MANUAL TECNICO

Amaretto Datawarehouse

Contenido

Datawarehouse – Scripts implementados en Talend Open Studio.

Creación del Datawarehouse.

Para crear el datawarehouse se debe de ejecutar el script *DW.sql.* Este script permite la creación de la base de datos AmarettoDW la cual alojará la información transformada y adecuada al modelo dimensional de nuestro datawarehouse.

DW.sql:

```
DROP DATABASE IF EXISTS AmarettoDW
go
CREATE DATABASE AmarettoDW
go
USE AmarettoDW

if exists (select 1
   from sys.sysreferences r join sys.sysobjects o on (o.id = r.constid
and o.type = 'F')
   where r.fkeyid = object_id('FACTTRANSACCIONES') and o.name =
'FK_FACTTRAN_RELATIONS_DIMPRODU')
alter table FACTTRANSACCIONES
   drop constraint FK_FACTTRAN_RELATIONS_DIMPRODU
go

if exists (select 1
   from sys.sysreferences r join sys.sysobjects o on (o.id = r.constid
and o.type = 'F')
   where r.fkeyid = object_id('FACTTRANSACCIONES') and o.name =
'FK_FACTTRAN_RELATIONS_DIMUBICA')
alter table FACTTRANSACCIONES
   drop constraint FK_FACTTRAN_RELATIONS_DIMUBICA
go

if exists (select 1
   from sys.sysreferences r join sys.sysobjects o on (o.id = r.constid
and o.type = 'F')
   where r.fkeyid = object_id('FACTTRANSACCIONES') and o.name =
'FK_FACTTRAN_RELATIONS_DIMCLIEN')
alter table FACTTRANSACCIONES
   drop constraint FK_FACTTRAN_RELATIONS_DIMCLIEN
go

if exists (select 1
   from sys.sysreferences r join sys.sysobjects o on (o.id = r.constid
and o.type = 'F')
   where r.fkeyid = object_id('FACTTRANSACCIONES') and o.name =
'FK_FACTTRAN_RELATIONS_DIMTIEMP')
alter table FACTTRANSACCIONES
```

```
   drop constraint FK_FACTTRAN_RELATIONS_DIMTIEMP
go

if exists (select 1
            from  sysobjects
           where  id = object_id('DIMCLIENTE')
            and   type = 'U')
   drop table DIMCLIENTE
go

if exists (select 1
            from  sysobjects
           where  id = object_id('DIMPRODUCTO')
            and   type = 'U')
   drop table DIMPRODUCTO
go

if exists (select 1
            from  sysobjects
           where  id = object_id('DIMTIEMPO')
            and   type = 'U')
   drop table DIMTIEMPO
go

if exists (select 1
            from  sysobjects
           where  id = object_id('DIMUBICACION')
            and   type = 'U')
   drop table DIMUBICACION
go

if exists (select 1
            from  sysobjects
           where  id = object_id('FACTTRANSACCIONES')
            and   type = 'U')
   drop table FACTTRANSACCIONES
go

/*==============================================================*/
/* Table: DIMCLIENTE                                            */
/*==============================================================*/
create table DIMCLIENTE (
   IDCLIENTE            int identity(1,1)    not null,
   IDCLIENTEN           int                  not null,
   NOMBRE               varchar(250)         not null,
   DIRECCION            varchar(300)         not null,
   CIUDAD               varchar(300)         not null,
   PAIS                 varchar(100)         not null,
   EMAIL                varchar(250)         not null,
   FECHADENACIMIENTO    datetime             not null,
   GENERO               varchar(10)          not null,
   constraint PK_DIMCLIENTE primary key (IDCLIENTE)
)
```

```
go

/*==============================================================*/
/* Table: DIMPRODUCTO                                           */
/*==============================================================*/
create table DIMPRODUCTO (
   IDPRODUCTO           int identity(1,1)    not null,
   IDPRODUCTON          int                  not null,
   NOMBRE               varchar(250)         not null,
   CATEGORIA            varchar(250)         not null,
   DESCRIPCION          varchar(500)         not null,
   PRECIOUNITARIO       decimal(5,2)         not null,
   STOCK                int                  not null,
   constraint PK_DIMPRODUCTO primary key (IDPRODUCTO)
)
go

/*==============================================================*/
/* Table: DIMTIEMPO                                             */
/*==============================================================*/
create table DIMTIEMPO (
   IDTIEMPO             int    not null,
   FECHACOMPLETA        datetime             not null,
   DIADELASEMANA        int                  not null,
   NUMERODEDIADELMES    int                  not null,
   NUMERODEDIAENGENERAL int                  not null,
   NOMBREDELDIA         varchar(15)          not null,
   NOMBREDELDIAABREVIADO varchar(3)           not null,
   BANDERDIALUNESAVIERNRES varchar(25)          not null,
   NUMEROSEMANAENELANO  int                  not null,
   NUMERODESEMANAENGENERAL int                  not null,
   FECHADEINICIODESEMANA datetime              not null,
   CLAVEFECHAINICIODESEMANA int                  not null,
   NUMERODEMES          int                  not null,
   NUMERODEMESENGENERAL int                  not null,
   NOMBREDELMES         varchar(15)          not null,
   NOMBREDELMESABREVIADO varchar(3)           not null,
   CUARTO               int                  not null,
   NUMERODEANO          int                  not null,
   ANOMES               int                  not null,
   MESFISCAL            int                  not null,
   CUARTOFISCAL         int                  not null,
   ANOFISCAL            int                  not null,
   BANDERAFINDEMES      varchar(50)          not null,
   FECHADEMISMODIAHACEUNANO datetime              not null,
   constraint PK_DIMTIEMPO primary key (IDTIEMPO)
)
go

/*==============================================================*/
/* Table: DIMUBICACION                                          */
/*==============================================================*/
create table DIMUBICACION (
   IDUBICACION          int identity(1,1)    not null,
   IDUBICACIONN         int                  not null,
   PAIS                 varchar(100)         not null,
   CIUDAD               varchar(300)         not null,
   DIRECCION            varchar(300)         not null,
   MONEDA               varchar(200)         not null,
   constraint PK_DIMUBICACION primary key (IDUBICACION)
)
go

/*==============================================================*/
/* Table: FACTTRANSACCIONES                                     */
/*==============================================================*/
create table FACTTRANSACCIONES (
   IDPRODUCTO           int                  not null,
   IDUBICACION          int                  not null,
   IDCLIENTE            int                  not null,
   IDTIEMPO             int                  not null,
   TIPOTRANSACCION      varchar(50)          not null,
   CANTIDAD             int                  not null,
   PRECIOUNITARIO       decimal(5,2)         not null,
   TOTALTRANSACCION     decimal(8,2)         not null,
   COSTOTRASLADO        decimal(8,2)         not null,
   COSTOALMACEN         decimal(8,2)         not null,
   COSTOINVENTARIO      decimal(8,2)         not null,
   TIEMPOENTREGADIAS    int                  not null,
   DESCUENTO            decimal(8,2)         not null,
   COSTOTOTALTRANSACCION decimal(10,2)       not null,
   constraint PK_FACTTRANSACCIONES primary key (IDPRODUCTO, IDUBICACION,
IDCLIENTE, IDTIEMPO)
)
go

alter table FACTTRANSACCIONES
   add constraint FK_FACTTRAN_RELATIONS_DIMPRODU foreign key (IDPRODUCTO)
      references DIMPRODUCTO (IDPRODUCTO)
go

alter table FACTTRANSACCIONES
   add constraint FK_FACTTRAN_RELATIONS_DIMUBICA foreign key
(IDUBICACION)
      references DIMUBICACION (IDUBICACION)
go

alter table FACTTRANSACCIONES
   add constraint FK_FACTTRAN_RELATIONS_DIMCLIEN foreign key (IDCLIENTE)
      references DIMCLIENTE (IDCLIENTE)
go

alter table FACTTRANSACCIONES
   add constraint FK_FACTTRAN_RELATIONS_DIMTIEMP foreign key (IDTIEMPO)
      references DIMTIEMPO (IDTIEMPO)
go
```

