CSE250 - Database Management System

Winter 23'

Project Title :

<u>Wildlife Sanctuary Management System</u>

Group Details

Parth Joshi	AU2140002	parth.j2@ahduni.edu.in
Milee Bajpai	AU2140158	milee.b@ahduni.edu.in
Krutarth Vora	AU2140162	krutarth.v@ahduni.edu.in
Shubham Apat	AU2140220	shubham.a1@ahduni.edu.in

Project Description :

The Wildlife Sanctuary Management System is a database-driven application that provides a comprehensive solution for managing a wildlife sanctuary's operations. The system includes modules for managing sanctuaries, animals, employees, doctors, habitat areas, trees, birds, passengers, safari ride bookings, medical supplies, and food supplies. The database management system (DBMS) provides a centralized platform for storing, retrieving, and managing data, ensuring data accuracy, consistency, and security. The Wildlife Sanctuary Management System offers an efficient and effective way to manage a wildlife sanctuary's diverse activities, streamlining operations and enhancing productivity.

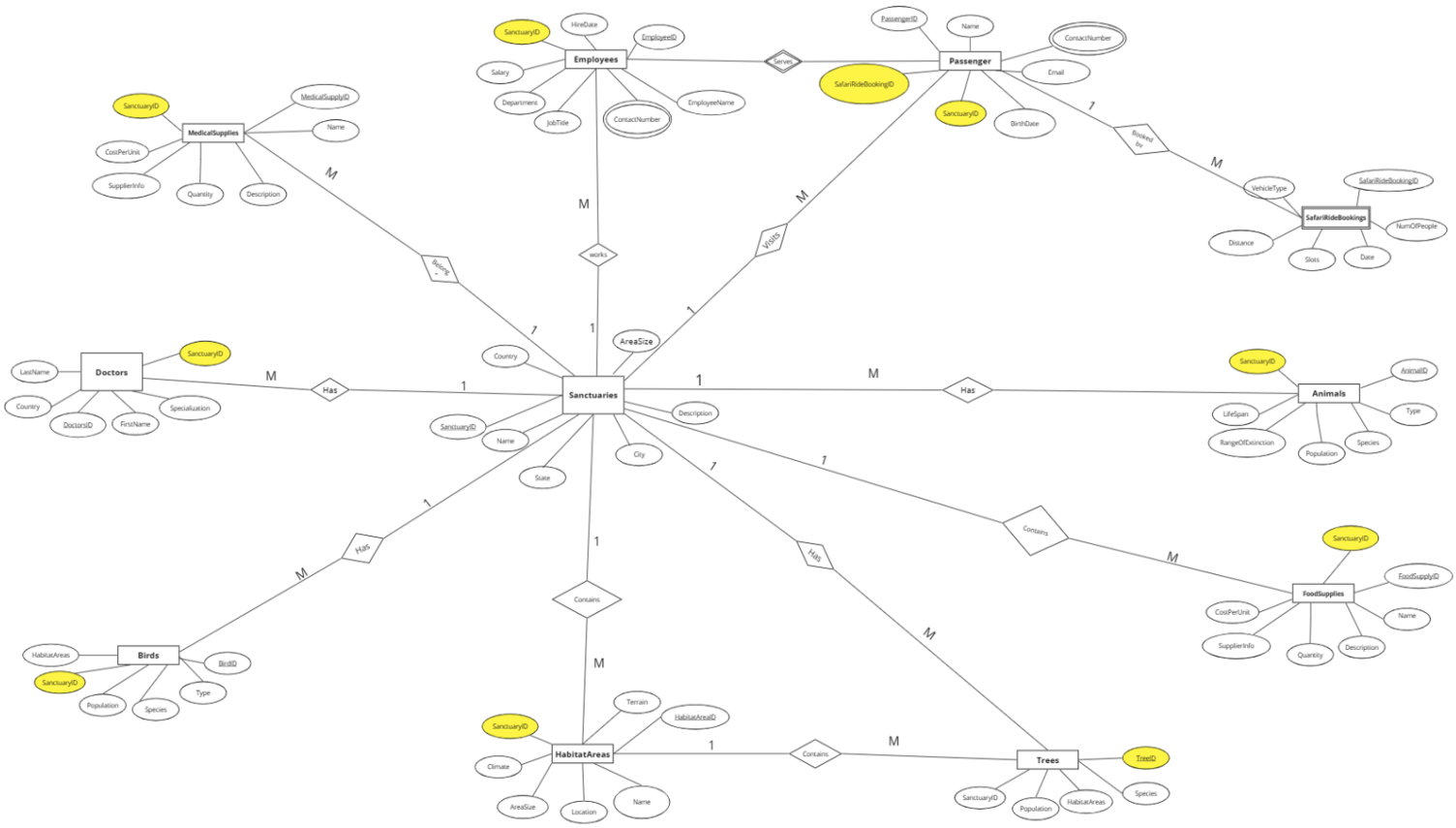
Tools Used :

Database	MySQL
Framework	Node-Js
Other Platforms	VS Code, MySQL Workbench, Miro, Lucid Chart

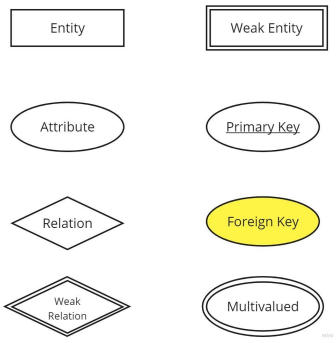
Requirements of the System :

- The database system should contain information about different species of animals and plants found in the sanctuary, including their scientific and common names, taxonomy, physical characteristics, behavior, and habitat preferences.
- Each animal and plant species should be associated with a unique ID that can be used to track their location and movements within the sanctuary.
- The database should also keep track of the location and status of each animal within the sanctuary, including their health, age, sex, and reproductive status.
- The system should maintain information about the different habitats within the sanctuary, including their size, location, topography, climate, and vegetation.
- The database should be able to generate reports on animal populations, habitat health, and other key indicators to help wildlife managers make informed decisions about conservation and management practices.
- The system should also keep track of visitor information, including their demographics, visit history, and feedback, to help improve visitor experiences and better manage tourism impacts on the sanctuary.
- The system should provide tools for researchers and scientists to access and analyze data from the database for scientific research and conservation purposes.

E-R Diagram :



LEGENDS



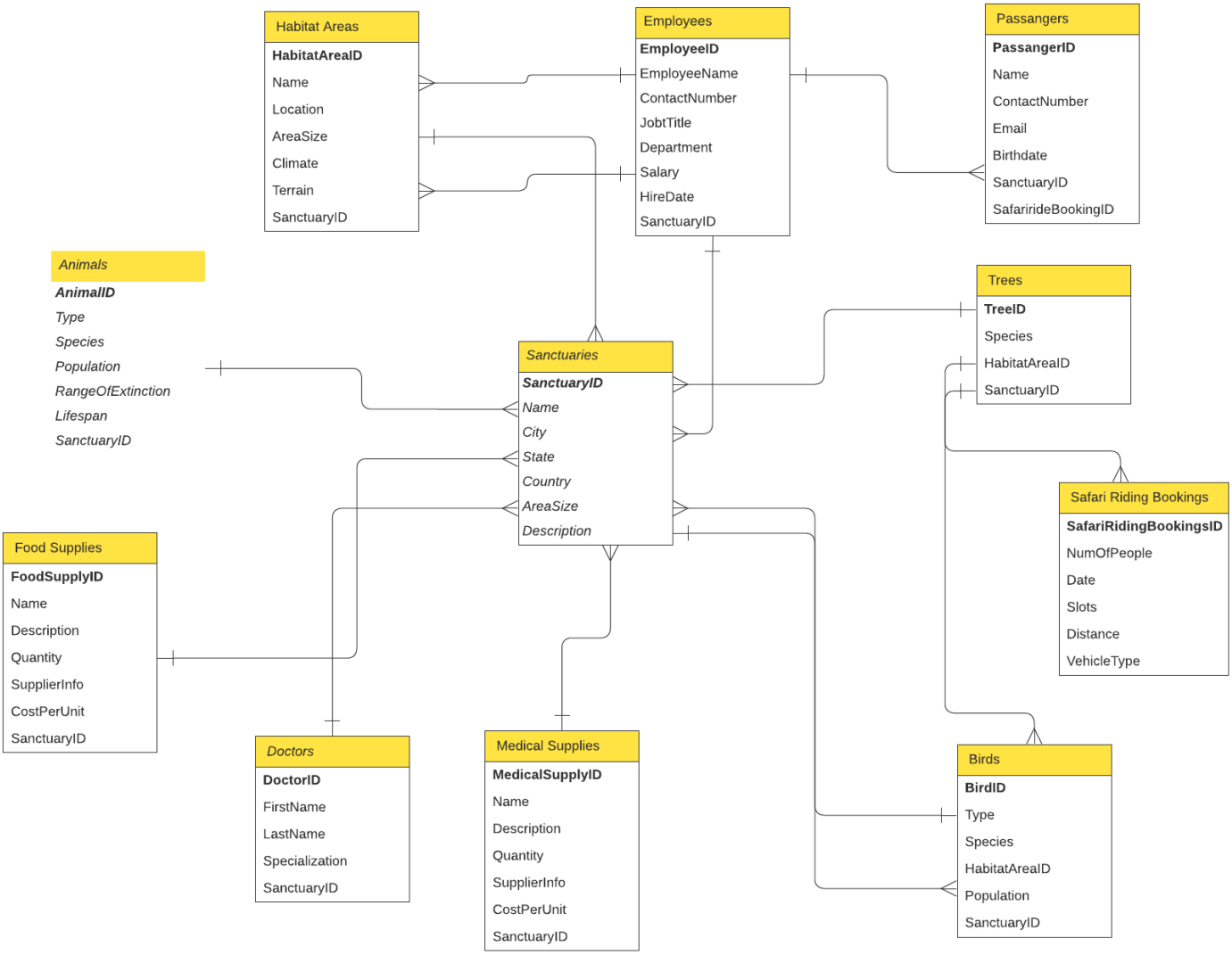
[Click here](#) for a better view.

