Click Here

En el da de hoy quiero escribir sobre un tema muy controversial cuando se est diseando una base de datos y no se tiene una idea del concepto de relaciones. Por tal razn, hoy les explicare como se establece una relacin entre dos o ms entidades para lograr establecer integridad referencia, definir una clave primaria para una entidad y dems.Tenemos que recordar que una entidad no es ms que un objeto o una persona del mundo real. Vamos a suponer que tenemos dos entidades: Cliente, Pedido. El cliente es una persona mientras que el pedido es una cosa. El cliente en este caso realiza cierta cantidad de pedidos mensualmente, por tal razn existe una relacin de que el cliente puede tener muchos pedidos. Lo importante en este caso va hacer como van hacer las estructuras de las tablas, los tipos de datos que debemos seleccionar y como definir la clave primaria usando la restriccin (PRIMARY KEY) y establecer un enlace de datos entre las tablas usando una restriccin de clave foranea (FOREIGN KEY).Lo primero es determinar cules van hacer los atributos de cada entidad, empezaremos por la entidad Cliente:CREATE TABLE dbo.Cliente(Id_cliente int identity(1,1),nombre varchar(60) NOT NULL,apellido varchar(60) NOT NULL,cedula varchar(13) NOT NULL,direccion varchar(100) NOT NULL,telefono varchar(10) NOT NULL,correo_electronico varchar(60),fecha_registro date,estado_cliente bit DEFAULT 0,CONSTRAINT pk1 PRIMARY KEY(Id_cliente))Declaramos un campo Id_cliente con el objetivo de identificar cada cliente con un Id el cual se va a generar automticamente por identity. La importancia es con este campo vamos a identificar rpidamente el cliente dentro de la tabla. La cdula no podra ser porque ese campo identifica al cliente pero en el mundo real por tal razn es que debemos declarar un campo para identificarlo de manera nica pero, dentro de la tabla.Usamos la restriccin de columna NOT NULL para indicar que esos campos no se pueden quedar vacos, que van hacer requeridos de manera obligatoria a la hora que se vaya a insertar un nuevo registro. El campo estado_cliente lo explicarems adelante porquees importante.CREATE TABLE dbo.Pedido(Id_pedido int identity(1,1),Id_empleado int NOT NULL,Id_cliente int NOT NULL,Id_detalle int NOT NULL,fecha_pedido date NOT NULL,CONSTRAINT pk2 PRIMARY KEY(Id_pedido),CONSTRAINT fk1 FOREIGN KEY(Id_cliente)REFERENCES dbo.Cliente (Id_cliente))En la tabla de pedidos, vamos a declarar un campo que identifique el pedido de forma nica dentro de esa tabla. Cuando hago referencia al Id_empleado, Id_cliente, Id_detalle, es que estos campos van a servir para hacer una referencia de esos campos con otras tablas para establecer un enlace de datos (Relacin). Por ejemplo, el Id_empleado este campo se utilizara para establecer un enlace de datos entre la tabla de empleados que conlleve ese sistema y la tabla de pedido, el enlace estara basado en el campo Id_empleado de la tabla empleado y el campo Id_empleado de la tabla pedido, porque cada pedido lo efecta un empleado, al igual que el Id_detalle, cada pedido es nico, pero tiene un detalle de todos los productos que compro el cliente, entonces con ese campo establecemos un enlace de datos (Relacin) para que cada pedido este referenciado a un detalle.En la tabla pedido, se hizo una definicin de una restriccin de tabla PRIMARY Key y FOREIGN KEY, la restriccin PRIMARY KEY recuerden que su funcin es declarar un campo como clave principal, sea que los datos que recaigan en este campo deben ser nicos, sea no se pueden repetir. La restriccin FOREIGN KEY en cambio, lo que hace es en este caso es establecer un enlace de datos entre el campo Id_cliente de la tabla cliente y el campo Id_cliente de la tabla pedido.Ahora la pregunta seria, porque no declare la restriccin FOREIGN KEY en la tabla Cliente? La razn es sencilla, porque la tabla pedido es la que est haciendo referencia a la tabla cliente a travs del campo Id_cliente por tal razn es que esta restriccin se debe definir en la tabla en la cual se estn haciendo referencia.En la tabla Cliente hay un campo muy especial, este es estado_cliente, cual es