

Proyecto Bases de Datos

Raya Pérez José Luis

Muñoz Sandoval Alan Sebastián

Cruz Martinez Giovanni

Diciembre 2023

Base de Datos para tienda de Comics

1. Requerimientos

1. **Gestión de Inventario de Cómic:** Manejo de cómics disponibles, incluyendo detalles como título, género, autor, año de publicación y formato.
2. **Registro de Clientes:** Información de los clientes, incluyendo nombre, dirección y historial de compras.
3. **Transacciones (Compras):** Procesamiento de ventas de cómics, incluyendo gestión de precios, fechas de venta y empleados que realizan la venta.
4. **Gestión de Empleados:** Información del personal, incluyendo turnos y detalles de contacto.

2. Modelo Conceptual

2.1. Entidades y Atributos

■ Comic

- IDcomic: Identificador único (Llave primaria).
- Personaje: Nombre del personaje principal.
- NUmero: Numero del cómic.
- Autor: Autor del cómic.
- Precio: Precio de venta del cómic.
- Año: Año en que se publicó el cómic.
- Cantidad: Cantidad disponible en la tienda.

■ Cliente

- IDcliente: Identificador único (Llave primaria).
- Nombre: Nombre del cliente.
- Apellido: Apellido del cliente.
- Teléfono: Número de teléfono.

■ Compra

- IDcompra: Identificador único (Llave primaria).
- Fecha: Fecha de la venta.
- Total: Monto total de la venta.
- IDcliente: Identificador del cliente (Llave foránea).
- IDempleado: Identificador del empleado (Llave foránea).

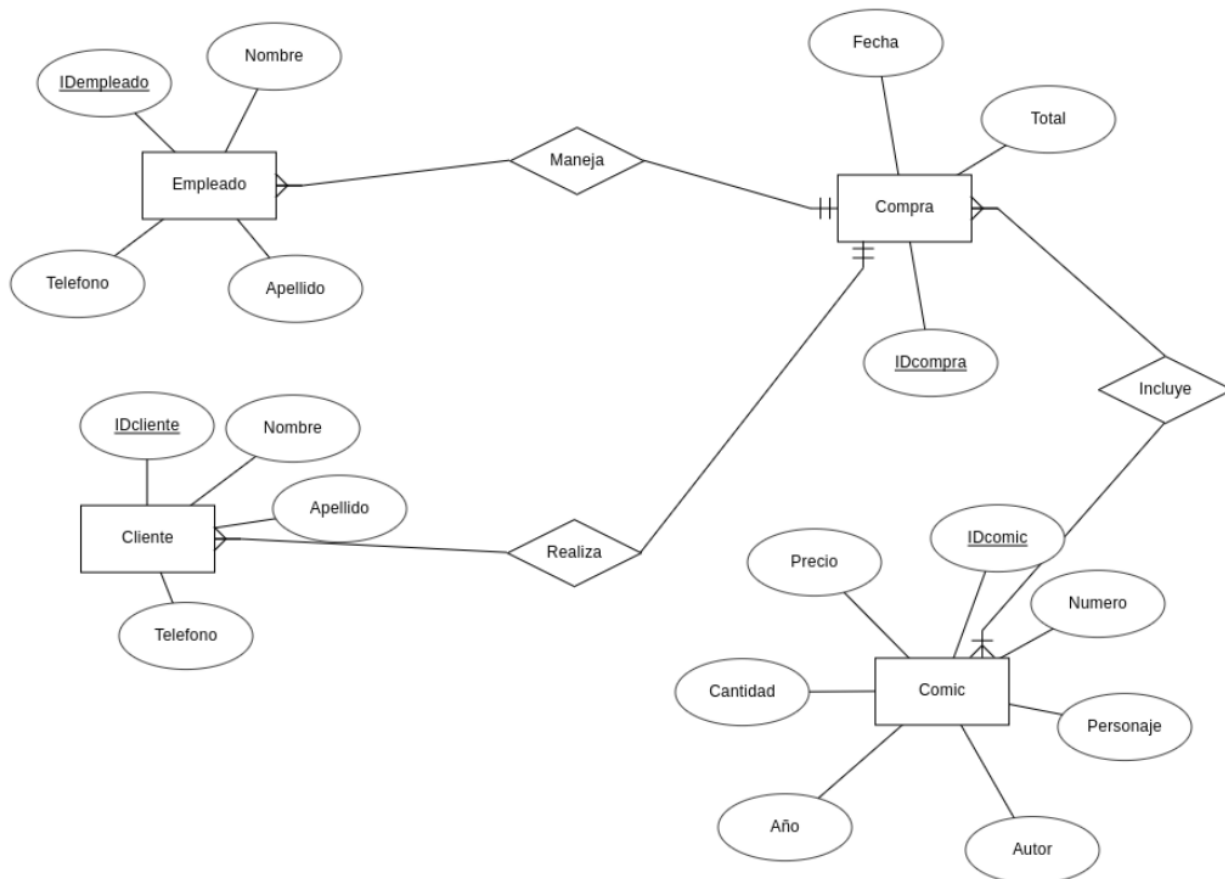
■ Empleado

- IDempleado: Identificador único (Llave primaria).
- Nombre: Nombre del empleado.
- Apellido: Apellido del empleado.
- Teléfono: Número de teléfono.

