4096	2	0	2798cb8f-4a85-425a-a157-1e995833de86	1	30	0	0	0
16	2018-12-29 13:20:05	ocp.infra	16385	4	0	f0bb443c-5dff-42d3-8776-6e9b2e26f7cc	1	30	0	0	0
17	2018-12-29 13:20:38	ocp.node	4096	2	0	74afc18d-10ab-4a5b-a5e3-844ec05c2aff	1	30	0	0	0
18	2018-12-29 13:21:01	ocp.bastion	4096	1	0	a0474894-27de-46d7-8444-89aeb1ba5d42	1	30	0	0	0
19	9999-12-31 00:00:00	N/A	0	0	0	0	0	0	0	0	0

Figura 28: Ejemplo de tabla con id "-1"

Una vez creadas las tablas auxiliares para la base de datos “OpenStackTR” ya se encuentra preparada dicha base de datos para el proceso ETL. La estructura final de la base de datos se muestra en la siguiente figura.

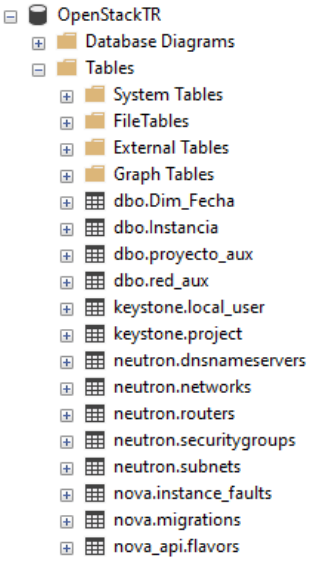


Figura 29: Base de datos que alberga el contenido transaccional

3.4.3 Desarrollo del proceso ETL

Una vez se tienen ambas bases de datos preparadas, es posible empezar el desarrollo de los procesos comentados en el apartado 3.4 donde se puede resumir la existencia de tres grandes procesos que le dan nombre a este apartado: Extracción, Transformación y Carga.

Este desarrollo se hará posible con la ayuda de la herramienta "Integration Services" ofrecida por SSDT en Visual Studio donde es posible realizar una plantilla para desarrollar la solución del modelo multidimensional.

Una vez creado el proyecto, Visual Studio presenta la interfaz mostrada en la figura 30. Esta interfaz muestra los elementos más importantes que ofrece SSIS, donde se encuentra el explorador de soluciones a la derecha que permite crear el paquete que usaremos para la creación del paquete que contiene el proceso ETL. En la parte izquierda de la interfaz se encuentran las tareas que se pueden realizar en el flujo de control, pudiendo mostrar las tareas del flujo de datos cuando se selecciona el mismo. Por último, en la parte central se encuentra una ventana donde introducir las tareas, modificarlas, relacionarlas, etc.

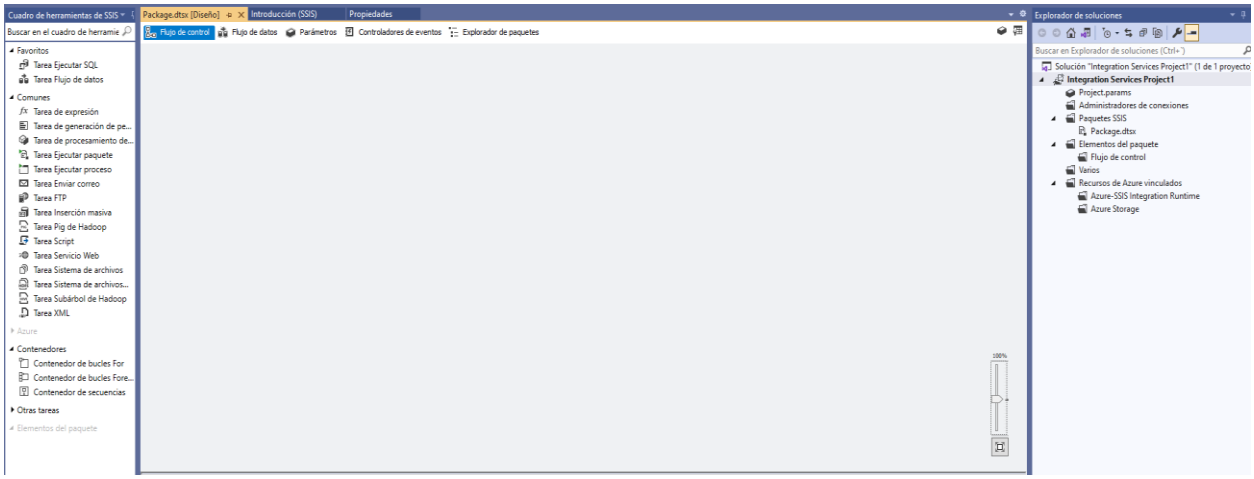


Figura 30: Interfaz SSIS

Las tareas de flujo de control proporcionan la funcionalidad y restricciones de precedencia que conectan los ejecutables, mientras que las tareas de flujo de datos permiten mover los datos entre orígenes y destinos, además de tener tareas para transformar, limpiar y modificar datos a medida que se mueven.

Para realizar el proceso ETL, se ha creado un paquete SSIS que contendrá estos procesos. El contenido de este paquete SSIS se irá actualizando a medida que se realicen los procesos ETL.

3.4.3.1 Extracción

La extracción es la parte inicial del proceso, y el proceso consiste en extraer los datos que se van a utilizar desde una o más fuentes, siendo para este proyecto una única fuente, ya que las condiciones requerían juntar las tablas en la base de datos "OpenStackTR"

El primer paso que realizar es la identificación de la fuente, que en este caso es la base de datos "OpenStackTR" la cual es la unión, escogiendo lo necesario, de varias bases de datos relacionales, provocando que esta base final sea una base de datos no relacional. Esta base de datos se encuentra almacenada en SQL Server por lo que habría que realizar una conexión OLE DB a SQL Sever Native Cliente.

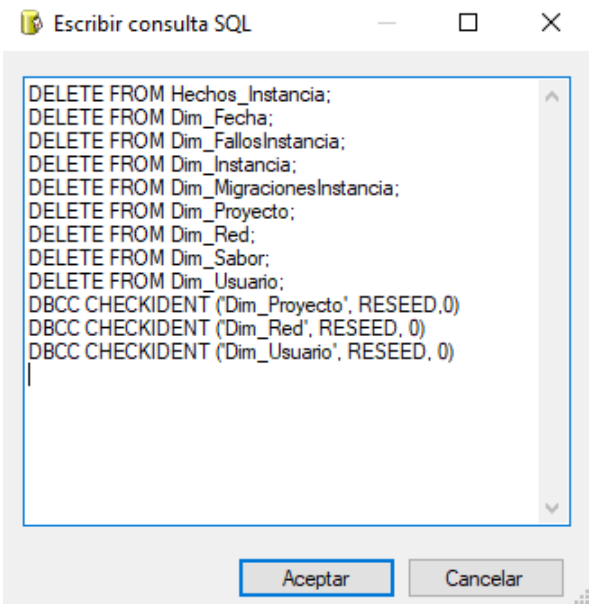
Antes de realizar la extracción de datos hay que determinar cuál de los tres modos de extracción queremos incluir en nuestro proceso. Estos modos fueron detallados en el apartado 3.4 donde se puede resumir la existencia de tres modos: Extracción total, extracción incremental y notificación de actualizaciones.

Debido a que este proyecto se realiza por primera vez y no se pretende incluir más datos por el momento, además de no contar con tablas con una gran cantidad de registros, se ha llegado a la conclusión de que el mejor modo de extracción para este proyecto es la extracción total.

Esta modalidad consiste en extraer la totalidad de los datos, barriendo todas las tablas seleccionadas completamente.

Una vez se ha concretado como realizar el primer paso del proceso ETL hay que trasladarlo a la herramienta de Visual Studio.

En primer lugar, para evitar errores a la hora de insertar registros si se desea realizar de nuevo el proceso, ya sea por tener nuevos registros u otro motivo, es necesario tener una primera tarea de limpieza de tablas, donde se elimine el contenido que poseen las mismas y se reinicie el identificador de aquellas que poseen un identificador auto incremental. Este script SQL consiste en una tarea de ejecución SQL con el contenido que se muestra en la siguiente figura.



Fragmento 4: Consulta SQL para limpiar las tablas

Por otro lado, para realizar la extracción de las tablas del origen de datos es necesario crear una tarea de flujo de datos para cada dimensión y tabla de hechos en la que se quieren insertar registros y dentro de esta tarea de flujo de datos, facilitando así las siguientes etapas del proceso.

Dentro de cada tarea de flujo de datos, la primera etapa será extraer los datos con una conexión a la base de datos “OpenStackTR” que es el origen de los datos del proyecto mediante el controlador OLE DB para SQL Sever Native.

Esta extracción se realiza con la tarea denominada “Origen de OLE DB” la cual permite seleccionar una conexión, para este proyecto la mostrada arriba, y elegir el modo de acceso a los datos que contiene esta conexión a la base de datos como se muestra en la siguiente figura.

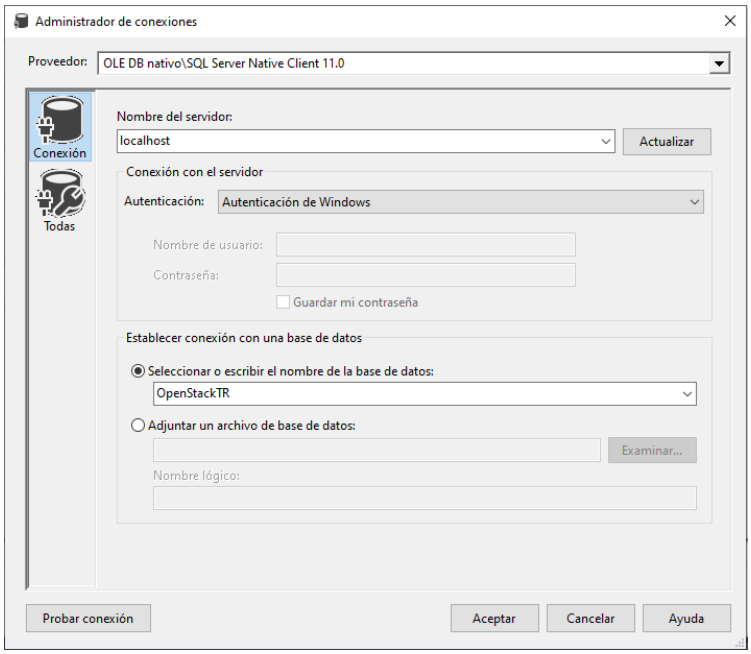


Figura 31: Conexión al origen de datos

Respecto al modo de acceder a los datos, se puede acceder de cuatro maneras diferentes: Tabla o vista, comando SQL, variable de nombre de tabla o nombre de vista y comando de SQL variable.

Debido a que para realizar la extracción de datos a las dimensiones hay que realizar uniones entre varias tablas, realizar pequeñas transformaciones en algunos campos y seleccionar las columnas deseadas se accederá mediante comandos sql, como el presentando en la siguiente figura que se encarga de extraer los datos de las tablas proyecto y grupos de seguridad.

```
SELECT  DISTINCT pro.[id] as id_proyecto
    ,pro.[name] as name_proyecto
    ,pro.[description]
    ,pro.[enabled]
    ,pro.[domain_id]
    ,pro.[parent_id]
    ,pro.[is_domain]
    ,sec.[id] as id_gruposec
    ,sec.[name] as name_gruposec
    ,sec.[standard_attr_id]
  FROM [keystone].[project] pro LEFT JOIN [neutron].[securitygroups]
sec ON(pro.id=sec.project_id)
```

Figura 32: Comando SQL de extracción

Este comando sql, en profundidad, consiste en unir dos tablas mediante un “left join” el cual devolverá todos los resultados de la tabla proyecto, ya que nos interesa tener todas sus filas, además de insertar los resultados coincidentes con las claves de grupo de seguridad. Debido a este uso de “left join” algunos campos tendrán valores “NULL” y será necesario tenerlo en cuenta en el proceso de transformación.

Además, la función “select” selecciona aquellos registros deseados, que en este caso son aquellos cuyas columnas tienen registros, y por tanto aquellas columnas que se encuentran rellenadas solamente de registros “NULL” son desechadas en este proceso.

El resto de tareas donde se extraen los datos para las dimensiones son similares a la descrita, pero con ciertas excepciones, una excepción es la modificación del campo fecha recibido, ya que el campo fecha contiene tanto la fecha en formato fecha y hora, pero en este proyecto no necesitamos la hora ya que la dimensión fecha ha sido construida sin horas. Por tanto, para poder comparar fechas es necesario que la hora se encuentre a 0 y es de lo que se encarga la conversión propuesta cuando se llama al campo fecha. Esta conversión recibe el campo fecha con la hora y devuelve el campo con la hora a 0 y se realiza como se muestra en la siguiente figura.

```
SELECT DISTINCT CAST(CONVERT(NVARCHAR,created_at,112)AS
DATETIME)  AS created_at_sabor
,[id]
    ,[name]
    ,[memory_mb]
    ,[vcpus]
    ,[swap]
    ,[flavorid]
    ,[rxtx_factor]
    ,[root_gb]
    ,[ephemeral_gb]
    ,[disabled]
    ,[is_public]
 FROM [nova_api].[flavors]
```

Fragmento 5: Comando SQL de extracción modificando el campo fecha.

Otra excepción es que la dimensión de redes realiza su unión mediante “inner join” ya que lo que interesa es tener toda la información de redes al completo.

La vista general de una tarea de extracción se muestra en la siguiente figura, donde encontramos la conexión OLE DB a la base de datos de origen, el modo de acceso mediante comando SQL, y, por último, el propio comando SQL que se encarga de extraer y unir las tablas de origen. Todas estas secciones se muestran en la siguiente figura.

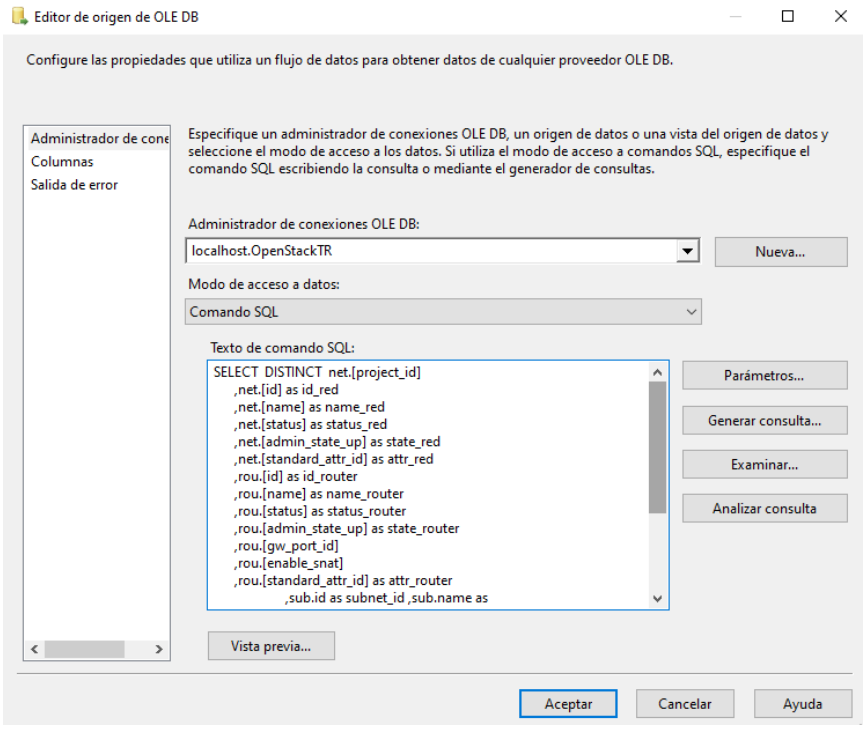


Figura 33: Ejemplo origen OLE DB

Por otra parte, para la tarea de origen OLE DB que se encarga de extraer los datos para la tabla de hechos, se realiza el mismo proceso, pero con una diferencia en los comandos sql que cogen los datos ya que hay que tener ciertas consideraciones y el volumen de tablas a unir aumenta considerablemente.

Como se ha visto anteriormente, la tabla de hechos tiene los identificadores de todas las dimensiones por lo que hay que realizar uniones con todas las tablas de cada dimensión y es aquí donde entran las tablas auxiliares realizadas en el apartado 3.4.2 donde se sacarán algunos identificadores para la tabla de hechos. También se podría incluir en este apartado las medidas, pero no se ha visto ninguna propiedad numérica que sea aditiva o semi aditiva para esta tabla de hechos, por lo que las medidas se formarán junto a la creación del cubo.

El fragmento 6 contempla el código donde se extraen todos los identificadores necesarios para la tabla de hechos, aunque también es posible encontrar el código en el siguiente enlace: Enlace código extracción tabla de hechos. Estos identificadores provienen de las tablas encontradas en "OpenStackTR" y son obtenidos comparando los identificadores en común de las tablas.

