***Asignatura:***

*Bases de Datos*

*Primer cuatrimestre*

***Ciclo Lectivo 2023***

***Universidad Tecnológica Nacional***

*Facultad Regional Resistencia*

*Ingeniería en Sistemas de Información*

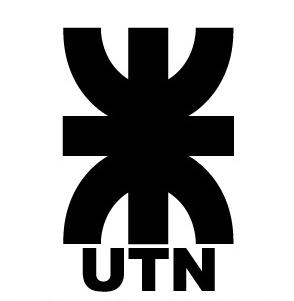
**Alumnos:**

* **Córdoba, Rodrigo Julián**
* **Guzmán, Tomás Ignacio**
* **Ramírez, Eduardo Manuel**
* **Saucedo, Gonzalo Nicolás**
* **Stegmayer, Tobías Sebastián**
* **Vallejos, Enzo Nahuel**

**Docentes:**

* **Fernández, Juan Carlos**
* **Eiman, Luis**
* **Orcola, Carolina**

**Comisión: K3.1 Grupo: 1**



Trabajo Práctico Integrador

Escenario Farmacia

Contenido

[Escenario 2](#_Toc139108076)

[Sistema Gestor de Base de Datos 4](#_Toc139108077)

[Características principales 4](#_Toc139108078)

[Conceptos de la unidad 1 6](#_Toc139108079)

[Análisis del escenario y Modelado 7](#_Toc139108080)

[Diagrama Entidad-Relación planteado: 7](#_Toc139108081)

[Consideraciones: 7](#_Toc139108082)

[Esquema Relacional 8](#_Toc139108083)

[Diccionario de datos 10](#_Toc139108084)

[Esquema Físico 15](#_Toc139108085)

[Consultas SQL 21](#_Toc139108086)

[INSERCIÓN, MODIFICACIÓN, BORRADO DE DATOS 22](#_Toc139108087)

[Consultas UPDATE 22](#_Toc139108088)

[Consultas INSERT 27](#_Toc139108089)

[Consultas DELETE 34](#_Toc139108090)

[Consultas SELECT 36](#_Toc139108091)

# Escenario

Se desea mantener una base de datos para la gestión de una cadena de farmacias distribuida en diferentes ciudades. De la ciudad se sabe su nombre y su código postal.

De cada farmacia, su ID, su dirección (calle y número) y los días que le corresponde guardia. Una farmacia está ubicada en una sola ciudad, pero en una ciudad hay más de una farmacia. A su vez, sabemos que por cada ciudad existe un único farmacéutico; es decir, en las ciudades en las que hubiere más de una farmacia, el mismo farmacéutico estará afectado a todas las farmacias de esa ciudad. En cada farmacia trabajan varios empleados.

De cada empleado queremos saber su CUIT, su nombre, la fecha de ingreso laboral. Tenga en cuenta que cada empleado trabaja en una sola farmacia. Esta cadena de farmacias venden medicamentos solo a sus afiliados. Los datos que se deben guardar son: Id de afiliado, apellido y nombre, tipo y número de documento, dirección, localidad, fecha de ingreso, fecha de nacimiento.

Existen dos tipos de afiliados: eventuales, que reciben un 20% de descuento sobre las compras realizadas, y crónicos, cuyo descuento es del 70%. Para el caso de los crónicos se debe guardar información sobre código de diagnóstico y fecha de diagnóstico. Un afiliado crónico puede tener varios diagnósticos.

De cada venta se debe generar un comprobante que contenga: número, fecha, id de afiliado. A su vez cada comprobante contiene un conjunto de medicamentos que se deben cargar teniendo en cuenta los siguientes datos: código de medicamento, precio, descuento (directamente relacionado al tipo de afiliado) y total. Para poder efectivizar una venta, cada farmacia, posee su stock de cada medicamento. Cada medicamento se identifica por código, descripción, presentación (ej: ampollas de 5 unidades, jarabe de 100ml, inyecciones por 10 unidades) y precio, que es el mismo para todas las farmacias. También se conoce la o las monodrogas que componen cada medicamento, el laboratorio que lo comercializa y las acciones terapéuticas que tiene.

De cada monodroga sabemos el nombre científico y el nombre comercial. De cada laboratorio sabemos CUIT, razón social, domicilio. Un laboratorio provee varios medicamentos a esta cadena de farmacias. De las acciones terapéuticas conocemos el nombre y el tiempo que tarda en hacer efecto. Tenga en cuenta que una acción terapéutica puede repetirse para distintos medicamentos. Por ejemplo, el medicamento Dorixina Forte es un medicamento que cuesta $1360 y su presentación es en caja de 20 comprimidos. Tiene como monodrogas Clonixinato de lisina (nombre científico) en 125,00 mg y Dextropropoxifeno napsilato 2 (nombre científico) en 98,00 mg. Sus acciones terapéuticas son analgésicas y antiinflamatorias y tardan 4 horas en hacer efecto en la persona que toma el medicamento.

El sistema deberá permitir consultar la base de datos de diferentes alternativas para medicamentos compuestos por una monodroga, medicamentos de un laboratorio, medicamentos con el mismo nombre y distinta presentación, entre otras. La cadena de farmacias cuenta con un depósito central, que realiza todas las compras y recibe todos los medicamentos solicitados por cada farmacia a los proveedores (laboratorios).

Este depósito posee su propio stock de medicamentos que luego redistribuye a cada farmacia que lo solicite. La forma de ingresar medicamentos al stock del depósito central es a través de los INGRESOS. De cada ingreso se registra: código de Ingreso, fecha de ingreso, código de transporte (es la denominación de la empresa que efectuó el traslado de los medicamentos), CUIT del proveedor. Cada ingreso contiene un conjunto de medicamentos que se deben cargar teniendo en cuenta los siguientes datos: código de medicamento, cantidad.

Se debe tener en cuenta que es necesario poder determinar en qué estado se encuentra cada Ingreso (en confección, terminado, procesado), ya que sólo se puede impactar en el stock del depósito un ingreso terminado.

Existe una lista de empresas de transportes a quienes se le puede designar la tarea de trasladar medicamentos de un depósito a otro. Los datos con que se cuentan son: código de transporte, razón social, CUIT/CUIL, e-mail, teléfono y domicilio. Además, se debe conocer de cada transportista a qué localidades (sucursales) alcanza su servicio.

Cada farmacia que necesite medicamentos, lo debe requerir al depósito central por medio de TRANSFERENCIAS, las cuales deben contener: número de solicitud, Id de farmacia, fecha de confección (es la fecha actual y no puede modificarse). Cada transferencia contiene un conjunto de medicamentos que se deben cargar teniendo en cuenta los siguientes datos: código de medicamento y cantidad. Se debe tener en cuenta que es necesario poder determinar en qué estado se encuentra cada solicitud (Pendiente,, terminado, procesado), ya que sólo se puede impactar en el stock de la farmacia una transferencia terminada.

