ITESM Campus Querétaro
Bases de Datos Avanzadas
Lab: Ejemplos de ETL
Fecha de entrega: martes 26 abril de 2022

*Para obtener la calificación del presente laboratorio, es necesario tener la evidencia del desarrollo del mismo. La evidencia será un archivo PDF el cual contendrá las imágenes de pantallas que se van a ir indicando en el desarrollo del laboratorio como ***EVIDENCIA****. *El archivo lo suben a canvas en la tarea nombrada "***Lab: Éjemplos ETL" El nombre del archivo será: Matricula_LabETL.pdf**.

En el presente laboratorio revisaremos varios ejemplos que puedan ayudar con el proceso **ETL** de sus **cubos** del **DWH**.

Parte I. Configurar el ambiente

Para esta parte sigue las instrucciones del profesor

a) Para facilitar el manejo del DWH ejecuta el script **OLTP_EM2022.sql** en el esquema de tu matricula. El script creara los objetos (base de datos transaccional). Una vez creados los objetos carga los datos a la tabla VENTAS a partir del archivo **datos_ventas.csv**
b) Adicional en la base de datos OLAP (base de datos del repositorio del DWH con la cuenta DWHA0XXXXXX) ejecutar el script **OLAP_EM2022.sql** el cual creara los objetos que nos ayudaran en el desarrollo del presente laboratorio.

Parte II. Revisión de vistas, tablas y procedimientos.

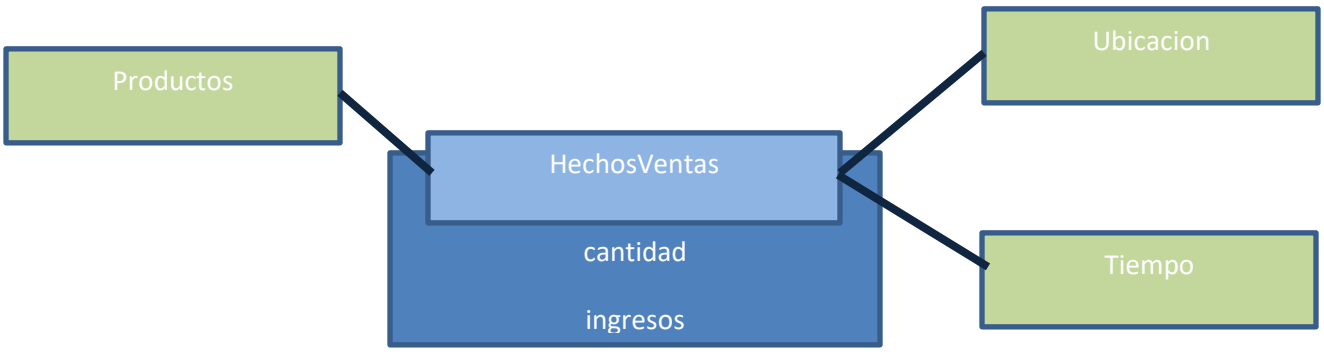
Conéctate al esquema **DWHMatricula -OLAP-** y ejecuta las siguientes instrucciones

```
select count (1) from v_catalogoproductos;
select count (1) from v_catalogoproveedores;
select count (1) from v_catalogosucursales;
select count (1) from v_ventas;
```

*****EVIDENCIA*****
Captura la pantalla con la sentencia y su salida

Revisa la estructura de las vistas.

Considerando el siguiente **esquema estrella** para un **cubo** de un **datawarehouse**:



Para la **dimensión** de "**Productos**", vamos a tener los siguientes atributos:

- *Descripción del producto*
- *Nombre del proveedor*
- *Código del producto (base de datos transaccional)*

Para la **dimensión** "**Ubicación**", vamos a tener los siguientes atributos:

- *Region*
- *Código de sucursal (base de datos transaccional)*

Para la **dimensión** "**Tiempo**", vamos a tener los siguientes atributos:

- Fecha
- Año
- Mes
- Día
- Nombre del día

Revisa las estructuras de las tablas **D_PRODUCTOS, D_TIEMPO, D_UBICACION** y **H_VENTAS.** De la misma manera revisa los **constraints** entre cada una de las tablas.

*****EVIDENCIA*****
Captura la pantalla con la estructura de las tablas y las relaciones entre las dimensiones y hechos

Parte III. Implementación de procedimiento de ETL para tablas de dimensiones.
Para mantener **actualizadas** las **tablas** de **dimensiones** es necesario extraer la información de las bases de datos transaccionales. En nuestro caso para facilitar la extracción creamos las vistas que nos ayudaran en el proceso. Solamente vamos a insertar los registros "**nuevos**" de nuestra base de datos transaccional. Para esta práctica consideramos que no debemos borrar ningún registro del **DWH** aun cuando se borren en la base de datos transaccional, esto para mantener **información** "**histórica**" en el **DWH**.

