Universidad Tecnológica Nacional

Facultad Regional Mendoza

Sistemas de Gestión II

Implementación de Datawarehouse

**Caso: “Data Warehousing y OLAP para la Industria de Comidas Rápidas”**

**Alumno:**

* Ojeda Juan Manuel Leg. 25197

Indice de contenidos del informe

[Descripción del Caso 3](#_Toc299895300)

[Introducción 3](#_Toc299895301)

[Tecnología del almacén 3](#_Toc299895302)

[Áreas Temáticas: 3](#_Toc299895303)

[Requisitos particulares: 4](#_Toc299895304)

[Índices a resolver: 4](#_Toc299895305)

[Implementación de la Solución propuesta 6](#_Toc299895306)

[Diseño del Datawarehouse: 6](#_Toc299895307)

[Descripción del Datawarehouse: 7](#_Toc299895309)

[Descripción de las tablas de Hechos: 7](#_Toc299895310)

[Descripción de las tablas de dimensiones: 7](#_Toc299895311)

[Hecho Ventas: 8](#_Toc299895312)

[Índices resueltos por el hecho “Ventas”: 9](#_Toc299895313)

[Hecho “Volumen de Movimientos”: 9](#_Toc299895314)

[Índices resueltos por el hecho “Volumen de Movimientos”: 10](#_Toc299895315)

[Tecnología utilizada en el diseño y la Implementación del DW. 10](#_Toc299895316)

[Herramientas y programación 10](#_Toc299895317)

[Resumen de la configuración de Microstrategy 11](#_Toc299895318)

[Diseño ETL 17](#_Toc299895319)

[Indicadores Utilizados 19](#_Toc299895320)

[ANEXO 1 – Extracto de codigo para cargar la BD. 20](#_Toc299895321)

Descripción del Caso

Introducción

La industria de comida rápida es altamente competitiva, un cambio muy pequeño en las operaciones pueden tener un impacto significativo en la línea de operaciones. Por esta razón, el acceso rápido a la información completa para la presentación de informes estándar y bajo demanda, es esencial.

DecisionSoft ha diseñado e implementado un Data Warehouse para hacer frente a esta exigencia de Springwood Corporation, una franquicia de comida rápida que opera aproximadamente 20 Taco TABASCO y restaurantes Kentucky Fried Chicken alrededor del país. El Springwood Data Warehouse proporciona ahora soporte de decisiones estratégicas y tácticas para todos los niveles de gestión dentro de Springwood.

Tecnología del almacén

El almacén de datos se implementa en Microsoft SQL Server 2000, e incorpora los datos de dos fuentes principales:

* Todos los días información de ventas es suministrada de forma automática por el sistema de facturación.
* Información contable desde la base de datos de contabilidad Dynamics (Microsoft Great Plains).

Estos datos multidimensionales, se actualizan automáticamente de forma periódica (o bajo demanda, si es necesario) y se mantienen históricamente durante varios años con fines comparativos.

A efectos de información y análisis, los datos en el almacén se transforman en cubos OLAP. Los cubos se acceden a través de Excel utilizando MD BusinessQuery.

Los datos pueden ser analizados por tienda, por empresa, por área y zona, por año, trimestre y el mes, y por la marca y el concepto.

Áreas Temáticas:

1. Datos de pérdidas y ganancias y flujo de caja para Springwood. Las cantidades se pueden ver durante un período, trimestre a la fecha, de año a la fecha, y se puede comparar con cualquiera de los dos presupuestos, en comparación con el mismo período del año anterior, o como porcentaje de las ventas.
2. Mezcla de Ventas: Contiene las ventas diarias de todos los elementos del menú en todas las tiendas. Además de los parámetros de análisis estándar, estos datos también puede ser analizado por marca, por tipo de posición o elemento de menú, por año calendario, mes y semana, y por el nivel de precios. Este cubo se puede utilizar para calcular los importes de ventas, los costos y la variación de precio de lista.
3. Cubos de Ventas (partes del día). Contiene los importes de ventas y cuenta en intervalos de 15 minutos. Además de los parámetros de análisis estándar, los datos de este cubo también se pueden analizar por año calendario, mes y semana, y por ocho horas, cuatro horas, dos horas, una hora y 15 minutos de intervalo, o específicos comida (por ejemplo, almuerzo, cena, desayuno, entre comidas, etc.)
4. Resumen de Ventas Diarias. Contiene resumen diario de ventas de cada tienda. Además de los parámetros de análisis estándar, estos datos también pueden ser dimensionado por un indicador comparable, por año calendario, mes y semana, y por el nivel de precios. Las ventas brutas, las ventas gravadas, las ventas no gravadas.

Muchas cantidades se pueden ver, opcionalmente, como las variaciones, como un porcentaje de las ventas, o se resumen en la semana a la fecha, período, mes.

1. Reportes de Costos de Ventas. Muestra la cantidad y el costo promedio de artículos, considerando como unidad mínima de análisis, el ítem de menú. Debe realizarse un análisis geográfico de costos, e incluyendo la estructura corporativa.

La estructura de datos implementado para Springwood les permite mantener varias estructuras organizativas distintas con el fin de representar adecuadamente a cada tienda en un (1) la estructura corporativa, es decir, la filial de que pertenecen, (2) la estructura de operaciones, es decir, la zona o área y (3) el concepto de estructura, es decir, KFC, la TB (Taco TABASCO ), Burger King, etc.

