Tablas de muchos a muchos

Relaciones de muchos a muchos

Una relación de muchos a muchos se produce cuando varios registros de una tabla se asocian a varios registros de otra tabla. Por ejemplo, existe una relación de muchos a muchos entre los clientes y los productos: los clientes pueden comprar varios productos y los productos pueden ser comprados por muchos clientes.

Por lo general, los sistemas de bases de datos relacionales no permiten implementar una relación directa de muchos a muchos entre dos tablas. Tenga en cuenta el ejemplo de seguimiento de facturas. Si había muchas facturas con el mismo número de factura y uno de sus clientes preguntó acerca de ese número de factura, no sabría a qué número se refería. Este es el motivo por el que se debe asignar un valor exclusivo a cada factura.

Para evitar este problema, puede dividir la relación de muchos a muchos en dos relaciones de uno a muchos mediante el uso de una tercera tabla denominada tabla de unión, tabla de rompimientos o tabla intermedia. Cada registro de una tabla de unión incluye un campo de coincidencia que contiene el valor de las claves principales de las dos tablas que se unen. (En la tabla de unión, estos campos de coincidencia son claves externas). Estos campos de clave externa se rellenan con datos, ya que los registros de la tabla de unión se crean desde cualquiera de las tablas que se unen.

Instrucciones

A continuación, se le brindaran tres tablas con una relación o cardinalidad de muchos a muchos,analicé,identifique y realice las tablas de unión o rompimiento para cada relación.

1.-Relación muchos a muchos entre usuarios y grupo:

Crear una tabla usuarios(No confundir con la tabla realizada anteriormente llamada "usuario")

Sus atributos serán: id int llave primaria, nombre varchar (25) NOT NULL,apellido varchar (25) NOT NULL,fecha_nacimiento DATE.

Crear la tabla grupo con los atributos:

id int llave primaria,nombre varchar (25) NOT NULL,descripcion varchar (75), fecha_creacion DATE.

Cree la tabla de unión usuario_grupo donde se almacene la información del usuario y la información del grupo:

us_id int not null(llave foránea referencia de usuarios)int,gr_id int not null (llave foránea referencia de grupo).

Crear la llave compuesta usando el id de usarios y de grupo

2.-Relacion muchos a muchos entre habitaciones y huéspedes

Crear la tabla habitaciones con los siguientes atributos:

habitacion_numero int llave primaria,precio_por_noche decimal NOT NULL,piso int NOT NULL ,max_personas int

Cree la tabla huéspedes con los siguientes atributos :

id int llave primaria,nombres varchar (45)NOT NULL,apellidos varchar (45) NOT NULL,telefono char (10),correo varchar (45),direccon varchar (45),ciudad varchar (45),pais varchar (45)

Cree la tabla de unión reservas con los atributos:

inicio_fecha,fin_fecha y la información de las habitaciones y los huéspedes :habitacion int not null llave foránea ,huesped_id int not null llave foránea.

Crear la llave compuesta usando la habitación y el id de huésped.

3.-Relación muchos a muchos entre municipio y proyecto

Cree la tabla ciudad con los siguientes atributos:

id int llave primaria,nombre varchar (45) NOT NULL

Cree la tabla municpio

id int llave primaria,nombre varchar (45) ,ciudad_id int llave foranea.

Cree la tabla proyecto

id int llave primaria,proyecto varchar (50)NOT NULL,monto money NOT NULL, fecha_inicio date,fecha_entrega date

Cree la tabla de unión proyecto_municpio que almacene la información del proyecto y del municipio

municipio_id int not null llave foránea , (proyecto_id int not null)llave foránea

Crear la llave compuesta usando el id del proyecto y el id del municpio

Querys Relación muchos a muchos

A continuación, se les presentará instrucciones para realizar Querys utilizando los inserts realizados anteriormente(Si desea puedes hacer más inserts para poder entender más ampliamente la relación muchos a muchos)

Instrucciones para las consultas SQL:

Relación muchos a muchos entre usuarios y grupo:

Consulta:

- Debes seleccionar los nombres de los usuarios y los nombres de los grupos.

- Utiliza las tablas "usuarios", "grupo" y "usuario_grupo".