Como se puede apreciar encontramos dos tipos de uniones, "inner join y left join". Cuando se utiliza "inner join" estamos buscando las coincidencias de los identificadores en ambas tablas, es decir estamos buscando instancias cuyo proyecto, usuario y red se encuentra en las tablas de dimensión de proyecto, usuario y red.

```
SELECT insN.id_aux_instancia as id_instancia, proN.id_proyecto_aux as id_proyecto,userN.id_aux_auto as id_usuario,netN.id_red_auto as id_red,fechaN.FechaSK as fechaCreacion
,CONVERT(varchar,insN.updated_at_instancia,112) as fechaActualizacion,CONVERT(varchar,insN.deleted_at_instancia,112) as fechaEliminacion,migN.id as id_migracion,
faultN.id as id_falloIns, insN.instance_type_id_instancia as id_sabor

FROM OpenStackTR.dbo.Instancia insN INNER JOIN OpenStackTR.dbo.proyecto_aux proN ON(insN.project_id_instancia=proN.id_proyecto)
                                    INNER JOIN OpenStackTR.keystone.local_user userN ON(insN.user_id_instancia=userN.user_id)
                                    INNER JOIN OpenStackTR.dbo.red_aux netN ON(insN.project_id_instancia=netN.project_id)
                                    LEFT JOIN OpenStackTR.nova.migrations migN ON(insN.uuid_instancia=migN.instance_uuid)
                                    INNER JOIN OpenStackTR.dbo.Dim_Fecha fechaN ON(CAST(CONVERT(NVARCHAR,insN.created_at_instancia,112)AS DATETIME)=fechaN.FechaCompleta)
                                    LEFT JOIN OpenStackTR.nova.instance_faults faultN ON(insN.uuid_instancia=faultN.instance_uuid)
```

Fragmento 6: Extracción para la tabla de hechos

Por otro lado, cuando se utiliza "left join" se le da prioridad al identificador de la izquierda, en este caso el identificador de la instancia, frente al identificador de migración o fallo. Este uso de left join es debido a que no todas las instancias tienen un fallo o migración, y lo que interesa es tener todas las instancias posibles sin necesidad de que incluyan un fallo o migración.

Debido a esto se ha de tener en cuenta, que cuando se realiza un "left join" las filas que no tengan coincidencian pasarán a tener un valor nulo y habrá que realizar una transformación para asignarles un valor que se pueda encontrar en las dimensiones de migración y fallo.

Con esto se obtendrían todos los registros de todas las tablas necesarias para rellenar el modelo estrella y a continuación habría que estudiar si estos registros necesitan algún tipo de transformación.

3.4.3.2 Transformación

La transformación es la segunda parte del proceso ETL y consiste en aplicar una serie de funciones o reglas de negocio sobre los datos extraídos para convertirlo en datos que se puedan entender o manejar según las necesidades del proyecto para ser posteriormente cargados en la nueva fuente.

Las acciones o procesos más habituales son:

- Reformateo de datos: Este proceso se realizará en nuestro proyecto debido a que el formato de las fechas cambia entre la tabla origen y destino.

- Conversión de unidades: Este proceso no hace falta realizarlo ya que solamente existen dos campos que contienen valores numéricos con unidades y se encuentran en la unidad deseada.
- Selección de columnas para su carga posterior: Esta transformación se realizará en este proyecto ya que consiste en hacer que las columnas con valores nulos no se carguen, y en nuestras tablas hay varias columnas con dichos valores.
- División de una columna en varias: Esta acción no se realizará en este proyecto debido a que normalmente se realiza sobre nombres, y en nuestra base de datos, los nombres vienen cifrados por lo que no podemos separar las columnas
- Traducción de códigos: Este proceso consiste en cambiar por valores enteros, el código almacenado en la columna. Dicha acción no se realizará en este proyecto debido a que no se considera de utilidad según las columnas disponibles.
- Transformación de valores nulos: Este proceso se realizará en este proyecto debido a que existe columnas que mezclan valores conocidos y valores nulos.

Existen más transformaciones habituales como la agregación de columnas, obtención de valores calculados, agrupación de filas y lookups que no son comentadas en este apartado debido a que serán comentadas en la construcción del backend.

Respecto a las transformaciones comentadas que se puede aplicar a nuestro proyecto tenemos el reformateo de datos, la selección de columnas y la transformación de valores nulos.

La selección de columnas es un proceso que ya ha sido realizado en la extracción, debido a que en el comando sql al llamar a las tablas de origen, hay que elegir que columnas queremos añadir. En este proceso han sido eliminadas columnas en las que mínimo el 90% de sus registros eran “NULL” y han sido seleccionadas el resto de las columnas que aportaban valor, incluso teniendo registros “NULL” en algunas filas.

Una vez seleccionadas las columnas, la siguiente etapa es reformatear los datos para que el formato de los valores sea el mismo en la tabla de origen que destino. En esta ocasión, solo hay que reformatear las columnas que contienen el valor fecha ya que están en un formato diferente al que se quiere obtener en las fechas destino.

Este proceso se realiza mediante la tarea “Conversión de datos” que recibe como entrada la columna deseada a formatear, establece el nuevo tipo de datos siempre que la conversión de datos sea posible, y devuelve como salida la columna reformateada. El proceso se muestra en la figura 32.

Columna de entrada	Alias de salida	Tipo de datos
created_at_instancia_sin	Copia de created_at_i...	fecha [DT_DATE]
updated_at_instancia_...	Copia de updated_at_i...	fecha [DT_DATE]

Figura 34: Ejemplo de reformateo de datos

Por otro lado, hay que tratar con las columnas seleccionadas que tienen registros “NULL” mediante una transformación del valor. Esta transformación se puede realizar gracias a la tarea “columna derivada” que permite insertar funciones en las columnas. Una de las funciones disponibles es ‘REPLACENULL’ que recibe el valor de una columna, comprueba si este valor es nulo, y si lo es, reemplaza este valor nulo, por el valor deseado introducido en la expresión.

Como se puede ver en la figura 34, la expresión recibe la columna introducida, y si es nulo, cambia su valor por un 0 debido a que el resto de los valores son numéricos [15].

Nombre de columna d...	Columna derivada	Expresión	Tipo de datos
gw_port_id	Reemplazar 'gw_port_id'	REPLACENULL(gw_port_id,0)	cadena Unicode [DT_W...
gateway_ip	Reemplazar 'gateway_ip'	REPLACENULL(gateway_ip,0)	cadena Unicode [DT_W...

Figura 35: Primer ejemplo de transformación de valores nulos

Por otro lado, si el resto de los valores no nulos de una columna no son enteros, la expresión reemplazaría el valor "NULL" por el valor "N/A" el cual quiere expresar que el valor es desconocido o no aplicable como muestra la siguiente figura.

Nombre de columna d...	Columna derivada	Expresión
image_ref_instancia	Reemplazar 'image_ref...	REPLACENULL(image_ref_instancia,"N/A")
key_name_instancia	Reemplazar 'key_name...	REPLACENULL(key_name_instancia,"N/A")
host_instancia	Reemplazar 'host_insta...	REPLACENULL(host_instancia,"N/A")
availability_zone_insta...	Reemplazar 'availabilit...	REPLACENULL(availability_zone_instancia,"N/A")
node_instancia	Reemplazar 'node_inst...	REPLACENULL(node_instancia,"N/A")

Figura 36: Segundo ejemplo de transformación de valores nulos

También hay que tener en cuenta que puede haber valores nulos en la fecha, y en este caso la expresión reemplazaría el valor "NULL" una fecha lejana o temprana en el tiempo como "9999-12-31", como se puede ver en la siguiente figura.

Copia de launched_at_i...	Reemplazar 'Copia de l...	REPLACENULL([Copia de launched_at_instancia_sin],"9999-12-31")
Copia de terminated_a...	Reemplazar 'Copia de t...	REPLACENULL([Copia de terminated_at_instancia_sin],"9999-12-31")
Copia de deleted_at_in...	Reemplazar 'Copia de ...	REPLACENULL([Copia de deleted_at_instancia_sin],"9999-12-31")

Figura 37: Tercer ejemplo de transformación de valores nulos

Por último, en esta transformación de valores nulos, se encuentran los valores desconocidos de la tabla de hechos y en estos casos se utiliza el valor '-1', el cual ha sido introducido como identificador en aquellas dimensiones necesarias. Esta transformación se puede ver en la figura 36.

id_migracion	Reemplazar 'id_migraci...	REPLACENULL(id_migracion,-1)
id_sabor	Reemplazar 'id_sabor'	REPLACENULL(id_sabor,-1)
id_fallolns	Reemplazar 'id_fallolns'	REPLACENULL(id_fallolns,-1)

Figura 38: Transformación de los identificadores nulos

Este tratamiento de valores nulos se ha realizado principalmente para evitar violar la integridad referencial entre dimensiones y hechos y, además, para facilitar el análisis de datos.

3.4.3.3 Carga

La última etapa del proceso ETL es la carga, etapa en la cual los datos procedentes de la fase de transformación son cargados en el sistema de destino.

Como se expuso en el apartado 3.4 la carga se realizará mediante el tipo de carga "insert" donde acumularemos todos los datos y serán insertados en la base de datos destino "InstanciaDW". Ha sido elegido este método debido a que al tener un volumen pequeño de datos, las operaciones del proceso ETL se realizan en un breve periodo de tiempo, y al ser un método más sencillo de construir se ha seleccionado este modo de carga.

Para realizar la carga se ha utilizado otra tarea dentro del flujo de datos de cada dimensión o tabla de hechos que será de tipo "Destino de OLE DB" que realizará una conexión SQL Server Native Cliente con la base de datos destino "InstanciaDW".

En esta tarea el modo de acceso a los datos es mediante carga rápida de la tabla o vista donde solo es necesario establecer la tabla a la cual se insertarían los registros y relacionar las columnas de origen y destino como se ve en la siguiente figura donde se carga la tabla "Dim_Instancia".

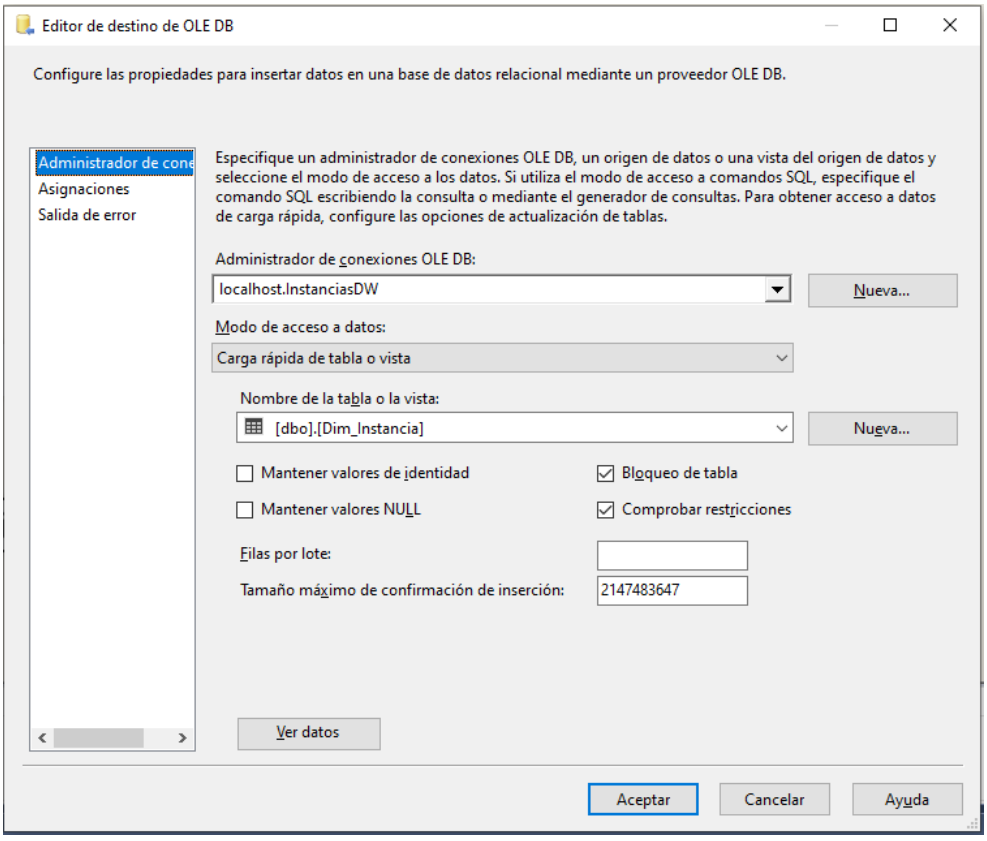


Figura 39: Ejemplo destino OLE DB

En las asignaciones de columnas se han de seleccionar aquellas columnas transformadas ya que el proceso ya ha sido realizado, y el resto son las columnas derivadas del proceso de selección.

Este proceso de selección es un apartado del editor de destino OLE DB, y su interfaz y columnas se pueden ver en la siguiente figura.

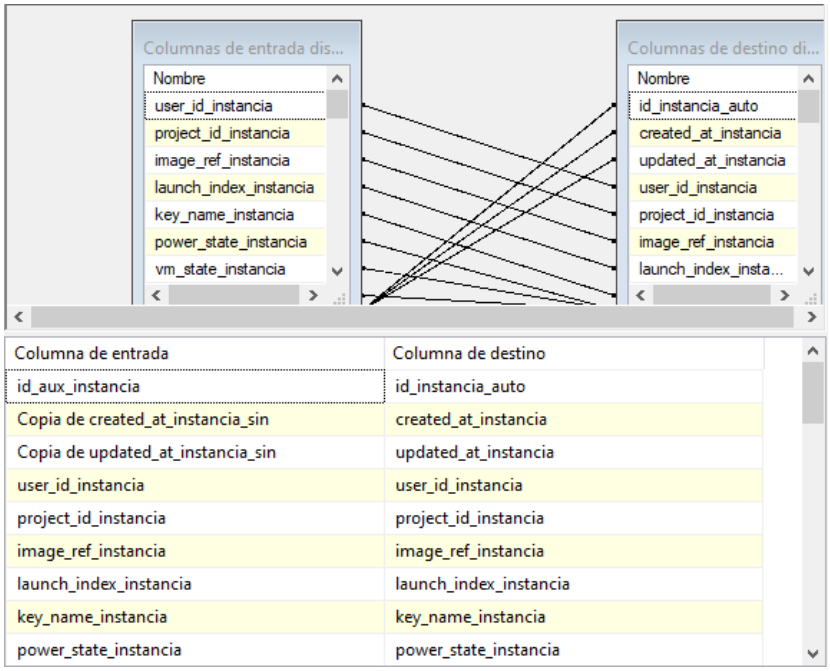


Figura 40: Ejemplo de asignación de columnas destino

Este proceso se repite en cada tarea de flujo de datos de dimensión y tabla de hechos, quedando por tanto el esquema mostrado en la figura 41 para cada tarea.

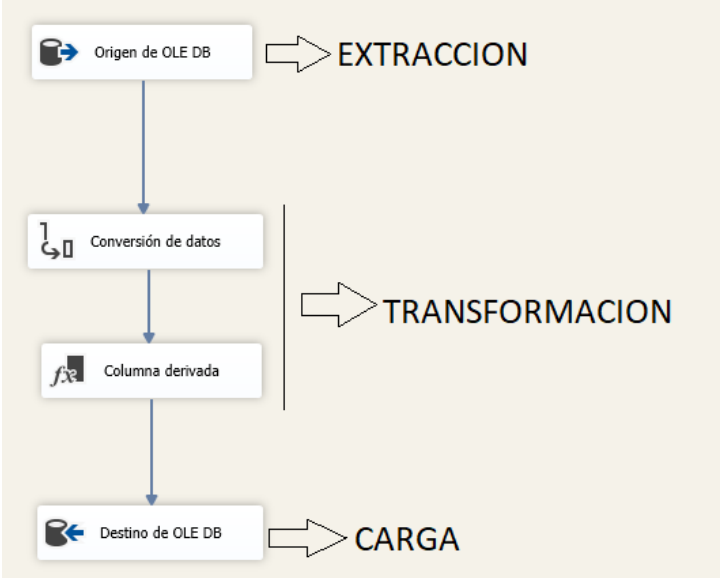


Figura 41: Proceso ETL en una tarea de flujo de datos

Una vez finalizada cada tarea de flujo de datos, es posible unir todas las tareas para automatizar el proceso ETL y ejecutar cada paquete cuando otro ha finalizado, evitando así que se carguen todas las tablas si una tarea ha fallado.

Resumiendo para finalizar este apartado del proceso ETL, en la figura 42 se encuentra el proceso de flujo final, el cual empieza con una tarea sql que se encarga de la limpieza comentada previamente, si esta tarea se ejecuta con éxito, se ejecuta cada tarea de flujo de datos de la dimensión, donde dentro de cada tarea de flujo de datos encontramos el flujo encontrado en la figura 41, y por último si todas

las tareas de las dimensiones han sido completadas con éxito, se ejecutará la tarea de la tabla de hechos que rellena todos los identificadores relacionados con las dimensiones terminado el proceso.

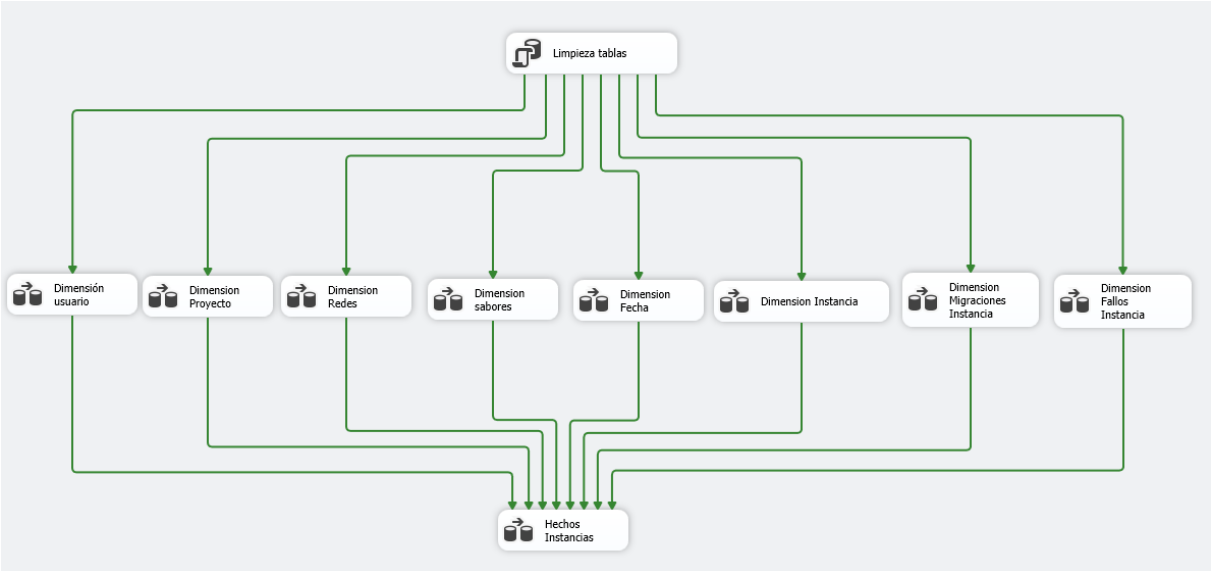


Figura 42: Flujo de control proceso ETL

3.5 Construcción del BackEnd

La construcción del BackEnd de este proyecto consiste en montar una base de datos multidimensional dentro de la estructura del Data Warehouse, fuertemente diferenciada de una base de datos relacional en su forma de guardar y procesar la información, encontrando la principal diferencia en la estructura que forman las tablas, estudiada y realizado en el apartado 3.3 de este documento donde se realizó el modelo estrella y se destacaron las diferencias con las estructuras de las bases de datos relacionales.

