



75.15 / 95.05 / TA044
Base de Datos - 2do cuatrimestre, 2025

Docentes:

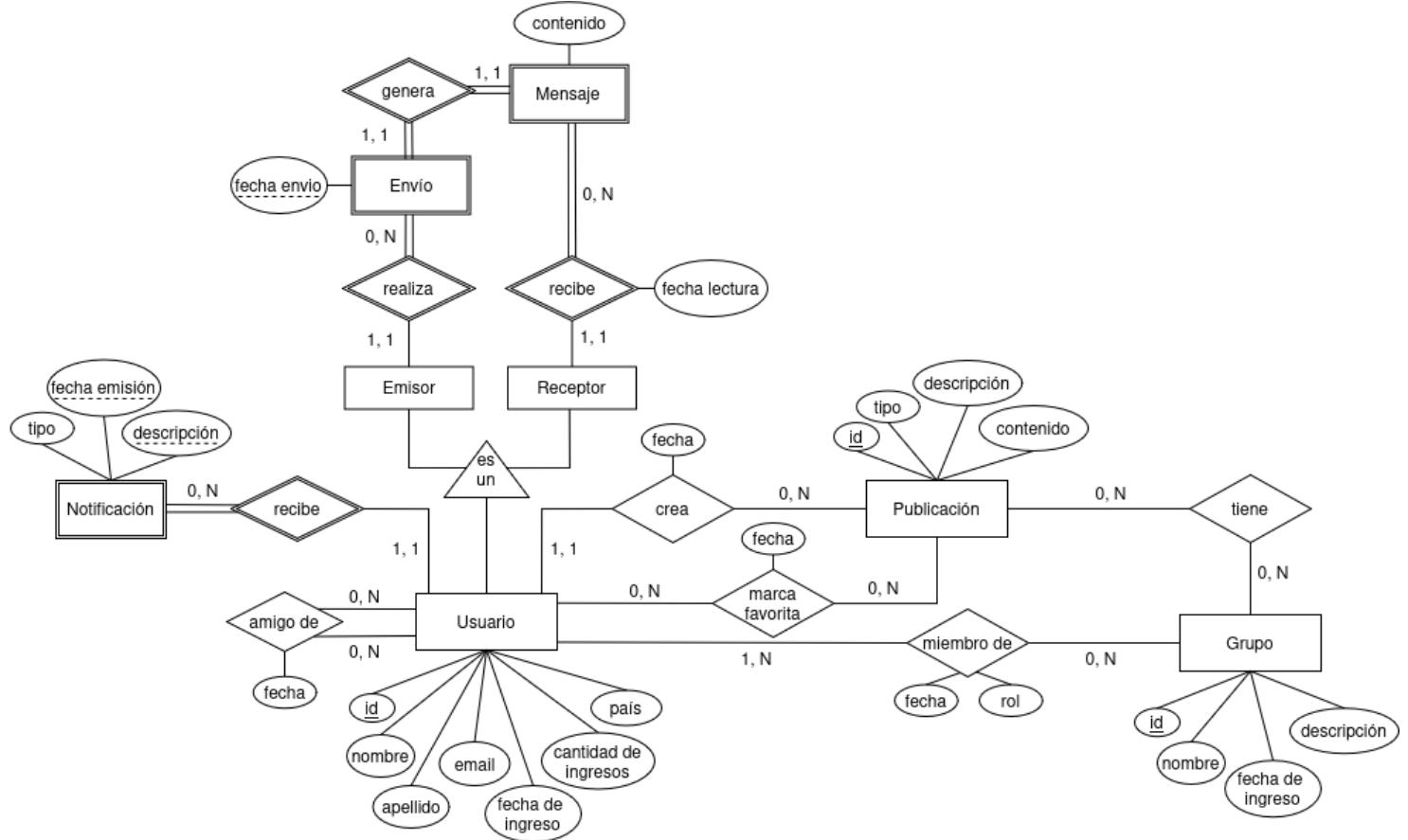
Cabrera, José Luis – Jefe de cátedra
Bernardi, Adrián Sergio – Jefe de trabajos prácticos
Aquista, Kailásh – Ayudante

Grupo 5:

Trebejo, Luis – 109988
Laos Pinto, Luis Gonzalo – 109967
Fiorilo, Roy – 108419
Pranteda, Joani – 108303
Maddalena, Martin Andres – 107610

Modelo Entidad-Relación

Dada la consigna se decidió realizar el siguiente diagrama de entidad-relación:



Hipótesis realizadas:

- Un usuario puede ser emisor y receptor de mensajes, por eso usamos especialización en la entidad **Usuario**.
- Un usuario puede subir una publicación a más de un grupo o tener publicaciones que no pertenezcan a ningún grupo (publicaciones personales).
- Un usuario puede crear varias publicaciones pero una publicación pertenece a un sólo usuario, no existen publicaciones creadas por varios usuarios.
- El primer usuario miembro de un grupo es el creador, por lo cual siempre los grupos tienen un usuario miembro como mínimo.

