Profesor
L. en C.C. Erick Orlando Matla Cruz
Ayudantes
L. en C.C. Efraín Hipólito Chamú
L. en C.C. Anahí Quiroz Jiménez
L. en C.C. Karen Monserrat Zavala Correa
Grupo: 9119

Práctica 06

Lenguaje de Manipulación de Datos (DML)

1. Objetivo General

Insertar, actualizar, consultar y eliminar los datos de una base de datos.

2. Objetivos Secundarios

- Conocer la sintaxis correcta para realizar funciones de manipulación de datos dentro de un SMBD.
- Introducir el concepto de consulta básica.

3. Introducción

Una base de datos, cualquiera que ésta sea, por ejemplo: un archivero, tiene entre sus finalidades, el almacenar datos de una manera ordenada para su posterior consulta, actualización, inserción y eliminación, dependiendo de las necesidades del cliente. Esta misma finalidad aplica para las bases de datos computacionales, es por ello que resulta fundamental conocer el mecanismo para realizar cualquiera de estas cuatro actividades dentro del SMBD que se encarga de la administración de la base de datos sobre la cual se pretende trabajar.

Como se vio en *Lenguaje de Definición de Datos (DDL)* (Práctica 05), el lenguaje que se propone en estas prácticas para comunicarse con él SMBD es SQL. Este lenguaje maneja varias palabras reservadas para la parte de DDL pero también para la parte de DML.

Para su estudio, podemos dividir las palabras reservadas de SQL para DML en cuatro categorías:

1. Inserción de datos
2. Actualización de datos
3. Consulta de datos
4. Eliminación de registros

3.1. Inserción de datos

Una base de datos relacional contiene tablas con columnas que poseen características singulares y que se relacionan entre sí para modelar un problema de la realidad. Estas tablas han sido construidas y detalladas mediante el *Lenguaje de Definición de Datos (DDL)* y cuestiones de *Integridad* (Práctica 05) para almacenar información en forma de registros. Mediante DDL es posible definir el esquema que permitirá almacenar los datos.

En la Tabla 6.1 se puede observar una tabla que almacenará por cada registro la temperatura máxima y mínima de una ciudad.

Tabla 6.1 - Esquema de la tabla Temperatura_Ciudad.

Ciudad	*Temperatura Mínima*	*Temperatura Máxima*

La tabla recién creada se encuentra vacía, haciendo uso del DML es posible agregar renglones a través del SMBD. La Tabla 6.2 se muestra ya a Temperatura_Ciudad con algunos registros.

Tabla 6.2 - Tabla con dos inserciones.

Ciudad	*Temperatura Mínima*	*Temperatura Máxima*
Los Angeles	12°C	28°C
Washington	3°C	15°C

Para llevar a cabo la inserción de datos a través de un SMBD, SQL trabaja con palabras reservadas cuya sintaxis se muestra en la Figura 6.1.

```
INSERT INTO nombre_tabla (columna1, columna2, ...,columnaN)
VALUES (valor1, valor2, ...,valorN);
```

Figura 6.1 - Sintaxis de inserción de SQL.

En esta sintaxis encontramos:

1. INSERT INTO
   Palabras reservadas que en conjunto, comienzan la instrucción de insertar un registro en una tabla específica.
2. nombre_tabla
   Es el nombre de la tabla destino de la inserción, es decir, la tabla donde se insertará el nuevo registro.
3. (
   El símbolo de apertura de paréntesis, después del nombre de la tabla, sirve para indicar que comenzará con la lista de columnas donde se insertarán los nuevos

valores (el nuevo registro). En esta lista podemos definir de manera personalizada el orden en el que queremos que se inserten los datos del registro.

