ITP

**IMPLEMENTACIÓN DE BASE DE DATOS PARA LA GESTIÓN DE UN RESTAURANTE EN MONGODB**

**ENTREGADO POR:**

**JOHAN FERNANDO BURBANO CALPA**

**ENTREGADO A:**

**BRAYAN ARCOS**

**INSTITUTO TEGNOLOGIO DEL PUTUMAYO**

**MOCOA – PUTUMAYO**

**2024**

Contenido

[Contenido 2](#_Toc180038469)

[Resumen Ejecutivo 3](#_Toc180038470)

[Introducción 3](#_Toc180038471)

[Contexto y Motivación 3](#_Toc180038472)

[Alcance del Informe 3](#_Toc180038473)

[Objetivos 3](#_Toc180038474)

[Metodología 4](#_Toc180038475)

[Herramientas Utilizadas 4](#_Toc180038476)

[Procedimientos 4](#_Toc180038477)

[Realización del Taller: 4](#_Toc180038478)

[Desarrollo del Informe 5](#_Toc180038479)

[Esquema de la Base de Datos 5](#_Toc180038480)

[Modelo de Datos 7](#_Toc180038481)

[ Relaciones 7](#_Toc180038482)

[ Cardinalidad 7](#_Toc180038483)

[Métodos de Captura 8](#_Toc180038484)

[Consultas 9](#_Toc180038485)

[Análisis y Discusión 10](#_Toc180038486)

[Interpretación de Resultados 10](#_Toc180038487)

[Conclusiones 11](#_Toc180038488)

[Referencias 11](#_Toc180038489)

# Resumen Ejecutivo

Este informe presenta el diseño, desarrollo e implementación de una base de datos para un restaurante, utilizando MongoDB como sistema de gestión de bases de datos. Se destaca la creación de una base de datos con colecciones que incluyen restaurantes, administradores, empleados, clientes, mesas, reservas, platos, pedidos y facturas. El diseño se centró en optimizar las relaciones entre estas colecciones para reflejar de manera eficiente las interacciones dentro del restaurante. Las principales decisiones de diseño incluyen la implementación de relaciones uno a uno, uno a muchos y muchos a muchos. Los resultados obtenidos muestran que la base de datos es capaz de manejar consultas complejas, actualizar datos y realizar inserciones de manera efectiva, cumpliendo con los objetivos de optimización y gestión de la información del restaurante.

Introducción

## Contexto y Motivación

La base de datos fue diseñada con el objetivo de mejorar la gestión de un restaurante, permitiendo una administración eficiente de las relaciones entre empleados, clientes, mesas, reservas, pedidos y facturas. La motivación detrás de este proyecto es facilitar el análisis, la consulta y la manipulación de la información crítica del restaurante, como el seguimiento de las reservas, la gestión de empleados y el control de pedidos. Este sistema permitirá una mejora significativa en la eficiencia operativa del restaurante.

## Alcance del Informe

El informe abarca el diseño de la base de datos utilizando MongoDB, el desarrollo de relaciones entre colecciones, la ejecución de consultas para la inserción y actualización de datos, así como la validación de dichas relaciones. Se incluyen ejemplos de relaciones uno a uno, uno a muchos y muchos a muchos.

## Objetivos

* Diseñar una base de datos clara, escalable y eficiente.
* Implementar relaciones entre colecciones usando las mejores prácticas.
* Ejecutar consultas para la manipulación de datos (insert, update, upsert).
* Facilitar el análisis y manejo de información crítica del restaurante mediante consultas optimizadas.

# Metodología

## Herramientas Utilizadas

* MongoDB: Sistema NoSQL utilizado para la creación y gestión de la base de datos.
* Studio 3T: Herramienta gráfica empleada para la creación, visualización y análisis de las colecciones y las relaciones entre ellas.

## Procedimientos

Creación de la Base de Datos: Se diseñó la estructura de la base de datos en MongoDB, empleando Studio 3T para la creación y manipulación visual de las colecciones.

