

DICCIONARIO DB

- 1) ----- Sentencias SQL utilizadas
- 2) ----- NO SQL
- 3) ----- Plan de contingencia
- 4) ----- Tablas
- 5) ----- Procedimiento (equivalente a PL/SQL)

Consultas Select utilizadas

SELECT * FROM Historial_Puntos;

Obtiene todos los registros del historial de puntos acumulados y canjeados por los usuarios, permitiendo un seguimiento completo de los movimientos de puntos en el sistema de fidelización.

SELECT * FROM Puntos;

Recupera el estado actual de puntos acumulados por cada usuario, usado para mostrar su saldo disponible y gestionar recompensas.

SELECT * FROM Detalles_Facturas;

Extrae todos los detalles asociados a las facturas, incluyendo precios unitarios, tasas y descuentos, para poder mostrar y procesar el desglose de cada factura.

SELECT * FROM Pagos;

Obtiene todos los registros de pagos realizados, incluyendo método, fecha y estado, para la gestión y verificación de transacciones.

SELECT * FROM Facturas;

Recupera todas las facturas generadas en el sistema, mostrando fecha e importe total, fundamental para el control financiero y facturación.

SELECT * FROM Detalles_Pedidos;

Extrae los detalles específicos de cada pedido, como productos y cantidades, para poder mostrar y gestionar los pedidos realizados.

SELECT * FROM Pedidos;

Obtiene todos los pedidos registrados, relacionando factura, usuario y restaurante, usado para la gestión de pedidos y seguimiento.

SELECT * FROM Resenas;

Recupera todas las reseñas que los usuarios han dejado sobre restaurantes, incluyendo valoraciones y fechas, para mostrar opiniones y calificaciones.

SELECT * FROM Empleados;

Extrae información de todos los empleados, incluyendo datos personales, salario y fecha de contratación, para la administración del personal.

SELECT * FROM Proveedores_Ingredientes;

Obtiene la relación entre proveedores y los ingredientes que suministran, junto con precios y tiempos de entrega, para la gestión de inventario y compras.

SELECT * FROM Productos_Ingredientes;

Obtiene la relación entre proveedores y los ingredientes que suministran, junto con precios y tiempos de entrega, para la gestión de inventario y compras.

SELECT * FROM Productos;

Obtiene todos los productos disponibles, con sus categorías, nombres, precios y descripciones, para ser mostrados en la aplicación.

SELECT * FROM Ofertas;

Extrae todas las ofertas vigentes o pasadas, mostrando nombre, precio, descripción y fecha de expiración, para promociones en la tienda.

SELECT * FROM Usuarios;

Recupera la información completa de los usuarios registrados, incluyendo datos personales y credenciales, esencial para autenticación y gestión de perfiles.

SELECT * FROM Ingredientes;

Obtiene todos los ingredientes disponibles en el sistema, con detalles sobre stock, unidad de medida y estado, para gestión de inventario.

SELECT * FROM Restaurante;

Extrae la información de los restaurantes registrados, con sus datos de contacto y aforo, para mostrar en la aplicación y gestionar ubicaciones.

SELECT * FROM Proveedores;

Recupera datos de los proveedores, incluyendo empresa, contacto y email, para la gestión de relaciones comerciales.

SELECT * FROM Almacen;

Obtiene todos los almacenes, con nombre, capacidad y ubicación, para controlar la logística y almacenamiento de productos e ingredientes. Aunque solamente haya uno.

SELECT * FROM Categorías;

Extrae todas las categorías de productos, usadas para organizar y filtrar el catálogo en la aplicación.

SELECT * FROM Alertas_Stock;

Recupera las alertas generadas cuando el stock de ingredientes está bajo, permitiendo acciones preventivas en la gestión de inventarios.

NO SQL

Nos hemos creado una cuenta en la web de mongodb Atlas. Hemos creado una BD nueva y una colección (el equivalente a una tabla en sql) llamada candidatos (creada con el back).

The screenshot shows the MongoDB Atlas web interface. At the top, there's a breadcrumb trail: "SERGIO (ISW)'S ORG - 2025-05-30 > PROJECT 0 > DATABASES". Below this is the cluster name "ClusterSIOPrueba" with a MongoDB logo icon. A navigation bar contains tabs: "Overview", "Real Time", "Metrics", "Collections" (which is selected and underlined), "Atlas Search", and "Query Insights".