El almacén de datos Springwood y la consiguiente estructura de cubos OLAP puede navegarse a lo largo de cualquiera de estas jerarquías corporativas - es decir, por la compañía operadora, según la zona o zonas, o por marca o concepto. Esto permite comparaciones entre los conceptos, por ejemplo, o de todas las tiendas dentro de un concepto. Del mismo modo, es fácil hacer comparaciones de zona a zona, o para ver el desempeño de todas las tiendas dentro de un área.  
El almacén de datos tiene una dimensión de tiempo. Este calendario se ha construido en el almacén, lo que permite una fácil comparación de cualquier período del ejercicio anterior o el mismo período de un año antes. En lugar de comparar a nivel período, las comparaciones y las tendencias se pueden hacer a nivel trimestral o anual. (Excepto para los casos de decisiones de más bajo nivel, por ejemplo, comparar semana a semana e incluso día a día).  
En caso de Beneficios y Pérdidas se compara cada período durante los últimos cinco años, hasta el nivel de cuenta. Esto hace que sea fácil para Springwood para evaluar las tendencias en cualquier categoría de gasto, comparando tienda a tienda, período a período, una zona a zona, o concepto-concepto.

Los cubos de Ventas Diarias se actualizan durante la noche con la información capturada por las cajas registradoras en cada tienda. Esto permite a los gerentes para evaluar y comparar las tendencias en la rapidez del servicio, el uso de mano de obra, las compras de alimentos de los empleados, etc., por el almacén, zona, área, concepto, etc. Debido a las ventas se registran en intervalos de 60 minutos, llamadas Partes del día, los administradores pueden usar esto para encontrar patrones de ventas extraños, posiblemente indicativos de robo de los empleados, durante las horas de la medianoche.

Requisitos particulares:

* Las ventas diarias se analizan en el segundo Nivel de la organización y podría darse a cada gerente de Tienda la autorización para el análisis de sus propios datos.

Índices a resolver:

1. Mostrar Análisis de costos, ya sea histórico y además geográfico.
2. Analizar históricamente las ventas, podría desagregarse por Estructura Corporativa (para determinar si hay mayores o menores incidencias de tiendas)
3. Ranking de Ventas por Estructura de Productos de manera de estudiar preferencias.
4. Análisis de Tendencia para pérdidas y ganancias, Ranking de gastos por estructura corporativa.
5. Análisis de Importes de Ventas por intervalos de tiempo, para Gerentes de segundo nivel de la organización.
6. Reporte de cifras de Ventas gravadas, no gravadas, etc.
7. Tabla dinámica de Mix de Ventas, detallando hasta el Ítem de Menú, posición de menú y sus costos.
8. Relacionar en una tabla dinámica, Tiendas (y correspondiente estructura corporativa) con las estructura de productos vendidos.
9. Comparación entre distintos periodos de pérdidas y ganancias según las cuentas contables. Tabla y gráfico.
10. Total de ventas realizadas a Empleados. (horarios y productos).
11. Análisis geográfico de ventas mensuales, para estudiar posibilidades de abrir nuevas sucursales
12. Rentabilidad Económica de Productos del último mes, comparando contra un valor objetivo.

Implementación de la Solución propuesta

Diseño del Datawarehouse:



Descripción del Datawarehouse:

Debido a las características del caso presentado y al tamaño del datawarehouse a partir del cual se pretende implementar la solución se ha logrado llegar a un modelo de DW con formato de estrella. Con este tipo de formato de warehouse se consigue un mayor rendimiento a la hora de procesar. Esta decisión de implementación se tomó en base a que el negocio en cuestión lo permite, es decir, no quedan tablas excesivamente grandes que puedan ralentizar el procesamiento.

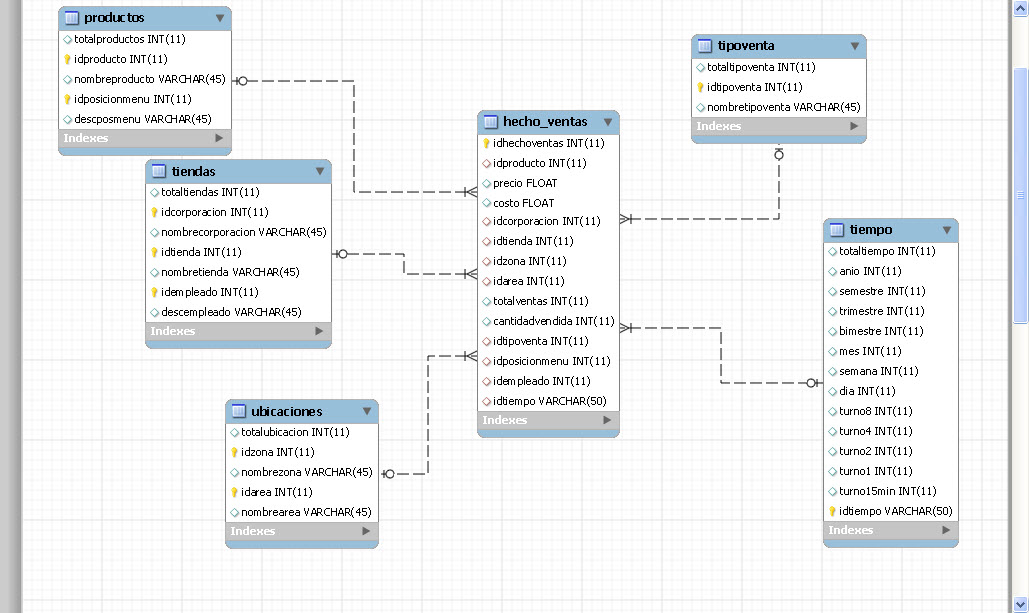
Descripción de las tablas de Hechos:

* Ventas:
  + Monto de las ventas realizadas de los productos de ambas empresas.
  + Cantidad Vendida
  + Tipo de Comida
  + Tipo de Venta
  + Cantidad de Artículos
* Volumen de movimientos:
  + Resultado del ejercicio
  + Total Caja
  + Promedio de Costos
  + Contable: montos de ingresos que son imputados a cada empresa.

