



UNIVERSIDAD TÉCNICA DE BABAHOYO

FACULTAD DE ADMINISTRACIÓN, FINANZAS E INFORMÁTICA.

PROCESO DE TITULACIÓN

DICIEMBRE 2021- ABRIL 2022

EXAMEN COMPLEXIVO DE GRADO O DE FIN DE CARRERA

PRUEBA PRÁCTICA

INGENIERÍA EN SISTEMAS

PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN SISTEMAS

TEMA:

**ESTUDIO COMPARATIVO ENTRE BASES DE DATOS RELACIONAL Y NO
RELACIONAL**

EGRESADO:

LUIS ARTURO SALTOS PEREZ

TUTOR:

ING. FREDY MAXIMILIANO JORDÁN CORDONES

AÑO 2022

Resumen

La plataforma DB-Engines representa información clave sobre el nivel de popularidad que existe entre los dos tipos de bases de datos, con base en los datos proporcionados se identificó que la base de datos relacional cubre un gran segmento en el mercado de bases de datos. Esto se debe a que es una tecnología antigua que se enfoca en soluciones de almacenamiento estructurado.

Además, también muestra una perspectiva de los diferentes modelos de bases de datos no relacionales, siendo una tecnología que se encuentra en pleno crecimiento, se observó cierto grado de aceptación por tener un alto grado de compatibilidad en nuevas aplicaciones que requieren almacenar datos no estructurados de grandes bancos de información.

En la base de datos relacional es necesario establecer una estructura en forma de tabla para almacenar los datos. Donde cada tabla contiene una clave primaria que se relaciona con las demás tablas por medio de una clave secundaria para facilitar el acceso a los datos. Las bases de datos no relacionales están diseñadas para operar en una arquitectura distribuida que facilita la replicación de la información en diferentes centros de almacenamiento, no utilizan una estructura relacionada definida ya que implementan APIs para acceder a la información.

El estudio comparativo entre bases de datos relacionales y no relacionales se realiza con el objetivo de brindar información sobre las principales características, ventajas y desventajas que constituyen cada modelo de base de datos relacional y no relacional, para posteriormente establecer un cuadro comparativo que brinde información a futuras investigaciones.

Palabras clave: SQL, base de dato relacional, base de datos no relacional, modelo Acid, modelo Base.

Summary

The DB-Engines platform represents key information about the level of popularity that exists between the two types of databases, based on the data provided it was identified that the relational database covers a large segment in the database market. This is because it is an old technology that focuses on structured storage solutions.

In addition, it also shows a perspective of the different models of non-relational databases, being a technology that is in full growth, a certain degree of acceptance was observed for having a high degree of compatibility in new applications that require storing unstructured data of large information banks.

In the relational database it is necessary to establish a structure in the form of a table to store the data. Where each table contains a primary key that is related to the other tables by means of a secondary key to facilitate access to the data. Non-relational databases are designed to operate in a distributed architecture that facilitates the replication of information in different storage centers, they do not use a defined related structure since they implement APIs to access information.

The comparative study between relational and non-relational databases is carried out with the aim of providing information on the main characteristics, advantages and disadvantages that constitute each relational and non-relational database model, to subsequently establish a comparative table that provides information to Future investigations.

Keywords: SQL, relational database, non-relational database, acid model, base model.

INTRODUCCIÓN

La investigación actual estará encaminada en el desarrollo de una comparativa a nivel de bases de datos tomando como referencia el modelo base de datos relacional y no relacional, que son unos de los principales medios de almacenamiento que sean utilizado para guardar información a lo largo de la historia. Dichos datos pueden ser almacenados y gestionados mediante la implementación de instrucciones en un gestor de base de datos o APIs.

Es importante recalcar que por los nuevos avances tecnológicos que se están dando hoy en día muchos de los desarrolladores y empresas están optando por utilizar base de datos relacional y no relacional con el objetivo de aportar soluciones que contribuyan en una mejora en la administración de los datos.

Por ello el modelo de base de datos relacional proporciona soluciones robustas donde se requiere aplicar un almacenamiento estructurado de la información, es ahí donde surgen los principales problemas ya que no son compatibles con los nuevos modelos de desarrollo en las nuevas aplicaciones debido a que no están sujetas a cambios repentinos en su diseño, no cuentan con una estructura distribuida, consumen muchos recursos de hardware, necesitan ser desconectadas para solucionar los errores. Mientras que en las bases de datos no relacionales los principales problemas están relacionados con la inconsistencia de la información, no implementan reglas para la normalización de datos y tienden a presentar problemas de inestabilidad.

Por ende, estos son unos de los principales problemas que se deben tomar en consideración para la aplicación de una solución de base de datos independientemente del área de desarrollo donde se necesite aplicar una solución específica ya que cada modelo de base de datos relacional y no relacional aporta funcionalidades diferentes que pueden ser una ventaja o desventaja en el desarrollo de un sistema o aplicación.

Por esta razón el principal objetivo será realizar un estudio comparativo entre bases de datos relacional y no relacional con la finalidad de conocer sus principales estructuras, características, ventajas y desventajas que estas aportarían para la gestión de los datos dependiendo del tipo de información que se pretenda manipular o analizar.

Por consiguiente, la línea de investigación en la cual estará enfocado la investigación es la línea de sistemas de información y comunicación, emprendimiento e innovación y la sublínea es redes y tecnologías inteligentes de software y hardware.

El método de investigación que se empleará será el método analítico ya que este nos facilitara descomponer la información recopilada en diferentes fragmentos que serán primordiales para establecer las diferencias que aportarían en el desarrollo del estudio comparativo. Además, se empleará la investigación explicativa ya que esta permite describir y establecer relaciones de las principales funcionalidades con las que están constituidas las bases de datos relacional y no relacional.

Por ello la investigación bibliográfica o documental facilitará la interpretación y análisis de la información obtenida de las diversas fuentes documentales y bibliográficas que permitirán generar nuevas ideas que contribuyan en el desarrollo del caso de estudio sobre los cuales se sustentará los objetivos planteados en la investigación.

DESARROLLO

La principal necesidad de elaborar este estudio comparativo de bases de datos es que muchas veces los programadores entran en un dilema sobre cuál será el mejor modelo de bases de datos que empleará para dar solución a un problema del mundo real ya que se desconoce de todas las funcionalidades que posee el modelo de base de datos relacional y no relacional.

Es por eso que se debe tener en consideración cuales son los principales problemas que presentan cada base de datos. La base de datos relacional al ser una tecnología antigua necesita de una gran infraestructura para ponerlas en funcionamiento, solo se las puede implementar en soluciones donde se necesite tener un almacenamiento estructurado de los datos lo cual con lleva a tener problemas de compatibilidad con el desarrollo de las nuevas aplicaciones donde se necesita efectuar cambios repentinos en su diseño. La base de datos no relacionales suele presentar problemas de inconsistencia en la información, no soportan transacciones complejas que se sustenten en el modelo ACID.