Below the navigation bar, it says "DATABASES: 1" and "COLLECTIONS: 1". On the left side, there's a sidebar with a "+ Create Database" button and a "Search Namespaces" search bar. Below the search bar, a tree view shows the database "reto_final" expanded, revealing a collection named "candidatos".

The main content area displays the details for the "reto_final.candidatos" collection. It shows "STORAGE SIZE: 36KB", "LOGICAL DATA SIZE: 689B", and "TOTAL DOCUMENTS: 3". There are tabs for "Find", "Indexes", "Schema Anti-Patterns" (with a notification icon), and "Aggre". A link "Generate queries from natural language in Compass" is present. At the bottom, there's a "Filter" section with a placeholder text "Type a query: { field: 'value' }".

Tras ello, hemos configurado la DB para que se pueda conectar desde la IP conectada a nuestro dominio.
Le damos a Network Access

The screenshot displays the MongoDB Atlas interface. On the left, the 'Clusters' sidebar is visible, with the 'Network Access' option highlighted by a yellow circle. The main content area shows the 'Overview' tab selected, displaying 'DATABASES: 1' and 'COLLECTIONS: 1'. Below this, there is a '+ Create Database' button and a search bar labeled 'Search Namespaces'. A dropdown menu is open for the cluster 'reto_final', showing the 'candidatos' namespace.

Clusters

SERVICES

- Atlas Search
- Stream Processing
- Triggers
- Migration**
- Data Federation

SECURITY

- Quickstart
- Backup
- Database Access
- Network Access**
- Advanced

Overview Real Time Me

DATABASES: 1 COLLECTIONS: 1

+ Create Database

Search Namespaces

▼ **reto_final**

- candidatos**

Después, le damos a Add IP Address y metemos nuestra IP para que la clase del backend se pueda conectar con la DB y pueda insertar los datos en ella.

SERGIO (ISW)'S ORG - 2025-05-30 > PROJECT 0

Network Access

IP Access List Peering Private Endpoint

+ ADD IP ADDRESS

Current IP Address not added. You will not be able to connect to databases from this address. **Add Current IP Address** **Do not show me again**

You will only be able to connect to your cluster from the following list of IP Addresses:

IP Address	Comment	Status	Actions
62.57.55.160/32	Created as part of the Auto Setup process	Active	EDIT DELETE
3.232.93.217/32	IP AWS VSCode	Active	EDIT DELETE

Add IP Access List Entry

Atlas only allows client connections to a cluster from entries in the project's IP Access List. Each entry should either be a single IP address or a CIDR-notated range of addresses. [Learn more](#)

ADD CURRENT IP ADDRESS **ALLOW ACCESS FROM ANYWHERE**

Access List Entry:

Comment:

☐ This entry is temporary and will be deleted in

Cancel **Confirm**

Todo esto junto con un fetch que llama al servlet del backend que mostraremos a continuación y le envía los datos que inserta el usuario en la web con un POST, y este, a través de su conexión con la DB de mongo se conecta a ella y le envía los datos metiéndolos en la colección candidatos.

Fetch que gestiona el envío de currículums así como todos los datos cumplimentados con relación a estos.

```
//GESTIONAR CV
window.addEventListener('DOMContentLoaded', () => {
  console.log("JS cargado correctamente");
  document.getElementById('cv').addEventListener('submit', async function (e) {
    e.preventDefault();
    console.log("Interceptado el submit");
    const form = e.target;
    const formData = new FormData(form);

    for (let [key, value] of formData.entries()) {
      console.log(key, value);
    }

    try {
      const res = await fetch('http://3.232.93.217:8080/api/uploadCV', {
        method: 'POST',
        body: formData,
      });

      const data = await res.json();
      alert(data.message);
    } catch (err) {
      console.error(err);
      alert('Error al subir el archivo');
    }
  });
});
```

Servlet que gestiona la conexión con la MongoDB así como la publicación de los datos recogidos del frontend con la api a la colección “candidatos”.

