

TECNOLOGICO NACIONAL DE MEXICO

INSTITUTO TECNOLÓGICO DE MORELIA

“José María Morelos y Pavón”

Administración De Base De Datos

Profesor: Trujillo Ramos Roque

Proyecto Administración de Base de Datos

Carlos Jahir Castro Cázares

Jaime Isai Velázquez Aguilar

Diego Zamora Delgado

Jesús Iván Lemus Cervantes

Brandon Alexis Rodríguez Molina

Semestre Ago-Dic 2019

13 de diciembre del 2019

Definición del Proyecto

En base a nuestro problema planteado, nos dimos a la tarea de crear una base de datos que permita el control y registro de pacientes en una determinada área, la cual es la médica. Nos dimos a la tarea de hacer esto debido a que hemos notado que varios médicos particulares se mantienen haciendo registros aun en formato físico, lo cual hace que su trabajo sea menos eficiente, por tanto, esta idea podría llegar a implementarse como una alternativa en dichos casos, propiciando un mejor desempeño del médico en cuestión. Para lograr esto haremos uso de un sistema gestor de base de datos, lo que nos permitirá poder crear la base de datos, administrar los usuarios y sus roles, así como poder proteger la información de cada paciente. Esto estará implementado en un programa de aplicación sencillo que será manejado por el médico para efectuar las operaciones que él requiera realizar.

Objetivo

Creación de la base de datos para la elaboración de citas médicas, así como del registro de datos personales del cliente, eliminación de estos, consulta, modificación y la posibilidad de generar y registrar una factura de un cliente en específico. Todo esto mediante la base de datos implementada en una aplicación sencilla, intuitiva y fácil de usar por el médico, para agilizar parte de su trabajo, con el fin de que este logre una mejor eficiencia en el control de sus registros. Pese a no ser un proyecto nuevo, lo que haremos será tratar de extender este proyecto para su libre distribución, para que cada vez más, algún medico particular pueda hacer uso de esta herramienta.

Requisitos

Requisitos Funcionales

Identificación de Requisito	RF01
Nombre de Requisito	Registrar pacientes
Características	El usuario (medico) deberá registrar sus pacientes en el sistema
Descripción de Requisito	El sistema permitirá al usuario (medico) gestionar la información de sus pacientes (Citas, contacto y datos personales)
Requisito No Funcional	RNF1 RNF2
Prioridad de Requisito	Alta

Identificación de Requisito	RF02
Nombre de Requisito	Consultar información de pacientes
Características	El usuario (medico) tendrá total acceso a la información de cada uno de sus pacientes
Descripción de Requisito	El sistema permitirá al usuario (medico) consultar toda la información necesaria de los pacientes
Requisito No Funcional	RNF1 RNF2
Prioridad de Requisito	Alta

Identificación de Requisito	RF03
Nombre de Requisito	Modificar información de pacientes
Características	El usuario (medico) podrá modificar la información agregada previamente para su conveniencia.
Descripción de Requisito	El sistema permitirá al usuario (medico) modificar los datos de los pacientes
Requisito No Funcional	RNF1 RNF2
Prioridad de Requisito	Alta

Identificación de Requisito	RF04
Nombre de Requisito	Programar citas
Características	El usuario (medico) podrá agregar una o más citas para cada paciente que lo requiera.
Descripción de Requisito	El sistema permitirá al usuario (medico) programar citas para sus pacientes.
Requisito No Funcional	RNF1 RNF2
Prioridad de Requisito	Alta

Identificación de Requisito	RF05
Nombre de Requisito	Consultar información de citas
Características	El usuario (medico) podrá consultar datos acerca de las citas de cada paciente.
Descripción de Requisito	El sistema permitirá al usuario (medico) consultar información de las citas como diagnostico, fecha, pago, temperatura, peso y presión arterial.
Requisito No Funcional	RNF1 RNF2
Prioridad de Requisito	Alta

Identificación de Requisito	RF06
Nombre de Requisito	Modificar citas
Características	El usuario (medico) podrá modificar algún dato de una cita.
Descripción de Requisito	El sistema permitirá al usuario (medico) modificar información respecto a las citas como la fecha o
Requisito No Funcional	RNF1 RNF2
Prioridad de Requisito	Alta

Identificación de Requisito	RF07
Nombre de Requisito	Borrar cita
Características	El usuario (medico) tendrá la opción de eliminar (cancelar) una cita
Descripción de Requisito	El sistema permitirá al usuario (medico) eliminar citas de sus pacientes.
Requisito No Funcional	RNF1 RNF2
Prioridad de Requisito	Alta

Identificación de Requisito	RF08
Nombre de Requisito	Borrar pacientes
Características	El usuario (medico) podrá eliminar (dar de baja) a un paciente
Descripción de Requisito	El sistema permitirá al usuario (medico) dar de baja o eliminar a un paciente que ya no sea relevante o útil para él
Requisito No Funcional	RNF1 RNF2
Prioridad de Requisito	Alta

Identificación de Requisito	RF09
Nombre de Requisito	Facturar
Características	El usuario (medico) podrá hacer facturas
Descripción de Requisito	El sistema permitirá al usuario (medico) crear facturas para cada cita y para cada paciente que
Requisito No Funcional	RNF1 RNF2
Prioridad de Requisito	Alta

Requisitos No Funcionales

Identificación de Requisito	RNF1
Nombre de Requisito	Interfaz del sistema.
Características	El sistema presentará una interfaz de usuario sencilla y con buena estética para que sea de fácil manejo a los usuarios del sistema.
Descripción de Requisito	El sistema debe tener una interfaz de uso intuitiva y sencilla.
Prioridad de Requisito	Alta

Identificación de Requisito	RNF2
Nombre de Requisito	Seguridad en información
Características	El sistema garantizará a los usuarios y a los clientes una seguridad en cuanto a la información con la que se trata puesto que es información sensible.
Descripción de Requisito	Garantizar la seguridad del sistema con respecto a la información y datos que se manejan, así como el respaldo de la información en caso de alguna contingencia.
Prioridad de Requisito	Alta

Identificación de Requisito	RNF3
Nombre de Requisito	Mantenimiento
Características	El sistema deberá de tener un manual de usuario para facilitar los uso al ser utilizado por cualquier tipo de usuario.
Descripción de Requisito	El sistema debe disponer de una documentación fácilmente actualizable que permita a los usuarios realizar operaciones con el menor esfuerzo posible.
Prioridad de Requisito	Alta

Identificación de Requisito	RNF4
Nombre de Requisito	Diseño de interfaz intuitiva
Características	El sistema deberá de tener una interfaz fácil e intuitiva de usuario, teniendo en cuenta las características solicitadas.
Descripción de Requisito	La interfaz de usuario debe ajustarse a las características solicitadas, dentro de la cual estará incorporado el sistema de gestión de citas y de los pacientes atendidos así como los estudios realizados a cada unos de los clientes.
Prioridad de Requisito	Alta

Identificación de Requisito	RNF5
Nombre de Requisito	Desempeño
Características	El sistema garantizará a los usuarios un desempeño en cuanto a los datos almacenado en el sistema ofreciéndole una confiabilidad a esta misma.
Descripción de Requisito	Garantizar el desempeño del sistema informático a los diferentes usuarios. En este sentido la información almacenada o registros realizados podrán ser consultados y actualizados permanente y simultáneamente, sin que se afecte el tiempo de respuesta.
Prioridad de Requisito	Alta

Identificación de Requisito	RNF6
Nombre de Requisito	Nivel de Usuario
Características	Garantizar al usuario el acceso de información de acuerdo al nivel que posee.
Descripción de Requisito	Facilidades y controles para permitir el acceso a la información al personal autorizado a través del sistema, con la intención de consultar y subir información pertinente para cada una de ellas.
Prioridad de Requisito	Alta