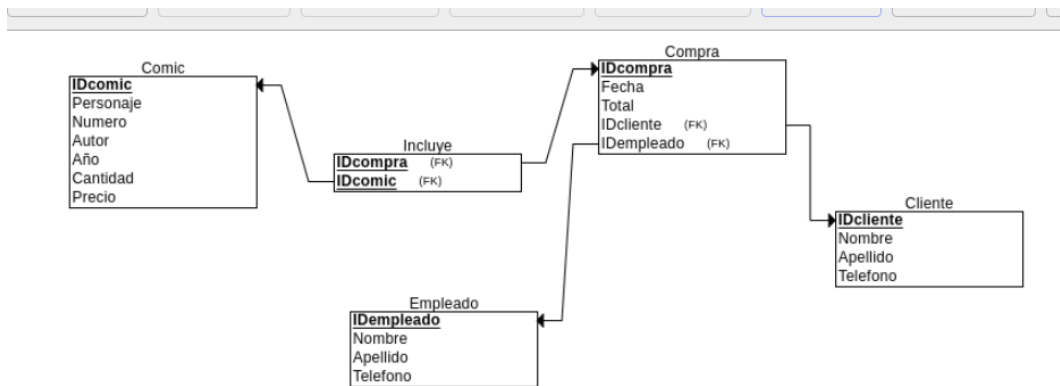
2.1.1. Relaciones

- **Realiza (Cliente - Compra):** Un cliente puede realizar múltiples compras.
- **Incluye (Compra - Comic):** Una compra puede incluir varios cómics.
- **Maneja (Empleado - Compra):** Un empleado es responsable de gestionar las compras.

3. Modelo E/R



4. Modelo Relacional



5. Script para la generación de tablas

```
CREATE TABLE Cliente
```

```
(
  IDcliente SERIAL PRIMARY KEY,
  Nombre VARCHAR(50) NOT NULL,
  Apellido VARCHAR(50) NOT NULL,
  Telefono VARCHAR(15) NOT NULL
);
```

```
CREATE TABLE Empleado
```

```
(
  IDempleado SERIAL PRIMARY KEY,
  Nombre VARCHAR(50) NOT NULL,
  Apellido VARCHAR(50) NOT NULL,
  Telefono VARCHAR(15) NOT NULL
);
```

```
CREATE TABLE Comic
```

```
(
  IDcomic SERIAL PRIMARY KEY,
  Personaje VARCHAR(50) NOT NULL,
  Numero INT NOT NULL,
  Autor VARCHAR(50) NOT NULL,
  Año INT NOT NULL,
  Cantidad INT NOT NULL,
  Precio DECIMAL NOT NULL
);
```

```
CREATE TABLE Compra
```

```
(
  IDcompra SERIAL PRIMARY KEY,
  Fecha DATE NOT NULL,
  Total DECIMAL NOT NULL,
  IDcliente INT,
  IDempleado INT,
  FOREIGN KEY (IDcliente) REFERENCES Cliente(IDcliente) ON DELETE SET NULL,
  FOREIGN KEY (IDempleado) REFERENCES Empleado(IDempleado) ON DELETE SET NULL ON UPDATE CASCADE
);
```

```

CREATE TABLE Incluye
(
    IDcompra INT NOT NULL,
    IDcomic INT NOT NULL,
    PRIMARY KEY (IDcompra, IDcomic),
    FOREIGN KEY (IDcompra) REFERENCES Compra(IDcompra) ON DELETE CASCADE ON UPDATE CASCADE,
    FOREIGN KEY (IDcomic) REFERENCES Comic(IDcomic) ON DELETE CASCADE ON UPDATE CASCADE
);

-- Check
ALTER TABLE Comic
ADD CONSTRAINT check_cantidad_nonnegative
CHECK (Cantidad >= 0);

ALTER TABLE Compra
ADD CONSTRAINT check_total_nonnegative
CHECK (Total >= 0);

ALTER TABLE Comic
ADD CONSTRAINT check_año_valido
CHECK (Año >= 1900 AND Año <= EXTRACT(YEAR FROM CURRENT_DATE));

-- Dominios
CREATE DOMAIN precio_comic AS DECIMAL
CHECK (VALUE >= 0 AND VALUE <= 500);

CREATE DOMAIN telefono_standard AS VARCHAR(15)
CHECK (VALUE ~ '[0-9]{10}$');

CREATE DOMAIN año_comic AS INT
CHECK (VALUE >= 1900 AND VALUE <= EXTRACT(YEAR FROM CURRENT_DATE));

-- Tuplas
ALTER TABLE Compra
ADD CONSTRAINT check_fecha_total_compra
CHECK (NOT (Total > 500 AND EXTRACT(YEAR FROM Fecha) < 2000));

ALTER TABLE Comic
ADD CONSTRAINT check_precio_cantidad
CHECK (NOT (Precio = 0 AND Cantidad = 0));