# 

# Sistema Gestor de Base de Datos

## Características principales

MySQL es un sistema de gestión de bases de datos relacional (RDBMS) de código abierto y ampliamente utilizado. A continuación, se detallan algunas de las características principales de MySQL basadas en la documentación oficial:

* **Fiabilidad y rendimiento:** MySQL es conocido por su alta fiabilidad y rendimiento. Está diseñado para manejar grandes volúmenes de datos y cargas de trabajo intensivas. Ofrece una gestión eficiente de transacciones y consultas, así como una rápida recuperación de fallos.
* **Escalabilidad y capacidad de carga:** MySQL es escalable tanto en términos de tamaño de la base de datos como de capacidad de carga. Puede manejar aplicaciones de pequeña a gran escala y puede crecer junto con los requisitos de negocio. Admite la replicación y la partición de datos para distribuir la carga entre varios servidores.
* **Flexibilidad:** MySQL es flexible y se adapta a diversas aplicaciones y entornos. Puede utilizarse en una amplia gama de sistemas operativos, incluyendo Windows, Linux, macOS, etc. Además, es compatible con una variedad de lenguajes de programación y ofrece conectividad a través de APIs, como JDBC, ODBC, y .NET.
* **Seguridad:** MySQL ofrece características de seguridad sólidas para proteger los datos. Soporta autenticación de usuarios, cifrado de datos en tránsito y en reposo, y control de acceso basado en privilegios. También es posible auditar y registrar eventos de seguridad para cumplir con los requisitos de cumplimiento.
* **Replicación y alta disponibilidad:** MySQL permite la replicación de bases de datos para mejorar la disponibilidad y la redundancia. Puede configurarse para replicar datos en tiempo real a servidores secundarios, lo que proporciona mayor tolerancia a fallos y la capacidad de realizar copias de seguridad en línea sin interrupciones.
* **Soporte transaccional:** MySQL admite transacciones ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad), lo que garantiza la integridad y consistencia de los datos. Puede utilizar el control de concurrencia multiversión (MVCC) para gestionar eficientemente las transacciones concurrentes sin bloqueos innecesarios.
* **Motor de almacenamiento:** MySQL es modular y admite múltiples motores de almacenamiento. El motor de almacenamiento predeterminado es InnoDB, que proporciona transacciones y recuperación de fallos. También ofrece MyISAM, que es adecuado para aplicaciones con requisitos de lectura intensiva, y otros motores como MEMORY, NDB Cluster y ARCHIVE.
* **Herramientas y utilidades**: MySQL proporciona una amplia gama de herramientas y utilidades para administrar y trabajar con bases de datos. Incluye la interfaz de línea de comandos (CLI) de MySQL, así como interfaces gráficas como MySQL Workbench y phpMyAdmin. También ofrece utilidades para la importación/exportación de datos, copias de seguridad y restauración, y optimización del rendimiento