* Colecciones principales restaurantes, administradores, empleados, clientes, mesas, reservas, platos, pedidos y facturas
* Relaciones: Se implementaron relaciones uno a uno, uno a muchos y muchos a muchos entre las colecciones, según la naturaleza de los datos.

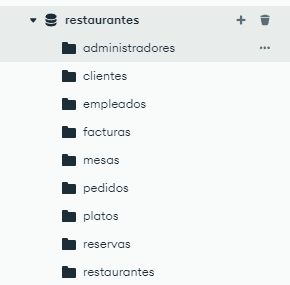
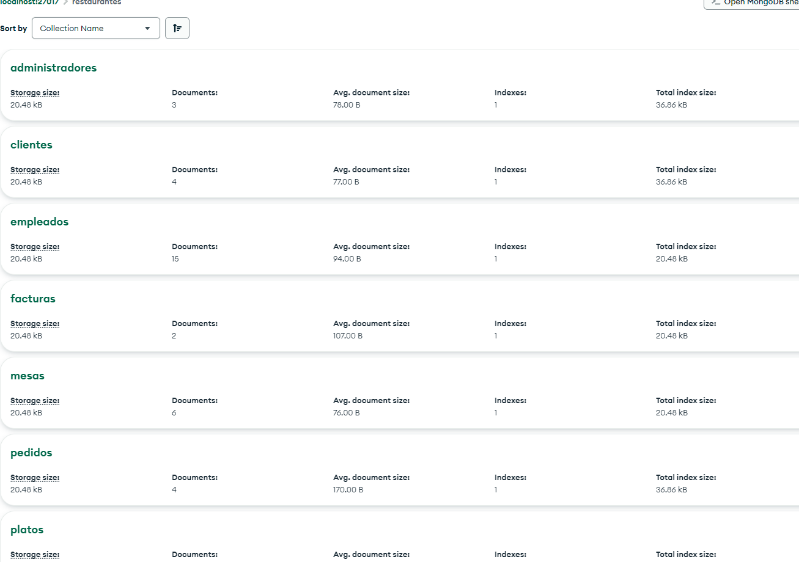
## Realización del Taller:

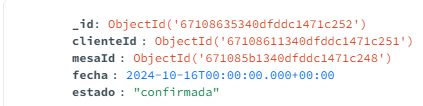
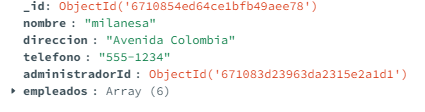
Se utilizaron archivos mediante consola para generar las colecciones, y se normalizó el modelo de datos para reducir redundancias y mejorar el rendimiento de las consultas.

Se llevaron a cabo operaciones de inserción y actualización de datos utilizando las funciones updateOne(), setOnInsert y each() para garantizar la integridad y consistencia de los datos.

# Desarrollo del Informe

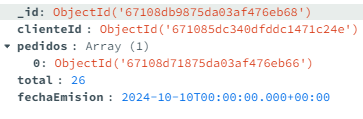
# Esquema de la Base de Datos

El esquema de la base de datos se diseñó para representar todas las entidades principales del restaurante y sus relaciones:

* Empleados: Cada empleado está asociado a un restaurante (relación 1 a muchos).
* Mesas y Reservas: Las reservas están vinculadas tanto a las mesas como a los clientes (relación muchos a muchos).
* Pedidos y Facturas: Los pedidos realizados por los clientes generan facturas (relación uno a muchos).
* Administrador y Restaurantes: cada administrador tiene un restaurante asignado y cada restaurante tiene un único administrador. (relación de uno a uno)

