Tarea

Evidencia de Desempeño Construcción de Base de Datos Unidad 3

Axel David Pérez Hernández, Guadalupe Joan Anaya Cárdenas, Alejandro Yahir Mérida Cruz, Ángel Yahir Martínez Gómez, Alexis Yahir Villar Escorcia

Universidad Politécnica De Tulancingo, Calle Ingenierías #100, Huapalcalco, Tulancingo

Base de Datos

Nombre del profesor: Víctor Hugo Fernández Cruz

Tulancingo Hidalgo 11 de abril de 2024

U P T

Contenido

RESUMEN 3
INTRODUCCION 4
DESARROLLO 6
SISTEMA MANEJADOR SELECCIONADO 6
MODELADO DE BASE DE DATOS 8
MODELO MERE 10
..... 10
DIAGRAMA DE TRANSICIÓN 11
NORMALIZACIÓN DE LA BASE DE DATOS 12
SENTENCIAS DEL LENGUAJE DE DEFINICIÓN Y MANIPULACIÓN DE DATOS UTILIZADAS 15
CONCLUSIONES 23
REFERENCIAS 24

RESUMEN

Se muestra una base de datos para gestionar los alquileres de viviendas de particulares. La base de datos se compone de: Propietarios que es donde se almacena la información de los propietarios que ofrecen sus viviendas en alquiler. Las viviendas que se ofrecen en alquiler.

Los inquilinos que se interesan por alquilar las viviendas. Los Alquileres almacenan la información de los alquileres que se han realizado.

Las tablas están relacionadas entre sí mediante claves foráneas. La clave foránea de la tabla Alquileres a la tabla Propietarios indica el propietario de la vivienda alquilada. La clave foránea de la tabla Alquileres a la tabla Viviendas indica la vivienda que se ha alquilado. La clave foránea de la tabla Alquileres a la tabla Inquilinos indica el inquilino que ha alquilado la vivienda.

INTRODUCCION

La gestión de los alquileres de viviendas de particulares puede ser una tarea compleja. Es necesario tener información sobre los propietarios, las viviendas, los inquilinos y los alquileres. Esta información debe ser precisa y estar actualizada.

Una base de datos bien diseñada y estructurada puede facilitar la gestión de datos y mejorar la toma de decisiones. La base de datos debe ser capaz de almacenar la información de forma eficiente y permitir el acceso a la información de forma rápida y sencilla.

Se muestra una base de datos que puede ser utilizada para gestionar los alquileres de viviendas de particulares. La base de datos está diseñada para almacenar la información de forma eficiente y permitir el acceso a la información de forma rápida y sencilla.

La base de datos se compone de las siguientes tablas:

Propietarios: Esta tabla almacena la información de los propietarios que ofrecen sus viviendas en alquiler. La información incluye el nombre, la dirección, el teléfono y el correo electrónico.

Campos:

IdPropietario (clave primaria)

Nombre, Dirección, Teléfono, CorreoElectronico.

Viviendas: Esta tabla almacena la información de las viviendas que se ofrecen en alquiler. La información incluye la dirección, la superficie, el número de habitaciones, el precio del alquiler y las características de la vivienda.

Campos:

IdVivienda (clave primaria), IdPropietario (clave foránea), Dirección, Superficie, NumHabitaciones, PrecioAlquiler, Caracteristicas.

Inquilinos: Esta tabla almacena la información de los inquilinos que se interesan por alquilar las viviendas. La información incluye el nombre, la dirección, el teléfono y el correo electrónico.

Campos:

IdInquilino (clave primaria), Nombre, Dirección, Teléfono, CorreoElectronico.

Alquileres: Esta tabla almacena la información de los alquileres que se han realizado. La información incluye la fecha de inicio, la fecha de fin, el precio del alquiler y la vivienda alquilada.

Campos:

IdAlquiler (clave primaria), IdVivienda (clave foránea), IdInquilino (clave foránea), FechaInicio, FechaFin, PrecioAlquiler.

Esta base de datos puede ser utilizada para realizar las siguientes tareas:

Almacenar la información de los propietarios, las viviendas, los inquilinos y los alquileres.

