# Actividad de la unidad 2 – SQL Server

Cristian Enrique Paez Valencia

Ingeniería de Software, Universidad de Cartagena

Bases de datos I

Jhon Carlos Arrieta Arrieta

Cartagena, Colombia

9 de noviembre del 2024

# Tabla de contenido

1. **Introducción** ............................................................................. 3
2. **Objetivos**………………………………………………………4
3. **Justificación** ............................................................................... 5
4. **Desarrollo** ...................................................................................6
5. **Bibliografía** .................................................................................63

# Introducción

En este trabajo vamos a poner en práctica los contenidos vistos en la unidad 2, asi mismo se hace énfasis en a lo que se refiere a las diferentes sentencias SQL Sever en ejercicios propuestos que a continuación se presentan, estaremos usando una base de datos ya anteriormente creada pero ahora vamos a usar la GUI para ejecutar los distintos comandos, que nos van a permitir insertar, actualizar, eliminar y asi mismo consultar.

# Objetivos

* Practicar y aplicar los conceptos de SQL y DML para insertar, modificar, eliminar y consultar información en sistemas para bases de datos relaciones.
* Interpretar una solicitud de información y transformarla en una operación de SQL DML según sea su naturaleza y objetivo.
* Realizar backup y restauración de la base de datos con sus estructura y datos intactos.
* Manipular la información almacenada en un sistema para bases de datos de acuerdo con las necesidades de información que requiera el cliente o una aplicación de software que deba interactuar con dicha base de datos.

# Justificación

Este trabajo tiene como finalidad, poner en práctica los conceptos vistos en la unidad dos, ya que es muy importante no solamente tener el concepto de cómo hacerlo, sino que tambien es de suma importancia ponerlo en ejecución, por tal motivo se hace este trabajo.

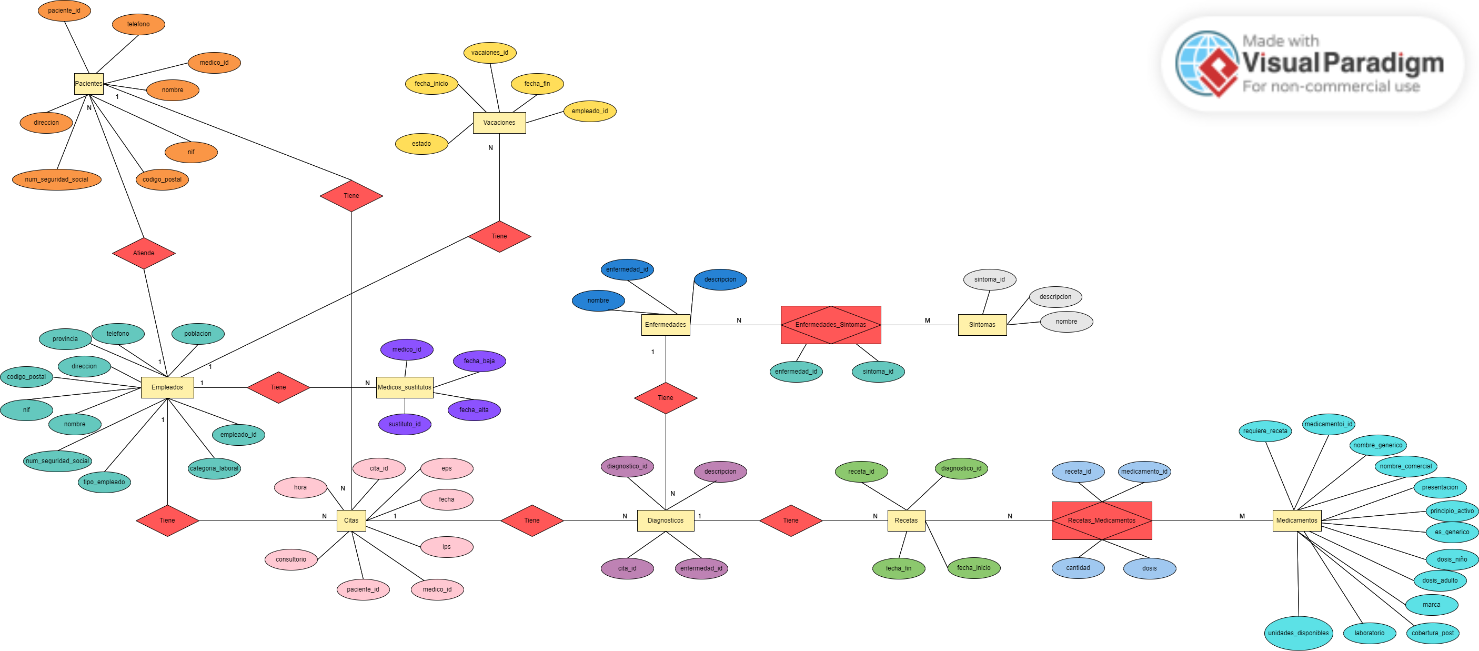
# Desarrollo

1. **Texto del enunciado del ejercicio que le fue asignado.**

[Ejercicio 24- Hospital](https://drive.google.com/file/d/1DicRAeFWLo6gmHpw8KOG4Chpo43i-HZm/view?usp=sharing)

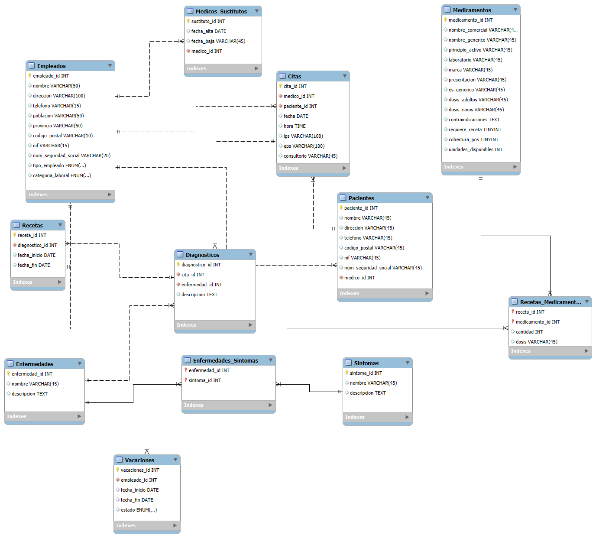
1. **Modelo Diagrama entidad relación**

[Diagrama entidad-relacion](https://drive.google.com/file/d/13UfZu5zTzqm61Wo1CnY4uzYhBDX94yBo/view?usp=sharing)



1. **Modelo relacional**

[Modelo relacional](https://drive.google.com/file/d/1jJuJcs2LWM2HTonlGQDjc4s3d5swKX7U/view?usp=sharing)



1. **Enlace para descargar el contenido de la actividad 1**
2. **Enlace para descargar el contenido de la actividad 2**
3. **Dearrollo de los puntos que se deben resolver descritos en el contenido de la actividad.**

**PRIMERA PARTE: 30%**

1. Por cada uno de los 3 motores servidores de bases de datos relaciones utilizados para desarrollar esta actividad, usted debe descargar, instalar y utilizar la aplicación Cliente SQL con GUI oficiales y recomendada por el respectivo fabricante o proveedor del Servidor Motor de BD relacional.

DESCRIPCIÓN DETALLADA DE LOS PASOS PARA REALIZAR LA CONEXIÓN (Utilice toda la didáctica e ilustraciones necesarias para que el proceso quede bien documentado y explicado)

Entramos a sql server management y ya estando dentro nos pide la conexión a donde queremos conectarnos, hacemos dicha conexión, se nos abre una carpeta en la parte izquierda que contiene las bases de datos le damos en click derecho para crear la base de datos colocamos el nombre en la ventana que se nos abrió y le damos en crear y listo ya tendríamos la conexión a la base de datos.

1. Utilizar el archivo **backup\_bd\_ejercicio\_xyz.sql** para realizar el proceso de restauración de la bd

DESCRIPCIÓN DETALLADA DE LOS PASOS PARA RESOLVER ESTE PUNTO

Estando dentro del SQL server le damos click derecho y seleccionamos restaurar base de datos le damos en la opción de seleccionar un file y se nos abre una ventana y seleccionamos la ruta en donde se encuentra nuestra backup y listo.

1. Verificar que se han creado todas las tablas de la **bd\_ejercicio\_xyz**

DESCRIPCIÓN DETALLADA DE LOS PASOS PARA RESOLVER ESTE PUNTO

Hacemos una consulta con una query a la base de datos para verificar que se hizo la conexión.

1. Vuelva a general el Modelo Relacional usando la funcionalidad que le ofrece el Cliente SMBDR propio de cada uno de los 3 motores de BD con los que ha decidido desarrollar esta actividad. El modelo relacional generado debe ser igual en cuanto a estructura (Tablas y relaciones) en los tres motores de bases de datos, igual al modelo relacional utilizado inicialmente construir las bases de datos en la actividad #1

DESCRIPCIÓN DETALLADA DE LOS PASOS PARA RESOLVER ESTE PUNTO

Puedes ejecutar el script SQL directamente si lo tienes para crear las tablas y relaciones.

1. Volver a crear el usuario para **usuario\_xyz** con las mismas credenciales utilizadas en el punto la 2da Actividad.

DESCRIPCIÓN DETALLADA DE LOS PASOS PARA RESOLVER ESTE PUNTO

Ingresamos a SQL Server y creamos un usuario llamado cristian\_xyz con la misma contraseña.

1. Desconectarse el Cliente SQL del respectivo servidor de BD relacional y volver a conectarse utilizando las credenciales del usuario **usuario\_xyz.**

DESCRIPCIÓN DETALLADA DE LOS PASOS PARA RESOLVER ESTE PUNTO

Nos salimos de la conexión de la base de datos e ingresamos con las credenciales del nuevo usuario.

1. Mostrar las bases de datos del usuario **usuario\_xyz**

DESCRIPCIÓN DETALLADA DE LOS PASOS PARA RESOLVER ESTE PUNTO

Ejecutamoes este comando:

SELECT name

FROM sys.databases

WHERE SUSER\_SNAME(owner\_sid) = 'cristian\_xyz';

1. Descargar e instalar cualquier IDE (Entorno Integrado de desarrollo) de su preferencia p.e: Visual Studio Code, Atom, Visual Studio .Net, Netbeans, Eclipse, IntelliJ IDEA, Aptana Studio, DreamWaver, etc. (escoger al menos 3) y hacer que estos se conecten a los 3 motores de bases de datos utilizados para desarrollar esta actividad.

DESCRIPCIÓN DETALLADA DE LOS PASOS PARA RESOLVER ESTE PUNTO

Visual Studio Code:

Lo primero que hacemos es descargar una extensión llamada SQLTools y cada extensión correspondiente para cada base de datos si es necesario, instalamos plugins adicionales para cada base de datos: SQLTools Oracle, SQLTools PostgreSQL, SQLTools MySQL/MariaDB, SQLTools MSSQL, seguido de eso presionamos las teclas Ctrl + Shift + P y buscamos SQLTools: Add new connection) y agregamos la conexión a la base de datos, configuramos host, port, user, password, y database. Con las credenciales que le hayamos pasado.