Las tablas de las bases de datos multidimensionales se organizan en forma de cubo, concretamente denominado cubo OLAP (On-Line Analytical Processing). En este tipo de estructura, cada dimensión del cubo equivale a un campo de dimensiones en la tabla, mientras que la información almacenada en cada celda del cubo corresponde a los hechos.

Existen tres tipos de implementaciones OLAP:

- ROLAP (OLAP Relacional): Los datos que almacena esta base de datos son relacionales. Este sistema es adecuado en:
    - Escalabilidad en el número de hechos
    - Redefiniciones del cubo
    - Actualizaciones frecuentes
- MOLAP (OLAP Multidimensional): Los datos se almacenan en estructuras multidimensionales especializadas. Estos sistemas manejan eficientemente cubos y son por tanto eficientes en consultas y en la gestión del almacenamiento.
- HOLAP (OLAP Híbrido): Es un sistema hibrido ya que por una parte contiene los datos resumidos y agregados en MOLAP y por otra parte los datos detallados en ROLAP.

En este proyecto se va a utilizar la implementación MOLAP debido a que se considera la mejor opción según el diseño de las bases de datos.

Esta implementación MOLAP, por tanto, se compone de unas tablas de dimensión y en este proyecto, una tabla de hechos, desarrollado en el apartado 3.3, unas medidas correspondientes a los valores numéricos de la tabla de hechos, las jerarquías que permiten organizar la base de datos multidimensional en base a diferentes niveles de abstracción, y, por último, el cubo como resultado final de la combinación de estos elementos.

Para este desarrollo se usará la herramienta Analysis Services ofrecida por SQL Server Data Tools en Visual Studio, donde se creará un proyecto de tipo “Proyecto multidimensional y de minería de datos de Analysis Services”.

Una vez creado el proyecto, hay que conectarlo con el modelo estrella que contiene la estructura y los datos introducidos en el apartado 3.4, ofreciendo Analysis Services un apartado dentro del proyecto denominado “Origen de datos”, el cual permite realizar una conexión con la base de datos deseada como se muestra en la siguiente figura que se realiza la conexión con “InstanciasDW”.

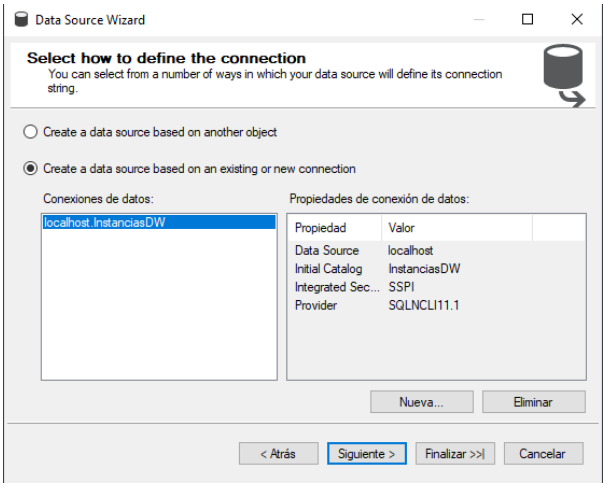


Figura 43: Conexión origen de datos

Una vez tenemos en nuestro proyecto el origen de datos de la base de datos correspondiente al modelo estrella, si queremos poder visualizar gráficamente y realizar cambios sobre el modelo, las tablas, o crear campos, se puede realizar una vista al origen de datos.

La vista de origen de datos se realiza mediante un asistente donde simplemente se selecciona el origen de datos anterior, se eligen aquellas tablas sobre las que se quiere trabajar y se establece un nombre a la vista, el cual como se puede observar contiene ya las relaciones, debido a que el modelo estrella ya ha sido construido en el apartado 3.3, obteniendo así el resultado de la figura 44.

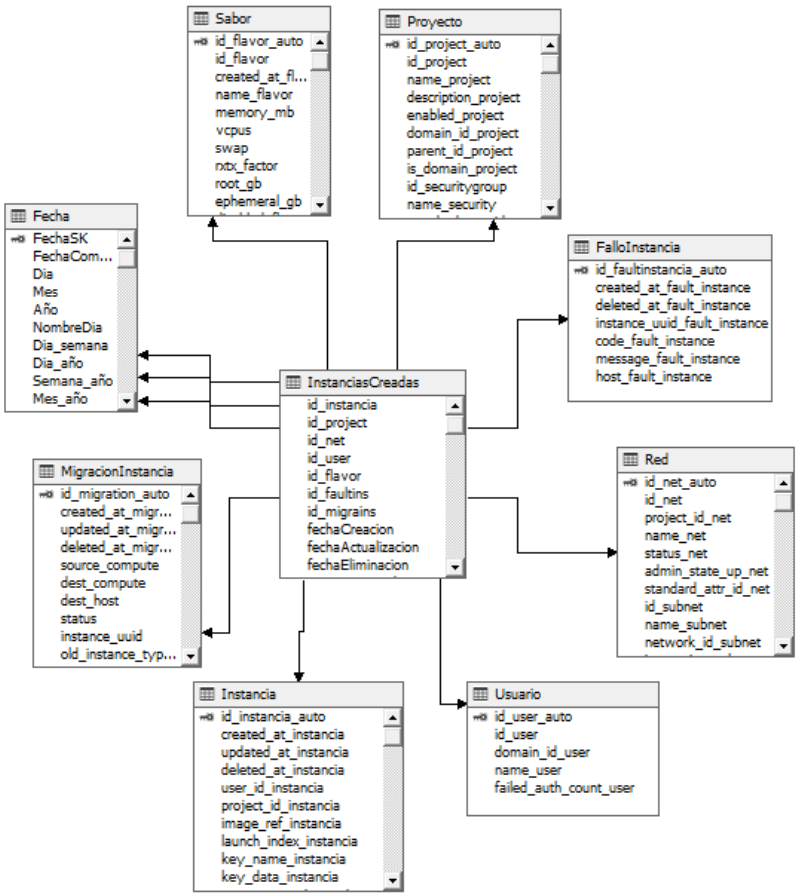


Figura 44: Vista origen de datos

Además de visualizar el diagrama se pueden realizar otras acciones sobre la vista del origen de datos. Es posible añadir otro diagrama donde se incluyan otras tablas y así poder construir más cubos, es posible modificar, añadir o eliminar las relaciones entre tablas, y lo más importante, y de lo que se hará uso a continuación es la creación de cálculos con nombre, que funciona como un nuevo campo añadido en la tabla deseada que realiza la expresión encontrada a la hora de crear dicho campo.

Principalmente este cálculo con nombre se realizan traducciones tales como cambiar valores numéricos por expresiones, traducciones de lenguaje como se va a realizar en nuestro proyecto, aunque también es posible realizar expresiones matemáticas.

Estos cálculos con nombre se han añadido en las tablas de dimensión de “Tiempo” y “MigraciónInstancia” dónde se va a realizar la traducción a ingles de los campos “Mes” y “NombreDia” de la dimensión “Fecha” y la traducción al español de los campos “status” y “migration_type” de la dimensión “MigraciónInstancia”.

En la creación de este cálculo con nombre hay que especificar el nombre que tendrá el campo dentro de la tabla y la expresión que realizará la traducción, especificando pues, dentro de la expresión el campo del cual se realiza dicha traducción.

En la figura 45 se puede ver la creación de este cálculo donde se realiza la traducción del campo “Mes” a su equivalente inglés.

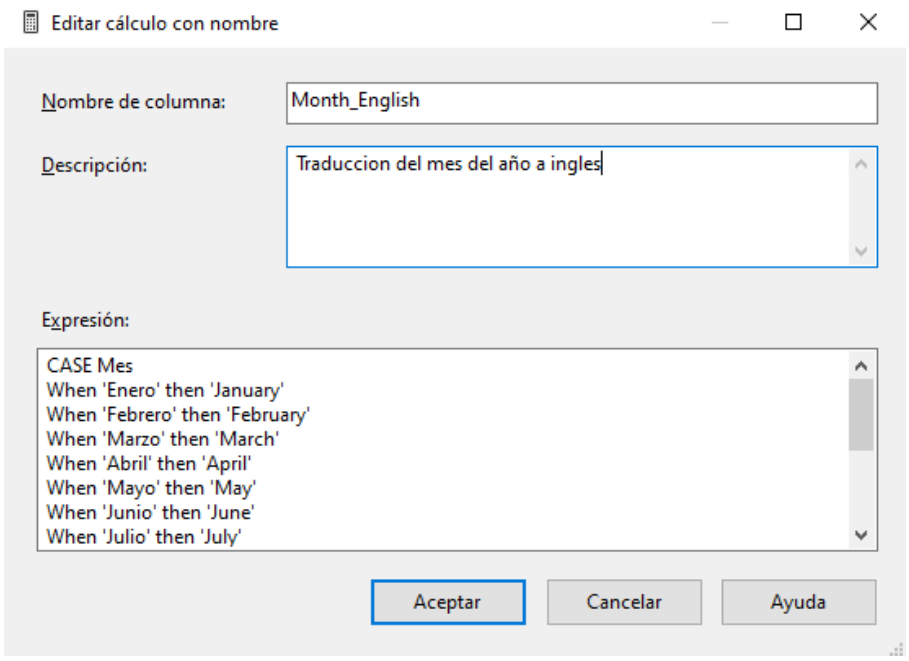


Figura 45: Cálculo con nombre para traducción

Una vez obtenido el origen de datos deseado y modificado a petición del proyecto obtendríamos el esquema de la figura 46.

Antes de la creación del cubo, es conveniente crear cada dimensión para poder modificar la misma y todo este proceso explicado a continuación, ha sido realizado tantas veces como tablas se deseen añadir como dimensión, especificando los detalles en la creación de aquellas dimensiones que precisen un tratamiento necesario.

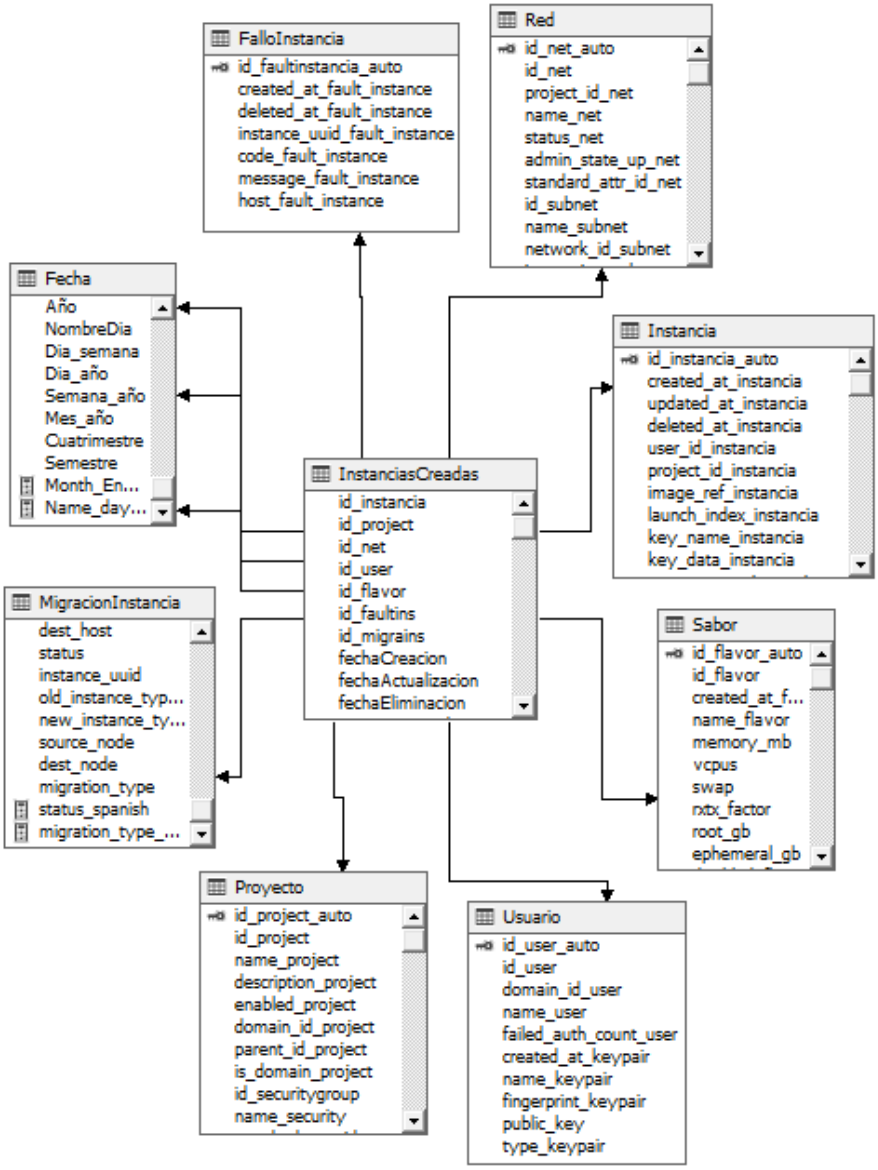


Figura 46: Origen de datos con cálculos

Para crear las dimensiones, existe un asistente de creación de dimensiones el cual contiene una serie de pasos para la creación de esta.

El asistente de dimensiones, en primer lugar, el asistente nos permite seleccionar el método de creación de la dimensión como muestra la siguiente figura, siendo la mejor opción para nuestro proyecto la creación de la dimensión usando las tablas existentes en el origen de datos.

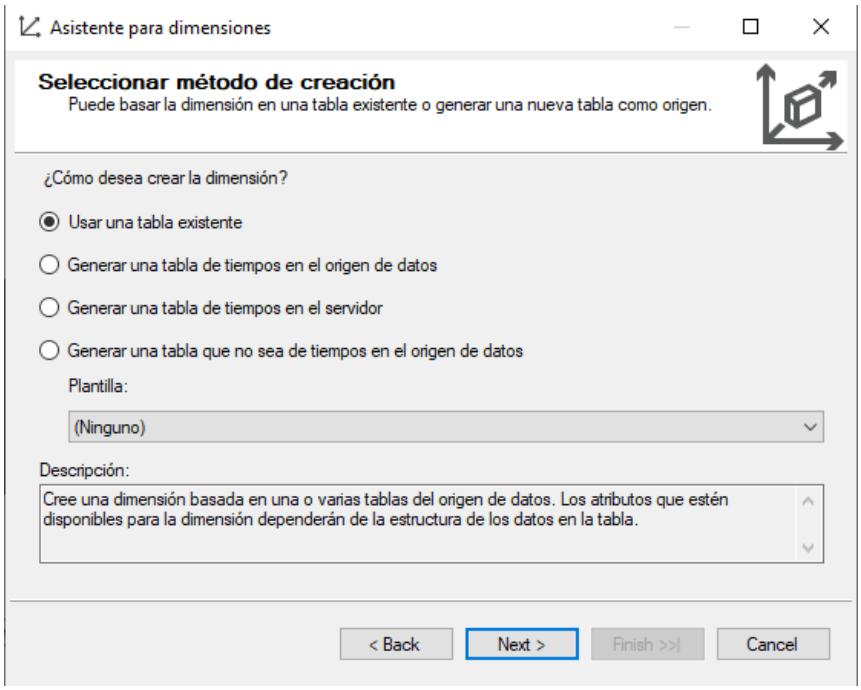


Figura 47: Método de creación dimensiones

En segundo lugar, el asistente solicita que información de origen deseamos incluir como dimensión, es decir, sobre que tabla se va a establecer que el proyecto de Analysis Services cree esta tabla como una dimensión.

En este paso, por tanto, se elige la tabla deseada de la vista del origen de datos y se establece cual será la columna de clave de nuestra dimensión como muestra la figura 48, dejando en nuestro proyecto, la columna por defecto, ya que estas claves fueron creadas al desarrollar el modelo estrella.