Inserción de valores no aplicables.

Luego de creada la base de datos AmarettoDW que corresponde a nuestro Datawarehouse, se debe ejecutar el archivo *Insert-ValoresNoAplicablesDimensiones.sql*. Que corresponde a una serie de consultas que sirven para insertar un registro genérico con id = -1; que servirá para identificar a aquellos registros en los cuales no se posee un valor de parte de la base de datos transaccional. Esto ocurre en las transacciones de compra, ya que el sistema Odoo no almacena los datos de cliente, producto y ubicación. Por ello este script afecta las dimensiones DimCliente, DimProducto y DimUbicacion.

Script *Insert-ValoresNoAplicablesDimensiones.sql:*

```
SET IDENTITY_INSERT DIMCLIENTE ON
INSERT INTO DIMCLIENTE
(IDCLIENTE,IDCLIENTEN,NOMBRE,DIRECCION,CIUDAD,PAIS,EMAIL,FECHADENACIMIENT
O,GENERO) VALUES(-1,-1,'No aplica','No aplica','No aplica','No
aplica','No aplica',GETDATE(),'No aplica')
SET IDENTITY_INSERT DIMCLIENTE OFF

SET IDENTITY_INSERT DIMPRODUCTO ON
INSERT INTO DIMPRODUCTO
(IDPRODUCTO,IDPRODUCTON,NOMBRE,CATEGORIA,DESCRIPCION,PRECIOUNITARIO,STOCK
) VALUES(-1,-1,'No aplica','No aplica','No aplica',0.00,0)
SET IDENTITY_INSERT DIMPRODUCTO OFF

SET IDENTITY_INSERT DIMUBICACION ON
INSERT INTO DIMUBICACION
(IDUBICACION,IDUBICACIONN,PAIS,CIUDAD,DIRECCION,MONEDA) VALUES(-1,-1,'No
aplica','No aplica','No aplica','No aplica')
SET IDENTITY_INSERT DIMUBICACION OFF
```

Llenado de DimCliente.

Para extraer los datos necesarios para su transformación y posterior carga en la dimensión DimCliente se utiliza una consulta sql en la configuración del proceso ETL en Talend Open Studio.