NetBeans:

**Abrir la Ventana de Servicios**:

* Ve a Window > Services para abrir el panel de Databases.

**Agregar Controladores**:

* Si no tienes los controladores de base de datos, debes descargarlos:
  + **Oracle**
  + **PostgreSQL**
  + **MySQL**
  + **SQL Server**

**Configurar Conexiones**:

* En Services > Databases, haz clic derecho y selecciona New Connection....
* Selecciona el controlador correspondiente (agrega el archivo .jar si es necesario).
* Completa los detalles de conexión (host, puerto, usuario, contraseña, nombre de la base de datos).

**Probar Conexión**:

* Una vez configurada la conexión, haz clic en Test Connection para verificar. Si la prueba es exitosa, guarda la conexión y úsala para realizar consultas.

IntelliJ:

**Abrir el Panel de Base de Datos**:

* Ve a View > Tool Windows > Database.

**Agregar Controladores**:

* Si no tienes los controladores JDBC instalados, IntelliJ los descargará automáticamente cuando selecciones un tipo de base de datos. También puedes descargarlos manualmente como en NetBeans:
  + **Oracle**
  + **PostgreSQL**
  + **MySQL**
  + **SQL Server**

**Configurar Conexiones**:

* En el panel de Database, haz clic en + y selecciona Data Source.
* Escoge el motor de base de datos (Oracle, PostgreSQL, MySQL, SQL Server) y rellena los detalles:
  + Host, Port: Dirección y puerto del servidor.
  + User y Password: Credenciales de usuario.
  + Database: Nombre de la base de datos (ejemplo: ORCL para Oracle).

**Probar Conexión**:

* Haz clic en Test Connection para verificar que los datos son correctos.

**SEGUNDA PARTE 70%**

Desde este punto en adelante, usted debe responder cada punto escribiendo el texto de la instrucción SQL correcta, explicación de su funcionamiento e imagen (recorte de la zona de pantalla) del resultado de generado por la instrucción SQL ingresada desde el editor SQL del Cliente para BD que esté utilizando:

1. **Escribir el texto SQL (copiar y pegar desde el editor SQL):**
2. **Explicación de la instrucción**
3. **Recorte de pantalla de la instrucción ingresada** en el editor del Cliente SQL y su respectivo resultado.

Leer detalladamente cada uno de los siguientes ítems y resolverlo siguiendo al pie de la letra las instrucciones que se le solicitan.

Tome el modelo relacional de su ejercicio **bd\_ejercicio\_xyz** y escoja al menos dos tablas que estén relacionadas entre sí mediante llave foránea.

Para poder explicar de una manera más simple el desarrollo del ejercicio práctico, a manera de referencia el profesor ha llamado TablaX y TablaZ a las dos tablas que usted ha seleccionado del modelo relacional de su ejercicio individual de BD (actividad #1), cuya esquema se puede expresar así como [**TablaX**](id, columna1, columna2, columna3, columnaN)

[**TablaZ**](id, columna1, columna2, columna3, tablax\_id)

y su relación se puede representar así:

**[TablaX]**--|------------<-**[TablaZ]**

Donde claramente cada registro de **TablaZ** depende o está relacionado con un registro de **TablaX.**

1. Insertar en TablaZ un al menos 1 registro usando la forma de inserción completa de SQL.

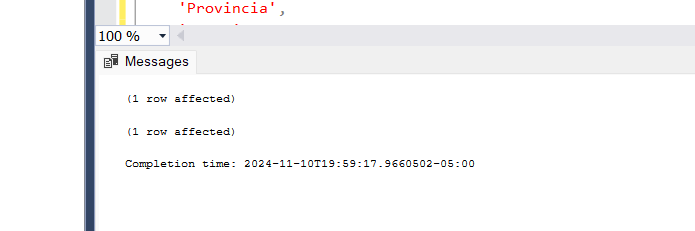
* INSTRUCCIÓN SQL:

|  |
| --- |
| * -- Inserción en la tabla Empleados * INSERT INTO Empleados ( * nombre, * direccion, * telefono, * poblacion, * provincia, * codigo\_postal, * nif, * num\_seguridad\_social, * tipo\_empleado, * categoria\_laboral * ) VALUES ( * 'Juan Perez', * '123 Calle Principal', * '123456789', * 'Ciudad', * 'Provincia', * '12345', * '12345678A', * 'SS12345678', * 'medico', * 'titular' * ); * -- Inserción en la tabla Medicos\_Sustitutos usando el ID del médico * INSERT INTO Medicos\_Sustitutos ( * medico\_id, * fecha\_alta, * fecha\_baja * ) VALUES ( * (SELECT TOP 1 empleado\_id FROM Empleados WHERE nombre = 'Juan Perez'), * '2024-01-01', * '2024-12-31' * ); |

* Explicación:

Ya con esto primero creamos un empleado para que no nos genere error y despue si creamos el medico sustituto.

* Recorte de pantalla del resultado:



1. Mostrar todos los registros Insertados en la TablaZ

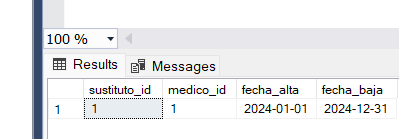
* INSTRUCCIÓN SQL:

|  |
| --- |
| * select \* from medicos\_sustitutos; |

* Explicación:

Con esto consultamos la tabla de medicos

* Recorte de pantalla del resultado:



1. Insertar en TablaX al menos 5 registros usando el formato de **Inserción completo** de SQL.

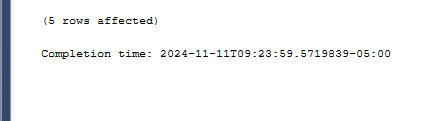
* INSTRUCCIÓN SQL:

|  |
| --- |
| INSERT INTO Empleados (  nombre,  direccion,  telefono,  poblacion,  provincia,  codigo\_postal,  nif,  num\_seguridad\_social,  tipo\_empleado,  categoria\_laboral  ) VALUES  ('Ana Gomez', '456 Calle Secundaria', '987654321', 'Ciudad1', 'Provincia1', '54321', '87654321B', 'SS87654321', 'medico', 'interino'),  ('Carlos Ruiz', '789 Calle Tercera', '567890123', 'Ciudad2', 'Provincia2', '67890', '87654321C', 'SS98765432', 'ats', 'titular'),  ('Lucia Torres', '321 Calle Cuarta', '345678901', 'Ciudad3', 'Provincia3', '98765', '87654321D', 'SS12345678', 'auxiliar', 'sustituto'),  ('Mario Sanchez', '654 Calle Quinta', '234567890', 'Ciudad4', 'Provincia4', '87654', '87654321E', 'SS87654322', 'celador', 'interino'),  ('Patricia Molina', '987 Calle Sexta', '123098765', 'Ciudad5', 'Provincia5', '76543', '87654321F', 'SS23456789', 'admin', 'titular'); |

* Explicación:

Con esto insertamos a los 5 empleados

* Recorte de pantalla del resultado:

****

1. Mostrar todos los registros Insertados en la TablaX

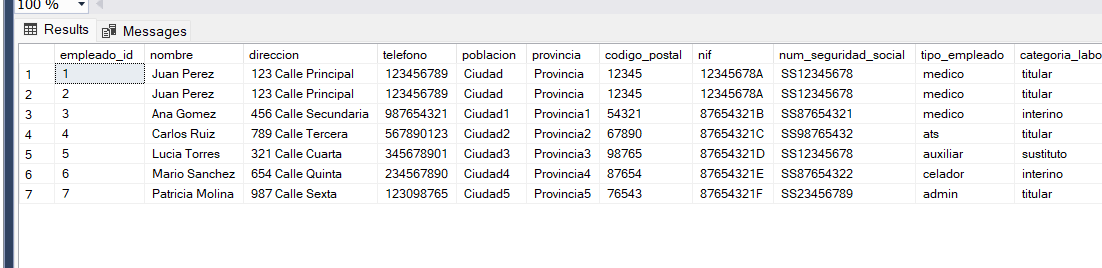
* INSTRUCCIÓN SQL:

|  |
| --- |
| * select \* from Empleados; |

* Explicación:

Consultamos a los empleados

* Recorte de pantalla del resultado:



1. Insertar en TablaZ al menos 5 registros usando el formato de **Inserción parcial** de SQL, pero que estén relacionados con los registros insertados previamente en TablaX

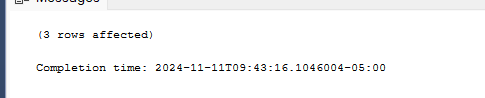
* INSTRUCCIÓN SQL:

|  |
| --- |
| -- Para SQL Server  INSERT INTO Medicos\_Sustitutos (medico\_id, fecha\_alta, fecha\_baja)  SELECT TOP 5 empleado\_id, '2024-01-01', '2024-12-31'  FROM Empleados  WHERE tipo\_empleado = 'medico'; |

* Explicación:

Insertamos los registros parcialmente

* Recorte de pantalla del resultado:

****

1. Mostrar todos los registros Insertados en la TablaX y los insertados en TablaZ, debe utilizar una instrucciones SQL por separado para mostrar los resultados de cada consulta.

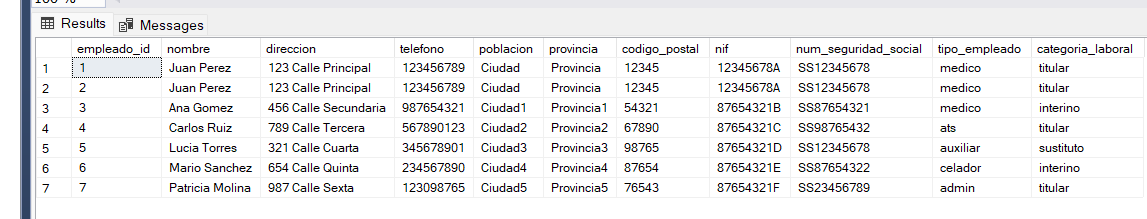
* INSTRUCCIÓN SQL:

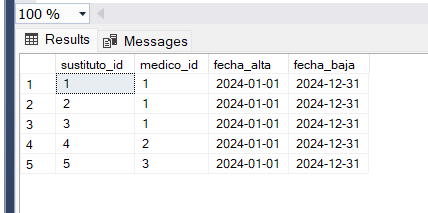
|  |
| --- |
| * select \* from medicos\_sustitutos; * select \* from Empleados; |

* Explicación:

Consultamos las dos tablas

* Recorte de pantalla del resultado:





1. Insertar en TablaX al menos 15 registros usando el formato de **Inserción múltiple** de SQL.

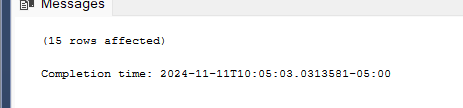
* INSTRUCCIÓN SQL:

|  |
| --- |
| -- Para SQL Server  INSERT INTO Empleados (nombre, direccion, telefono, poblacion, provincia, codigo\_postal, nif, num\_seguridad\_social, tipo\_empleado, categoria\_laboral)  VALUES  ('Juan Pérez', 'Calle Falsa 123', '123456789', 'Madrid', 'Madrid', '28001', '12345678A', '12345678901', 'medico', 'titular'),  ('Ana Gómez', 'Avenida de la Paz 456', '987654321', 'Barcelona', 'Cataluña', '08002', '87654321B', '98765432101', 'ats', 'interino'),  ('Luis Martínez', 'Calle Sol 789', '543216789', 'Valencia', 'Comunidad Valenciana', '46003', '23456789C', '54321678902', 'medico', 'sustituto'),  ('Laura Sánchez', 'Calle Luna 123', '111223344', 'Sevilla', 'Andalucía', '41004', '34567890D', '11122334403', 'auxiliar', 'titular'),  ('Pedro López', 'Calle Estrella 456', '222334455', 'Granada', 'Andalucía', '18005', '45678901E', '22233445504', 'celador', 'interino'),  ('María Fernández', 'Calle Mar 789', '333445566', 'Zaragoza', 'Aragón', '50006', '56789012F', '33344556605', 'admin', 'sustituto'),  ('Carlos Ruiz', 'Calle Rosa 321', '444556677', 'Murcia', 'Murcia', '30007', '67890123G', '44455667706', 'medico', 'titular'),  ('Sofía Álvarez', 'Calle Árbol 654', '555667788', 'Alicante', 'Comunidad Valenciana', '03008', '78901234H', '55566778807', 'ats', 'interino'),  ('David Pérez', 'Calle Río 987', '666778899', 'Córdoba', 'Andalucía', '14009', '89012345I', '66677889908', 'medico', 'sustituto'),  ('Elena Castro', 'Calle Puente 135', '777889900', 'Madrid', 'Madrid', '28010', '90123456J', '77788990009', 'auxiliar', 'titular'),  ('José Díaz', 'Calle Valle 246', '888990011', 'Bilbao', 'País Vasco', '48011', '01234567K', '88899001110', 'celador', 'interino'),  ('Irene Martínez', 'Calle Sol 357', '999100122', 'Valencia', 'Comunidad Valenciana', '46012', '12345678L', '99910012211', 'admin', 'sustituto'),  ('Raúl Sánchez', 'Calle Luna 468', '111223344', 'Sevilla', 'Andalucía', '41013', '23456789M', '11122334412', 'medico', 'titular'),  ('Carmen Rodríguez', 'Calle Ria 579', '222334455', 'Granada', 'Andalucía', '18014', '34567890N', '22233445513', 'ats', 'interino'),  ('Javier Pérez', 'Calle Estrella 680', '333445566', 'Zaragoza', 'Aragón', '50015', '45678901O', '33344556614', 'medico', 'sustituto'); |

* Explicación:

Hacemos la inserción múltiple

* Recorte de pantalla del resultado:



1. Mostrar el contenido completo de TablaX

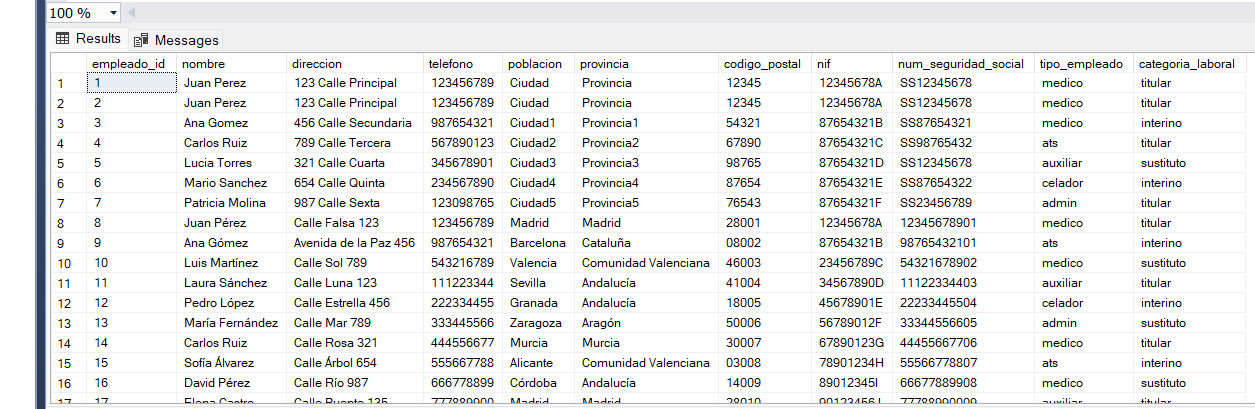
* INSTRUCCIÓN SQL:

|  |
| --- |
| * select \* from Empleados; |

* Explicación:

Consultamos a los empleados

* Recorte de pantalla del resultado:



1. Inserte al menos un registro en TablaX usando la forma completa de inserción de SQL, pero en la lista de valores omita el valor de al menos una columna.

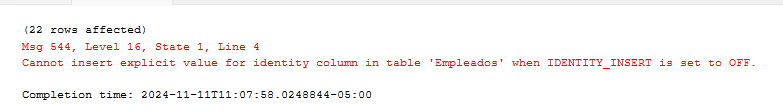
* INSTRUCCIÓN SQL:

|  |
| --- |
| -- Insertar en la tabla Empleados omitiendo la columna 'categoria\_laboral'  INSERT INTO Empleados (empleado\_id, nombre, direccion, telefono, poblacion, provincia, codigo\_postal, nif, num\_seguridad\_social, tipo\_empleado)  VALUES (7, 'Juan Pérez', 'Calle Falsa 123', '123456789', 'Madrid', 'Madrid', '28001', '12345678A', '12345678901', 'medico'); |

* Explicación:

Insertamos, pero nos da un error ya que no permite null

* Recorte de pantalla del resultado:



1. Mostrar el contenido completo de TablaX

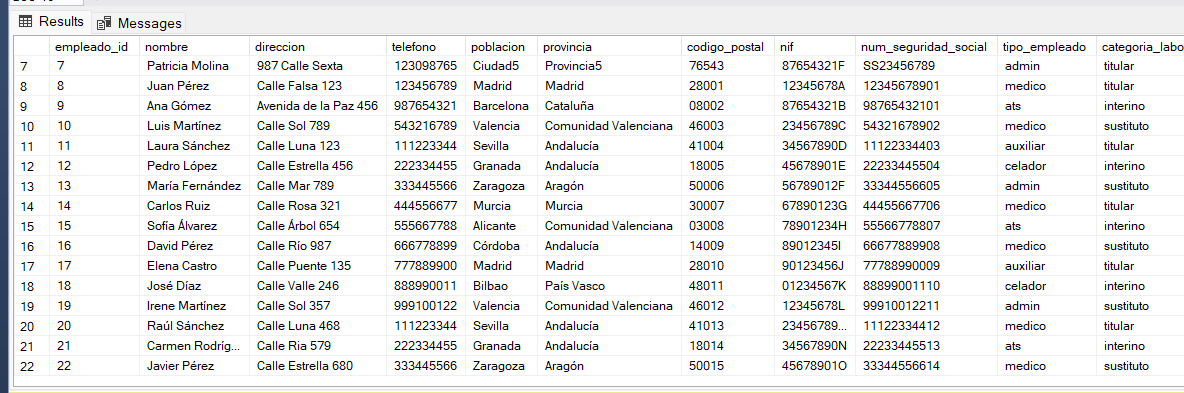
* INSTRUCCIÓN SQL:

|  |
| --- |
| * select \* from Empleados; |

* Explicación:

Consultamos los datos en la tabla de empleados

* Recorte de pantalla del resultado:



1. Inserte al menos un registro en TablaX usando la forma de inserción parcial de SQL, pero no incluya la lista de columnas al menos una columna cuyo valor no acepta NULL.

* INSTRUCCIÓN SQL:

|  |
| --- |
| INSERT INTO Empleados (nombre, direccion, telefono, poblacion, provincia, codigo\_postal, nif, num\_seguridad\_social, tipo\_empleado, categoria\_laboral)  VALUES ('Juan Pérez', 'Calle Falsa 123', '123456789', 'Madrid', 'Madrid', '28001', '12345678A', '12345678901', 'medico', 'titular'); |

* Explicación:

Me deja insertar y no da error

* Recorte de pantalla del resultado:



1. Mostrar el contenido completo de TablaX

* INSTRUCCIÓN SQL:

|  |
| --- |
| * select \* from Empleados; |

* Explicación:

Lo que hacemos es consultar la tabla

* Recorte de pantalla del resultado:



1. Inserte al menos un registro en TablaX usando cualquiera de las formas de inserción SQL (completa, parcial o múltiple), cuyo nuevo registro en la columna de llave primaria tenga un valor que ya esté previamente insertado en esa misma columna Primary Key de esa misma tabla.

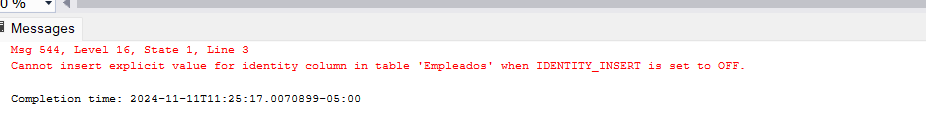
* INSTRUCCIÓN SQL:

|  |
| --- |
| -- Inserción con valor duplicado en la columna de llave primaria 'empleado\_id'  -- Esto generará un error de duplicidad debido a la restricción de la clave primaria  INSERT INTO Empleados (empleado\_id, nombre, direccion, telefono, poblacion, provincia, codigo\_postal, nif, num\_seguridad\_social, tipo\_empleado, categoria\_laboral)  VALUES (1, 'Juan Pérez', 'Calle Falsa 123', '123456789', 'Madrid', 'Madrid', '28001', '12345678A', '12345678901', 'medico', 'titular'); |

* Explicación:

Tratamos de insertar un dato duplicado, pero nos da error

* Recorte de pantalla del resultado:



1. Mostrar el contenido completo de TablaX

* INSTRUCCIÓN SQL:

|  |
| --- |
| * select \* from Empleados; |

* Explicación:

Mostramos la tabla empleados

Recorte de pantalla del resultado:



1. Tomar al menos un registro de TablaX cuya llave primaria está relacionada con al menos un registro de TablaZ, luego proceda a cambiar, modificar o actualizar el valor actual de la llave primaria de ese registro en la TablaX, por un otro nuevo valor que no haya sido ingresado en esa misma primary key.

