دانشگاه صنعتی شریف
دانشکده مهندسی کامپیوتر

عنوان:

سیستم مدیریت کتابخانه

اعضای گروه

آرش ضیایی رازبان ۴۰۰۱۰۵۱۰۹

علیرضا سلیمیان ۴۰۰۱۰۵۰۳۶

محمد لطفی ۴۰۰۱۰۵۲۲۸

نام درس

طراحی پایگاه داده ها

نیم‌سال دوم ۱۴۰۱-۱۴۰۲

نام استاد درس

مهدی دادبخش

فهرست مطالب

آ عملیات سیستم پایگاه داده

با توجه به توضیحی که از جزییات فاز چهارم، در سامانه درس افزار نوشته شده بود و طبق برداشت خودمان از این فاز، به شرح این فاز به صورت زیر می پردازیم.
لیست عملیاتی که قرار است از پروژه خواسته شود:

- **عملیات عمومی**:

۱. دریافت لیست کتاب ها، روزنامه ها، مجلات مختص به یک نویسنده
۲. دریافت لیست کتاب ها، روزنامه ها، مجلات مختص به یک ناشر
۳. دریافت لیست کتاب ها، روزنامه ها، مجلات با جست و جوی اسم
۴. دریافت لیست نویسنده ها، ناشران
۵. بروزرسانی (تغییر) اطلاعات فردی

- **عملیات مشتری**:

۱. دریافت لیست کتاب ها، روزنامه ها، مجلات مختص به خود مشتری
۲. بروزرسانی (اضافه کردن، حذف کردن، تغییر) کتاب ها، مجلات، روزنامه ها مختص به خود مشتری

- **عملیات مشترک مدیر و کارمندان**:

۱. دریافت لیست مشتریان
۲. بروزرسانی (اضافه کردن، حذف کردن، تغییر) کتاب ها، مجلات، روزنامه ها
۳. بروزرسانی (اضافه کردن، حذف کردن، تغییر) نویسنده، ناشر

- **عملیات مدیر**:

۱. دریافت لیست کارکنان
۲. بروزرسانی (اضافه کردن، حذف کردن، تغییر) کارمندان

ب فاز پنجم

ب_۱ ساختن جدول پروژه

با توجه به موجودیت ها و ارتباطات بین آنها که در فاز اول پروژه تعریف کردیم، جدوال را در محیط sql_server توصیف می کنیم:

۱. **Book**:

```
create table Book
(
   ISBN nvarchar(30) primary key,
   Bauthor nvarchar(30) FOREIGN KEY REFERENCES Author(author_id),
   Bcategory nvarchar(30) not null,
   Bname nvarchar(30) not null,
   shelf_number int not null,
   [row_number] int not null,
   rental_price float not null,
   price float not null,
   [status] bit not null,
   [count] int not null,
   libID int FOREIGN key references [Library](libID),
)
```

۲. **Magazine**:

```
create table Magazine
(
   MID nvarchar(30) primary key,
   Mcategory nvarchar(30) not null,
   Mname nvarchar(30) not null,
   shelf_number int not null,
   [row_number] int not null,
   rental_price float not null,
   price float not null,
   [status] bit not null,
   [count] int not null,
   libID int FOREIGN key references [Library](libID),
)
```

۳. Newspaper:

```
create table Newspaper
(
   NID nvarchar(30) primary key,
   Ncategory nvarchar(30) not null,
   Nname nvarchar(30) not null,
   shelf_number int not null,
   [row_number] int not null,
   rental_price float not null,
   price float not null,
   [status] bit not null,
   [count] int not null,
   libID int FOREIGN key references [Library](libID)
)
```

۴. manager Library:

```
create table Library_manager
(
    employee_id nvarchar(30) primary key,
    Mpassword nvarchar(3) null,
    [address] nvarchar(30) null,
    phone nvarchar(30) null,
    Mname nvarchar(30) null,
    position nvarchar(30) not null,
    salary float,
    [start_Date] Datetime2,
    end_Date Datetime2,
    libID int foreign key references [Library](libID)
)
```

۵. **Employee**:

```
create table Employee
(
    employee_id int primary key,
    Ename nvarchar(30) null,
    Epassword nvarchar(30) null,
    [address] nvarchar(30) null,
    phone nvarchar(30) null,
    position nvarchar(30) not null,
    salary float,
    [start_Date] Datetime2,
    end_Date Datetime2,
    libID int foreign key references [Library](libID)
)
```

