UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES

ARAGÓN

INGENIERÍA EN COMPUTACIÓN

CREACIÓN DE UN CUBO MULTIDIMENSIONAL EN PFIZER

T E S I S

QUE PARA OBTENER EL TÍTULO DE:

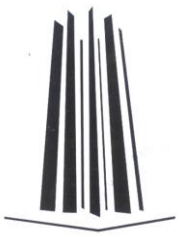
INGENIERO EN COMPUTACIÓN

P R E S E N T A:

CARLOS ALBERTO GARCÍA HURTADO

ASESOR DE TESIS

M. EN I. ARCELIA BERNAL DÍAZ



MÉXICO, 2009.

AGRADECIMIENTOS

Esta tesis está dedicada a mis padres, quienes me han dado la dicha de poder existir, quienes con todo su amor, cariño, dedicación, comprensión, sinceridad y ejemplo me han indicado el camino adecuado para salir adelante en esta vida y me siento muy orgulloso de ellos, gracias padres por todo.

Agradezco a mis hermanos por la confianza que depositaron en mí desde que era un niño, ya que con su apoyo y confianza el camino recorrido lo he disfrutado para poder salir adelante, gracias por ser un ejemplo a seguir en todo momento y por nunca dejarme solo en los momentos difíciles.

Doy gracias a Dios por cada unos de los momentos que he tenido la oportunidad de vivir y por aquellos que aun deseo seguir viviendo.

Gracias a mi prometida por apoyarme en todo y por darme la oportunidad de conocer y disfrutar el verdadero amor.

A mis amigos y compañeros les doy las gracias por su lealtad en todas mis etapas como estudiante, ya que en todo momento de cada uno ellos siempre aprendí algo diferente.

Agradezco a mis maestros y un especial reconocimiento para mi asesora Arcelia quien me apoyó, ayudó y guió para lograr concluir uno de los proyectos más importantes de mi vida.

Por último gracias a la Universidad Nacional Autónoma de México por permitir mi desarrollo como profesionista y hacer de mi una persona capaz de ofrecer algo a todos los que me rodean.

Creación de un Cubo Multidimensional en Pfizer

INTRODUCCIÓN

Objetivo General

Tener la información actualizada y confiable en un cubo multidimensional que contenga la integración de cada uno de los sistemas utilizados por el área de ventas en la empresa farmacéutica Pfizer que ayude para la toma de decisiones en distintos niveles de la compañía y con ello mejorar la productividad de la empresa.

Objetivos Particulares

- Levantamiento de información con el área involucrada tomando en cuenta los procesos de facturación, modificaciones de clientes, proveedores, productos.
- Disminuir los tiempos de entrega en los reportes.
- Ser un sistema capaz de adaptarse a los cambios de la organización, que pueda reflejar en el tiempo las necesidades analíticas futuras del negocio de manera automática.
- Proporcionar información relevante de la compañía a través de un repositorio único de información.
- Tener un sistema que integre de manera natural, el ambiente operativo (plataforma, esquema de seguridad, apariencia y política de operación) de las aplicaciones existentes en el área y permita a los diferentes usuarios acceder a la información que requieran y les corresponda analizar.
- Optimizar y automatizar las tareas que actualmente realiza el personal de operación y facturación.
- Crear un **ETL (Extraer, Transformar y Cargar)** para la carga e integración de información de los diferentes sistemas.
- Elaborar una base de datos integra y segura.
- Crear un sistema multidimensional que permita la interacción del usuario final con la Información procesada en el **Data wareHouse (Repositorio de Datos)**

CAPÍTULO I ANTECEDENTES

I.I Entorno del negocio

Hoy en día las organizaciones se enfrentan a un gran problema ya que la cantidad de información almacenada en sus bases de datos cada día es mayor, lo cual dificulta su acceso y obtención de datos relevantes e importantes para la toma de decisiones. Debido a esto en las áreas de planeación para **Business Intelligence (Inteligencia de Negocios)** el concepto de cubos multidimensionales está teniendo una gran importancia en la actualidad para el desarrollo de las empresas. Sus objetivos incluyen directamente la reducción de los costos de almacenamiento y una mayor velocidad de respuesta frente a las consultas de los usuarios ya que estos pueden ahora analizar y realizar preguntas sobre años, meses, días o bien periodos que sean requeridos incluso con la información histórica.

Pfizer es la compañía farmacéutica número uno en el mundo, actualmente cuenta con presencia en las de 150 países desarrollando actividades de investigación y desarrollo de productos farmacéuticos, tiene dos principales líneas de negocio: Salud Humana donde tiene productos en las principales áreas terapéuticas atendiendo así enfermedades cardiovasculares, enfermedades infecciosas y desorden en los sistemas nerviosos. Por otro lado está la división de Salud Animal comprometida a proveer soluciones médicas innovadoras para todas aquellas personas dedicadas a garantizar la salud y bienestar de los animales. Es una de las compañías más comprometidas y preparadas en el desarrollo científico y la investigación de medicamentos que ayudan a mantener la salud de los seres humanos en México y el mundo desde 1849 ofrece soluciones integrales para la prevención y el tratamiento de los problemas de salud, a través de ideas innovadoras en la investigación, el desarrollo y la manufactura de medicamentos de calidad, seguros y efectivos. Para ello Pfizer cuenta con más de 85 mil empleados a nivel mundial que trabajan en alianza con la comunidad médica, pacientes, autoridades y la sociedad en general.

Pfizer es una empresa líder y por tal motivo se encuentra en desarrollo constante en sus sistemas de información que permitan a la empresa tener los datos de forma veraz y actualizada en todo momento con la finalidad de mantener a la vanguardia cada uno de sus procesos y las herramientas utilizadas dentro de la compañía. Actualmente se están desarrollando nuevas estrategias que permitan un mejor desempeño y análisis de la información contenida en sus diferentes sistemas para ofrecer a sus clientes ofertas de acuerdo al comportamiento del mercado global y con ello mejorar la distribución con la finalidad de no tener perdidas hasta ahora debido a la falta de información.

I.II Planteamiento del problema

Actualmente los reportes que requieren información sobre ventas lo hacen consultando directamente sobre el sistema transaccional lo cual ocasiona que considerablemente baje el Performance de las bases de datos al estar realizando consultas continuas y con ello retardos en la operación diaria para la facturación.

Actualmente el área de ventas para salud humana en Pfizer tiene muchas carencias de acceso a la información financiera ya que en todos los casos es necesario levantar un requerimiento para cada reporte solicitado, este reporte es generado por el administrador de cada unos de los sistemas transaccionales y entregados a las áreas correspondientes según la información solicitada, posteriormente es necesario pasar por un proceso de homologación y validación de datos antes de ser entregados al usuario para poder garantizar la integridad de los datos. Este procedimiento es bastante tardado e impide tener la información cuando es requerida por los usuarios al instante lo cual influye directamente en los tiempos de entrega para la toma de decisiones y con ello se aumentan los costos de mantenimiento - operación dentro de la empresa al tener asignados varios recursos para una sola actividad.

Para solventar dicho problema se creará un cubo multidimensional que permitirá consultar y analizar la información de forma íntegra y actualizada al momento de ser

requerida por cada uno de los usuarios y con ello disminuirán considerablemente los tiempos para obtener la información deseada y con ello ayudar a la toma de decisiones dentro de la empresa.

La situación actual en el área se describe en el diagrama 1.1 donde podemos observar que la generación de un simple reporte como las ventas del mes puede tardar más de un día en caso de no tener la información integra al momento de ser requerida.

Diagrama 1.1: Situación actual para la solicitud de nuevos reportes dentro del área de ventas.

I.III Propuesta

Tener un proceso de extracción, transformación y carga de la información que ayude a mejorar el proceso de análisis de los datos en forma automatizada y que les permita en el área establecer objetivos por medio de una consulta ágil y flexible, teniendo la capacidad de comparar diferentes métricas dentro de la empresa. Así mismo, generar un cubo multidimensional que ayude a reducir el tiempo en el análisis y consulta de información ya que se podrán obtener consultas mensuales, trimestrales y anuales con un detalle de productos en sus diferentes niveles en pesos, dólares y aquellas métricas que sean requeridas por lo usuarios, dentro de la realización del proyecto se hará un

análisis detallado de las fuentes de información que actualmente se tienen con la finalidad de crear un proceso que permita la integración de dichas fuentes.

Este sistema deberá estar aislado del sistema transaccional para garantizar la efectividad, rapidez e integridad cuando los usuarios requieran consultar información relevante dentro de la empresa.

El proceso tendrá la capacidad de cargar la información de ventas diarias así como la integración con la información proporcionada por el área de finanzas para la carga del presupuesto y con ello el cubo multidimensional tenga la capacidad de mostrar la información de ventas ligada al presupuesto para poder realizar el análisis necesario que ayude a la toma de decisiones importantes para un mejor desempeño del corporativo.

Para lograr lo anterior el sistema deberá tener las siguientes características: integración del sistema, análisis flexible y reportes comprensivos, tomando en cuenta que la información proviene de diferentes fuentes lo cual hace complejo su análisis:

- **Integración del sistema**

  Datos transferidos desde sistemas de apoyo dispares o archivos de texto promoviendo la integridad de los datos, así mismo, debe ser una plataforma escalable que cumpla con las necesidades de análisis de información actual y futura con la finalidad de poder realizar proyecciones o estimaciones en beneficio de la compañía.

- **Análisis Flexible**

  Incluir el detalle de ventas, escenarios, análisis dimensional y la habilidad de satisfacer requerimientos únicos de comparativos entre versiones y cálculos de las diferentes métricas existentes.

- **Reportes comprensivos**

  Los propios usuarios deben tener la capacidad de realizar sus propios reportes en el cubo multidimensional con la flexibilidad que requieren de acuerdo a sus

requerimientos y necesidades, obteniendo la información que necesitan de manera rápida y oportuna cuando esta sea requerida tomando en cuenta que la información debe mostrarse de forma íntegra.

El diagrama 1.2 muestra la estructura que se está implementando para la consolidación de las diferentes fuentes pasando por un proceso de extracción, transformación y carga lo cual nos permitirá crear reportes y cubos multidimensionales para la interacción con el usuario final.

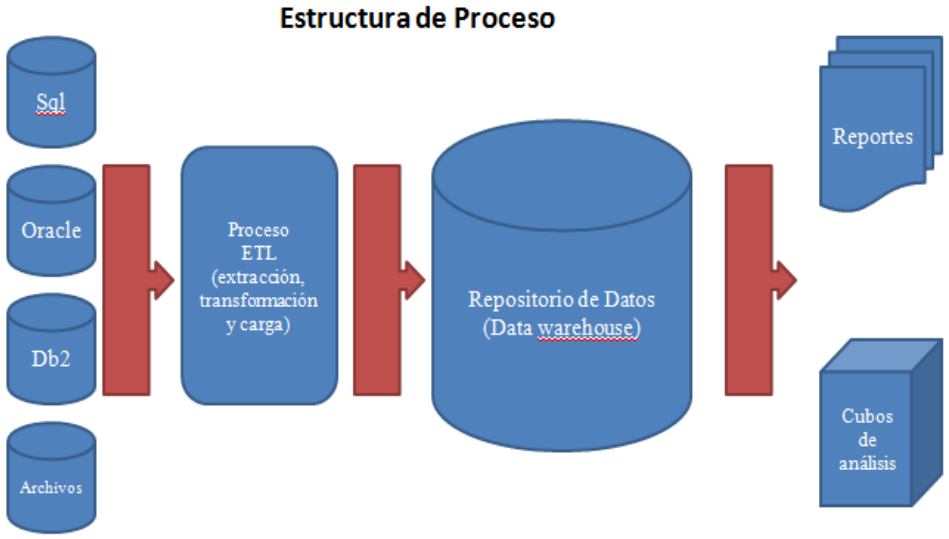


Diagrama 1.2: Muestra el flujo de los procesos para extraer, transformar y cargar la información para lograr la interacción con el usuario final

En este diagrama se observan los beneficios de utilizar un **ETL (Extraer Transformar y Cargar)** capaz de interactuar con diferentes fuentes o bases de datos sin importar su origen como SQL, Oracle o archivos planos ya que de esta forma se podrá integrar la información y transformarla para posteriormente cargarla en nuestro repositorio de datos el cual nos servirá como base para poder generar nuestro cubo dimensional al igual que los reportes solicitados por los usuarios para realizar un análisis dinámico.

Es importante mencionar que el análisis de información con cubos multidimensionales está teniendo grandes beneficios para las empresas ya que de forma ágil y rápida permite a los niveles gerenciales consultar la información de forma dinámica con lo cual ayuda a la toma de decisiones para poder generar un mejor desempeño de la empresa en base a la información obtenida y proyecciones realizadas.

Posteriormente en el capítulo 2 se explicará la utilización que actualmente tienen los cubos multidimensionales para entender el porqué la necesidad de está implementación dentro del área de ventas para Pfizer, por otro lado en el área se ha encontrado que son muchas las necesidades que tienen en cuanto a la información, para definir claramente en qué consiste el tipo de análisis que desean hacer se llegó a los siguientes puntos clave donde se puede observar la conformación de las necesidades esenciales para el cumplimiento de las peticiones realizadas por el área en cuanto a los datos.

I.IV Necesidades de información

Las siguientes dimensiones y métricas serán mostradas en el cubo para permitir un análisis adecuado de la información y con ello ayudar a la toma de decisiones dentro de la empresa partiendo de la información general a lo particular según, ya que en el cubo se tiene la facilidad de desglosar por niveles según se requiera, las siguientes definiciones será la información esencial que deberá contener el cubo multidimensional para poder realizar un análisis de los datos adquiridos.

- Venta Nacional.
  Contiene la información de ventas de todo el territorio donde actualmente tiene presencia Pfizer a nivel nacional, con la finalidad de obtener las ventas de cada región.

- Venta por División.

El área de ventas actualmente tiene el territorio dividido en diferentes divisiones de acuerdo a su ubicación lo cual es necesario mostrar en el cubo para tener un control adecuado.

- Venta por Línea Terapéutica.
  Cada división está dividida en diferentes líneas terapéuticas lo cual se verá en el cubo como un nivel más de acuerdo a las necesidades de los usuarios ya que es importante identificar aquellas divisiones que están vendiendo más.

- Venta por Grupo de Productos y Producto.
  Los productos actualmente están divididos por grupos según las definiciones del negocio, por tal motivo el cubo tendrá la división y subdivisión de los productos.

- Información de Venta por Cliente.
  Se tendrá una dimensión de clientes que ayudará a identificar a cuales clientes se está vendiendo mas, o bien, donde se están teniendo perdidas considerables de acuerdo al periodo, producto, o bien, lo que sea requerido.

- Información de Venta Privada.
  El mercado farmacéutico está dividido en dos grandes mercados y por ello el cubo mostrará la información del mercado de acuerdo a lo solicitado.

Actualmente el área de ventas maneja más de tres fuentes de información distintas debido a que las necesidades del negocio así lo requiere y para consolidar esta información en un repositorio de datos (base de datos) es necesario homologar cada una de las fuentes que tienen el área por lo cual se debe implementar un **ETL (Extraer Transformar y Cargar)** que permita cargar la Información en la base de datos y con ello integrar los diferentes sistemas como uno solo para que pueda ser consultado

cuando sea requerido por los usuarios, además que ayudara a tener la información actualizada al terminar la facturación diaria.

El ETL es un proceso que permite a las organizaciones mover datos desde múltiples fuentes, reformatearlos, limpiarlos, homologarlos y cargarlos en otra base de datos, o bien, en otro sistema operacional para apoyar un proceso de negocio, en este caso será utilizado para cargar las tablas que se utilizarán en nuestro cubo multidimensional.
Actualmente la interfaz de entrada contemplada es el sistema de facturación para catálogos de productos, territorios, clientes y datos reales de venta diaria, por medio de Excel se realizarán las agrupaciones de productos y territorios, así como la alimentación de los datos del presupuesto, mientras que la interfaz de salida será un cubo multidimensional, en caso de requerir información adicional de otra fuente el ETL permitirá cargar los nuevos datos siempre y cuando cumplan con las necesidades mínimas de información para ser considerado dentro del cubo multidimensional.
El Diagrama 1.3 esquematiza las entidades externas, sistemas externos y el flujo de datos (entradas y salidas) desde y hacia el sistema.

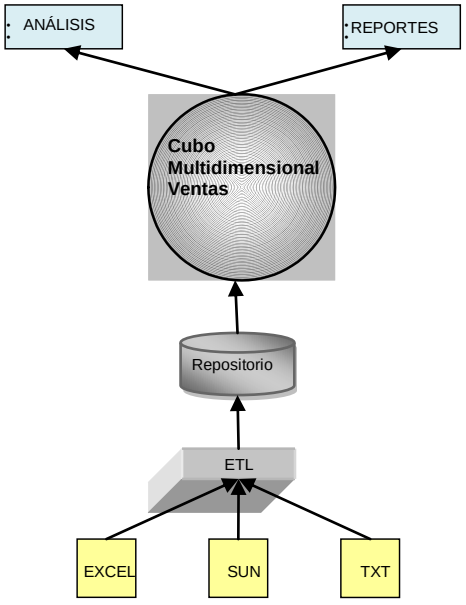


Diagrama 1.3: Muestra el proceso de carga para crear u cubo multidimensional

I.V Requerimientos de hardware y software

Para garantizar el correcto funcionamiento de nuestra aplicación es necesario tener como mínimo los requerimientos de sistema indicados a continuación ya que de esta manera se podrán integrar cada unos de los procesos involucrados para la creación de nuestro cubo multidimensional:

Software

- Sistema Operativo
  Windows NT, WindowsXP o Windows2000 Server
- Base de Datos
  SQL Server 2000
- Herramientas Cognos
  DecisionStream (ETL)
  Impromptu (reporteador)
  PowerPlay (cubo multidimensional)

Hardware

- Memoria RAM de 2 Gb
- Espacio en Disco Duro100 Gb

Actualmente la compañía tiene la infraestructura necesaria para implementar nuestro desarrollo ya que podemos explotar al máximo las capacidades de hardware y software obtenidas por lo cual el cubo multidimensional tendrá como objetivo cumplir con las necesidades de los usuarios en tiempo y forma garantizando ante todo la integridad de la información en el momento de ser requerida para su consulta y con ello apoyar a la toma de decisiones dentro de la organización.

En el capítulo II observaremos los beneficios de utilizar un cubo multidimensional y la importancia que tiene su aplicación dentro de las empresas que manejan grandes cantidades de información.

CAPÍTULO II Cubo Multidimensional

II.I Procesamiento analítico en línea

Es un almacén de datos donde se realizan operaciones de procesamiento analítico en línea **(OLAP – On-Line Analytical Processing)**, cuya operación consiste principalmente de consultas sobre grandes volúmenes de datos y de proveer una interfaz en línea que ofrece reportes y gráficos. OLAP es una tecnología que permite a los analistas y administradores visualizar y navegar los datos accediendo a una amplia variedad de vistas posibles de la información de manera interactiva, rápida y eficiente.

II.II Data warehouse y data marts

Por lo general, los sistemas operacionales tienen requerimientos de rendimiento estrictos, cargas de trabajo predecibles, pequeñas unidades de trabajo y una alta utilización. Por el contrario, los sistemas de apoyo para la toma de decisiones tienen por lo general requerimientos de rendimiento variantes, cargas de trabajo impredecibles y grandes unidades de trabajo.

La importancia de un sistema hecho para la toma decisiones radica en que tiene la capacidad de recolectar información de diferentes sistemas operacionales y estar en un almacén de datos propio e independiente, a ese almacén de datos es conocido como un data warehouse.

- **Data warehouse**

Al igual que los almacenes de datos operacionales, un data warehouse es un tipo especial de base de datos. Al parecer el término se originó a finales de los ochenta aunque el concepto es de alguna manera más antiguo.

Un data warehouse se define como un almacén de datos orientado a un tema, integrado, no volátil y variante en el tiempo, que soporte decisiones de administración.

Los data warehouses surgieron principalmente por dos razones: primero, la necesidad de proporcionar una única fuente de datos limpia y consciente para sus propósitos de

apoyo para la toma de decisiones; segundo, la necesidad de hacerlo sin afectar a los sistemas operacionales.

- **Data marts**

Por lo general, los data warehouses están hechos para proporcionar una fuente de datos única para todas las actividades de apoyo para la toma de decisiones. Sin embargo, cuando los data warehouses se hicieron populares se empezaron a notar tendencias donde los usuarios a menudo realizaban amplias operaciones de informes y análisis de datos sobre un subconjunto relativamente pequeño de todo el data warehouse.

Por tal motivo, la ejecución repetida de tales operaciones sobre el mismo subconjunto de todo el almacén no era muy eficiente, por lo tanto, se inició con la creación de algún tipo de almacén limitado de propósito general que estuviera hecho a la medida de algún tema en específico, con esto se proporcionaba un acceso más rápido a los datos y fue cuando surgió el concepto data mart.

Debido a esto entenderemos por data mart un almacén de datos especializado, orientado a un tema, integrado, volátil y variante en el tiempo orientado para apoyar un subconjunto específico de decisiones de administración.

II.III Modelo Multidimensional

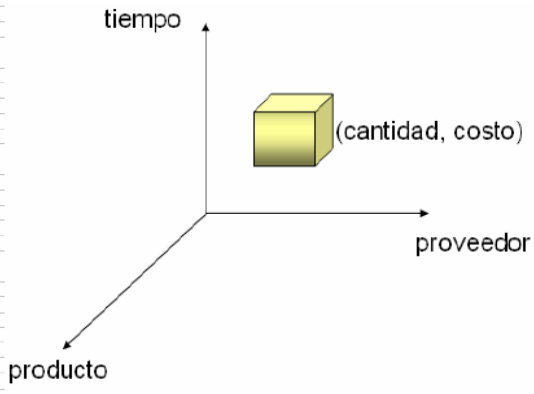
Un modelo de datos multidimensional es una colección de medidas las cuales dependen de un conjunto de dimensiones, o bien, es una representación de los datos que permite organizarlos en la forma de hechos, dimensiones y agregados, donde los hechos contienen medidas, es decir, la información a nivel de transacción que vamos a analizar, por ejemplo: compras, ventas, gastos, etc.

Las dimensiones contienen información descriptiva de esas transacciones como pueden ser: fecha, cliente, producto, etc. en esencia un modelo multidimensional se utiliza principalmente para el análisis de información que ayude a la toma de decisiones.

El diagrama 2.1 muestra el concepto de modelo multidimensional en el área de compras, donde a la izquierda se muestran los datos transaccionales de base, a la derecha y abajo se tiene un diagrama de cubo donde hay dos medidas a analizar: la cantidad y el costo, estos atributos o campos formarán la tabla de hechos de nuestro cubo y serán analizados mediante tres dimensiones.

La dimensión producto que contiene los atributos del producto que se compró, la dimensión proveedor que contiene los atributos del proveedor a quien se le compró ese producto y finalmente la dimensión tiempo que contiene la fecha en la que se realizó esa transacción de compra.

	FECHA	COMPRA-$	CANTIDAD	PROVEEDOR	PRODUCTO
2	Ene-04	1302	2	AMAZON	LIBRO
3	Ene-04	232	34	GANDHI	REVISTA
4	Ene-04	234	2	BARNES	REVISTA
5	Feb-04	2	46	CRYSTAL	LIBRO
6	Feb-04	486	3	GANDHI	LIBRO
7	Mar-04	746	2	CRYSTAL	LIBRO
8	Mar-04	224	8	AMAZON	REVISTA
9	Abr-04	1248	5	CRYSTAL	LIBRO
10	Abr-04	121287	32	GANDHI	REVISTA
11	May-04	41212	6	CRYSTAL	LIBRO
12	May-04	122	21	CRYSTAL	LIBRO
13	Jun-04	1545	1	AMAZON	LIBRO
14	Jun-04	4545	3	SOTANO	REVISTA
15	Jul-04	4545	45	SOTANO	LIBRO
16	Jul-04	1212	21	SOTANO	REVISTA
17	Jul-04	4548	1	AMAZON	LIBRO
18	Ago-04	121285	3	SOTANO	LIBRO
19	Ago-04	42	6	GANDHI	REVISTA
20	Sep-04	1212	2	AMAZON	LIBRO
21	Oct-04	21245	1	GANDHI	REVISTA
22	Oct-04	1515	5	SOTANO	REVISTA
23	Nov-04	07054	2	SOTANO	LIBRO
24	Nov-04	6668	1	SOTANO	LIBRO
25	Nov-04	266	4	AMAZON	REVISTA
26	Dic-04	56497	1	AMAZON	LIBRO
27	Dic-04	266	5	BARNES	LIBRO



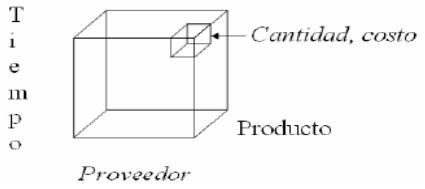


Diagrama 2.1: Muestra los conceptos de dimensión y hechos

Se tienen diferentes esquemas relacionales para hacer el modelado multidimensional, ver la figura 2.2 donde se muestran los tres esquemas: estrella, copo de nieve y constelación que a continuación se describirán para poder comprender el beneficio y la utilización de cada uno de ellos, considerando las necesidades del negocio requeridas en cuanto a la información.

