NOAH'S PET CLINIC

Context

1. **System Analysis and Design**
2. **Database Designs and Implementation**
3. **Project Plan**
4. **SQL Schema**

Task: ***Part I – System Analysis and Designs***

1. Executive Summary:

Background: Noah's Pet Clinic is a specialized dog clinic located in rural Manchester. The clinic currently uses a paper-based system for managing appointments, pet and owner information, and consultation details, which is inefficient and not easily accessible to all staff.

Purpose: The aim of this report is to propose a solution to improve the clinic's appointment and patient management processes by developing a computer-based system.

Problem statement: The paper-based system currently in use at Noah's Pet Clinic is inefficient and not easily accessible to all staff, leading to difficulties in scheduling appointments and managing patient information.

Key points:

- Develop a computer-based system with features such as a calendar for scheduling appointments, a database for storing and organizing pet and owner information, and electronic forms for recording consultation details
- Implement measures to ensure the system adheres to constraints such as primary keys for unique identification of pets, unique email addresses for doctors, and foreign key constraints linking appointments to valid doctors
- The new system will greatly improve the efficiency and accessibility of the clinic's appointment and patient management processes

Summary: This report outlines a solution to improve the appointment and patient management processes at Noah's Pet Clinic by developing a computer-based system with appropriate features and constraints. Implementing this system will significantly enhance the efficiency and accessibility of the clinic's operations.

2.

a. Main goals for the new system being developed for Noah's Pet Clinic:

- Improve the efficiency and accessibility of the clinic's appointment and patient management processes
- Reduce the reliance on paper-based systems and manual data entry
- Provide a centralized, easily accessible location for storing and organizing pet and owner information
- Allow staff to easily schedule, view, and cancel appointments
- Enable staff to record and access consultation details electronically
- Facilitate communication between staff and pet owners, such as through appointment reminders or the ability to view past visit records

- Improve the accuracy and reliability of the clinic's data by reducing the risk of errors or lost information
- Provide an improved experience for both staff and pet owners by streamlining processes and reducing the time spent on administrative tasks.

b. Functional requirements and non-functional requirements.

In order to develop a system that is sufficient enough for the clinic to want to use to replace their existing paper system, it was essential to map out the key requirements of the system prior to development. The main requirements were classified based on functionality and limitations.

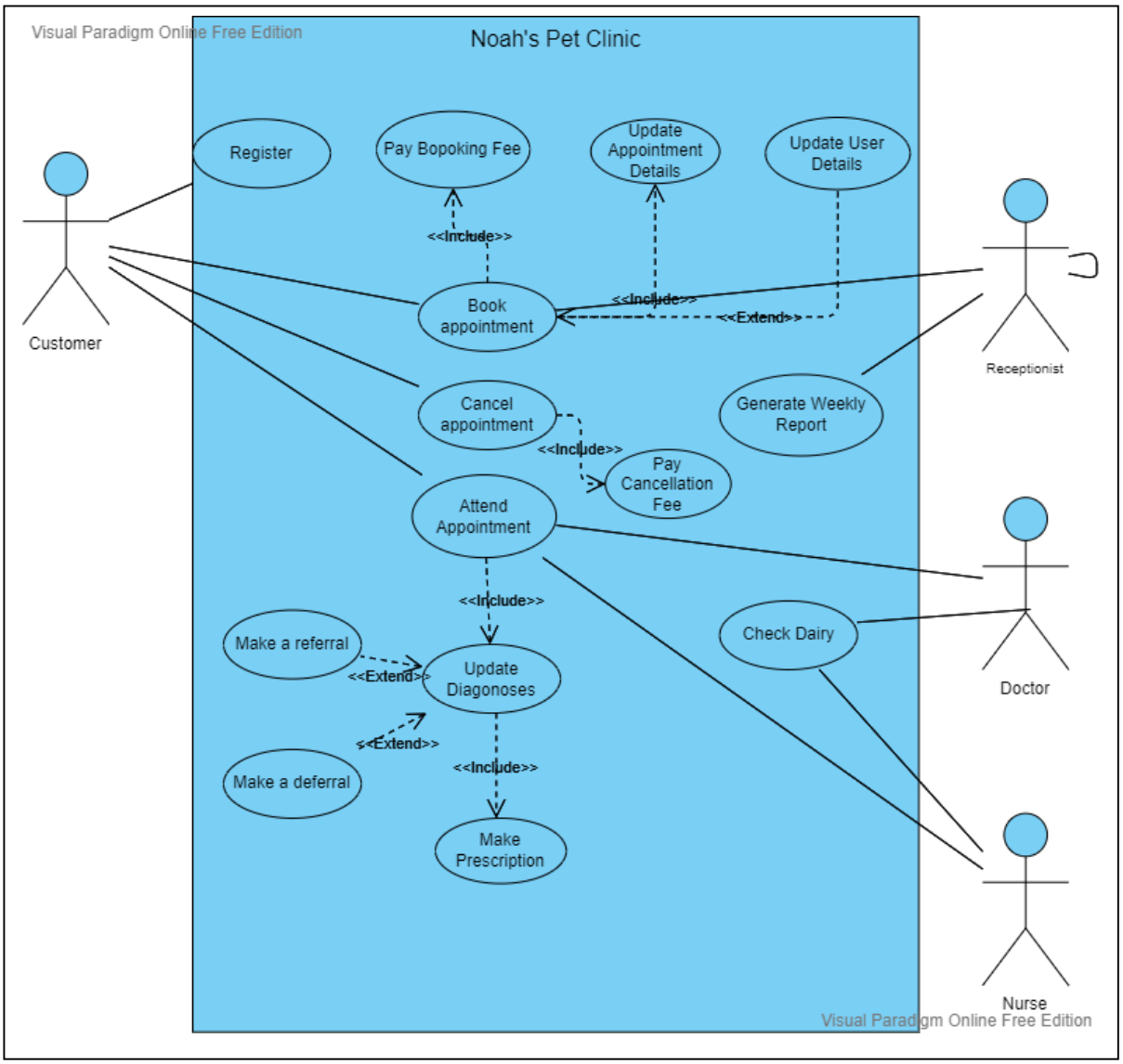
Functional requirements classified per actor:

- Administrator/ receptionist:
  - Collect information
  - Set up confirmation emails to be sent to pet owners (regarding appointment and payment reminders, notifications etc)
  - Help book an appointment for customers wishing to book by phone.
  - Check doctors and nurses' availability
  - Set up invoices
  - Update diary
  - Answer customer enquiries
- Manager:
  - Ensure things are working smoothly
  - Assist admin with answering enquiries
  - Check transactions and payments received (monitor balance)
- Doctors:
  - Attend appointment
  - Provide diagnosis
  - Decide outcome
    - Make a referral
    - Make a deferral
    - Prescribe medication.
- Nurses:
  - Attend appointments
  - Assist doctors
  - Help the customers
- Pet owners:
  - Turn up with their pets to the appointment
  - Ensure the information they provide is correct
  - Make payments
- System:
  - Alerts the admin to missing information
  - Allow for customers' self-booking
  - Process payments
  - Generate documents:
    - Invoices
    - Diagnosis descriptions
  - Store the information correctly

Non-functional requirements:

- Appointments can only be on Mondays or Fridays
- Unable to register pets older than 12 years.
- No more than two nurses are assigned per appointment.
- 4-5 doctors are available per day, appointments limited accordingly.
- A diagnosis has the date it was made, description and in case of medication prescribed, it may contain a recommendation for a pharmacy.
- Doctors each must have the following information stored:
    - Id → must be unique
    - Name
    - Office number
    - Telephone number
    - Email → must be unique
    - Part-time/full-time?
- Nurses each must have the following information in the system:
    - Name
    - Id → unique
- Each pet must have the following stored:
    - Id → unique and ranges between 1000 to 3000.
    - Owner's name
    - Name
    - Age
    - Owner's address
    - Breed
    - Weight
    - Colour
    - Gender.
- The cost of the appointment varies with the age of the pet:
    - £10: 0-5 years.
    - £15: 5-10 years.
    - £20: 10-12 years.

3. Use Case Diagram for the New Application



a. The Use Case diagram above is structure in such a way that the primary user (customers) are on the left side while the secondary and tertiary users (Receptionist, Doctors, and Nurses) are on the right side.

Customers:

- Register
- Book Appointments
- Cancel Appointments
- Attend Appointments

When customers log in, the system verifies their details. Depending on the details input, the system either log the user in or displays an error message with incorrect details.

When customers cancel appointments, they have to pay the cancellation fee

Receptionist:

- Book Appointments for customers
- Generate weekly reports

Doctors:

- Attend Appointments which includes making diagnoses.
- Check appointment diary.

Based on the diagnoses, the doctor can make a recommendation on the medicines, he can also make referrals and deferrals. Making deferrals means that the customer has to book another appointment.

