Tecnológico de Monterrey

Actividad 4: Diseño de Base de Datos del Reto.

Integración de Seguridad Informática en Redes y Sistemas de Software (TC2007B).

Grupo 452.

Santiago Ramírez Niño - A01665906.

Gabriel Gutiérrez Guerra - A01660505.

Alejandro Ignacio Vargas Cruz - A01659714.

Omar Llano Tostado - A01666730.

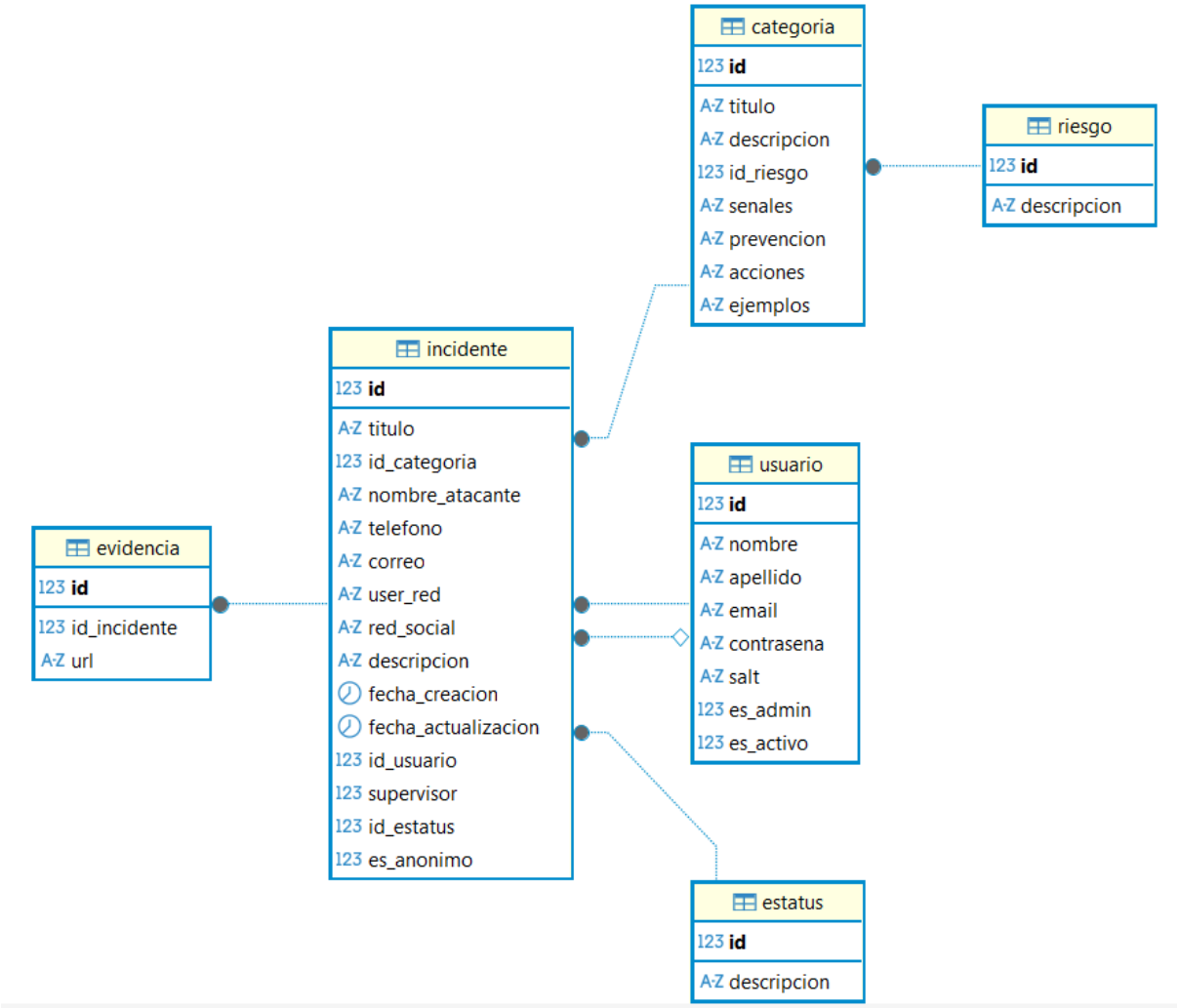
Equipo 10.

Profesor: Diogo Miguel Burnay Rojas.

Miércoles, 01 de octubre de 2025.

1. DIAGRAMA ENTIDAD-RELACIÓN.

Diseño de las tablas de la base de datos:



2. DOCUMENTACIÓN DEL DISEÑO.

2.1 Descripción de Entidades.

USUARIO.

