DATA BROS

Shane Seeley and Jozef Maselek

Contact Management System

CMPT 308N-113

11161

Outline of our system

- Contact management system where users can add and interact with their contacts
  - Mainly meant to be at the disposal of a small company
- Admins can go and manage users including adding and deleting them and can manage the system as a whole

Outline of our system

EACH USER SHOULD BE ABLE TO:

a. ADD MULTIPLE SETS OF CONTACT INFORMATION WITH THE FOLLOWING DETAILS: FIRST NAME, SURNAME, CELL PHONE, WORKPLACE PHONE, FAX NUMBER, EMAIL ADDRESS, GENDER, AND AGE

b. REMOVE A CONTACT RECORD

c. EDIT THE CONTACT RECORD’S DETAILS

d. SEARCH THROUGH CONTACTS BASED ON ONE OR SEVERAL FEATURES AND LIST THE RESULTS ON THE SCREEN. FOR INSTANCE, IT SHOULD BE ABLE TO RETURN THE CELL PHONE NUMBER OF A SPECIFIC NAME

References:

(1) "Google Contacts." *Google Contacts Software Reviews, Demo & Pricing - 2023*, www.softwareadvice.com/crm/google-contacts-profile/. Accessed 25 Oct. 2023.

(2) "Customer Relationship Management Definition - Salesforce Us." *Salesforce*, www.salesforce.com/crm/what-is-crm/. Accessed 25 Oct. 2023.

(3) "WhatCMS - Microsoft Excel." *What CMS?*, whatcms.org/c/Microsoft-Excel. Accessed 25 Oct. 2023.

Describe how you created this mini world and how you selected the entities, attributes, relationships, participations, and cardinality.

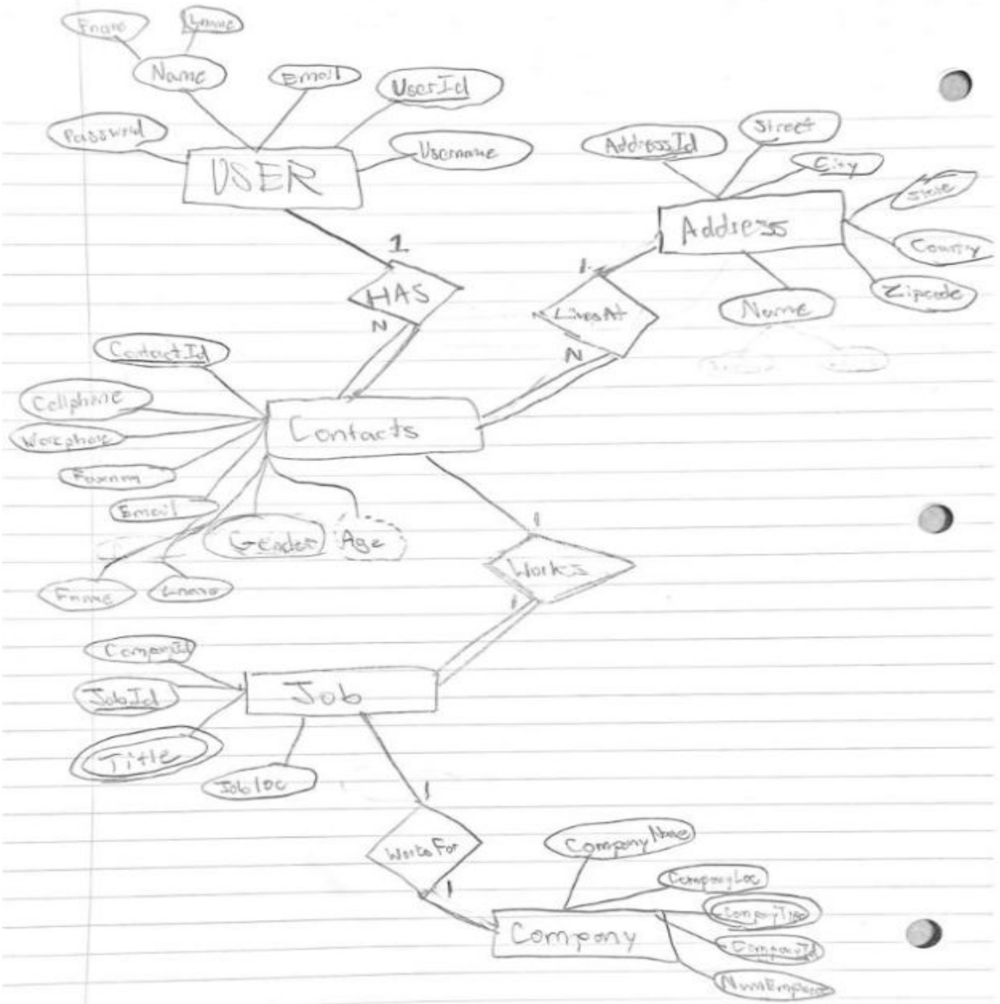
To create this mini world, we thought about what would be important for a contact management system to function and what kind of amenities users might want. Because it's acontact-focusedd database the entities are centered around the contacts entity. Attributes belonging to each entity are self-explanatory and are indicative of the table they are assigned to. The entities not directly related to the contacts entity are meant to further elaborate and give insight on information like departments or notes on interactions. Participations are tied together by necessity for the most part.

Address is an entity that stores an AddressID, the street address, city, state, country, zip code, and the name of the person associated. It is related to the contacts entity and stores the address information of specific contacts. Many contacts can store many addresses.

Contacts is an entity with the attributes Fname, Lname, Cellphone, Workphone, FaxNum, Email, Gender, and Birthday. It stores the contact info for a person. It's related to most of the other entities in many different ways.

User is an entity with the attributes Username, Password, UserID, IsAdmin, Email, Fname, and Lname. It is an entity to store the user’s information and has an attribute to discern whether it's an admin. Its connected to contacts in a 1:N configuration.

Job, Company, and Department are related entities, describing the jobs of individual contacts, the company they work for, and the department they work in that company. Job has the attributes JobID, Title, JobLocation, CompanyID, and ContactID. It is connected to contacts and company in a 1:1 relationship. Company has CompanyID, CompanyName, CompanyLocation, CompanyType, and Employees and is connected to Job and Department in a 1:1 relationship. Department is connect to Company and Contacts in a 1:1 relationship and has the attributes DepartmentID, Dname, Dlocation, DcompanyID, Demployees, and Dmanagername.