la funcin de este campo?La importancia de este campo es la siguiente:Como el Id_cliente est referenciado en la tabla Pedido, si existe un registro en la tabla cliente, por ejemplo:INSERT INTO Cliente(nombre,apellido,cedula,direccion,telefono,correo_electronico,fecha_registro)VALUES(Joseph Arquimedes,Collado Tineo,00000000000,Palmar de tigua,8090000000,joseph@sqltrainning.com,GETDATE())SELECT*FROM dbo.ClienteTenemos un registro en la tabla cliente, que pasa, que si queremos eliminar ese registro fcilmente lo podemos hacer usando la sentencia DELETE FROMDELETE FROM ClienteWHERE Id_cliente=1Vuelvo a insertar:INSERT INTO Cliente(nombre,apellido,cedula,direccion,telefono,correo_electronico,fecha_registro)VALUES(Joseph Arquimedes,Collado Tineo,00000000000,Palmar de tigua,8090000000,joseph@sqltrainning.com,GETDATE())Pero que pasara si el cliente tiene un pedido, osea que exista un pedido en el cual se este haciendo referencia al Id_cliente de ese registro en la tabla cliente en la tabla pedido.INSERT INTO PedidoVALUES(56,2,1,GETDATE())SELECT*FROM dbo.PedidoNo especifico los campos por que voy a insertar en todosAhora, si tratamos de eliminar el cliente de Id_cliente=2, miren lo que pasaDELETE FROM ClienteWHERE Id_cliente=2El error es que no podemos eliminar ese cliente por que el esta referenciado en la tabla pedido, y si esta referenciado y no podemos eliminarlo, aqu es donde entra en juego el campo estado_cliente, porque si tenemos una aplicacin en la cual se requiere eliminar un cliente que tiene mucho tiempo que no realiza pedidos o cualquier otra razn, lo que vamos hacer es que en vez de eliminarlo vamos ha actualizar su campo estado_cliente=1 y en la aplicacin la consulta seria la que se va a desplegar en la pantalla del usuarioSELECT*FROM ClienteWHERE estado_cliente=0Los que estn en cero son los que me representan los clientes activos, mientras que los 1 representan que estn inactivos, y as, si la empresa requiere hacer un reporte de pedidos antiguos, como los cliente inactivos se encuentras en la tabla, podemos determinar que cantidad de pedidos hicieron esos cliente, por que no se va a perder esa relacin que tiene con los pedidos.VentajasLa ventaja de todo esto es que si tenemos una aplicacin que se encarga de validar que no se puede eliminar un cliente por que tiene referencia de pedidos por ejemplo, en la base de datos tambin se tiene que validar eso, por que si un usuario se conecta va remota al servidor, si no hay una integridad referencial el podra fcilmente eliminar un registro de la tabla cliente o pedido y se perdera la relacin con cualquiera de los registro que fuesen eliminados. Tambin buscamos establecer mecanismos de seguridad e integridad de informacin de nuestro sistema.Espero que aprovechen este articulo, En el mundo de los sistemas de gestin de bases de datos, establecer relaciones entre tablas es crucial para extraer informacin valiosa y realizar anlisis de datos complejos. En este tutorial, exploraremos el concepto de crear mltiples relaciones entre las mismas tablas en SQL Server.Antes de adentrarnos en los detalles tcnicos, vamos a entender los diferentes tipos de relaciones que pueden existir entre tablas:Uno a uno: Esta relacin existe cuando un solo registro en una tabla est relacionado con solo un registro en otra tabla y viceversa.Uno a muchos: Esta relacin existe cuando un solo registro en una tabla puede estar relacionado con mltiples registros en otra tabla, pero cada registro en la segunda tabla solo puede estar relacionado con un registro en la primera tabla.Muchos a muchos: Esta relacin existe cuando mltiples registros en una tabla pueden estar relacionados con mltiples registros en otra tabla. Normalmente se utiliza una tabla de unin para evitar la duplicacin de datos.Ahora, veamos cmo podemos implementar prcticamente mltiples relaciones entre las mismas tablas en SQL Server.Primero, necesitamos crear una base de datos y las tablas necesarias. Por ejemplo, consideremos un escenario en el que tenemos una base de datos de ventas de una librera con una tabla llamada VentasMensuales que contiene informacin