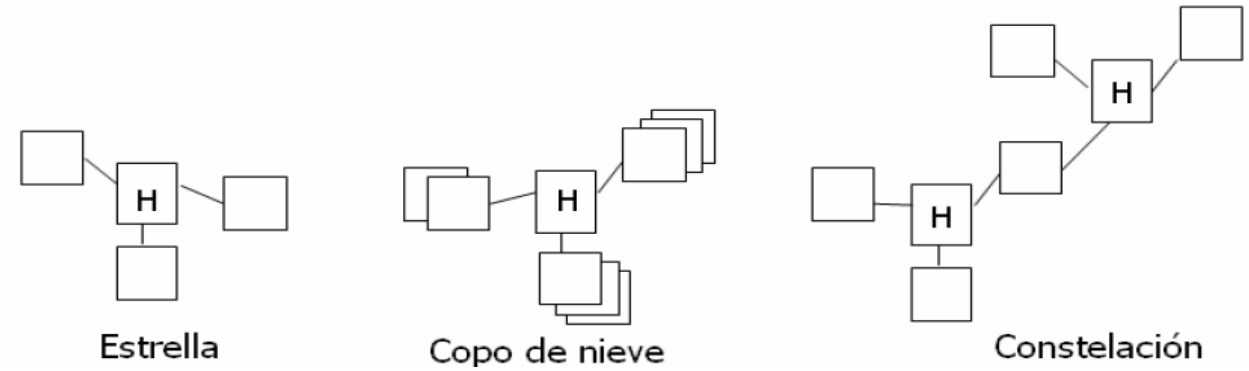


Figura 2.2: Esquemas para representar modelos multidimensionales

- **Modelo o esquema estrella tradicional**

El esquema estrella es una técnica de modelado de datos usada para hacer corresponder un modelo multidimensional a una base de datos relacional, su nombre se debe básicamente a que gráficamente parece una estrella. El esquema de estrella tiene cuatro componentes: hechos, dimensiones, atributos y jerarquías de atributos, donde, los hechos y dimensiones son representados por tablas físicas en el almacén de datos, la tabla de hecho esta relacionada a cada dimensión en una relación uno a muchos.

- **Modelo copo de nieve**

El esquema copo de nieve es una variación de la estrella tradicional, lo que se hace es que en cada dimensión se almacenan jerarquías de atributos, o bien, simplemente se separan atributos en otra entidad por razones de desempeño y mejor utilización del espacio.

- **Modelo de constelación**

El modelo de constelación nuevamente es una variación del esquema estrella tradicional, solo que en este modelo algunos atributos de las dimensiones se separan

formando una nueva entidad que puede ser compartida con otros cubos, es decir, pueden ser dimensiones de otros cubos.

II.IV Cubo Multidimensional

Consiste en una representación multidimensional de datos de detalle y resumen, por lo que un cubo tiene como objetivo mejorar el rendimiento empresarial dentro de una organización ya que es un subconjunto de datos de la base de datos original, además son capaces de administrar de forma rápida y eficiente grandes cantidades de información. Este tipo de aplicaciones es utilizada principalmente por el personal responsable de preparar y diseñar informes que sean presentados a un ejecutivo o director y esto ayude con la toma de decisiones acertadas que permitan alcanzar metas bien definidas para el negocio, de igual forma los cubos pueden ser utilizados con información que puede ser presentada para externos, tales como grupos financieros, inversionistas y analistas de mercado que puedan explorar sobre los datos financieros de la empresa.

Los componentes que integran un cubo nos muestran la importancia de tener bien definido el tipo de información que estaremos mostrando a los usuarios.

- **Métricas o indicadores**

Para conocer el estado actual de un negocio se necesitan identificar y medir diferentes aspectos para poder responder a preguntas que generalmente tienen los usuarios, tales como, cuanto se ha vendido durante el año, cual es el porcentaje de cobertura comparado con su presupuesto, en que región las ventas están bajando, etc.
Una métrica es precisamente el valor que representa un aspecto del negocio el cual puede consultado según las necesidades de los usuarios.

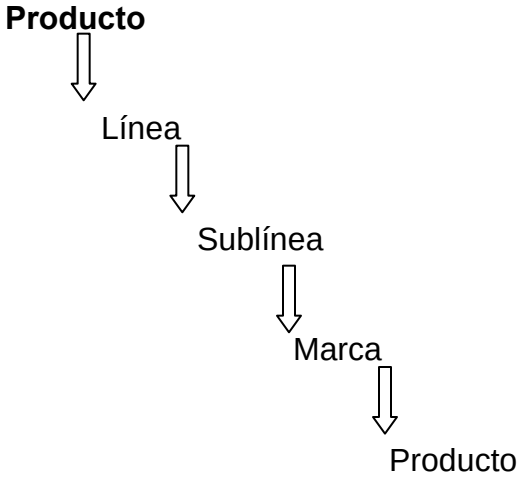
- **Dimensiones**

Es el enfoque bajo el cual se está evaluando un indicador, esto es, que se podrá definir un indicador como ventas netas, pero este valor se desea ver desglosado por tipo de

cliente y luego por zona. Al tipo de cliente y zona es lo que llamamos dimensiones dentro de nuestro cubo multidimensional.

- **Jerarquías**

  Se refiere al orden lógico en el que se desglosa normalmente un indicador por más de una dimensión, suponiendo el caso donde tenemos las dimensiones de Marca, Línea, Sublínea, Producto, normalmente los ejecutivos analizan con cierto orden pre-establecido al indicador de ventas, se puede definir una jerarquía con el nombre de producto con el siguiente orden de acuerdo a las necesidades del negocio.



Así como esta jerarquía se pueden crear varias que ayuden al análisis de los datos según sean las necesidades del los usuarios en el área.

II.V Operaciones con cubos multidimensionales

Una vez que ya se tienen los cubos se pueden realizar diferentes operaciones sobre ellos para visualizar y analizar la información como las siguientes:

- **Generalizar y especializar**

Son operaciones que sirven para navegar un cubo sobre sus dimensiones, donde, las dimensiones que ofrecen diferentes niveles de abstracción pueden ser navegables, por ejemplo supongamos que tenemos nuestra dimensión de tiempo donde tenemos los

niveles de abstracción: año, semestre, mes, semana y día, una operación de especialización nos permitirá interactivamente visualizar los hechos desde el agregado total por año e ir descendiendo hasta el detalle por día, mientras que la operación generalizar permitirá lo contrario ya que desde cualquier nivel podrá generalizar a un nivel superior como se muestra en la figura 2.3.

Figure 2.3: Muestra las operaciones sobre el cubo.

- **Corte de cubo**

Sirve para ver subconjuntos de cubos ya que como resultado generan cubos de menor proporción de acuerdo a los datos solicitados. La operación corte como la palabra lo indica es una operación que genera una parte de cubo, por ejemplo, cuando en la dimensión de tiempo indicamos el mes de marzo para cierto proveedor y un producto definido, esto nos genera solo una parte de toda la información que posee nuestro cubo.

- **Filtrar y pivotear**

Filtrar consiste en realizar una selección sobre los datos de un cubo utilizando una constante, mientras que la operación pivotear sirve para visualizar desde distintos ángulos el cubo ya que permite rotar los ejes del cubo para examinarlo desde diferentes puntos de vista.

En al capítulo III se obtendrán los elementos necesarios para crear nuestra base de datos que servirá para mostrar los datos en nuestro cubo multidimensional.

CAPÍTULO III Análisis y diseño de la base de datos para Pfizer

III.I Base de Datos

Una base de datos es una colección de información organizada donde se muestra un conjunto de datos relacionados que se almacenan de forma que se pueda acceder a ellos de manera sencilla como se muestra en la figura 3.1, con la posibilidad de relacionarlos, ordenarlos en base a diferentes criterios, además es un conjunto de información almacenada en memoria auxiliar que permite acceso directo y un conjunto de programas que manipulan esos datos.

Base de Datos también se puede definir como un conjunto exhaustivo no redundante de datos estructurados organizados independientemente de su utilización y su implementación en máquina accesibles en tiempo real y compatibles con usuarios concurrentes con necesidad de información diferente y no predicable en tiempo.

Es una colección de datos estructurada y organizada para permitir el rápido acceso a la información de interés y los elementos que la forman se denominan registros los cuales, a su vez, están compuestos por campos.

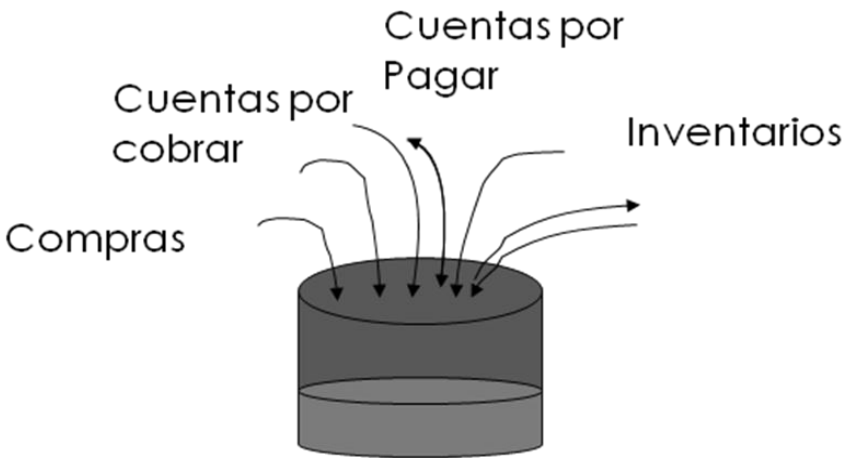


Figura 3.1: Muestra la conectividad que puede tener una base de datos para almacenar información mediante interfaces de usuario.

III.II Gestor de Bases de Datos

Seleccionar el correcto gestor de base de datos es de vital importancia para el funcionamiento correcto de las aplicaciones, ya que es donde se almacena toda la información relacionada con una empresa para poder ser consultada.

Un gestor es un conjunto de programas no visibles al usuario final que se encargan de la privacidad, integridad y seguridad de los datos, así como la interacción con el sistema operativo, proporciona una interfaz entre los datos, los programas que los manejan y los usuarios finales, por esta razón cualquier operación que el usuario hace contra la base de datos está controlada por el gestor, internamente almacena una descripción de datos lo cual es llamado como diccionario de datos al igual que los permisos de usuarios permitidos y no permitidos.

Para la creación de nuestro repositorio utilizaremos SQL debido a que tiene grandes ventajas que explicaremos a continuación, esto en comparación con los demás gestores de bases de datos como: *Oracle*, *Sybase ASE*, *PostgreSQL*, *Interbase*, *Firebird* o *MySQL,* por otro lado el software de Pfizer está basado en la tecnología Microsoft lo cual reduce los costos considerablemente ya que no será necesario utilizar otra tecnología y se pueden explotar los recursos actuales de la empresa, principalmente por tener las características necesarias para nuestro repositorio de datos.

III.3 Beneficios de SQL Server

A continuación se explican las principales razones para elegir nuestro gestor de bases de datos SQL server, ya que SQL server es un sistema administrador para Bases de Datos relacionales basadas en la arquitectura Cliente / Servidor (RDBMS) que usa Transact-SQL para mandar peticiones entre un cliente y el SQL server.

SQL Server usa la arquitectura Cliente / Servidor como se muestra en la figura 3.2 para separar la carga de trabajo en tareas que corran en computadoras tipo Servidor y tareas que corran en computadoras tipo Cliente:

Figura 3.2: Funcionamiento lógico del gestor de base de datos para interactuar con el usuario final.

- El Cliente es responsable de la parte lógica y de presentar la información al usuario. Generalmente, el cliente corre en una o más computadoras Cliente, aunque también puede correr en una computadora Servidor con SQL Server.

- SQL Server administra Bases de Datos y distribuye los recursos disponibles del servidor (tales como memoria, operaciones de disco, etc) entre las múltiples peticiones.

El Sistema Administrador para Bases de Datos Relacionales (RDBMS) es parte esencial de SQL server ya que mediante su gestión permite:

- Mantener las relaciones entre la información y la Base de Datos.
- Asegurarse de que la información es almacenada correctamente, es decir, que las reglas que definen las relaciones ente los datos no sean violadas.

- Recuperar toda la información en un punto conocido en caso de que el sistema falle.

SQL se encuentra totalmente integrado con Windows NT y toma ventaja de muchas de sus características como las que mencionaremos a continuación:

- **Seguridad**

  SQL Server está integrado con el sistema de seguridad de Windows NT. Esta integración permite ingresar tanto a Windows NT como a SQL Server con el mismo nombre de usuario y password. Además SQL Server usa las características de encriptación que Windows NT para la seguridad en red. SQL Server está provisto de su propia seguridad para clientes no-Microsoft

- **Soporte multiprocesador**

  SQL Server soporta las capacidades de multiprocesamiento simétrico (SMP) de Windows NT. SQL Server automáticamente toma ventaja de cualquier procesador adicional que sea agregado al Servidor

- **Servicios de Windows NT**

  SQL Server corre como un servicio dentro de Windows NT, permitiendo operarlo remotamente

- **Microsoft clúster server**

  Es un componente de Windows NT Enterprise Edition. Soporta la conexión de dos servidores, o nodos, en un clúster para aumentar las habilidades y tener un mejor manejo de la información y las aplicaciones. SQL Server trabaja en conjunto con el clúster Server para intercambiar papeles automáticamente en caso de que el nodo primario falle.

Debido a estas características utilizaremos SQL server para el almacenamiento de la información con la finalidad de tener una administración local en caso de ser requerida

y por otro lado permite la interacción con nuestro cubo multidimensional para mostrar los datos al usuarios final.

III.IV Modelo relacional

El modelo relacional de datos es un modelo simple, potente y formal para representar la realidad, también ofrece una base firme para enfocar y analizar formalmente muchos problemas relacionados con la gestión de bases de datos, como el diseño de la base de datos, redundancia (almacenamiento de los mismos datos varias veces en diferentes lugares) y la distribución. El elemento básico del modelo es la relación, por lo cual un esquema de bases de datos relacional es una colección de definiciones de relaciones. El esquema de cada relación es una agregación de atributos. Un caso de relación es una tabla con filas y columnas, donde, las columnas de las relaciones corresponden a los atributos; y las filas denominadas tuplas (conjunto de datos) son colecciones de valores tomados de cada atributo.

Para que una tabla cumpla con los requisitos de la estructura relacional debe cumplir las siguientes condiciones:

- Debe tener un solo tipo de filas dándole un formato que debe mantenerse durante todo su el desarrollo.
- Cada fila debe ser única y no pueden existir filas duplicadas.
- Cada columna debe ser única y no deben existir columnas duplicadas.
- Cada columna debe ser definida con un nombre específico.
- El valor de una columna para una fila debe ser único, no pueden existir múltiples valores en una posición de la columna.

Claves (llaves): Para acceder a la información en las tablas relacionales se debe elegir una clave que distinga a una tupla en particular, la clave se forma tomando un atributo o un conjunto de ellos. Como condición de que exista una tabla relacional no pueden existir filas duplicadas lo que implica que siempre debe existir al menos una clave. Para encontrar la llave adecuada se deben buscar en los dominios de los atributos junto con

los enlaces conceptuales que existen entre ellos con la finalidad de encontrar valores determinados que identifiquen la tupla de la tabla. No se debe buscar la clave en los valores concretos de las tuplas sino que en todos los posibles valores de los atributos, es decir su dominio.

La figura 3.3 muestra nuestro modelo relacional que permitirá ver la relación e integridad que tenernos con nuestras diferentes tablas, y se utilizará como referencia para crear la base de datos de nuestro data warehouse, esto nos ayudará a comprender de manera más sencilla el comportamiento de cada tabla.

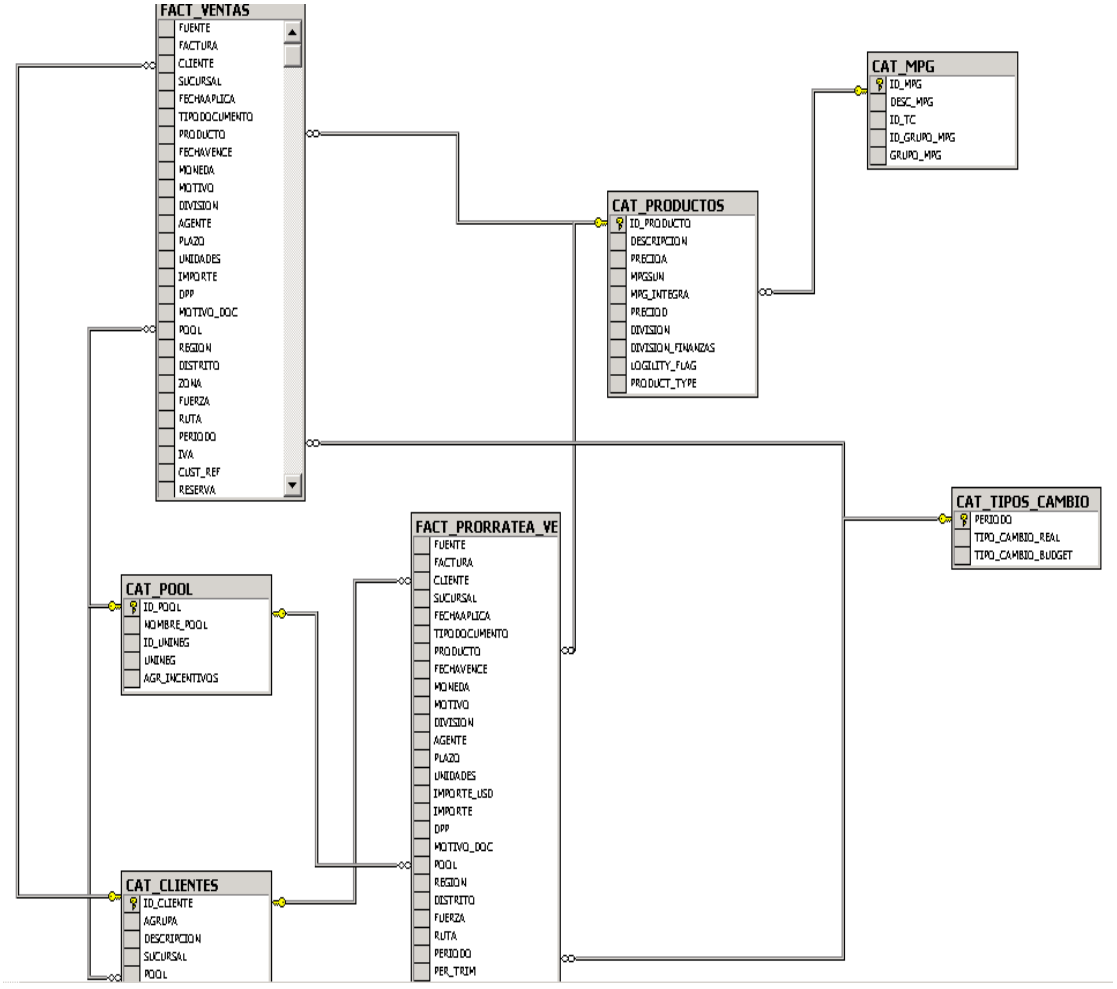


Figura 3.3: Modelo Relacional de la base de datos creada como repositorios de datos para el cubo multidimensional de Pfizer

III.V Arquitectura cliente – servidor

Una arquitectura es una serie de elementos funcionales que aprovechando diferentes estándares, convenciones, reglas y procesos, permite integrar una amplia gama de productos y servicios informáticos, de manera que pueden ser utilizados eficazmente dentro de la organización, para seleccionar el modelo de una arquitectura, hay que partir del contexto tecnológico y organizativo del momento ya que la arquitectura Cliente/Servidor requiere una determinada especialización de cada uno de los diferentes componentes que la integran.

Cliente es el que inicia un requerimiento de servicio, donde, el requerimiento inicial puede convertirse en múltiples requerimientos de trabajo y sin importar la ubicación de los datos o de las aplicaciones la información obtenida es totalmente transparente en base a la solicitud realizada al servidor.

Servidor es cualquier recurso de cómputo dedicado a responder a los requerimientos del cliente, los servidores pueden estar conectados a los clientes a través de redes para proveer de múltiples servicios a los clientes tales como impresión, acceso a bases de datos, fax, procesamiento de imágenes, reportes, etc.

La arquitectura cliente – servidor se divide en dos partes claramente diferenciadas, la primera es la parte del servidor y la segunda la de un conjunto de clientes, normalmente el servidor es una máquina bastante potente que actúa de depósito de datos y funciona como un sistema gestor de base de datos (SGBD), por otro lado los clientes suelen ser estaciones de trabajo que solicitan varios servicios al servidor. Ambas partes deben estar conectadas entre sí mediante una red de comunicaciones.
Una representación gráfica de este tipo de arquitectura sería como se muestra en el diagrama 3.4.

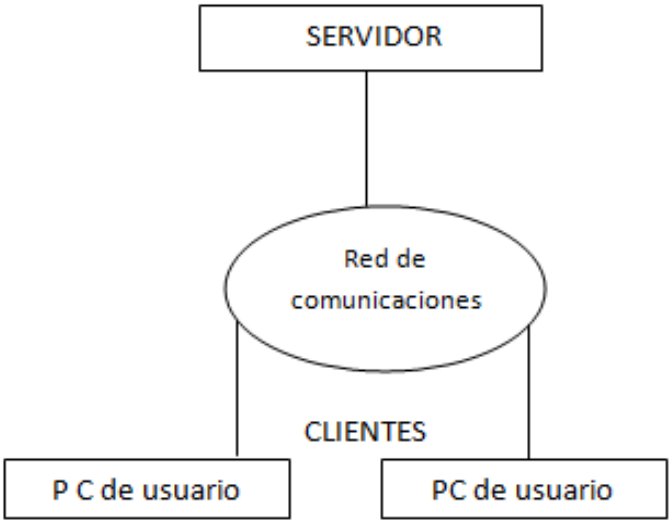


Diagrama 3.4: Representación de la arquitectura cliente – servidor

Este tipo de arquitectura es la más utilizada en la actualidad, debido a que es la más avanzada y la que mejor ha evolucionado en estos últimos años, en general esta arquitectura necesita tres tipos de software para su correcto funcionamiento:

- Software de gestión de datos: Este software se encarga de la manipulación y gestión de los datos almacenados y requeridos por las diferentes aplicaciones, normalmente este software se aloja en el servidor.
- Software de desarrollo: este tipo de software se aloja en los clientes y solo en aquellos que se dedique al desarrollo de aplicaciones.
- Software de interacción con los usuarios: También reside en los clientes y es la aplicación gráfica de usuario para la manipulación de datos, siempre a nivel usuario (consultas principalmente).

Ventajas de la arquitectura cliente – servidor

El modelo cliente/servidor en particular es recomendable para redes que requieran un alto grado de fiabilidad, donde, las principales ventajas son:

- Recursos centralizados: debido a que el servidor es el centro de la red, puede administrar los recursos que son comunes a todos los usuarios, por ejemplo: una base de datos centralizada se utilizaría para evitar problemas provocados por datos contradictorios y redundantes.
- Seguridad mejorada: ya que la cantidad de puntos de entrada que permite el acceso a los datos no es importante.
- Administración al nivel del servidor: ya que los clientes no juegan un papel importante en este modelo, requieren menos administración.
- Red escalable: gracias a esta arquitectura, es posible quitar o agregar clientes sin afectar el funcionamiento de la red y sin la necesidad de realizar mayores modificaciones.

III.VI Creación de la base de datos

Para crear la base de datos que nos servirá para crear las tablas de lo que será nuestro repositorio de datos (data warehouse), para ello ingresamos al SQL Enterprise Manager como se muestra en la figura 3.5.