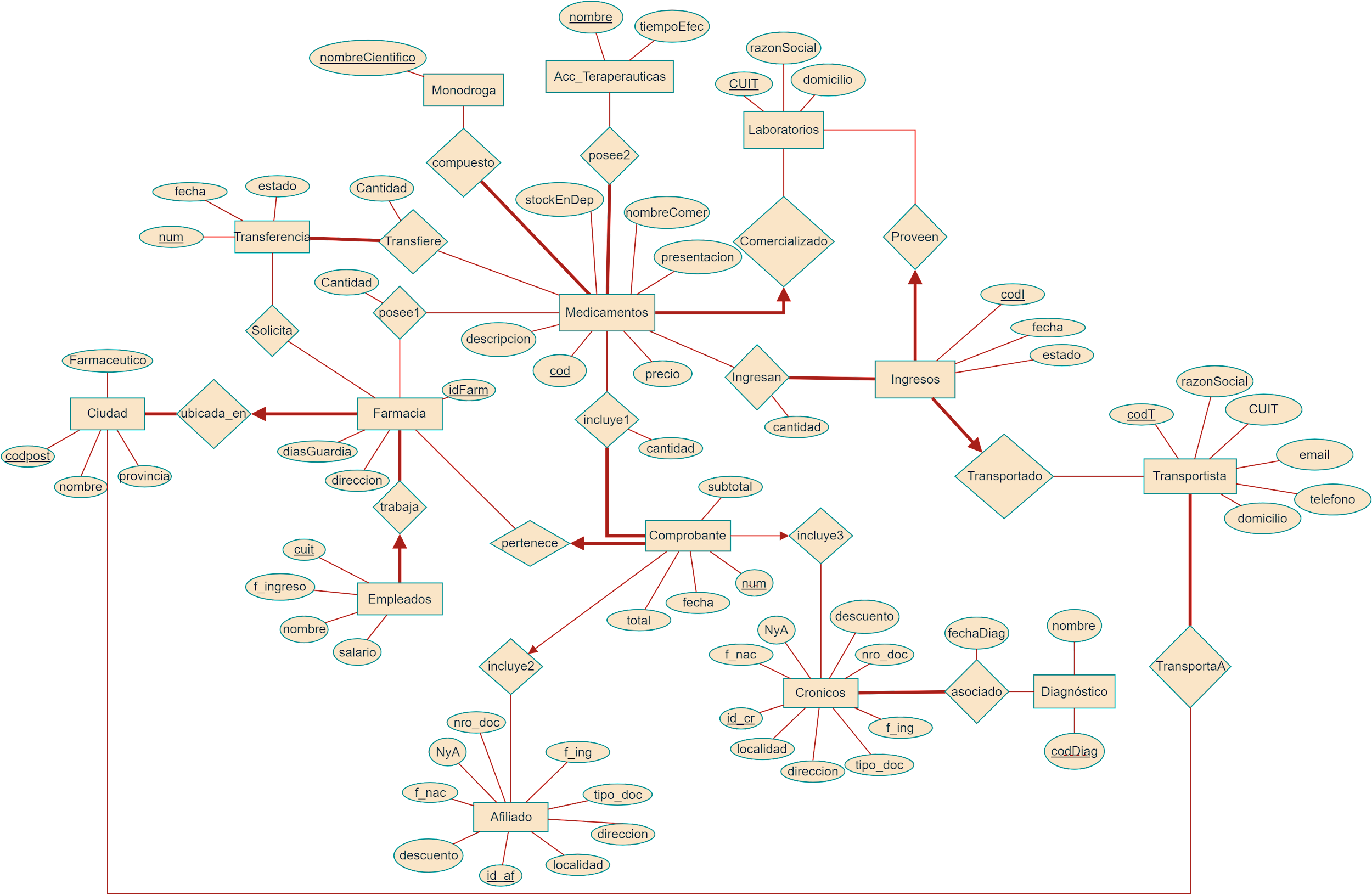
## 

## Conceptos de la unidad 1

* **Independencia de datos:** MySQL, como cualquier otro SGBD relacional, proporciona independencia de datos al separar la estructura lógica de la base de datos de su representación física. Esto significa que los cambios en la estructura física de la base de datos (como la forma en que los datos se almacenan en el disco) no afectan a las aplicaciones ni a los usuarios que acceden a los datos a través de la estructura lógica (como las tablas y las columnas).
* **Soporte para múltiples vistas de datos:** MySQL permite a los usuarios definir "vistas", que son representaciones personalizadas de los datos que pueden simplificar las consultas y mejorar la seguridad. Una vista es esencialmente una consulta almacenada que puede ser tratada como una tabla virtual.
* **Control de redundancia de datos:** MySQL ayuda a controlar la redundancia de datos a través de su soporte para relaciones entre tablas. Por ejemplo, puedes usar claves primarias y foráneas para asegurarte de que los datos no se dupliquen innecesariamente en diferentes tablas.
* **Restricciones de integridad:** MySQL permite definir restricciones de integridad en las tablas, como claves primarias, claves foráneas y restricciones UNIQUE, NOT NULL y CHECK. Estas restricciones ayudan a garantizar que los datos cumplen con ciertas reglas y que la calidad de los datos se mantiene.
* **Seguridad de los datos:** MySQL proporciona una serie de características de seguridad para proteger los datos, incluyendo la autenticación de usuarios, el control de acceso basado en privilegios y la encriptación de datos. Por ejemplo, puedes otorgar a diferentes usuarios diferentes niveles de acceso a la base de datos y a sus objetos (como tablas y vistas), y puedes usar SSL para encriptar las comunicaciones entre el cliente de MySQL y el servidor.

# Análisis del escenario y Modelado

## Diagrama Entidad-Relación planteado:

[](https://app.diagrams.net/?page-id=R2lEEEUBdFMjLlhIrx00&scale=auto" \l "G1C_JBskfHOxcKy5gQI_Ss0E70aFCfRe8M)

Link: <https://app.diagrams.net/?page-id=R2lEEEUBdFMjLlhIrx00&scale=auto#G1C_JBskfHOxcKy5gQI_Ss0E70aFCfRe8M>

## Consideraciones:

1. **“Farmacéutico”** es un atributo de “**Ciudad”**, ya que un único farmacéutico estará afectado a todas las farmacias de esa ciudad.
2. Se considera la entidad “**afiliado”** como un afiliado eventual.
3. Un “**Comprobante”** puede incluir tanto cero o un “**Afiliado”** como cero o un “**Cronicos”**, por lo tanto no se puede evitar que algún comprobante pueda incluir un afiliado crónico como uno eventual a la vez.
4. En “**posee1”** el atributo descriptivo **“cantidad”** representa que cada farmacia tiene su propio stock de ese medicamento.
5. La entidad medicamento almacena en stockEnDep la cantidad de stock total de ese medicamento en el depósito.

# Esquema Relacional

|  |
| --- |
| Ciudad(codpost: integer, nombre: string, provincia: string, farmaceutico: string): cp: codpost  Afiliado(id\_af: integer, descuento: integer, f\_nac: date, NyA: string, nro\_doc: integer, f\_ing: date, tipo\_doc: integer, direccion: integer, localidad: integer): cp: id\_af  Cronicos(id\_af: integer, localidad: string, direccion: string, tipo\_doc: string, f\_ing: date, nro\_doc: integer, descuento: integer, NyA: string, f\_nac: date): cp: id\_af  Diagnostico (nombre: string, codDiag: integer): cp: codDiag  Transportista(codT: integer, razonSocial: string, CUIT: integer, email: string, telefono: integer, domicilio: string): cp: codT  Laboratorios (CUIT: integr, razonSocial: string, domicilio: string): cp: CUIT  Acc\_Teraperauticas (nombre: string, tiempoEfec: integer): cp: nombre  Monodroga (nombreCientifico: string): cp: nombreCientifico  Transferencia (num: integer, fecha: date, estado: string): cp: num  Farmacia(diasGuardia: date, direccion: string, idFarm: **int**, codpost:integer): cp: idFarm cf: codpost→Ciudad(codpost) cnn: codpost  Empleados(cuit: integer, f\_ingreso: date, nombre: string, idFarm:**int,** salario: float): cp: cuit cf: idFarm→Farmacia(idFarm) cnn: idFarm  Comprobantes(num: integer, fecha: date, total: **float**, id\_af:integer, id\_cronico:integer, idFarm: int, subtotal: float ): cp: num, idFarm cf: id\_af→ Afiliado(id\_af), id\_cronico→ Cronicos(id\_af), idFarm → Farmacia(idFarm)  Ingresos (codl: integer, fecha: date, estado: string, cantidad: integer, CUIT: integer, codT: integer): cp: codl cf: CUIT → Laboratorios(CUIT), codT → Transportistas(codT) cnn: CUIT  Medicamentos (cod: integer, CUIT: string, nombreComer: string, descripcion: string, precio: **float**, presentación: string, stockEnDep:integer): cp: cod cf: CUIT→ Laboratorios(CUIT)  Transfiere(num: integer, cod:integer, cantidad:integer): cp:num, cod cf:num→Transferencia(num), cod → Medicamentos (Cod)  cnn: cantidad  Incluye1(cod:integer, num: integer, cantidad: integer): cp:num, cod cf: num→ Comprobante(num), cod → Medicamentos(cod)   posee1 (cod: integer, idFarm: integer, Cantidad: integer): cp: (cod, idFarm) cf: cod → Medicamentos(cod), idFarm→ Farmacia(idFarm)  posee2 (cod: integer, nombre: string): cp: (cod, nombre) cf: cod → Medicamentos(cod), nombre → Acc\_Teraperauticas(nombre)  compuesto(cod: integer, nombreCientifico): cp: (cod, nombreCientifico) cf: cod → Medicamentos(cod), nombreCientifico → Monodroga(nombreCientifico)  ingresan(codI: integer, cod: integer, cantidad: integer): cp:(codI, cod) cf: codI → Ingresos(codI), cod → Medicamentos(cod)  asociado(codDiag: integer, id\_af: integer, fechaDiag: date): cp: (codDiag, id\_af) cf: codDiag → Diagnostico(codDiag), id\_af → Cronicos(id\_af)  TransportaA(codT: integer, codpost: integer): cp: (codT, codpost) cf: codpost → Ciudad(codpost), codT → Transportista(codT)  Solicita(num: integer, idFarm: integer): cp: (num, idFarm) cf: num →Transferencia(num), idFarm → Farmacia(idFarm) |