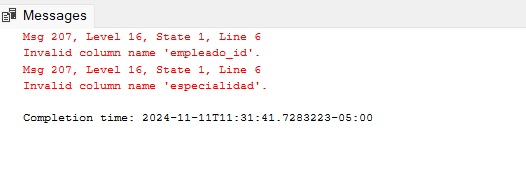
* INSTRUCCIÓN SQL:

|  |
| --- |
| -- Inserción en medicos\_sustitutos  INSERT INTO medicos\_sustitutos (empleado\_id, especialidad)  VALUES (1, 'Medicina General');  -- Actualización de la llave primaria en Empleados  UPDATE Empleados  SET empleado\_id = 2 -- Nuevo valor único  WHERE empleado\_id = 1; |

* Explicación:

Tratamos de cambiar los datos, pero no se puede y nos da error

* Recorte de pantalla del resultado:



1. Mostrar el contenido completo de TablaX

* INSTRUCCIÓN SQL:

|  |
| --- |
| * select \* from Empleados; |

* Explicación:

Consultamos los datos de la tabla

* Recorte de pantalla del resultado:



1. Realice el proceso inverso, es decir, seleccione un registro de la TablaZ y modifique el valor de la llave foránea que referencia a otro registro de TablaX, y cambielo por un nuevo valor que no exista en la llave primaria de TablaX

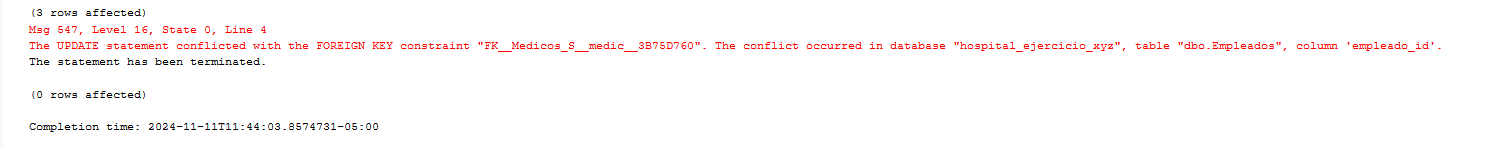
* INSTRUCCIÓN SQL:

|  |
| --- |
| SELECT \* FROM Medicos\_Sustitutos WHERE medico\_id = 1; -- Ahora, intentamos actualizar el valor de medico\_id a un nuevo valor que no exista en Empleados UPDATE Medicos\_Sustitutos SET medico\_id = 999 -- Asumimos que el valor 999 no existe en Empleados WHERE medico\_id = 1; -- Verificamos la actualización SELECT \* FROM Medicos\_Sustitutos WHERE medico\_id = 999; |

* Explicación:

Hace la modificación pero nos da error

* Recorte de pantalla del resultado:



1. Mostrar el contenido completo de TablaZ.

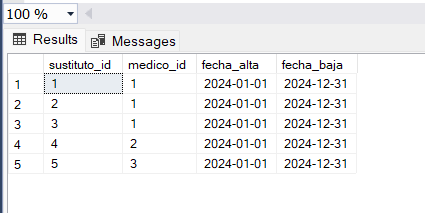
* INSTRUCCIÓN SQL:

|  |
| --- |
| * select \* from medicos\_sustitutos; |

* Explicación:

Mostramos el contenido de la tabla

* Recorte de pantalla del resultado:



1. Mostrar el contenido completo de TablaX

* INSTRUCCIÓN SQL:

|  |
| --- |
| -- Supongamos que tenemos un registro en Medicos\_Sustitutos con medico\_id = 1  -- Primero, seleccionamos el registro de Medicos\_Sustitutos  SELECT \* FROM Medicos\_Sustitutos WHERE medico\_id = 1;  -- Ahora, intentamos actualizar el valor de medico\_id a un nuevo valor que no exista en Empleados  UPDATE Medicos\_Sustitutos  SET medico\_id = 999 -- Asumimos que el valor 999 no existe en Empleados  WHERE medico\_id = 1;  -- Verificamos la actualización  SELECT \* FROM Medicos\_Sustitutos WHERE medico\_id = 999; |

* Explicación:

Mostramos el contenido de la tabla empleados

* Recorte de pantalla del resultado:



1. Seleccionar una columna de TablaX que no acepte valores NULL, tomar un registro en dicha tabla y modificar el valor a esa columna ingresado un valor NULL.

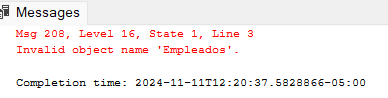
* INSTRUCCIÓN SQL:

|  |
| --- |
| UPDATE Empleados  SET empleado\_id = NULL  WHERE empleado\_id = 1; |

* EXPLICACIÓN:

Noa da error al tratar de hacer la modificación

* Recorte de pantalla del resultado:



1. Mostrar el contenido completo de TablaX

* INSTRUCCIÓN SQL:

|  |
| --- |
| * select \* from Empleados; |

* Explicación:

Consultamos la tabla empleados

* Recorte de pantalla del resultado:



1. Seleccionar una columna de TablaX que tenga tipo de dato numérico (int, float, double, decimal o numeric) según el tipo de dato del motor de BD que sobre el cual esté desarrollando la actividad. Seleccione un registro en esa tabla y modifique el valor de esa columna ingresando un valor textual (no numérico)

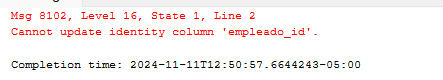
* INSTRUCCIÓN SQL:

|  |
| --- |
| -- Intentamos actualizar el campo numérico 'empleado\_id' con un valor no numérico  UPDATE Empleados  SET empleado\_id = 'TextoInvalido'  WHERE empleado\_id = 1;  -- SQL Server generará un error de conversión de tipo de datos. |

* EXPLICACIÓN:

Intentaremos ingresar un dato en el cual no se puede ingresar datos de ese tipo y nos generara error

* Recorte de pantalla del resultado:



1. Mostrar el contenido completo de TablaX

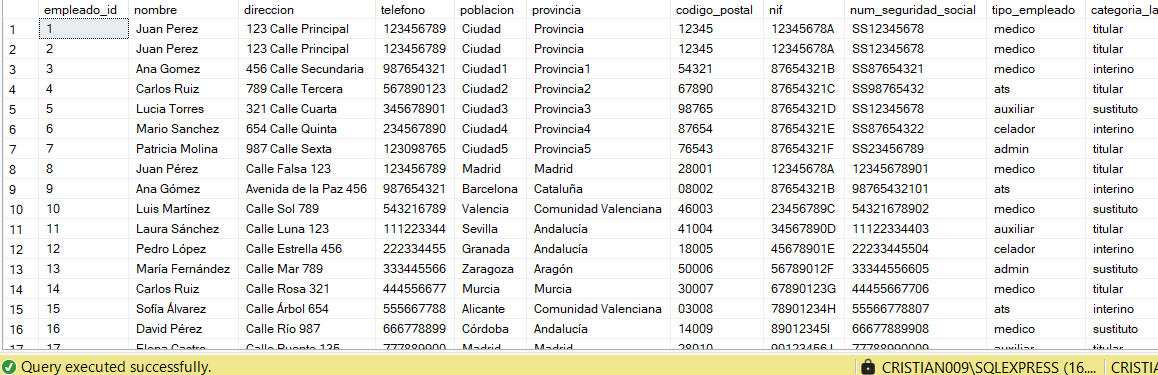
* INSTRUCCIÓN SQL:

|  |
| --- |
| * select \* from Empleados; |

* Explicación:

Consultamos la tabla empleados

* Recorte de pantalla del resultado:



1. Seleccionar una columna de TablaX que tenga tipo de dato numérico (int, float, double, decimal o numeric) según el tipo de dato del motor de BD que sobre el cual esté desarrollando la actividad. Seleccione un registro en esa tabla y modifique el valor de esa columna ingresando un valor numérico en formato texto, es decir, encerrado entre comillas simples.

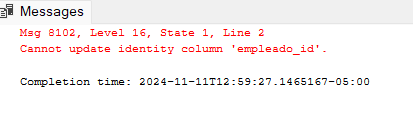
* INSTRUCCIÓN SQL:

|  |
| --- |
| UPDATE Empleados  SET empleado\_id = '100'  WHERE empleado\_id = 1; |

* EXPLICACIÓN:

Nos genera error ya que estamos intentado ingresar un valor que no coresponde al que tiene designado

* Recorte de pantalla del resultado:



1. Mostrar el contenido completo de TablaX

* INSTRUCCIÓN SQL:

|  |
| --- |
| * select \* from Empleados; |

* Explicación:

Consultamos a la tabla empleados

* Recorte de pantalla del resultado:

\_

1. Seleccionar una columna de TablaX que tenga tipo de dato texto (char, chr, nchar, ntext, nvarchar, character, charvaring, varchar, varchar2, etc) según el tipo de dato del motor de BD que sobre el cual esté desarrollando la actividad. Seleccione un registro en esa tabla y modifique el valor de esa columna ingresando un valor numérico, es decir, un dato en formato número, sin comillas simples

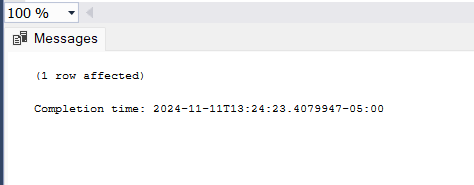
* INSTRUCCIÓN SQL:

|  |
| --- |
| UPDATE Empleados SET nombre = 12345 WHERE empleado\_id = 1; |

* EXPLICACIÓN:

Lo que hacemos es modificar la columna de ese empleados

* Recorte de pantalla del resultado:



1. Mostrar el contenido completo de TablaX

* INSTRUCCIÓN SQL:

|  |
| --- |
| * select \* from Empleados; |

* Explicación:

Lo que hacemos es consultar la tabla empleados

* Recorte de pantalla del resultado:



1. Seleccionar en TablaX una columna de tipo Fecha (Date, Time, DateTime, SmallDateTime, DateTime2, TimeStamp, Year, Mont, Day, etc) según el motor de bases de datos que esté usando para desarrollar esta actividad. Luego seleccione al menos un registro de TablaX e inserte un registro usando la forma completa de inserción de SQL, de tal menara que el valor de la columna de tipo fecha tenga el siguiente dato: **29-02-2023**

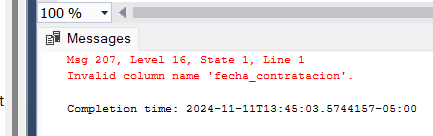
* INSTRUCCIÓN SQL:

|  |
| --- |
| INSERT INTO Empleados (empleado\_id, nombre, direccion, telefono, poblacion, provincia, codigo\_postal, nif, num\_seguridad\_social, tipo\_empleado, categoria\_laboral, fecha\_contratacion) VALUES (100, 'Carlos Test', 'Calle Prueba 123', '123456789', 'Ciudad Test', 'Provincia Test', '12345', '12345678X', '12345678910', 'medico', 'titular', '2023-02-29'); |

* Explicación:

Tratamos de cosultar pero nos da error

* Recorte de pantalla del resultado:



1. Mostrar el contenido completo de TablaX

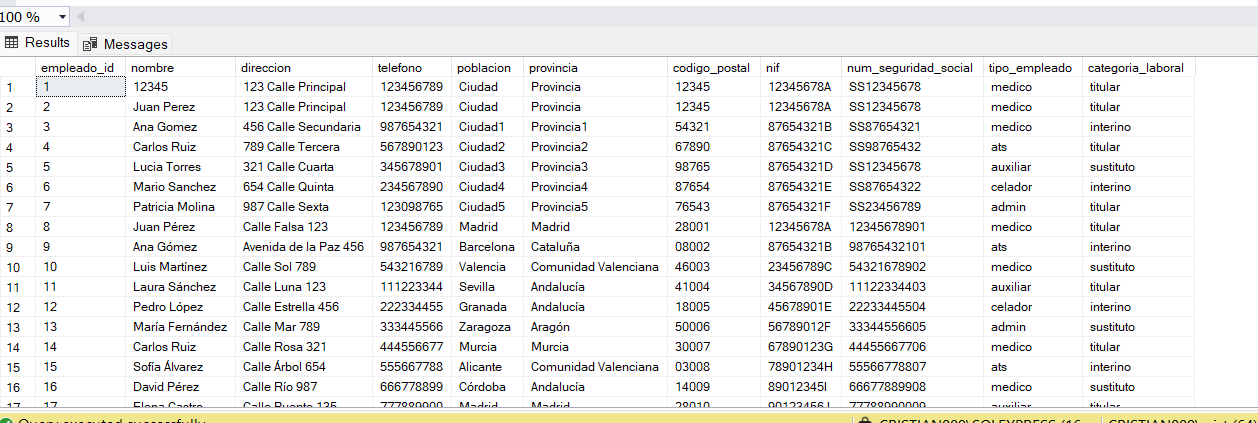
* INSTRUCCIÓN SQL:

|  |
| --- |
| * select \* from Empleados; |

* Explicación:

Mostramos la tabla empleados

* Recorte de pantalla del resultado:



1. Repita la operación con los siguientes practicando con las siguientes fechas, mostrando :
   1. 23-12-10
   2. 01/14/2023
   3. -10/08/2023
   4. 12/30-2023
   5. 12-30-2023
   6. 2023-50-20
   7. 2023-06-21

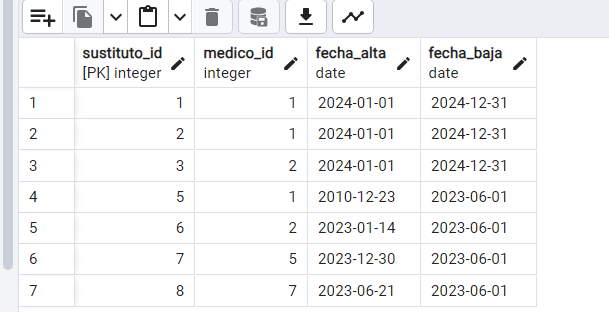
* INSTRUCCIÓN SQL:

|  |
| --- |
| -- a. 23-12-10  INSERT INTO Medicos\_Sustitutos (medico\_id, fecha\_alta, fecha\_baja)  VALUES (1, CONVERT(DATE, '23-12-10', 3), '2023-06-01');  -- b. 01/14/2023  INSERT INTO Medicos\_Sustitutos (medico\_id, fecha\_alta, fecha\_baja)  VALUES (2, '2023-01-14', '2023-06-01');  -- c. -10/08/2023  -- Este formato no es válido, SQL Server lanzará error.  -- d. 12/30-2023  -- Este formato no es válido, SQL Server lanzará error.  -- e. 12-30-2023  INSERT INTO Medicos\_Sustitutos (medico\_id, fecha\_alta, fecha\_baja)  VALUES (5, '2023-12-30', '2023-06-01');  -- f. 2023-50-20  -- Este formato no es válido, SQL Server lanzará error.  -- g. 2023-06-21  INSERT INTO Medicos\_Sustitutos (medico\_id, fecha\_alta, fecha\_baja)  VALUES (7, '2023-06-21', '2023-06-01'); |

* Explicación:

Lo que hacemos es modificar los datos en la tabla de médicos sustitutos ya que empleados no tenía un tipo de datos de esos

* Recorte de pantalla del resultado:



1. Mostrar el contenido completo de TablaX

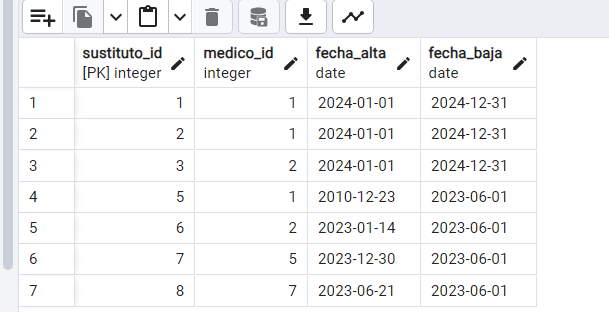
* INSTRUCCIÓN SQL:

|  |
| --- |
| * select \* from medicos\_sustitutos; |

* Explicación:

Mostramos la tabla de médicos sustitutos

* Recorte de pantalla del resultado:



1. En TablaX debe cambiar o actualizar el valor actual de la columna1 y columna2 por otros datos diferentes, pero sólo en aquellos registros cuyo valor en la clave primaria tenga sea igual x dato específico.

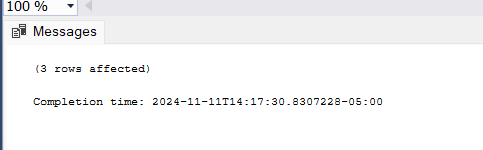
INSTRUCCIÓN SQL:

|  |
| --- |
| UPDATE ms  SET ms.fecha\_alta = '2023-12-15', ms.fecha\_baja = '2023-12-31'  FROM Medicos\_Sustitutos ms  JOIN Empleados e ON ms.medico\_id = e.empleado\_id  WHERE e.empleado\_id = 1; -- Especifica el valor de la clave primaria de Empleados |

* Explicación:

La actualización en Medicos\_Sustitutos basada en la clave primaria de Empleados. Supongamos que quieres actualizar las columnas nombre y direccion en la tabla Empleados y fecha\_alta y fecha\_baja en la tabla Medicos\_Sustitutos, pero solo para los registros cuyo empleado\_id (de Empleados) esté relacionado con un medico\_id en Medicos\_Sustitutos.

* Recorte de pantalla del resultado:

\_\_

1. Mostrar el contenido completo de TablaX

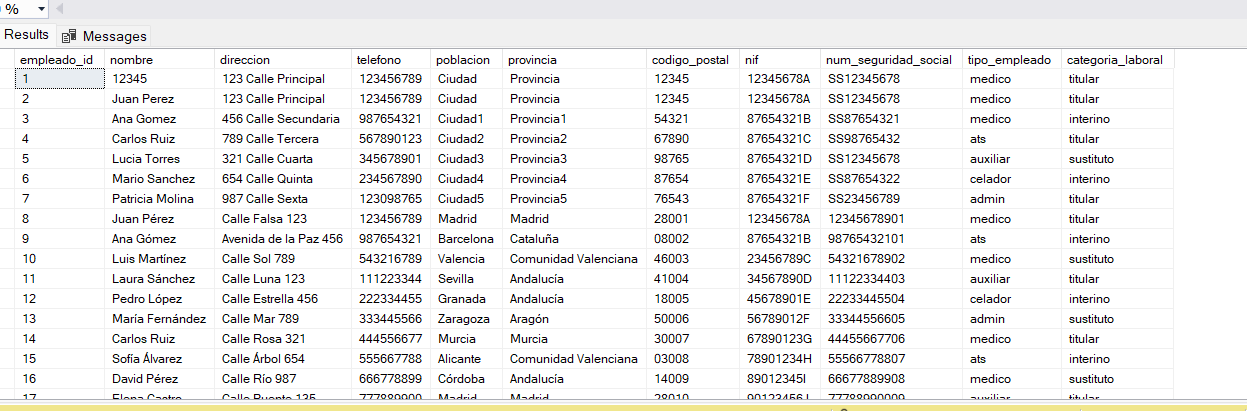
* INSTRUCCIÓN SQL:

|  |
| --- |
| * select \* from Empleados; |

* Explicación:

Consultamos a la tabla

* Recorte de pantalla del resultado:



1. En TablaX debe cambiar o actualizar el valor actual de la columna1 y columna2 por otros datos diferentes, pero sólo en aquellos registros cuyo valor en la clave primaria sea igual a alguno de los siguientes valores A, B, C, D, E

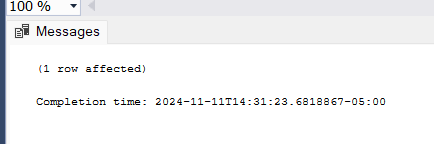
INSTRUCCIÓN SQL:

|  |
| --- |
| UPDATE Empleados  SET nombre = 'Nuevo Nombre', direccion = 'Nueva Dirección'  WHERE empleado\_id IN (A, B, C, D, E); -- Reemplaza A, B, C, D, E con los valores específicos de empleado\_id |

* Explicación:

Modificamos según el id que le pasemos

* Recorte de pantalla del resultado:



1. Mostrar el contenido completo de TablaX

* INSTRUCCIÓN SQL:

|  |
| --- |
| * select \* from Empleados; |

* Explicación:

Consultamos la tabla empleados

* Recorte de pantalla del resultado:



1. En TablaX debe cambiar o actualizar el valor actual de la columna1 y columna2 por otros datos diferentes, pero sólo en aquellos registros cuyo valor en la clave primaria sea igual a X o Y dato

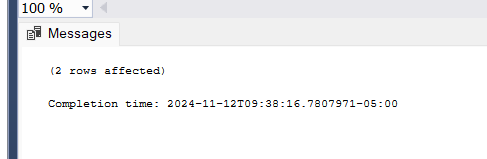
INSTRUCCIÓN SQL:

|  |
| --- |
| UPDATE Empleados  SET nombre = 'Nuevo Nombre', direccion = 'Nueva Dirección'  WHERE empleado\_id IN (7, 8); |

* Explicación:

Lo que hacemos es modificar según el dato que le pasemos

* Recorte de pantalla del resultado:



1. Mostrar el contenido completo de TablaX

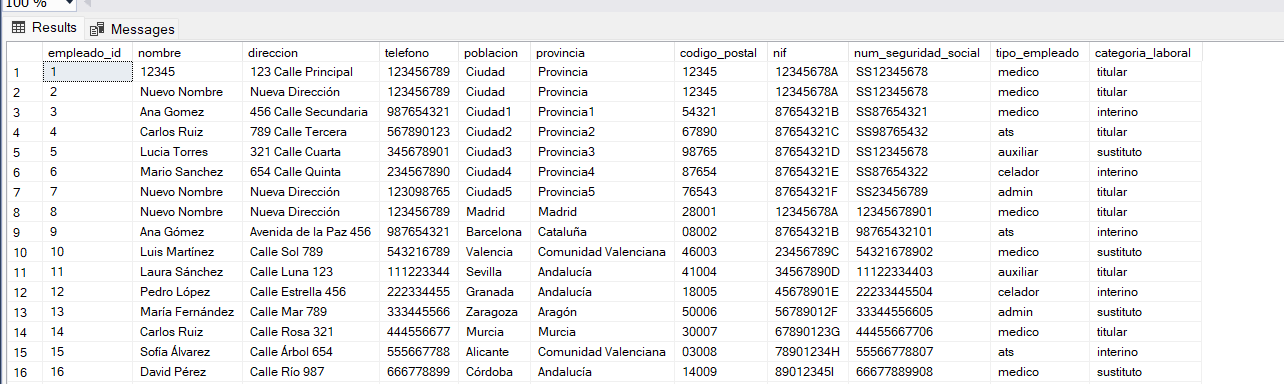
* INSTRUCCIÓN SQL:

|  |
| --- |
| * select \* from Empleados; |

* Explicación:

Mostrar la tabla empleados

* Recorte de pantalla del resultado:



1. En TablaX debe cambiar o actualizar el valor actual de la columna1 y columna2 por otros datos diferentes, pero sólo en aquellos registros cuyo valor en la clave primaria sea igual a A y B dato

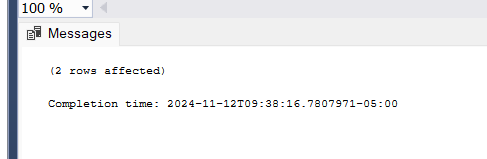
INSTRUCCIÓN SQL:

|  |
| --- |
| UPDATE Empleados  SET nombre = 'Nuevo Nombre', direccion = 'Nueva Dirección'  WHERE empleado\_id IN (7, 8); |

* Explicación:

Lo que hacemos es modificar según el dato que le pasemos

* Recorte de pantalla del resultado:



1. Mostrar el contenido completo de TablaX

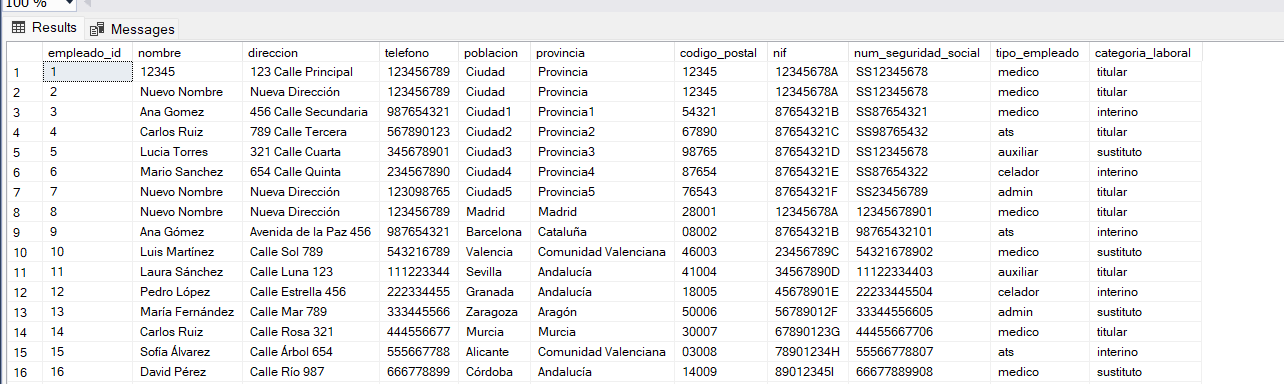
* INSTRUCCIÓN SQL:

|  |
| --- |
| * select \* from Empleados; |

* Explicación:

Mostrar la tabla empleados

* Recorte de pantalla del resultado:



1. En TablaX debe cambiar o actualizar el valor actual de la columna1 y columna2 por otros datos diferentes, pero sólo en aquellos registros cuyo valor en la columnaX este entre A y Z rango de valores

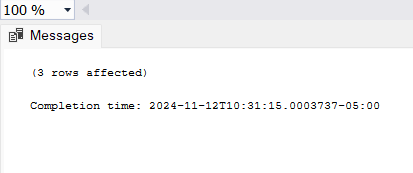
INSTRUCCIÓN SQL:

|  |
| --- |
| UPDATE Empleados  SET nombre = 'Nuevo Nombre', direccion = 'Nueva Dirección'  WHERE empleado\_id BETWEEN 5 AND 7; |

* Explicación:

Modificar los registros según lo que pasamos

* Recorte de pantalla del resultado:



1. Mostrar el contenido completo de TablaX

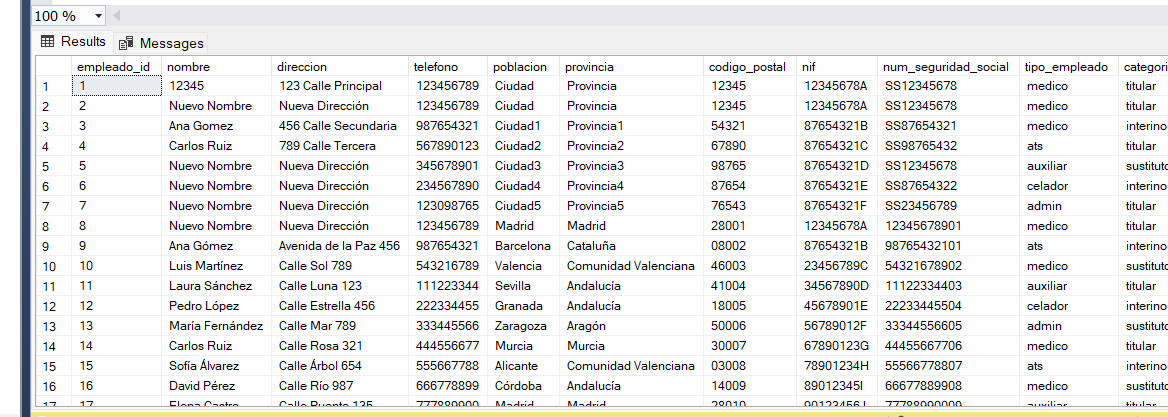
* INSTRUCCIÓN SQL:

|  |
| --- |
| * select \* from Empleados; |

* Explicación:

Consultamos la tabla empleados

* Recorte de pantalla del resultado:



1. En TablaX debe cambiar o actualizar el valor actual de la columna1 y columna2 por otros datos diferentes, pero sólo en aquellos registros cuyo valor en la columna3 sea mayor o igual a X dato

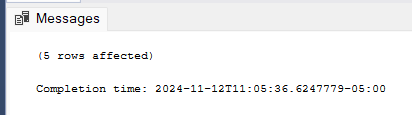
INSTRUCCIÓN SQL:

|  |
| --- |
| UPDATE Empleados  SET nombre = 'Nuevo Nombre', direccion = 'Nueva Dirección'  WHERE empleado\_id >= 19; |

* Explicación:

Modificamos la tabla que cumpla con estas características

* Recorte de pantalla del resultado:



1. Mostrar el contenido completo de TablaX

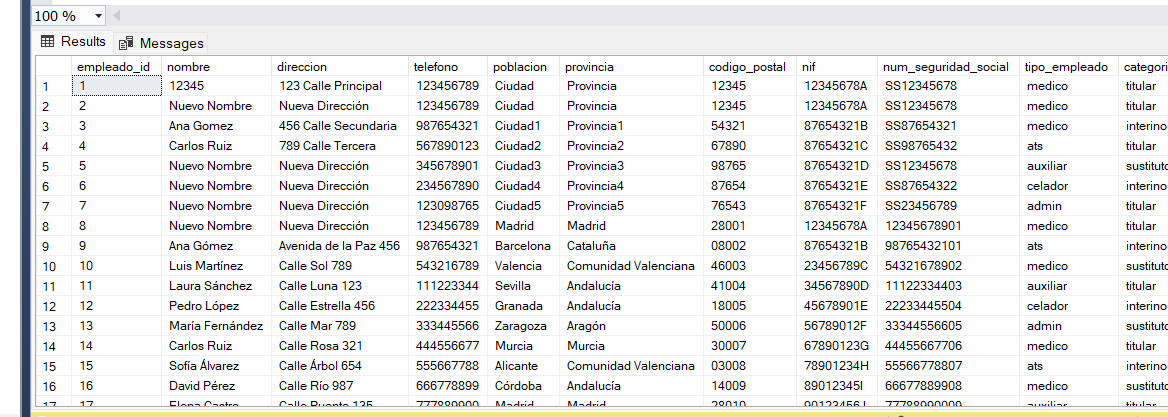
* INSTRUCCIÓN SQL:

|  |
| --- |
| * select \* from Empleados; |

* Explicación:

Consultamos la tabla empleados

* Recorte de pantalla del resultado:



1. En TablaX debe cambiar o actualizar el valor actual de la columna1 y columna2 por otros datos diferentes, pero sólo en aquellos registros cuyo valor en la columna3 esté entre diferente a X dato

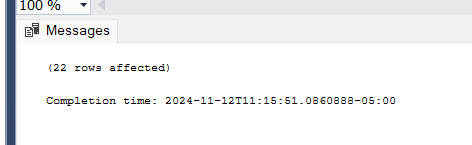
INSTRUCCIÓN SQL:

|  |
| --- |
| UPDATE Empleados  SET nombre = 'Nuevo Nombre', direccion = 'Nueva Dirección'  WHERE empleado\_id <> 1; |

* Explicación:

Modificamos los datos que cumplan con esa característica

* Recorte de pantalla del resultado:



1. Mostrar el contenido completo de TablaX

* INSTRUCCIÓN SQL:

|  |
| --- |
| * select \* from Empleados; |

* Explicación:

Consultamos la tabla empleados

* Recorte de pantalla del resultado:



1. En TablaX debe cambiar o actualizar el valor actual de la columna1 y columna2 por otros datos diferentes, pero sólo en aquellos registros cuyo valor en la columnaX esté entre A y Z rango de valores

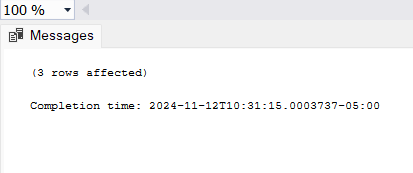
INSTRUCCIÓN SQL:

|  |
| --- |
| UPDATE Empleados  SET nombre = 'Nuevo Nombre', direccion = 'Nueva Dirección'  WHERE empleado\_id BETWEEN 5 AND 7; |

* Explicación:

Modificamos según los datos pasados

* Recorte de pantalla del resultado:



1. Mostrar el contenido completo de TablaX

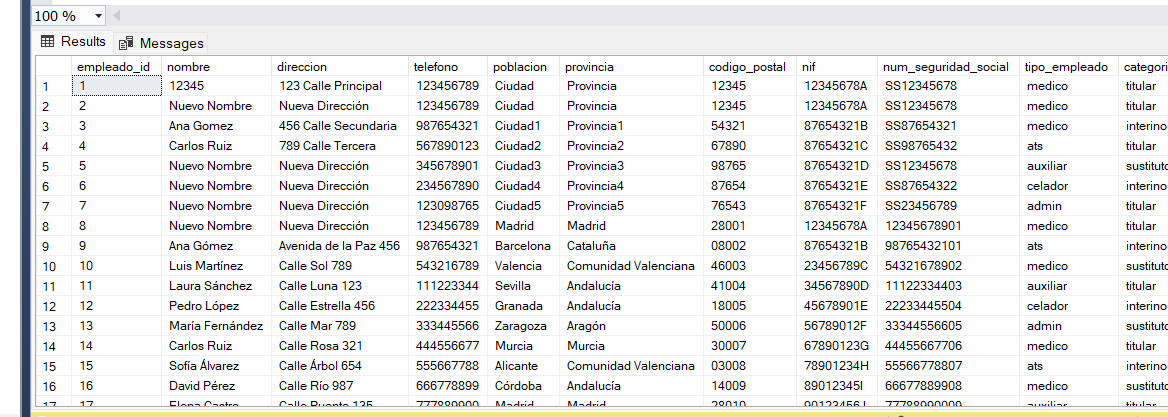
* INSTRUCCIÓN SQL:

|  |
| --- |
| * select \* from Empleados; |

* Explicación:

Consultamos los datos

* Recorte de pantalla del resultado:



1. En TablaX debe cambiar o actualizar el valor actual de la columna1 y columna2 por otros datos diferentes, pero sólo en aquellos registros cuyo valor en la columnaX inicie con el dato ABC

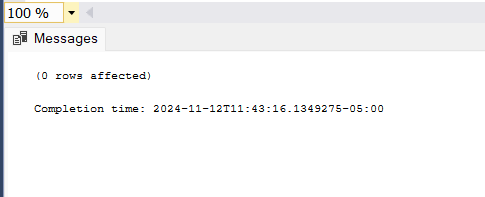
INSTRUCCIÓN SQL:

|  |
| --- |
| UPDATE Empleados SET nombre = 'Nuevo Nombre', direccion = 'Nueva Dirección' WHERE nombre LIKE 'ABC%'; |

* Explicación:

Tratamos de modificar la tabla empleados

* Recorte de pantalla del resultado:



1. Mostrar el contenido completo de TablaX

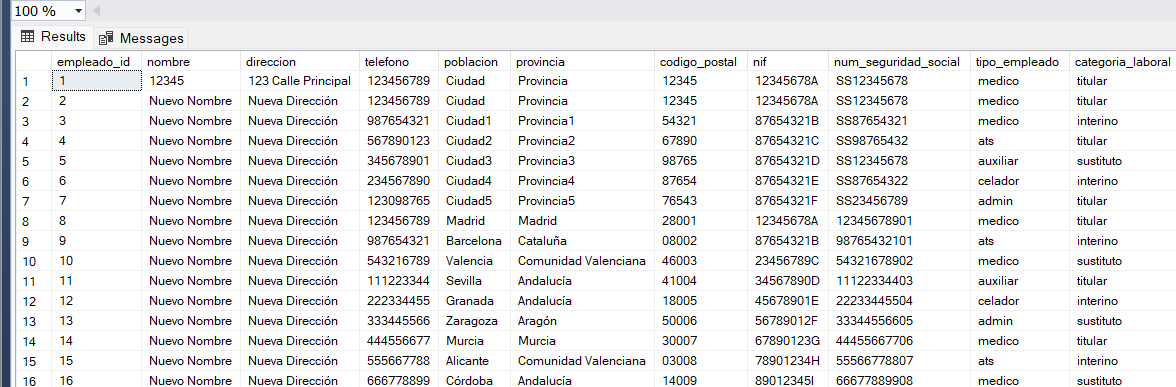
* INSTRUCCIÓN SQL:

|  |
| --- |
| * select \* from Empleados; |

* Explicación:

Consultamos la tabla empleados

* Recorte de pantalla del resultado:



1. En TablaX debe cambiar o actualizar el valor actual de la columna1 y columna2 por otros datos diferentes, pero sólo en aquellos registros cuyo valor en la columnaX termine con el dato XYZ

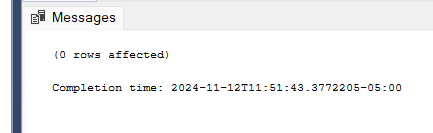
INSTRUCCIÓN SQL:

|  |
| --- |
| UPDATE Empleados  SET nombre = 'Nuevo Nombre', direccion = 'Nueva Dirección'  WHERE nombre LIKE '%XYZ'; |

* Explicación:

Tratamos de modificar los que cocida con la descripción pasada

* Recorte de pantalla del resultado:



1. Mostrar el contenido completo de TablaX

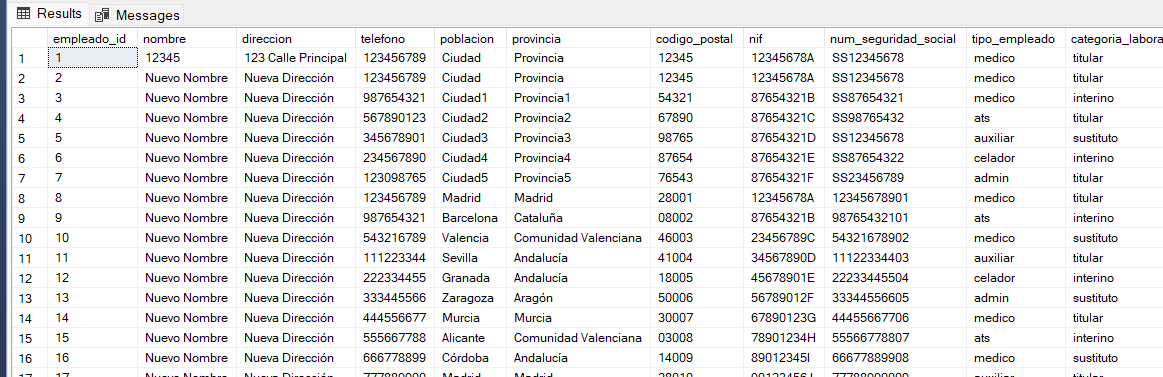
* INSTRUCCIÓN SQL:

|  |
| --- |
| * select \* from Empleados; |

* Explicación:

Consultamos la tabla

* Recorte de pantalla del resultado:



1. En TablaX debe cambiar o actualizar el valor actual de la columna1 y columna2 por otros datos diferentes, pero sólo en aquellos registros cuyo valor en la columnaX contenga con el dato ABCD

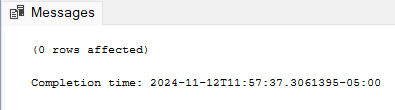
INSTRUCCIÓN SQL:

|  |
| --- |
| UPDATE Empleados  SET nombre = 'Nuevo Nombre', direccion = 'Nueva Dirección'  WHERE nombre LIKE '%ABCD%'; |

* Explicación:

Modificamos según los datos pasados

* Recorte de pantalla del resultado:



1. Mostrar el contenido completo de TablaX

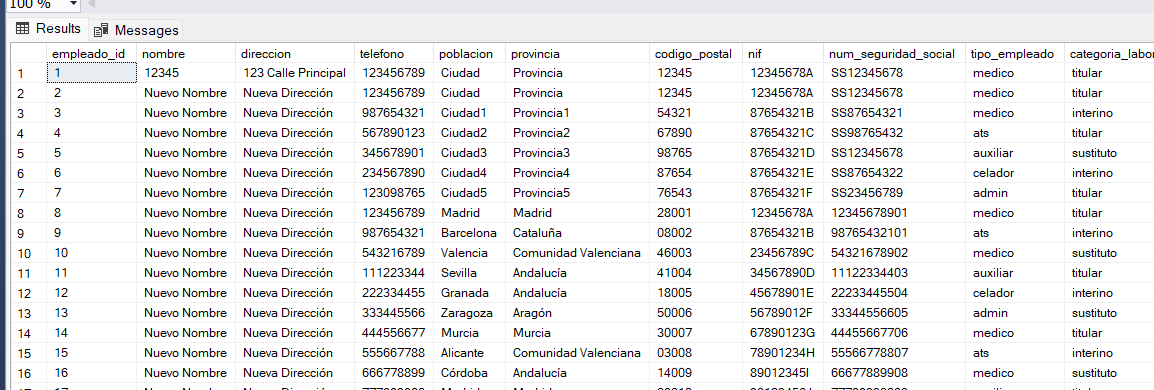
* INSTRUCCIÓN SQL:

|  |
| --- |
| * select \* from Empleados; |

* Explicación:

Consultamos la tabla empleados

* Recorte de pantalla del resultado:



1. En TablaX debe cambiar o actualizar el valor actual de la columna1 y columna2 por otros datos diferentes, pero sólo en aquellos registros cuyo valor en la columnaX sea igual a NULL

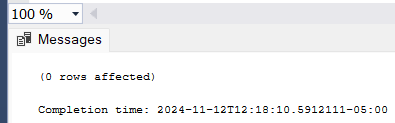
INSTRUCCIÓN SQL:

|  |
| --- |
| UPDATE Empleados  SET nombre = 'Nuevo Nombre', direccion = 'Nueva Dirección'  WHERE direccion IS NULL; |

* Explicación:

Tratamos de modificar con los datos pasados

* Recorte de pantalla del resultado:



1. Mostrar el contenido completo de TablaX

* INSTRUCCIÓN SQL:

|  |
| --- |
| * select \* from Empleados; |

* Explicación:

Consultamos la tabla empleados

* Recorte de pantalla del resultado:



1. En TablaX debe cambiar o actualizar el valor actual de la columna1 y columna2 por otros datos diferentes, pero sólo en aquellos registros cuyo valor en la columnaX sea diferente a NULL

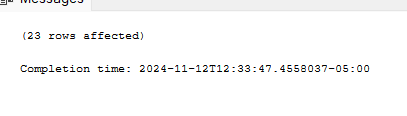
INSTRUCCIÓN SQL:

|  |
| --- |
| UPDATE Empleados  SET nombre = 'Nuevo Nombre', direccion = 'Nueva Dirección'  WHERE direccion IS NOT NULL; |

* Explicación:

Modificamos los datos que cumplan con esa característica

* Recorte de pantalla del resultado:



1. Mostrar el contenido completo de TablaX

* INSTRUCCIÓN SQL:

|  |
| --- |
| * select \* from Empleados; |

* Explicación:

Consultamos la tabla empleados

* Recorte de pantalla del resultado:



1. En la base de **bd\_ejercicio\_xyz** creen una nueva tabla llamada **TablaY**, de acuerdo al siguiente esquela de tabla:

**TablaY**(id, fecha, nombre, apellidos, asignatura, nota1, nota2, nota3, nota\_definitiva, promedio, mensaje)

Según el motor de bases de datos que esté utilizando:

* Columna **id** debe ser llave primaria artificial (autoincremental, secuencia, serie, identidad, etc)
* Columna **fecha** debe ser tener formato año-mes-dia hora:minuto:segundo
* Columnas **nombre**, **apellidos**, **asignatura** y **mensaje** deben ser de tipo caracteres
* Las columnas **nota1**, **nota2**, **nota3**, **definitiva** y **promedio** deben ser de tipo número decimal.