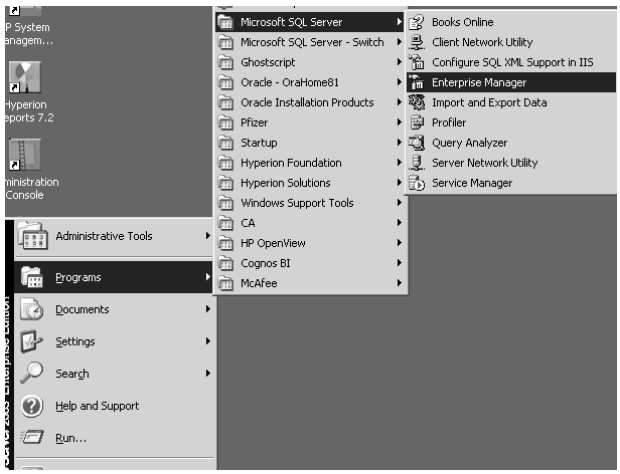


Figura 3.5: Muestra como ingresar al SQL enterprise manager

Posteriormente aparecerá una pantalla como en la figura 3.6 que nos pide ingresar ciertos datos como: Server y los datos de usuario para la conexión en caso de querer

realizar la conexión hacia otro servidor. Los cuales deberán ser llenados con los datos correspondientes al servidor, o bien, la computadora donde estará la Base de Datos que será utilizada.

En nuestro caso en server utilizaremos **localhost (servidor local)** que hace referencia a nuestra computadora (en caso de tener SQL instalado en otro servidor bastará con poner el nombre del servidor), así como la autenticación de Windows ya que en este momento no tenemos ningún usuario creado.

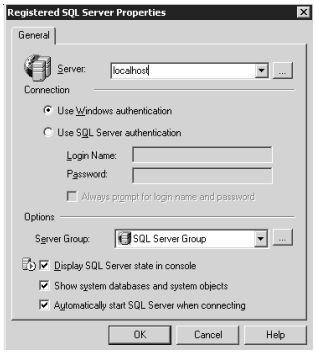


Figura 3.6: Opción para elegir el servidor en el que vamos a trabajar

Una vez que ingresamos en SQL enterprise manager para crear nuestra base de datos es necesario posicionarnos en la parte de Microsoft SQL servers – SQL server group – Server – Databases – New Database como se muestra en la figura 3.7.

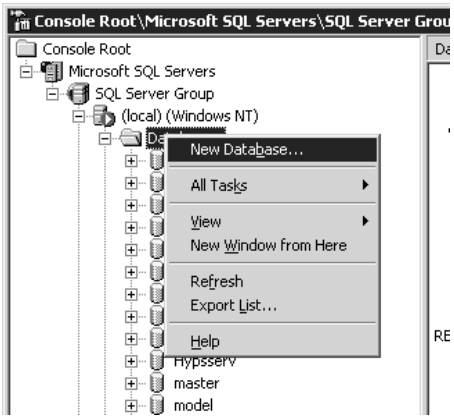


Figura 3.7: Método gráfico para crear una base de datos dentro del enterprise manager.

Es en esta parte donde vamos a ingresar los datos correspondientes para la nueva base de datos, que en nuestro caso llamaremos DWH_titulacion.

Adicionalmente es necesario crear un usuario que sea administrador de nuestra base datos el cual deber tener permisos de administrador sobre la base de datos ya que es necesario crear las tablas que nos ayudarán para la creación de nuestro repositorio de datos, en la figura 3.8 se muestra la ubicación donde podemos crear nuestro usuario con los permisos correspondientes de administrador: Microsoft SQL servers – SQL server group – Server – Security – Logins – New Login, en caso de requerir mas usuarios debemos seguir este mismo procedimiento para cada unos de los usuarios requeridos.

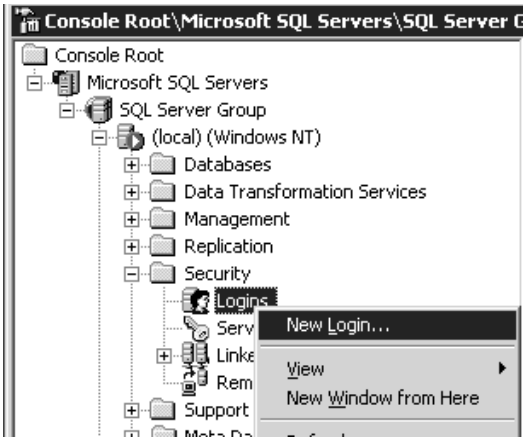


Figura 3.8: Muestra el área correspondiente para la creación de usuarios que tendrá nuestra base de datos.

III.VI Creación de tablas

El repositorio de datos deberá tener diferentes tablas que serán creadas mediante scripts definidos para cada unos de ellas. Dichas tablas serán cargadas de información por el ETL mediante una serie procesos que se ejecutarán para obtener las cifras del cierre diario el cual actualizará cada uno de los catálogos necesarios para la consolidación de los datos como se muestra en diagrama 3.9, así como la información de ventas, en este diagrama podemos observar la importancia de tener bien definidas

nuestras tablas al igual que nuestro ETL ya que son las partes esenciales de nuestro proyecto para garantizar la integridad de la información que estará alimentando nuestro cubo multidimensional y reportes necesarios.

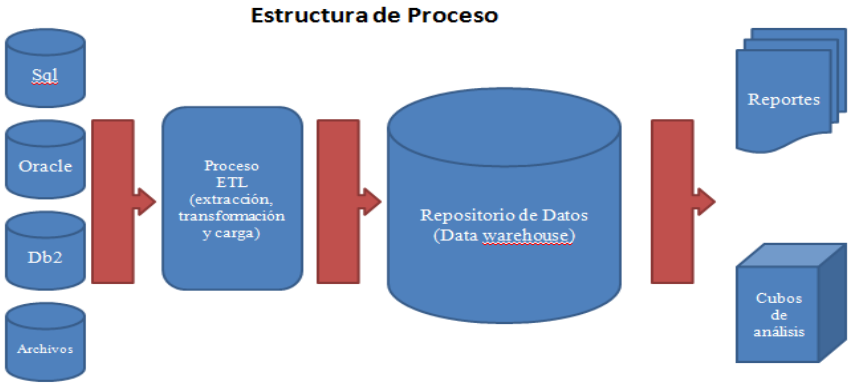


Diagrama 3.9: Estructura general para cargar la información en nuestro repositorio de datos para la creación del cubo multidimensional

Para ejecutar cada uno de los scripts utilizaremos **SQL query analyzer (analizador de consultas SQL)**, es una herramienta que nos permite interactuar con las diferentes bases de datos que tenemos ya sea para ejecutar consultas simples, o bien, procesos específicos que requieran mayor procesamiento. La forma de ingresar una vez estando dentro de nuestro Enterprise Manager es dando clic sobre Tools – SQL Query Analyzer como se puede observar en la figura 3.10 y posteriormente nos mostrará el área de trabajo para correr nuestros scripts.

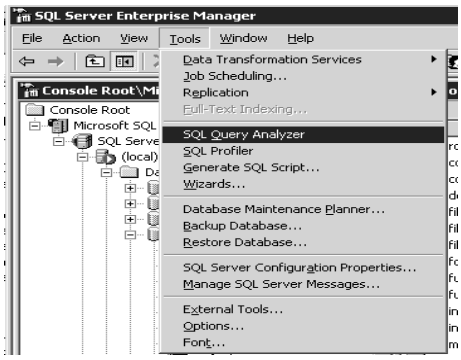


Figura 3.10: Forma para ingresar al SQL query analyzer

Una vez que estamos en SQL query analyzer debemos seleccionar nuestra base de datos y bastará con ejecutar cada uno de nuestros scripts como se muestra en la figura 3.11, o bien, se pueden crear procedimientos almacenados donde que contengan todos nuestros scripts en un solo procedimiento.

Los scripts que estaremos utilizando para la creación de nuestras tablas se explicarán a continuación.

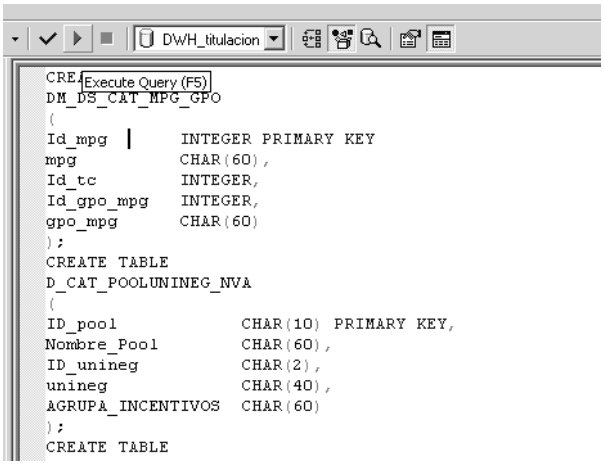


Figura 3.11: Muestra el área donde ejecutaremos cada uno de nuestros scripts.

- **Script 1: CAT_MPG**

En primera instancia vamos a crear la tabla CAT_MPG, la cual nos servirá para guardar los datos que identificarán a qué grupo pertenece cada unos de los productos que se están facturando mediante el siguiente script.

```
CREATE TABLE CAT_MPG
        (
        ID_MPG      CHAR(15) PRIMARY KEY,
        DESC_MPG        CHAR(60),
        ID_TC       INTEGER,
        ID_GRUPO_MPG   INTEGER,
        GRUPO_MPG     CHAR(60)
        );
```

Como podemos observar en el script anterior la tabla cuenta con cinco campos, donde quedará definido como llave primaria el campo ID_MPG que nos servirá para hacer nuestras relaciones con tablas que sean necesarias, en los otros campos se guardará la información de las descripciones correspondientes.

- **Script 2: CAT_POOL**

En la tabla que crearemos a continuación se guardan los datos que ayudan al área de ventas de ventas a dividir sus ventas por regiones lo cual es conocido como Pool, con esta tabla será mas fácil identificar las regiones de venta.

```
CREATE TABLE CAT_POOL
        (
        ID_POOL          CHAR(10) PRIMARY KEY,
        NOMBRE_POOL      CHAR(60),
        ID_UNINEG        CHAR(2),
        UNINEG           CHAR(40),
        AGR_INCENTIVOS  CHAR(60)
        );
```

Como podemos observar en el script la tabla contiene cinco campos donde nuestra llave primaria será el campo ID_POOL, así como la descripción requerida en los otros campos, esta tabla ayudará para agrupar cada unos de los mpgs en sus diferentes divisiones.

- **Script 3: CAT_PRODUCTOS**

Es necesario tener un catálogo que nos permita tener una relación de aquellos productos que actualmente se está vendiendo dentro de la compañía, así como permitir sean agregados nuevos productos en caso de ser requeridos lo cual se observará con más detalle al momento de implementar nuestro ETL.

```
CREATE TABLE CAT_PRODUCTOS
(
```

```
ID_PRODUCTO          INTEGER PRIMARY KEY,
DESCRIPCION       CHAR(60),
PRECIOA           FLOAT,
MPGSUN            INTEGER,
MPG_INTEGRA       CHAR(15) FOREIGN KEY REFERENCES CAT_MPG(ID_MPG),
PRECIOD           FLOAT,
DIVISION          INTEGER,
DIVISION_FINANZAS  INTEGER,
LOGILITY_FLAG     CHAR(15),
PRODUCT_TYPE      CHAR(15)
);
```

Nuestro script tiene diez campos que son requeridos para la generación del cubo multidimensional, como se puede observar queda como llave primaria el campo ID_PRODUCTO, el cual por ningún motivo puede duplicarse en nuestra tabla, además tiene otro aspecto importante ya que hace una referencia (relación) hacia la tabla CA_MPG tomando como base el campo ID_MPG de dicha tabla, por lo cual no podrá ser agregado un producto que no tenga relación con la tabla CAT_MPG.

- **Script 4: CAT_CLIENTES**

Nuestra tabla de clientes permitirá tener la información de todos aquellos clientes a los que la compañía vende sus productos, por tal motivo en caso de tener un cliente nuevo primero deberá estar dado de alta en este catálogo ya que de lo contrario no aparecerá dentro de nuestro cubo multidimensional el dato de facturación.

```
CREATE TABLE CAT_CLIENTES
(
ID_CLIENTE   CHAR(10) PRIMARY KEY,
AGRUPA       CHAR(15),
DESCRIPCION  CHAR(60),
SUCURSAL     CHAR(60),
POOL         CHAR(10) FOREIGN KEY REFERENCES CAT_POOL(ID_POOL)
);
```

Como se puede ver en el script la tabla cuenta con 6 campos, donde, el campo ID_CLIENTE será nuestra llave primaria y con ello no se permitirá duplicidad de información para un mismo cliente, además es necesario saber a que división pertenece cada cliente y por tal motivo se hace una referencia con la tabla CAT_POOL para poder obtener dicho dato y poder mostrar la agrupación correcta en el cubo.

- **Script 5: CAT_TIPOS_CAMBIO**

Para las organizaciones es muy importante tener la información de sus ventas en moneda local, pero adicionalmente se requiere tener dichos datos de venta en dólares, o bien, cualquier otra moneda que sea requerida para realizar las consultas sobre el cubo y con ello facilitar la creación de los reportes gerenciales, por tal motivo se creó la tabla CAT_TIPOS_CAMBIO que mediante el campo periodo nos permitirá identificar el tipo de cambio actual para representar la información de ventas en otra moneda.

```
CREATE TABLE
CAT_TIPOS_CAMBIO
(
PERIODO          INTEGER PRIMARY KEY,
TIPO_CAMBIO_REAL    FLOAT,
TIPO_CAMBIO_BUDGET  FLOAT
);
```

Al crear nuestra tabla anterior podemos ver que tiene 3 campos, donde, el campo PERIODO será nuestra llave primaria ya que un periodo no se puede repetir por ningún, además de que los tipos de cambio en el mercado global se manejan por periodos. En los otros dos campos tenemos los tipos de cambio correspondientes a las ventas, así como el tipo de cambio para el presupuesto en base las proyecciones establecidas.

- **Script 6: FACT_VENTAS**

Uno de los problemas más frecuentes al almacenar información dentro de la base de datos es que los usuarios pueden guardar cualquier información aunque no sea

necesaria, por tal motivo la importancia de nuestras tablas anteriores radica en la posibilidad de interactuar en nuestra tabla de hechos principal y con ello no permitir la carga de datos innecesarios, en esta tabla donde se estará guardando la información diaria de facturación tomando en cuenta que no se podrán cargar productos, clientes, periodos y pools que no estén previamente dados de alta en nuestros catálogos y esto nos permite garantizar la integridad de la información.

```
CREATE TABLE
FACT_VENTAS
(
FUENTE        INTEGER,
FACTURA       CHAR(30),
CLIENTE       CHAR(10) FOREIGN KEY REFERENCES CAT_CLIENTES(ID_CLIENTE),
SUCURSAL      CHAR(15),
FECHAAPLICA   INTEGER,
TIPODOCUMENTO CHAR(5),
PRODUCTO      INTEGER FOREIGN KEY REFERENCES CAT_PRODUCTOS (ID_PRODUCTO),
FECHAVENCE    INTEGER,
MONEDA        CHAR(15),
MOTIVO        INTEGER,
DIVISION      CHAR(15),
AGENTE        CHAR(3),
PLAZO         CHAR(4),
UNIDADES      FLOAT,
IMPORTE       FLOAT,
DPP           FLOAT,
MOTIVO_DOC    CHAR(15),
POOL          CHAR(10) FOREIGN KEY REFERENCES CAT_POOL(ID_POOL),
REGION        CHAR(30),
DISTRITO      CHAR(10),
ZONA          CHAR(10),
FUERZA        CHAR(15),
RUTA          CHAR(15),
PERIODO       INTEGER FOREIGN KEY REFERENCES CAT_TIPOS_CAMBIO(PERIODO),
IVA           FLOAT,
CUST_REF      CHAR(25),
RESERVA       FLOAT
```

```
);
```

Este script es más complejo ya que la tabla tendrá 27 campos, donde los campos de CLIENTE, PRODUCTO, POOL y PERIODO hacen referencia a otras tablas con lo cual podremos obtener las descripciones correspondientes de cada campo y con ello mostrar adecuados dentro de nuestro cubo multidimensional.

- **Script 7: FACT_PRORRATEA_VENTAS**

Una vez que cargamos la información de venta diaria, es necesario tener otra tabla que nos permita visualizar la información en dólares, así como las descripciones correspondientes que lo usuarios requieren para poder tener los datos necesarios dentro del cubo.

```
CREATE TABLE
FACT_PRORRATEA_VENTAS
(
FUENTE        INTEGER,
FACTURA       CHAR(30),
CLIENTE       CHAR(10) FOREIGN KEY REFERENCES CAT_CLIENTES(ID_CLIENTE),
SUCURSAL      CHAR(15),
FECHAAPLICA   INTEGER,
TIPODOCUMENTO CHAR(5),
PRODUCTO      INTEGER FOREIGN KEY REFERENCES CAT_PRODUCTOS(ID_PRODUCTO),
FECHAVENCE    INTEGER,
MONEDA        CHAR(15),
MOTIVO        INTEGER,
DIVISION      CHAR(15),
AGENTE        CHAR(3),
PLAZO         CHAR(4),
UNIDADES      FLOAT,
IMPORTE_USD   FLOAT,
IMPORTE       FLOAT,
DPP           FLOAT,
MOTIVO_DOC    CHAR(15),
POOL          CHAR(10) FOREIGN KEY REFERENCES CAT_POOL(ID_POOL),
REGION        CHAR(30),
```

```
DISTRITO       CHAR(10),
FUERZA         CHAR(15),
RUTA           CHAR(15),
PERIODO        INTEGER FOREIGN KEY REFERENCES CAT_TIPOS_CAMBIO(PERIODO),
PER_TRIM       CHAR(10),
TIPO_VENTA     CHAR(3),
ID_INST        CHAR(3),
DPP_USD        FLOAT
);
```

Uno de los principales campos creados en este script es donde la información se podrá mostrar en dólares, lo cual es de vital importancia para poder incluirlo dentro de las métricas del cubo.

Para le correcta creación de nuestras tablas es importante que los scripts sean ejecutados en este orden ya que de lo contrario podemos tener errores de consistencia con nuestras tablas, esto debido a las relaciones que existen entre las tablas y las llaves primarias correspondientes a cada una de ellas, una vez ejecutados los scripts ya tendremos todo listo en nuestra base de datos para poder cargar la información mediante nuestro ETL necesaria que servirá como base para nuestro cubo multidimensional.

Los procesos necesarios para la carga de información se explicarán a detalle en el siguiente capítulo tomando como referencia que cada unos de los procesos implementados deberá conducir con el llenado de nuestras tablas creadas en este capítulo garantizando la integridad de la información que se estará mostrando en el cubo multidimensional.

En nuestro siguiente capítulo de hará la creación del ETL donde se definirán las consultas que utilizaremos en cada uno de los procesos, se definirán los filtros correspondientes para cada una de las tablas, así como el tipo de datos que estaremos entregando en cada tabla.

CAPÍTULO IV IMPLEMENTACIÓN DEL ETL

IV.I Introducción al ETL

ETL (Extract, Transform and Load) es el proceso que permite a las organizaciones mover datos desde múltiples fuentes, reformatearlos, limpiarlos, y cargarlos en otra base de datos, data mart, o data warehouse para analizar, o en otro sistema operacional para apoyar un proceso de negocio, en este caso será utilizado para cargar las tablas que se utilizarán en nuestro cubo multidimensional.

- **Extraer (Extract)**

La primera parte del proceso ETL consiste en extraer los datos desde los sistemas de origen. La mayoría de los proyectos de almacenamiento de datos fusionan datos provenientes de diferentes sistemas de origen. Cada sistema separado puede usar una organización diferente de los datos o formatos distintos. Los formatos de las fuentes normalmente se encuentran en bases de datos relacionales o ficheros planos, pero pueden incluir bases de datos no relacionales u otras estructuras diferentes. La extracción convierte los datos a un formato preparado para iniciar el proceso de transformación.

Un requerimiento importante que se debe exigir a la tarea de extracción es que ésta cause un impacto mínimo en el sistema origen. Si los datos a extraer son muchos, el sistema de origen se podría ralentizar e incluso colapsar, provocando que éste no pueda utilizarse con normalidad para su uso cotidiano. Por esta razón, en sistemas grandes las operaciones de extracción suelen programarse en horarios o días donde este impacto sea nulo o mínimo.

- **Transformar (Transform)**

La fase de transformación aplica una serie de reglas de negocio o funciones sobre los datos extraídos para convertirlos en datos que serán cargados. Algunas fuentes de datos requerirán alguna pequeña manipulación de los datos. No obstante en otros

casos pueden ser necesarias aplicar algunas de las siguientes transformaciones: Seleccionar sólo ciertas columnas para su carga, traducir códigos, codificar valores libres, obtener nuevos valores calculados y unir datos de múltiples fuentes (por ejemplo, búsquedas, combinaciones, etc.).

- **Carga (Load)**

Los fabricantes de bases de datos han puesto considerable importancia en la eficiencia de las operaciones de carga. Las operaciones de carga incluyen el movimiento de los datos transformados y consolidados hacia la base de datos de apoyo para la toma de decisiones, la verificación de su consistencia y la construcción de cualquier índice necesario, existen pasos esenciales como los que mencionaremos a continuación:

1.-Movimiento de datos. Por lo general, los sistemas modernos proporcionan herramientas de carga en paralelo. En ocasiones formatearán previamente los datos para darles el formato físico interno requerido de destino antes de la carga real.

2.-Verificación de integridad. La mayor parte de la verificación de integridad de los datos a ser cargados puede ser realizada antes de la carga real, sin hacer referencia a los datos que ya están en la base de datos. Sin embargo, ciertas restricciones no pueden verificarse sin examinar la base de datos existente.

3.-Construcción de índices. La presencia de índices puede hacer significativamente lento el proceso de carga, debido a que la mayoría de los productos actualiza los índices conforme cada fila es insertada en la tabla subyacente. Por esta razón, en ocasiones es buena idea eliminar los índices antes de la carga y luego volverlos a crear. Sin embargo, este enfoque no vale la pena cuando la proporción de los nuevos datos (con respecto a los existentes) es pequeña, ya que el costo de crear un índice no se compensa con el tamaño de la tabla a indexar.

- **Procesamiento paralelo**

La aplicación de procesamiento paralelo ha permitido desarrollar una serie de métodos para mejorar el rendimiento general de los procesos ETL cuando se trata de grandes volúmenes de datos. Principalmente hay 3 tipos principales que se pueden implementar en las aplicaciones ETL:

- De datos: Consiste en dividir un único archivo secuencial en pequeños archivos de datos para proporcionar acceso paralelo.
- De segmentación: Permitir el funcionamiento simultáneo de varios componentes en el mismo flujo de datos.
- De componente: Consiste en el funcionamiento simultáneo de múltiples procesos en diferentes flujos de datos en el mismo puesto de trabajo.

Una característica de un sistema ETL es asegurar que los datos que se cargan sean relativamente consistentes, ya que las múltiples bases de datos de origen tienen diferentes ciclos de actualización (algunas pueden ser actualizadas cada pocos minutos, mientras que otras pueden tardar días o semanas), en un sistema de ETL será necesario que se puedan detener ciertos procesos hasta que todas las fuentes estén sincronizadas, del mismo modo, cuando un almacén de datos tiene que ser actualizado con los contenidos en un sistema de origen, es necesario establecer puntos de sincronización y de actualización.

IV.II Definición del catálogo

La herramienta que utilizaremos para la creación de nuestro ETL como se había definido anteriormente será DecisionStream ya que actualmente esta herramienta tiene la capacidad de procesar diferentes fuentes de información y cargaras a nuestro repositorio de datos, tomando como ventaja que en Pfizer actualmente es la herramienta estándar para este tipo de procesamiento. DecisionStream es una solución ETL que permite construir e implementar rápidamente un data warehouse o una serie

de Data Marts coordinados entre sí, en función de cada área funcional, en este sentido, garantiza que los datos procedentes de diversas fuentes o depósitos son organizados y estructurados según las dimensiones deseadas por el usuario final para la creación del cubo multidimensional.

Antes de iniciar con la creación de procesos es necesario definir una base de datos la cual servirá como repositorio para nuestro catalogo en DecisionStream, esta base de datos es independiente de nuestro Datawahouse que será donde guardaremos la información obtenida.
Nuestro catalogo estará en una base de datos de SQL, esto es de gran utilidad ya que de ser necesario migrar nuestro catalogo de servidor lo podríamos hacer realizando un backup de la base de datos y posteriormente restaurarla en otro servidor.

