**ESCUELA POLITÉCNICA NACIONAL**

ESCUELA DE FORMACIÓN DE TECNÓLOGOS

**BASE DE DATOS**

|  |  |
| --- | --- |
|  | |
| PROFESOR: | Ing. Yadira Franco R |
| PERÍODO ACADÉMICO: | 2025 A |
|  | |

**TAREA**

|  |
| --- |
|  |
| TÍTULO:  **TIPOS DE JOIN** |
|  |
| Qué es una base de datos y cuál es su importancia? | Atlantic Tech |
|  |
| Estudiante  xxxxxxxxxxxxxx |
|  |

**2025-A**

TRABAJO GRUPAL MAXIMO 2 INTEGRANTES

**TIPOS DE JOIN**

El JOIN se usa para combinar filas de dos o más tablas basadas en una condición de relación entre ellas, generalmente usando claves foráneas.

**2. Tipos de JOIN:**

1. **INNER JOIN**: Este tipo de JOIN retorna solo las filas que tienen coincidencias en ambas tablas.
2. **LEFT JOIN (o LEFT OUTER JOIN)**: Retorna todas las filas de la tabla de la izquierda y las filas coincidentes de la tabla de la derecha. Si no hay coincidencias, devuelve NULL para la tabla de la derecha.
3. **RIGHT JOIN (o RIGHT OUTER JOIN)**: Similar al LEFT JOIN, pero devuelve todas las filas de la tabla de la derecha.
4. **FULL JOIN (o FULL OUTER JOIN)**: Retorna todas las filas cuando hay una coincidencia en una de las tablas, o ambas.

**ADICIONAL:**

**Agrega registros de prueba** a las tablas correspondientes para garantizar que los resultados de cada tipo de JOIN sean representativos y completos. Los registros deben ser consistentes con los datos que se requieren para cada consulta y deben permitir visualizar claramente las diferencias en los resultados de cada JOIN.

**INSTRUCCIONES**

* 1. **Sube un archivo al git hub** que contenga todos los elementos solicitados.
  2. **Incluye los scripts SQL** correspondientes a cada tipo de JOIN (INNER, LEFT, RIGHT, FULL OUTER JOIN) con **comentarios detallados** explicando cada consulta.
  3. **Adjunta las capturas de pantalla** de los resultados obtenidos para cada consulta, mostrando los datos de la ejecución de las consultas.

|  |  |
| --- | --- |
| CLÍNICA | |
| **INNER JOIN** | **LEFT JOIN** |
| 1. Relacionar **pacientes** con **consultas** cuando ambos están registrados. | 1. Mostrar todos los **pacientes**, incluyendo aquellos que aún no tienen **consultas** agendadas. |
|  |  |
| 1. Relacionar **consultas** con **doctores** asignados. | 1. Mostrar todas las **consultas**, incluyendo las que no tienen **doctor** asignado. |
|  |  |
| 1. Relacionar **medicamentos** recetados con los **pacientes** que los reciben. | 1. Mostrar todos los **pacientes**, incluyendo aquellos que no reciben **medicamentos**. |
|  |  |
| **RIGHT JOIN** | **FULL OUTER JOIN** |
| 1. Mostrar todas las **consultas**, incluso aquellas que no están asignadas a un **paciente**. | 1. Mostrar todos los **pacientes** y **consultas**, incluidas las que no están relacionadas entre sí. |
|  |  |
| 1. Mostrar todos los **doctores**, incluso aquellos que no tienen **consultas** asignadas. | 1. Mostrar todas las **consultas** y **doctores**, incluso las no relacionadas. |
|  |  |
| 1. Mostrar todos los **medicamentos**, incluso los que no han sido recetados. | 1. Mostrar todos los **pacientes** y **medicamentos**, incluso los no relacionados entre sí. |
|  |  |