Para la **dimensión producto** actualiza el procedimiento **ACTUALIZA_PRODUCTO** de la siguiente manera (recuerda estar ya conectado al esquema dwhMatricula):

```
create or replace PROCEDURE ACTUALIZA_PRODUCTO AS
BEGIN
  insert into d_producto
  select seq_d_producto.nextval, codproducto,descripcion, razonsocial
  from v_catalogoproductos cp, v_catalogoproveedores cv
  where cp.rfcproveedor = cv.rfcproveedor
  and cp.codproducto not in (select codproducto from d_producto);
  commit;
END ACTUALIZA_PRODUCTO;
```

Para la **dimensión ubicación** actualiza el procedimiento **ACTUALIZA_UBICACION** a partir de la siguiente consulta:

```
  select seq_d_ubicacion.nextval,region,codsucursal
  from v_catalogosucursales cs
  where cs.codsucursal not in (select codsucursal from d_ubicacion);
```

Ejecuta los **dos procedimientos** que acabas de modificar y en este punto ya tenemos pobladas las dos tablas de dimensiones del cubo.

*****EVIDENCIA*****
REALIZA LA CONSULTA A CADA UNA DE LAS TABLAS QUE SE ACABAN DE POBLAR Y TOMA LA CAPTURA DE LA MISMA – CONSULTA Y DATOS Y/O PESTAÑA DE DATOS DESDE SQL DEVELOPER-

Para la **dimensión tiempo** es necesario ejecutar el procedimiento **PDIMTIEMPO** para poblar con datos entre las fechas **1 de enero de 2022** y **21 de abril de 2022**.

*****EVIDENCIA*****
REALIZA LA CONSULTA A LA TABLA S QUE SE ACABAN DE POBLAR Y TOMA LA CAPTURA DE LA MISMA – CONSULTA Y DATOS Y/O PESTAÑA DE DATOS DESDE SQL DEVELOPER-

Parte IV Implementación de procedimiento de ETL para la tabla de hechos.

Para la implementación de la **tabla** de **hechos** vamos a tomar las siguientes consideraciones:

- Debemos especificar la **fecha** de **inicio** y la **fecha** de **fin** de la extracción de datos. Esto puede ser muy útil para una **carga inicial** de datos y después para **cargas incrementales** (reducción de cargas de trabajo tanto en las bases de datos transaccionales como la del DWH), por ejemplo, cargas diarias o semanales.
- Para evitar la carga de **datos** "**duplicados**" o **errores** en los **constraints** para periodos de tiempo ya capturados, los procedimientos de cargas de hechos deben poder actualizar y/o sustituir la información ya existente. Una manera fácil de realizar esto es **borrando** antes de la inserción los registros que correspondan al periodo de tiempo que se va a actualizar. En nuestro caso el procedimiento de carga debe **iniciar con** el **borrado** de los **registros** que corresponden al periodo del tiempo a insertar.
- Los indicadores de cantidad e ingresos se calculan de la siguiente manera
    - **cantidad** → sum(ventas.cantidad)
    - **ingresos** → sum(ventas.cantidad*ventas.precio)
- Todas las **dimensiones** deben **agruparse** juntas, considerando el tiempo, producto y ubicación.
- Para el caso de la dimensión tiempo se debe usar la **función trunc** de la fecha para que sea fácil la comparación de las fechas.

Teniendo en cuenta las consideraciones anteriores, **escribe** una **consulta** que obtenga el **identificador** del **producto**, **ubicación** y **tiempo** de las **dimensiones** y se calcule los **indicadores** a partir de la **vista** de **ventas**. Usa la **función** de **agrupación** para que puedas realizar la consulta. Las condiciones deben cumplir con lo siguiente:

- Para la dimensión producto, usar el correspondiente código de producto transaccional con su correspondiente registro en la vista de ventas.
- Para la dimensión ubicación, usar el correspondiente código de ubicación transaccional con su correspondiente registro en la vista de ventas.
- Para la dimensión tiempo usa la función **between fechainicio and fechafinal** para obtener los registros de la vista de ventas y realizar el **join** con la función **trunc** a la correspondiente columna de la dimensión del tiempo.

