Wikicoa Project’s DataBase

ALMEIDA BENAVIDES ROMER ALEXANDER

DESARROLLO DE BASES DE DATOS

TECNOLOGÍA EN DESARROLLO DE SOFTWARE

QUINTO SEMESTRE

MOCOA PUTUMAYO

2024

Contenido

[RESUMEN EJECUTIVO 3](#_Toc176945633)

[Introducción 4](#_Toc176945634)

[Metodología 5](#_Toc176945635)

[Herramientas utilizadas: 5](#_Toc176945636)

[Procedimientos 5](#_Toc176945637)

[1. Identificación de Requerimientos 5](#_Toc176945638)

[2. Diseño del Diagrama Entidad-Relación (ER) 5](#_Toc176945639)

[3. Proceso de Normalización 6](#_Toc176945640)

[4. Implementación de Tablas en SQL 6](#_Toc176945641)

[5. Inserción de Datos de Prueba 6](#_Toc176945642)

[6. Ejecución de Consultas SQL 6](#_Toc176945643)

[Descripción de la base de datos 7](#_Toc176945644)

[Consultas SQL 8](#_Toc176945645)

[Consultas realizadas, sus debidos resultados y explicación: 12](#_Toc176945646)

[Diseño de la base de datos WIKICOA 15](#_Toc176945647)

[1. Entidades y atributos 15](#_Toc176945648)

[2. Relaciones 15](#_Toc176945649)

[3. Modelo ER 16](#_Toc176945650)

[4. Cardinalidad 16](#_Toc176945651)

[3. Normalización 16](#_Toc176945652)

[4. Tablas Normalizadas 17](#_Toc176945653)

[Análisis y Discusión 19](#_Toc176945654)

[Interpretación de Resultados 19](#_Toc176945655)

[1. Objetivos de la Base de Datos: 19](#_Toc176945656)

[2. Evaluación del Rendimiento: 19](#_Toc176945657)

[Conclusiones: 20](#_Toc176945658)

[Recomendaciones 21](#_Toc176945659)

# RESUMEN EJECUTIVO

Este informe presenta el diseño, implementación y normalización de una base de datos relacional para un sistema Wiki, enfocada en gestionar eficientemente la información de usuarios, artículos y categorías. El principal objetivo fue crear una base de datos bien estructurada, aplicando los principios de normalización hasta la Tercera Forma Normal (3NF) para garantizar la integridad y minimizar redundancias.

Se utilizaron herramientas como MySQL Workbench para el diseño y gestión de la base de datos, junto con SQL para la ejecución de consultas. Los métodos empleados incluyen la creación de un modelo entidad-relación (ERD), la definición de tablas con sus respectivas claves primarias y foráneas, y la ejecución de operaciones CRUD (Crear, Leer, Actualizar y Eliminar) para verificar el correcto funcionamiento de las relaciones.

El análisis incluyó la evaluación de consultas SQL sobre las tablas, optimización del esquema de la base de datos y una verificación detallada de las relaciones entre las entidades. Los resultados mostraron que las consultas se ejecutaron correctamente, y la estructura de la base de datos cumplió con los objetivos de rendimiento y organización de los datos.

# Introducción

El documento describe el diseño y desarrollo de una base de datos relacional para gestionar información en un sistema Wiki, destacando la importancia de la normalización y el uso de SQL en la administración de datos.

Cubre aspectos esenciales de SQL como consultas, optimización y diseño de bases de datos, con un enfoque en la implementación de un CRUD y normalización.

Finalmente, el objetivo que se quiere alcanzar es, crear una base de datos funcional y normalizada para un sistema de Wiki, aplicando buenas prácticas en la estructura y ejecución de consultas SQL.

# Metodología

## Herramientas utilizadas:

* MySQL Workbench
* Heidi SQL
* LUCID App
* AI

## Procedimientos

Detalladamente se describen los métodos y pasos aplicados durante el análisis y construcción de la Wikicoa’s Database, asegurando el cumplimiento de los alineamientos con los conceptos aprendidos previamente:

### 1. Identificación de Requerimientos

Se llevó a cabo un análisis detallado para identificar las principales entidades que formarían parte del sistema, como Usuarios, Artículos y Categorías, junto con sus respectivos atributos y relaciones. Este paso fue clave para asegurar un diseño de base de datos que cubriera las necesidades del sistema de forma eficiente y organizada.

