University of British Columbia, Vancouver
Department of Computer Science

`CPSC 304 Project Cover Page

Milestone #: __2_____

Date: ___October 20, 2023_____

Group Number: ____65_____

Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
Sharon Marfatia	79155529	c8c4d	ssmubc@students.cs.ubc.ca
Chris Jung	14231609	b4i0j	eunhocj@students.cs.ubc.ca
Ayan Qadir	90759622	z7e0k	aqadir01@students.cs.ubc.ca

By typing our names and student numbers in the above table, we certify that the work in the attached assignment was performed solely by those whose names and student IDs are included above. (In the case of Project Milestone 0, the main purpose of this page is for you to let us know your e-mail address, and then let us assign you to a TA for your project supervisor.)

In addition, we indicate that we are fully aware of the rules and consequences of plagiarism, as set forth by the Department of Computer Science and the University of British Columbia

CPSC 304 Project Milestone 2: Aquarium Database

Project Description

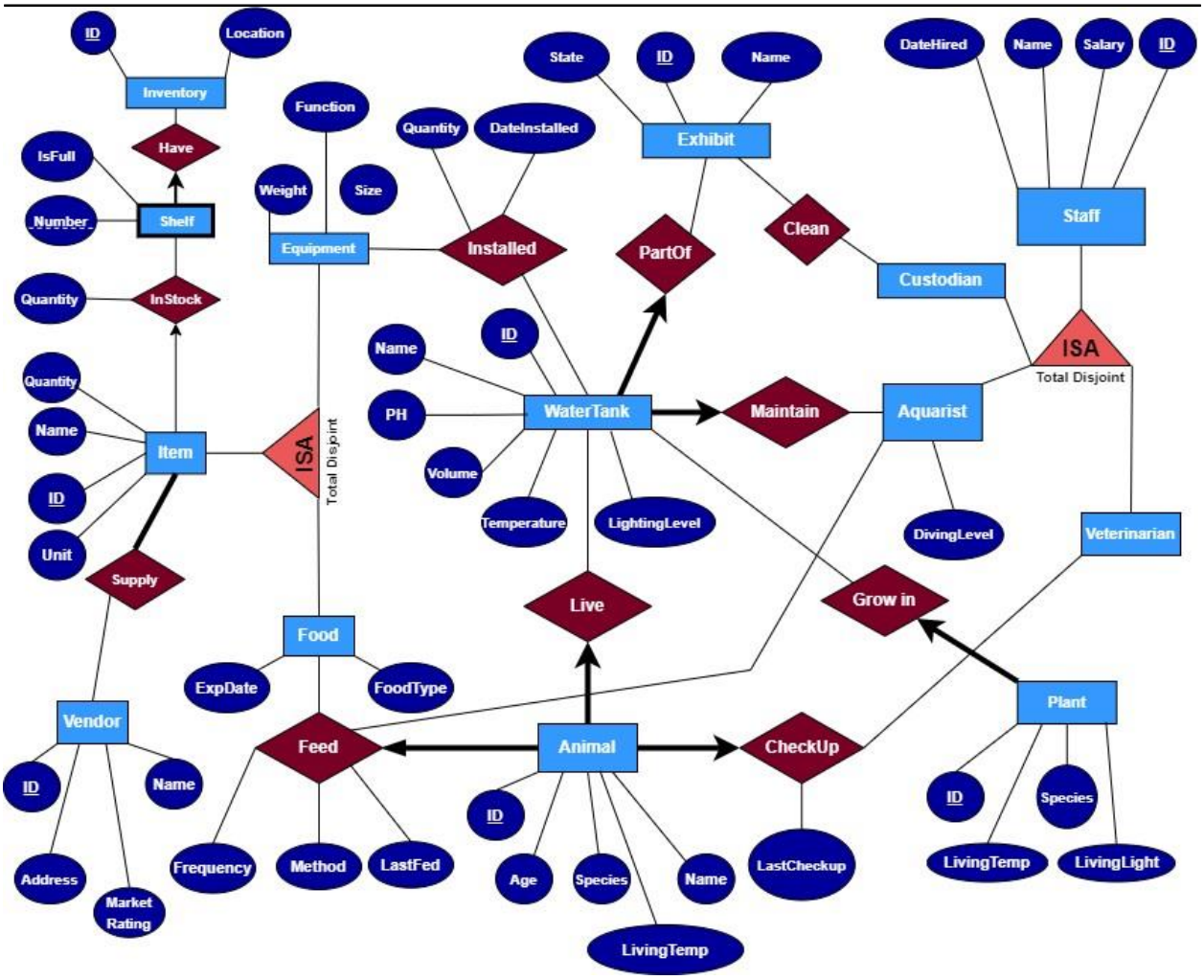
A brief (~2-3 sentences) summary of your project. Many of your TAs are managing multiple projects so this will help them remember details about your project.

