UNIVERSIDAD NACIONAL DE INGENIERÍA
E-LEARNING UNI

TF_SQL Y FUNDAMENTOS DE BASE DE DATOS CON IA

Proyecto Final Asistido por IA y Buenas Prácticas

Universidad / Centro: Centro de Capacitación E-Learning (UNI / UNI E-learning)
Curso: SQL Y fundamentos de Base de datos con IA
Módulo: Proyecto Final Asistido por IA y Buenas Prácticas (Módulo 8)
Docente: Mag. Roberto Tello Yuen
Alumno: Arles Joshua Novoa Rojas

ETAPA 1: Presentación del caso, definición del problema y análisis:

1.1 Presentación del Caso - Sector Seguros

El presente proyecto se desarrolla en el sector asegurador, específicamente en la empresa Nova Insurance Company S.A., dedicada a la gestión de pólizas y siniestros.
Las empresas de seguros manejan grandes volúmenes de información relacionada con clientes, contratos de pólizas, coberturas contratadas, siniestros reportados y pagos realizados.

Actualmente, la información se encuentra dispersa en documentos físicos y archivos digitales no estructurados, lo que dificulta el control, análisis y toma de decisiones oportunas.

1.2 Definición del problema

Las compañías de seguros requieren administrar información crítica relacionada con clientes, pólizas contratadas, coberturas incluidas, siniestros reportados y pagos realizados.

Sin embargo, cuando esta información no está estructurada adecuadamente, pueden presentarse los siguientes problemas:

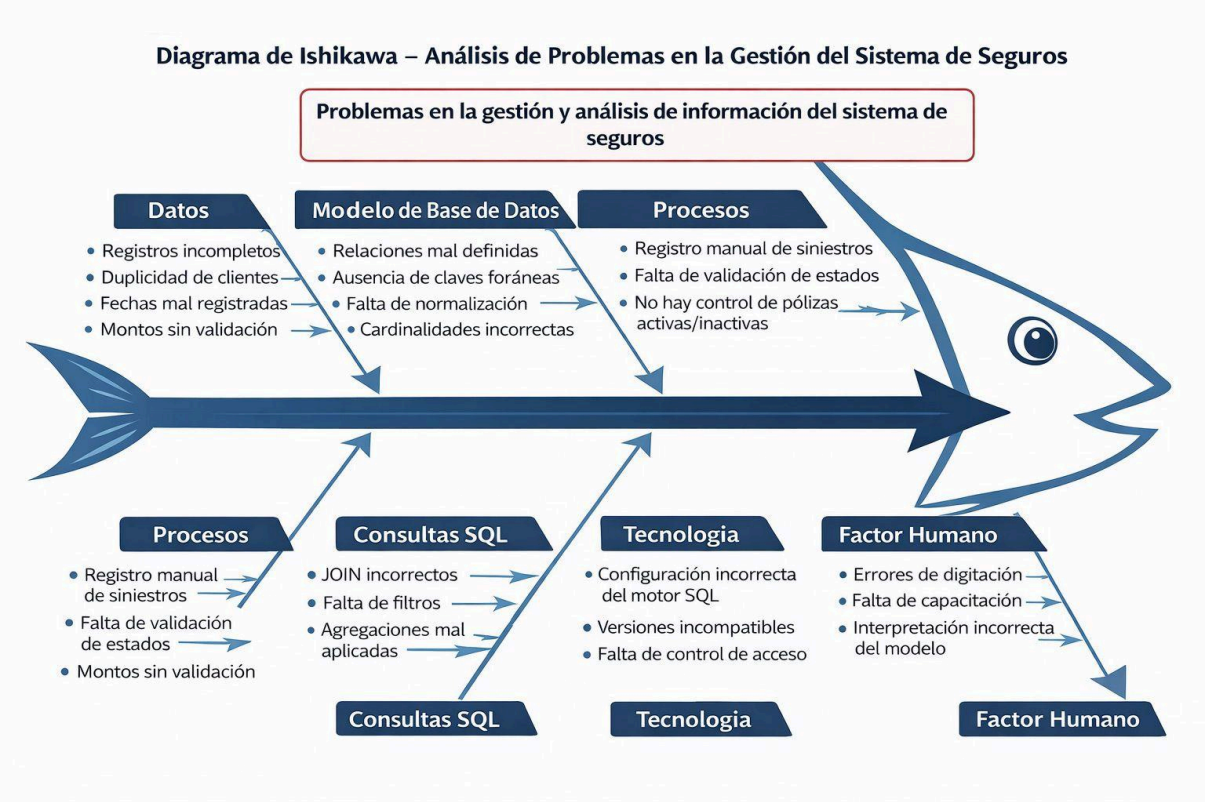
- Duplicidad de registros de clientes
- Dificultad para verificar vigencia de pólizas
- Falta de trazabilidad en siniestros
- Errores en el cálculo de pagos
- Retrasos en la toma de decisiones