Script de *DimCliente.sql*:

```sql
SELECT DISTINCT c.entity_id AS 'IdClienteN',
CONCAT_WS(' ', c.firstname, c.middlename, c.lastname) AS Nombre,
CONCAT_WS(', ', a.street, a.region) AS 'Direccion',
a.city AS 'Ciudad',
CASE
    WHEN a.country_id = 'BZ' THEN 'Belice'
    WHEN a.country_id = 'CR' THEN 'Costa Rica'
    WHEN a.country_id = 'SV' THEN 'El Salvador'
    WHEN a.country_id = 'GT' THEN 'Guatemala'
    WHEN a.country_id = 'HN' THEN 'Honduras'
    WHEN a.country_id = 'NI' THEN 'Nicaragua'
    WHEN a.country_id = 'PA' THEN 'Panama'
    WHEN a.country_id = 'US' THEN 'Estados Unidos'
    ELSE 'Sin Especificar'
END AS 'Pais',
c.email AS 'email',
COALESCE(c.dob, 'Sin especificar') AS 'FechaDeNacimiento',
CASE
    WHEN c.gender = 1 THEN 'Masculino'
    WHEN c.gender = 2 THEN 'Femenino'
    ELSE 'No definido'
END AS 'Genero'
FROM customer_entity AS c
INNER JOIN sales_order AS s ON s.customer_id = c.entity_id
INNER JOIN customer_address_entity AS a ON a.entity_id =
c.default_billing
WHERE s.status NOT IN ('canceled')
```

Llenado de DimProducto.

Para extraer los datos necesarios para su transformación y posterior carga en la dimensión DimProducto se utiliza una consulta sql en la configuración del proceso ETL en Talend Open Studio.

Script de *DimProducto.sql*:

```sql
SELECT
    DISTINCT(e.entity_id) as Idproducton,
    n.value AS Nombre,
    GROUP_CONCAT(DISTINCT cv.value SEPARATOR ', ') as Categoria,
    COALESCE(REPLACE(REPLACE(d.value, '<p>', ''), '</p>', ''),'Sin
especificar') AS Descripcion,
    p.value as Preciounitario,
    si.qty as Stock
FROM
    catalog_product_entity AS e
LEFT JOIN
    catalog_product_entity_varchar AS n
    ON e.entity_id = n.entity_id
    AND n.attribute_id = 73
LEFT JOIN
    catalog_product_entity_text AS d
    ON e.entity_id = d.entity_id
    AND d.attribute_id = 75
INNER JOIN `catalog_product_entity_decimal` as p on e.entity_id =
p.entity_id
LEFT JOIN
    catalog_category_product AS cp
    ON e.entity_id = cp.product_id
LEFT JOIN
    catalog_category_entity_varchar AS cv
    ON cp.category_id = cv.entity_id
    AND cv.attribute_id = 45
INNER JOIN `cataloginventory_stock_item` as si on e.entity_id =
si.product_id
GROUP BY
    e.entity_id
```

Llenado de DimUbicacion.

Para extraer los datos necesarios para su transformación y posterior carga en la dimensión DimUbicacion se utiliza una consulta sql en la configuración del proceso ETL en Talend Open Studio.

Script de *DimUbicacion.sql*:

```
SELECT sor.entity_id as 'IdUbicacionN',
CASE sor.country_id
WHEN 'BZ' THEN 'Belice'
WHEN 'CR' THEN 'Costa Rica'
WHEN 'SV' THEN 'El Salvador'
WHEN 'GT' THEN 'Guatemala'
WHEN 'HN' THEN 'Honduras'
WHEN 'NI' THEN 'Nicaragua'
WHEN 'PA' THEN 'Panama'
WHEN 'MX' THEN 'Mexico'
ELSE 'Sin especificar'
END AS 'Pais',
sor.city as 'Ciudad',
CONCAT_WS(', ', sor.street, sor.region) as 'Direccion',
CASE so.order_currency_code
WHEN 'BZD' THEN 'Dolar beliceño'
WHEN 'CRC' THEN 'Colon costarricense'
WHEN 'USD' THEN 'Dolar estadounidense'
WHEN 'GTQ' THEN 'Quetzal'
WHEN 'HNL' THEN 'Lempira hondureño'
WHEN 'NIC' THEN 'Cordoba nicaraguense'
WHEN 'PAB' THEN 'Balboa'
WHEN 'MXN' THEN 'Peso mexicano'
ELSE 'Sin especificar'
END AS 'Moneda'
from sales_order_address sor inner join sales_order so on so.entity_id =
sor.parent_id where sor.address_type = 'shipping'
```

Llenado de FactTransacciones.

Para extraer los datos necesarios para su transformación y posterior carga en la tabla de hechos FactTransacciones se utilizan dos consultas sql en la configuración del proceso ETL en Talend Open Studio, una consulta para las transacciones de compra y otra para las de venta.

