a. Derive relations from the conceptual model.

Strong Entities

Clinic (clinicNo, clinicName, clinicAddress, clinicPhone) **Primary Key** clinicNo	**Staff** (staffNo, staffName, staffAddress, staffPhone, dateOfBirth, position, salary, clinicNo) **Primary Key** staffNo **Foreign Key** clinicNo references Clinic(clinicNo)
Owner (ownerNo, ownerName, ownerAddress, ownerPhone) **Primary Key** ownerNo	**Pet** (petNo, petName, petDateOfBirth, species, breed, color, ownerNo, clinicNo) **Primary Key** petNo **Foreign Key** ownerNo references Owner(ownerNo) **Foreign Key** clinicNo references Clinic(clinicNo)
Examination (examNo, chiefComplaint, description dateSeen, actionsTaken, petNo, staffNo) **Primary Key:** examNo **Foreign Key** petNo references Pet(petNo) **Foreign Key** staffNo references Staff(staffNo)	

Binary Relationships

Entity A	**Entity B**	**Multiplicity (A)**	**Relationship (A)**	**Verb/Relationship Description**
Clinic	Staff	1:*	employs	A clinic employs multiple staff members.
Staff	Clinic	*:1	works at	A staff member works at one clinic.
Owner	Pet	1:*	owns	An owner owns multiple pets.
Pet	Owner	*:1	belongs to	A pet belongs to one owner.
Clinic	Pet	1:*	registers	A clinic registers multiple pets.
Pet	Clinic	*:1	is registered at	A pet is registered at one clinic.
Pet	Examination	1:*	undergoes	A pet undergoes multiple examinations.
Examination	Pet	*:1	is conducted for	An examination is conducted for one pet.
Staff	Examination	1:*	conducts	A staff member conducts multiple examinations.
Examination	Staff	*:1	is conducted by	An examination is conducted by one staff member.

1. Clinic <-> Staff

- Multiplicity:
    - A clinic employs multiple staff members (1:).
    - A staff member works at one clinic (**:1*).

- Relationship Type: *1: (One-to-Many)
- Participation Constraints:
  - Mandatory on the staff side: Every staff member must belong to a clinic.
  - Optional on the clinic side: Not all clinics may have staff at a given time.
- Implementation:
  - Foreign key clinicNo is added to the Staff table to reference Clinic.

2. Owner <-> Pet

- Multiplicity:
  - An owner can own multiple pets (1:).
  - A pet belongs to one owner (**:1*).
- Relationship Type: *1: (One-to-Many)
- Participation Constraints:
  - Mandatory on the pet side: Every pet must have an owner.
  - Optional on the owner side: An owner may not own any pets initially.
- Implementation:
  - Foreign key ownerNo is added to the Pet table to reference Owner.

3. Clinic <-> Pet

- Multiplicity:
  - A clinic registers multiple pets (1:).
  - A pet is registered at one clinic (**:1*).
- Relationship Type: *1: (One-to-Many)
- Participation Constraints:
  - Mandatory on the pet side: Every pet must be registered at a clinic.
  - Optional on the clinic side: A clinic may not have registered pets initially.
- Implementation:
  - Foreign key clinicNo is added to the Pet table to reference Clinic.

4. Pet <-> Examination

- Multiplicity:
  - A pet undergoes multiple examinations (1:).
  - An examination is conducted for one pet (**:1*).
- Relationship Type: *1: (One-to-Many)
- Participation Constraints:
  - Mandatory on the examination side: Every examination must be tied to a pet.
  - Optional on the pet side: A pet may not have any examinations recorded yet.
- Implementation:
  - Foreign key petNo is added to the Examination table to reference Pet.

5. Staff <-> Examination

- Multiplicity:
  - A staff member conducts multiple examinations (1:).
  - An examination is conducted by one staff member (**:1*).
- Relationship Type: *1: (One-to-Many)
- Participation Constraints:
  - Mandatory on the examination side: Every examination must be tied to a staff member.
  - Optional on the staff side: A staff member may not have conducted any examinations yet.
- Implementation:
  - Foreign key staffNo is added to the Examination table to reference Staff.

b. Validate the logical model using normalization to 3NF.