Acceder a la información de forma rápida y sencilla.

Generar informes sobre los alquileres.

Tomar decisiones sobre la gestión de los alquileres.

La base de datos puede ser ampliada para incluir información adicional, como las fotos de las viviendas, los contratos de alquiler y los pagos realizados.

DESARROLLO

SISTEMA MANEJADOR SELECCIONADO

Un sistema gestor de base de datos (SGBD) o Database Management System (DBMS) es un conjunto de programas invisibles para el usuario final con el que se administra y gestiona la información que incluye una base de datos. Los gestores de datos o gestores de base de datos permiten administrar todo acceso a la base de datos, pues tienen el objetivo de servir de interfaz entre esta, el usuario y las aplicaciones.

En la actualidad hay gestores de bases de datos que cumplen un modelo para acceder a la misma de modo más sencillo, con lenguajes de consulta que permiten generar informes, analizar, garantizar la seguridad y la integridad de los datos.

1. MySQL MySQL es un SGBD que está escrito en C y C++ y, además, está provisto de un analizador sintáctico de SQL basado en Yacc con un tokenizador (escáner léxico) propio. Además, el sistema de gestión de bases de datos se destaca por su amplio soporte de sistemas operativos. 4

2. Microsoft SQL Server Microsoft SQL Server es una herramienta para la gestión de bases de datos cuyo principal lenguaje de consulta es Transact-SQL, una aplicación de las normas ANSI/ISO estándar Structured Query Language (SQL). Algunas de sus características son las siguientes:

Soporte de transacciones.

- Escalabilidad, estabilidad y seguridad.
- Soporta procedimientos almacenados.
- Incluye un potente entorno gráfico de administración que permite el uso de comandos DDL y DML gráficamente.
- Permite trabajar en modo cliente-servidor, donde la información y datos se alojan en el servidor y las terminales o clientes de la red sólo acceden a la información.

- Permite administrar información de otros servidores de datos. Microsoft SQL Server es uno de los principales sistemas de gestión de base de datos del mercado, dispone de un amplio abanico de aplicaciones de software destinados a la inteligencia empresarial y analítica de mercado.

(Pérez, Gestor de Base de Datos: Qué es, funcionalidades y ejemplos 2023)

MySQL se utiliza para almacenar información para contactar a los propietarios en caso de que haya algún problema con la vivienda, en las viviendas se puede utilizar para gestionar las propiedades y para encontrar inquilinos.

En el caso de los alquileres se puede utilizar para rastrear los pagos del alquiler y para generar informes. Con los inquilinos esta información se puede utilizar para verificar la identidad de los inquilinos y para contactarlos en caso de que haya algún problema.

En las agencias la información se puede utilizar para contactar a las agencias inmobiliarias en caso de que haya algún problema con la vivienda. La información de los servicios se puede utilizar para gestionar los servicios que se ofrecen en las viviendas.

MySQL se utiliza para esta información porque es un SGBD fácil de usar, flexible, escalable, rentable y con una gran comunidad de usuarios:

Seguridad: MySQL ofrece una serie de características de seguridad que ayudan a proteger la información contra accesos no autorizados.

Rendimiento: MySQL es un SGBD muy rápido que puede manejar grandes cantidades de datos sin problemas.

Confiabilidad: MySQL es un SGBD muy confiable que ofrece un alto tiempo de actividad.

MODELADO DE BASE DE DATOS

Un modelo de base de datos describe la estructura lógica de una base de datos, incluidas las relaciones y restricciones que determinan cómo se almacenan y acceden a los datos.

Los modelos de bases de datos individuales se diseñan basándose en reglas y conceptos de modelos de datos más amplios adoptados por el diseñador.

La mayoría de los modelos de datos pueden representarse mediante los diagramas de bases de datos adjuntos.

Modelo de relación de entidades Este modelo captura las relaciones entre entidades del mundo real de una manera muy similar al modelo de red, pero no está directamente vinculado a la estructura física de la base de datos.