El surgimiento de las bases de datos data de la época antigua cuando los trabajadores se encargaban de recopilar y guardar la información en un almacén de bibliotecas que simulaba un sistema de gestión de almacenamiento que permitía guardar los datos obtenidos de cosechas de los cultivos y censos poblacionales. Sin embargo, toda la información de los datos guardados en esos registros no era posible ser accedidos de forma rápida debido a que no se contaba con una herramienta automatizada que facilite la gestión de esos datos ya todo el proceso de extracción de la información se efectuaba de forma manual. (Primera base de datos, 2021)

La primera base de datos dio su aparición en el mercado en 1963 cuando un grupo de informáticos estaban celebrando una reunión en la ciudad de California ya que ellos pretendían buscar nuevos métodos que les permitiese almacenar los datos de sus investigaciones con el objetivo de crear un conjunto de datos agrupados de forma estructurada y ordenados. (Joaquinaraujo, s.f.)

En 1970 el informático Edgar Frank Codd fue uno de los pioneros en encaminar la teoría de las bases de datos relacional cuyo significado está basado en un modelo de datos relacional para grandes bancos de datos compartidos que implementa relaciones basado en filas, las cuales

aportarían información sobre las cual estarían sentado los cimientos que darían el surgimiento de los sistemas gestores de bases de datos relacional. (Ferraris, 2010)

En los ochenta se creó el primer lenguaje de consultas que estaría basado en el modelo relacional que introdujo Edgar Frank Codd, este permitía efectuar consultas complejas de datos específicos, cambiar la estructura de la base de datos. Dicho lenguaje fue introducido al mercado por la empresa Oracle en los años 1979 logrando una gran acogida por las operaciones que permitía ejecutar mediante la utilización de comandos SQL. (Llaca, 2017)

En el año de 1998 se introdujeron las bases de datos no relacional en el mercado por el científico Carlo Strozzi quien fue el que acuñó la terminología NoSQL que se refería a un tipo de almacenamiento de código abierto en donde no se tenía en consideración un lenguaje estructurado para la manipulación de los datos. Debido al crecimiento de las grandes plataformas que hoy en día se conocen como Facebook, Twitter, Youtube permitieron que el nuevo modelo de base de datos no relacional tenga mayor acogida ya que estas les brindan una mayor escalabilidad, accesibilidad en cuanto al almacenaje de grandes volúmenes de información. (Base de datos NOSQL, 2017)

Base de datos

Se conoce como base de dato a todo almacén lógico que permita almacenar la información del mismo tipo de forma estructurada y organizada de tal manera que esta pueda ser gestionada a través de un programa de cómputo mediante la utilización de comandos que permiten realizar las principales operaciones de un CRUD.

Mosquera Palacios, Wanumen Silva, & Rivas Trujillo (2018) indicó: Una base de datos es una colección de datos almacenados de modo que no exista la posibilidad de duplicación de información innecesaria y que puede ser accesada desde muchas aplicaciones por medio de métodos o funciones ya preestablecidas para la inserción, captura o eliminación de información de la colección mencionada. (pág. 1)

Sistema gestor de base de datos

Es un software que es utilizado como una herramienta para crear la estructura de una base de datos mediante la implementación de un lenguaje estructurado de consulta que facilita la accesibilidad, extracción y la administración de los datos almacenado de una base de datos.

Regalado Jalca, García, & Gómez (2017) ha afirmado los siguiente: Un sistema de gestión de bases de datos (SGBD) consiste en una colección de datos interrelacionados y

un conjunto de programas para acceder a dichos datos. El objetivo principal de un SGBD es proporcionar una forma de almacenar y recuperar la información de una base de datos de manera que sea tanto práctica como eficiente. (pág. 11)

Base de dato relacional

Es una base de dato que implementa el modelo relacional para establecer las relaciones entre los diversos tipos de datos que son representados mediante tablas. Es decir, cada tabla está compuesta por una fila que vendría hacer un registro que representa una clave principal, cada columna los atributos, los registro tiene un valor por cada uno de los atributos lo que permitiría establecer cada una de las relaciones que existen entre clave principal y clave foránea. (Oracle, 2022)

SQL

Es un lenguaje de alto nivel considerado como un estándar para trabajar con las bases de datos relacionales con la finalidad de ser utilizado como un lenguaje declarativo de la estructura de una base de datos permitiéndole a los programadores administrar y extraer la información de los datos mediante la implementación de sentencias y cláusulas. (Silveira, 2021)

El autor (Narro, 2017) afirma que el “SQL está compuesto por sentencias, cláusulas, operadores y funciones de agregado para la manipulación de elementos en una base de datos, las sentencias SQL se dividen en cuatro tipos de lenguajes de manipulación de datos, control de datos, control de transacciones”.

DDL: es un lenguaje que se lo emplea para definir los objetos por los cuales estará conformada la base de datos, incluye ordenes como: create, alter, drop, truncate.

DML: es un lenguaje de control que permite manipular los registros de una tabla a través de las sentencias: insert, select, update, delete, merge.

TCL: es un lenguaje de control de transacciones que se lo implementa para el control y la gestión de las transacciones, permite sentencias como: commit, rollback.

DCL: es un lenguaje de control orientado principalmente al control de los accesos de los usuarios sobre los diversos objetos que conforman la base de datos, entre ellos se encuentran: grant, revoke.

Modelo ACID

Es uno de los principales modelos sobre los cuales se basa la administración de las bases de datos relacionales y sobre las cuales se sustentan las transacciones de una base de datos relacional, entre ellas se nombran las siguientes propiedades:

Atomicidad: “Todas las operaciones se realizan, o que ninguna de ellas se realiza, de modo que, si una parte de la transacción falla, entonces toda la transacción falla”. (International, 2017)

Consistencia: “Permite mantener la integridad de la información al inicio, durante, al finalizar la transacción según las reglas establecidas. Es decir, ningún campo que sea modificado deberá contener un tipo de dato que no esté especificado en el create table”. (DBA dixit, 2020)

Aislamiento: “El aislamiento garantiza la individualidad de cada transacción, evita que se vean afectados por otras transacciones. Garantiza que las transacciones se procesen de forma segura e independiente al mismo tiempo sin interferencias, pero no garantiza el orden de las transacciones” (Socrates, 2022)

Durabilidad: “Una transacción debe ser recuperable, por tanto, debe tener durabilidad. Si se confirma una transacción, el sistema garantiza que sus actualizaciones pueden persistir incluso si el equipo se bloquea inmediatamente después de la confirmación”. (Microsoft, 2021)

Tipos de bases de datos relacionales

En la actualidad existen un sin número de bases de datos relacionales, pero para conocer las importantes nos enfocaremos en los resultados de la plataforma DB-Engines ya que esta nos proporciona un ranking por año de cuáles son las más utilizadas en el mercado de bases de datos relacional.