Entidades

• **Usuario:**

Clave candidata = clave primaria = *id_usuario*.

Se decidió que **Usuario** tenga clave subrogada *id_usuario* para asegurar un identificador único, inmutable y eficiente, independiente de los datos personales del usuario.

• **Emisor:**

Clave candidata = clave primaria = *id_usuario*.

Emisor es una especialización de **Usuario** sin atributos propios.

• **Receptor:**

Clave candidata = clave primaria = *id_usuario*.

Receptor es una especialización de **Usuario** sin atributos propios.

- **Publicación:**

Clave candidata = clave primaria = *id_publicacion*.

Usamos la clave subrogada *id_publicacion* para asegurar eficiencia en la identificación de cada publicación, dado que sus atributos naturales lo garantizan.

- **Grupo:**

Clave candidata = clave primaria = *id_grupo*.

Se optó por una clave subrogada para simplificar la identificación y la relación con otras entidades como **Usuarios** o **Publicaciones**.

- **Notificación:**

Clave candidata = clave primaria = *id_receptor, descripción, fecha_emisión*.

Entidad débil dado que no tiene suficientes atributos para poder identificarse por si misma, tiene los atributos discriminantes (*descripción* y *fecha_emisión*) y junto con *id_receptor* forman la clave candidata y primaria.

- **Envío:**

Clave candidata = clave primaria = *id_emisor, fecha_envío*.

Entidad débil que depende de **Emisor**, creada para representar la acción de envío y almacenar el atributo *fecha_envío*, permitiendo modelar correctamente la relación entre **Emisor** y **Mensaje**. Su identificación se logra a partir de la clave de la entidad fuerte **Emisor** junto con el discriminante *fecha_envío*.

- **Mensaje:**

Clave candidata = clave primaria = *id_emisor, id_receptor, fecha_envío*

Entidad débil que depende de **Envío**, la cual a su vez es una entidad débil dependiente de **Emisor**. Por lo tanto, **Mensaje** no puede existir sin la existencia previa de un **Envío** realizado por un **Emisor**. Su identificación se logra a partir de las claves primarias de las entidad fuertes finales **Emisor** y **Receptor** y del atributo discriminante (*fecha_envío*) de la entidad intermedia **Envío**, sin requerir un atributo discriminante, dado que cada envío genera un único mensaje.

Relaciones

- **Amigo de:**

Clave candidata = clave primaria = *id_usuario*.

- **Crea:**

Clave candidata = clave primaria = *id_usuario*.

- **Marca favorita:**

Clave candidata = clave primaria = *id_usuario*.

- **Tiene:**

Clave candidata = clave primaria = *id_publicacion*.

- **Miembro de:**

Clave candidata = clave primaria = *id_grupo*.

- **Realiza:**

Clave candidata = clave primaria = *id_emisor, fecha_envío*.

- **Genera:**

Clave candidata = clave primaria = *id_emisor, id_receptor, fecha_envío*.

- **Recibe:**

(Emisor → Mensaje)

Clave candidata = clave primaria = *id_emisor, id_receptor, fecha_envío*.

(Usuario → Notificación)

Clave candidata = clave primaria = *id_receptor, descripción, fecha_emisión*.

Pasaje a Modelo Relacional

Del diagrama del modelo entidad-relación mostrado anteriormente se define el modelo relacional según las siguientes relaciones:

- **Usuarios**(*id_usuario*, nombre, apellido, email, fecha_ingreso, cantidad_ingresos, pais).

- **Amistades**(*id_usuario_1*, *id_usuario_2*, *fecha_amistad*).
- **Grupos**(*id_grupo*, nombre, descripcion, fecha_creacion).
- **Usuarios_grupos**(*id_usuario*, *id_grupo*, *fecha_ingreso*, *rol_usuario*).
- **Publicaciones**(*id_publicacion*,*id_usuario*, tipo, *fecha_creacion*, descripcion, contenido).
- **Publicaciones_grupos**(*id_publicacion*, *id_grupo*).
- **Usuarios_publicaciones_favoritas**(*id_usuario*, *id_publicacion*, *fecha_favorito*).
- **Notificaciones**(*id_receptor*, descripcion, *fecha_emision*, tipo).
- **Mensajes**(*id_receptor*, *id_emisor*, *fecha_envio*, *fecha_lectura*, contenido).

