**Create tables in database**

|  |  |
| --- | --- |
| User | CREATE TABLE User(  Username varchar(50),  Email varchar(50) NOT NULL,  Password char(32) NOT NULL,  UserType enum('ADMIN', 'OWNER', 'VISITOR') NOT NULL,  PRIMARY KEY(Username),  UNIQUE(Email)  ) ENGINE = InnoDB; |
| Property | CREATE TABLE Property(  ID int CHECK (ID < 100,000),  Name varchar(255) NOT NULL,  Size float NOT NULL,  IsCommercial boolean NOT NULL,  IsPublic boolean NOT NULL,  Street varchar(255) NOT NULL,  City varchar(100) NOT NULL,  Zip int NOT NULL,  PropertyType enum('FARM', 'GARDEN', 'ORCHARD') NOT NULL,  Owner varchar(50) NOT NULL,  ApprovedBy varchar(50),  PRIMARY KEY(ID),  UNIQUE(Name),  FOREIGN KEY (Owner) REFERENCES User(Username)  ON DELETE CASCADE ON UPDATE CASCADE,  FOREIGN KEY(ApprovedBy) REFERENCES User(Username)  ON DELETE SET NULL ON UPDATE CASCADE,  ) ENGINE = InnoDB; |
| Visit | CREATE TABLE Visit(  Username varchar(50),  PropertyID int,  VisitDate timestamp NOT NULL,  Rating int NOT NULL CHECK (Rating >= 1 AND Raing <= 5),  CONSTRAINT Pk\_Visit PRIMARY KEY (Username, PropertyID),  FOREIGN KEY (Username) REFERENCES User(Username)  ON DELETE CASCADE ON UPDATE CASCADE,  FOREIGN KEY (PropertyID) REFERENCES Property(ID)  ON DELETE CASCADE ON UPDATE CASCADE,  ) ENGINE = InnoDB; |
| Has | CREATE TABLE Has(  PropertyID int,  ItemName varchar(100),  CONSTRAINT Pk\_Has PRIMARY KEY(PropertyID, ItemName),  FOREIGN KEY (PropertyID) REFERENCES Property(ID)  ON DELETE CASCADE ON UPDATE CASCADE,  FOREIGN KEY (ItemName) REFERENCES FarmItem(Name)  ON DELETE CASCADE ON UPDATE CASCADE  ) ENGINE = InnoDB; |
| FarmItem | CREATE TABLE FarmItem(  Name varchar(100),  IsApproved boolean NOT NULL,  Type enum('ANIMAL', 'FRUIT', 'FLOWER', 'VEGETABLE', 'NUT') NOT NULL,  PRIMARY KEY(Name)  ) ENGINE = InnoDB; |

**Login page**

|  |  |
| --- | --- |
| verify valid credentials | SELECT \*  FROM User  WHERE Username = $username AND Password = $password; |

**New visitor registration**

|  |  |
| --- | --- |
| verify registration is valid | SELECT \*  FROM User  WHERE Username = $username OR Email = $email; |
| insert new visitor account | INSERT INTO User  VALUES ($username, $email, $password, 'VISITOR'); |

**New owner registration**

|  |  |
| --- | --- |
| check that owner does not exist | SELECT \*  FROM User  WHERE Username = $username AND Email = $email; |
| populate animal dropdown | SELECT \*  FROM FarmItem  WHERE Type = 'ANIMAL' AND IsApproved = 1; |
| populate crop dropdown | SELECT \*  FROM FarmItem  WHERE Type != 'ANIMAL' AND IsApproved = 1; |
| add new owner | INSERT INTO User  VALUES ($username, $email, $password, 'OWNER'); |
| find max property id | SELECT MAX(ID) FROM Property |
| add new property | INSERT INTO Property  VALUES ($propertyid, $name, $size, $iscommercial, $ispublic, $street, $city, $zip, $propertytype, $owner, NULL); |
| add initial crop/animal | INSERT INTO Has  VALUES ($propertyid, $itemName); |