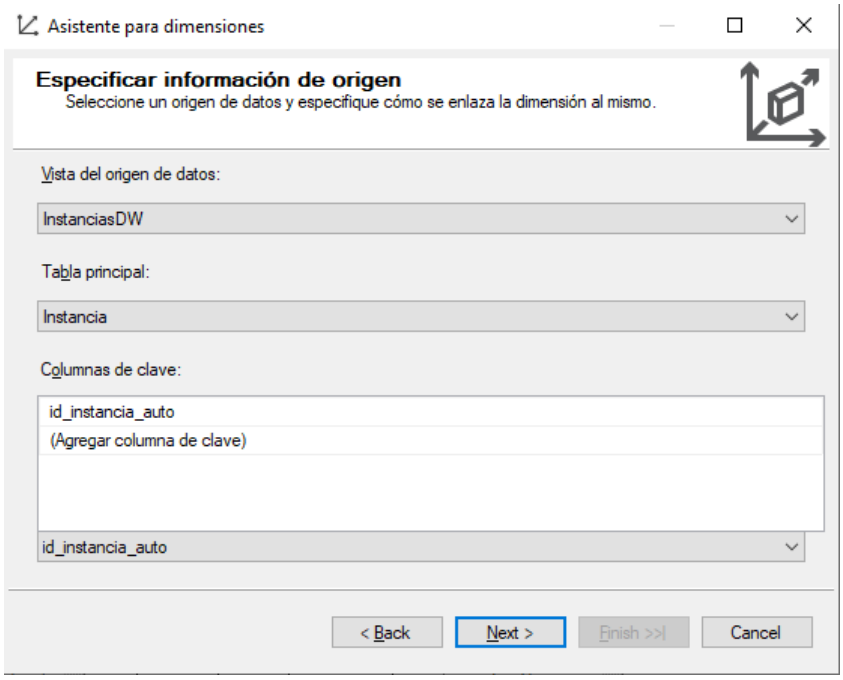


Figura 48: Información de origen en creación de dimensiones

Como último paso en la creación de la dimensión, el asistente nos permite seleccionar, que atributos, de los disponibles en la tabla, queremos añadir a la dimensión. En nuestro proyecto, como todos los atributos ya han sido revisados anteriormente, y en las tablas de la vista del origen de datos ya han sido seleccionados aquellos que queremos en nuestro proyecto final, añadiremos todos los atributos a la dimensión, salvo en la dimensión "Tiempo" donde seleccionaremos los deseados y en aquellas dimensiones donde se han creado campos con traducciones, estos campos no es necesario añadirlos ya que las traducciones se asignan de una manera detallada más adelante.

En este paso, también se puede modificar el tipo de atributo. Esta modificación es necesaria realizar para la correcta creación de la dimensión tiempo y el correcto funcionamiento de las jerarquías.

Por ende, es necesario pararse en este paso en la creación de la dimensión "Tiempo" ya que es necesario establecer el tipo de atributo para cada atributo relacionado con el tiempo que se desee añadir.

Como se ve en la Figura 49, para cada atributo que se desea añadir a la dimensión, se ha de seleccionar que tipo de fecha contiene el campo, si es una Fecha completa, si es un día de la semana, etc. Gracias a esta especificación de atributos, Analysis Services comprenderá que se trata de una dimensión tiempo.

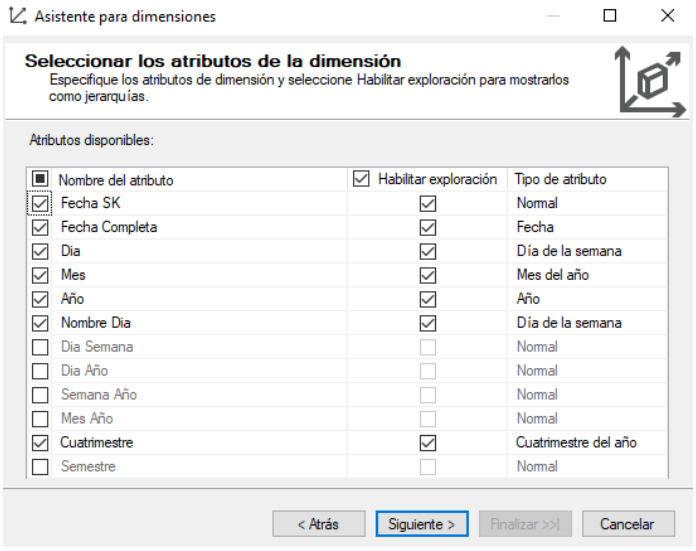


Figura 49: Atributos creación dimensión tiempo

Una vez completado este paso, la dimensión será agregada en el apartado correspondiente, donde quedará el listado de todas las dimensiones diseñadas mostrado en la siguiente figura.

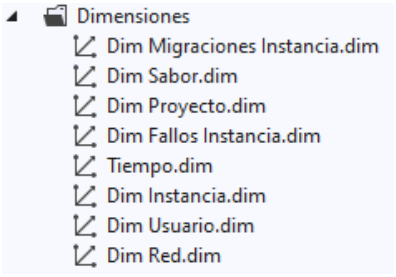


Figura 50: Dimensiones diseñadas

Estas dimensiones pueden ser modificadas para obtener un mejor resultado a la hora de explorar el cubo. Sobre las dimensiones es posible crear tantas jerarquías como se desee, y tantas traducciones como campos traducidos se tengan.

En nuestro proyecto, no todas las dimensiones dispondrán de jerarquías o traducciones.

A continuación, se va a desarrollar en el siguiente apartado las traducciones realizadas en las dimensiones.

3.5.1. Traducciones de las dimensiones

Las dimensiones que contienen traducciones ya han sido comentadas cuando se han creado los campos con estas traducciones y son las dimensiones “Tiempo” y “MigraciónInstancia”. Para asignar las traducciones Analysis Services contiene un apartado de “Traducciones” dentro de cada dimensión, que permite agregar un idioma y seleccionar aquellos campos que contienen una traducción.

En primer lugar, abrimos la dimensión que queremos modificar, y seleccionamos el apartado de traducciones. Dentro de este apartado encontramos un botón que crea una nueva traducción. Este botón abre un asistente para seleccionar un idioma entre un listado de los mismos, en nuestro proyecto, seleccionaremos Español (España) como se muestra en la figura 51, en la dimensión “MigraciónInstancia” donde se realizará la traducción Inglés-Español, y en la dimensión “Tiempo” seleccionaremos Inglés (Estados Unidos) para realizar la traducción Español-Inglés.

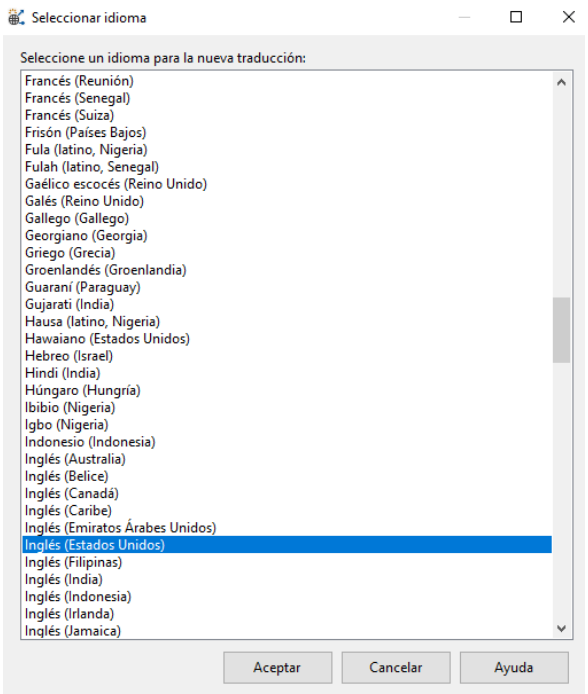


Figura 51: Idiomas disponibles para traducciones

Una vez seleccionado el idioma al que se desea realizar la traducción, hay que insertar el campo donde se ha realizado el cálculo con nombre que contiene la traducción en aquel campo donde se desea realizar la traducción como muestra la figura 52.

Figura 52: Campos de traducciones en la dimensión Fecha

Como se puede ver en la figura 53 se ha seleccionado el atributo mes de la dimensión "Tiempo" y se le ha asignado que para el idioma Inglés (Estados Unidos) el valor que deben mostrar los campos son los contenidos en el cálculo con nombre "Month_English"

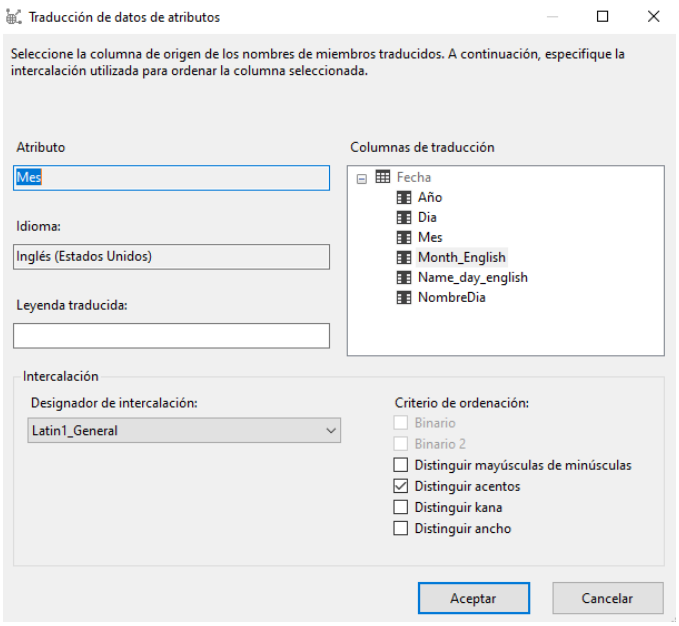


Figura 53: Asignación campo traducción

Este proceso hay que realizar en aquellos campos donde se desea realizar la traducción, dando como resultado la traducción encontrada en la figura 54, que cambia el nombre de los meses y los días de la semana a su equivalente inglés cuando el explorador del cubo tiene por defecto el idioma Inglés (Estados Unidos).

Figura 54: Traducción dimensión Tiempo

Una manera de comprobar que se han realizado las traducciones en las dimensiones cuando el idioma en el que se desea explorar el cubo es en Inglés es mediante el explorador de dimensiones dónde se puede establecer el idioma de exploración, donde vemos en la figura 55 como los días se encuentran en ingles al seleccionar dicho idioma en el explorador.

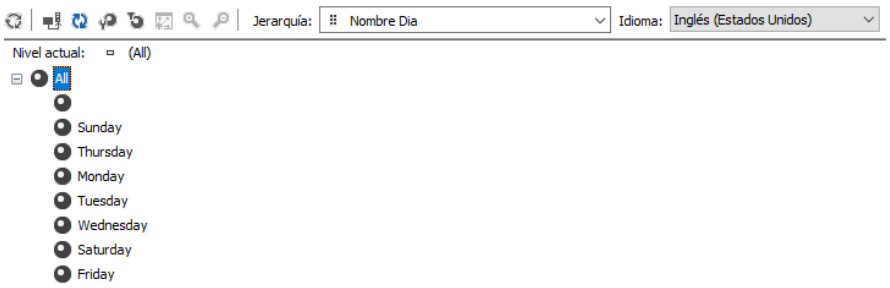


Figura 55: Ejemplo traducción

3.5.2. Jerarquías de las dimensiones

Otra posible modificación es la creación de jerarquías en las dimensiones. Debido a la estructura de las tablas del proyecto y como han sido dispuestas en el modelo, no es posible realizar jerarquías, ya que para realizar estas jerarquías tiene que existir una conexión en los datos y no todas las dimensiones disponen de ello.

Las dimensiones sobre las que se pueden realizar estas jerarquías son las dimensiones “Proyecto”, “Red”, “Instancia” y la más importante en la creación de cualquier estructura multidimensional es la jerarquía de la dimensión “Tiempo”.

Estas jerarquías son de utilidad a la hora de explorar los datos y poder obtener diferentes niveles de abstracción, como por ejemplo obtener los hechos correspondientes de un mes y año deseado dentro de las fechas que permite la dimensión.

Parar crear estas jerarquías Analysis Services también ofrece la posibilidad de crearlas mediante un apartado gráfico en la modificación de la propia dimensión. Estas jerarquías son creadas arrastrando y ordenando los elementos que queremos que formen parte de la jerarquía desde la lista de atributos a la ventana de Jerarquías.

En la figura 56 se puede ver el resultado de la jerarquía establecida en la dimensión tiempo y este proceso hay que repetirlo en aquellas dimensiones donde queramos realizar una jerarquía.

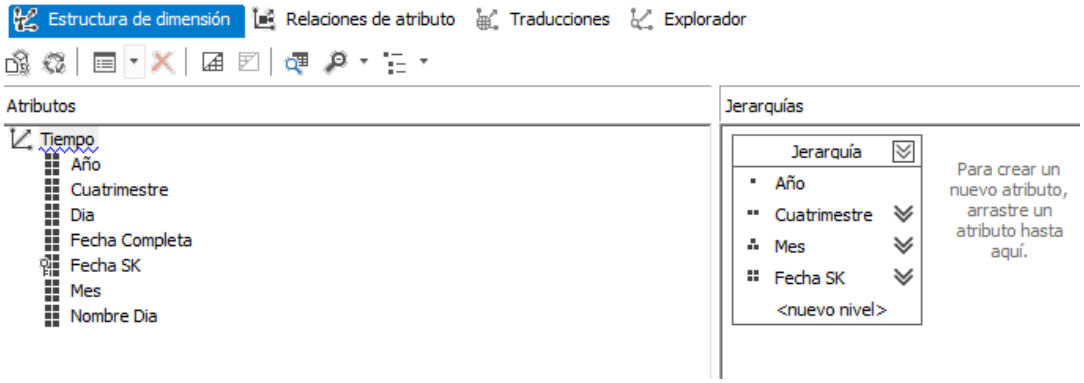


Figura 56: Jerarquía dimensión tiempo

En nuestro proyecto se han creado también jerarquías en las dimensiones “Red”, “Proyecto” e “Instancia” estableciendo en la dimensión “Proyecto” una relación entre el proyecto y el grupo de seguridad del proyecto como muestra la figura 57, ya que el id del proyecto es la granularidad más baja dentro de la dimensión, y cada proyecto tiene varios grupos de seguridad.

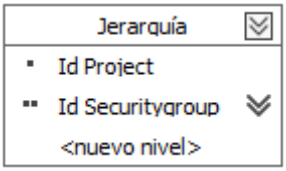


Figura 57: Jerarquía dimensión proyecto

Para la dimensión “Red” la relación es similar, estableciendo la jerarquía entre el identificador del proyecto y la red, ya que en esta dimensión cada proyecto puede tener varias redes como muestra la figura 58.

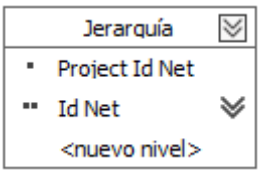


Figura 58: Jerarquía dimensión red

En la dimensión “Instancia” se han construido tres jerarquías visualizadas en la figura 59, ya que esta dimensión contiene muchos niveles detalles al contener campos que son identificadores de otras dimensiones, como proyecto y usuario que tienen mayor granularidad que instancia.

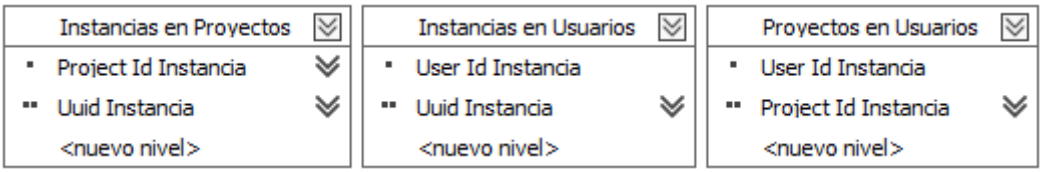


Figura 59: Jerarquía dimensión instancia

Una vez finalizadas las jerarquías, la herramienta muestra una advertencia de que no están definidas las relaciones de atributos para todos los niveles incluidos en la jerarquía. Estas relaciones de atributo especifican las restricciones de cardinalidad existentes entre los niveles de una jerarquía y por tanto sin dichas relaciones Analysis Services no computará correctamente los agregados que existen entre los distintos niveles. Para solucionar esto, la propia herramienta, dispone una pestaña de Relaciones de atributo, dónde se pueden ver las relaciones que tienen los elementos de las jerarquías establecidas en cada dimensión. SSAS proporciona por defecto unas relaciones entre la clave y los demás elementos de la jerarquía, pero no ofrece las relaciones entre los propios miembros, que es lo que debemos modificar nosotros.

La solución propuesta es relacionar desde la clave, que es el nivel más bajo, hasta el máximo nivel de detalle, como se puede ver en la figura 60 donde se encuentran las relaciones de atributo de la dimensión tiempo.

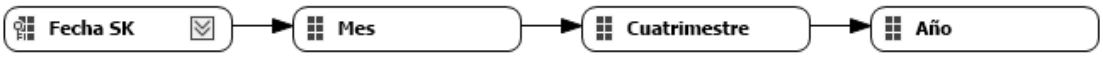


Figura 60: Relaciones de atributo dimensión tiempo

Básicamente, esta relación nos dice, que, desde el nivel más bajo, la clave, podemos acceder al mes, desde el mes podemos saber en qué cuatrimestre se encuentra, y de este último conocer el año al que pertenece. Además, estas relaciones son definidas como "rígidas" ya que las relaciones entre los miembros no cambiarán con el tiempo.

Estas relaciones también hay que realizarlas para las demás dimensiones al contener varios atributos en las jerarquías como se muestra en la figura 61 con las relaciones de atributo para la dimensión instancia.

Figura 61: Relaciones de atributo dimensión instancia

Finalizando la creación de las jerarquías, cuando una de estas es creada, lo que se espera es tener relaciones "m:1" desde los niveles inferiores a los niveles superiores de forma que si, por ejemplo, hablamos de nuestra dimensión tiempo, se espera que todos los cuatrimestres y meses correspondan a un año en concreto, pero este mismo cuatrimestre o mes, puede pertenecer a 4 años diferentes. Es decir, Analysis Services no sabría identificar donde pertenece el mes "Enero", porque lo encuentra duplicado en los distintos años establecidos en la dimensión. Esto mismo pasaría con los cuatrimestres, y si tuviéramos niveles más detallados por debajo de Año, Mes o cualquier otro nivel inferior también encontraríamos este problema.

Este problema también se encuentra con el campo "Project_id" de la dimensión Instancia, ya que en la jerarquía donde se relaciona la instancia y el proyecto, puede repetirse un proyecto para distintas instancias y por tanto necesitaríamos otro campo para poder identificarlo correctamente.

A continuación, se va a detallar como resolver el problema para un campo de la dimensión Tiempo, pero habría que repetir el proceso con aquellos campos que presentan este problema.

