

Trabajo Práctico Integrador

Escenario Farmacia



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

Contenido

Escenario	2
Sistema Gestor de Base de Datos	4
Características principales.....	4
Conceptos de la unidad 1	6
Análisis del escenario y Modelado	7
Diagrama Entidad-Relación planteado:	7
Consideraciones:	7
Esquema Relacional.....	8
Diccionario de datos	10
Esquema Físico	15
Consultas SQL	21
INSERCIÓN, MODIFICACIÓN, BORRADO DE DATOS	22
Consultas UPDATE.....	22
Consultas INSERT.....	27
Consultas DELETE	32
Consultas SELECT.....	34

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

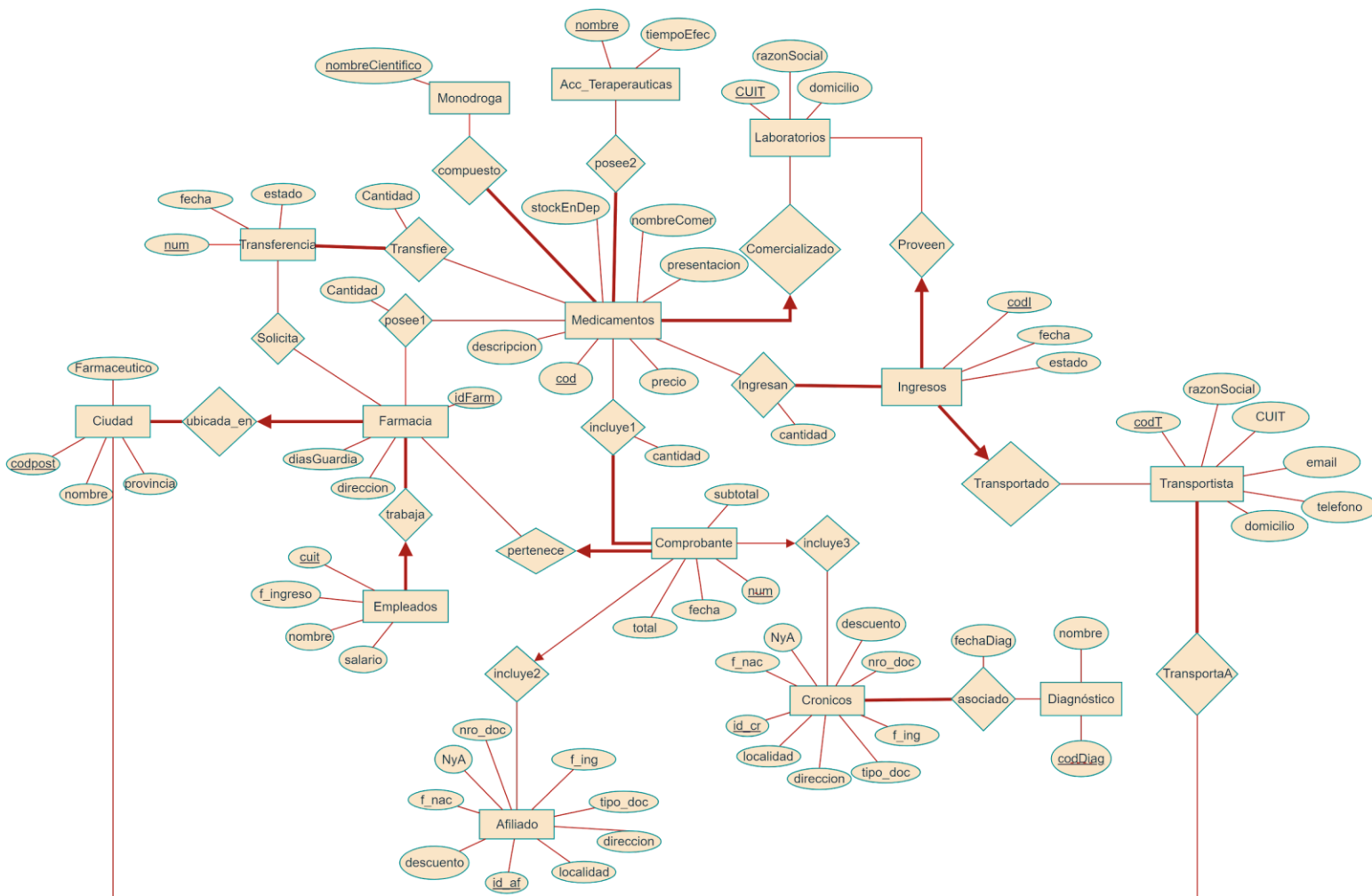
comandos (CLI) de MySQL, así como interfaces gráficas como MySQL Workbench y phpMyAdmin. También ofrece utilidades para la importación/exportación de datos, copias de seguridad y restauración, y optimización del rendimiento

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:



Link: https://app.diagrams.net/?page-id=R2IEEEUBdFMjLlhrx00&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: integer, 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
codI	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](#)

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;
```

✓ 19999 filas afectadas. (La consulta tardó 0,6421 segundos.)