Identificación de Requisito	RNF7
Nombre de Requisito	Base de datos
Características	Se utilizará un Sistema Gestor de Bases de datos
Descripción de Requisito	Se usará algun gestor SQL para la administración de la base de datos de la aplicación.
Prioridad de Requisito	Alta

Identificación de Requisito	RNF8
Nombre de Requisito	Herramientas de desarrollo
Características	Se usarán herramientas de desarrollo como lo son java y c#
Descripción de Requisito	Versiones: Java 8 y c# 6.0
Prioridad de Requisito	Alta

Descripción del Sistema

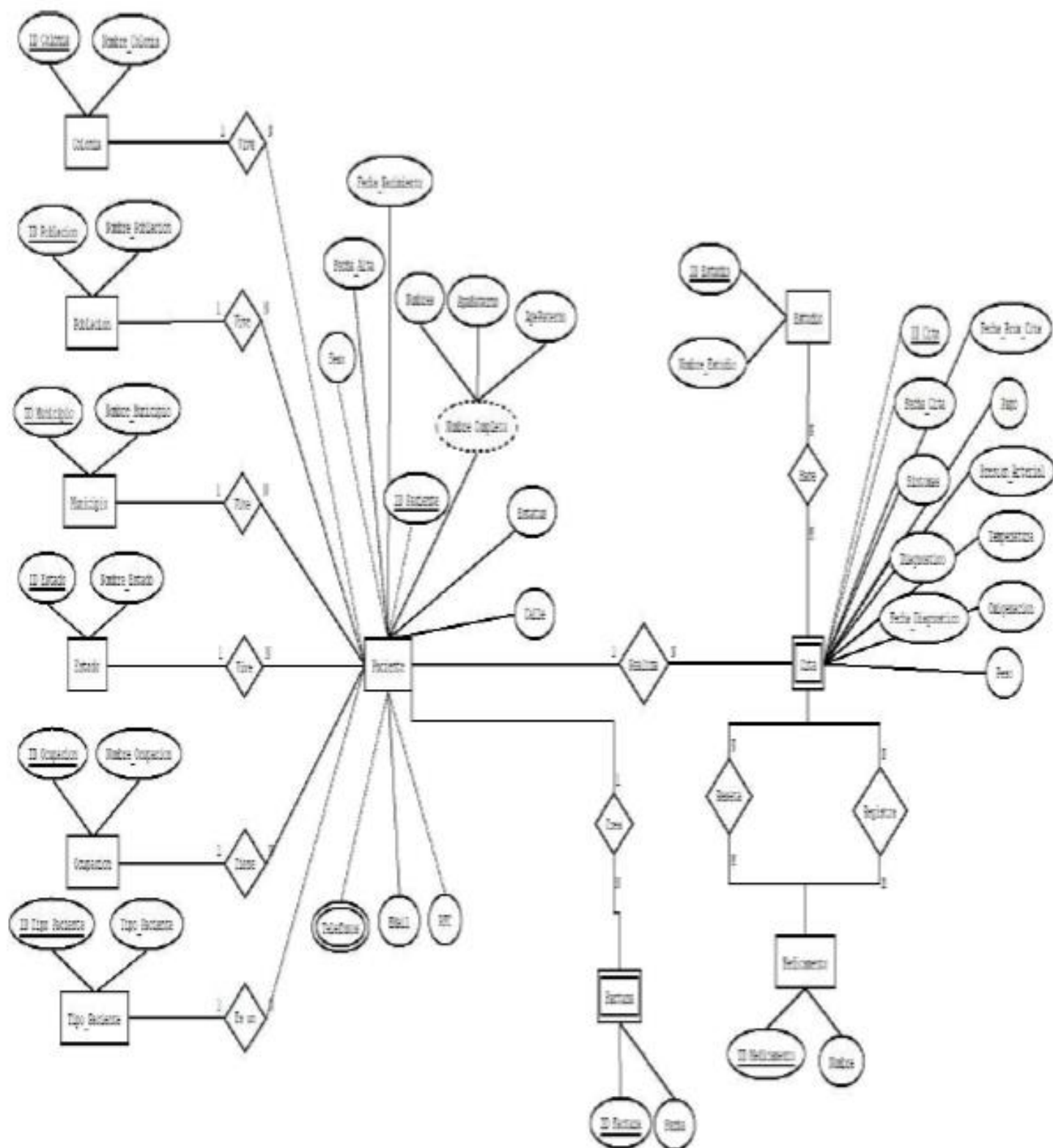
En la clínica medicas existen pacientes que pueden tener varia información, que se desea almacenar tales como el nombre completo, sexo, calle donde habita, e-mail, RFC, números telefónicos, fecha de nacimiento, fecha de alta, estatus. Además de información adicional como colonia, población, municipio, estado, ocupación y tipo de cliente.

En estas el paciente puede facturar una cantidad indefinida de facturas, donde solo se desea aguardar la fecha de la factura.

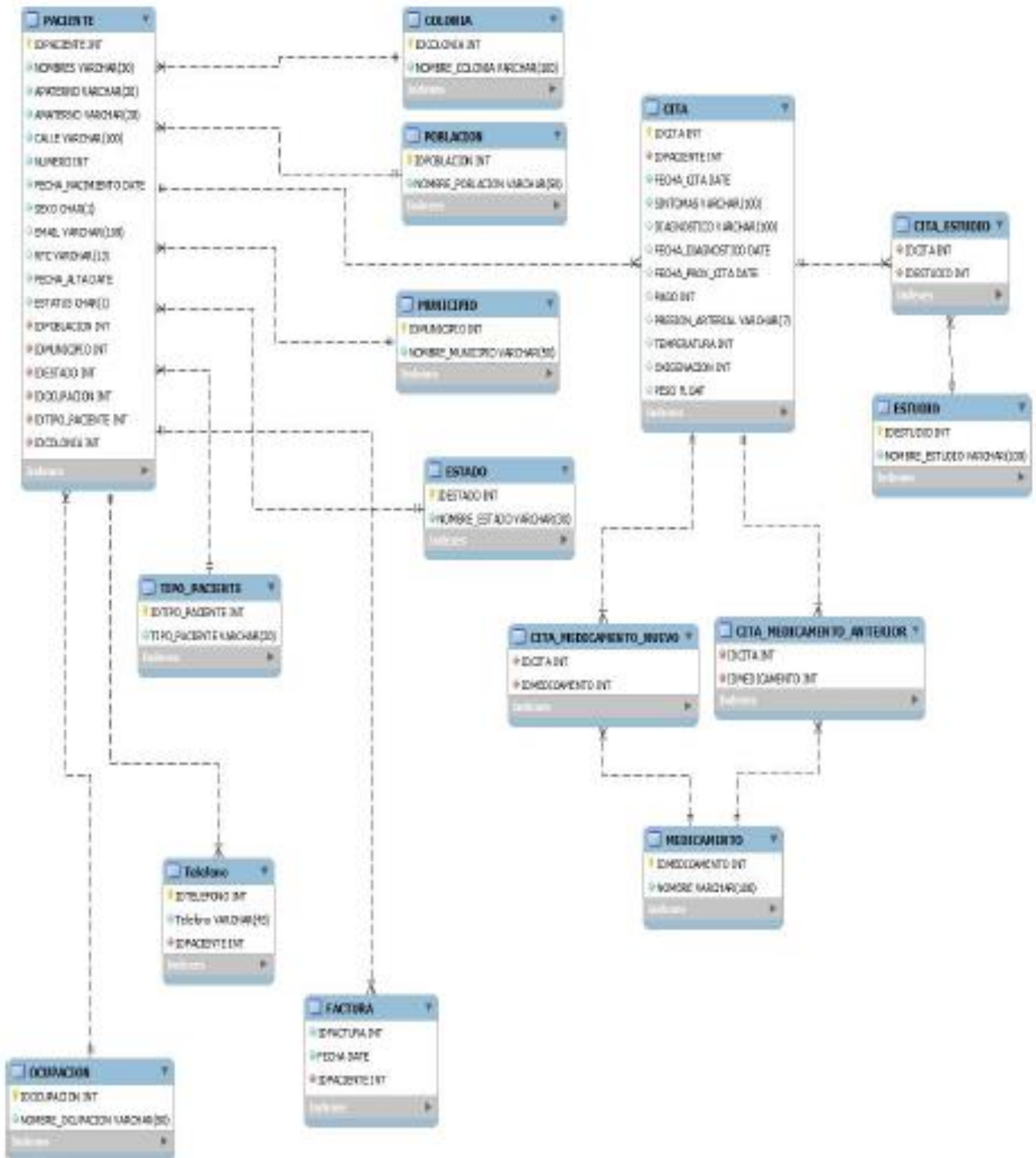
Igualmente, el paciente puede tener una cantidad indefinida de citas con el médico, las cuales tienen como información fecha de la cita, síntomas, diagnostico, fecha de diagnóstico, pago, presión arterial, temperatura, oxigenación, peso y la fecha de la siguiente cita.

