Coding Challenges - PetPals, The Pet Adoption Platform

J1701-Deva R

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 shelters(
shelterId int primary key auto_increment,
   name varchar(50) not null,
   location varchar(150) not null
   );


create table pets(
petId int primary key auto_increment,
   name varchar(50) not null,
   age int not null,
   breed varchar(50) not null,
   type varchar(50) not null,
   availableForAdoption bit not null,
   shelterId int,
   foreign key(shelterId) references shelters(shelterId) on delete cascade
   );

   create table donations (
   donationId int primary key auto_increment,
   shelterId int,
   donorname varchar(255) not null,
   donationtype enum('cash', 'item') not null,
   donationamount decimal(10,2) default null,
   donationitem varchar(255) default null,
   donationdate datetime not null,
   foreign key (shelterId) references shelters(shelterId) on delete cascade
);

create table adoptionevents (
   eventid int primary key auto_increment,
   eventname varchar(50) not null,
   eventdate datetime not null,
   location varchar(145) not null
);
```

```
create table participants (
  participantid int primary key auto_increment,
  participantname varchar(255) not null,
  participanttype enum('shelter', 'adopter') not null,
  eventid int,
  foreign key (eventid) references adoptionevents(eventid) on delete cascade
);
```

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

```
participant_id INT PRIMARY KEY
event_id INT PRIMARY KEY AUTO
pet_id INT PRIMARY KEY
FOREIGN KEY (event_id) REFERENCES adoption_events(event_id)
donation_id INT PRIMARY KEY
FOREIGN KEY (shelter_id) REFERENCES shelters(shelter_id)
```

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

```
create database if not exists petpals;
```

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 * from pets;

insert into shelters (name, location) values
('happy shelter', 'chennai'),
('safe shelter', 'bangalore'),
('home shelter', 'hyderbad');

insert into pets (name, age, breed, type, availableforadoption, shelterid) values
('bella', 2, 'labrador', 'dog', 1, 1),
('max', 3, 'beagle', 'dog', 1, 2),
('luna', 1, 'persian', 'cat', 1, 3),
('charlie', 5, 'golden retriever', 'dog', 0, 1),
('milo', 4, 'shitzhu', 'cat', 1, 2),
('daisy', 2, 'parrot', 'bird', 1, 3);

select * from pets;

insert into donations (shelterid, donorname, donationtype, donationamount, donationitem,
donationdate) values
(1, 'alice johnson', 'cash', 500.00, null, '2025-03-01 10:30:00'),
(2, 'bob smith', 'item', null, 'dog food', '2025-03-02 12:00:00'),
(3, 'charlie davis', 'cash', 250.00, null, '2025-03-05 14:20:00'),
(1, 'diana green', 'item', null, 'cat toys', '2025-03-07 09:10:00');


insert into adoptionevents (eventname, eventdate, location) values
```

('spring adoption', '2025-04-10 11:00:00', 'new york central park'),
('puppy love day', '2025-05-15 10:00:00', 'bangalore community center'),
('friends meetup', '2025-06-20 12:30:00', 'chennai pet plaza');

insert into participants (participantname, participanttype, eventid) values
('happy shelter', 'shelter', 1),
('safe shelter', 'shelter', 2),
('Deva', 'adopter', 1),
('SriGanesh', 'adopter', 2),
('Sathish', 'adopter', 3);

-- 5th qstn

select name, age, breed, type from pets where availableforadoption = 1;

select participantname, participanttype from participants
where eventid = 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 participantname, participanttype from participants
where eventid = 1;

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.

update shelters set name='new home shelter', location='coimbatore' where shelterid =3;

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 shelternname, sum(d.donationamount) as totaldonation
from shelters s
left join donations d on s.shelterid = d.shelterid
group by s.shelterid, 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;

alter table pets add column ownerId int default null,
add foreign key(ownerId) references participants(participantId) on delete cascade;
```

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(donationdate, '%y-%m') as monthyear, sum(donationamount) as
totaldonation
from donations
group by year(donationdate), month(donationdate)
order by year(donationdate), month(donationdate);
```

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.age, p.breed, p.type, s.name as shelternname
from pets p
join shelters 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(pa.participantid) as totalparticipants
from participants pa
join adoptionevents ae on pa.eventid = ae.eventid
join shelters s on ae.location = s.location
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.

```
create table adoption (
  adoptionid int primary key auto_increment,
```

```
    petid int unique not null,
    adopterid int not null,
    adoptiondate datetime not null default current_timestamp,
    foreign key (petid) references pets(petid) on delete cascade,
    foreign key (adopterid) references participants(participantid) on delete cascade
);


insert into adoption (petid, adopterid, adoptiondate) values
(4, 1, '2024-04-12 13:00:00'),
(5, 2, '2024-05-18 14:30:00');


select p.petid, p.name as petname, p.age, p.breed, p.type
from pets p
left join adoption a on p.petid = a.petid
where a.petid is null;
```

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

```
create table users (
  userid int primary key auto_increment,
  name varchar(50) not null,
  email varchar(55) unique,
  phonenumber varchar(20)
);

insert into users (userid, name, email, phonenumber) values
(1, 'john doe', 'johndoe@email.com', '9876543210'),
(2, 'alice smith', 'alicesmith@email.com', '8765432109');



select p.name as petname, u.name as adoptername
from adoption a
join pets p on a.petid = p.petid
join users u on a.adopterid = u.userid;
```

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

```
select s.shelterid, 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
order by availablepets desc;
```

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, p1.shelterid
from pets p1
join pets p2 on p1.shelterid = p2.shelterid
and p1.breed = p2.breed
and p1.petid < p2.petid
order by p1.shelterid, p1.breed;
```

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

```
    select s.shelterid, s.name as shelternname, e.eventid, e.eventname, e.eventdate
    from shelters s
    cross join adoptionevents e
order by s.shelterid, e.eventdate;
```

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

```
select s.shelterid, s.name as shelternname, count(a.petid) as adoptedpets
from shelters s
join pets p on s.shelterid = p.shelterid
join adoption a on p.petid = a.petid
group by s.shelterid, s.name
order by adoptedpets desc;
```