- **Propósito**: Almacena información de los usuarios del sistema, tanto usuarios finales que reportan incidentes en la aplicación, como administradores que los supervisan desde el sistema web.
- **Campos principales**:
  - id (BIGINT): Clave primaria, no nula, auto-incremental.
  - nombre, apellido (VARCHAR 120): Datos personales del usuario, no nulos.
  - email (VARCHAR 190, UNIQUE): Identificador único para login, no nulo.

- contrasena (VARCHAR 256): Hash de la contraseña, no nulo.
- salt (VARCHAR 64): Salt para seguridad de contraseña, no nulo.
- es_admin (BOOLEAN): Indica si el usuario tiene permisos administrativos para el sistema web, no nulo, y por default será falso (si es registro desde la app, si es desde el sitio web, será true).
- es_activo (BOOLEAN): Estatus del usuario, útil para el borrado lógico de usuarios, no nulo, y por default será true.

RIESGO.

- **Propósito**: Catálogo que clasifica el nivel de riesgo de las categorías a las que pertenecen los incidentes de ciberseguridad.
- **Campos principales**:
  - id (TINYINT): Clave primaria (suficiente para pocos niveles: bajo, medio, alto, crítico), no nula, auto-incremental.
  - descripcion (VARCHAR 255): Descripción del nivel de riesgo, no nula.

CATEGORÍA.

- **Propósito**: Define las categorías de incidentes de ciberseguridad con información detallada sobre cada tipo.
- **Campos principales**:
  - id (TINYINT): Clave primaria (suficiente para categorías específicas), no nula, auto-incremental.
  - titulo (VARCHAR 255): Nombre de la categoría, no nulo.
  - descripcion (TEXT): Explicación detallada de la categoría, no nula.
  - id_riesgo (TINYINT, FK): Nivel de riesgo asociado a la tabla RIESGO, no nulo.
  - senales, prevencion, acciones, ejemplos (TEXT): Información educativa característica y referente a cada categoría.

ESTATUS.

- **Propósito**: Catálogo que define los estados posibles de un incidente a lo largo de su ciclo de análisis en el proceso de operación del sistema.
- **Campos principales**:
  - id (TINYINT): Clave primaria (suficiente para los 3 posibles estados contemplados), no nulo, auto-incremental.
  - descripcion (VARCHAR 100): Estado del incidente (Pendiente, Aprobado, Rechazado), no nula.

INCIDENTE.

- **Propósito**: Entidad central que almacena los reportes de incidentes de ciberseguridad realizados por usuarios desde la aplicación, mismos que serán evaluados por administradores desde el sistema web.
- **Campos principales**:

- id (BIGINT): Clave primaria de cada incidente, no nulo, auto-incremental.
- titulo (VARCHAR 255): Título descriptivo del incidente, no nulo.
- id_categoria (TINYINT, FK): Tipo de incidente asociado a la tabla de CATEGORÍA, no nulo.
- nombre_atacante, telefono, correo, user_red, red_social (VARCHAR): Datos referentes de la persona, empresa o entidad que efectuó el ciberataque, o estafa, hacia el usuario, así como sus medios de contacto. Pueden ser nulos algunos o todos ellos.
- descripcion (TEXT): Detalles del incidente, no nula.
- fecha_creacion, fecha_actualizacion (TIMESTAMP): Datos referentes a fechas del levantamiento o edición del reporte por el usuario desde la aplicación, no nulos, por default toman el valor de la fecha actual, incluso en el caso de actualización.
- id_usuario (BIGINT, FK): Clave foránea que hace referencia al usuario que levanta el reporte, hacia la tabla USUARIO, no nulo.
- supervisor (BIGINT, FK): Clave foránea que hace referencia al administrador que evalúa el reporte desde el sistema web, presente en la tabla USUARIO, puede ser nulo mientras aún no se evalúa por ningún administrador, sólo entonces tendrá un id asociado.
- id_estatus (TINYINT, FK): Clave foránea que hace referencia la tabla ESTATUS referente al estado actual del reporte, no nulo.
- es_anonimo (BOOLEAN): Indica si el reporte es anónimo, es decir, si se desea compartir el nombre de la persona que lo levantó, no nulo, y tiene como valor predeterminado false.

EVIDENCIA.

- **Propósito**: Almacena URLs de archivos (capturas, documentos) que respaldan un incidente, subidos directamente por el usuario desde la aplicación móvil.
- **Campos principales**:
  - id (BIGINT): Clave primaria asociada a cada archivo, no nula, auto-incremental.
  - id_incidente (BIGINT, FK): Incidente al que pertenece, una llave foránea que hace referencia a la tabla INCIDENTE, no nula.
  - url (VARCHAR 500): Ruta o URL del archivo, no nula.