```

Notas:

Para el ID de las tablas, se usó SERIAL para que automáticamente se generaran el id uno por uno. Se crean solo 20 empleados, pues no es lógico crear más de 100 empleados para administrar el negocio con 100 clientes.

6. Restricciones de integridad referencial

1. Ejemplo 1

Tablas involucradas en la restricción:

- Incluye
- Compra

FK de la tabla que referencia y PK de la tabla referenciada:

- FK: Incluye.IDcompra

- PK: Compra.IDcompra

Justificación del trigger de integridad referencial elegido:

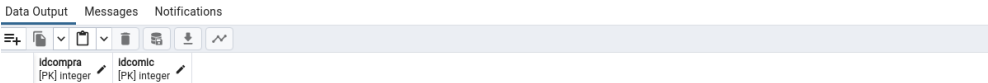
- Elegimos el trigger *CASCADE* para la llave foránea porque si un comic es eliminado, es lógico eliminar todas las referencias de ese comic en la tabla 'Incluye', ya que el objeto ya no existe y no puede ser parte de ninguna compra.

Instrucción DELETE para evidenciar que la restricción está funcionando:

```

1  -- Insertamos el ejemplo
2  insert into Comic (IDcomic, Personaje, Numero, Autor, Año, Cantidad, Precio) values (110, 'Ejemplo', 42, 'Stan Lee', 1964, 10, '5.80');
3
4  -- Como el ID obtenido es 105, insertamos una entrada en Incluye
5  INSERT INTO Incluye (IDcompra, IDcomic) VALUES (57, 100);
6
7  -- Ahora eliminamos el comic
8  DELETE FROM Compra WHERE IDcompra = 110;
9
10 -- Verificamos si las entradas en Incluye asociadas han sido eliminadas
11 SELECT * FROM Incluye WHERE IDcomic = 110;
12

```



Vemos que no se nos devuelve ningún dato, indicando que la restricción funciona

2. Ejemplo 2

Tablas involucradas en la restricción:

- Incluye
- Compra

FK de la tabla que referencia y PK de la tabla referenciada:

- FK: Incluye.IDcompra
- PK: Compra.IDcompra

Justificación del trigger de integridad referencial elegido:

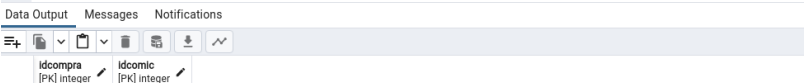
- Elegimos el trigger *CASCADE* para la llave foránea porque si una compra es eliminada, todas las referencias a esa operación en la tabla 'Incluye' también deben eliminarse automáticamente. Esto asegura que no tengamos registros huérfanos en 'Incluye' que hagan referencia a compras inexistentes.

Instrucción DELETE para evidenciar que la restricción está funcionando:

```

1  -- Primero insertamos una compra y un comic de prueba
2  insert into Compra (IDcompra, Fecha, Total, IDcliente, IDempleado) values (110, '2023-02-02', '34.16', 27, 4);
3
4  -- Como el ID obtenido es 110, insertamos una entrada en Incluye
5  INSERT INTO Incluye (IDcompra, IDcomic) VALUES (110, 56);
6
7  -- Ahora eliminamos la compra
8  DELETE FROM Compra WHERE IDcompra = 110;
9
10 -- Verificamos si la entrada en Incluye asociada ha sido eliminada
11 SELECT * FROM Incluye WHERE IDcompra = 110;
12

```



Vemos que no se nos devuelve ningún dato, indicando que la restricción funciona

3. Ejemplo 3

Tablas involucradas en la restricción:

- Compra

- Cliente

FK de la tabla que referencia y PK de la tabla referenciada:

- FK: Compra.IDcliente
- PK: Cliente.IDcliente

Justificación del trigger de integridad referencial elegido:

- Elegimos el trigger *SET NULL* para la llave foránea porque si un cliente es eliminado, no queremos perder los registros de las compras que hizo, pero debemos indicar que el cliente ya no existe en nuestra base de datos. Por lo tanto, establecemos el valor de IDcliente en las compras relacionados como NULL.