Nurses:

- Attend Appointments
- Check appointment diary.

b. Use Case Specifications:

Use Case: Book appointment
Owner: Customer, Admin
Pre-Conditions
- Customer registration and Pet registration - Primary information provided for customer and pet and pet's unique ID generated. - Availability confirmed with the suitable doctor and nurse/s
Post-Conditions
- Fee payment received then the Appointment confirmation, receipt, Notifications/reminders generated. - Appointment takes place and diagnosis made.
Primary Path
1. Customer logs into the online system and checks the designated vet's availability. 2. Customer chooses the best suited time. 3. Customer confirms and clicks the button that takes to the checkout page. 4. Customer confirms the order and pays the fee assigned to their pet based on age. 5. Appointment confirmation generated.
Alternate Path
1.a. Customer's login information do not match the database. 1.a.1. Customer tries again. 1.a.2. Customer resits password 1.b. Customer chooses to book appointment by phone. 1.b.1. Customer calls the main desk 1.b.2. Admin asks the customer identifying questions to access customer's account 1.b.2.a. Customer's answers do not match 1.b.2.b. Admin helps retrieve account information 2.a. Customer's chosen time clashes with vet's availability.

2.a.1. System alerts the customer to the error.

2.a.2. Customer chooses a different time.

2.a.3. Customer chooses an available slot

2.a.4. Process proceeds to the next step

2.b. Customer booking by phone

2.b.1. Admin asks the customer about their preferred time for the appointment and checks against the doctor's availability.

2.b.1.a. Initial time chosen doesn't work

2.b.1.b. *go back to 1.b.3*

2.b.1.c. Customer chooses an available slot.

2.b.1.d. Process proceeds to the next step

4.a. Insufficient funds

4.a.1. Customer ensures the right amount is available in their account and that their card is not blocked then tries again.

Notes

None

Use Case: Attend appointment

Owner: Customer, Doctor, Nurses

Pre-Conditions

- Successful appointment booking and fee payment. NO CANCELLATION.

Post-Conditions

- Diagnosis finalised with details about symptoms, decision and where to get the medication.
- Further appointments booked OR pet discharged.

Primary Path

6. Customer, Doctor and Nurses attend the appointment with the pet.
7. Doctor asks for any symptoms observed and makes initial predictions.
8. Doctor carries out checks and tests with the help of the nurses.
9. Doctor notes the diagnosis details and decides on the next steps.

Alternate Path

1.a. Customer fails to bring the pet on time

1.a.1. Customer pays the cancellation fee

1.a.2. appointment rebooked and fees paid again.

1.b. Doctor fails to attend the appointment.

1.b.1. customer is given the choice to get a refund and rebook or never come again!

Notes

None

Use Case: Cancel appointment

Owner: Customer (the pet owner)

Pre-Conditions

- Customer completes personal registration along with pet's registration
- Customer successfully books an appointment.

Post-Conditions

- Appointment cancellation fee payment processing.
- Update cancelled appointment status.
- Rebook appointment.

Primary Path

1. Customer logs into the system
2. Customer navigates to the confirmed appointment.
3. Customer cancels the appointment.
4. System takes customer to the cancellation payment page
5. Customer makes a payment
6. Transaction complete, system confirms payment is received.
7. System updates the appointment diary for the day of the cancelled appointment.

Alternate Path

1.a. Customer's login information do not match the database.

1.a.1. Customer tries again.

1.a.2. Customer resits password

1.b. Customer chooses to book appointment by phone.

1.b.1. Customer calls the main desk

1.b.2. Admin asks the customer identifying questions to access customer's account

1.b.2.a. Customer's answers do not match

1.b.2.b. Admin helps retrieve account information

Notes

None

Use Case: Check diary

Owner: Doctor, Nurse/s

Pre-Conditions

- Customer registers their pets

- Customer books appointment successfully
- System updates record

Post-Conditions

- Doctors and Nurses attend the appointment on time.

Primary Path

1. Doctors/Nurses log into their accounts using their unique emails or IDs.
2. They navigate the system to the diary section to look at the booked appointments for the day.

Alternate Path

2.a. appointments might not show to the right staff.

2.a.1. Staff contact the administrator for possible issues and to request updates.

Notes

none

Use Case: Register (complete registration)

Owner: Pet owners (Customers)

Pre-Conditions

- Customers visit the website and click on Register

Post-Conditions

- Customers verify that the information displayed on their accounts are correct
- Customers make note of their password for future use.

Primary Path

3. Customers answer the questions to fill in the fields required (name, address, contact details).
4. Customers specify the number of pets they have and create a file for each (pet name, gender, colour, age)
5. Customers verify the details are correct
6. Customers create password and confirm registration.
7. System sends confirmation email to alert for the profile creation.

Alternate Path

1.a. Customers call the front desk to register

1.a.1. Admin asks customer about the required information and fill in the required fields for them.

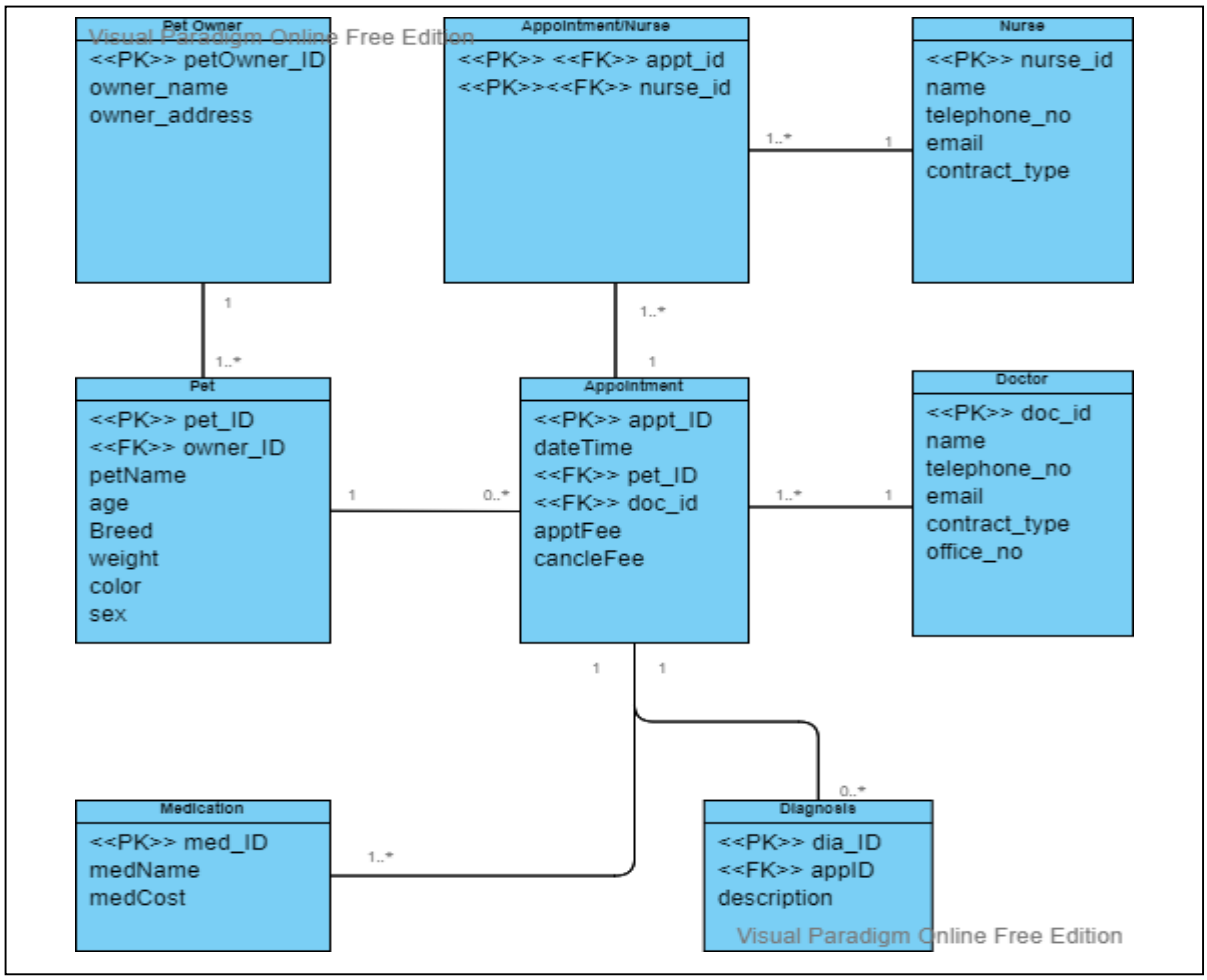
Notes

none

Use Case: weekly report generation
Owner: Administrator (Receptionist)
Pre-Conditions
- Customers booked appointments following their registrations. - Transactions completed successfully - Doctors and nurses as well as customers attended appointments and details of diagnoses updated on the system.
Post-Conditions
- Reports generated and raised to the management team - Clinic's performance is evaluated for profit analysis and/or improvement.
Primary Path
8. Administrators check the appointments that took place over the past week 9. Administrators access transaction history 10. Administrators gather the data observed in steps 1 and 2 and present them appropriately in an executive report.
Alternate Path
N/A
Notes
none