4. columna1, columna2, ... , columnaN
   Es el listado de las columnas donde se insertarán los datos de un registro. El orden en el que se escriban las columnas es el orden en el que se insertarán los nuevos datos.
5.)
   El símbolo de cierre de paréntesis, indica el término de la lista de columnas donde se insertarán los nuevos valores.
6. VALUES
   Palabra reservada para indicarle al SMBD que la lista que sigue a continuación son los valores del nuevo registro.
7. (
   El símbolo de apertura de paréntesis, indica la apertura de la lista de valores que se insertarán para formar el nuevo registro.
8. valor1, valor2, ... , valorN
   Es el listado de valores que se insertarán para formar el nuevo registro. El orden en el que se escriban estos valores es el orden en el que se insertarán los nuevos datos.
9.)
   El símbolo de cierre de paréntesis, indica el cierre de la lista de valores que se insertarán para formar el nuevo registro.
10. ;
    Indica la finalización de la instrucción.

Como ejemplo, se muestra la sintaxis de la inserción de un registro en la tabla Temperatura_Ciudad. Ver Figura 6.2.

```
INSERT INTO Temperatura_Ciudad (Ciudad,Temperatura_Minima,Temperatura_Maxima)
VALUES ('Los Angeles','12ºC','28ºC');
```

Figura 6.2 - Ejemplo de inserción en la tabla Temperatura_Ciudad.

Cabe mencionar que los elementos descritos en los puntos 3, 4 y 5 son opcionales, en caso de no colocarlos, el SMBD asume que se utilizarán todas las columnas de la tabla en el orden en el que se encuentran en definidos de izquierda a derecha. En el caso que se requiera insertar los datos en columnas específicas, la lista de las columnas puede variar en orden, así como en el número de columnas. Por ejemplo, si quisiéramos insertar solo el nombre de la ciudad, el código se vería como se muestra en la Figura 6.3.

```
INSERT INTO Temperatura_Ciudad (Ciudad) VALUES ('Ciudad de México');
```

Figura 6.3 - Ejemplo de inserción en la tabla Temperatura_Ciudad sobre un sólo atributo.

3.2. **Consulta de datos**

Los datos almacenados en una base de datos, no generan información por sí solos. Es por eso que se requiere consultarlos, ordenarlos, contarlos, sumarlos, etc., para generar dicha información.

Por ejemplo, retomemos la tabla de las ciudades y sus temperaturas. Si trabajáramos para un centro climatológico, podríamos querer saber qué ciudades han presentado las temperaturas más bajas para poder hacer predicciones sobre el clima.

Como otro ejemplo, supongamos que somos los dueños de una pizzería de entrega a domicilio. Dicha pizzería cuenta con una base de datos que almacena todos los pedidos y sus detalles, así como los clientes con sus teléfonos y domicilios. Para incrementar la utilidad se ha decidido invertir en promociones y publicidad.

Conocer las características de las pizzas que más se consumen y el tipo de cliente que más ordena, es primordial para ofrecer las mejores promociones e invertir correctamente en publicidad. Para lograr esto, necesitaríamos hacer una explotación específica de los datos.

SQL ofrece la capacidad de poder realizar consultas con palabras reservadas. A continuación se presenta la sintaxis básica para realizar consultas. Ver Figura 6.4.

```
SELECT nombre_columna
FROM nombre_tabla
WHERE condicion;
```

Figura 6.4 - Sintaxis de consulta de SQL.

En esta sintaxis encontramos:

1. SELECT
   Palabra reservada para indicar que atributos se mostrarán como resultado de la consulta.
2. nombre_columna
   Es el nombre de la columna, o columnas separadas por comas, que queremos consultar.
3. FROM

Palabra reservada para indicarle al SMBD la tabla de la cual obtener la columna que necesitamos.

4. nombre_tabla
   Es el nombre de la tabla que contiene la columna o columnas que solicitamos.
5. WHERE
   Palabra reservada para indicar bajo qué condición seleccionaremos los registros que queremos consultar.
6. condicion
   Es una característica o conjunto de características que tiene que cumplir los registros para ser seleccionados.
7. ;
   Indica la finalización de la instrucción.