IV.II.I Configuración del repositorio de datos

Para crear la base de datos que nos servirá para nuestro catalogo realizaremos los mismos paso realizados en nuestro capítulo tres para la creación de nuestra base de datos, en este caso ingresamos los datos correspondientes para la nueva base de datos como se muestra en la figura 4.1 y en nuestro caso le ponemos DS_titulacion.

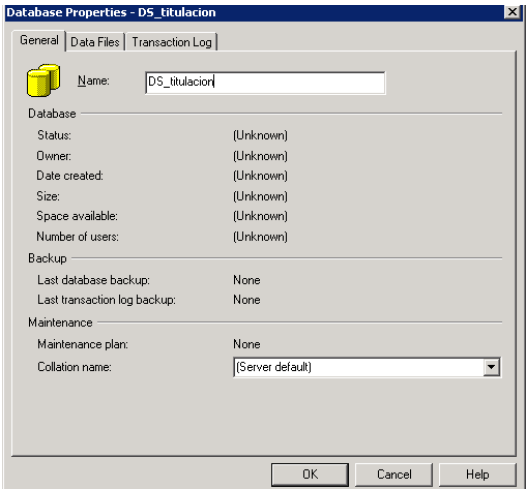


Figura 4.1: En esta parte se define el nombre de nuestra base de datos.

Una vez creada la base de datos es necesario crear un usuario que debe tener permisos de administrador, ya que servirá para poder hacer nuestras conexiones, así como la creación de los objetos necesarios para nuestro catálogo, tomando como referencia la creación de usuarios hecha en el capitulo tres, crearemos un usuario de nombre ds_titulacion, el cual deberá tener permisos sobre nuestra base de datos DS_titulación que nos ayudará a guardar todas las configuraciones y cambios aplicados en nuestro catálogo de DecisionStream.

IV.II.II Creación del catálogo

Antes de iniciar con DecisionStream es necesario crear una conexión a nuestra base de datos mediante un ODBC que deberá contener los datos necesarios para conectarnos.

ODBC (Open Database Connectivity) es un programa de interface de aplicaciones (API) para acceder a datos en sistemas manejadores de bases de datos tanto relacionales como no relacional, utilizando para ello SQL (lenguaje de consulta estructurado) con la finalidad de explotar los datos contenidos dentro de una base de datos.

Para crear una conexión ODBC en Windows es necesario entrar al control panel – administrative tolos – data sources como se muestra en la figura 4.2.

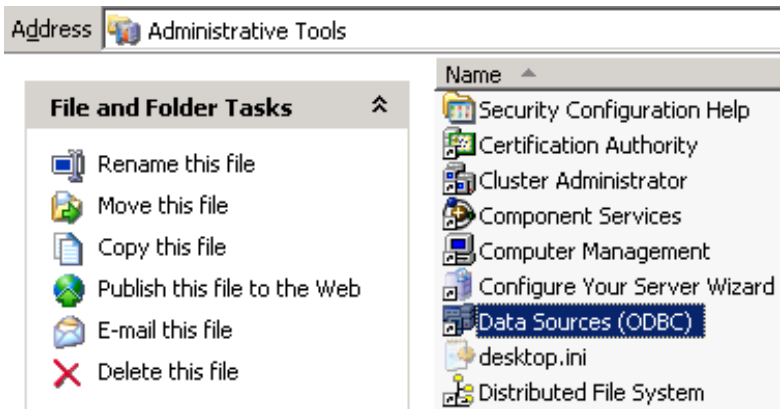


Figura 4.2: Muestra el área para crear un ODBC desde Windows.

Posteriormente es necesario configurar nuestro ODBC de acuerdo a los datos de conexión configurados para nuestro catálogo anteriormente, para ello vamos a seleccionar un system DSN utilizando el driver de SQL como se ven en la figura 4.3 ya que la base de datos dedicada para nuestro catálogo está desarrolla en SQL, en caso de tener otro tipo de conexión bastaría con definir la conexión correspondiente.

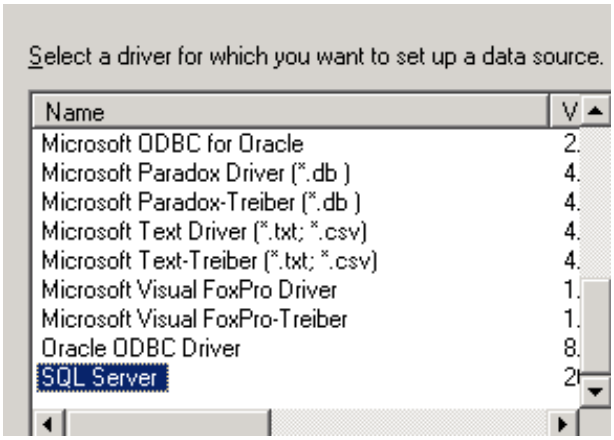


Figura 4.3: En esta parte indicamos el tipo de conexión que se desea

Una vez que ya tenemos el driver adecuado será necesario indicar el nombre de nuestro ODBC, así como nombre del servidor, base de datos y usuario que será utilizado para poder conectarse. La base de datos que utilizaremos, así como el usuario serán los que se crearon en el capítulo IV.II.I.

Una vez que ya definimos nuestro ODBC de la base de datos que utilizaremos para la creación de nuestro catálogo es necesario ingresar a DecisionStream (figura 4.4) y realizar la conexión correspondiente a nuestro repositorio. En primera instancia definimos el nombre del catálogo en el cual estaremos trabajando como se muestra en la figura 4.5 y lo llamaremos DS_TITULACION, es importante mencionar que cada que se requiera un cambio sobre el ETL será necesario ingresar con los datos que definiremos a continuación.

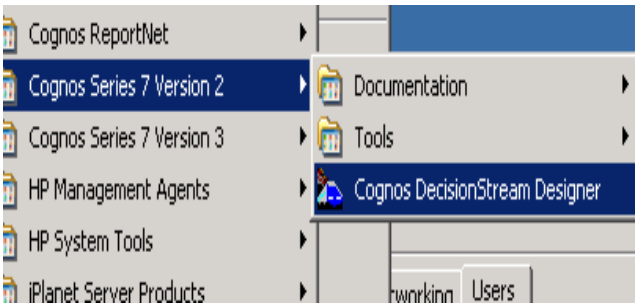


Figura 4.4: Icono principal de acceso a Decisionstream en Windows.

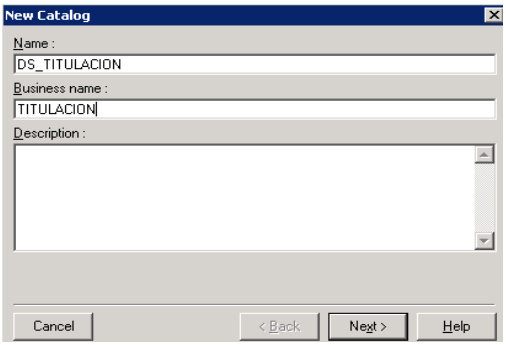


Figura 4.5: Pantalla para definir el nombre de nuestro catálogo.

Enseguida damos clic en Next para indicar los datos correspondientes de conexión para a base de datos que utilizaremos para nuestro catálogo. En este caso lo haremos mediante un ODBC el cual definimos previamente y tiene los datos necesarios de configuración, damos clic en test connection (figura 4.6) para validar los datos y asegurarnos de que nuestro catálogo está definido adecuadamente.

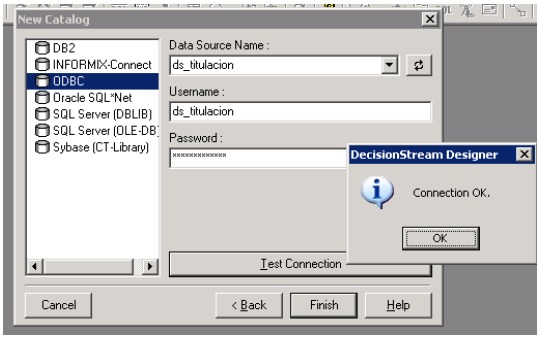


Figura 4.6: Pantalla de validación al definir el catálogo para nuestro ETL.

Una vez ingresando a nuestro catálogo la primer pantalla que veremos será como en la figura 4.7, es en este espacio donde podemos iniciar a trabajar para la creación de nuestros procesos que servirán el desarrollo de nuestro ETL, posteriormente se definirán cada uno de los procesos que serán necesarios para la carga de información que ayudará para alimentar de datos nuestro repositorio de datos del cubo multidimensional.

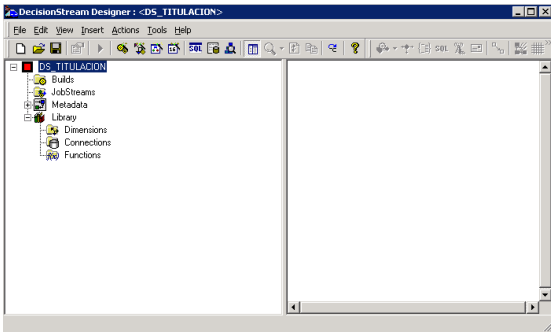


Figura 4.7: Área de trabajo principal para la creación de procesos en decisionstream.

IV.III Procesos de carga

La fuente de información principal para las ventas diarias será el sistema de facturación de Pfizer como se explicó en el capítulo 1, ya que en este sistema actualmente se carga todo el proceso de facturación, por otro lado tenemos archivos planos (txt, Excel, csv, etc.) que servirán para la carga de algunos catálogos relevantes como los productos.

En la secuencia definida para obtener los datos se utilizarán una serie de consultas predefinida las cuales servirán para cargar cada una de nuestras tablas creadas en el capítulo III para nuestro repositorio de datos y con ello poder mostrar datos en nuestro cubo multidimensional.

Antes de crear cada unos de los **builds (procesos)** es necesario crear las conexiones hacia nuestras diferentes fuentes que vamos a necesitar como se muestra en la figura 4.8, dichas conexiones serán mediante ODBC y se definen en la parte de conexiones.

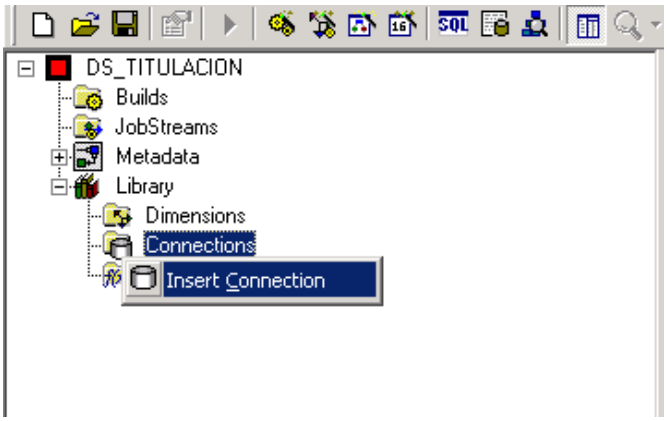


Figura 4.8: Área para definir las conexiones hacia nuestras diferentes fuentes de datos y entrega.

Proceso 1 CARGA_CATALOGO_MPG

Carga los archivos previamente definidos por los usuarios para consolidar la información de las unidades de negocio en Pfizer, lo cual ayudará a tener una correcta administración en cuanto a las corporaciones de la empresa.

Para la carga se utilizará un jobstream donde estarán integrados varios procesos que nos ayudarán con la definición de nuestra dimensión, su principal función de este proceso es cargar la información en el repositorio de datos para generar la dimensión de mpg, tomando en consideración que se debe omitir información duplicada lo cual nos servirá para actualizar los datos de productos que será la base para la generación del cubo multidimensional.

Iniciamos creando un jobstream (trabajo) como se muestra en la figura 4.9 al cual llamaremos genera_dimensiones y nos ayudará para cargar nuestra dimensión de mpgs y pools respectivamente, el jobstream deberá contener los procesos necesarios que se utilizarán para la carga de la dimensión de mpg.

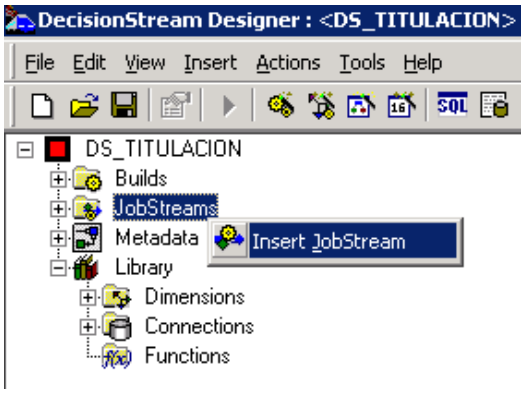


Figura 4.9: Muestra la creación de un jobstream para la ejecución de diversos procesos.

Dentro de nuestro job vamos a tener un build llamado CARGA_CATALOGO_MPG el cual contiene el proceso que nos servirá para cargar los datos de SUN mediante consultas SQL.

En este proceso se utilizarán dos herramientas importantes de DecisionStream, por un lado las jerarquías, así como las dimensiones de referencia, esto para asignar niveles a las fuentes de información que estamos procesando, para este proceso debemos crear una dimensión de referencia (reference dimension), la cual llamaremos DS_CATALOGOS_INTEGRA como se muestra en figura 4.10:

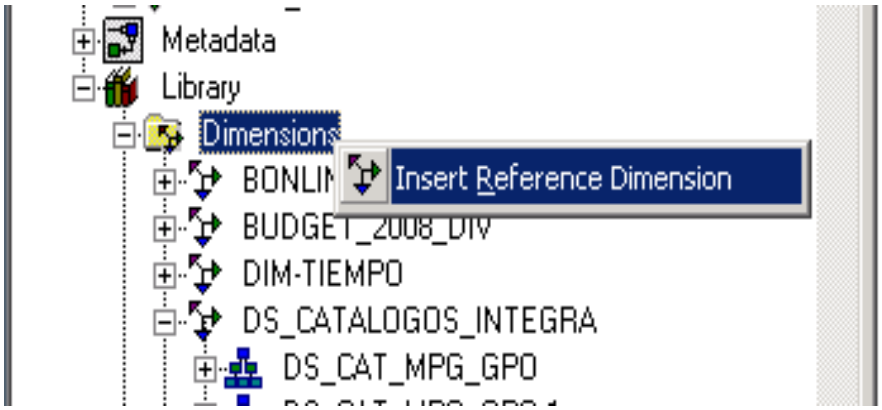


Figura 4.10: Espacio para crear las dimensiones de referencia.

Una vez que tengamos nuestra Dimensión de Referencia debemos agregar una jerarquía como se muestra en la figura 4.11 que nos ayudará para agrupar la información en los diferentes niveles requeridos a la cual llamaremos DS_CAT_MPG_GPO

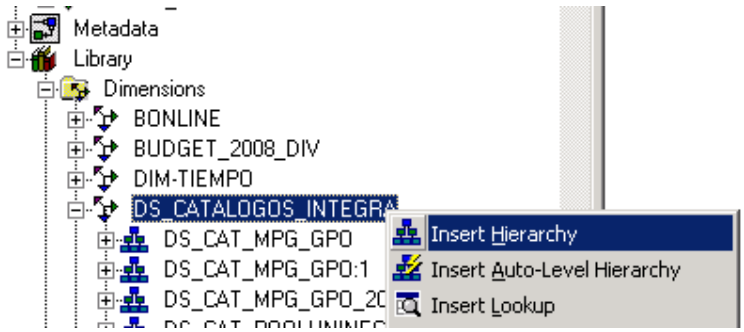


Figura 4.11: Creación de una jerarquía para agrupar diferentes niveles de datos.

En la jerarquía debemos definir los diferente niveles que vamos a utilizar (figura 4.12) en este caso utilizaremos dos niveles que se llamarán_ id_gpo_mpg y id_mpg respectivamente.

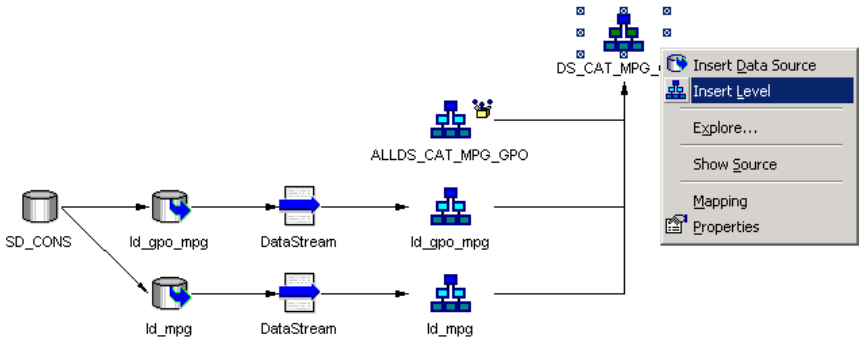


Figura 4.12: Definición de niveles dentro de una jerarquía.

En el nivel id_mpg vamos a tener un **data source (fuente de datos)** que utilizaremos para poder cargar los datos de los catálogos creados por el área, el cuál que contendrá los datos de los Mpgs que agrupan a una serie de productos para llevar un mejor control dentro de la organización, para esto se utilizará una consulta SQL como se

muestra en la figura 4.13, donde, un **SELECT (sentencia SQL de consulta)** lo que utilizamos para consultar información específica sobre una o varias tablas según sean las necesidades de los datos.

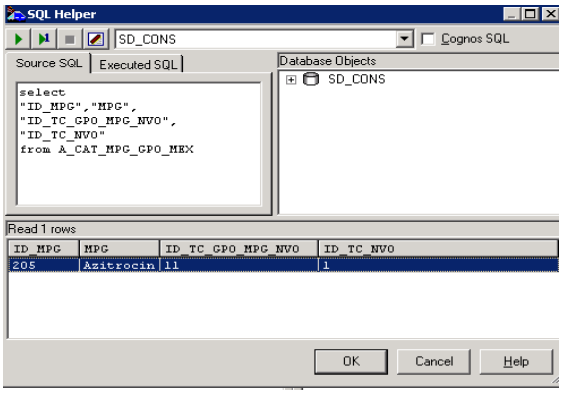


Figura 4.13: Estructura de un data source para la carga de datos.

La consulta que utilizaremos para cargar los datos de este nivel estará definida como se muestra a continuación, la cual importará la información de una tabla que contiene los datos solicitados dentro de nuestra conexión SD_CONS previamente definida (archivo de texto).

```
select ID_MPG,MPG,ID_TC_GPO_MPG_NVO,ID_TC_NVO
from
A_CAT_MPG_GPO_MEX
```

Adicionalmente agregamos otro nivel el cual llamaremos id_gpo_mpg que contendrá distintas definiciones que nos ayudarán para agrupar los Mpgs en uno solo tomando como referencia el campo id_tc_gpo_mpg_nvo el cuál servirá para poder crear la relación con nuestro nivel anterior y con ello definir de forma adecuada la dimensión de Mpgs.

En la figura 4.14 se muestra la sentencia **Select Distict (campos) from tabla** nos ayuda para obtener aquellos registros que tienen diferente información en su contenido de cada campo, omitiendo así información duplicada.

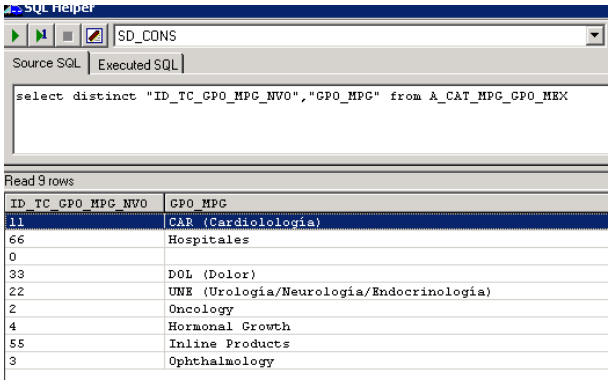


Figura 4.14: Muestra la utilización de la sentencia distinct en una consulta.

Una vez definidos los dos niveles que vamos a utilizar debemos hacer un **mapping (relación)** de los datos obtenidos con aquella información que deseamos entregar en nuestra tabla final, para ello damos doble clic sobre la el datastream y posteriormente mapeamos todos nuestros campos como se puede observar en la figura 4.15.

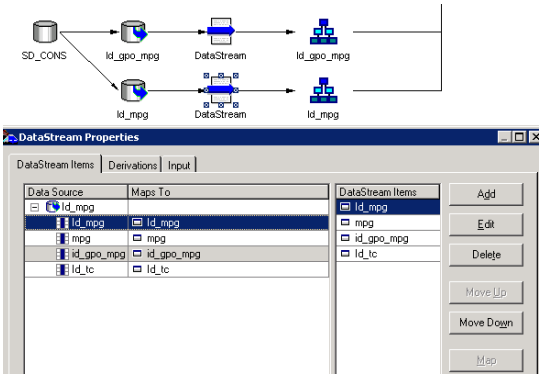


Figura 4.15: Mapeo de campos solicitados

Adicionalmente debemos indicar en cada uno de nuestros niveles el valor que tendrá cada uno de los campos mapeados anteriormente como observamos en la figura 4.16, en este caso en particular tenemos que nuestro id_mpg será nuestra llave y por tal motivo es un valor que debe ser único, el campo que indicamos como Caption será el mpg el cual nos indica que es el valor que estaremos entregando y finalmente el campo

id_gpo_mpg queda definido como Parent lo cual nos indica que deberá tener relación con nuestro nivel superior.

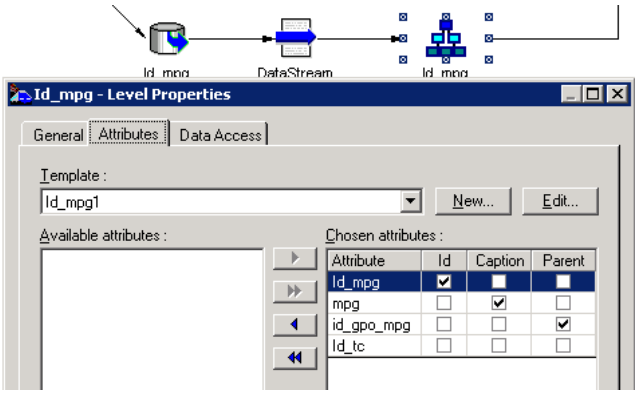


Figura 4.16: Pantalla para indicar los tipos de dato correspondientes a cada campo.

En nuestro nivel superior vamos a definir la relación que tendrá con nuestro nivel inferior creado anteriormente como se indica en la figura 4.17, en este caso el campo que definimos como llave es el id_gpo_mpg ya que tiene una relación directa con el campo que definimos como parent anteriormente, además del caption que será el valor a mostrar en nuestra tabla al momento de entregar los datos.

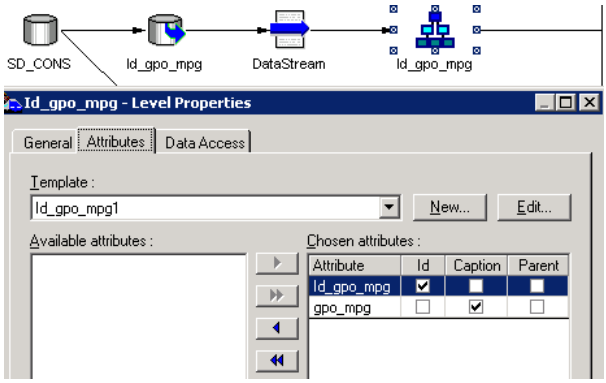


Figura 4.17: Definición de tipo de dato y relación con otros niveles.

Una vez terminado el build se verá como se muestra en la figura 4.18, el cual bastará con ejecutarlo para poder cargar de datos en la tabla que deseamos.

Figura 4.18: Estructura de un build con jerarquías.

La forma de realizar consultas sobre nuestras tablas es mediante sentencias SQL, en específico la sentencia Select como se muestra en la figura 4.19, mostrando los datos de una tabla.

Source SQL | Executed SQL

```
select * from DM_DS_CAT_MPG_GPO
```

Read 143 rows

Id_mpg	mpg	Id_tc	Id_gpo_mpg	gpo_mpg
107	Terramicina	5	55	Inline Products
122	Cefobid	5	55	Inline Products
123	Unasyn Oral	5	55	Inline Products
125	Unasyn IM/IV	5	55	Inline Products
134	Vibramicina	5	55	Inline Products
136	Accupril	5	55	Inline Products
137	Accuretic	5	55	Inline Products
138	CHLORO/PROLOID	5	55	Inline Products
145	Dilantin	5	55	Inline Products
146	Dilzem	5	55	Inline Products
151	FemHRT	5	55	Inline Products
159	Lopid	5	55	Inline Products
161	Neurontin	3	33	DOL (Dolor)
164	Ponstan	5	55	Inline Products

Figura 4.19: Estructura de un select.