El problema central identificado es la falta de un sistema estructurado que permita gestionar de manera integrada la información de pólizas y siniestros, garantizando integridad de datos y consultas confiables para análisis y toma de decisiones.

Por ello, se plantea diseñar un modelo entidad–relación que permita organizar y relacionar adecuadamente la información del negocio asegurador.

1.3 Análisis de Documentos Fuente

FIGURA 1: Diagrama de Ishikawa - Análisis de causas del problema en la gestión de seguros



Para comprender la estructura de información en el sector asegurador, se identificaron los siguientes documentos fuente que sirven como base para el diseño de la base de datos:

1. ***Contrato de Póliza***

Documento que contiene:

- Datos del cliente
- Número de póliza
- Tipo de seguro
- Coberturas incluidas
- Fecha de inicio y vencimiento
- Prima contratada

Este documento permite identificar las entidades CLIENTE, PÓLIZA y COBERTURA.

2. *Reporte de Siniestro*

Documento generado cuando el asegurado informa un evento cubierto por la póliza.
Contiene:

- Número de póliza
- Fecha del siniestro
- Tipo de siniestro
- Descripción del evento
- Monto reclamado
- Estado del proceso

Este documento permite identificar la entidad SINIESTRO y su relación con PÓLIZA.

3. *Registro de Pago*

Documento que respalda la indemnización realizada por la aseguradora.
Contiene:

- Número de siniestro
- Monto pagado
- Fecha de pago
- Medio de pago
- Estado del pago

Este documento permite identificar la entidad PAGO y su relación con SINIESTRO.

Análisis Técnico

Del análisis documental se concluye que la información presenta una estructura relacional clara, donde:

- Un cliente puede tener múltiples pólizas.
- Una póliza puede generar múltiples siniestros.
- Un siniestro puede generar uno o más pagos (según el caso).

Este análisis preliminar servirá como base para el diseño del modelo entidad–relación en la siguiente etapa.

1.4 Identificación Preliminar de Entidades

A partir del análisis del caso y los documentos fuente, se identifican las siguientes entidades preliminares para el modelo conceptual:

1. ***CLIENTE***

Representa a la persona natural o jurídica que contrata una póliza de seguro.

2. ***PÓLIZA***

Contrato formal entre el cliente y la aseguradora, que define coberturas, vigencia y prima.

3. ***COBERTURA***

Define los riesgos o eventos específicos que la póliza cubre.

4. ***SINIESTRO***

Evento reportado por el cliente que activa la cobertura de la póliza.

5. ***PAGO***

Registro de indemnización realizada por la aseguradora ante un siniestro aprobado.

Relación Preliminar Entre Entidades

- Un CLIENTE puede tener múltiples PÓLIZAS.
- Una PÓLIZA puede incluir múltiples COBERTURAS.
- Una PÓLIZA puede generar múltiples SINIESTROS.
- Un SINIESTRO puede generar uno o más pagos.

Estas entidades serán formalizadas y estructuradas en la Etapa 2 mediante el modelo ENTIDAD-RELACIÓN.

1.5 Análisis Causal del Problema - Enfoque Ishikawa

Con el objetivo de comprender de manera estructurada las causas del problema identificado en la gestión de pólizas y siniestros, se aplicó la metodología del Diagrama de Ishikawa (causa–efecto).