La Figura 6.5 muestra un ejemplo del uso de esta estructura de consulta.

```
SELECT Ciudad, Temperatura Mínima, Temperatura Máxima
FROM Temperatura_Ciudad
WHERE Ciudad = 'Washington';
```

Figura 6.5 - Ejemplo de consulta a la tabla Temperatura_Ciudad.

El resultado de la anterior consulta se muestra en la Tabla 6.3.

Tabla 6.3. Ejemplo de consulta a la tabla Temperatura_Ciudad.

Ciudad	*Temperatura Mínima*	*Temperatura Máxima*
Los Angeles	12°C	28°C

3.3. **Actualización de datos**

Aunque el modelo de la base de datos y la creación de las tablas haya sido eficiente, es probable que a través del tiempo, se necesiten hacer cambios a los datos almacenados. Esto se debe a que la información cambia debido a múltiples razones, como por ejemplo: errores en la captura de los datos, cambios en las políticas internas de la organización dueña de la base de datos, cambios en particular como la dirección o teléfono de clientes o proveedores, entre otros.

A manera de ejemplo, en la Tabla 6.4 se muestra una tabla con cuatro registros previamente insertados.

Tabla 6.4 - Tabla de Temperatura_Ciudad.

Ciudad	*Temperatura Mínima*	*Temperatura Máxima*
Los Angeles	12°C	28°C
Washington	3°C	15°C
Monterrey	8°C	39°C
Rio de Janeiro	6°C	37°C

Cuando existen datos en una tabla, podemos realizar cambios en ellos utilizando lenguaje SQL. La Tabla 6.5 muestra la misma tabla con algunos cambios o actualizaciones.

Tabla 6.5 - Tabla Temperatura_Ciudad con actualizaciones.

Ciudad	*Temperatura Mínima*	*Temperatura Máxima*
Los Angeles	8°C	28°C
Washington	2°C	15°C
Monterrey	8°C	42°C
Rio de Janeiro	6°C	38°C

Para llevar a cabo las actualizaciones necesarias en los datos de una tabla, los SMBD trabajan con palabras reservadas de SQL, cuya sintaxis se muestra en la Figura 6.6.

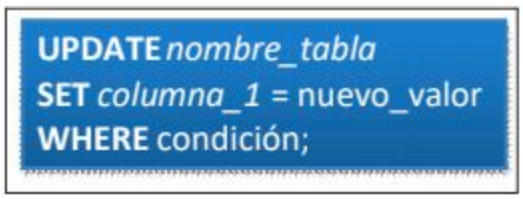

Figura 6.6. Sintaxis de actualización de SQL.

En esta sintaxis encontramos:

1. UPDATE
   Palabra reservada para comenzar la instrucción de actualización de datos.
2. nombre_tabla
   Es el nombre de la tabla en donde se encuentran los datos a los que se les realizarán los cambios.
3. SET

Palabra reservada para definir la columna o conjunto de columnas que se actualizarán.

4. columna_1
   Es el nombre de la columna donde se encuentran los datos que se actualizarán.
5. =
   El signo igual sirve para definir el nuevo valor que tomará la columna que se declaró anteriormente.
6. nuevo_valor
   Es el valor que aparecerá en lugar del anterior.
7. WHERE
   Palabra reservada para indicar bajo que condición se seleccionarán los registros donde se hará la actualización.
8. condicion
   Es una característica o conjunto de características que tiene que cumplir los registros para ser actualizados.
9. ;
   Indica la finalización de la instrucción.

Como ejemplo, se muestra en la Figura 6.7, la sintaxis de la actualización de un campo en un registro de la tabla Temperatura_Ciudad.

```
UPDATE Temperatura_Ciudad
SET Temperatura_Minima = '8ºC'
WHERE Ciudad = 'Los Angeles';
```

Figura 6.7 - Ejemplo de actualización en la tabla Temperatura_Ciudad.