۶. **Author**:

```
CREATE TABLE Author
(
   author_id nvarchar(30) primary key,
   Aname nvarchar(30) not null,
   Apassword nvarchar(30) not null,
   [address] nvarchar(30) not null,
   phone nvarchar(30) not null,
   salary float,
   [start_Date] Datetime2,
   end_Date Datetime2,
)
```

۷. **Publisher**:

```
CREATE TABLE Publisher
(
   publisher_id nvarchar(30) primary key,
   Pname nvarchar(30) not null,
   Ppassword nvarchar(30) not null,
   [address] nvarchar(30) not null,
   phone nvarchar(30) not null,
   [start_Date] Datetime2,
   end_Date Datetime2,
)
```

۸. **Customer**:

```
CREATE TABLE Customer
(
   customer_id nvarchar(30) primary key,
   Cname nvarchar(30) not null,
   Cpassword nvarchar(30) not null,
   [address] nvarchar(30) not null,
   phone nvarchar(30) not null,
   Csubscription nvarchar(30) not null,
   [start_Date] Datetime2,
   end_Date Datetime2
)
```

حال باتوجه به اینکه جدوال مربوط به موجودیت ها را به طور جداگانه تعریف کردیم، باید با استفاده از یکسری جدوال واسطه، ارتباطات بین این موجودیت ها را نیز برقرار کنیم.

۹. **Customer and Books for Relations** :$NtoN$

ارتباط بین کتاب و مشتری به صورت چند به چند چرا که هر مشتری می تواند چند کتاب را از کتابخانه قرض بگیرد و لذا، هر کتاب می تواند در دست چند مشتری باشد(چرا که از هر کتاب به تعدادی در کتابخانه وجود دارد).

```
CREATE TABLE CBook(
   customer_id nvarchar(30) FOREIGN key REFERENCES Customer(customer_id),
   ISBN nvarchar(30) FOREIGN key references Book(ISBN)
)
```

۱۰. **Customer and Magazine for Relations** $NtoN$:

با توجه به اینکه ممکن است از هر مجله به تعدادی در کتابخانه موجود باشد، پس ارتباط بین مشتری و مجله نیز به صورت چند به چند است، لذا باید با استفاده از یک جدول این ارتباط را برقرار کرد.

```
CREATE TABLE Cmagazine(
    customer_id nvarchar(30) FOREIGN key REFERENCES Customer(customer_id),
    MID nvarchar(30) FOREIGN key references Magazine(MID)
)
```

۱۱. **Customer and Magazine for Relations** $NtoN$:

همچنین ارتباط بین روزنامه ها و مشتری نیز به صورت چند به چند است و مشابه جدوال بالا، برای برقراری این ارتباط نیاز به یک جدول داریم.

```
CREATE TABLE CNewspaper(
    customer_id nvarchar(30) FOREIGN key REFERENCES Customer(customer_id),
    NID nvarchar(30) FOREIGN key references Newspaper(NID)
)
```

۱۲. **Library**:

```
create table [Library]
(
    libID int primary key,
    lib_name nvarchar(30) not null,
    [address] nvarchar(30) not null,
    phone_number nvarchar(30) not null,
    manager nvarchar(30) not null, -- to be replaced with a foreign key
    income float,
)
```

۱۳. **relations Publisher – Book** $NtoN$:

با توجه به اینکه هر کتاب می تواند توسط چند ناشر به چاپ برسد و هر انتشارات می تواند چندین کتاب را چاپ کند، پس رابطه ی بین کتاب و انتشارات به صورت چند به چند است. بنابراین، به جدول نیاز داریم تا این ارتباط را برقرار سازد:

```
CREATE TABLE BDetails(
    ISBN nvarchar(30) FOREIGN key REFERENCES Book(ISBN),
    publisher_id nvarchar(30) FOREIGN key references Publisher(publisher_id),
)
```

۱۴. **Relation Author – Publisher – Magazine** $NtoN$:

با توجه به اینکه هر انتشارات می تواند چندین مجله را منتشر کند و هر مجله میتواند توسط جندین انتشارات منتشر شود، پس ارتباط بین مجله و انتشارات به صورت چند به چند است. همچنین، در هر مجله چندین نویسنده می توانند مشارکت داشته باشند. پس، ارتباط بین نویسنده و مجلات به صورت چند به چند است. لذا برای برقراری این ارتباطات، باید جدولی را بسازیم

```
CREATE TABLE MDetails(
    MID nvarchar(30) FOREIGN key REFERENCES Magazine(MID),
    publisher_id nvarchar(30) FOREIGN key references Publisher(publisher_id),
    author_id nvarchar(30) FOREIGN key references Author(author_id)
)
```