Our database is designed to comprehensively model the many aspects of an aquarium exhibition, capturing both its internal and external components. Our database models core entities of an aquarium which are water tanks, devices within the tanks/maintenance equipment, marine animals, aquatic plants, the supply chain of curated animal diet, and the individuals responsible for operations and maintenance. Our project aims to address and represent the intricacies of the aquarium domain, and as an example of a real-life situation, the database can be applied to efficiently manage large-scale.

ER Diagram

The ER diagram you are basing your item #3 (below) on. This ER diagram may be the same as your milestone 1 submission or it might be different. If you have made changes from the version submitted in milestone 1, attach a note indicating what changes have been made and why.

- The addition of pH to the WaterTank entity is to add a non-trivial FD to perform normalization for the entity table
- The attribute 'Status' was changed to 'State' since Status is a keyword in SQL
- The addition of vendor_market_rating is to add a non-trivial FD to perform normalization for the entity Vendor
- 'IsFull' attribute was added to Shelf enity for better management of the shelf



Schemas/Constraints/FDs (Question 4 and 5)

The schema derived from your ER diagram (above). For the translation of the ER diagram to the relational model, follow the same instructions as in your lectures. Also, identify the functional dependencies in your relations, including the ones involving all candidate keys (including the primary key).

- **Underline for Primary Key**
- **Bold for Foreign Key**

- Staff(<u>id</u>: integer, salary: float, name: varchar[255], datehired: date)
  - PK: id
  - Constraints: salary, name, datehired should all be not null
  - Functional Dependencies (FDs):
    id → salary
    id → name

id → datahired

- Custodian(<u>id</u>: integer, <u>exhibit_id</u>: integer) (Subtype of Staff)
  - PK: (id, exhibit_id)
  - FK: (id, exhibit_id)

- Veterinarian(<u>id</u>: integer) (Subtype of Staff)
  - PK: id
  - FK: id

- Aquarist(<u>id</u>: integer, diving_level: float, **<u>water_tank_id</u>**: integer) (Subtype of Staff)
  - PK: id,
  - FK: (id, water_tank_id)
  - Constraints: diving_level (NOT NULL)
  - Functional Dependencies (FDs):
    id → diving_level

- Custodian_Clean_Exhibit_Table(<u>exhibit_id</u>: integer, <u>custodian_id</u>: integer)
  - PK: (exhibit_id, custodian_id)
  - FK: (exhibit_id, custodian_id)

- Grown_In_Plant(<u>plant_id</u>: integer, species: varchar[255], living_temp: float, living_light: float, **<u>water_tank_id</u>**: int)
  - PK: (plant_id, water_tank_id)
  - FK: water_tank_id
  - Constraints: species, living_temp, and living_light (NOT NULL)
  - Functional Dependencies (FDs):
    plant_id → species
    plant_id → living_temp
    plant_id → living_light
    plant_id → water_tank_id

- Vendor(<u>id</u>: integer, name: varchar[255], address: varchar[255], vendor_market_rating: enum)
  - PK: id
  - Constraints: name, address, and vendor_market_rating (NOT NULL)
  - Functional Dependencies (FDs):
    id → name
    id → address
    id → vendor_market_rating
    name → vendor_market_rating

- Supply(**ItemID**: integer, **VendorID**: integer)
  - PK: (ItemID, VendorID)
  - FK: (ItemID, VendorID)

- Inventory(id: integer, location: varchar[255])
  - Primary Key (PK): inventory_id
  - Constraints: location (NOT NULL)
  - Functional Dependencies (FDs):
    id → location