3.4. **Eliminación de registros.**

De la misma manera que se comentó en el punto de Actualización de datos, los datos son sensibles a cambios de diversa naturaleza, uno de estos cambios puede ser también la eliminación de registros. Este proceso consiste en borrar un renglón o renglones específicos de una tabla. Es importante saber que si se requiriera eliminar un campo específico de un renglón, se deberá hacer por el método de Actualización de datos y no por Eliminación de registros.

En la Tabla 6.6 se muestra la tabla completa de ciudades y temperaturas de una base de datos. Por cuestiones de políticas es necesario eliminar la ciudad de Rio de Janeiro. Este proceso se realiza a través de la Eliminación de registros.

Tabla 6.6. Tabla con eliminación del registro para la ciudad Rio de Janeiro.

Ciudad	*Temperatura Mínima*	*Temperatura Máxima*
Los Angeles	8°C	28°C
Washington	2°C	15°C
Monterrey	8°C	42°C

SQL ofrece la capacidad de poder hacer estas consultas con palabras reservadas. A continuación, en la Figura 6.8, se muestra la sintaxis para realizar el proceso de eliminación.

```
DELETE FROM nombre_tabla
WHERE condicion;
```

Figura 6.8 - Sintaxis de eliminación de SQL.

En esta sintaxis podemos encontrar:

1. DELETE FROM
   Palabras reservadas para indicarle al SMBD que se hará la eliminación de un registro completo.
2. nombre_tabla
   Es el nombre de la tabla de donde se eliminará el registro.
3. WHERE
   Palabra reservada para indicar bajo qué condición se seleccionará el registro que se desea eliminar.
4. condicion
   Es una característica o conjunto de características que tiene que cumplir los registros para ser eliminados.
5. ;
   Indica la finalización de la instrucción.

Como ejemplo, en la Figura 6.9, se muestra la sintaxis de la eliminación de un registro de la tabla Temperatura_Ciudad.

```
DELETE FROM Temperatura_ciudad
WHERE Ciudad = 'Rio de Janeiro';
```

Figura 6.9 - Ejemplo de eliminación de registro en la tabla Temperatura_Ciudad.

Profesor
L. en C.C. Erick Orlando Matla Cruz
Ayudantes
L. en C.C. Efraín Hipólito Chamú
L. en C.C. Anahí Quiroz Jiménez
L. en C.C. Karen Monserrat Zavala Correa
Grupo: 9119

4. Ejercicios

(NOTA: Resuelve los siguientes ejercicios en relación al proyecto que realizarás durante el curso, en dado caso que no tengas un proyecto, utiliza la información en el apéndice NFL-ONEFA parte 07 al final de esta práctica para realizarlos)

1. Realiza al menos 15 a 20 **Inserciones para cada tabla de la base de datos**. Guarda y entrega la sintaxis que utilizaste en un archivo .sql.
2. Realiza 2 modificaciones de los datos insertados para cada tabla. Guarda y entrega la sintaxis que utilizaste en un archivo .sql.
3. Realiza 2 consultas por tabla. Guarda y entrega la sintaxis que utilizaste en un archivo .sql.
4. Realiza 2 eliminaciones de cada tabla. Guarda y entrega la sintaxis que utilizaste en un archivo .sql.

Entregables requeridos para prácticas subsecuentes:

- Inserciones para cada tabla de la base de datos

Profesor
L. en C.C. Erick Orlando Matla Cruz
Ayudantes
L. en C.C. Efraín Hipólito Chamú
L. en C.C. Anahí Quiroz Jiménez
L. en C.C. Karen Monserrat Zavala Correa
Grupo: 9119