Cada vez que se realiza una cita puede o no puede recetarse, algún o muchos medicamentos, y si ya se tenían medicamentos en las anteriores citas este es registrado. Igualmente se pueden realizar varios o ningún estudio médico.

- Diagrama Entidad / Relación



- Esquema Relacional



Definición del SMBD

Como podemos ver existen multitud de base de datos que podemos utilizar. MySQL es un sistema de base de datos relacional muy popular, y de los más utilizados para los sistemas de gestión de contenidos.

Al ser una base de datos que se utiliza en multitud de aplicaciones web existen multitud de tutoriales, foros, en la red en los que podemos encontrar la información que necesitamos. La condición de open source de MySQL, hace que su utilización sea gratuita e incluso se pueda configurarse con total libertad.

Es un servidor multi-usuarios muy rápido y robusto de ejecución de instrucciones en paralelo, es decir, que múltiples usuarios distribuidos a lo largo de una red local o Internet podrán ejecutar distintas tareas sobre las bases de datos localizadas en un mismo servidor.

Este tipo de bases de datos puede ejecutar desde acciones tan básicas, como insertar y borrar registros, actualizar información ó hacer consultas simples, hasta realizar tareas tan complejas como nuestra la aplicación lo requiera. Los

Requerimientos básicos para MySQL son:

- 512 Mb de memoria Ram
- 1024 Mb maquina virtual
- 1 GB de espacio de disco duro
- Sistema operativo:Windows,Linux y Unix
- Arquitectura del sistema 32/64 bit
- Protocolo de red TCP/IP

El servidor de base de datos MySQL proporciona lo último en escalabilidad, luciendo la capacidad de manejar aplicaciones profundamente arraigadas con una huella de tan sólo 1 MB a la ejecución de los almacenes de datos masivos que llevan a cabo terabytes de información.

Permite a los DBA configurar el servidor de base de datos MySQL de una manera mas sencilla, con el resultado final de un rendimiento increíble. MySQL tiene una variedad de opciones de alta disponibilidad de las configuraciones de replicación maestro / esclavo a alta velocidad. Apoyo transaccional robusto: MySQL ofrece uno de los más poderosos motores de bases de datos transaccionales en el mercado (InnoDB). Estas características aseguran el concepto "ACID", (aislamiento, durabilidad, atomicidad, consistencia) de transacciones, ilimitada bloqueo de filas, la capacidad de transacción distribuida, y el soporte de transacciones múltiples, donde los lectores no bloquean a los escritores y viceversa. La integridad de datos completa está asegurada a través de la integridad referencial obligadas por el servidor, los niveles de aislamiento de transacciones especializadas, y la detección de estancamiento instantánea. También proporciona SSH y soporte SSL para asegurar conexiones seguras y protegidas. Se pueden crear usuarios con ciertos privilegios para que los

usuarios sólo vean los datos que deberían, y potentes funciones de cifrado y descifrado de datos para asegurar que los datos sensibles se protejan de accesos no autorizados. Por último, las utilidades de copia de seguridad y recuperación proporcionadas a través de MySQL y proveedores de software de terceros permiten la copia de seguridad lógica y física completa, así como la recuperación completa y el punto en el tiempo.

Definición del espacio de la base de datos y tablespace

Para la asignación del espacio que será usado por nuestra base de datos, usaremos tablespace que serán asignados a cada una de las tablas de la base de datos, ya que nuestro sistema gestor de base de datos, no nos permite la asignación global de un solo tablespace para toda la base de datos si no que al momento de la creación de la base de datos se crea un directorio donde se aguardan cada una de las tablas de la base de datos, así que se crearan tablespace para cada una de las tablas de la base de datos para así llevar un control del espacio de la base de datos médico. Los tamaños de los tablespace por cada tabla serán los siguientes:

Tabla	Espacio
PACIENTE	16 KB
FACTURA	16 KB
CITA	16 KB
CITA_ESTUDIO	8 KB
CITA_MEDICAMENTO_NUEVO	8 KB
CITA_MEDICAMENTO_ANTERIOR	8 KB
COLONIA	4 KB
POBLACION	4 KB
MUNICIPIO	4 KB
ESTADO	4 KB
TIPO_PACIENTE	4 KB
TELEFONO	4 KB
OCUPACION	4 KB
ESTUDIO	4 KB
MEDICAMENTO	4 KB

Estos son los valores iniciales de los tablespace que se usaran por cada una de las tablas, aunque parece que el espacio reservado para los tablespace es muy grande, se indicaron así ya que este espacio es suficiente como para tener 1000 registros por cada tabla más sus índices y otros elementos, así que esto ayudara a que no se tenga que alterar los tablespace en mucho tiempo, aun así si el espacio es superado se configurara que los tablespace se expandan en una cantidad de 5KB cada vez que se necesite más espacio hasta un máximo de 1GB.

El tipo de almacenamiento secundario que se utilizará será en un disco duro (HDD), que contendrá la base de datos y la memoria primario será administrada por el sistema gestor de base de datos igualmente que la forma de organización de archivos.

Tablespace por usuario

En nuestra base de datos se usarán dos usuarios que serán las entidades que interactuarán con la base de datos, el usuario del médico que será quien tendrá control completo sobre la base de datos y recepcionista quien ingresará algunos registros sobre algunas tablas, para que estos usuarios trabajen sobre un espacio de memoria asignado se usarán 2 tablespace para cada usuario.

Tabla	Espacio
MEDICO	16 KB
RECEPCIONISTA	16 KB

Se usará así ya que tanto el médico como la recepcionista ingresan datos a la base de datos continuamente sobre todo en las tablas de paciente y cita. Con un espacio de estos tamaños es suficiente para que se trabaje continuamente durante un gran periodo de tiempo sin hacer modificaciones en los tablespace igualmente si estos sobrepasan la capacidad asignada, crecerán automáticamente en 10KB, hasta un máximo de 1GB.

Usuarios y roles

Para nuestro proyecto (bd_medico) definiremos dos tipos de usuarios que serán médico y recepcionista.

```
-- crear usuarios
create user 'medico'@'localhost'
identified by '12345';

create user 'receptionista'@'localhost'
identified by '12345';
```

Cada usuario desempeñará un rol distinto dentro de la base de datos, mientras que el médico tiene todos los privilegios de la base de datos, el recepcionista no tiene por qué tener acceso a todo, como al historial de la cirugía o a la historia clínica de un paciente, ya que es algo que no le corresponde. A continuación, se muestra imágenes con los privilegios de cada usuario:

```
grant all privileges on bd_medico.*
to 'medico'@'localhost';
```

*El médico con todos los privilegios en la bd_medico para todas las tablas

```
grant delete, insert, update
on bd_medico.cita
to 'receptionista'@'localhost';
```

*permisos de recepcionista en la bd_medico en la tabla cita.