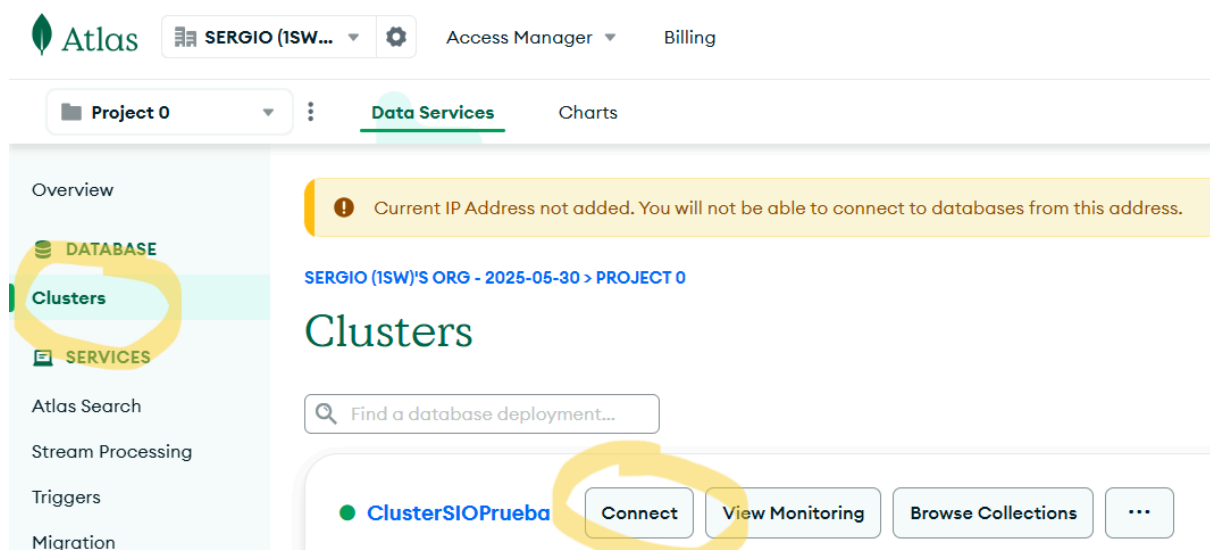
```
@WebServlet("/api/uploadCV") new *
@MultipartConfig
public class CVServlet extends HttpServlet {
    private static final String UPLOAD_DIR = "/home/ec2-user/retoBueno/CV-candidatos"; 2 usages
    private static final String MONGO_URI = "mongodb+srv://root:Aa12021888.@clustersiopruoba.ykpwkjn.mor

    @Override no usages new *
    protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws IOException {
        try {
            // Crear carpeta si no existe
            File uploadDir = new File(UPLOAD_DIR);
            if (!uploadDir.exists()) uploadDir.mkdirs();

            Part filePart = req.getPart(s: "cv");
