****

**Tecnicatura Universitaria en Programación**

**Base de Datos II**

**Base de Datos para Gestión de Reseñas Gastronómicas**

**Equipo # 34**

**Integrantes:**

**Parlangeli, Marcos – Legajo 31191**

**Della Savia, Natali - 24960**

**Perosa Croses, Lisandro - 29859**

**Salas, Gabriel Edgardo - 31190**

## Explicación del Sistema

En este trabajo se ha desarrollado un sistema para una aplicación de reseñas de platos de restaurantes. La base de datos permite almacenar la información de los usuarios clientes y restaurantes, donde los clientes pueden navegar entre distintas opciones gastronómicas según sus preferencias, calificar y comentar sobre los platos consumidos, y los restaurantes pueden gestionar su oferta mostrando sus platos, revisando sus reseñas y analizando cuál es su plato más destacado o aquellos que presentan oportunidades de mejora.

Entre las funcionalidades principales tenemos:

* Listar restaurantes ubicados en la misma localidad que el usuario cliente.
* Permitir a los restaurantes conocer cuáles de sus platos tienen mejores calificaciones.
* Facilitar la consulta de las reseñas y el análisis de tendencias, gracias a la utilización de vistas.
* Garantizar la integridad de los datos y automatizar procesos críticos mediante procedimientos almacenados y triggers (como por ejemplo, prevenir inconsistencias en las fechas de reseñas o transformar la eliminación física en una baja lógica).

## Diagrama de Entidad Relación

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

## Código

La sección de Código abarca los aspectos fundamentales para el desarrollo de la base de datos, incluyendo la creación de la base de datos y sus tablas, así como la definición de relaciones entre ellas. También se detallan los procesos de carga de datos, las consultas SQL más relevantes, el uso de triggers para automatizar acciones y la implementación de procedimientos almacenados para optimizar operaciones repetitivas, todo orientado a la eficiencia y funcionalidad del sistema.

## Creación de Tablas

Texto

El contenido generado por IA puede ser incorrecto.

Texto

El contenido generado por IA puede ser incorrecto.

## Carga de Datos

Texto

El contenido generado por IA puede ser incorrecto.

Texto

El contenido generado por IA puede ser incorrecto.

Texto

El contenido generado por IA puede ser incorrecto.

Texto

El contenido generado por IA puede ser incorrecto.

Texto

El contenido generado por IA puede ser incorrecto.

Texto

El contenido generado por IA puede ser incorrecto.

Texto

El contenido generado por IA puede ser incorrecto.

Texto

El contenido generado por IA puede ser incorrecto.

Texto

El contenido generado por IA puede ser incorrecto.

Texto

El contenido generado por IA puede ser incorrecto.

Imagen que contiene Texto

El contenido generado por IA puede ser incorrecto.

## Objetos de Base de Datos clave en el sistema

### Vista 1:

La vista **View\_MejorPlato** que muestra para cada restaurante, la siguiente información: IDRestaurante, IDPlatoRestaurante, Cantidad de calificaciones y Promedio de calificaciones

Esta vista permite identificar el plato que más “suena” en función del volumen y la calidad de las reseñas, facilitando a los restaurantes conocer cuál de sus platos destaca dentro de la plataforma.

Texto

El contenido generado por IA puede ser incorrecto.

### Vista 2:

Se utiliza la vista **View\_PromedioPlatoPorRestaurante** que consolida, por cada restaurante y por tipo de plato, el promedio de calificaciones recibidas. La información mostrada incluye: IDRestaurante, NombreRestaurante, IDTipoPlato, NombrePlato y Promedio de las calificaciones.

Esta vista permite analizar y comparar la percepción de los clientes sobre las diferentes categorías del menú, dejando más en claro las fortalezas y áreas de mejora de cada restaurante.

Texto

El contenido generado por IA puede ser incorrecto.

### Vista 3

Se utiliza la vista **View\_ClienteMasCalificaciones** que identifica al cliente con la mayor cantidad de calificaciones realizadas en el sistema. La vista presenta los siguientes datos: Cuil, NombreCliente, ApellidoCliente y Total de calificaciones.

Esta información es clave para reconocer a los usuarios más activos en la plataforma, lo que puede ayudar en el análisis del comportamiento del cliente.