2.2 Reglas de Negocio.

1. **Autenticación segura**: Las contraseñas se almacenan hasheadas con un salt generado de manera única por cada usuario nuevo registrado.
2. **Roles diferenciados**: es_admin determina si un usuario puede supervisar incidentes y puede tener acceso al sistema web para evaluar los reportes.
3. **Borrado lógico**: es_active permite desactivar usuarios sin eliminar su historial, y en caso que estén desactivados, no puedan ingresar ni a la aplicación móvil, ni al sistema web.

4. **Reportes anónimos**: es_anonimo permite reportes sin exponer la identidad del usuario, para mayor privacidad.
5. **Trazabilidad**: Cada incidente registra quién lo creó y quién lo supervisa por medio de llaves foráneas a la tabla USUARIO.
6. **Auditoría temporal**: fecha_creacion y fecha_actualizacion rastrean cambios en la línea de tiempo para cada incidente.
7. **Clasificación jerárquica**: Categorías a las que pertenecen los incidentes, están vinculadas a niveles de riesgo.
8. **Integridad referencial**: Todas las FK tienen constraints para evitar datos huérfanos.
9. **Evidencias**: Cada incidente puede tener una o más evicencias (se tiene contemplado hasta 5 archivos) que brinden respaldo y veracidad de su reporte.
10. **Seguimiento a reportes:** Por medio de la asignación de su estatus. Todos los nuevos reportes cuentan con estatus “Pendiente”, y un administrador al evaluarlo desde el sistema web, lo aprueba o rechaza.

2.3 Justificación de Decisiones de Diseño.

- **TINYINT para catálogos**: Optimiza espacio para tablas con pocos registros (menos de 128).
- **BIGINT para datos transaccionales**: Permite escalar a millones de incidentes, evidencias y usuarios (mayor escalabilidad).
- **Separación de evidencias**: Permite múltiples archivos por incidente sin redundancia.
- **Supervisor opcional**: Un incidente puede estar sin asignar (supervisor nullable, cuando se evalúa desde el sitio web, se asigna el valor del id de dicho usuario con privilegio de administrador).
- **Timestamps automáticos**: MySQL actualiza fecha_actualizacion automáticamente, y al crear un reporte, también toma el valor actual.
- **VARCHAR(190) para email**: Compatible con índices UTF8MB4 en MySQL
- **Campos opcionales del atacante**: Permite flexibilidad en la información disponible, puede ser que se tengan datos o no de la entidad atacante.
- **Borrado lógico de Usuarios:** Permite trazabilidad y evitar riesgos de incompatibilidad. Por ejemplo, si un usuario se desactiva, se quiere mantener su registro en la base de datos para mostrar sus aportaciones a los demás usuarios y mantener un historial organizado sin inconsistencias.
- **Entidades separadas:** Catálogos como la tabla Categoría, Riesgo o Estatus permiten asignar sólo unos cuantos registros, y serán los mismos para cada levantamiento de reportes, o tal vez con ligeras variaciones. Permite una mejor organización y jerarquización de la información. Así, cumpliendo también con la 3NF.

3. CONSULTAS EN MySQL.

3.1. CONSULTA 1: INSERT - Registro de nuevo incidente.

Propósito: Crear un nuevo reporte de incidente de phishing.

Regla de negocio: Todo incidente debe tener categoría, usuario reportante y estatus inicial previamente creados en sus tablas. Suponiendo que:

- En Categoría existe un registro con id=1 y descripción = “Phishing”.
- En Usuario existe el usuario que levanta el reporte con id=1.
- En Estatus existe un registro con id=1 y descripción=”Pendiente”.

```SQL
INSERT INTO incidente (
        titulo, id_categoria, nombre_atacante, telefono, correo,
        user_red, red_social, descripcion,
        id_usuario, supervisor, id_estatus, es_anonimo
      ) VALUES (
        'Correo sospechoso de banco', 1, 'Diogo Burnay','55-1234-5678',
'diogo@gmail.com', 'diogodev', 'Instagram',
        'Recibí un correo que pretende ser de mi banco solicitando
actualizar datos personales. El enlace redirige a un sitio no oficial.', 1,
        null, 1, 0
);
```

3.2. CONSULTA 2: SELECT con JOIN - Listar incidentes con información completa.

Propósito: Obtener reporte completo de incidentes incluyendo categoría, riesgo y estatus.