Las claves foráneas son las siguientes:

- *id_usuario_1* e *id_usuario_2* en la relación **Amistades** que referencian ambos a **Usuarios**(*id_usuario*).
- *id_usuario* e *id_grupo* en la relación **Usuarios_grupos** que referencian a **Usuarios**(*id_usuario*) y **Grupos**(*id_grupo*), respectivamente.
- *id_usuario* en la relación **Publicaciones** que hace referencia a **Usuarios**(*id_usuario*).
- *id_publicacion* e *id_grupo* en la relación **Publicaciones_grupos** que referencian a **Publicaciones**(*id_publicacion*) y **Grupos**(*id_grupo*), respectivamente.
- *id_usuario* e *id_publicacion* en la relación **Usuarios_publicaciones_favoritas** que referencia a **Usuarios**(*id_usuario*) y **Publicaciones**(*id_publicacion*), respectivamente.
- *id_receptor* en la relación **Notificaciones** que referencia a **Usuarios**(*id_usuario*).
- *id_receptor* e *id_emisor* en la relación **Mensajes** que referencian ambos a **Usuarios**(*id_usuario*).

Justificación de pasaje

- **Usuarios:** Posee su *id_usuario* como clave primaria subrogada, junto con atributos que describen al usuario como su nombre, apellido, email, fecha de ingreso, cantidad de ingresos y país.
- **Amistades:** Tiene como clave primaria compuesta los atributos *id_usuario_1*, *id_usuario_2*, junto con la fecha en que se concretó la amistad. Este modelo permite representar relaciones recíprocas entre usuarios sin redundancia, evitando almacenar la misma amistad dos veces en distinto orden. Ambas claves foráneas hacen referencia a la entidad **Usuarios**.
- **Grupos:** Posee su *id_grupo* como clave primaria subrogada y atributos descriptivos como nombre, descripción y fecha de creación.
- **Usuarios_grupos:** Tiene una clave primaria compuesta por *id_usuario* e *id_grupo*, que además son claves foráneas de las respectivas tablas **Usuarios** y **Grupos**. Incluye atributos adicionales como la fecha de ingreso y el rol del usuario dentro del grupo. Este modelo permite representar correctamente la relación de pertenencia entre usuarios y grupos, donde un usuario puede formar parte de varios grupos.
- **Publicaciones:** Posee su *id_publicacion* como clave primaria subrogada, junto con atributos como tipo, fecha de creación, descripción y contenido. Incluye una clave foránea *id_usuario* que identifica al autor de la publicación.
- **Publicaciones_grupos:** Tiene una clave primaria compuesta por *id_publicacion* e *id_grupo*, ambas claves foráneas hacia **Publicaciones** y **Grupos**. Permite asociar una publicación a uno o varios grupos, evitando duplicidades y representando la relación de forma eficiente.
- **Usuarios_publicaciones_favoritas:** Tiene una clave primaria compuesta por *id_usuario* e *id_publicacion*, que indican qué publicación fue marcada como favorita por cada usuario. La fecha en que se marcó como favorita se almacena como atributo adicional. Este diseño refleja una relación de muchos a muchos entre usuarios y publicaciones, permitiendo que un usuario tenga múltiples publicaciones favoritas.

- **Notificaciones:** Posee una clave primaria compuesta por *id_receptor*, *descripcion*, *fecha_emision*, identificando de forma única cada notificación recibida por un usuario. El atributo *id_receptor* es clave foránea hacia **Usuarios**. Se optó por incluir la descripción y la fecha de emisión en la clave, ya que un mismo usuario puede recibir múltiples notificaciones con la misma descripción en diferentes momentos, siendo la fecha de emisión el atributo que permite distinguirlas de forma única.
- **Mensajes:** Posee una clave primaria compuesta por *id_emisor*, *id_receptor*, *fecha_envio*, que identifica de forma única cada mensaje enviado entre usuarios. Ambos identificadores son claves foráneas que referencian a la entidad **Usuarios**. Se incluyen atributos como fecha de lectura y contenido. Este diseño permite representar de forma completa la comunicación entre usuarios, garantizando unicidad de cada envío.