Cardinality of Relationships :

Entity - 1	Relationship Name	Entity - 2	Cardinality	Description
Sanctuaries	Has	Animals	1:M	A sanctuary can have many animals, but an animal can only belong to one sanctuary.
Sanctuaries	Has	Doctors	1:M	A sanctuary can have many doctors, but a doctor can only work at one sanctuary.
Sanctuaries	Has	Employees	1:M	A sanctuary can have many employees, but an employee can only work at one sanctuary.
Sanctuaries	Has	HabitatAreas	1:M	A sanctuary can have many habitat areas, but a habitat area can only belong to one sanctuary.
Sanctuaries	Has	Trees	1:M	A sanctuary can have many trees, but a tree can only belong to one sanctuary.
Sanctuaries	Has	MedicalSupplies	1:M	A sanctuary can have many medical supplies, but a medical supply can only belong to one sanctuary.
Sanctuaries	Has	FoodSupplies	1:M	A sanctuary can have many food supplies, but a food supply can only belong to one sanctuary.

Animals	Belongs	Sanctuaries	M:1	An animal belongs to a single sanctuary, but a sanctuary can have many animals.
Doctors	Works	Sanctuaries	M:1	A doctor works at a single sanctuary, but a sanctuary can have many doctors.
Employees	Works	Sanctuaries	M:1	An employee works at a single sanctuary, but a sanctuary can have many employees.
HabitatAreas	Belongs	Sanctuaries	M:1	A habitat area belongs to a single sanctuary, but a sanctuary can have many habitat areas.
Trees	Belongs	HabitatAreas	M:1	A tree belongs to a single habitat area, but a habitat area can have many trees.
Passenger	Makes	SafariRideBookings	M:1	A passenger can make many safari ride bookings, but a safari ride booking can only be made by one passenger.
Passenger	in	Sanctuaries	M:1	A passenger can visit only one sanctuary, but a sanctuary can have many visitors.
HabitatAreas	Has	Trees	1:M	A habitat area can have many trees, but a tree can only belong to one habitat area.
SafariRideBookings	Has	Passengers	1:M	A safari ride booking can have many passengers, but a passenger can only make one safari ride booking.

Database Design :



[Click here](#) for a better view.

ENTITIES :

(1) Sanctuaries :

	Field	Type	Null	Key	Default	Extra
▶	SanctuaryID	int	NO	PRI	NULL	auto_increment
	Name	varchar(100)	YES		NULL	
	City	varchar(100)	YES		NULL	
	State	varchar(100)	YES		NULL	
	Country	varchar(100)	YES		NULL	
	AreaSize	int	YES		NULL	
	Description	varchar(100)	YES		NULL	

Contains information about the different wildlife sanctuaries, including their name, location (city, state, and country), area size, and a description. The primary key is SanctuaryID.

(2) Animals :

	Field	Type	Null	Key	Default	Extra
▶	AnimalID	int	NO	PRI	NULL	auto_increment
	Type	varchar(100)	YES		NULL	
	Species	varchar(100)	YES		NULL	
	Population	int	YES		NULL	
	RangeOfExtinction	varchar(100)	YES		NULL	
	Lifespan	int	YES		NULL	
	SanctuaryID	int	YES	MUL	NULL	

Contains information about the animals present in each sanctuary, including their type, species, population, range of extinction, and lifespan. The table also has a foreign key reference to the Sanctuaries table. The primary key is AnimalID.

(3) Employees :

Field	Type	Null	Key	Default	Extra
EmployeeID	int	NO	PRI	NULL	auto_increment
EmployeeName	varchar(100)	YES		NULL	
ContactNumber	varchar(100)	YES		NULL	
JobTitle	varchar(100)	YES		NULL	
Department	varchar(100)	YES		NULL	
Salary	decimal(10,2)	YES		NULL	
HireDate	date	YES		NULL	
SanctuaryID	int	YES	MUL	NULL	

Contains information about the employees working in each sanctuary, including their name, contact number, job title, department, salary, and hire date. The table also has a foreign key reference to the Sanctuaries table. The primary key is EmployeeID.

(4) Doctors:

Field	Type	Null	Key	Default	Extra
DoctorID	int	NO	PRI	NULL	auto_increment
FirstName	varchar(100)	YES		NULL	
LastName	varchar(100)	YES		NULL	
Specialization	varchar(100)	YES		NULL	
SanctuaryID	int	YES	MUL	NULL	

Contains information about the doctors who work in the sanctuaries, including their name, specialization, and the sanctuary they work in. The table also has a foreign key reference to the Sanctuaries table. The primary key is DoctorID.

(5) Habitat Areas:

Field	Type	Null	Key	Default	Extra
HabitatAreaID	int	NO	PRI	NULL	auto_increment
Name	varchar(100)	YES		NULL	
Location	varchar(100)	YES		NULL	
AreaSize	int	YES		NULL	
Climate	varchar(100)	YES		NULL	
Terrain	varchar(100)	YES		NULL	
SanctuaryID	int	YES	MUL	NULL	

Contains information about the different habitat areas in each sanctuary, including their name, location, area size, climate, and terrain. The table also has a foreign key reference to the Sanctuaries table. The primary key is HabitatAreaID.

(6) Trees :

Field	Type	Null	Key	Default	Extra
TreeID	int	NO	PRI	NULL	auto_increment
Species	varchar(100)	YES		NULL	
HabitatAreaID	int	YES	MUL	NULL	
SanctuaryID	int	YES	MUL	NULL	

Contains information about the different trees in each habitat area, including their species. The table has foreign key references to both the HabitatAreas and Sanctuaries tables. The primary key is TreeID.

(7) Birds :

Field	Type	Null	Key	Default	Extra
BirdID	int	NO	PRI	NULL	auto_increment
Type	varchar(100)	YES		NULL	
Species	varchar(100)	YES		NULL	
HabitatAreaID	int	YES	MUL	NULL	
Population	int	YES		NULL	
SanctuaryID	int	YES	MUL	NULL	

Contains information about the different bird species in each habitat area, including their type, species, and population. The table has foreign key references to both the HabitatAreas and Sanctuaries tables. The primary key is BirdID.

(8) SafariRideBookings:

Field	Type	Null	Key	Default	Extra
SafariRideBookingID	int	NO	PRI	NULL	auto_increment
NumOfPeople	int	YES		NULL	
Date	date	YES		NULL	
Slots	varchar(100)	YES		NULL	
Distance	int	YES		NULL	
VehicleType	varchar(100)	YES		NULL	

Contains information about the bookings for safari rides, including the number of people, date, slots, distance, and vehicle type. The table also has a foreign key reference to the Sanctuaries table. The primary key is SafariRideBookingID.

(9) Passengers:

Field	Type	Null	Key	Default	Extra
PassengerID	int	NO	PRI	NULL	auto_increment
Name	varchar(100)	YES		NULL	
ContactNumber	varchar(100)	YES		NULL	
Email	varchar(100)	YES		NULL	
Birthdate	date	YES		NULL	
SanctuaryID	int	YES	MUL	NULL	
SafariRideBookingID	int	YES	MUL	NULL	

Contains information about the visitors to the sanctuaries, including their name, contact number, email, and birthdate. The table also has a foreign key reference to the Sanctuaries table. The primary key is PassengerID.

(10) MedicalSupplies:

Field	Type	Null	Key	Default	Extra
MedicalSupplyID	int	NO	PRI	NULL	auto_increment
Name	varchar(100)	YES		NULL	
Description	varchar(100)	YES		NULL	
Quantity	int	YES		NULL	
SupplierInfo	varchar(100)	YES		NULL	
CostPerUnit	int	YES		NULL	
SanctuaryID	int	YES	MUL	NULL	

Contains information about the medical supplies used in the sanctuaries, including their name, description, quantity, supplier information, and cost per unit. The table also has a foreign key reference to the Sanctuaries table. The primary key is MedicalSupplyID.

(11) FoodSupplies:

Field	Type	Null	Key	Default	Extra
FoodSupplyID	int	NO	PRI	NULL	auto_increment
Name	varchar(100)	YES		NULL	
Description	varchar(100)	YES		NULL	
Quantity	int	YES		NULL	
SupplierInfo	varchar(100)	YES		NULL	
CostPerUnit	int	YES		NULL	
SanctuaryID	int	YES	MUL	NULL	

Contains information about the food supplies used in the sanctuaries, including their name, description, quantity, supplier information, and cost per unit. The table also has a foreign key reference to the Sanctuaries table. The primary key is FoodSupplyID.

Stored Procedures :

(1) Stored Procedure to add/ insert an animal.

```
-- 1. Procedure to add an animal

DELIMITER $$

CREATE PROCEDURE AddNewAnimal (
  IN animalType VARCHAR(100),
  IN animalSpecies VARCHAR(100),
  IN animalPopulation INT,
  IN rangeOfExtinction VARCHAR(100),
  IN animalLifespan INT,
  IN sanctuaryId INT
)
BEGIN
  INSERT INTO Animals (Type, Species, Population, RangeOfExtinction, Lifespan, SanctuaryID)
     VALUES (animalType, animalSpecies, animalPopulation, rangeOfExtinction, animalLifespan,
sanctuaryId);
END $$

DELIMITER ;


CALL AddNewAnimal('Mammal', 'Snow Leopard', 7000, 'Endangered', 12, 1);

-- ------------------------------------------------------------------------------------
```

Before :

```
1    SELECT * FROM Animals;
2
3
4
5
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

AnimalID	Type	Species	Population	RangeOfExtinction	Lifespan	SanctuaryID
1	Mammal	Bengal Tiger	2500	Endangered	15	1
2	Mammal	Indian Rhinoceros	3500	Vulnerable	35	1
3	Bird	Indian Peafowl	10000	Least Concern	15	1
4	Mammal	Asiatic Lion	523	Endangered	16	2
5	Mammal	Indian Elephant	27000	Endangered	60	2
6	Bird	Indian Vulture	150	Critically Endangered	20	2
7	Reptile	King Cobra	100	Least Concern	20	3
8	Mammal	Sloth Bear	7000	Vulnerable	25	3
9	Mammal	Indian Wild Dog	2500	Endangered	12	3
10	Bird	Sarus Crane	1500	Vulnerable	30	4
11	Mammal	Blackbuck	5000	Least Concern	10	4
12	Reptile	Gharial	200	Critically Endangered	40	4
NULL	NULL	NULL	NULL	NULL	NULL	NULL

After :

```
1    SELECT * FROM Animals;
2
3
4
5
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

