TASKS:

1. Provide a SQL script that initializes the database for the Pet Adoption Platform ”PetPals”.

Query:

```
CREATE DATABASE IF NOT EXISTS petadoptiondatabase;
usepetadoptiondatabaase
```

2. Create tables for pets, shelters, donations, adoption events, and participants.

```
-- Pets Table
CREATE TABLE IF NOT EXISTS Pets (
    PetID INT PRIMARY KEY,
    Name VARCHAR(255),
    Age INT,
    Breed VARCHAR(255),
    Type VARCHAR(50),
    AvailableForAdoption BIT
);

-- Shelters Table
CREATE TABLE IF NOT EXISTS Shelters (
    ShelterID INT PRIMARY KEY,
    Name VARCHAR(255),
    Location VARCHAR(255)
);

-- Donations Table
CREATE TABLE IF NOT EXISTS Donations (
    DonationID INT PRIMARY KEY,
    DonorName VARCHAR(255),
    DonationType VARCHAR(50),
    DonationAmount DECIMAL(10, 2),
    DonationItem VARCHAR(255),
    DonationDate DATETIME
);

-- AdoptionEvents Table
CREATE TABLE IF NOT EXISTS AdoptionEvents (
    EventID INT PRIMARY KEY,
    EventName VARCHAR(255),
    EventDate DATETIME,
    Location VARCHAR(255)
);
```

```
-- Participants Table
CREATE TABLE IF NOT EXIST Participants (
    ParticipantID INT PRIMARY KEY,
    ParticipantName VARCHAR(255),
    ParticipantType VARCHAR(50),
    EventID INT,
    FOREIGN KEY (EventID) REFERENCES AdoptionEvents(EventID)
);
```

3. Define appropriate primary keys, foreign keys, and constraints:

Ans: That was created in the above table

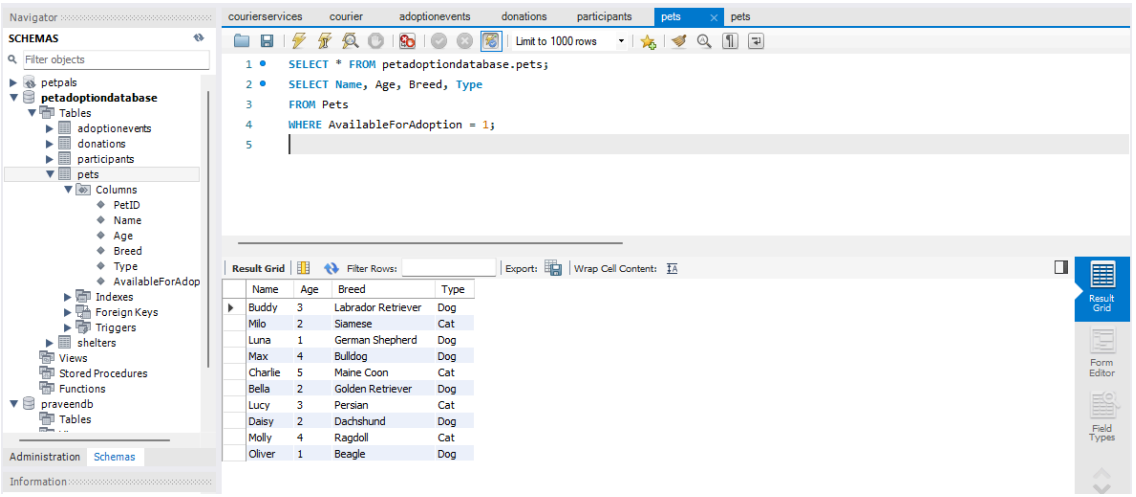
4. Ensure the script handles potential errors, such as if the database or tables already exist.

Ans: IF NOT EXISTS clause is used in the table and database creation.

5.Write an SQL query that retrieves a list of available pets (those marked as available for adoption) from the "Pets" table. Include the pet's name, age, breed, and type in the result set. Ensure that the query filters out pets that are not available for adoption.

query:

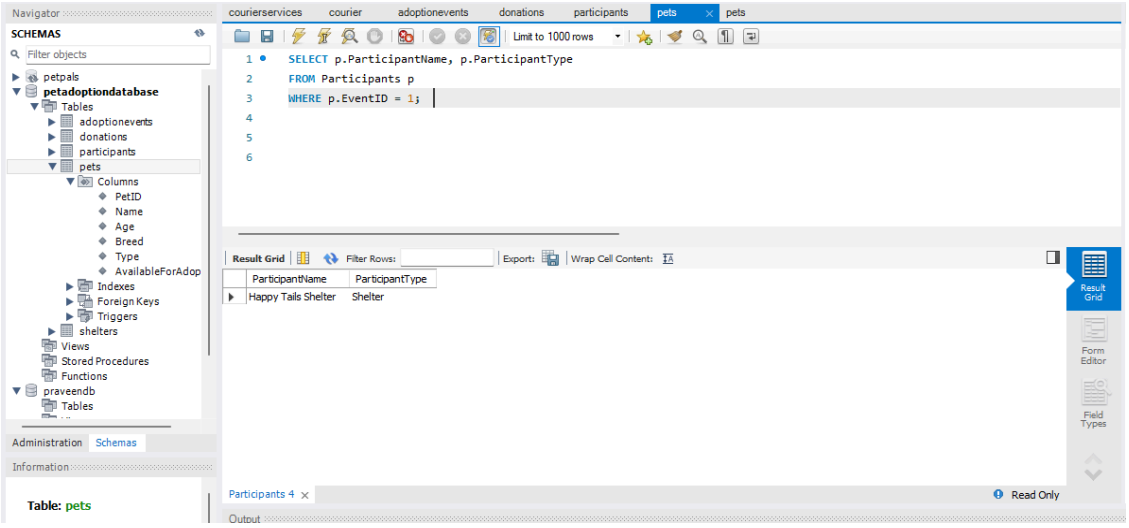
```
SELECT Name, Age, Breed, Type
FROM Pets
WHERE AvailableForAdoption = 1;
```



6.Write an SQL query that retrieves the names of participants (shelters and adopters) registered for a specific adoption event. Use a parameter to specify the event ID. Ensure that the query joins the necessary tables to retrieve the participant names and types

query:

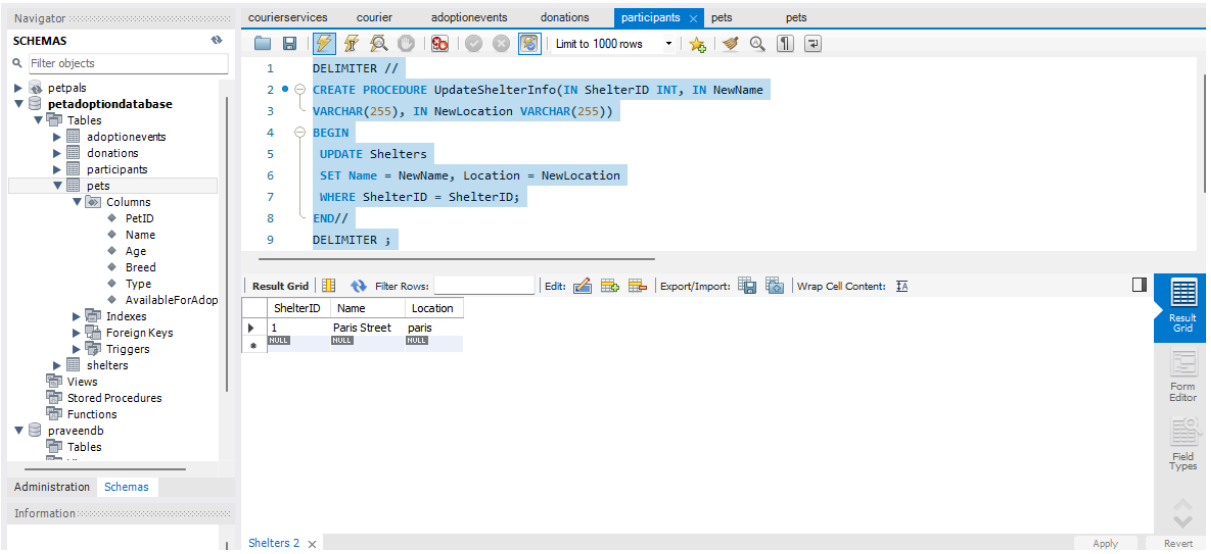
```
SELECT p.ParticipantName, p.ParticipantType
FROM Participants p
WHERE p.EventID = 1;
```



7.Update Shelter Information Procedure

Query:

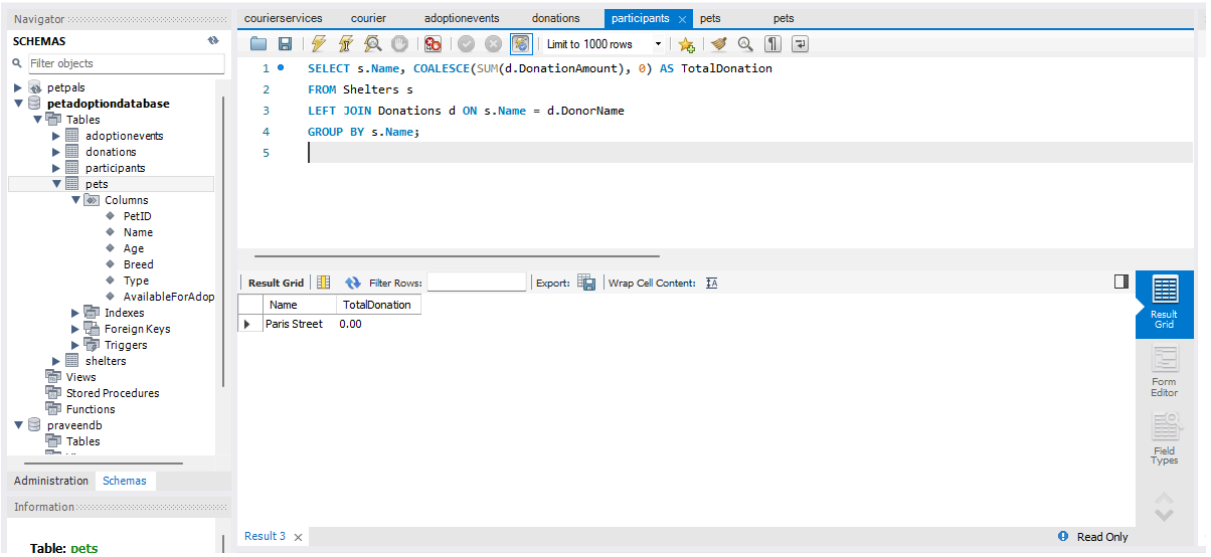
```
DELIMITER //
CREATE PROCEDURE UpdateShelterInfo(IN ShelterID INT, IN
NewName VARCHAR(255), IN NewLocation VARCHAR(255)) BEGIN
UPDATE Shelters SET Name = NewName, Location = NewLocation
WHERE ShelterID = ShelterID; END//
DELIMITER ; CALL UpdateShelterInfo(1, 'Paris Street', 'paris');
SELECT * FROM Shelters WHERE ShelterID = 1; Set
sql_safe_updates=0;
```



8. Total Donation Amount per Shelter

Query:

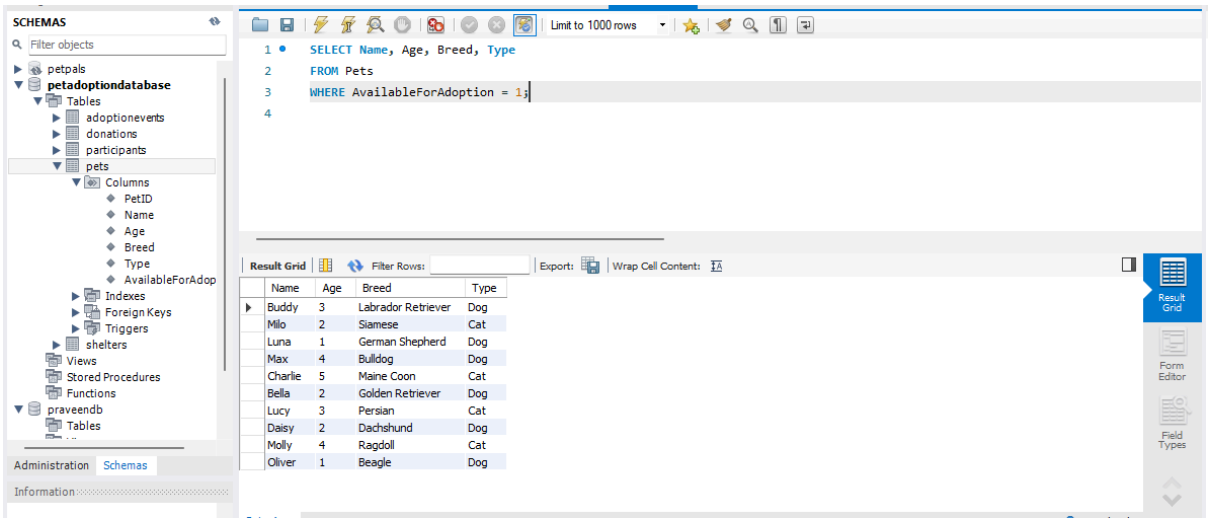
SELECT s.Name, COALESCE(SUM(d.DonationAmount), 0) AS TotalDonation
FROM Shelters s
LEFT JOIN Donations d ON s.ShelterID = d.ShelterID
GROUP BY s.Name;



9. Pets Without Owners

Query:

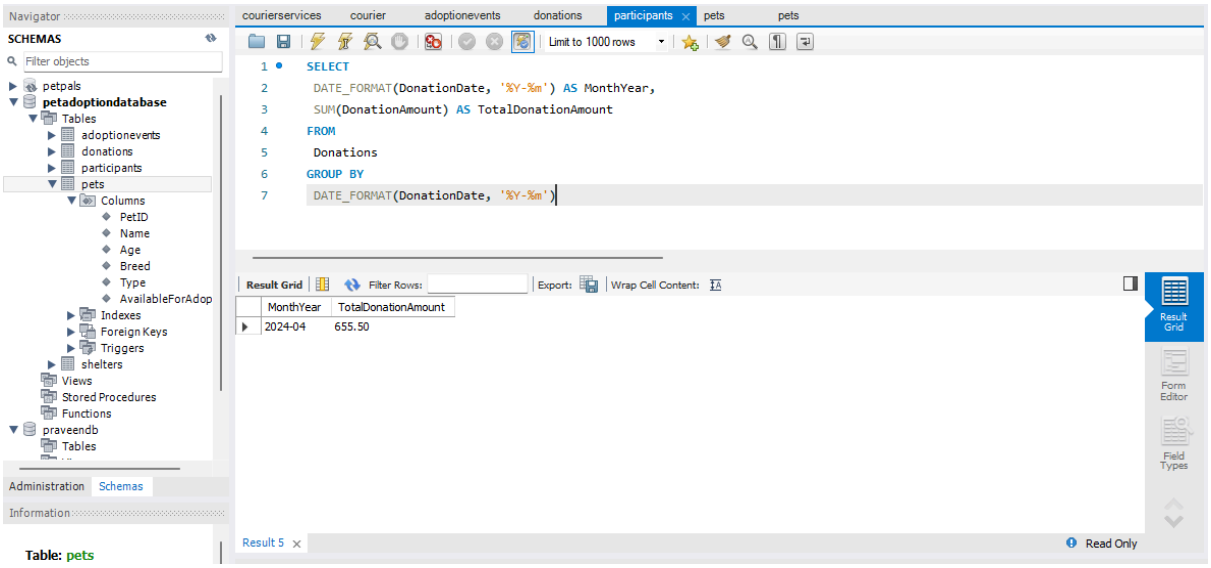
```
SELECT Name, Age, Breed, Type
FROM Pets
WHERE AvailableForAdoption = 1;
```



10. Total Donation per Month-Year

Query:

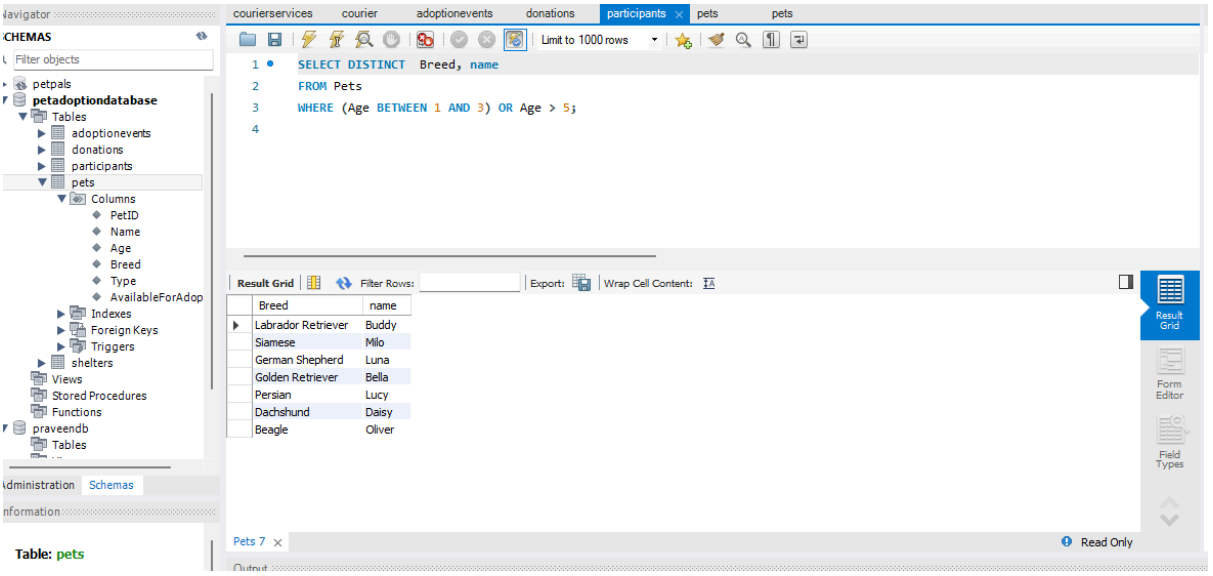
```
SELECT DATE_FORMAT(DonationDate, '%Y-%m') AS MonthYear,
SUM(DonationAmount) AS TotalDonationAmount
FROM Donations
GROUP BY DATE_FORMAT(DonationDate, '%Y-%m')
```



11. Distinct Breeds for Pets Aged 1-3 or >5 Years

Query:

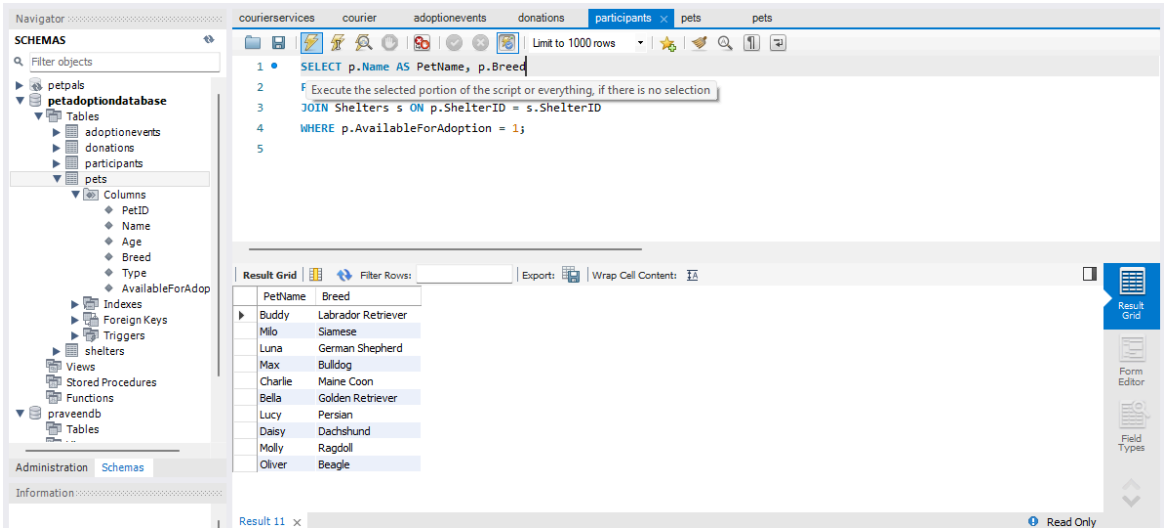
```
SELECT DISTINCT Breed
FROM Pets
WHERE (Age BETWEEN 1 AND 3) OR Age > 5;
```



12. Pets and Their Respective Shelters (Available for Adoption)

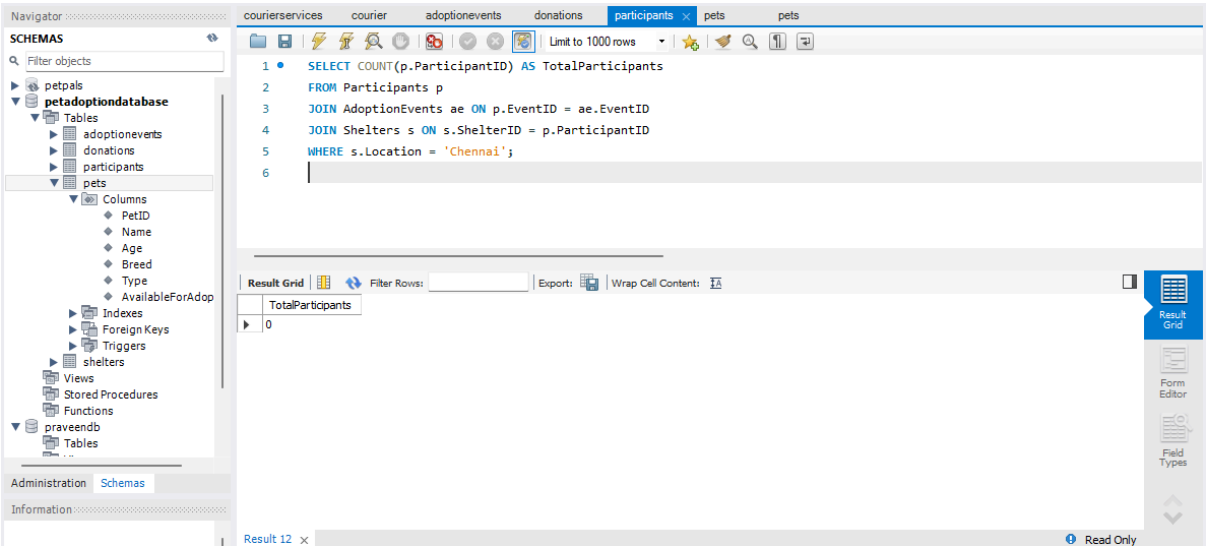
Query:

```
SELECT p.Name AS PetName, p.Breed, s.Name AS ShelterName
FROM Pets p
JOIN Shelters s ON p.ShelterID = s.ShelterID
WHERE p.AvailableForAdoption = 1;
```



13. Total Participants in Events by City
Query:

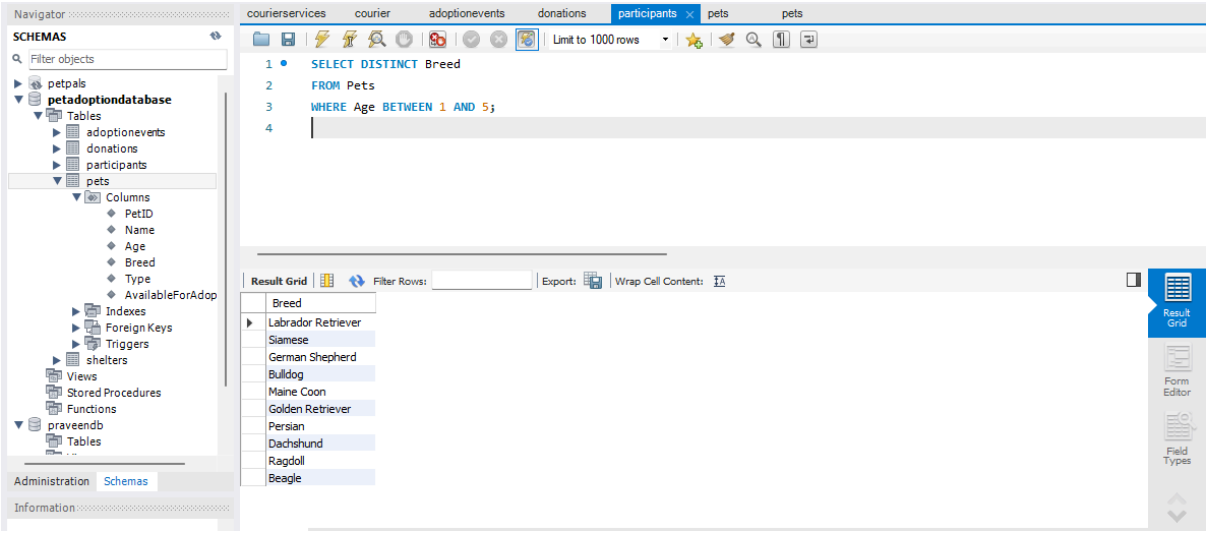
```
SELECT COUNT(p.ParticipantID) AS TotalParticipants
FROM Participants p
JOIN AdoptionEvents ae ON p.EventID = ae.EventID
JOIN Shelters s ON s.ShelterID = p.ParticipantID
WHERE s.Location = 'Chennai';
```