**Owner home**

|  |  |
| --- | --- |
| default view of owners properties | SELECT Name,  Street AS Address,  City,  Zip,  Size,  PropertyType AS Type,  (CASE WHEN IsPublic =1 THEN 'True' ELSE 'False' END) AS Public,  (CASE WHEN IsCommercial =1 THEN 'True' ELSE 'False' END) AS Commercial,  ID,  (CASE WHEN ApprovedBy IS NULL THEN 'False' ELSE 'True' END) AS isValid,  COUNT( \* ) as Visits,  AVG( Rating) as Avg.Rating  FROM Property JOIN Visit ON Visit.PropertyID = Property.ID  WHERE Owner = $owner  GROUP BY ID; |
| search by term filter | SELECT Name,  Street AS Address,  City,  Zip,  Size,  PropertyType AS Type,  (CASE WHEN IsPublic =1 THEN 'True' ELSE 'False' END) AS Public,  (CASE WHEN IsCommercial =1 THEN 'True' ELSE 'False' END) AS Commercial,  ID,  (CASE WHEN ApprovedBy IS NULL THEN 'False' ELSE 'True' END) AS isValid,  COUNT( \* ) as Visits,  AVG( Rating) as Avg.Rating  FROM Property JOIN Visit ON Visit.PropertyID = Property.ID  WHERE $searchby = $search  GROUP BY ID; |

**View other owner's properties**

|  |  |
| --- | --- |
| get all other valid properties (not including current logged in owner) | SELECT Name,  Street AS Address,  City,  Zip,  Size,  PropertyType AS Type,  (CASE WHEN IsPublic =1 THEN 'True' ELSE 'False' END) AS Public,  (CASE WHEN IsCommercial =1 THEN 'True' ELSE 'False' END) AS Commercial,  ID,  (CASE WHEN ApprovedBy IS NULL THEN 'False' ELSE 'True' END) AS isValid,  COUNT( \* ) as Visits,  AVG( Rating) as Avg.Rating  FROM Property JOIN Visit ON Visit.PropertyID = Property.ID  WHERE Owner != $owner  GROUP BY ID; |
| search by term filter | SELECT Name,  Street AS Address,  City,  Zip,  Size,  PropertyType AS Type,  (CASE WHEN IsPublic =1 THEN 'True' ELSE 'False' END) AS Public,  (CASE WHEN IsCommercial =1 THEN 'True' ELSE 'False' END) AS Commercial,  ID,  (CASE WHEN ApprovedBy IS NULL THEN 'False' ELSE 'True' END) AS isValid,  COUNT( \* ) as Visits,  AVG( Rating) as Avg.Rating  FROM Property JOIN Visit ON Visit.PropertyID = Property.ID  WHERE Owner != $owner  GROUP BY ID; |
| view property details | SELECT P . \* ,  FarmItem.Name as Item,  (CASE WHEN FarmItem.Type = 'ANIMAL' THEN 'Animals' ELSE 'Crops' END) as Type  FROM (SELECT Property.Name,  Property.Owner,  Email AS 'Owner Email',  Street AS Address,  City,  Zip,  Size AS 'Size (acres)',  AVG( Rating ) as Avg.Rating,  PropertyType AS Type ,  (CASE WHEN IsPublic =1 THEN 'True' ELSE 'False' END) AS Public,  (CASE WHEN IsCommercial =1 THEN 'True' ELSE 'False' END) AS Commercial,  Property.ID AS ID  FROM Property  JOIN User ON Property.Owner = User.Username  JOIN Has ON Property.ID = Has.PropertyID  JOIN FarmItem ON FarmItem.Name = Has.ItemName  JOIN Visit ON Visit.PropertyID = Property.ID  WHERE Property.ID =$id  ) AS P  JOIN Has ON Has.PropertyID = P.ID  JOIN FarmItem ON FarmItem.Name = Has.ItemName |