# Diccionario de datos

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **NombreCampo** | **Descripción** | **Entidad/es a las que pertenece** | **Función (PK, FK)** | **Dominio / Longitud** | **Acepta Nulos?** |
| cantidad | Cantidad de ingreso | Ingresos, Transfiere, posee1, ingresan |  | INTEGER | No |
| cod | Código del comprobante | Comprobantes, Transfiere, Incluye1 | PK | INTEGER | No |
| cod | Código del medicamento | Medicamentos, Transfiere, Incluye1, posee1, posee2, compuesto, ingresan | PK | INTEGER | No |
| codDiag | Código de diagnóstico | Diagnostico | PK | INTEGER | No |
| codI | Código de ingreso en ingresan | ingresan | FK | INTEGER | No |
| codl | Código de ingreso | Ingresos | PK | INTEGER | No |
| codpost | Código postal | Ciudad | PK | INTEGER | No |
| codpost | Código postal de la farmacia | Farmacia | FK | INTEGER | No |
| codpost | Código postal del transporte | Transporta | FK | INTEGER | No |
| codT | Código del transportista | Transportista | PK | INTEGER | No |
| codT | Código del transportista en los ingresos | Ingresos | FK | INTEGER | No |
| CUIT | CUIT (Clave Única de Identificación Tributaria) del transportista | Transportista, Laboratorios, Medicamentos |  | INTEGER | No |
| cuit | CUIT (Clave Única de Identificación Tributaria) del empleado | Empleados | PK | INTEGER | No |
| CUIT | CUIT (Clave Única de Identificación Tributaria) en los ingresos | Ingresos | FK | INTEGER | No |
| descripcion | Descripción del medicamento | Medicamentos |  | VARCHAR(100) | No |
| descuento | Descuento del afiliado | Afiliado |  | INTEGER | No |
| diasGuardia | Días de guardia de la farmacia | Farmacia |  | DATE | No |
| direccion | Dirección del afiliado | Afiliado, Cronicos, Empleados |  | INTEGER | No |
| domicilio | Domicilio del transportista | Transportista, Laboratorios |  | VARCHAR(80) | No |
| domicilio | Domicilio de la farmacia | Farmacia, Empleados |  | VARCHAR(80) | No |
| email | Correo electrónico del transportista | Transportista |  | VARCHAR(60) | No |
| estado | Estado de la transferencia | Transferencia |  | VARCHAR(10) | No |
| f\_ing | Fecha de ingreso del afiliado | Afiliado |  | DATE | No |
| f\_ingreso | Fecha de ingreso del empleado | Empleados |  | DATE | No |
| f\_nac | Fecha de nacimiento del afiliado | Afiliado |  | DATE | No |
| farmaceutico | Nombre del farmacéutico de la ciudad | Ciudad |  | VARCHAR(50) | No |
| fecha | Fecha de la transferencia | Transferencia, Ingresos, Comprobantes |  | DATE | No |
| id\_af | Identificador del afiliado | Afiliado | PK | INTEGER | No |
| id\_af | Identificador del afiliado en el comprobante | Comprobantes | FK | INTEGER | No |
| id\_af | Identificador del afiliado en el comprobante | Medicamentos, Incluye1, asociado, Comprobantes | FK | INTEGER | No |
| id\_cronico | Identificador del afiliado crónico en el comprobante | Comprobantes | FK | INTEGER | No |
| id\_cronico | Identificador del afiliado crónico en el comprobante | Medicamentos, Incluye1, asociado, Comprobantes | FK | INTEGER | No |
| idFarm | Identificador de la farmacia | Farmacia, Empleados | PK | INTEGER | No |
| idFarm | Identificador de la farmacia en solicita | Solicita | FK | INTEGER | No |
| localidad | Localidad del afiliado | Afiliado, Cronicos |  | INTEGER | No |
| nombre | Nombre de la ciudad | Ciudad |  | VARCHAR(50) | No |
| nombre | Nombre de la terapia | Acc\_Teraperauticas, posee2 | PK | VARCHAR(50) | No |
| nombre | Nombre del empleado | Empleados |  | VARCHAR(50) | No |
| nombreCient | Nombre científico de la monodroga en compuesto | compuesto |  | VARCHAR(50) | No |
| nombreClient | Nombre de la monodroga | Monodroga, compuesto | PK | VARCHAR(50) | No |
| nombreComer | Nombre comercial de la monodroga | Monodroga |  | VARCHAR(80) | No |
| nro\_doc | Número de documento del afiliado | Afiliado |  | INTEGER | No |
| num | Número de transferencia | Transferencia, Transfiere, Comprobantes, Solicita | PK | INTEGER | No |
| num | Número de transferencia en solicita | Solicita | FK | INTEGER | No |
| NyA | Nombre y Apellido del afiliado | Afiliado |  | VARCHAR(50) | No |
| precio | Precio del medicamento | Medicamentos |  | FLOAT | No |
| presentación | Presentación del medicamento | Medicamentos |  | VARCHAR(100) | No |
| provincia | Provincia de la ciudad | Ciudad |  | VARCHAR(50) | No |
| razonSocial | Razón social del transportista | Transportista |  | VARCHAR(80) | No |
| telefono | Número de teléfono del transportista | Transportista |  | INTEGER | No |
| tiempoEfec | Tiempo efectivo de la terapia | Acc\_Teraperauticas |  | INTEGER | No |
| tipo\_doc | Tipo de documento del afiliado | Afiliado |  | INTEGER | No |
| total | Total del comprobante | Comprobantes |  | FLOAT | No |

Link

[TPI BDD - Diccionario de Datos](https://docs.google.com/spreadsheets/d/1V8Z5XSRgavkA4P5xmLrfUw9o5jyPH9TkMR7kQOmB_u0/edit?usp=sharing)