Task: ***Part II – Database Design and Implementation***

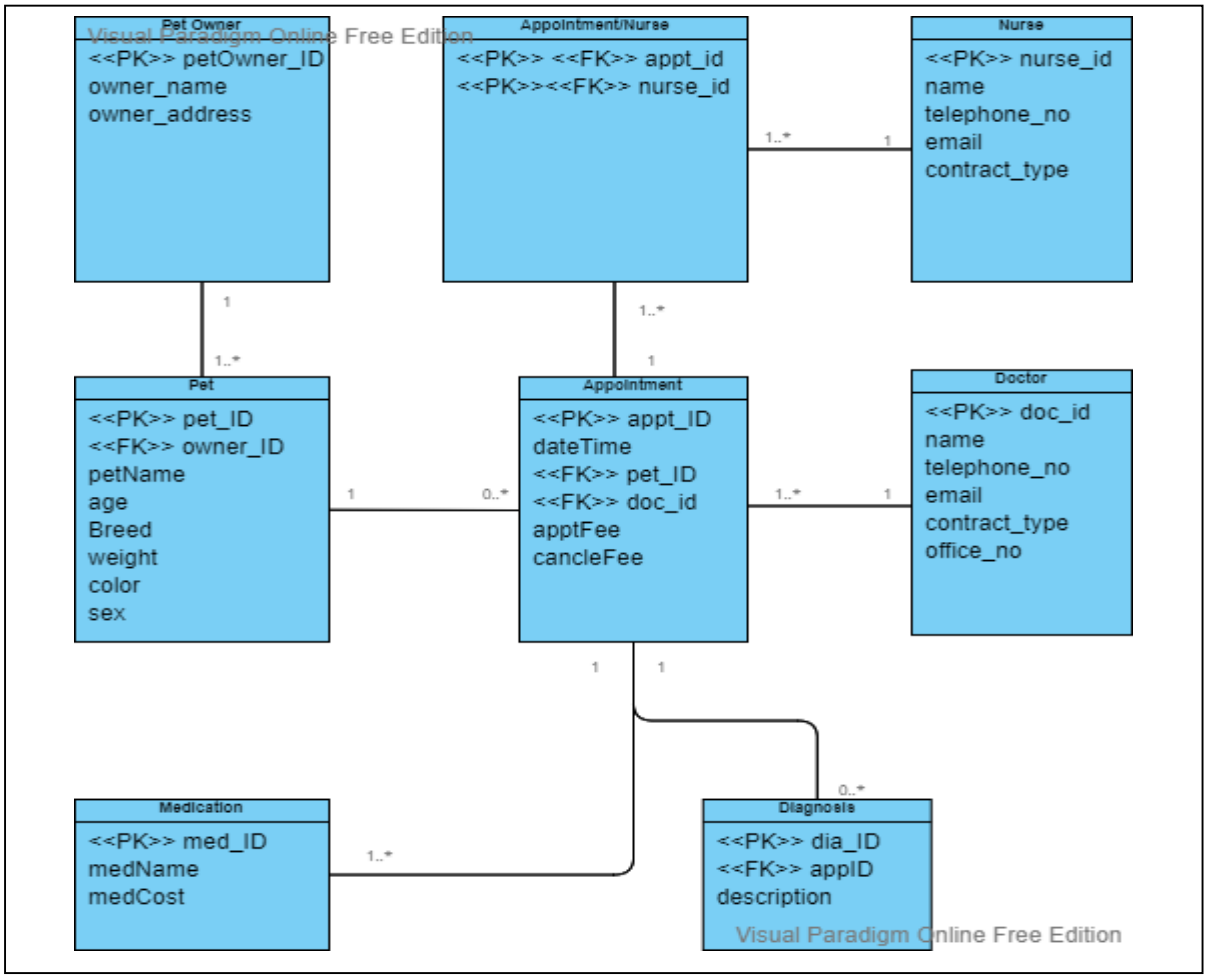
1. **Identify the entity and attribute and draw a top-down Entity Relational Diagram ERD for the System.**



2. **Normalise the appointment diary, the pet registration form, and the pet consultation card showing all stages of the normalisation process (UNF, 1NF, 2NF, 3NF). Then produce a bottom-up ERD from the tables produced at 3NF**

--See attached Excel file (Normalise the appointment diary and pet registration form.exe)—

3. **Merge top-down and bottom-up data models to produce a final ERD**



4. **CREATE Oracle tables from your design. Pay attention to constraints on particular attributes as outlined in the constraints section. Include the CREATE and DROP SQL statements in your report.**

   --See attached sql script (noahsPetClinic.sql)--

5. **Insert the data into tables as specified in the appendix. You should also insert FIVE (5) additional tuples in EACH table. All INSERT statements should be included in your script (see at base of this section). Include one example SQL INSERT statement for each table in your report.**

```
--
INSERT INTO pet_owners (owner_id, name, address, telephone_number, email_address) VALUES (1, 'John Smith', '123 Main Street', '555-555-1212', 'john@example.com');

INSERT INTO pets (pet_id, owner_id, name, type, gender, age, colour, weight) VALUES (1, 1, 'Fluffy', 'Cat', 'F', 5, 'White', 10);

INSERT INTO doctors (doctor_id, name, office_number, telephone_number, email_address, is_full_time) VALUES (1, 'Dr. Jane Smith', 101, '555-555-1212', 'jane@example.com',
'Y');

INSERT INTO nurses_new (nurse_id, name, office_number, telephone_number, email_address) VALUES (1, 'Nurse A', 101, '111-111-1111', 'nurseA@email.com');
INSERT INTO nurses_new (nurse_id, name, office_number, telephone_number, email_address) VALUES (2, 'Nurse B', 102, '222-222-2222', 'nurseB@email.com');

INSERT INTO appointments (appointment_id, pet_id, doctor_id, appt_date, appt_time, cost) VALUES (1, 1, 1, '05-Aug-2022', TO_DATE('13:00:00'), 15);

INSERT INTO appointment_nursed (id, appointment_id, nurse_id) VALUES (1, 1, 1);
INSERT INTO appointment_nursed (id, appointment_id, nurse_id) VALUES (2, 1, 2);

INSERT INTO diagnoses (diagnosis_id, appointment_id, diag_date, description, referral, deferral) VALUES (1, 1, '05-Aug-2022', 'Flu', 'N', 'N');

INSERT INTO medication (medication_id, appointment_id, name, dosage, frequency) VALUES (1, 1, 'Ibuprofen', '200mg', '3 times a day');
```

6. **Retrieve all data from each table using a SELECT query. Include screen shots of the query results in your report to show that all data has been inserted.**

```
SELECT * FROM pet_owners;
```

Results | Explain | Describe | Saved SQL | History

OWNER_ID	NAME	ADDRESS	TELEPHONE_NUMBER	EMAIL_ADDRESS
1	John Smith	123 Main Street	555-555-1212	john@example.com
2	Jane Doe	456 Market Avenue	555-555-1213	jane@example.com
3	Bob Williams	789 Elm Street	555-555-1214	bob@example.com
4	Sally Johnson	321 Oak Avenue	555-555-1215	sally@example.com
5	Tom Brown	654 Pine Street	555-555-1216	tom@example.com
6	Sarah Davis	987 Cedar Boulevard	555-555-1217	sarah@example.com
7	Mike Thompson	246 Maple Drive	555-555-1218	mike@example.com

7 rows returned in 0.00 seconds Download

```
SELECT * FROM pets;
```

Results | Explain | Describe | Saved SQL | History

PET_ID	OWNER_ID	NAME	TYPE	GENDER	AGE	COLOUR	WEIGHT
1001	1	Fluffy	Cat	F	5	White	10
1002	1	Buddy	Dog	M	3	Black	20
1003	2	Mittens	Cat	F	2	Grey	8
1004	2	Rover	Dog	M	4	Brown	25
1005	3	Smokey	Cat	M	6	Black	15
1006	3	Daisy	Dog	F	1	White	10
1007	4	Tiger	Cat	M	3	Orange	12
1008	4	Lucy	Dog	F	2	Brown	20
1009	5	Garfield	Cat	M	4	Grey	15
1010	5	Charlie	Dog	M	1	Black	10