El usuario recepcionista podrá visualizar todas las citas, además de poder cambiar de fecha una, así como también cancelar o eliminar alguna o todas.

```
grant INSERT, UPDATE, Delete
on bd_medico.factura
to 'receptionista'@'%';
```

*permisos de recepcionista en la bd_medico en la tabla factura.

El usuario recepcionista podrá agregar facturas, además de poder cambiar de fecha una, así como también cancelar o eliminar alguna o todas.

```
grant delete, insert, update, select      *permisos de recepcionista
on bd_medico.cita_estudio                en la bd_medico en la tabla
to 'recepcionista'@'localhost';          cita_estudio.
```

El usuario recepcionista podrá agregar estudios por cita, además de poder cambiar de fecha una, así como también cancelar o eliminar alguna o todas.

```
grant insert, select                    *permisos de recepcionista en
on bd_medico.cita_medimento_anterior    la bd_medico en la tabla
to 'recepcionista'@'localhost';          cita_medimento_anterior.
```

El usuario recepcionista podrá agregar medicamentos a una cita, así como también visualizarlos.

```
grant insert, select                    *permisos de recepcionista en la
on bd_medico.cita_medimento_nuevo       bd_medico en la tabla
to 'recepcionista'@'localhost';          cita_medimento_anterior.
```

El usuario recepcionista podrá agregar medicamentos a una cita, así como también visualizarlas.

Configuración del SMBD para soportar la carga de trabajo

En la base de datos “Medico” es necesario tener una configuración para tener un buen funcionamiento de esta, así que en este apartado explicaremos cómo optimizaremos un servidor MySQL para soportar nuestra carga de trabajo. ¿Para qué queremos optimizar un servidor MySQL? Pues para que pueda servir las peticiones más rápido usando menos recursos de CPU, RAM e I/O de disco, o simplemente para que aproveche mejor los recursos del sistema para garantizar una mejor estabilidad y una mayor velocidad de respuesta al acceder a datos almacenados en las bases de datos. Las configuraciones del servidor MySQL se realizarán desde un único archivo (my.cnf)

A continuación, vamos a especificar algunos de los parámetros generales que influyen en el rendimiento y en la estabilidad de MySQL y que se deben parametrizar en el my.cnf. Lo dividiremos en tres categorías Memoria, Procesos y conexiones

Memoria

- **query_cache_type:** Sirve para activar o desactivar cache, pondremos 1 para activar el cache de consultas.
- **query_cache_size:** Este parámetro especifica el tamaño del cache de consultas, asignaremos 64 MB de RAM por cada 1 GB de memoria física usable.
- **sort_buffer_size:** Con este parámetro configuramos el tamaño del cache de búsquedas de MySQL, configuraremos 1 MB por cada 1 GB de memoria RAM física disponible por ser lo recomendable.
- **read_buffer_size:** Con este parámetro configuramos el tamaño del cache de lecturas de MySQL, configuraremos 1 MB por cada 1 GB de memoria RAM física disponible por ser lo recomendable.

Procesos

- **thread_cache_size:** Es el número máximo de hilos de ejecución que se pueden cachear y reusar, configuraremos entre 32 y 64 para un uso normal.
- **thread_concurrency:** Especifica el número máximo de hijos de ejecución o procesos abiertos de MySQL, configuraremos 2 por cada núcleo de CPU disponible.
- **table_cache:** Especifica el máximo de tablas abiertas entre todos los threads o hilos de ejecución de MySQL, daremos un valor de 64 por ser lo recomendable.

Conexiones

- **max_connections:** Especifica el número máximo de conexiones totales que puede aceptar el servidor MySQL al mismo tiempo. Asignaremos 2, debido a que en la base de datos “medico” solo existen dos usuarios que continuamente estarán conectados.
- **wait_timeout:** Es el tiempo de espera que tarda MySQL en cerrar una conexión. Asignaremos 2 segundos.

Índices

Para construir los índices para nuestro proyecto, nos hemos basado en la forma en que los datos serán consultados (Consultas SQL), por la aplicación final que usara el médico.

Para explicar mejor los índices iremos tabla por tabla explicando que tipos de índices serán creados y el por qué lo decidimos así. Las tablas de nuestra base de datos es la siguientes.

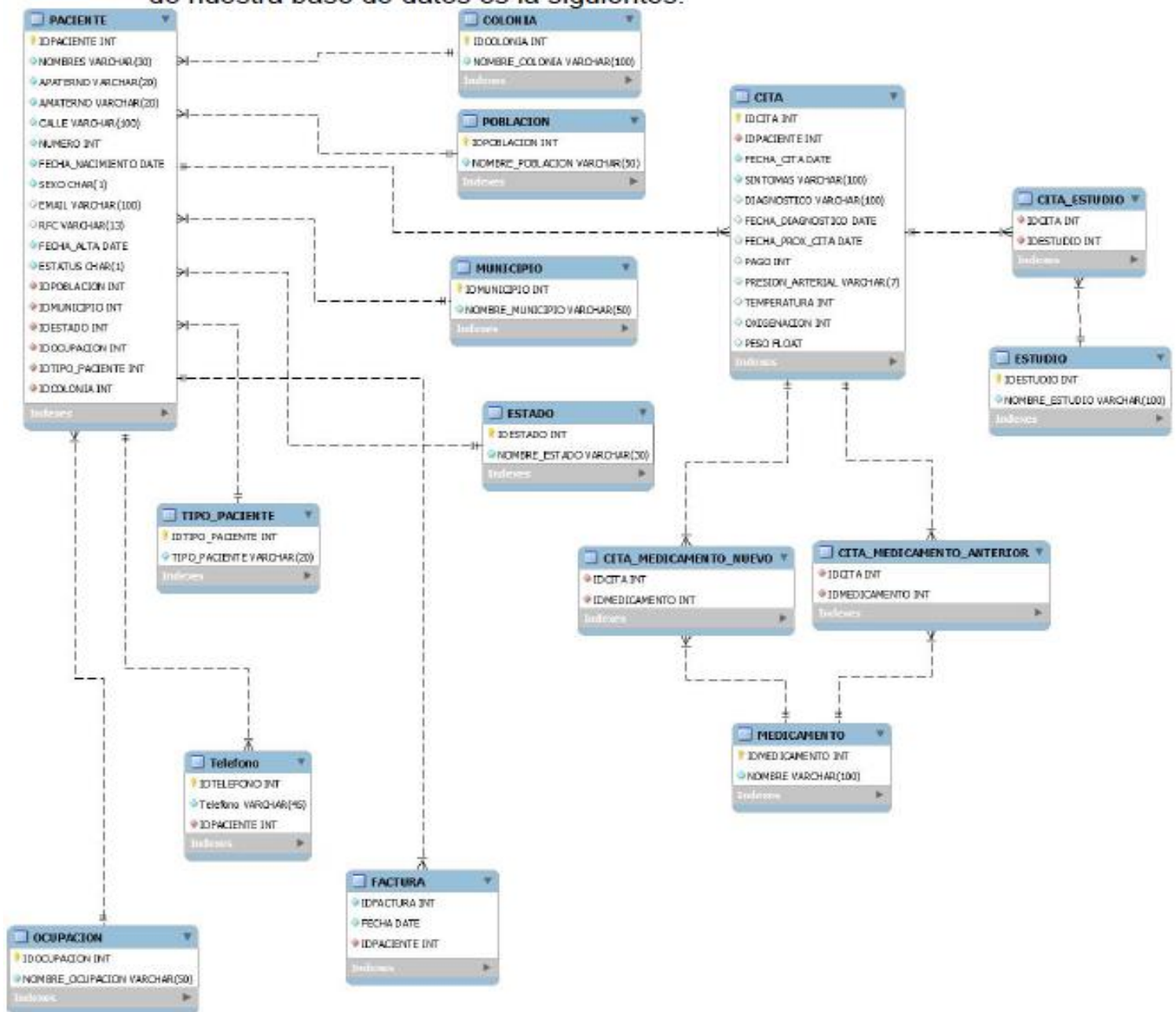


TABLA PACIENTE

Para esta tabla hemos decidido el construir un **índice primario**, ya que en todas las consultas que se realizaran se usara la llave primaria de este para relacionar esta tabla con otras y así obtener más información, por esta razón se construirá este tipo de índice para que agilice el proceso de búsqueda del id.