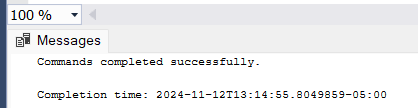
INSTRUCCIÓN SQL:

|  |
| --- |
| CREATE TABLE hospital\_ejercicio\_xyz.TablaY (  id INT IDENTITY(1,1) PRIMARY KEY,  fecha DATETIME,  nombre NVARCHAR(50),  apellidos NVARCHAR(50),  asignatura NVARCHAR(100),  nota1 DECIMAL(5, 2),  nota2 DECIMAL(5, 2),  nota3 DECIMAL(5, 2),  nota\_definitiva DECIMAL(5, 2),  promedio DECIMAL(5, 2),  mensaje NVARCHAR(255)  ); |

* Explicación:

Consultamos la tabla

* Recorte de pantalla del resultado:



1. Mostrar el contenido completo de **TablaY**

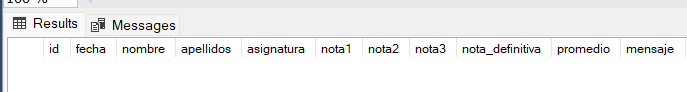
* INSTRUCCIÓN SQL:

|  |
| --- |
| * select \* from tablay; |

* Explicación:

Consultamos la tablay

* Recorte de pantalla del resultado:



1. Observe la siguiente tabla con datos, usted debe insertar cada registro en TablaY de tal manera que la tabla quede exactamente (con los datos)
2. Mostrar el contenido completo de **TablaY**

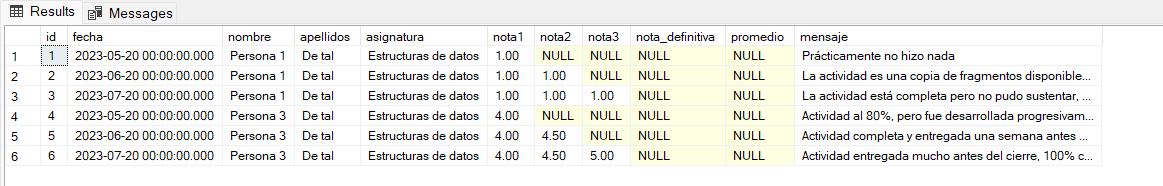
* INSTRUCCIÓN SQL:

|  |
| --- |
| select \* from tablay; |

* Explicación:

Consultamos la tablay

* Recorte de pantalla del resultado:



1. Escriba la instrucción SQL necesaria que permita actualizar automáticamente el valor correcto del cálculo para de las columnas **definitiva** y **promedio** en cada uno de los registros de **TablaY**, tomando como base el las siguientes reglas:
   1. nota1 y nota2 pesan 20% y 20% y nota3 pesa el resto del 100%.
   2. La columna **definitiva** debe contener el valor calculado según el valor que se encuentre en nota1, nota2 y nota3 en cada registro.
   3. La columna **promedio** debe contener el valor calculado según el valor que tengan las columnas nota1, nota2 y nota3.

Por ejemplo:

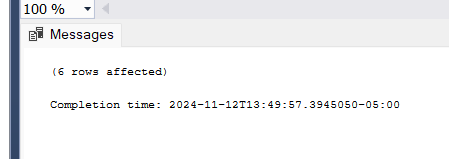
INSTRUCCIÓN SQL:

|  |
| --- |
| UPDATE TablaY  SET  definitiva =  (CASE  WHEN nota3 IS NOT NULL THEN (nota1 \* 0.2 + nota2 \* 0.2 + nota3 \* 0.6)  ELSE (nota1 \* 0.2 + nota2 \* 0.2)  END),  promedio =  (CASE  WHEN nota3 IS NOT NULL THEN (nota1 + nota2 + nota3) / 3  ELSE (nota1 + nota2) / 2  END); |

* Explicación:

Hacemos la actualización solicitada

* Recorte de pantalla del resultado:



1. Mostrar el contenido completo de **TablaY**

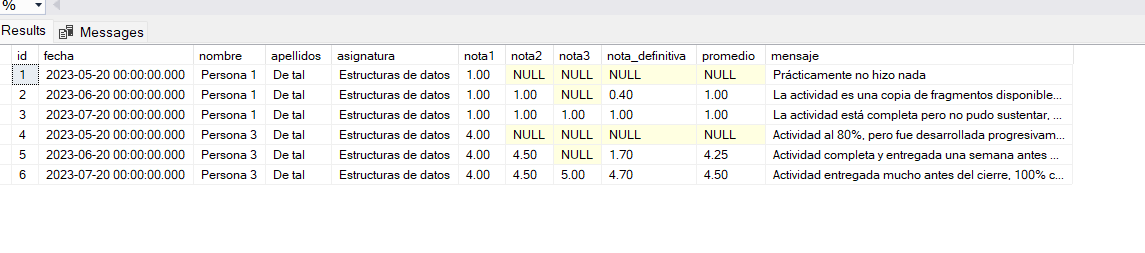
* INSTRUCCIÓN SQL:

|  |
| --- |
| * select \* from tablay; |

* Explicación:

Consultamos la tablay

* Recorte de pantalla del resultado:



1. Modificar todos las columnas de tipo carácter en **TablaY** para que ahora queden en mayúscula sostenida.

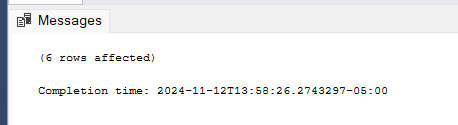
* INSTRUCCIÓN SQL:

|  |
| --- |
| UPDATE TablaY  SET  nombre = UPPER(nombre),  apellidos = UPPER(apellidos),  asignatura = UPPER(asignatura),  mensaje = UPPER(mensaje); |

* Explicación:

Modificamos la tabla según los requerimientos solicitados

* Recorte de pantalla del resultado:



1. Mostrar el contenido completo de **TablaY**

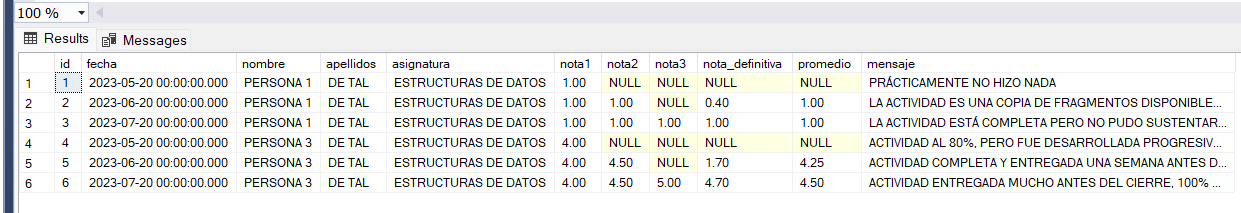
* INSTRUCCIÓN SQL:

|  |
| --- |
| * select \* from tablay; |

* Explicación:

Consultamos los datos de la tablay

* Recorte de pantalla del resultado:



1. Eliminar solo un registro de la TablaX de datos cuya llave primaria sea igual a Y valor

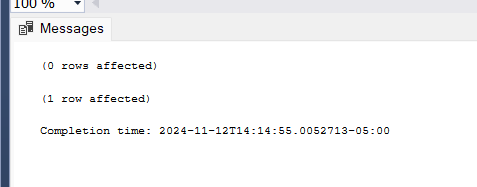
* INSTRUCCIÓN SQL:

|  |
| --- |
| DELETE FROM medicos\_sustitutos  WHERE medico\_id = 5;  DELETE FROM Empleados  WHERE empleado\_id = 5; |

* Explicación:

Lo primero que hacemos es eliminar de al que depende de el y después si eliminamos al que necesitamos eliminar, y que tiene una llave foránea

* Recorte de pantalla del resultado:



1. Mostrar el contenido completo de **TablaX**

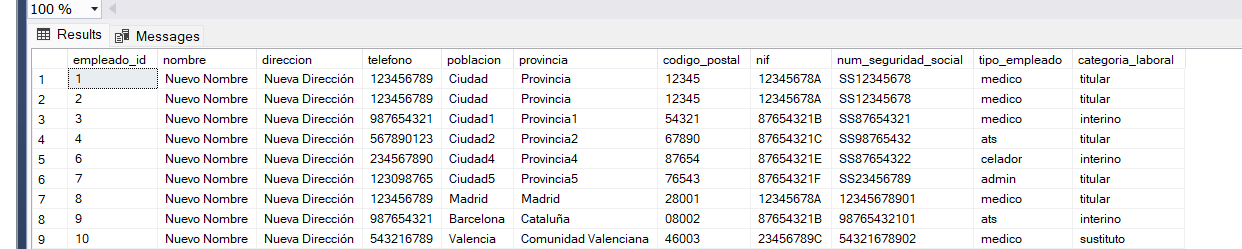
* INSTRUCCIÓN SQL:

|  |
| --- |
| * select \* from empleados; |

* Explicación:

Mostramos la tabla empleados

* Recorte de pantalla del resultado:



1. Eliminar todos los registros de la TablaX

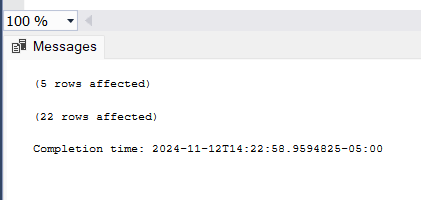
* INSTRUCCIÓN SQL:

|  |
| --- |
| DELETE FROM medicos\_sustitutos WHERE medico\_id IN (SELECT empleado\_id FROM Empleados);  DELETE FROM Empleados; |

* Explicación:

Eliminar los registros

* Recorte de pantalla del resultado:



1. Mostrar el contenido completo de **TablaY**

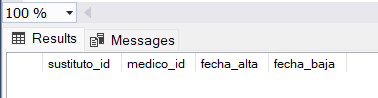
* INSTRUCCIÓN SQL:

|  |
| --- |
| * select \* from medicos\_sustitutos; |

* Explicación:

Consultamos la tabla

* Recorte de pantalla del resultado:



1. **Enlace de GitHub del repositorio para descargar los proyectos o archivos de código fuente.**

# Bibliografía

BIBLIOTECA UNICARTAGENA. (s/f). Elogim.com. Recuperado el 29 de junio de 2023,

de: <https://elibro.unicartagenaproxy.elogim.com/es/ereader/unicartagena/121283?page=12> páginas 12 - 13

BIBLIOTECA UNICARTAGENA. (s/f). Elogim.com. Recuperado el 29 de junio de 2023, páginas 29 - 32

de: <https://elibro.unicartagenaproxy.elogim.com/es/ereader/unicartagena/121283?page=29>

BIBLIOTECA UNICARTAGENA. (s/f). Elogim.com. Recuperado el 29 de junio de 2023,páginas 24 - 25

de: [https://elibro.unicartagenaproxy.elogim.com/es/ereader/unicartagena/44139?page=55](https://bdigital.uvhm.edu.mx/wp-content/uploads/2020/05/Bases-de-Datos.pdf)