Formas Normales

- **Primera forma normal (1FN):** No hay atributos compuestos ni multivaluados y todas las columnas de todas las tablas son atómicas.
- **Segunda forma normal (2FN):** Se cumple la primera forma normal y todo atributo no primo depende de la clave primaria completa, no de una parte de ella (aplica solo a claves compuestas):
 - **Usuarios:**
 $\text{id_usuario} \rightarrow \text{nombre, apellido, email, fecha_ingreso, cantidad_ingresos, pais.}$
 - **Amistades:**
 $\text{id_usuario_1, id_usuario_2} \rightarrow \text{fecha_amistad.}$
 - **Grupos:**
 $\text{id_grupo_1} \rightarrow \text{nombre, descripcion, fecha_creacion.}$
 - **Usuarios_grupos:**
 $\text{id_usuario, id_grupo} \rightarrow \text{fecha_ingreso, rol_usuario.}$
 - **Publicaciones:**
 $\text{id_publicacion} \rightarrow \text{id_usuario, tipo, fecha_creacion, descripcion, contenido.}$
 - **Publicaciones_grupos:**
 No posee atributos no primos.
 - **Usuarios_publicaciones_favoritas:**
 $\text{id_usuario, id_publicacion} \rightarrow \text{fecha_favorito.}$
 - **Notificaciones:**
 $\text{id_receptor, descripcion, fecha_emision} \rightarrow \text{tipo.}$
 - **Mensajes:**
 $\text{id_receptor, id_emisor, fecha_envio} \rightarrow \text{fecha_lectura, contenido.}$
- **Tercera forma normal (3FN):** Se cumple la segunda forma normal y no existen dependencias transitivas (es decir, ningún atributo no primo depende de otro atributo no primo):
 - En las relaciones **Usuarios**, **Amistades**, **Grupos**, **Usuarios_grupos**, **Publicaciones**, **Usuarios_publicaciones_favoritas**, **Notificaciones** y **Mensajes** todos los atributos no primos dependen directamente de la clave primaria.
 - La relación **Publicaciones_grupos** no tiene atributos adicionales que pueda generar dependencias transicionales.
- **Forma Normal de Boyce-Codd (FNBC):** Una relación está en BCNF si y sólo si para toda dependencia funcional no trivial $X \rightarrow Y$, X es superclave: En todas las relaciones, toda dependencia funcional no trivial tiene como determinante a una superclave. Esto elimina redundancias lógicas y garantiza la máxima descomposición sin pérdida de información ni introducción de dependencias espurias.