Instrucción DELETE para evidenciar que la restricción está funcionando:

```

1  -- Primero insertamos un cliente de prueba
2  insert into Cliente (IDcliente, Nombre, Apellido, Telefono) values (111,'Doralynn', 'Coneybeer', '4242714944');
3
4  -- Como el ID obtenido es 11, insertamos una compra asociada a este cliente
5  insert into Compra (Fecha, Total, IDcliente, IDempleado) values ('2023-02-02', '34.16', 111, 4);
6
7  -- Ahora eliminamos el cliente
8  DELETE FROM cliente WHERE IDcliente = 111;
9
10 -- Verificamos si el IDcliente en la compra asociada se ha establecido como NULL
11 SELECT * FROM Compra WHERE IDcliente IS NULL;
12

```

Data Output Messages Notifications						
	idcompra [PK] integer	fecha date	total numeric	idcliente integer	idempleado integer	
1	101	2023-02-02	34.16	[null]	4	

Vemos que aparece el cliente como null, indicando que la restricción funciona

4. Ejemplo 4

Tablas involucradas en la restricción:

- Compra
- Empleado

FK de la tabla que referencia y PK de la tabla referenciada:

- FK: Compra.IDempleado
- PK: Empleado.IDempleado

Justificación del trigger de integridad referencial elegido:

- Elegimos el trigger *SET NULL* para la llave foránea porque si un empleado es eliminado, no queremos perder los registros de las compras que gestionó, pero necesitamos reflejar que el empleado ya no está en la empresa. Por lo tanto, establecemos el valor de IDempleado en las compras relacionadas como NULL.

Instrucción DELETE para evidenciar que la restricción está funcionando:

```

1  -- Primero insertamos un empleado de prueba
2  insert into Empleado (IDempleado,Nombre, Apellido, Telefono) values (111,'Cthrine', 'Randals', '5937788725');
3
4  -- Suponiendo que el ID obtenido es 103, insertamos un ticket asociado a este empleado
5  insert into Compra (Fecha, Total, IDcliente, IDempleado) values ('2023-02-02', '34.16', 27, 111);
6
7  -- Ahora eliminamos el empleado
8  DELETE FROM Empleado WHERE IDempleado = 111;
9
10 -- Verificamos si el IDempleado en la compra asociada se ha establecido como NULL
11 SELECT * FROM Compra WHERE IDempleado IS NULL;
12

```

Data Output Messages Notifications						
	idcompra [PK] integer	fecha date	total numeric	idcliente integer	idempleado integer	
1	103	2023-02-02	34.16	27	[null]	

Vemos que aparece el empleado como null, indicando que la restricción funciona

7. Restricciones Check

1. Restricción 1

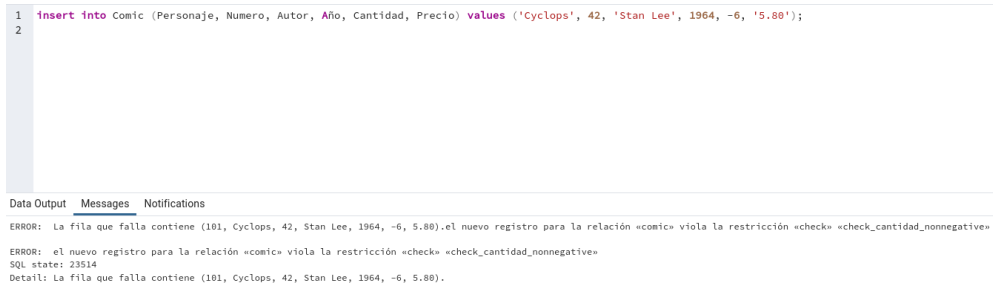
Tabla elegida: Comic

Atributo elegido: Cantidad

Breve descripción de la restricción: La cantidad de cómics en inventario debe ser mayor o igual a cero.

Instrucción para la creación de la restricción:

```
ALTER TABLE Comic
ADD CONSTRAINT check_cantidad_nonnegative
CHECK (Cantidad >= 0);
```



```
1 insert into Comic (Personaje, Numero, Autor, Año, Cantidad, Precio) values ('Cyclops', 42, 'Stan Lee', 1964, -6, '5.80');
2
```

Data Output Messages Notifications

ERROR: La fila que falla contiene (101, Cyclops, 42, Stan Lee, 1964, -6, 5.80).el nuevo registro para la relación «comic» viola la restricción «check» «check_cantidad_nonnegative»

ERROR: el nuevo registro para la relación «comic» viola la restricción «check» «check_cantidad_nonnegative»

SQL state: 23514

Detail: La fila que falla contiene (101, Cyclops, 42, Stan Lee, 1964, -6, 5.80).