14. Retrieve a list of unique breeds for pets with ages between 1 and 5 years.

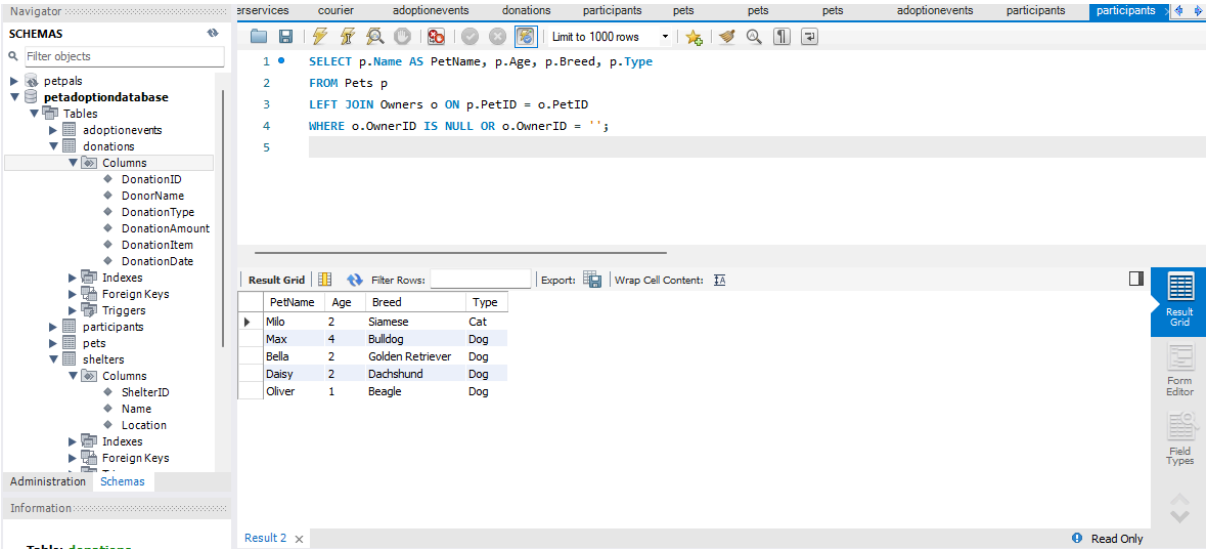
query:

```
SELECT DISTINCT Breed
FROM Pets
WHERE Age BETWEEN 1 AND 5;
```



15. Find the pets that have not been adopted by selecting their information from the 'Pet' table.

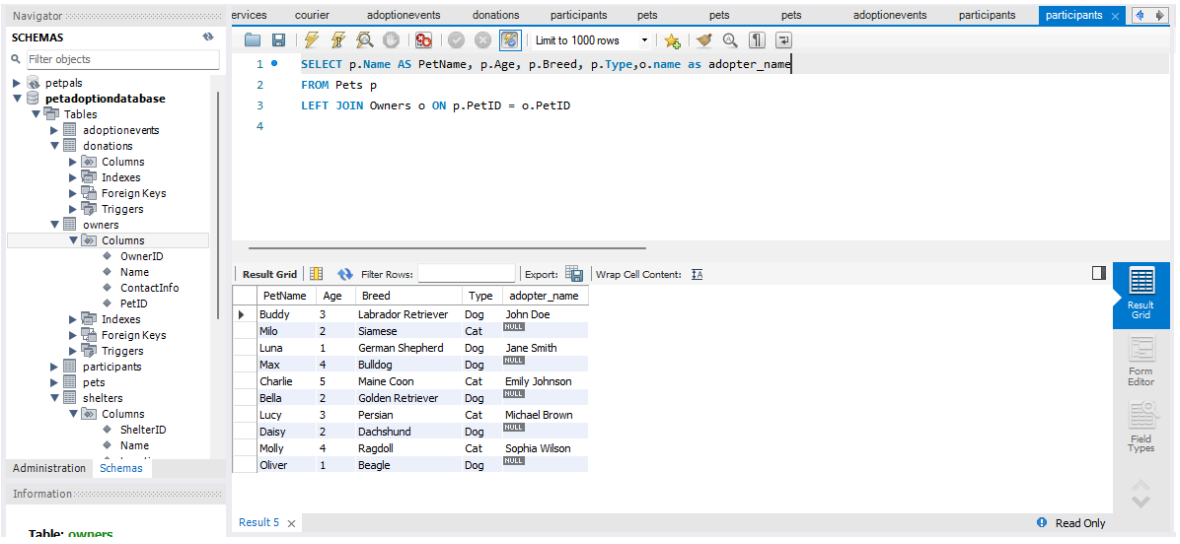
SELECT p.Name AS PetName, p.Age, p.Breed, p.Type
FROM Pets p
LEFT JOIN Owners o ON p.PetID = o.PetID
WHERE o.OwnerID IS NULL OR o.OwnerID = '';