Creación de tablas

A continuación se mostrará el código para la creación de las tablas:

```
1 CREATE TYPE PAIS AS ENUM ('Argentina', 'Chile', 'Paraguay', 'Bolivia', 'Peru', 'Uruguay');
2
3 CREATE TABLE Usuarios (
4     id_usuario INT GENERATED ALWAYS AS IDENTITY PRIMARY KEY,
5     nombre VARCHAR(100) NOT NULL,
6     apellido VARCHAR(100) NOT NULL,
7     email VARCHAR(255) UNIQUE NOT NULL,
8     fecha_ingreso DATE NOT NULL,
9     cantidad_ingresos INT DEFAULT 0,
10    pais PAIS NOT NULL,
11    CONSTRAINT chk_email CHECK (email LIKE '%@gmail.com')
12);
13
14 CREATE TABLE Grupos (
15     id_grupo INT PRIMARY KEY,
16     nombre VARCHAR(255) NOT NULL,
17     descripcion VARCHAR(255) NOT NULL,
18     fecha_creacion DATE NOT NULL
19);
20
21 CREATE TABLE Usuarios_Grupos (
22     id_usuario INT NOT NULL,
23     id_grupo INT NOT NULL,
24     fecha_ingreso DATE NOT NULL,
25     rol_usuario VARCHAR(50) NOT NULL,
26     PRIMARY KEY (id_usuario, id_grupo),
27     FOREIGN KEY (id_usuario) REFERENCES Usuarios(id_usuario) ON DELETE CASCADE,
28     FOREIGN KEY (id_grupo) REFERENCES Grupos(id_grupo)
29);
30
31 CREATE TABLE Amistades (
32     id_usuario_1 INT NOT NULL,
33     id_usuario_2 INT NOT NULL,
34     fecha_amistad DATE NOT NULL,
35     PRIMARY KEY (id_usuario_1, id_usuario_2),
36     FOREIGN KEY (id_usuario_1) REFERENCES Usuarios(id_usuario) ON DELETE CASCADE,
37     FOREIGN KEY (id_usuario_2) REFERENCES Usuarios(id_usuario) ON DELETE CASCADE,
38     CHECK (id_usuario_1 < id_usuario_2)
39);
40
41 CREATE TABLE Notificaciones (
42     tipo VARCHAR(50) NOT NULL,
43     fecha_emision TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
44     descripcion VARCHAR(255) NOT NULL,
45     id_receptor INT NOT NULL,
46     PRIMARY KEY (id_receptor, descripcion, fecha_emision),
47     FOREIGN KEY (id_receptor) REFERENCES Usuarios(id_usuario) ON DELETE CASCADE
48);
49
50 CREATE TABLE Mensajes (
51     contenido VARCHAR(255) NOT NULL,
52     fecha_envio TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
53     fecha_lectura TIMESTAMP,
54     id_emisor INT NOT NULL,
55     id_receptor INT NOT NULL,
56     PRIMARY KEY (id_emisor, id_receptor, fecha_envio),
57     FOREIGN KEY (id_emisor) REFERENCES Usuarios(id_usuario) ON DELETE CASCADE,
58     FOREIGN KEY (id_receptor) REFERENCES Usuarios(id_usuario) ON DELETE CASCADE,
59     CHECK (fecha_lectura IS NULL OR fecha_lectura >= fecha_envio)
60);
```

```

61 CREATE TABLE Publicaciones (
62     id_publicacion INT GENERATED ALWAYS AS IDENTITY PRIMARY KEY,
63     tipo VARCHAR(50) NOT NULL,
64     fecha_creacion TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
65     descripcion VARCHAR(255) NOT NULL,
66     id_usuario INT NOT NULL,
67     contenido BYTEA NOT NULL,
68     FOREIGN KEY (id_usuario) REFERENCES Usuarios(id_usuario) ON DELETE CASCADE
69 );
70
71 CREATE TABLE Publicaciones_Grupo (
72     id_publicacion INT NOT NULL,
73     id_grupo INT NOT NULL,
74     PRIMARY KEY (id_publicacion, id_grupo),
75     FOREIGN KEY (id_publicacion) REFERENCES Publicaciones(id_publicacion) ON DELETE CASCADE,
76     FOREIGN KEY (id_grupo) REFERENCES Grupos(id_grupo)
77 );
78
79 CREATE TABLE Usuarios_Publicaciones_Favoritas (
80     id_usuario INT NOT NULL,
81     id_publicacion INT NOT NULL,
82     fecha_favorito TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
83     PRIMARY KEY (id_usuario, id_publicacion),
84     FOREIGN KEY (id_usuario) REFERENCES Usuarios(id_usuario) ON DELETE CASCADE,
85     FOREIGN KEY (id_publicacion) REFERENCES Publicaciones(id_publicacion) ON DELETE CASCADE
86 );

```

Permisos

A continuación se mostrará el código para la creación de los 2 tipos de roles, mostrando sobre que tablas va a tener los respectivos permisos:

```

1 --Creacion de roles.
2 CREATE ROLE administrador LOGIN;
3 CREATE ROLE usuario LOGIN;
4
5 --Permisos para el administrador (acceso total).
6 GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA tp TO administrador;
7
8 --Permisos para el usuario.
9 GRANT INSERT ON
10    Publicaciones ,
11    Grupos ,
12    Mensajes
13 TO usuario;

```

La justificación de los permisos para cada rol:

- **Administrador:** Este debe ser capaz de interactuar con todas las tablas de la base (o esquema), con lo cual se le da permiso general de todas las tablas pertenecientes al esquema tp (esquema en el cual fueron creadas las tablas).
- **Usuario:** Se le otorgó el permiso de INSERT al Usuario porque, dentro del sistema, los usuarios deben poder crear publicaciones, formar nuevos grupos y enviar mensajes a otros.