2. Restricción 2

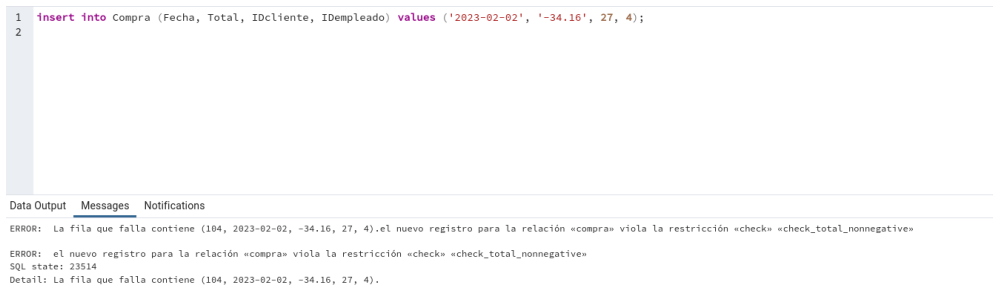
Tabla elegida: Compra

Atributo elegido: Total

Breve descripción de la restricción: El total de la compra debe ser mayor o igual a cero.

Instrucción para la creación de la restricción:

```
ALTER TABLE Compra
ADD CONSTRAINT check_total_nonnegative
CHECK (Total >= 0);
```



```
1 insert into Compra (Fecha, Total, IDcliente, IDempleado) values ('2023-02-02', '-34.16', 27, 4);
2
```

Data Output Messages Notifications

ERROR: La fila que falla contiene (104, 2023-02-02, -34.16, 27, 4).el nuevo registro para la relación «compra» viola la restricción «check» «check_total_nonnegative»

ERROR: el nuevo registro para la relación «compra» viola la restricción «check» «check_total_nonnegative»

SQL state: 23514

Detail: La fila que falla contiene (104, 2023-02-02, -34.16, 27, 4).

3. Restricción 3

Tabla elegida: Comic

Atributo elegido: Año

Breve descripción de la restricción: El año de publicación del cómic debe estar entre 1900 y el año actual.

Instrucción para la creación de la restricción:

```
ALTER TABLE Comic
ADD CONSTRAINT check_año_valido
CHECK (Año >= 1900 AND Año <= EXTRACT(YEAR FROM CURRENT_DATE));
```

```
1 insert into Comic (Personaje, Numero, Autor, Año, Cantidad, Precio) values ('Cyclops', 42, 'Stan Lee', 2024, 6, '5.80');
2
```

Data Output Messages Notifications

ERROR: La fila que falla contiene (102, Cyclops, 42, Stan Lee, 2024, 6, 5.80).el nuevo registro para la relación «comic» viola la restricción «check» «check_año_valido»

ERROR: el nuevo registro para la relación «comic» viola la restricción «check» «check_año_valido»

SQL state: 23514

Detail: La fila que falla contiene (102, Cyclops, 42, Stan Lee, 2024, 6, 5.80).

8. Dominios Personalizados

1. Dominio 1

Tabla elegida: Comic

Atributo elegido: Precio

Breve descripción del dominio y de la restricción check propuesta: El dominio para 'Precio' asegura que el precio sea no negativo y no exceda un máximo establecido, por ejemplo, 500.

Instrucción para la creación del dominio personalizado:

```
CREATE DOMAIN precio_comic AS DECIMAL
CHECK (VALUE >= 0 AND VALUE <= 500);
```

```
1 -- Ejemplo válido para Comic
2 INSERT INTO Comic (Personaje, Numero, Autor, Año, Cantidad, Precio)
3 VALUES ('Batman', 1, 'Bob Kane', 1940, 10, 250.00);
4
5 -- Ejemplo inválido para Comic (Precio fuera de rango)
6 INSERT INTO Comic (Personaje, Numero, Autor, Año, Cantidad, Precio)
7 VALUES ('Batman', 2, 'Bob Kane', 1940, 10, 600.00);
8
9
10
11
12
```

Data Output Messages Notifications

ERROR: el valor para el dominio precio_comic viola la restricción «check» «precio_comic_check»

SQL state: 23514

2. Dominio 2

Tabla elegida: Cliente, Empleado

Atributo elegido: Telefono

Breve descripción del dominio y de la restricción check propuesta: El dominio para 'Telefono' asegura que el número tenga exactamente 10 dígitos.