Una de las ventajas al utilizar SQL son las diferentes clausulas o condiciones que podemos ejecutar sobre las consultas, por ejemplo, en caso de requerir información específica de un grupo de mpg implementamos la clausula where, donde indicaremos el filtro de lo que deseamos obtener, la clausula where tiene como finalidad devolver el resultado de un conjunto de registros que necesitamos para ser analizados y su sentencia es la siguiente:

```
Select * from tabla
where campo = valor
```

El operando = se puede cambiar por cualquier operando que sea valido dentro de SQL como por ejemplo: <>, >, < y es necesario considerar que en caso de ser un campo de tipo carácter es necesario utilizar comillas simples para hacer la igualación.

```
Select * from tabla
where campo = 'valor caracter'
```

Proceso 2: CARGA_CATALOGO_POOL

Carga los archivos previamente definidos por los usuarios para consolidar la información de las unidades de negocio en Pfizer, lo cual ayudará a tener una correcta administración en cuanto a las corporaciones de la empresa.

Utilizando el mismo procedimiento aplicado en el proceso anterior, en este proceso se va a cargar la tabla que nos servirá para consolidar nuestra dimensión de divisiones por pool, vamos a tener un dimension build como se muestra en la figura 4.20 tomando como fuentes diferentes tablas para cargar los datos correspondientes a la dimensión de divisiones.

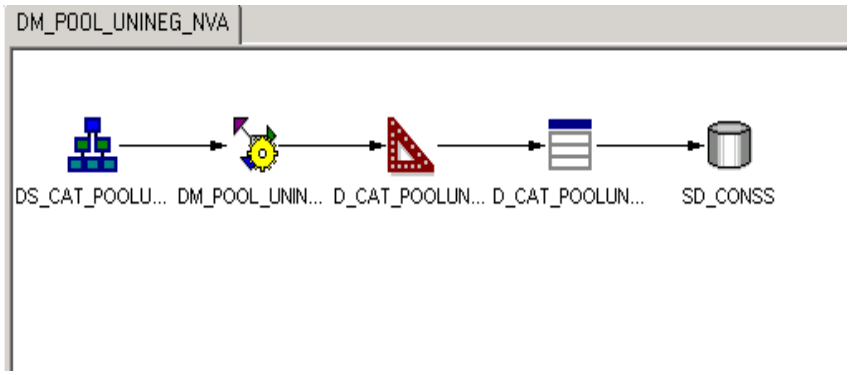


Figura 4.20: Proceso para la carga de la dimensión de divisiones.

Las consultas que utilizaremos para este proceso tomarán la información de dos archivos Excel definidos por los usuarios, los cuales bastará con ser modificados para actualizar nuestra dimensión de divisiones.

Archivo 1:

```
SELECT ID_pool, Nombre_Pool, id_unineg_NVA, Agrupa_Incentivos
FROM
Cat_Pool_2007` where `ID_pool` <> 'ND'
```

Archivo 2:

```
SELECT ID_unineg_nvo,
unineg_nvo
FROM
Cat_unineg_nvo where ID_unineg_nvo <> 'ND'
```

Este proceso contiene una jerarquía que nos sirve para definir de nuestras diferentes fuentes de datos y que nivel les corresponde a cada una de ellas respectivamente como se había observado en el proceso anterior. Por último nos permitirá llenar la tabla que servirá para nuestra dimensión de Divisiones en nuestro cubo multidimensional, la tabla como se muestra en la figura 4.21 contendrá la siguiente estructura al término de nuestro proceso.

Source SQL | Executed SQL

```
select * from D_CAT_POOLUNINEG_NVA
```

Read 133 rows

ID_pool	Nombre_Pool	ID_unineg	unineg	AGRUPA_INCENTIVOS
-	- ERROR	-	Error	ND
00	00_CUENTA...	I	INST...	DIV POR DIST
01I	01I_Migue...	-	Error	01I
02	02_TIJUAN...	0	OESTE	BAJA CALIFORNIA
02G	02G_TIJUA...	0	OESTE	BAJA CALIFORNIA
02I	02I_Isidr...	-	Error	02I
02N	02N_TIJUA...	0	OESTE	BAJA CALIFORNIA
03	03_TIJUAN...	0	OESTE	BAJA CALIFORNIA
03I	03I_Alfon...	-	Error	03I
04	04_HERMOS...	0	OESTE	SONORA

Figura 4.21: Estructura final de la tabla para la dimensión de divisiones.

Proceso 3: CARGA_CAT_PRODUCTOS

Se obtiene toda la información de los productos vendidos por Pfizer, así como su descripción y sus agrupaciones en diferentes niveles que ayudará a la creación del catalogo de productos considerando nuevos productos que se agregarán de forma automatizada.

En este proceso se cargarán los productos que están dados de alta en el sistema de facturación utilizando una consulta SQL que permitirá cargar los datos correspondientes omitiendo los registros duplicados, una vez extraídos estos datos se utilizará un lookup, el cual nos ayudará para leer nuestro catalogo de mpgs cargado anteriormente, esto ayudará para alimentar la tabla que nos servirá como base para nuestra dimensión de productos y cuando un nuevo código es agregado en el área de facturación este será cargado automáticamente dentro de la tabla ya que tiene le peculiaridad de tener el atributo UDATE/INSERT que explicaremos posteriormente.
En la figura 4.22 se muestra como debe quedar nuestro proceso final, del cual se describirá posteriormente cada una de sus etapas que lo conforman para un mejor entendimiento.

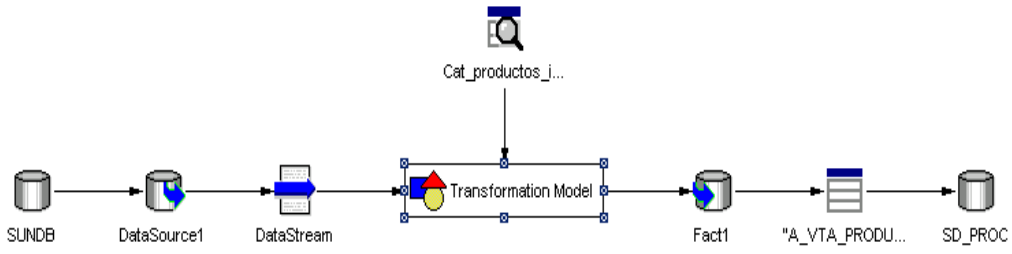


Figura 4.22: Muestra el proceso final para la carga de nuestra dimensión de productos.

En nuestro datasource1 visto en la figura anterior vamos a extraer la información de los productos dados de alta en sistema de facturación SUN, para ello utilizaremos la sentencia **select**, además de un **from** para seleccionar las tablas y un **distinct** que nos ayudará para evitar tener registros duplicados ya que nuestro código de producto es

único y deberá ser así durante todo el proceso, por otro lado utilizamos la clausula **where** para relacionar nuestras diferentes tablas que tenemos y con ello poder obtener datos como descripción y precio del producto.

La consulta que utilizaremos para la carga nuestro catalogo estará definida como se muestra a continuación, donde podemos observar la utilización de las sentencias explicadas anteriormente.

```
select distinct
b.ITEM_CODE Producto, b.DESCRIPTN Descripcion, b.BASE_PRICE1 PrecioA,  b.ANAL_I2
MPGSUN,b.BASE_PRICE4 PrecioD, b.ANAL_I1 division, b.ANAL_I5 ISPC, b.UNIT_STOCK
Measure_units, b.COMMENTS Logility_flag

from
SOMFHDRMPL h, SOMFDETMPL d, SSRFADB A, SSRFITM b

where
A.ADB_CODE = h.CUST_CODE and d.REC_TYPE = 'D' and h.TRANS_REF=d.INV_NO and
h.REC_TYPE='I' and (h.ANAL_M5 = '' or h.ANAL_M5 = 'USD') and h.ANAL_M7 <= '599' and h.ANAL_M3 =
'262' and b.SUN_DB = A.SUN_DB and d.ITEM_CODE = b.ITEM_CODE and d.ITEM_CODE >= '00001'
```

En esta consulta podemos observar los beneficios que podemos tener al utilizar la clausula where combinada con otros atributos para mostrar datos de diferentes tablas tomando cierto criterios.

Posteriormente realizamos el **mapping (relación)** de nuestras columnas que deseamos entregar, esto lo hacemos uno a uno (figura 4.23) dando clic sobre al campo que deseamos hacer relación ya que vamos a utilizarlo para hacer la relación con nuestro lookup que definiremos posteriormente, esto para poder obtener la relación producto – mpg en cada uno de ellos y poder tener nuestra tabla completa para que nuestra dimensión de productos nos muestre los datos de cada uno de los productos, así como su agrupador correspondiente (mpg).

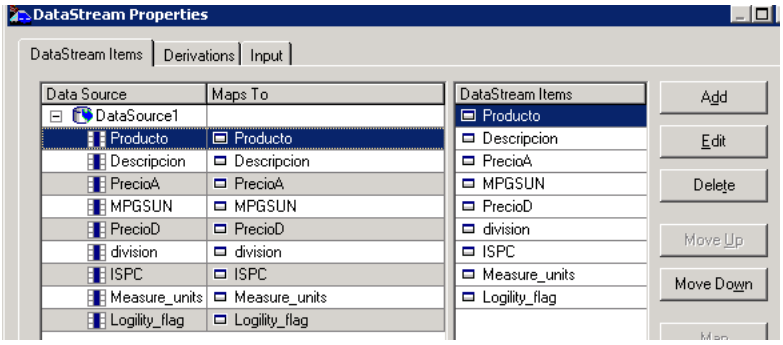


Figura 4.23: Mapeo de atributos

En el tranformation model (transformación del modelo) definimos la columna producto como llave lo cual nos servirá para hacer una relación con nuestro catalogo de mpgs mediante un lookup que nos estará entregando los datos de mpgs para cada producto como se ve en la figura 4.24, en este caso tiene el efecto como si aplicará un JOIN (relación) dentro de una consulta SQL, con la particularidad que podemos aplicar cálculos sobre los datos obtenidos lo cual nos ayuda a manipular los datos de mejor manera y poder transformar la información según las necesidades del negocio.

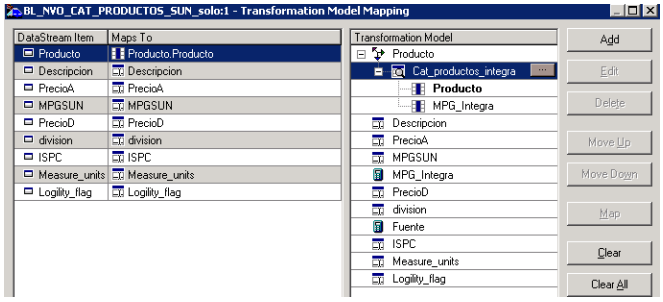


Figura 4.24: Forma para obtener los datos de un lookup mediante atributos.

Una vez que ya tenemos nuestro lookup definido en el tranformation model es necesario entregar los datos obtenidos, para ello es necesario agregar un derivation (campo calculado) por cada una de las columnas que deseamos mostrar como se muestra en la figura 4.25.

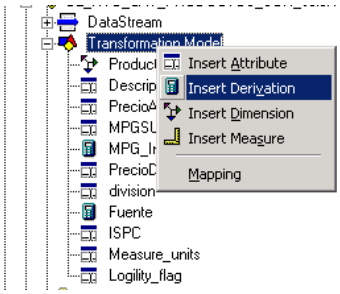


Figura 4.25: Área para entregar campos calculados.

Una de las propiedades del derivation es que dentro de la pestaña de reference tributes tenemos definida una condición (figura 4.26) para asignarle un mpg a cada producto, la cual nos indica que utilizará el valor de nuestro lookup siempre y cuando los datos provenientes de nuestro datasource no contengan la información necesaria, es importante mencionar que podemos utilizar diferentes tipos de condiciones, cálculos y funciones en caso de ser requerido dependiendo de las necesidades del negocio.

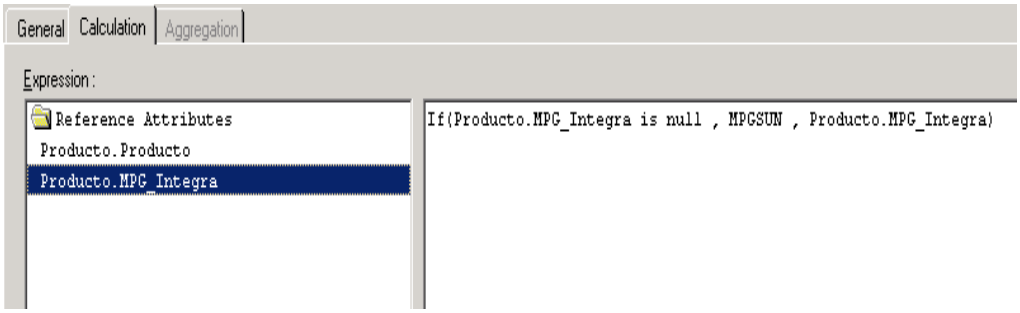


Figura 4.26: Agregar condiciones a un campo calculado

Una vez definidas las columnas deseamos, vamos a entregar los datos en la tabla CAT_PRODUCTOS creada en el capítulo III, donde id_producto quedará definido como llave (Primary Key) como se muestra en la figura 4.27, lo cual indica que no puede haber datos duplicado en esa columna y en caso de existir nos manda un error de integridad con nuestra tabla.

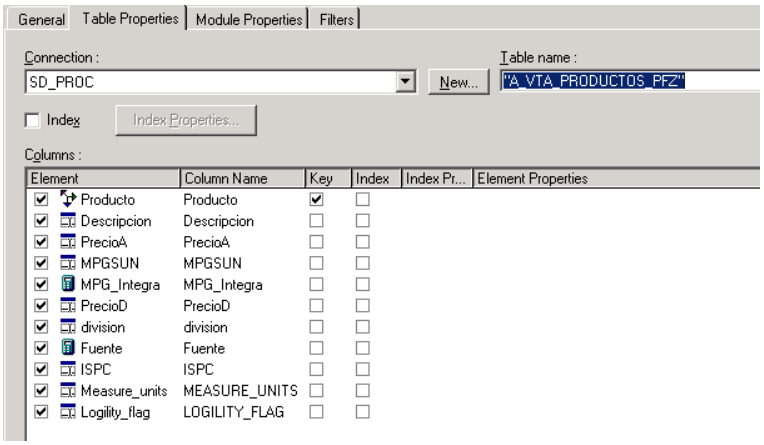


Figura 4.27: Muestra cómo podemos definir un campo como llave dentro del proceso

Una vez que ya tenemos definido en que tabla (table delivery) vamos a entregar la información, dentro de la pestaña module properties está definido que sea de UPDATE/INSERT (figura 4.28) que explicaremos posteriormente, ya que con esto vamos a permitir que al existir nuevos códigos dentro del sistema de facturación SUN en automático serán agregados en nuestra tabla y con esto se garantizará la integridad de los datos en la dimensión de productos ya que también permite actualizar las descripciones en caso de ser necesario.

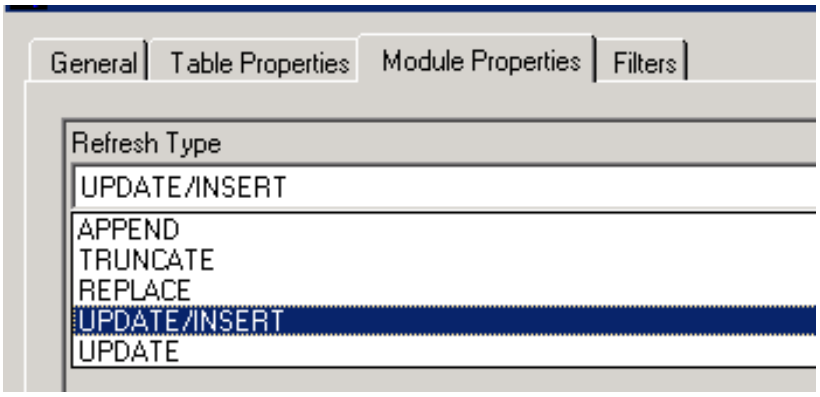


Figura 4.28: Muestra la forma de como estaremos entregando la información en una tabla

Otras opciones que tenemos además de UPDATE/INSERT para este tipo de entrega de datos son: APPEND, TRUNCATE, REPLACE, UPDATE, las cuales se utilizan dependiendo de las necesidades que el proceso requiera

- **APPEND**: Esta función sirve para agregar toda la información proveniente de nuestro datasource, por lo que nuestra tabla se irá incrementando en cada una de las actualizaciones.

- **TRUNCATE**: Esta función es utilizada principalmente cuando deseamos borrar por completo el contenido de una tabla y posteriormente agregar nuevos datos.

- **REPLACE**: Reemplaza los datos existentes en la tabla que tenga relación con nuestro datasource previamente definido.

- **UPDATE**: Actualiza la información de los campos que hagan relación definiendo para ello previamente un campo como llave.

Otra opción que tenemos en nuestro Table delivery son los filtros (figura 4.29), lo cual nos permite como su nombre lo indica filtrar datos mediante diferentes condiciones que pueden ser definidas según las necesidades de los usuarios.

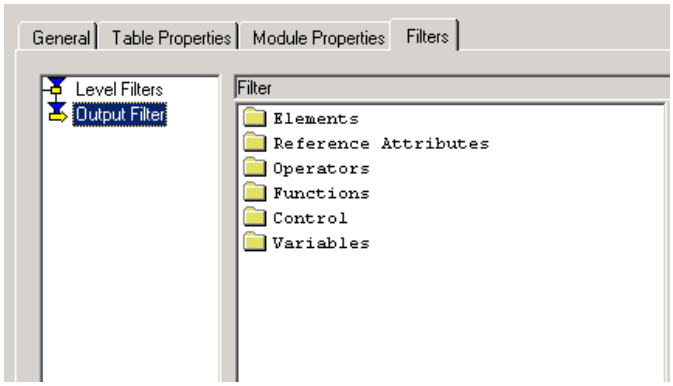


Figura 4.29: Área para aplicar filtros al entregar datos en una tabla.

Proceso 4: CAT_CLIENTES

Tomando como base la información histórica de ventas, así como los clientes actuales carga los datos de los diferentes clientes de Pfizer, con lo cual se elabora el catalogo de Clientes para la elaboración del cubo. Esto nos ayudará a tener una correcta administración de los clientes que por alguna razón han dejado de facturar, o bien, poder ver su tendencia a la alta según sea el caso, lo cual permitirá poder analizar a detalle la información por cliente y poder definir estrategias de negocio si son requeridas

Para cargar nuestro catálogo de clientes es necesario realizar previamente una homologación de los clientes ya que el área de ventas maneja diferentes descripciones que no son igual que las capturadas dentro del sistema de facturación, para ello se tienen diferentes catálogos que utilizaremos mediante lookups que nos ayudarán con dicha homologación como: HOM_CLIENTES, HOM_AGRUPADOR, HOM_SUCURSAL como se muestra en la figura 4.30, los cuales son administrados por el área de ventas, con ello permitiremos que la información mostrada dentro de nuestro cubo multidimensional sea mostrada como el usuario lo requiere.

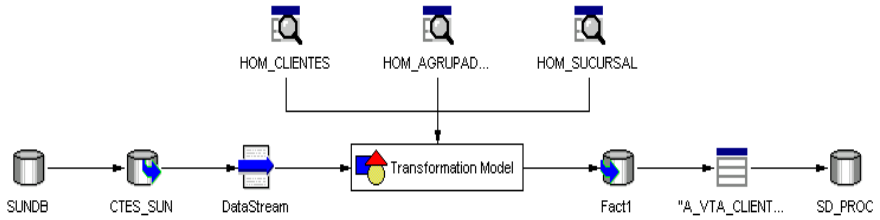


Figura 4.30: Muestra el proceso de clientes con sus diferentes catálogos.

En el datasource vamos cargar lo información proveniente del sistema de facturación para nuestra dimensión de clientes mediante una consulta, la cual tiene la peculiaridad de utilizar la clausula **case**, la cual nos permite tener mediante una serie de condiciones un valor predeterminado para un campo de nuestra tabla.

Además se implementó la función **substring** que es una sub-cadena en SQL que se utiliza para tomar una parte de los datos almacenados, esta función puede ser invocada con diferentes nombres según las diferentes bases de datos que se estén utilizando, una de sus variantes puede ser utilizada como **substr**.

La consulta que utilizaremos para cargar la información de los diferentes clientes facturados será la siguiente:

```
select distinct d.ACCNT_CODE Id_Cliente, case when case when substring(e.ADDRESS_2,1,3) in
'VER','CHI','TUX','MEX','ACA','SUC','TIJ','GUA','JUA','LEO','HOR','MET','VAL','MER','GTO','MON','MUN','HE
R','VIL') then case when e.ADDRESS_2 = 'VALLAHERMOSA' then 'VILLAHERMOSA' else e.ADDRESS_2
end else rtrim(isnull(c.LOOKUP,(isnull(e.ADDRESS_2,e.ADDRESS_6)))) end ='GRUPO CAS' then
substring(e.ADDRESS_6,7,34)) else case when substring(e.ADDRESS_2,1,3) in
'TOR','CED','PUE','CUL','TOL','MOR','VER','CHI','TUX','MEX','ACA','SUC','TIJ','GUA','JUA','LEO','HOR','MET
','VAL','MER','GTO','MON','MUN','HER','VIL') then case when e.ADDRESS_2 = 'VALLAHERMOSA' then
'VILLAHERMOSA' else e.ADDRESS_2 end else
rtrim(isnull(c.LOOKUP,(isnull(e.ADDRESS_2,e.ADDRESS_6)))) end end Sucursal,
(rtrim(d.ANAL_C2)) Agrupa , c.ACCNT_NAME Descripción, ANAL_C5 POOL,
(substring(e.ADDRESS_6,7,34)) ESTADO
from SSRFADB d,SSRFACC c ,SSRFADD e
where
d.SUN_DB='MPL' and d.SUN_DB=c.SUN_DB and d.SUN_DB=e.SUN_DB and
d.ACCNT_CODE=c.ACCNT_CODE and d.ADB_CODE=e.ADD_CODE and substring(ADB_CODE,10,1) =
'1'
```

En el script anterior podemos observar un nuevo parámetro utilizado dentro de nuestra consulta, este parámetro es el case que se explicará a continuación.

- CASE: Cuando tenemos una columna en una tabla de SQL que puede contener diferentes valores y según el valor de cada tupla nosotros queremos mostrar un texto, número, etc. al momento de hacer el SELECT disponemos de la sentencia CASE de SQL que tiene la sintaxis:

  SELECT CASE columna WHEN valor1 THEN ‘es valor 1′ WHEN valor2 THEN ‘es valor 2′ …. END From Tabla

Donde valor1, valor2, etc. son los valores que se encuentran almacenados en la base de datos y los textos “es valor 1″, “es valor 2″ son los textos que se mostrarán en el resultado del SELECT para cada tupla que cumpla con valor1 y valor2 respectivamente, valor1 y valor2 y los valores a mostrar pueden ser de cualquier tipo.

Proceso 5: CAT_TIPOS_CAMBIO

Cada cierre de periodo es necesario actualizar los tipos de cambio para lo cual nos basamos en el sistema de facturación para obtener el tipo de cambio actual y con ello poder realizar los cálculos en dólares, así como la obtención de tipos de cambio para presupuesto tomando como base archivos planos.

Durante este proceso cargarán los tipos de cambio tomando como base el sistema SUN, donde existen los tipos de cambio por periodo, se implementó un lookup (figura 4.31) debido a que también es necesario que nuestra tabla contenga los tipos de cambio para presupuesto los cuales son definidos por el área de ventas en base a sus proyecciones estimadas para el año.