- Realiza una unión entre las tablas "usuarios" y "usuario_grupo" utilizando la condición "usuarios.id = usuario_grupo.us_id".

- Realiza una unión entre las tablas "grupo" y "usuario_grupo" utilizando la condición "grupo.id = usuario_grupo.gr_id".

Subconsulta:

- Debes seleccionar el nombre de los usuarios.

- Utiliza la tabla "usuarios".

- Utiliza una subconsulta para obtener los "us_id" de la tabla "usuario_grupo" donde "gr_id" sea igual a 1.

- Utiliza la condición "id IN (SELECT ***** FROM ****** WHERE ******* = 1)".

Función de agregación:

- Debes seleccionar el nombre del grupo y contar la cantidad de "us_id" en la tabla "usuario_grupo".

- Utiliza las tablas "grupo" y "usuario_grupo".

- Realiza una unión entre las tablas "grupo" y "usuario_grupo" utilizando la condición "grupo.id = usuario_grupo.gr_id".

- Utiliza la función de agregación "COUNT(usuario_grupo.us_id)" para contar la cantidad de usuarios en cada grupo.

- Utiliza la cláusula "GROUP BY grupo.nombre" para agrupar los resultados por el nombre del grupo.

Consulta 2 :

- Debes seleccionar los nombres de los usuarios y los nombres de los grupos.

- Utiliza las tablas "usuarios", "grupo" y "usuario_grupo".

- Realiza una unión entre las tablas "usuarios" y "usuario_grupo" utilizando la condición "usuarios.id = usuario_grupo.us_id".

- Realiza una unión entre las tablas "grupo" y "usuario_grupo" utilizando la condición "grupo.id = usuario_grupo.gr_id".

- Utiliza la condición "grupo.nombre LIKE '%intensivo%'" para filtrar los grupos que contengan la palabra "intensivo" en su nombre.

Subconsulta 2 :

- Debes seleccionar el nombre de los usuarios.

- Utiliza la tabla "usuarios".

- Utiliza una subconsulta para obtener los "us_id" de la tabla "usuario_grupo" donde "gr_id" sea igual a 2.

- Utiliza la condición "id IN (SELECT **** FROM ********* WHERE ****** = 2)".

Función de agregación 2 :

- Debes seleccionar el nombre del grupo y obtener el máximo valor de "us_id" en la tabla "usuario_grupo".

- Utiliza las tablas "grupo" y "usuario_grupo".

- Realiza una unión entre las tablas "grupo" y "usuario_grupo" utilizando la condición "grupo.id = usuario_grupo.gr_id".

- Utiliza la función de agregación "MAX(usuario_grupo.us_id)" para obtener el máximo valor de usuarios en cada grupo y también realizalo con la función de agregación MIN.

- Utiliza la cláusula "GROUP BY grupo.nombre" para agrupar los resultados por el nombre del grupo.

Consulta 3 :

- Debes seleccionar los nombres de los usuarios y las fechas de creación de los grupos.

- Utiliza las tablas "usuarios", "grupo" y "usuario_grupo".

- Realiza una unión entre las tablas "usuarios" y "usuario_grupo" utilizando la condición "usuarios.id = usuario_grupo.us_id".

- Realiza una unión entre las tablas "grupo" y "usuario_grupo" utilizando la condición "grupo.id = usuario_grupo.gr_id".

- Utiliza la condición "grupo.fecha_creacion entre '2020-03-08' y '2022-03-08'" para filtrar los grupos creados entre esas fechas.

Subconsultas 3 :

- Debes seleccionar el nombre de los usuarios.

- Utiliza la tabla "usuarios".

- Utiliza una subconsulta para obtener los "us_id" de la tabla "usuario_grupo" donde "gr_id" sea igual a 3.

- Utiliza la condición "id IN (SELECT ****** FROM ******** WHERE ******** = 3)".

Función de agregación 3:

- Debes seleccionar la descripción del grupo y contar la cantidad de "us_id" en la tabla "usuario_grupo".

- Utiliza las tablas "grupo" y "usuario_grupo".

- Realiza una unión entre las tablas "grupo" y "usuario_grupo" utilizando la condición "grupo.id = usuario_grupo.gr_id".