Para solucionar este problema los atributos de la dimensión tienen una propiedad llamada "KeyColumns" que especifica cual es el valor que realmente toma el atributo, por lo que, se le puede establecer un conjunto de atributos, es decir, una colección para poder identificar correctamente en nuestro caso tanto el Mes como el Cuatrimestre.

Como se puede ver en la figura 62 el campo cuatrimestre tiene ahora como claves el cuatrimestre y el año, por tanto, ahora se pueden identificar cada cuatrimestre repetido ya que pertenece a un único año identificando ahora correctamente el campo.

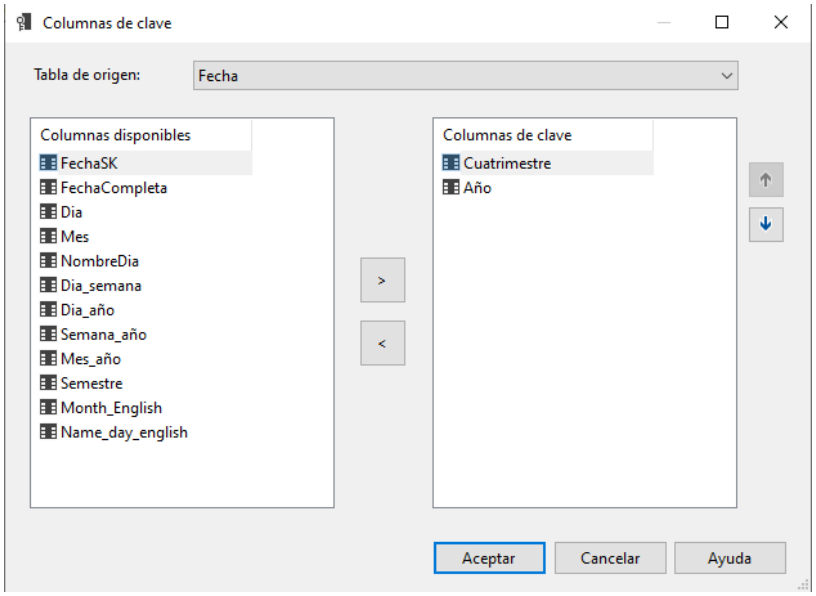


Figura 62: Colección de valores para cuatrimestre

Una vez asignadas las colecciones en aquellos atributos con claves duplicadas, se puede seleccionar cual es el valor que se desea mostrar en pantalla, dejando solamente el atributo del nivel de detalle más bajo como se ve en la siguiente figura.

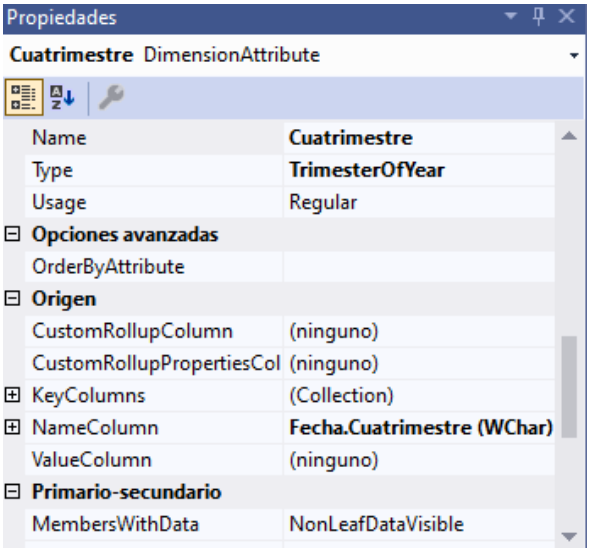


Figura 63: Propiedades atributo cuatrimestre

Una vez establecido todo lo necesario para el funcionamiento de las jerarquías podemos observar en la figura 64 como quedaría la jerarquía en el explorar de datos de la dimensión tiempo.

Figura 64: Resultado jerarquía tiempo

Cuando se ha conseguido tener la vista de origen de datos, con las relaciones y campos deseados y las dimensiones creadas y configuradas según la necesidad, es posible crear el cubo OLAP.

3.5.3. Creación cubo OLAP

Un cubo OLAP (Online Analytical Processing) es la estructura de datos, en la cual el almacenamiento físico de los datos se realiza en un vector multidimensional. Este sistema OLAP no va a eliminar ni modificar los datos, solo va a añadir datos si es necesario realizando de nuevo los procesos ETL, ya que es un sistema optimizado de consultas para recuperar grandes cantidades de datos [16].

Para crear este cubo, Analysis Services dispone de un asistente para facilitar su creación. En primer lugar, debemos seleccionar el método de creación del cubo, existiendo la posibilidad de crear el cubo usando las tablas del origen de datos, creando el cubo vacío para desarrollarlo manualmente o generando tablas en el origen de datos subyacente. Para nuestro proyecto se va a seleccionar el uso de tablas existentes, como se muestra en la figura 65, ya que en pasos anteriores ha sido creado todo lo necesario.

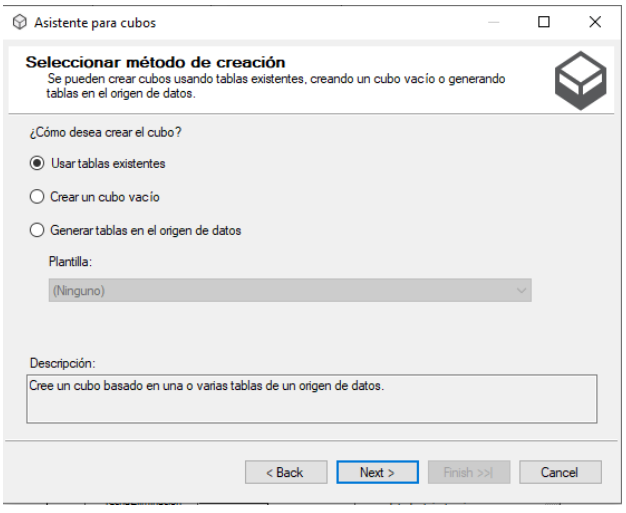


Figura 65: Método creación cubo

A continuación, se realizan dos acciones en el mismo paso, ya que se selecciona la vista del origen de datos, que en nuestro proyecto solo ha sido creada una vista. Por otro lado, hay que establecer cuál será la tabla que actuará como tabla de grupo de medida, o como ha sido citada anteriormente, tabla de hechos.

Esta tabla puede ser sugerida por el propio asistente de Analysis Services, pero como el modelo ha sido construido a consciencia, esta tabla ya ha sido preparada, y por tanto la tabla "InstanciasCreadas" pasará a ser denominada tabla de grupo de medida como se muestra en la siguiente figura.

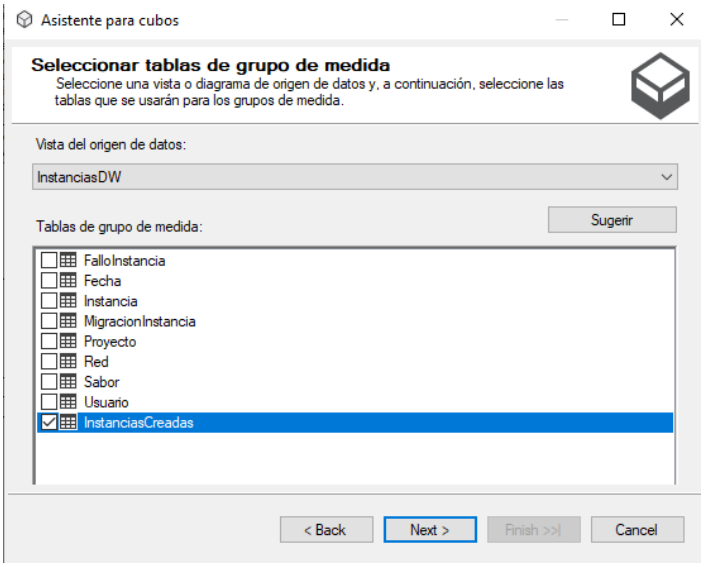


Figura 66: Asignación vista y grupo de medida

El siguiente paso de la creación del cubo es la selección de medidas que dispondrá la tabla de grupo de medida previamente elegida. Como no ha sido insertada ninguna medida previamente, y tiene que tener mínimo una medida el cubo, seleccionaremos la medida que crea automáticamente el cubo como se puede ver en la figura 67, aunque, las medidas que se utilizarán para el análisis de datos serán creadas en la modificación del cubo donde se presentan más oportunidades.

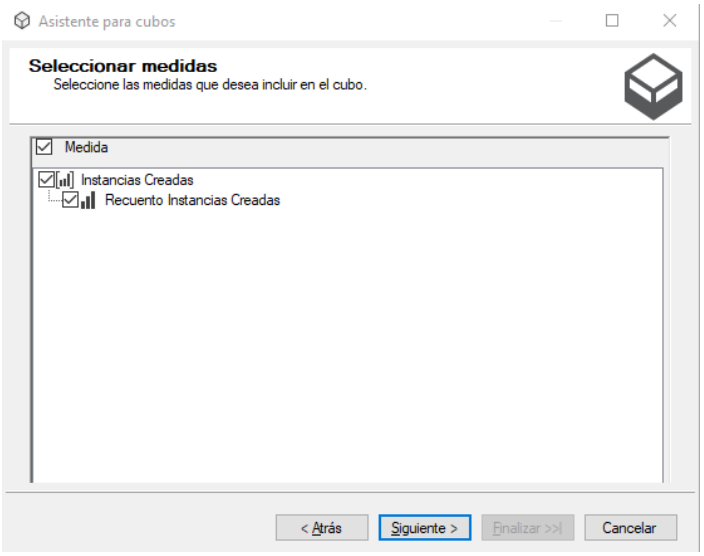


Figura 67: Selección medidas cubo

Por último, una vez finalizada la selección de medidas, dispondremos de una última selección, las dimensiones que queremos incluir en el cubo, que volverán a ser todas las dimensiones creadas y modificadas anteriormente como se puede apreciar en la figura 68.

Una vez elegidas las dimensiones es finalizado el asistente para cubos, y se abre en pantalla un visualizador del cubo, donde podemos realizar operaciones sobre el mismo.

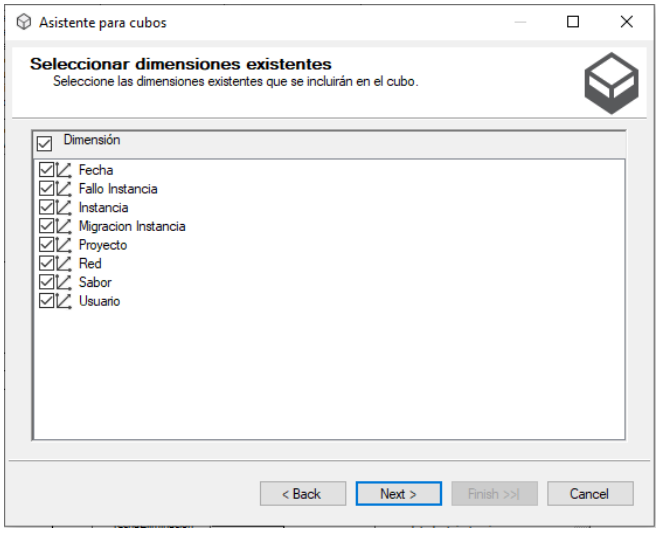


Figura 68: Selección dimensiones cubo

Ahora es posible visualizar el resultado del cubo en la figura 69, el cual ha sido proporcionado por el asistente para cubos y lo que hemos seleccionado durante el proceso.

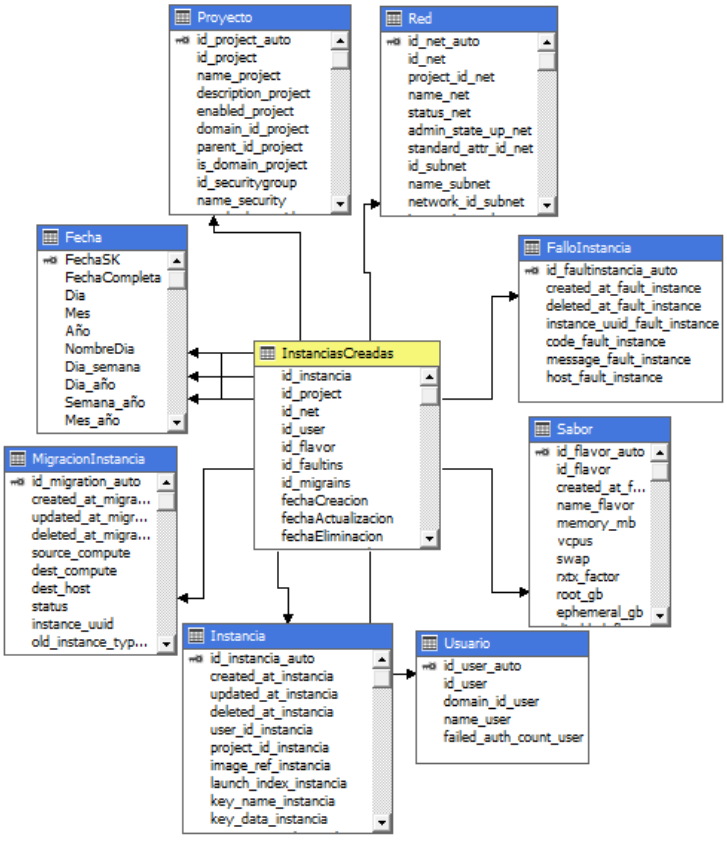


Figura 69: Cubo OLAP sin modificaciones

En este explorador, disponemos de diversas pestañas para realizar operaciones y así modificar y mejorar el cubo. Es posible añadir nuevas medidas o grupos de medidas, realizar cálculos y añadirlos como medidas, diseñar particiones, crear agregaciones, añadir perspectivas para tener más grupos de medida, realizar discretizaciones en un numero de rangos de un atributo, agregar traducciones para la tabla de hechos y un explorador para visualizar los datos contenidos.

De estas acciones, por la naturaleza de nuestro proyecto y como ha llegado hasta este proceso solo es necesario modificar y crear las medidas que se finalmente se utilizarán para el análisis.

Respecto a las medidas, según se vio en el estudio de las tablas y dimensiones, en la dimensión Instancia tenemos dos valores numéricos que pueden ser medidas, la memoria consumida por una Instancia, y las cpus virtuales que utiliza. Estas medidas no han sido añadidas previamente en la tabla de hechos debido a que se tratan de medidas no aditivas en la tabla de hechos ya que no se pueden agregar sobre ninguna columna al haber filas repetidas sin necesidad de significar que se trata de una nueva instancia. Sin embargo, en la dimensión Instancia funcionan como medidas aditivas ya que se puede agregar en todas las columnas que lo permiten.

Para añadir estos dos valores numéricos como medidas al cubo, es necesario crear un nuevo grupo de medida y establecerlo sobre la dimensión Instancia como se puede ver en la siguiente figura, donde se selecciona entre el listado de posibles grupos de medidas, la dimensión que actuará como grupo de medida en el cubo.

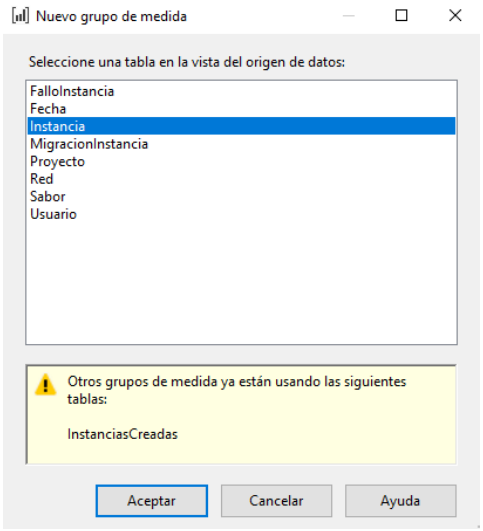


Figura 70: Creación nuevo grupo de medida

Una vez creado el grupo de medida eliminamos todos los campos que no se desean tener como medidas hasta dejar los campos deseados que se pueden ver en la figura 71.

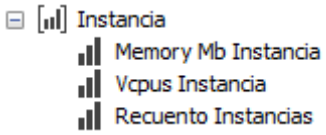


Figura 71: Grupo de medida sobre la Dimensión Instancia

Como se puede apreciar en la figura 71 el asistente ha añadido una nueva medida que se encarga de contar las instancias pudiendo también agruparlas, añadiendo así una nueva medida a las dos elegidas previamente.

Por otro lado, podemos crear tres medidas aditivas sobre la tabla de hechos que se encarguen de contar las instancias, usuarios, redes y proyectos como se puede ver en la figura 72.

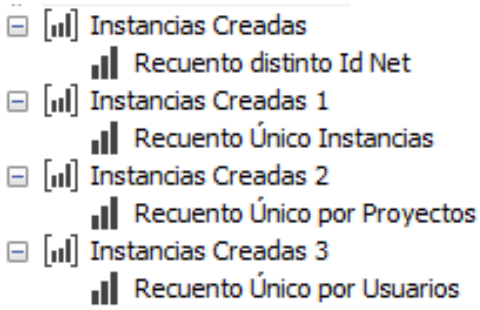


Figura 72: Medidas de la tabla de hechos

La particularidad de estas medidas es que en su creación se ha de seleccionar que hagan un recuento distinto de las filas como se muestra en la figura 73, debido a que, como se ha comentado anteriormente las filas se repiten sin necesidad de suponer una nueva transacción.