Texto

El contenido generado por IA puede ser incorrecto.

### Procedimiento Almacenado 1

El procedimiento almacenado **SP\_AgregarPlato** para ingresar un nuevo plato en la base de datos. Este procedimiento recibe los siguientes parámetros: IDRestaurante, TituloPlato, PrecioUnitario e IDTipoPlato.

Al ejecutarse, el procedimiento inserta el nuevo registro en la tabla PlatoRestaurante y muestra un mensaje de éxito, validando la operación de inserción del plato en el sistema.

**Texto

El contenido generado por IA puede ser incorrecto.**

### Procedimiento Almacenado 2

Se utiliza el procedimiento almacenado **SP\_ModificarDireccionIDLocalidad** para actualizar la dirección y la localidad de un restaurante. Este procedimiento recibe: IDRestaurante, Nueva Dirección y Nuevo IDLocalidad.

De esta manera, se facilita la gestión de la información de ubicación, permitiendo a los restaurantes mantener actualizada su dirección y localidad en la plataforma.

Texto

El contenido generado por IA puede ser incorrecto.

### Procedimiento Almacenado 3

Se utiliza el procedimiento almacenado **SP\_EliminarPlato** para eliminar un plato. Sin embargo, en conjunto con el trigger **BorrarPlato**, que se explicara más abajo, este procedimiento no borra físicamente el registro, sino que activa una baja lógica. Al ejecutarse, se actualiza el campo FechaBaja de la tabla PlatoRestaurante, preservando el historial y evitando la pérdida de información.

Texto

El contenido generado por IA puede ser incorrecto.

### Trigger 1

El trigger **TR\_Calificaciones\_ValidarFechas**, el cual se ejecuta automáticamente tras un INSERT o UPDATE en la tabla Calificaciones. Este trigger verifica que el valor de FechaCalificacion no sea anterior a FechaConsumo.  
Si se detecta que la fecha de la calificación es menor a la fecha del consumo, el trigger emite un mensaje de error y revierte la transacción mediante un rollback.

Texto

El contenido generado por IA puede ser incorrecto.

### Trigger 2

Finalmente, se utiliza el trigger **TR\_BorrarPlato**, que se activa en lugar de la eliminación física (DELETE) de un registro en la tabla PlatoRestaurante. La función de este trigger es la siguiente:  
Al intentarse eliminar un plato, se captura el ID del plato desde la tabla virtual *deleted*.  
Así, en lugar de borrar el registro, se actualiza el campo FechaBaja con la fecha actual, registrando así la baja lógica del plato.

Texto

El contenido generado por IA puede ser incorrecto.

**Trigger 3 y 4**

Dentro del sistema de gestión de reseñas gastronómicas, se implementaron dos triggers que garantizan la unicidad del campo Email en las tablas Cliente y Restaurante. Estas validaciones son fundamentales para asegurar la integridad y confiabilidad del sistema, especialmente considerando que el correo electrónico es un identificador clave para el ingreso a la plataforma, la recuperación de credenciales y la comunicación con los usuarios.

TR\_EmailUnicoCliente

Este trigger de tipo INSTEAD OF INSERT actúa sobre la tabla Cliente y tiene como propósito evitar la inserción de clientes con un correo electrónico que ya esté registrado. Al interceptar la operación de inserción:

* Comprueba si en la tabla ya existe un registro con el mismo email (Email = i.Email).
* Si detecta una coincidencia, cancela la operación con ROLLBACK e imprime un mensaje de advertencia, evitando la duplicación.
* Si no hay conflictos, realiza la inserción de manera controlada a partir de la tabla virtual inserted.

Esta medida permite mantener la unicidad lógica de los clientes incluso cuando no se haya definido una restricción UNIQUE directamente sobre el campo Email.

TR\_EmailUnicoRestaurante

El funcionamiento de este trigger es análogo al del cliente, pero aplicado a la tabla Restaurante. Su objetivo es impedir que dos restaurantes compartan la misma dirección de correo electrónico, lo cual podría comprometer la autenticación o gestión de contenido dentro del sistema. Al igual que el anterior:

* Realiza una verificación previa antes de insertar.
* En caso de duplicación, detiene el proceso y emite un mensaje.
* Permite la inserción solamente si se respeta la unicidad del email.