- ShelfInInventory(shelf_number: integer, **inventory_id**: integer, is_full: ENUM('true', false'))
  - PK: (shelf_number, inventory_id)
  - FK: inventory_id
  - Constraints: location (NOT NULL)
  - FDs:
    (shelf_number, inventory_id) → is_full

- Item(id: integer, name: varchar[255], quantity: integer, unit: varchar[255])
  - PK: id
  - Constraints: name, quantity, unit (NOT NULL)
  - FDs:
    id → quantity
    id → unit
    id → name

- Equipment(**item_id**: integer, function: varchar[255], weight: decimal(10,2), size: varchar[255]) (Subtype of Item)
  - PK: item_id
  - FK: item_id
  - Constraints: function, weight, size, date_installed (NOT NULL)
  - FDs:
    item_id → function
    item_id → weight
    item_id → size

- Food(**item_id**: integer, exp_date: date, food_type: varchar[255]) (Subtype of Item)
  - PK: item_id
  - FK: item_id
  - Constraints: exp_date, food_type (NOT NULL)
  - FDs:
    item_id → item_id
    item_id→ exp_date

item_id → food_type

- InStock(**item_id**: integer, **shelf_number**: integer, **inventory_id**: integer, quantity: integer)
  - PK: (item_id, shelf_number, inventory_id)
  - FK: (shelf_number, inventory_id), item_id
  - FDs:
    (item_id, shelf_number, inventory_id) → quantity

- Installed(**equipment_id**: integer, **water_tank_id**: integer, quantity: integer, date_installed: date)
  - PK: (equipment_id, water_tank_id)
  - FK: equipment _id, water_tank_id
  - Constraints: date_installed, quantity (NOT NULL)
  - FDs:
    (equipment_id, water_tank_id) → (quantity, date_installed)

- Feed(**food_id**: integer, **animal_id**: integer, **aquarist_id**: integer, quantity: integer, last_fed: integer, method: varchar[255])
  - PK: (animal_id, food_id, aquarist_id)
  - FK: food_id, animal_id, aquarist_id
  - Constraints: method, quantity (NOT NULL)
  - FDs:
    (food_id, animal_id) → (aquarist_id, quantity, last_fed, method)

- Exhibit(id: integer, name: char, state: enum)
  - PK: id
  - Constraints: name, state (NOT NULL)
  - Functional Dependencies (FDs):
    id → name
    id → state

- WaterTank(id: integer, name: varchar[255], volume: decimal, temperature: decimal, lightinglevel: enum, pH: decimal, **exhibit_id**: integer)
  - PK: id
  - FK: exhibit_id
  - Constraints: name, volume, temperature, lightinglevel, pH (NOT NULL)
  - Functional Dependencies (FDs):
    id → name
    id → volume
    id → temperature
    id → lightinglevel
    id → pH

temperature → pH

- Animal(id: integer, name: varchar[255], species: varchar[255], age: integer, livingtemp: varchar[255], water_tank_id: integer, veterinarian_id: integer)
  - PK: (id)
  - FK: (water_tank_id, veterinarian_id)
  - Constraints: name, species, integer, livingtemp (NOT NULL)
  - Functional Dependencies (FDs):
    id → name
    id → species
    id → age
    id → livingtemp

- AquaristMaintainWaterTank (aquarist_id: integer, water_tank_id: integer)
  - PK: (aquarist_id, water_tank_id)
  - FK: (aquarist_id, water_tank_id)

Normalization

Normalize each of your tables to be in 3NF or BCNF. Give the list of tables, their primary keys, their candidate keys, and their foreign keys after normalization

Out of total 20 entity/relation tables (Appendix.1) before normalization, we figured that 17 of them are already normalized. Thus, we normalized the remaining 3 tables (Vendor, Item, and WaterTank) as below. Each of the three normalization processes is slightly different as each team member normalized each table for learning purposes.

a) Normalization - Vendor

- Vendor(id: int, name: char, address: char, vendor_market_rating: enum)
  - PK: id
  - Constraints: name, address, and vendor_market_rating should all be NOT NULL
  - Functional Dependencies (FDs):
    id → name
    id → address
    id → vendor_market_rating
    name → vendor_market_rating

Minimum Cover
(removed id → vendor_market_rating because it's redundant):

id → name

id → address
name → vendor_market_rating

Since id is a superkey, the relationships with id on the LHS do not violate BCNF so we do not need to decompose on those relationships. However, the relationship, **name → vendor_market_rating** needs to be decomposed.