### 2. Diseño del Diagrama Entidad-Relación (ER)

A partir del análisis anterior, se creó un diagrama entidad-relación (ER) que incluyó las principales entidades del sistema y cómo se relacionan entre sí. En este punto, se identificaron las claves primarias (Primary Key) y las claves foráneas (Foreign Key) necesarias para mantener la integridad referencial entre las tablas. Por ejemplo, la relación entre Usuarios y Artículos fue representada por la tabla "Crea", mientras que la relación entre Artículos y Categorías se gestionó a través de la tabla "Asigna".

### 3. Proceso de Normalización

Para evitar la duplicación de información y mantener la consistencia de los datos, se aplicaron las tres primeras formas normales (1NF, 2NF, y 3NF). En la 1NF, se garantizaron atributos atómicos, sin conjuntos repetidos de datos. La 2NF eliminó dependencias parciales, asegurando que todas las columnas dependieran completamente de la clave primaria en tablas con claves compuestas. Finalmente, en la 3NF, se eliminaron las dependencias transitivas, asegurando que cada atributo dependiera únicamente de la clave primaria.

### 4. Implementación de Tablas en SQL

Usando sentencias SQL en MySQL Workbench, se crearon las tablas necesarias según el modelo entidad-relación. Para cada tabla, se definieron los tipos de datos apropiados para las columnas, junto con restricciones como NOT NULL y claves primarias/foráneas que conectan las diferentes entidades. Este paso fue esencial para asegurar que las tablas pudieran manejar los datos de manera coherente y respetando las reglas de integridad.

### 5. Inserción de Datos de Prueba

Después de la creación de las tablas, se introdujeron datos de prueba utilizando comandos INSERT INTO. Estos datos incluyeron registros en las tablas de Usuarios, Artículos, y Categorías para validar que la base de datos funcionara correctamente y que las relaciones entre las tablas estuvieran establecidas correctamente. Los datos de prueba permitieron comprobar la funcionalidad de la base de datos y ajustar cualquier aspecto que no estuviera alineado con el diseño.

### 6. Ejecución de Consultas SQL

Para asegurar la operatividad y cumplimiento con las necesidades del sistema, se ejecutaron varias consultas SQL. Entre estas, se incluyeron búsquedas de artículos creados por un usuario específico, la obtención de listas de artículos dentro de determinadas categorías y el uso de combinaciones de tablas (joins) para extraer información relevante de múltiples entidades. Estas consultas permitieron confirmar que los datos estaban organizados correctamente y que el sistema podía extraer la información requerida de manera eficiente.

# Descripción de la base de datos

**Objetivo:** La base de datos está diseñada para gestionar información sobre usuarios, personas, artículos y categorías dentro de un sistema de tipo wiki, orientado a preservar y divulgar información autóctona del municipio de Mocoa. Dado que actualmente no existe un sistema que gestione eficientemente este tipo de recursos culturales, este proyecto tiene como objetivo abrir el acceso a este conocimiento a la comunidad local y extranjera, superando las limitaciones de acceso que tradicionalmente ha estado reservado a círculos privados.

Descripción de las tablas:

users: Esta tabla almacena los datos básicos de los usuarios registrados en el sistema, como el nombre de usuario (username), su contraseña encriptada (passwordKey), y la fecha de registro (registrationDate). Los usuarios pueden ser administradores o contribuyentes de artículos.

people: Relacionada con los usuarios, esta tabla gestiona información personal más detallada como el nombre completo (fullName), fecha de nacimiento (birthdate), género (gender) y correo electrónico (email). Esto permitirá tener un perfil más completo de los usuarios que participan en la creación y edición de artículos, además de permitir la integración de roles adicionales dentro del sistema en el futuro.

articles: Tabla central del sistema que almacena los artículos creados por los usuarios. Incluye el título (title), el contenido del artículo (content), la fecha de creación (creationDate) y la fecha de modificación (modificationDate). Cada artículo está relacionado con el usuario que lo creó.

categories: Esta tabla permite la clasificación de los artículos dentro del sistema. Cada categoría tiene un nombre (name) y una descripción (description), lo que facilita organizar los artículos por temas o áreas de conocimiento autóctono.

creates: Relaciona a los usuarios con los artículos que han creado. Almacena el ID del usuario (userId), el ID del artículo (articleId), y la fecha de creación (creationDate), sirviendo para llevar un control más detallado de las contribuciones dentro del sistema.

assigns: Anteriormente llamada Article\_Category, esta tabla establece la relación entre los artículos y sus categorías. Permite clasificar un artículo dentro de una o más categorías, asegurando una mejor organización del contenido.

