Task 1. Changes

The “Dam_Ranking” relation that we initially had was removed, instead we will be doing the ranking through SQL queries (like SELECT returning the score for the given query), and that information will not be saved in the database.

Task 2. Normalize to BCNF

For table Animal, the non-prime attributes {sex, dam, status} are all functionally dependent on animal_id. No other FDs exist, and thus every attribute is functionally dependent on animal_id, the key of Animal.

For table Dam, the only non-derived attribute is animal_id, which is the key of the relation. There are no FDs in Dam.

For Dam_Ranking, there is only one FD, in which the attribute position is functionally dependent on animal-id.

For Session_Dam, every attribute {milk_rating, num_of_kids, mothering, mother_score, kids_weaned, observations, birth_weight} are functionally dependent on the key {animal_id, session_id}.

- Session_Dam
    - Is already in BCNF, as there is a single primary key (session_id), no partial or transitive dependencies.

- Dam
    - Is already in BCNF, as there is a single primary key (animal_id), and all attributes are derived without having any transitive dependencies.

Task 3. Views

For our project, we will not require any views to be able to retrieve the data needed to rank the dams.

Note: Dam_Ranked might need view

Task 4. Queries

```
CREATE TABLE animal (
        animal_id SERIAL PRIMARY KEY,
        sex CHAR(1),
        status VARCHAR(20),
        dam SERIAL REFERENCES animal(animal_id)
);

CREATE TABLE session_dam (
        session_id SERIAL PRIMARY KEY,
        animal_id SERIAL REFERENCES animal(animal_id),
        mother_score INTEGER,
        milk_rating INTEGER,
        num_of_kids INTEGER,
        mothering INTEGER,
        observations TEXT,
        birth_weight REAL,
        kids_weaned INTEGER
);

INSERT INTO animal (animal_id, sex, status, dam) VALUES
        (1, 'M', 'Sold', NULL),
        (2, 'M', 'Sold', NULL),
        (3, 'F', 'Active', NULL),
        (4, 'M', 'Active', 3);

INSERT INTO session_dam (animal_id, mother_score, milk_rating, num_of_kids, mothering, birth_weight,
kids_weaned) VALUES
        (3, 3, 2, 2, 4, 4.3, 2),
        (3, 4, 3, 2, 3, 3.9, 2),
        (2, 10, 10, 10, 10, 10, 10);

SELECT animal_id, AVG(mother_score), AVG(milk_rating), AVG(num_of_kids), AVG(mothering),
AVG(birth_weight), AVG(kids_weaned)
        FROM session_dam
        GROUP BY animal_id;
```

Task 5. Discussion

The queries and views relate to the use cases by allowing the data entrant to submit data and the data analyst to view the data.