Script de compras *FactTransacciones–compras.sql*:

```
SELECT ci.magento_product_id as IdProducto, -1 as IdUbicacion, -1 as
IdCliente, ci.date as  IdTiempo, 'Compra' as TipoTransaccion,
ci.stock_added as Cantidad,
(SELECT (ROUND(CAST((SELECT value_float FROM ir_property WHERE
name='standard_price' and SPLIT_PART(res_id, ',', 2)::INTEGER =
ci.product_id) AS NUMERIC),2))) as PrecioUnitario,
(SELECT (ROUND(CAST((ci.stock_added) * (SELECT value_float FROM
ir_property WHERE name='standard_price' and SPLIT_PART(res_id, ',',
2)::INTEGER = ci.product_id) AS NUMERIC),2))) as TotalTransaccion,
CASE
    WHEN ci.stock_added BETWEEN 0 AND 15 THEN 1
    WHEN ci.stock_added BETWEEN 16 AND 30 THEN 3
    WHEN ci.stock_added BETWEEN 31 AND 50 THEN 5
    WHEN ci.stock_added BETWEEN 51 AND 100 THEN 7
    WHEN ci.stock_added > 100 THEN 10
END as CostoTraslado,
CASE
    WHEN ci.stock_added BETWEEN 0 AND 10 THEN 5
    WHEN ci.stock_added BETWEEN 11 AND 20 THEN 10
    WHEN ci.stock_added BETWEEN 21 AND 50 THEN 15
    WHEN ci.stock_added BETWEEN 51 AND 100 THEN 25
    WHEN ci.stock_added > 100 THEN 35
END as CostoAlmacen,
(CASE --Costo traslado
    WHEN ci.stock_added BETWEEN 0 AND 15 THEN 1
    WHEN ci.stock_added BETWEEN 16 AND 30 THEN 3
    WHEN ci.stock_added BETWEEN 31 AND 50 THEN 5
    WHEN ci.stock_added BETWEEN 51 AND 100 THEN 7
    WHEN ci.stock_added > 100 THEN 10
```

```sql
END +
CASE --Costo almacen
    WHEN ci.stock_added BETWEEN 0 AND 10 THEN 5
    WHEN ci.stock_added BETWEEN 11 AND 20 THEN 10
    WHEN ci.stock_added BETWEEN 21 AND 50 THEN 15
    WHEN ci.stock_added BETWEEN 51 AND 100 THEN 25
    WHEN ci.stock_added > 100 THEN 35
    END) as CostoInventario,
30 AS TiempoEntregaDias,
CASE
    WHEN ci.stock_added BETWEEN 0 AND 15 THEN 0.15
    WHEN ci.stock_added BETWEEN 16 AND 30 THEN 0.25
    WHEN ci.stock_added BETWEEN 31 AND 50 THEN 0.35
    WHEN ci.stock_added BETWEEN 51 AND 100 THEN 0.50
    WHEN ci.stock_added > 100 THEN 0.60
END as Descuento,
(CASE --cantidad * (Precio unitario - precio unitario*descuento)
    WHEN ci.stock_added BETWEEN 0 AND 15 THEN (ci.stock_added) * ((SELECT
ROUND(CAST((SELECT value_float FROM ir_property WHERE
name='standard_price' and SPLIT_PART(res_id, ',', 2)::INTEGER =
ci.product_id) AS NUMERIC),2)) - (SELECT ROUND(CAST((SELECT value_float
FROM ir_property WHERE name='standard_price' and SPLIT_PART(res_id, ',',
2)::INTEGER = ci.product_id) AS NUMERIC),2)) * 0.15)
    WHEN ci.stock_added BETWEEN 16 AND 30 THEN (ci.stock_added) *
((SELECT ROUND(CAST((SELECT value_float FROM ir_property WHERE
name='standard_price' and SPLIT_PART(res_id, ',', 2)::INTEGER =
ci.product_id) AS NUMERIC),2)) - (SELECT ROUND(CAST((SELECT value_float
FROM ir_property WHERE name='standard_price' and SPLIT_PART(res_id, ',',
2)::INTEGER = ci.product_id) AS NUMERIC),2)) * 0.25)
    WHEN ci.stock_added BETWEEN 31 AND 50 THEN (ci.stock_added) *
((SELECT ROUND(CAST((SELECT value_float FROM ir_property WHERE
name='standard_price' and SPLIT_PART(res_id, ',', 2)::INTEGER =
ci.product_id) AS NUMERIC),2)) - (SELECT ROUND(CAST((SELECT value_float
FROM ir_property WHERE name='standard_price' and SPLIT_PART(res_id, ',',
2)::INTEGER = ci.product_id) AS NUMERIC),2)) * 0.35)
    WHEN ci.stock_added BETWEEN 51 AND 100 THEN (ci.stock_added) *
((SELECT ROUND(CAST((SELECT value_float FROM ir_property WHERE
name='standard_price' and SPLIT_PART(res_id, ',', 2)::INTEGER =
ci.product_id) AS NUMERIC),2)) - (SELECT ROUND(CAST((SELECT value_float
FROM ir_property WHERE name='standard_price' and SPLIT_PART(res_id, ',',
2)::INTEGER = ci.product_id) AS NUMERIC),2)) * 0.50)
    WHEN ci.stock_added > 100 THEN (ci.stock_added) * ((SELECT
ROUND(CAST((SELECT value_float FROM ir_property WHERE
name='standard_price' and SPLIT_PART(res_id, ',', 2)::INTEGER =
ci.product_id) AS NUMERIC),2)) - (SELECT ROUND(CAST((SELECT value_float
FROM ir_property WHERE name='standard_price' and SPLIT_PART(res_id, ',',
2)::INTEGER = ci.product_id) AS NUMERIC),2)) * 0.60)
END +
CASE --Costo traslado
    WHEN ci.stock_added BETWEEN 0 AND 15 THEN 1
    WHEN ci.stock_added BETWEEN 16 AND 30 THEN 3
    WHEN ci.stock_added BETWEEN 31 AND 50 THEN 5
    WHEN ci.stock_added BETWEEN 51 AND 100 THEN 7
    WHEN ci.stock_added > 100 THEN 10
END +
CASE --Costo almacen
    WHEN ci.stock_added BETWEEN 0 AND 10 THEN 5
    WHEN ci.stock_added BETWEEN 11 AND 20 THEN 10
    WHEN ci.stock_added BETWEEN 21 AND 50 THEN 15
    WHEN ci.stock_added BETWEEN 51 AND 100 THEN 25
    WHEN ci.stock_added > 100 THEN 35
END) AS CostoTotalTransaccion
FROM compra_inventario ci
```