Problema central:

Deficiente gestión de información en la administración de pólizas y siniestros.

Se identificaron las siguientes categorías de causa:

1. Personas

- Errores en el registro manual de datos.
- Falta de capacitación en sistemas de gestión.
- Inconsistencia en la actualización de estados de siniestros.

2. Procesos

- Procedimientos no estandarizados para registrar siniestros.
- Ausencia de validación previa antes de aprobar pagos.

- Duplicidad en registros de clientes.

3. Tecnología

- Uso de archivos Excel aislados.
- Sistemas no integrados.
- Falta de trazabilidad automatizada.

4. Información

- Datos incompletos en pólizas.
- Inconsistencia en los montos reclamados.
- Ausencia de reportes consolidados.

5. Gestión

- Dificultad para analizar volumen de siniestros.
- Falta de indicadores para la toma de decisiones.
- Retrasos en aprobación de pagos.

Conclusión del Análisis

El análisis causal evidencia la necesidad de implementar una base de datos relacional estructurada que permita: **Integridad de datos, Relaciones claras entre entidades, Consultas confiables, Generación de reportes para toma de decisiones.**

Este análisis fundamenta el diseño del modelo entidad–relación presentado en la siguiente etapa.

ETAPA 2: Diseño del Modelo Entidad-Relación:

2.1 Definición de Entidades y Atributos

En esta etapa se realiza el diseño del modelo Entidad-Relación del sistema propuesto, identificando las entidades principales del negocio, sus atributos y las relaciones existentes entre ellas.

A continuación, se describen las entidades definidas:

1. ***Cliente***

Entidad que representa a la persona que contrata una póliza de seguro.

Atributos:

- id_cliente (PK)
- dni

- nombres
- apellidos
- teléfono
- correo
- dirección

2. ***Póliza***

Entidad que representa el contrato de seguro asociado a un cliente.

Atributos:

- id_poliza (PK)
- numero_poliza
- fecha_fin
- estado
- prima
- id_cliente (FK)

3. ***Cobertura***

Entidad que representa las coberturas incluidas dentro de una póliza.

Atributos:

- id_cobertura (PK)
- descripcion
- monto_maximo
- id_poliza (FK)

4. ***Siniestro***

Entidad que representa un evento reportado por el cliente.

Atributos:

- id_siniestro (PK)
- fecha_reporte
- descripcion
- monto_reclamado
- estado
- id_poliza (FK)

5. ***Pago***

Entidad que representa el pago realizado por un siniestro aprobado.

Atributos:

- id_pago (PK)
- fecha_pago

- monto_pagado
- metodo_pago
- estado_pago
- id_siniestro (FK)

2.2 <u>Definición de claves primarias</u>

Cada entidad definida cuenta con una clave primaria (PK), la cual permite identificar de manera única cada registro dentro de la base de datos.

Las claves primarias del modelo son las siguientes:

- Cliente → id_cliente
- Poliza → id_poliza
- Cobertura → id_cobertura
- Siniestro → id_siniestro
- Pago → id_pago

Estas claves garantizan unicidad e integridad dentro del sistema.

2.3 <u>Establecer relaciones (Cardinalidades)</u>

Las relaciones del modelo se definen en función de las reglas del negocio (seguros) y su cardinalidad:

- **Cliente (1) - (N) Poliza**
  Un cliente puede contratar varias pólizas, pero cada póliza pertenece a un solo cliente.
- **Poliza (1) - (N) Siniestro**
  Una póliza puede registrar varios siniestros a lo largo del tiempo, pero cada siniestro se asocia a una sola póliza.
- **Siniestro (1) - (N) Pago**
- Un siniestro puede generar uno o varios pagos (parciales o por etapas), pero cada pago corresponde a un solo siniestro.
- **Poliza (1) - (N) Cobertura**
  Una póliza puede incluir varias coberturas, y cada cobertura pertenece a una sola póliza.

2.4 Justificación del modelo

El modelo Entidad–Relación propuesto fue diseñado con el objetivo de representar de manera estructurada el proceso de gestión de pólizas y siniestros en Nova Insurance Company S.A.