Ejemplos de inserción

A continuación se mostrará el código con algunos ejemplos de inserción:

```

1  INSERT INTO Usuarios (nombre, apellido, email, fecha_ingreso, cantidad_ingresos, pais) VALUES
2  ('Joani', 'Pranteda', 'joani_pranteda@gmail.com', '2023-03-15', 25, 'Argentina'),
3  ('Daira', 'Gomez', 'daira_gomez@gmail.com', '2023-05-10', 15, 'Argentina'),
4  ('Naomy', 'Lopez', 'naomy_lopez@gmail.com', '2022-11-20', 40, 'Chile'),
5  ('Lucas', 'Martinez', 'lucas_martinez@gmail.com', '2024-01-05', 8, 'Uruguay'),
6  ('Sofia', 'Perez', 'sofia_perez@gmail.com', '2024-07-02', 5, 'Argentina'),
7  ('Diego', 'Gomez Perci', 'diego_gomezperci@gmail.com', '2022-09-13', 60, 'Argentina'),
8  ('Camila', 'Fernandez', 'camila_fernandez@gmail.com', '2023-10-01', 12, 'Paraguay'),
9  ('Mateo', 'Sanchez', 'mateo_sanchez@gmail.com', '2024-03-18', 10, 'Bolivia'),
10 ('Valentina', 'Suarez', 'valentina_suarez@gmail.com', '2023-12-28', 22, 'Peru'),
11 ('Martina', 'Rossi', 'martina_rossi@gmail.com', '2022-06-19', 33, 'Argentina');

12
13 INSERT INTO Grupos (id_grupo, nombre, descripcion, fecha_creacion) VALUES
14 (1, 'Programadores Java', 'Grupo dedicado al intercambio de conocimientos sobre Java', '2023-04-12'),
15 (2, 'Cafe y Charlas', 'Un espacio para conversar sobre cualquier tema con buena compania', '2023-09-02'),
16 (3, 'Modelacion Numerica', 'Estudiantes y docentes compartiendo ejercicios y tips', '2024-03-06'),
17 (4, 'Gamers', 'Comunidad para organizar partidas online y hablar de videojuegos', '2023-10-15'),
18 (5, 'Musica y Produccion', 'Intercambio de ideas y recursos sobre produccion musical', '2023-07-20');

19
20 INSERT INTO Usuarios_Grupos (id_usuario, id_grupo, fecha_ingreso, rol_usuario) VALUES
21 (1, 1, '2023-04-12', 'Administrador'),
22 (1, 2, '2023-09-01', 'Miembro'),
23 (2, 2, '2023-09-02', 'Administrador'),
24 (3, 3, '2024-03-06', 'Administrador'),
25 (4, 4, '2023-10-15', 'Administrador'),
26 (5, 4, '2023-10-16', 'Miembro'),
27 (6, 1, '2023-04-13', 'Miembro'),
28 (7, 5, '2023-07-20', 'Administrador'),
29 (8, 5, '2023-08-01', 'Miembro'),
30 (9, 3, '2024-03-07', 'Miembro'),
31 (10, 2, '2023-09-03', 'Miembro');

32
33 INSERT INTO Amistades (id_usuario_1, id_usuario_2, fecha_amistad) VALUES
34 (1, 2, '2023-06-10'),
35 (1, 3, '2023-07-21'),
36 (1, 4, '2024-02-01'),
37 (2, 5, '2023-10-10'),
38 (3, 6, '2022-12-05'),
39 (4, 7, '2024-01-14'),
40 (5, 8, '2024-03-30'),
41 (6, 9, '2023-05-22'),
42 (7, 10, '2023-12-12'),
43 (9, 10, '2024-05-02');

44
45 INSERT INTO Notificaciones (tipo, fecha_emision, descripcion, id_receptor) VALUES
46 ('Solicitud de amistad', '2023-06-10 10:15:00', 'Daira te ha enviado una solicitud de amistad .', 1),
47 ('Publicacion en grupo', '2023-09-01 08:30:00', 'Nueva publicacion en el grupo "Cafe y Charlas".', 1),
48 ('Publicacion de amigo', '2023-07-21 14:45:00', 'Tu amiga Naomy ha compartido una nueva publicacion.', 1),
49 ('Publicacion en grupo', '2023-10-05 09:00:00', 'Nueva publicacion en el grupo "Modelacion Numerica".', 9),
50 ('Solicitud de amistad', '2024-01-14 18:00:00', 'Lucas te ha enviado una solicitud de amistad .', 7),
51 ('Solicitud de amistad', '2024-04-09 16:20:00', 'Sofia te ha enviado una solicitud de amistad .', 6);