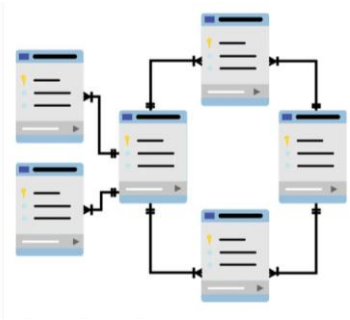

Diagrama de modelo E-R. *Fuente: Ipesanz, CC BY 4.0*

Aquí, las personas, los lugares y las cosas para los que se almacenan los puntos de datos son las llamadas entidades, cada una de las cuales tiene ciertos atributos que forman un dominio.

También se representan gráficamente las cardinalidades y relaciones entre entidades.

El modelo de base de datos presentado en este documento refleja la estructura de la información de manera visual y comprensible. El diagrama entidad-relación (ER) permite identificar las entidades relevantes, sus atributos, relaciones y cardinalidad.

Un ejemplo sencillo es que un Propietario (Juan Pérez) puede tener varias Viviendas (Vivienda 1 y Vivienda 2) en alquiler. Cada Vivienda puede tener varios Alquileres (Alquiler 1 y Alquiler 2) a lo largo del tiempo. Un Alquiler puede tener varios Inquilinos (Inquilino 1 e Inquilino 2), como en el caso de una familia.

MODELO MERE

Se desea gestionar los alquileres de viviendas de particulares, se requiere tener información de los propietarios que ofrecen sus viviendas en alquiler, así como de los inquilinos que se interesan por alquilar estas, obteniendo información sobre los alquileres y las viviendas alquiladas correspondientemente, saber de la duración de cada alquiler (histórico) y la información de renovación de un alquiler, por él inquilino.