|  |
| --- |
| SCRIPT |
| create database clinicas; -- Se crea la base de datos clinica  use clinicas;  -- Se crea la tabla pacientes  create table pacientes(  pacientesID int primary key not null, -- Se define la clave primaria y demas campos de la tabla  nombre varchar(50) not null,  apellido varchar(50) not null,  edad int not null,  ciudad varchar(50) not null,  direccion varchar(100) not null  );  -- Se crea la tabla doctores  create table doctores(  doctorID int primary key not null, -- Se define la clave primaria y demas campos de la tabla  nombre varchar(50) not null,  apellido varchar(50) not null,  especialidad varchar(50) not null  );  -- Se crea la tabla medicamentos  create table medicamentos(  medicamentoID int primary key not null, -- Se define la clave primaria y demas campos de la tabla  nombre varchar(50) not null,  descripcion varchar(100) not null  );  -- Se crea la tabla consultas  create table consultas(  consultaID int primary key not null, -- Se define la clave primaria y demas campos de la tabla  pacienteID int, -- Campo que almacenará el ID del paciente  foreign key (pacienteID) references pacientes(pacientesID), -- Se establece una relacion con la tabla pacientes  doctorID int,  foreign key (doctorID) references doctores(doctorID),  fecha date not null  );  -- Se crea la tabla recetas  create table recetas(  recetaID int primary key not null, -- Se define la clave primaria y demas campos de la tabla  pacienteID int, -- Campo que almacenará el ID del paciente  foreign key (pacienteID) references pacientes(pacientesID), -- Se establece una relacion con la tabla pacientes  medicamentoID int, -- Campo que almacenará el ID del medicamento  foreign key (medicamentoID) references medicamentos(medicamentoID), -- Se establece relación con la tabla medicamentos  doctorID int, -- Campo que almacenará el ID del doctor  foreign key (doctorID) references doctores(doctorID), -- Se establece una relacion con la tabla doctores  fecha date not null  );  -- Se insertan datos en la tabla pacientes  insert into pacientes(pacientesID, nombre, apellido, edad, ciudad, direccion) values  (101, 'Laura', 'García', 28, 'Madrid', 'Calle Alcalá 123'),  (102, 'Carlos', 'Martínez', 35, 'Barcelona', 'Avenida Diagonal 456'),  (103, 'Ana', 'López', 42, 'Valencia', 'Calle Colón 789'),  (104, 'Javier', 'Sánchez', 50, 'Sevilla', 'Plaza Nueva 321'),  (105, 'Marta', 'Fernández', 22, 'Málaga', 'Calle Larios 654'),  (106, 'David', 'Pérez', 65, 'Zaragoza', 'Paseo Independencia 101'),  (107, 'Sofía', 'Gómez', 38, 'Bilbao', 'Gran Vía 202'),  (108, 'Daniel', 'Ruiz', 29, 'Murcia', 'Plaza Circular 303'),  (109, 'Elena', 'Hernández', 45, 'Palma', 'Paseo Marítimo 404'),  (110, 'Pablo', 'Díaz', 31, 'Alicante', 'Explanada España 505'),  (111, 'Alejandro', 'Martín', 29, 'Alicante', 'Avenida de Denia 210'),  (112, 'Sara', 'González', 33, 'Alicante', 'Calle San Fernando 75'),  (113, 'Jorge', 'Soler', 30, 'Alicante', 'Paseo de la Explanada 12');  -- Se insertan datos en la tabla doctores  insert into doctores(doctorID, nombre, apellido, especialidad) values  (201, 'Juan', 'Rodríguez', 'Cardiología'),  (202, 'María', 'Gutiérrez', 'Pediatría'),  (203, 'Luis', 'Moreno', 'Traumatología'),  (204, 'Carmen', 'Jiménez', 'Dermatología'),  (205, 'Pedro', 'Navarro', 'Neurología'),  (206, 'Teresa', 'Romero', 'Ginecología'),  (207, 'Alberto', 'Torres', 'Oftalmología'),  (208, 'Isabel', 'Serrano', 'Endocrinología'),  (209, 'Francisco', 'Molina', 'Psiquiatría'),  (210, 'Rosa', 'Ortega', 