Clinic (clinicNo, clinicName, clinicAddress, clinicPhone)
Primary Key: clinicNo

1NF: All attributes are atomic (no multi-valued phone numbers)

2NF: No partial dependencies since there is a single primary key (clinicNo)

3NF: No transitive dependencies (clinicName does not depend on clinicAddress)

Staff (staffNo, staffName, staffAddress, staffPhone, dateOfBirth, position, salary, clinicNo)
Primary Key: staffNo
Foreign Key: clinicNo references Clinic(clinicNo)

1NF: All attributes are atomic (all values in one cell)

2NF: No partial dependencies since there is a single primary key (staffNo)

3NF: No transitive dependencies (clinicNo does not determine any other attributes in Staff)

Owner (ownerNo, ownerName, ownerAddress, ownerPhone)
Primary Key: ownerNo

1NF: All attributes are atomic (all values in one cell)

2NF: No partial dependencies since there is a single primary key (ownerNo)

3NF: No transitive dependencies (ownerNo does not determine any other attributes in Owner)

Pet (petNo, petName, petDateOfBirth, species, breed, color, ownerNo, clinicNo)
Primary Key: petNo
Foreign Key: ownerNo references Owner(ownerNo)

1NF: All attributes are atomic (all values in one cell)

2NF: No partial dependencies since because the primary key is petNo, and all attributes depend on it

3NF: No transitive dependencies, as ownerNo and clinicNo are foreign keys which do not determine any other attributes

Examination (examNo, chiefComplaint, description, dateSeen, actionsTaken, petNo, staffNo)

```
Primary Key: examNo
Foreign Keys: petNo references Pet(petNo), staffNo references Staff(staffNo)
```

1NF: All attributes are atomic (all values in one cell)

2NF: No partial dependencies since the primary key is examNo, and all attributes depend on it

3NF: No transitive dependencies, as petNo and staffNo do not determine any other attributes

c. Validate the logical model against 5 user transactions. (Note: These will be then implemented in 3c).

1. Add a new pet to the database

- Scenario: A pet owner registers a new pet at a clinic.
- Steps:
    1. Insert a new record into the Owner table if the owner is not already present.
    2. Insert a new record into the Pet table with the owner's ID (ownerNo) and clinic ID (clinicNo).

```
-- Step 1: Insert the owner (if not already present)
INSERT INTO Owner (ownerNo, ownerName, ownerAddress, ownerPhone)
VALUES (“O113423”, “John Smith”, “102 Old Dorwart St, Lancaster, PA 17602”, “717-321-5555”);

-- Step 2: Insert the pet
INSERT INTO Pet (petNo, petName, petDateOfBirth, species, breed, color, ownerNo, clinicNo)
VALUES ("P148213", "Sunny", "2021-06-15", "Dog", "Retriever", "Golden", "O113423, “C013123”);
```

Validation:

- Owner: ownerNo is unique and references the primary key of Owner.
- Pet: petNo is unique and references ownerNo in Owner and clinicNo in Clinic

2. Record an examination for a pet

- Scenario: A staff member records an examination for a pet.
- Steps:

1. Ensure the pet exists in the Pet table and the staff member exists in the Staff table.
2. Insert a new record into the Examination table.

```
-- Step 1: Insert the examination
INSERT INTO Examination (examNo, chiefComplaint, description, dateSeen, actionsTaken, petNo,
staffNo)
VALUES (“E7893121”, “Limping”, “Examined left leg; no fractures detected”, “2024-01-15”, “Prescribed
anti-inflammatory medication”, “P148213”, “S101310”);
```

Validation:

- Examination: examNo is unique.
- Foreign Keys:
  - petNo references Pet(petNo).
  - staffNo references Staff(staffNo).

3. Update a staff member's assigned clinic

- Scenario: A staff member is reassigned to another clinic.
- Steps:
  1. Update the clinicNo in the Staff table for the specified staffNo.

```
-- Update the staff member's assigned clinic
UPDATE Staff
SET clinicNo = "'C013123'"
WHERE staffNo = "S101310";
```

Validation:

- Staff: Ensure staffNo exists in the Staff table.
- Foreign Key: Ensure the new clinicNo references Clinic(clinicNo).

4. Retrieve all pets registered at a specific clinic

- Scenario: The system needs to generate a list of all pets registered at a particular clinic.
- Steps:
  1. Query the Pet table for pets with the specified clinicNo.

```
-- Retrieve all pets registered at clinic C001
SELECT petNo, petName, petDateOfBirth, species, breed, color, ownerNo
FROM Pet
WHERE clinicNo = “C013123”;
```

Validation:

- Pet: clinicNo is a valid foreign key in the Pet table.
- Output: The query returns the expected pets registered at the specified clinic.

5. Generate a report of examinations conducted by a staff member

- Scenario: The system needs to generate a report of all examinations conducted by a specific staff member.
- Steps:
    1. Query the Examination table for examinations with the specified staffNo.
    2. Join with the Pet table to include pet details in the report.

```
-- Generate a report of examinations conducted by staff member S101
SELECT e.examNo, e.chiefComplaint, e.description, e.dateSeen, e.actionsTaken, p.petName, p.species,
p.breed, p.ownerNo
FROM Examination e
JOIN Pet p ON e.petNo = p.petNo
WHERE e.staffNo = “S101310”;
```

Validation:

- Examination: Ensure staffNo is valid and exists in the Examination table.
- Joins:
    - petNo in Examination matches petNo in Pet.
- Output: Verify the report includes all relevant examination details.

d. Define integrity constraints:

i. Primary key constraints.