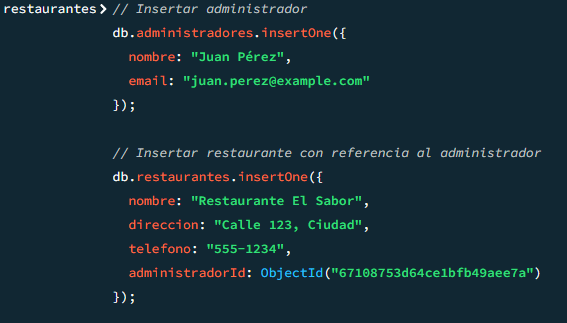
# Modelo de Datos

El modelo fue normalizado para minimizar redundancias y asegurar una estructura eficiente:

* Relaciones: Las relaciones entre colecciones fueron representadas mediante referencias (ObjectId), como en el caso de los empleados y su restaurante, o embebiendo documentos en casos donde se necesitaba información relacionada directamente dentro de la colección (por ejemplo, detalles del pedido dentro de la factura).
* Cardinalidad:
  + Uno a uno: Administrador y restaurantes.
  + Uno a muchos: Empleado y restaurante.
  + Muchos a muchos: Mesas y reservas, gestionado a través de colecciones intermediarias.

# Métodos de Captura

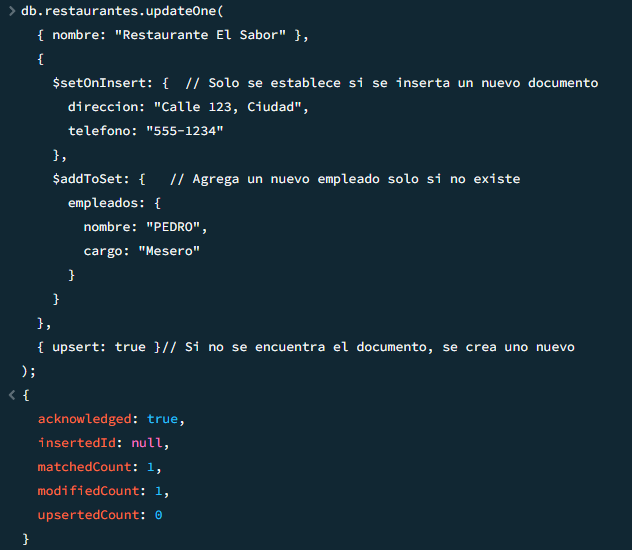
* Se crearon las diferentes colección e inserción de datos por medio de la consola de MongoDB.



* Posteriormente, se aplicaron relaciones mediante el uso de updateOne() con las opciones upsert y setOnInsert para insertar o actualizar datos cuando fuese necesario.

# Consultas

* Inserciones: Se insertaron empleados, clientes, reservas y pedidos utilizando insertMany().
* Actualizaciones: Se empleó updateOne() y setOnInsert para actualizar datos de clientes y reservas cuando ya existían entradas.



* Búsquedas: Se realizaron consultas para obtener la información de pedidos por cliente, reservas por mesa y facturación por restaurante.

# Análisis y Discusión

## Interpretación de Resultados

* El análisis de las consultas ejecutadas y las relaciones definidas demuestra que la base de datos cumple con los objetivos propuestos. La estructura diseñada permite:
* Un manejo eficiente de las reservas y la facturación, al optimizar las relaciones entre las colecciones involucradas.
* El control y actualización de datos de manera coherente utilizando las funciones updateOne() con coincidencias puntuales, lo que mejora la integridad de los datos.
* Las consultas son rápidas y eficientes, gracias a la correcta definición de relaciones y la normalización de los datos.

# Conclusiones

* La base de datos diseñada cubre todas las áreas críticas del restaurante, permitiendo una gestión fluida y organizada de los empleados, clientes, mesas, reservas, pedidos y facturas.
* MongoDB ha demostrado ser una solución adecuada para manejar relaciones complejas entre colecciones en un contexto de negocio como el de un restaurante.
* Las funciones de actualización y manejo de datos, como updateOne(), setOnInsert, y each(), permiten mantener la consistencia de los datos en todo momento.

# Referencias

* MongoDB Documentación. [https://docs.mongodb.com](https://docs.mongodb.com/)
* Studio 3T User Guide. <https://studio3t.com/knowledge-base/>