Por otro lado, se creará un **índice secundario por campo clave**, con el atributo del **RFC** ya que al momento de hacer facturas necesitaremos acceder a este campo con mayor frecuencia, así que nos conviene ya tener optimizada la búsqueda de este campo.

Por último, crearemos **índices secundarios por campo no clave**, a los atributos de **sexo, fecha_alta y estatus**, ya que a veces necesitamos hacer filtros sobre esta información, así que tenerla indexada nos ayudara a realizar estos filtros de manera más eficiente y rápida.

TABLA TIPO_PACIENTE

Esta tabla es un campo especial, ya que en los requisitos del paciente, este solo maneja dos tipos de pacientes, activos e inactivos así que en mucho tiempo esta tabla no tendrá nuevos registros y solo contendrá 2 registros, pensamos al principio el crear un **índice primario**, ya que usamos mayormente el **id** para buscar el estatus del paciente, pero al ver que solo contendrá dos registros en su vida y si llegara un nuevo tipo de paciente esta tabla realmente no crecerá mucho, por lo cual vemos que son muy pocos datos, por lo que concluimos que esta **no necesitara de algún índice**.

TABLA TELEFONO

En esta tabla vemos necesaria la creación, de un **índice primario** sobre el **id** de la tabla ya que este es el que necesitamos para relacionarlo con otras tablas y obtener información de este.

Pero más importante vemos la creación de un **índice secundario por campo clave** sobre el campo de **teléfono**, ya que al ser números de teléfonos estos deben de ser únicos por paciente y aunque un paciente pueda tener varios números de teléfono un paciente no puede tener el mismo número de teléfono que otro. Además, que, al momento de hacer una búsqueda por número de teléfono, este índice nos ayudara a optimizar esta búsqueda.

TABLA FACTURA

En esta tabla vemos la necesidad de hacer un **Índice Agrupado** sobre el campo de **Fecha**, Ya que las características del negocio implican que la amplia mayoría de los pacientes no piden factura, así que, aunque esta tabla tenga varios inserts, durante su uso estos no serán frecuentes. Además de que como será sobre un campo de fecha, los nuevos registros siempre serán más recientes en la fecha que los antiguos haciendo que al momento de reorganizar el índice este no tenga que hacer demasiado trabajo, alterando la optimización de nuestras consultas. Por último, hacemos este índice ya que se necesita saber a través de consultas las facturas de un día específico y que estas estén de la más reciente a la más antiguas, por el cual tener optimizado la búsqueda de fechas, sería de gran ayuda.

TABLA OCUPACION

Por las características de esta tabla conviene, la creación de dos índices primeramente un **índice primario** por el campo **ID_OCUPACION**, ya que al momento de acceder a la tabla a través de una unión con otra usaremos este campo para obtener la información pertinente. Y el segundo índice será un **índice agrupado**, sobre el campo de **NOMBRE_OCUPACION** ya que nos conviene tener la información organizada y ordenada, al momento de saber el contenido de la tabla. Esta tabla es un catálogo por lo que casi no habrá inserciones de registros nuevos sobre esta, por lo cual nos conviene tener estos tipos de índices para así mejorar el tiempo de respuesta de las consultas.

TABLA COLONIA

Por las características de esta tabla conviene, la creación de dos índices primeramente un **índice primario** por el campo **ID_COLONIA**, ya que al momento de acceder a la tabla a través de una unión con otra usaremos este campo para obtener la información pertinente. Y el segundo índice será un **índice agrupado**, sobre el campo de **NOMBRE_COLONIA** ya que nos conviene tener la información organizada y ordenada, al momento de saber el contenido de la tabla.

Esta tabla es un catálogo por lo que casi no habrá inserciones de registros nuevos sobre esta, por lo cual nos conviene tener estos tipos de índices para así mejorar el tiempo de respuesta de las consultas.

TABLA POBLACION

Por las características de esta tabla conviene, la creación de dos índices primeramente un **índice primario** por el campo **ID_POBLACION**, ya que al momento de acceder a la tabla a través de una unión con otra usaremos este campo para obtener la información pertinente. Y el segundo índice será un **índice agrupado**, sobre el campo de **NOMBRE_POBLACION** ya que nos conviene tener la información organizada y ordenada, al momento de saber el contenido de la tabla. Esta tabla es un catálogo por lo que casi no habrá inserciones de registros nuevos sobre esta, por lo cual nos conviene tener estos tipos de índices para así mejorar el tiempo de respuesta de las consultas.

TABLA MUNICIPIO

Por las características de esta tabla conviene, la creación de dos índices primeramente un **índice primario** por el campo **ID_MUNICIPIO**, ya que al momento de acceder a la tabla a través de una unión con otra usaremos este campo para obtener la información pertinente. Y el segundo índice será un **índice agrupado**, sobre el campo de **NOMBRE_MUNICIPIO** ya que nos conviene tener la información organizada y ordenada, al momento de saber el contenido de la tabla.

Esta tabla es un catálogo por lo que casi no habrá inserciones de registros nuevos sobre esta, por lo cual nos conviene tener estos tipos de índices para así mejorar el tiempo de respuesta de las consultas.

TABLA ESTADO

Por las características de esta tabla conviene, la creación de dos índices primeramente un **índice primario** por el campo **ID_ESTADO**, ya que al momento de acceder a la tabla a través de una unión con otra usaremos este campo para obtener la información pertinente. Y el segundo índice será un **índice agrupado**, sobre el campo de **NOMBRE_ESTADO** ya que nos conviene tener la información organizada y ordenada, al momento de saber el contenido de la tabla.

Esta tabla es un catálogo por lo que casi no habrá inserciones de registros nuevos sobre esta, por lo cual nos conviene tener estos tipos de índices para así mejorar el tiempo de respuesta de las consultas.

TABLA CITA

En esta tabla hemos decidido usar un **índice de tipo agrupado** para la columna **idestudio**, ya que inicialmente todos los datos de esta columna están en completo desorden (numéricamente) y si aplicamos un índice de tipo agrupado nos soluciona el 'problema' y ahora nuestros datos aparecerán ordenados y así optimizando tiempo para la consulta, y además en esta tabla no se están haciendo inserts de manera constante ya que los estudios casi siempre son los mismos y además los datos van ordenados en forma ascendente por lo que es recomendable usar este tipo de índice.

TABLA CITA_ESTUDIO

En esta tabla decidimos agregar **dos índices no agrupados**, uno para **idcita** y otro para **idestudio**, ya que los datos pueden estar organizados de forma aleatoria pero internamente les da un orden lógico y se van a estar alterando de manera constante. Será muy eficaz al momento de utilizar las instrucciones de where y join.

TABLA CITA

En esta tabla decidimos utilizar un **índice agrupado** para la columna **idcita**, ya que así será más eficaz para nuestro gestor el realizar las consultas, y como los datos van en forma ascendente no va a haber problemas al momento de agregar un nuevo registro.