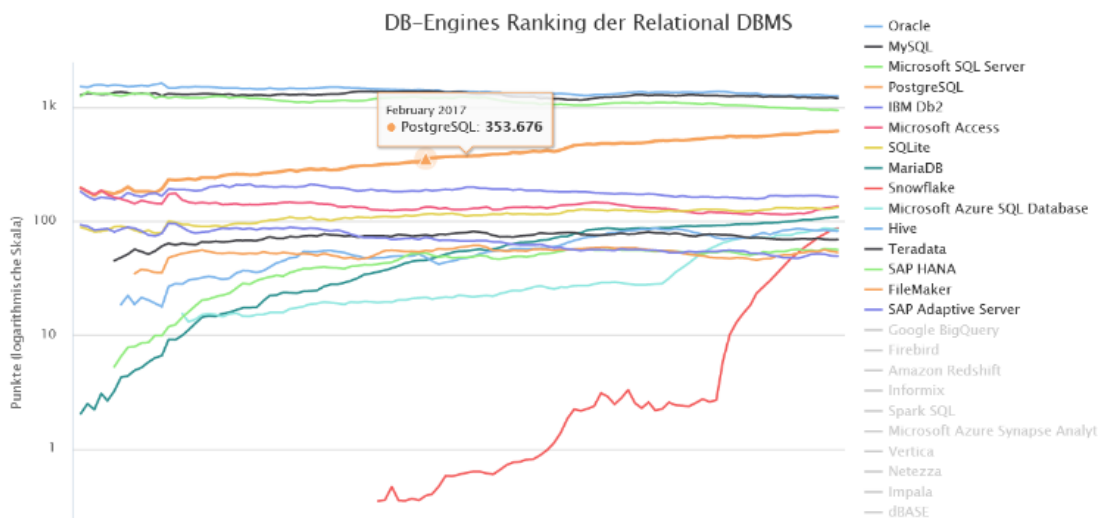


Ilustración 1. Ranking de las bases de datos relacionales más usadas por año.

Tomada de la plataforma (DB-ENGINES, 2022)

Tabla 1. Bases de datos relacionales

	Oracle	MySQL	Microsoft SQL Server	PostgreSQL
Tipo de licencia	Privativa	GLP o Uso comercial	Acceso de cliente + servidor Licencia (CAL). Basada en núcleos	PostgreSQL License
Desarrollado	Oracle Corporation	Oracle Corporation	Microsoft	PostgreSQL Global Development Group
Lanzamiento	1977	1994	1989	1996
Lenguaje de fabricación	C++ Java C	C C++	C C++	C
Lenguaje de programación que soporta	PL/SQL. Java. C. C++.	C. C++. C#. Pascal. Python. Ruby. Gambas. REALbasic. Delphi. Eiffel. Smalltalk. Java.	C# con ADO.NET. Entity Framework (EF) con C#. Java y JDBC. Node.js. ODBC para C++. PHP. Python. Ruby.	C. C++. Java Pl/Java web. Php. Python. Ruby. Sh. Tcl. Perl. R.

		Lisp. Perl. PHP.		
Versión	21c	8.0.25	15.0	14.2
Plataforma	AIX. HP-UX. Linux. OS X. Solaris. Windows. z/OS.	FreeBSD. Linux. OS X. Solaris. Windows.	Windows.	Solaris. Unix. Windows. FreeBSD. HP-UX. Linux. NetBSD. OpenBSD. OS X.
Características	Brinda una interfaz gráfica para la administración. Proporciona tecnologías para el control de acceso. Posee un lenguaje para el diseño de la base de datos. Está pensada con una orientación hacia el internet.	Permite agrupar las transacciones. Se puede efectuar una replicación. Está disponible para diversas plataformas. Permite realizar búsquedas e indexación para campos de texto.	Permite elegir el lenguaje y plataforma. Brinda inteligencia con clústeres de big data. Proporciona un tipo de cifrado en los datos. Brinda compatibilidad con cualquier dispositivo y servicio Azure.	Soporta muchos clientes al mismo tiempo. Permite crear Triggers. Es capaz de soportar consultas en vistas de diferentes maneras al modo en que son almacenadas. Tiene soporte para bases de datos distribuidas.

<p style="text-align: center;">Ventajas</p>	<p>Se puede ejecutar en diferentes plataformas.</p> <p>Soporta funciones de lado del servidor.</p> <p>Permite la administración de bases de datos distribuidas.</p>	<p>Ofrece un alto rendimiento al momento de efectuar las operaciones.</p> <p>Su instalación no es complicada.</p> <p>Brinda una encriptación a los datos.</p> <p>Es un software gratuito.</p>	<p>Brinda escalabilidad, estabilidad y seguridad.</p> <p>Soporta procedimientos de almacenado.</p> <p>Permite la ejecución de permisos de administración de todo tipo.</p>	<p>Se ejecuta en diversas plataformas.</p> <p>Opera con el lenguaje SQL.</p> <p>Es fiable y robusto.</p> <p>La configuración es sencilla de realizar.</p>
<p style="text-align: center;">Desventajas</p>	<p>Las licencias son muy costosas.</p> <p>El soporte y mantenimiento es muy costoso.</p> <p>No existen manuales sobre cómo usarlo a no ser que se compre la licencia.</p>	<p>Falta documentación de diversas funciones ya que no está documentada por ser gratuito.</p> <p>Al momento de efectuar consultas en las vistas la operación se realiza con lentitud.</p>	<p>La licencia es muy costosa.</p> <p>Consume mucha RAM.</p> <p>Tiene muchas restricciones.</p> <p>Solo se pueden ejecutar en servidores de Windows.</p>	<p>Es muy lento cuando se están escribiendo datos en una base de datos pequeñas.</p> <p>Solo se lo puede emplear en ambientes en donde se requiera de un alto volumen de información.</p> <p>La sintaxis de</p>

				algunos comandos es demasiado compleja.
Script en el servidor	Si	Si	Transact SQL. Lenguajes .NET. R. Python. Java.	Funciones definidas por el usuario
Triggers	Si	Si	Si	Si
Api y método de acceso	DBC. ODBC. ODP.NET. Interfaz de llamadas de Oracle (OCI).	ADO.NET. JDBC. ODBC. API nativa.	ADO.NET. JDBC. ODBC. OLE DB. Flujo de datos tabulares (TDS).	API nativa de PostgreSQL en PHP.
Modelo	ACID	ACID	ACID	ACID

Elaborado por: Luis Saltos Perez

Bases de datos no relacional

Las bases de datos no relacionales también son conocidas como NoSQL en este tipo de base de datos no se necesita especificar la estructura para poder almacenar los datos, no se requiere de un identificador que apunte a las diferentes relaciones que existen entre tablas ya que la información esta organizada en documentos en formato JSON, tampoco utilizan un lenguaje estructurado para manipular los datos. (Yecid Alexis Rendón, 2019)

Dentro de esta familia de bases de datos no relacionales destacan los siguientes tipos:

Clave-valor

Esta base de dato se caracteriza por implementar un método básico de clave-valor en donde cada grupo de datos se almacenan en pares de clave-valor cuya clave representa un indicador único y en cada una de ella se le asigna un valor que puede ser un elemento simple o elementos compuesto complejos. (Aws, 2022)

