

Tema:

* Practica 2

Curso:

* SQL Server Implementación.

Docente:

* Eric Gustavo Coronel Castillo

Integrantes:

* Joan Martin Echandia Muñoz
* Feliz Augusto Tejada Villar
* Alexander Gracey Ramos

Índice

[1. Datos de la Empresa 3](#_Toc64471886)

[2. Descripción de los procesos 3](#_Toc64471887)

[3. Tablas 4](#_Toc64471888)

[4. Erwin Data Modeler 4](#_Toc64471889)

[5. Generación del script 5](#_Toc64471890)

[5.1. Restricciones 5](#_Toc64471891)

[6. Conclusiones 7](#_Toc64471892)

[7. Recomendaciones 7](#_Toc64471893)

[8. Enlace del video 7](#_Toc64471894)

Practica 2



# Datos de la Empresa

Molitalia S.A. es una empresa encargada de la producción y distribución de productos de primera necesidad a nivel nacional. Por ello, la base de sus procesos se centra en la adquisición de insumos y transporte de productos terminados a los distintos puntos de venta. Es por esta razón que la información utilizada para este trabajo se centrará exclusivamente en el aprovisionamiento y transporte de productos realizado el último trimestre del año 2019.

# Descripción de los procesos

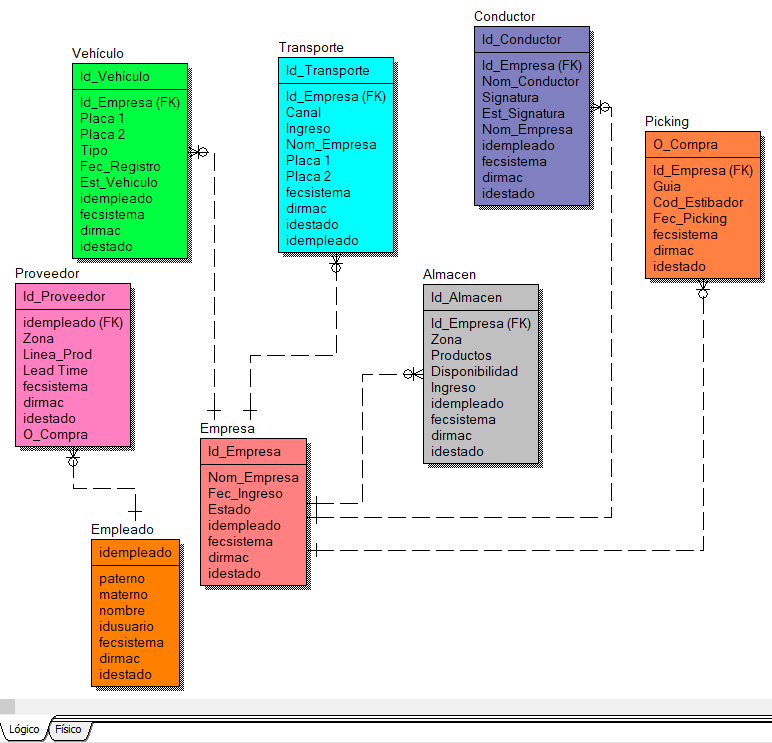
Durante la gestión de insumos se debe considerar las órdenes de compra, armado de la programación de transporte y “picking”, y selección de estibadores para la distribución de estos en el almacén predefinido anteriormente. Por otro lado, la gestión de transporte de productos terminados corresponde a la enumeración de las ordenes de transporte, programación de la zona y punto de venta respectivo, además de la selección del “courier” apropiado para dicho transporte.