Fname
Lname
Name
Email
UserId
Password
USER
Username
1
HAS
N
AddressId
Street
City
State
Address
Country
Zipcode
Name
1
LivesAt
N
ContactId
Cellphone
Workphone
Faxnum
Email
Gender
Age
Fname
Lname
Contacts
1
WorksI
1
CompanyId
JobId
Title
Job
JobLoc
1
WorksFor
1
CompanyName
CompanyLoc
CompanyType
CompanyId
NumEmployees
Company

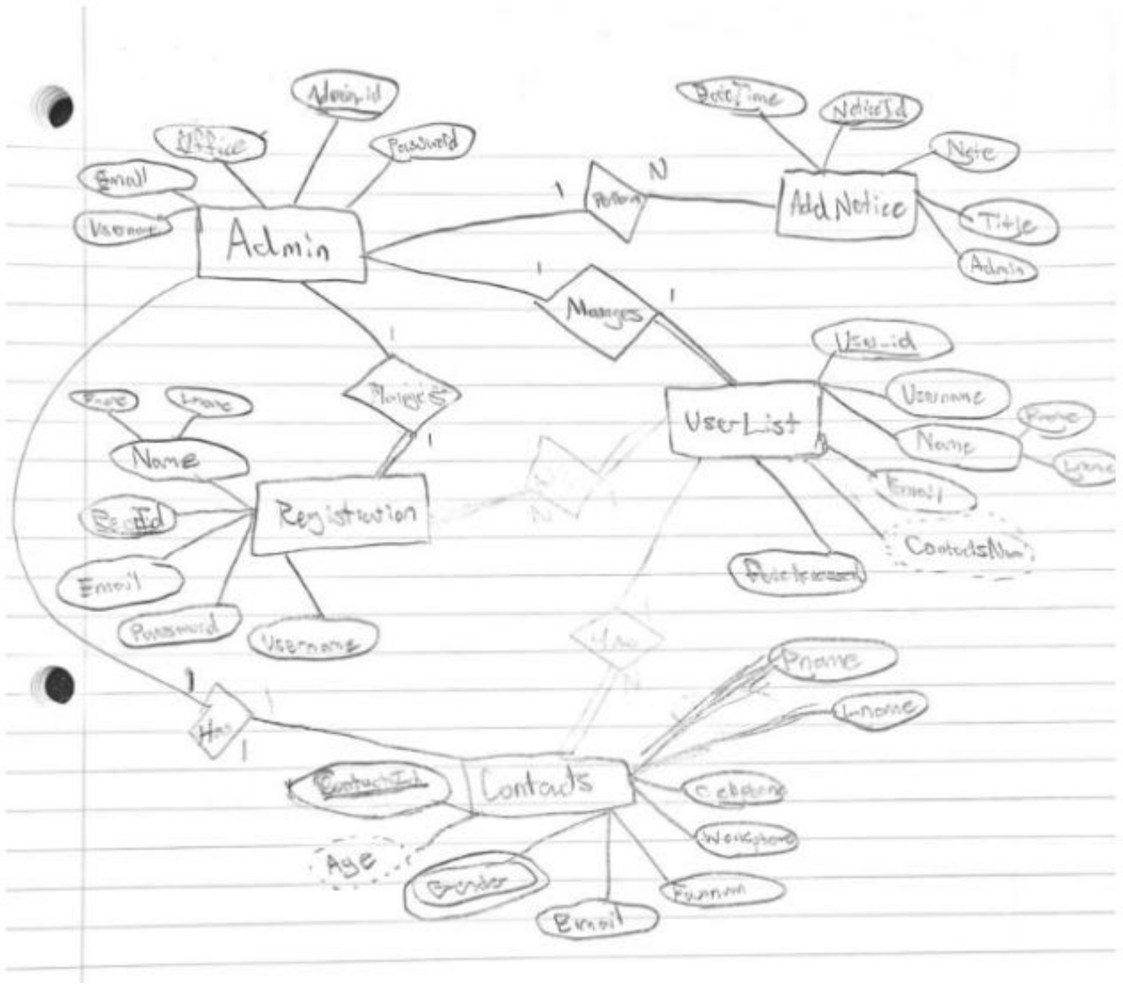
Contacts is an entity with the attributes Fname, Lname, Cellphone, Workphone, FaxNum, Email, Gender, and Birthday. It stores the contact info for a person. Its function in this ER model is in the case an admin wants to access a list of Admin-related contacts.

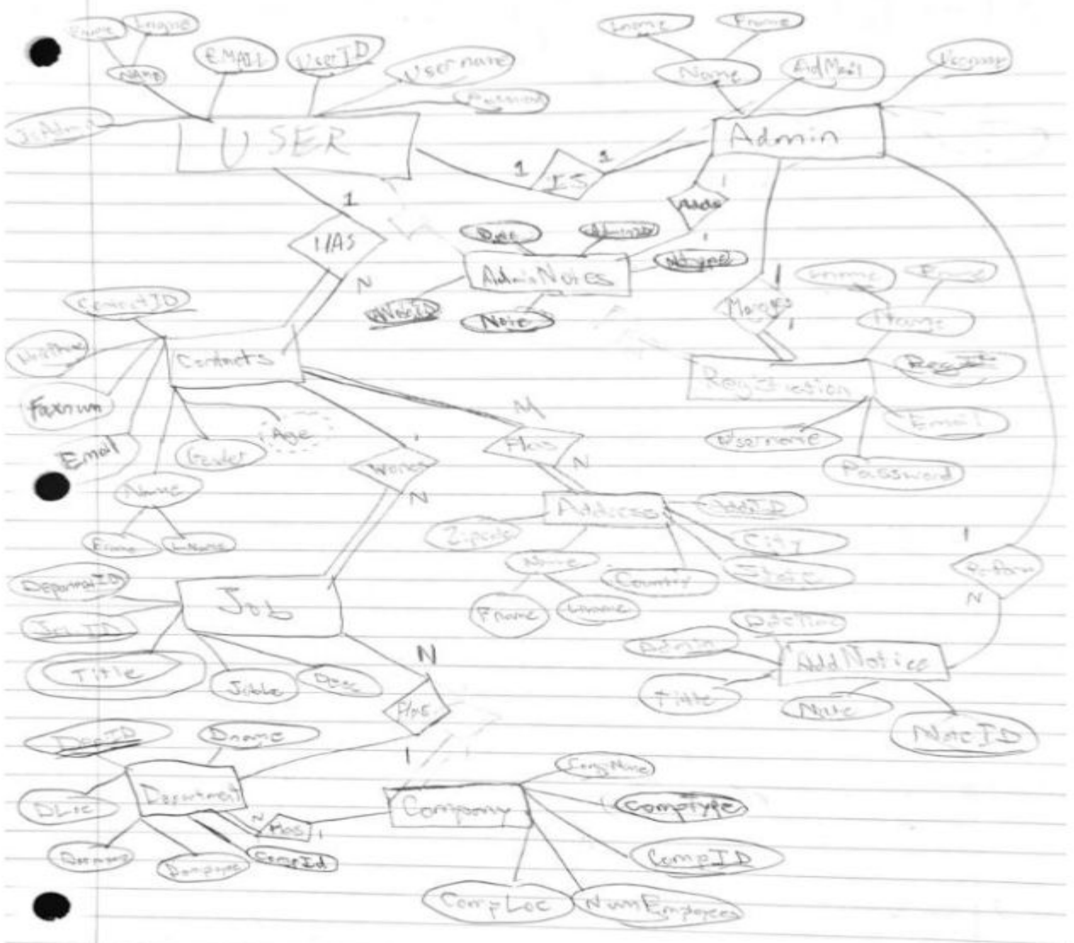
Admin is an entity with the attributes Name, Admin_ID, Password, Email, and Username. It stores the info of Admins for the database system. It connects to all of the other entities other than contacts.