As a result we can divide our attributes as we did in class (below):

Left side	Middle	Right side
id, address	name	vendor_market_rating

VendorLogistics(<u>id</u>, address, **name**)
VendorReputation(<u>name</u>, vendor_market_rating)

After normalization:

Final Tables in BCNF:

VendorLogistics(<u>id</u>, address, **name**)
PK: id
FK: name REFERENCES VendorReputation(name)

VendorReputation(<u>name</u>, vendor_market_rating)
PK: name

b) Normalization - WaterTank

<u>WaterTank:</u>

- WaterTank(<u>id</u>: integer, name: char, volume: decimal, temperature: decimal, lightinglevel: enum, pH: decimal, **<u>exhibit_id</u>:** integer)
  - PK: id
  - FK: exhibit_id
  - Constraints: name, volume, temperature, lightinglevel, pH (NOT NULL)
  - Functional Dependencies (FDs):
    id → name
    id → volume
    id → temperature
    id → lightinglevel
    id → pH
    temperature → pH

Minimum Cover
(removed id → pH because it's redundant):

id → name
id → volume
id → temperature
id → lightinglevel
temperature → pH

Since id is a superkey, the relationships with id on the LHS do not violate BCNF so we do not need to decompose on those relationships. However, the relationship, **temperature → pH** needs to be decomposed.

As a result we can divide our attributes as we did in class (below):

Left side	Middle	Right side
id, name, volume, lightinglevel	temperature	pH

- WaterTankLogistics(id: integer, name: char, volume: decimal, temperature: decimal, lightinglevel: enum, **exhibit_id**: integer)

- WaterTankpH(temperature: decimal, pH: decimal)

As all FDs are preserved now, normalization is finished.

Final Tables in BCNF

- WaterTankLogistics(**id**: integer, name: char, volume: decimal, temperature: decimal, lightinglevel: enum, exhibit_id: integer)
  - PK: id
  - FK: exhibit_id
  - FDs:
    id → name
    id → volume
    id → temperature
    id → lightinglevel

- WaterTankpH(temperature: decimal, pH: decimal)
  - PK: temperature
  - FDs:
    temperature → pH

c) Normalization - Item

- Item(**id**: integer, name: varchar[255], quantity: integer, unit: varchar[255])
  - PK: id
  - Constraints: name, quantity, unit (NOT NULL)

- FDs:
  id → quantity
  id → unit
  id → name
  name → unit

Minimum Cover
(removed id → unit because it's redundant):

id → quantity
id → name
name → unit

name → unit violates 3NF as name is not a super key nor unit is a part of the key. Thus, decompose this relation into following using lossless-join:

ItemQuantity(id, name, quantity),
ItemUnit(name, unit)

As all FDs are preserved now, normalization is finished.

Final Tables in BCNF:

- ItemQuantity(**id**: integer, name: varchar[255], quantity: integer])
  - PK: id
  - FK: name
  - Constraints: quantity, name(NOT NULL)
  - FDs:
    id → quantity
    id → name

- ItemUnit(**name**: varchar[255], unit: varchar[255])
  - PK: name
  - Constraints: unit (NOT NULL)
  - FDs:
    name → unit

Post-Normalization

After normalizing the three table, we now have total 23 normalized table (Appendix.2)

