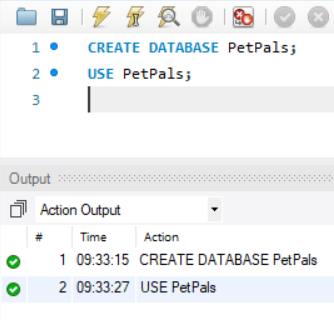
**Coding Challenges - PetPals, The Pet Adoption Platform**

**Tasks**:

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

CREATE DATABASE PetPals;

USE PetPals;



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

CREATE TABLE Pets(

PetID INT PRIMARY KEY auto\_increment,

Name VARCHAR(100) NOT NULL,

Age INT,

Breed VARCHAR(100),

Type VARCHAR(50),

AvailableForAdoption BIT NOT NULL DEFAULT 1

);

CREATE TABLE Shelters(

ShelterID INT PRIMARY KEY auto\_increment,

Name VARCHAR(100) NOT NULL,

Location VARCHAR(250)

);

CREATE TABLE Donations(

DonationID INT PRIMARY KEY auto\_increment,

DonarName VARCHAR(100),

DonationType VARCHAR(50),

DonationAmount DECIMAL(10,2),

DonationItem VARCHAR(100),

DonationDate DATETIME NOT NULL DEFAULT current\_timestamp

);

CREATE TABLE AdoptionEvents(

EventID INT PRIMARY KEY auto\_increment,

EventName VARCHAR(100) NOT NULL,

EventDate DATETIME NOT NULL,

Location VARCHAR(250)

);

CREATE TABLE Participants (

ParticipantID INT PRIMARY KEY AUTO\_INCREMENT,

ParticipantName VARCHAR(100) NOT NULL,

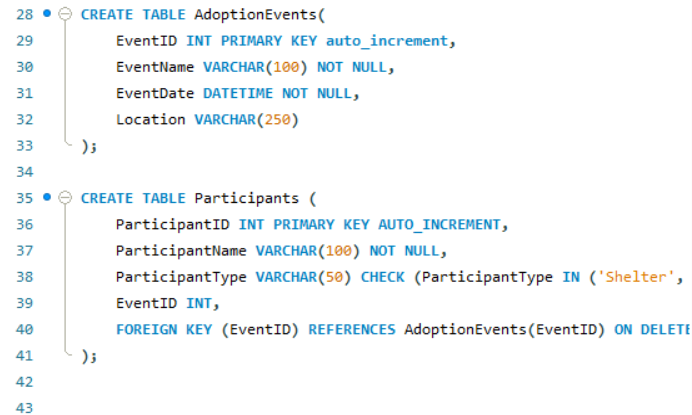
ParticipantType VARCHAR(50) CHECK (ParticipantType IN ('Shelter', 'Adopter')),

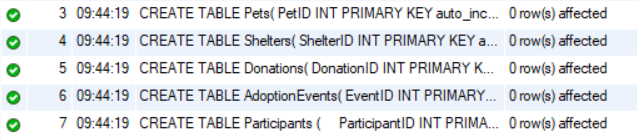
EventID INT,

FOREIGN KEY (EventID) REFERENCES AdoptionEvents(EventID) ON DELETE SET NULL

);







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

ALTER TABLE Donations

ADD CONSTRAINT chk\_DonationType

CHECK (DonationType IN ('Cash', 'Item'));

ALTER TABLE Donations

ADD CONSTRAINT chk\_DonationFields

CHECK (

(DonationType = 'Cash' AND DonationAmount IS NOT NULL AND DonationItem IS NULL) OR

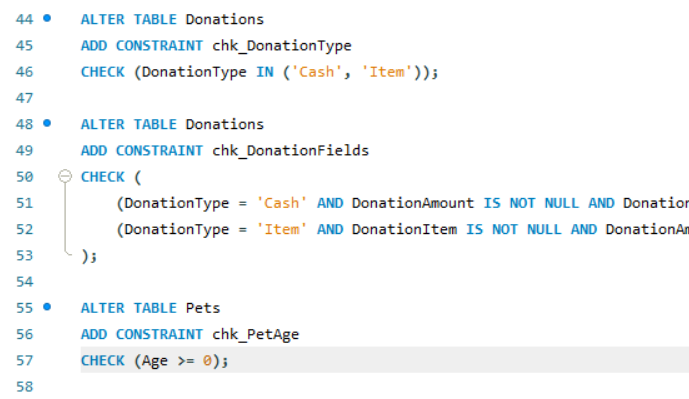
(DonationType = 'Item' AND DonationItem IS NOT NULL AND DonationAmount IS NULL)