            String nombre = req.getParameter(s: "nombre");
            String apellido = req.getParameter(s: "last-name");
            String email = req.getParameter(s: "email");
            String restaurant = req.getParameter(s: "candidacy");
            String job = req.getParameter(s: "job");
            String policy = req.getParameter(s: "policy");

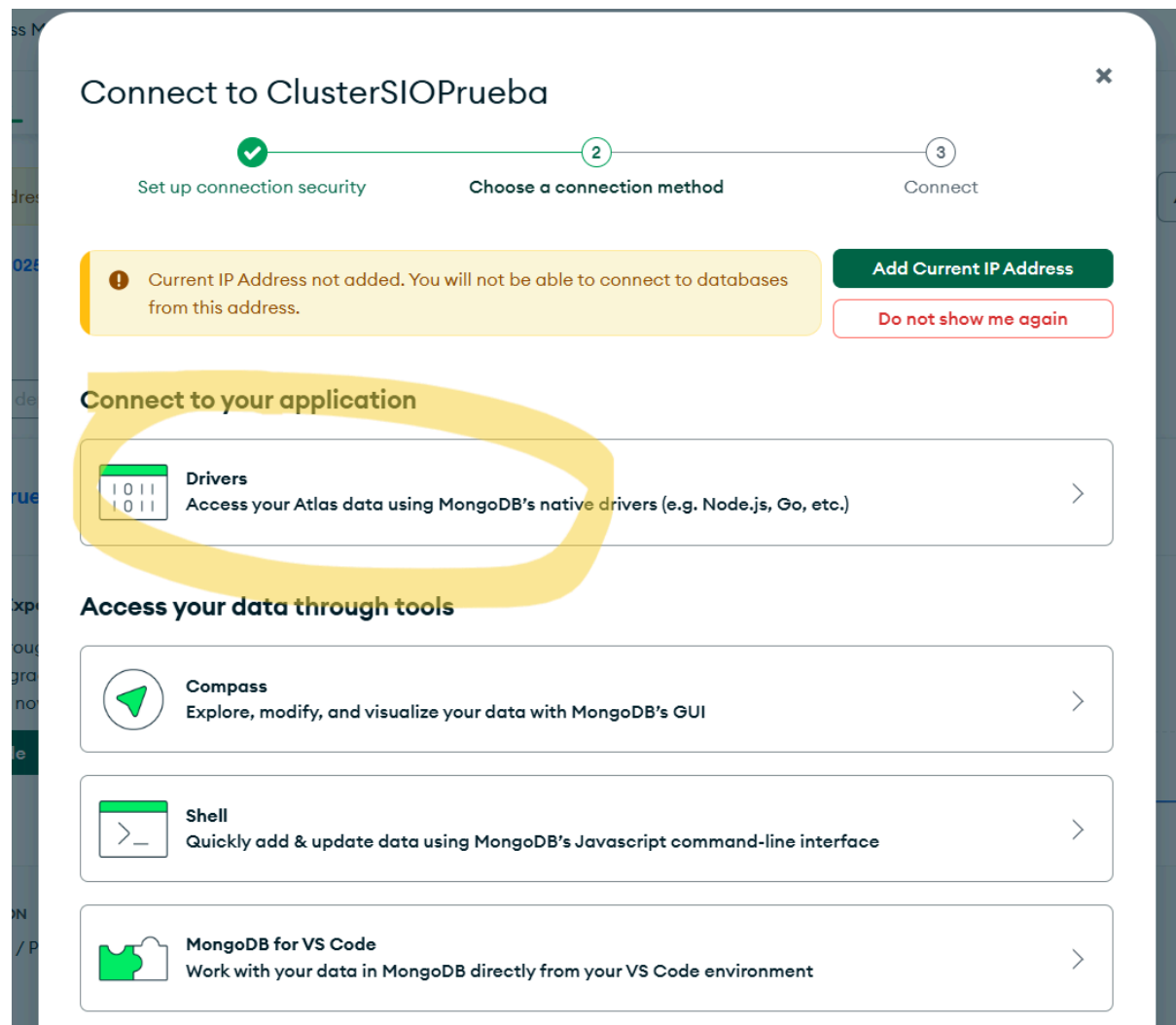
            String fileName = Paths.get(filePart.getSubmittedFileName()).getFileName().toString();
            String fullPath = UPLOAD_DIR + File.separator + fileName;
            filePart.write(fullPath);
        }
    }
}
```