UserList is an entity with the attributes User_ID, username, name, email, contactsNum, and Date accessed. It stores the list of Users in the database and allows admins to change or view that information. It has a 1:1 relationship with Registration.

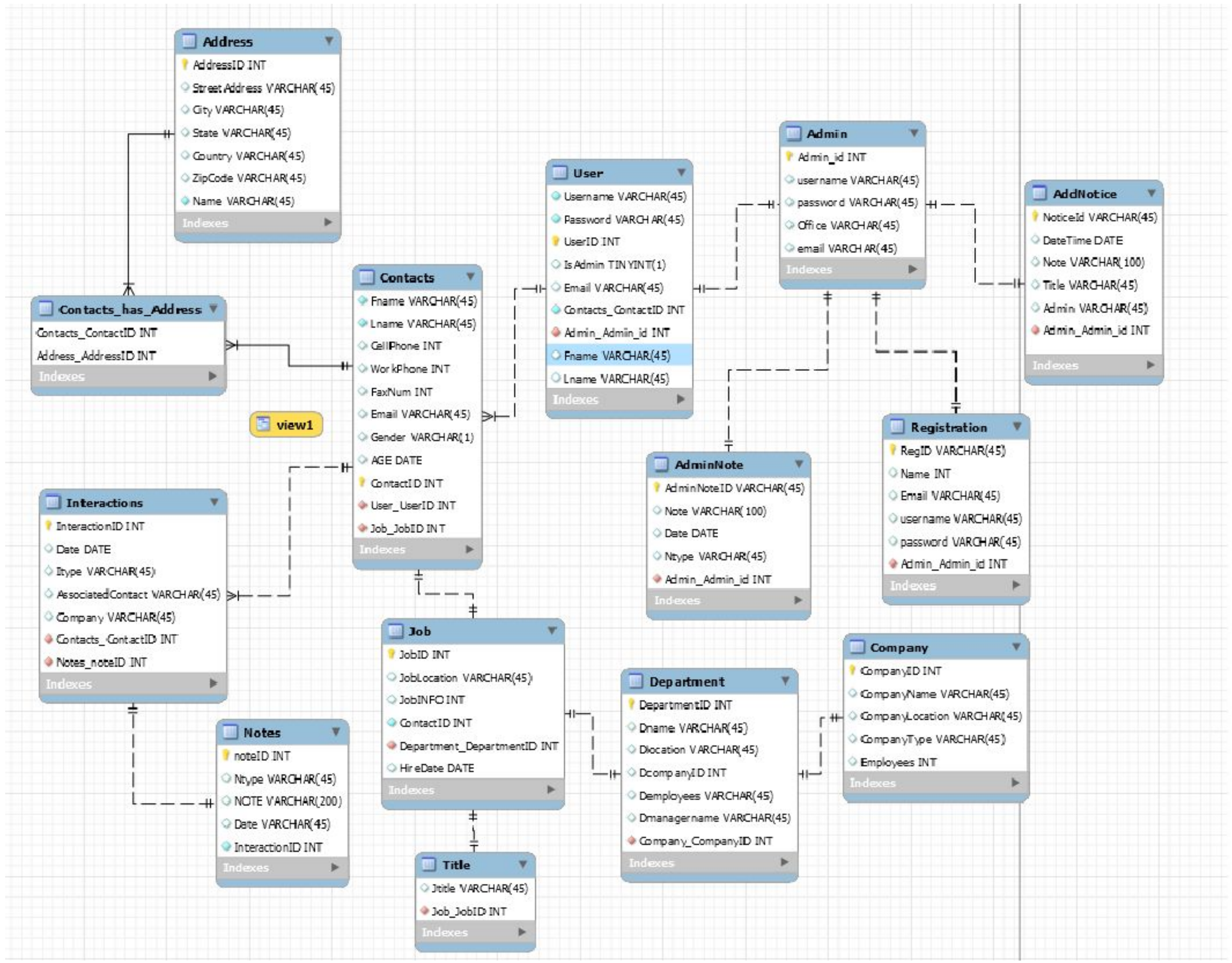
Registration is an entity with the attributes Name, RegID, Email, Password, and Username. This entity stores the list of users that can be added but aren't yet. The admin would have to go through and approve of the users in this list to be added to UserList. It has a 1:1 relationship with Admin.

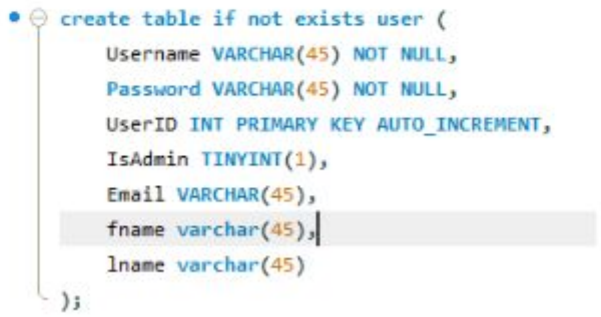
AddNotice is an entity with the attribute DateTime, NoticeID, Note, Title, and Admin. This is an entity that stores the information for notices about the Contact management system. It has a 1:N relationship with Admin as an admin could have many notices.