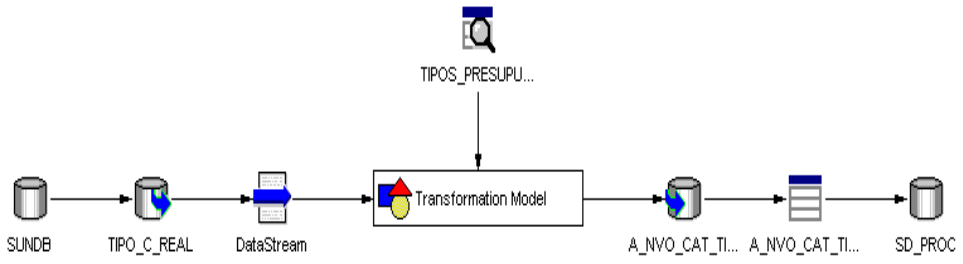


Figura 4.31: Proceso para la carga de los tipos de cambio.

Mediante la siguiente consulta se obtienen los tipos de cambio actuales del sistema de facturación tomando como base el periodo máximo en el cual se está facturando, lo cual ayudará para mostrar las ventas en dólares dentro del cubo multidimensional.

```
select PERIOD PERIODO, RATE TIPO_CAMBIO_REAL
from SSRFCNV
where SUN_DB = 'MPL' and CODE = 'USD' and PERIOD <> 0 and PERIOD = {$Per_max}
```

Donde {$Per_max} es una variable que posteriormente definiremos dentro de nuestro proceso principal para la carga de los datos solicitados y la clausula where nos sirve para extraer solo aquella información que deseamos como se observa en la figura 4.32.

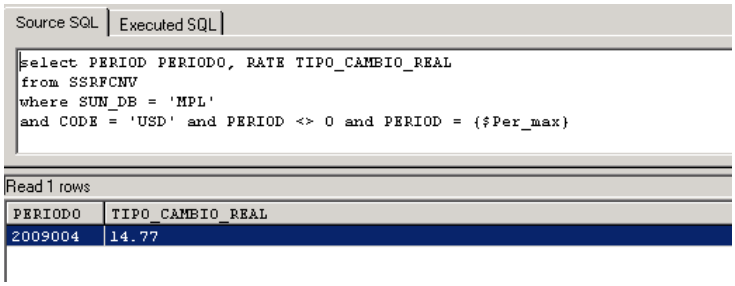


Figura 4.32: Muestra el contenido de la tabla para la carga del periodo.

Posteriormente se hace relación mediante un lookup (como se observó en el proceso 1) tomando como campo llave el periodo ya que los datos que se van a extraer para los tipos de cambio en presupuesto provienen de un archivo en Excel (figura 4.33), el cual es cargado por los usuarios de ventas.

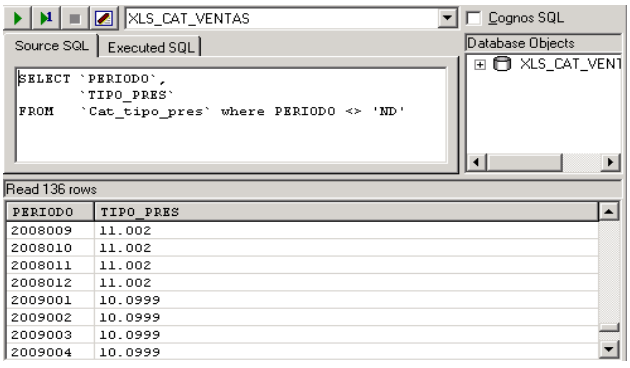


Figura 4.33: Tipos de cambio por periodo tomado de un archivo plano.

Proceso 6: FACT_VENTAS

Servirá para actualizar las ventas diarias mediante este proceso se cargarán los datos relevantes para el área de ventas como: ventas en pesos, unidades y dólares lo cual se unirá con nuestros diferentes catálogos para poder ver las descripciones correspondientes en cada una de las dimensiones.

Es en este proceso donde diariamente vamos extraer las ventas del sistema de facturación SUN mediante consultas SQL que permitirán consolidar correctamente los datos tomando como referencia las reglas de negocio definidas, además tiene dos lookups (figura 4.34) que nos permitirá añadir campos a la información obtenida, esto nos permitirá tener la información actualizada al final de cada cierre de ventas, pero en caso de ser necesario tener la información actualizada en el transcurso del día los usuarios claves pueden solicitar al administrador que se actualice la información en el cubo para lo cual será necesario correr todo el proceso de cierre.

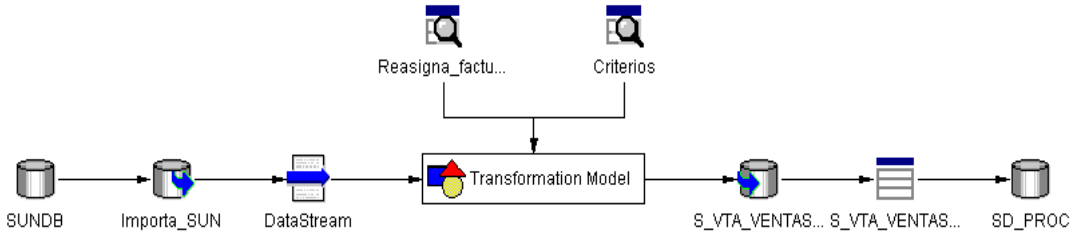


Figura 4.34: Proceso para la carga de venta diaria desde el sistema de facturación.

En nuestro datasource utilizamos la sentencia select que nos permiten obtener los datos que necesitamos tomando como base nuestro sistema da facturación, dicha consulta contiene clausulas where (figura 4.35) para cumplir con ciertos criterios que deben ser considerados para obtener aquella información que es relevante para el área de ventas, en esta consulta se están utilizando cuatro diferentes tablas definidas en from, esto es de mucha utilidad cuando ya que en la mayoría de los sistemas

transaccionales los datos se guardan en diferentes tablas y se desea mostrar datos de ellas unidas mediante campos llave.

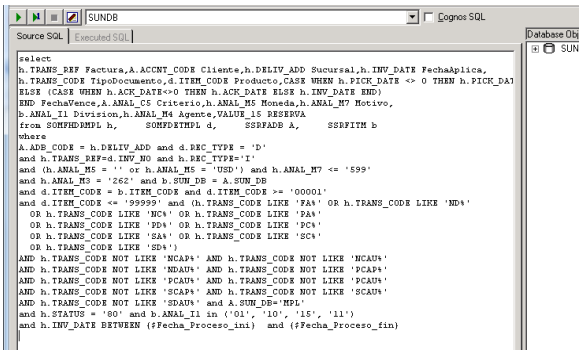


Figura 4.35: Área de trabajo para cargar las consultas de un datasource.

La consulta que estaremos utilizando para la carga de venta diaria se muestra a continuación, donde utilizamos diferentes tablas que serán unidas mediante distintos campos que cumplan las condiciones de relación.

```
select
h.TRANS_REF Factura,
A.ACCNT_CODE Cliente,
h.DELIV_ADD Sucursal,
h.INV_DATE FechaAplica,
h.TRANS_CODE TipoDocumento,
d.ITEM_CODE Producto,
CASE WHEN h.PICK_DATE <> 0 THEN h.PICK_DATE ELSE (CASE WHEN h.ACK_DATE<>0 THEN h.ACK_DATE
ELSE h.INV_DATE END) END FechaVence, A.ANAL_C5 Criterio, h.ANAL_M5 Moneda, h.ANAL_M7 Motivo,
b.ANAL_I1 Division, h.ANAL_M4 Agente, h.ANAL_M8 Plazo, d.VALUE_5 Unidades, CASE h.ANAL_M5 WHEN ''
THEN VALUE_18 - (VALUE_20 +  ALUE_15)  WHEN 'USD' THEN d.VALUE_20 END Importe, d.VALUE_20 DPP,
d.ANAL_M2 Motivo_doc, --{$Periodo} Periodo,    d.INV_PRD Periodo, d.VALUE_17 IVA, LEN(ltrim(A.ANAL_C5))
Conteo, ltrim(h.CUST_REF) Pedido, VALUE_15 RESERVA
from
SOMF h, SOMPL d, SSRFADB A, STM b
where
A.ADB_CODE = h.DELIV_ADD and d.REC_TYPE = 'D'
and h.TRANS_REF=d.INV_NO and h.REC_TYPE='I'
and (h.ANAL_M5 = '' or h.ANAL_M5 = 'USD')
```

```
and h.ANAL_M7 <= '599' and h.ANAL_M3 = '262'
and b.SUN_DB =  A.SUN_DB
and d.ITEM_CODE = b.ITEM_CODE
and d.ITEM_CODE >= '233201'
and d.ITEM_CODE <= '2339'
 BETWEEN {$Fecha_Proceso_ini} and {$Fecha_Proceso_fin}
```

Como podemos observar esta consulta requiere de una complejidad mayor ya que utiliza diferentes condiciones debido a la complejidad con la que está hecha la base de datos en SUN. Por mencionar una de ellas podemos ver la clausula between que nos indica el criterio para definir un rango de valores que deseamos visualizar dentro de nuestra consulta. Los valores de {$Fecha_Proceso_ini} y {$Fecha_Proceso_fin} son variables que estarán definidas dentro de nuestro proceso principal y que posteriormente definiremos al tener nuestro ETL finalizado.

En el transformation model (figura 4.36) vamos a tener dos lookups tomando como base lo hecho en el proceso 1, los cuales van a tener relación con los campos de factura y producto respectivamente. Además habrá varios derivations (campos calculados) que nos permitirán visualizar la información deseada en nuestro cubo multidimensional con las descripciones correspondientes ya que la información proveniente de nuestra fuente de datos normalmente no viene como el usuario la desea.

Para implementar el lookup es necesitan tener dos dimensiones asociadas a cada unos de ellos, en este caso tendremos factura y id_producto respectivamente los cuales serán explicados posteriormente, además de diferentes derivations que nos van a servir para dividir un campo por segmentos con la finalidad de mostrar los datos como el usuario los desea ya que debido a las necesidades del negocio el detalle de la información debe ser mayor proporcionando así datos a los niveles más bajos en las dimensiones que sea requerido.

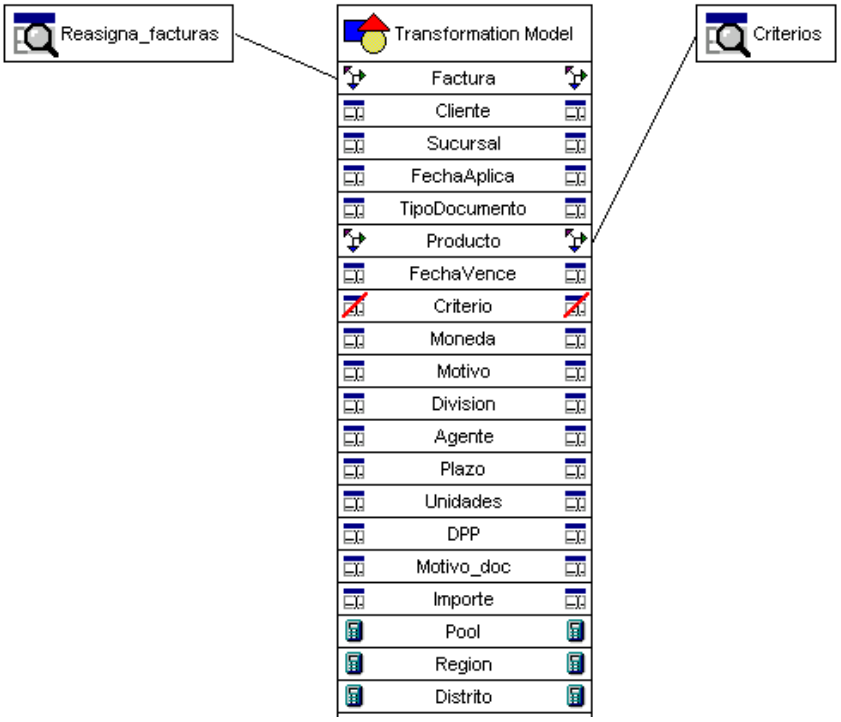


Figura 4.36: Estructura que representa un transformation model

- **Lookup: Reasigna_facturas**

Este lookup será utilizado para leer un archivo Excel que es administrado por el área de ventas ya que debido a las necesidades del negocio es necesario que manualmente se reasignen facturas hacia otros territorios de donde fueron vendidas. Con esto se evita principalmente realizar afectaciones directas a la base de datos mediante UPDATES y con ello se lleva una mejor administración sobre aquellas facturas que están siendo reasignadas. En caso de requerir reasignar otra factura basta con que el usuario responsable agregue la factura requerida en el archivo plano. Se está utilizando la sentencia SELECT para la carga de los datos donde tenemos los campos de: Factura y POOLNVO. El campo Factura nos servirá como llave para hacer la relación con nuestros datos que estamos cargando de SUN y por tal motivo es un registro único y no puede ser duplicado ya que esto ocasionaría que la información no se muestre

como es deseada. El campo POOLNVO es el que estaremos entregando en nuestra tabla y con ello se estará realizando una reasignación de territorio para ciertas facturas como se observa en la figura 4.37.

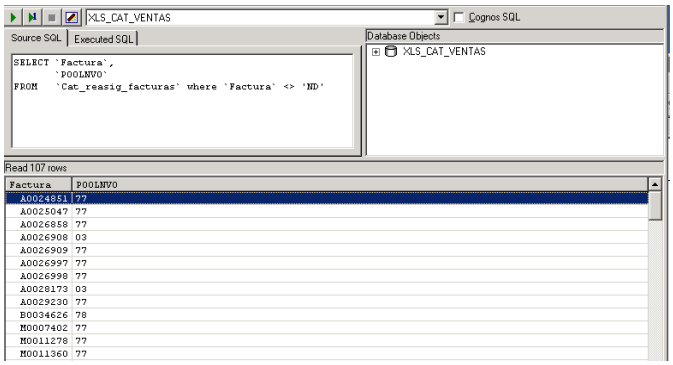


Figura 4.37: Fuente de datos para la carga de nuestro lookup.

Como podemos observar en nuestra tabla de atributos al estar definiendo nuestro lookup ponemos el campo Id_Factura como llave el cual será mapeado con nuestro campo de Factura mientras que el POOLNVO es el campo que entregaremos como nuevo valor cuando exista relación (figura 4.38).

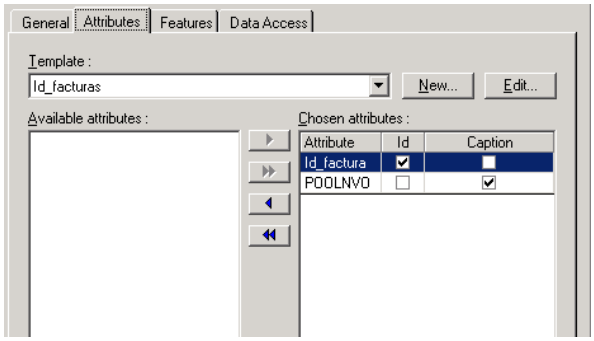


Figura 4.38: Forma para identificar atributos de cada campo.

En este proceso no es obligatorio que todos las facturas sean reasignadas, por tal motivo debemos permitir que aquellos registros que no existan dentro de nuestra relación en el lookup puedan ser entregados con su descripción original, para ello

debemos activar la opción Accept unmatched member identifiers como se muestra en la figura 4.39.

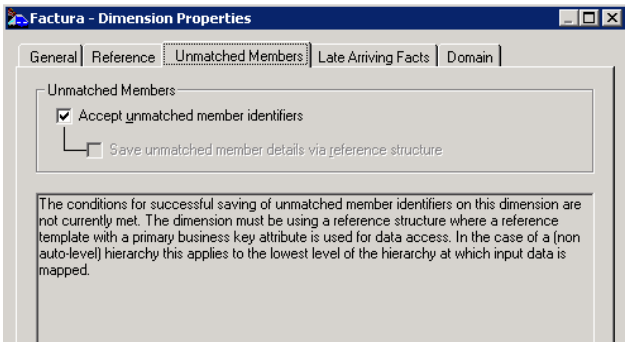


Figura 4.39: Opción para permitir la entrega de valores sin relación

- **Lookup Criterios**

En este lookup se están utilizando 2 Fuentes (figura 4.40) de datos ya que una de ellas es la información proporcionada por el área de ventas mediante un archivo en Excel, mientras que la otra proviene de una tabla de productos la cual fue cargada en uno de los procesos anteriores.

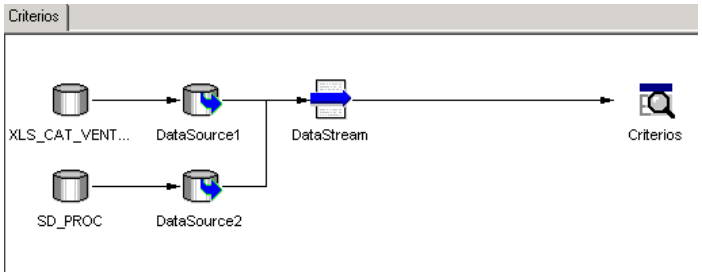


Figura 4.40: Vista de la lectura de dos fuentes en un mismo datasource

Para que ambas fuentes sean unidas y puedan ser entregadas como una sola fuente deben contener el mismo número de campos al momento de estar realizando nuestro mapping (relación), como podemos observar el número de columnas que estamos leyendo son cuatro, pero en la entrega solo utilizamos dos y es aquí donde definimos cuales son los campos que vamos a cargar (figura 4.41).

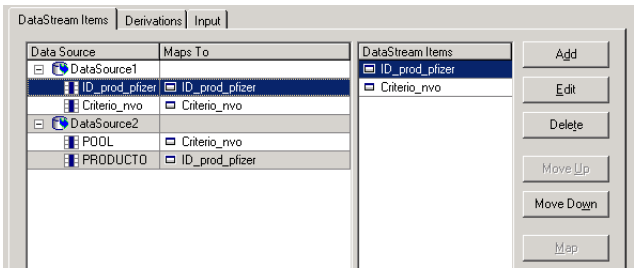


Figura 4.41: Mapeo de las fuentes

En la entrega utilizaremos un derivation donde vamos a crear una condición como se muestra en la figura 4.42 la cual nos indicará que tomará el campo de criterio siempre y cuando en la fuente no contenga datos en ese campo y le pegará el valor que obtengamos de nuestro lookup hecho anteriormente.

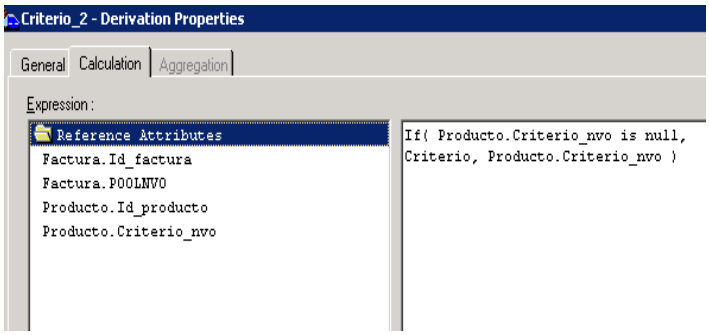


Figura 4.42: Muestra la forma de crear una condición para un derivation.

Proceso 7: FACT_PRORRATEA_VENTAS

La necesidad de tener este proceso es debido a que en la fuente original SUN no vienen datos necesarios como el trimestre el cual es necesario tenerlo para el análisis de los datos, así como la información en dólares tomando como base la tabla de tipos de cambio. Para ello se utilizaron dos lookups como se muestra en la figura 4.43, los cuales nos permitirán tener los datos en nuestro Fact table como es requerido para la creación de nuestro cubo multidimensional. Una vez cargado este proceso se iniciará

con la creación de nuestro modelo ya que se cuenta con la información necesaria en los catálogos y nuestra tabla de hechos.

Los datos que se cargarán están definidos en la siguiente consulta los cuales serán provenientes de nuestra tabla llenada en el proceso anterior mediante dos datasources, el cual contendrá dos consultas SQL utilizando la sentencia SELECT, además se implementará el UNION ALL que nos sirve para unir los datos de dos o más tablas según sea requerido, otra condición que utilizaremos es el NOT IN el cual nos indica que vamos a traer solo aquellos datos que no estén dentro del rango especificado.

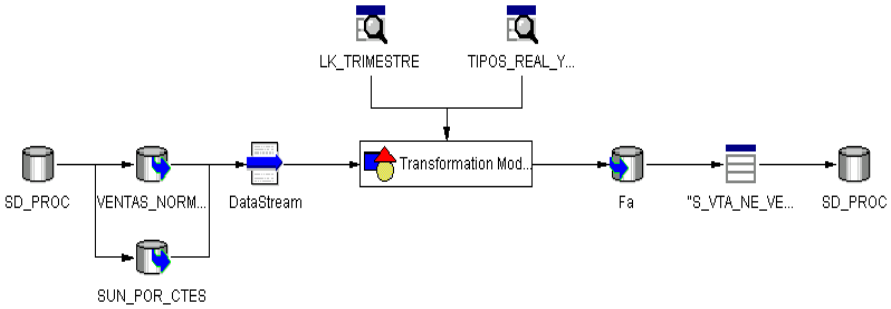


Figura 4.43: Esquema que representa el proceso de carga y transformación de las fuentes

En el primer datasource se van a cargar los datos provenientes de nuestras tablas llenadas previamente, como podemos observar antes de cada campo se pone un identificador (a, b) el cual nos sirve para indicarle a nuestro gestor de base de datos cual de las dos tablas definidas en nuestro FROM es la que va a utilizar para entregar el valor del campo correspondiente.

```
SELECT
a."FUENTE", a."FACTURA", a."CLIENTE", a."SUCURSAL", a."FECHAAPLICA", a."TIPODOCUMENTO",
a."PRODUCTO", a."FECHAVENCE", a."MONEDA", a."MOTIVO", a."DIVISION", a."AGENTE", a."PLAZO",
a."UNIDADES"*b."PROCENTAJE" UNIDADES, a."IMPORTE"*b."PROCENTAJE" IMPORTE, a."DPP",
a."MOTIVO_DOC", b."POOL_DES" POOL, a."REGION", a."DISTRITO", a."ZONA", a."FUERZA", a."RUTA",
a."PERIODO", a."IVA", concat(a."FACTURA",a."PRODUCTO") FACT_PROD, a.RESERVA
FROM
```

```
"S_VTA_VENTAS_SUNDB" a,
"S_VTA_NE_DIS_PRORRATEO" b
WHERE
a."POOL" = b."POOL_ORIG" and a.FechaAplica BETWEEN {$Fecha_Proceso_ini}
and {$Fecha_Proceso_fin}
```

En el segundo datasource (figura 4.44) se van a cargar los datos filtrados por cliente utilizando la sentencia IN que nos ayudará a definir un rango de datos que deseamos cargar, una vez teniendo los dos datasorces los vamos a mapear para que nos entreguen la misma cantidad de campos. La razón de utilizar dos dataources con la misma fuente es debido a que estos clientes tienen un trato especial en las unidades y el importe, mientras que en el datasurce anterior todo es multiplicado previamente definido en una tabla.

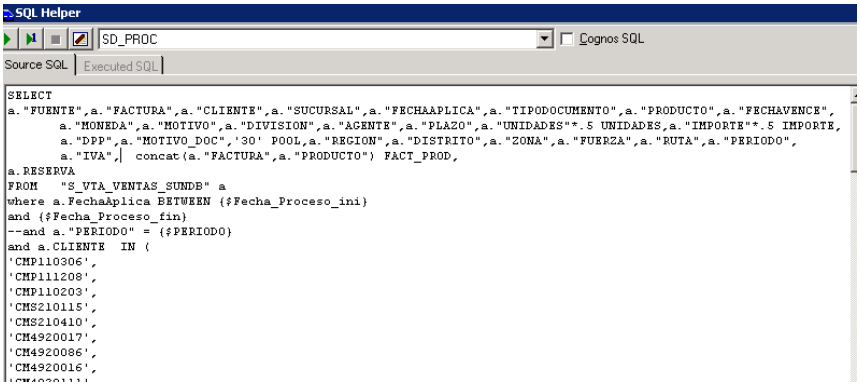


Figura 4.44: Carga de datos para clientes específicos.

La forma cómo quedará nuestro mapeo con las dos fuentes cargadas será como se muestra en la figura 4.45, donde podemos observar que el mismo número de campos es entregado en ambos casos para tener un mismo mapeo y posteriormente poder utilizar el transformation model, de esta forma estaremos homologando las dos fuentes extraídas anteriormente.