Regla de negocio: Los reportes deben mostrar información consolidada de múltiples tablas.

```SQL
SELECT
    i.id,
    i.titulo,
    c.titulo AS categoria,
    r.descripcion AS nivel_riesgo,
    e.descripcion AS estatus,
    CONCAT(u.nombre, ' ', u.apellido) AS reportado_por,
    i.fecha_creacion,
    i.es_anonimo
FROM incidente i
INNER JOIN categoria c ON i.id_categoria = c.id
INNER JOIN riesgo r ON c.id_riesgo = r.id
INNER JOIN estatus e ON i.id_estatus = e.id
```

```
LEFT JOIN usuario u ON i.id_usuario = u.id
WHERE i.es_anonimo = FALSE
ORDER BY i.fecha_creacion DESC
LIMIT 50;
```

3.3. CONSULTA 3: UPDATE - Asignar supervisor a incidente y Evaluar incidente.

Propósito: Asignar un administrador para dar seguimiento a un incidente al evaluar un incidente, así como un estatus del mismo.

Regla de negocio: Solo los administradores pueden ser supervisores (es_admin=1) y se debe asignar un id_estatus= 2 para aprobar, o 3 para rechazar.

```
SQL
UPDATE incidente
SET
    supervisor = 2,
    id_estatus = 2
WHERE id = 1
AND EXISTS (
    SELECT 1 FROM usuario
    WHERE id = 2 AND es_admin = TRUE
);
```

3.4. CONSULTA 4: DELETE lógico - Desactivar usuario.

Propósito: Deshabilitar cuenta de usuario sin eliminar su historial

Regla de negocio: Se usa borrado lógico para mantener trazabilidad de incidentes reportados. Para ello se modifica el campo es_activo de un usuario existente.

```
SQL
UPDATE usuario
SET is_active = FALSE
WHERE id = 1;
```

En el presente proyecto se realizarán borrados lógicos, sin embargo, para borrados físicos, de un incidente, por ejemplo, se ejecutaría el siguiente comando (incluyendo evidencias asociadas a dicho incidente, para evitar inconsistencias):

```SQL
DELETE FROM evidencia WHERE id_incidente = 10;
DELETE FROM incidente WHERE id = 10;
```

3.5. CONSULTA 5: Autenticación - Validar credenciales de usuario.

Propósito: Verificar login de usuario con contraseña hasheada y con la salt.

Regla de negocio: Solo usuarios existentes y activos pueden iniciar sesión.

```SQL
SELECT * FROM usuario
WHERE email = 'santiago@tec.mx'
AND es_activo = TRUE
LIMIT 1;
```

En la aplicación se hace la comparación de la contraseña hasheada con la sal para permitir la autenticación y el acceso al sistema.

3.6. CONSULTA 6: REPORTE - Estadísticas de incidentes por categoría y riesgo.

Propósito: Generar datos para dashboard con distribución de incidentes.

Regla de negocio: Los reportes deben agrupar por categoría y nivel de riesgo para análisis.

```SQL
SELECT
    r.descripcion AS nivel_riesgo,
    c.titulo AS categoria,
    COUNT(i.id) AS total_incidentes,
    SUM(CASE WHEN e.descripcion = 'Pendiente' THEN 1 ELSE 0 END) AS nuevos,
    SUM(CASE WHEN e.descripcion = 'Aprobado' THEN 1 ELSE 0 END) AS
aprobados,
```

```
    SUM(CASE WHEN e.descripcion = 'Rechazado' THEN 1 ELSE 0 END) AS
rechazados,
    ROUND(AVG(TIMESTAMPDIFF(HOUR, i.fecha_creacion, i.fecha_actualizacion)),
2) AS promedio_horas_resolucion
FROM riesgo r
INNER JOIN categoria c ON c.id_riesgo = r.id
LEFT JOIN incidente i ON i.id_categoria = c.id
LEFT JOIN estatus e ON i.id_estatus = e.id
WHERE i.fecha_creacion >= DATE_SUB(CURDATE(), INTERVAL 30 DAY)
GROUP BY r.id, c.id
ORDER BY r.id DESC, total_incidentes DESC;
```

3.7. CONSULTA 7: REVISIÓN- Obtener incidentes pendientes.

Propósito: Consultar incidentes pendientes de revisión para analizarlos.

Regla de negocio: Los reportes deben ser existentes en la tabla, con id_estatus=1 (pendientes).

```
SQL
SELECT * FROM incidente WHERE id_estatus = 1;
```

4. REGLAS DEL NEGOCIO.

4.1 Restricciones de Integridad.

Regla.	Implementación.	Ejemplo en Consultas.
Email único por usuario.	UNIQUE constraint en usuario.email.	Búsqueda por email.
Contraseñas seguras.	NOT NULL en password_hash y salt.	Retorna salt para validación.
Relaciones obligatorias.	FOREIGN KEY constraints.	Los JOINs garantizan integridad.