Además, agregamos un **índice no agrupado** para la columna **idpaciente**, ya que este lo podemos encontrar de forma aleatoria y se va a estar alterando de manera constante por lo que este tipo de índice resulta es más eficaz para esta columna.

TABLA CITA_MEDICAMENTO_NUEVO

En la tabla **CITA_MEDICAMENTO_NUEVO** decidimos hacer uso de los índices no agrupados, uno para el **ID CITA**, como se mencionó anteriormente en la tabla **cita_estudio** y el otro para el **ID MEDICAMENTO**, todo esto ya que igualmente se mencionó que estos datos pueden estar ordenados de manera aleatoria, pero tienen ordenamiento lógico de manera interna del gestor, y puede que se estén alterando constantemente. Esto se usa para incrementar la eficacia en algunas instrucciones del tipo WHERE o JOIN.

TABLA CITA_MEDICAMENTO_ANTERIOR

En la tabla **CITA_MEDICAMENTO_ANTERIOR** decidimos hacer uso de los índices no agrupados, uno para el **ID CITA**, como se mencionó anteriormente en la tabla **cita_estudio** y el otro para el **ID MEDICAMENTO**, todo esto ya que igualmente se mencionó que estos datos pueden estar ordenados de manera aleatoria, pero tienen ordenamiento lógico de manera interna del gestor, y puede que se estén alterando constantemente. Esto se usa para incrementar la eficacia en algunas instrucciones del tipo WHERE o JOIN.

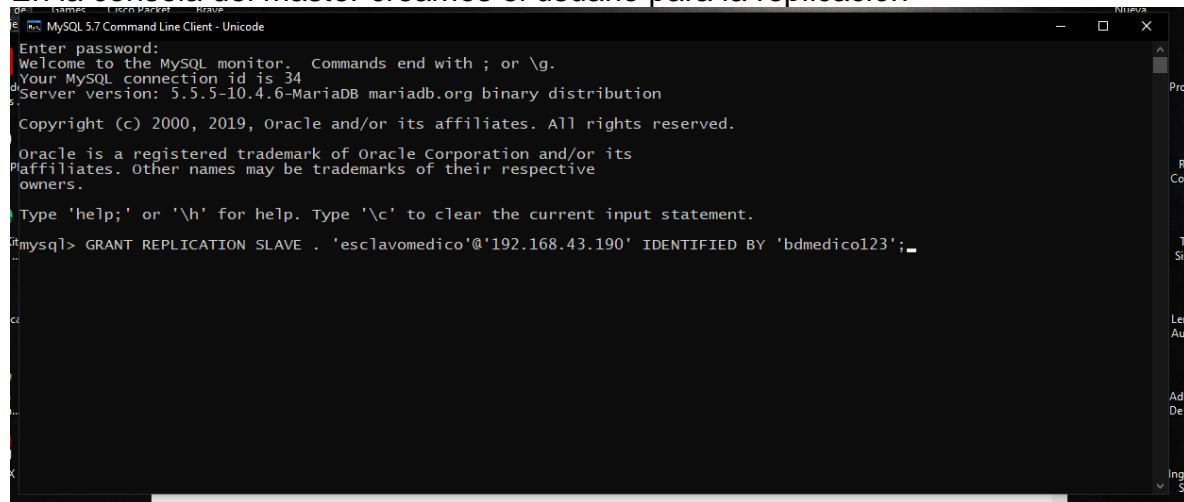
TABLA MEDICAMENTO

En esta tabla se decidió usar los índices agrupados para la columna **ID MEDICAMENTO**, esto lo hará más eficaz al realizar ciertas consultas ya que es gestor las interpretara mejor con este tipo de índices. Los datos vienen en forma ascendente y por esto no habrá problema al realizar nuevas inserciones. Además, la columna **NOMBRE** usará igualmente un índice agrupado, igualmente porque se organiza en orden alfabético(ascendente) y nos servirá para la eficacia.

Replicación

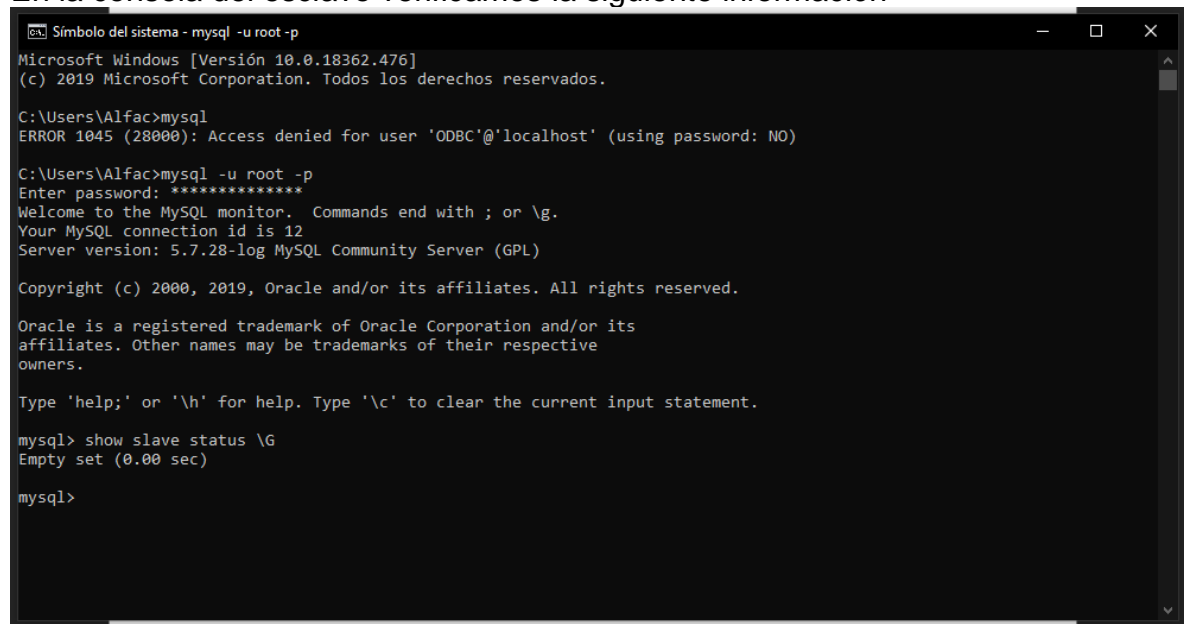
Para la replicación de la base de datos, usamos el gestor de base de datos de MySQL en su versión 5.7, para esto hicimos configuraciones en la consola y en los archivos de configuración del gestor, primero conectamos la computadora esclava con el master, y una vez conectadas abrimos la terminal en las computadoras y configuramos lo siguiente.