Script de ventas *FactTransacciones.sql*:

```
SELECT e.entity_id as 'IdProducto', sor.entity_id as 'IdUbicacion',
c.entity_id as 'IdCliente', DATE_FORMAT(so.created_at, '%Y%m%d') as
'IdTiempo', 'Venta' as 'TipoTransaccion', soi.qty_ordered as 'Cantidad',
cped.value as 'PrecioUnitario', ((soi.qty_ordered * cped.value) -
soi.discount_amount) as 'TotalTransaccion',
CASE so.total_qty_ordered
    WHEN 1 THEN ABS(so.base_shipping_amount - 1)
    ELSE ABS(((SELECT value FROM core_config_data WHERE path =
'carriers/flatrate/price')*soi.qty_ordered))
END as 'CostoTraslado',
1/(SELECT COUNT(*) FROM sales_order_item soi WHERE soi.order_id =
so.entity_id) as 'CostoAlmacen',
(1/(SELECT COUNT(*) FROM sales_order_item soi WHERE soi.order_id =
so.entity_id) +
CASE so.total_qty_ordered
    WHEN 1 THEN ABS(so.base_shipping_amount - 1)
    ELSE ABS(((SELECT value FROM core_config_data WHERE path =
'carriers/flatrate/price')*soi.qty_ordered))
END) as 'CostoInventario',
CASE sor.country_id
    WHEN 'BZ' THEN 15
    WHEN 'CR' THEN 7
    WHEN 'SV' THEN 2
    WHEN 'GT' THEN 10
    WHEN 'HN' THEN 12
    WHEN 'NI' THEN 14
    WHEN 'PA' THEN 20
    WHEN 'MX' THEN 25
    ELSE 300000
END AS 'TiempoEntregaDias',
soi.discount_amount as 'Descuento',
((soi.qty_ordered * cped.value) - soi.discount_amount) +
    (1/(SELECT COUNT(*) FROM sales_order_item soi WHERE soi.order_id =
so.entity_id) +
        CASE so.total_qty_ordered
        WHEN 1 THEN ABS(so.base_shipping_amount - 1)
        ELSE ABS(((SELECT value FROM core_config_data WHERE path =
'carriers/flatrate/price')*soi.qty_ordered))
        END) AS 'CostoTotalTransaccion'
```

```
FROM catalog_product_entity e
INNER JOIN sales_order_item soi on e.entity_id = soi.product_id
INNER JOIN sales_order so ON so.entity_id = soi.order_id
INNER JOIN sales_order_address sor ON so.entity_id = sor.parent_id
INNER JOIN customer_entity c on so.customer_id = c.entity_id
INNER JOIN catalog_product_entity_decimal as cped on e.entity_id =
cped.entity_id
WHERE sor.address_type = 'shipping'
```

Llenado de dimensión Tiempo

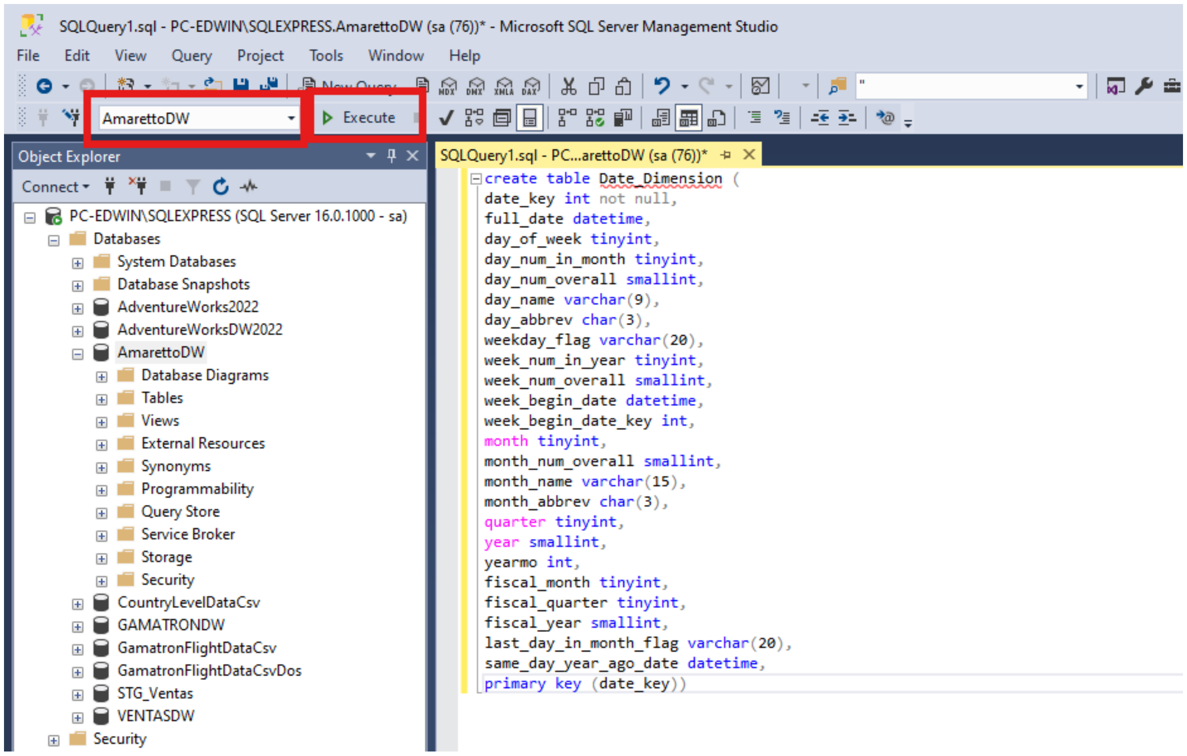
El llenado de la dimensión de tiempo se hace una sola vez y se lleva a cabo de una manera diferente.