- Utiliza la función de agregación "COUNT(usuario_grupo.us_id)" para contar la cantidad de usuarios en cada grupo.

- Utiliza la condición "grupo.descripcion LIKE '%matutino%'" para filtrar los grupos que contengan la palabra "matutino" en su descripción.

- Utiliza la cláusula "GROUP BY grupo.descripcion" para agrupar los resultados por la descripción del grupo.

Relación muchos a muchos entre habitaciones y huéspedes:

Consulta:

- Debes seleccionar el número de habitación, los nombres y apellidos de los huéspedes.

- Utiliza las tablas "habitaciones", "huespedes" y "reservas".

- Realiza una unión entre las tablas "habitaciones" y "reservas" utilizando la condición "habitaciones.habitacion_numero = reservas.habitacion".

- Realiza una unión entre las tablas "huespedes" y "reservas" utilizando la condición "huespedes.id = reservas.huesped_id".

Subconsulta:

- Debes seleccionar los nombres y apellidos de los huéspedes.

- Utiliza la tabla "huespedes".

- Utiliza una subconsulta para obtener los "huesped_id" de la tabla "reservas" donde "habitacion" sea igual a 2.

- Utiliza la condición "id IN (SELECT huesped_id FROM reservas WHERE habitacion = 2)".

Función de agregación:

- Debes seleccionar el número de habitación y contar la cantidad de "huesped_id" en la tabla "reservas".

- Utiliza las tablas "habitaciones" y "reservas".

- Realiza una unión entre las tablas "habitaciones" y "reservas" utilizando la condición "habitaciones.habitacion_numero = reservas.habitacion".

- Utiliza la función de agregación "COUNT(reservas.huesped_id)" para contar la cantidad de huéspedes en cada habitación.

- Utiliza la cláusula "GROUP BY habitaciones.habitacion_numero" para agrupar los resultados por el número de habitación.

Consulta 2:

- Debes seleccionar el número de habitación, el piso, los nombres y apellidos de los huéspedes.

- Utiliza las tablas "habitaciones", "huespedes" y "reservas".

- Utiliza la condición "habitaciones.piso = 4" para filtrar las habitaciones que estén en el piso 4.

- Realiza una unión entre las tablas "habitaciones" y "reservas" utilizando la condición "habitaciones.habitacion_numero = reservas.habitacion".

- Realiza una unión entre las tablas "huespedes" y "reservas" utilizando la condición "huespedes.id = reservas.huesped_id".

Subconsulta 2:

- Debes seleccionar los nombres y apellidos de los huéspedes.

- Utiliza la tabla "huespedes".

- Utiliza una subconsulta para obtener los "huesped_id" de la tabla "reservas" donde "habitacion" sea igual a 3.

- Utiliza la condición "id IN (SELECT huesped_id FROM reservas WHERE habitacion = 3)".

Función de agregación 2:

- Debes seleccionar el número de habitación y obtener el promedio de "huesped_id" en la tabla "reservas".

- Utiliza las tablas "habitaciones" y "reservas".

- Realiza una unión entre las tablas "habitaciones" y "reservas" utilizando la condición "habitaciones.habitacion_numero = reservas.habitacion".

- Utiliza la función de agregación "AVG(reservas.huesped_id)" para obtener el promedio de huéspedes en cada habitación.

- Utiliza la cláusula "GROUP BY habitaciones.habitacion_numero" para agrupar los resultados por el número de habitación.

Consulta 3:

- Debes seleccionar el número de habitación, los nombres y apellidos de los huéspedes.

- Utiliza las tablas "habitaciones", "huespedes" y "reservas".

- Realiza una unión entre las tablas "habitaciones" y "reservas" utilizando la condición "habitaciones.habitacion_numero = reservas.habitacion".

- Realiza una unión entre las tablas "huespedes" y "reservas" utilizando la condición "huespedes.id = reservas.huesped_id".

Subconsulta 3:

- Debes seleccionar los nombres y apellidos de los huéspedes.

- Utiliza la tabla "huespedes".

- Utiliza una subconsulta para obtener los "huesped_id" de la tabla "reservas" donde "habitacion" sea igual a 4.