```
UPDATE comprobantes c JOIN ( SELECT c.num, CASE WHEN c.id_af IS NULL THEN SUM(m.precio * i.cantidad) *
0.3 ELSE SUM(m.precio * i.cantidad) * 0.8 END AS total FROM medicamentos m JOIN incluye1 i ON m.cod =
i.cod JOIN comprobantes c ON c.num = i.num GROUP BY c.num, c.id_af ) t ON c.num = t.num SET c.total =
t.total;
```

num	id_af	id_cr	fecha	total	idFarm	subtotal
1	11590	NULL	2023-03-09	5039.15	8	6298.94
2	NULL	10389	2022-09-17	5898.04	25	19660.2
3	NULL	1686	2022-09-29	769.053	44	2563.51
4	NULL	4505	2021-10-06	6307.29	6	21024.3
5	5461	NULL	2023-01-01	6700.46	35	8375.58
6	10433	NULL	2023-06-07	1640.08	46	2050.1
7	2713	NULL	2022-05-31	8397.17	28	10496.5
8	NULL	1937	2022-03-05	4637.13	26	15457.1
9	NULL	193	2022-10-28	432.696	2	1442.32
10	5078	NULL	2023-05-05	22.56	15	28.2
11	7126	NULL	2023-01-04	7948.72	9	9935.9
12	NULL	1213	2021-07-26	4842.89	23	16143
13	5489	NULL	2022-11-19	14626.8	10	18283.5
14	NULL	2558	2021-05-30	5202.62	1	17342.1
15	NULL	3576	2021-05-16	1354.69	48	4515.64
16	5956	NULL	2022-04-19	14153.2	24	17691.5
17	NULL	8569	2022-09-23	7439.07	37	24796.9
18	8631	NULL	2022-05-13	31144.9	29	38931.1
19	10184	NULL	2022-06-01	10676.6	6	13345.8

2. 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);
```

✓ 294 filas afectadas. (La consulta tardó 0,0002 segundos.)

```
UPDATE Empleados SET salario = salario * 1.20 WHERE f_ingreso <= DATE_SUB(CURDATE(), INTERVAL 5 YEAR);
```

3. 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';
```

✓ 8 filas afectadas. (La consulta tardó 0,0002 segundos.)

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

[Editar en línea] [Editar] [Crear código PHP]

4. 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;
```

✓ 48 filas afectadas. (La consulta tardó 0,0031 segundos.)

```
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;
```

[[Editar en línea](#)] [[Editar](#)] [[Crear código PHP](#)]

- 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:

cod	CUIT	precio	presentación	nombreComer
108	3401271491	452.55	10 mg comp.subl.x 10	ALPRAZOLAM CEVALLOS

Después:

cod	CUIT	precio	presentación	nombreComer
108	<u>3401271491</u>	407.295	10 mg comp.subl.x 10	ALPRAZOLAM CEVALLOS

✓ 1 fila afectada. (La consulta tardó 0,0003 segundos.)

```
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;
```

[Editar en línea] [Editar] [Crear código PHP]

- 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;
```

✔ 63 filas afectadas. (La consulta tardó 0,0004 segundos.)

```
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;
```

[\[Editar en línea \]](#) [\[Editar \]](#) [\[Crear código PHP \]](#)

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')
);
```

✓ 1 fila insertada. (La consulta tardó 0,0035 segundos.)

```
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') );
```

[[Editar en línea](#)] [[Editar](#)] [[Crear código PHP](#)]

2. 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
```

✓ MySQL ha devuelto un conjunto de valores vacío (es decir: cero columnas). (La consulta tardó 0,0744 segundos.)

```
-- Obtener el siguiente número de diagnóstico SELECT MAX(codDiag) + 1 INTO  
@siguienteCodDiag FROM Diagnostico;
```

[[Editar en línea](#)] [[Editar](#)] [[Crear código PHP](#)]

Operaciones sobre los resultados de la consulta

 Crear vista

✓ 1 fila insertada. (La consulta tardó 0,0087 segundos.)