1. Apéndice Seguros parte 07

Crea la base de datos Seguros (si no la tienes) utilizando el código que se encuentra al término de estas instrucciones. Una vez creadas las tablas realiza los ejercicios de la sección 4 de esta práctica.

```
-- CREATE DATABASE Seguros
-- DROP DATABASE Seguros


CREATE TABLE poliza (
        id_poliza               INTEGER,
        id_paquete              INTEGER,
        id_aseguradora          INTEGER,
        id_persona              INTEGER,
        fecha_inicio            DATE NOT NULL,
        fecha_fin               DATE NOT NULL,
        fecha_emision           DATE NOT NULL,
        status                  VARCHAR (10) NOT NULL,
        PRIMARY KEY (id_poliza),
        FOREIGN KEY (id_paquete) REFERENCES paquete(id_paquete) ON UPDATE CASCADE,
        FOREIGN KEY (id_aseguradora) REFERENCES aseguradora(id_aseguradora) ON UPDATE CASCADE,
        FOREIGN KEY (id_persona) REFERENCES persona(id_persona) ON UPDATE CASCADE,
        CHECK (status in('Vigente','Vencida')),
        CHECK (fecha_emision < fecha_fin ),
        CHECK (fecha_inicio >= fecha_emision AND fecha_inicio < fecha_fin)
);

CREATE TABLE beneficiario (
        id_beneficiario         INTEGER,
        id_persona              INTEGER,
        PRIMARY KEY (id_beneficiario),
        FOREIGN KEY (id_persona) REFERENCES persona(id_persona) ON UPDATE CASCADE
);


CREATE TABLE beneficiario_poliza (
        id_beneficiario         INTEGER,
        id_poliza               INTEGER,

        PRIMARY KEY (id_beneficiario,id_poliza),
        FOREIGN KEY (id_poliza) REFERENCES poliza(id_poliza)
        FOREIGN KEY (id_beneficiario) REFERENCES beneficiario(id_beneficiario)
);


CREATE TABLE reclamacion (
        id_reclamacion          INTEGER,
        id_poliza               INTEGER,
        fecha_reclamacion       DATE NOT NULL,
        fecha_liquidacion       DATE NOT NULL,
        monto                   NUMERIC(15,2) NOT NULL CHECK (monto > 0 ),
        PRIMARY KEY (id_reclamacion),
```

Profesor
L. en C.C. Erick Orlando Matla Cruz
Ayudantes
L. en C.C. Efraín Hipólito Chamú
L. en C.C. Anahí Quiroz Jiménez
L. en C.C. Karen Monserrat Zavala Correa
Grupo: 9119

```
        FOREIGN KEY (id_poliza) REFERENCES poliza(id_poliza) ON UPDATE CASCADE,
        CHECK (fecha_liquidacion >= fecha_reclamacion)                  );
CREATE TABLE monto (
        id_monto            INTEGER,
        id_poliza                   INTEGER,
        id_cmoneda                  INTEGER,
        id_cforma_pago              INTEGER,
        temporalidad                VARCHAR(32) NOT NULL,
        fecha_pago                  DATE NOT NULL,
        PRIMARY KEY (id_monto),
        FOREIGN KEY (id_poliza) REFERENCES poliza(id_poliza) ON UPDATE CASCADE,
        FOREIGN KEY (id_cmoneda) REFERENCES cmoneda(id_cmoneda) ON UPDATE CASCADE,
        FOREIGN KEY (id_cforma_pago) REFERENCES cforma_pago(id_cforma_Pago) ON UPDATE CASCADE,
        CHECK (temporalidad in ('Anual','Semestral','Mensual','Única'))
);


CREATE TABLE cmoneda (
        id_cmoneda                  INTEGER,
        moneda                      VARCHAR(24) NOT NULL,
        PRIMARY KEY (id_cmoneda),
        CHECK (moneda in ('Dólares','Pesos'))
);

CREATE TABLE cforma_pago(
        id_cforma_pago              INTEGER,
        forma_Pago                  VARCHAR(24) NOT NULL,
        PRIMARY KEY (id_cforma_pago),
        CHECK (forma_pago in ('Efectivo','Débito','PayPal','Crédito','Transferencia','Cheque'))
);


CREATE TABLE paquete(
        id_paquete                  INTEGER,
        id_ctipo_seguro             INTEGER,
        nombre_paquete              VARCHAR(64) NOT NULL,
        PRIMARY KEY (id_paquete),
        FOREIGN KEY (id_ctipo_seguro) REFERENCES ctipo_seguro(id_ctipo_seguro) ON UPDATE CASCADE
);


CREATE TABLE cobertura_paquete (
        id_cobertura                INTEGER,
        id_paquete                  INTEGER,
        PRIMARY KEY (id_cobertura,id_paquete),
        FOREIGN KEY (id_paquete) REFERENCES paquete(id_paquete),
        FOREIGN KEY (id_cobertura) REFERENCES cobertura(id_cobertura)
);


CREATE TABLE cobertura (
        id_cobertura                INTEGER,
        nombre_cobertura            VARCHAR(64) NOT NULL,
        PRIMARY KEY (id_cobertura)
);
```