```

```

52 INSERT INTO Mensajes (contenido, fecha_envio, fecha_lectura, id_emisor, id_receptor) VALUES
53 ('Hola Daira, como estas?', '2023-06-10 10:00:00', '2023-06-10 10:05:00', 1, 2),
54 ('Todo bien Joani, vos?', '2023-06-10 10:06:00', '2023-06-10 10:07:00', 2, 1),
55 ('Te pase el codigo del ejercicio de LU', '2023-07-21 15:10:00', '2023-07-21 15:20:00', 3, 1)
56 ,
57 ('Nos juntamos a estudiar?', '2023-07-21 16:00:00', '2024-03-07 16:10:00', 1, 3),
58 ('Jugamos esta noche?', '2024-07-10 18:00:00', '2024-07-10 18:10:00', 4, 5),
59 ('Dale, me conecto a las 22', '2024-07-10 18:20:00', NULL, 5, 4),
60 ('Che Diego, te debo algo?', '2023-11-01 11:00:00', '2023-11-01 11:05:00', 1, 6),
61 ('No, al contrario, yo te debo a vos jajaja', '2023-11-02 11:10:00', NULL, 6, 1);

62 INSERT INTO Publicaciones (tipo, fecha_creacion, descripcion, id_usuario, contenido) VALUES
63 ('Texto', '2023-06-01 09:15:00', 'Reflexion matutina', 1, convert_to('Cada error me acerca un
    paso mas al exito.', 'UTF8')),
64 ('Texto', '2023-09-02 08:45:00', 'Pensamiento del dia', 3, convert_to('Estudiar sin pausa,
    pero sin prisa.', 'UTF8')),
65 ('Texto', '2024-02-10 11:30:00', 'Motivacion del dia', 4, convert_to('El codigo limpio
    tambien es arte.', 'UTF8')),
66 ('Texto', '2024-04-01 18:00:00', 'Inspiracion musical', 9, convert_to('La musica habla cuando
    las palabras fallan.', 'UTF8')),
67 ('Texto', '2024-07-25 22:15:00', 'Cierre de jornada', 10, convert_to('Hoy aprendi que siempre
    hay algo nuevo que descubrir.', 'UTF8')),
68 ('Imagen', '2023-09-02 09:30:00', 'Foto del grupo en el cafe', 2, decode(
    FFD8FFE000104A4649460001AABBCCDDEEFF', 'hex')),
69 ('Imagen', '2024-03-06 14:10:00', 'Grafico del proyecto final', 3, decode('89504
    E470D0A1A0A0000000D49484452', 'hex')),
70 ('Video', '2023-07-20 13:00:00', 'Demo de nueva cancion', 7, decode('52494646
    AABBCDD574156456D70342066696C65', 'hex')));

71 INSERT INTO Publicaciones_Grupo (id_publicacion, id_grupo) VALUES
72 (2, 3),
73 (3, 4),
74 (4, 3),
75 (5, 2),
76 (6, 2),
77 (7, 3),
78 (8, 5);

81 INSERT INTO Usuarios_Publicaciones_Favoritas (id_usuario, id_publicacion, fecha_favorito)
     VALUES
82 (1, 2, '2023-09-03 11:00:00'),
83 (1, 5, '2024-07-26 13:15:00'),
84 (2, 1, '2023-06-02 10:30:00'),
85 (3, 3, '2024-03-10 09:00:00'),
86 (4, 4, '2024-04-02 20:10:00'),
87 (5, 2, '2023-09-05 15:00:00'),
88 (6, 1, '2023-06-02 08:00:00'),
89 (7, 5, '2024-07-26 14:30:00'),
90 (8, 6, '2024-03-08 16:45:00'),
91 (9, 7, '2024-05-03 11:10:00'),
92 (10, 3, '2024-03-12 09:20:00');

```

Resolución de consultas

A continuación se mostrará el código para la realización de las consultas solicitadas en el enunciado:

```

1 -- Registrar un usuario.
2 INSERT INTO Usuarios (nombre, apellido, email, fecha_ingreso, cantidad_ingresos, pais) VALUES
3 ('Rogelio', 'Buen Dia', 'rogelio_buendia@gmail.com', '2024-03-18', 1, 'Argentina');
4
5 -- Listar todos los usuarios de la red social.
6 SELECT *
7 FROM usuarios;