Datos completos.	NOT NULL en campos críticos.	El INSERT requiere campos obligatorios.

4.2 Validaciones de Estados.

- **Borrado lógico**: es_activo = FALSE mantiene usuarios deshabilitados.
- **Control de estatus**: Solo administradores asignan supervisores.
- **Auditoría temporal**: fecha_actualizacion se actualiza automáticamente.
- **Privacidad**: es_anonimo = TRUE oculta reportantes en consultas.

4.3 Filtros y Permisos.

- **Por rol**: Se verifica es_admin = TRUE antes de asignar supervisor y evaluar incidentes.
- **Por fecha**: Se filtra últimos 30 días con DATE_SUB(CURDATE(), INTERVAL 30 DAY).
- **Por estado**: Consultas agrupan por estatus para control de workflow.
- **Por activo**: Solo permite login de usuarios con es_activo = TRUE.

4.4 Cálculos para Reportes.

- **Conteo**: COUNT() para totales de incidentes por categoría.
- **Agregación condicional**: SUM(CASE WHEN...) para desglose por estatus.
- **Promedios**: AVG(TIMESTAMPDIFF()) calcula tiempo de resolución.
- **Agrupación**: GROUP BY con múltiples dimensiones (riesgo + categoría).

5. DATOS DE EJEMPLO Y CATÁLOGOS.

5.1. Inserción de datos de ejemplo.

En Riesgo:

```sql
INSERT INTO riesgo (id, descripcion) VALUES
(1, 'Bajo'),
(2, 'Medio'),
(3, 'Alto'),
(4, 'Crítico');
```

mysql> select*from riesgo;

id	descripcion
1	Bajo
2	Medio
3	Alto
4	Crítico

4 rows in set (0.00 sec)

En Incidente:

```sql
INSERT INTO incidente (
        titulo, id_categoria, nombre_atacante, telefono, correo,
        user_red, red_social, descripcion,
        id_usuario, supervisor, id_estatus, es_anonimo
      ) VALUES (
        'Correo sospechoso de banco', 1, 'Diogo Burnay','55-1234-5678',
'diogo@gmail.com', 'diogodev', 'Instagram',
        'Recibí un correo que pretende ser de mi banco solicitando
actualizar datos personales. El enlace redirige a un sitio no oficial.', 1,
        null, 1, 0
),
(
        'Robo de identidad', 2, 'Artemio Urbina','55-1234-5678', null, null,
null, 'Se hicieron pasar por un familiar.', 3,
        null, 1, 1
);
```

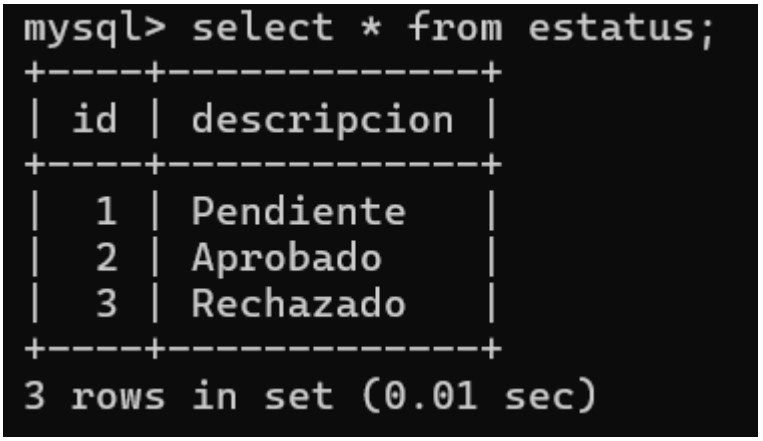
mysql> select*from incidente;