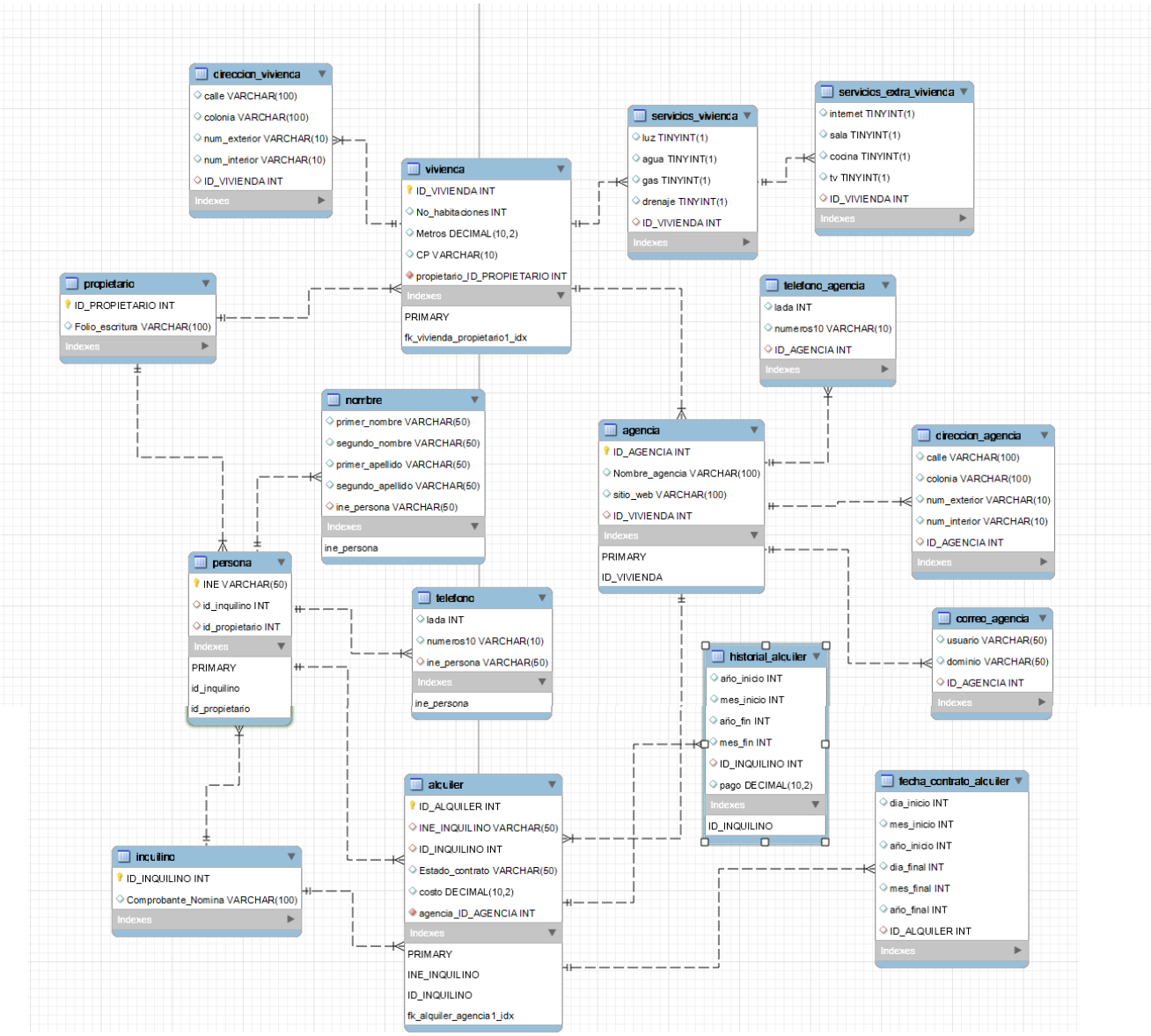

DIAGRAMA DE TRANSICIÓN

PROPIETARIO (ID_PROPIETARIO, Folio_escritura) ID_PROPIETARIO es una clave primaria de PROPIETARIO

PERSONA (INE) INE es una clave primaria de PERSONA

TELEFONO (lada, 10_numeros)

NOMBRE (primer_nombre, segundo_nombre, primer_apellido, segundo_apellido)

INQUILINO (ID_INQUILINO, Comprobante_Nomina) ID_INQUILINO es una clave primaria de INQUILINO

VIVIENDA (ID_VIVIENDA, No_haitaciones, Metros, CP) ID_VIVIENDA es una clave primaria de VIVIENDA

DIRECCION_VIVIENDA (calle, colonia, num_exterior, num_interior)

SERVICIOS_VIVIENDA (luz, agua, gas, drenaje)

SERVICIOS_EXTRA_VIVIENDA (internet, sala, cocina, tv)

AGENCIA (ID_AGENCIA, Nombre_agencia, sitio_web, ID_VIVIENDA) ID_VIVIENDA es una clave foránea de VIVIENDA, ID_AGENCIA es una clave primaria de AGENCIA

DIRECCION_AGENCIA (calle, colonia, num_exterior, num_interior)

TELEFONO_AGENCIA (lada, 10_numeros)

CORREO_AGENCIA (usuario, dominio)

ALQUILER (ID_ALQUILER, INE_INQUILINO, Estado_contrato, costo) INE_INQUILINO es una clave foránea de INQUILINO, ID_ALQUILER es una clave primaria de ALQUILER

FECHA_CONTRATO_ALQUILER (dia_inicio, mes_inicio, año_inicio, dia_final, mes_final, año_final)

HISTORIAL_ALQULER (año_inicio, mes_inico, año_fin, mes_fin, ID_INQUILINO, pago) ID_INQUILNO es una clave foránea de INQUILINO

NORMALIZACIÓN DE LA BASE DE DATOS

La normalización de bases de datos en simples palabras es organizar nuestro conjunto de datos para evitar redundancias y duplicaciones. Es un proceso que busca eliminar la redundancia, es decir, la duplicación innecesaria de información. Para lograr esto utilizaremos las formas normales; recordando que las formas normales son un conjunto de reglas que sirven como guía para estructurar y organizar las bases de datos de manera eficiente. Su objetivo principal es evitar la redundancia, mejorar la integridad de los datos y facilitar su acceso y manejo. En nuestro caso de estudio (Alquiler-Vivienda) hemos aplicado estas formas normales de la siguiente manera:

→*Primera forma normal:* En esta etapa, nos enfocamos en eliminar la repetición innecesaria de datos dentro de la base de datos. Para ello comenzamos identificando y desglosando los atributos compuestos de cada entidad. Tomando en cuenta que de la entidad supertipo “Persona” se desprenden dos entidades subtipos (“Propietario” e “Inquilino”) que heredan sus atributos, podemos definir que los atributos nombre y teléfono son atributos compuestos, por lo que se convertirían en nuevas tablas con sus atributos atómicos.