);

ALTER TABLE Pets

ADD CONSTRAINT chk\_PetAge

CHECK (Age >= 0);



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

CREATE DATABASE IF NOT EXISTS PetPals;

USE PetPals;

DROP TABLE IF EXISTS Participants;

DROP TABLE IF EXISTS AdoptionEvents;

DROP TABLE IF EXISTS Donations;

DROP TABLE IF EXISTS Pets;

DROP TABLE IF EXISTS Shelters;

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.

SELECT

Name,

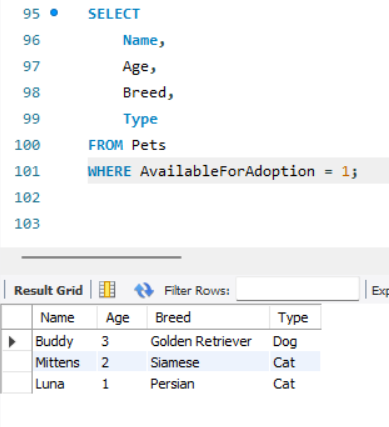
Age,

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.

SELECT

P.ParticipantName,

P.ParticipantType

FROM

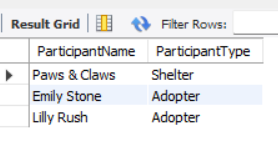
Participants P

JOIN

AdoptionEvents AE ON P.EventID = AE.EventID

WHERE

P.EventID = 2;



7. Create a stored procedure in SQL that allows a shelter to update its information (name and

location) in the "Shelters" table. Use parameters to pass the shelter ID and the new information. Ensure that the procedure performs the update and handles potential errors, such as an invalid shelter ID.

DELIMITER //

CREATE PROCEDURE UpdateShelterInfo(

IN shelterId INT,

IN newName VARCHAR(100),

IN newLocation VARCHAR(250)

)

BEGIN

IF EXISTS (SELECT \* FROM Shelters WHERE ShelterID = shelterId) THEN

UPDATE Shelters

SET Name = newName,

Location = newLocation

WHERE ShelterID = shelterId;

ELSE

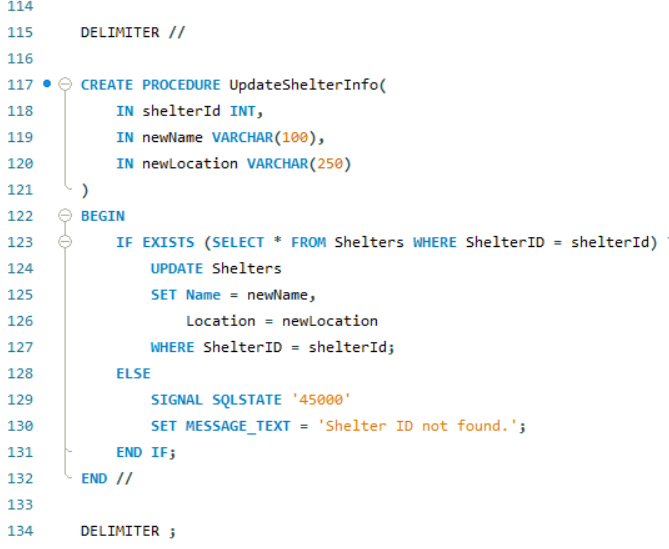
SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = 'Shelter ID not found.';

END IF;

END //

DELIMITER ;



8. Write an SQL query that calculates and retrieves the total donation amount for each shelter (by shelter name) from the "Donations" table. The result should include the shelter name and the total donation amount. Ensure that the query handles cases where a shelter has received no donations.