sobre las ventas mensuales, incluyendo la fecha del pedido, fecha de envo, clave del producto, ID del cliente y ventas totales realizadas.Aqu tienes un ejemplo de cmo podemos crear la tabla VentasMensuales:CREATE TABLE VentasMensuales(FechaPedido DATE, FechaEnvio DATE, ClaveProducto VARCHAR(6), IDCliente VARCHAR(8), VentasTotales INT);Una vez que tengamos nuestra estructura de tabla en su lugar, podemos poblarla con datos utilizando la instruccin INSERT.Ahora, pasemos a crear las relaciones entre las tablas. En SQL Server, podemos utilizar restricciones de clave externa (foreign key constraints) para establecer relaciones entre tablas. Podemos crear mltiples claves externas entre las mismas tablas para representar diferentes relaciones.Por ejemplo, supongamos que queremos establecer dos relaciones entre la tabla VentasMensuales y una tabla Fecha. Una relacin se basar en la fecha del pedido y la otra relacin se basar en la fecha de envo.Para crear las relaciones, necesitamos definir las claves externas en la tabla VentasMensuales que hagan referencia a la clave primaria en la tabla Fecha. Aqu tienes un ejemplo de cmo podemos crear las claves externas:ALTER TABLE VentasMensualesADD CONSTRAINT FK_FechaPedido FOREIGN KEY (FechaPedido) REFERENCES Fecha(Fecha);ALTER TABLE VentasMensualesADD CONSTRAINT FK_FechaEnvio FOREIGN KEY (FechaEnvio) REFERENCES Fecha(Fecha);Con estas claves externas en su lugar, ahora podemos consultar los datos y realizar varios clculos y visualizaciones basados en las diferentes relaciones.Por ejemplo, podemos calcular las ventas totales por fecha de pedido y fecha de envo utilizando consultas SQL:SELECT FechaPedido, SUM(VentasTotales) AS VentasPorFechaPedidoFROM VentasMensualesGROUP BY FechaPedido;SELECT FechaEnvio, SUM(VentasTotales) AS VentasPorFechaEnvioFROM VentasMensualesGROUP BY FechaEnvio;Al utilizar estas mltiples relaciones, podemos obtener informacin valiosa sobre los patrones de ventas de la librera y optimizar su estrategia de cadena de suministro.En conclusin, crear mltiples relaciones entre las mismas tablas en SQL Server nos permite analizar datos desde diferentes perspectivas y extraer informacin significativa. Al entender los fundamentos de las relaciones de bases de datos y utilizar claves externas, podemos aprovechar todo el potencial de nuestros datos.Gracias por leer este tutorial sobre cmo crear mltiples relaciones entre las mismas tablas en SQL Server. Estn atentos para ms consejos y trucos de SQL Server!Click to rate this post!Preguntado por: Jan Ortz Tercero|ltima actualizacin: 10 de abril de 2022Puntuacin: 4.2/5 (29 valoraciones) Para ver los objetos de los que depende una tabla En el Explorador de objetos, expanda Bases de datos, expanda una base de datos y, a continuacin, Tablas. Haga clic con el botn derecho en una tabla y, despus, haga clic en Ver dependencias.Cmo ver los atributos de una tabla en SQL Server?Uso de SQL Server Management Studio Expanda Bases de datos, haga clic con el botn derecho en la base de datos que quiera ver y, despus, haga clic en Propiedades. En el cuadro de dilogo Propiedades de la base de datos , seleccione una pgina para ver la informacin correspondiente.Para realizar consultas sobre las tablas de las bases de datos disponemos de la instruccin SELECT. Con ella podemos consultar una o varias tablas. Es sin duda el comando ms verstil del lenguaje SQL. Existen muchas clusulas asociadas a la sentencia SELECT (GROUP BY, ORDER, HAVING, UNION).Para ver los que tenemos debemos usar las tablas del sistema, concretamente: INFORMATION_SCHEMA. TABLE_CONSTRAINTS. En esta tabla podemos encontrar la siguiente informacin: CONSTRAINT_CATALOG = nombre de la base de datos.Las restricciones en SQL Server son reglas y restricciones predefinidas que se aplican en una sola columna o en varias columnas, relacionados a los valores permitidos en las columnas, para mantener la integridad, precisin y confiabilidad de los datos de esa columna.27 preguntas relacionadas encontradasPuede colocar restricciones para limitar el tipo de dato que puede ingresarse en una tabla. Dichas