Primero hay que crear una tabla auxiliar Data_Dimension, en la base de datos de nuestro datawarehouse, es decir en AmarettoDW. Esta tabla auxiliar nos servirá para insertar data de fechas.

- Ejecutar la consulta del script *Query Crear Tabla Auxiliar Data_Dimension.sql* en SQL Server Management Studio conectado a la base AmarettoDW

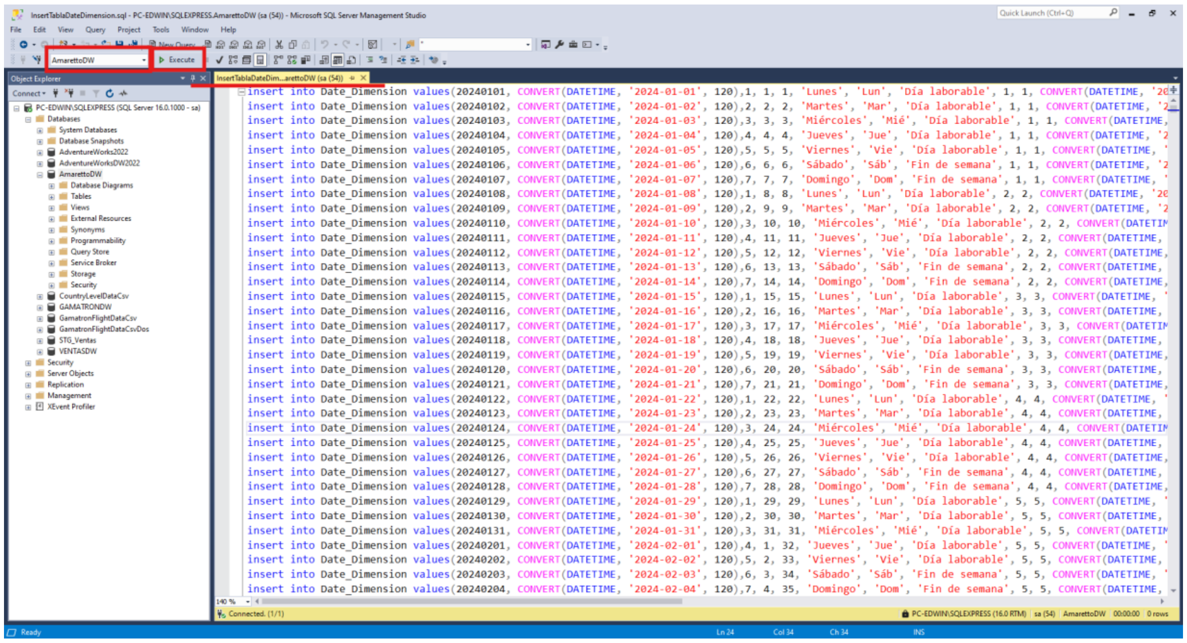
Script de Data_Dimension:

```
create table Date_Dimension (
date_key int not null,
full_date datetime,
day_of_week tinyint,
day_num_in_month tinyint,
day_num_overall smallint,
day_name varchar(9),
day_abbrev char(3),
weekday_flag varchar(20),
week_num_in_year tinyint,
week_num_overall smallint,
week_begin_date datetime,
week_begin_date_key int,
month tinyint,
month_num_overall smallint,
month_name varchar(15),
month_abbrev char(3),
quarter tinyint,
year smallint,
yearmo int,
fiscal_month tinyint,
fiscal_quarter tinyint,
fiscal_year smallint,
last_day_in_month_flag varchar(20),
same_day_year_ago_date datetime,
primary key (date_key))
```



```sql
create table Date_Dimension (
date_key int not null,
full_date datetime,
day_of_week tinyint,
day_num_in_month tinyint,
day_num_overall smallint,
day_name varchar(9),
day_abbrev char(3),
weekday_flag varchar(20),
week_num_in_year tinyint,
week_num_overall smallint,
week_begin_date datetime,
week_begin_date_key int,
month tinyint,
month_num_overall smallint,
month_name varchar(15),
month_abbrev char(3),
quarter tinyint,
year smallint,
yearmo int,
fiscal_month tinyint,
fiscal_quarter tinyint,
fiscal_year smallint,
last_day_in_month_flag varchar(20),
same_day_year_ago_date datetime,
primary key (date_key))
```

