REPORTE DE PRÁCTICA

NOMBRE DE LA PRÁCTICA: "PRACICA 0"

ALUMNO: LUIS ALONSO LOZADA CASTELÁN
Dr. Eduardo Cornejo-Velázquez

1. Introducción

En el ámbito de la ingeniería del software, la construcción de sistemas robustos y eficientes descansa sobre una base fundamental: la aplicación de procesos sistemáticos y metodológicos. Este principio, que es la columna vertebral de disciplinas como el diseño de bases de datos relacionales y la teoría de los lenguajes formales, garantiza que los productos finales no sean fruto de la improvisación, sino el resultado de un diseño riguroso y premeditado.

Tal como lo establecen autores como C.J. Date y Piattini y Marcos, el diseño de una base de datos es un proceso que se estructura en niveles de abstracción. Se inicia en el "Mundo Real", con un análisis detallado de los requisitos, para luego ser plasmado en un "Esquema Conceptual" que modela las entidades y sus relaciones, independientemente de la tecnología. Este esquema se transforma metódicamente en un "Esquema Lógico Relacional", donde las tablas, claves y restricciones se definen con precisión matemática. Saltarse estas fases o ejecutarlas de manera desordenada conduce inevitablemente a anomalías en los datos, redundancia e inconsistencias, comprometiendo la integridad de todo el sistema.

Antes de que un compilador o intérprete pueda ejecutar una línea de código, debe realizar un análisis metódico que incluye las fases léxica, sintáctica y semántica. En la fase léxica, un "autómata finito" descompone el código en tokens o componentes básicos; en la sintáctica, una "gramática libre de contexto" valida la estructura de las sentencias; y en la semántica, se verifica la lógica y los tipos de datos. Este riguroso proceso de "compilación" es el equivalente funcional al diseño por niveles de una base de datos.

Por lo tanto, este reporte busca fundamentar, a partir de los conceptos de Lenguajes Formales y de la teoría de Bases de Datos, que la metodología sistemática no es una mera recomendación, sino una necesidad imperante. Es el puente de ingeniería que nos permite transitar de manera confiable y eficiente desde un problema complejo del mundo real hasta una solución software funcional, consistente y mantenible.

2. Marco teórico

Análisis de requerimientos

El análisis de requerimientos constituye la fase inicial y fundamental en el diseño de cualquier sistema de base de datos. Según [3], este proceso comienza en el "Mundo Real", donde el diseñador debe recopilar y analizar exhaustivamente la información del dominio del problema. El objetivo es comprender y documentar los datos que se deben almacenar, sus relaciones y las reglas de negocio que los gobiernan. [1] enfatiza que un sistema de bases de datos existe para "almacenar información y permitir a los usuarios obtener y actualizar esa información según sea necesario" (p. 4), por lo que un análisis de requerimientos preciso es la base para cumplir con este propósito. Omitir o realizar deficientemente esta fase conduce a un diseño que no refleja las necesidades reales de la organización.

Modelo Entidad - Relación

Una vez capturados los requerimientos, se procede a su modelado conceptual utilizando el Modelo Entidad-Relación. [3] lo presentan como el "Esquema Conceptual", una abstracción que describe la estructura de los datos sin considerar detalles de implementación. Este modelo representa el universo del discurso mediante **entidades** (objetos distinguibles, como AUTOR o LIBRO), sus **atributos** (propiedades) y las **relaciones** entre ellas (como "Escribe"). Esta herramienta proporciona una representación gráfica clara y comprensible tanto para usuarios como para desarrolladores, sirviendo como un plano conceptual que guía el diseño posterior.

Modelo relacional

El siguiente paso en la metodología sistemática es la transformación del esquema conceptual en un Modelo Relacional. [3] identifican esta etapa como la creación del "Esquema Lógico (relacional)". [2] fundamentan este modelo en una base matemática sólida, donde los datos se organizan en **relaciones** (tablas bidimensionales). Cada tabla se compone de tuplas (filas) y atributos (columnas). Las **claves primarias** garantizan la identificación única de cada fila, y las **claves foráneas** establecen y mantienen las relaciones entre las tablas, garantizando la integridad referencial. Este modelo es la base teórica sobre la que se construyen los Sistemas Gestores de Bases de Datos (SGBD) relacionales.