**Manage properties for owners**

|  |  |
| --- | --- |
| initial data population of screen | SELECT  Name,  Street as Address,  City,  Zip,  Size,  PropertyType as Type,  (CASE WHEN IsPublic =1 THEN 'True' ELSE 'False' END) AS Public,  (CASE WHEN IsCommercial =1 THEN 'True' ELSE 'False' END) AS Commercial,  ID  FROM Property  WHERE Property.Owner = $owner and Property.ID = $id |
| check if new property name doesnt exist yet | SELECT \*  FROM Property  WHERE ID != $id AND Name = $name |
| get list of crops that this property grows | SELECT ItemName  FROM Has  JOIN FarmItem ON FarmItem.Name = Has.ItemName  WHERE Has.PropertyID = $id AND FarmItem.Type != 'ANIMAL' |
| get list of animals that this farm raises | SELECT ItemName  FROM Has  JOIN FarmItem ON FarmItem.Name = Has.ItemName  WHERE Has.PropertyID = $id AND FarmItem.Type = 'ANIMAL' |
| get list of new crops that can be added to the property | SELECT Name  FROM FarmItem  WHERE FarmItem.Type != 'ANIMAL'  AND FarmItem.IsApproved = True  AND NOT EXISTS (  SELECT \*  FROM Has  WHERE Has.ItemName = FarmItem.Name  AND Has.PropertyID =$id); |
| get list of previous crops to check if it already exists | SELECT Name  FROM FarmItem  WHERE FarmItem.Name = $na |
| add new crop/animal to farmitem to be reviewed | INSERT INTO FarmItem VALUES ($name, False, $type); |
| delete Property | DELETE FROM Property  WHERE ID = $id |
| update property | UPDATE Property  SET Name = $name, Size = $size, IsCommercial = $commercial, IsPublic = $public, Street = $street,City = $city, Zip = $zip  WHERE ID = $id |
| update new crop/animal (approved) | INSERT INTO Has VALUES ($propertyid, $itemname) |
| Delete crop or animal | DELETE FROM Has WHERE ItemName = $itemname AND PropertyID = $id |

**Add property for owner**

|  |  |
| --- | --- |
| check that new property name doesnt already exist | SELECT \* FROM Property WHERE Name = $name; |
| find largest property id | SELECT MAX(ID) FROM Property |
| Add new farm property | INSERT INTO Property VALUES ($id, $name, $size, $commerical, $public, $street, $city, $zip, $propertytype, $ownerusername, NULL);  INSERT INTO Has VALUES ($propertyid, $animal);  INSERT INTO Has Values ($propertyid, $crop); |
| Add new garden/orchard property | INSERT INTO Property VALUES ($id, $name, $size, $commerical, $public, $street, $city, $zip, $propertytype, $ownerusername, NULL);  INSERT INTO Has Values ($propertyid, $crop); |