10 rows returned in 0.01 seconds Download

```
SELECT * FROM doctors;
```

Results | Explain | Describe | Saved SQL | History

DOCTOR_ID	NAME	OFFICE_NUMBER	TELEPHONE_NUMBER	EMAIL_ADDRESS	IS_FULL_TIME
1	Dr. Jane Smith	101	555-555-1212	jane@example.com	Y
2	Dr. John Doe	102	555-555-1213	john@example.com	Y
3	Dr. Bob Williams	103	555-555-1214	bob@example.com	Y
4	Dr. Sally Johnson	104	555-555-1215	sally@example.com	N

4 rows returned in 0.01 seconds Download

```
SELECT * FROM nurses;
```

Results | Explain | Describe | Saved SQL | History

NURSE_ID	NURSE_NAME	TELEPHONE_NUMBER	EMAIL	CONTRACT_TYPE
5000	Titilayo Aluko	445576276255	titi@aluko.com	Part-Time
5001	Anabell Sandra	447792929001	anabell@sandra.com	Full-Time
5002	Jessica Rains	441372142781	jessica@rains.com	Full-Time

3 rows returned in 0.00 seconds Download

```
SELECT * FROM appointments;
```

Results | Explain | Describe | Saved SQL | History

APPOINTMENT_ID	PET_ID	DOCTOR_ID	APPT_DATE	APPT_TIME	COST
1	1001	1	05-Aug-2022	01-Jan-2023	15
2	1002	2	05-Aug-2022	01-Jan-2023	5
3	1003	3	09-Aug-2022	01-Jan-2023	5
4	1004	4	16-Aug-2022	01-Jan-2023	5
5	1005	1	07-Oct-2022	01-Jan-2023	15
6	1006	2	11-Oct-2022	01-Jan-2023	5
7	1007	3	30-Nov-2022	01-Jan-2023	5
8	1008	4	27-Nov-2022	01-Jan-2023	5
9	1002	1	24-Nov-2022	01-Jan-2023	5
10	1002	1	29-Aug-2022	01-Jan-2023	5

More than 10 rows available. Increase rows selector to view more rows.

10 rows returned in 0.00 seconds Download

```
SELECT * FROM appointment_nurses;
```

Results | Explain | Describe | Saved SQL | History

ID	APPOINTMENT_ID	NURSE_ID
1	1	1
2	1	2
3	2	3
4	2	4
5	3	5
6	3	6
7	4	7
8	4	8
9	5	1
10	5	2

More than 10 rows available. Increase rows selector to view more rows.

10 rows returned in 0.01 seconds Download

```
SELECT * FROM diagnoses;
```

Results | Explain | Describe | Saved SQL | History

DIAGNOSIS_ID	APPOINTMENT_ID	DIAG_DATE	DESCRIPTION	REFERRAL	DEFERRAL
1	1	05-Aug-2022	Flu	N	N
2	2	05-Aug-2022	Allergy	Y	N
3	3	09-Aug-2022	Skin infection	N	Y
4	4	16-Aug-2022	socialisation	N	N
5	5	07-Oct-2022	Bladder infection	Y	N
6	6	11-Oct-2022	Liver disease	N	Y
7	7	30-Nov-2022	socialisation	N	N
8	8	27-Nov-2022	dental	N	Y
9	9	24-Nov-2022	socialisation	N	N
10	10	29-Aug-2022	socialisation	N	N

More than 10 rows available. Increase rows selector to view more rows.

10 rows returned in 0.00 seconds Download

```
SELECT * FROM medication;
```

Results | Explain | Describe | Saved SQL | History

MEDICATION_ID	APPOINTMENT_ID	NAME	DOSAGE	FREQUENCY
1	1	Ibuprofen	200mg	3 times a day
2	2	Antihistamine	10mg	2 times a day
3	3	Antibiotic ointment	apply topically	3 times a day
4	4	Vitalmin Active	10 tablets	1 time a day
5	5	Antibiotic	500mg	3 times a day
6	6	Liver support suppliement	2 tablets	2 times a day
7	7	Vitalmin Active	10 tablets	1 time a day
8	8	Liver support suppliement	2 tablets	2 times a day
9	9	Vitalmin Active	10 tablets	1 times a day
10	10	Vitalmin Active	10 tablets	1 times a day

More than 10 rows available. Increase rows selector to view more rows.

10 rows returned in 0.00 seconds Download

7. **A large vet franchise is interested in your solution. Critically analyse your current design and the state of the current database and suggest at least three ways in which you could redesign the database to improve it for a larger client. Justify your decisions**

As it stands, the current database design is sufficient for a small-scale operation, but it may not be scalable enough for a larger vet franchise. Here are three ways in which the database could be redesigned to improve it for a larger client:

a. I would suggest normalizing the database to reduce redundancy and improve data integrity. For example, the "doctors" and "nurses_new" tables could be combined into a single "employees" table, with a column indicating the employee's role (doctor or nurse).
b. I would recommend adding a "locations" table: Currently, the database does not track which location each appointment or diagnosis took place at. A larger franchise may have multiple locations, and it would be useful to be able to track which location each appointment or diagnosis occurred at. To address this, we could add a "locations" table that stores information about each location (e.g. address, phone number, etc.). We could then add a foreign key field to the appointments and diagnoses tables that reference the location where the appointment or diagnosis took place. This would allow the franchise to track which location each appointment or diagnosis occurred at and make it easier to generate reports for specific locations.
c. I would recommend adding more granularity to the appointments table: Currently, the appointments table only has the date and time of the appointment, but a larger franchise may have multiple appointments at the same time. To address this, we could add a new field to the appointments table called "appointment_slot" that indicates the specific time slot for the appointment (e.g. 9:00-9:30 am). This would allow the franchise to schedule multiple appointments at the same time without any conflicts.
d. I would suggest normalizing the pet_owners table: Currently, the pet_owners table stores all of the owner's information in a single row. However, as the franchise grows, it may be necessary to store more information about the pet owners (e.g. additional contact information, emergency contact details, etc.). To address this, we could normalize the pet_owners table by breaking it up into multiple tables (e.g. one table for contact information, another for emergency contact details, etc.). This would allow the franchise to store more information about their pet owners without making the table too large or unwieldy.
e. +0oImplementing security measures: For a larger client, it may be necessary to implement additional security measures to protect the data in the database. This could include measures such as encrypting sensitive data, implementing user access controls, and regularly backing up the database.

Overall, these changes would enable the franchise to better manage and track its data as it grows, and ensure that the database remains efficient and scalable as the business expands.

Task: ***Part III – Query and Implementation***

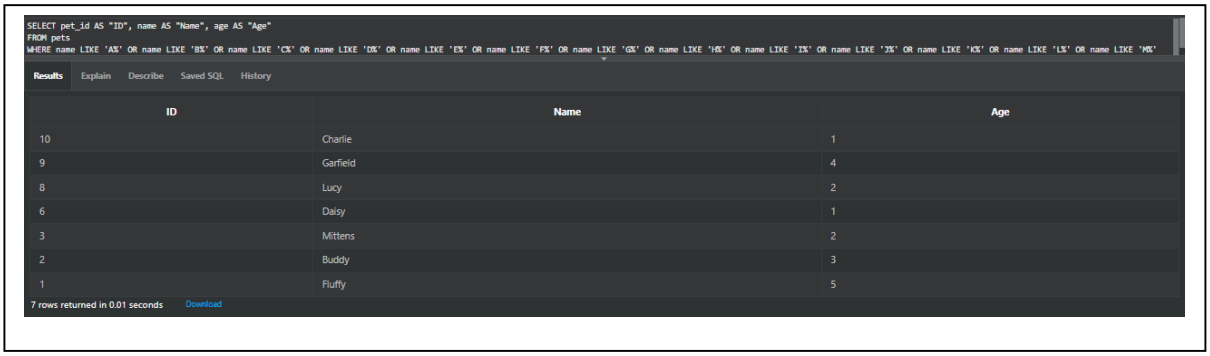
Testing and Query Explanation

1. *Write an SQL statement that lists the pet id, pet name, pet age showing the columns as "ID", "Name" and "Age". Their names should be those starting with letters A to M. Sort the results by the pet id in descending order.*

```
SELECT pet_id AS "ID", name AS "Name", age AS "Age"
FROM pets
WHERE name LIKE 'A%' OR name LIKE 'B%' OR name LIKE 'C%' OR name LIKE 'D%' OR name LIKE 'E%' OR name LIKE 'F%' OR name LIKE 'G%' OR
name LIKE 'H%' OR name LIKE 'I%' OR name LIKE 'J%' OR name LIKE 'K%' OR name LIKE 'L%' OR name LIKE 'M%'
ORDER BY pet_id DESC;
```

Explanation:

This query selects the pet_id, name, and age columns from the pets table and renames them as "ID", "Name", and "Age", respectively. It filters the rows to only include those where the name column starts with any of the letters A through M (case-insensitive). Finally, it orders the rows in descending order by the pet_id column.



```
SELECT pet_id AS "ID", name AS "Name", age AS "Age"
FROM pets
WHERE name LIKE 'A%' OR name LIKE 'B%' OR name LIKE 'C%' OR name LIKE 'D%' OR name LIKE 'E%' OR name LIKE 'F%' OR name LIKE 'G%' OR name LIKE 'H%' OR name LIKE 'I%' OR name LIKE 'J%' OR name LIKE 'K%' OR name LIKE 'L%' OR name LIKE 'M%'
```

Results | Explain | Describe | Saved SQL | History

ID	Name	Age
10	Charlie	1
9	Garfield	4
8	Lucy	2
6	Daisy	1
3	Mittens	2
2	Buddy	3
1	Fluffy	5

7 rows returned in 0.01 seconds Download

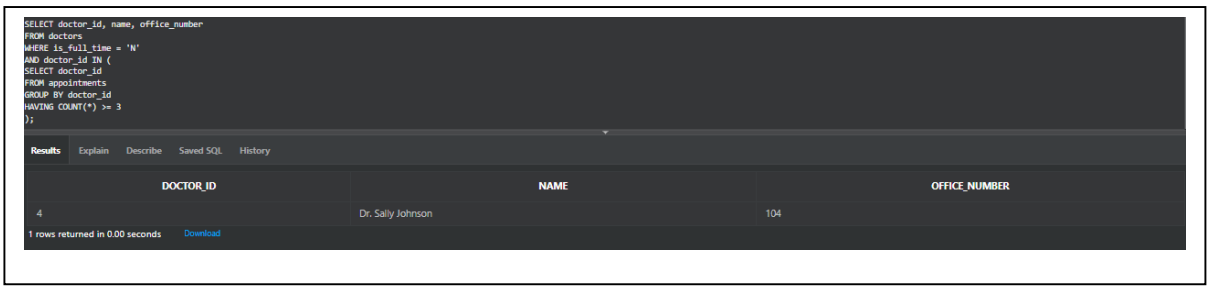
2. *Write an SQL statement that shows the most overworked part-time vet doctor, i.e., any vet with 3 or more appointments for pet clinics, listing details such as id, name and office number.*

```
1  SELECT doctor_id, name, office_number
2  FROM doctors
3  WHERE is_full_time = 'N'
4  AND doctor_id IN (
5  SELECT doctor_id
6  FROM appointments
7  GROUP BY doctor_id
```

Explanation:

This query selects the doctor_id, name, and office_number of doctors who are not full-time (is_full_time = 'N') and have handled at least 3 appointments.

It does this by first selecting all doctors who are not full-time from the doctors table. It then filters this result set using the IN operator, which allows the query to check if the doctor_id of each doctor is present in a subquery. The subquery selects the doctor_id of all doctors who have handled at least 3 appointments by grouping the appointments by doctor_id and counting the number of appointments for each doctor_id. Only doctor_ids that have a count of at least 3 are included in the subquery's result set, which is used to filter the main query's result set. Finally, the resulting rows are ordered by doctor_id in descending order.



```
SELECT doctor_id, name, office_number
FROM doctors
WHERE is_full_time = 'N'
AND doctor_id IN (
SELECT doctor_id
FROM appointments
GROUP BY doctor_id
HAVING COUNT(*) >= 3
);
```

Results | Explain | Describe | Saved SQL | History

DOCTOR_ID	NAME	OFFICE_NUMBER
4	Dr. Sally Johnson	104

1 rows returned in 0.00 seconds Download

3. *Write an SQL query that lists pet details with 2 or more appointments between 2nd Jan 2022 and 26th Sep 2022. The columns should include pet id, name and a count of the number of appointments.*

```
SELECT p.pet_id as "ID", p.name as "Name", COUNT(a.appointment_id) as "Appointments"
FROM pets p
JOIN appointments a ON p.pet_id = a.pet_id
WHERE a.appt_date BETWEEN '02-Jan-2022' AND '26-Sep-2022'
GROUP BY p.pet_id, p.name
```

Explanation:

This query selects the pet ID and name, and the number of appointments for each pet, from the pets and appointments tables, where the appointment date is between '02-Jan-2022' and '26-Sep-2022'. The query groups the results by pet ID and name, and only displays results where the number of appointments is greater than or equal to 2.

```
SELECT p.pet_id as "ID", p.name as "Name", COUNT(a.appointment_id) as "Appointments"
FROM pets p
JOIN appointments a ON p.pet_id = a.pet_id
WHERE a.appt_date BETWEEN '02-Jan-2022' AND '26-Sep-2022'
GROUP BY p.pet_id, p.name
HAVING COUNT(a.appointment_id) >= 2;
```

Results | Explain | Describe | Saved SQL | History

ID	Name	Appointments
2	Buddy	2

1 rows returned in 0.00 seconds Download

4. *Write an SQL query that finds pet id, name and cost of appointments, such that the cost of the appointment is less than the average appointment cost of all pet appointments.*

```
SELECT p.pet_id, p.name, a.cost
FROM pets p
JOIN appointments a ON p.pet_id = a.pet_id
GROUP BY p.pet_id, p.name, a.cost
HAVING a.cost < (SELECT AVG(cost) FROM appointments);
```

Explanation:
This query selects the pet id, name, and cost of appointments for each pet, and then groups them by pet id, name, and cost. It then uses a HAVING clause to filter the results and only include rows where the cost of the appointment is less than the average cost of all appointments. This means that it will show the pets that have had appointments that are cheaper than the average cost of all appointments.

```
SELECT p.pet_id, p.name, a.cost
FROM pets p
JOIN appointments a ON p.pet_id = a.pet_id
GROUP BY p.pet_id, p.name, a.cost
HAVING a.cost < (SELECT AVG(cost) FROM appointments);
```

Results | Explain | Describe | Saved SQL | History

PET_ID	NAME	COST
7	Tiger	5
9	Garfield	5
3	Mittens	5
4	Rover	5
2	Buddy	5
6	Daisy	5
8	Lucy	5

7 rows returned in 0.01 seconds Download

5. *Write an SQL statement that lists a doctor id, status, the total number of appointments he/she has handled, the pet involved (id, name) and the dates these took place. The query should only show cases whose diagnoses involved "socialisation" or "dental" work and where the appointment date was at least 4 weeks ago.*

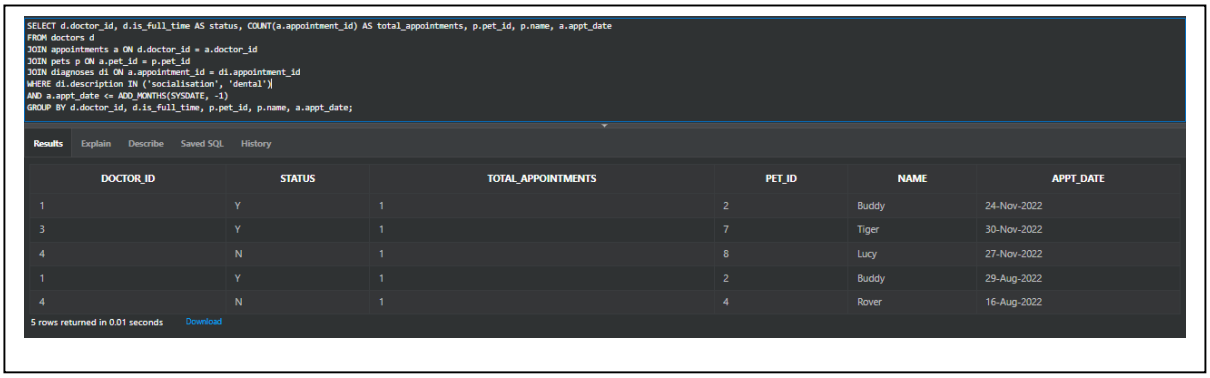
```
SELECT d.doctor_id, d.is_full_time AS status, COUNT(a.appointment_id) AS total_appointments, p.pet_id, p.name, a.appt_date
FROM doctors d
JOIN appointments a ON d.doctor_id = a.doctor_id
JOIN pets p ON a.pet_id = p.pet_id
JOIN diagnoses di ON a.appointment_id = di.appointment_id
WHERE di.description IN ('socialisation', 'dental')
AND a.appt_date <= ADD_MONTHS(SYSDATE, -1)
GROUP BY d.doctor_id, d.is_full_time, p.pet_id, p.name, a.appt_date;
```

Explanation:

This query retrieves the doctor ID, status, total number of appointments handled, pet ID and name, and appointment date of all appointments that meet the following criteria:

- The diagnosis description is either 'socialisation' or 'dental'.
- The appointment date is at least 1 month before the current date.