۱۵. **Relation Author – Publisher – Newspaper** $NtoN$:

به طور مشابه، ارتباط روزنامه – انتشارات و روزنامه – نویسنده، چند به چند است. بنابراین، باید یک جدول برای برقراری این ارتباطات بسازیم

```
CREATE TABLE NDetails(
    NID nvarchar(30) FOREIGN key REFERENCES Newspaper(NID),
    publisher_id nvarchar(30) FOREIGN key references Publisher(publisher_id),
    author_id nvarchar(30) FOREIGN key references Author(author_id)
)
```

۱۶. **Relation Publisher – Author** $NtoN$:

ارتباط بین نویسنده و انتشارات نیز به صورت چند به چند است چرا که یک نویسنده می تواند در چند انتشارات کار کند و یک انتشارات می تواند چند نویسنده داشته باشد. پس، ارتباط بین نویسنده و انتشارات نیز به صورت چند به چند است. لذا به جدولی واسطه برای برقراری ارتباط نیاز داریم.

```
CREATE TABLE  PARelation(
    publisher_id nvarchar(30) FOREIGN key references Publisher(publisher_id),
    author_id nvarchar(30) FOREIGN key references Author(author_id)
)
```

ب ـ ۲ selects and Functions

(a) **get books list by Author**

```
create function get_book_by_name (@name_ nvarchar(30))
returns table
as
return select Book.Bcategory, Book.Bname from Book inner join Author as BA on BA.author_id = Book.Bauthor where BA.Aname = @name_
go
```

test:

```
select * from get_book_by_name(N'۱ نویسنده')
```

Results | Messages

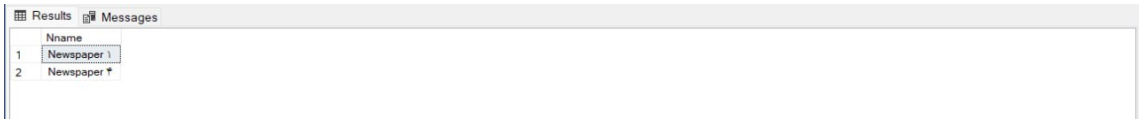
	Bcategory	Bname
1	داستان	کتاب ۱
2	داستان	کتاب ۳
3	تاریخی	کتاب ۴
4	داستان	کتاب ?

(b) **get newspaper list by Author**

```
create function get_newspaper_by_name (@name_ nvarchar(30))
returns table
as
return select Newspaper.Nname from Newspaper inner join NDetails on Newspaper.NID = NDetails.NID
 where NDetails.author_id in (Select author_id from Author where Author.Aname = @name_)

go
select * from get_newspaper_by_name(N'نویسنده ')
```

test:



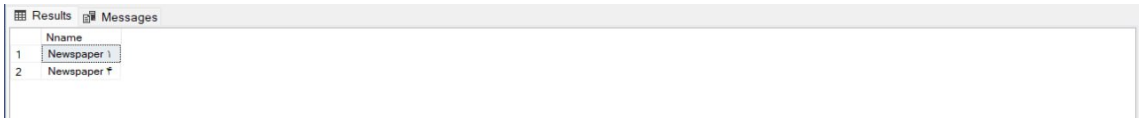
(c) **get magazine list by Author**

```
create function get_magazine_by_name (@name_ nvarchar(30))
returns table
as

return select Magazine.Mname from Magazine inner join MDetails on Magazine.MID = MDetails.MID where MDetails.author_id
in (Select author_id from Author where Author.Aname = @name_)

go
select * from get_magazine_by_name(N'نویسنده ')
```

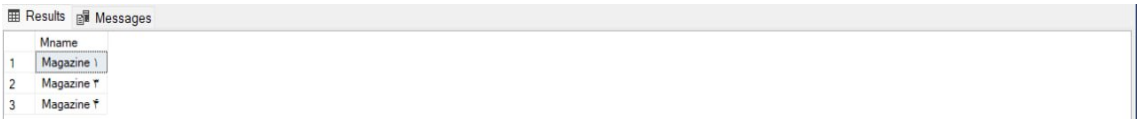
test:



(d) **get magazines list by Publisher**

```
create function get_mag_by_Pubname (@name_ nvarchar(30))
returns table
as
return select Magazine.Mname from Magazine inner join MDetails on Magazine.MID = MDetails.MID where MDetails.publisher_id in
(Select publisher_id from Publisher where Publisher.Pname = @name_)
go
select * from get_mag_by_Pubname(N'انتشارات فارسی')
```

test:



(e) **get newspaper list by Publisher**

```
create function get_news_by_Pubname (@name_ nvarchar(30))
returns table
as
return select Newspaper.Nname from Newspaper inner join NDetails on Newspaper.NID = NDetails.NID where NDetails.publisher_id in
(Select publisher_id from Publisher where Publisher.Pname = @name_)
go
select * from get_news_by_Pubname(N'انتشارات فارسی')
```

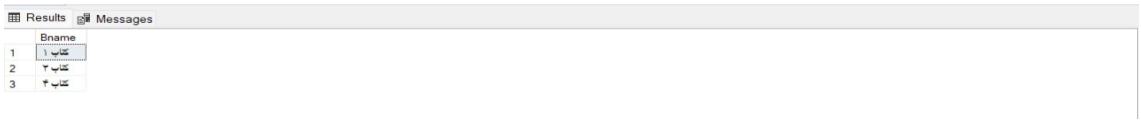
test:

(f) **get book by publisher**

```
create function get_Book_by_Pubname (@name_ nvarchar(30))
returns table
as
return select Book.Bname from Book inner join BDetails on Book.ISBN = BDetails.ISBN where BDetails.publisher_id in
(Select publisher_id from Publisher where Publisher.Pname = @name_)

go
select * from get_Book_by_Pubname(N'انتشارات فارسی')
```

test:



(g) **get book by book name**

```
create function get_book_by_book_name (@name_ nvarchar(30))
returns table
as
return select * from Book where Book.Bname = @name_
go
select * from get_book_by_book_name(N'کتاب ۱')
```

test:

(h) **get magazine by magazine name**

```
create function get_magazine_by_magazine_name (@name_ nvarchar(30))
returns table
as
return select * from Magazine where Magazine.Mname = @name_
go
select * from get_magazine_by_magazine_name('magazine1')
```

test:

Results Messages

	MID	Mcategory	Mname	shelf_number	row_number	rental_price	price	status	count	libID
1	MID008	مجله	magazine1	4	1	5500	22000	1	6	1

(i) **get newspaper by newspaper name**

```
create function get_newspaper_by_newspaper_name (@name_ nvarchar(30))
returns table
as
return select * from Newspaper where Newspaper.Nname = @name_
go
select * from get_newspaper_by_newspaper_name(N'Newspaper1')
```

test:

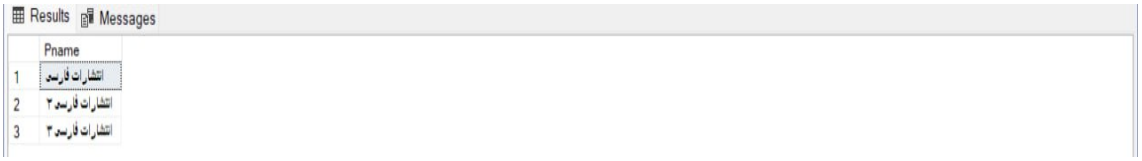
Results Messages

	NID	Ncategory	Nname	shelf_number	row_number	rental_price	price	status	count	libID
1	NID007	روزنامه	Newspaper1	2	1	3000	10000	1	10	1

(j) **get Publishser list**

```
select Publisher.Pname from Publisher
```

test:

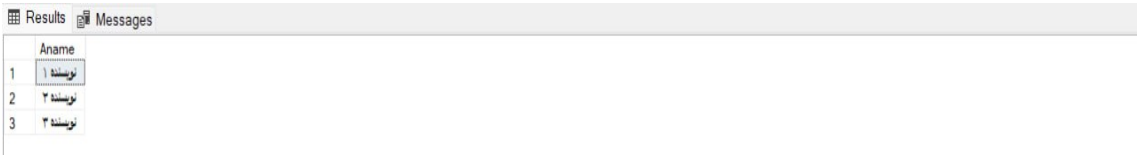


	Pname
1	انتشارات فارسی
2	انتشارات فارسی ۲
3	انتشارات فارسی ۳

(k) **get Author list**

```
select Author.Aname from Author
```

test:



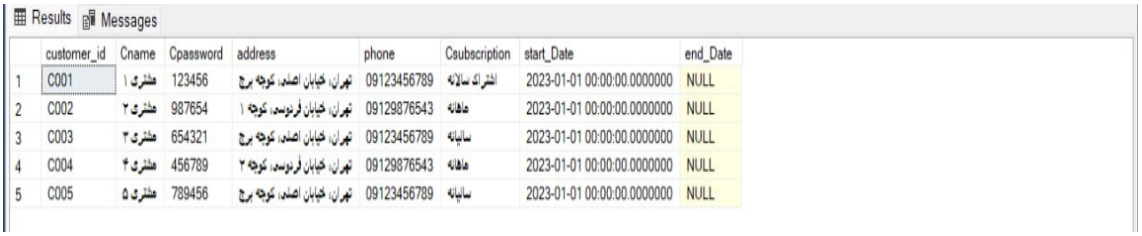
	Aname
1	نویسنده ۱
2	نویسنده ۲
3	نویسنده ۳

(l) **get Customer list**

```
create function get_customer_list()
returns table
as
    return select * from Customer
go

select * from get_customer_list()
```

test:

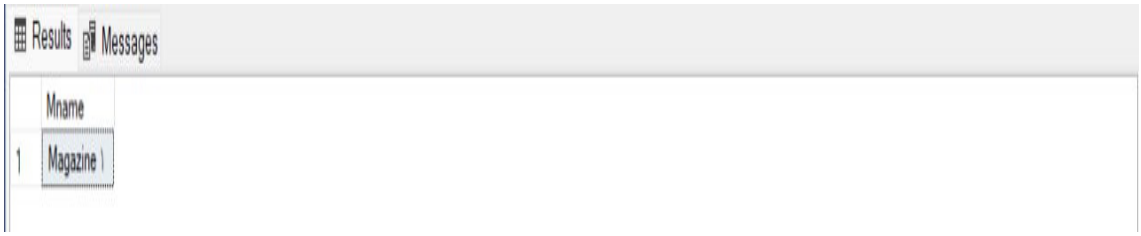


	customer_id	Cname	Cpassword	address	phone	Csubscription	start_Date	end_Date
1	C001	مشتری ۱	123456	تهران، خیابان اصلی، کوچه برج	09123456789	اشتراک سالانه	2023-01-01 00:00:00.0000000	NULL
2	C002	مشتری ۲	987654	تهران، خیابان فرعی، کوچه ۱	09129876543	ماهانه	2023-01-01 00:00:00.0000000	NULL
3	C003	مشتری ۳	654321	تهران، خیابان اصلی، کوچه برج	09123456789	سالیانه	2023-01-01 00:00:00.0000000	NULL
4	C004	مشتری ۴	456789	تهران، خیابان فرعی، کوچه ۲	09129876543	ماهانه	2023-01-01 00:00:00.0000000	NULL
5	C005	مشتری ۵	789456	تهران، خیابان اصلی، کوچه برج	09123456789	سالیانه	2023-01-01 00:00:00.0000000	NULL

(m) **get Customers Magazines**

```
create function get_Customer_mag
(@id nvarchar(30))
returns table
as
  return select Magazine.Mname from Magazine inner join Cmagazine on Magazine.MID = Cmagazine.MID where Cmagazine.customer_id = @id
go
select * from get_Customer_mag('C001')
```

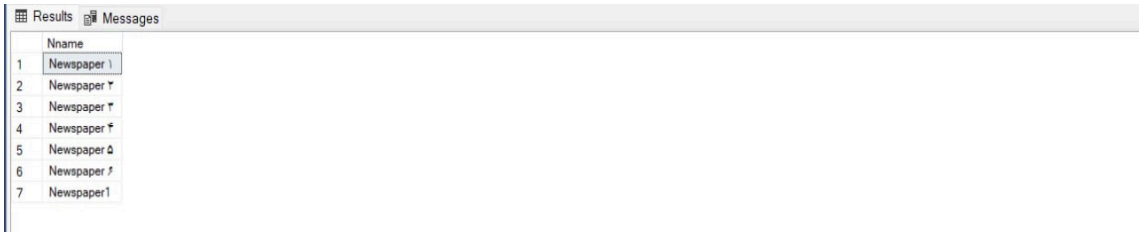
test:



(n) **get Customers Newspaper**

```
create function get_Customer_news
(@id nvarchar(30))
returns table
as
  return select Newspaper.Nname from Newspaper inner join CNewspaper on Newspaper.NID = Newspaper.NID where CNewspaper.customer_id = @id
go
select * from get_Customer_news('C001')
```