**Visitor home**

|  |  |
| --- | --- |
| initial public, validated properties table population | SELECT  Name,  Street AS Address,  City,  Zip,  Size,  PropertyType AS Type ,  (CASE WHEN IsPublic =1 THEN 'True' ELSE 'False' END) AS Public,  (CASE WHEN IsCommercial =1 THEN 'True' ELSE 'False' END) AS Commercial,  ID,  COUNT( \* ) AS Visits,  AVG( Rating ) AS 'Avg. Rating'  FROM Property  JOIN Visit ON Visit.PropertyID = Property.ID  WHERE Property.IsPublic = 1  AND Property.ApprovedBy IS NOT NULL  GROUP BY Property.ID |
| search by term filter if its just 2 parameters | SELECT  Name,  Street AS Address,  City,  Zip,  Size,  PropertyType AS Type ,  (CASE WHEN IsPublic =1 THEN 'True' ELSE 'False' END) AS Public,  (CASE WHEN IsCommercial =1 THEN 'True' ELSE 'False' END) AS Commercial,  ID,  COUNT( \* ) AS Visits,  AVG( Rating ) AS 'Avg. Rating'  FROM Property  JOIN Visit ON Visit.PropertyID = Property.ID  WHERE Property.IsPublic = 1  AND Property.ApprovedBy IS NOT NULL  AND $searchby = $search  GROUP BY Property.ID |
| search by term filter if user passes in a range for Visits | SELECT  Name,  Street AS Address,  City,  Zip,  Size,  PropertyType AS Type ,  (CASE WHEN IsPublic =1 THEN 'True' ELSE 'False' END) AS Public,  (CASE WHEN IsCommercial =1 THEN 'True' ELSE 'False' END) AS Commercial,  ID,  COUNT( \* ) AS Visits,  AVG( Rating ) AS 'Avg. Rating'  FROM Property  JOIN Visit ON Visit.PropertyID = Property.ID  WHERE Property.IsPublic = 1  AND Property.ApprovedBy IS NOT NULL  GROUP BY Property.ID  HAVING COUNT(\*) BETWEEN $min AND $max |
| search by term filter if user passes in a range for avg rating | SELECT  Name,  Street AS Address,  City,  Zip,  Size,  PropertyType AS Type ,  (CASE WHEN IsPublic =1 THEN 'True' ELSE 'False' END) AS Public,  (CASE WHEN IsCommercial =1 THEN 'True' ELSE 'False' END) AS Commercial,  ID,  COUNT( \* ) AS Visits,  AVG( Rating ) AS 'Avg. Rating'  FROM Property  JOIN Visit ON Visit.PropertyID = Property.ID  WHERE Property.IsPublic = 1  AND Property.ApprovedBy IS NOT NULL  GROUP BY Property.ID  HAVING AVG(Rating) BETWEEN $min AND $max |
| search by term filter if user passes in one number for Visits | SELECT  Name,  Street AS Address,  City,  Zip,  Size,  PropertyType AS Type ,  (CASE WHEN IsPublic =1 THEN 'True' ELSE 'False' END) AS Public,  (CASE WHEN IsCommercial =1 THEN 'True' ELSE 'False' END) AS Commercial,  ID,  COUNT( \* ) AS Visits,  AVG( Rating ) AS 'Avg. Rating'  FROM Property  JOIN Visit ON Visit.PropertyID = Property.ID  WHERE Property.IsPublic = 1  AND Property.ApprovedBy IS NOT NULL  GROUP BY Property.ID  HAVING COUNT(\*) = $search |
| search by term filter if user passes in one number for avg rating | SELECT  Name,  Street AS Address,  City,  Zip,  Size,  PropertyType AS Type ,  (CASE WHEN IsPublic =1 THEN 'True' ELSE 'False' END) AS Public,  (CASE WHEN IsCommercial =1 THEN 'True' ELSE 'False' END) AS Commercial,  ID,  COUNT( \* ) AS Visits,  AVG( Rating ) AS 'Avg. Rating'  FROM Property  JOIN Visit ON Visit.PropertyID = Property.ID  WHERE Property.IsPublic = 1  AND Property.ApprovedBy IS NOT NULL  GROUP BY Property.ID  HAVING AVG(Rating) = $search |

**Log/unlog visit page for visitors**

|  |  |
| --- | --- |
| Initial population of page | SELECT  P.\*,  FarmItem.Name as FarmItem,  (CASE WHEN FarmItem.Type = 'ANIMAL' THEN 'Animals' ELSE 'Crops' END) as Type  FROM (  SELECT  Property.Name,  Property.Owner,  Email AS 'Owner Email',  Street AS Address,  City,  Zip,  AVG(Rating) as 'Avg.Rating',  Size AS 'Size (acres)',  PropertyType AS Type,  COUNT(\* ) as Visits,  (CASE WHEN IsPublic =1 THEN 'True' ELSE 'False' END) AS Public,  (CASE WHEN IsCommercial =1 THEN 'True' ELSE 'False' END) AS Commercial,  ID  FROM Property  JOIN User ON Property.Owner = User.Username  JOIN Has ON Property.ID = Has.PropertyID  JOIN FarmItem ON FarmItem.Name = Has.ItemName  JOIN Visit ON Visit.PropertyID = Property.ID  WHERE Property.ID =$id  ) AS P  JOIN Has ON Has.PropertyID = P.ID  JOIN FarmItem ON FarmItem.Name = Has.ItemName |
| log new rating with logged date | INSERT INTO Visit VALUES ($username, $propertyid, CURRENT\_TIMESTAMP, $rating); |
| unlog a visit | DELETE FROM Visit WHERE Username = $username AND PropertyID = $id |
| see if visitor has already logged for a property | SELECT \*  FROM Visit  WHERE PropertyID = $id AND Username = $username |