Each table must have a unique identifier that serves as the primary key:

Clinic:

```
CONSTRAINT pk_clinic PRIMARY KEY (clinicNo)
```

Staff:

```
CONSTRAINT pk_staff PRIMARY KEY (staffNo)
```

Owner:

```
CONSTRAINT pk_owner PRIMARY KEY (ownerNo)
```

Pet:

```
CONSTRAINT pk_pet PRIMARY KEY (petNo)
```

Examination:

```
CONSTRAINT pk_examination PRIMARY KEY (examNo)
```

ii. Referential integrity/Foreign key constraints.

Relationships between entities are maintained through foreign key constraints:

Staff -> Clinic:

```
CONSTRAINT fk_staff_clinic FOREIGN KEY (clinicNo) REFERENCES Clinic (clinicNo)
```

Pet -> Owner:

```
CONSTRAINT fk_pet_owner FOREIGN KEY (ownerNo) REFERENCES Owner (ownerNo)
```

Pet -> Clinic:

```
CONSTRAINT fk_pet_clinic FOREIGN KEY (clinicNo) REFERENCES Clinic (clinicNo)
```

Examination -> Pet:

```
CONSTRAINT fk_examination_pet FOREIGN KEY (petNo) REFERENCES Pet (petNo)
```

Examination -> Staff:

```
CONSTRAINT fk_examination_staff FOREIGN KEY (staffNo) REFERENCES Staff (staffNo)
```

iii. Alternate key constraints (if any).

Define additional unique keys as alternate keys for critical attributes:

Owner Phone Number:

```
CONSTRAINT ak_owner_phone UNIQUE (ownerPhone)
```

Clinic Phone Number:

```
CONSTRAINT ak_clinic_phone UNIQUE (clinicPhone)
```

Staff Phone Number

```
CONSTRAINT ak_staff_phone UNIQUE (staffPhone)
```

iv. Required data.

Clinic :

```
clinicNo NOT NULL, clinicName NOT NULL, clinicAddress NOT NULL, clinicPhone NOT NULL
```

Staff:

```
staffNo NOT NULL, staffName NOT NULL, staffAddress NOT NULL, staffPhone NOT NULL, clinicNo NOT
NULL
```

Owner:

```
ownerNo NOT NULL, ownerName NOT NULL, ownerAddress NOT NULL, ownerPhone NOT NULL
```

Pet:

```
petNo NOT NULL, petName NOT NULL, petDateOfBirth NOT NULL, species NOT NULL, ownerNo NOT
NULL, clinicNo NOT NULL
```

Examination:

```
examNo NOT NULL, chiefComplaint NOT NULL, dateSeen NOT NULL, petNo NOT NULL, staffNo NOT
NULL
```

v. Attribute domain constraints

Date:

- Format: YYYY-MM-DD

Example:

```
CHECK (dateSeen ~ '^\d{4}-\d{2}-\d{2}$')
```

IDs (Owner, Clinic, Pet):

- Format: A letter (O, C, P) followed by 6 digits (assuming the client is okay with just 6 digits)

Example:

```
CHECK (clinicNo ~ '^C\d{6}$'),
CHECK (ownerNo ~ '^O\d{6}$'),
CHECK (petNo ~ '^P\d{6}$')
```

Address:

- Format: {Street Address}, {City}, {State Abbrev} {Zip Code}

Example:

```
CHECK (clinicAddress ~ '^[^,]+, [^,]+, [A-Z]{2} \d{5}$')
```

Phone Number:

- Format: 10 digits separated by hyphens (XXX-XXX-XXXX).

Example:

```
CHECK (clinicPhone ~ '^\d{3}-\d{3}-\d{4}$'),
CHECK (staffPhone ~ '^\d{3}-\d{3}-\d{4}$'),
CHECK (ownerPhone ~ '^\d{3}-\d{3}-\d{4}$')
```

Names (Owner, Pet, Staff):

- Must contain non-numeric characters only.

Example:

```
CHECK (staffName ~ '^[A-Za-z ]+$'),
CHECK (ownerName ~ '^[A-Za-z ]+$'),
CHECK (petName ~ '^[A-Za-z ]+$')
```

Open-Ended Fields (Examination):

- chiefComplaint and description:
  - Maximum 500 characters.

```
CHECK (LENGTH(chiefComplaint) <= 500),

CHECK (LENGTH(description) <= 500)
```

vi. General constraints (if any).

Salary Validation:

- Staff salary must be greater than $0.

```
CHECK (salary > 0)
```

Species:

- Restrict species to predefined values (e.g., Dog, Cat, etc.).

```
CHECK (species IN ('Dog', 'Cat', 'Bird', 'Rabbit', 'Other'))
```

Position Validation:

- Restrict species to predefined values (e.g., Dog, Cat, etc.).

```
CHECK (species IN ('Dog', 'Cat', 'Bird', 'Rabbit', 'Other'))
```