De igual forma se aplicó en el resto de entidades, en el caso de la entidad “Vivienda” sus atributos compuestos que se convirtieron en nuevas tablas fueron Dirección, Servicios y Servicios extra; para la entidad “Agencia” los atributos desglosados fueron Dirección, Teléfono y Correo_Electrónico.; y por último en la entidad “Alquiler” se organizó los atributos Fecha_Contrato e Historial para tener un mejor control de los días, meses y años.

→*Segunda forma normal:* En la segunda forma normal cada atributo no-clave debe depender de la clave primaria completa, no solo de una parte de ella. Una dependencia funcional es una relación entre dos atributos, donde un atributo (el determinante) determina el valor del otro (el dependiente).

Para que una tabla cumpla con la segunda forma normal debe cumplir con dos requisitos:

1. Que cumpla con la primera forma normal.

2. No tener dependencias parciales.

Una vez que se había cumplido la primera condición en nuestro modelo procedimos a verificar que efectivamente no existieran dependencias parciales en nuestras tablas.

Inicialmente el atributo compuesto de servicios extras pertenecía como un campo a el atributo compuesto de servicios; sin embargo, esto no era correcto ya que este no tenía dependencia total con el Id_vivienda ya que al considerarse como servicios extras al paquete básico de alquiler, este solo depende de si el inquilino los solicita y por lo tanto se convirtió en una nueva tabla.

→*Tercera forma normal:* La tercera forma normal se basa en la segunda forma normal y busca eliminar las dependencias transitivas, que pueden generar inconsistencias en los datos.

Para que una tabla cumpla con la tercera forma normal, debe cumplir con dos requisitos:

1. Estar en la segunda forma normal, en otras palabras, no tener dependencias parciales.

2. No tener dependencias transitivas, es decir que un atributo no-clave no puede depender de otro atributo no-clave, a menos que este último dependa de la clave primaria completa.

Llegados a este punto lo que teníamos que hacer era verificar que no existieran dependencias transitivas en nuestro modelo, y dado que esto se cumplía no fue necesario aplicar más modificaciones.