SQL

Finalmente, para interactuar con una base de datos relacional, se utiliza un lenguaje de propósito específico. SQL (Structured Query Language) es el lenguaje estándar y universal para la definición, manipulación y control de datos en un SGBD relacional. Aunque no se muestra explícitamente en las portadas, el libro de [2], titulado *Fundamentos de Sistemas de Bases de Datos*, aborda SQL en profundidad como la herramienta práctica para materializar el modelo relacional. SQL permite operacionalizar el diseño al proporcionar comandos para crear las estructuras de la base de datos (**DDL - Data Definition Language**), insertar, modificar y consultar los datos (**DML - Data Manipulation Language**), y administrar los permisos de acceso (**DCL - Data Control Language**). Es el puente indispensable entre el diseño teórico y la gestión práctica de la información.

3. Herramientas empleadas

Para el desarrollo del modelado e implementación de la base de datos, se han utilizado las siguientes herramientas especializadas que facilitan el proceso sistemático de diseño:

1. **ERD Plus**. Es una herramienta de **modelado conceptual y lógico** en línea que se utiliza específicamente para crear diagramas Entidad-Relación (ER) y esquemas relacionales. Permite visualizar de manera gráfica la estructura de la base de datos, definiendo entidades, atributos, relaciones y cardinalidades. Su utilidad principal radica en la transición directa desde el modelo ER hacia el esquema relacional[3].

2. **MySQL Server**. Es un **sistema gestor de bases de datos relacional** (RDBMS) de código abierto que implementa el modelo lógico relacional. Se utiliza para crear, mantener y operar la base de datos física mediante el lenguaje SQL. MySQL permite materializar el diseño teórico en una base de datos funcional, ejecutando consultas DDL (Data Definition Language) para crear tablas basadas en el esquema relacional, y consultas DML (Data Manipulation Language) para manipular los datos almacenados, cumpliendo así con los principios establecidos por [1] y [2].

4. Desarrollo

Análisis de requisitos

Para el desarrollo del sistema de gestión de fraccionamientos, se han identificado y analizado los siguientes requisitos principales del caso de estudio, siguiendo la metodología propuesta por [3] que inicia en el "Mundo Real".

Modelo Entidad - Relación

En la Tabla 1 se presenta la propuesta para el sistema de gestión de fraccionamientos, que representa las relaciones entre las entidades identificadas durante el análisis de requisitos.

Table 1: Matriz de relaciones del sistema de fraccionamientos.

Entidades	cliente	casa	fraccionamiento	trabajador	servicio
cliente	X				
casa		X			
fraccionamiento			X		
trabajador				X	
servicio					X

Las relaciones representadas en la matriz se fundamentan en el modelo conceptual descrito por [3], donde:

- **Cliente - Casa:** Relación "tener" (uno a muchos) - Un cliente puede tener una o múltiples casas
- **Cliente - Fraccionamiento:** Relación indirecta a través de la casa que posee en el fraccionamiento
- **Cliente - Servicio:** Relación "recibir" (uno a muchos) - Un cliente puede recibir múltiples servicios
- **Casa - Fraccionamiento:** Relación "ubicar" (muchos a uno) - Cada casa se ubica en un único fraccionamiento
- **Trabajador - Servicio:** Relación "hacer" (muchos a muchos) - Un trabajador puede realizar múltiples servicios y un servicio puede ser realizado por varios trabajadores