'Medicina General'),  (211, 'Gabriela', 'Méndez', 'Oncología'),  (212, 'Ricardo', 'Fernández', 'Cardiología Pediátrica'),  (213, 'Patricia', 'Silva', 'Medicina Interna');  -- Se insertan datos en la tabla medicamentos  insert into medicamentos(medicamentoID, nombre, descripcion) values  (301, 'Ibuprofeno', 'Antiinflamatorio no esteroideo'),  (302, 'Paracetamol', 'Analgésico y antipirético'),  (303, 'Amoxicilina', 'Antibiótico betalactámico'),  (304, 'Omeprazol', 'Inhibidor de la bomba de protones'),  (305, 'Loratadina', 'Antihistamínico para alergias'),  (306, 'Simvastatina', 'Para reducir el colesterol'),  (307, 'Metformina', 'Antidiabético oral'),  (308, 'Salbutamol', 'Broncodilatador para el asma'),  (309, 'Diazepam', 'Ansiolítico y relajante muscular'),  (310, 'Atorvastatina', 'Hipolipemiante'),  (311, 'Rosuvastatina', 'Hipolipemiante de última generación'),  (312, 'Metoprolol', 'Bloqueador beta para hipertensión'),  (313, 'Losartán', 'Antihipertensivo antagonista de receptores de angiotensina II');  -- Se insertan datos en la tabla consultas  insert into consultas(consultaID, pacienteID, doctorID, fecha) values  (401, 101, 201, '2025-05-10'),  (402, 102, 202, '2025-05-11'),  (403, 103, 203, '2025-05-12'),  (404, 104, 204, '2025-05-13'),  (405, 105, 205, '2025-05-14'),  (406, 106, 206, '2025-05-15'),  (407, 107, 207, '2025-05-16'),  (408, 108, 208, '2025-05-17'),  (409, 109, 209, '2025-05-18'),  (410, 110, 210, '2025-05-19'),  (411, null, null, '2025-05-12'),  (412, null, null, '2025-05-15'),  (413, null, null, '2025-05-19');  -- Se insertan datos en la tabla recetas  insert into recetas(recetaID, pacienteID, medicamentoID, doctorID, fecha) values  (501, 101, 301, 201, '2025-05-10'),  (502, 101, 302, 201, '2025-05-10'),  (503, 102, 303, 202, '2025-05-11'),  (504, 103, 304, 203, '2025-05-12'),  (505, 104, 305, 204, '2025-05-13'),  (506, 105, 306, 205, '2025-05-14'),  (507, 106, 307, 206, '2025-05-15'),  (508, 107, 308, 207, '2025-05-16'),  (509, 108, 309, 208, '2025-05-17'),  (510, 109, 310, 209, '2025-05-18'),  (511, 110, 301, 210, '2025-05-19'),  (512, 110, 302, 210, '2025-05-20'),  (513, null, null, 211, '2025-05-21'),  (514, null, null, 212, '2025-05-22'),  (515, null, null, 210, '2025-05-23');  -- CONSULTAS  -- INNER JOIN  -- Se usa inner join para combinar los registros de dos o mas tablas(con una condicion de relacion entre ellas), en este caso la tabla pacientes y la tabla consultas  -- 1. Relacionar pacientes con consultas cuando ambos están registrados.  -- En este caso la relacion es encontrar pacientes con consultas registradas  select p.pacientesID, p.nombre as nombre\_paciemte, p.apellido as apellido\_paciente,  c.consultaID, c.fecha as fecha\_consulta from pacientes p  inner join consultas c on p.pacientesID = c.pacienteID;  -- 2. Relacionar consultas con doctores asignados.  -- La relacion que comparten las tablas consultas y doctores es:  -- "los doctores asigandos a cada consulta"  select c.consultaID, c.fecha as fecha\_consulta,  d.doctorID, d.nombre as nombre\_doctor,  d.apellido as apellido\_doctor, d.especialidad as especialidad\_doctor  from doctores d inner join consultas c on d.doctorID = c.doctorID;  -- 3. Relacionar medicamentos recetados con los pacientes que los reciben.  -- Se relacionan cuatro tablas (medicamentos, recetas, pacientes y doctores)  select r.recetaID, r.fecha as fecha\_receta, p.pacientesID, p.nombre as nombre\_paciente,  p.apellido as apellido\_paciente, m.medicamentoID,  m.nombre as nombre\_medicamento, d.nombre as nombre\_doctor,  d.apellido as apellido\_doctor from recetas r  -- Se relaciona una receta con el paciente al que le corresponde  inner join pacientes p on r.pacienteID = p.pacientesID  -- Relacion entre la receta y el medicamento que se ha prescrito  inner join medicamentos m on r.medicamentoID = m.medicamentoID  -- La receta se conecta con el doctor que la ha emitido  inner join doctores d on r.doctorID = d.doctorID;  -- LEFT JOIN  -- Sirve para combinar los datos de las dos tablas manteniendo los registros de la tabla izquierda  -- incluso cuando no hay coincidencias en la tabla derecha  -- 1. Mostrar todos los pacientes, incluyendo aquellos que aún no tienen consultas agendadas.  select p.pacientesID, p.nombre as nombre\_paciemte, p.apellido as apellido\_paciente,  c.consultaID, c.fecha as fecha\_consulta  from pacientes p -- Esta es la tabla izquierda (primera tabla mencionada) de la cual todos los registros se mantienen  left join consultas c -- Esta tabla derecha solo contiene coincidencias  on p.pacientesID = c.pacienteID;  -- 2. Mostrar todas las consultas, incluyendo las que no tienen doctor asignado.  -- Si una consulta no tiene un doctor asignado se mostraran las columnas con los datos de los doctores llena valores nulos  select c.consultaID, c.fecha as fecha\_consulta,  d.nombre as nombre\_doctor, d.apellido as apellido\_doctor, d.especialidad as especialidad\_doctor  from consultas c -- Tabla principal con todas las consultas  left join doctores d -- Tabla secundaria con las coincidencias  on c.doctorID =d.doctorID;  -- 3. Mostrar todos los pacientes, incluyendo aquellos que no reciben medicamentos.  -- Si un paciente no tiene una receta entonces el ID de la receta y el medicamento se mostraran con valores nulos  select p.pacientesID, p.nombre as nombre\_paciente, p.apellido as apellido\_paciente,  r.recetaID, m.nombre as nombre\_medicamento  from pacientes p -- Tabla principal de la cual se mostraran todos los registros  -- Se conecta a los pacientes con las recetas, se mantienen todos los pacientes aunque no tengan recetas  left join recetas r on p.pacientesID = r.pacienteID  -- Se conecta recetas con medicamentos, se aplica solo a los pacientes que tienen recetas  left join medicamentos m on r.medicamentoID = m.medicamentoID;  -- RIGHT JOIN  -- Es una operacion de union entre tablas que mantiene todos los registros de la tabla principal (segunda tabla mencionada)  -- y solo muestra los registros de coincidencias de la tabla secundaria (primera tabla mencionada)  -- 1. Mostrar todas las consultas, incluso aquellas que no están asignadas a un paciente.  -- Se muestran todas las consultas medicas y las que no tienen un paciente asignado se muestran con valores nulos  select c.consultaID, c.fecha as fecha\_consulta,  p.pacientesID, p.nombre as nombre\_paciente, p.apellido as apellido\_paciente  from pacientes p -- Tabla secundaria que contiene los datos de los pacientes  right join consultas c -- Tabla principal con todos los datos de las consultas  on p.pacientesID = c.pacienteID;  -- 2. Mostrar todos los doctores, incluso aquellos que no tienen consultas asignadas.  -- Se mantienen todos los registros de la tabla doctores y se relacionan con la tabla consultas  -- Si no hay consultas asignadas a un doctor se muestran valores nulos  select d.doctorID, d.nombre as nombre\_doctor, d.apellido as apellido\_doctor,  d.especialidad as especialidad\_doctor,  c.consultaID, c.fecha as fecha\_consulta  from consultas c -- Tabla secundaria  right join doctores d -- Tabla principal  on d.doctorID = c.doctorID;  -- 3. Mostrar todos los medicamentos, incluso los que no han sido recetados.  -- Se mantienen todos los registros de la tabla medicamentos y se relacionan con la tabla recetas  -- Si los medicamentos no tienen una receta se muestran valores nulos  select m.medicamentoID, m.nombre as nombre\_medicamento,  r.recetaID, r.fecha as fecha\_receta  from recetas r -- Tabla secundaria  right join medicamentos m -- Tabla principal  on m.medicamentoID = r.medicamentoID;  -- FULL OUTER JOIN  -- Es una operacion de union que preserva todos los registros de ambas tablas  -- MySQL no soporta directamente FULL OUTER JOIN, por lo que debemos simularlo con una combinación de LEFT JOIN y RIGHT JOIN.  -- 1. Mostrar todos los pacientes y consultas, incluidas las que no están relacionadas entre sí.  -- Todos los pacientes con sus consultas si existen  select p.pacientesID, p.nombre as nombre\_paciente, p.apellido as apellido\_paciente,  c.consultaID, c.fecha as fecha\_consulta, d.nombre as doctor  from pacientes p -- Tabla principal, todos los registros se mantienen  left join consultas c on p.pacientesID = c.pacienteID -- Une con consultas si existen  left join doctores d on c.doctorID = d.doctorID -- Une doctor si hay consultas  union -- Combina resultados eliminando duplicados  -- Consultas sin pacientes  select p.pacientesID, p.nombre as nombre\_paciente, p.apellido as apellido\_paciente,  c.consultaID, c.fecha as fecha\_consulta, d.nombre as nombre\_doctor  from pacientes p  -- Prioriza todas las consultas (incluso sin pacientes)  right join consultas c on p.pacientesID = c.pacienteID  -- Añade información del doctor para cada consulta  left join doctores d on c.doctorID = d.doctorID  where p.pacientesID is null;  -- 2. Mostrar todas las consultas y doctores, incluso las no relacionadas.  -- Muestra todos los doctores y los relacionan con la tabla consultas  -- Si no tienen consultas los valores se llenaran con null  select d.doctorID, d.nombre as nombre\_doctor, d.apellido as apellido\_doctor,  d.especialidad, c.consultaID, c.fecha as fecha\_consulta, p.nombre as nombre\_paciente  from doctores d  left join consultas c on d.doctorID = c.doctorID  left join pacientes p on c.pacienteID = p.pacientesID  union  -- Muestra consultas que no tienen un doctor asignado  select null as doctorID, null as nombre\_doctor,  null as apellido\_doctor, null as especialidad,  c.consultaID, c.fecha as fecha\_consulta,  p.nombre as nombre\_paciente  from consultas c  left join pacientes p on c.pacienteID = p.pacientesID  where c.doctorID is null; -- Se filtran los registros de consultas sin doctor  -- 3. Mostrar todos los pacientes y medicamentos, incluso los no relacionados entre sí.  select p.pacientesID, p.nombre as nombre\_pacientes, p.apellido as apellido\_paciente,  m.medicamentoID, m.nombre as medicamento  from pacientes p -- Tabla principal  -- Conecta a los pacientes con las recetas  left join recetas r on p.pacientesID = r.pacienteID  -- Conecta recetas con medicamentos  left join medicamentos m on r.medicamentoID = m.medicamentoID  union -- Combina resultados eliminando duplicados  select null as pacientesID, null as nombre\_paciente,  null as apellido\_paciente, m.medicamentoID, m.nombre as medicamento  from medicamentos m -- Todos los medicamentos  -- filtra solo medicamentos sin registros en recetas  where not exists (select 1 from recetas r where r.medicamentoID = m.medicamentoID) |