# Esquema Físico

|  |
| --- |
| **CREATE TABLE Ciudad (  codpost INTEGER,  nombre VARCHAR(50),  provincia VARCHAR(50),  farmaceutico VARCHAR(50),  PRIMARY KEY (codpost) );  CREATE TABLE Afiliado (  id\_af INTEGER,  descuento INTEGER,  f\_nac DATE,  NyA VARCHAR(50),  nro\_doc INTEGER,  f\_ing DATE,  tipo\_doc VARCHAR(3),  direccion VARCHAR(80),  localidad VARCHAR(50),  PRIMARY KEY (id\_af) );  CREATE TABLE Cronicos (  id\_cr INTEGER,  descuento INTEGER,  PRIMARY KEY (id\_cr) );  CREATE TABLE Diagnostico (  codDiag INTEGER,  nombre VARCHAR(80),  PRIMARY KEY(codDiag) );  CREATE TABLE Transportista (  codT INTEGER,  razonSocial VARCHAR(80),  CUIT VARCHAR(11),  email VARCHAR(60),  telefono INTEGER,  domicilio VARCHAR(80),  PRIMARY KEY(codT) );  CREATE TABLE Laboratorios (  CUIT VARCHAR(11),  razonSocial VARCHAR(80),  domicilio VARCHAR(80),  PRIMARY KEY(CUIT) );  CREATE TABLE acc\_terapeuticas (  nombre VARCHAR(50),  tiempoEfec INTEGER,  PRIMARY KEY(nombre) );  CREATE TABLE Monodroga (  nombreCientifico VARCHAR(50),  PRIMARY KEY(nombreCientifico) );  CREATE TABLE Transferencia (  num INTEGER,  fecha DATE,  estado VARCHAR(10),  PRIMARY KEY (num) );  CREATE TABLE Farmacia (  idFarm INTEGER,  codpost INTEGER NOT NULL,  diasGuardia VARCHAR(10),  direccion VARCHAR(80),  PRIMARY KEY(idFarm),  FOREIGN KEY (codpost) REFERENCES Ciudad(codpost)  ON DELETE NO ACTION  ON UPDATE NO ACTION  );  CREATE TABLE Empleados (  cuit VARCHAR(11),  idFarm INTEGER NOT NULL,  nombre VARCHAR(50),  f\_ingreso DATE,**  **salario: FLOAT,  PRIMARY KEY(cuit),  FOREIGN KEY (idFarm) REFERENCES Farmacia(idFarm)  ON DELETE NO ACTION   ON UPDATE NO ACTION );  CREATE TABLE Comprobantes (  num INTEGER,  id\_af INTEGER,  id\_cr INTEGER,  fecha DATE,  total FLOAT,  idFarm INTEGER NOT NULL,**  **subtotal FLOAT**, **PRIMARY KEY (num),  FOREIGN KEY (id\_af) REFERENCES Afiliado(id\_af)  ON DELETE CASCADE  ON UPDATE NO ACTION,  FOREIGN KEY (id\_cr) REFERENCES Cronicos(id\_cr)  ON DELETE CASCADE  ON UPDATE NO ACTION,  FOREIGN KEY (idFarm) REFERENCES Farmacia(idFarm)  ON DELETE CASCADE  ON UPDATE NO ACTION; );  CREATE TABLE Ingresos (  codI INTEGER,  CUIT VARCHAR(11) NOT NULL,  codT INTEGER,  fecha DATE,  estado VARCHAR(10),  PRIMARY KEY (codI),  FOREIGN KEY (CUIT) REFERENCES Laboratorios(CUIT)  ON DELETE NO ACTION  ON UPDATE NO ACTION,  FOREIGN KEY (codT) REFERENCES Transportista(codT)  ON DELETE NO ACTION  ON UPDATE NO ACTION );  CREATE TABLE Medicamentos (  cod INTEGER,  CUIT VARCHAR(11),  nombreComer VARCHAR(80),  descripcion VARCHAR(100),  precio FLOAT,  presentación VARCHAR(100),  stockEnDep INTEGER,  PRIMARY KEY (cod),  FOREIGN KEY (CUIT) REFERENCES Laboratorios(CUIT)  ON DELETE NO ACTION   ON UPDATE CASCADE );  CREATE TABLE Transfiere (  num INTEGER,  cod INTEGER,  cantidad INTEGER,  PRIMARY KEY (num, cod),  FOREIGN KEY (num) REFERENCES Transferencia(num)  ON DELETE NO ACTION  ON UPDATE NO ACTION,  FOREIGN KEY (cod) REFERENCES Medicamentos(cod)  ON DELETE NO ACTION  ON UPDATE NO ACTION  );  CREATE TABLE Incluye1 (  cod INTEGER,  num INTEGER,  cantidad INTEGER,  PRIMARY KEY (num, cod),  FOREIGN KEY (num) REFERENCES Comprobantes(num)  ON DELETE CASCADE  ON UPDATE NO ACTION,  FOREIGN KEY (cod) REFERENCES Medicamentos(cod)  ON DELETE NO ACTION  ON UPDATE NO ACTION );  CREATE TABLE posee1 (  cod INTEGER,  idFarm INTEGER,  cantidad INTEGER,  PRIMARY KEY (cod, idFarm),  FOREIGN KEY (cod) REFERENCES Medicamentos(cod)  ON DELETE NO ACTION  ON UPDATE NO ACTION,  FOREIGN KEY (idFarm) REFERENCES Farmacia(idFarm)  ON DELETE NO ACTION  ON UPDATE NO ACTION );  CREATE TABLE posee2 (  cod INTEGER,  nombre VARCHAR(50),  PRIMARY KEY (cod, nombre),  FOREIGN KEY (cod) REFERENCES Medicamentos(cod)  ON DELETE NO ACTION  ON UPDATE NO ACTION,  FOREIGN KEY (nombre) REFERENCES acc\_terapeuticas(nombre)  ON DELETE NO ACTION  ON UPDATE CASCADE );  CREATE TABLE compuesto (  cod INTEGER,  nombreCientifico VARCHAR(50),  PRIMARY KEY (cod, nombreCientifico),  FOREIGN KEY (cod) REFERENCES Medicamentos(cod)  ON DELETE NO ACTION  ON UPDATE NO ACTION,  FOREIGN KEY (nombreCientifico) REFERENCES Monodroga(nombreCientifico)  ON DELETE NO ACTION  ON UPDATE CASCADE );  CREATE TABLE ingresan (  codI INTEGER,  cod INTEGER,  cantidad INTEGER,  PRIMARY KEY (codI, cod),  FOREIGN KEY (codI) REFERENCES Ingresos(codI)  ON DELETE CASCADE  ON UPDATE NO ACTION,  FOREIGN KEY (cod) REFERENCES Medicamentos(cod)  ON DELETE NO ACTION  ON UPDATE NO ACTION );  CREATE TABLE asociado (  codDiag INTEGER,  id\_af INTEGER,  fechaDiag DATE,  PRIMARY KEY (codDiag, id\_af),  FOREIGN KEY (codDiag) REFERENCES Diagnostico(codDiag)  ON DELETE NO ACTION  ON UPDATE NO ACTION,  FOREIGN KEY (id\_af) REFERENCES Cronicos(id\_cr)  ON DELETE NO ACTION  ON UPDATE NO ACTION );  CREATE TABLE TransportaA (  codT INTEGER,  codpost INTEGER,  PRIMARY KEY (codT, codpost),  FOREIGN KEY (codpost) REFERENCES Ciudad(codpost)  ON DELETE NO ACTION  ON UPDATE NO ACTION,  FOREIGN KEY (codT) REFERENCES Transportista(codT)  ON DELETE NO ACTION  ON UPDATE NO ACTION );  CREATE TABLE Solicita (  num INTEGER,  idFarm INTEGER,  PRIMARY KEY (num, idFarm),  FOREIGN KEY (num) REFERENCES Transferencia(num)  ON DELETE NO ACTION  ON UPDATE NO ACTION,  FOREIGN KEY (idFarm) REFERENCES Farmacia(idFarm)  ON DELETE NO ACTION  ON UPDATE NO ACTION );** |
|  |

# Consultas SQL

**INSERCIÓN, MODIFICACIÓN, BORRADO DE DATOS**

Una vez cargadas las tablas con datos, realizar a criterio del grupo:

* seis consultas distintas para inserción de datos (distintos a los ya cargados en forma masiva),
* seis consultas distintas para modificación de datos,
* seis consultas sql distintas para borrado de filas.