Instrucción para la creación del dominio personalizado:

```
CREATE DOMAIN telefono_standard AS VARCHAR(15)
CHECK (VALUE ~ '^[0-9]{10}$');
```

```
1
2 -- Ejemplo válido para Cliente
3 INSERT INTO Cliente (Nombre, Apellido, Telefono)
4 VALUES ('Juan', 'Pérez', '1234567890');
5
6 -- Ejemplo inválido para Cliente (Teléfono no tiene 10 dígitos)
7 INSERT INTO Cliente (Nombre, Apellido, Telefono)
8 VALUES ('Juan', 'Pérez', '123');
```

Data Output Messages Notifications

ERROR: el valor para el dominio telefono_standard viola la restricción «check» «telefono_standard_check»

SQL state: 23514

3. Dominio 3

Tabla elegida: Comic

Atributo elegido: Año

Breve descripción del dominio y de la restricción check propuesta: El dominio para 'Año' asegura que el año esté entre 1900 y el año actual.

Instrucción para la creación del dominio personalizado:

```
CREATE DOMAIN año_comic AS INT
CHECK (VALUE >= 1900 AND VALUE <= EXTRACT(YEAR FROM CURRENT_DATE));
```

```
1
2 -- Ejemplo válido para Comic (Año dentro del rango)
3 INSERT INTO Comic (Personaje, Numero, Autor, Año, Cantidad, Precio)
4 VALUES ('Superman', 1, 'Jerry Siegel', 1950, 20, 300.00);
5
6 -- Ejemplo inválido para Comic (Año fuera de rango)
7 INSERT INTO Comic (Personaje, Numero, Autor, Año, Cantidad, Precio)
8 VALUES ('Superman', 2, 'Jerry Siegel', 1800, 20, 300.00);
```

Data Output Messages Notifications

ERROR: el valor para el dominio "año_comic" viola la restricción «check» «año_comic_check»

SQL state: 23514

9. Restricciones para Tuplas

1. Restricción 1

Tabla elegida: Compra

Breve descripción de la restricción: Las compras con un total mayor a 500 no pueden ser anteriores al año 2000.

Instrucción para la creación de la restricción:

```
ALTER TABLE Compra
ADD CONSTRAINT check_fecha_total_compra
CHECK (NOT (Total > 500 AND EXTRACT(YEAR FROM Fecha) < 2000));
```

Instrucción INSERT que permita evidenciar que la restricción está funcionando:

```
1 -- Intento de insertar una compra con total alto y fecha reciente (tendrá éxito)
2 INSERT INTO Compra (Fecha, Total, IDcliente, IDempleado)
3 VALUES (CURRENT_DATE, 1000.00, 1, 1);
4
5 -- Intento de insertar una compra con total alto y fecha antigua (fallará)
6 INSERT INTO Compra (Fecha, Total, IDcliente, IDempleado)
7 VALUES ('1995-01-01', 1000.00, 1, 1);
8
9
10
```

Data Output Messages Notifications

ERROR: La fila que falla contiene (106, 1995-01-01, 1000.00, 1, 1).el nuevo registro para la relación «compra» viola la restricción «check» «check_fecha_total_compra»

ERROR: el nuevo registro para la relación «compra» viola la restricción «check» «check_fecha_total_compra»

SQL state: 23514

Detail: La fila que falla contiene (106, 1995-01-01, 1000.00, 1, 1).

2. Restricción 2

Tabla elegida: Comic

Breve descripción de la restricción: Ni la cantidad ni el precio de un cómic pueden ser simultáneamente los valores mínimos.

Instrucción para la creación de la restricción:

```
ALTER TABLE Comic
ADD CONSTRAINT check_precio_cantidad
CHECK (NOT (Precio = 0 AND Cantidad = 0));
```

Instrucción INSERT que permita evidenciar que la restricción está funcionando:

```
1 -- Intento de insertar un cómic con precio o cantidad válida (tendrá éxito)
2 INSERT INTO Comic (Personaje, Numero, Autor, Año, Cantidad, Precio)
3 VALUES ('Prueba', 1, 'Autor', 2020, 1, 0);
4
5 -- Intento de insertar un cómic con cantidad y precio ambos en 0 (fallará)
6 INSERT INTO Comic (Personaje, Numero, Autor, Año, Cantidad, Precio)
7 VALUES ('Prueba', 1, 'Autor', 2020, 0, 0);
8
9
10
```

Data Output Messages Notifications

ERROR: La fila que falla contiene (110, Prueba, 1, Autor, 2020, 0, 0).el nuevo registro para la relación «comic» viola la restricción «check» «check_precio_cantidad»

ERROR: el nuevo registro para la relación «comic» viola la restricción «check» «check_precio_cantidad»

SQL state: 23514

Detail: La fila que falla contiene (110, Prueba, 1, Autor, 2020, 0, 0).