En la constante MONGO_URI definimos la url de conexión por la que se van a comunicar backend de Java y MongoDB. Que en la web de MongoDB lo obtenemos así:



Tras darle a Clusters, nos saldrá el nuestro y cuando lo visualicemos, le daremos a Connect para elegir el método por el que nos vamos a comunicar con el back.

Elegimos por Drivers:



A continuación, debemos bajar en el display que nos aparece hasta ver la cadena de conexión que nos sale en "3. Add your connection string into your application code".

Con ella, cambiando el `<db_password>` por nuestra contraseña y el `root` por nuestro nombre de usuario de la DB para que la conexión sea exitosa. Como se muestra en la siguiente imagen.

Connect to ClusterSIOPrueba



Connecting with MongoDB Driver

1. Select your driver and version

We recommend installing and using the latest driver version.

Driver	Version
Python ▼	4.7 or later ▼

2. Install your driver

Run the following on the command line

Note: Use appropriate Python 3 executable

```
python -m pip install "pymongo[srv]"
```



[View MongoDB Python Driver installation instructions.](#)

3. Add your connection string into your application code

Use this connection string in your application

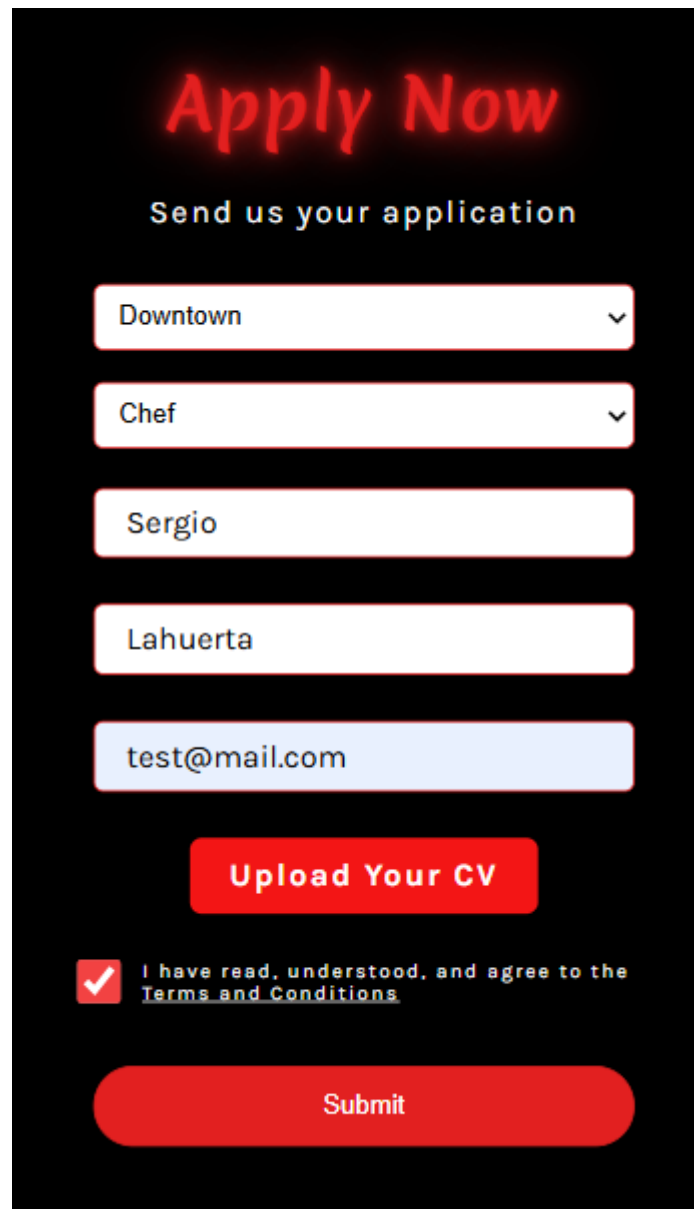
☐ View full code sample

```
mongodb+srv://root:<db_password>@clustersiopueba.ykpwkjm.mongodb.net/?  
retryWrites=true&w=majority&appName=ClusterSIOPrueba
```



Replace **<db_password>** with the password for the **root** database user. Ensure any option params are [URL encoded](#).

Por último, el usuario al ingresar los datos en la web y apretar el botón Submit, envía los datos al backend con el fetch y este los publica en la MongoDB en la colección candidatos. De tal manera:



Apply Now

Send us your application

Downtown

Chef

Sergio

Lahuerta

test@mail.com

Upload Your CV

☒ I have read, understood, and agree to the [Terms and Conditions](#)

Submit

Así luego en la web de MongoDB en nuestra colección vemos los datos introducidos;

The screenshot shows the MongoDB Atlas interface for a cluster named 'ClusterSIOPrueba'. The 'Collections' tab is selected, displaying the 'reto_final.candidatos' collection. The collection statistics show a storage size of 36KB, logical data size of 919B, 4 total documents, and a total index size of 36KB. The 'Find' tab is active, showing a filter bar with the query '{ field: 'value' }'. Below the filter, the query results are displayed, showing a single document. The document fields are: _id (ObjectId), nombre (Sergio), apellido (Lahuerta), email (test@mail.com), restaurant (downtown), job (Chef), acepta_politica (true), and ruta_cv (a file path). A yellow circle highlights the query results section.