SELECT

S.Name AS ShelterName,

IFNULL(SUM(D.DonationAmount), 0) AS TotalDonations

FROM

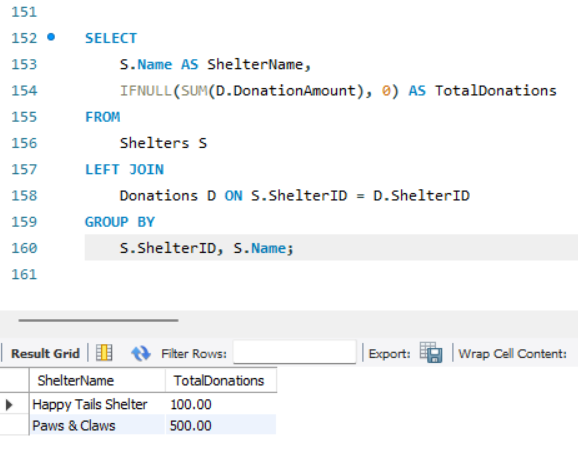
Shelters S

LEFT JOIN

Donations D ON S.ShelterID = D.ShelterID

GROUP BY

S.ShelterID, [S.Name](http://s.name);



9. Write an SQL query that retrieves the names of pets from the "Pets" table that do not have an owner (i.e., where "OwnerID" is null). Include the pet's name, age, breed, and type in the result set.

SELECT

Name,

Age,

Breed,

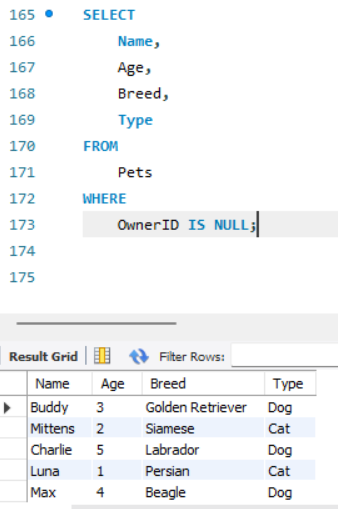
Type

FROM

Pets

WHERE

OwnerID IS NULL;



10. Write an SQL query that retrieves the total donation amount for each month and year (e.g., January 2023) from the "Donations" table. The result should include the month-year and the corresponding total donation amount. Ensure that the query handles cases where no donations were made in a specific month-year.

SELECT

DATE\_FORMAT(MonthStart, '%M %Y') AS MonthYear,

TotalDonations

FROM (

SELECT

DATE\_FORMAT(DonationDate, '%Y-%m-01') AS MonthStart,

SUM(DonationAmount) AS TotalDonations

FROM

Donations

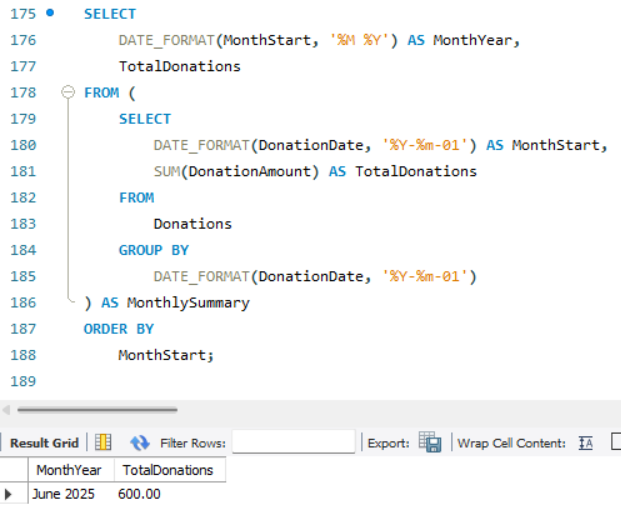
GROUP BY

DATE\_FORMAT(DonationDate, '%Y-%m-01')

) AS MonthlySummary

ORDER BY

MonthStart;

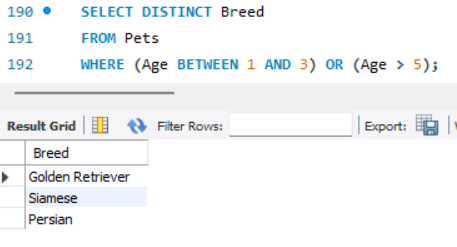


11. Retrieve a list of distinct breeds for all pets that are either aged between 1 and 3 years or older than 5 years.

SELECT DISTINCT Breed

FROM Pets

WHERE (Age BETWEEN 1 AND 3) OR (Age > 5);