|  |  |  |
| --- | --- | --- |
| REDES SOCIALES | | |
| **INNER JOIN** | **LEFT JOIN** | |
| 1. Relacionar **usuarios** con sus **fotos** publicadas. | 1. Mostrar todos los **usuarios**, incluyendo aquellos que no han subido ninguna **foto**. | |
|  |  | |
| 1. Relacionar **fotos** con los **comentarios** recibidos. | 1. Mostrar todas las **fotos**, incluyendo las que no tienen **comentarios**. | |
|  |  | |
| 1. Relacionar **usuarios** con sus **comentarios** realizados | 1. Mostrar todos los **usuarios**, incluyendo aquellos que no han realizado ningún **comentario**. | |
|  |  | |
| **RIGHT JOIN** | **FULL OUTER JOIN** | |
| 1. Mostrar todas las **fotos**, incluso las que no están asociadas a ningún **usuario**. | 1. Mostrar todos los **usuarios** y **fotos**, incluidas las que no tienen relación entre sí. | |
|  |  | |
| 1. Mostrar todos los **comentarios**, incluso los que no están asociados a ninguna **foto**. | 1. Mostrar todas las **fotos** y **comentarios**, incluidos los no relacionados entre sí. | |
|  |  | |
| 1. Mostrar todos los **comentarios**, incluso los que no están asociados a ningún **usuario**. | 1. Mostrar todos los **usuarios** y **comentarios**, incluidos los no relacionados entre sí. | |
|  |  | |
| SCRIPT | |
| -- Crear la base de datos  create database red\_social;  use red\_social;  -- Se crea la tabla usuarios  create table usuarios (  usuarioID int primary key not null, -- Se define la clave primaria y demas campos de la tabla  nombre varchar(50) not null,  apellido varchar(50) not null,  edad int not null,  ciudad varchar(50) not null,  seguidores int not null,  fotos\_publicadas int not null  );  -- Se crea la tabla fotos  create table fotos (  fotoID int primary key not null, -- Se define la clave primaria y demas campos de la tabla  usuarioID int, -- Campo que almacenara el ID del usuario  foreign key (usuarioID) references usuarios(usuarioID), -- Se establece una relacion con la tabla usuarios  descripcion varchar(255) not null,  fecha date not null,  me\_gusta int not null  );  -- Se crea la tabla comentarios  create table comentarios (  comentarioID int primary key not null, -- Se define la clave primaria y demas campos de la tabla  fotoID int, -- Campo que almacenara el ID del de las fotos  foreign key (fotoID) references fotos(fotoID), -- Se establece una relacion con la tabla fotos  usuarioID int, -- Campo que almacenara el ID del usuario  foreign key (usuarioID) references usuarios(usuarioID), -- Se establece una relacion con la tabla usuarios  comentario text not null,  fecha date not null  );  insert into usuarios (usuarioID, nombre, apellido, edad, ciudad, seguidores, fotos\_publicadas) values  (1, 'María', 'Gómez', 25, 'Madrid', 350, 12),  (2, 'Carlos', 'López', 30, 'Barcelona', 420, 8),  (3, 'Ana', 'Martínez', 22, 'Valencia', 180, 5),  (4, 'Javier', 'Sánchez', 28, 'Sevilla', 600, 15),  (5, 'Lucía', 'Fernández', 35, 'Bilbao', 290, 7),  (6, 'Pedro', 'García', 40, 'Málaga', 150, 3),  (7, 'Sofía', 'Ruiz', 27, 'Zaragoza', 380, 9),  (8, 'David', 'Hernández', 31, 'Murcia', 210, 6),  (9, 'Elena', 'Jiménez', 29, 'Palma', 470, 11),  (10, 'Pablo', 'Díaz', 33, 'Alicante', 520, 14);  insert into fotos (fotoID, usuarioID, descripcion, fecha, me\_gusta) values  (101, 1, 'Atardecer en la playa', '2023-05-15', 45),  (102, 2, 'Mi nuevo perro', '2023-06-20', 78),  (103, 3, 'Comida casera', '2023-04-10', 32),  (104, 1, 'Viaje a la montaña', '2023-07-05', 91),  (105, 4, 'Concierto de verano', '2023-08-12', 120),  (106, 5, 'Día de