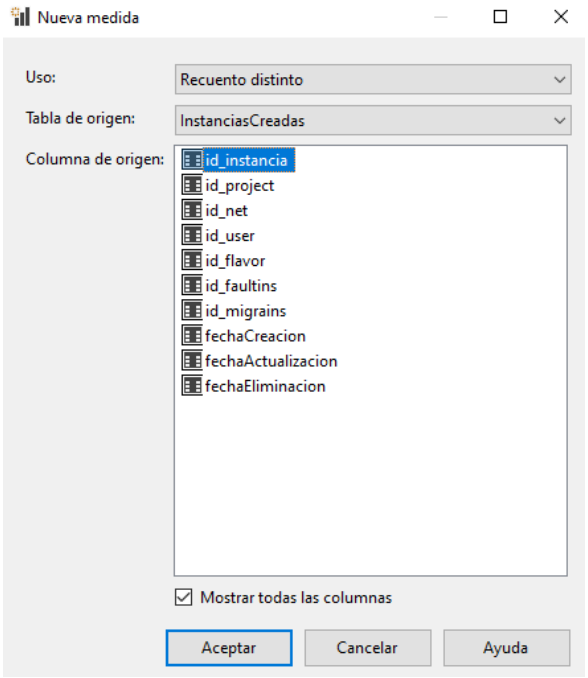


Figura 73: Creación medida de recuento distinto sobre instancia en la tabla de hechos

Una vez finalizada la creación de estas medidas tenemos que el cubo ahora tiene dos grupos de medida distintos que contienen un total de 7 medidas como se puede ver en la siguiente figura.

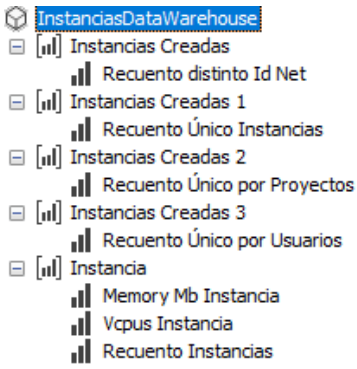


Figura 74: Medidas cubo

En la figura 75 se puede visualizar el cubo OLAP que ha sido fruto del desarrollo de todo lo comentado en este documento, desde la construcción del modelo estrella que establecía la estructura multidimensional, pasando por el proceso ETL que poblaba de datos este modelo, hasta el desarrollo del BackEnd realizado en este apartado que cogía estos procesos realizados y los convertía en el cubo que tenemos ahora.

Para finalizar, lo ideal es usar el cubo fuera de SSAS, en programas como el propio SQL Sever Management Studio, Excel o Power Bi, y para esto, es necesario implementar el proyecto que guardará la estructura en SQL Server Analysis Services y procesará cada apartado del proyecto para comprobar si están o no bien diseñados o contiene algún tipo de error.

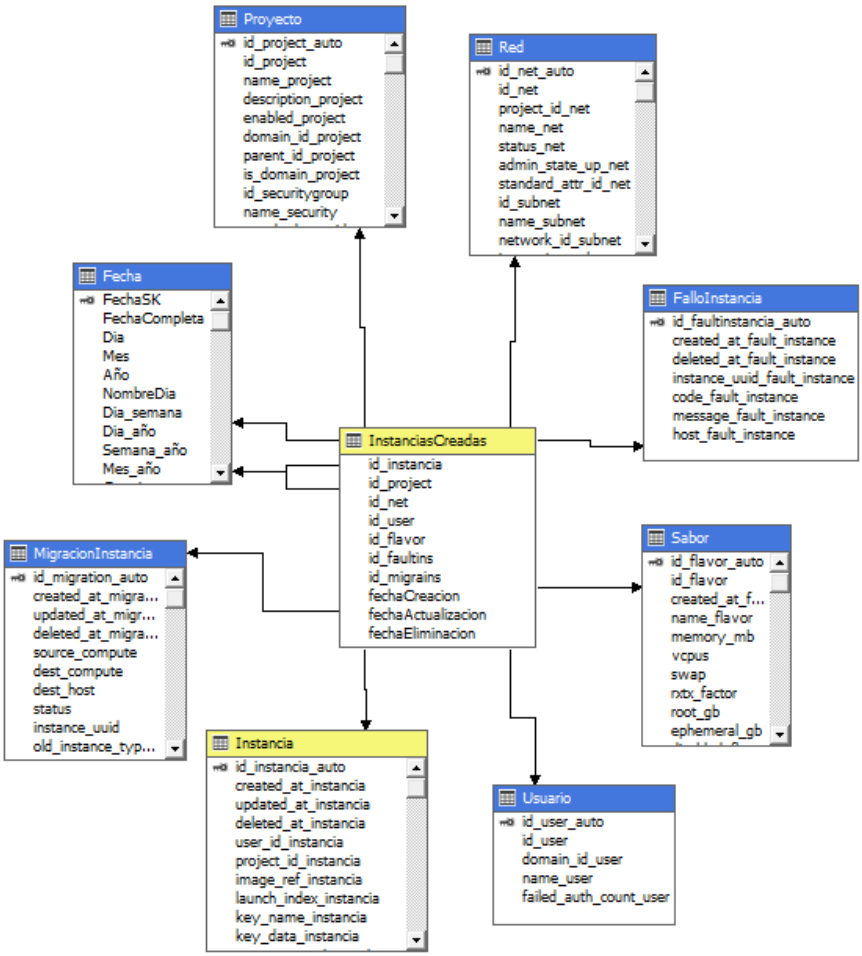


Figura 75: cubo OLAP modificado

Si el resultado de la implementación es positivo como se muestra en la figura 76 esto significa que la implementación finalizó correctamente y ahora se puede tratar el cubo en los programas mencionados anteriormente.

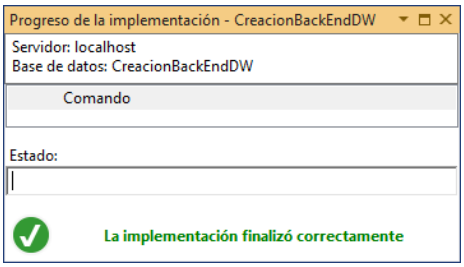


Figura 76: Implementación proyecto BackEnd

Una vez finalizada la implementación se puede comprobar dicha implementación realizando una conexión a Analysis Services desde SQL Server Management Studio como se muestra en la siguiente figura.

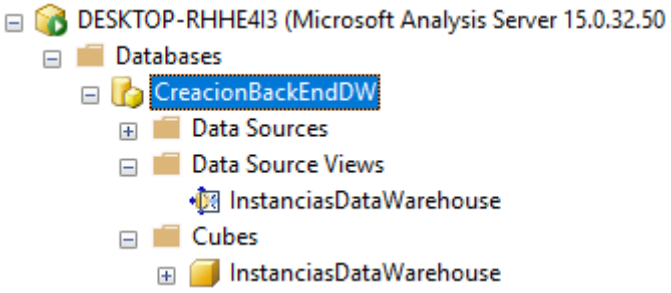


Figura 77: Comprobación implementación proyecto BackEnd

Esta implementación del proyecto no conlleva que no se puedan realizar más cambios sobre el mismo, ya que es posible actualizar los esquemas de la vista del origen de datos, eliminando tablas, columnas o relaciones o bien añadiendo columnas y relaciones a tablas que ya se encuentran en la vista del origen de datos. También se pueden realizar nuevos cambios sobre las dimensiones y el cubo, teniendo que procesar dicho elemento para llevar los cambios al proyecto implementado fuera de Analysis Services.

La estructura final del proyecto que contiene al cubo y el desarrollo del BackEnd la podemos visualizar en la figura 78.

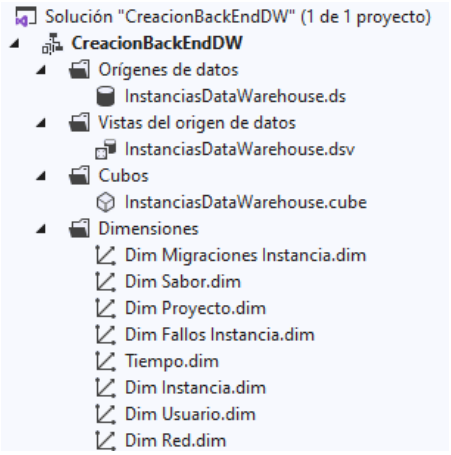


Figura 78: Estructura BackEnd

Llegados a este punto, podemos dar por finalizada la creación del cubo OLAP que será utilizado para extraer información relevante sobre la utilización de la plataforma OpenStack.

En el siguiente apartado se van a enseñar algunas herramientas para visualizar y manejar fácilmente el contenido de este cubo OLAP, así como enseñar y explicar qué tipo de información podemos obtener de esta estructura multidimensional.

3.5.1 Uso y visualización del BackEnd

Este proyecto ha sido realizado con la intención de poder tener una herramienta de visualización sencilla que permita poder explorar los datos de la plataforma OpenStack y así poder obtener información del rendimiento, uso, problemas, etc que ofrece esta plataforma.

Para visualizar esta información, en primer lugar, hay que realizar el desarrollo de un cubo, de tipo OLAP en nuestro proyecto que contenga una estructura multidimensional, desarrollo el cual ha sido detallado en anteriores apartados, y, en segundo lugar, la elección por parte del usuario final, de que herramienta desea utilizar para inspeccionar el cubo.

Básicamente se pueden encontrar tres tipos de herramientas, herramientas como la conexión a Analysis Services de SQL Server Management Studio que ofrece la posibilidad de realizar consultas multidimensionales, herramientas como Excel que mediante una conexión a una base de datos Analysis Services permite realizar consultas arrastrando los elementos de una manera gráfica y herramientas profesionales como Power BI que permite obtener gráficos de una manera sencilla.

En este proyecto se va a explicar cómo usar el cubo y visualizar los datos con Analysis Services de SQL Server Management Studio y Excel, debido a la facilidad y rapidez con la cual puede hacerse, y en otro proyecto, se desarrollará la construcción del entorno necesario para preparar Power BI como solución final.

3.5.1.1 Uso en SQL Server Management Studio de Analysis Services

En primer lugar, se mostrará el uso del cubo en SQL Server Management con una conexión a Analysis Services y una serie de consultas MDX. Como la implantación del proyecto ya ha sido realizada, y esto significa que se crea una base de datos en SQL Server Analysis Services no es necesario realizar ninguna nueva conexión, ya que ya tenemos el cubo listo para usar.

Para realizar consultas, basta con seleccionar el cubo sobre el que se quiere realizar la consulta y el tipo de consulta entre MDX (MultiDimensional eXpressions), DMX (Data Mining Extensions), XMLA (XML for Analysis), DAX (Data Analysis Expressions), siendo para este ejemplo el tipo de consulta seleccionada MDX ya que permite consultar objetos multidimensionales y devolver conjuntos de celdas multidimensionales que contenga los datos del cubo.

Una vez realizada la conexión, se nos muestra una ventana donde podemos elegir el cubo sobre el que queremos realizar la consulta, cambiar el grupo de medida de este cubo, y visualizar medidas, dimensiones y jerarquías, además de un espacio para realizar la consulta como se muestra en la siguiente figura.

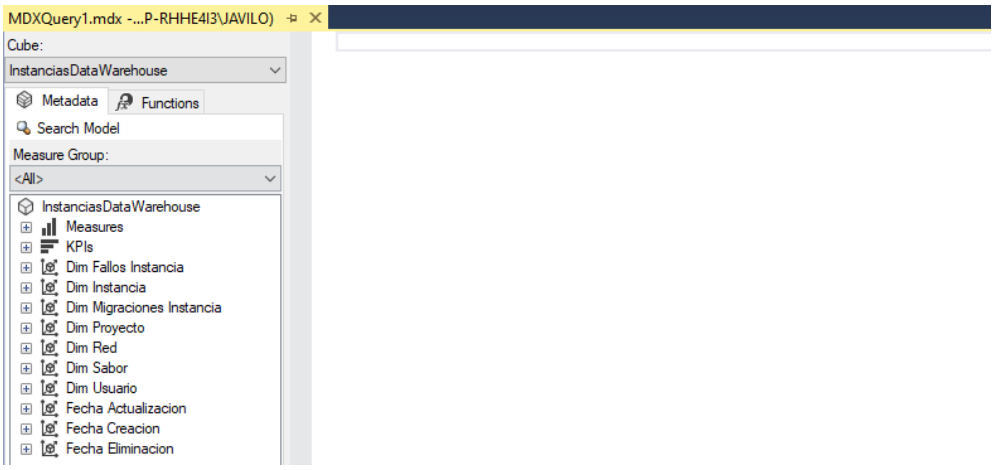


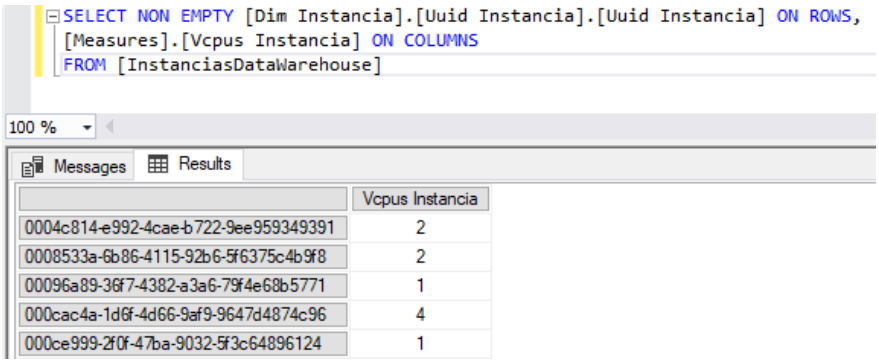
Figura 79: Ventana consulta MDX

Respecto a la consulta, el formato de la sintaxis de la instrucción MDX SELECT es similar al de la sintaxis de SQL. En MDX, la instrucción SELECT especifica un conjunto de resultados que contiene un subconjunto de datos multidimensionales que se ha devuelto desde el cubo. Una consulta MDX debe contener la siguiente información [17]:

- El número de ejes que se desea que contenga el conjunto de resultados.
- El conjunto de miembros o tuplas que se incluyen en cada eje de la consulta.
- El nombre del cubo que define el contexto de la consulta.
- El conjunto de miembros o tuplas que se incluyen en el eje segmentador.

A continuación, se van a presentar una serie de consultas realizadas al cubo "InstanciasDataWarehouse" resultado de este proyecto.

La primera consulta es una consulta sencilla que se puede ver en la figura 80 donde se obtienen todos los identificadores de la instancia y las cpus virtuales que ha utilizado dicha instancia.



```
SELECT NON EMPTY [Dim Instancia].[Uuid Instancia].[Uuid Instancia] ON ROWS,
  [Measures].[Vcpus Instancia] ON COLUMNS
  FROM [InstanciasDataWarehouse]
```

100 %

Messages | Results

	Vcpus Instancia
0004c814-e992-4cae-b722-9ee959349391	2
0008533a-6b86-4115-92b6-5f6375c4b9f8	2
00096a89-36f7-4382-a3a6-79f4e68b5771	1
000cac4a-1d6f-4d66-9af9-9647d4874c96	4
000ce999-2f0f-47ba-9032-5f3c64896124	1

Figura 80: Ejemplo nº1 consulta MDX

Si se desea algo más complejo y útil dentro del apartado de estudiar el rendimiento de la plataforma, se puede realizar la consulta de la figura 79 en la que vamos a averiguar que 5 usuarios consumieron más cpus virtuales en la creación de sus instancias. El resultado de la consulta se puede ver en la siguiente figura.

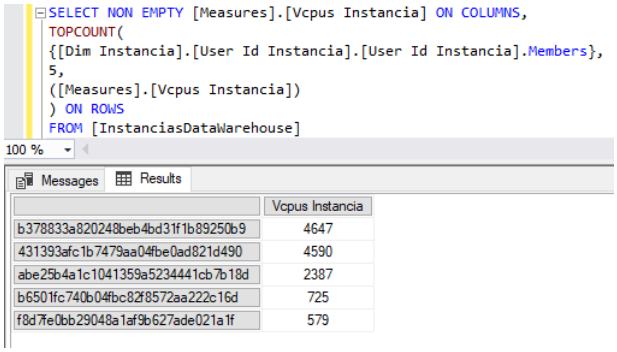


Figura 81: Ejemplo nº2 consulta MDX

De la misma manera, es posible averiguar que cinco proyectos son los que consumen más memoria de las instancias que se encuentran activas actualmente. El resultado de la consulta se puede ver en la siguiente figura.

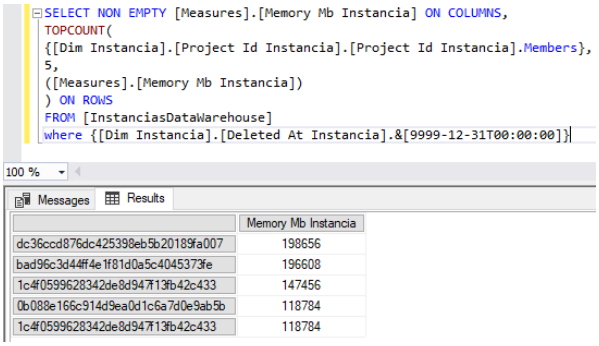


Figura 82: Ejemplo nº3 consulta MDX

Son muchas más las posibilidades que podemos realizar con las consultas MDX, pero como no es el objetivo de este proyecto no se mostrarán más ejemplos.

3.5.1.2 Uso con Microsoft Excel

La característica de usar el cubo en Excel está pensada para proporcionar también una manera rápida y sencilla de probar la eficacia del diseño del modelo, con la particularidad de no realizar consultas de manera escrita y ofreciendo la posibilidad de utilizar las herramientas de análisis que ofrece Excel.

Debido a que el cubo no ha tenido relación con Excel, es necesario establecer una conexión a Analysis Services que contiene al cubo para poder trabajar sobre él. Esta conexión se muestra en la siguiente figura.

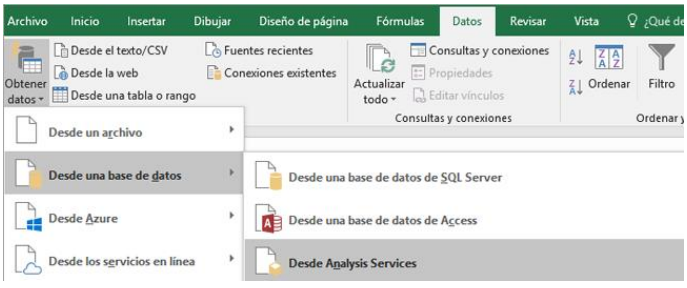


Figura 83: Elección de la conexión a Analysis Services desde Excel

Una vez seleccionada la fuente de los datos, para realizar la conexión es necesario especificar el servidor al que debe conectarse y las credenciales de conexión como se muestra en la siguiente figura.