AnimalID	Type	Species	Population	RangeOfExtinction	Lifespan	SanctuaryID
1	Mammal	Bengal Tiger	2500	Endangered	15	1
2	Mammal	Indian Rhinoceros	3500	Vulnerable	35	1
3	Bird	Indian Peafowl	10000	Least Concern	15	1
4	Mammal	Asiatic Lion	523	Endangered	16	2
5	Mammal	Indian Elephant	27000	Endangered	60	2
6	Bird	Indian Vulture	150	Critically Endangered	20	2
7	Reptile	King Cobra	100	Least Concern	20	3
8	Mammal	Sloth Bear	7000	Vulnerable	25	3
9	Mammal	Indian Wild Dog	2500	Endangered	12	3
10	Bird	Sarus Crane	1500	Vulnerable	30	4
11	Mammal	Blackbuck	5000	Least Concern	10	4
12	Reptile	Gharial	200	Critically Endangered	40	4
13	Mammal	Snow Leopard	7000	Endangered	12	1
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Call Block : Procedure - 1 (Node-Js)

```javascript
async function AddNewAnimal(animalType, animalSpecies, animalPopulation, rangeOfExtinction, animalLifespan, sanctuaryId) {
  try {
    const connection = await mysql.createConnection({
        host: 'localhost',
        user: 'root',
        password: 'Parth@3060',
        database: 'wildlife_sanctuary_up'
    });

    await connection.execute('CALL AddNewAnimal(?, ?, ?, ?, ?, ?)',
      [animalType, animalSpecies, animalPopulation, rangeOfExtinction, animalLifespan, sanctuaryId]);

    console.log('New animal added successfully!');
    connection.end();

  } catch (error) {
    console.error(error);
  }
}

AddNewAnimal('Mammal', 'African Elephant', 700, 'Endangered', 23, 3);
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL COMMENTS

```
PS C:\Users\parth\OneDrive\Desktop\dbms_project> node "c:\Users\parth\OneDrive\Desktop\dbms_project\stored_procedures\1.js"
New animal added successfully!
PS C:\Users\parth\OneDrive\Desktop\dbms_project>
```

Code
Code
Code

(2) Stored Procedure to book safari tickets.

```
DELIMITER $$

CREATE PROCEDURE AddSafariRideBooking(IN p_NumOfPeople INT, IN p_Date DATE, IN p_Slots VARCHAR(100), IN p_Distance INT, IN p_VehicleType VARCHAR(100), IN p_Name VARCHAR(100), IN p_ContactNumber VARCHAR(100), IN p_Email VARCHAR(100), IN p_Birthdate DATE, IN p_SanctuaryID INT)
BEGIN
  DECLARE v_SafariRideBookingID INT;

  INSERT INTO SafariRideBookings (NumOfPeople, Date, Slots, Distance, VehicleType,SanctuaryID)
  VALUES (p_NumOfPeople, p_Date, p_Slots, p_Distance, p_VehicleType,p_SanctuaryID);

  SET v_SafariRideBookingID = LAST_INSERT_ID();

      INSERT INTO Passengers (Name, ContactNumber, Email, Birthdate, SanctuaryID, SafariRideBookingID)
          VALUES (p_Name, p_ContactNumber, p_Email, p_Birthdate, p_SanctuaryID, v_SafariRideBookingID);

    UPDATE SafariRideBookings SET NumOfPeople = NumOfPeople + p_NumOfPeople WHERE SafariRideBookingID = v_SafariRideBookingID;
END $$

DELIMITER ;

CALL AddSafariRideBooking(30, '2021-05-31', 'evening', 10, 'Jeep', 'Aman Singh', '897654321', 'Aman@example.com', '1983-05-01', 4);

-- ----------------------------------------------------------------------------------------------------------------
```

Before :

```
28 •  SELECT * FROM Passengers;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

PassengerID	Name	ContactNumber	Email	Birthdate	SanctuaryID	SafariRideBookingID
1	Priya Patel	9876543210	priya.patel@example.com	1990-01-01	1	301
2	Rahul Singh	8765432109	rahul.singh@example.com	1992-05-12	2	302
3	Kavita Sharma	7654321098	kavita.sharma@example.com	1985-09-23	3	303
4	Amit Kumar	6543210987	amit.kumar@example.com	1987-03-15	4	304
5	Anjali Gupta	5432109876	anjali.gupta@example.com	1998-11-30	5	305
6	Sachin Sharma	4321098765	sachin.sharma@example.com	1995-07-08	1	306
7	Neha Patel	3210987654	neha.patel@example.com	1993-02-20	2	307
8	Kunal Gupta	2109876543	kunal.gupta@example.com	1991-06-18	3	308
9	Manoj Singh	1098765432	manoj.singh@example.com	1989-12-10	4	309
10	Radhika Khanna	0987654321	radhika.khanna@example.com	1997-08-25	5	310
11	John Doe	1234567890	johndoe@example.com	2000-01-01	2	311
NULL	NULL	NULL	NULL	NULL	NULL	NULL

```
28 •  SELECT * FROM SafariRideBookings;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap

SafariRideBookingID	NumOfPeople	Date	Slots	Distance	VehicleType	SanctuaryID
301	2	2022-06-15	morning	10	jeep	1
302	4	2021-12-20	noon	20	bus	2
303	1	2023-01-05	evening	5	jeep	5
304	3	2022-09-10	morning	15	jeep	4
305	2	2023-03-18	noon	20	bus	3
306	5	2021-11-30	evening	10	jeep	5
307	2	2022-05-12	morning	5	jeep	2
308	3	2022-07-25	noon	15	bus	4
309	1	2022-07-25	noon	20	jeep	1
310	4	2022-07-25	morning	10	jeep	5
311	8	2023-05-01	evening	10	Jeep	1
NULL	NULL	NULL	NULL	NULL	NULL	NULL

After :

```
28 •  SELECT * FROM Passengers;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

PassengerID	Name	ContactNumber	Email	Birthdate	SanctuaryID	SafariRideBookingID
1	Priya Patel	9876543210	priya.patel@example.com	1990-01-01	1	301
2	Rahul Singh	8765432109	rahul.singh@example.com	1992-05-12	2	302
3	Kavita Sharma	7654321098	kavita.sharma@example.com	1985-09-23	3	303
4	Amit Kumar	6543210987	amit.kumar@example.com	1987-03-15	4	304
5	Anjali Gupta	5432109876	anjali.gupta@example.com	1998-11-30	5	305
6	Sachin Sharma	4321098765	sachin.sharma@example.com	1995-07-08	1	306
7	Neha Patel	3210987654	neha.patel@example.com	1993-02-20	2	307
8	Kunal Gupta	2109876543	kunal.gupta@example.com	1991-06-18	3	308
9	Manoj Singh	1098765432	manoj.singh@example.com	1989-12-10	4	309
10	Radhika Khanna	0987654321	radhika.khanna@example.com	1997-08-25	5	310
11	John Doe	1234567890	johndoe@example.com	2000-01-01	2	311
12	Aman Singh	897654321	Aman@example.com	1983-05-01	4	312
NULL	NULL	NULL	NULL	NULL	NULL	NULL

SafariRideBookingID	NumOfPeople	Date	Slots	Distance	VehicleType	SanctuaryID
301	2	2022-06-15	morning	10	jeep	1
302	4	2021-12-20	noon	20	bus	2
303	1	2023-01-05	evening	5	jeep	5
304	3	2022-09-10	morning	15	jeep	4
305	2	2023-03-18	noon	20	bus	3
306	5	2021-11-30	evening	10	jeep	5
307	2	2022-05-12	morning	5	jeep	2
308	3	2022-07-25	noon	15	bus	4
309	1	2022-07-25	noon	20	jeep	1
310	4	2022-07-25	morning	10	jeep	5
311	8	2023-05-01	evening	10	Jeep	1
312	30	2022-05-31	evening	10	Jeep	4
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Procedure - 2

```
async function addSafariRideBooking(p_NumOfPeople, p_Date, p_Slots, p_Distance, p_VehicleType, p_Name, p_ContactNumber, p_Email, p_Birthdate, p_SanctuaryID) {
    const connection = await mysql.createConnection({
    host: 'localhost',
    user: 'root',
    password: 'Parth@3060',
    database: 'wildlife_sanctuary_up'
});

try {
    const [rows] = await connection.execute(`CALL AddSafariRideBooking(?, ?, ?, ?, ?, ?, ?, ?, ?, ?)`, [p_NumOfPeople, p_Date, p_Slots, p_Distance, p_VehicleType, p_Name, p_ContactNumber

    console.log(`Safari ride booking added with ID ${rows[0][0].v_SafariRideBookingID}`);
    } catch (error) {
    console.log(error.message);
    } finally {
    await connection.end();
    }
}

// Example usage:
addSafariRideBooking(4, '2023-05-01', '9:00 AM - 10:00 AM', 5, 'Jeep', 'John Doe', '555-555-5555', 'johndoe@example.com', '1990-01-01', 1);
```

3) Procedure to retrieve the list of passengers who have booked safari rides for a given date and time slot , given sanctuary and safari ride booking information:

```
DELIMITER $$

CREATE PROCEDURE GetPassengersForSafariRide(IN DateParam DATE, IN SlotParam VARCHAR(100),IN SanctuaryIDParam INT)
BEGIN
    SELECT p.Name, s.Name AS Sanctuary, s.City, s.State, s.Country, s.AreaSize, s.Description, s.SanctuaryID, srb.NumOfPeople, srb.Distance, srb.VehicleType
  FROM Passengers p
  JOIN SafariRideBookings srb ON p.SafariRideBookingID = srb.SafariRideBookingID
  JOIN Sanctuaries s ON srb.SanctuaryID = s.SanctuaryID
  WHERE srb.Date = DateParam
  AND srb.Slots = SlotParam
  AND s.SanctuaryID = SanctuaryIDParam;

END $$

DELIMITER ;


CALL GetPassengersForSafariRide('2022-07-25','noon',1);
```

– —----------

Output :

```
76 •  CALL GetPassengersForSafariRide('2022-07-25','noon',1);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Name	Sanctuary	City	State	Country	AreaSize	Description	SanctuaryID	NumOfPeople	Distance	VehicleType
Kunal Gupta	Kanha National Park	Mandla	Madhya Pradesh	India	940	Famous for Bengal Tigers and Barasingha	1	3	15	bus
Manoj Singh	Kanha National Park	Mandla	Madhya Pradesh	India	940	Famous for Bengal Tigers and Barasingha	1	1	20	jeep

Procedure - 3