ClusterSIOPrueba

Overview Real Time Metrics Collections Atlas Search Query Insights Query Profiler Perform

DATABASES: 1 COLLECTIONS: 1 PREVIEW New Data Explorer

+ Create Database

Search Namespaces

▼ reto_final

candidate

reto_final.candidatos

STORAGE SIZE: 36KB LOGICAL DATA SIZE: 919B TOTAL DOCUMENTS: 4 INDEXES TOTAL SIZE: 36KB

Find Indexes Schema Anti-Patterns Aggregation Search Indexes

Generate queries from natural language in Compass

Filter Type a query: { field: 'value' }

QUERY RESULTS: 1-1 OF 1

```
{
  "_id": ObjectId('6839e3b0cc5e3719cffc3e9d'),
  "nombre": "Sergio",
  "apellido": "Lahuerta",
  "email": "test@mail.com",
  "restaurant": "downtown",
  "job": "Chef",
  "acepta_politica": true,
  "ruta_cv": "/home/ec2-user/retoBueno/CV-candidatos/Curriculum Sergio Lahuerta.pdf"
}
```

Plan de contingencia

Drop schema if exists esquema_reto → Elimina el esquema con ese nombre si existe, si no, se ejecuta la línea siguiente.

Create schema esquema_reto → Crea un esquema con ese nombre, por el cual no estará llamado otro esquema gracias a la consulta anterior.

Use esquema_reto → Comunica al gestor de la base de datos que las consultas posteriores las aplique al esquema especificado.

Drop table if exists en el siguiente orden de las tablas → Elimina las tablas en el orden en el que se encuentren con esos nombres si existen, en caso contrario, se ejecuta la siguiente consulta.

Historial_Puntos,
Puntos,
Detalles_Facturas,
Pagos,
Facturas,
Detalles_Pedidos,
Pedidos,
Resenas,
Empleados,
Proveedores_Ingredientes,
Productos_Ingredientes,
Productos,
Ofertas,
Usuarios,
Ingredientes,
Restaurante,
Proveedores,
Almacen,
Categorias,
Alertas_Stock;

Tablas:

Tabla	Campo	Tipo de dato	Restricciones	Descripción
Categorías	ID_Categoria	INT	PK, Auto_increment	Identificador único de la categoría
	Nombre	Varchar (100)	NOT NULL	Nombre de la Categoría

Tabla	Campo	Tipo de dato	Restricciones	Descripción
Ofertas	ID_Oferta	INT	PK, Auto_increment	Identificador de la oferta
	Nombre	Varchar (100)	NOT NULL	Nombre de la oferta
	Precio	Decimal (10,2)	NOT NULL	Precio de la oferta
	Descripción	TEXT	NULL permitido	Descripción de oferta
	columnaImagen	Varchar (255)	NULL permitido	Imagen de la oferta
	fechaExpiracion	DATE	NOT NULL	fecha de caducidad de la oferta

Tabla	Campo	Tipo de dato	Restricciones	Descripción
Productos	ID_Producto	INT	PK, Auto_increment	Identificador del producto
	Nombre	Varchar (100)	NOT NULL	Nombre del producto
	Precio	Decimal (10,2)	NOT NULL	Precio del producto
	Descripción	TEXT	NULL permitido	Descripción del producto
	ImagenProducto	Varchar (250)	NOT NULL	Imagen del producto
	ID_Oferta	INT	FK → Ofertas	Oferta asociada
	ID_Categoria	INT	FK → Categorías	Categoría del producto

Tabla	Campo	Tipo de dato	Restricciones	Descripción
Ingredientes	ID_Ingrediente	INT	PK, Auto_increment	Identificador del ingrediente
	Nombre	Varchar (100)	NOT NULL	Nombre del ingrediente
	UnidadMedida	Varchar(50)	NULL permitido	Unidad de Medida
	StockDisponible	Decimal (10,2)	NOT NULL	Cantidad disponible en stock
	TipoAlmacenamiento	Varchar(50)	NULL permitido	Condiciones almacenamiento
	EstaActivo	Boolean	NULL permitido	Si el ingrediente está disponible

Tabla	Campo	Tipo de dato	Restricciones	Descripción
Productos_Ingr edientes	ID_ProductoIngr ediente	INT	PK, Auto_increment	Identificador del vínculo producto-ingredi ente
	ID_Ingrediente	INT	FK → Ingredientes	Ingrediente relacionado
	ID_Producto	INT	FK → Productos	Producto relacionado
	Cantidad	Decimal (10,2)	NOT NULL	Cantidad de ingrediente usado

Tabla	Campo	Tipo de dato	Restricciones	Descripción
Proveedores	ID_Proveedor	INT	PK, Auto_increment	Identificador del proveedor
	NombreEmpres a	Varchar (100)	NOT NULL	Nombre de la empresa proveedora
	Telefono	Varchar (100)	NOT NULL	Telefono del proveedor
	Email	Varchar (100)	NULL permitido	Email del proveedor