Profesor
L. en C.C. Erick Orlando Matla Cruz
Ayudantes
L. en C.C. Efraín Hipólito Chamú
L. en C.C. Anahí Quiroz Jiménez
L. en C.C. Karen Monserrat Zavala Correa
Grupo: 9119

```
CREATE TABLE ctipo_seguro (
        id_ctipo_seguro         INTEGER  ,
        nombre_seguro           VARCHAR(32) NOT NULL,

        PRIMARY KEY     (id_ctipo_seguro),
        CHECK   (nombre_seguro  in  ('Vida','Gastos  Médicos','Accidentes  Personales','Automóviles  y
        Motocicletas','Vivienda'))
);
CREATE TABLE aseguradora (
        id_aseguradora          INTEGER,
        id_direccion            INTEGER,
        nombre_aseguradora      VARCHAR(140) NOT NULL,
        correo                  VARCHAR(64),
        telefono                NUMERIC(10,0) NOT NULL,

        PRIMARY KEY     (id_aseguradora),
        FOREIGN KEY (id_direccion) REFERENCES direccion(id_direccion) ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE agente (
        id_agente               INTEGER,
        id_persona              INTEGER,
        id_aseguradora          INTEGER,
        fecha_inicio            DATE NOT NULL,
        fecha_fin               DATE,
        PRIMARY KEY (id_agente),
        FOREIGN KEY (id_persona) REFERENCES persona(id_persona) ON UPDATE CASCADE,
        FOREIGN KEY (id_aseguradora) REFERENCES aseguradora(id_aseguradora) ON UPDATE CASCADE,
        CHECK (fecha_inicio < fecha_fin )
);


CREATE TABLE persona (
        id_persona              INTEGER,
        id_cgenero              INTEGER,
        id_direccion            INTEGER,
        nombre                  VARCHAR (64) NOT NULL,
        app                     VARCHAR(32) NOT NULL,
        apm                     VARCHAR(32),
        rfc                     VARCHAR(13) NOT NULL UNIQUE,
        fecha_nac               DATE NOT NULL,
        correo                  VARCHAR(64),
        telefono                NUMERIC(10,0) NOT NULL,
        PRIMARY KEY     (id_persona),
        FOREIGN KEY (id_cgenero) REFERENCES cgenero(id_cgenero) ON UPDATE CASCADE,
        FOREIGN KEY (id_direccion) REFERENCES direccion(id_direccion) ON UPDATE CASCADE,
        CHECK (correo in ('%@%'))
);

CREATE TABLE cgenero (
        id_cgenero              INTEGER,
        genero                  VARCHAR(10) NOT NULL,
        PRIMARY KEY (id_cgenero),
        CHECK (genero in ('Masculino','Femenino'))
```