12. Retrieve a list of pets and their respective shelters where the pets are currently available for adoption.

SELECT

P.Name AS PetName,

P.Breed,

P.Type,

S.Name AS ShelterName

FROM

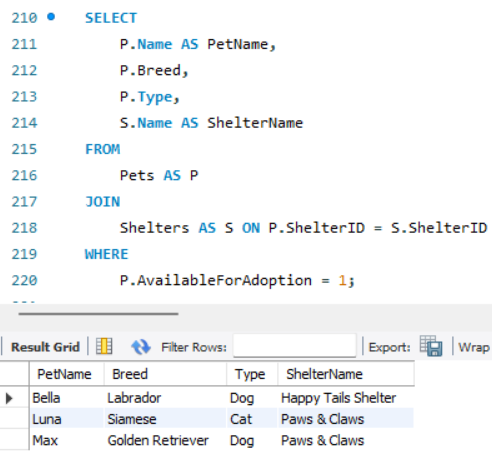
Pets AS P

JOIN

Shelters AS S ON P.ShelterID = S.ShelterID

WHERE

P.AvailableForAdoption = 1;



13. Find the total number of participants in events organized by shelters located in specific city. Example: City=Chennai

SELECT

COUNT(\*) AS TotalParticipants

FROM

Participants P

JOIN

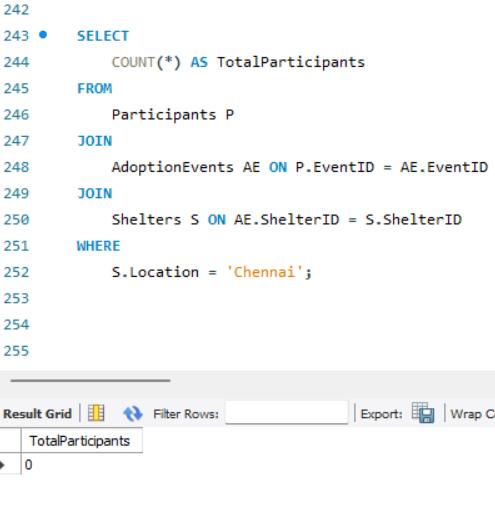
AdoptionEvents AE ON P.EventID = AE.EventID

JOIN

Shelters S ON AE.ShelterID = S.ShelterID

WHERE

S.Location = 'Chennai';

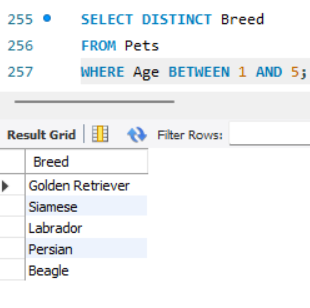


14. Retrieve a list of unique breeds for pets with ages between 1 and 5 years.

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

PetID,

Name,

Age,

Breed,

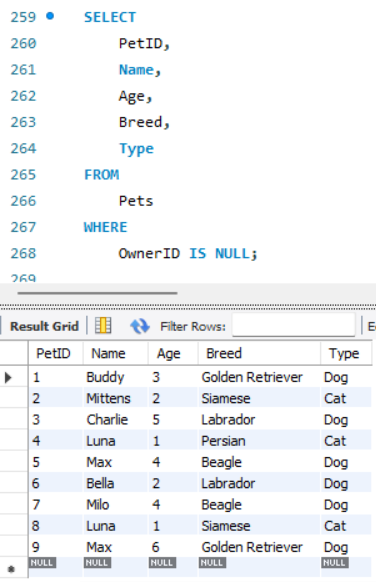
Type

FROM

Pets

WHERE

OwnerID IS NULL;



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

SELECT

P.Name AS PetName,

U.UserName AS AdopterName

FROM

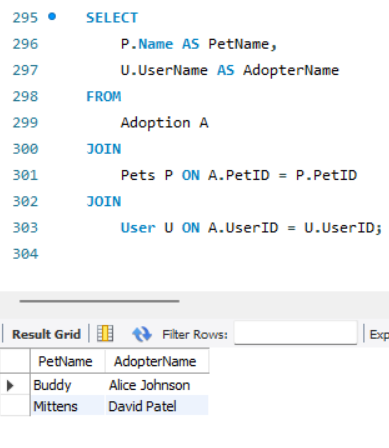
Adoption A

JOIN

Pets P ON A.PetID = P.PetID

JOIN

User U ON A.UserID = U.UserID;