La entidad Cliente constituye el punto de partida del sistema, ya que toda póliza debe estar asociada a un cliente registrado. Esto permite mantener trazabilidad y control sobre los contratos vigentes.

La entidad Poliza actúa como elemento central del modelo, ya que conecta al cliente con las coberturas contratadas y con los siniestros que puedan generarse durante su vigencia. Esta estructura permite analizar el historial completo de cada póliza.

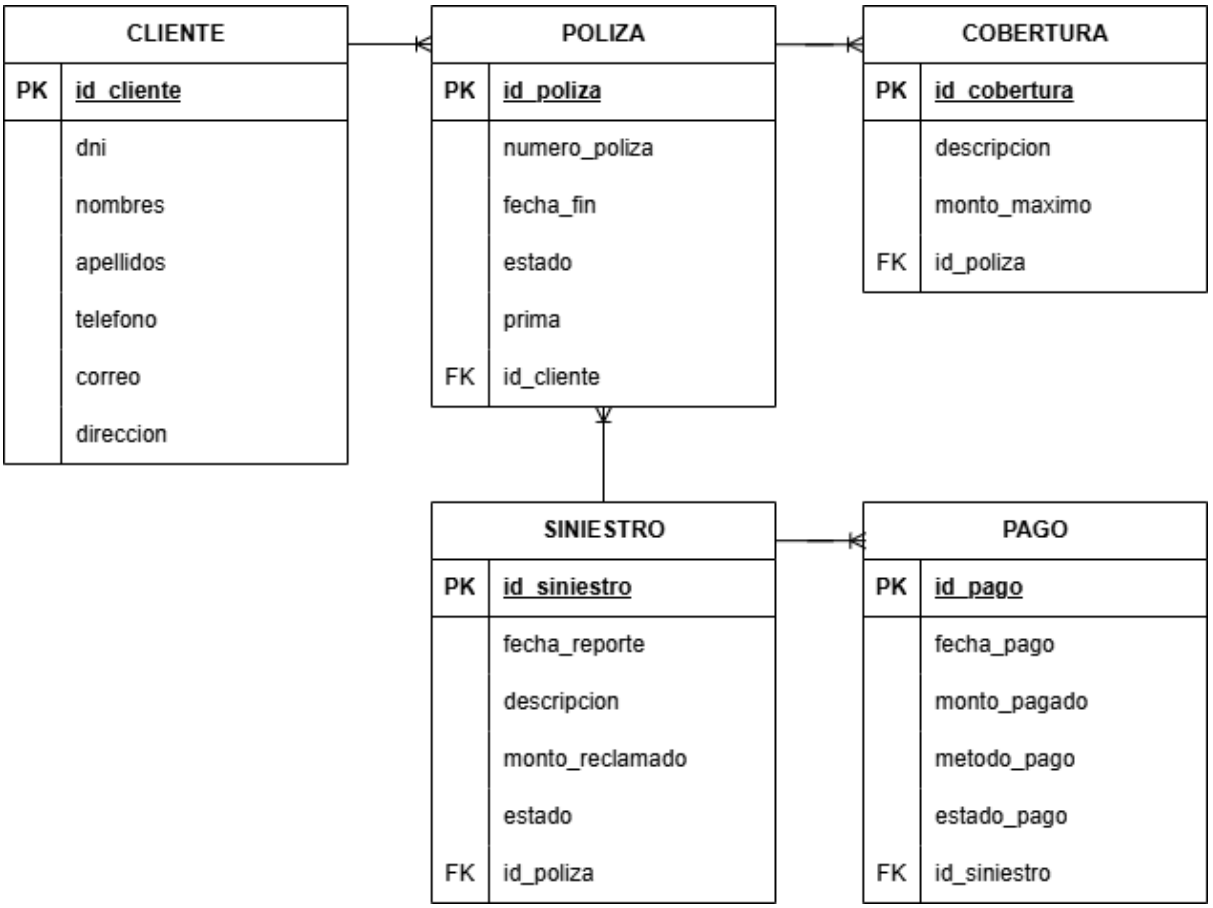
La entidad Cobertura se define de manera independiente para permitir que una póliza incluya múltiples coberturas, lo cual brinda flexibilidad y escalabilidad al sistema.