restricciones pueden especificarse cuando la tabla se crea por primera vez a travs de la instruccin CREATE TABLE, o luego de crear la tabla a travs de la instruccin ALTER TABLE.Consultar datos de una tabla El comando SELECT va seguido de la lista de campos de la tabla (separados por coma), luego se escribe la palabra reservada FROM , y por ltimo, el nombre de la tabla que deseas consultar.Relacionar los atributos de una tabla con otraHaga clic con el botn derecho en la capa que desea relacionar, elija Uniones & Relaciones y, a continuacin, haga clic en Relacin. ... Elija el campo de la capa en que se va a basar la relacin.Para crear una relacin entre dos tablas de datos Se abre el cuadro de dilogo Relacin, que rellenar el cuadro Tabla secundaria con la tabla a la que arrastr el objeto Relation.Abra la base de datos de destino de Access. En la pestaa Datos externos, haga clic en Base de datos ODBC. Haga clic en Vincular al origen de datos creando una tabla vinculada > Aceptar y siga los pasos del asistente. Si el archivo .Excel solo puede crear la relacin si una columna contiene valores nicos....Si Relaciones est atenuado, significa que la hoja de clculo contiene una sola tabla.En el cuadro Administrar relaciones, haga clic en Nueva.En el cuadro Crear relacin, haga clic en la flecha abajo de Tabla y seleccione una tabla en la lista.Una relacin de tabla hace coincidir los datos de los campos clave (a menudo un campo con el mismo nombre en ambas tablas). En la mayora de los casos, estos campos coincidentes son la clave principal de una tabla, que proporciona un identificador nico para cada registro, y una clave externa de la otra tabla.En este nuevo post vamos a ver la sentencia de MySQL SELECT, su funcionalidad es la de realizar consultas sobre una o varias tablas de una base de datos para extraer un determinado nmero de filas (resultados)....MySQL SELECT: Realizar consultas a una base de datosALTER TABLE.UPDATE.INSERT.TRUNCATE TABLE.La clusula CONSTRAINT se usa en las instrucciones ALTER TABLE y CREATE TABLE para crear o eliminar restricciones. Hay dos tipos de clusulas CONSTRAINT: uno para crear una restriccin en un nico campo y otro para crear una restriccin en varios campos.La constraint CHECK permite restringir el rango de valores permitidos para un campo dado. Este rango puede especificarse de manera explcita para una columna en particular, o utilizando los valores presentes en otras columnas.Las limitaciones o (Constraints) de SQL se utilizan para especificar reglas para los datos de una tabla. Si hay alguna violacin entre la restriccin y accion de datos, la accin se aborta por la restriccin.Se utilizan obligatoriamente las restricciones de tipo 2 cuando la restriccin afecta a un grupo de columnas o cuando queremos definir ms de una CONSTRAINT para una columna (slo se puede definir una restriccin1 en cada columna). La clusula PRIMARY KEY se utiliza para definir la clave principal de la tabla.Es simplemente la ausencia de cualquier valor. Cuando definimos un campo en una tabla, es posible forzar que dicho campo acepte o rechace guardar valores nulos en l. Esto asegurar que el campo en cuestin es que rechace -o acepte- cualquier intento (en un Insert o Update) de dejar el campo con un valor nulo.NULL Palabras clave Devuelve verdadero si el valor proporcionado es NULO y falso si el valor proporcionado no es NULO. NOT NULL es la palabra clave que realiza la comparacin booleana. Devuelve verdadero si el valor proporcionado no es NULO y falso si el valor proporcionado es nulo.Existen tres tipos de relaciones entre tablas:Uno a uno. Cuando cada elemento de cada tabla solo aparece una vez. ... Uno a varios. Cuando un elemento en una tabla puede tener una relacin con varios elementos de otra tabla. ... Varios a varios.La relacin de una base de datos es el vnculo que se establece entre distintos elementos de las tablas que la conforman. En este tipo de relaciones es fundamental el uso de los campos de llave primaria (primary key) que son los que se relacionan con otros registros de otras tablas.Las relaciones que se establecen entre los diferentes