Este diseño básico de la base de datos facilita la creación y administración de contenido sobre la cultura local, permitiendo que cualquier usuario contribuya y clasifique artículos de interés para la comunidad. La inclusión de la tabla people agrega una capa adicional de información sobre los usuarios que puede ser útil en la gestión de roles o permisos más personalizados.

Diagrama

Descripción generada automáticamente

Esquema de la base de datos con tablas principales como users como la principal entidad distinguida con su debida PK id, people que complementa a users almacenando información adicional de las personas relacionadas, con su PK id y FK userId que referencia a users. articles como segunda entidad fundamental, su PK id, categories como la última tabla y su PK id. creates es la tabla relacional entre artículos y usuarios que utiliza las FK en users y articles para generar la relación, y assign que vendría siendo la relación de muchos a muchos, temporalmente modificada para crear la relación entre artículos y categorías. Esta tabla almacena las FK en articles y categories, permitiendo así su relación y funcionando como intermediario para asignar a muchos artículos, muchas categorías.

# Diseño de la base de datos WIKICOA

1. Entidades y atributos:

* users (Entidad fuerte)

id (Primary Key)

username

passwordKey

registrationDate

* roles

id (Primary Key)

roleName

user\_roles (Tabla intermedia para usuarios y roles)

userId (Foreign Key)

roleId (Foreign Key)

* people (Entidad fuerte)

id (Primary Key)

firstName

lastName

birthDate

gender

* articles (Entidad fuerte)

id (Primary Key)

title

content

creationDate

modificationDate

userId (Foreign Key en users)

* categories (Entidad fuerte)

id (Primary Key)

name

description

* comments (Entidad fuerte)

id (Primary Key)

content

userId (Foreign Key)

articleId (Foreign Key)

creationDate

* article\_revisions (Entidad débil)

id (Primary Key)

articleId (Foreign Key)

userId (Foreign Key en users)

revisionDate

content

2. Relaciones

1. **creates** (entre users y articles):  
   Relación entre usuarios y artículos donde un usuario puede crear múltiples artículos.
   * userId (Foreign Key en articles)
   * articleId (Foreign Key en articles)
   * creationDate (atributo de la relación)
2. **assigns** (entre articles y categories):  
   Relación entre artículos y categorías, donde un artículo puede pertenecer a varias categorías y una categoría puede contener varios artículos.
   * articleId (Foreign Key en articles)
   * categoryId (Foreign Key en categories)
3. **comments** (entre users y articles):  
   Relación entre usuarios y artículos, ya que un usuario puede comentar en múltiples artículos, y cada artículo puede tener múltiples comentarios de diferentes usuarios.

* userId (Foreign Key en users)
* articleId (Foreign Key en articles)
* creationDate (atributo propio del comentario)

1. **user\_roles** (entre users y roles):  
   Relación entre usuarios y roles para asignar permisos específicos a cada usuario.

* userId (Foreign Key en users)
* roleId (Foreign Key en roles)

1. **article\_revisions** (entre articles y users):  
   Relación que permite almacenar revisiones de artículos y registrar al usuario que hizo cada cambio. Cada artículo puede tener múltiples revisiones, y cada revisión está asociada a un usuario que realizó la edición.

* articleId (Foreign Key en articles)
* userId (Foreign Key en users)
* revisionDate (atributo propio de la relación)
* content (contenido actualizado en cada revisión)

**3. Cardinalidad Actualizada**

1. Un usuario puede crear múltiples artículos, y cada artículo es creado por un único usuario:  
   Usuario (1) ---- (N) Crea (N) ---- (1) Artículo
2. Un artículo puede pertenecer a múltiples categorías, y una categoría puede contener varios artículos:  
   Artículo (N) ---- (N) Asigna (N) ---- (1) Categoría
3. Un usuario puede realizar múltiples comentarios en varios artículos, y un artículo puede tener múltiples comentarios de distintos usuarios:  
   Usuario (1) ---- (N) Comenta (N) ---- (1) Artículo
4. Un usuario puede tener múltiples roles y un rol puede ser asignado a varios usuarios:  
   Usuario (1) ---- (N) User\_roles (N) ---- (1) Rol
5. Un artículo puede tener múltiples revisiones hechas por distintos usuarios:  
   Artículo (1) ---- (N) Article\_revisions (N) ---- (1) Usuario

Diagrama ER Lucid

5. Normalización

Primera Forma Normal (1NF)

Todas las tablas cumplen con la 1NF, pues los valores de cada campo son atómicos.

Segunda Forma Normal (2NF)

Todas las tablas cuentan con una clave primaria completa, sin dependencias parciales en las tablas de relaciones.

Tercera Forma Normal (3NF)

No existen dependencias transitivas. Cada atributo en las tablas depende directamente de la clave primaria.