**Visitor’s visit history**

|  |  |
| --- | --- |
| Populate list of visit history | SELECT Property.Name, Visit.VisitDate, Visit.Rating, Property.ID  FROM Visit JOIN Property ON Property.ID = Visit.PropertyID  WHERE Visit.Username = $username |
| View property from visit history | SELECT  P.\*,  FarmItem.Name as FarmItem,  (CASE WHEN FarmItem.Type = 'ANIMAL' THEN 'Animals' ELSE 'Crops' END) as Type  FROM (  SELECT  Property.Name,  Property.Owner,  Email AS 'Owner Email',  Street AS Address,  City,  Zip,  AVG(Rating) as 'Avg.Rating',  Size AS 'Size (acres)',  PropertyType AS Type,  COUNT(\* ) as Visits,  (CASE WHEN IsPublic =1 THEN 'True' ELSE 'False' END) AS Public,  (CASE WHEN IsCommercial =1 THEN 'True' ELSE 'False' END) AS Commercial,  ID  FROM Property  JOIN User ON Property.Owner = User.Username  JOIN Has ON Property.ID = Has.PropertyID  JOIN FarmItem ON FarmItem.Name = Has.ItemName  JOIN Visit ON Visit.PropertyID = Property.ID  WHERE Property.ID =$id  ) AS P  JOIN Has ON Has.PropertyID = P.ID  JOIN FarmItem ON FarmItem.Name = Has.ItemName |

**View unconfirmed property as an admin**

|  |  |
| --- | --- |
| Population of unconfirmed list | SELECT  Name,  Street,  City,  Zip,  Size,  PropertyType as Type,  (CASE WHEN IsPublic =1 THEN 'True' ELSE 'False' END) AS Public,  (CASE WHEN IsCommercial =1 THEN 'True' ELSE 'False' END) AS Commercial,  ID,  Owner  FROM Property  WHERE ApprovedBy IS NULL; |
| Search by filter terms | SELECT  Name,  Street,  City,  Zip,  Size,  PropertyType as Type,  (CASE WHEN IsPublic =1 THEN 'True' ELSE 'False' END) AS Public,  (CASE WHEN IsCommercial =1 THEN 'True' ELSE 'False' END) AS Commercial,  ID,  Owner  FROM Property  WHERE ApprovedBy IS NULL and $searchby = $search |

**View confirmed property as an admin**

|  |  |
| --- | --- |
| Population of confirmed list | SELECT  Name,  Street,  City,  Zip,  Size,  PropertyType as Type,  (CASE WHEN IsPublic =1 THEN 'True' ELSE 'False' END) AS Public,  (CASE WHEN IsCommercial =1 THEN 'True' ELSE 'False' END) AS Commercial, ID, ApprovedBy as VerifiedBy, AVG(Rating)  FROM Property JOIN Visit ON Visit.PropertyID = Property.ID  WHERE ApprovedBy IS NOT NULL  GROUP BY Name; |
|  | SELECT  Name,  Street,  City,  Zip,  Size,  PropertyType as Type,  (CASE WHEN IsPublic =1 THEN 'True' ELSE 'False' END) AS Public,  (CASE WHEN IsCommercial =1 THEN 'True' ELSE 'False' END) AS Commercial,  ID,  Owner  FROM Property  WHERE ApprovedBy IS NOT NULL and $searchby = $search |