SENTENCIAS DEL LENGUAJE DE DEFINICIÓN Y MANIPULACIÓN DE DATOS UTILIZADAS

```
CREATE TABLE PROPIETARIO (
    ID_PROPIETARIO INT AUTO_INCREMENT PRIMARY KEY,
    Folio_escritura VARCHAR(100)
);

CREATE TABLE PERSONA (
    INE VARCHAR(50) PRIMARY KEY,
    id_inquilino INT,
    id_propietario INT,
    FOREIGN KEY (id_inquilino) REFERENCES INQUILINO(ID_INQUILINO),
    FOREIGN KEY (id_propietario) REFERENCES PROPIETARIO(ID_PROPIETARIO)
);

CREATE TABLE TELEFONO (
    lada INT,
    numeros10 VARCHAR(10),
    ine_persona VARCHAR(50),
    FOREIGN KEY (ine_persona) REFERENCES PERSONA(ine)
);

CREATE TABLE NOMBRE (
    primer_nombre VARCHAR(50),
    segundo_nombre VARCHAR(50),
    primer_apellido VARCHAR(50),
    segundo_apellido VARCHAR(50),
```

```
    ine_persona VARCHAR(50),
    FOREIGN KEY (ine_persona) REFERENCES PERSONA(ine)
);

CREATE TABLE INQUILINO (
    ID_INQUILINO INT AUTO_INCREMENT PRIMARY KEY,
    Comprobante_Nomina VARCHAR(100)
);

CREATE TABLE VIVIENDA (
    ID_VIVIENDA INT AUTO_INCREMENT PRIMARY KEY,
    No_habitaciones INT,
    Metros DECIMAL(10, 2),
    CP VARCHAR(10)
);

CREATE TABLE DIRECCION_VIVIENDA (
    calle VARCHAR(100),
    colonia VARCHAR(100),
    num_exterior VARCHAR(10),
    num_interior VARCHAR(10),
    ID_VIVIENDA INT,
    FOREIGN KEY (ID_VIVIENDA) REFERENCES VIVIENDA(ID_VIVIENDA)
);

CREATE TABLE SERVICIOS_VIVIENDA (
    luz BOOLEAN,
```

```
    agua BOOLEAN,
    gas BOOLEAN,
    drenaje BOOLEAN,
    ID_VIVIENDA INT,
    FOREIGN KEY (ID_VIVIENDA) REFERENCES VIVIENDA(ID_VIVIENDA)
);

CREATE TABLE SERVICIOS_EXTRA_VIVIENDA (
    internet BOOLEAN,
    sala BOOLEAN,
    cocina BOOLEAN,
    tv BOOLEAN,
    ID_VIVIENDA INT,
    FOREIGN KEY (ID_VIVIENDA) REFERENCES SERVICIOS_VIVIENDA(ID_VIVIENDA)
);

CREATE TABLE AGENCIA (
    ID_AGENCIA INT AUTO_INCREMENT PRIMARY KEY,
    Nombre_agencia VARCHAR(100),
    sitio_web VARCHAR(100)
);

CREATE TABLE DIRECCION_AGENCIA (
    calle VARCHAR(100),
    colonia VARCHAR(100),
    num_exterior VARCHAR(10),
    num_interior VARCHAR(10),
```

```
    ID_AGENCIA INT,
    FOREIGN KEY (ID_AGENCIA) REFERENCES AGENCIA(ID_AGENCIA)
);

CREATE TABLE TELEFONO_AGENCIA (
    lada INT,
    numeros10 VARCHAR(10),
    ID_AGENCIA INT,
    FOREIGN KEY (ID_AGENCIA) REFERENCES AGENCIA(ID_AGENCIA)
);

CREATE TABLE CORREO_AGENCIA (
    usuario VARCHAR(50),
    dominio VARCHAR(50),
    ID_AGENCIA INT,
    FOREIGN KEY (ID_AGENCIA) REFERENCES AGENCIA(ID_AGENCIA)
);

CREATE TABLE ALQUILER (
    ID_ALQUILER INT AUTO_INCREMENT PRIMARY KEY,
    INE_INQUILINO VARCHAR(50),
    ID_INQUILINO INT,
    Estado_contrato VARCHAR(50),
    ID_AGENCIA INT,
    costo DECIMAL(10, 2),
    FOREIGN KEY (INE_INQUILINO) REFERENCES PERSONA(INE),
    FOREIGN KEY (ID_INQUILINO) REFERENCES INQUILINO(ID_INQUILINO),
```

```
    FOREIGN KEY (ID_AGENCIA) REFERENCES AGENCIA(ID_AGENCIA)
);

CREATE TABLE FECHA_CONTRATO_ALQUILER (
    dia_inicio INT,
    mes_inicio INT,
    año_inicio INT,
    dia_final INT,
    mes_final INT,
    año_final INT,
    ID_ALQUILER INT,
    FOREIGN KEY (ID_ALQUILER) REFERENCES ALQUILER(ID_ALQUILER)
);

CREATE TABLE HISTORIAL_ALQUILER (
    año_inicio INT,
    mes_inicio INT,
    año_fin INT,
    mes_fin INT,
    ID_INQUILINO INT,
    pago DECIMAL(10, 2),
    FOREIGN KEY (ID_INQUILINO) REFERENCES ALQUILER(ID_INQUILINO)
);
```