```javascript
async function getPassengersForSafariRide(DateParam, SlotParam, SanctuaryIDParam) {
  try {
    const connection = await mysql.createConnection({
      host: 'localhost',
      user: 'root',
      password: 'Parth@3060',
      database: 'wildlife_sanctuary_up'
    });

    const [rows, fields] = await connection.execute(
      `CALL GetPassengersForSafariRide(?, ?, ?)`,
      [DateParam, SlotParam, SanctuaryIDParam]
    );

    connection.end();

    return rows;
  } catch (error) {
    console.error(error);
    throw error;
  }
}
getPassengersForSafariRide('2023-04-20', 'morning', 1)
  .then(rows => console.log(rows))
  .catch(error => console.error(error));
```

4) Stored Procedure to get full details regarding a particular sanctuary

```
DELIMITER $$


CREATE PROCEDURE GetSanctuaryStats(IN SanctuaryIDParam INT)
BEGIN
   DECLARE TotalAnimals INT DEFAULT 0;
   DECLARE TotalEmployees INT DEFAULT 0;
   DECLARE TotalDoctors INT DEFAULT 0;
   DECLARE TotalHabitatAreas INT DEFAULT 0;
   DECLARE TotalTrees INT DEFAULT 0;
   DECLARE TotalBirds INT DEFAULT 0;
   DECLARE TotalPassengers INT DEFAULT 0;
   DECLARE TotalSafariRideBookings INT DEFAULT 0;
   DECLARE TotalMedicalSupplies INT DEFAULT 0;
   DECLARE TotalFoodSupplies INT DEFAULT 0;
   DECLARE i INT DEFAULT 0;
   DECLARE numHabitatAreas INT DEFAULT 0;
   DECLARE numTrees INT DEFAULT 0;
   DECLARE numBirds INT DEFAULT 0;
   DECLARE numPassengers INT DEFAULT 0;
   DECLARE numSafariRideBookings INT DEFAULT 0;
   DECLARE numMedicalSupplies INT DEFAULT 0;
   DECLARE numFoodSupplies INT DEFAULT 0;
   DECLARE No_Of_People_Visited INT DEFAULT 0;

   -- Get total number of animals in the sanctuary
         SELECT COUNT(*) INTO TotalAnimals FROM Animals WHERE SanctuaryID =
SanctuaryIDParam;

   -- Get total number of employees in the sanctuary
        SELECT COUNT(*) INTO TotalEmployees FROM Employees WHERE SanctuaryID =
SanctuaryIDParam;

   -- Get total number of doctors in the sanctuary
   SELECT COUNT(*) INTO TotalDoctors FROM Doctors WHERE SanctuaryID = SanctuaryIDParam;
```

```
-- Get total number of habitat areas in the sanctuary
SELECT COUNT(*) INTO TotalHabitatAreas FROM HabitatAreas WHERE SanctuaryID = SanctuaryIDParam;

-- Get total number of trees in the sanctuary
SELECT COUNT(*) INTO TotalTrees FROM Trees WHERE HabitatAreaID IN (SELECT HabitatAreaID FROM HabitatAreas WHERE SanctuaryID = SanctuaryIDParam);

-- Get total number of birds in the sanctuary
SELECT COUNT(*) INTO TotalBirds FROM Birds WHERE SanctuaryID = SanctuaryIDParam;

-- Get total number of safari ride bookings in the sanctuary
SELECT COUNT(*) INTO TotalSafariRideBookings FROM SafariRideBookings WHERE SanctuaryID = SanctuaryIDParam;

SELECT SUM(NumOfPeople) INTO No_Of_People_Visited FROM SafariRideBookings WHERE SanctuaryID = SanctuaryIDParam;
-- Get total number of medical supplies in the sanctuary
SELECT COUNT(*) INTO TotalMedicalSupplies FROM MedicalSupplies WHERE SanctuaryID = SanctuaryIDParam;

-- Get total number of food supplies in the sanctuary
SELECT COUNT(*) INTO TotalFoodSupplies FROM FoodSupplies WHERE SanctuaryID = SanctuaryIDParam;

SELECT TotalAnimals,TotalEmployees,TotalDoctors,TotalHabitatAreas,TotalTrees,TotalBirds,TotalSafariRideBookings,No_Of_People_Visited,TotalMedicalSupplies,TotalFoodSupplies;

END $$

DELIMITER ;

CALL GetSanctuaryStats(4);

– —----------
```

Output :

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

TotalAnimals	TotalEmployees	TotalDoctors	TotalHabitatAreas	TotalTrees	TotalBirds	TotalSafariRideBookings	No_Of_People_Visited	TotalMedicalSupplies	TotalFoodSupplies
3	2	2	2	2	2	2	33	3	3

Procedure - 4

```
async function getSanctuaryStats() {
  const connection = await mysql.createConnection({
    host: 'localhost',
    user: 'root',
    password: 'Parth@3060',
    database: 'wildlife_sanctuary_up'
  });

  const [rows] = await connection.execute('CALL GetSanctuaryStats(1)');
  console.log(rows);

  connection.end();
}

getSanctuaryStats();
```

5) Stored procedure to increase the salary of all employees in a given department at a given sanctuary by a given percentage.

```
DELIMITER $$

CREATE PROCEDURE IncreaseSalaryByDepartment(
   IN SanctuaryIDParam INT,
   IN DepartmentParam VARCHAR(100),
   IN PercentageParam DECIMAL(10,2)
)
BEGIN
   UPDATE Employees
   SET Salary = Salary * (1 + PercentageParam/100)
   WHERE SanctuaryID = SanctuaryIDParam AND Department = DepartmentParam;
END $$

DELIMITER ;

CALL IncreaseSalaryByDepartment(1,'Tourism',15);
```

– —----------

Before :

```
165 •   SELECT * FROM Employees WHERE Department = 'Tourism' AND SanctuaryID = 1;
166
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

EmployeeID	EmployeeName	ContactNumber	JobTitle	Department	Salary	HireDate	SanctuaryID
102	Aanya Singh	9876543211	Tour Guide	Tourism	20000.00	2021-02-01	1
111	Aarsh Patel	987654320	Tour Guide	Tourism	25000.00	2021-01-31	1
112	Anuj Singh	9876543211	Tour Guide	Tourism	20000.00	2021-02-01	1
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

After :

```
67 •    SELECT * FROM Employees WHERE Department = 'Tourism' AND SanctuaryID = 1;
68
```

esult Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

EmployeeID	EmployeeName	ContactNumber	JobTitle	Department	Salary	HireDate	SanctuaryID
102	Aanya Singh	9876543211	Tour Guide	Tourism	23000.00	2021-02-01	1
111	Aarsh Patel	987654320	Tour Guide	Tourism	28750.00	2021-01-31	1
112	Anuj Singh	9876543211	Tour Guide	Tourism	23000.00	2021-02-01	1
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Procedure - 5

```
8
9   async function increaseSalaryByDepartment() {
10    try {
11      const connection = await mysql.createConnection(connectionConfig);
12      const SanctuaryIDParam = 1;
13      const DepartmentParam = 'IT';
14      const PercentageParam = 10.0;
15
16      const [rows] = await connection.execute(
17        `UPDATE Employees
18         SET Salary = Salary * (1 + ?/100)
19         WHERE SanctuaryID = ? AND Department = ?`,
20        [PercentageParam, SanctuaryIDParam, DepartmentParam]
21      );
22
23      console.log(`Affected rows: ${rows.affectedRows}`);
24
25      connection.end();
26    } catch (err) {
27      console.error(err);
28    }
29  }
30
31  increaseSalaryByDepartment();
32
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL COMMENTS

```
PS C:\Users\parth\OneDrive\Desktop\dbms_project> node "c:\Users\parth\OneDrive\Desktop\dbms_project\stored_procedures\5.js"
Affected rows: 0
PS C:\Users\parth\OneDrive\Desktop\dbms_project>
```

Code
Code
Code

6) Stored Procedure to get the top 3 most booked safari ride dates for a specific sanctuary.

```
DROP PROCEDURE IF EXISTS GetTop3SafariRideDates;
DELIMITER $$

CREATE PROCEDURE GetTop3SafariRideDates(
   IN SanctuaryNameParam VARCHAR(100)
)
BEGIN
   SELECT Date, COUNT(*) AS NumOfBookings
   FROM (
      SELECT DISTINCT sr.SafariRideBookingID, sr.Date
      FROM Sanctuaries s
      JOIN SafariRideBookings sr ON s.SanctuaryID = sr.SanctuaryID
      WHERE s.Name = SanctuaryNameParam
   ) AS temp
   GROUP BY Date
   ORDER BY NumOfBookings DESC
   LIMIT 3;
END $$

DELIMITER ;

CALL GetTop3SafariRideDates('Kanha National Park');

-- —----------------------------------------------------------------------------------------------------------------
```

Output :

```
194 •   CALL GetTop3SafariRideDates('Kanha National Park');
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Date	NumOfBookings
2022-07-25	2
2022-06-15	1
2023-05-01	1

Procedure - 6 :

```javascript
async function getTop3SafariRideDates(SanctuaryNameParam) {
  try {
    const connection = await mysql.createConnection({
      host: 'localhost',
      user: 'root',
      password: 'Parth@3060',
      database: 'wildlife_sanctuary_up'
    });

    const [rows, fields] = await connection.execute(`CALL GetTop3SafariRideDates('${SanctuaryNameParam}')`);

    connection.end();

    return rows;
  } catch (err) {
    console.error(err);
    throw new Error('Failed to get top 3 safari ride dates');
  }
}

getTop3SafariRideDates('Sanctuary A')
  .then((result) => console.log(result))
  .catch((error) => console.error(error));
```

(7) Stored Procedure to get the list of all habitat areas that have a population of birds greater than the average population of birds in all habitat areas of a specific sanctuary.

```
DELIMITER $$

CREATE PROCEDURE GetHabitatAreasWithHighBirdPopulation(
   IN SanctuaryNameParam VARCHAR(100)
)
BEGIN
   SELECT ha.Name AS HabitatArea, b.Population
   FROM Sanctuaries s
   JOIN HabitatAreas ha ON s.SanctuaryID = ha.SanctuaryID
   JOIN Birds b ON ha.HabitatAreaID = b.HabitatAreaID
   WHERE s.Name = SanctuaryNameParam AND b.Population > (
      SELECT AVG(b2.Population)
      FROM Birds b2
      JOIN HabitatAreas ha2 ON b2.HabitatAreaID = ha2.HabitatAreaID
      JOIN Sanctuaries s2 ON ha2.SanctuaryID = s2.SanctuaryID
      WHERE s2.Name = SanctuaryNameParam
   );
END $$

DELIMITER ;

CALL GetHabitatAreasWithHighBirdPopulation('Sundarbans National Park');

-- ----------------------------------------------------------------------------------------
```