The results are grouped by doctor ID, status, pet ID, name, and appointment date. The status column is an alias for the is_full_time column from the doctors table, and the total_appointments column is a count of the number of appointment IDs from the appointments table for each group.



```
SELECT d.doctor_id, d.is_full_time AS status, COUNT(a.appointment_id) AS total_appointments, p.pet_id, p.name, a.appt_date
FROM doctors d
JOIN appointments a ON d.doctor_id = a.doctor_id
JOIN pets p ON a.pet_id = p.pet_id
JOIN diagnoses di ON a.appointment_id = di.appointment_id
WHERE di.description IN ('socialisation', 'dental')
AND a.appt_date <= ADD_MONTHS(SYSDATE, -1)
GROUP BY d.doctor_id, d.is_full_time, p.pet_id, p.name, a.appt_date;
```

Results | Explain | Describe | Saved SQL | History

DOCTOR_ID	STATUS	TOTAL_APPOINTMENTS	PET_ID	NAME	APPT_DATE
1	Y	1	2	Buddy	24-Nov-2022
3	Y	1	7	Tiger	30-Nov-2022
4	N	1	8	Lucy	27-Nov-2022
1	Y	1	2	Buddy	29-Aug-2022
4	N	1	4	Rover	16-Aug-2022

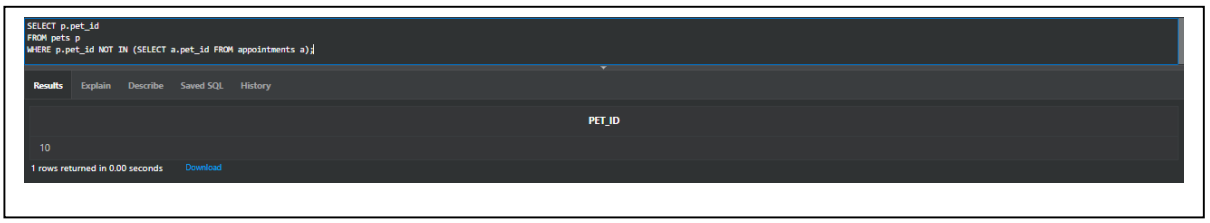
5 rows returned in 0.01 seconds Download

6. *Write an SQL query that finds pet ids with no appointments or diagnoses so far.*

```
SELECT p.pet_id
FROM pets p
WHERE p.pet_id NOT IN (SELECT a.pet_id FROM appointments a);
```

Explanation:

This query will select the pet id's from the pets table that are not in the appointments table. It will return a list of pet id's that have no appointments.



```
SELECT p.pet_id
FROM pets p
WHERE p.pet_id NOT IN (SELECT a.pet_id FROM appointments a);
```

Results | Explain | Describe | Saved SQL | History

PET_ID
10

1 rows returned in 0.00 seconds Download

Noah's Pet Cline Project Plan/Report

Introduction

Noah's Pet Clinic (NPC) is a specialized dog clinic located in rural Manchester. The clinic offers veterinary services to a wide range of pets. The clinic is currently facing a problem with its outdated and inefficient paper-based appointment and registration system. This system requires pet owners to call NPC reception to make appointments for their pets. The receptionist would then make a note of the appointments in the NPC appointment diary. This system is not very accessible or efficient, and it can be difficult for the clinic staff to constantly find and access the diary to search for appointment details. Additionally, there is a filing cabinet for Pet registration forms for the Owner to fill and the receptionist files all the Owner and Pet registration forms in a filing cabinet in alphabetical order (under the owner's name). The managing director at PNC is looking to upgrade this system to a digital one.

In order to achieve this goal, a team of 4 individuals, Ibrahim (22450374), Rama (22531617), Chinonye (22469807), and Promise (22504972), was formed. Each member of the team had specific tasks to complete in order to successfully implement the new digital system. This report will outline the project goals, timeline, methodology, and results of the project.

Project Goal

1. Implement a user-friendly and efficient digital system for managing appointments, record keeping, and registration at Noah's Pet Clinic.
2. Ensure that the new system is fully compliant with relevant regulations.
3. Improve accessibility and ease of use for clinic staff when searching for appointment and registration details.
4. Streamline the process of making and cancelling appointments, and reduce the likelihood of appointment diary misplacement.
5. Facilitate easier storage and retrieval of pet and owner registration information.
6. Provide a more accurate and reliable system for tracking the cost of appointments, consultations, and medication.
7. Increase the overall efficiency and effectiveness of the clinic's operations.
8. Improve the overall experience for pet owners when making appointments and registering their pets at Noah's Pet Clinic.

Project Timeline:

Week 1-2:

- Identify functional and non-functional requirements
- List the main goals of the system

Week 3-4:

- Draw a Use-case Diagram
- Produce a use case specification for at least one UC per student

Week 5-6:

- Identify the entities and attributes and draw a top-down Entity Relationship Diagram (ERD) for the system.
- Normalize the appointment diary, the registration forms and consult document cards

Week 7-8:

- Design and develop the SQL database for the system

Week 9-10:

- Test the system
- Address any bugs or issues that arise
- Finalize the project and submit the report.

Methodology:

The project team used the Waterfall methodology for this project. Waterfall methodology is an iterative and incremental approach to software development that emphasizes flexibility, collaboration, and customer satisfaction. This methodology allowed the team to quickly adapt to changing requirements and make adjustments as needed.

Tasks were divided among team members based on their skills and expertise. Each team member was responsible for completing specific tasks and reporting their progress to the team. Team members communicated and collaborated regularly on Teams and JIRA to ensure that the project was on track and to address any issues that arose. ***(See table 1 for the breakdown of tasks completed by each member of the team).***

The team used a variety of tools and technologies to complete the project, including:

Visual Paradigm Tool – For designing the system Use Case Diagrams.

Visual Paradigm Tool – Entity Relationship Diagrams for data modelling

Microsoft Excel – Normalizing Entity Relational Diagram.

ORACLE APEX – Creating SQL database.

Microsoft Word – Compiling project report

JIRA – Assigning and monitoring tasks.

Teams – Meetings.

The team also sought feedback from the lecturer throughout the project to ensure that the new system met the intended needs and was user-friendly.

Result

The new system has significantly improved the accessibility and ease of use when searching for appointment and registration details. Appointment and registration processes have been streamlined, and the likelihood of appointment diary misplacement has been reduced. The new system also facilitates the storage and retrieval of pet and owner registration information. The new system also provides more accurate and reliable tracking of the cost of appointments, consultations, and medication. This will help the clinic to better manage its finances and improve overall efficiency.

Table 1: Tasks completed by team members.

S/N	Tasks	Ibrahim	Rama	Chinonye	Promise
1	Identify Functional and Non-Functional Requirements		✓		
2	List the main goals of the system	✓	✓	✓	✓

3	Draw Use-case Diagram	✓	✓		
4	Produce a use case specification for at least one UC per student		✓		
5	Identify the entities and attributes and draw a top-down Entity Relationship Diagram (ERD) for the system.	✓	✓	✓	✓
6	Normalise the appointment diary, the pet registration form, the pet consultation card showing all stages of the normalisation process(UNF, 1NF, 2NF, 3NF). Then produce a bottom up ERDfrom the tables produced at 3NF	✓	✓	✓	✓
7	Merge top-down and bottom-up data models to produce a final ERD.	✓	✓	✓	✓
8	CREATE Oracle tables from your design. Pay attention to constraints on particular attributes as outlined in the constraints section. Include the CREATE and DROP SQL statements in your report.	✓			
9	Insert the data into tables. You should also insert FIVE (5)additional tuples in EACH table. All INSERT statements should be included in your script.	✓			
10	Retrieve all data from each table using a SELECT query. Include	✓			

	screenshots of the query results in your report to show that all data has been inserted.				
11	Suggest three (3) ways in which the database could be redesigned.				✓
12	Write an SQL statement that implements the queries				✓
13	Written an executive summary for your project report			✓	
14	Included a bullet pointed list to explain the requirements of the new system			✓	
15	Final report compilation.	✓	✓	✓	✓
16	Selling to the franchise that made an offer.			✓	✓

Conclusion

The project team successfully implemented a new digital system for managing appointments, record keeping, and registration at Noah's Pet Clinic. The new system is efficient, and fully compliant with relevant regulations. It improves accessibility and ease of use for clinic staff when searching for appointment and registration details, streamlines the process of making and cancelling appointments, and facilitates the storage and retrieval of pet and owner registration information.

In conclusion, the new digital system has met the project's objectives and helped Noah's Pet Clinic to improve its services.