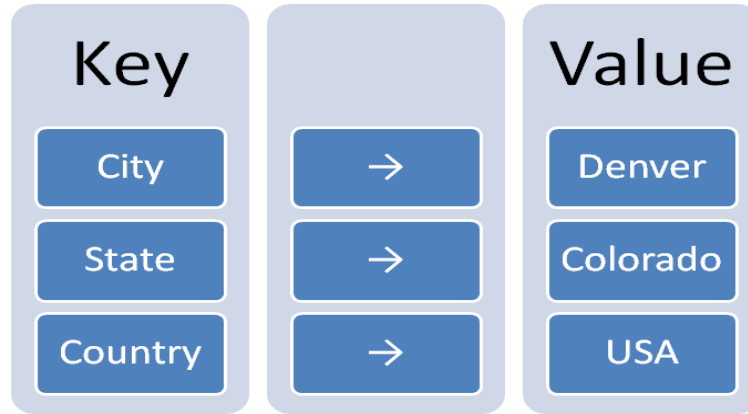


Ilustración 2. Esquema de base de datos de clave-valor. Tomada de (Williams, 2021)

Documentos

Es una base de datos que esta orientada principalmente al manejo de datos mediante el uso de documentos JSON y en cada uno de ellos se le asigna un identificador único para cada registro, los datos están agrupados en un solo documento en donde cada uno de los documentos contiene datos semiestructurados.

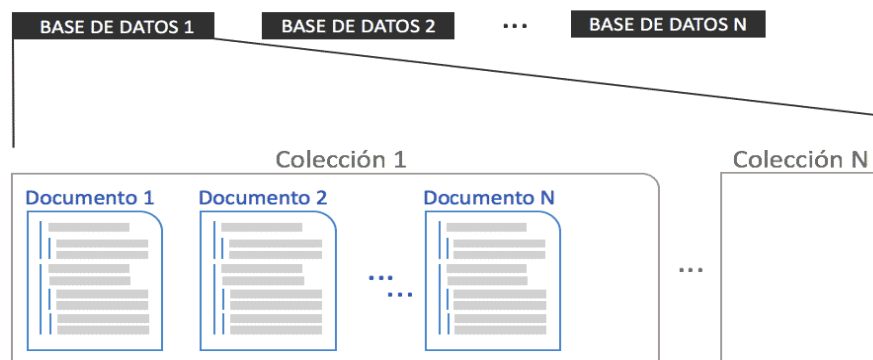


Ilustración 3. Esquema de base de datos de documentos. Tomada de (Calvo, 2017)

Grafo

Según Redacción KeepCoding (2022) define que: “La base de datos grafo representan la información como objetos, en este caso, los datos se representan como grafos, es decir, una estructura de vértices y aristas que simbolizan los conjuntos de datos y las relaciones que existen entre ellos”.

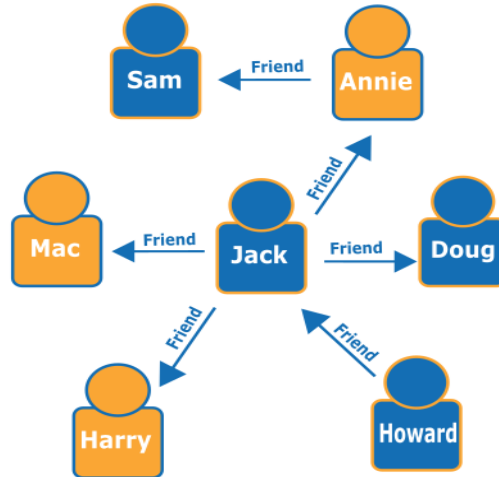


Ilustración 4. Esquema de base de datos de grafo. Tomada de (Aws, 2022)

Columna

Según Macarrón (2021) define: “Que son bases de datos de columnas anchas similar a las bases de datos relacionales, están pensadas para requerir menos consultas a la hora de recuperar datos puesto que guardan todos los datos relacionados en grupos, teniendo cada familia de columnas un conjunto de columnas con una relación lógica entre ellas”

Row Key	Column Family			
Row Key	Customers		Products	
Customer ID	Customer Name	City & Country	Product Name	Price
1	Sam Smith	California, US	Mike	\$500
2	Arijit Singh	Goa, India	Speakers	\$1000
3	Ellie Goulding	London, UK	Headphones	\$800
4	Wiz Khalifa	North Dakota, US	Guitar	\$2500

Ilustración 5. Esquema de base de datos de columnar. Tomada de (Sinha, 2021)

Modelo BASE

Es uno de los modelos sobre los cuales se enfocan las bases de datos no relacionales ya que este modelo es más sencillo que el modelo ACID ya que este proporciona mayor flexibilidad para la gestión de los datos no estructurados y sus propiedades son:

Disponibilidad: se refiere a que el almacén de los datos estará operativo todo el tiempo sin importar las múltiples fallas que se presenten esto es debido a que se emplea una distribución

y replicado de los almacenes de datos en diferentes lugares de almacenamiento con la finalidad que si falla cierta fracción esta no represente un problema para toda la base de dato.

Estado relajado: lo que representa son las réplicas de los almacenes de los datos en donde cada replica no tiene que ser similar a la anterior.

Consistencia: lo que permite es tener el almacén de los datos actualizado ante posibles modificaciones y si no existe algún cambio pone a disposición de los usuarios la información de los datos almacenados.

Tabla 2. Bases de datos no relacionales

	Mongodb	Apache Cassandra	Neo4j	Redis
Tipo de licencia	GNU AGPL	Apache License 2.0	Dual AGPLv3/ comercial	BSD
Desarrollado	MongoDB Inc.	Apache Software Foundation	Neo Technology	Salvatore Sanfilippo
Lanzamiento	2009	2008	2007	2009
Modelo de desarrollo	Open source	Open source	Open source	Open source
Lenguaje de fabricación	PHP. Python. Ruby. C++ Builder. C. C++. C# / .NET. Java. JavaScript. Node.js.	Python. Ruby. C/C++. C#. Java. JavaScript. PHP.	Python. Ruby. Scala. Net. Clojure. Elixir. Haskell. Java. JavaScript. Perl. PHP.	Haskell Haxe Java Clojure Crystal D Swift Fancy Go Dart Elixir.