10. Consultas

1. Consulta 1

Redacción clara de la consulta: Listar todos los cómics de 'Spider-Man', ordenados por su número.

Código en lenguaje SQL de la consulta:

```
SELECT IDcomic, Personaje, Numero
FROM Comic
WHERE Personaje = 'Spider-Man'
ORDER BY Numero;
```

```

1  -- Consulta 1
2  SELECT IDcomic, Personaje, Numero
3  FROM Comic
4  WHERE Personaje = 'Spider-Man'
5  ORDER BY Numero;
6
7
8
9
10
11
12
13

```

Data Output Messages Notifications

	Idcomic [PK] integer	personaje character varying (50)	numero integer
1	1	Spider-Man	1
2	26	Spider-Man	2
3	51	Spider-Man	18

2. Consulta 2

Redacción clara de la consulta: Calcular el total de ventas realizadas por cada empleado.

Código en lenguaje SQL de la consulta:

```

SELECT Empleado.IDempleado, Empleado.Nombre, Empleado.Apellido,
       SUM(Compra.Total) AS TotalVentas
FROM Compra
JOIN Empleado ON Compra.IDempleado = Empleado.IDempleado
GROUP BY Empleado.IDempleado, Empleado.Nombre, Empleado.Apellido;

```

```

1  -- Consulta 2
2  SELECT Empleado.IDEmpleado, Empleado.Nombre, Empleado.Apellido, SUM(Compra.Total) AS TotalVentas
3  FROM Compra
4  JOIN Empleado ON Compra.IDEmpleado = Empleado.IDEmpleado
5  GROUP BY Empleado.IDEmpleado, Empleado.Nombre, Empleado.Apellido;
6

```

Data Output Messages Notifications				
	Idempleado [PK] integer	nombre character varying (50)	apellido character varying (50)	totalventas numeric
1	11	Gabbi	Pracy	213.60
2	8	Hertha	Guilaem	188.98
3	19	Celisse	Findon	85.91
4	4	Gregoire	Getley	204.62
5	14	Rubina	Bockett	167.06
6	3	Patten	Stowe	295.07
7	17	Isahella	Gasking	416.09
8	20	Cthrine	Randals	53.84
9	9	Darci	Stormes	99.82
10	13	Atlante	Childerley	291.14
11	7	Rupert	Upham	339.39
12	10	Ermengarde	Orth	102.48
13	1	Janelle	Bleeze	234.38
14	5	Hyatt	Aleshintsev	149.06
15	18	Stella	Theuff	273.36
16	2	Lil	Firebrace	147.58
17	15	Bryana	Brunsen	370.12
18	16	Fonz	Warnock	244.04
19	6	Sander	Dumphries	467.93
20	12	Valerye	Halwood	401.67

3. Consulta 3

Redacción clara de la consulta: Listar los cómics más vendidos, basándose en la cantidad de veces que aparecen en las compras.

Código en lenguaje SQL de la consulta:

```

SELECT Comic.IDcomic, Comic.Personaje, COUNT(Incluye.IDcomic) AS VecesVendido
FROM Incluye
JOIN Comic ON Incluye.IDcomic = Comic.IDcomic
GROUP BY Comic.IDcomic, Comic.Personaje
ORDER BY VecesVendido DESC;

```

```

1  -- Consulta 3
2  SELECT Comic.IDcomic, Comic.Personaje, COUNT(Incluye.IDcomic) AS VecesVendido
3  FROM Incluye
4  JOIN Comic ON Incluye.IDcomic = Comic.IDcomic
5  GROUP BY Comic.IDcomic, Comic.Personaje
6  ORDER BY VecesVendido DESC;

```

Data Output Messages Notifications

	Idcomic [PK] integer	personaje character varying (50)	vecesvendido bigint
1	46	Nightwing	5
2	93	Harley Quinn	3
3	90	Daredevil	3
4	19	Joker	3
5	20	Catwoman	3
6	16	Doctor Strange	3
7	57	Green Lantern	2
8	80	Iron Man	2
9	45	Catwoman	2
10	84	Hulk	2
11	50	Cyclops	2
12	22	Deadpool	2
13	59	Hulk	2
14	62	Aquaman	2
15	98	Wolverine	2
16	42	Green Arrow	2
17	88	Black Panther	2
18	82	Green Lantern	2
19	32	Green Lantern	2
20	38	Black Panther	2
21	78	Wonder Woman	2
22	33	Thor	2
23	18	Harley Quinn	2
24	55	Iron Man	2

11. Vistas

1. Vista 1

Redacción clara de la vista planteada: Esta vista muestra el total de compras realizadas por cada cliente.