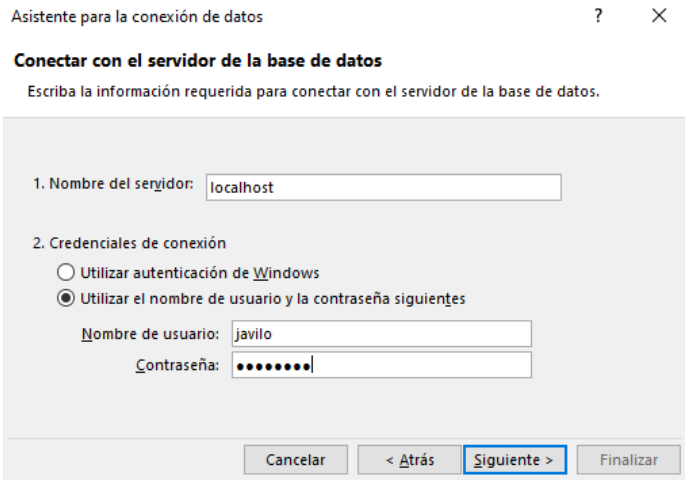


Figura 84: Conexión servidor Analysis Services desde Excel

Ahora ya se dispone de todas las bases de datos que contiene esta conexión, y debemos seleccionar aquella donde se encuentra el proyecto creado. En este paso Excel ofrece la posibilidad de elegir a que tabla o cubo se quiere conectar, pero como en este proyecto solo ha sido creado un cubo, no hace falta realizar esta elección y por tanto se ha de seleccionar el cubo creado anteriormente como muestra la figura 85.

Al finalizar el asistente para la conexión de datos, Excel pregunta cómo se desea importar los datos, mediante un informe de tabla dinámica, un gráfico dinámico o simplemente crear la conexión. Para este apartado, se van a importar los datos en un informe de tabla dinámica y se van a realizar algunas consultas para comparar su utilidad con las consultas realizadas en SQL Sever Analysis Services.

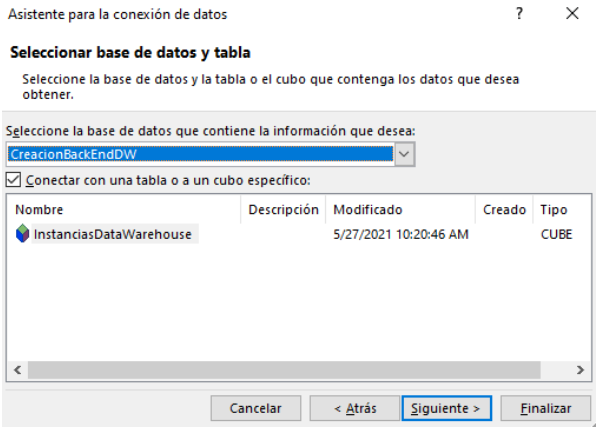
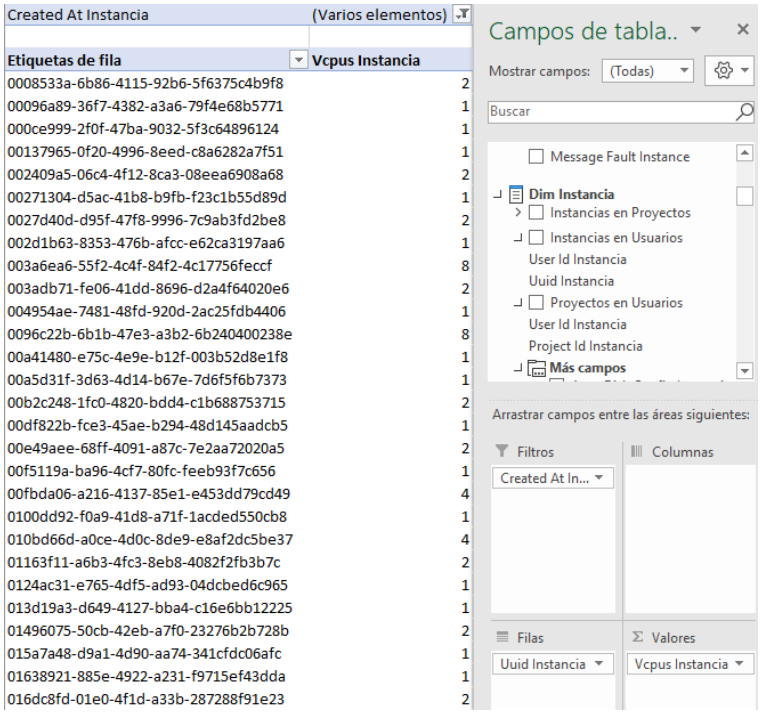


Figura 85: Selección base de datos desde Excel

Cuando ha sido creado el informe de tabla dinámica, lo que encontramos en pantalla es una hoja Excel donde se colocarán los campos seleccionados, una lista de los campos de las tablas y las medidas, y un área donde arrastrar los campos y seleccionar si queremos que sea un filtro, columna, fila o valor.

Para comprobar la facilidad de uso se van a copiar algunas de las consultas realizadas en SSAS.

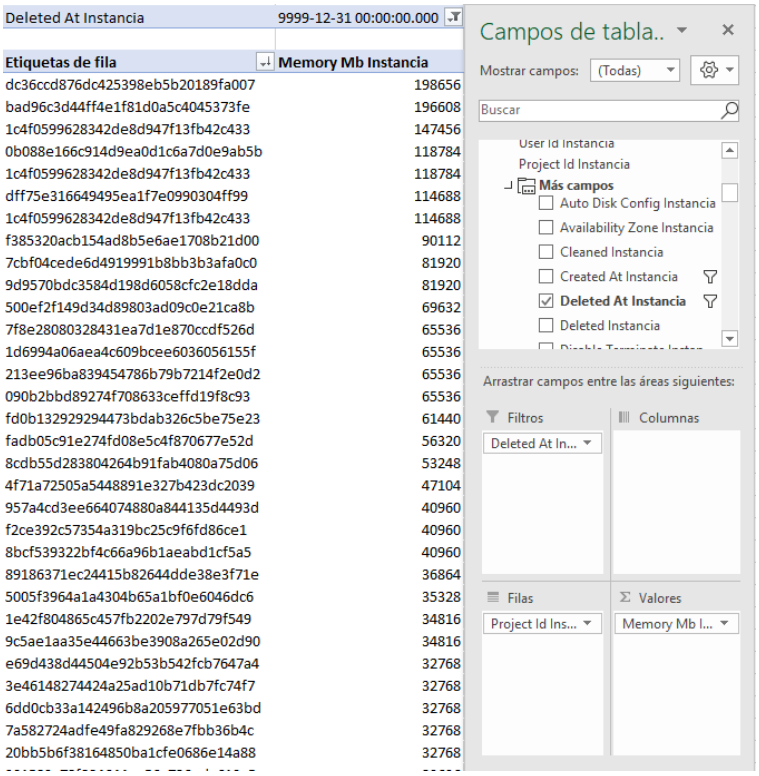
La primera consulta que copiar es aquella donde se han obtenido los identificadores de las instancias y las cpus virtuales utilizadas por cada una de ellas en función de una fecha, en este caso todos los identificadores de 2018. El resultado se encuentra en la siguiente figura.



Created At Instancia	(Varios elementos)
Etiquetas de fila	**Vcpus Instancia**
0008533a-6b86-4115-92b6-5f6375c4b9f8	2
00096a89-36f7-4382-a3a6-79f4e68b5771	1
000ce999-2f0f-47ba-9032-5f3c64896124	1
00137965-0f20-4996-8eed-c8a6282a7f51	1
002409a5-06c4-4f12-8ca3-08eea6908a68	2
00271304-d5ac-41b8-b9fb-f23c1b55d89d	1
0027d40d-d95f-47f8-9996-7c9ab3fd2be8	2
002d1b63-8353-476b-afcc-e62ca3197aa6	1
003a6ea6-55f2-4c4f-84f2-4c17756feccf	8
003adb71-fe06-41dd-8696-d2a4f64020e6	2
004954ae-7481-48fd-920d-2ac25fdb4406	1
0096c22b-6b1b-47e3-a3b2-6b240400238e	8
00a41480-e75c-4e9e-b12f-003b52d8e1f8	1
00a5d31f-3d63-4d14-b67e-7d6f5f6b7373	1
00b2c248-1fc0-4820-bdd4-c1b688753715	2
00df822b-fce3-45ae-b294-48d145aadcb5	1
00e49aee-68ff-4091-a87c-7e2aa72020a5	2
00f5119a-ba96-4cf7-80fc-feeb93f7c656	1
00fbda06-a216-4137-85e1-e453dd79cd49	4
0100dd92-f0a9-41d8-a71f-1acded550cb8	1
010bd66d-a0ce-4d0c-8de9-e8af2dc5be37	4
01163f11-a6b3-4fc3-8eb8-4082f2fb3b7c	2
0124ac31-e765-4df5-ad93-04dcbed6c965	1
013d19a3-d649-4127-bba4-c16e6bb12225	1
01496075-50cb-42eb-a7f0-23276b2b728b	2
015a7a48-d9a1-4d90-aa74-341cfdc06afc	1
01638921-885e-4922-a231-f9715ef43dda	1
016dc8fd-01e0-4f1d-a33b-287288f91e23	2

Figura 86: Ejemplo consulta nº1 Excel

En segundo lugar, en la consulta de la figura 87 se ha extraido los proyectos que consumen más megabytes de memoria de aquellas instancias activas actualmente.



Deleted At Instancia	9999-12-31 00:00:00.000
Etiquetas de fila	**Memory Mb Instancia**
dc36ccd876dc425398eb5b20189fa007	198656
bad96c3d44ff4e1f81d0a5c4045373fe	196608
1c4f0599628342de8d947f13fb42c433	147456
0b088e166c914d9ea0d1c6a7d0e9ab5b	118784
1c4f0599628342de8d947f13fb42c433	118784
dff75e316649495ea1f7e0990304ff99	114688
1c4f0599628342de8d947f13fb42c433	114688
f385320acb154ad8b5e6ae1708b21d00	90112
7cbf04cede6d4919991b8bb3b3afa0c0	81920
9d9570bdc3584d198d6058cfc2e18dda	81920
500ef2f149d34d89803ad09c0e21ca8b	69632
7f8e28080328431ea7d1e870ccdf526d	65536
1d6994a06aea4c609bcee6036056155f	65536
213ee96ba839454786b79b7214f2e0d2	65536
090b2bbd89274f708633ceffd19f8c93	65536
fd0b132929294473bdab326c5be75e23	61440
fadb05c91e274fd08e5c4f870677e52d	56320
8cdb55d283804264b91fab4080a75d06	53248
4f71a72505a5448891e327b423dc2039	47104
957a4cd3ee664074880a844135d4493d	40960
f2ce392c57354a319bc25c9f6fd86ce1	40960
8bcf539322bf4c66a96b1aeabd1cf5a5	40960
89186371ec24415b82644dde38e3f71e	36864
5005f3964a1a4304b65a1bf0e6046dc6	35328
1e42f804865c457fb2202e797d79f549	34816
9c5ae1aa35e44663be3908a265e02d90	34816
e69d438d44504e92b53b542fcb7647a4	32768
3e46148274424a25ad10b71db7fc74f7	32768
6dd0cb33a142496b8a205977051e63bd	32768
7a582724adfe49fa829268e7fbb36b4c	32768
20bb5b6f38164850ba1cfe0686e14a88	32768

Figura 87: Ejemplo consulta nº2 en Excel

Como se puede ver, el resultado es el mismo que el obtenido en SSAS, si se ven las cinco primeras filas obtenidas en la figura 86 se trata de los mismos campos que los obtenidos en la figura 81 que realiza la misma consulta en SSAS, pero como se puede apreciar, en Excel solamente se debe arrastrar los campos deseados a las áreas, establecer el filtro de la fecha que hace obtener las instancias no eliminadas y ordenar los valores gracias a la función de ordenar de Excel.

Una de las ventajas que ofrece Excel sobre el modelo realizado, es que es más sencillo aplicar filtros, sobre todo al grupo de medidas de Instancia ya que en este grupo de medidas las fechas no se encuentran tratadas, pero Excel permite seleccionar varios campos o filtrar un año más fácilmente mientras que si queremos hacer lo mismo en SSAS deberíamos introducir todas las fechas.

Realizando otros ejemplos de consultas más complejas utilizando diferentes dimensiones, encontramos los siguientes ejemplos de consultas:

- Cantidad de instancias por usuario y mes. Esta consulta mostrada en la figura 88 devuelve como resultado todos los usuarios que han creado instancias durante los meses de Febrero de todos los años en los que este mes ha sufrido una creación de instancias además de ofrecer la suma de todas las instancias realizadas en los meses de Febrero de los años 2018,2019 y 2020

Recuento Único Instancias	Etiquetas			
	Febrero	Febrero	Febrero	Total general
Etiquetas de fila	2018	2019	2020	
0105ea93289f403b93d533a774b22e80			2	2
076575444f934f6199523113ed63d69f	2			2
0a6bca68660d4ac49bc9230b1efa33f5			2	2
0d47121686544efa81cb9d42169405de	3			3
11004ad7c4c848e28fc4566bce60c5ae			7	7
262b9f6321cc46fe9fa5f3e8724e07f0			10	10
26838da7194345bb8f05f97031ba8d5b			1	1
2fb98e6d316448d6b93042cf2e867f58			3	3
3153f9a3989d4ae99da57c51b2255eec			1	1
3c97bbc5f54d479587f5714191ff6e5c	1		4	5
431393afc1b7479aa04fbe0ad821d490	4	13	13	30
4fe69dd0ddd94c2995acc833fcc2203e		13		13
5409a21769b944ec846f5fe147caad10	2		4	6
5f5e2b9cb34d438d9a3f75569725f323	2			2
62698a6b3c1147fdadc29bb5925bb635			9	9
63bfb9716688452d845b2f107186e58f	2			2
6843c3f40e0348f5b268666830a7cd8e	1			1
6d19e446b49348aa8feaf0468a1f9fef			1	1
6d275ded1bd646a68c913a076caa63f0			13	13
76213e0d2b4c48c8846eec7d3dd63dfd	2			2
79405564552c4054ac930cce13bcb5d4			1	1
7bb9a2fadbbf472fa94d186bec75217b			1	1
804242f76da14d1e91fa351a5afd9264			5	5
824191b812d74ec79e859ec6de00bcc3			9	9
853148e7ce054a1c8d4c97dbc12a3944	2			2
86b819b027784a3a9163b0d2c440b033			1	1
87b3143210664551be9ada1d7cf2ddf3			3	3
89d0228a0dd44170ba1f8db20449322e			1	1

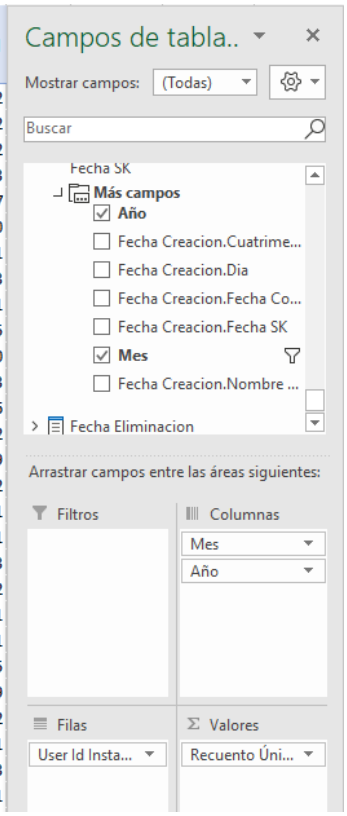
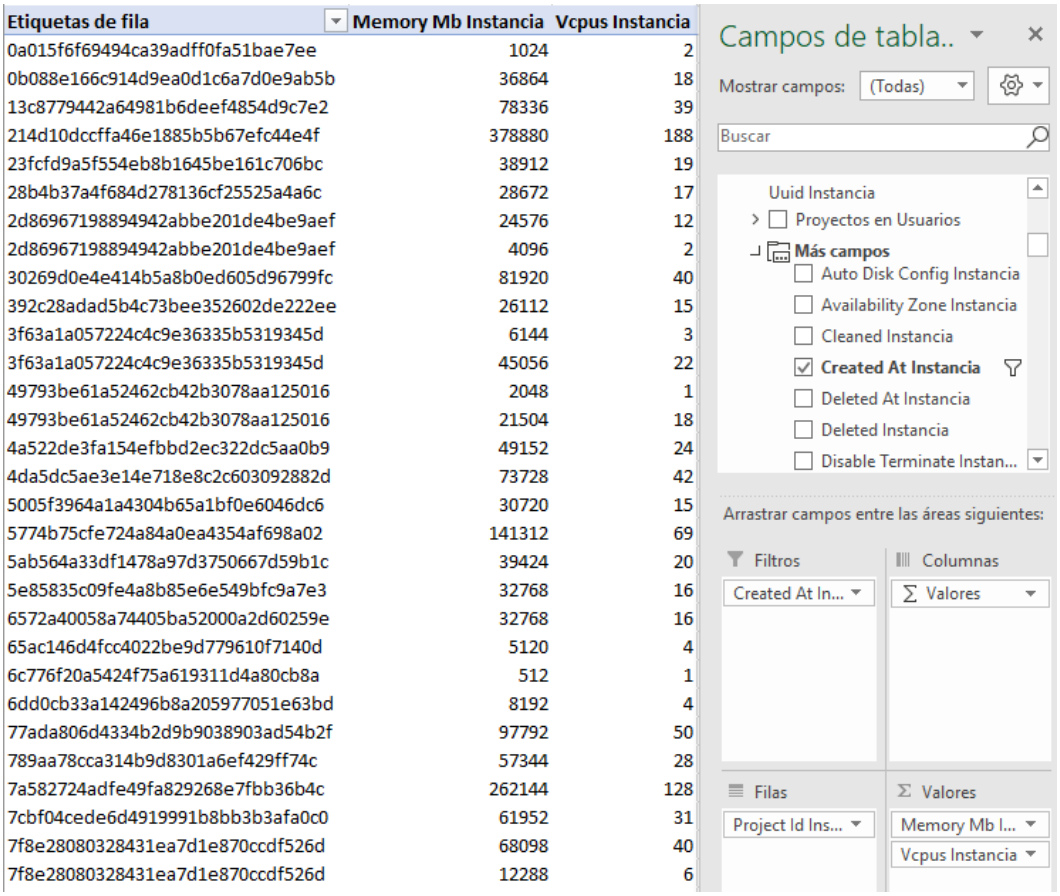