SQL Statements

Creation

CREATE TABLE Exhibit (

```
  id INTEGER PRIMARY KEY,
  Name VARCHAR(255) NOT NULL,
  Status ENUM('OPEN', 'CLOSED', 'UNDER MAINTENANCE', 'COMING SOON') NOT NULL
 );


CREATE TABLE Staff (
  id INTEGER PRIMARY KEY,
  salary DECIMAL(10, 2) NOT NULL,
  name VARCHAR(255) NOT NULL,
  datehired DATE NOT NULL
);

CREATE TABLE Custodian (
  id INTEGER PRIMARY KEY,
  exhibit_id INTEGER,
  FOREIGN KEY (exhibit_id) REFERENCES Exhibit(id) ON DELETE CASCADE ON UPDATE CASCADE,
  FOREIGN KEY (id) REFERENCES
  Staff(id) ON DELETE CASCADE);


  CREATE TABLE Veterinarian (
  id INTEGER PRIMARY KEY,
  FOREIGN KEY (id) REFERENCES
  Staff(id) ON DELETE CASCADE
);

CREATE TABLE Aquarist (
  id INTEGER PRIMARY KEY,
  FOREIGN KEY (id) REFERENCES
  Staff(id) ON DELETE CASCADE,
  diving_level DECIMAL (10, 2) NOT NULL,
  water_tank_id INTEGER,
  FOREIGN KEY (water_tank_id) REFERENCES WaterTank(id) ON DELETE CASCADE ON UPDATE
CASCADE
);

CREATE TABLE WaterTankLogistics (
  id INTEGER PRIMARY KEY,
  Name VARCHAR(255) NOT NULL,
  Volume DECIMAL (10, 2) NOT NULL,
  Temperature DECIMAL (3, 1) NOT NULL,
  LightingLevel ENUM('LOW', 'MEDIUM', 'HIGH') NOT NULL,
```

```
  exhibit_id INTEGER,
  FOREIGN KEY (exhibit_id) REFERENCES Exhibit(id) ON DELETE CASCADE ON  UPDATE CASCADE
);

CREATE TABLE WaterTankpH (
Temperature DECIMAL (3, 1) PRIMARY KEY,
pH DECIMAL (2,1) NOT NULL
);


CREATE TABLE Animal (
  id INTEGER PRIMARY KEY,
  Name VARCHAR(255) NOT NULL,
  Species VARCHAR(255) NOT NULL,
  Age INTEGER NOT NULL,
  LivingTemp DECIMAL (3, 1) NOT NULL,
  water_tank_id INTEGER,
  veterinarian_id INTEGER,
  FOREIGN KEY (water_tank_id) REFERENCES WaterTankLogistics(id) ON DELETE CASCADE ON
UPDATE CASCADE,
  FOREIGN KEY (veterinarian_id) REFERENCES Veterinarian(id) ON DELETE CASCADE ON UPDATE
CASCADE
);

CREATE TABLE Inventory (
    id INTEGER PRIMARY KEY,
    location VARCHAR(255) NOT NULL
);


CREATE TABLE ShelfInInventory (
    shelf_number INTEGER,
    inventory_id INTEGER,
    is_full ENUM('true', 'false') NOT NULL,
    PRIMARY KEY (shelf_number, inventory_id),
    FOREIGN KEY (inventory_id) REFERENCES Inventory(id)
        ON DELETE CASCADE ON UPDATE CASCADE
);


CREATE TABLE ItemQuantity (
    id INTEGER PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
```

```
  quantity INTEGER NOT NULL
);


CREATE TABLE ItemUnit (
  name VARCHAR(255) PRIMARY KEY,
  unit VARCHAR(255) NOT NULL
);


CREATE TABLE Equipment (
  item_id INTEGER PRIMARY KEY,
  function VARCHAR(255) NOT NULL,
  weight DECIMAL(10,2) NOT NULL,
  size VARCHAR(255) NOT NULL,
  date_installed DATE NOT NULL,
  FOREIGN KEY (item_id) REFERENCES ItemQuantity(id)
    ON DELETE CASCADE ON UPDATE CASCADE
);


CREATE TABLE Food (
  item_id INTEGER PRIMARY KEY,
  exp_date DATE NOT NULL,