	Perl. Scala. Delphi.		Go. Groovy.	Visual Basic
Versión	4.0.8	3.11.7	3.3.0	6.0.6
Tipo de almacenamiento	Documento	Hash Tables	Grafo	Hash Tables
Formato	JSON, BSON	*****	*****	*****
Plataforma	Multiplataforma	Multiplataforma	Multiplataforma	Multiplataforma
Características	Consultas. Indexación. Replicación. Balanceo de carga. Almacenamiento de archivos. Ejecuta código JavaScript en el servidor.	Descentralizado. Escalabilidad. Tolera fallos. Consistencia. Soporta replicación en varios data center.	Permite ejecutar transacciones. No utiliza esquema. Tiene un buen balanceo de carga en la lectura de datos.	Permite un rápido acceso a los datos. Los datos los estructura en memoria. Tiene un tipo de duplicado asíncrono. Es compatible con más de un lenguaje de programación.
Ventajas	Se puede ejecutar con pocos recursos de computación. Es de código abierto. Soporta estructuras	Tolera fallos. Permite la réplica de los almacenes de datos. Permite tener escalabilidad incremental.	Permite establecer relaciones intercambiables. Permiten tener un mayor rendimiento. Ofrecen	Ofrece una fácil configuración. Brinda alta disponibilidad. Permite usar Lua scripting. Ofrece mayor velocidad

	<p>distribuidas.</p> <p>No hay que definir la estructura de la base de datos.</p> <p>Permite la replicación del almacén de los datos.</p>	<p>Implementa su propio lenguaje CQL.</p>	<p>flexibilidad y escalabilidad.</p> <p>Permiten tener una mejor gestión de los datos.</p>	<p>cuando se consulta de diferentes instancias.</p> <p>Tiene escalabilidad horizontal.</p>
Desventajas	<p>No soportas transacciones complejas.</p> <p>No implementa joins para realizar consultas.</p> <p>No admite documentos con tamaño de más de 16 MB.</p> <p>Solo se puede agrupar 100 documentos.</p>	<p>No soporta joins.</p> <p>No facilita ordenar resultados en tiempo de consulta.</p>	<p>Posee un Cql que no es un lenguaje estandarizado.</p> <p>La replicación de grafo solo lo efectúa en grafos completos.</p> <p>No disponen de tantas herramientas.</p>	<p>Consume demasiada memoria.</p> <p>Si falla los datos se pierden.</p> <p>No ejecuta transacciones anidadas.</p>
Empresas	<p>Facebook.</p> <p>Nokia.</p> <p>eBay.</p> <p>Adobe.</p> <p>Google.</p>	<p>Twitter.</p> <p>Netflix.</p>	<p>Walmart.</p> <p>Telenor.</p> <p>Cisco.</p> <p>UBS.</p> <p>Hewlett-Packard.</p> <p>Lufthansa.</p>	<p>Walmart.</p> <p>Telenor.</p> <p>Cisco.</p> <p>UBS.</p> <p>Hewlett-Packard.</p> <p>Lufthansa.</p>
Script en el servidor	<p>JavaScript</p>	<p>Si</p>	<p>Si</p>	<p>Lua funciones de redis</p>

Triggers	Si	Si	Si	Si
Esquema de datos	Sin esquema	Sin esquema	Sin esquema	Sin esquema
Api y método de acceso	Protocolo propietario usando JSON	Protocolo Bolt. Lenguaje de consulta Cypher. API Java.	API Java nativa de Neo4j. Neo4j Cypher Java API.	RDB y AOF

Elaborado por: Luis Saltos Perez

Comparativa de base de datos relacional y no relacional

Mediante el análisis de popularidad que nos muestra la plataforma DB-Engines se logra tener una perspectiva de cómo está la situación actual en el mercado de las bases de datos relacionales y no relacionales. El grafico permite visualizar por categorías el nivel de popularidad que existe en los diversos tipos de base de datos, entre los cuales destacan los gestores de bases de datos relacional con un porcentaje del 71.8%, esto quiere decir que muchas de las empresas y desarrolladores lo prefieren por que brinda soluciones donde se requiere un almacenamiento estructurado basado en el modelo ACID.

Por otro lado, podemos establecer que las bases de datos no relacionales presentan un nivel de popularidad que esta categorizado por los diferentes modelos que ofrecen estas bases de datos no relacionales.

- Documento el 10%.
- Grafo el 1.8%.
- Key-values el 5.7%.
- Wide column 3.1%.

Estos resultados permiten determinar que en el futuro estas bases de datos no relacionales terminaran por reemplazar el modelo tradicional de base de datos relacional, esto se debe a que es una tecnología que está en pleno desarrollo, es más compatible con el desarrollo de las nuevas aplicaciones que se están lanzando al mercado ya que permite efectuar cambios repentinos en sus diseños, porque utilizan una arquitectura distribuida que facilita la réplica de la información.

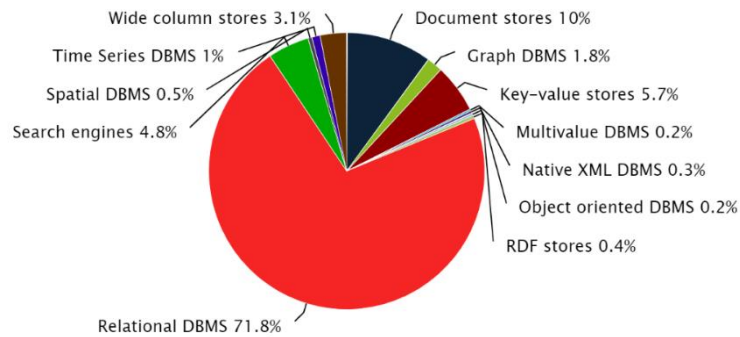


Ilustración 6. Popularidad de diferentes tipos de bases de datos. Tomada de (DB-Engines , 2022)

Tabla 3. Comparativa de base de datos relacional y no relacional

	Base de datos relacional	Base de dato no relacional
Lenguaje de programación	Lenguaje estructurado de consultas SQL	No utilizan un lenguaje de consulta
Modelo de desarrollo	Código abierto, pago	Código abierto, pago
Tipo de almacenamiento	Estructurado	No estructurado
Modelo de datos	Se normalizan los datos en tablas compuestas por filas, columnas	Existen varios modelos de datos documento, clave-valor, grafo, columna

Tipo de arquitectura	Cliente-servidor	Distribuida
Tipo de escalabilidad	Vertical	Horizontal
Propiedades sobre las cuales se sustentan las bases de datos	<p>Modelo ACID es más complejo y está compuesto por propiedades como:</p> <ul style="list-style-type: none"> ➤ Atomicidad. ➤ Uniformidad. ➤ Aislamiento. ➤ Durabilidad. 	<p>Modelo BASE se caracteriza por ser más flexible destacando las propiedades:</p> <ul style="list-style-type: none"> ➤ Disponibilidad. ➤ Estado relajado. ➤ Consistencia.
Características	<ul style="list-style-type: none"> ➤ No permite la duplicación de los datos. ➤ Implementa tablas para el almacenaje de los datos. ➤ Contiene claves que relacionan los datos para poder ser consultados. ➤ Soporta JOINS. 	<ul style="list-style-type: none"> ➤ La información se almacena en colecciones de documentos. ➤ No soportan operaciones de consultas JOINS. ➤ Hacen uso de Apis para el acceso a los datos en formato JSON.
Ventajas	<ul style="list-style-type: none"> ➤ Esta no permite guardar las operaciones que no se completaron antes posibles errores. 	<ul style="list-style-type: none"> ➤ Permite efectuar cambios en tiempo real sin que los demás módulos de la base de datos se vean