Profesor
L. en C.C. Erick Orlando Matla Cruz
Ayudantes
L. en C.C. Efraín Hipólito Chamú
L. en C.C. Anahí Quiroz Jiménez
L. en C.C. Karen Monserrat Zavala Correa
Grupo: 9119

```
);

CREATE TABLE direccion (
        id_direccion            INTEGER,
        id_cmunicipio           INTEGER,
        calle                   VARCHAR(75) NOT NULL,
        num_int                 INTEGER CHECK (num_int > 0),
        num_ext                 INTEGER NOT NULL CHECK (num_ext > 0),
        colonia                 VARCHAR(64) NOT NULL,
        cp                      NUMERIC(5,0) NOT NULL,
        PRIMARY KEY     (id_direccion),
        FOREIGN KEY (id_cmunicipio) REFERENCES cmunicipio(id_cmunicipio) ON UPDATE CASCADE      );
CREATE TABLE cmunicipio (
        id_cmunicipio           INTEGER,
        id_cestado              INTEGER,
        municipio               VARCHAR(60) NOT NULL,
        PRIMARY KEY (id_cmunicipio),
        FOREIGN KEY (id_cestado) REFERENCES cestado(id_cestado) ON UPDATE CASCADE
);


CREATE TABLE cestado (
        id_cestado              INTEGER,
        estado                  VARCHAR(24) NOT NULL,
        PRIMARY KEY (id_cestado)
);


CREATE TABLE poliza_vivienda (
        id_poliza_vivienda      INTEGER,
        id_poliza               INTEGER,
        id_direccion            INTEGER,
        id_ctipo_vivienda       INTEGER,
        suma_aseg               NUMERIC(10,2) NOT NULL CHECK (suma_aseg > 0),
        deducible       VARCHAR(3) NOT NULL CHECK (deducible > '0%'),
        no_habitaciones         INTEGER NOT NULL CHECK (no_habitaciones > 0),
        no_ventanas             INTEGER NOT NULL CHECK(no_ventanas > 0),
        no_pisos                INTEGER NOT NULL CHECK (no_pisos > 0),
        PRIMARY KEY (id_poliza_vivienda),
        FOREIGN KEY (id_poliza) REFERENCES poliza(id_poliza) ON UPDATE CASCADE,
        FOREIGN KEY (id_ctipo_vivienda) REFERENCES ctipo_vivienda(id_ctipo_vivienda) ON UPDATE CASCADE,
        FOREIGN KEY (id_direccion) REFERENCES direccion(id_direccion) ON DELETE CASCADE ON UPDATE CASCADE
);


CREATE TABLE ctipo_vivienda (
        id_ctipo_vivienda   INTEGER,
        tipo_vivienda           VARCHAR(32) NOT NULL,
        PRIMARY KEY (id_ctipo_vivienda)
);


CREATE TABLE poliza_vida (
        id_poliza_vida          INTEGER,
        id_poliza               INTEGER,
```