Tabla	Campo	Tipo de dato	Restricciones	Descripción
Almacen	ID_Almacen	INT	PK, Auto_increment	Identificador del almacén
	Nombre	Varchar (100)	NOT NULL	Nombre del almacén
	Capacidad	INT	NOT NULL	Capacidad total del almacén
	Ubicación	Varchar (100)	NULL permitido	Dirección o Ubicación

Tabla	Campo	Tipo de dato	Restricciones	Descripción
Proveedores_Ingredientes	ID_ProveedorIngrediente	INT	PK, Auto_increment	ID de relación proveedor-ingrediente
	ID_Proveedor	INT	FK → Proveedores	Proveedor del ingrediente
	ID_Ingrediente	DECIMAL(10,2)	FK → Ingredientes	Ingrediente suministrado
	precioUnitario	Varchar (100)	NOT NULL	Precio por unidad
	tiempoEntregaDias	INT	NULL permitido	Tiempo estimado de entrega

Tabla	Campo	Tipo de dato	Restricciones	Descripción
Restaurante	ID_Restaurante	INT	PK, Auto_increment	Identificador del restaurante
	Nombre	Varchar (100)	NOT NULL	Nombre restaurante
	Direccion	Varchar (100)	NULL permitido	Dirección
	Telefono	Varchar (100)	NULL permitido	Teléfono
	Email	Varchar (100)	NULL permitido	Correo Electrónico
	Aforo	INT	NULL permitido	Capacidad máxima del restaurante
	imagenRestaurante	Varchar (250)	NULL permitido	Imagen del restaurante

Tabla	Campo	Tipo de dato	Restricciones	Descripción
Empleados	ID_Empleado	INT	PK, Auto_increment	Identificador del empleado
	ID_Restaurante	INT	FK → Restaurante	Restaurante asignado
	Nombre	Varchar (100)	NOT NULL	Nombre empleados
	Apellidos	Varchar (100)	NOT NULL	Apellidos empleados
	DNI	Varchar (20)	NOT NULL	Documento de Identidad
	Telefono	Varchar (20)	NULL permitido	Teléfono
	Sueldo	DECIMAL(10,2)	NOT NULL	Salario
	FechaContratacion	DATE	NOT NULL	Fecha de contratación

Tabla	Campo	Tipo de dato	Restricciones	Descripción
Usuarios	ID_Usuarios	INT	PK, Auto_increment	Identificador del usuario
	Nombre	Varchar (100)	NULL permitido	Nombre usuarios
	DNI	Varchar (20)	NULL permitido	Documento de Identidad
	Telefono	Varchar (20)	NULL permitido	Teléfono
	Email	Varchar (100)	NOT NULL UNIQUE	Correo Electrónico único
	Contrasena	Varchar (100)	NOT NULL	Contraseña
	DNI	Varchar (20)	NULL permitido	Documento de identidad
	Direccion	Varchar (200)	NULL permitido	Dirección

Tabla	Campo	Tipo de dato	Restricciones	Descripción
Reseñas	ID_Reseñas	INT	PK, Auto_increment	ID de reseña
	ID_Usuario	INT	FK → Usuarios	Usuario que opina
	ID_Restaurante	INT	FK → Restaurante	Restaurante reseñado
	Valoracion	INT	NULL permitido	Puntuación
	Fecha	DATE	NULL permitido	Fecha de la reseña

Tabla	Campo	Tipo de dato	Restricciones	Descripción
Pedidos	ID_Pedido	INT	PK, Auto_increment	ID del pedido
	ID_Factura	INT	FK → Facturas	Factura asociada
	ID_Restaurante	INT	FK → Restaurante	Restaurante donde se realiza
	ID_Usuario	INT	FK → usuarios	Usuario que realiza el pedido
	Numero	INT	NOT NULL	Número o folio del pedido

Tabla	Campo	Tipo de dato	Restricciones	Descripción
Detalles_Pedidos	ID_DetallePedido	INT	PK, Auto_increment	ID de detalle del pedido
	ID_Pedido	INT	FK → Pedidos	Pedido al que pertenece
	ID_Producto	INT	FK → Productos	Producto solicitado
	Cantidad	INT	NOT NULL	Cantidad solicitada
	Observaciones	TEXT	NULL permitido	Comentarios o notas