	<ul style="list-style-type: none"> ➤ Permite efectuar búsquedas complejas. 	<p>afectado.</p> <ul style="list-style-type: none"> ➤ Permite almacenar cualquier tipo de información sin la necesidad de estar validando. ➤ Esta no utiliza una estructura fija.
Desventajas	<ul style="list-style-type: none"> ➤ Tiene un alto costo de mantenimiento cuando el volumen de la información es demasiado alto. ➤ Si hay cambios en algunos de los registros debemos modificar la estructura o si no presentaría errores. ➤ La instalación suele tener complicaciones dependiendo del tipo del sistema operativo y servidor. ➤ Presentan una interfaz compleja para poder manipular los datos mediante comandos. 	<ul style="list-style-type: none"> ➤ Pueden presentar inconsistencias en la información almacenada. ➤ No presentan una interfaz gráfica que facilite la manipulación de los datos se necesita de conocimientos avanzados. ➤ No permite efectuar transacciones. ➤ Los datos no están relacionados por claves que apunte a una tabla. ➤ Presentan problemas de inestabilidad.
Api	Para acceder a los datos se implementa el lenguaje de consultas	Se acceden a los datos mediante APIs que están orientadas en objetos

Rendimiento	Se necesita tener bien optimizadas las consultas, índices y estructura de tabla para lograr el máximo rendimiento	Depende del tamaño del clúster de hardware subyacente, la latencia de red y la aplicación que efectúa la llamada
-------------	---	--

Elaborado por: Luis Saltos Perez

Conclusión

Se pudo constatar en base a la información proporcionada en el desarrollo de la investigación las siguientes conclusiones:

- Se identificó que las bases de datos relacionales tienen un gran dominio en el mercado actual de bases de datos debido a que aportan soluciones robustas en entornos donde se requiera almacenar información estructurada.
- Se observó que todas las operaciones que se efectúen dentro de la base de dato relacional están sujetas en el modelo Acid, esto se debe a que estas propiedades garantizan que todas las transacciones se ejecuten de forma segura.
- Se logro identificar que las bases de datos no relacionales se las utiliza en entorno distribuidos ya que permiten hacer réplicas de la información en diferentes lugares al mismo tiempo con la finalidad de poner a disposición dichas copias cuando los demás nodos tengan fallos.
- Se comprobó que estas bases de datos no relacionales aportan varios modelos de almacenamiento de datos que pueden ser implementados en cualquier tipo de entorno donde se requieran desarrollar aplicaciones modernas.
- Se constato que tanto las bases de datos relacionales y no relacionales se las puede utilizar en diferentes entornos donde se requiera aplicar soluciones específicas independientemente del tipo de área de desarrollo.

Referencias

Redacción KeepCoding. (19 de enero de 2022). *¿Qué es una base de datos y cómo funciona?* . Obtenido de KeepCoding Tech School: <https://keepcoding.io/blog/que-es-una-base-de-datos-y-como-funciona/>

Aws. (2022). *¿Qué es una base de datos clave-valor?* Obtenido de *¿Qué es NoSQL?*: <https://aws.amazon.com/es/nosql/key-value/>

Aws. (2022). *¿Qué es una base de datos de gráficos? [imagen]*. Obtenido de aws: <https://aws.amazon.com/es/nosql/graph/>

Base de datos NOSQL. (30 de octubre de 2017). Obtenido de Modelo de red: <https://modelodered.wordpress.com/2017/10/30/base-de-datos-nosql/>

Calvo, D. (3 de noviembre de 2017). *Base de datos orientada a documentos [imagen]*. Obtenido de Diego Calvo: <https://www.diegocalvo.es/base-de-datos-orientada-a-documentos/>

DBA dixit. (26 de marzo de 2020). *Modelo Acid de transacciones*. Obtenido de Recursos en línea para el administrador y el desarrollador de base de datos: <http://dbadixit.com/modelo-acid-transacciones/>

DB-ENGINES. (2022). *DB-Engines Ranking - Trend of Relational DBMS Popularity [captura de pantalla]*. Obtenido de DB-ENGINES: https://db-engines.com/en/ranking_trend/relational+dbms

Ferraris, D. R. (2010). *Fundamentos de informática y programación en C* (Primera ed.). España. Obtenido de <https://books.google.com.ec/books?id=FfEfCB-hXCgC&pg=PT297&dq=Edgar+Frank+Codd&hl=es&sa=X&ved=2ahUKEwjJ8bSXkqn2AhViVzABHT7rA-UQ6AF6BAgJEAI#v=onepage&q=Edgar%20Frank%20Codd&f=false>

International, D. (2017). *DAMA-DMBOK: Guía Del Conocimiento Para La Gestión De Datos (Spanish Edition)* (Segunda ed.). (T. Publications, Ed.) Obtenido de https://books.google.com.ec/books?id=5fnvDwAAQBAJ&printsec=frontcover&hl=es&source=gs_ge_summary_r&cad=0#v=onepage&q&f=false

Joaquinaraujo. (s.f.). *Breve historia del origen de las bases de datos*. Obtenido de Platzi: <https://platzi.com/blog/historia-origen-bases-de-datos/>

Llaca, M. (8 de enero de 2017). *Historia del lenguaje más popular: desde SEQUEL hasta SQL*. Obtenido de Parceladigital: <https://parceladigital.com/2017/01/08/historia-del-lenguaje-mas-popular-desde-sequel-hasta-sql>

Macarrón, P. (8 de Marzo de 2021). *Tipos de bases de datos no relacionales*. Obtenido de Certia: <https://www.certia.net/tipos-de-bases-de-datos-no-relacionales/>

Microsoft. (14 de septiembre de 2021). *Propiedades ACID*. Obtenido de Microsoft: <https://docs.microsoft.com/es-es/windows/win32/cos-sdk/acid-properties>

Mosquera Palacios, D. J., Wanumen Silva, L. F., & Rivas Trujillo, E. (2018). *Base de datos en SQL Server* (Primera ed.). Bogotá: Ecoe. Obtenido de https://books.google.com.ec/books?id=OtdJEAAAQBAJ&printsec=frontcover&dq=fundamentos+de+sql&hl=es&sa=X&redir_esc=y#v=onepage&q=fundamentos%20de%20sql&f=false

Narro, W. (15 de diciembre de 2017). *Lenguaje SQL*. Obtenido de Bitácora de software: <http://www.bitacorasoftware.com/2017/12/lenguaje-sql.html>

Oracle. (2022). *El significado de base de datos relacional*. Obtenido de Qué es una base de datos relacional: <https://www.oracle.com/ar/database/what-is-a-relational-database/>

Primera base de datos. (22 de mayo de 2021). Obtenido de Pregunte las bibliotecas responden: <https://www.pregunte.es/consulta/inicio.cmd?FORMULARIO=buscador&ACCION=getDetalleConsulta&NIC=78874>

Regalado Jalca, J. J., García, J. G., & Gómez, Á. P. (2017). *Fundamentos sobre la gestión de base de datos* (Primera ed.). Alicante: Área de innovación y desarrollo,S.L. Obtenido de <http://repositorio.unesum.edu.ec/bitstream/53000/2077/1/Fundamento%20sobre%20la%20Gestio%20de%20bases%20de%20datos.pdf>