Una vez que tengas la consulta, ajusta el procedimiento **ACTUALIZA_VENTAS** con la siguiente estructura:

```
create or replace
PROCEDURE ACTUALIZA_VENTAS
(
  FECHAINICIAL IN DATE
, FECHAFINAL IN DATE
) AS
  vFechaInicial date;
  vFechaFinal   date;
  vdProPk       number;
  vdTiempoPk    number;
  vdUbiPk       number;
  v_cantidad    number;
  v_ingresos    number;

  cursor c_tiempo is
  select dtiempopk
  from d_tiempo
  where fecha between vFechaInicial and vFechaFinal;

  cursor c_ventas is
  -- código de la consulta desarrollada_


BEGIN
  vFechaInicial := FECHAINICIAL;
  vFechaFinal   := FECHAFINAL;

  open c_tiempo;
  LOOP
    fetch c_tiempo into vdTiempoPk;
    exit when c_tiempo%NOTFOUND;
    delete from h_ventas where dtiempopk=vdTiempoPk;
    commit;
  END LOOP;
  close c_tiempo;

  open c_ventas;
  LOOP
    fetch c_ventas into vdProPk, vdUbiPk, vdTiempoPk, v_cantidad, v_ingresos;
    exit when c_ventas%NOTFOUND;
    insert into h_ventas (hventaspk,dprodpk,dubipk,dtiempopk,cantidad,ingresos)
    values (seq_h_ventas.nextval, vdProPk, vdUbiPk, vdTiempoPk, v_cantidad, v_ingresos);
    commit;
  END LOOP;
  close c_ventas;
END ACTUALIZA_VENTAS;
/
```

Ejecuta el procedimiento ACTUALIZA_VENTAS para poblar con datos entre las fechas **1 de enero de 2022** y **18 de marzo de 2022**.

EVIDENCIA

CAPTURA DE PANTALLA EN DONDE SE MUESTRE EL TOTAL DE REGISTROS DE LA TABLA DE HECHOS

Parte V Consultas

La siguiente consulta nos da como resultado las ventas por producto acumulado por mes desde el 1 de enero hasta el 1 de abril del 2022. Impleméntala en una vista que se llame ventaProductoMes

```
select descproducto as "PRODUCTO",
sum(cantidad) as "CANTIDAD VENDIDA",
MES
from d_producto dp,
h_ventas hv,
d_tiempo dt
where dp.dprodpk = hv.dprodpk
and dt.dtiempopk = hv.dtiempopk
and dt.fecha between to_date ('01-01-2022','DD-MM-YYYY') and to_date('01-04-2022','DD-MM-YYYY')
group by descproducto, MEs
order by 1,to_char(to_date(MES,'Month'),'MM') asc
```

El resultado de la consulta les debe dar algo como esto:

	PRODUCTO	CANTIDAD VENDIDA	MES
1	Coca cola	3291	JANUARY
2	Coca cola	2751	FEBRUARY
3	Coca cola	1899	MARCH
4	Fritos	3353	JANUARY
5	Fritos	2912	FEBRUARY
6	Fritos	1897	MARCH
7	Manzana Lift	3500	JANUARY
8	Manzana Lift	2801	FEBRUARY
9	Manzana Lift	1986	MARCH
10	Pay de nuez	3416	JANUARY
11	Pay de nuez	2968	FEBRUARY
12	Pay de nuez	1736	MARCH
13	Pinguinos	3154	JANUARY
14	Pinguinos	3188	FEBRUARY
15	Pinguinos	1698	MARCH
16	Sabritas	3297	JANUARY
17	Sabritas	2973	FEBRUARY
18	Sabritas	1854	MARCH
19	Sabritones	3009	JANUARY
20	Sabritones	3203	FEBRUARY
21	Sabritones	1977	MARCH
22	Sprite	3623	JANUARY
23	Sprite	2679	FEBRUARY
24	Sprite	1610	MARCH
25	Submarinos	3156	JANUARY
26	Submarinos	2982	FEBRUARY
27	Submarinos	1864	MARCH
28	Victoria	3372	JANUARY
29	Victoria	2827	FEBRUARY
30	Victoria	1913	MARCH

Siguiendo el mismo ejemplo anterior implementen las vistas siguientes:

- Ventas de cada uno de los productos para todos los viernes del mes de FEBRERO
- Ventas de “Pay de Nuez” en cada sucursal
- Ventas de “Pinguinos” en cada sucursal, pero solamente los miércoles
- Ventas de TODOS los productos por sucursal y mes

EVIDENCIA

CAPTURA DE PANTALLA EN DONDE SE MUESTRE LA CONSULTA Y EL RESULTADO DE LAS MISMAS – 5 REPORTES – ANTERIORES.