La entidad Siniestro permite registrar los eventos reportados por los clientes, asegurando control sobre los montos reclamados y el estado del proceso.

Finalmente, la entidad Pago permite registrar las indemnizaciones asociadas a los siniestros aprobados, garantizando trazabilidad financiera y control de desembolsos.

La estructura propuesta cumple con principios de integridad referencial, evita duplicidad de información y permite generar consultas para análisis y toma de decisiones.

FIGURA 2: Diagrama Entidad-Relación del Sistema de Gestión de Seguros



Metodología:
El diagrama Entidad–Relación fue modelado utilizando la herramienta diagrams.net (Draw.io), aplicando notación Crow's Foot para la representación de cardinalidades y claves primarias/foráneas.

ETAPA 3: Implementación del Modelo Relacional

3.1 Herramienta utilizada y justificación:

Para la implementación se utilizó PostgreSQL, debido a que es un motor de Base de Datos sólido y ampliamente utilizado en el ámbito académico como profesional.

Se eligió Postgre SQL porque permite trabajar correctamente con claves primarias y foráneas, lo cuál era fundamental para asegurar que las relaciones entre cliente, póliza, siniestro, y pago se mantengan consistentes. Además, maneja de forma adecuada tipos de datos como fechas (DATE) y montos decimales (DECIMAL), necesarios para el caso del sistema de seguros.

Otro motivo de elección fue que PostgreSQL cumple con los estándares SQL y facilita la validación del modelo mediante consultas SELECT, JOIN y funciones de agregación como SUM y GROUP BY.

Finalmente, se utilizó el entorno DB Fiddle con motor PostgreSQL 17 para ejecutar y comprobar todo el script (creación de tablas, inserción de datos y consultas), verificando que el modelo funcione correctamente.

- ***BATCH 1*:** ESTRUCTURA (DDL)

```sql
CREATE TABLE cliente (
    id_cliente SERIAL PRIMARY KEY,
    dni VARCHAR(15) NOT NULL,
    nombres VARCHAR(100),
    apellidos VARCHAR(100),
    telefono VARCHAR(15),
    correo VARCHAR(100),
    direccion VARCHAR(150)
);

CREATE TABLE poliza (
    id_poliza SERIAL PRIMARY KEY,
    numero_poliza VARCHAR(20),
    fecha_fin DATE,
    estado VARCHAR(20),
    prima DECIMAL(10,2),
    id_cliente INT,
    FOREIGN KEY (id_cliente) REFERENCES cliente(id_cliente)
);

CREATE TABLE cobertura (
    id_cobertura SERIAL PRIMARY KEY,
    descripcion VARCHAR(150),
    monto_maximo DECIMAL(10,2),
    id_poliza INT,
    FOREIGN KEY (id_poliza) REFERENCES poliza(id_poliza)
);

CREATE TABLE siniestro (
    id_siniestro SERIAL PRIMARY KEY,
    fecha_reporte DATE,
    descripcion VARCHAR(150),
    monto_reclamado DECIMAL(10,2),
    estado VARCHAR(20),
    id_poliza INT,
    FOREIGN KEY (id_poliza) REFERENCES poliza(id_poliza)
);

CREATE TABLE pago (
    id_pago SERIAL PRIMARY KEY,
    fecha_pago DATE,
    monto_pagado DECIMAL(10,2),
    metodo_pago VARCHAR(50),
    estado_pago VARCHAR(20),
    id_siniestro INT,
    FOREIGN KEY (id_siniestro) REFERENCES siniestro(id_siniestro)
);
```