id	titulo	id_categoria	nombre_atacante	telefono	correo	user_red	red_social	descripcion	fecha_creacion	fecha_actualizacion	id_usuario	supervisor	id_estatus	es_anonimo
1	Fraude	1	Gabo	123	alex@tec.mx	@gaboawewew	Facebook	Blabla	2025-09-28 18:23:52	2025-10-01 13:32:59	1	2	2	1
3	Prueba	1	Ary	123	Ary@tec.mx	Santiago niño	Facebook	undefined	2025-09-30 16:42:52	2025-09-30 16:42:52	1	NULL	1	0
4	Prueba	1	Ary	123	Ary@tec.mx	Santiago niño	Facebook	undefined	2025-09-30 16:43:00	2025-09-30 16:43:00	1	NULL	1	0
5	Prueba	1	Aryam	123	Ary@tec.mx	@nino	Twitter/X	Aryam esta preocupada porque Nacho esta raro y le va a regalar hot wheels	2025-09-30 16:45:05	2025-09-30 16:45:05	1	NULL	1	0
6	ME VIOLARON	1	TECMX	5530529710	NULL	NULL	NULL	AYUDA ME VIOLARON	2025-09-30 19:21:50	2025-09-30 19:21:50	1	NULL	1	1
7	Correo sospechoso de banco	1	Diogo Burnay	55-1234-5678	diogo@gmail.com	diogodev	Instagram	Recibí un correo que pretende ser de mi banco solicitando actualizar datos personales. El enlace redirige a un sitio no oficial.	2025-10-01 14:42:59	2025-10-01 14:42:59	1	NULL	1	0
8	Robo de identidad	2	Artemio Urbina	55-1234-5678	NULL	NULL	NULL	Se hicieron pasar por un familiar.	2025-10-01 14:42:59	2025-10-01 14:42:59	3	NULL	1	1

7 rows in set (0.00 sec)

En Estatus:

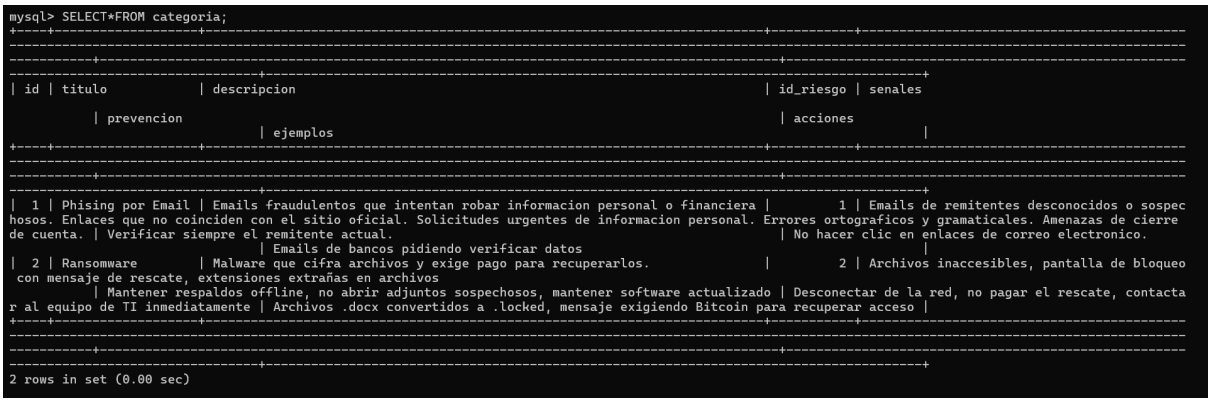
```sql
INSERT INTO estatus (id, descripcion) VALUES
(1, 'Pendiente'),
(2, 'Aprobado'),
(3, 'Rechazado');
```



En Categoría:

```sql
INSERT INTO categoria (id, titulo, descripcion, id_riesgo, senales,
prevencion, acciones, ejemplos) VALUES
(1, 'Phishing por Email',
    'Emails fraudulentos que intentan robar informacion personal o
financiera',
    1,
    'Emails de remitentes desconocidos o sospechosos. Enlaces que no
coinciden con el sitio oficial. Solicitudes urgentes de informacion
personal. Errores ortograficos y gramaticales. Amenazas de cierre de
cuenta.',
    'Verificar siempre el remitente actual. ',
    'No hacer clic en enlaces de correo electronico.',
    'Emails de bancos pidiendo verificar datos'),
(2, 'Ransomware',
    'Malware que cifra archivos y exige pago para recuperarlos.',
    2,
    'Archivos inaccesibles, pantalla de bloqueo con mensaje de rescate,
extensiones extrañas en archivos',
```

```
    'Mantener respaldos offline, no abrir adjuntos sospechosos, mantener
software actualizado',
    'Desconectar de la red, no pagar el rescate, contactar al equipo de TI
inmediatamente',
    'Archivos .docx convertidos a .locked, mensaje exigiendo Bitcoin para
recuperar acceso')
;
```



En Usuario:

SQL

```
INSERT INTO usuario (nombre, apellido, email, contrasena, salt, es_admin,
es_activo) VALUES
('Juan', 'Pérez', 'juan.perez@example.com',
    '$2a$10$abcdefghijklmnopqrstuv',
    'abc123salt456',
    FALSE, TRUE),
('María', 'García', 'maria.garcia@example.com',
    '$2a$10$xyzabcdefghijklmnopqrst',
    'xyz789salt012',
    TRUE, TRUE),
('Carlos', 'Rodríguez', 'carlos.rodriguez@example.com',
    '$2a$10$mnopqrstuvwxyzabcdefgh',
    'mno345salt678',
    FALSE, TRUE),
('Ana', 'Martínez', 'ana.martinez@example.com',
    '$2a$10$ijklmnopqrstuvwxyzabcd',
    'ijk901salt234',
    TRUE, TRUE),
('Pedro', 'López', 'pedro.lopez@example.com',
```

```
    '$2a$10$efghijklmnopqrstuvwxyz',
    'efg567salt890',
    FALSE, FALSE);
```

```
mysql> select*from usuario;
+----+---------+----------+---------------------------+--------------------------------------------------------------+----------------------------
-----+----------+-----------+
| id | nombre  | apellido | email                     | contrasena                                                   | salt
     | es_admin | es_activo |
+----+---------+----------+---------------------------+--------------------------------------------------------------+----------------------------
-----+----------+-----------+
|  1 | Santiago | Nino     | santiago@tec.mx           | 12345678                                                     | misal
     |        0 |         1 |
|  2 | Jorge    | Niño     | juanHola@example          | ef92b778bafe771e89245b89ecbc08a44a4e166c06659911881f383d4473e94f | mysalt
     |        1 |         0 |
|  3 | Aaron    | Mercury  | aaron@example.mx          | c9d98026237c01762987ceb1be4e335e093cb6f350c3763e60c78d0d407f8824 | 2604e0cf44d42ed8b356b96681e5
a37d |        0 |         1 |
|  4 | Romina   | Niño     | romi@lasalle.mx           | 76132e1601cc31cb7f6a2a13410a678480167215c9cc8a7152d694a199a4982c | ee3d051bd156cfd763a7cc171c30
ba22 |        0 |         1 |
|  6 | Rominaa  | Ramirez  | romi@lasallee.mx          | ad7503386ce5b4aef4bc8c3e609a35e348438a26f4831023bb4a2cfa8f39b8dd | 3b285a60d232ac628f3d3a1439e2
e4c0 |        0 |         1 |
| 12 | Sara     | Gil      | sara@tec.mx               | 83453f13cd6a3dd1c7bbcf4e9c40cbe6eb6baab50b655f550aeedde48a2dd594 | 1895b5c7debc5fae4881c34714e6
8164 |        0 |         1 |
| 13 | Juan     | Pérez    | juan.perez@example.com    | $2a$10$abcdefghijklmnopqrstuv                                | abc123salt456
     |        0 |         1 |
| 14 | María    | García   | maria.garcia@example.com  | $2a$10$xyzabcdefghijklmnopqrst                               | xyz789salt012
     |        1 |         1 |
| 15 | Carlos   | Rodríguez | carlos.rodriguez@example.com | $2a$10$mnopqrstuvwxyzabcdefgh                             | mno345salt678
     |        0 |         1 |
| 16 | Ana      | Martínez | ana.martinez@example.com  | $2a$10$ijklmnopqrstuvwxyzabcd                                | ijk901salt234
     |        1 |         1 |
| 17 | Pedro    | López    | pedro.lopez@example.com   | $2a$10$efghijklmnopqrstuvwxyz                                | efg567salt890
     |        0 |         0 |
+----+---------+----------+---------------------------+--------------------------------------------------------------+----------------------------
-----+----------+-----------+
11 rows in set (0.01 sec)
```

En Evidencia:

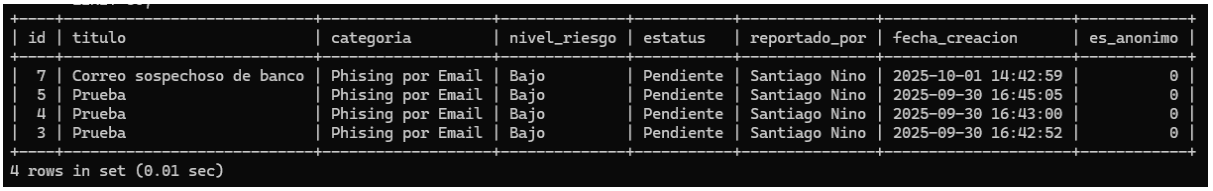
```
SQL
INSERT INTO evidencia (id_incidente, url) VALUES
(1, 'https://storage.example.com/evidencias/phishing_email_screenshot.png'),
(1, 'https://storage.example.com/evidencias/phishing_url_details.pdf'),
(3, 'https://storage.example.com/evidencias/ransomware_screen.jpg'),
(3, 'https://storage.example.com/evidencias/encrypted_files_list.txt'),
(4, 'https://storage.example.com/evidencias/credit_notification.pdf'),
(4, 'https://storage.example.com/evidencias/threatening_messages.png'),
(5, 'https://storage.example.com/evidencias/malware_task_manager.png');
```

```
mysql> select*from evidencia;
+----+--------------+-----------------------------------------------------------------+
| id | id_incidente | url                                                             |
+----+--------------+-----------------------------------------------------------------+
|  8 |            1 | https://storage.example.com/evidencias/phishing_email_screenshot.png |
|  9 |            1 | https://storage.example.com/evidencias/phishing_url_details.pdf      |
| 10 |            3 | https://storage.example.com/evidencias/ransomware_screen.jpg         |
| 11 |            3 | https://storage.example.com/evidencias/encrypted_files_list.txt      |
| 12 |            4 | https://storage.example.com/evidencias/credit_notification.pdf       |
| 13 |            4 | https://storage.example.com/evidencias/threatening_messages.png      |
| 14 |            5 | https://storage.example.com/evidencias/malware_task_manager.png      |
+----+--------------+-----------------------------------------------------------------+
7 rows in set (0.00 sec)
```