Descripción de las tablas de dimensiones:

* + - Producto:
      * Contiene la descripción de cada uno de los productos y de los menúes ofrecidos por alguna de las 2 compañías en estudio.
    - Ubicación:
      * Se encuentra la información relevante a las zonas y áreas en donde existe instalada alguna tienda perteneciente alguna de las firmas de comida rápida en cuestión.
    - Tienda:
      * En esta tabla podemos encontrar información acerca del tipo de tienda y de los empleados afectados a cada una de las mismas además de poder conocer a que corporación pertenece una tienda en particular.
    - Tipo de Venta:
      * Contiene los tipos de ventas que pueden existir en estos negocios.
    - Tiempo:
      * En esta tabla se encuentran todas y cada una de las posibles divisiones de tiempo solicitadas para el estudio de los umbrales temporales que exige el estudio.

Hecho Ventas:



**Índices resueltos por el hecho “Ventas”**:

* Mostrar Análisis de costos, ya sea histórico y además geográfico. (1)
* Analizar históricamente las ventas, podría desagregarse por Estructura Corporativa (para determinar si hay mayores o menores incidencias de tiendas) (2)
* Ranking de Ventas por Estructura de Productos de manera de estudiar preferencias. (3)
* Análisis de Importes de Ventas por intervalos de tiempo, para Gerentes de segundo nivel de la organización. (5)
* Reporte de cifras de Ventas gravadas, no gravadas, etc. (6)
* Tabla dinámica de Mix de Ventas, detallando hasta el Ítem de Menú, posición de menú y sus costos. (7)
* Relacionar en una tabla dinámica, Tiendas (y correspondiente estructura corporativa) con las estructura de productos vendidos. (8)
* Total de ventas realizadas a Empleados. (horarios y productos). (10)
* Análisis geográfico de ventas mensuales, para estudiar posibilidades de abrir nuevas sucursales. (11)

Hecho “Volumen de Movimientos”:

# E:\Documents and Settings\Administrador\Escritorio\hechoVlumen.jpg

Índices resueltos por el hecho “Volumen de Movimientos”:

* Análisis de Tendencia para pérdidas y ganancias, Ranking de gastos por estructura corporativa. (4)
* Comparación entre distintos periodos de pérdidas y ganancias según las cuentas contables. Tabla y gráfico. (9)
* Rentabilidad Económica de Productos del último mes, comparando contra un valor objetivo. (12)

Tecnología utilizada en el diseño y la Implementación del DW.

Los diagramas de BD y de DW fueron realizados utilizando las herramientas E.Architect y Microsoft Visio respectivamente.

La BD se implementó utilizando el motor MySQL por lo que el esquema resultante es del tipo ROLAP

Herramientas y programación

|  |  |  |
| --- | --- | --- |
| **Nombre Herramienta** | **Descripción** | **Logo** |
| Python | Lenguaje de Programación con el que se desarrolló el ETL y los scripts responsables de llenar la BD . |  |
| MySQL | Base de Datos utilizada para implementar el modelo relacional . |  |
| MySQL - workbench | Interfaz usuario para BD MySQL utilizada para realizar la ingeniería inversa |  |
| Microstrategy V.9 | Permite realizar Businness Intelligence, para crear y gestionar data warehouse y sistemas OLAP, tomando los datos almacenados en una base de datos relacional y de otras fuentes. También posee una interfaz gráfica para crear informes. |  |
| Visio | Herramienta de modelado visual utilizada para crear diagramas de forma rápida |  |
| navicat | Interfaz usuario para utilizar con MySQL | E:\Documents and Settings\Administrador\Escritorio\navicat.jpg |

Resumen de la configuración de Microstrategy

* Paso 1 : Se procede a construir el almacén de metadatos para que el software pueda absorver datos con los que realizará su trabajo
* Paso 2: Ingresamos al wizard que nos provee este s.w. para poder configurar el proyecto nuevo
* Paso 3: dentro del wizard seleccionemos “Crear Metadata de la base de metadatos”



* Paso 4 : Seleccionamos los tipos de repositorio a configurar ( Tablas de Historial y Metadata )



* Paso 5 : Ingresamos el nombre del repositorio donde se almacenan los datos y los datos del usuario que accederá a los mismos



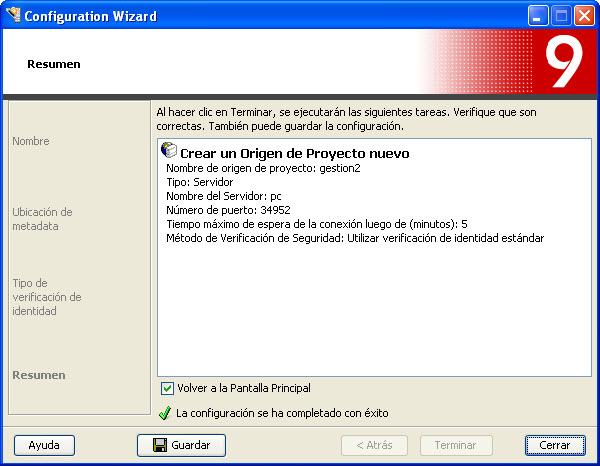
* Paso 6: Se configura a continuación el Servidor que se utilizará para la comunicación con la base de metadatos