Output :

```
220 •   CALL GetHabitatAreasWithHighBirdPopulation('Sundarbans National Park');
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

HabitatArea	Population
Mangrove Forest	75

Procedure - 7

```javascript
async function getHabitatAreasWithHighBirdPopulation(sanctuaryName) {
  try {
    const connection = await mysql.createConnection({
      host: 'localhost',
      user: 'root',
      password: 'Parth@3060',
      database: 'wildlife_sanctuary_up',
    });

    const [rows] = await connection.execute(
      `CALL GetHabitatAreasWithHighBirdPopulation(?)`,
      [sanctuaryName]
    );

    connection.end();
    return rows;
  } catch (err) {
    console.error(err);
  }
}
getHabitatAreasWithHighBirdPopulation('Kanha National Park')
  .then((rows) => console.log(rows))
  .catch((error) => console.error(error));
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL COMMENTS

```
PS C:\Users\parth\OneDrive\Desktop\dbms_project> node "c:\Users\parth\OneDrive\Desktop\dbms_project\stored_procedures\7.js"
[
  [ { HabitatArea: 'Bamboo Forest', Population: 100 } ],
  ResultSetHeader {
    fieldCount: 0,
    affectedRows: 0,
    insertId: 0,
    info: '',
    serverStatus: 34,
    warningStatus: 0
  }
]
PS C:\Users\parth\OneDrive\Desktop\dbms_project>
```

Code
Code
Code

(8) Stored procedure to update the population of a given bird species in all sanctuaries by a given percentage:

```
SELECT * FROM Birds WHERE Type = 'Raptor';


DELIMITER $$
CREATE PROCEDURE UpdateBirdPopulation(
   IN BirdTypeParam VARCHAR(100),
   IN PopulationIncreasePercentage DECIMAL(5, 2)
)
BEGIN
   DECLARE done INT DEFAULT FALSE;
   DECLARE currentSanctuaryID INT;
   DECLARE currentPopulation INT;

   -- Declare cursor for the Sanctuaries table
   DECLARE sanctuaryCursor CURSOR FOR SELECT SanctuaryID FROM Sanctuaries;

   -- Declare continue handler for the cursor
   DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

   -- Open the cursor
   OPEN sanctuaryCursor;

   -- Loop through each row of the cursor
   sanctuary_loop: LOOP
      -- Fetch the next row of the cursor
      FETCH sanctuaryCursor INTO currentSanctuaryID;

      -- Exit loop if there are no more rows to fetch
      IF done THEN
         LEAVE sanctuary_loop;
      END IF;

      -- Get the current population of the given bird species in the sanctuary
        SELECT Population INTO currentPopulation FROM Birds WHERE Type = BirdTypeParam AND
SanctuaryID = currentSanctuaryID;

      -- Calculate the new population after the percentage increase
        SET currentPopulation = currentPopulation + (currentPopulation * PopulationIncreasePercentage /
100);
```

```
    -- Update the bird population in the current sanctuary
        UPDATE Birds SET Population = currentPopulation WHERE Type = BirdTypeParam AND SanctuaryID = currentSanctuaryID;
  END LOOP;

  -- Close the cursor
  CLOSE sanctuaryCursor;
END $$
DELIMITER ;
```

```
CALL UpdateBirdPopulation('Raptor',50);
```

- - —----------

Before :

BirdID	Type	Species	HabitatAreaID	Population	SanctuaryID
2	Raptor	Crested Serpent Eagle	102	20	2
5	Raptor	Indian Vulture	105	10	5
7	Raptor	Shikra	107	15	2
10	Raptor	Tawny Eagle	110	25	5
NULL	NULL	NULL	NULL	NULL	NULL

After :

```
226 •   SELECT * FROM Birds WHERE Type = 'Raptor';
```

Result Grid | Filter Rows: | Edit: | Export/Import:

	BirdID	Type	Species	HabitatAreaID	Population	SanctuaryID
▶	2	Raptor	Crested Serpent Eagle	102	30	2
	5	Raptor	Indian Vulture	105	15	5
	7	Raptor	Shikra	107	22	2
	10	Raptor	Tawny Eagle	110	37	5
*	NULL	NULL	NULL	NULL	NULL	NULL

Procedure - 8

```javascript
async function updateBirdPopulation(BirdTypeParam, PopulationIncreasePercentage) {
  try {
    const connection = await mysql.createConnection({
      host: 'localhost',
      user: 'root',
      password: 'Parth@3060',
      database: 'wildlife_sanctuary_up'
    });

    // Call the stored procedure
    const [rows, fields] = await connection.execute(
      'CALL UpdateBirdPopulation(?, ?)', [BirdTypeParam, PopulationIncreasePercentage]);

    connection.end();

    // Return the results
    return rows;
  } catch (error) {
    // Handle the error
    console.error(error);
  }
}

// Call the function
updateBirdPopulation('sparrow', 10.5)
  .then(result => console.log(result))
  .catch(error => console.error(error));
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL COMMENTS

```
PS C:\Users\parth\OneDrive\Desktop\dbms_project> node "c:\Users\parth\OneDrive\Desktop\dbms_project\stored_procedures\8.js"
ResultSetHeader {
  fieldCount: 0,
  affectedRows: 0,
  insertId: 0,
  info: '',
  serverStatus: 66,
  warningStatus: 0
}
PS C:\Users\parth\OneDrive\Desktop\dbms_project>
```

Functions :

(1) Function to get a List of Medical Supplies and their cost for a particular Sanctuary

```
DELIMITER $$

CREATE FUNCTION GetMedicalSuppliesInSanctuary(sanctuaryID INT)
RETURNS VARCHAR(5000)
DETERMINISTIC
BEGIN
   DECLARE medicalSupplyName VARCHAR(100);
   DECLARE totalCost DECIMAL(10, 2);
   DECLARE output varchar(5000) DEFAULT '';
   DECLARE done INT DEFAULT FALSE;

   DECLARE cur CURSOR FOR
        SELECT MedicalSupplies.Name, SUM(MedicalSupplies.Quantity * MedicalSupplies.CostPerUnit)
AS TotalCost
      FROM MedicalSupplies
      INNER JOIN Sanctuaries ON MedicalSupplies.SanctuaryID = Sanctuaries.SanctuaryID
      WHERE Sanctuaries.SanctuaryID = sanctuaryID
      GROUP BY MedicalSupplies.Name;

   DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

   OPEN cur;



  WHILE NOT done DO
      FETCH cur INTO medicalSupplyName, totalCost;
      IF NOT done THEN
         SET output = CONCAT(output, medicalSupplyName, ' ', totalCost, '--------');
      END IF;
   END WHILE;

   CLOSE cur;

   return output;
```

END $$

DELIMITER ;

SELECT (GetMedicalSuppliesInSanctuary(1)) AS MedicalSuppliesCosting;

– —----------

Output :

```
41 •  SELECT (GetMedicalSuppliesInSanctuary(1)) AS MedicalSuppliesCosting;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

MedicalSuppliesCosting
Vitamin C Tablets 2000.00--------Antibiotic Ointment 1500.00--------Disinfectant Spray 3000.00--------

Stored Function-1

```
const mysql = require('mysql2/promise');

async function getSafariRideReport(startDate, endDate) {
  try {
    const connection = await mysql.createConnection({
      host: 'localhost',
      user: 'root',
      password: 'Parth@3060',
      database: 'wildlife_sanctuary_up'
    });

    const [rows] = await connection.query(`SELECT GetSafariRideReport('${startDate}', '${endDate}') as report`);

    const report = rows[0].report;

    console.log(report);

    connection.end();
  } catch (error) {
    console.error(error);
  }
}

getSafariRideReport('2022-01-01', '2023-12-31');
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL COMMENTS

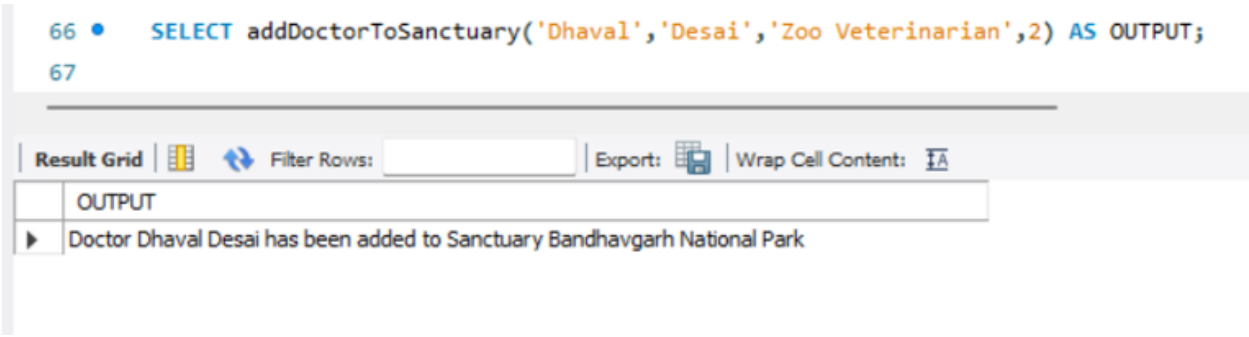
```
PS C:\Users\parth\OneDrive\Desktop\dbms_project> node "c:\Users\parth\OneDrive\Desktop\dbms_project\stored_functions\1.js"
null
PS C:\Users\parth\OneDrive\Desktop\dbms_project>
```

(2) Function for adding a doctor to a sanctuary

```
DELIMITER $$
CREATE FUNCTION addDoctorToSanctuary(
  doctorFirstName VARCHAR(100),
  doctorLastName VARCHAR(100),
  doctorSpecialization VARCHAR(100),
  sanctuaryIDParam INT
)
RETURNS VARCHAR(200)
DETERMINISTIC
BEGIN
  DECLARE SanctuaryName VARCHAR(50);
  INSERT INTO Doctors (FirstName, LastName, Specialization, SanctuaryID)
  VALUES (doctorFirstName, doctorLastName, doctorSpecialization, sanctuaryIDParam);