Silveira, P. (2021 de junio de 2021). *Qué es SQL*. Obtenido de Alura Latam: <https://www.aluracursos.com/blog/que-es-sql>

Sinha, S. (7 de julio de 2021). *HBase Architecture: HBase Data Model & HBase Read/Write Mechanism[imagen]*. Obtenido de edureka: <https://www.edureka.co/blog/hbase-architecture/>

Socrates. (2022). *Atomicidad consistencia aislamiento durabilidad (ácido)*. Obtenido de Techinfo: <https://techinfo.wiki/atomicidad-consistencia-aislamiento-durabilidad-acido/>

Williams, A. (8 de abril de 2021). *Key-Value Databases, Explained[imagen]*. Obtenido de KDnuggets: <https://www.kdnuggets.com/2021/04/nosql-explained-understanding-key-value-databases.html>

Yecid Alexis Rendón. (28 de mayo de 2019). *Bases de datos relacionales vs. no relacionales*. Obtenido de Academia praoma: <https://www.pragma.com.co/academia/lecciones/bases-de-datos-relacionales-vs.-no-relacionales>

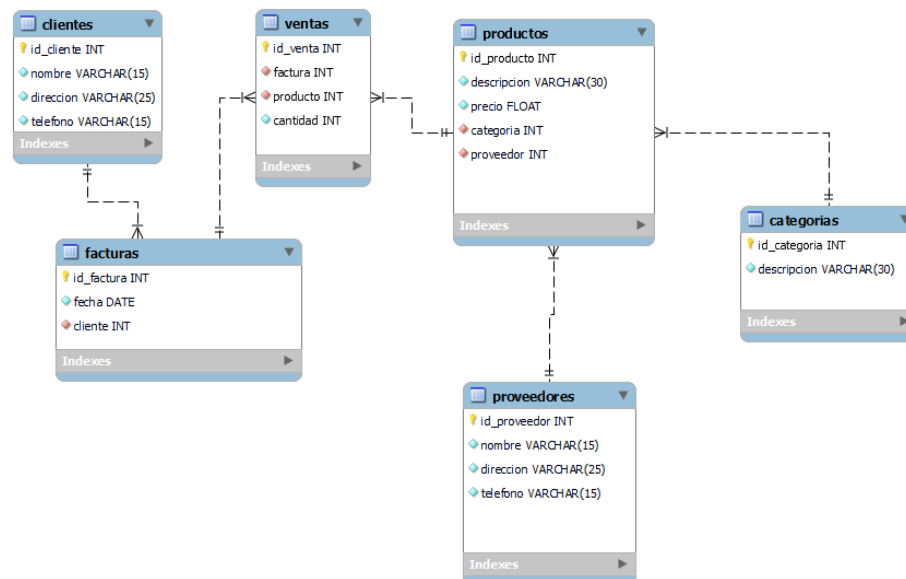
Anexo

Práctica

Mediante el gestor de base de datos relacional MySQL y no relacional MongoDB se efectuarán diversas sentencias que son implementadas para la creación y manipulación de los datos con la finalidad de conocer cómo se estructura cada sentencia.

Base de datos relacional.

El siguiente diagrama de base de datos relacional se lo utiliza como ejemplo para mostrar las diversas relaciones que existen entre las tablas, los campos que constituyen cada una de las tablas con las cuales está compuesta la base de datos relacional.



Instrucciones en MySQL.

La instrucción create database se la implementa para crea una base de datos.

```
create database tienda;
```

La sentencia use permite poner en funcionamiento la base de datos relacional.

```
use tienda;
```

Se crea la estructura de la tabla mediante el comando create table, se define las columnas con las cuales estará compuesta la tabla, se crea la clave principal que es el identificador que servirá para establecer las relaciones con las demás tablas con el comando primary key.

```
create table clientes(  
  id_cliente int not null auto_increment,  
  nombre varchar(15) not null,  
  direccion varchar(25) not null,  
  telefono varchar(15) not null,  
  primary key(id_cliente)  
);
```

Se modifica la tabla con alter table para establecer las respectivas relaciones con las demás tablas mediante la clave secundaria.

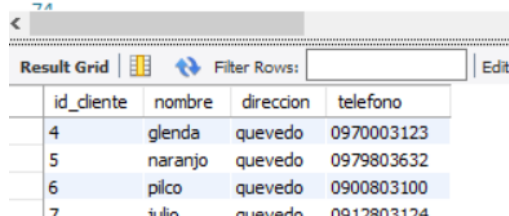
```
alter table facturas add foreign key(cliente) references clientes(id_cliente);  
alter table ventas add foreign key(factura) references facturas(id_factura);
```

Insert into permite insertar registros en la tabla de la base de datos relacional.

```
insert into clientes (id_cliente,nombre,direccion,telefono) values  
( '1', 'pedro', 'quevedo', '0979803123' );
```

Select * from se lo utiliza para consultar los datos que se encuentran almacenados en la tabla clientes.

```
73 • select * from clientes;
```

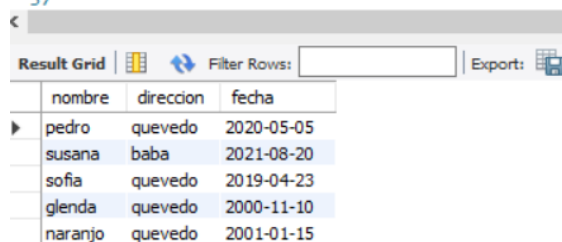


Result Grid | Filter Rows: | Edit

id_cliente	nombre	direccion	telefono
4	glenda	quevedo	0970003123
5	naranja	quevedo	0979803632
6	pilco	quevedo	0900803100
7	idlin	quevedo	0012803124

Consulta
la implementación

```
34 • select c.nombre,c.direccion,f.fecha  
35 from clientes c inner join facturas f  
36 on c.id_cliente=f.cliente;
```



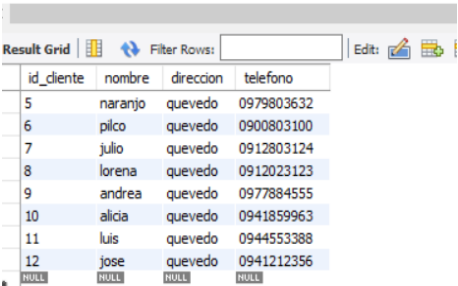
Result Grid | Filter Rows: | Export: |

nombre	direccion	fecha
pedro	quevedo	2020-05-05
susana	baba	2021-08-20
sofia	quevedo	2019-04-23
glenda	quevedo	2000-11-10
naranja	quevedo	2001-01-15

relacionada de dos tablas con
de inner join.

La instrucción delete se la utiliza para eliminar un registro de la tabla clientes implementado la cláusula where para especificar un dato en específico dentro de la tabla.