Profesor
L. en C.C. Erick Orlando Matla Cruz
Ayudantes
L. en C.C. Efraín Hipólito Chamú
L. en C.C. Anahí Quiroz Jiménez
L. en C.C. Karen Monserrat Zavala Correa
Grupo: 9119

```
        id_cocupacion           INTEGER,
        suma_aseg               NUMERIC(10,2) NOT NULL CHECK (suma_aseg > 500000 and suma_aseg
        <5000000),
        fumador                 VARCHAR(2) NOT NULL CHECK (fumador in ('Si','No')),
        PRIMARY KEY (id_poliza_vida),
        FOREIGN KEY (id_poliza)  REFERENCES poliza(id_poliza) ON UPDATE CASCADE,
        FOREIGN KEY (id_cocupacion) REFERENCES cocupacion(id_cocupacion) ON UPDATE CASCADE
);


CREATE TABLE cocupacion (
        id_cocupacion           INTEGER,
        ocupacion               VARCHAR(140) NOT NULL,
        PRIMARY KEY (id_cocupacion)
);

CREATE TABLE poliza_ap_gm(
        id_poliza_ap_gm         INTEGER,
        id_poliza               INTEGER,
        id_ctipo_seguro_ap_gm   INTEGER,
        id_cocupacion           INTEGER,
        deducible       VARCHAR(3)  NOT NULL CHECK (deducible > '0%' )
        suma_aseg               INTEGER NOT NULL CHECK (suma_aseg > 100000 AND suma_aseg <= 5000000)
        PRIMARY KEY (id_poliza_ap_gm),
        FOREIGN KEY (id_cocupacion) REFERENCES cocupacion(id_cocupacion) ON UPDATE CASCADE,
        FOREIGN KEY (id_poliza) REFERENCES poliza(id_poliza) ON UPDATE CASCADE,
        FOREIGN KEY (id_ctipo_seguro_ap_gm) REFERENCES ctipo_seguro_ap_gm(id_ctipo_seguro_ap_gm) ON
UPDATE CASCADE
);

CREATE TABLE chospital (
        id_chospital            INTEGER,
        nombre_hospital         VARCHAR(64) NOT NULL,
        PRIMARY KEY (id_chospital)
);

CREATE TABLE poliza_ap_gm_chospital (
        id_poliza_ap_gm         INTEGER,
        id_chospital            INTEGER,
        PRIMARY KEY (id_poliza_ap_gm, id_chospital),
        FOREIGN KEY (id_poliza_ap_gm) REFERENCES poliza_ap_gm(id_poliza_ap_gm),
);


CREATE TABLE ctipo_seguro_ap_gm (
        id_ctipo_seguro_ap_gm   INTEGER,
        nombre_seguro           VARCHAR(32) NOT NULL,
        PRIMARY KEY (id_ctipo_seguro_ap_gm),
        FOREIGN KEY     (id_chospital) REFERENCES chospital(id_chospital)
);


CREATE TABLE poliza_vehiculo (
        id_poliza_vehiculo      INTEGER,
        id_poliza               INTEGER,
        id_cmodelo              INTEGER,
```

Profesor
L. en C.C. Erick Orlando Matla Cruz
Ayudantes
L. en C.C. Efraín Hipólito Chamú
L. en C.C. Anahí Quiroz Jiménez
L. en C.C. Karen Monserrat Zavala Correa
Grupo: 9119

```
        id_ccolor               INTEGER,
        id_ctipo_vehiculo       INTEGER,
        no_serie                VARCHAR(16) NOT NULL CHECK (no_serie in('[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-
        9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]')),
        placa                   VARCHAR(7) NOT NULL CHECK (placa in('[0-9][0-9][0-9]-[A-Z][A-Z][A-Z]')),
        deducible       VARCHAR(3) NOT NULL CHECK (deducible > '0%'),
        PRIMARY KEY (id_poliza_vehiculo),
        FOREIGN KEY (id_poliza) REFERENCES poliza(id_poliza) ON UPDATE CASCADE,
        FOREIGN KEY (id_cmodelo) REFERENCES cmodelo(id_cmodelo) ON UPDATE CASCADE,
        FOREIGN KEY (id_ccolor) REFERENCES ccolor(id_ccolor) ON UPDATE CASCADE,
        FOREIGN KEY (id_ctipo_vehiculo) REFERENCES ctipo_vehiculo(id_ctipo_vehiculo) ON UPDATE CASCADE
);


CREATE TABLE ctipo_vehiculo (
        id_ctipo_vehiculo       INTEGER,
        tipo_vehiculo           VARCHAR(24) NOT NULL,
        PRIMARY KEY (id_ctipo_vehiculo)
);


CREATE TABLE cmodelo (
        id_cmodelo              INTEGER,
        modelo                  VARCHAR(72) NOT NULL,
        PRIMARY KEY (id_cmodelo)
    );


CREATE TABLE ccolor (
        id_ccolor               INTEGER,
        color                   VARCHAR(16) NOT NULL,
        PRIMARY KEY     (id_ccolor)
);
```