- ***BATCH 2*:** INSERTS (DATOS)

```sql
INSERT INTO cliente (dni, nombres, apellidos, telefono, correo, direccion)
VALUES
('12345678','Carlos','Ramirez Soto','987654321','carlos@email.com','Av. Lima 123'),
('87654321','Maria','Gonzales Perez','912345678','maria@email.com','Jr. Arequipa 456');

INSERT INTO poliza (numero_poliza, fecha_fin, estado, prima, id_cliente)
VALUES
('POL-001','2026-12-31','Activa',1200.00,1),
('POL-002','2026-06-30','Activa',950.00,2);

INSERT INTO cobertura (descripcion, monto_maximo, id_poliza)
VALUES
('Cobertura contra incendios',50000.00,1),
('Cobertura contra robos',30000.00,1),
('Cobertura vehicular total',80000.00,2);

INSERT INTO siniestro (fecha_reporte, descripcion, monto_reclamado, estado, id_poliza)
VALUES
('2026-02-10','Incendio parcial en vivienda',20000.00,'En evaluacion',1),
('2026-03-05','Choque vehicular leve',15000.00,'Aprobado',2);

INSERT INTO pago (fecha_pago, monto_pagado, metodo_pago, estado_pago, id_siniestro)
VALUES
('2026-02-20',18000.00,'Transferencia','Completado',1),
('2026-03-10',15000.00,'Deposito bancario','Completado',2);
```

- ***BATCH 3:*** VALIDACIÓN

Se ejecutaron consultas SELECT para verificar la correcta inserción de los datos en cada tabla del modelo relacional.

```sql
SELECT * FROM cliente;
SELECT * FROM poliza;
SELECT * FROM cobertura;
SELECT * FROM siniestro;
SELECT * FROM pago;
```

Resultado de ejecución:

db<>fiddle Postgres 17 run markdown donate feedback about

```
INSERT INTO poliza (numero_poliza, fecha_fin, estado, prima, id_cliente)
VALUES
('POL-001','2026-12-31','Activa',1200.00,1),
('POL-002','2026-06-30','Activa',950.00,2);

INSERT INTO cobertura (descripcion, monto_maximo, id_poliza)
VALUES
('Cobertura contra incendios',50000.00,1),
('Cobertura contra robos',30000.00,1),
('Cobertura vehicular total',80000.00,2);

INSERT INTO siniestro (fecha_reporte, descripcion, monto_reclamado, estado, id_poliza)
VALUES
('2026-02-10','Incendio parcial en vivienda',20000.00,'En evaluacion',1),
('2026-03-05','Choque vehicular leve',15000.00,'Aprobado',2);

INSERT INTO pago (fecha_pago, monto_pagado, metodo_pago, estado_pago, id_siniestro)
VALUES
('2026-02-20',18000.00,'Transferencia','Completado',1),
('2026-03-10',15000.00,'Deposito bancario','Completado',2);
```

INSERT 0 2

```
SELECT * FROM cliente;
SELECT * FROM poliza;
SELECT * FROM cobertura;
SELECT * FROM siniestro;
SELECT * FROM pago;
```

id_cliente	dni	nombres	apellidos	telefono	correo	direccion
1	12345678	Carlos	Ramirez Soto	987654321	carlos@email.com	Av. Lima 123
2	87654321	Maria	Gonzales Perez	912345678	maria@email.com	Jr. Arequipa 456

SELECT 2

id_poliza	numero_poliza	fecha_fin	estado	prima	id_cliente
1	POL-001	2026-12-31	Activa	1200.00	1
2	POL-002	2026-06-30	Activa	950.00	2

SELECT 2

id_cobertura	descripcion	monto_maximo	id_poliza
1	Cobertura contra incendios	50000.00	1
2	Cobertura contra robos	30000.00	1
3	Cobertura vehicular total	80000.00	2

SELECT 3

id_siniestro	fecha_reporte	descripcion	monto_reclamado	estado	id_poliza
1	2026-02-10	Incendio parcial en vivienda	20000.00	En evaluacion	1
2	2026-03-05	Choque vehicular leve	15000.00	Aprobado	2

SELECT 2

id_pago	fecha_pago	monto_pagado	metodo_pago	estado_pago	id_siniestro
1	2026-02-20	18000.00	Transferencia	Completado	1
2	2026-03-10	15000.00	Deposito bancario	Completado	2

SELECT 2

ETAPA 4: Gestión de la Base de Datos y toma de decisiones

Consultas SQL y Análisis:

- ***BATCH 4:*** Clientes con pólizas

```sql
SELECT c.nombres, c.apellidos, p.numero_poliza, p.prima
FROM cliente c
JOIN poliza p ON c.id_cliente = p.id_cliente;
```

nombres	apellidos	numero_poliza	prima
Carlos	Ramirez Soto	POL-001	1200.00
Maria	Gonzales Perez	POL-002	950.00

SELECT 2

- ***BATCH 5:*** Pólizas con coberturas

```sql
SELECT p.numero_poliza, co.descripcion, co.monto_maximo
FROM poliza p
JOIN cobertura co ON p.id_poliza = co.id_poliza;
```

numero_poliza	descripcion	monto_maximo
POL-001	Cobertura contra incendios	50000.00
POL-001	Cobertura contra robos	30000.00
POL-002	Cobertura vehicular total	80000.00

SELECT 3

- ***BATCH 6:*** Siniestros con cliente

```sql
SELECT c.nombres, s.descripcion, s.monto_reclamado, s.estado
FROM siniestro s
JOIN poliza p ON s.id_poliza = p.id_poliza
JOIN cliente c ON p.id_cliente = c.id_cliente;
```

nombres	descripcion	monto_reclamado	estado
Carlos	Incendio parcial en vivienda	20000.00	En evaluacion
Maria	Choque vehicular leve	15000.00	Aprobado

SELECT 2

- ***BATCH 7:*** Total reclamado por cliente

```sql
SELECT c.nombres, SUM(s.monto_reclamado) AS total_reclamado
FROM cliente c
JOIN poliza p ON c.id_cliente = p.id_cliente
JOIN siniestro s ON p.id_poliza = s.id_poliza
GROUP BY c.nombres;
```

nombres	total_reclamado
Carlos	20000.00
Maria	15000.00

SELECT 2

REGISTRO DE USO DE INTELIGENCIA ARTIFICIAL

Herramienta utilizada: Chat GPT - Modelo de Lenguaje para asistencia técnica en base de datos

Prompts utilizados y aplicación en el proyecto

Prompt 1 - Modelado conceptual:

Diseña un modelo entidad-relación para un sistema de seguros que incluya las entidades cliente, póliza, cobertura, siniestro y pago, indicando claves primarias, foráneas y cardinalidades.

- ***Aplicación en el proyecto:***

Se utilizó como referencia inicial para estructurar el modelo conceptual y verificar la coherencia de las relaciones 1:N entre cliente-póliza y póliza-cobertura.

- ***Decisión propia:***

Se ajustaron manualmente las cardinalidades y atributos para asegurar consistencia con el caso planteado.

Prompt 2 - Diseño del modelo relacional en Postgre SQL:

Genera el script SQL en Postgre SQL para crear las tablas del sistema de seguros con claves primarias, claves foráneas e integridad referencial.

- ***Aplicación en el proyecto:***

Sirvió como guía para estructurar el modelo físico.

- ***Decisión propia:***

Se definieron tipos de datos adecuados (Decimal para montos, Date para fechas) y se validó la integridad referencial mediante FOREIGN KEY.

Prompt 3 - Validación técnica del modelo:

Revisa si las relaciones entre las tablas y las claves foráneas cumplen con principios de normalización e integridad referencial.

- ***Aplicación en el proyecto:***

Se utilizó para verificar consistencia estructural del esquema antes de la ejecución de pruebas.

- ***Decisión propia:***

Se corrigieron detalles menores en nombres de atributos y relaciones para mantener coherencia con el modelo ER presentado.

<u>Prompt 4 - Consultas avanzadas y análisis:</u>

Genera consultas SQL utilizando JOIN y funciones de agregación (SUM, GROUP BY) para analizar el total reclamado por cliente y listar con sus coberturas.

- ***Aplicación en el proyecto:***

Se empleó como base para estructurar consultas de análisis.

- ***Decisión propia:***

Se implementaron alias de tablas, agregaciones y filtros alineados al modelo desarrollado, asegurando que las consultas reflejen correctamente las relaciones definidas.