```

```

8   -- Listar todas las amistades de la red social.
9   SELECT u.nombre AS nombre_usuario, u.apellido AS apellido_usuario, us.nombre AS nombre_amigo,
10    us.apellido AS apellido_amigo, a.fecha_amistad
11   FROM Usuarios u
12   JOIN Amistades a ON u.id_usuario = a.id_usuario_1
13   JOIN Usuarios us ON a.id_usuario_2 = us.id_usuario;
14
15   -- Listar los amigos de un usuario particular de la red social.
16   SELECT
17     u2.id_usuario,
18     u2.nombre,
19     u2.apellido
20   FROM Amistades a
21   INNER JOIN Usuarios u2 ON a.id_usuario_2 = u2.id_usuario
22   WHERE a.id_usuario_1 = 10
23
24   UNION
25
26   SELECT
27     u1.id_usuario,
28     u1.nombre,
29     u1.apellido
30   FROM Amistades a
31   INNER JOIN Usuarios u1 ON a.id_usuario_1 = u1.id_usuario
32   WHERE a.id_usuario_2 = 10
33   ORDER BY nombre, apellido;
34
35   -- Listar todos los mensajes de la red social.
36   SELECT *
37   FROM Mensajes;
38
39   -- Contabilizar la cantidad de usuarios, agrupados por pais.
40   SELECT pais, COUNT(*) AS cantidad_usuarios
41   FROM Usuarios
42   GROUP BY pais
43   ORDER BY cantidad_usuarios DESC;
44
45   -- Realizar una publicacion (dar un ejemplo de cada tipo).
46   -- Texto.
47   INSERT INTO Publicaciones (tipo, fecha_creacion, descripcion, id_usuario, contenido) VALUES
('Texto', '2024-08-10 10:00:00', 'Inicio de proyecto', 5, convert_to('Empezando el desarrollo del nuevo sistema, motivacion al maximo', 'UTF8'));
48
49   -- Imagen.
50   INSERT INTO Publicaciones (tipo, fecha_creacion, descripcion, id_usuario, contenido) VALUES
('Imagen', '2024-09-12 14:25:00', 'Vista desde el laboratorio', 6, decode('89504E470D0A1AOA0000000D49484452AABBCCDD', 'hex'));
51
52   -- Video.
53   INSERT INTO Publicaciones (tipo, fecha_creacion, descripcion, id_usuario, contenido) VALUES
('Video', '2024-10-05 19:45:00', 'Resumen de la hackathon', 8, decode('52494646BBAACDD574156456D7034323032', 'hex'));
54
55   -- Actualizar una publicacion (dar un ejemplo de cada tipo).
56   -- Texto.
57   UPDATE Publicaciones
58   SET descripcion = 'Actualizacion de avance del proyecto',
59     contenido = convert_to('El modulo de autenticacion ya esta completo y funcionando correctamente.', 'UTF8')
60   WHERE id_publicacion = 12;
61
62

```

```

63      -- Imagen.
64 UPDATE Publicaciones
65 SET descripcion = 'Nueva foto del laboratorio con el equipo',
66     contenido = decode('FFD8FFE000104A4649460001AABBCCDD99887766', 'hex')
67 WHERE id_publicacion = 13;
68
69      -- Video.
70 UPDATE Publicaciones
71 SET descripcion = 'Video editado: resumen final de la hackathon',
72     contenido = decode('52494646CCBBAADD574156456D70343230335F757064', 'hex')
73 WHERE id_publicacion = 14;
74
75      -- Eliminar una publicacion (dar un ejemplo de cada tipo).
76 DELETE from Publicaciones
77 WHERE id_publicacion = 1;
78
79      -- Desegistrar a un usuario de la aplicacion (dar un ejemplo).
80 DELETE FROM Usuarios WHERE id_usuario = 1;
81 -- SELECT * FROM Usuarios; --El usuario desaparecio de la tabla usuarios
82 -- SELECT * FROM Amistades; --El usurio se elimina junto a sus amistades
83 -- SELECT * FROM Publicaciones_Grupo; --Se elimina la referencia a esa id_publicacion que
84     realizo ese usurio
85 -- SELECT * FROM Publicaciones --Se elimina todo registro done el usurio haya hecho una
86     publicacion
87 -- SELECT * FROM Usuarios_publicaciones_favoritas --Se elimina los registros del usurio con
88     sus publicaciones favoritas y si un usurio tenia como favorito una publicacion del
89     usurio eliminado este registro tambien se elimina
90 ;
91
92      -- Mostrar las publicaciones mas populares ordenadas por cantidad de favoritos que poseen.
93 SELECT p.id_publicacion, p.descripcion, p.tipo, p.fecha_creacion,
94       u.nombre || ' ' || u.apellido AS autor,
95       COUNT(upf.id_publicacion) AS cantidad_favoritos
96 FROM Publicaciones p
97 JOIN Usuarios u ON p.id_usuario = u.id_usuario
98 LEFT JOIN Usuarios_publicaciones_favoritas upf ON p.id_publicacion = upf.id_publicacion
99 GROUP BY p.id_publicacion, p.descripcion, p.tipo, p.fecha_creacion, u.nombre, u.apellido
100 ORDER BY cantidad_favoritos DESC;
101
102      -- Mostrar los usuarios mas populares basandose en la cantidad de publicaciones favoritas que
103     poseen sus publicaciones.
104 WITH publicaciones_favoritas(publicacion, cantidad) AS (
105     SELECT id_publicacion, COUNT(*)
106     FROM usuarios_publicaciones_favoritas
107     GROUP BY id_publicacion
108 )
109
110     SELECT *
111     FROM usuarios
112     WHERE id_usuario IN (
113         SELECT pu.id_usuario
114         FROM publicaciones pu
115         JOIN publicaciones_favoritas pf ON pu.id_publicacion = pf.publicacion
116         WHERE pf.cantidad = (SELECT MAX(cantidad) FROM publicaciones_favoritas)
117     );

```