Introduction

In the current landscape of pet care, there is a large demand for dependable daycare services catering to our beloved dogs. In response, my initiative is dedicated to crafting a streamlined SQL database system utilizing SQL Developer, specifically designed to streamline the management of a doggy daycare facility. My primary objective is to create essential tables for client data, appointment scheduling, dog profiles. Moreover, I aim to include more advanced features like integrating automated reminders, comprehensive reporting features, and front-end application compatibility. This endeavor stems from personal experience; having recently adopted a dog and utilized Rover for overnight boarding, I wondered if I could create a database that could rival Rover and decrease the overall cost of boarding your pup.

Exploring

To start this topic, I did not have a great understanding of how databases in this industry worked. To gather information, I utilized Reddit review of Wag and Rover and talked to a friend that worked at a vet clinic. Using the knowledge gained from these interactions, I gained a firm understanding of how to create my database!

Building

To build the database, I designed tables for clients, dogs, and appointments, ensuring data integrity with foreign key constraints. I used mock data to populate the tables, including owner details, dog profiles, vaccination records, and appointment details. I implemented basic SQL queries for appointment scheduling, modification, and developed simple views for staff reference. Additionally, I explored advanced features such as stored procedures, triggers, and integration options with front-end applications.

Discovering

Through this project, I learned the importance of database design in addressing specific business needs and ensuring data integrity. I gained hands-on experience in SQL development, including query optimization and automation using stored procedures and triggers. While I successfully implemented the minimal viable project goal, I acknowledge areas for improvement, such as linking with front-end interfaces which would’ve required more time but now that I have more experience working with triggers and procedures, I can make the those faster leaving more time to work with front-end applications.

1. **Database Normalization:**

My project heavily relies on fundamental database design principles. For instance, I utilize primary keys, foreign keys, and constraints to ensure data integrity within the database schema. The tables for clients, dogs, and appointments demonstrate the normalization process, reducing redundancy and enhancing efficiency.

-- Example of creating tables with primary and foreign keys

CREATE TABLE Clients (

client\_id INT PRIMARY KEY,

name VARCHAR(50),

email VARCHAR(50),

phone VARCHAR(15)

);

CREATE TABLE Dogs (

dog\_id INT PRIMARY KEY,

client\_id INT,

name VARCHAR(50),

breed VARCHAR(50),

age INT,

vaccination\_status VARCHAR(20),

FOREIGN KEY (client\_id) REFERENCES Clients(client\_id) ON DELETE CASCADE

);

1. **SQL Query Optimization:**

In this project, I optimize SQL queries to enhance database performance. For example, when updating appointment times, I utilize stored procedures for efficient execution. This approach reduces network traffic and improves scalability.

-- Example of a stored procedure for updating appointment time

CREATE OR REPLACE PROCEDURE UpdateAppointmentTime(

p\_appointment\_id INT,

p\_new\_time TIMESTAMP

) AS

BEGIN

UPDATE Appointments

SET appointment\_time = p\_new\_time

WHERE appointment\_id = p\_appointment\_id;

END;

/

1. **SQL Triggers:**

I utilize triggers to automate certain actions within the database. For instance, I implement a trigger for appointment reminders, which logs messages when new appointments are scheduled. Triggers enhance data consistency and enforce business rules automatically.

-- Example trigger for appointment reminders

CREATE OR REPLACE TRIGGER reminder\_trigger

BEFORE INSERT ON Appointments

FOR EACH ROW

DECLARE

v\_appointment\_date DATE;

BEGIN

v\_appointment\_date := :NEW.appointment\_date;

-- Log a message when a new appointment is inserted

INSERT INTO Appointment\_Logs (appointment\_id, log\_message, log\_date)

VALUES (:NEW.appointment\_id, 'New appointment scheduled. Client ID: ' || :NEW.client\_id || ', Service: ' || :NEW.service || ', Date: ' || TO\_CHAR(v\_appointment\_date, 'YYYY-MM-DD') || ', Time: ' || TO\_CHAR(:NEW.appointment\_time, 'HH24:MI:SS'), SYSDATE);

END;

/

1. **Data Population and Manipulation:**

Throughout the project, I focus on populating the database with mock data and manipulating it to simulate real-world scenarios. For this project I used mockaroo to insert data into the Clients table and the Dogs table some examples of the inserted data are in the attached files named Clients.sql and Dogs.sql

1. **Using SQL Queries to join tables and search for entries:**

I utilize SQL queries to join tables and search for specific entries, enabling efficient data retrieval and analysis.

-- Query to retrieve dog and owner names by joining Dogs and Clients tables

SELECT Dogs.Name AS DogName, Clients.Name AS OwnerName

FROM Dogs

INNER JOIN Clients ON Dogs.Client\_ID = Clients.Client\_ID;

-- Query to retrieve all bookings for a specific dog

SELECT \* FROM Appointments WHERE dog\_id = 23;

This integrated approach illustrates the practical application of database concepts in developing a comprehensive solution for managing a doggy daycare facility.