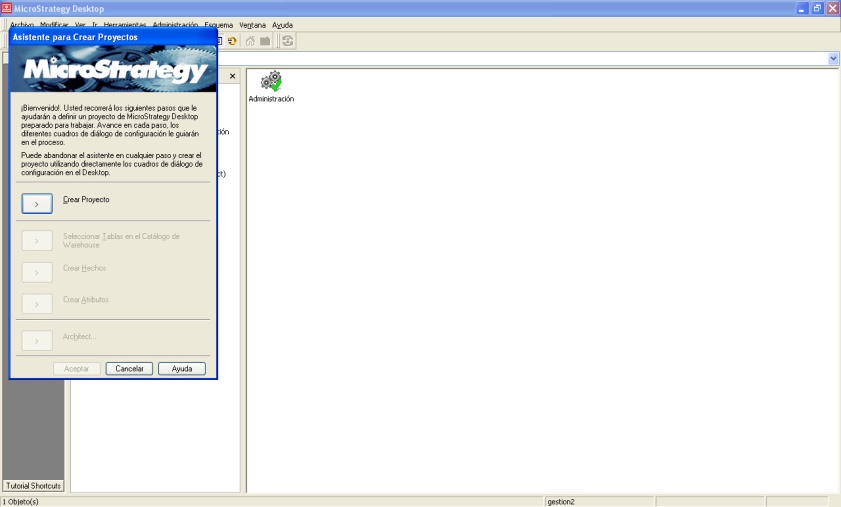
* Paso 7: Al igual que en el caso anterior se ingresa el nombre de usuario pero en este caso dejamos el que ofrece el sw por defecto.



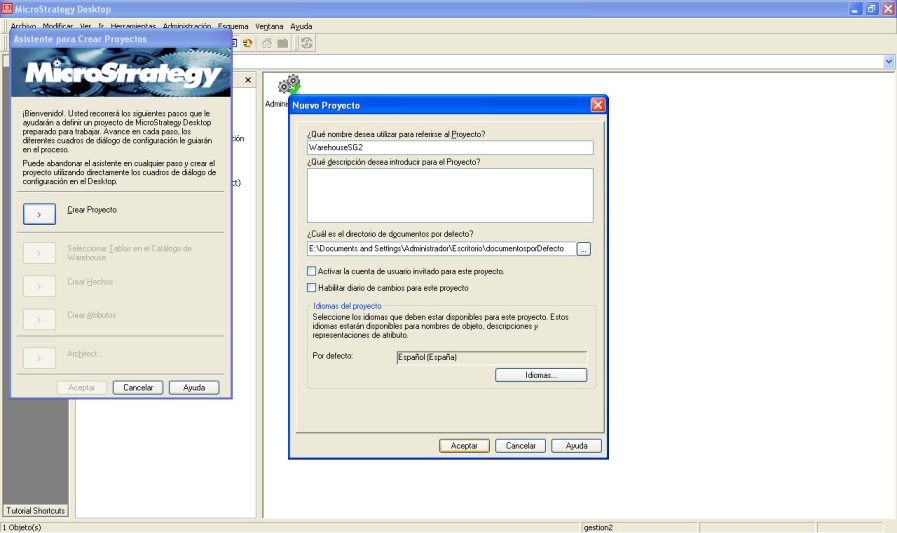
* Paso 8: Despues de navegar a través del wizard y configurar parámetros básicos (tipo de autenticación, etc)de configuración llegamos a la pantalla siguiente.



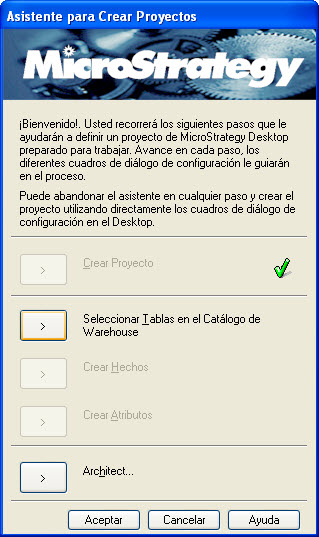
* Paso 9: iniciar el asistente para crear un nuevo proyecto