En la consola del master creamos el usuario para la replicación



```
MySQL 5.7 Command Line Client - Unicode
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 34
Server version: 5.5.5-10.4.6-MariaDB mariadb.org binary distribution
Copyright (c) 2000, 2019, Oracle and/or its affiliates. All rights reserved.
Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
mysql> GRANT REPLICATION SLAVE . 'esclavomedico'@'192.168.43.190' IDENTIFIED BY 'bdmedico123';
```

En la consola del esclavo verificamos la siguiente información



```
Símbolo del sistema - mysql -u root -p
Microsoft Windows [Versión 10.0.18362.476]
(c) 2019 Microsoft Corporation. Todos los derechos reservados.

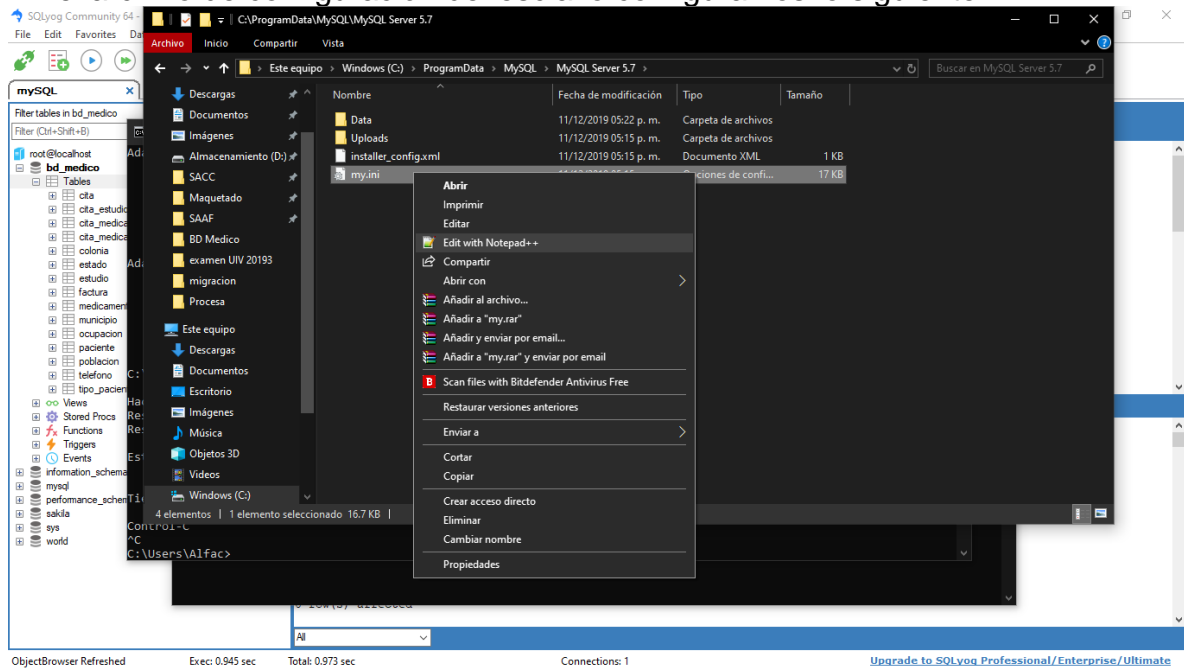
C:\Users\Alfac>mysql
ERROR 1045 (28000): Access denied for user 'ODBC'@'localhost' (using password: NO)

C:\Users\Alfac>mysql -u root -p
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 12
Server version: 5.7.28-log MySQL Community Server (GPL)
Copyright (c) 2000, 2019, Oracle and/or its affiliates. All rights reserved.
Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show slave status \G
Empty set (0.00 sec)

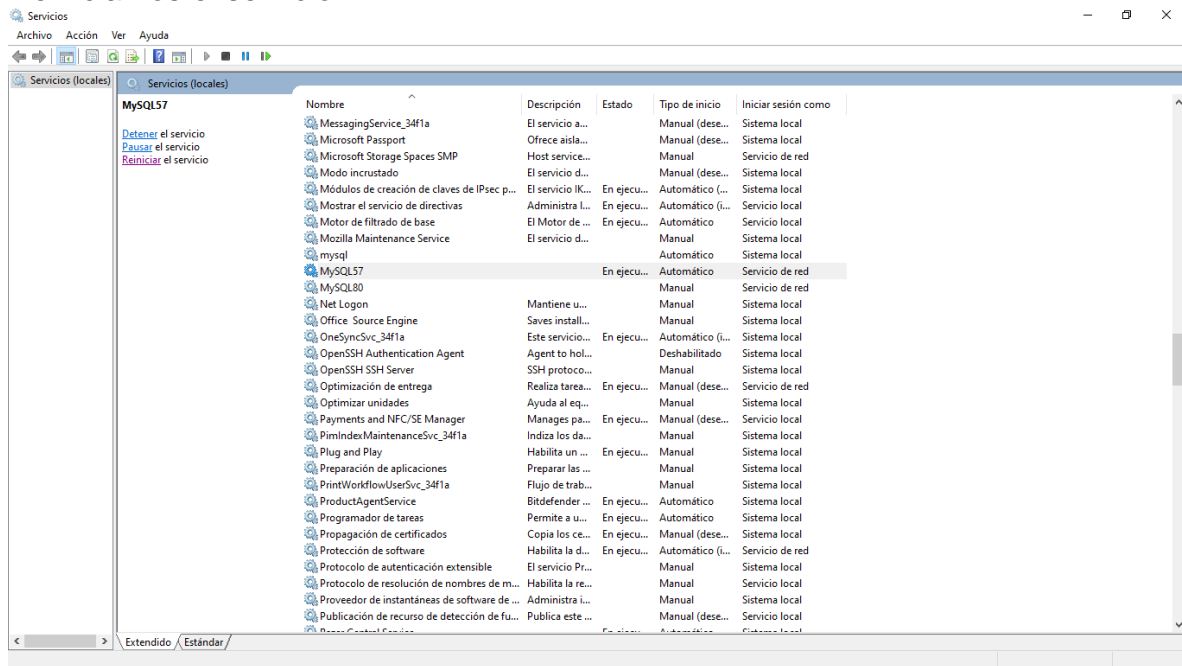
mysql>
```

En el archivo de configuración del esclavo configuramos lo siguiente.

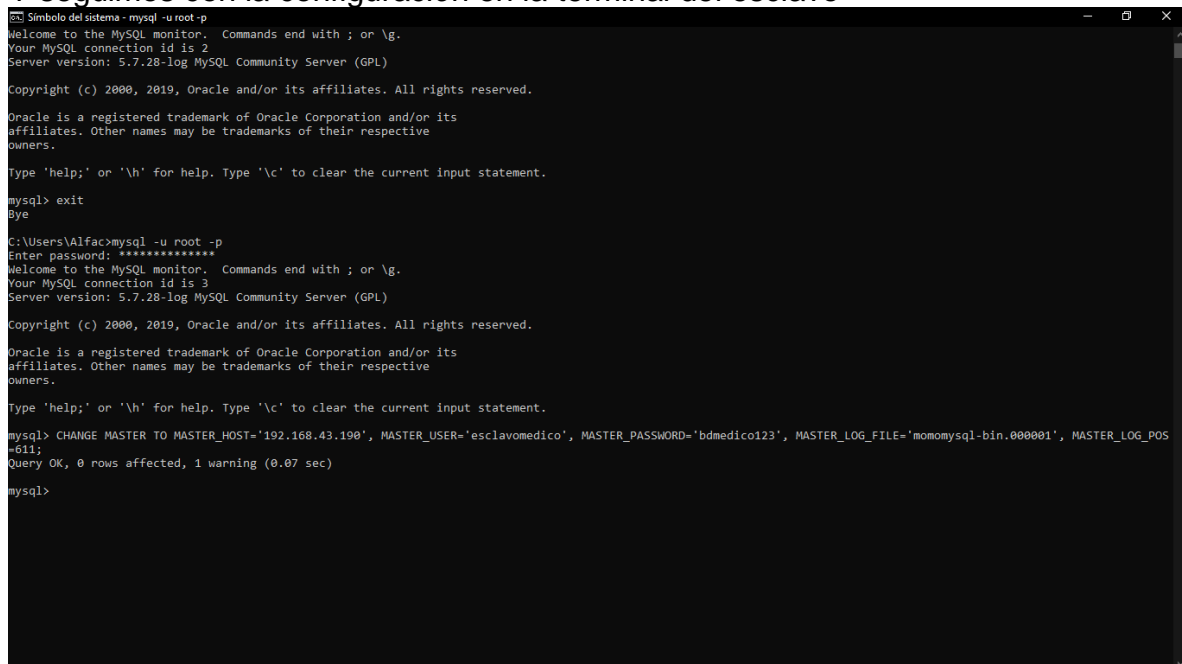


```
# ***** Group Replication Related *****
# Specifies the server ID. For servers that are used in a replication topology,
# you must specify a unique server ID for each replication server, in the
# range from 1 to 2^32 - 1. "Unique" means that each ID must be different
# from every other ID in use by any other replication master or slave.
server-id=2
binlog_do_db=bd_medico
# ***** Group Replication Related *****
# The host name or IP address of the slave to be reported to the master
# during slave registration. This value appears in the output of SHOW SLAVE HOSTS
# on the master server. Leave the value unset if you do not want the slave to
# register itself with the master.
# report_host=0.0
# ***** Group Replication Related *****
```