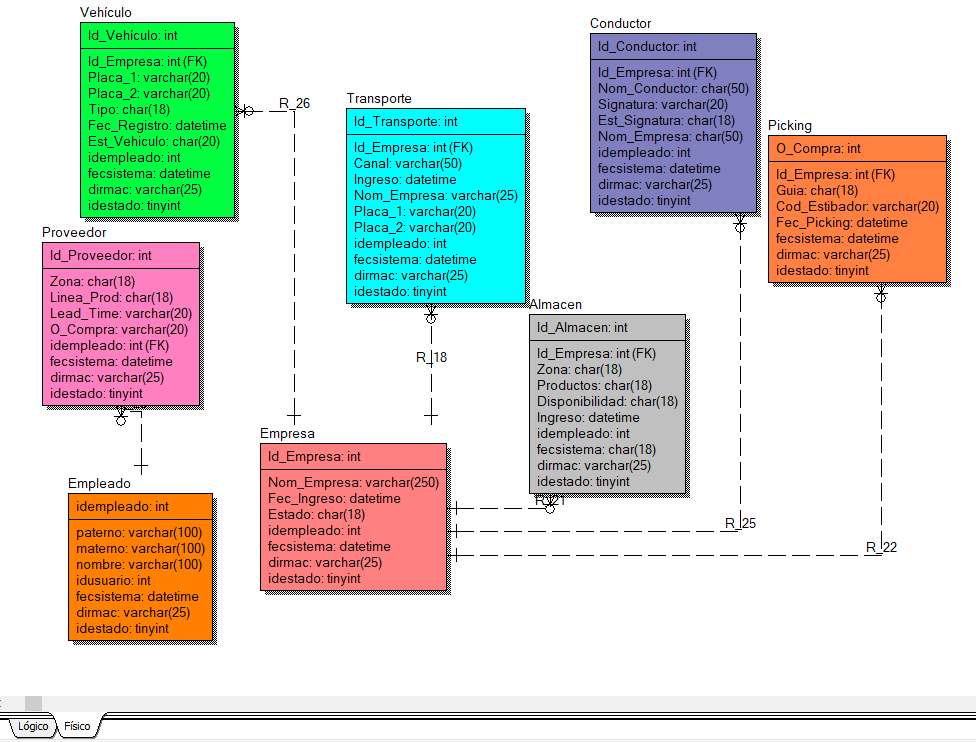
* Abastecimiento:
* Distribución:

# Tablas

Las tablas utilizadas para este trabajo son 7: Transporte, Almacén, Vehículo, Proveedor, Empresa, Conductor y Picking. Estas se enviarán en conjunto con este trabajo para reducir el número de hojas posible y será enviado en formato xslx. Asimismo, cabe resaltar que para la generación de las tablas en SQL se utilizará como máximo 8 filas de cada tabla para que la información sea más manejable.

# Erwin Data Modeler

* Modelo Lógico
* Modelo Físico



# Generación del script

El script se enviará en conjunto con este trabajo para reducir el número de hojas posible. Este será enviado en formato .txt y .sql para que sea revisado con detenimiento.

# Restricciones

Se plantea utilizar las siguientes restricciones para especificar el tipo de información que se planteará en cada una de las tablas de la base de datos.

* Primary Key: La clave primaria (PK) identifica de manera única cada fila de una tabla.

CREATE TABLE Trabajo\_01.dbo.Transporte(

id\_transporte INT NOT NULL

CONSTRAINT PK\_Transporte PRIMARY KEY,

canal VARCHAR (50) NOT NULL,

ingreso DATE NOT NULL,

id\_empresa INTEGER NOT NULL,

Nom\_empresa VARCHAR (250) NOT NULL,

placa\_1 VARCHAR (50) NOT NULL,

placa\_2 VARCHAR (50) NOT NULL DEFAULT 'NONE',

CONSTRAINT U\_Transporte\_id\_transporte UNIQUE (id\_transporte),

CONSTRAINT CK\_Transporte\_canal CHECK (canal = 'Mercados' or

canal = 'Distribuidores' or canal = 'Mayoristas'),

CONSTRAINT CK\_Transporte\_ingreso CHECK (ingreso <= '2020-01-01')

)

GO

El siguiente ejemplo muestra cómo se utiliza la función. En el caso de la tabla Transporte se considera a “id\_transporte” como su PK debido a que es el registro único que lleva cada envío. Con este se obtiene información relacionada a la zona, mercadería y fecha en la se realizó dicha actividad.

* Foreign Key:

CREATE TABLE Trabajo\_01.dbo.Transporte(

id\_transporte INT NOT NULL

CONSTRAINT PK\_Transporte PRIMARY KEY,

canal VARCHAR(50) NOT NULL,

ingreso DATE NOT NULL,

id\_empresa INTEGER NOT NULL

CONSTRAINT FK\_Transporte\_Empresa

FOREIGN KEY(id\_empresa)

REFERENCES dbo.Empresa(id\_empresa),

Nom\_empresa VARCHAR(250) NOT NULL,

placa\_1 VARCHAR(50) NOT NULL,

placa\_2 VARCHAR(50) NOT NULL DEFAULT 'NONE',

CONSTRAINT U\_Transporte\_id\_transporte UNIQUE (id\_transporte),

CONSTRAINT CK\_Transporte\_canal CHECK (canal = 'Mercados' or

canal = 'Distribuidores' or canal = 'Mayoristas'),

CONSTRAINT CK\_Transporte\_ingreso CHECK (ingreso <= '2020-01-01')

)

GO

El siguiente ejemplo muestra cómo se utiliza la función. Se crearán los datos del transporte, según los datos asociados al id de la empresa, según las referencia anteriores del dbo.Empresa.