16. Retrieve the names of all adopted pets along with the adopter's name from the 'Adoption' and 'User' tables.

Query:

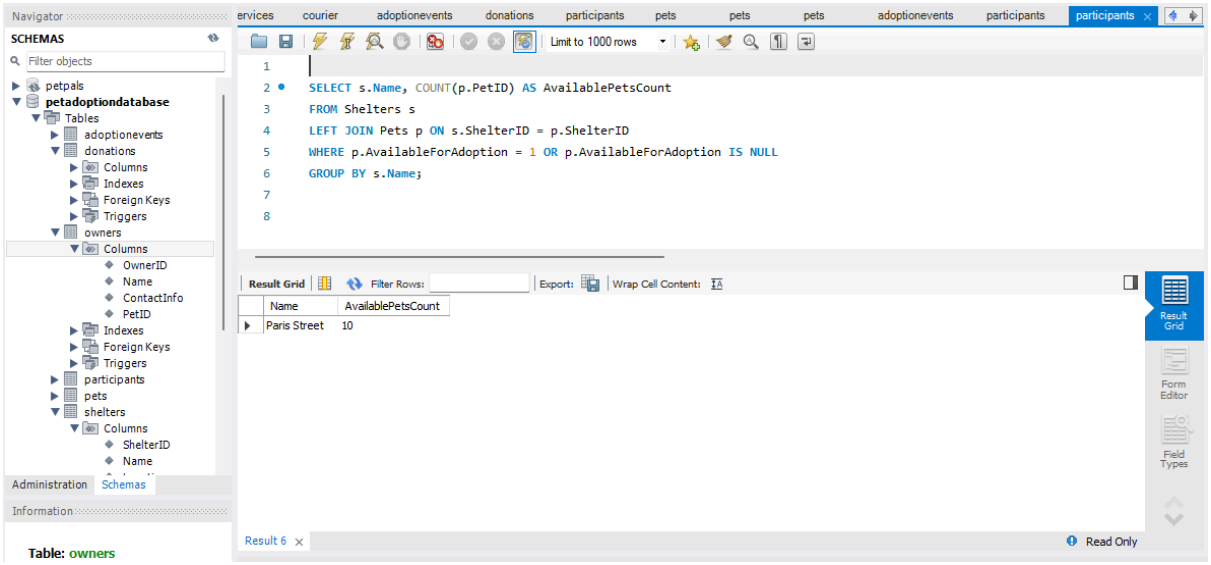
SELECT p.Name AS PetName, p.Age, p.Breed, p.Type,o.name as adopter_name
From Pets p
LEFT JOIN Owners o ON p.PetID = o.PetID



17. Retrieve a list of all shelters along with the count of pets currently available for adoption in each shelter.

Query:

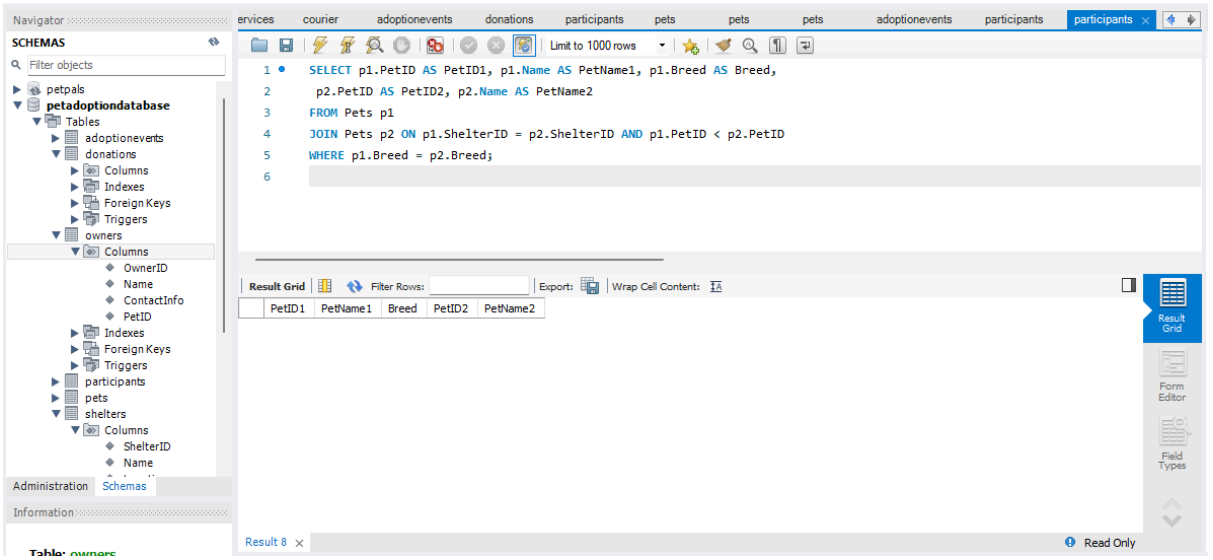
```
SELECT s.Name, COUNT(p.PetID) AS AvailablePetsCount
FROM Shelters s
LEFT JOIN Pets p ON s.ShelterID = p.ShelterID
WHERE p.AvailableForAdoption = 1 OR p.AvailableForAdoption IS NULL
GROUP BY s.Name;
```



18. Find pairs of pets from the same shelter that have the same breed.

Query:

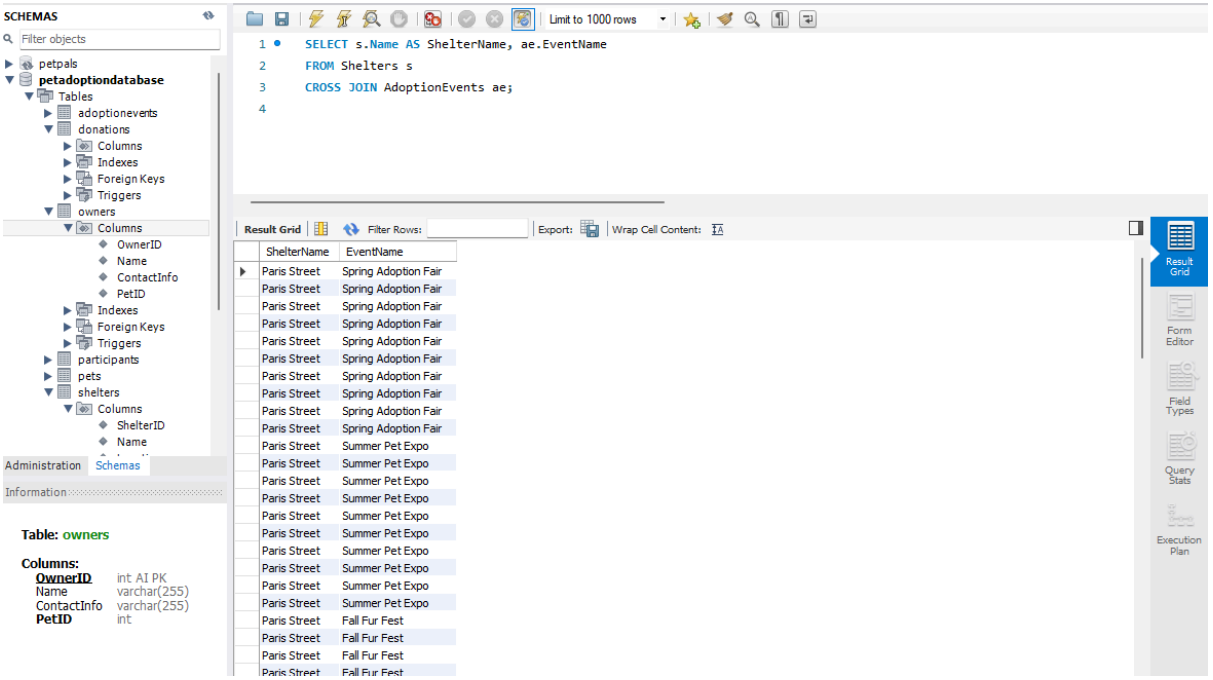
```
SELECT p1.PetID AS PetID1, p1.Name AS PetName1, p1.Breed AS Breed, p2.PetID AS PetID2, p2.Name AS PetName2 FROM Pets p1
JOIN Pets p2 ON p1.ShelterID = p2.ShelterID AND p1.PetID < p2.PetID
WHERE p1.Breed = p2.Breed;
```



19. List all possible combinations of shelters and adoption events.

Query:

```
SELECT s.Name AS ShelterName, ae.EventName
FROM Shelters s
CROSS JOIN AdoptionEvents ae;
```



20. Determine the shelter that has the highest number of adopted pets.

Query:

```
SELECT s.Name, COUNT(*) AS Count
FROM Shelters s
JOIN Pets p ON s.ShelterID = p.ShelterID
WHERE p.AvailableForAdoption = 0
 GROUP BY s.Name
```