```
-- 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' );
```

[[Editar en línea](#)] [[Editar](#)] [[Crear código PHP](#)]

✓ 3 filas insertadas. (La consulta tardó 0,0106 segundos.)

```
-- 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;
```

[[Editar en línea](#)] [[Editar](#)] [[Crear código PHP](#)]

3. Crear un nuevo comprobante para Bernardino Chamorro que realice 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 fila insertada. (La consulta tardó 0,0495 segundos.)

```
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 );
```

[[Editar en línea](#)] [[Editar](#)] [[Crear código PHP](#)]

4. Insertar nuevo laboratorio

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

✓ 1 fila insertada. (La consulta tardó 0,0031 segundos.)

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

5. Insertar nueva Farmacia

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

✓ 1 row inserted. (Query took 0.0016 seconds.)

```
insert INTO farmacia (idFarm, codpost, diasGuardia, direccion) VALUES (49, 1000, 'martes', 'San martin 324 Chaco, 30494');
```

6. 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;
```

✓ 1 fila insertada. (La consulta tardó 0,0021 segundos.)

```
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' );
```

[[Editar en línea](#)] [[Editar](#)] [[Crear código PHP](#)]


✓ MySQL ha devuelto un conjunto de valores vacío (es decir: cero columnas). (La consulta tardó 0,0003 segundos.)

```
SELECT MAX(codI) INTO @ultimoCodI FROM ingresos;
```

☐ Perfilando

[[Editar en línea](#)] [[Editar](#)] [[Explicar SQL](#)] [[Crear código PHP](#)] [[Actualizar](#)]

Operaciones sobre los resultados de la consulta

 Crear vista

✓ 1 fila insertada. (La consulta tardó 0,0473 segundos.)

```
INSERT INTO ingresan (codI, cod, cantidad) SELECT @ultimoCodI, (SELECT m.cod FROM  
medicamentos m WHERE m.nombreComer = 'IBUPROFENO ILAB 600') ,10;
```

[[Editar en línea](#)] [[Editar](#)] [[Crear código PHP](#)]

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;
```

✓ 1 fila eliminada. (La consulta tardó 0,0269 segundos.)

```
DELETE FROM comprobantes c WHERE datediff(CURRENT_DATE, c.fecha) >= (365*5) AND c.id_af
IS NOT NULL;
```

[[Editar en línea](#)] [[Editar](#)] [[Crear código PHP](#)]

2. Eliminar las monodrogas que no compongan ningún medicamento.

```
DELETE FROM Monodroga
WHERE nombreCientifico NOT IN (
    SELECT nombreCientifico FROM compuesto
);
```

✓ 5 filas eliminadas. (La consulta tardó 0,0003 segundos.)

```
DELETE FROM Monodroga WHERE nombreCientifico NOT IN ( SELECT nombreCientifico FROM
compuesto );
```

[[Editar en línea](#)] [[Editar](#)] [[Crear código PHP](#)]

3. 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;
```

✓ 3 filas eliminadas. (La consulta tardó 0,0004 segundos.)

```
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;
```

[[Editar en línea](#)] [[Editar](#)] [[Crear código PHP](#)]

4. 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);
```

✓ 27 filas eliminadas. (La consulta tardó 0,0019 segundos.)

```
DELETE FROM Afiliado WHERE id_af NOT IN (SELECT id_af FROM Asociado) AND f_ing <  
DATE_SUB(CURDATE(), INTERVAL 5 YEAR);
```

[[Editar en línea](#)] [[Editar](#)] [[Crear código PHP](#)]

5. 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';
```

✓ 13 filas eliminadas. (La consulta tardó 0,0034 segundos.)

```
DELETE FROM ingresos i WHERE datediff(CURRENT_DATE, i.fecha)>=365 AND i.estado =  
'pendiente';
```

[[Editar en línea](#)] [[Editar](#)] [[Crear código PHP](#)]

6. 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';
```

✓ 511 filas eliminadas. (La consulta tardó 0,0013 segundos.)