Admin_id
Office
Password
Email
Username
Admin
1
Perform
N
Add Notice
DateTime
NoticeId
Note
Title
Admin
1
Manages
1
User_id
Username
UserList
Fname
Name
Lname
Email
ContactsNum
DateCreated
1
Manages
1
Fname
Lname
Name
RegId
Registration
Email
Password
Username
1
Has
1
ContactId
Contacts
Fname
L-name
Cellphone
Workphone
Funnum
Age
Gender
Email

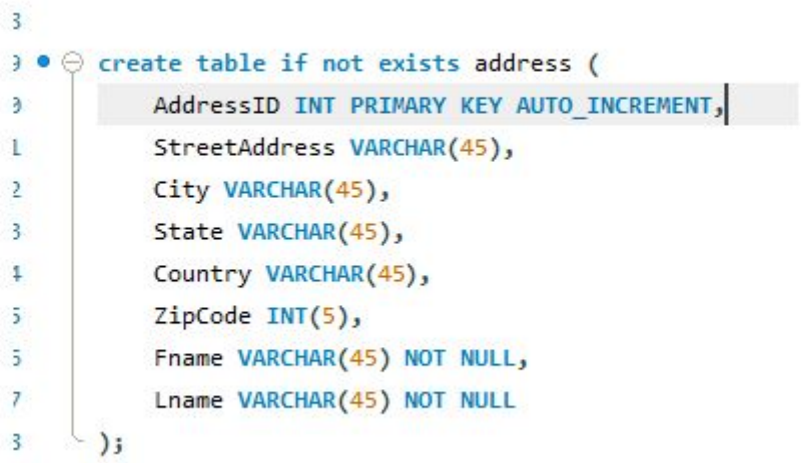


USER
Fname
Lname
NAME
EMAIL
UserID
Username
Password
IsAdmin
1
IS
1
Admin
Lname
Fname
Name
AdMail
Username
1
HAS
N
Adds
1
AdminNotes
Date
AdminID
NoteID
Note
Manages
1
1
Fname
Email
Name
RegID
Registration
Username
Email
Password
ContactID
Contacts
WorkPhone
Faxnum
Email
Gender
Age
Name
Fname
Lname
M
Has
N
Works
N
Address
AddID
Zipcode
City
State
Country
Name
Fname
Lname
1
Perform
N
AddNotice
Admin
Title
Note
NoteID
Job
DepartmentID
JobID
Title
JobTitle
Desc
N
Has
1
DeptID
Dname
Department
DLoc
Company
CompName
CompType
CompID
CompLoc
NumEmployees
N
Has
1
CompId

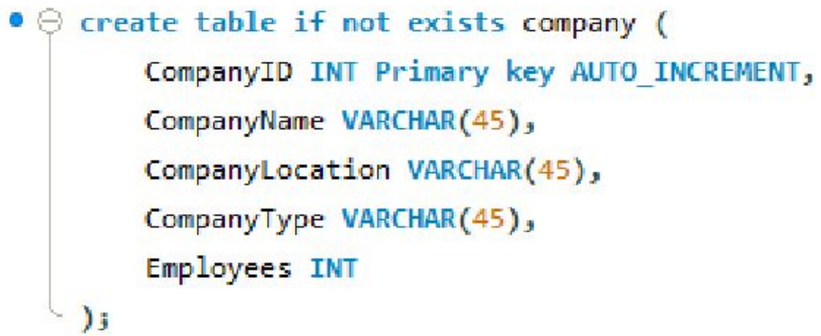

Address
AddressID INT
Street Address VARCHAR(45)
City VARCHAR(45)
State VARCHAR(45)
Country VARCHAR(45)
ZipCode VARCHAR(45)
Name VARCHAR(45)
Indexes
Contacts_has_Address
Contacts_ContactID INT
Address_AddressID INT
Indexes
view1
Contacts
Fname VARCHAR(45)
Lname VARCHAR(45)
CellPhone INT
WorkPhone INT
FaxNum INT
Email VARCHAR(45)
Gender VARCHAR(1)
AGE DATE
ContactID INT
User_UserID INT
Job_JobID INT
Indexes
User
Username VARCHAR(45)
Password VARCHAR(45)
UserID INT
IsAdmin TINYINT(1)
Email VARCHAR(45)
Contacts_ContactID INT
Admin_Admin_id INT
Fname VARCHAR(45)
Lname VARCHAR(45)
Indexes
Admin
Admin_id INT
username VARCHAR(45)
password VARCHAR(45)
Office VARCHAR(45)
email VARCHAR(45)
Indexes
AddNotice
NoticeId VARCHAR(45)
DateTime DATE
Note VARCHAR(100)
Title VARCHAR(45)
Admin VARCHAR(45)
Admin_Admin_id INT
Indexes
Registration
RegID VARCHAR(45)
Name INT
Email VARCHAR(45)
username VARCHAR(45)
password VARCHAR(45)
Admin_Admin_id INT
Indexes
AdminNote
AdminNoteID VARCHAR(45)
Note VARCHAR(100)
Date DATE
Ntype VARCHAR(45)
Admin_Admin_id INT
Indexes
Interactions
InteractionID INT
Date DATE
Itype VARCHAR(45)
AssociatedContact VARCHAR(45)
Company VARCHAR(45)
Contacts_ContactID INT
Notes_noteID INT
Indexes
Notes
noteID INT
Ntype VARCHAR(45)
NOTE VARCHAR(200)
Date VARCHAR(45)
InteractionID INT
Indexes
Job
JobID INT
JobLocation VARCHAR(45)
JobINFO INT
ContactID INT
Department_DepartmentID INT
HireDate DATE
Indexes
Title
Jtitle VARCHAR(45)
Job_JobID INT
Indexes
Department
DepartmentID INT
Dname VARCHAR(45)
Dlocation VARCHAR(45)
DcompanyID INT
Demployees VARCHAR(45)
Dmanagername VARCHAR(45)
Company_CompanyID INT
Indexes
Company
CompanyID INT
CompanyName VARCHAR(45)
CompanyLocation VARCHAR(45)
CompanyType VARCHAR(45)
Employees INT
Indexes



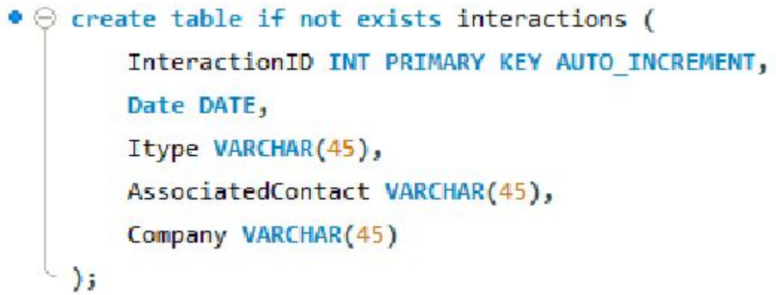
```sql
create table if not exists user (
    Username VARCHAR(45) NOT NULL,
    Password VARCHAR(45) NOT NULL,
    UserID INT PRIMARY KEY AUTO_INCREMENT,
    IsAdmin TINYINT(1),
    Email VARCHAR(45),
    fname varchar(45),
    lname varchar(45)
);
```