  SELECT name INTO SanctuaryName FROM Sanctuaries WHERE SanctuaryID = sanctuaryIDParam;

   RETURN CONCAT('Doctor ', doctorFirstName, ' ', doctorLastName, ' has been added to Sanctuary ',
SanctuaryName);
END$$
DELIMITER ;

SELECT addDoctorToSanctuary('Dhaval','Desai','Zoo Veterinarian',2) AS OUTPUT;
```

-- ----------

Output :



Stored Function - 2

```javascript
async function addDoctorToSanctuary(doctorFirstName, doctorLastName, doctorSpecialization, sanctuaryIDParam) {
  try {
    // Create a connection to the database
    const connection = await mysql.createConnection({
      host: 'localhost',
      user: 'root',
      password: 'Parth@3060',
      database: 'wildlife_sanctuary_up'
    });

    // Call the addDoctorToSanctuary function and get the result
    const [rows, fields] = await connection.execute(
      `SELECT addDoctorToSanctuary(?, ?, ?, ?) AS result`,
      [doctorFirstName, doctorLastName, doctorSpecialization, sanctuaryIDParam]
    );

    // Close the connection to the database
    await connection.end();

    // Return the result
    return rows[0].result;
  } catch (error) {
    // Handle any errors that may occur
    console.error(error);
    return 'An error occurred while adding the doctor to the sanctuary.';
  }
}

// Call the addDoctorToSanctuary function
addDoctorToSanctuary('John', 'Doe', 'Cardiology', 1)
  .then(result => console.log(result))
  .catch(error => console.error(error));
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL COMMENTS

```
PS C:\Users\parth\OneDrive\Desktop\dbms_project> node "c:\Users\parth\OneDrive\Desktop\dbms_project\stored_functions\2.js"
Doctor John Doe has been added to Sanctuary Kanha National Park
PS C:\Users\parth\OneDrive\Desktop\dbms_project>
```

(3) Function to get the total number of bookings for a given passenger in all sanctuaries:

```
DELIMITER $$

CREATE FUNCTION getTotalBookingsForPassenger(passenger_name VARCHAR(200))
RETURNS VARCHAR(1000)
DETERMINISTIC
BEGIN

DECLARE Safari_date  DATE;
DECLARE safari_slot VARCHAR(20);
DECLARE SanctuaryName VARCHAR(50);
DECLARE done INT DEFAULT FALSE;
DECLARE output varchar(1000) DEFAULT '';

        DECLARE cur CURSOR FOR SELECT
SafariRideBookings.Date,SafariRideBookings.Slots,Sanctuaries.name  FROM SafariRideBookings
                JOIN Passengers ON SafariRideBookings.SafariRideBookingID =
Passengers.SafariRideBookingID
      JOIN Sanctuaries ON Sanctuaries.SanctuaryID = SafariRideBookings.SanctuaryID
      WHERE Passengers.name = passenger_name;



      DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

                OPEN cur;

      SET output = 'Previous Safari Trips ';

      WHILE NOT done DO
      FETCH cur INTO Safari_date,safari_slot,SanctuaryName;
      IF NOT done THEN
            SET output = CONCAT(output,  '  Safari Date : ', Safari_date , '   Safari Slot : ',safari_slot,'
Sanctuary Name : ',SanctuaryName,
        '.        .');
      END IF;
   END WHILE;

   CLOSE cur;
```

```
    return output;
END $$

DELIMITER ;

SELECT getTotalBookingsForPassenger('Priya Patel') AS OUTPUT;

-- ----------------------------------------------------------------------

SELECT * FROM Animals;

-- ----------------------------------------------------------------
```

Output :

```
116 •   SELECT getTotalBookingsForPassenger('Priya Patel') AS OUTPUT;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

OUTPUT
Previous Safari Trips Safari Date : 2022-06-15 Safari Slot : morning Sanctuary Name : Kanha National Park. . Safari Date : 2022-07-25 Safari Slot : noon Sanctuary Name : Kanha National Park.

Stored Function - 3

```
1  async function getTotalBookingsForPassenger(passengerName) {
2    // create connection to MySQL database
3    const connection = await mysql.createConnection({
4      host: 'localhost',
5      user: 'root',
6      password: 'Parth@1068',
7      database: 'wildlife_sanctuary_up'
8    });
9
10   // call MySQL function and fetch result
11   const [rows] = await connection.execute(
12     `SELECT getTotalBookingsForPassenger('${passengerName}')`
13   );
14
15   // close database connection
16   await connection.end();
17
18   // extract output string from result
19   const output = rows[0]['getTotalBookingsForPassenger(\'' + passengerName + '\')'];
20
21   return output;
22 }
23
24 // example usage
25 getTotalBookingsForPassenger('John Doe')
26 .then((result) => {
27     console.log(result);
28     })
29 .catch((error) => {
30     console.error(error);
31     });
```

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   COMMENTS

PS C:\Users\parth\OneDrive\Desktop\dbms_project> node "c:\Users\parth\OneDrive\Desktop\dbms_project\stored_functions\3.js"
Error: Unknown column 'SafariRideBookings.SanctuaryID' in 'on clause'
    at PromiseConnection.execute (c:\Users\parth\OneDrive\Desktop\dbms_project\node_modules\@mysql2\promise.js:111:22)
    at getTotalBookingsForPassenger (c:\Users\parth\OneDrive\Desktop\dbms_project\stored_functions\3.js:13:35)
    at processTicksAndRejections (node:internal/process/task_queues:96:5) {
  code: 'ER_BAD_FIELD_ERROR',
  errno: 1054,
  sql: "SELECT getTotalBookingsForPassenger('John Doe')",
  sqlState: '42S22',
  sqlMessage: "Unknown column 'SafariRideBookings.SanctuaryID' in 'on clause'"
}
```

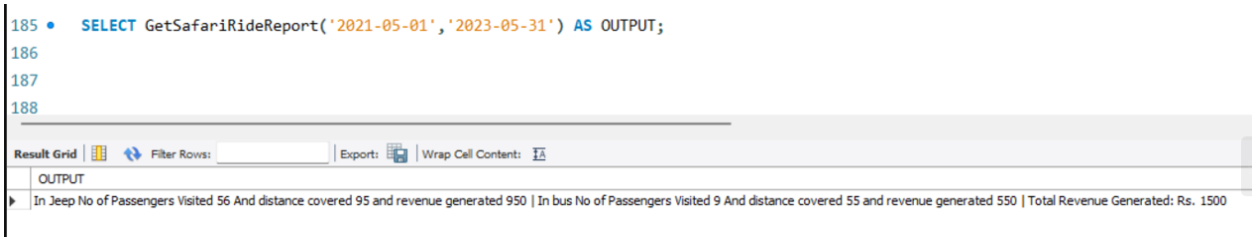
(4) Generate Safari Report and Calculate total revenue within date Range

```
DROP FUNCTION IF EXISTS GetSafariRideReport;
DELIMITER $$

CREATE FUNCTION GetSafariRideReport(startDate DATE, endDate DATE)
RETURNS VARCHAR(1000)
DETERMINISTIC
BEGIN

DECLARE vehicleTypeParam VARCHAR(100);
        DECLARE finished INT DEFAULT 0;
  DECLARE numOfPassengers INT;
  DECLARE distance INT;
  DECLARE revenue DECIMAL(10, 2);
  DECLARE totalRevenue DECIMAL(10, 2) DEFAULT 0;
  DECLARE report VARCHAR(1000) DEFAULT '';




  DECLARE vehicleCursor CURSOR FOR
    SELECT DISTINCT VehicleType
    FROM SafariRideBookings
    WHERE Date >= startDate AND Date <= endDate;

  DECLARE CONTINUE HANDLER FOR NOT FOUND SET finished = 1;

  -- loop through each vehicle type
  OPEN vehicleCursor;

  read_loop: LOOP
    -- fetch the next row
    FETCH vehicleCursor INTO vehicleTypeParam;

    -- exit loop if no more rows
    IF finished = 1 THEN
      LEAVE read_loop;
    END IF;
```

```
        -- calculate total passengers, distance, and revenue for the vehicle type
        SELECT SUM(NumOfPeople), SUM(Distance)
        INTO numOfPassengers, distance
        FROM SafariRideBookings
        WHERE VehicleType = vehicleTypeParam AND Date >= startDate AND Date <= endDate;

        SET revenue = distance * 10.0;

        -- concatenate the intermedia

                SET report = CONCAT(report, 'In ',vehicleTypeParam, ' No of passengers visited', numOfPassengers , ' And distance covered ', distance , ' and revenue generated ',revenue ,' | ');

        SET totalRevenue = totalRevenue + revenue;
    END LOOP;

    -- concatenate the total revenue to the report string
    SET report = CONCAT(report, '| Total Revenue Generated: Rs. ', totalRevenue);

   return report;
END$$

DELIMITER ;

SELECT GetSafariRideReport('2021-05-01','2023-05-31') AS OUTPUT;
```

– —----------

Output :



Stored Function - 4

```javascript
async function getSafariRideReport(startDate, endDate) {
  try {
    // create connection pool
    const pool = mysql.createPool({
      host: 'localhost',
      user: 'root',
      password: 'Parth@3060',
      database: 'wildlife_sanctuary_up',
      waitForConnections: true,
      connectionLimit: 10,
      queueLimit: 0
    });
    const connection = await pool.getConnection();
    const [rows] = await connection.query('SELECT GetSafariRideReport(?, ?) as report', [startDate, endDate]);
    connection.release();
    return rows[0].report;
  } catch (error) {
    console.error(error);
    throw new Error('Error executing MySQL function');
  }
}
getSafariRideReport('2022-01-01', '2022-12-31')
  .then(report => console.log(report))
  .catch(error => console.error(error));
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL COMMENTS

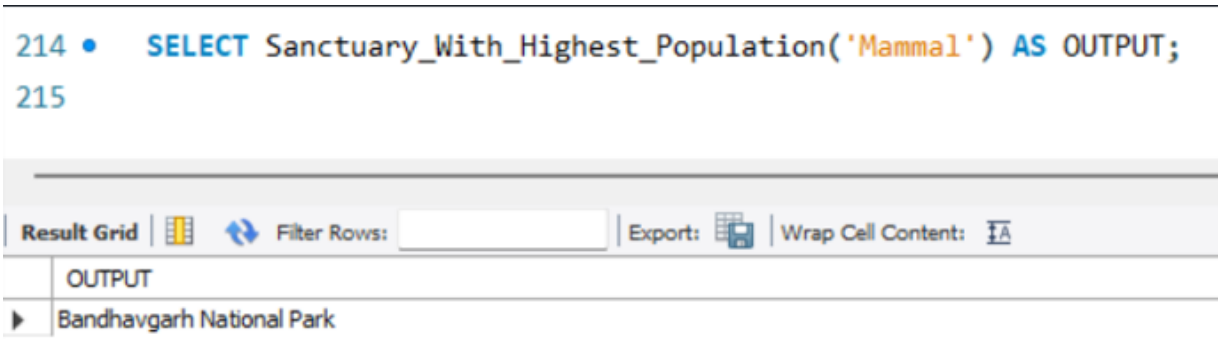
```
PS C:\Users\parth\OneDrive\Desktop\dbms_project> node "c:\Users\parth\OneDrive\Desktop\dbms_project\stored_functions\4.js"
null
```

(5) retrieve the name of the sanctuary with the highest population of a given animal species.