APPENDIX: SQL Schema

DROP TABLE medication;

DROP TABLE diagnoses;

DROP TABLE appointment_nursed;

DROP TABLE appointments;

DROP TABLE nurses_new;

DROP TABLE doctors;

```
DROP TABLE pets;
DROP TABLE pet_owners;

CREATE TABLE pet_owners (
  owner_id INTEGER PRIMARY KEY,
  name VARCHAR(255) NOT NULL,
  address VARCHAR(255) NOT NULL,
  telephone_number VARCHAR(255) NOT NULL,
  email_address VARCHAR(255) NOT NULL
);

CREATE TABLE pets (
  pet_id INTEGER PRIMARY KEY,
  owner_id INTEGER NOT NULL,
  name VARCHAR(255) NOT NULL,
  type VARCHAR(255) NOT NULL,
  breed VARCHAR(255) NOT NULL,
  gender CHAR(1) NOT NULL,
  age INTEGER NOT NULL,
  colour VARCHAR(255) NOT NULL,
  weight INTEGER NOT NULL,
  FOREIGN KEY (owner_id) REFERENCES pet_owners (owner_id)
);

CREATE TABLE doctors (
  doctor_id INTEGER PRIMARY KEY,
  name VARCHAR(255) NOT NULL,
  office_number INTEGER NOT NULL,
  telephone_number VARCHAR(255) NOT NULL,
  email_address VARCHAR(255) UNIQUE NOT NULL,
  is_full_time CHAR(1) NOT NULL
);
```

```
CREATE TABLE nurses_new (
  nurse_id INTEGER PRIMARY KEY,
  name VARCHAR(255) NOT NULL,
  office_number INTEGER NOT NULL,
  telephone_number VARCHAR(255) NOT NULL,
  email_address VARCHAR(255) NOT NULL
);

CREATE TABLE appointments (
  appointment_id INTEGER PRIMARY KEY,
  pet_id INTEGER NOT NULL,
  doctor_id INTEGER NOT NULL,
  appt_date DATE NOT NULL,
  appt_time DATE NOT NULL,
  cost INTEGER NOT NULL,
  FOREIGN KEY (pet_id) REFERENCES pets (pet_id),
  FOREIGN KEY (doctor_id) REFERENCES doctors (doctor_id)
);

CREATE TABLE appointment_nursed (
  id INTEGER PRIMARY KEY,
  appointment_id INTEGER NOT NULL,
  nurse_id INTEGER NOT NULL,
  FOREIGN KEY (appointment_id) REFERENCES appointments (appointment_id),
  FOREIGN KEY (nurse_id) REFERENCES nurses_new (nurse_id)
);

CREATE TABLE diagnoses (
  diagnosis_id INTEGER PRIMARY KEY,
  appointment_id INTEGER NOT NULL,
  diag_date DATE NOT NULL,
```

```
  description VARCHAR(255) NOT NULL,
  referral CHAR(1) NOT NULL,
  deferral CHAR(1) NOT NULL,
  FOREIGN KEY (appointment_id) REFERENCES appointments (appointment_id)
);

CREATE TABLE medication (
  medication_id INTEGER PRIMARY KEY,
  appointment_id INTEGER NOT NULL,
  name VARCHAR(255) NOT NULL,
  dosage VARCHAR(255) NOT NULL,
  frequency VARCHAR(255) NOT NULL,
  FOREIGN KEY (appointment_id) REFERENCES appointments (appointment_id)
);

--

INSERT INTO pet_owners (owner_id, name, address, telephone_number, email_address) VALUES (1, 'John Smith', '123 Main Street', '555-555-1212', 'john@example.com');

INSERT INTO pet_owners (owner_id, name, address, telephone_number, email_address) VALUES (2, 'Jane Doe', '456 Market Avenue', '555-555-1213', 'jane@example.com');

INSERT INTO pet_owners (owner_id, name, address, telephone_number, email_address) VALUES (3, 'Bob Williams', '789 Elm Street', '555-555-1214', 'bob@example.com');

INSERT INTO pet_owners (owner_id, name, address, telephone_number, email_address) VALUES (4, 'Sally Johnson', '321 Oak Avenue', '555-555-1215', 'sally@example.com');

INSERT INTO pet_owners (owner_id, name, address, telephone_number, email_address) VALUES (5, 'Tom Brown', '654 Pine Street', '555-555-1216', 'tom@example.com');

INSERT INTO pet_owners (owner_id, name, address, telephone_number, email_address) VALUES (6, 'Sarah Davis', '987 Cedar Boulevard', '555-555-1217', 'sarah@example.com');

INSERT INTO pet_owners (owner_id, name, address, telephone_number, email_address) VALUES (7, 'Mike Thompson', '246 Maple Drive', '555-555-1218', 'mike@example.com');

--

INSERT INTO pets (pet_id, owner_id, name, type, breed, gender, age, colour, weight) VALUES (1001, 1, 'Fluffy', 'Cat', 'Sianene', 'F', 5, 'White', 10);
```

INSERT INTO pets (pet_id, owner_id, name, type, breed, gender, age, colour, weight) VALUES (1002, 1, 'Buddy', 'Dog', 'Chiwawa', 'M', 3, 'Black', 20);

INSERT INTO pets (pet_id, owner_id, name, type, breed, gender, age, colour, weight) VALUES (1003, 2, 'Mittens', 'Cat', 'Ragdoll', 'F', 2, 'Grey', 8);

INSERT INTO pets (pet_id, owner_id, name, type, breed, gender, age, colour, weight) VALUES (1004, 2, 'Rover', 'Dog', 'Bull dog', 'M', 4, 'Brown', 25);

INSERT INTO pets (pet_id, owner_id, name, type, breed, gender, age, colour, weight) VALUES (1005, 3, 'Smokey', 'Cat', 'Persian', 'M', 6, 'Black', 15);

INSERT INTO pets (pet_id, owner_id, name, type, breed, gender, age, colour, weight) VALUES (1006, 3, 'Daisy', 'Dog', 'Terrier', 'F', 1, 'White', 10);

INSERT INTO pets (pet_id, owner_id, name, type, breed, gender, age, colour, weight) VALUES (1007, 4, 'Tiger', 'Cat', 'Birman', 'M', 3, 'Orange', 12);

INSERT INTO pets (pet_id, owner_id, name, type, breed, gender, age, colour, weight) VALUES (1008, 4, 'Lucy', 'Dog', 'Alsatian', 'F', 2, 'Brown', 20);

INSERT INTO pets (pet_id, owner_id, name, type, breed, gender, age, colour, weight) VALUES (1009, 5, 'Garfield', 'Cat', 'Birman', 'M', 4, 'Grey', 15);

INSERT INTO pets (pet_id, owner_id, name, type, breed, gender, age, colour, weight) VALUES (1010, 5, 'Charlie', 'Dog', 'Alsatian', 'M', 1, 'Black', 10);

INSERT INTO doctors (doctor_id, name, office_number, telephone_number, email_address, is_full_time) VALUES (1, 'Dr. Jane Smith', 101, '555-555-1212', 'jane@example.com', 'Y');

INSERT INTO doctors (doctor_id, name, office_number, telephone_number, email_address, is_full_time) VALUES (2, 'Dr. John Doe', 102, '555-555-1213', 'john@example.com', 'Y');

INSERT INTO doctors (doctor_id, name, office_number, telephone_number, email_address, is_full_time) VALUES (3, 'Dr. Bob Williams', 103, '555-555-1214', 'bob@example.com', 'Y');

INSERT INTO doctors (doctor_id, name, office_number, telephone_number, email_address, is_full_time) VALUES (4, 'Dr. Sally Johnson', 104, '555-555-1215', 'sally@example.com', 'N');

INSERT INTO nurses_new (nurse_id, name, office_number, telephone_number, email_address) VALUES (1, 'Nurse A', 101, '111-111-1111', 'nurseA@email.com');

INSERT INTO nurses_new (nurse_id, name, office_number, telephone_number, email_address) VALUES (2, 'Nurse B', 102, '222-222-2222', 'nurseB@email.com');

INSERT INTO nurses_new (nurse_id, name, office_number, telephone_number, email_address) VALUES (3, 'Nurse C', 103, '333-333-3333', 'nurseC@email.com');

INSERT INTO nurses_new (nurse_id, name, office_number, telephone_number, email_address) VALUES (4, 'Nurse D', 104, '444-444-4444', 'nurseD@email.com');

INSERT INTO nurses_new (nurse_id, name, office_number, telephone_number, email_address) VALUES (5, 'Nurse E', 105, '555-555-5555', 'nurseE@email.com');

INSERT INTO nurses_new (nurse_id, name, office_number, telephone_number, email_address) VALUES (6, 'Nurse F', 106, '666-666-6666', 'nurseF@email.com');

INSERT INTO nurses_new (nurse_id, name, office_number, telephone_number, email_address) VALUES (7, 'Nurse G', 107, '777-777-7777', 'nurseG@email.com');

INSERT INTO nurses_new (nurse_id, name, office_number, telephone_number, email_address) VALUES (8, 'Nurse H', 108, '888-888-8888', 'nurseH@email.com');