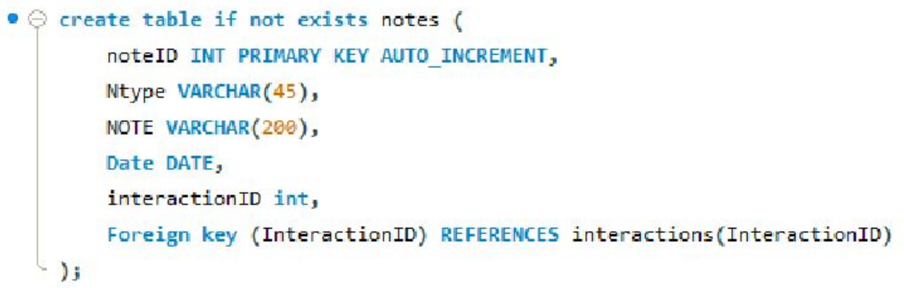
```sql
create table if not exists address (
    AddressID INT PRIMARY KEY AUTO_INCREMENT,
    StreetAddress VARCHAR(45),
    City VARCHAR(45),
    State VARCHAR(45),
    Country VARCHAR(45),
    ZipCode INT(5),
    Fname VARCHAR(45) NOT NULL,
    Lname VARCHAR(45) NOT NULL
);
```



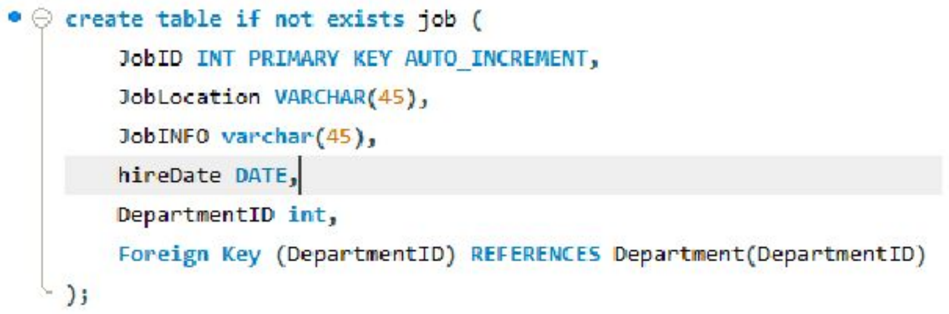
```sql
create table if not exists company (
    CompanyID INT Primary key AUTO_INCREMENT,
    CompanyName VARCHAR(45),
    CompanyLocation VARCHAR(45),
    CompanyType VARCHAR(45),
    Employees INT
);
```



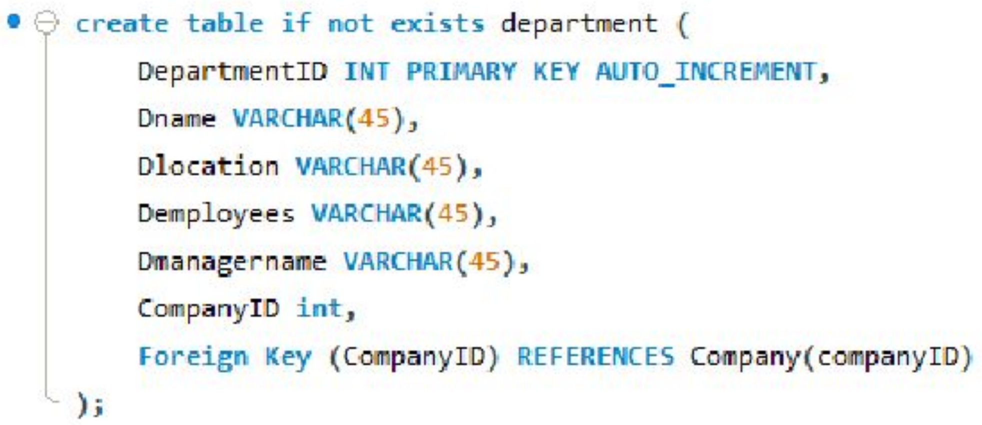
```sql
create table if not exists interactions (
    InteractionID INT PRIMARY KEY AUTO_INCREMENT,
    Date DATE,
    Itype VARCHAR(45),
    AssociatedContact VARCHAR(45),
    Company VARCHAR(45)
);
```



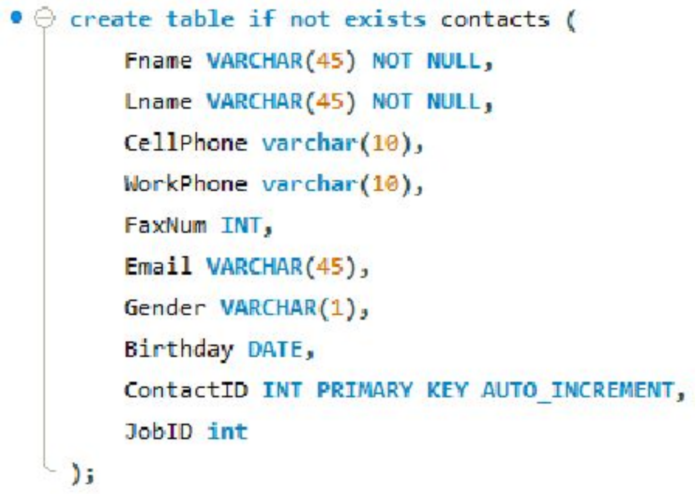
```sql
create table if not exists notes (
    noteID INT PRIMARY KEY AUTO_INCREMENT,
    Ntype VARCHAR(45),
    NOTE VARCHAR(200),
    Date DATE,
    interactionID int,
    Foreign key (InteractionID) REFERENCES interactions(InteractionID)
);
```



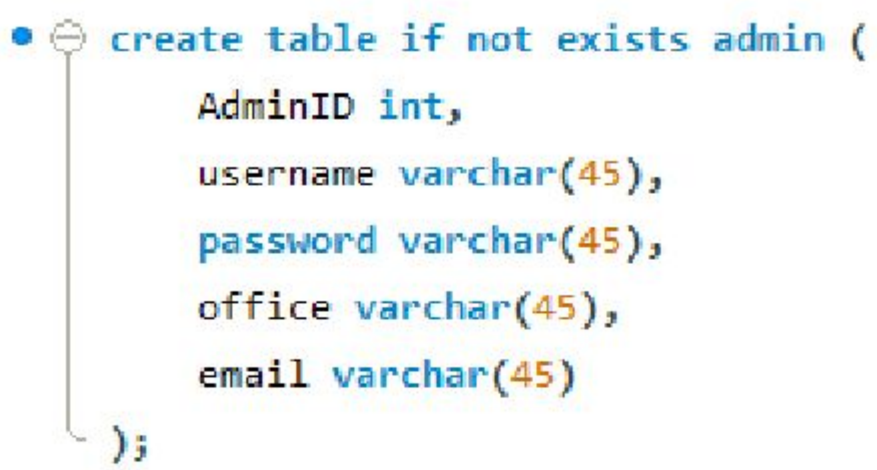
```sql
create table if not exists job (
    JobID INT PRIMARY KEY AUTO_INCREMENT,
    JobLocation VARCHAR(45),
    JobINFO varchar(45),
    hireDate DATE,
    DepartmentID int,
    Foreign Key (DepartmentID) REFERENCES Department(DepartmentID)
);
```