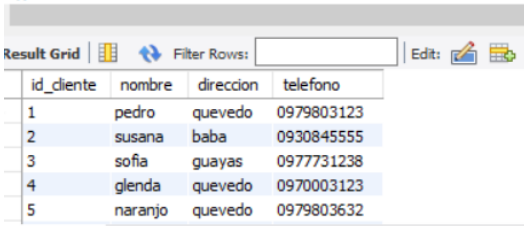
```
42 • delete from clientes where id_cliente='13';
```



id_cliente	nombre	direccion	telefono
5	naranja	quevedo	0979803632
6	pico	quevedo	0900803100
7	julio	quevedo	0912803124
8	lorena	quevedo	0912023123
9	andrea	quevedo	0977884555
10	alicia	quevedo	0941859963
11	luis	quevedo	0944553388
12	jose	quevedo	0941212356
NULL	NULL	NULL	NULL

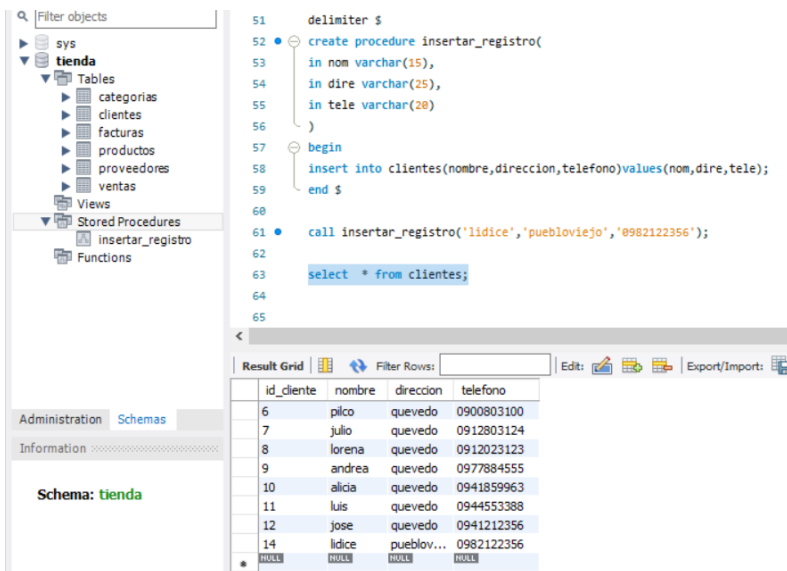
Update sirve para modificar un registro.

```
45 • update clientes set direccion='guayas'
46   where id_cliente='3';
47
```



id_cliente	nombre	direccion	telefono
1	pedro	quevedo	0979803123
2	susana	baba	0930845555
3	sofia	guayas	0977731238
4	glenda	quevedo	0970003123
5	naranja	quevedo	0979803632

Creo un procedimiento de almacenado con la instrucción create procedure para insertar datos en la tabla clientes y se envían los datos como parámetros con la instrucción call.



```
51 delimiter $
52 • create procedure insertar_registro(
53   in nom varchar(15),
54   in dire varchar(25),
55   in tele varchar(20)
56 )
57 begin
58   insert into clientes(nombre,direccion,telefono)values(nom,dire,tele);
59 end $
60
61 • call insertar_registro('lidice','pueblviejo','0982122356');
62
63 select * from clientes;
64
65
```

id_cliente	nombre	direccion	telefono
6	pico	quevedo	0900803100
7	julio	quevedo	0912803124
8	lorena	quevedo	0912023123
9	andrea	quevedo	0977884555
10	alicia	quevedo	0941859963
11	luis	quevedo	0944553388
12	jose	quevedo	0941212356
14	lidice	pueblv...	0982122356
NULL	NULL	NULL	NULL

Drop table permite eliminar una tabla de la base de datos.

```
drop table proveedores;
```

Truncate table sirve para vaciar todos los datos de la tabla clientes.

```
truncate table clientes;
```

Base de datos no relacional MongoDB.

En mongodb no se necesita establecer una estructura para poder almacenar los datos esto se debe a que utiliza un esquema dinámico que facilita el almacenaje de los datos en documentos json.

Instrucciones en MongoDB.

El comando show dbs nos permite listar las bases de datos que están creadas en mongodb.

```
> show dbs
< admin      41 kB
  config    111 kB
  local     73.7 kB
test >
```

Use nos verifica en mongodb si hay alguna base de datos creada con ese nombre que se especifica y si no la hay la crea, la pone en funcionamiento.

```
> use tienda
< 'switched to db tienda'
tienda > |
```

Crea la colección dentro de base de datos mongodb y al mismo tiempo se insertan los datos y se agrega un identificador.

```
> _MONGOSH
> db.clientes.insert({nombre:"pedro",direccion:"ventanas",telefono:"097412359"})
< { acknowledged: true,
  insertedIds: { '0': ObjectId("62321c68363c0cc4701fa4d2") } }
tienda >
```

Show collections nos muestra las colecciones que se han creado en la base de datos.

```
> show collections
< clientes
tienda >
```

La instrucción find nos permite mostrar los registros que sean ingresado en la colección del documento.

```
> db.clientes.find()
< { _id: ObjectId("62321c68363c0cc4701fa4d2"),
  nombre: 'pedro',
  direccion: 'ventanas',
  telefono: '097412359' }
tienda >
```

La instrucción insertmany permite insertar varios documentos al mismo tiempo.

```
> db.proveedores.insertMany([ {nombre:"tonia",direccion:"napo",telefono:"0123457856"},
  < { acknowledged: true,
    insertedIds:
      { '0': ObjectId("62323161363c0cc4701fa4d4"),
        '1': ObjectId("62323161363c0cc4701fa4d5"),
        '2': ObjectId("62323161363c0cc4701fa4d6") } } ]
tienda >
```

Mediante el comando find junto con pretty crea un mejor formato de visualización de los registros.

```
> db.proveedores.find().pretty()
< { _id: ObjectId("62323161363c0cc4701fa4d4"),
  nombre: 'tonia',
  direccion: 'napo',
  telefono: '0123457856' }
  { _id: ObjectId("62323161363c0cc4701fa4d5"),
  nombre: 'glenda',
  direccion: 'guayas',
  telefono: '097412356' }
  { _id: ObjectId("62323161363c0cc4701fa4d6"),
  nombre: 'lorena',
  direccion: 'salinas',
  telefono: '1456238978' }
tienda >
```

Elimina un registro de la colección con la instrucción deleteone.

```
> db.proveedores.deleteOne({nombre:"glenda"})
< { acknowledged: true, deletedCount: 1 }
tienda >
```

Se realiza una actualización mediante la instrucción updateOne en el campo dirección.

```
> db.proveedores.updateOne({direccion:"salinas"},{$set:{direccion:"los rios"}}
< { acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0 }
tienda >
```

```
> db.proveedores.find().pretty()
< { _id: ObjectId("62323161363c0cc4701fa4d6"),
  nombre: 'lorena',
  direccion: 'los rios',
  telefono: '1456238978' }
tienda > |
```