  food_type VARCHAR(255) NOT NULL,
  FOREIGN KEY (item_id) REFERENCES ItemQuantity(id)
    ON DELETE CASCADE ON UPDATE CASCADE
);


CREATE TABLE InStock (
  item_id INTEGER,
  shelf_number INTEGER,
  inventory_id INTEGER,
  quantity INTEGER NOT NULL,
  PRIMARY KEY (item_id, shelf_number, inventory_id),
  FOREIGN KEY (shelf_number, inventory_id) REFERENCES ShelfInInventory(shelf_number,
inventory_id)
    ON DELETE CASCADE ON UPDATE CASCADE,
  FOREIGN KEY (item_id) REFERENCES ItemQuantity(id)
    ON DELETE CASCADE ON UPDATE CASCADE
);
```

```
CREATE TABLE Installed (
   equipment_id INTEGER,
   water_tank_id INTEGER,
   quantity INTEGER NOT NULL,
   date_installed DATE NOT NULL,
   PRIMARY KEY (equipment_id, water_tank_id),
   FOREIGN KEY (equipment_id) REFERENCES Equipment(item_id)
      ON DELETE CASCADE ON UPDATE CASCADE,
   FOREIGN KEY (water_tank_id) REFERENCES ItemQuantity(id)
      ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE Feed (
   food_id INTEGER,
   animal_id INTEGER,
   aquarist_id INTEGER,
   quantity INTEGER NOT NULL,
   last_fed DATE NOT NULL,
   method VARCHAR(255) NOT NULL,
   PRIMARY KEY (food_id, animal_id, aquarist_id),
   FOREIGN KEY (food_id) REFERENCES Food(item_id)
      ON DELETE CASCADE ON UPDATE CASCADE
);


CREATE TABLE Custodian_Clean_Exhibit_Table (
  exhibit_id INTEGER,
  custodian_id INTEGER,
  PRIMARY KEY (exhibit_id, custodian_id),
  FOREIGN KEY (exhibit_id) REFERENCES Exhibit(id) ON DELETE CASCADE ON UPDATE CASCADE,
  FOREIGN KEY (custodian_id) REFERENCES Custodian(id) ON DELETE CASCADE ON UPDATE
CASCADE
);


CREATE TABLE VendorLogistics (
  id INTEGER PRIMARY KEY,
  FOREIGN KEY (name) REFERENCES VendorReputation(name) ON DELETE CASCADE ON UPDATE
CASCADE,
  address VARCHAR(255) NOT NULL
);

CREATE TABLE VendorReputation (
```

```
  name VARCHAR(255) NOT NULL,
  vendor_market_rating VARCHAR(255) NOT NULL
);


CREATE TABLE Grown_In_Plant (
  plant_id INTEGER,
  species VARCHAR (255) NOT NULL,
  living_temp DECIMAL(10, 2) NOT NULL,
  living_light DECIMAL(10, 2) NOT NULL,
  water_tank_id INTEGER NOT NULL,
  PRIMARY KEY (plant_id),
  FOREIGN KEY (water_tank_id) REFERENCES WaterTank
  ON DELETE NO ACTION
  ON UPDATE CASCADE
);

CREATE TABLE Aquarist_Maintain_WaterTank (
  aquarist_id INTEGER,
  water_tank_id INTEGER,
  FOREIGN KEY (aquarist_id) REFERENCES Aquarist(id) ON DELETE CASCADE ON UPDATE
CASCADE,
  FOREIGN KEY (water_tank_id REFERENCES WaterTank(ID) ON DELETE CASCADE ON UPDATE
CASCADE
);

CREATE TABLE Supply (
    ItemID INTEGER,
    VendorID INTEGER,
    PRIMARY KEY (ItemID, VendorID),
    FOREIGN KEY (ItemID) REFERENCES Item(ID) ON DELETE CASCADE ON UPDATE CASCADE,
    FOREIGN KEY (VendorID) REFERENCES Vendor(id) ON DELETE CASCADE ON UPDATE CASCADE
);
```