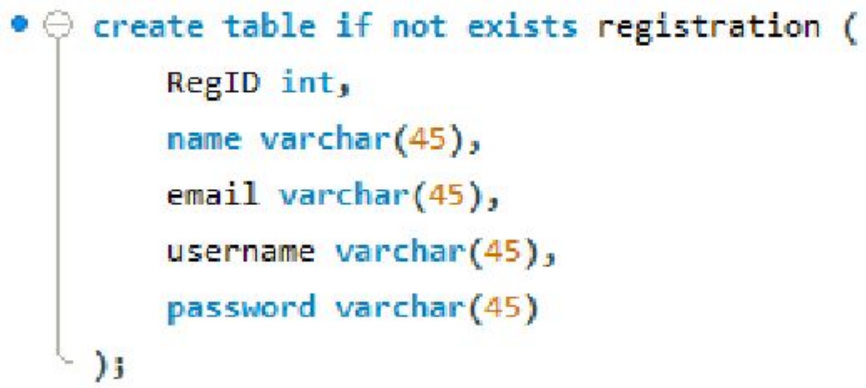
```sql
create table if not exists department (
    DepartmentID INT PRIMARY KEY AUTO_INCREMENT,
    Dname VARCHAR(45),
    Dlocation VARCHAR(45),
    Demployees VARCHAR(45),
    Dmanagername VARCHAR(45),
    CompanyID int,
    Foreign Key (CompanyID) REFERENCES Company(companyID)
);
```



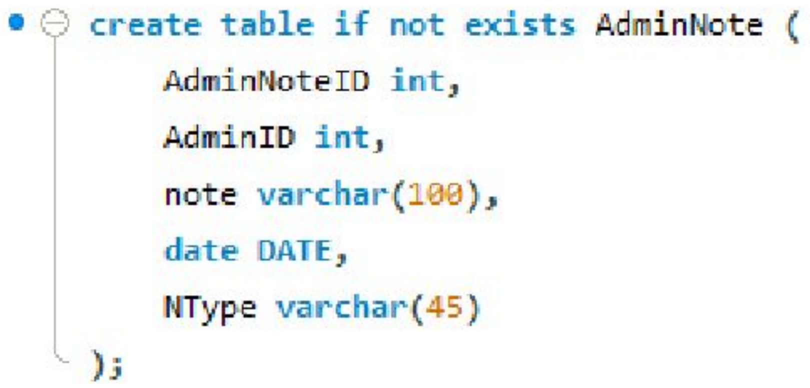
```sql
create table if not exists contacts (
    Fname VARCHAR(45) NOT NULL,
    Lname VARCHAR(45) NOT NULL,
    CellPhone varchar(10),
    WorkPhone varchar(10),
    FaxNum INT,
    Email VARCHAR(45),
    Gender VARCHAR(1),
    Birthday DATE,
    ContactID INT PRIMARY KEY AUTO_INCREMENT,
    JobID int
);
```



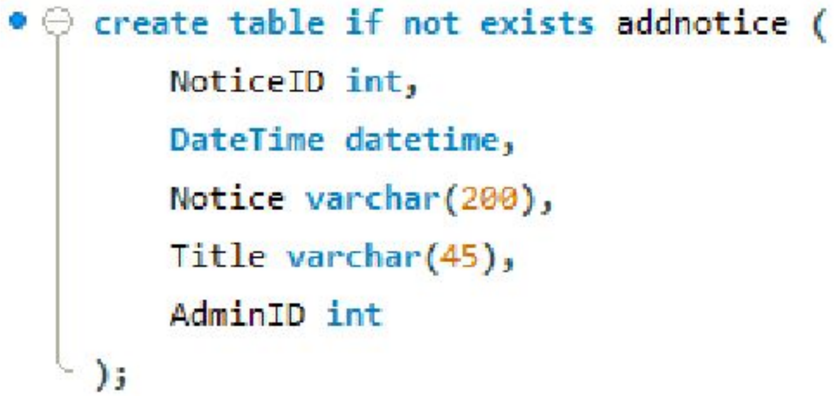
```
create table if not exists admin (
    AdminID int,
    username varchar(45),
    password varchar(45),
    office varchar(45),
    email varchar(45)
);
```



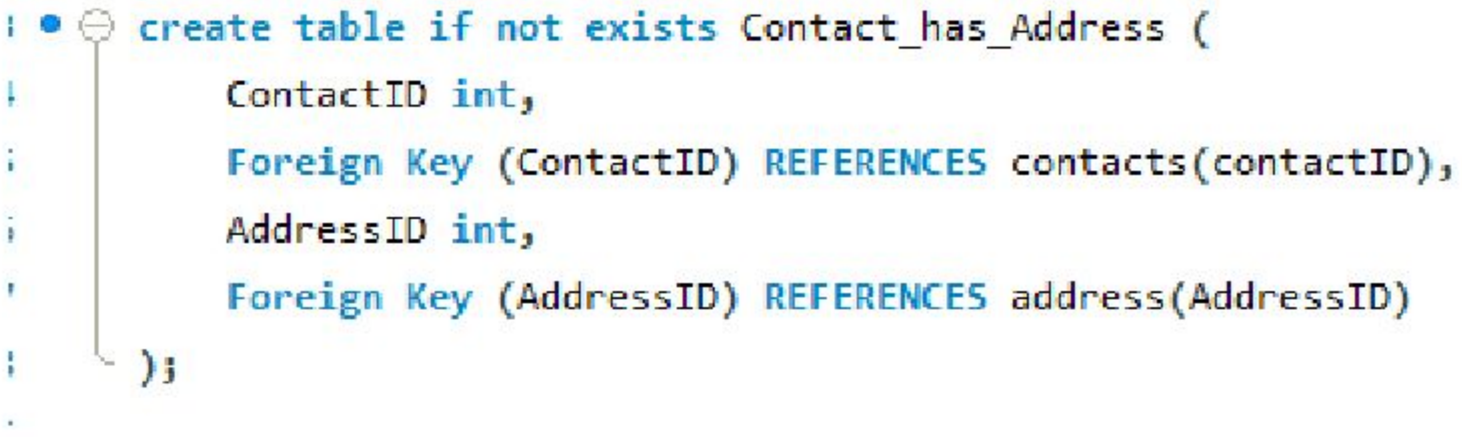
```
create table if not exists registration (
    RegID int,
    name varchar(45),
    email varchar(45),
    username varchar(45),
    password varchar(45)
);
```