```
DROP PROCEDURE IF EXISTS Sanctuary_With_Highest_Population;
DELIMITER $$


CREATE FUNCTION Sanctuary_With_Highest_Population(TypeParam VARCHAR(100))
RETURNS VARCHAR(100)
DETERMINISTIC
BEGIN
   DECLARE sanctuary_name VARCHAR(100);

   SELECT s.Name
   INTO sanctuary_name
   FROM Sanctuaries s
   INNER JOIN Animals a ON s.SanctuaryID = a.SanctuaryID
   WHERE a.Type = TypeParam
   ORDER BY a.Population DESC
   LIMIT 1;

   RETURN sanctuary_name;
END$$

DELIMITER ;

SELECT Sanctuary_With_Highest_Population('Mammal') AS OUTPUT;
```

– —----------

Output :



Stored Function - 5

```javascript
const connection = mysql.createConnection({
  host: 'localhost',
  user: 'root',
  password: 'Parth@3060',
  database: 'wildlife_sanctuary_up',
});

function sanctuaryWithHighestPopulation(typeParam) {
  return new Promise((resolve, reject) => {
    const query = `SELECT Sanctuary_With_Highest_Population('${typeParam}') AS sanctuaryName`;

    connection.query(query, (error, results) => {
      if (error) {
        reject(error);
      } else {
        resolve(results[0].sanctuaryName);
      }
    });
  });
}

// Example usage
sanctuaryWithHighestPopulation('Mammal')
  .then((sanctuaryName) => console.log(`Sanctuary with highest lion population: ${sanctuaryName}`))
  .catch((error) => console.error(error));
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL COMMENTS

```
PS C:\Users\parth\OneDrive\Desktop\dbms_project> node "c:\Users\parth\OneDrive\Desktop\dbms_project\stored_functions\5.js"
Sanctuary with highest lion population: Bandhavgarh National Park
```

Code
Code
Code

Triggers :

(1) Trigger to prevent deletion of a sanctuary record if it is referenced by other tables

```
DELIMITER $$

CREATE TRIGGER prevent_sanctuary_deletion BEFORE DELETE ON Sanctuaries
FOR EACH ROW

BEGIN

  IF EXISTS(SELECT * FROM Animals WHERE SanctuaryID = OLD.SanctuaryID)
    OR EXISTS(SELECT * FROM Employees WHERE SanctuaryID = OLD.SanctuaryID)
    OR EXISTS(SELECT * FROM Doctors WHERE SanctuaryID = OLD.SanctuaryID)
    OR EXISTS(SELECT * FROM HabitatAreas WHERE SanctuaryID = OLD.SanctuaryID)
    OR EXISTS(SELECT * FROM Trees WHERE SanctuaryID = OLD.SanctuaryID)
    OR EXISTS(SELECT * FROM Birds WHERE SanctuaryID = OLD.SanctuaryID)
    OR EXISTS(SELECT * FROM SafariRideBookings WHERE SanctuaryID = OLD.SanctuaryID)
    OR EXISTS(SELECT * FROM Passengers WHERE SanctuaryID = OLD.SanctuaryID)
    OR EXISTS(SELECT * FROM MedicalSupplies WHERE SanctuaryID = OLD.SanctuaryID)
    OR EXISTS(SELECT * FROM FoodSupplies WHERE SanctuaryID = OLD.SanctuaryID)
  THEN
    SIGNAL SQLSTATE '45000'
            SET MESSAGE_TEXT = 'Cannot delete sanctuary record because it is referenced by other
tables';
  END IF;
END$$

DELIMITER ;

DELETE FROM Sanctuaries WHERE SanctuaryID = 5;

-- ---------------------------------------------------------------------------

SELECT * FROM FoodSupplies;
```

Output :

```
27 •   DELETE FROM Sanctuaries WHERE SanctuaryID = 5;
28
29
```

Output

Action Output

#	Time	Action	Message
444	01:14:30	CREATE TRIGGER prevent_sanctuary_deletion BEFORE DELETE ON Sanctuaries FOR EACH ROW BEGIN ...	Error Code: 1359. Trigger already exists
445	01:15:07	DROP TRIGGER IF EXISTS prevent_sanctuary_deletion	0 row(s) affected
446	01:15:12	CREATE TRIGGER prevent_sanctuary_deletion BEFORE DELETE ON Sanctuaries FOR EACH ROW BEGIN ...	0 row(s) affected
447	01:26:47	DELETE FROM Sanctuaries WHERE SanctuaryID = 5	Error Code: 1644. Discount amount is more than allowed.
448	01:27:16	DROP TRIGGER prevent_sanctuary_deletion	0 row(s) affected
449	01:27:16	CREATE TRIGGER prevent_sanctuary_deletion BEFORE DELETE ON Sanctuaries FOR EACH ROW BEGIN ...	0 row(s) affected
450	01:30:07	DELETE FROM Sanctuaries WHERE SanctuaryID = 5	Error Code: 1644. Cannot delete sanctuary record because it is referenced by other tables

Trigger -1

```
3  async function deleteSanctuaryRecord(sanctuaryId) {
4    const connection = await mysql.createConnection({
5      host: 'localhost',
6      user: 'root',
7      password: 'Parth@3060',
8      database: 'wildlife_sanctuary_up'
9    });
10
11   try {
12     await connection.execute(`DELETE FROM Sanctuaries WHERE SanctuaryID = ${sanctuaryId}`);
13     console.log(`Sanctuary record with ID ${sanctuaryId} deleted successfully!`);
14   } catch (err) {
15     console.error(err);
16   } finally {
17     connection.end();
18   }
19 }
20
21 deleteSanctuaryRecord(1);
22
```

(2) If a sanctuary has more than 2 doctors then it will not allow to insert any more doctors.

```
DROP TRIGGER IF EXISTS check_Doctor_count;
DELIMITER $$

CREATE TRIGGER check_Doctor_count
AFTER INSERT ON Doctors
FOR EACH ROW
BEGIN

   DECLARE num_doctors INT;

   SELECT COUNT(*) INTO num_doctors
   FROM Doctors
   WHERE SanctuaryID = NEW.SanctuaryID;


   IF num_doctors > 2 THEN
      SIGNAL SQLSTATE '45000'
         SET MESSAGE_TEXT = 'There are more than 2 doctors working in this sanctuary.';
  END IF;
END$$

DELIMITER ;

INSERT INTO Doctors (FirstName, LastName, Specialization, SanctuaryID) VALUES
('Peyush', 'Bansal', 'Zoo Veterinarian', 1);

-- ----------------------------------------------------------------
```

Output :

```
INSERT INTO Doctors (FirstName, LastName, Specialization, SanctuaryID) VALUES
('Peyush', 'Bansal', 'Zoo Veterinarian', 1);
```

Context Help

Output

Action Output

#	Time	Action		Message
473	02:06:49	CREATE TRIGGER check_medical_supplies AFTER INSERT ON Doctors FOR EACH ROW BEGIN	-- Decla...	Error Code: 1064. You have an error in your SQL syntax; check the manual that corresponds to your MySQL serv...
474	02:08:46	CREATE TRIGGER check_medical_supplies AFTER INSERT ON Doctors FOR EACH ROW BEGIN	-- Decla...	0 row(s) affected
475	02:09:12	SELECT * FROM Doctors LIMIT 0, 1000		10 row(s) returned
476	02:10:28	DROP PROCEDURE IF EXISTS check_medical_supplies		0 row(s) affected, 1 warning(s): 1305 PROCEDURE wildlife_sanctuary_up.check_medical_supplies does not exist
477	02:10:39	CREATE TRIGGER check_medical_supplies AFTER INSERT ON Doctors FOR EACH ROW BEGIN	-- Decla...	Error Code: 1359. Trigger already exists
478	02:10:53	DROP TRIGGER IF EXISTS check_medical_supplies		0 row(s) affected
479	02:11:01	CREATE TRIGGER check_medical_supplies AFTER INSERT ON Doctors FOR EACH ROW BEGIN	-- Decla...	0 row(s) affected
480	02:11:11	INSERT INTO Doctors (FirstName, LastName, Specialization, SanctuaryID) VALUES ('Peyush', 'Bansal', 'Zoo V...		Error Code: 1644. There are more than 2 doctors working in this sanctuary.