Figura 88: Ejemplo consulta nº3 en Excel

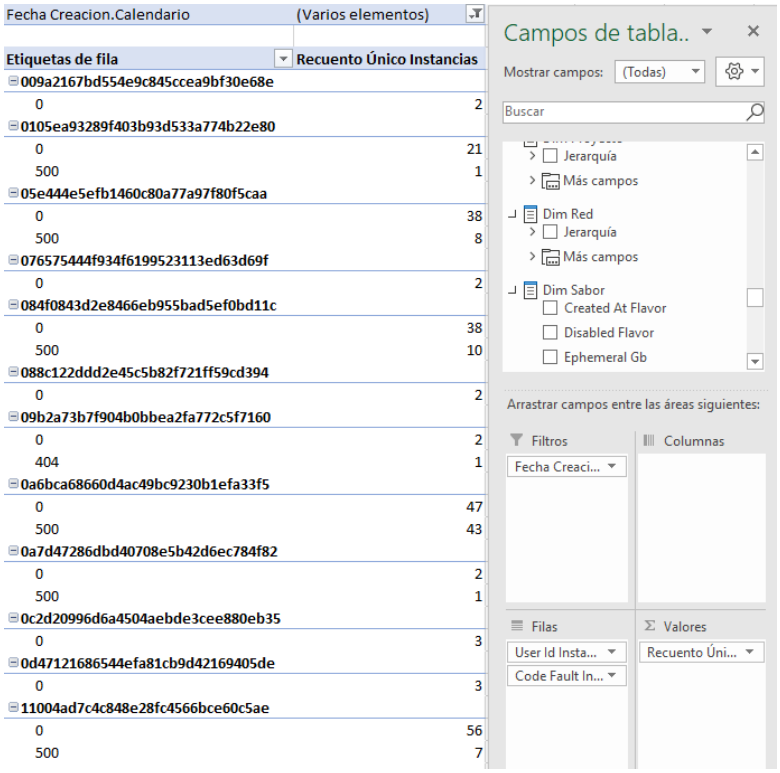
- Suma de RAM y CPUS por proyecto y año. Esta consulta muestra la sumatoria de toda la memoria y cpus que ha consumido un proyecto a lo largo de un año, en este caso, en 2017. El resultado se puede ver en la siguiente figura.



Etiquetas de fila	Memory Mb Instancia	Vcpus Instancia
0a015f6f69494ca39adff0fa51bae7ee	1024	2
0b088e166c914d9ea0d1c6a7d0e9ab5b	36864	18
13c8779442a64981b6deef4854d9c7e2	78336	39
214d10dccffa46e1885b5b67efc44e4f	378880	188
23fcfd9a5f554eb8b1645be161c706bc	38912	19
28b4b37a4f684d278136cf25525a4a6c	28672	17
2d86967198894942abbe201de4be9aef	24576	12
2d86967198894942abbe201de4be9aef	4096	2
30269d0e4e414b5a8b0ed605d96799fc	81920	40
392c28adad5b4c73bee352602de222ee	26112	15
3f63a1a057224c4c9e36335b5319345d	6144	3
3f63a1a057224c4c9e36335b5319345d	45056	22
49793be61a52462cb42b3078aa125016	2048	1
49793be61a52462cb42b3078aa125016	21504	18
4a522de3fa154efbbd2ec322dc5aa0b9	49152	24
4da5dc5ae3e14e718e8c2c603092882d	73728	42
5005f3964a1a4304b65a1bf0e6046dc6	30720	15
5774b75cfe724a84a0ea4354af698a02	141312	69
5ab564a33df1478a97d3750667d59b1c	39424	20
5e85835c09fe4a8b85e6e549bfc9a7e3	32768	16
6572a40058a74405ba52000a2d60259e	32768	16
65ac146d4fcc4022be9d779610f7140d	5120	4
6c776f20a5424f75a619311d4a80cb8a	512	1
6dd0cb33a142496b8a205977051e63bd	8192	4
77ada806d4334b2d9b9038903ad54b2f	97792	50
789aa78cca314b9d8301a6ef429ff74c	57344	28
7a582724adfe49fa829268e7fbb36b4c	262144	128
7cbf04cede6d4919991b8bb3b3afa0c0	61952	31
7f8e28080328431ea7d1e870ccdf526d	68098	40
7f8e28080328431ea7d1e870ccdf526d	12288	6

Figura 89: Ejemplo consulta nº4 en Excel

- Recuento de instancias fallidas para cada usuario de aquellas instancias creadas en 2018 y 2019. Esta consulta se puede visualizar mediante la siguiente figura.



Fecha Creacion.Calendario	(Varios elementos)
Etiquetas de fila	Recuento Único Instancias
009a2167bd554e9c845ccea9bf30e68e	
0	2
0105ea93289f403b93d533a774b22e80	
0	21
500	1
05e444e5efb1460c80a77a97f80f5caa	
0	38
500	8
076575444f934f6199523113ed63d69f	
0	2
084f0843d2e8466eb955bad5ef0bd11c	
0	38
500	10
088c122ddd2e45c5b82f721ff59cd394	
0	2
09b2a73b7f904b0bbea2fa772c5f7160	
0	2
404	1
0a6bca68660d4ac49bc9230b1efa33f5	
0	47
500	43
0a7d47286dbd40708e5b42d6ec784f82	
0	2
500	1
0c2d20996d6a4504aebde3cee880eb35	
0	3
0d47121686544efa81cb9d42169405de	
0	3
11004ad7c4c848e28fc4566bce60c5ae	
0	56
500	7

Figura 90: Ejemplo consulta nº5 en Excel

- Gráfico dinámico. El grafico de la siguiente figura muestra el estado actual de las instancias creadas en 2020.

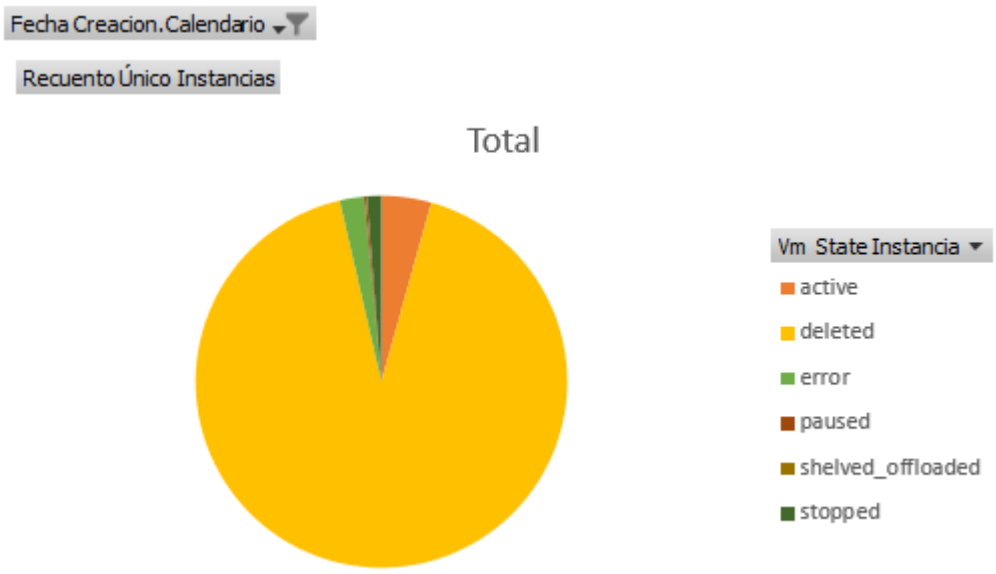


Figura 91: Ejemplo gráfico dinámico en Excel

Como se puede apreciar en las consultas, para un usuario inexperto o que desconoce el lenguaje de bases de datos, es más sencillo utilizar Excel, ya que aun desconociendo el significado de Filtro, Columna, Fila o Valor puede ir fácilmente probando hasta obtener el resultado deseado.

4. Conclusiones y trabajo futuro

En este último capítulo se comentarán las conclusiones obtenidas de la realización del proyecto y como se podría mejorar o continuar el desarrollo de este.

El objetivo de este proyecto era proporcionar una herramienta de análisis a la plataforma OpenStack-DI del departamento de informática de la universidad de Almería para poder visualizar fácilmente el comportamiento y uso de su plataforma. Como se ha visto en las herramientas de visualización ahora el departamento posee diferentes maneras de acceder a un conjunto de datos preparado para el análisis de datos.

Por otro lado, el trabajo realizado durante estos meses ha proporcionado un aumento en los conocimientos de algunas técnicas aplicadas en el área de BI. Este proyecto ha servido para desarrollar las siguientes tecnologías de BI:

- Se ha aprendido a construir por completo un Data Warehouse, teniendo como origen la unión de distintas bases de datos, a través de procesos ETL.
- Se ha aprendido a realizar procesos ETL mediante la herramienta SSIS para obtener los datos deseados, realizar ciertas transformaciones y posteriormente cargar el resultado esperado.
- Se ha conseguido crear un cubo OLAP con SSAS.

Respecto a los objetivos y planificación planteada ha sido útil incluir los mismos para marcar un ritmo de progresión adecuado. Se han cumplido todos los objetivos presentados tanto personales como a nivel de proyecto y se han cumplido los tiempos establecidos en el cronograma salvo en el desarrollo de los procesos ETL dónde se ha tenido que invertir algo más de tiempo debido a que era la primera vez que realizaba algo parecido, y no hay demasiada información útil en la que basarse, por lo que tuve que resolver bastantes problemas por mi cuenta, y, aun así, se pudieron terminar.

Sobre el resultado final cabe mencionar que ha sido satisfactorio ya que se ha obtenido lo propuesto en un principio, que es poder estudiar fácilmente el rendimiento de la plataforma, pero, por otro lado, creo que habría sido un proyecto más interesante, aunque también más extenso en su construcción, y por tanto el tiempo invertido, realizar el proyecto sobre un departamento con más procesos de negocio en los cuales hubiese más medidas y dimensiones sobre las que tratar. Aun así como se ha comentado previamente, el resultado ha sido satisfactorio.

Para finalizar las conclusiones de este proyecto hay que comentar que no termina el proyecto aquí, en este TFG, ya que en el CTFG se va a realizar una mejora presentando al usuario final una herramienta de visualización más completa que las enseñadas anteriormente, enseñando también como usar la herramienta y cómo leer los campos disponibles.

Como trabajo futuro, considero que la principal mejorar es cambiar la estructura del modelo, ya que en un principio no se consideraba tener dos grupos de medida, pero al realizar las primeras consultas se comprobó que no podían hacerse las agregaciones esperadas y hubo que crear otro grupo de medida, el cual presenta el problema de que no puede aprovechar la estructura que tiene la tabla de hechos, y solamente puede realizar las agrupaciones con filtros del propio grupo de medida.

Por tanto, una de las propuestas es cambiar el modelo en estrella a un modelo en copo de nieve, donde la tabla de hechos no repita las instancias y se puedan importar, por tanto, los valores a la tabla

de hechos para realizar agregaciones aprovechando la estructura dimensional al completo, con la desventaja de que este modelo podría presentar problemas de rendimiento.

La segunda propuesta consiste en realizar una API conectada a Analysis Services donde se realicen algunas consultas personalizadas y así poder integrar los datos en otras aplicaciones.

El futuro de esta implantación sería ofrecer también técnicas como *Machine Learning* y *Data Mining* que incrementaría aún más la calidad y el valor de la información.

Bibliografía

[1] Jiménez, P. M. (20 de Marzo de 2019). *¿Qué es el Business Intelligence? Soluciones más comunes del BI*. Recuperado el 15 de Enero de 2021, de IEBS: https://www.iebschool.com/blog/business-intelligence-ventajas-digital-business/

[2] Gustavo. (3 de Diciembre de 2020). *¿Qué es MYSQL? Explicación Detallada Para Principiantes*. Recuperado el 8 de Enero de 2021, de Hostinger Tutoriales: https://www.hostinger.es/tutoriales/que-es-mysql/

[3] Castillo, J. A. (13 de Diciembre de 2018). *Cómo instalar MYSQL en Windows 10 paso a paso*. Recuperado el 8 de 2021 de Enero, de Profesional review: https://www.profesionalreview.com/2018/12/13/mysql-windows-10/

[4] DBA. (29 de Enero de 2020). *Instalar SQL Server Windows*. Recuperado el 9 de Enero de 2021, de DBA dixit: http://dbadixit.com/instalar-sql-server-windows/

[5] Microsoft. (31 de Marzo de 2021). *Microsoft Documentación: Conexión a proyectos en Team Explorer*. Recuperado el 30 de Mayo de 2021, de Microsoft: https://docs.microsoft.com/es-es/visualstudio/ide/connect-team-project?view=vs-2019

[6] Parilli, M. (11 de Junio de 2020). *Carecterísticas del MYSQL*. Recuperado el 9 de Enero de 2021, de Tecnoinformatic: https://tecnoinformatic.com/c-programacion/caracteristicas-del-mysql/

[7] Parada, M. (23 de Noviembre de 2019). *Qué es SQL Server*. Recuperado el 10 de Enero de 2021, de OpenWebinars: https://openwebinars.net/blog/que-es-sql-server/

[8] Microsoft. (27 de Enero de 2020). *Microsoft Documentación: Escenario de Tutorial de Analysis Services*. Recuperado el 2 de Marzo de 2021, de Microsoft: https://docs.microsoft.com/es-es/analysis-services/multidimensional-tutorial/analysis-services-tutorial-scenario?view=asallproducts-allversions

[9] Orfila, X. (2 de Junio de 2017). *¿Qué es y para qué sirve el Data Warehouse a las empresas?* Recuperado el 15 de Enero de 2021, de Zeus: https://datablog.zeus.vision/2017/06/02/que-es-data-warehouse/

[10] Emilio. (18 de Julio de 2020). *Comparativa de las mejores soluciones de Data Warehouse*. Recuperado el 21 de Febrero de 2021, de TodoBI: https://todobi.com/comparativa-de-las-15-mejores-soluciones-de-data-warehouse/

[11] Adamson, C. (2010). *Star Schema The Complete Reference.* McGraw-Hill Education.

[12] Inmon, W., Strauss, D., & Neushloss, G. (2008). *DW 2.0: The Architecture for the Next Generation of Data Warehousing.* Morgan Kaufmann.

[13] Diegocond. (11 de Mayo de 2017). *Herramientas ETL y algunos ejemplos de herramientas ETL, MOLAP, Data warehouse, DataMarts*. Recuperado el Febrero de 2021, de Datos Mineria Información: https://datosmineriainformacion.wordpress.com/2017/05/22/herramientas-etl-y-algunos-ejemplos-de-herramientas-etl-molap-data-warehouse-datamarts/

[14] Anónimo. (30 de Julio de 2013). *Procesos ETL: Carga. ¿En qué consiste?* Recuperado el 13 de Febrero de 2021, de PowerData Blog: https://blog.powerdata.es/el-valor-de-la-gestion-de-datos/bid/312592/Procesos-ETL-Carga-En-qu-consiste

[15] Calvo, D. (29 de Mayo de 2017). *Tratamiento de los valores nulos*. Obtenido de Diego Calvo: https://www.diegocalvo.es/tratamiento-de-los-valores-nulos/

[16] Microsoft. (6 de Mayo de 2019). *Microsoft Documentación: Uso de cubos OLAP para análisis avanzado*. Recuperado el 4 de Marzo de 2021, de Microsoft: https://docs.microsoft.com/es-es/system-center/scsm/olap-cubes-overview?view=sc-sm-2019

[17] Microsoft. (2 de Mayo de 2018). *Microsoft Documentación: Consulta MDX*. Recuperado el 3 de Mayo de 2021, de Microsoft: https://docs.microsoft.com/es-es/analysis-services/multidimensional-models/mdx/mdx-query-the-basic-query?view=asallproducts-allversions

UNIVERSIDAD DE ALMERÍA

Resumen/Abstract

Este proyecto muestra el uso de diferentes herramientas, tecnologías y lenguajes de bases de datos para la creación de una estructura de datos sobre la plataforma OpenStack-DI del departamento de informática de la Universidad de Almería. El presente Trabajo Fin de Grado pretende ayudar al departamento en el estudio del uso de la plataforma OpenStack-DI ofreciendo una solución donde poder consultar rápidamente la información de análisis deseada. Por tanto, queda demostrado con el proyecto el potencial de análisis y la capacidad para mejorar los procesos de negocio de una empresa que tiene el área de las bases de datos, en este caso las estructuras multidimensionales.

This Project shows the use of different tools, technologies and data base languages for the creation of a data structure from OpenStack-DI platform of the IT department. This Final Degree Project aim to help the department into study the use of the OpenStack-DI platform offering a solution where the departamento can see quickly the analysis information. So, the Project shows the potential of data analysis and the capacity to improve the business processes of a company because of the help of the database área, specifically multidimensional structures.

GRADO EN INGENIERÍA INFORMÁTICA, 2020/2021