elementos de dos tablas en una base de datos relacional pueden ser de tres tipos distintos: Relaciones uno a uno, se establecen entre una entidad de una tabla y otra entidad de otra tabla. Un ejemplo aparece en la figura 90.Insertar una tabla dinmica en Excel es muy sencillo pues solo basta con seleccionar el rango de celas que contiene los valores para operar la tabla dinmica y luego solo basta con aplicar el diseo ms adecuado para representar la informacin. Como hemos visto, el modelo relacional se basa en las relaciones existentes entre las diferentes tablas de base de datos (BBDD). En esta seccin nos centraremos en explicar y ejemplificar los diferentes tipos de relaciones imprescindibles en la administracin de bases de datos relacionales.Los ejemplos de esta seccin se basan en capturas de diagramas definidos desde el editor de Sql Server Management Studio (SSMS). Esta seccin solo tiene objetivo de explicar conceptualmente los fundamentos de los diferentes tipos de relaciones.Relacin 1 a 1Este tipo de relacin entre dos tablas se establece cuando un registro de una tabla solo puede estar vinculado a un nico registro en otra tabla. Este tipo de relacin se utiliza generalmente para relaciones exclusivas cuando tenemos gran cantidad de campos. Dicha relacin nos permite dividir la informacin en tablas ms pequeas con menos cantidad de campos y facilitar la gestin de nuestras bases de datos.Dicha relacin se establece a travs de una Foreign Key que se vincula directamente con la Primary Key de la tabla principal. En este tipo de relaciones ser indistinto que tabla considerar principal y que tabla dependiente para establecer la Foreign Key, siendo ello decisin de diseo del administrador de acuerdo a los datos que guardar en las tablas. En este ejemplo podemos observar cmo se establece una relacin 1 a 1 entre las tablas paises y banderas, ya que un pas solo puede tener asociada una bandera y una bandera solo puede pertenecer a un pas. Constituye un tipo de relacin muy utilizada que nos permite organizar los datos de manera eficiente y sencilla.Se ha considerado la tabla paises como tabla principal. De esta manera nos aseguramos que no se puedan guardar banderas en el sistema de pases que an no se han registrado en la base de datos. Relacin 1 a nEste tipo de relacin entre dos tablas, tambin llamada relacin 1 a muchos, se establece cuando un registro de una tabla puede estar asociado a varios registros de otra tabla, pero no de manera inversa.En este caso la Foreign Key siempre se definir en la tabla secundaria o dependiente, la cual podr repetirse n veces en funcin a la Primary Key con la que se establece la relacin. En este ejemplo podemos ver como se establece una relacin 1 a n entre la tabla clientes y la tabla pedidos ya que un cliente puede tener varios pedidos y varios pedidos solo pueden pertenecer a un cliente.La Primary Key de un cliente podra repetirse varias veces en la Foreign Key de la tabla pedidos (id_clientes) para guardar la informacin de todos los pedidos realizados por un cliente determinado. Relacin n a nEl tipo de relacin n a n o muchos a muchos se establece cuando varios registros de una tabla se asocian a varios registros de la otra. Para ello es necesario definir una tercera tabla intermedia para establecer primero dos relaciones 1 a n y poder vincular correctamente las dos tablas principales.Usualmente la Primary Key de la tercera tabla intermedia ser una clave Compuesta o Composite key de las Primary Key de las dos tablas principales y a travs de dichas claves formar dos Foreign Key a cada tabla con la que establece relacin. Adems, esta tercera tabla aparte de establecer las relaciones intermedias, tambin podr guardar informacin adicional como por ejemplo informacin de auditora u otros datos importantes del establecimiento de la relacin. En este ejemplo podemos ver como se establece una relacin n a n entre la tabla alumnos y asignaturas ya que los alumnos pueden tener varias asignaturas y las asignaturas pueden tener varios alumnos.Vemos que para establecer dicha relacin se necesita una tercera tabla intermedia