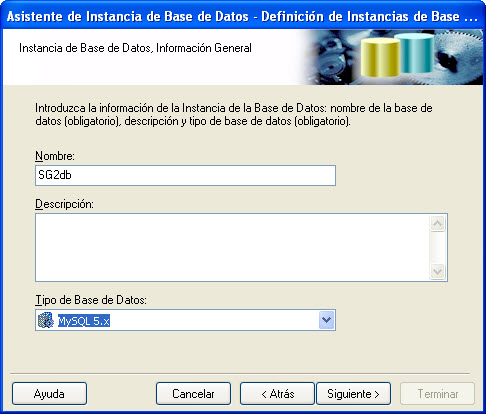


* Paso 10: iniciar el asistente para crear un nuevo proyecto

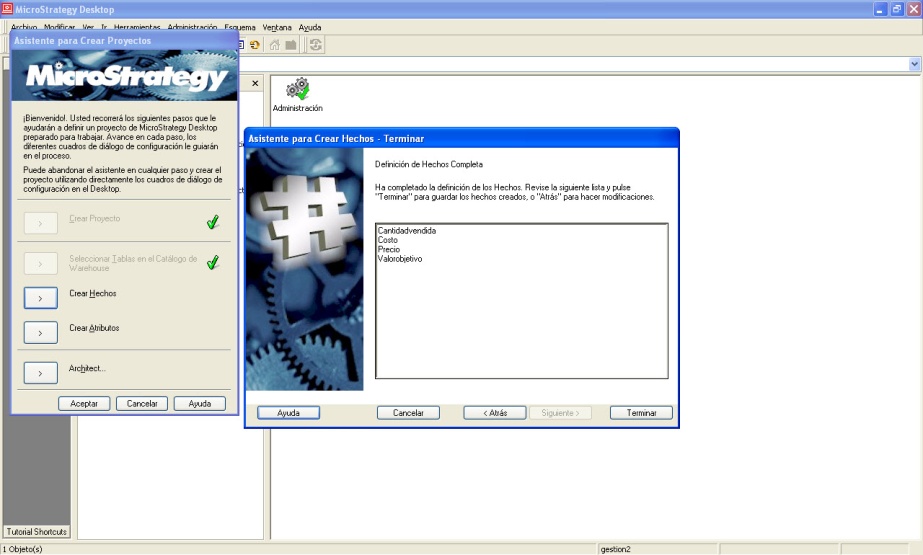


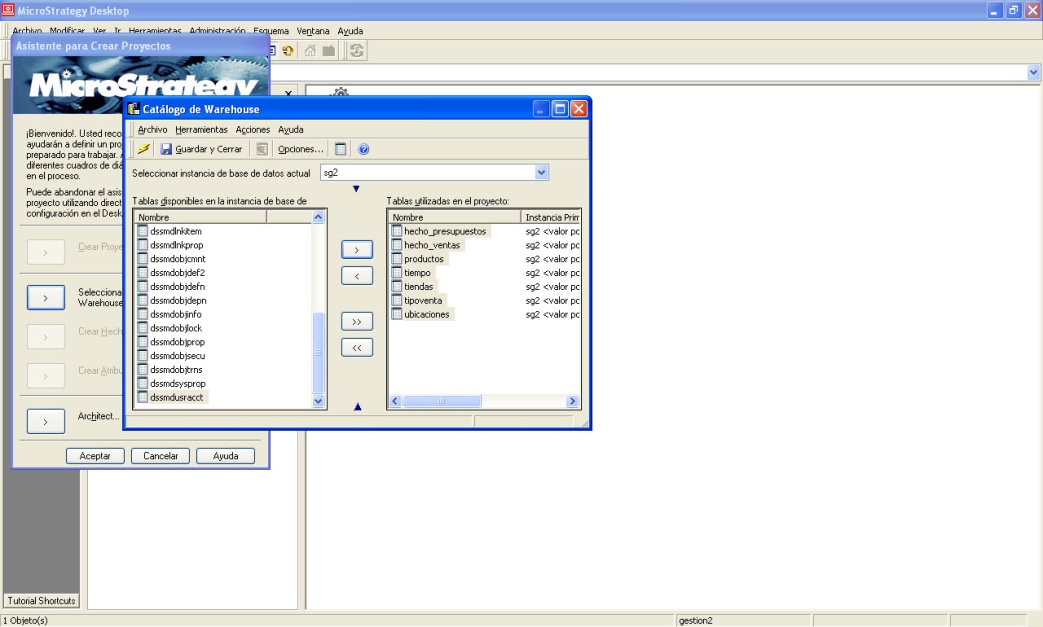
* Paso 11: Seleccionamos las tablas de origen y el tipo de BD.