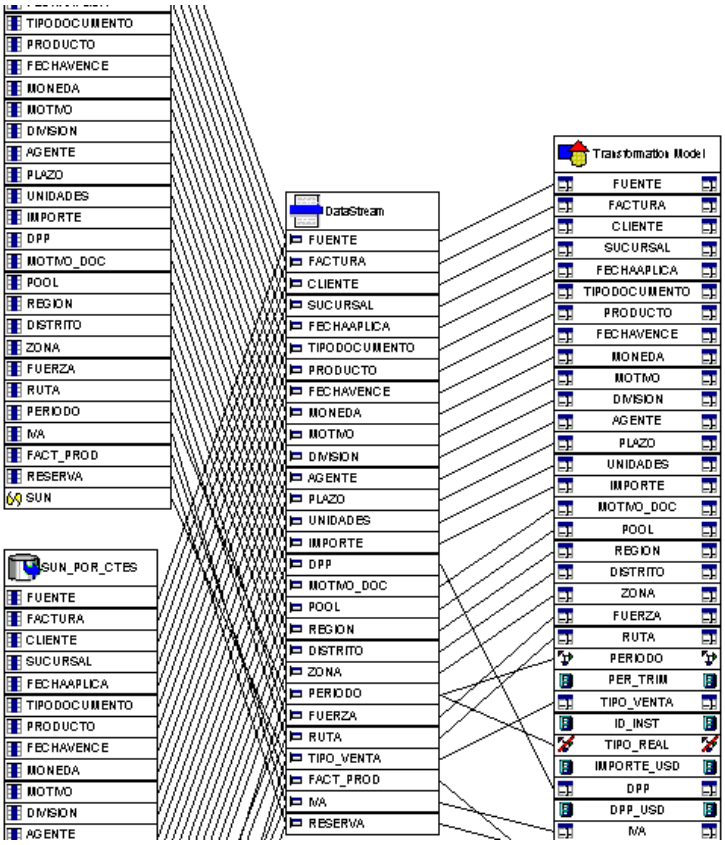


Figura 4.45: Mapeo de dos diferentes fuentes en una sola entrega de datos.

En el tranformation model vamos a utilizar los datos provenientes de nuestro lookup para unirlos con nuestra fuente original (figura 4.46), es importante mencionar que aquellos campos mostrados en la parte inferior son nuestros derivations que nos sirven para poder entregar datos posteriores provenientes de nuestro lookup. Una vez definido todo lo anterior podemos cargar los datos en nuestra tabla de hechos, tomando como referencia todo lo hecho en los procesos anteriores nuestros catálogos al igual que nuestra tabla final deberán estar homologados con la finalidad de no duplicar la información y mostrar los datos de forma íntegra para cada cierre de ventas. Con estos procesos se asegura que la información consultada por los usuarios va a estar actualizada al término de cada cierre de ventas y los datos podrán ser analizados por las áreas requeridas en el cubo multidimensional.

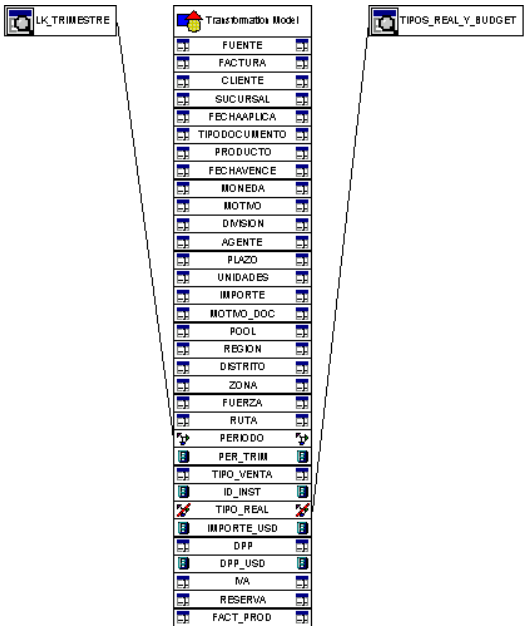


Figura 4.46: Integración de nuestras fuentes de datos con cada lookup.

Proceso 8: BORRA_REGISTROS

Este proceso es creado con la finalidad de no borrar toda la información histórica en cada cierre de ventas, por lo cual se ejecutará un proceso intermedio Borra_Registros el cual nos servirá mediante la sentencia DELETE (figura 4.47) para borrar aquellos registros cargados dentro de nuestra base de datos tomando como base una fecha inicial y una final para lo cual se utiliza la clausula WHERE, en caso de no utilizarla borraríamos todos los registros contenidos en nuestras tablas.

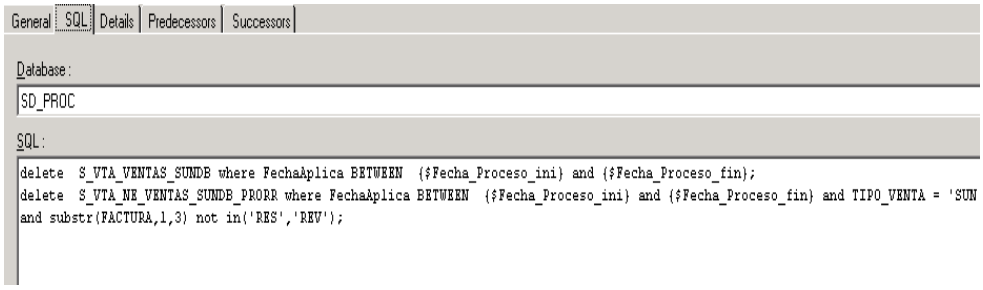


Figura 4.47: Estructura del proceso para el borrado de registros de una fecha específica.

Cada uno de los procesos anteriores sirven para alimentar las tablas que nos servirán como base para la creación de nuestras dimensiones en el cubo multidimensional, como podemos ver es una forma fácil de manipular los catálogos ya que en ocasiones basta con que lo usuarios cambien las descripciones en simple archivo plano lo cual permite la interacción directa de los datos que se muestran en un cubo con los usuarios. Para ello es necesario definir usuarios lideres que puedan tener acceso a dichos archivos en caso de ser necesario cambiar alguna descripción lo cual deberá ser definido por el área de ventas.

Proceso 9: VENTAS_DIARIAS

Para realizar la ejecución secuencial de cada uno de los procesos anteriores, es necesario crear un Jobstream (trabajo) como se muestra en la figura 4.48 que contendrá la forma y orden de cómo se estará ejecutando cada uno de los procesos, tomando en cuenta que la ejecución de cada unos de ellos deberá hacerse en el orden definido para garantizar la correcta actualización de los catálogos, así como la información de venta diaria.

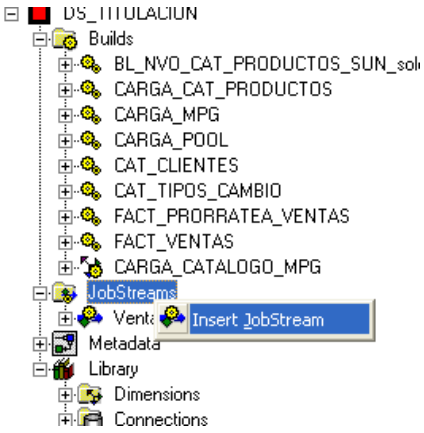


Figura 4.48: Muestra la creación de un jobstream.

Una vez creado nuestro nuestro jobstream, es necesario agregar cada uno de los procesos (builds) hechos anteriormente, para nuestro cubo multidimensional es

necesario iniciar con el borrado del periodo actual de las ventas diarias, posteriormente se cargarán los catálogos de cada una de nuestras dimensiones, al final se cargarán las ventas diarias del sistema de facturación y al final se ejecutará el cubo multidimensional de forma automatizada como se muestra en la figura 4.49 donde podemos observar la ejecución de algunos procesos en paralelo.

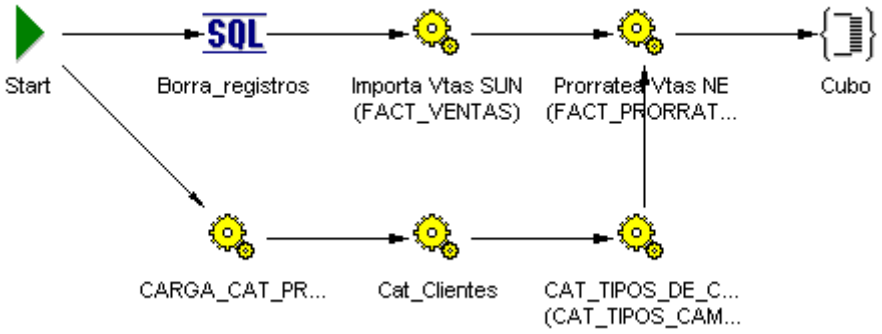


Figura 4.49: Muestra la secuencia del proceso final para la carga de ventas.

Una de las ventajas al utilizar jobstreams es que podemos ejecutar las tareas necesarias, e incluso si se desean agregar otras bastaría con agregarlas a nuestro proyecto, esto permite una fácil administración de nuestras tareas, otra de sus ventajas es que podemos definir nuestro trabajo para que sea detenido en caso de que alguno de nuestros builds falle, lo cual garantiza que la información no será procesada en caso de tener algún error, con esto los datos de consulta para el usuario final deberán estar siempre con datos reales.

Una vez creados los procesos necesarios para la carga de información en nuestro data warehouse, es necesario diseñar el cubo multidimensional que permitirá a los usuarios consultar la información de forma para poder realizar un análisis dinámico de los datos, la creación y definición del cubo multidimensional se explicará a detalle en capítulo V tomando como base todo lo realizado anteriormente.

CAPÍTULO V CREACIÓN DE UN CUBO MULTIDIMENSIONAL PARA PFIZER

V.1 Definición de Dimensiones

El modelo final creado para el área de ventas en Pfizer deberá tener las dimensiones y niveles que se muestran en la tabla 5.1, así como las métricas correspondientes para la generación de reportes y estadísticas, esto para poder satisfacer las necesidades del área en cuanto a la información se refiere, los datos que se estarán mostrando en el cubo son provenientes de las tablas cargadas anteriormente en nuestro ETL desarrollado en el capítulo anterior.

DIMENSIONES	NIVELES
PERIODO	- Periodo - Trimestre - Mes - Fecha
DIVISIONES	- Unidad de negocio - Región
LÍNEA TERAPÉUTICA	- Clase Terapéutica - Grupo Mpg - Mpg - Descripción
MERCADO	- Id Institución
CLIENTE	- Cliente - Sucursal
PRODUCTO	- Mpg - Descripción
TIPO FACTURACIÓN	- Tipo Facturación
TIPO DOCUMENTO	- Tipo Documento
MÉTRICAS	- Venta Importe - Venta Unidades - Venta importe Dólares - Venta Bruta Pesos - Venta Bruta Dólares

Tabla 5.1: Muestra el contenido de las métricas y dimensiones que debe contener nuestro cubo multidimensional.

- **PERIODO**

Esta dimensión va a contener los datos más relevantes en cuanto al tiempo, donde se podrá navegar desde el nivel superior hasta el mínimo que será la fecha, en esta dimensión los usuarios pueden visualizar los datos en cada nivel dependiendo de la información que requieran.

- **DIVISIONES**

En Pfizer se utilizan diferentes unidades de negocio para llevar un mejor control en su facturación, además estas son divididas en regiones, para ello tenemos esta dimensión que no ayudará a realizar un análisis detallado de los datos dependiendo de sus unidad de negocio donde se está vendiendo.

- **LÍNEA TERAPÉUTICA**

Cada producto vendido está asignado a una área terapéutica que a su vez esta subdividida en otros niveles, por tal motivo esta dimensión puede mostrar la información por producto, o bien por sus niveles superiores según sea requerido por lo usuarios al momento de estar analizando la información.

- **MECARDO**

Para el área de facturación es importante tener la información del mercado donde se están vendiendo sus productos, ya que actualmente tienen dividido su sector en dos grandes mercados que son: trade e institucional por lo que dicha información podrá ser consultada en esta dimensión.

- **CLIENTE**

La dimensión de cliente es muy importante dentro de nuestro cubo multidimensional, ya que gracias a esta dimensión se pueden identificar los clientes que se facturan durante todo el año y cuales presentan cambios dentro de sus compras en la compañía.

- **PRODUCTO**

Actualmente existe una gran cantidad de productos que son vendidos dentro de la compañía, en esta dimensión se podrá llegar hasta el nivel más bajo que es el producto partiendo de un nivel superior que es el MPG el cual agrupa ciertos productos con características similares lo cual es definido por el área de facturación.

- **TIPO FACTURACIÓN**

Es importante tener identificado en qué forma se está metiendo la facturación en el sistema y para ello esta dimensión nos ayudará a identificar el tipo de facturación que se está realizando ya que en ocasiones se utilizan archivos para afectar ciertas ventas lo cual se puede identificar dentro de nuestro cubo.

- **TIPO DOCUMENTO**

Está dimensión nos ayudará a identificar los cambios que se realizan cada cierre de periodo, ya que debido a que las condiciones del negocio son muy cambiantes es necesario cargar información extra según sea requerido. Por tal motivo la dimensión nos ayudará a identificar dichos cambios que en otros sistemas no son detectables.

- **MÉTRICAS**

Una métrica, o también conocido como indicador es el valor que representa la medición matemática de un aspecto del negocio, en un cubo multidimensional se pueden encontrar un sin número de métricas que van a depender directamente de las necesidades del negocio que ayude a análisis de los datos contenidos dentro de un cubo multidimensional.

Una vez que ya tenemos definidas nuestras dimensiones es necesario iniciar con la creación de nuestro cubo multidimensional y para la creación de nuestro metadatos utilizaremos Impromptu (generador de reportes), el cual nos sirve para crear nuestro catálogo con las relaciones correspondientes a cada una de las tablas creadas para

nuestro data warehouse, después de haber definido nuestro catálogo crearemos un reporte y después generaremos un IQD (Impromptu Query Definition) que contendrá los datos cargados en nuestras tablas mediante nuestro ETL para poder explotarlos con nuestro software generador de cubos Powerplay Transformer (generador de cubos multidimensionales) y con ello poder tener nuestro cubo que cumpla con las necesidades requeridas por parte de los usuarios y con ello permitir un análisis eficaz de la información.

Impromptu en una parte integral de la solución de inteligencia de negocios, ya que gracias a este podemos generar reportes cómodos para los usuarios que les permitan moverse fácilmente dentro de los reportes de resultados generados teniendo así la funcionalidad OLAP (On Line Analitical Processing o procesamiento analítico en línea). Además permite crear el catálogo que será utilizado para la generación de nuestro cubo.

V.2 Creación de Metadatos

El primer paso para la el desarrollo de nuestro metadatos es la definición de nuestro catalogo que contendrá las relaciones entre las tablas de nuestro data warehouse con lo cual garantizaremos la integridad de los datos que se estén mostrando en el cubo multidimensional, para este proceso utilizaremos Impromptu y vamos a ingresar a la aplicación como se muestra en la figura 5.1 dentro de los programas de Cognos.

Figura 5.1: Muestra la forma de ingresar a la herramienta impromptu para la generación de metadatos y reportes.

Una vez que ingresamos es necesario crear nuestro catálogo con las tablas empleadas para nuestro data warehouse, en este catálogo quedarán definidas todas las relaciones necesarias para crear nuestro IQD que será implementado el estar desarrollando el cubo multidimensional, para esto es necesario posicionarnos en la parte de catalog – new. Posteriormente vamos a llenar los campos solicitados con los datos correspondientes (figura 5.2), en la sección file name se deberá poner la ubicación donde deseamos sea guardado nuestro catalogo, mientras que en la parte description debemos poner una breve descripción sobre el contenido de nuestro catalogo con la finalidad de identificarlo fácilmente, en la sección databases vamos a elegir la conexión correspondiente a nuestro data warehouse, en nuestro caso utilizaremos una conexión ODBC la cual fue definida en la creación del ETL.

En caso de requerir otro tipo de conexión se puede elegir dependiendo de las necesidades de cada proyecto, de igual forma se pueden tener diferentes tipos de conexiones dentro de un mismo catálogo y con ello gestionar de diferentes bases de datos.

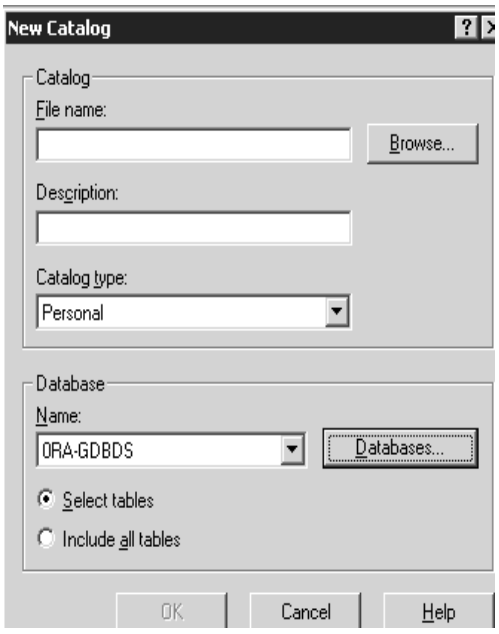


Figura 5.2: Área donde se indican las propiedades de conexión para nuestro catalogo.

Una vez que ya tenemos creada nuestra conexión a la base de datos es necesario asignar un password que servirá para darle seguridad a nuestro catalogo como se muestra en la figura 5.3 (e nuestro caso será dwh_titulacion), donde debemos considerar que el password asignado será indispensable para poder realizar cualquier cambio sobre nuestro catálogo.

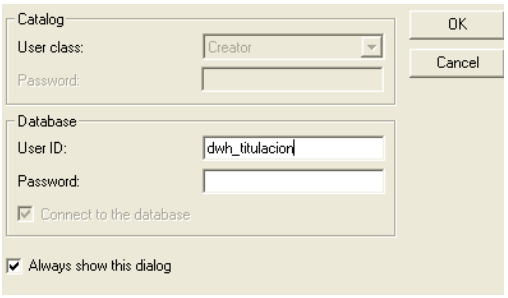


Figura 5.3: Sirve para definir un password a nuestro catalogo de impromptu.

Posteriormente es necesario hacer los joins correspondientes con cada una de las tablas, tomando en consideración que estas relaciones deben las mismas que se utilizaron en nuestro modelo relacional, ya que con esto garantizaremos la integridad de la información en nuestra base de datos respetando los criterios de integridad definidos previamente.

La creación de estos joins (figura 5.4) es primordial para que la información en nuestro cubo multidimensional se muestre de forma adecuada, ya que es en esta parte donde se creará el reporte necesario (IQD) para el desarrollo del cubo.

Una vez terminado el reporte con los datos solicitados por el usuario se iniciará con el desarrollo del cubo multidimensional tomando como base todo lo desarrollado con anterioridad. Las dimensiones del cubo estarán en función de las columnas generadas en este punto todo tomando como base las necesidades del área solicitante.

Figura 5.4: creación de relaciones en un catalogo de impromptu

Posterior a la creación de las relaciones con cada una de las tablas es necesario realizar un reporte (IMR) que no ayudará para definir nuestro el archivo con extensión IQD que utilizará Powerplay para definir cada una de las dimensiones de nuestro cubo.

En la figura 5.5 podemos observar la generación de un nuevo reporte tomando como base las tablas agregadas en nuestro catalogo definido previamente, los datos que tendremos en nuestro reporte será la información que puede ser procesada por nuestro cubo multidimensional y con ello mostrar la información al usuario para que sea consultada cuando esta sea requerida.

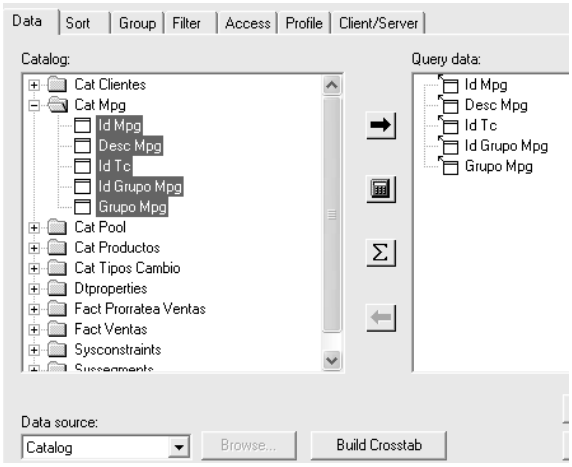
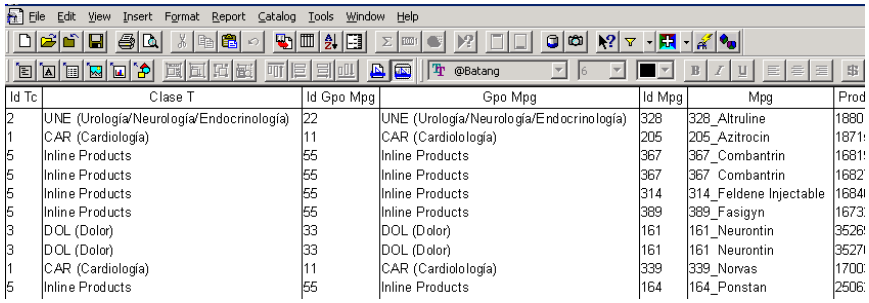


Figura 5.5: Área para agregar los campos necesarios en la creación de nuestro reporte

El reporte generado (Figura 5.6) servirá para que nuestro cubo obtenga la información proveniente de cada una de las tablas creadas y alimentadas por el ETL, dando así una vista al usuario sobre los datos procesados con anterioridad garantizando la integridad en la información mostrada.

Este reporte será la base del cubo multidimensional por lo que es necesario guardarlo con la extensión IQD, una vez terminado podemos iniciar con la creación del cubo.



Id Tc	Clase T	Id Gpo Mpg	Gpo Mpg	Id Mpg	Mpg	Prod
2	UNE (Urología/Neurología/Endocrinología)	22	UNE (Urología/Neurología/Endocrinología)	328	328_Altruline	1880
1	CAR (Cardiología)	11	CAR (Cardiolología)	205	205_Azitrocin	1871
5	Inline Products	55	Inline Products	367	367_Combantrin	1681
5	Inline Products	55	Inline Products	367	367 Combantrin	1682
5	Inline Products	55	Inline Products	314	314_Feldene Injectable	1684
5	Inline Products	55	Inline Products	389	389_Fasigyn	1673
3	DOL (Dolor)	33	DOL (Dolor)	161	161_Neurontin	3526
3	DOL (Dolor)	33	DOL (Dolor)	161	161 Neurontin	3527
1	CAR (Cardiología)	11	CAR (Cardiolología)	339	339_Norvas	1700
5	Inline Products	55	Inline Products	164	164_Ponstan	2506

Figura 5.6: Vista de un reporte para generar un IQD.

V.3 Creación del cubo multidimensional

Utilizando powerplay transformer se realizará la creación del cubo multidimensional, una vez ingresado a la herramienta (Figura 5.7) vamos a crear un nuevo proyecto, donde nos pedirá los datos necesarios para el desarrollo de nuestro cubo multidimensional, debemos considerar que nuestra fuente de datos será el IQD creado previamente.

Figura 5.7: Ingresando a powerplay transformer.

Una dimensión está compuesta por uno o más niveles de categorías ordenadas jerárquicamente lo cual ofrece una forma para obtener progresivamente detalle en la información asociada a la dimensión, los gerentes y personas con poder de decisión dentro de la organización requieren analizar una variedad de reportes para ver las tendencias y otros aspectos de la información, lo cual gracias a la implementación de cubos multidimensionales hoy en día es posible, gracias a la capacidad de consultar en datos, un cubo está compuesto de diferentes dimensiones o perspectivas del negocio y los cubos son objetos que estructuran la información en un formato multidimensional y ofrecen un medio interactivo para investigar y navegar dentro de la información de la organización.

El área para definir que nuestra fuente de datos se muestra en la figura 5.8, posteriormente es necesario indicar la ruta de donde estaremos tomando nuestro IQD, es importante mencionar que un IQD bien definido es la base principal para el buen funcionamiento de un cubo multidimensional en sus diferentes dimensiones y niveles.

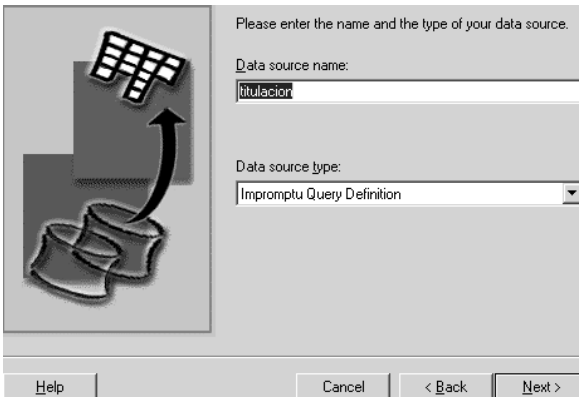


Figura 5.8: definición de nuestra fuente de datos para la creación de un cubo.