spa', '2023-03-18', 65),  (107, 6, 'Nueva bicicleta', '2023-02-22', 43),  (108, 1, 'Libro favorito', '2023-01-30', 28),  (109, 2, 'Jardín botánico', '2023-09-14', 87),  (110, null, 'Cena con amigos', '2023-10-05', 95),  (111, null, 'Deporte al aire libre', '2023-11-11', 110),  (112, null, 'Museo de arte', '2023-12-03', 52);  insert into comentarios (comentarioID, fotoID, usuarioID, comentario, fecha) values  (1001, 101, 2, '¡Qué bonito atardecer!', '2023-05-16'),  (1002, 101, 3, 'Me encantaría estar ahí', '2023-05-17'),  (1003, 104, 1, 'Las vistas son impresionantes', '2023-07-06'),  (1004, 105, 3, '¡Qué envidia!', '2023-08-13'),  (1005, 102, 4, 'Tu perro es adorable', '2023-06-21'),  (1006, 106, 1, 'Se ve relajante', '2023-03-19'),  (1007, 107, 3, 'Buena elección de modelo', '2023-02-23'),  (1008, 108, 5, 'Lo leí el mes pasado, es genial', '2023-01-31'),  (1009, 109, 6, 'Hermosas flores', '2023-09-15'),  (1010, 110, 6, 'Se ve delicioso todo', '2023-10-06'),  (1011, null, null, 'Qué energía', '2023-11-12'),  (1012, null, null, 'Fui la semana pasada', '2023-12-04'),  (1013, null, null, 'Espectacular foto astronómica', '2023-04-26');  -- CONSULTAS  -- INNER JOIN  -- 1. Relacionar usuarios con sus fotos publicadas.  select u.usuarioID, u.nombre, u.apellido,  f.fotoID, f.descripcion, f.fecha, f.me\_gusta  from usuarios u  -- Se usa inner join para mostrar solo coincidencias exactas entre ambas tablas  inner join fotos f on u.usuarioID = f.usuarioID;  -- 2. Relacionar fotos con los comentarios recibidos.  select f.fotoID, f.descripcion as Descripcion\_foto, f.fecha as fecha\_foto,  c.comentarioID, c.comentario, c.fecha as fecha\_comentario  from fotos f  -- Solo se mostraran fotos que tienen comentarios y comentarios que tienen fotos válidas  inner join comentarios c on f.fotoID=c.fotoID;  -- Se relaciona las fotos con sus comentarios  -- 3. Relacionar usuarios con sus comentarios realizados.  select u.usuarioID, u.nombre, u.apellido,  c.comentarioID, c.comentario, c.fecha as fecha\_comentario  from usuarios u  -- Solo se muestran los usuarios con comentarios  inner join comentarios c on u.usuarioID = c.usuarioID;  -- Relaciona usuarios con sus comentarios a traves del ID  -- LEFT JOIN  -- 1. Mostrar todos los usuarios, incluyendo aquellos que no han subido ninguna foto.  -- Se muestran todos los registros, incluso los que no se relacionan  select u.usuarioID, u.nombre, u.apellido,  f.fotoID, f.descripcion, f.fecha, f.me\_gusta  from usuarios u -- Se mantienen todos los registros de esta tabla  -- Mantiene todos los usuarios aunque no tengan fotos  left join fotos f -- Esta tabla solo contiene coincidencias  on u.usuarioID = f.usuarioID;  -- 2. Mostrar todas las fotos, incluyendo las que no tienen comentarios.  select f.fotoID, f.descripcion, f.fecha, f.me\_gusta,  c.comentarioID, c.comentario, c.fecha as fecha\_comentario  from fotos f -- Se mantienen todos los registros de esta tabla  -- Mantiene todas las fotos aunque no tengan comentarios  left join comentarios c -- Solo contiene coincidencias  on f.fotoID = c.fotoID;  -- 3. Mostrar todos los usuarios, incluyendo aquellos que no han realizado ningún comentario.  select u.usuarioID, u.nombre, u.apellido,  c.comentarioID, c.comentario, c.fecha as fecha\_comentario  from usuarios u -- Se mantienen todos los registros de esta tabla  -- Mantiene todos los usuarios aunque no tengan comentarios  left join comentarios c -- Solo contiene coincidencias  on u.usuarioID = c.usuarioID; | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Temática** | **INNER JOIN** | **LEFT JOIN** | **RIGHT JOIN** | **FULL OUTER JOIN** |