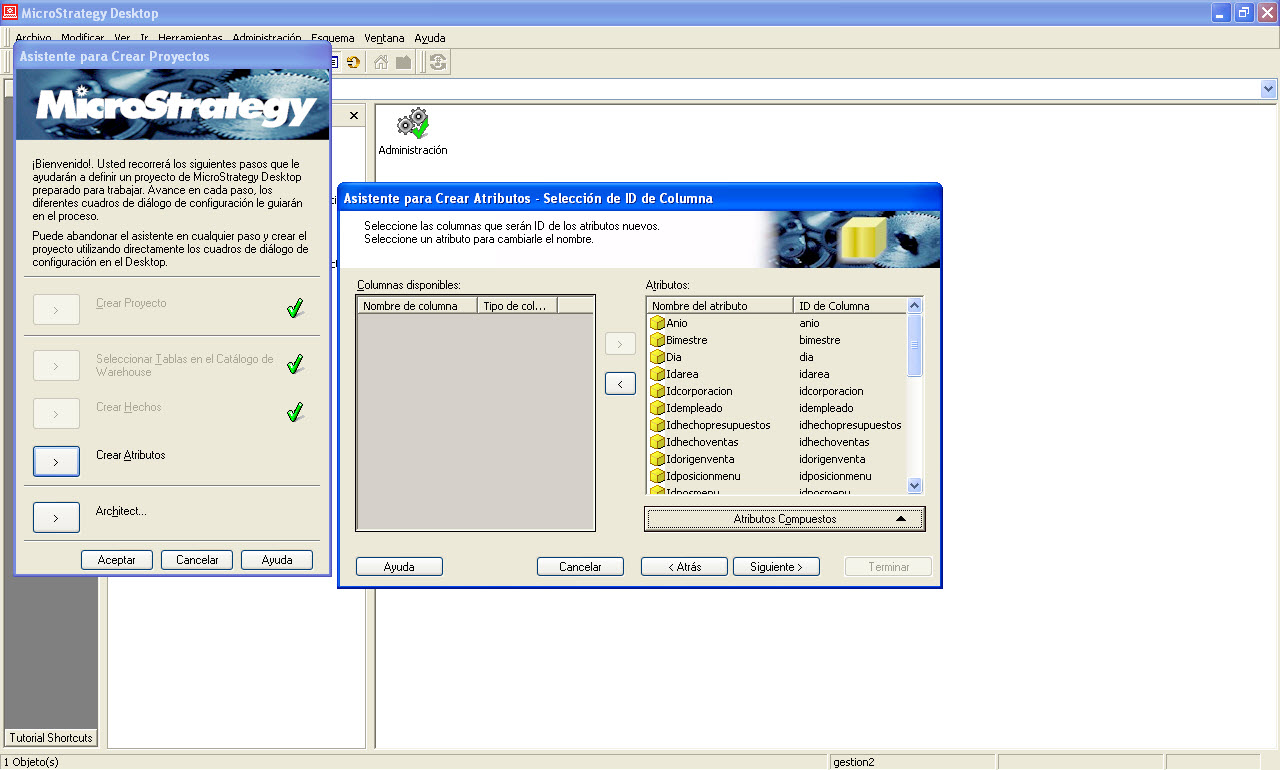


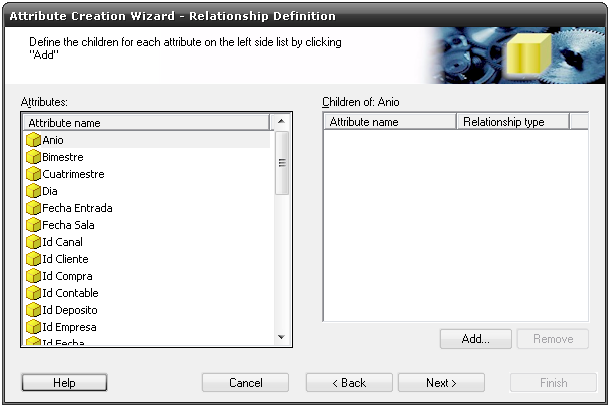
* Paso 12: Definimos el catálogo y los hechos para el DW.



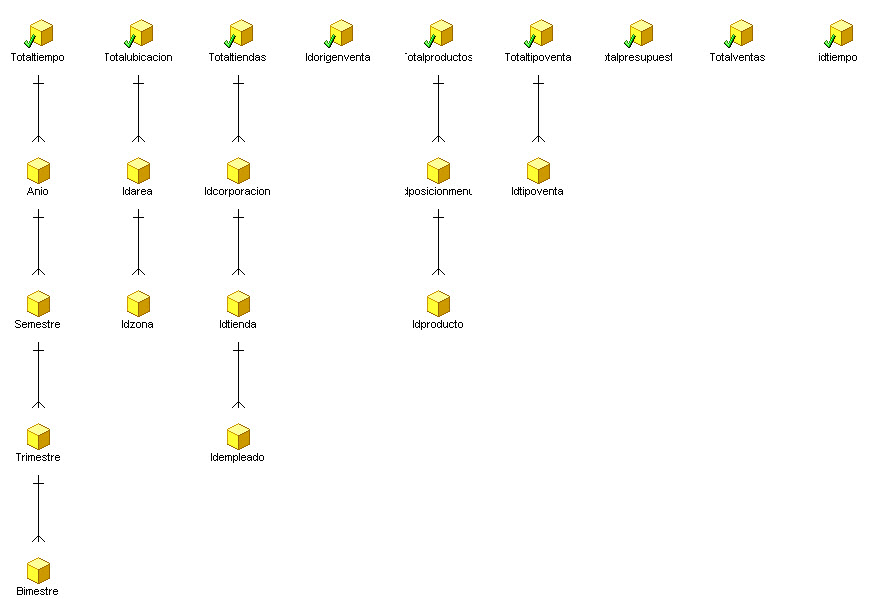


* Paso 13: Indicamos las Relaciones dentro de la Creación de atributos.

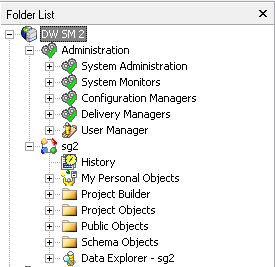




* Paso 14: Ingresamos al Architect de la aplicación y definimos la jerarquía de lo seleccionado previamente



* Paso 15: Una vez finalizado los pasos anteriores podremos navegar a través del proyecto.



Diseño ETL

Debido a que no se cuenta con un diseño previo de bases de datos o registros en donde se tiene la información necesaria para realizar los análisis solicitados hemos diseñado un ETL asumiendo origen y formato de datos estándar con los cuales se trabaja y se procesan para poder cargar el DW.

Con respecto a los datos con los que se han poblado las tablas de nuestra BD relacional es necesario aclarar los siguientes aspectos:

* Se asume que por una decisión de la alta gerencia al momento de comenzar con el negocio se realizaron cálculos para definir valores objetivos que era necesario alcanzar para cubrir los costos donde resultó lo siguiente:
  + Para cubrir los costos (en el año 2000) era necesario que las ventas de Taco Tabasco fuera de $13 y de KFC de $15.
  + Con el correr de los años estos valores objetivos han ido modificándose para estar a la altura de los incrementos de costos y precios de venta.
* Debido a la inflación a nivel mundial se han realizado los cálculos necesarios para incrementar los costos y precios de TODOS los productos ofrecidos entre un 10% y 20% anual.
* Debido a restricciones en el manejo de BD con gran cantidad de filas se ha restringido la cantidad de ventas diarias y de productos. Las mismas son de entre 10 y 30 por cada tienda en cada día y la venta no tiene mas5 productos incluidos.

|  |  |  |  |
| --- | --- | --- | --- |
| Tabla de Hechos | Atributo | Origen de datos | Origen del dato particular |
| Venta | Precio | Registro de ventas de la tienda | Precio unitario del producto \* cantidad vendida |
| Costo | Costo unitario del producto \* cantidad vendida |
| cantidad Vendida | cantidad vendida |
| Volumen de movimientos | precio | Registro de ventas de la tienda | Precio unitario del producto \* cantidad vendida |
| costo | Costo unitario del producto \* cantidad vendida |
| valorobjetivo | Políticas de ventas | Monto de la venta promedio esperada por la gerencia de la cadena |
|  |  |  |  |
| Tabla de Dimensiones | Atributo | Origen de datos | Origen del dato particular |
| Productos | nombreproducto | registro de productos disponibles | Nombre del producto |
| descposmenu | Registro de posiciones de menú | Nombre de la posición dentro del menú |
| Tiempo | Anio | Registro de ventas realizadas | Se descompone la fecha que se obtiene del registro en el formato aaaa/MM/dd hh:mm y se carga cada uno de los atributos según corresponda. |
| Semestre |
| Trimestre |
| bimestre |
| mes |
| semana |
| dia |
| turno8 |
| turno4 |
| turno2 |
| turno 15min |
| Tiendas | nombrecorporacion | Tabla de corporaciones | Dependiendo del ID de la fila se completa con el nombre de la corporación a la que pertenece. La información de correspondencia de Ids y nombres se encuentra en el ETL |
| nombretienda | Registro de sucursales | Nombre de la sucursal particular |
| descempleado | Registro de empleados | Nombre del empleado particular |
| tipoventa | nombretipoventa | registro de ventas | tipo de venta que fue realizada |
| ubicaciones | nombrezona | registro de tiendas | Se descompone el nombre completo de la sucursal en el formato Zona/ Area y se completan los atributos según corresponda |
| nombrearea |