Después de haber definido la fuente de datos, así como el nombre de nuestro cubo, vamos a crear las dimensiones necesarias del modelo que nos permitirán realizar un análisis detallado de la información de acuerdo a las necesidades del área, cada dimensión debe contener los niveles predefinidos en nuestra tabla de definición.

La creación de dimensiones se hará dando clic derecho sobre el área de dimensión map (figura 5.9) y se crearán las dimensiones que sean necesarias según los requerimientos del área, en nuestro caso crearemos 9 dimensiones de acuerdo a lo solicitado por el área de ventas, cada una de las dimensiones deberá contener sus niveles correspondientes definidos anteriormente, con la finalidad de poder consultar la información dentro del cubo de lo general a lo particular, realizando cruces entre las diferentes dimensiones y con ello experimentar las ventajas y beneficios de utilizar un modelo multidimensional dentro del área ya que la información podrá ser consultada de forma más dinámica y eficiente.

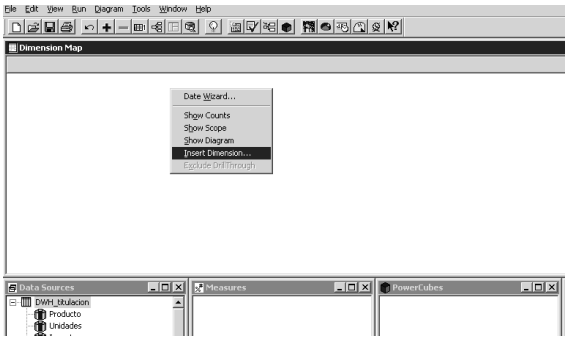


Figura 5.9: creación de dimensiones dentro de un modelo en powerplay.

Una vez que ya tenemos todas nuestras dimensiones (Figura 5.10) definidas es necesario asignar los niveles correspondientes a cada una de ellas, esto se hará tomando la información proveniente del data source generado previamente.

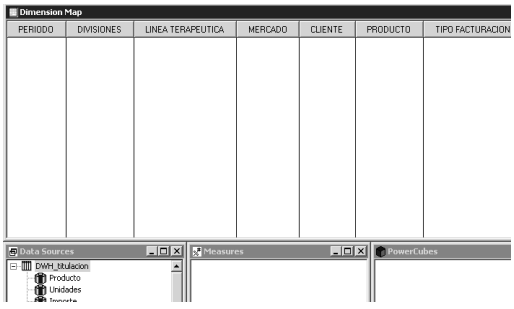


Figura: 5.10: Vista de las dimensiones creadas

Para agregar los niveles basta con arrastrar los datos de nuestro data source hacia la dimensión donde deseamos sea mostrado cada nivel, al terminar de definir cada uno de los niveles aun requerimos la creación de las métricas que servirán para consultar la información de ventas, una vez definidas estaremos en posibilidad de generar nuestro cubo multidimensional para ser publicado y consultado por lo usuarios.

En la figura 5.11 se muestra la estructura del modelo que será utilizado para la creación del subo multidimensional. En caso de requerir algún nivel, bastará con agregarlo en nuestro desde la fuente de datos, pero si el nivel solicitado no existe en la fuente será necesario modificar la definición de nuestro IQD para poder agregar los campos que sean necesarios.

El tipo de actualización de un cubo depende de la cantidad que este contenga, por lo cual es de vital importancia mostrar solo aquella información que sea relevante para la empresa.

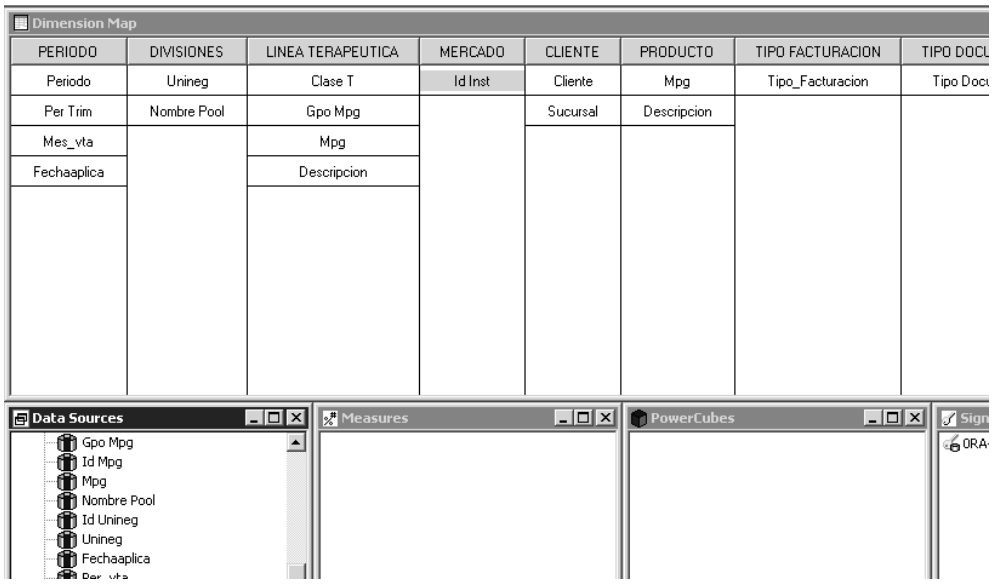


Figura 5.11: Muestra los niveles de cada una de las dimensiones del cubo.

La parte de las métricas es de vital importancia, ya que serán los indicadores con los cuales podremos consultar la información, estos datos provienen de nuestra fuente de

datos y se definen de acuerdo a las necesidades del área, en la Figura 5.12, se muestran las métricas que estaremos utilizando para analizar la información del cubo multidimensional.

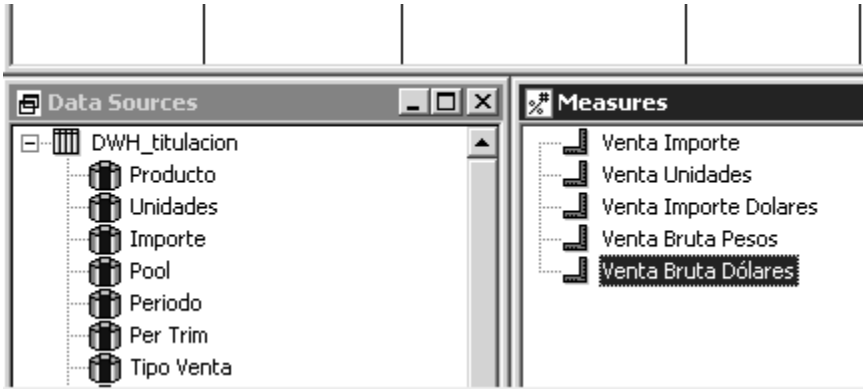


Figura 5.12: Área para definir las métricas para nuestro modelo.

V.4 Funcionalidad del cubo multidimensional

Dentro del cubo tenemos varias formas de hacer cruces de información con nuestras diferentes dimensiones y estos se realizan según las necesidades requeridas por el usuario.

En nuestro cubo en caso de requerir conocer la información de una fecha en específica tenemos que posicionarnos en la dimensión de PERIODO e inmediatamente despliega los años disponibles a consultar como se muestra en la figura 5.13.

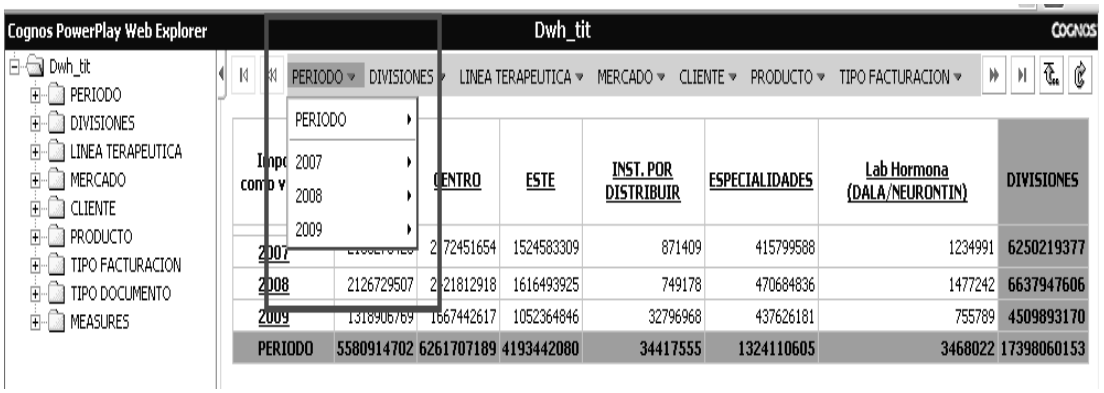


Figura 5.13: Navegación de dimensiones

Si deseamos ir a un nivel inferior bastará con seleccionar un año en específico e inmediatamente las cifras cambian como se puede observar en la figura 5.14.

Figura 5.14: Bajando el nivel de una dimensión

Otro ejemplo de navegación es para seleccionar un TRIMESTRE, PERIODO y DIA como se muestra en las figuras 5.15, 5.16, 5.17 partiendo de un nivel superior, a un nivel inferior según el detalle de la información que sea requerida.

Figura 5.15: Navegando en una dimensión

Figura 5.16: Bajando el nivel de una dimensión

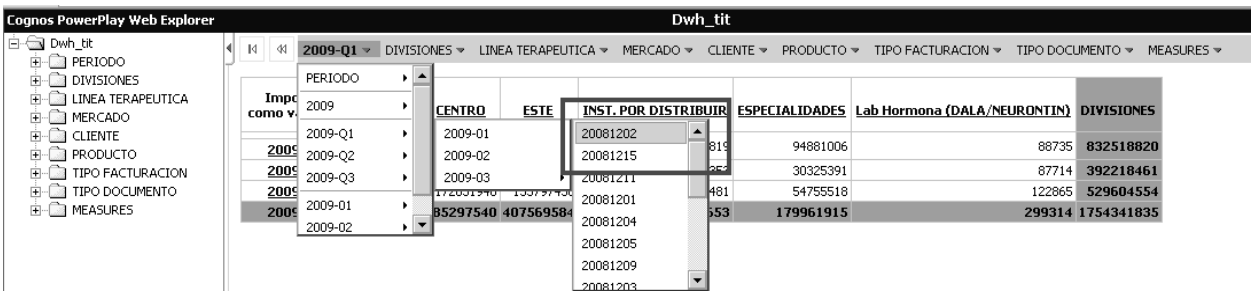


Figura 5.17: Obteniendo el nivel mínimo de una dimensión.

Estas mismas tareas las podemos hacer con todas las dimensiones de **DIVISION, LINEA TERAPEUTICA, MERCADO, CLIENTE, PRODUCTO, TIPO FACTURACION, TIPO DOCUMENTO Y MEASURES.**

Como podemos ver en las siguientes dos imágenes (5.18 y 5.19), cada que aplicamos un filtro los nombres de las dimensiones van cambiando según en el nivel en que estemos posicionados, de ésta forma podemos identificar los cruces de información que tenemos al momento.

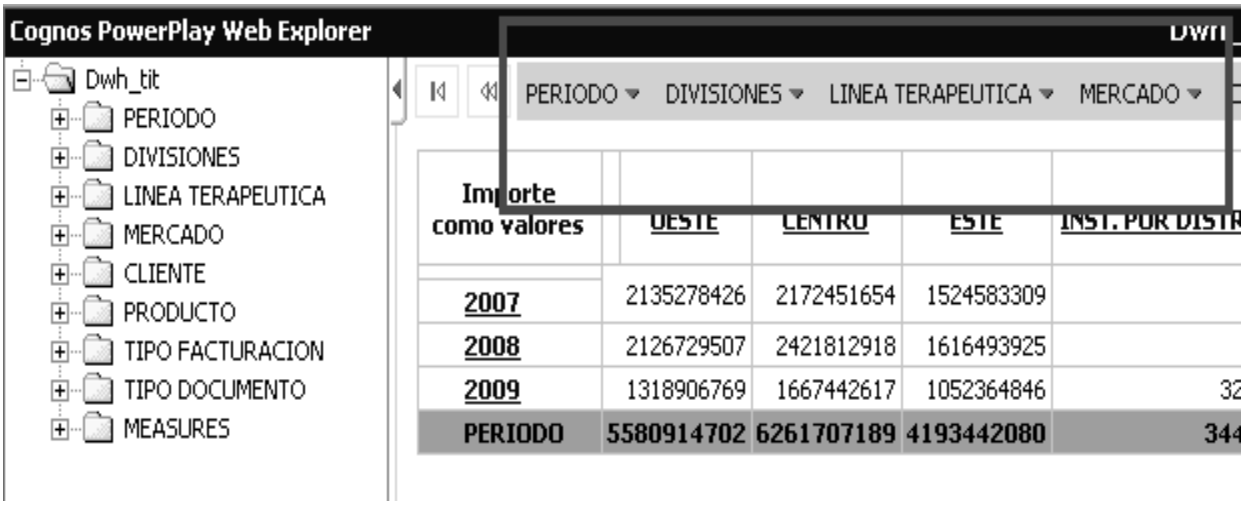


Figura 5.18: Muestra el cruce de las dimensiones que estamos consultando.

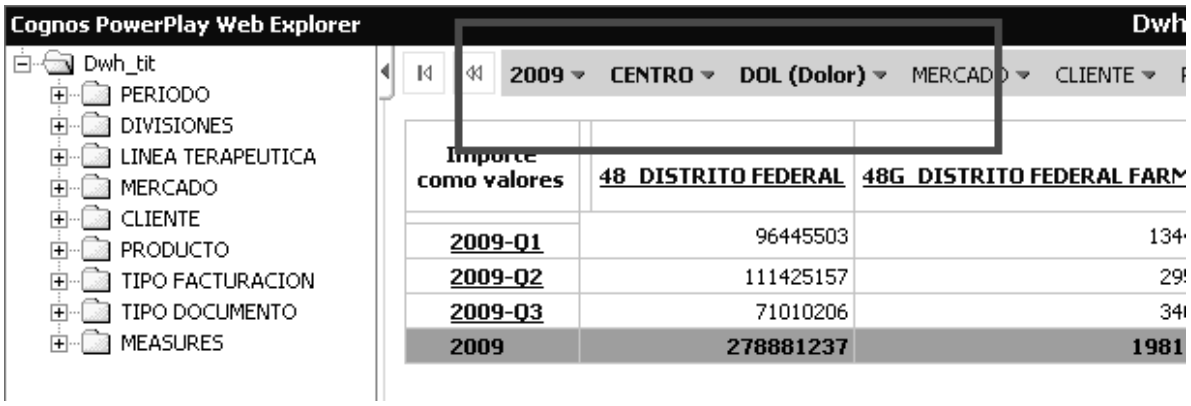


Figura 5.19: Muestra los datos de las dimensiones consultadas.

Otra gran utilidad de los cubos multidimensionales es que se puede visualizar todo el contenido de una dimensión arrastrándola hacia la parte inferior y después aplicarle los filtros requeridos.

Por ejemplo: Se requiere ver la VENTA BRUTA en DOLARES de todos los CLIENTES en el año 2008, de la división CENTRO, bastaría con arrastrar la dimensión requerida como se muestra en la figura 5.20

.

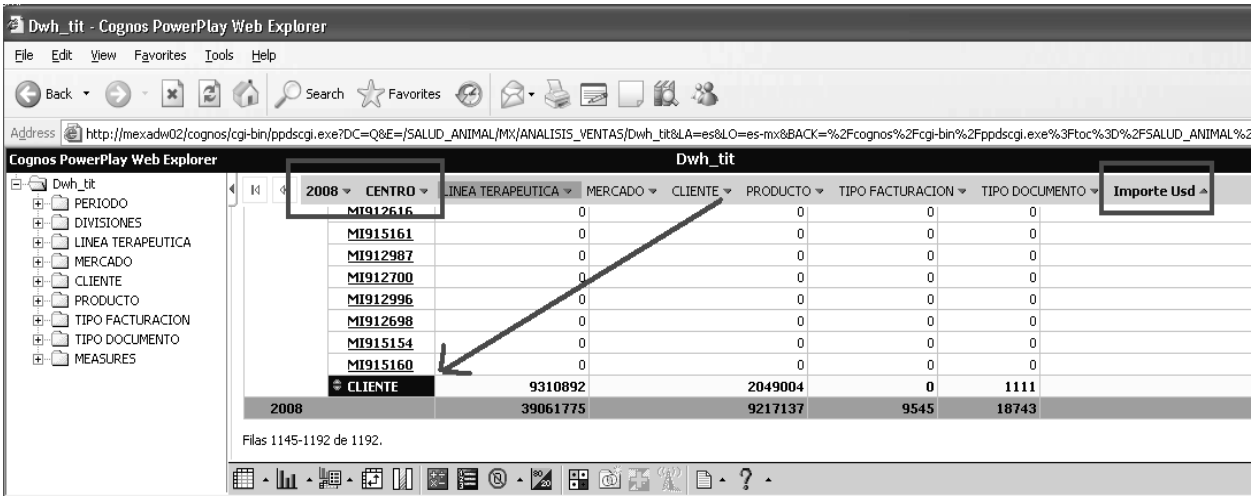


Figura 5.20: Método para intercalar las dimensiones según las necesidades requeridas.

Una opción de mucha utilidad para los usuarios es que se puede graficar de diferentes formas cada uno de los cruces que realizamos como se puede ver en figura 5.21, esto es de gran importancia ya que permite la generación de reportes y graficas dinámicamente al instante de modificar los cruces en nuestro cubo multidimensional, lo cual ayuda al análisis de información para la toma de decisiones dentro de la compañía.

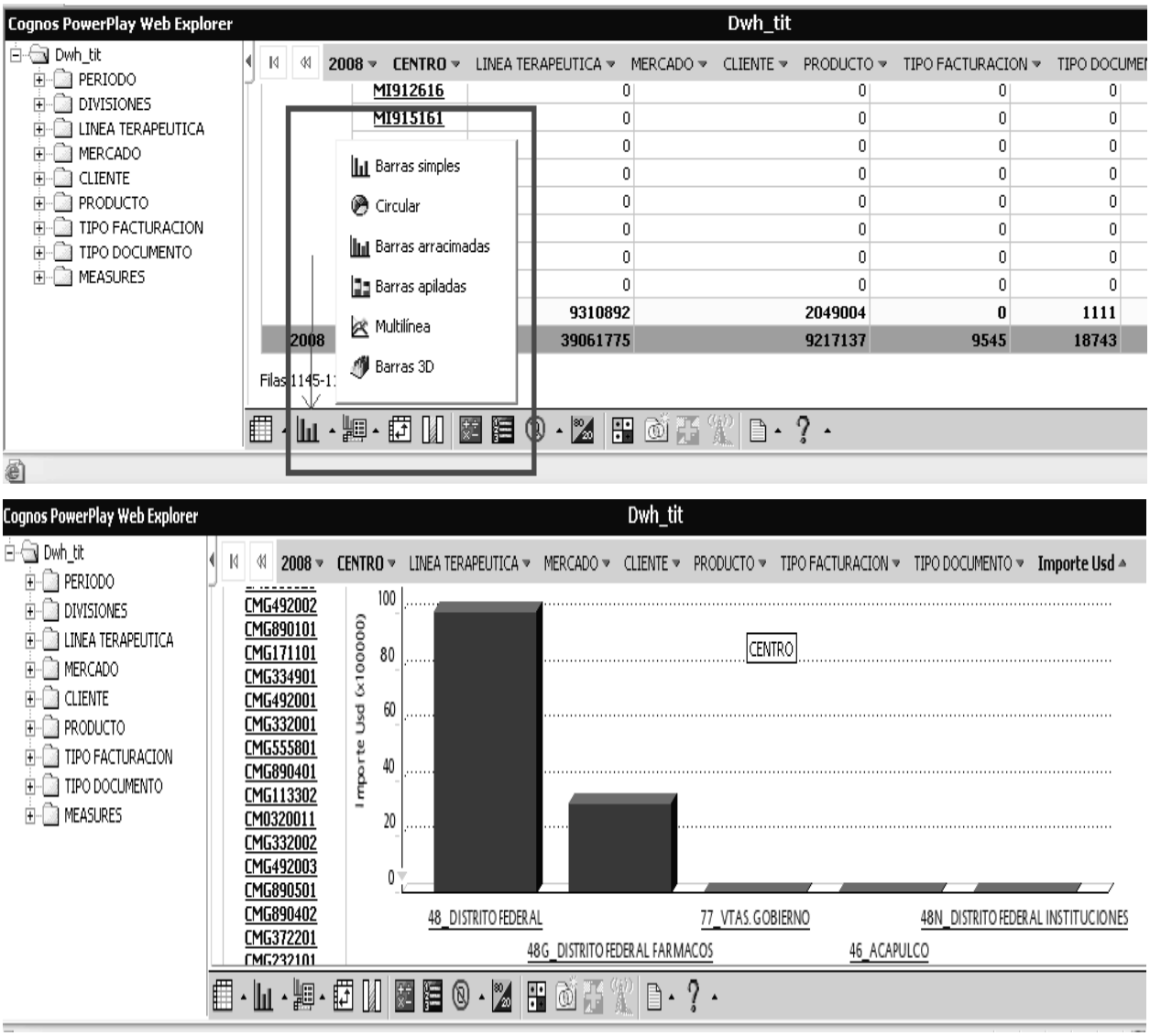


Figura 5.21: Muestra la forma para graficar dentro de un cubo multidimensional.

Cuando hay muchos ceros en la información y no es necesario visualizarlos dentro del cubo para la generación de un reporte se puede aplicar la opción para suprimirlos y con ello mostrar solo aquella información relevante para el área como se observa en la figura 5.22.

Figura 5.22: Opción para eliminar ceros dentro del cubo multidimensional.

Para la generación de reportes tenemos otra opción de gran utilidad ya que la información se puede exportar a un Excel, CSV ó PDF (figura 5.23) en el instante de estar realizando la consulta según las necesidades del usuario.

Figura 5.23: Método para exportar los datos del cubo a un archivo.

CONCLUSIONES

Hoy en día es de vital importancia que las empresas cuenten con la información actualizada sobre sus indicadores de negocio principales, debido a esto la creación de un cubo multidimensional, así como la implementación de un ETL tiene grandes beneficios ya que permite a los analistas de una organización tener los recursos necesarios de información para gestionar y hacer crecer de mejor manera la empresa basando sus decisiones en los datos proporcionados en un cubo multidimensional.

El crecimiento en la implementación de cubos multidimensionales está en auge debido a que las empresas tienen la necesidad de información sobre sus métricas principales, esto sin afectar los sistemas transaccionales, por ello es importante la realización de un análisis previo para definir las características de una base de datos integra que ayude a satisfacer en todo momento los requerimientos de los usuarios.

Gracias a la implementación del proyecto se cumplieron con los objetivos definidos, ya que las necesidades de información requeridas por parte de los usuarios quedarán cubiertas con la implementación del cubo multidimensional.

La implementación de este proyecto se logró con la ayuda de los conocimientos adquiridos en el diplomado Diseño de Negocios con SQL Server y Oracle ya que durante el desarrollo del mismo se aplicaron para lograr un buen desempeño de las herramientas implementadas.

BIBLIOGRAFÍA

1.- Fundamentos de Bases de Datos
Silberschatz Abraham, Korth Heny F. y Sudarshan S.
3ra Edición. Ed. Mc Graw Hill, 1998, 756 pp.

2.- Introducción a los Sistemas de Bases de Datos
C. J. Date.
Prentice Hall, 1999, 545 pp.

3.- Procesamiento de Bases de Datos
David M. Kroenke.
8va Edición. Ed. Prentice Hall, 281 pp.

4.- Sistemas de Administración de Bases de Datos
Gerald V. Post.
Ed. Mc Graw Hill, 193 pp.

5.- Análisis y diseño de Sistemas de Información
Witten Jeffrey L., Bentely Lonnied D. y Barlow Victor M.
3ra Edición, Ed. Mc Graw Hill, 1996, 890 pp.

PÁGINAS WEB

http://es.wikipedia.org/wiki/Base_de_datos

http://www-01.ibm.com/software/data/cognos/

http://www.gestiopolis.com/recursos/documentos/fulldocs/ger/bintna.htm

http://www.inegi.org.mx/inegi/contenidos/espanol/prensa/Contenidos/Articulos/tecnologia/relacional.pdf