llamada asignaturasAlumnos que adems guarda informacin adicional como la fecha en la que un alumno determinado se matricula de una asignatura especfica.Observamos igualmente que la Primary Key de la tabla asignaturasAlumnos es la concatenacin de idAlumnos e idAsignaturas dando lugar a una CompositeKey. A su vez idAlumnos forma la ForeignKey que se relaciona con la tabla alumnos e idAsignaturas la ForeignKey vinculada a la tabla asignaturas. Buen da amigos de IncanatoIt, en este artculo se muestra elvdeodondeimplementan las ltimas tablas del modelo relacional de la base de datos, se disea el diagrama relacional de base de datos, se explica de manera detallada las relaciones entre las tablas, relaciones de uno a muchos, utilizando el asistente grfico de Microsoft SQL Server Management Studio 2014.Puedes seguir el curso Completo desde: Puedes descargar el Material del Curso desde,Archivos del Curso, Backup y Presentaciones: modo de relacionar registros entre tablas es por tanto mediante referencias, para lo cual se usan los identificadores definidos como claves primarias y forneas. Clave primariaEn el diseo de bases de datos relacionales, se llama clave primaria (Primary Key) a un campo o a una combinacin de campos que identifica de forma nica a cada fila de una tabla. Una clave primaria comprende de esta manera una columna o conjunto de columnas. No puede haber dos filas en una tabla que tengan la misma clave primaria.Una clave primaria debe identificar a todas las posibles filas de una tabla y no nicamente a las filas que se encuentran en un momento determinado. Ejemplos de claves primarias son DNI (asociado a una persona) o ISBN (asociado a un libro). Las guas telefnicas y diccionarios no pueden usar nombres o palabras o nmeros del sistema decimal de Dewey como claves candidatas, porque no identifican unvocamente nmeros de telfono o palabras. Clave forneaEn el contexto de bases de datos relacionales, una clave fornea o clave ajena (o Foreign Key FK) es una limitacin referencial entre dos tablas. La clave fornea identifica una columna o grupo de columnas en una tabla (tabla hija o referendo) que se refiere a una columna o grupo de columnas en otra tabla (tabla maestra o referenciada). Las columnas en la tabla referendo deben ser la clave primaria u otra clave candidata en la tabla referenciada. Los valores en una fila de las columnas referendo deben existir solo en una fila en la tabla referenciada. As, una fila en la tabla referendo no puede contener valores que no existen en la tabla referenciada. De esta forma, las referencias pueden ser creadas para vincular o relacionar informacin. Esto es una parte esencial de la normalizacin de base de datos. Mltiples filas en la tabla referendo pueden hacer referencia, vincularse o relacionarse a la misma fila en la tabla referenciada. Mayormente esto se ve reflejado en una relacin uno (tabla maestra o referenciada) a muchos (tabla hija o referendo).Diagrama Relacional Base de Datos en Sql Server 2014 Dentro de las opciones que nos ofrece nuestra base de datos, encontramos la opcin Diagrama de la Base de Datos, que nos permite realizar las relaciones de las tablas en forma Grfica. Para acceder a esta opcin le damos click derecho y crearemos un nuevo Diagrama de Base de Datos.Imagen 1: Diagrama relacional de la Base de datos del Curso.Relaciones entre tablas en Sql Server 2014 (4-35) Bases de Datos en Microsoft Sql Server 2014Saludos Imperio, un abrazo a la distancia. Uno a Uno (1:1): Cada fila de una tabla est relacionada con una sola fila de otra tabla.Uno a Muchos (1:N): Una fila de una tabla puede estar relacionada con mltiples filas de otra tabla.Muchos a Muchos (M:N): Muchas filas de una tabla pueden estar relacionadas con muchas filas de otra tabla. Para implementar esta relacin, se utiliza una tabla intermedia.Clave Primaria (Primary Key): Un campo o combinacin de campos que identifican de manera nica cada fila en una tabla.Clave Fornea (Foreign Key): Un campo en una tabla que establece una relacin con la clave primaria de otra tabla. Imaginemos que estamos creando un sistema para