## Tablas Normalizadas

1. Tabla Users

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

1. Tabla People

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

1. Tabla Articles

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

1. Tabla Categories

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

1. Tabla Creates

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

1. Tabla Assigns

Interfaz de usuario gráfica

Descripción generada automáticamente

1. Tabla comments  
   Interfaz de usuario gráfica, Texto, Aplicación

   Descripción generada automáticamente
2. Tabla tagsInterfaz de usuario gráfica, Texto, Aplicación

   Descripción generada automáticamente
3. Tabla article\_tag Interfaz de usuario gráfica, Texto

   Descripción generada automáticamente
4. Tabla article\_revision Interfaz de usuario gráfica, Texto, Aplicación

   Descripción generada automáticamente
5. Tabla de roles Interfaz de usuario gráfica, Texto, Aplicación

   Descripción generada automáticamente
6. Tabla de user\_roles Interfaz de usuario gráfica, Texto, Aplicación

   Descripción generada automáticamente
7. Tabla de article\_likes Interfaz de usuario gráfica, Texto, Aplicación

   Descripción generada automáticamente

# Análisis y Discusión

## Interpretación de Resultados

### 1. Objetivos de la Base de Datos:

**Optimización de Gestión:** El objetivo de la base de datos de WIKICOA es mejorar la eficiencia en la administración de usuarios, artículos, roles y revisiones mediante una estructura que facilite operaciones rápidas y precisas. La integración de roles de usuario y revisiones de artículos asegura un sistema robusto y de calidad.

### 2. Evaluación del Rendimiento:

* **Consultas:** Se evaluó si las consultas diseñadas para la base de datos se ejecutaban de manera eficiente. El uso de claves primarias y foráneas, junto con los índices adecuados, debería haber mejorado significativamente el rendimiento de las consultas, especialmente aquellas relacionadas con la recuperación y gestión de artículos y la administración de usuarios. Las pruebas preliminares indicaron que las consultas se ejecutaban rápidamente, cumpliendo con las expectativas de rendimiento.
* **Integridad de Datos:** La integridad referencial se mantuvo utilizando claves foráneas, y las restricciones y validaciones implementadas en la base de datos aseguraron que los datos se mantuvieran consistentes y precisos. Esto es crucial en un sistema Wiki donde la precisión de la información y la trazabilidad de las ediciones son fundamentales.

# Conclusiones:

La estructura relacional de WIKICOA ha logrado un balance óptimo entre eficiencia, escalabilidad y facilidad de mantenimiento. La base de datos, organizada y normalizada en 3NF, permite un manejo eficaz y preciso de la información, lo que es esencial en un sistema Wiki donde la confiabilidad de los datos es crítica. No obstante, al observar un crecimiento continuo en el volumen de datos, se sugiere implementar mecanismos de optimización adicionales como la denormalización selectiva y el uso de caché. Estas medidas contribuirán a mantener un rendimiento adecuado y asegurarán que la base de datos pueda manejar de manera eficaz el aumento de carga y complejidad en el futuro.

# Recomendaciones

**Observaciones:** A partir del análisis de los resultados, se observó que la base de datos es robusta en términos de manejar las operaciones diarias de gestión de contenido y usuarios. Sin embargo, se identificaron áreas de mejora en términos de escalabilidad y manejo de cargas de consulta intensivas durante los períodos de alta actividad.

**Recomendaciones:** Basado en las observaciones, se hacen las siguientes recomendaciones para futuras mejoras:

* **Revisión de Índices:** Evaluar y posiblemente reconfigurar los índices actuales para optimizar el rendimiento durante picos de alta demanda.
* **Ajustes en la Normalización:** Considerar ajustes en la normalización de algunas tablas para reducir la complejidad de las consultas frecuentes, posiblemente mediante la denormalización selectiva donde sea beneficioso para el rendimiento.
* **Implementación de Caché de Consultas:** Introducir un sistema de caché para las consultas más frecuentes podría reducir significativamente los tiempos de respuesta y la carga sobre la base de datos durante los periodos de acceso intenso.

Estas recomendaciones están diseñadas para mejorar la eficiencia, escalabilidad y confiabilidad de la base de datos de Wikicoa, asegurando que pueda manejar un crecimiento en el número de usuarios y en la cantidad de contenido sin degradar la experiencia del usuario.

Referencias

* Moodle ITP
* Lucid.app
* SQL Performance Explained
* ChatGPT
* Github’s Repository: <https://github.com/Romerinjs/Wikicoa_DB_Proyect>