* Unique: Esta restricción es utilizada para definir admitir sólo valores únicos en la columna.

CREATE TABLE Trabajo\_01.dbo.Conductor(

id\_conductor VARCHAR (50) NOT NULL

CONSTRAINT PK\_Conductor PRIMARY KEY,

Nom\_conductor VARCHAR (250) NOT NULL,

Signatura VARCHAR (50) NOT NULL,

Est\_signatura VARCHAR (50) NOT NULL,

Nom\_empresa VARCHAR (250) NOT NULL,

CONSTRAINT u\_conductor\_id\_conductor UNIQUE (id\_conductor),

CONSTRAINT u\_conductor\_signatura UNIQUE(Signatura),

CONSTRAINT CK\_conductor\_Est\_signatura CHECK (Est\_signatura = 'Vigente' or Est\_signatura = 'Caducado')

)

GO

El siguiente ejemplo muestra cómo se utiliza la función. Las columnas id\_conductor y signatura tendrán esta característica debido a que son códigos que varían según el conductor y no pueden repetirse.

* Check: La restricción “Check” indica limitaciones o condiciones para la columna durante el ingreso de datos.

CREATE TABLE Trabajo\_01.dbo.Proveedor(

id\_proveedor INT NOT NULL

CONSTRAINT PK\_Proveedor PRIMARY KEY,

Zona VARCHAR (50) NOT NULL,

Linea\_prod VARCHAR (50) NOT NULL,

Lead\_time VARCHAR (50) NOT NULL,

O\_Compras VARCHAR (50) NOT NULL,

CONSTRAINT u\_proveedor\_id\_proveedor UNIQUE (id\_proveedor),

CONSTRAINT u\_proveedor\_o\_compras UNIQUE (O\_compras),

CONSTRAINT CK\_proveedor\_Zona CHECK (Zona = 'Norte'

or Zona = 'Sur' or Zona = 'Centro'),

CONSTRAINT CK\_proveedor\_Linea\_prod CHECK (Linea\_prod = 'Mascotas'

or Linea\_prod = 'Dulces' or Linea\_prod = 'Harinas' or

Linea\_prod = 'Avenas' or Linea\_prod = 'Panetones')

)

GO

El siguiente ejemplo muestra cómo se utiliza la función. Las columnas con esta condición son “Zona” y “Linea\_prod”. En el primero se planteó una restricción en la que sólo se puede ser admitido que la celda tenga sea Norte, Sur o Centro. Por otro lado, en la “Linea\_prod” sólo se permite ingresar la línea de productos que se indica en la programación.

* NULL / NOT NULL: Es el comando utilizado para especificar si en las celdas puede ser admitido los valores nulos o no.

CREATE TABLE Trabajo\_01.dbo.Empresa(

id\_empresa INT NOT NULL

CONSTRAINT PK\_Empresa PRIMARY KEY,

Nom\_empresa VARCHAR(250) NOT NULL,

Fec\_ingreso DATE NOT NULL,

Estado VARCHAR(50) NOT NULL,

CONSTRAINT CK\_empresa\_Estado CHECK (Estado =

'Disponible' or Estado = 'No disponible'),

CONSTRAINT CK\_empresa\_fec\_ingreso CHECK (fec\_ingreso <=

'2020-01-01')

)

GO

El siguiente ejemplo muestra cómo se utiliza la función. Una vez definido el tipo de datos que almacenará cada columna se procede a permitir o no los datos nulos. Con esto, sabemos que en las columnas “id\_empresa”, “Nom\_empresa”, “Fec\_ingreso” y “Estado” no se admitirán las celdas vacías.

# Conclusiones

* Con ayuda del programa SQL la búsqueda de datos específicos y realización de consultas en estos es más sencillo y rápido, a diferencia de las hojas de Excel.
* El programa SQL tiene capacidad de almacenar más datos y realizar consultas en tablas distintas. Realizar dichas tareas en el programa de Excel implicaría una demora y, en el peor de los casos, errores que podrían destruir la información.

# Recomendaciones

* Usar funciones en SQL que nos permita almacenar datos de manera rápida.
* Usar una tecnología de almacenamiento de datos en tiempo real.
* Actualizar los datos usando una tecnología REST.
* Para la importación de datos de Excel a SQL se recomienda realizar un tratamiento en la hoja para eliminar ceros o errores de cálculo. Con esto, se evita que haya problemas al pasar la información al programa. Caso contrario, especificar que los datos tratados pueden tener vacíos o “NULL”.

# Enlace del video

<https://youtu.be/RidkJ4Renns>