Trigger - 2

```
const connection = mysql.createConnection({
  host: 'localhost',
  user: 'root',
  password: 'Parth@3060',
  database: 'wildlife_sanctuary_up'
});

async function checkDoctorCountTrigger(newDoctor) {
  try {
    await connection.query('INSERT INTO Doctors SET ?', newDoctor);
    console.log('Doctor added successfully');
  } catch (error) {
    console.error(error);
    console.log('Error adding doctor');
  }
}

// example usage
const newDoctor = {
  DoctorID: 4,
  SanctuaryID: 1,
  FirstName: 'Jane',
  LastName: 'Doe',
  Specialization: 'Surgery'
};

checkDoctorCountTrigger(newDoctor);
```

(3) Trigger to calculate the total cost of medical supplies for the sanctuary after the new doctor works as compared to the total salaries of veterinarians working in the same sanctuary.

```
DROP TRIGGER IF EXISTS check_medical_supplies;

DELIMITER $$

CREATE TRIGGER check_medical_supplies
AFTER INSERT ON Doctors
FOR EACH ROW
BEGIN

        DECLARE total_cost INT;

   -- Calculate the total cost of medical supplies for the sanctuary that the new doctor works in
    SELECT SUM(CostPerUnit * Quantity) INTO total_cost
    FROM MedicalSupplies
    WHERE SanctuaryID = NEW.SanctuaryID;

      IF total_cost > 0.1 * (SELECT SUM(Salary) FROM Employees WHERE JobTitle = 'Veterinarian'
AND SanctuaryID = NEW.SanctuaryID) THEN
                        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'The total cost of
medical supplies for this sanctuary is too high compared to the salaries of the doctors.';
    END IF;
END$$

DELIMITER ;

INSERT INTO Doctors (FirstName, LastName, Specialization, SanctuaryID) VALUES
('Ajit', 'Pawar', 'Veterinarian', 1);



-- ----------------------------
```

Output :

```
INSERT INTO Doctors (FirstName, LastName, Specialization, SanctuaryID) VALUES
('Ajit', 'Pawar', 'Veterinarian', 1);

-- -------------------------------------------------------
```

Output

Action Output

#	Time	Action	Message
486	02:19:12	CREATE TRIGGER check_Doctor_count AFTER INSERT ON Doctors FOR EACH RO...	0 row(s) affected
487	02:19:19	INSERT INTO Doctors (FirstName, LastName, Specialization, SanctuaryID) VALUES ('P...	Error Code: 1644. There are more than 2 doctors working in this sanctuary.
488	02:21:01	CREATE TRIGGER check_medical_supplies AFTER INSERT ON Doctors FOR EACH R...	Error Code: 1327. Undeclared variable: total_cost
489	02:21:47	SELECT * FROM Employees LIMIT 0, 1000	12 row(s) returned
490	02:22:09	SELECT * FROM Employees LIMIT 0, 1000	12 row(s) returned
491	02:23:44	CREATE TRIGGER check_medical_supplies AFTER INSERT ON Doctors FOR EACH R...	Error Code: 1359. Trigger already exists
492	02:24:01	DROP TRIGGER IF EXISTS check_medical_supplies	0 row(s) affected
493	02:24:07	CREATE TRIGGER check_medical_supplies AFTER INSERT ON Doctors FOR EACH R...	0 row(s) affected
494	02:24:37	SELECT * FROM MedicalSupplies LIMIT 0, 1000	15 row(s) returned
495	02:25:02	INSERT INTO Doctors (FirstName, LastName, Specialization, SanctuaryID) VALUES ('Aji...	Error Code: 1644. There are more than 2 doctors working in this sanctuary.
496	02:26:40	DROP TRIGGER IF EXISTS check_Doctor_count	0 row(s) affected
497	02:26:40	CREATE TRIGGER check_Doctor_count AFTER INSERT ON Doctors FOR EACH RO...	0 row(s) affected
498	02:26:51	INSERT INTO Doctors (FirstName, LastName, Specialization, SanctuaryID) VALUES ('Aji...	Error Code: 1644. The total cost of medical supplies for this sanctuary is too high compared to the salaries of the doctors.

Trigger - 3

```
const connection = mysql.createConnection({
  host: 'localhost',
  user: 'root',
  password: 'Parth@3060',
  database: 'wildlife_sanctuary_up'
});

connection.connect((err) => {
  if (err) throw err;
  console.log('Connected to MySQL database');
});

const checkMedicalSuppliesTrigger = `
DELIMITER $$
CREATE TRIGGER check_medical_supplies
AFTER INSERT ON Doctors
FOR EACH ROW
BEGIN

  DECLARE total_cost INT;

  -- Calculate the total cost of medical supplies for the sanctuary that the new doctor works in
  SELECT SUM(CostPerUnit * Quantity) INTO total_cost
  FROM MedicalSupplies
  WHERE SanctuaryID = NEW.SanctuaryID;

  IF total_cost > 0.1 * (SELECT SUM(Salary) FROM Employees WHERE JobTitle = 'Veterinarian' AND SanctuaryID = NEW.SanctuaryID) THEN
    SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'The total cost of medical supplies for this sanctuary is too high compared to the salaries of the doctors.';
  END IF;
END$$
DELIMITER ;
`;

const createTrigger = async (triggerSql) => {
  return new Promise((resolve, reject) => {
    connection.query(triggerSql, (error, results, fields) => {
      if (error) {
        reject(error);
      } else {
        resolve(results);
      }
    });
  });
};

createTrigger(checkMedicalSuppliesTrigger)
  .then((results) => {
    console.log('Trigger created successfully');
  })
  .catch((error) => {
    console.error(error);
  });
```

(4) Trigger to update the salary of employees who have the job title of "Veterinarian" after a new record is inserted into the "Doctors" table.

```
DELIMITER $$

CREATE TRIGGER update_salary AFTER INSERT ON Doctors
FOR EACH ROW
BEGIN
   UPDATE Employees SET Salary = Salary + 5000 WHERE JobTitle = 'Veterinarian';
END$$

DELIMITER ;

INSERT INTO Doctors (FirstName, LastName, Specialization, SanctuaryID) VALUES
('Ajit', 'Pawar', 'Veterinarian', 1);


SELECT * FROM EMployees WHERE JobTitle = 'Veterinarian';
```

Before :

```
105 •   SELECT * FROM EMployees WHERE JobTitle = 'Veterinarian';
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

EmployeeID	EmployeeName	ContactNumber	JobTitle	Department	Salary	HireDate	SanctuaryID
103	Arjun Sharma	9876543212	Veterinarian	Animal Health	40000.00	2021-03-01	2
108	Kavya Patel	9876543217	Veterinarian	Animal Health	40000.00	2021-08-01	4
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

After :

```
SELECT * FROM EMployees WHERE JobTitle = 'Veterinarian';
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

EmployeeID	EmployeeName	ContactNumber	JobTitle	Department	Salary	HireDate	SanctuaryID
103	Arjun Sharma	9876543212	Veterinarian	Animal Health	45000.00	2021-03-01	2
108	Kavya Patel	9876543217	Veterinarian	Animal Health	45000.00	2021-08-01	4
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Trigger - 4

```javascript
async function createTrigger() {
  // create the connection to the database
  const connection = await mysql.createConnection({
    host: 'localhost',
    user: 'root',
    password: 'Parth@3060',
    database: 'wildlife_sanctuary_up'
  });

  try {
    // create the trigger
    await connection.execute(`
      DELIMITER $$
      CREATE TRIGGER update_salary AFTER INSERT ON Doctors
      FOR EACH ROW
      BEGIN
        UPDATE Employees SET Salary = Salary + 5000 WHERE JobTitle = 'Veterinarian';
      END$$
      DELIMITER ;
    `);

    console.log('Trigger created successfully');
  } catch (error) {
    console.error('Error creating trigger:', error);
  } finally {
    // close the connection
    connection.close();
  }
}

createTrigger();
```

(5) Trigger "bird_sanctuary_check" to design to execute automatically in response to an INSERT or UPDATE statement.

```
 DELIMITER$$
 CREATE TRIGGER bird_sanctuary_check
AFTER INSERT, UPDATE ON Animals
FOR EACH ROW
BEGIN
   DECLARE animal_type VARCHAR(20);
   DECLARE animal_id INT;
   DECLARE sanctuary_id INT;
   DECLARE bird_id INT;
   DECLARE bird_type VARCHAR(20);

   SELECT type INTO animal_type
   FROM Animals
   WHERE animal_id = NEW.animal_id;

   IF animal_type = 'Bird' THEN
      SELECT sanctuary_id INTO sanctuary_id
      FROM Sanctuaries
      WHERE sanctuary_id = NEW.sanctuary_id;

      IF sanctuary_id IS NULL THEN
         DELETE FROM Animals WHERE animal_id = NEW.animal_id;
         INSERT INTO Birds (bird_id, bird_type) VALUES (NEW.animal_id, 'Unknown');
      ELSE
         SELECT type INTO bird_type
         FROM Birds
         WHERE bird_id = NEW.animal_id;

         IF bird_type = 'Unknown' THEN
            UPDATE Birds SET bird_type = NEW.type WHERE bird_id = NEW.animal_id;
         END IF;
      END IF;
   END IF;
END $$
DELIMITER;
```

Trigger - 5

```
const connection = await mysql.createConnection({
  host: 'localhost',
  user: 'root',
  password: 'Parth@3060',
  database: 'wildlife_sanctuary_up'
});

const bird_sanctuary_check_trigger = `
  CREATE TRIGGER bird_sanctuary_check
  AFTER INSERT, UPDATE ON Animals
  FOR EACH ROW
  BEGIN
      DECLARE animal_type VARCHAR(20);
      DECLARE animal_id INT;
      DECLARE sanctuary_id INT;
      DECLARE bird_id INT;
      DECLARE bird_type VARCHAR(20);

      SELECT type INTO animal_type
      FROM Animals
      WHERE animal_id = NEW.animal_id;

      IF animal_type = 'Bird' THEN
          SELECT sanctuary_id INTO sanctuary_id
          FROM Sanctuaries
          WHERE sanctuary_id = NEW.sanctuary_id;

          IF sanctuary_id IS NULL THEN
              DELETE FROM Animals WHERE animal_id = NEW.animal_id;
              INSERT INTO Birds (bird_id, bird_type) VALUES (NEW.animal_id, 'Unknown');
          ELSE
              SELECT type INTO bird_type
              FROM Birds
              WHERE bird_id = NEW.animal_id;

              IF bird_type = 'Unknown' THEN
                  UPDATE Birds SET bird_type = NEW.type WHERE bird_id = NEW.animal_id;
              END IF;
          END IF;
      END IF;
  END;
`;

await connection.execute(`DELIMITER $$`);
await connection.execute(bird_sanctuary_check_trigger);
await connection.execute(`DELIMITER ;`);
```

References :

Ministry of Environment, Forest and Climate Change. (n.d.). Ministry of Environment, Forest and Climate Change, Government of India. Retrieved April 21, 2023, from **https://moef.gov.in/en/**

International Union for Conservation of Nature. (n.d.). IUCN Red List of Threatened Species. Retrieved April 21, 2023, from **https://www.iucnredlist.org/**

Simplilearn. (n.d.). Node.js MySQL Tutorial: A Beginner's Guide to Node.js MySQL. Retrieved from **https://www.simplilearn.com/tutorials/nodejs-tutorial/nodejs-mysql**