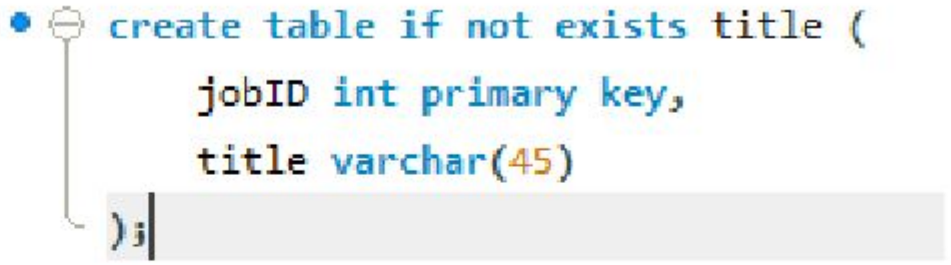
```
create table if not exists AdminNote (
    AdminNoteID int,
    AdminID int,
    note varchar(100),
    date DATE,
    NType varchar(45)
);
```



```
create table if not exists addnotice (
    NoticeID int,
    DateTime datetime,
    Notice varchar(200),
    Title varchar(45),
    AdminID int
);
```

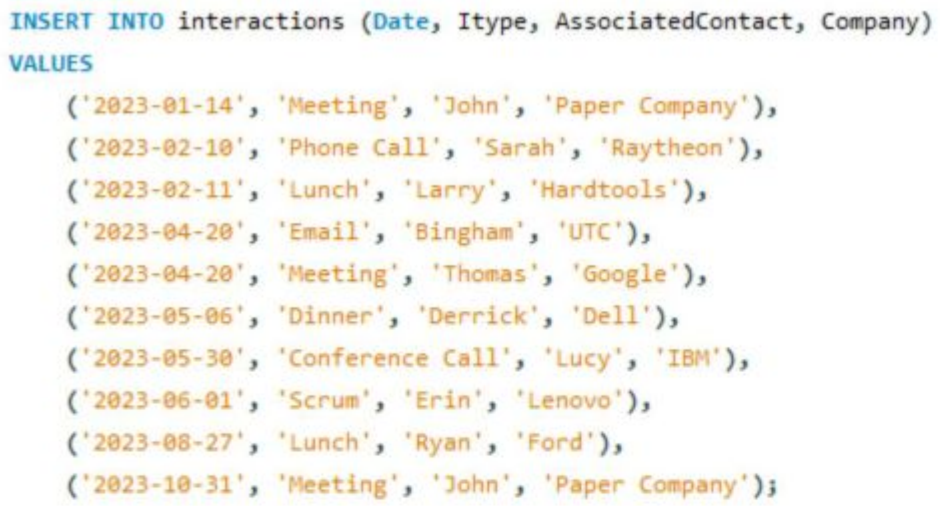


```sql
create table if not exists Contact_has_Address (
    ContactID int,
    Foreign Key (ContactID) REFERENCES contacts(contactID),
    AddressID int,
    Foreign Key (AddressID) REFERENCES address(AddressID)
);
```