Indicadores Utilizados

A continuación se expone un cuadro que resume los indicadores creados para que la herramienta “microstrategy” calcule los datos que se solicitan en los distintos reportes.

|  |  |
| --- | --- |
| Indicadores mas importantes | |
| Indicador | Descripción |
| Cantidad Vendida | Sumatoria de la cantidad vendida |
| Costo | sumatoria de costos según criterio |
| Incidencia Tiendas | porcentaje de participación de una tienda en el total de la corporación (Sum(Precio) {Idtienda} / Sum(Precio) {Idcorporacion} ) |
| Precio | Indica el monto de una venta |
| rentabilidad | porcentaje de ganancia que se obtiene en comparación con el costo del producto (Sum(Precio) {~} / Sum(Costo) {~} ) |
| Utilidad | Se calcula como el beneficio neto en dinero que se obtiene al vender un producto precio - costo |
| Rentabilidadvalorobjetivo | Rentabilidad comparando la totalidad de las ventas de un producto con la sumatoria de valores objetivo definidos por la gerencial ((Sum(Precio) {~} / Sum(Valorobjetivo) {~} ) - 1) |

ANEXO 1 – Codigo para cargar la BD.

import datetime

import random

import time

import mysql

import tiempo

import tiendas

import ubicacion

import productos

random.seed(time.time())

#detalleproductos = {1: (9,6.3), 2: (10,7), 3: (5,3.5), 4: (7,4.9), 5: (8,5.6), 6: (5,3.5), 7: (7,4.9), 8: (9,6.3), 9: (10,7), 10: (6,4.2), 11: (5,3.5), 12: (8,5.6), 13: (7,4.9), 14: (6,4.2), 15: (6,4.2), 16: (7,4.9), 17: (9,6.3), 18: (9,6.3), 19: (9,6.3), 20: (6,4.2), 21: (8,5.6), 22: (5,3.5), 23: (10,7), 24: (10,7), 25: (5,3.5), 26: (9,6.3), 27: (8,5.6), 28: (9,6.3), 29: (8,5.6), 30: (10,7), 31: (5,3.5), 32: (9,6.3), 33: (8,5.6), 34: (9,6.3), 35: (8,5.6), 36: (9,6.3), 37: (9,6.3), 38: (5,3.5), 39: (6,4.2), 40: (7,4.9), 41: (7,4.9), 42: (5,3.5), 43: (7,4.9), 44: (7,4.9), 45: (7,4.9), 46: (9,6.3), 47: (5,3.5), 48: (8,5.6), 49: (9,6.3), 50: (6,4.2), 51: (10,7), 52: (10,7), 53: (6,4.2), 54: (5,3.5), 55: (5,3.5), 56: (9,6.3), 57: (6,4.2), 58: (10,7), 59: (5,3.5), 60: (5,3.5), 61: (8,5.6), 62: (7,4.9), 63: (8,5.6), 64: (8,5.6), 65: (10,7), 66: (8,5.6), 67: (10,7), 68: (9,6.3), 69: (9,6.3), 70: (6,4.2), 71: (6,4.2), 72: (7,4.9), 73: (5,3.5), 74: (9,6.3), 75: (6,4.2), 76: (8,5.6), 77: (5,3.5), 78: (5,3.5), 79: (5,3.5), 80: (8,5.6), 81: (7,4.9), 82: (8,5.6), 83: (5,3.5), 84: (6,4.2), 85: (5,3.5), 86: (9,6.3), 87: (9,6.3), 88: (7,4.9), 89: (5,3.5), 90: (10,7), 91: (6,4.2), 92: (10,7), 93: (7,4.9), 94: (8,5.6), 95: (7,4.9), 96: (10,7)}

#valoresperadoTT = 14

#valoresperadoKFC = 17

def subirprecio(detalleproductos, valoresperadoTT, valoresperadoKFC):

cuanto = random.uniform(1.1, 1.2)

print cuanto

valoresperadoTT = valoresperadoTT \* cuanto

valoresperadoKFC = valoresperadoKFC \* cuanto

for producto in range(1, 96):

detalleproductos[producto] = (detalleproductos[producto][0] \* cuanto , detalleproductos[producto][1] \* cuanto )

print (detalleproductos, valoresperadoTT, valoresperadoKFC)

return (detalleproductos, valoresperadoTT, valoresperadoKFC)

def vender(hoy, hora, minuto, id\_origenventa, detalleproductos, valoresperadoTT, valoresperadoKFC):

#Inserto la fecha y hora en la tabla tiempo y obtengo el id insertado

id\_fecha = tiempo.fechadata(hoy, hora, minuto)

#Obtengo datos de la tabla tiendas. idCorporacion, idTienda, idEmpleado

id\_corporacion, id\_tienda, id\_empleado = tiendas.datosdetienda()

#Obtengo datos de la ubicacion

id\_area, id\_zona = ubicacion.datosdeubicacion(id\_tienda)