Insertion

```
INSERT INTO Grown_In_Plant (plant_id, species, living_temp, living_light, water_tank_id)
VALUES
  (101, 'Water Lily', 22.5, 100.0, 1),
  (102, 'Seaweed', 18.0, 80.0, 2),
  (103, 'Mangrove', 25.0, 70.0, 3),
  (104, 'Coral', 26.0, 90.0, 4),
  (105, 'Anubias', 24.0, 60.0, 5);
```

```
INSERT INTO VendorLogistics (id, name, address)
VALUES
  (5, 'AquaLife Supplies', '123 Ocean Drive, Marine City, 45678'),
  (6, 'WaterWorld Equipment', '789 Coral Blvd, Reef Town, 12345'),
  (7, 'Marine Essentials', ‘456 Habitat St, Salt Lake, 91011'),
  (8, 'Underwater Gadgets', '135 Reef Rd, Blue Sea, 11345'),
  (9, 'Oceanic Items', '8643 Sea Road, Blue Harbour, 14151');


INSERT INTO VendorReputation (name, vendor_market_rating)
VALUES
  ('AquaLife Supplies', 'High'),
  ('WaterWorld Equipment', 'Medium'),
  ('Marine Essentials', 'Low'),
  ('Underwater Gadgets', 'Medium'),
  ('Oceanic Items', 'High');


INSERT INTO Exhibit(id, Name, State, LastMaintenanceDate, Size, NoOfWaterTanks)
VALUES
(17, 'Octopus Exhibit', 'OPEN'),
(18, 'Shark Exhibit', 'OPEN'),
(19, 'Dolphin Exhibit', 'OPEN'),
(20, 'Turtle Exhibit', 'OPEN'),
(21, 'Squid Exhibit', 'OPEN')
;


INSERT INTO Staff (id, salary, name, datehired)
VALUES
  (100, 1000.50, 'Sam', '2023-10-15'),
  (101, 1000.50, 'Anna', '2023-10-15'),
  (102, 145000.80, 'Kevin', '2023-10-16'),
  (103, 145000.80, 'John', '2023-10-16'),
  (104, 145000.80, 'Mohammed', '2023-10-17'),
  (105, 145000.80, 'James', '2023-10-17'),
  (106, 145000.80, 'Wataru', '2023-10-18'),
  (107, 100000.80, 'Michael', '2022-10-05'),
```

```
 (108, 1000.50, 'Kim', '2022-10-30'),
 (109, 1000.50, 'Danny', '2022-09-15'),
 (110, 100000.80, 'Rachel', '2022-09-16'),
 (111, 100000.80, 'Baam', '2021-08-16'),
 (112, 100000.80, 'Megumi', '2021-07-16'),
 (113, 100000.80, 'Oshimhen', '2021-04-16'),
 (114, 1000.50, 'Messi', '2021-04-16')
;

INSERT INTO WaterTankLogistics(id, Name, Volume, Temperature, LightingLevel, exhibit_id)
VALUES
(1, 'Shark Tank', 1000.45, 27.5, 'Medium', 18),
(2, 'Octopus Tank', 1500.45, 27.8, 'Medium', 17),
(3, 'Dolphin Tank', 1000.45, 28.0, 'Medium', 19),
(4, 'Squid Tank', 1000.45, 27.9, 'Medium', 21),
(5, 'Turtle Tank', 500.50, 27.8,'Low', 20)
;

INSERT INTO WaterTankpH(Temperature, pH)
VALUES
(27.5, 7.0),
(27.8, 6.8),
(28, 6.7),
(27.9, 6.7),
(27.8, 6.8);


INSERT INTO Custodian (id, exhibit_id)
VALUES
 (100, 17),
 (101, 18),
 (108, 19),
 (109, 20),
 (114, 21)
;

INSERT INTO Aquarist (id, diving_level, water_tank_id)
VALUES
 (107, 100.00, 1),
 (110, 100.00, 2),
 (111, 100.00, 3),
 (112, 100.00, 4),
 (113, 100.00, 5)
```

```
;

INSERT INTO Veterinarian (id)
VALUES
  (102),
  (103),
  (104),
  (105),
  (106)
;


INSERT INTO Animal(id, Name, Species, Age, LivingTemp, water_tank_id, veterinarian_id)
VALUES
(31, 'Great White Shark', 'Carcharodon carcharias', 6, 27.5, 1, 102),
(32, 'Common Octopus', 'Octopus vulgaris', 7, 27.5, 2, 102),
(33, 'Orca', 'Orcinus orca', 5, 27.5, 3, 103),
(34, 'Vampire Squid', 'Vampyroteuthis infernalis', 3, 27.5, 4, 104),
(35, 'Aldabra giant tortoise', 'Aldabrachelys gigantea', 1, 27.5, 5, 105)
;

INSERT INTO ItemQuantity (id, name, quantity) VALUES
(1, 'Algae Wafers', 50),
(2, 'Coral Supplement', 30),
(3, 'Water Conditioner', 20),
(4, 'Medicated Feed', 65),
(5, 'Plankton', 80),
(6, 'Aquarium Salt', 120),
(7, 'Water Test Kits', 45);

INSERT INTO ItemUnit (name, unit) VALUES
('Algae Wafers', 'Packets'),
('Coral Supplement', 'Bottles'),
('Water Conditioner', 'Bottles'),
('Medicated Feed', 'Packets'),
('Plankton', 'Packets'),
('Aquarium Salt', 'Boxes'),
('Water Test Kits', 'Kits');

INSERT INTO Equipment (item_id, function, weight, size, date_installed) VALUES
(1, 'Water Filtration', 50.00, 'LARGE', '2020-01-15'),
(2, 'Protein Skimmer', 8.00, 'MEDIUM', '2021-07-30'),
(3, 'Heater', 2.00, 'SMALL', '2022-02-11'),
```

```
(4, 'LED Lighting', 5.00, 'MEDIUM', '2019-08-24'),
(5, 'UV Sterilizer', 6.00, 'SMALL', '2021-04-05'),
(6, 'Oxygen Pump', 4.50, 'SMALL', '2023-03-29'),
(7, 'CO2 System', 10.00, 'MEDIUM', '2018-12-10');

INSERT INTO Food (item_id, exp_date, food_type) VALUES
(8, '2024-01-15', 'Frozen Shrimp'),
(9, '2024-03-22', 'Fish Flakes'),
(10, '2024-02-11', 'Bloodworms'),
(11, '2024-05-19', 'Pellets'),
(12, '2024-04-13', 'Krill'),
(13, '2024-07-07', 'Squid'),
(14, '2024-06-25', 'Mysis Shrimp');

INSERT INTO Inventory (id, location) VALUES
(1, 'Main Storage'),
(2, 'Food Prep Area'),
(3, 'Maintenance Storage'),
(4, 'Medical Bay'),
(5, 'Temporary Storage');


INSERT INTO Custodian_Clean_Exhibit_Table (exhibit_id, custodian_id)
VALUES
  (17, 100),
  (18, 101),
  (19, 108),
  (20, 109),
  (21, 114);


INSERT INTO Supply (ItemID, VendorID)
VALUES
  (124, 5),
  (156, 6),
  (167, 7),
  (111, 8),
  (210, 9);


INSERT INTO ShelfInInventory (shelf_number, inventory_id, is_full) VALUES
(1, 1, 'true'),
(2, 1, 'false'),
```