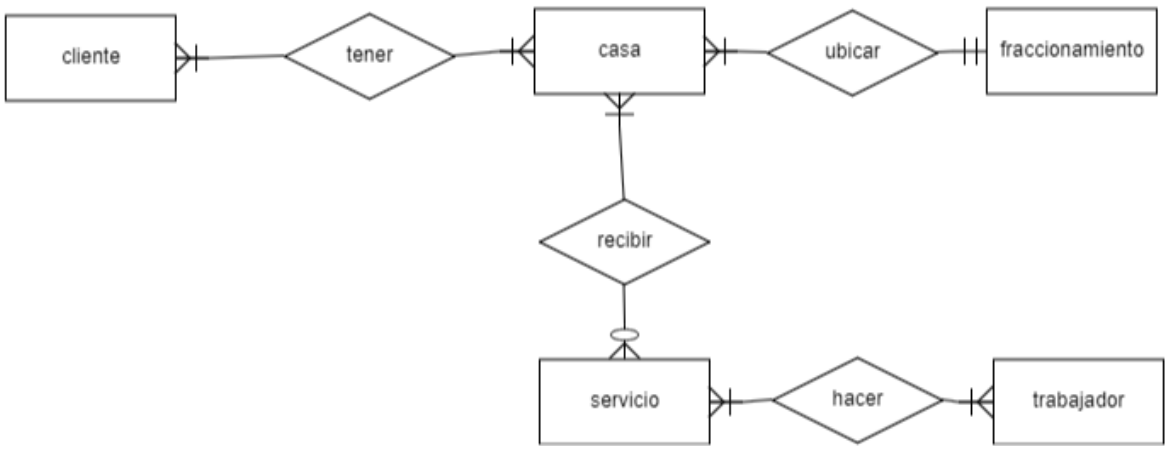

Figure 1: Modelo Entidad - Relación propuesto.

Modelo relacional

En la Figura 2 se presenta la propuesta de Modelo Entidad - Relación para.

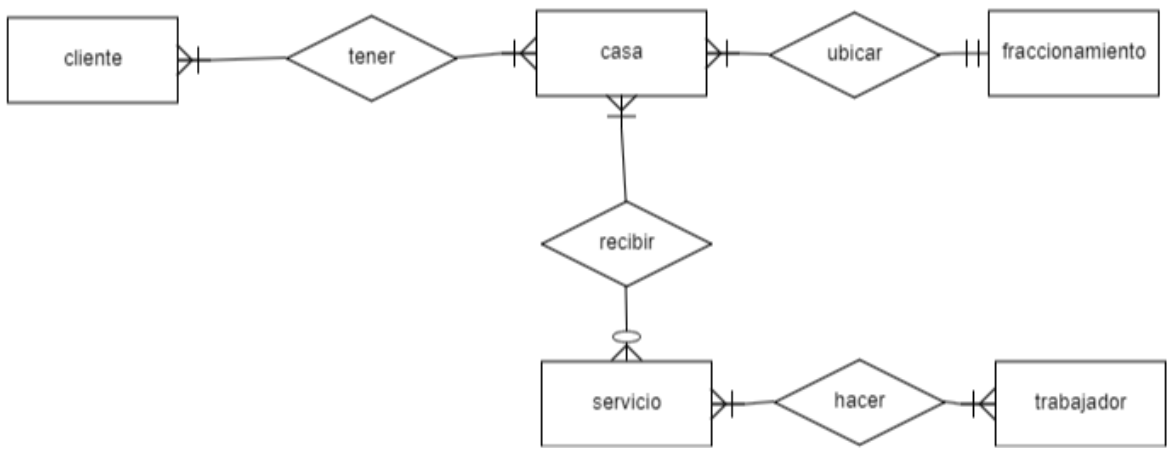

Figure 2: Modelo Relacional propuesto.

Sentencias SQL

Se presenta las sentencias para demostrar una BD, desde como se crea, se crean las tablas y se insertan datos.

Listing 1: Crear base de datos fraccionamiento.

```
CREATE DATABASE fraccionamiento;
USE fraccionamiento;
```

En el Listado 2 se presentan las sentencias SQL para crear las tablas del sistema.