Código en lenguaje SQL que permita crear la vista solicitada:

```

CREATE VIEW TotalComprasPorCliente AS
SELECT Cliente.IDcliente, Cliente.Nombre, Cliente.Apellido,
       SUM(Compra.Total) AS TotalGastado
FROM Cliente
JOIN Compra ON Cliente.IDcliente = Compra.IDcliente
GROUP BY Cliente.IDcliente, Cliente.Nombre, Cliente.Apellido;

```

```

1 SELECT * FROM TotalComprasPorCliente;
2

```

Data Output Messages Notifications				
	Idcliente integer	nombre character varying (50)	apellido character varying (50)	totalgastado numeric
1	55	Maryl	Chate	90.75
2	27	Lynnet	De Beneditti	68.32
3	23	Dewey	Syvret	86.13
4	56	Blondelle	McCaster	80.04
5	58	Ruthie	Havenhand	35.81
6	8	Urson	Grube	201.92
7	87	Sollie	Dudderidge	90.39
8	74	Alastair	Rowswell	1.07
9	54	Annmarie	Chevalier	4.66
10	29	Vance	Rosbotham	43.36
11	71	Tresa	Whitford	110.61
12	4	Jared	Billanie	26.52
13	68	Noelani	Brophy	113.96
14	51	Corinne	Schubbert	44.30
15	96	Raymond	Simkovitz	26.32
16	70	Judas	Pergens	29.90
17	52	Jamal	Bangle	13.65
18	67	Jessee	Franek	69.79
19	63	Junia	Jon	98.87
20	6	Reese	Gonthier	66.04
21	84	Alfie	Piscot	180.82
22	92	Rolph	Ahren	39.05
23	36	Wendall	Marshal	77.91
24	21	Frick	Thuinn	86.04

2. Vista 2

Redacción clara de la vista planteada: Muestra el detalle de cada compra, incluyendo los cómics que se compraron en cada una.

Código en lenguaje SQL que permita crear la vista solicitada:

```

CREATE VIEW DetalleCompras AS
SELECT Compra.IDcompra, Compra.Fecha, Comic.Personaje, Comic.Numero
FROM Compra
JOIN Incluye ON Compra.IDcompra = Incluye.IDcompra
JOIN Comic ON Incluye.IDcomic = Comic.IDcomic;

```

```

1 SELECT * FROM DetalleCompras;
2

```

Data Output					Messages	Notifications
	Idcompra integer	fecha date	personaje character varying (50)	numero integer		
1	55	2023-06-15	Harley Quinn	24		
2	29	2023-03-05	Harley Quinn	24		
3	81	2023-04-07	Doctor Strange	1		
4	30	2023-04-02	Catwoman	8		
5	61	2023-11-04	Catwoman	23		
6	35	2022-12-12	Black Widow	40		
7	88	2023-02-14	Batman	13		
8	18	2023-09-23	Hulk	28		
9	33	2023-09-22	Daredevil	21		
10	62	2023-10-03	Thor	13		
11	44	2023-06-05	Superman	33		
12	6	2023-04-23	Cyclops	4		
13	6	2023-04-23	Cyclops	42		
14	27	2023-06-02	Joker	31		
15	89	2023-11-25	Nightwing	4		
16	34	2023-02-01	Spider-Man	2		
17	45	2023-01-29	Green Lantern	2		
18	50	2023-05-09	Harley Quinn	24		
19	54	2023-10-17	Green Arrow	38		
20	63	2023-10-16	Black Panther	2		
21	90	2023-07-05	Nightwing	4		
22	88	2023-02-14	Harley Quinn	1		
23	15	2023-02-18	Green Arrow	15		
24	74	2023-03-03	Hawkeye	17		

3. Vista 3

Redacción clara de la vista planteada: Muestra la cantidad total de cómics disponibles para cada personaje.

Código en lenguaje SQL que permita crear la vista solicitada:

```

CREATE VIEW InventarioPorPersonaje AS
SELECT Personaje, SUM(Cantidad) AS TotalDisponible
FROM Comic
GROUP BY Personaje;

```

1

2

SELECT * FROM InventarioPorPersonaje;

Data Output

Messages

Notifications

	personaje character varying (50)	totaldisponible bigint
1	Aquaman	52
2	Deadpool	37
3	Daredevil	25
4	Catwoman	48
5	The Flash	53
6	Joker	25
7	Captain America	55
8	Cyclops	50
9	Superman	59
10	Batman	52
11	Rogue	30
12	Wonder Woman	56
13	Green Lantern	38
14	Thor	35
15	Hawkeye	28
16	Wolverine	45
17	Black Panther	28
18	Black Widow	57
19	Hulk	54
20	Green Arrow	44
21	Iron Man	39
22	Spider-Man	33
23	Nightwing	46
24	Harley Quinn	28