Reiniciamos el servicio



Y seguimos con la configuración en la terminal del esclavo



```

mysql> CHANGE MASTER TO MASTER_HOST='192.168.43.190', MASTER_USER='esclavomedico', MASTER_PASSWORD='bdmedico123', MASTER_LOG_FILE='momomysql-bin.000001', MASTER_LOG_POS=611;
Query OK, 0 rows affected, 1 warning (0.07 sec)

mysql> show slave status \G
***** 1. row *****
      Slave_IO_State:
        Master_Host: 192.168.43.190
        Master_User: esclavomedico
        Master_Port: 3306
        Connect_Retry: 60
        Master_Log_File: momomysql-bin.000001
        Read_Master_Log_Pos: 611
        Relay_Log_File: AYANAMI-relay.000001
        Relay_Log_Pos: 4
        Relay_Master_Log_File: momomysql-bin.000001
        Slave_IO_Running: No
        Slave_SQL_Running: No
        Replicate_Do_DB:
        Replicate_Ignore_DB:
        Replicate_Do_Table:
        Replicate_Ignore_Table:
        Replicate_Wild_Do_Table:
        Replicate_Wild_Ignore_Table:
        Last_Errno: 0
        Last_Error:
        Skip_Counter: 0
        Exec_Master_Log_Pos: 611
        Relay_Log_Space: 154
        Until_Condition: None
        Until_Log_File:
        Until_Log_Pos: 0
        Master_SSL_Allowed: No
        Master_SSL_CA_File:
        Master_SSL_CA_Path:
        Master_SSL_Cert:
        Master_SSL_Cipher:
        Master_SSL_Key:
        Seconds_Behind_Master: NULL
        Master_SSL_Verify_Server_Cert: No
        Last_IO_Errno: 0
        Last_IO_Error:
        Last_SQL_Errno: 0
        Last_SQL_Error:
        Replicate_Ignore_Server_Ids:

```

```

mysql> start slave;
Query OK, 0 rows affected (0.03 sec)

mysql> show slave status \G
***** 1. row *****
      Slave_IO_State: Connecting to master
        Master_Host: 192.168.43.190
        Master_User: esclavomedico
        Master_Port: 3306
        Connect_Retry: 60
        Master_Log_File: momomysql-bin.000001
        Read_Master_Log_Pos: 611
        Relay_Log_File: AYANAMI-relay.000001
        Relay_Log_Pos: 4
        Relay_Master_Log_File: momomysql-bin.000001
        Slave_IO_Running: Connecting
        Slave_SQL_Running: Yes
        Replicate_Do_DB:
        Replicate_Ignore_DB:
        Replicate_Do_Table:
        Replicate_Ignore_Table:
        Replicate_Wild_Do_Table:
        Replicate_Wild_Ignore_Table:
        Last_Errno: 0
        Last_Error:
        Skip_Counter: 0
        Exec_Master_Log_Pos: 611
        Relay_Log_Space: 154
        Until_Condition: None
        Until_Log_File:
        Until_Log_Pos: 0
        Master_SSL_Allowed: No
        Master_SSL_CA_File:
        Master_SSL_CA_Path:
        Master_SSL_Cert:
        Master_SSL_Cipher:
        Master_SSL_Key:
        Seconds_Behind_Master: NULL
        Master_SSL_Verify_Server_Cert: No
        Last_IO_Errno: 0
        Last_IO_Error:
        Last_SQL_Errno: 0
        Last_SQL_Error:
        Replicate_Ignore_Server_Ids:
        Master_Server_Id: 0
        Master_UUID:

```

Monitoreo

Para monitorear la base de datos decidimos usar el programa de software libre **MyTop**, que, aunque solo esta disponible para distribuciones Linux, decidimos usar este ya que casi no hay soluciones en Windows, y tomando en cuenta que la mayoría de los servidores donde están las bases de datos, son Linux es una solución factible y realista a este problema.

Mytop es un programa de monitoreo de fuente abierta y gratuito para las bases de datos MySQL y MariaDB, escrito por Jeremy Zawodny usando el lenguaje Perl . Es muy similar en apariencia a la herramienta de monitoreo de sistema Linux más famosa llamada top.

El programa Mytop proporciona una interfaz de shell de línea de comandos para monitorear subprocesos MySQL / MariaDB en tiempo real, consultas por segundo, lista de procesos y rendimiento de bases de datos, y da una idea para que el administrador de la base de datos optimice mejor el servidor para manejar cargas pesadas. Por defecto, la herramienta Mytop se incluye en los repositorios de Fedora y Debian / Ubuntu, por lo que solo tiene que instalarlo usando su administrador de paquetes predeterminado.

Si está utilizando distribuciones RHEL / CentOS , entonces necesita habilitar el repositorio de EPEL de terceros para instalarlo.

COMO LO USAMOS.

1. Instalamos mytop que será nuestro monitor de actividades de mysql:

```
sudo apt install mytop
```

2. Una vez instalado y configurado debemos iniciarlo con nuestro usuario y contraseña de la base de datos:

```
mytop --prompt
```

Anexamos capturas de los procesos, sesión de usuario, memoria y consultas que se están ejecutando:

```
parallels@parallels-vm: ~
1 row in set (0.00 sec)

mysql> insert into estado values(1, 'michoacan');
Query OK, 1 row affected (0.00 sec)

mysql> insert into estado values(2, 'Guerrero');
Query OK, 1 row affected (0.00 sec)

mysql> insert into estado values(2, 'Jalisco');
ERROR 1062 (23000): Duplicate entry '2' for key 'PRIMARY'
mysql> insert into estado values(3, 'Jalisco');
Query OK, 1 row affected (0.00 sec)

mysql> select * from estado;
+-----+-----+
| id_estado | nombre |
+-----+-----+
|          1 | michoacan |
|          2 | Guerrero |
|          3 | Jalisco |
+-----+-----+
3 rows in set (0.00 sec)

mysql> █
```

Terminal

```
parallels@parallels-vm: ~
MySQL on localhost (5.7.28)                               load 0.00 0.02 0.05 1/478 8831 up 0+00:31:
12 [22:49:45]
Queries: 943.0 qps: 1 Slow: 0.0 Se/In/Up/De(%): 06/01/00/00
Sorts: 0 qps now: 1 Slow qps: 0.0 Threads: 4 ( 1/ 0) 00/00/00/00
Key Efficiency: 77.3% Bps in/out: 17.8/ 1.5k Now in/out: 22.7/ 2.6k

  Id  User      Host/IP      DB      Time  Cmd  State Query
  --  ---
  7   root localhost:38204  --      608  Sleep
  8   root localhost:38206  prueba  608  Sleep
  15  root  localhost  bd_medico  107  Sleep
  14  root  localhost      0     Query starting show full processlist
```

Mytop

Processlist

```
mysql> show processlist;
```

Id	User	Host	db	Command	Time	State	Info
7	root	localhost:38204	NULL	Sleep	773		NULL
8	root	localhost:38206	prueba	Sleep	773		NULL
15	root	localhost	bd_medico	Query	0	starting	show processlist