INSERTAR DATOS

```
INSERT INTO PROPIETARIO (ID_PROPIETARIO, Folio_escritura)
VALUES
```

```
(1, 'Folio_001'),
(2, 'Folio_002'),
(3, 'Folio_003'),
(4, 'Folio_004'),
(5, 'Folio_005'),
(6, 'Folio_006'),
(7, 'Folio_007'),
(8, 'Folio_008'),
(9, 'Folio_009'),
(10, 'Folio_010');

INSERT INTO PERSONA (INE)
VALUES
('INE_001'),
('INE_002'),
('INE_003'),
('INE_004'),
('INE_005'),
('INE_006'),
('INE_007'),
('INE_008'),
('INE_009'),
('INE_010');

INSERT INTO TELEFONO (lada, numeros10)
```

```
VALUES
(52, '1234567890'),
(52, '2345678901'),
(52, '3456789012'),
(52, '4567890123'),
(52, '5678901234'),
(52, '6789012345'),
(52, '7890123456'),
(52, '8901234567'),
(52, '9012345678'),
(52, '0123456789');

INSERT INTO VIVIENDA (No_habitaciones, Metros, CP)
VALUES
(3, 120.5, '11000'),
(2, 80.3, '12000'),
(4, 150.0, '13000'),
(3, 110.0, '14000'),
(2, 90.0, '15000'),
(3, 100.0, '16000'),
(4, 180.0, '17000'),
(2, 70.0, '18000'),
(3, 95.0, '19000'),
(2, 85.0, '20000');
```

MANIPULACION DE DATOS

```
SELECT v.ID_VIVIENDA, v.Metros, v.No_habitaciones, p.Folio_escritura
FROM VIVIENDA v
INNER JOIN PROPIETARIO p ON v.ID_PROPIETARIO = p.ID_PROPIETARIO;

SELECT v.ID_VIVIENDA, se.internet, se.sala, se.cocina, se.tv
FROM VIVIENDA v
LEFT JOIN SERVICIOS_EXTRA_VIVIENDA se ON v.ID_VIVIENDA = se.ID_VIVIENDA;


SELECT v.ID_VIVIENDA, se.internet, se.sala, se.cocina, se.tv
FROM VIVIENDA v
RIGHT JOIN SERVICIOS_EXTRA_VIVIENDA se ON v.ID_VIVIENDA = se.ID_VIVIENDA;
```

CONCLUSIONES

El proceso de construcción de la base de datos para la gestión de alquileres de viviendas particulares ha sido fundamental para mejorar la eficiencia en la búsqueda de información y la toma de decisiones en el ámbito inmobiliario. La selección del sistema manejador, el diseño del modelo entidad-relación, la normalización de la base de datos y el uso de sentencias SQL han sido pasos clave en este proceso. Se espera que esta base de datos contribuya significativamente a la eficiencia operativa y la toma de decisiones en la empresa.

Este reporte ha destacado la importancia de un enfoque metódico y estructurado en el diseño y la implementación de bases de datos. Además, se subraya la necesidad de mantener la base de datos actualizada y optimizada para garantizar su eficacia a largo plazo. En resumen, una base de datos bien diseñada y gestionada puede ser un activo invaluable para cualquier empresa en la era digital.

REFERENCIAS

Helenclu. (n.d.). *Descripción de la normalización de la base de datos - Microsoft 365 Apps*. Microsoft Learn. https://learn.microsoft.com/es-es/office/troubleshoot/access/database-normalization-description

Pérez, S. D. (2023, March 23). Gestor de Base de Datos: Qué es, funcionalidades y ejemplos. Intelequia. https://intelequia.com/es/blog/post/gestor-de-base-de-datos-qu%C3%A9-es-funcionalidades-y-ejemplos

Inc., L. (2010, January 18). Qué es un modelo de base de datos. https://www.lucidchart.com/pages/es/que-es-un-modelo-de-base-de-datos#:~:text=Un%20modelo%20de%20base%20de%20datos%20muestra%20la%20estructura%20l%C3%B3gica,c%C3%B3mo%20se%20accede%20a%20ellos.

Torrejón, H. C. (2022, November 24). Cómo realizar la Normalización de bases de Datos y Por Qué. OpenWebinars.net. https://openwebinars.net/blog/como-realizar-la-normalizacion-de-bases-de-datos-y-por-que/