| **Clínica** | Relacionar **pacientes** con **consultas** cuando ambos están registrados. | Mostrar todos los **pacientes**, incluyendo aquellos que aún no tienen **consultas** agendadas. | Mostrar todas las **consultas**, incluso aquellas que no están asignadas a un **paciente**. | Mostrar todos los **pacientes** y **consultas**, incluidas las que no están relacionadas entre sí. |
| Relacionar **consultas** con **doctores** asignados. | Mostrar todas las **consultas**, incluyendo las que no tienen **doctor** asignado. | Mostrar todos los **doctores**, incluso aquellos que no tienen **consultas** asignadas. | Mostrar todas las **consultas** y **doctores**, incluso las no relacionadas. |
| Relacionar **medicamentos** recetados con los **pacientes** que los reciben. | Mostrar todos los **pacientes**, incluyendo aquellos que no reciben **medicamentos**. | Mostrar todos los **medicamentos**, incluso los que no han sido recetados. | Mostrar todos los **pacientes** y **medicamentos**, incluso los no relacionados entre sí. |
| **Curso** | Relacionar **estudiantes** con **cursos** en los que están inscritos. | Mostrar todos los **estudiantes**, incluso aquellos que no están inscritos en ningún **curso**. | Mostrar todos los **cursos**, incluso aquellos que no tienen **estudiantes** inscritos. | Mostrar todos los **estudiantes** y **cursos**, incluso los no relacionados entre sí. |
| Relacionar **cursos** con sus **instructores**. | Mostrar todos los **cursos**, incluyendo aquellos sin **instructor** asignado. | Mostrar todos los **instructores**, incluso aquellos que no tienen **cursos** a su cargo. | Mostrar todos los **cursos** y **instructores**, incluidos los no relacionados entre sí. |
| Relacionar **estudiantes** con **lecciones completadas**. | Mostrar todos los **estudiantes**, incluso aquellos que no han completado ninguna **lección**. | Mostrar todas las **lecciones**, incluyendo las que no han sido completadas por ningún **estudiante**. | Mostrar todos los **estudiantes** y **lecciones**, incluso las no relacionadas entre sí. |
| **Redes Sociales** | Relacionar **usuarios** con sus **fotos** publicadas. | Mostrar todos los **usuarios**, incluyendo aquellos que no han subido ninguna **foto**. | Mostrar todas las **fotos**, incluso las que no están asociadas a ningún **usuario**. | Mostrar todos los **usuarios** y **fotos**, incluidas las que no tienen relación entre sí. |
| Relacionar **fotos** con los **comentarios** recibidos. | Mostrar todas las **fotos**, incluyendo las que no tienen **comentarios**. | Mostrar todos los **comentarios**, incluso los que no están asociados a ninguna **foto**. | Mostrar todas las **fotos** y **comentarios**, incluidos los no relacionados entre sí. |
| Relacionar **usuarios** con sus **comentarios** realizados. | Mostrar todos los **usuarios**, incluyendo aquellos que no han realizado ningún **comentario**. | Mostrar todos los **comentarios**, incluso los que no están asociados a ningún **usuario**. | Mostrar todos los **usuarios** y **comentarios**, incluidos los no relacionados entre sí. |

Diagrama

Descripción generada automáticamenteCLÍNICA

DIAGRAMA

Diagrama

Descripción generada automáticamente

Cursos ONLINEDiagrama

Descripción generada automáticamente

REDES SOCIALESDiagrama

Descripción generada automáticamente

DIAGRAMA

Diagrama

Descripción generada automáticamente

Subir información del informe y Scritp al Git hub