(3, 2, 'true'),
(4, 2, 'false'),
(5, 3, 'true'),

INSERT INTO InStock (item_id, shelf_number, inventory_id, quantity) VALUES
(1, 1, 1, 10),
(2, 2, 1, 15),
(3, 3, 2, 20),
(4, 4, 2, 25),
(5, 5, 3, 30),
(6, 6, 3, 12),
(7, 7, 4, 18);

Appendix.1

List of relations before normalization

1. Staff
2. Custodian
3. Veterinarian
4. Aquarist
5. Custodian_Clean_Exhibit_Table
6. Grown_In_Plant
7. Vendor
8. Supply
9. Inventory
10. ShelfInInventory
11. Item
12. Equipment
13. Food
14. InStock
15. Installed
16. Feed
17. Exhibit
18. WaterTank
19. Animal
20. AquaristMaintainWaterTank

Appendix.2

List of relations after normalization (Decomposed tables after normalization are underlined)

1. Staff
2. Custodian

3. Veterinarian
4. Aquarist
5. Custodian_Clean_Exhibit_Table
6. Grown_In_Plant
7. VendorLogistics
8. VendorReputation
9. Supply
10. Inventory
11. ShelfInInventory
12. ItemQuantity
13. ItemUnit
14. Equipment
15. Food
16. InStock
17. Installed
18. Feed
19. Exhibit
20. WaterTankSimple
21. WaterTankPh
22. Animal
23. AquaristMaintainWaterTank