Listing 2: Crear tablas del sistema.

```
-- Tabla fraccionamiento
CREATE TABLE fraccionamiento (
    id_fraccionamiento INT AUTO_INCREMENT PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    direccion VARCHAR(200),
    telefono VARCHAR(15)
);

-- Tabla cliente
CREATE TABLE cliente (
    id_cliente INT AUTO_INCREMENT PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    email VARCHAR(100),
    telefono VARCHAR(15)
);

-- Tabla casa
CREATE TABLE casa (
    id_casa INT AUTO_INCREMENT PRIMARY KEY,
```

```
    id_cliente INT,
    id_fraccionamiento INT,
    direccion VARCHAR(200) NOT NULL,
    numero_casa VARCHAR(10),
    FOREIGN KEY (id_cliente) REFERENCES cliente(id_cliente),
    FOREIGN KEY (id_fraccionamiento) REFERENCES fraccionamiento(id_fraccionamiento)
);

-- Tabla trabajador
CREATE TABLE trabajador (
    id_trabajador INT AUTO_INCREMENT PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    especialidad VARCHAR(50),
    telefono VARCHAR(15)
);

-- Tabla servicio
CREATE TABLE servicio (
    id_servicio INT AUTO_INCREMENT PRIMARY KEY,
    id_cliente INT,
    id_trabajador INT,
    tipo_servicio VARCHAR(50) NOT NULL,
    fecha_servicio DATE,
    descripcion TEXT,
    FOREIGN KEY (id_cliente) REFERENCES cliente(id_cliente),
    FOREIGN KEY (id_trabajador) REFERENCES trabajador(id_trabajador)
);
```

En el Listado 3 se presentan las sentencias SQL para insertar registros de ejemplo.

Listing 3: Insertar registros de ejemplo.

```
-- Insertar datos en fraccionamiento
INSERT INTO fraccionamiento (nombre, direccion, telefono) VALUES
('Villas del Sol', 'Av. Principal 123', '555-1000'),
('Residencial Las Flores', 'Calle Jard n 456', '555-2000');

-- Insertar datos en cliente
INSERT INTO cliente (nombre, email, telefono) VALUES
('Mar a Gonz lez', 'maria.gonzalez@email.com', '555-3000'),
('Carlos Rodr guez', 'carlos.rodriguez@email.com', '555-4000');

-- Insertar datos en casa
INSERT INTO casa (id_cliente, id_fraccionamiento, direccion, numero_casa) VALUES
(1, 1, 'Av. Principal 123', 'A-101'),
(2, 1, 'Av. Principal 123', 'B-205');

-- Insertar datos en trabajador
INSERT INTO trabajador (nombre, especialidad, telefono) VALUES
('Juan P rez', 'Jardiner a', '555-5000'),
('Ana Mart nez', 'Mantenimiento', '555-6000');

-- Insertar datos en servicio
INSERT INTO servicio (id_cliente, id_trabajador, tipo_servicio, fecha_servicio, descripcion
(1, 1, 'Podado de jard n', '2024-01-15', 'Mantenimiento de reas verdes'),
```

5. Conclusiones

En conclusión, esta práctica fue completamente desafiante ya que ha sido realizada desde un archivo LaTEX (utilizando la página "OVERLEAF"), el cual llevo tiempo comprender,más sin embargo, he utilizado material y conocimientos ya aquiridos en semestres pasados. Además, me ha fortalecido los conceptos que vienen en los libros ya investigados y sobre todo el los videos proporcionados por mi catedrático, el cual había tenido dificultades con las expresiones regulares y aplicación en código como puede ser FLEX.

Referencias Bibliográficas

References

[1] Date, C. J. (2001). *Introducción a los sistemas de bases de datos* (5a ed., Vol. 1). Addison-Wesley Iberoamericana.

[2] Elmasri, R., y Navathe, S. B. (2007). *Fundamentos de sistemas de bases de datos* (5a ed.). Prentice Hall.

[3] Piattini, M., y Marcos, E. (2001). *Diseño de bases de datos relacionales*. Alfaomega.