#Obtengo el tipo de venta

id\_tipoventa = random.randint(1, 2)

valoresperado = valoresperadoTT

if(id\_corporacion <> 1):

valoresperado = valoresperadoKFC

cantidaddeproductos = random.randint(1,3)

for prods in range(1, cantidaddeproductos):

#Obtengo los datos del producto vendido idPosicionMenu, idProducto

id\_posicionmenu, id\_producto = productos.datosdeproducto(id\_corporacion)

cantidadvendida = random.randint(1, 8)

precio = (detalleproductos[id\_producto][0]) \* cantidadvendida

costo = (detalleproductos[id\_producto][1]) \* cantidadvendida

#print (id\_producto, precio, costo, id\_corporacion, id\_tienda, id\_zona, id\_area, 1, cantidadvendida, id\_origenventa, id\_tipoventa, id\_posicionmenu, id\_empleado, id\_fecha)

#print (id\_producto, precio, costo, id\_corporacion, id\_tienda, id\_zona, id\_area, 1, cantidadvendida, id\_posicionmenu, valoresperado, id\_empleado, id\_fecha)

#insert into ventas

sql\_ventas = "INSERT INTO hecho\_ventas(idproducto, precio, costo, idcorporacion, idtienda, idzona, idarea, totalventas, cantidadvendida, idorigenventa, idtipoventa, idposicionmenu, idempleado, idtiempo) values (%i, %f, %f, %i, %i, %i, %i, %i, %i, %i, %i, %i, %i, '%s');" % (id\_producto, precio, costo, id\_corporacion, id\_tienda, id\_zona, id\_area, 1, cantidadvendida, id\_origenventa, id\_tipoventa, id\_posicionmenu, id\_empleado, id\_fecha)

#print sql\_ventas

mysql.insert(sql\_ventas)

#insert into presupuesto

sql\_presupuesto = "INSERT INTO hecho\_presupuestos(idproducto, precio, costo, idcorporacion, idtienda, idzona, idarea, totalpresupuesto, cantidadvendida, idposicionmenu, valorobjetivo, idempleado, idtiempo) Values (%i, %f, %f, %i, %i, %i, %i, %i, %i, %i, %f, %i, '%s');" % (id\_producto, precio, costo, id\_corporacion, id\_tienda, id\_zona, id\_area, 1, cantidadvendida, id\_posicionmenu, valoresperado, id\_empleado, id\_fecha)

#print sql\_presupuesto

mysql.insert(sql\_presupuesto)

def load():

hoy = datetime.date(2000,1,1)

un\_dia = datetime.timedelta(days=1)

anio = hoy.year

id\_origenventa = 1

detalleproductos = {1: (9,6.3), 2: (10,7), 3: (5,3.5), 4: (7,4.9), 5: (8,5.6), 6: (5,3.5), 7: (7,4.9), 8: (9,6.3), 9: (10,7), 10: (6,4.2), 11: (5,3.5), 12: (8,5.6), 13: (7,4.9), 14: (6,4.2), 15: (6,4.2), 16: (7,4.9), 17: (9,6.3), 18: (9,6.3), 19: (9,6.3), 20: (6,4.2), 21: (8,5.6), 22: (5,3.5), 23: (10,7), 24: (10,7), 25: (5,3.5), 26: (9,6.3), 27: (8,5.6), 28: (9,6.3), 29: (8,5.6), 30: (10,7), 31: (5,3.5), 32: (9,6.3), 33: (8,5.6), 34: (9,6.3), 35: (8,5.6), 36: (9,6.3), 37: (9,6.3), 38: (5,3.5), 39: (6,4.2), 40: (7,4.9), 41: (7,4.9), 42: (5,3.5), 43: (7,4.9), 44: (7,4.9), 45: (7,4.9), 46: (9,6.3), 47: (5,3.5), 48: (8,5.6), 49: (9,6.3), 50: (6,4.2), 51: (10,7), 52: (10,7), 53: (6,4.2), 54: (5,3.5), 55: (5,3.5), 56: (9,6.3), 57: (6,4.2), 58: (10,7), 59: (5,3.5), 60: (5,3.5), 61: (8,5.6), 62: (7,4.9), 63: (8,5.6), 64: (8,5.6), 65: (10,7), 66: (8,5.6), 67: (10,7), 68: (9,6.3), 69: (9,6.3), 70: (6,4.2), 71: (6,4.2), 72: (7,4.9), 73: (5,3.5), 74: (9,6.3), 75: (6,4.2), 76: (8,5.6), 77: (5,3.5), 78: (5,3.5), 79: (5,3.5), 80: (8,5.6), 81: (7,4.9), 82: (8,5.6), 83: (5,3.5), 84: (6,4.2), 85: (5,3.5), 86: (9,6.3), 87: (9,6.3), 88: (7,4.9), 89: (5,3.5), 90: (10,7), 91: (6,4.2), 92: (10,7), 93: (7,4.9), 94: (8,5.6), 95: (7,4.9), 96: (10,7)}

valoresperadoTT = 14

valoresperadoKFC = 17

#4230

for i in range(4230):

# print "dia: ",i

if( anio <> hoy.year ):

anio = hoy.year

detalleproductos, valoresperadoTT, valoresperadoKFC = subirprecio(detalleproductos, valoresperadoTT, valoresperadoKFC)

cantventas = random.randint(10,30)

for j in range(cantventas):

hora = random.randint(1,24)

minuto = random.randint(0,59)

vender(hoy, hora, minuto, id\_origenventa, detalleproductos, valoresperadoTT, valoresperadoKFC)

id\_origenventa = id\_origenventa + 1

hoy = hoy + un\_dia

print hoy

print "id\_origenventa=%i" % (id\_origenventa)

load()