INSERT INTO appointments (appointment_id, pet_id, doctor_id, appt_date, appt_time, cost) VALUES (1, 1001, 1, '05-Aug-2022', TO_DATE('13:00:00', 'HH24:MI:SS'), 15);

INSERT INTO appointments (appointment_id, pet_id, doctor_id, appt_date, appt_time, cost) VALUES (2, 1002, 2, '05-Aug-2022', TO_DATE('14:00:00', 'HH24:MI:SS'), 5);

INSERT INTO appointments (appointment_id, pet_id, doctor_id, appt_date, appt_time, cost) VALUES (3, 1003, 3, '09-Aug-2022', TO_DATE('15:00:00', 'HH24:MI:SS'), 20);

INSERT INTO appointments (appointment_id, pet_id, doctor_id, appt_date, appt_time, cost) VALUES (4, 1004, 4, '16-Aug-2022', TO_DATE('16:00:00', 'HH24:MI:SS'), 5);

INSERT INTO appointments (appointment_id, pet_id, doctor_id, appt_date, appt_time, cost) VALUES (5, 1005, 1, '07-Oct-2022', TO_DATE('17:00:00', 'HH24:MI:SS'), 15);

INSERT INTO appointments (appointment_id, pet_id, doctor_id, appt_date, appt_time, cost) VALUES (6, 1006, 2, '11-Oct-2022', TO_DATE('18:00:00', 'HH24:MI:SS'), 15);

INSERT INTO appointments (appointment_id, pet_id, doctor_id, appt_date, appt_time, cost) VALUES (7, 1007, 3, '30-Nov-2022', TO_DATE('19:00:00', 'HH24:MI:SS'), 5);

INSERT INTO appointments (appointment_id, pet_id, doctor_id, appt_date, appt_time, cost) VALUES (8, 1008, 4, '27-Nov-2022', TO_DATE('20:00:00', 'HH24:MI:SS'), 5);

INSERT INTO appointments (appointment_id, pet_id, doctor_id, appt_date, appt_time, cost) VALUES (9, 1002, 1, '24-Nov-2022', TO_DATE('21:00:00', 'HH24:MI:SS'), 15);

INSERT INTO appointments (appointment_id, pet_id, doctor_id, appt_date, appt_time, cost) VALUES (10, 1002, 1, '29-Aug-2022', TO_DATE('22:00:00', 'HH24:MI:SS'), 20);

INSERT INTO appointments (appointment_id, pet_id, doctor_id, appt_date, appt_time, cost) VALUES (11, 1009, 4, '27-Nov-2022', TO_DATE('23:00:00', 'HH24:MI:SS'), 20);

INSERT INTO appointment_nursed (id, appointment_id, nurse_id) VALUES (1, 1, 1);

INSERT INTO appointment_nursed (id, appointment_id, nurse_id) VALUES (2, 1, 2);

INSERT INTO appointment_nursed (id, appointment_id, nurse_id) VALUES (3, 2, 3);

INSERT INTO appointment_nursed (id, appointment_id, nurse_id) VALUES (4, 2, 4);

INSERT INTO appointment_nursed (id, appointment_id, nurse_id) VALUES (5, 3, 5);

INSERT INTO appointment_nursed (id, appointment_id, nurse_id) VALUES (6, 3, 6);

INSERT INTO appointment_nursed (id, appointment_id, nurse_id) VALUES (7, 4, 7);

INSERT INTO appointment_nursed (id, appointment_id, nurse_id) VALUES (8, 4, 8);

INSERT INTO appointment_nursed (id, appointment_id, nurse_id) VALUES (9, 5, 1);

INSERT INTO appointment_nursed (id, appointment_id, nurse_id) VALUES (10, 5, 2);

INSERT INTO appointment_nursed (id, appointment_id, nurse_id) VALUES (11, 6, 3);

INSERT INTO appointment_nursed (id, appointment_id, nurse_id) VALUES (12, 6, 4);

INSERT INTO appointment_nursed (id, appointment_id, nurse_id) VALUES (13, 7, 1);

INSERT INTO appointment_nursed (id, appointment_id, nurse_id) VALUES (14, 7, 8);

INSERT INTO appointment_nursed (id, appointment_id, nurse_id) VALUES (15, 8, 2);

INSERT INTO appointment_nursed (id, appointment_id, nurse_id) VALUES (16, 8, 7);

INSERT INTO appointment_nursed (id, appointment_id, nurse_id) VALUES (17, 9, 3);

INSERT INTO appointment_nursed (id, appointment_id, nurse_id) VALUES (18, 9, 6);

INSERT INTO appointment_nursed (id, appointment_id, nurse_id) VALUES (19, 10, 4);

INSERT INTO appointment_nursed (id, appointment_id, nurse_id) VALUES (20, 10, 5);

INSERT INTO appointment_nursed (id, appointment_id, nurse_id) VALUES (21, 11, 1);

INSERT INTO appointment_nursed (id, appointment_id, nurse_id) VALUES (22, 11, 8);

INSERT INTO diagnoses (diagnosis_id, appointment_id, diag_date, description, referral, deferral) VALUES (1, 1, '05-Aug-2022', 'Flu', 'N', 'N');

INSERT INTO diagnoses (diagnosis_id, appointment_id, diag_date, description, referral, deferral) VALUES (2, 2, '05-Aug-2022', 'Allergy', 'Y', 'N');

INSERT INTO diagnoses (diagnosis_id, appointment_id, diag_date, description, referral, deferral) VALUES (3, 3, '09-Aug-2022', 'Skin infection', 'N', 'Y');

INSERT INTO diagnoses (diagnosis_id, appointment_id, diag_date, description, referral, deferral) VALUES (4, 4, '16-Aug-2022', 'socialisation', 'N', 'N');

INSERT INTO diagnoses (diagnosis_id, appointment_id, diag_date, description, referral, deferral) VALUES (5, 5, '07-Oct-2022', 'Bladder infection', 'Y', 'N');

INSERT INTO diagnoses (diagnosis_id, appointment_id, diag_date, description, referral, deferral) VALUES (6, 6, '11-Oct-2022', 'Liver disease', 'N', 'Y');

INSERT INTO diagnoses (diagnosis_id, appointment_id, diag_date, description, referral, deferral) VALUES (7, 7, '30-Nov-2022', 'socialisation', 'N', 'N');

INSERT INTO diagnoses (diagnosis_id, appointment_id, diag_date, description, referral, deferral) VALUES (8, 8, '27-Nov-2022', 'dental', 'N', 'Y');

INSERT INTO diagnoses (diagnosis_id, appointment_id, diag_date, description, referral, deferral) VALUES (9, 9, '24-Nov-2022', 'socialisation', 'N', 'N');

INSERT INTO diagnoses (diagnosis_id, appointment_id, diag_date, description, referral, deferral) VALUES (10, 10, '29-Aug-2022', 'socialisation', 'N', 'N');

INSERT INTO diagnoses (diagnosis_id, appointment_id, diag_date, description, referral, deferral) VALUES (11, 11, '27-Nov-2022', 'Allergy', 'Y', 'N');

INSERT INTO medication (medication_id, appointment_id, name, dosage, frequency) VALUES (1, 1, 'Ibuprofen', '200mg', '3 times a day');

INSERT INTO medication (medication_id, appointment_id, name, dosage, frequency) VALUES (2, 2, 'Antihistamine', '10mg', '2 times a day');

INSERT INTO medication (medication_id, appointment_id, name, dosage, frequency) VALUES (3, 3, 'Antibiotic ointment', 'apply topically', '3 times a day');

INSERT INTO medication (medication_id, appointment_id, name, dosage, frequency) VALUES (4, 4, 'Vitalmin Active', '10 tablets', '1 time a day');

INSERT INTO medication (medication_id, appointment_id, name, dosage, frequency) VALUES (5, 5, 'Antibiotic', '500mg', '3 times a day');

INSERT INTO medication (medication_id, appointment_id, name, dosage, frequency) VALUES (6, 6, 'Liver support supplement', '2 tablets', '2 times a day');

INSERT INTO medication (medication_id, appointment_id, name, dosage, frequency) VALUES (7, 7, 'Vitalmin Active', '10 tablets', '1 time a day');

INSERT INTO medication (medication_id, appointment_id, name, dosage, frequency) VALUES (8, 8, 'Liver support supplement', '2 tablets', '2 times a day');

INSERT INTO medication (medication_id, appointment_id, name, dosage, frequency) VALUES (9, 9, 'Vitalmin Active', '10 tablets', '1 times a day');

INSERT INTO medication (medication_id, appointment_id, name, dosage, frequency) VALUES (10, 10, 'Vitalmin Active', '10 tablets', '1 times a day');

INSERT INTO medication (medication_id, appointment_id, name, dosage, frequency) VALUES (11, 11, 'Antihistamine', '10mg', '2 times a day');