**Admin viewing details of unconfirmed/confirmed properties**

|  |  |
| --- | --- |
| Unconfirmed property details | SELECT  P.\*,  FarmItem.Name as FarmItem,  (CASE WHEN FarmItem.Type = 'ANIMAL' THEN 'Animals' ELSE 'Crops' END) as Type  FROM (  SELECT  Property.Name,  Property.Owner,  Email AS 'Owner Email',  Street AS Address,  City,  Zip,  AVG(Rating) as 'Avg.Rating',  Size AS 'Size (acres)',  PropertyType AS Type,  COUNT(\* ) as Visits,  (CASE WHEN IsPublic =1 THEN 'True' ELSE 'False' END) AS Public,  (CASE WHEN IsCommercial =1 THEN 'True' ELSE 'False' END) AS Commercial,  ID  FROM Property  JOIN User ON Property.Owner = User.Username  JOIN Has ON Property.ID = Has.PropertyID  JOIN FarmItem ON FarmItem.Name = Has.ItemName  JOIN Visit ON Visit.PropertyID = Property.ID  WHERE Property.ID =$id  ) AS P  JOIN Has ON Has.PropertyID = P.ID  JOIN FarmItem ON FarmItem.Name = Has.ItemName |
| check if new property name doesnt exist yet | SELECT \*  FROM Property  WHERE ID != $id AND Name = $name |
| get list of crops that this property grows | SELECT ItemName  FROM Has  JOIN FarmItem ON FarmItem.Name = Has.ItemName  WHERE Has.PropertyID = $id AND FarmItem.Type != 'ANIMAL' |
| get list of animals that this farm raises | SELECT ItemName  FROM Has  JOIN FarmItem ON FarmItem.Name = Has.ItemName  WHERE Has.PropertyID = $id AND FarmItem.Type = 'ANIMAL' |
| get list of new crops that can be added to the property | SELECT Name  FROM FarmItem  WHERE FarmItem.Type != 'ANIMAL'  AND FarmItem.IsApproved = True  AND NOT EXISTS (  SELECT \*  FROM Has  WHERE Has.ItemName = FarmItem.Name  AND Has.PropertyID =$id); |
| get list of new animals that can be added to the property | SELECT Name  FROM FarmItem  WHERE FarmItem.Type = 'ANIMAL'  AND FarmItem.IsApproved = True  AND NOT EXISTS (  SELECT \*  FROM Has  WHERE Has.ItemName = FarmItem.Name  AND Has.PropertyID =$id); |
| get list of previous crops to check if it already exists | SELECT Name  FROM FarmItem  WHERE FarmItem.Name = $name |
| add new crop/animal to farmitem to be reviewed | INSERT INTO FarmItem VALUES ($name, False, $type); |
| delete Property | DELETE FROM Property  WHERE ID = $id |
| save changes (confirm property) | UPDATE Property  SET Name = $name, Size = $size, IsCommercial = $commercial, IsPublic = $public, Street = $street,City = $city, Zip = $zip, ApprovedBy = $adminusername  WHERE ID = $id |
| update new crop/animal (approved) | INSERT INTO Has VALUES ($propertyid, $itemname) |

**All visitors list that admins can view**

|  |  |
| --- | --- |
| initial population of table | SELECT User.Username, User.Email, COUNT(VisitDate) as LoggedVisits  FROM User LEFT JOIN Visit ON Visit.Username = User.Username  WHERE User.UserType = 'VISITOR'  GROUP BY Username; |
| search by search term | SELECT User.Username, User.Email, COUNT(\*) as LoggedVisits  FROM User JOIN Visit ON Visit.Username = User.Username  WHERE User.UserType = 'VISITOR' AND $searchby = $search  GROUP BY Username; |
| delete visitor account | DELETE FROM User WHERE Username = $visitorusername |
| delete visitor's log history | DELETE FROM Vist WHERE Username = $visitorusername |

**All owners list that admins can view**

|  |  |
| --- | --- |
| initial population of table | SELECT User.Username, User.Email, COUNT(\*) as NumProperties  FROM User JOIN Property ON Property.Owner = User.Username  WHERE User.UserType = 'OWNER'  GROUP BY Username; |
| search by search term | SELECT User.Username, User.Email, COUNT(\*) as NumProperties  FROM User JOIN Property ON Property.Owner = User.Username  WHERE User.UserType = 'OWNER' AND $searchby = $search  GROUP BY Username; |
| delete owner account | DELETE FROM User WHERE Username = $visitorusername |

**Approved animals/crops list that admins can view**

|  |  |
| --- | --- |
| Make list | SELECT Name, Type  FROM FarmItem  WHERE IsApproved = True; |
| Add to approved list | INSERT INTO FarmItem VALUES ($itemname, 1, $type); |
| search by search term | SELECT Name, Type  FROM FarmItem  WHERE IsApproved = True AND $searchby = $search |

**Pending animals/crops list that admins can view**

|  |  |
| --- | --- |
| Make list | SELECT Name, Type  FROM FarmItem  WHERE IsApproved = False; |
| approve selection | UPDATE FarmItem  SET IsApproved = True  WHERE Name = $name; |
| delete selection | DELETE FROM FarmItem  WHERE Name = $name; |