Luego de esto se deben insertar registros de fechas a la tabla auxiliar recién creada. Para ello se puede ejecutar el archivo *InsertTablaDateDimension.sql* en SQL Server Management Studio. Estos registros corresponden a 5 años de fechas.



```sql
insert into Date_Dimension values(20240101, CONVERT(DATETIME, '2024-01-01', 120),1, 1, 1, 'Lunes', 'Lun', 'Día laborable', 1, 1, CONVERT(DATETIME, '20
insert into Date_Dimension values(20240102, CONVERT(DATETIME, '2024-01-02', 120),2, 2, 2, 'Martes', 'Mar', 'Día laborable', 1, 1, CONVERT(DATETIME, '2
insert into Date_Dimension values(20240103, CONVERT(DATETIME, '2024-01-03', 120),3, 3, 3, 'Miércoles', 'Mié', 'Día laborable', 1, 1, CONVERT(DATETIME,
insert into Date_Dimension values(20240104, CONVERT(DATETIME, '2024-01-04', 120),4, 4, 4, 'Jueves', 'Jue', 'Día laborable', 1, 1, CONVERT(DATETIME, '2
insert into Date_Dimension values(20240105, CONVERT(DATETIME, '2024-01-05', 120),5, 5, 5, 'Viernes', 'Vie', 'Día laborable', 1, 1, CONVERT(DATETIME, '
insert into Date_Dimension values(20240106, CONVERT(DATETIME, '2024-01-06', 120),6, 6, 6, 'Sábado', 'Sáb', 'Fin de semana', 1, 1, CONVERT(DATETIME, '2
insert into Date_Dimension values(20240107, CONVERT(DATETIME, '2024-01-07', 120),7, 7, 7, 'Domingo', 'Dom', 'Fin de semana', 1, 1, CONVERT(DATETIME, '
insert into Date_Dimension values(20240108, CONVERT(DATETIME, '2024-01-08', 120),1, 8, 8, 'Lunes', 'Lun', 'Día laborable', 2, 2, CONVERT(DATETIME, '20
insert into Date_Dimension values(20240109, CONVERT(DATETIME, '2024-01-09', 120),2, 9, 9, 'Martes', 'Mar', 'Día laborable', 2, 2, CONVERT(DATETIME, '2
insert into Date_Dimension values(20240110, CONVERT(DATETIME, '2024-01-10', 120),3, 10, 10, 'Miércoles', 'Mié', 'Día laborable', 2, 2, CONVERT(DATETIM
insert into Date_Dimension values(20240111, CONVERT(DATETIME, '2024-01-11', 120),4, 11, 11, 'Jueves', 'Jue', 'Día laborable', 2, 2, CONVERT(DATETIME,
insert into Date_Dimension values(20240112, CONVERT(DATETIME, '2024-01-12', 120),5, 12, 12, 'Viernes', 'Vie', 'Día laborable', 2, 2, CONVERT(DATETIME,
insert into Date_Dimension values(20240113, CONVERT(DATETIME, '2024-01-13', 120),6, 13, 13, 'Sábado', 'Sáb', 'Fin de semana', 2, 2, CONVERT(DATETIME,
insert into Date_Dimension values(20240114, CONVERT(DATETIME, '2024-01-14', 120),7, 14, 14, 'Domingo', 'Dom', 'Fin de semana', 2, 2, CONVERT(DATETIME,
insert into Date_Dimension values(20240115, CONVERT(DATETIME, '2024-01-15', 120),1, 15, 15, 'Lunes', 'Lun', 'Día laborable', 3, 3, CONVERT(DATETIME, '
insert into Date_Dimension values(20240116, CONVERT(DATETIME, '2024-01-16', 120),2, 16, 16, 'Martes', 'Mar', 'Día laborable', 3, 3, CONVERT(DATETIME,
insert into Date_Dimension values(20240117, CONVERT(DATETIME, '2024-01-17', 120),3, 17, 17, 'Miércoles', 'Mié', 'Día laborable', 3, 3, CONVERT(DATETIM
insert into Date_Dimension values(20240118, CONVERT(DATETIME, '2024-01-18', 120),4, 18, 18, 'Jueves', 'Jue', 'Día laborable', 3, 3, CONVERT(DATETIME,
insert into Date_Dimension values(20240119, CONVERT(DATETIME, '2024-01-19', 120),5, 19, 19, 'Viernes', 'Vie', 'Día laborable', 3, 3, CONVERT(DATETIME,
insert into Date_Dimension values(20240120, CONVERT(DATETIME, '2024-01-20', 120),6, 20, 20, 'Sábado', 'Sáb', 'Fin de semana', 3, 3, CONVERT(DATETIME,
insert into Date_Dimension values(20240121, CONVERT(DATETIME, '2024-01-21', 120),7, 21, 21, 'Domingo', 'Dom', 'Fin de semana', 3, 3, CONVERT(DATETIME,
insert into Date_Dimension values(20240122, CONVERT(DATETIME, '2024-01-22', 120),1, 22, 22, 'Lunes', 'Lun', 'Día laborable', 4, 4, CONVERT(DATETIME, '
insert into Date_Dimension values(20240123, CONVERT(DATETIME, '2024-01-23', 120),2, 23, 23, 'Martes', 'Mar', 'Día laborable', 4, 4, CONVERT(DATETIME,
insert into Date_Dimension values(20240124, CONVERT(DATETIME, '2024-01-24', 120),3, 24, 24, 'Miércoles', 'Mié', 'Día laborable', 4, 4, CONVERT(DATETIM
insert into Date_Dimension values(20240125, CONVERT(DATETIME, '2024-01-25', 120),4, 25, 25, 'Jueves', 'Jue', 'Día laborable', 4, 4, CONVERT(DATETIME,
insert into Date_Dimension values(20240126, CONVERT(DATETIME, '2024-01-26', 120),5, 26, 26, 'Viernes', 'Vie', 'Día laborable', 4, 4, CONVERT(DATETIME,
insert into Date_Dimension values(20240127, CONVERT(DATETIME, '2024-01-27', 120),6, 27, 27, 'Sábado', 'Sáb', 'Fin de semana', 4, 4, CONVERT(DATETIME,
insert into Date_Dimension values(20240128, CONVERT(DATETIME, '2024-01-28', 120),7, 28, 28, 'Domingo', 'Dom', 'Fin de semana', 4, 4, CONVERT(DATETIME,
insert into Date_Dimension values(20240129, CONVERT(DATETIME, '2024-01-29', 120),1, 29, 29, 'Lunes', 'Lun', 'Día laborable', 5, 5, CONVERT(DATETIME, '
insert into Date_Dimension values(20240130, CONVERT(DATETIME, '2024-01-30', 120),2, 30, 30, 'Martes', 'Mar', 'Día laborable', 5, 5, CONVERT(DATETIME,
insert into Date_Dimension values(20240131, CONVERT(DATETIME, '2024-01-31', 120),3, 31, 31, 'Miércoles', 'Mié', 'Día laborable', 5, 5, CONVERT(DATETIM
insert into Date_Dimension values(20240201, CONVERT(DATETIME, '2024-02-01', 120),4, 1, 32, 'Jueves', 'Jue', 'Día laborable', 5, 5, CONVERT(DATETIME, '
insert into Date_Dimension values(20240202, CONVERT(DATETIME, '2024-02-02', 120),5, 2, 33, 'Viernes', 'Vie', 'Día laborable', 5, 5, CONVERT(DATETIME,
insert into Date_Dimension values(20240203, CONVERT(DATETIME, '2024-02-03', 120),6, 3, 34, 'Sábado', 'Sáb', 'Fin de semana', 5, 5, CONVERT(DATETIME, '
insert into Date_Dimension values(20240204, CONVERT(DATETIME, '2024-02-04', 120),7, 4, 35, 'Domingo', 'Dom', 'Fin de semana', 5, 5, CONVERT(DATETIME,
```