```
DELETE FROM Afiliado WHERE id_af NOT IN (SELECT id_af FROM Asociado) AND localidad =  
'Sevilla';
```

[[Editar en línea](#)] [[Editar](#)] [[Crear código PHP](#)]

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;
```

<u>cod</u>	<u>nombreComer</u>	<u>Vendido</u> ↕ 1
92	OMEFAR	3157
29	AMLODIPINA ILAB	3091
1	AMOXIDAL DUO	3055
48	PANAPROST	3023
206	FABOACID PANTO	3017
8	CEFUROX	3016
45	CLORATEN	3014
146	KERAMIX RAPID	2993
50	AIRCOSALM	2991
101	KETOROLAC 20 MG	2960

2. 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;
```

← T → cod ↕ 1

<input type="checkbox"/>  Editar  Copiar  Borrar	2
<input type="checkbox"/>  Editar  Copiar  Borrar	3
<input type="checkbox"/>  Editar  Copiar  Borrar	4
<input type="checkbox"/>  Editar  Copiar  Borrar	5
<input type="checkbox"/>  Editar  Copiar  Borrar	6
<input type="checkbox"/>  Editar  Copiar  Borrar	7
<input type="checkbox"/>  Editar  Copiar  Borrar	8
<input type="checkbox"/>  Editar  Copiar  Borrar	9
<input type="checkbox"/>  Editar  Copiar  Borrar	10
<input type="checkbox"/>  Editar  Copiar  Borrar	11
<input type="checkbox"/>  Editar  Copiar  Borrar	12
<input type="checkbox"/>  Editar  Copiar  Borrar	13
<input type="checkbox"/>  Editar  Copiar  Borrar	14
<input type="checkbox"/>  Editar  Copiar  Borrar	15
<input type="checkbox"/>  Editar  Copiar  Borrar	16
<input type="checkbox"/>  Editar  Copiar  Borrar	17
<input type="checkbox"/>  Editar  Copiar  Borrar	18
<input type="checkbox"/>  Editar  Copiar  Borrar	19
<input type="checkbox"/>  Editar  Copiar  Borrar	20
<input type="checkbox"/>  Editar  Copiar  Borrar	21
<input type="checkbox"/>  Editar  Copiar  Borrar	22
<input type="checkbox"/>  Editar  Copiar  Borrar	23

3. 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);
```

nombre	Ventas a eventuales	Ventas a Cronicos
Resistencia	2	1

4. 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;
```

idFarm cantidad ↕ 1

15	999
14	919
36	611
9	289
26	89

5. 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);
```

idFarm cantidad pendiente

37	28
24	28

6. 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;
```

<u>idFarm</u>	<u>nombreComer</u>	<u>cantidad</u>	<u>stockEnDep</u>
2	AMOXIDAL DUO	0	491
26	AMOXIDAL DUO	0	491
28	AMOXIDAL DUO	0	491
2	IBUPROFENO ILAB 600	0	687
26	IBUPROFENO ILAB 600	0	687
28	IBUPROFENO ILAB 600	0	687
2	DICLONEX 50	0	982
26	DICLONEX 50	0	982
28	DICLONEX 50	0	982
2	IBUPROFENO ILAB SUSPENSION	0	176
26	IBUPROFENO ILAB SUSPENSION	0	176
28	IBUPROFENO ILAB SUSPENSION	0	176
2	KROLTONHEXINA ADULTOS	0	938
26	KROLTONHEXINA ADULTOS	0	938
28	KROLTONHEXINA ADULTOS	0	938
2	TAREMIS	0	140
26	TAREMIS	0	140
28	TAREMIS	0	140
2	AMITRIPTILINA LUAR	0	264
26	AMITRIPTILINA LUAR	0	264
28	AMITRIPTILINA LUAR	0	264
2	CEFUROX	0	165
26	CEFUROX	0	165
28	CEFUROX	0	165
2	CREMA DE BISMUTO ILAB	0	486

7. ¿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
);
```

codT **envios**

<u>2</u>	2
----------	---

8. 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;
```


[idFarm](#) [ganancias](#) ↕ 1

8	489225
26	477443
10	476922
12	471634
23	452283
14	446192
34	439395
42	438565
2	435499
28	435367
25	430380
38	417791
31	414464
4	399675
1	392867
36	392015
41	391173
47	374466
6	361124
21	352543
22	349857
3	346382
48	334398
9	333017
35	332693

[idFarm](#) [ganancias](#) ↕ 1

15	324078
13	322235
17	317900
27	317787
5	317180
24	314672
19	312953
20	308830
29	301455
18	298515
16	292661
32	279113
45	275925
30	273551
7	271630
43	269397
37	268611
46	265060
39	264628
44	258535
33	256439
40	245026
11	242756