Tabla	Campo	Tipo de dato	Restricciones	Descripción
Facturas	ID_Factura	INT	PK, Auto_increment	ID de la factura
	FechaFactura	DATE	NOT NULL	Fecha de emisión
	ImporteTotal	DECIMAL(10,2)	NOT NULL	Total de la factura

Tabla	Campo	Tipo de dato	Restricciones	Descripción
Detalles_Facturas	ID_DetalleFactura	INT	PK, Auto_increment	ID de detalle de la factura
	ID_Factura	INT	FK → Facturas	Factura correspondiente
	ID_DetallePedidos	INT	FK → Detalles_Pedidos	Línea de pedido asociada
	PrecioUnitario	DECIMAL(10,2)	NOT NULL	Precio por unidad
	TasaLocal	DECIMAL(10,2)	NULL permitido	Tasa local
	Descuento	DECIMAL(10,2)	NULL permitido	Descuento aplicado

Tabla	Campo	Tipo de dato	Restricciones	Descripción
Puntos	ID_puntos	INT	PK, Auto_increment	ID de puntos de usuario
	ID_Usuario	INT	FK → Usuarios	Usuario
	PuntosActuales	INT	NOT NULL	Puntos actuales acumulados

Tabla	Campo	Tipo de dato	Restricciones	Descripción
Historial_Puntos	ID_HistorialPuntos	INT	PK, Auto_increment	ID el movimiento de puntos
	ID_Factura	INT	FK → Facturas	Factura relacionada
	Fecha	DATE	NOT NULL	Fecha del movimiento
	Puntos	INT	NOT NULL	Cantidad de puntos ganados/canjeados
	TipoMovimiento	ENUM('GANADO', 'CANJEADO')	NOT NULL	Tipo de movimiento
	Descripcion	TEXT	NULL permitido	Descripción de movimiento

Tabla	Campo	Tipo de dato	Restricciones	Descripción
Pagos	ID_Pago	INT	PK, Auto_increment	ID del pago
	ID_Factura	INT	FK → Facturas	Factura pagada
	metodoPago	Varchar(50)	NOT NULL	Método de pago (efectivo, tarjeta...)
	fechaPago	DATE	NOT NULL	Fecha de pago
	estadoPago	Varchar(20)	NOT NULL	Estado del pago (pendiente, completado...)

Tabla	Campo	Tipo de dato	Restricciones	Descripción
Alertas_Stock	ID_Alerta	INT	PK, Auto_increment	ID del pago
	ID_Ingrediente	INT	FK → Facturas NOT NULL	ID del ingrediente
	fechaAlerta	DateTime	NOT NULL	fecha en la que se envía la alerta
	stockDisponible	decimal	NULL PERMITIDO	Cantidad de stock disponible
	mensaje	text	NULL PERMITIDO	Mensaje indicando el bajo stock de dicho producto

Procedimiento necesario:

Usamos el lenguaje procedural nativo de MySQL (procedimientos almacenados) que es el equivalente al PL/SQL de Oracle.

DELIMITER \$\$ → Cambia el delimitador de SQL por defecto que es ; a \$\$.

Drop Procedure if exists RevisarStockBajo → Elimina un procedimiento con ese nombre en caso de que exista.

Create Procedure RevisarStockBajo() → Crea el procedimiento RevisarStockBajo.

Begin y End\$\$ → Todo lo que ocurre dentro se ejecuta cuando se llama al procedimiento RevisarStockBajo().

```

INSERT INTO Alertas_Stock (ID_Ingrediente, StockDisponible, Mensaje)
SELECT
    ID_Ingrediente,
    StockDisponible,
    CONCAT('Stock bajo: ', StockDisponible, ' unidades disponibles para
    ingrediente ID ', Nombre, '')
FROM Ingredientes

```


WHERE StockDisponible < 10;

→ Consulta que comunica al procedimiento lo que debe hacer. En este caso se insertan los valores de ID_Ingrediente, StockDisponible, Mensaje en la tabla de Alertas_Stock cuando un ingrediente de la tabla Ingredientes tiene un StockDisponible menor a 10.

Delimiter ; → Final del procedimiento.

CALL RevisarStockBajo(); → Llama a ese procedimiento para luego poder revisar en la tabla Alertas_Stock que ingredientes hay que reponer.