Por último, solo queda insertar los datos de fechas de la tabla auxiliar en la tabla de a dimensión tiempo DimTiempo. Para ellos se utiliza la consulta SQL *Query Llenado DimTiempo.sql*

- Ejecutar la consulta del script *Query Llenado DimTiempo.sql* en SQL Server Management Studio conectado a la base AmarettoDW

Script DimTiempo:

```
insert into DIMTIEMPO
select date_key, full_date, day_of_week, day_num_in_month,
day_num_overall, day_name, day_abbrev, weekday_flag,
week_num_in_year, week_num_overall, week_begin_date, week_begin_date_key,
[month], month_num_overall, month_name,
month_abbrev, [quarter], [year], yearmo, fiscal_month, fiscal_quarter,
fiscal_year, last_day_in_month_flag,
same_day_year_ago_date from Date_Dimension
```

Scripts en PostgreSQL (Base de datos de Odoo)

Creación de tabla auxiliar para compras en el inventario.

Debido a que Odoo no registra como tal una tabla de compras que pueda rescatar la información necesaria para nuestra solución, se procede por ello a ejecutar un script en la base de datos de Odoo tipo PostgreSQL para la creación de una tabla auxiliar que almacene los registros de compra en el inventario.

Script *compra_inventario.sql*:

```
CREATE TABLE compra_inventario (
    id SERIAL PRIMARY KEY,
    product_id INT NOT NULL,
        magento_product_id INT,
    product_name VARCHAR,
    product_code VARCHAR,
    date INT,
    action_type VARCHAR(20), -- 'creation' o 'stock_increase'
    stock_added FLOAT DEFAULT 0
)
```

Trigger de actualización de Stock de producto.

Este trigger sirve para la actualización del stock de productos en el inventario cuando se realiza la sincronización de los dos sistemas transaccionales (Magento y Odoo). El script *update_stock_product_trigger.sql* se ejecuta en la base de datos PostgreSQL de Odoo.

Script *update_stock_product_trigger.sql:*

```
CREATE OR REPLACE FUNCTION log_stock_increase_in_quant()
RETURNS TRIGGER AS $$
BEGIN
    IF ((TG_OP = 'INSERT' AND NEW.quantity IS NOT NULL) OR (TG_OP =
'UPDATE' AND (NEW.quantity > OLD.quantity) and NEW.quantity > 0 and
OLD.quantity >= 0)) THEN
        INSERT INTO compra_inventario (product_id, magento_product_id,
product_name, product_code, action_type, stock_added, date)
        VALUES (
            NEW.product_id,
            COALESCE((SELECT CAST(magento_id AS INT) FROM product_product
WHERE id = NEW.product_id),-1),
            (SELECT name FROM product_template WHERE id = (SELECT
product_tmpl_id FROM product_product WHERE id = NEW.product_id)),
            (SELECT default_code FROM product_product WHERE id =
NEW.product_id),
            'stock_increase',
            (NEW.quantity - COALESCE(OLD.quantity, 0)),
            TO_CHAR(CURRENT_DATE, 'YYYYMMDD')::INT
        );
    END IF;
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER trigger_stock_increase_in_quant
AFTER INSERT OR UPDATE ON stock_quant
FOR EACH ROW
EXECUTE FUNCTION log_stock_increase_in_quant();
```