- Utiliza la condición "id IN (SELECT huesped_id FROM reservas WHERE habitacion = 4)".

Función de agregación 3:

- Debes seleccionar el número de habitación y obtener la suma de "precio_por_noche" en la tabla "habitaciones".

- Utiliza las tablas "habitaciones" y "reservas".

- Realiza una unión entre las tablas "habitaciones" y "reservas" utilizando la condición "habitaciones.habitacion_numero = reservas.habitacion".

- Utiliza la función de agregación "SUM(habitaciones.precio_por_noche)" para obtener el total recaudado por habitación.

- Utiliza la cláusula "GROUP BY habitaciones.habitacion_numero" para agrupar los resultados por el número de habitación.

Relación muchos a muchos entre municipio y proyectos:

Consulta:

- Debes seleccionar el nombre del municipio y el nombre del proyecto.

- Utiliza las tablas "municipio", "proyecto" y "proyecto_municipio".

- Realiza una unión entre las tablas "municipio" y "proyecto_municipio" utilizando la condición "municipio.id = proyecto_municipio.municipio_id".

- Realiza una unión entre las tablas "proyecto" y "proyecto_municipio" utilizando la condición "proyecto.id = proyecto_municipio.proyecto_id".

Subconsulta:

- Debes seleccionar el nombre del proyecto.

- Utiliza la tabla "proyecto".

- Utiliza una subconsulta para obtener los "proyecto_id" de la tabla "proyecto_municipio" donde "municipio_id" sea igual a 1.

- Utiliza la condición "id IN (SELECT ******* FROM ****** WHERE ***** = 1)".

Función de agregación:

- Debes seleccionar el nombre del municipio y contar la cantidad de "proyecto_id" en la tabla "proyecto_municipio".

- Utiliza las tablas "municipio" y "proyecto_municipio".

- Realiza una unión entre las tablas "municipio" y "proyecto_municipio" utilizando la condición "municipio.id = proyecto_municipio.municipio_id".

- Utiliza la función de agregación "COUNT(proyecto_municipio.proyecto_id)" para contar la cantidad de proyectos en cada municipio.

- Utiliza la cláusula "GROUP BY municipio.nombre" para agrupar los resultados por el nombre del municipio.

Consulta 2:

- Debes seleccionar el nombre del municipio y el nombre del proyecto.

- Utiliza las tablas "municipio", "proyecto" y "proyecto_municipio".

- Realiza una unión entre las tablas "municipio" y "proyecto_municipio" utilizando la condición "municipio.id = proyecto_municipio.municipio_id".

-Realiza una unión entre las tablas "proyecto" y "proyecto_municipio" utilizando la condición "proyecto.id = proyecto_municipio.proyecto_id".

-Utiliza la condición "municipio.nombre LIKE '%GAD%'" para filtrar los municipios que contengan la palabra "GAD" en su nombre.

Función de agregación 2:

-Debes seleccionar el nombre del municipio y obtener el mínimo valor de "proyecto_id" en la tabla "proyecto_municipio".

-Utiliza las tablas "municipio" y "proyecto_municipio".

-Realiza una unión entre las tablas "municipio" y "proyecto_municipio" utilizando la condición "municipio.id = proyecto_municipio.municipio_id".

-Utiliza la función de agregación "MIN(proyecto_municipio.proyecto_id)" para obtener el valor mínimo de proyectos en cada municipio.

-Utiliza la cláusula "GROUP BY municipio.nombre" para agrupar los resultados por el nombre del municipio.

Consulta 3 :

Selecciona el nombre del municipio y el nombre de la ciudad. Se realiza una unión entre las tablas "municipio" y "ciudad" utilizando la condición adecuada.

Subconsulta 3:

Selecciona el proyecto de la tabla "proyecto" donde el id está presente en la subconsulta que obtiene los proyecto_id de la tabla "proyecto_municipio" donde el municipio_id es igual a 3.

Función de agregación 3:

Selecciona el nombre del municipio y se obtiene el valor máximo de proyecto_id en la tabla "proyecto_municipio". Se realiza una unión entre las tablas "municipio" y "proyecto_municipio" y se agrupa por el nombre del municipio.