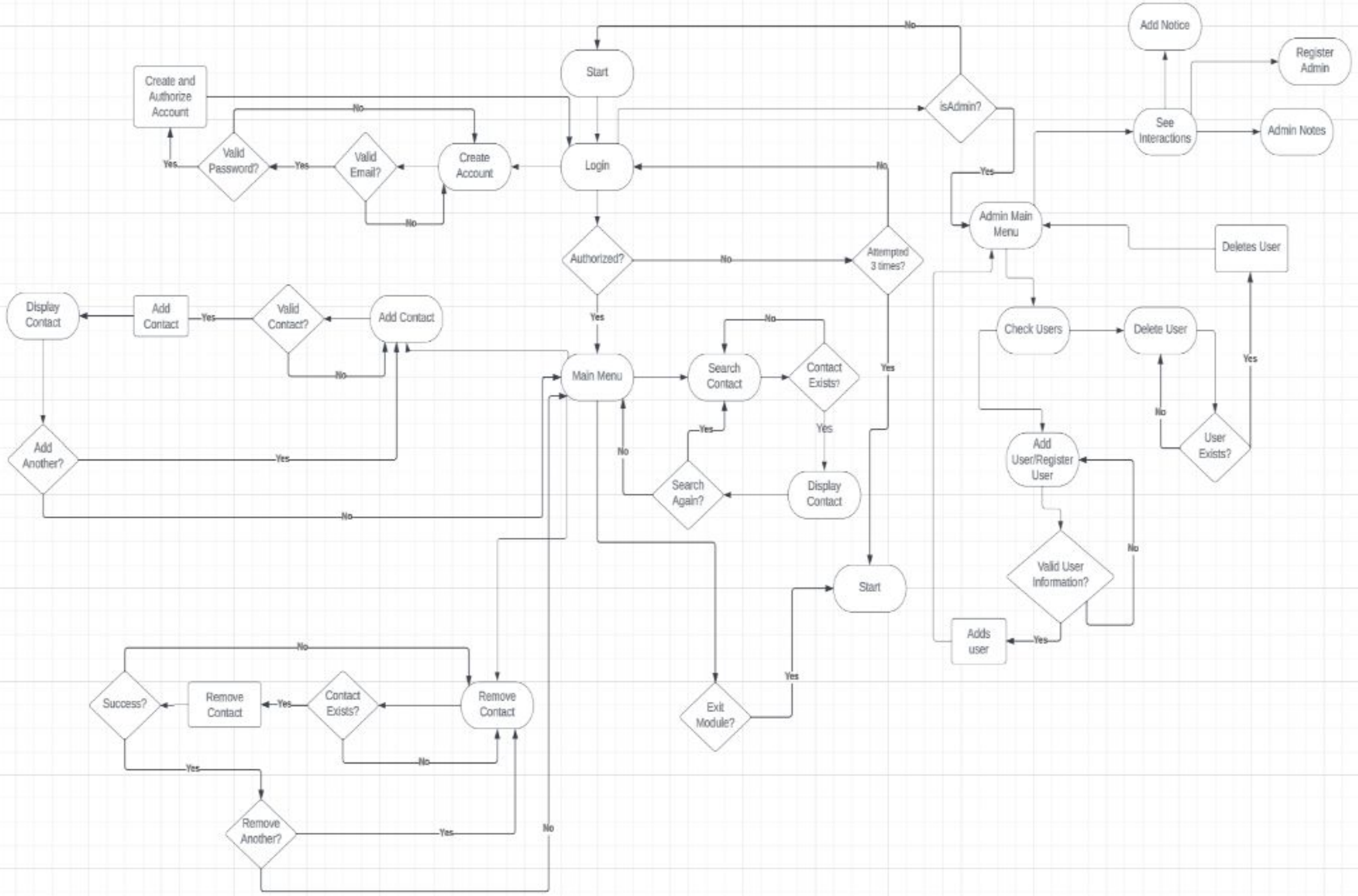


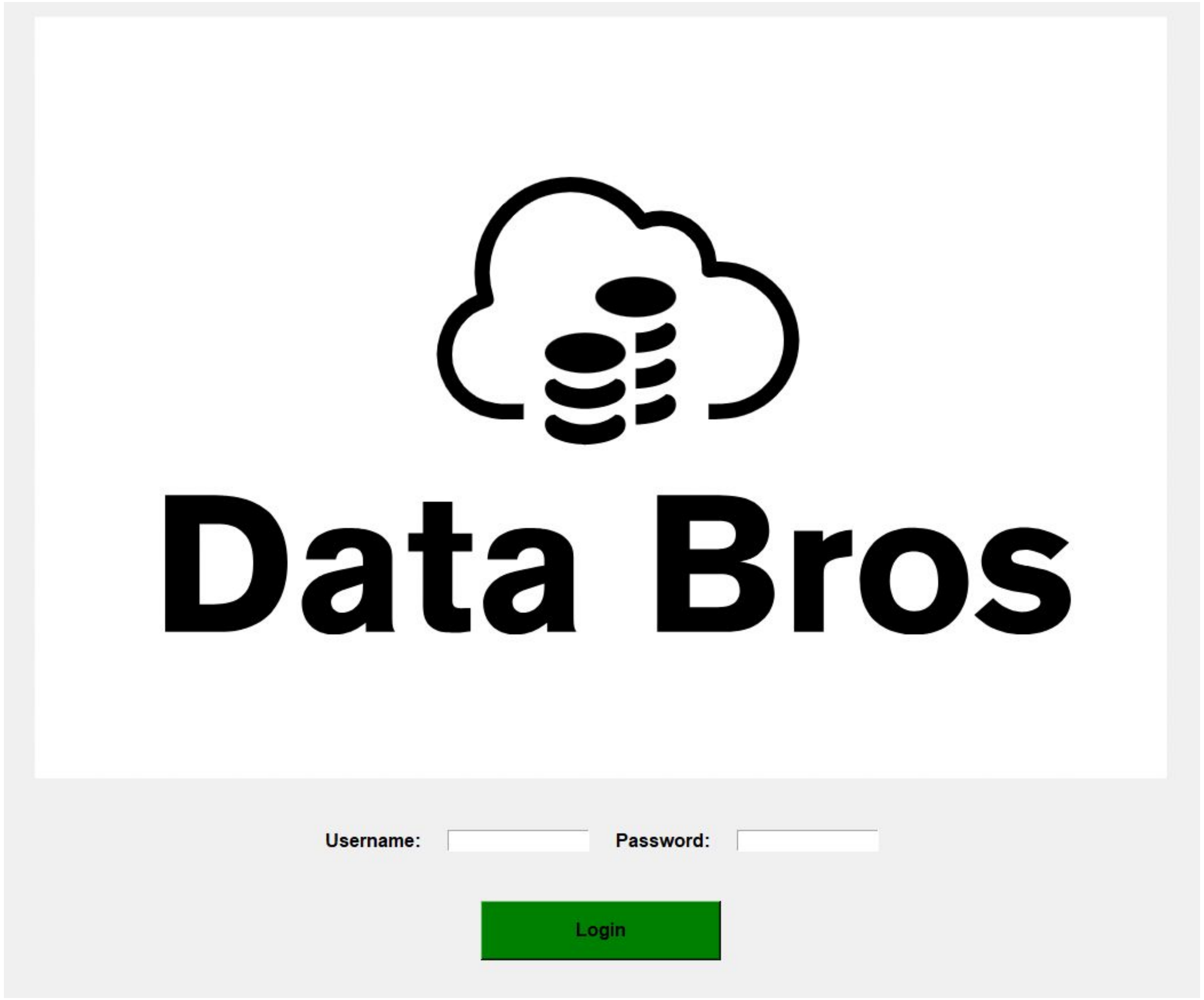
```sql
create table if not exists title (
    jobID int primary key,
    title varchar(45)
);
```

Inserting the data in one query through a multi-value insert statement, like the one shown in Figure 18, only took 0.016 sec. This applies all the data at once. Alternatively, doing a single insert statement for each piece of data takes 0.016 seconds to accomplish. To insert ten different statements would therefore take 0.16 seconds. This means that the multi-valued insert statement is 10x faster than the ten single insert statements.



```
INSERT INTO interactions (Date, Itype, AssociatedContact, Company)
VALUES
    ('2023-01-14', 'Meeting', 'John', 'Paper Company'),
    ('2023-02-10', 'Phone Call', 'Sarah', 'Raytheon'),
    ('2023-02-11', 'Lunch', 'Larry', 'Hardtools'),
    ('2023-04-20', 'Email', 'Bingham', 'UTC'),
    ('2023-04-20', 'Meeting', 'Thomas', 'Google'),
    ('2023-05-06', 'Dinner', 'Derrick', 'Dell'),
    ('2023-05-30', 'Conference Call', 'Lucy', 'IBM'),
    ('2023-06-01', 'Scrum', 'Erin', 'Lenovo'),
    ('2023-08-27', 'Lunch', 'Ryan', 'Ford'),
    ('2023-10-31', 'Meeting', 'John', 'Paper Company');
```


Start
Create and Authorize Account
Valid Password?
Valid Email?
Create Account
Login
isAdmin?
Add Notice
Register Admin
See Interactions
Admin Notes
Admin Main Menu
Authorized?
Attempted 3 times?
Deletes User
Display Contact
Add Contact
Valid Contact?
Add Contact
Main Menu
Search Contact
Contact Exists?
Check Users
Delete User
User Exists?
Add Another?
Search Again?
Display Contact
Add User/Register User
Valid User Information?
Start
Adds user
Success?
Remove Contact
Contact Exists?
Remove Contact
Exit Module?
Remove Another?
Yes
No


Data Bros
Username:
Password:
Login

Thank You