Importancia de estos triggers

Implementar esta lógica mediante triggers presenta varias ventajas:

* Permite centralizar la validación, evitando tener que repetirla desde la aplicación cliente.
* Ofrece una capa adicional de seguridad ante errores humanos o scripts que intenten insertar registros incorrectos.
* Ambos triggers fueron desarrollados utilizando bloques TRY...CATCH para el manejo seguro de errores y son parte fundamental del diseño robusto del sistema.

A screen shot of a computer program

AI-generated content may be incorrect.

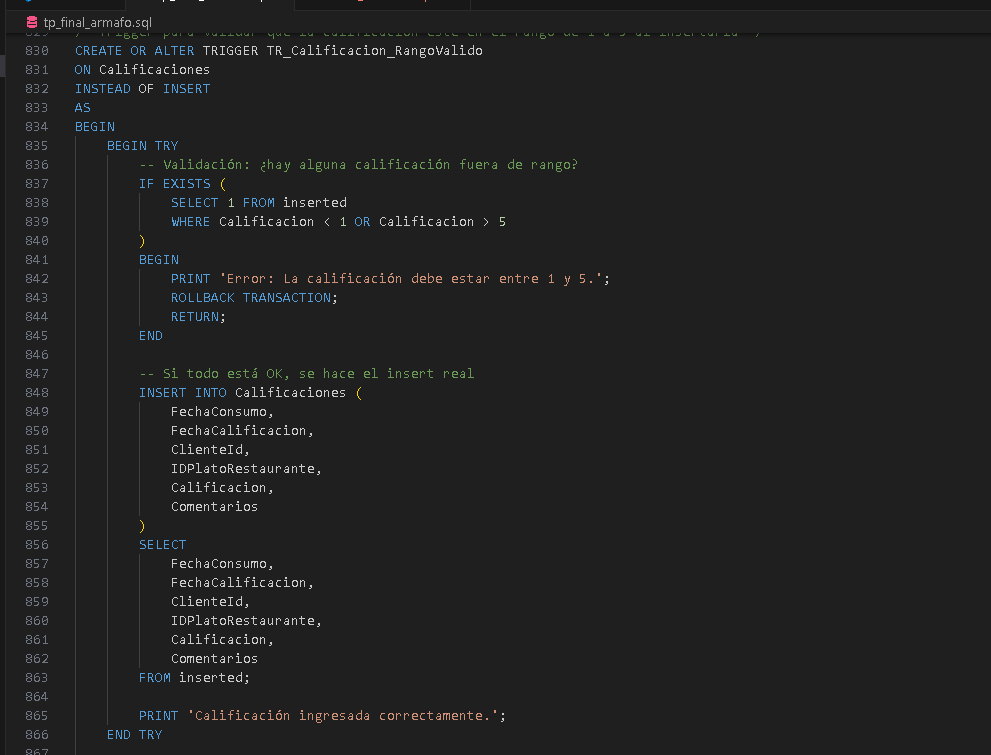
**Trigger 5**

En el contexto de una aplicación de reseñas de restaurantes, es esencial asegurar que los valores de calificación ingresados por los usuarios sean válidos y consistentes. Por ello, se diseñó e implementó el trigger TR\_ValidarRangoCalificacion, cuyo objetivo es **verificar que toda calificación registrada esté dentro del rango permitido de 1 a 5**.

**Funcionamiento del trigger**

Este trigger es de tipo INSTEAD OF INSERT y actúa sobre la tabla Calificaciones. La lógica que implementa es la siguiente:

* Al producirse una operación de inserción (INSERT), el trigger analiza los valores que llegarían a la base mediante la tabla virtual inserted.
* Si encuentra que **alguna de las calificaciones propuestas no está entre 1 y 5**, entonces:
  + Cancela la operación con un ROLLBACK.
  + Emite un mensaje con PRINT que indica que los valores están fuera del rango aceptado.
* Si todos los valores están dentro del rango permitido, entonces:
  + Inserta los datos en la tabla Calificaciones de manera segura.
  + Confirma la transacción con un COMMIT.



## Links a los recursos

**Script de creación de base de datos con datos**

<https://github.com/usuario/CodigoFuenteBD/>

**Video demo del sistema**

<https://youtube.com/usuario/VideoDemo>