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

SELECT

S.Name AS ShelterName,

COUNT(P.PetID) AS AvailablePets

FROM

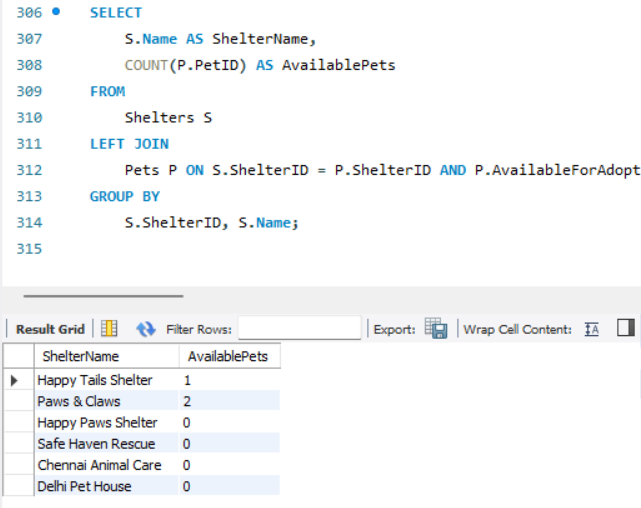
Shelters S

LEFT JOIN

Pets P ON S.ShelterID = P.ShelterID AND P.AvailableForAdoption = 1

GROUP BY

S.ShelterID, [S.Name](http://s.name);



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

SELECT

P1.PetID AS Pet1\_ID,

P1.Name AS Pet1\_Name,

P2.PetID AS Pet2\_ID,

P2.Name AS Pet2\_Name,

P1.Breed,

S.Name AS ShelterName

FROM

Pets P1

JOIN

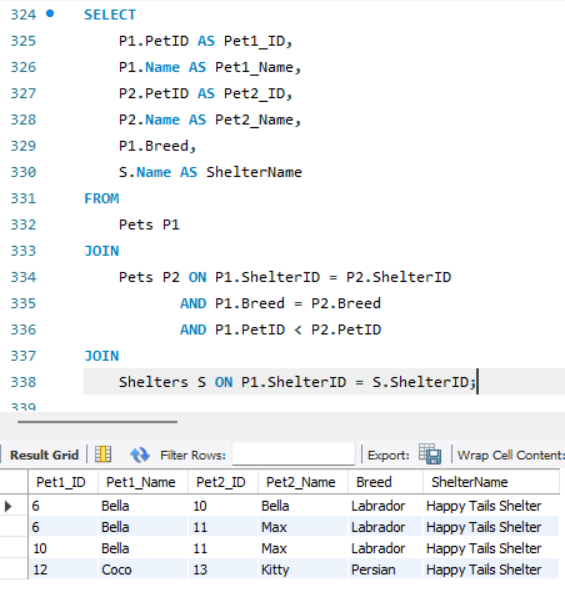
Pets P2 ON P1.ShelterID = P2.ShelterID

AND P1.Breed = P2.Breed

AND P1.PetID < P2.PetID

JOIN

Shelters S ON P1.ShelterID = S.ShelterID;



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

SELECT

S.Name AS ShelterName,

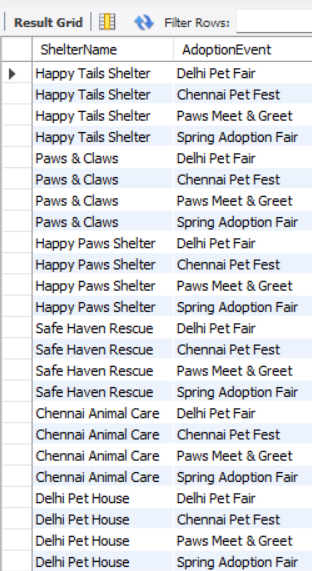
AE.EventName AS AdoptionEvent

FROM

Shelters S

CROSS JOIN

AdoptionEvents AE;



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

SELECT

S.Name AS ShelterName,

COUNT(A.AdoptionID) AS AdoptedPetCount

FROM

Adoption A

JOIN

Pets P ON A.PetID = P.PetID

JOIN

Shelters S ON P.ShelterID = S.ShelterID

GROUP BY

S.ShelterID, S.Name

ORDER BY

AdoptedPetCount DESC

LIMIT 1;