5.2. Resultados Esperados de Consultas Clave.

➔ SELECT con JOIN - Listar incidentes con información completa.

```sql
SELECT
    i.id,
    i.titulo,
    c.titulo AS categoria,
    r.descripcion AS nivel_riesgo,
    e.descripcion AS estatus,
    CONCAT(u.nombre, ' ', u.apellido) AS reportado_por,
    i.fecha_creacion,
    i.es_anonimo
FROM incidente i
INNER JOIN categoria c ON i.id_categoria = c.id
INNER JOIN riesgo r ON c.id_riesgo = r.id
INNER JOIN estatus e ON i.id_estatus = e.id
LEFT JOIN usuario u ON i.id_usuario = u.id
WHERE i.es_anonimo = FALSE
ORDER BY i.fecha_creacion DESC
LIMIT 50;
```



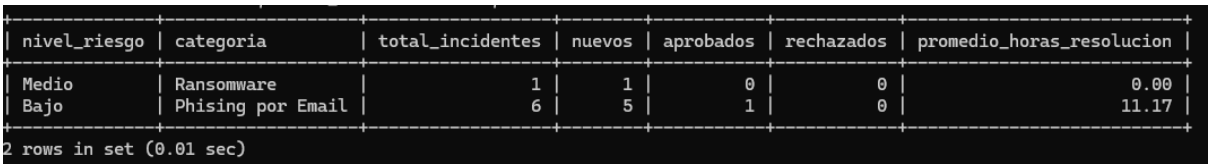
id	titulo	categoria	nivel_riesgo	estatus	reportado_por	fecha_creacion	es_anonimo
7	Correo sospechoso de banco	Phising por Email	Bajo	Pendiente	Santiago Nino	2025-10-01 14:42:59	0
5	Prueba	Phising por Email	Bajo	Pendiente	Santiago Nino	2025-09-30 16:45:05	0
4	Prueba	Phising por Email	Bajo	Pendiente	Santiago Nino	2025-09-30 16:43:00	0
3	Prueba	Phising por Email	Bajo	Pendiente	Santiago Nino	2025-09-30 16:42:52	0

4 rows in set (0.01 sec)

➔ Estadísticas de incidentes por categoría y riesgo.

```sql
SELECT
    r.descripcion AS nivel_riesgo,
    c.titulo AS categoria,
    COUNT(i.id) AS total_incidentes,
    SUM(CASE WHEN e.descripcion = 'Pendiente' THEN 1 ELSE 0 END) AS nuevos,
    SUM(CASE WHEN e.descripcion = 'Aprobado' THEN 1 ELSE 0 END) AS
aprobados,
    SUM(CASE WHEN e.descripcion = 'Rechazado' THEN 1 ELSE 0 END) AS
rechazados,
    ROUND(AVG(TIMESTAMPDIFF(HOUR, i.fecha_creacion, i.fecha_actualizacion)),
2) AS promedio_horas_resolucion
FROM riesgo r
```

```sql
INNER JOIN categoria c ON c.id_riesgo = r.id
LEFT JOIN incidente i ON i.id_categoria = c.id
LEFT JOIN estatus e ON i.id_estatus = e.id
WHERE i.fecha_creacion >= DATE_SUB(CURDATE(), INTERVAL 30 DAY)
GROUP BY r.id, c.id
ORDER BY r.id DESC, total_incidentes DESC;
```



nivel_riesgo	categoria	total_incidentes	nuevos	aprobados	rechazados	promedio_horas_resolucion
Medio	Ransomware	1	1	0	0	0.00
Bajo	Phising por Email	6	5	1	0	11.17

2 rows in set (0.01 sec)