Las consultas pueden ser ejecutadas sobre una misma tabla o distintas, pero todas deben cumplir condiciones medianamente complejas para ejecutarse (por ej. buscar valores en una tabla relacionada mediante una subconsulta).

**CONSULTAS SELECT**

1. Mostrar el ranking de los diez medicamentos con mayor cantidad de ventas en todas las farmacias de la cadena.
2. Listar los códigos y nombres de los medicamentos que fueron vendidos en todas las farmacias
3. Mostrar cantidad de afiliados crónicos y eventuales que compraron medicamentos en la farmacia de Resistencia en la última semana
4. Informar el top de las 5 farmacias que solicitaron mayores cantidades de amoxidal duo en los últimos 15 días.
5. Listado de farmacias con la menor cantidad de transferencias pendientes.
6. Identificar para una farmacia determinada cuales son los medicamentos sin stock en la misma pero con stock en deposito central
7. ¿Cuál es la empresa de transporte con mayor actividad en el último mes?
8. Informar el monto total de ventas por farmacia en el último trimestre ordenado en forma descendente.

## INSERCIÓN, MODIFICACIÓN, BORRADO DE DATOS

### Consultas UPDATE

1. Aplicar el descuento según el tipo de afiliado al precio total de todos los comprobantes

|  |
| --- |
| **UPDATE comprobantes c JOIN (  SELECT c.num,  CASE  WHEN c.id\_af IS NULL THEN (c.subtotal - (c.subtotal \* (cr.descuento/100)))  ELSE (c.subtotal - (c.subtotal \* (af.descuento/100)))  END AS total  FROM comprobantes c NATURAL JOIN afiliado af NATURAL JOIN cronicos cr  GROUP BY c.num, c.id\_af ) t ON c.num = t.num SET c.total = t.total;** |

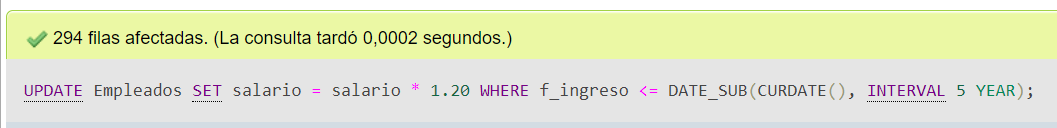
**Texto

Descripción generada automáticamente**

****

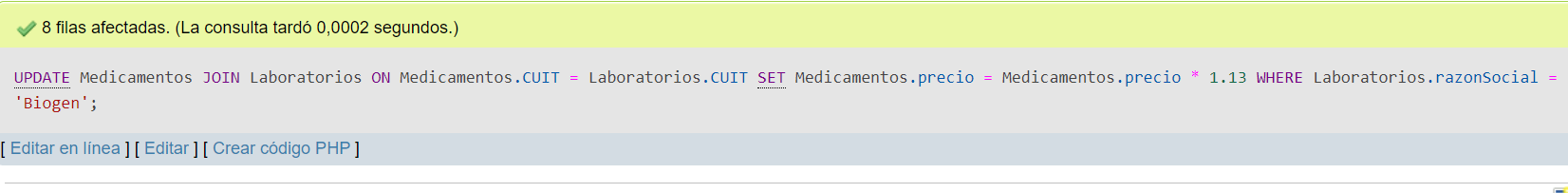
1. A aquellos empleados que tienen una antigüedad mayor a 5 años aumentar un 20% eL salario

|  |
| --- |
| **UPDATE** Empleados **SET** salario = salario \* 1.20 **WHERE** f\_ingreso <= **DATE\_SUB**(**CURDATE**(), INTERVAL 5 **YEAR**); |



1. A los medicamentos comercializados por Biogen incrementar su precio en un 13%.

|  |
| --- |
| **UPDATE** Medicamentos **JOIN** Laboratorios **ON** Medicamentos.CUIT = Laboratorios.CUIT **SET** Medicamentos.precio = Medicamentos.precio \* 1.13 **WHERE** Laboratorios.razonSocial = 'Biogen'; |



1. A aquellas farmacias que le corresponden el día de guardia “Lunes” cambiarlo por el día “Miércoles” o viceversa.

|  |
| --- |
| **UPDATE** farmacia f **JOIN** (  **SELECT** f.idFarm,  **CASE**  **WHEN** f.diasGuardia = 'Lunes' **THEN** f.diasGuardia = 'Miércoles'  **WHEN** f.diasGuardia = 'Miércoles' **THEN** f.diasGuardia = 'Lunes'  **END** **AS** diasGuardia  **FROM** farmacia f ) t **ON** f.idFarm = t.idFarm **SET** f.diasGuardia = t.diasGuardia; |

Texto

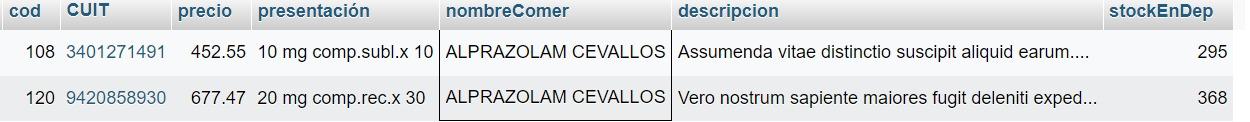
Descripción generada automáticamente

1. Realizar un descuento del 15% a los 10 medicamentos menos vendidos de todas las farmacias.

|  |
| --- |
| **UPDATE** medicamentos m **JOIN**(  **SELECT** m.cod, (m.precio\*0.9) **as** precio  **FROM** medicamentos m **NATURAL** **JOIN** incluye1 i1  **GROUP** **BY** i1.cod  **HAVING** **SUM**(i1.cantidad) =  (**SELECT** **MIN**(aux.vendido) **as** vendido  **FROM** (**SELECT** **SUM**(i.cantidad) **as** vendido  **FROM** incluye1 i  **GROUP** **BY** i.cod) aux)  ) aux2 **ON** m.cod = aux2.cod **SET** m.precio = aux2.precio; |



Antes:



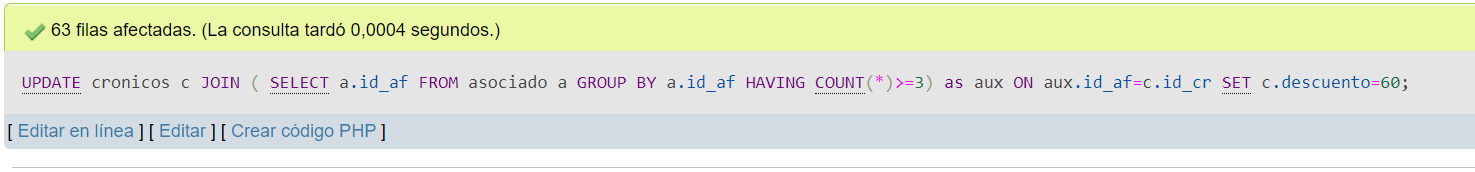
Después:

Interfaz de usuario gráfica

Descripción generada automáticamente

1. A aquellos socios Crónicos que tengan más de 3 diagnósticos, bajarles el descuento a un 60%.

|  |
| --- |
| **UPDATE** cronicos c **JOIN** ( **SELECT** a.id\_af  **FROM** asociado a   **GROUP** **BY** a.id\_af  **HAVING** **COUNT**(\*)>=3) **as** aux **ON** aux.id\_af=c.id\_cr **SET** c.descuento=60; |



### Consultas INSERT

1. Insertar en transportaA una nueva relación que vincule a la ciudad de resistencia con uno de los transportista que menor cantidad de envíos haya realizado en el último trimestre.

|  |
| --- |
| **INSERT** **INTO** transportaa (codT, codpost) **VALUES**((**SELECT** i.codT **FROM** ingresos i **WHERE** (30\*3) >= **DATEDIFF**(**CURRENT\_DATE**, i.fecha) **AND** i.estado = 'finalizado' **GROUP** **BY** i.codT **HAVING** **COUNT**(\*) = (  **SELECT** **MIN**(envios)  **FROM** (  **SELECT** **COUNT**(\*) **as** envios  **FROM** ingresos  **WHERE** 30 >= **DATEDIFF**(**CURRENT\_DATE**, fecha) **AND** estado = 'finalizado'  **GROUP** **BY** codT  ) **as** aux ) **ORDER** **BY** i.codT **LIMIT** 1 ), (**SELECT** c.codpost **FROM** ciudad c **WHERE** c.nombre = 'Resistencia') ); |

Texto

Descripción generada automáticamente

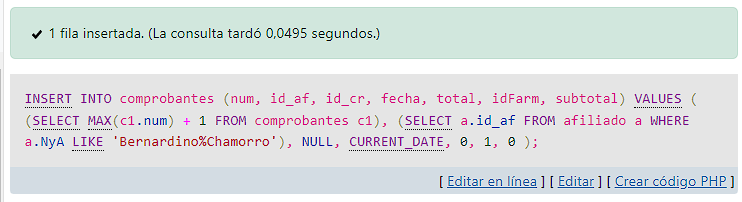
1. Si existe algún paciente crónico con 4 o más diagnósticos, crear un nuevo diagnóstico llamado Por morir y asociar a todos los que estén en esa condición

|  |
| --- |
| -- Obtener el siguiente número de diagnóstico **SELECT** **MAX**(codDiag) + 1 **INTO** @siguienteCodDiag **FROM** Diagnostico;  -- Insertar el nuevo diagnóstico "Por morir" si no existe **INSERT** **INTO** Diagnostico (codDiag, nombre) **SELECT** @siguienteCodDiag, 'Por morir' **FROM** dual **WHERE** **NOT** **EXISTS** (  **SELECT** \*  **FROM** Diagnostico  **WHERE** nombre = 'Por morir' );  -- Asociar el nuevo diagnóstico a los pacientes crónicos si existe **INSERT** **INTO** asociado (codDiag, id\_af, fechaDiag) **SELECT** @siguienteCodDiag, id\_af, **CURDATE**() **FROM** asociado a **GROUP** **BY** a.id\_af **HAVING** **COUNT**(\*) >= 4 |



1. Crear un nuevo comprobante para Bernardino Chamorro que realizo una compra en la farmacia con idFarm = 1

|  |
| --- |
| **INSERT** **INTO** comprobantes (**num**, id\_af, id\_cr, fecha, total, idFarm, subtotal) **VALUES** (  (**SELECT** **MAX**(c1.num) + 1 **FROM** comprobantes c1),  (**SELECT** a.id\_af **FROM** afiliado a **WHERE** a.NyA **LIKE** 'Bernardino%Chamorro'),  **NULL**, **CURRENT\_DATE**, 0, 1, 0  ); |



1. Insertar nuevo laboratorio

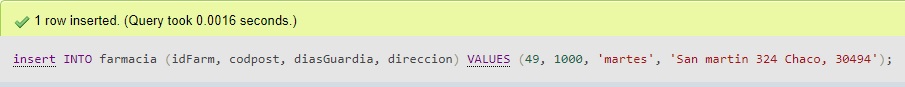
|  |
| --- |
| **INSERT INTO Laboratorios (CUIT, razonSocial, domicilio) VALUES ('1234567890', 'LabUrquiza', '420 pasaje Bosch, Corrientes, Argentina');** |

**Interfaz de usuario gráfica, Texto, Aplicación, Sitio web

Descripción generada automáticamente**

1. Insertar nueva Farmacia

|  |
| --- |
| **INSERT** **INTO** farmacia (idFarm, codpost, diasGuardia, direccion) **VALUES** (49, 1500, 'martes', 'San martin 324 Chaco'); |

****

1. Crear un nuevo ingreso de 10 unidades del medicamento IBUPROFENO ILAB 600 comercializado por el laboratorio BioGen y que será transportado por Rivas PLC.

|  |
| --- |
| **SELECT** **MAX**(codI) **INTO** @ultimoCodI **FROM** ingresos; **INSERT** **INTO** ingresan (codI, cod, cantidad) **INSERT** **INTO** ingresos (codI, CUIT, codT, fecha, estado) **VALUES** (  (**SELECT** **MAX**(i.codI) + 1 **FROM** ingresos i),  (**SELECT** l.CUIT  **FROM** laboratorios l  **WHERE** l.razonSocial = 'BioGen'),  (**SELECT** t.codT  **FROM** transportista t  **WHERE** t.razonSocial = 'Rivas PLC'),  **CURRENT\_DATE**,  'pendiente' ); **SELECT** @ultimoCodI,  (**SELECT** m.cod  **FROM** medicamentos m  **WHERE** m.nombreComer = 'IBUPROFENO ILAB 600')  ,10; |



### 

### Consultas DELETE

1. Borrar comprobantes que tengan una antigüedad mayor a 5 años y estén asociados a un afiliado eventual.

|  |
| --- |
| DELETE FROM comprobantes c WHERE datediff(CURRENT\_DATE, c.fecha) >= (365\*5) AND c.id\_af IS NOT NULL; |

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

1. Eliminar las monodrogas que no compongan ningún medicamento.

|  |
| --- |
| **DELETE** **FROM** Monodroga **WHERE** nombreCientifico **NOT** **IN** (  **SELECT** nombreCientifico **FROM** compuesto ); |

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

1. La sucursal de Resistencia ha sido dada de baja. Eliminar empleados vinculados a la farmacia de la ciudad de Resistencia que hayan trabajado por menos de dos años:

|  |
| --- |
| **DELETE** **FROM** Empleados **WHERE** idFarm **IN** (**SELECT** idFarm **FROM** Farmacia **WHERE** codpost **IN** (**SELECT** codpost **FROM** Ciudad **WHERE** nombre = 'Resistencia'))  **AND** **DATEDIFF**(**CURDATE**(), f\_ingreso) < 730; |