gestionar estudiantes y cursos. Necesitamos dos tablas: estudiantes y cursos, y una tabla intermedia inscripciones para establecer la relacin de muchos a muchos. Tabla: estudiantes CREATE TABLE estudiantes (id INT PRIMARY KEY, nombre VARCHAR(100), edad INT); Respuesta: | id | nombre | edad ||----|----------|------|| 1 | Urian Viera | 20 || 2 | Mara Gonzlez | 22 || 3 | Brenda Viera | 19 || 4 | Ana Martnez | 21 || 5 | Luis Fernndez | 23 | Tabla: cursos CREATE TABLE cursos (id INT PRIMARY KEY, nombre VARCHAR(100), creditos INT); Respuesta: | id | nombre | creditos ||----|----------|----------|| 1 | Matemticas | 3 || 2 | Historia | 4 || 3 | Ciencias | 3 || 4 | Lengua Espaola | 2 || 5 | Programacin | 5 | Tabla: cursos CREATE TABLE inscripciones (estudiante_id INT, curso_id INT, PRIMARY KEY (estudiante_id, curso_id), FOREIGN KEY (estudiante_id) REFERENCES estudiantes(id), FOREIGN KEY (curso_id) REFERENCES cursos(id)); La tabla inscripciones conecta a los estudiantes con los cursos, permitiendo manejar inscripciones de manera eficiente y asegurando la integridad referencial entre las tablas estudiantes y cursos.estudiante_id INT: Esta columna almacena el identificador del estudiante que se inscribe en un curso.curso_id INT: Esta columna almacena el identificador del curso al que el estudiante se inscribe.PRIMARY KEY (estudiante_id, curso_id): Define una clave primaria compuesta, asegurando que cada combinacin de estudiante y curso sea nica. Esto evita inscripciones duplicadas para el mismo estudiante en el mismo curso.FOREIGN KEY (estudiante_id) REFERENCES estudiantes(id): Establece una relacin entre estudiante_id en la tabla inscripciones y id en la tabla estudiantes, garantizando que solo se puedan inscribir estudiantes existentes.FOREIGN KEY (curso_id) REFERENCES cursos(id): Similarmente, establece una relacin entre curso_id en la tabla inscripciones y id en la tabla cursos, asegurando que solo se puedan inscribir en cursos existentes.Relaciones en Acciones 1. Insertar datos en las tablasInsertamos algunos estudiantes y cursos: INSERT INTO estudiantes (id, nombre, edad) VALUES (1, 'Urian Viera', 20),(2, 'Mara Gonzlez', 22); INSERT INTO cursos (id, nombre, creditos) VALUES (1, 'Matemticas', 3),(2, 'Historia', 4); 2. Inscribir estudiantes a cursosAhora inscribimos a los estudiantes en los cursos: INSERT INTO inscripciones (estudiante_id, curso_id) VALUES (1, 1), (1, 2), (2, 1); 3. Consultar inscripcionesPara ver qu cursos estn inscritos los estudiantes, puedes realizar una consulta con un JOIN: SELECT e.nombre AS estudiante, c.nombre AS curso FROM inscripciones iJOIN estudiantes e ON i.estudiante_id = e.idJOIN cursos c ON i.curso_id = c.id; Respuesta: | estudiante | curso ||----------|----------|| Urian Viera | Matemticas || Urian Viera | Historia || Mara Gonzlez | Matemticas | Al definir claves forneas, asegrate de que las filas en la tabla relacionada existan para mantener la integridad referencial.Es posible establecer restricciones adicionales en las claves forneas, como *ON DELETE CASCADE, que elimina automticamente las filas relacionadas si se elimina la fila principal. Estoy leyendo esta pregunta en 2015 y estoy usando SQL Server 2012. En este escenario, para ver las dependencias de una tabla, puede seguir estos pasos: 1. En la carpeta raz de su base de datos, hay una carpeta llamada Diagramas de bases de datos. Expanda esta base de datos y haga clic en 's' en la ventana emergente que aparecer; 3. Haga clic con el botn derecho en el campo que sospecha que tiene una dependencia, normalmente tienen las letras ID en sus nombres, por ejemplo, estoy en la base de datos EPM y en la tabla MSP_Projects tenemos el campo Proj_UID, haga clic con el botn derecho en el campo; 4. En el men contextual que aparece, seleccione el elemento Relaciones. En el sitio izquierdo de la ventana ver las claves forneas relacionadas con esta clave primaria, y en el lado derecho de la ventana ver las propiedades de la relacin existente.

Microsoft sql server management studio relationships. Sql server management studio. Sql server management studio table diagram.