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

1. Borrar todos los afiliados que no tienen ninguna enfermedad crónica y que se afiliaron hace más de 5 años:

|  |
| --- |
| **DELETE** **FROM** Afiliado **WHERE** id\_af **NOT** **IN** (**SELECT** id\_af **FROM** Asociado) **AND** f\_ing < **DATE\_SUB**(**CURDATE**(), INTERVAL 5 **YEAR**); |

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

1. Eliminar ingresos que tengan el estado finalizado y sean de hace más de 365 días:

|  |
| --- |
| DELETE FROM ingresos i  WHERE datediff(CURRENT\_DATE, i.fecha)>=365 AND i.estado = 'pendiente'; |

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

1. Borrar todos los afiliados que no tienen ninguna enfermedad crónica y que viven en una localidad específica:

|  |
| --- |
| **DELETE** **FROM** Afiliado **WHERE** id\_af **NOT** **IN** (**SELECT** id\_af **FROM** Asociado) **AND** localidad = 'Sevilla'; |

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

## Consultas SELECT

1. Mostrar el ranking de los diez medicamentos con mayor cantidad de ventas en todas las farmacias de la cadena.

|  |
| --- |
| **SELECT m.cod, m.nombreComer, SUM(i.cantidad) as Vendido FROM medicamentos m NATURAL JOIN incluye1 i NATURAL JOIN comprobantes c GROUP BY m.cod ORDER BY Vendido DESC LIMIT 10;** |

****

1. Listar los códigos y nombres de los medicamentos que fueron vendidos en todas las farmacias.

|  |
| --- |
| **SELECT m.cod FROM medicamentos m WHERE NOT EXISTS(  (SELECT f.idFarm  FROM farmacia f)  EXCEPT  (SELECT c.idFarm  FROM comprobantes c NATURAL JOIN incluye1 i  WHERE i.cod = m.cod) ) ORDER BY m.cod ASC;** |

****

1. Mostrar la cantidad de afiliados crónicos y eventuales que compraron medicamentos en la farmacia de Resistencia en la última semana.

|  |
| --- |
| **SELECT** c1.nombre ,**COUNT**(c.id\_af) **as** 'Ventas a eventuales', **COUNT**(c.id\_cr) **as** 'Ventas a Cronicos' **FROM** comprobantes c **NATURAL** **JOIN** farmacia f **NATURAL** **JOIN** ciudad c1 **WHERE** c1.nombre = 'resistencia' **AND** 7 >= **DATEDIFF**(**CURRENT\_DATE**, c.fecha); |

Interfaz de usuario gráfica, Texto, Aplicación, Word

Descripción generada automáticamente

1. Informar el top de las 5 farmacias que solicitaron mayores cantidades de amoxidal duo en los últimos 15 días.

|  |
| --- |
| **SELECT** f.idFarm, t.cantidad **FROM** farmacia f **NATURAL** **JOIN** solicita s **NATURAL** **JOIN** (**SELECT** t1.num ,**DATEDIFF**(**CURRENT\_DATE**, t1.fecha) **AS** dias **FROM** transferencia t1)t1 **NATURAL** **JOIN** transfiere t **NATURAL** **JOIN** medicamentos m **WHERE** m.nombreComer = 'AMOXIDAL DUO' **AND** 15 >= t1.dias **ORDER** **BY** t.cantidad **DESC** **LIMIT** 5; |

Tabla

Descripción generada automáticamente

1. Listado de farmacias con la menor cantidad de transferencias pendientes.

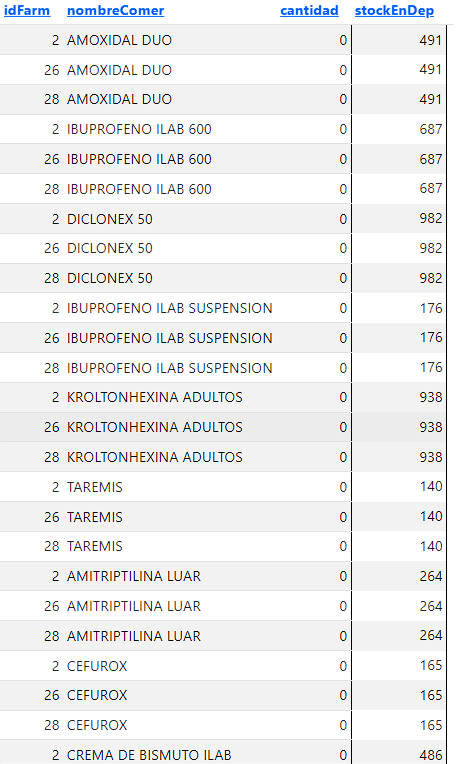
|  |
| --- |
| **SELECT** s.idFarm, **count**(\*) **as** 'cantidad pendiente' **FROM** transferencia t **NATURAL** **JOIN** solicita s **WHERE** t.estado = 'pendiente' **GROUP** **BY** s.idFarm **HAVING** **COUNT**(\*) = (**select** **MIN**(a.cant) **from** (**SELECT** s.idFarm,**count**(\*) **AS** cant **FROM** transferencia t **NATURAL** **JOIN** solicita s **WHERE** t.estado = 'pendiente' **GROUP** **BY** s.idFarm) a); |

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

1. Identificar para una farmacia determinada cuales son los medicamentos sin stock en la misma pero con stock en deposito central

|  |
| --- |
| **SELECT** p1.idFarm, m.nombreComer, p1.cantidad, m.stockEnDep **FROM** medicamentos m **NATURAL** **JOIN** posee1 p1 **WHERE** p1.Cantidad = 0 **AND** m.stockEnDep > 0; |



1. ¿Cuál es la empresa de transporte con mayor actividad en el último mes?

|  |
| --- |
| **SELECT** i.codT, **COUNT**(\*) **as** envios **FROM** ingresos i **WHERE** 30 >= **DATEDIFF**(**CURRENT\_DATE**, i.fecha) **AND** i.estado = 'finalizado' **GROUP** **BY** i.codT **HAVING** **COUNT**(\*) = (  **SELECT** **MAX**(envios)  **FROM** (  **SELECT** **COUNT**(\*) **as** envios  **FROM** ingresos  **WHERE** 30 >= **DATEDIFF**(**CURRENT\_DATE**, fecha) **AND** estado = 'finalizado'  **GROUP** **BY** codT  ) **as** aux ); |



1. Informar el monto total de ventas por farmacia en el último trimestre ordenado en forma descendente.

|  |
| --- |
| **SELECT** c.idFarm, **ROUND**(**SUM**(c.total), 0) **AS** ganancias **FROM** comprobantes c **WHERE** 90 >= **DATEDIFF**(**CURRENT\_DATE**,c.fecha) **GROUP** **BY** c.idFarm **ORDER** **by** ganancias **DESC**; |