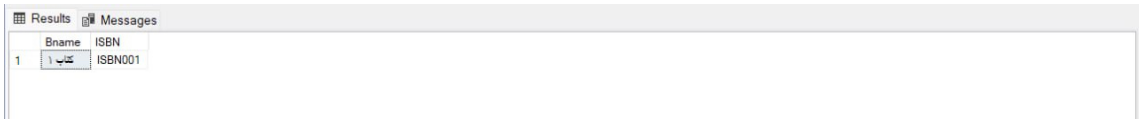
test:



(o) **get Customers Books**

```
create function get_Customer_book
(@id nvarchar(30))
returns table
as
  return select Book.Bname, Book.ISBN from Book inner join CBook on Book.ISBN = CBook.ISBN where CBook.customer_id = @id
go
select * from get_Customer_book('C001')
```

test:



ب-۳ Procedures

procedures for updating information data:

(a) **update customer data**

you can update your information data such as: 1. changing password 2. change your account name 3. change your address and 4. change your phone. Also, you can change your subscription.

```
create proc update_customer_info
@id nvarchar(30),
@cname nvarchar(30),
@newPass nvarchar(30),
@newPhone nvarchar(30),
@newAdd nvarchar(30),
@newEsh nvarchar(30)
as
begin
if @cname is not null begin
  update Customer set Cname = @cname where Customer.customer_id = @id
end
if @newPass is not null begin
  update Customer set Cpassword = @newPass where Customer.customer_id = @id
end
if @newPhone is not null begin
  update Customer set phone = @newPhone where Customer.customer_id = @id
end
if @newAdd is not null begin
  update Customer set [address] = @newAdd where Customer.customer_id = @id
end

if @newEsh is not null begin
  update Customer set Csubscription = @newEsh where Customer.customer_id = @id
end
end
go

Exec update_customer_info 'C001', N'۱ مشتری',null ,null , null, null
```

Employees and manager can update their information data such as: 1. changing password 2. changing account name 3. change address and 4. change phone.

(b) **update employee data**

```
create proc update_employee_info
@id nvarchar(30),
@Ename nvarchar(30),
@newPass nvarchar(30),
@newPhone nvarchar(30),
@newAdd nvarchar(30)
as
begin
if @Ename is not null begin
  update Employee set Ename = @Ename where Employee.employee_id = @id
end
if @newPass is not null begin
  update Employee set Epassword = @newPass where Employee.employee_id = @id
end
if @newPhone is not null begin
  update Employee set phone = @newPhone where Employee.employee_id = @id
end
if @newAdd is not null begin
  update Employee set [address] = @newAdd where Employee.employee_id = @id
end
end
go

Exec update_employee_info '1001', N'کارمند',null ,null , null
```

(c) **update manager data**

```
create proc update_manager_info
@id nvarchar(30),
@mname nvarchar(30),
@newPass nvarchar(30),
@newPhone nvarchar(30),
@newAdd nvarchar(30)
as
begin
if @mname is not null begin
  update Library_manager set Mname = @mname where Library_manager.employee_id = @id
end
if @newPass is not null begin
  update Library_manager set Mpassword = @newPass where Library_manager.employee_id = @id
end
if @newPhone is not null begin
  update Library_manager set phone = @newPhone where Library_manager.employee_id = @id
end
if @newAdd is not null begin
  update Library_manager set [address] = @newAdd where Library_manager.employee_id = @id
end
end
go


Exec update_manager_info 'E001', N'مدیر کتابخانه',null ,null , null
```

customers can add books, magazines and newspapers to their book, magazines and newspapers. But, there is limit for adding books and newspapers and magazines: if you want add book, if the library does not have the book, so you cannot have it. Unless you can(this limit is for all book, newspapers, magazines.)

(d) **add Books to customers books**

```
create proc add_books
@id nvarchar(30),
@ISBN nvarchar(30)
as
begin
  if ( select ISBN from CBook where CBook.customer_id = @id and CBook.ISBN = @ISBN) is not null begin
    print N'you already have this book'
  end
  else if (select [count] from Book where Book.ISBN = @ISBN) <= 0 begin
    print N'we dont have enough book'
  end
  else begin
    insert into Cbook values(@id, @ISBN)
    update Book set[count] = [count] - 1 where ISBN = @ISBN
  end
end
go
Exec add_books 'C001', 'ISBN001'
```

(e) **add newspaper to customers newspaper**

```
create proc add_newspaper
@id nvarchar(30),
@NID nvarchar(30)
as
begin
  if ( select NID from CNewspaper where CNewspaper.customer_id = @id and CNewspaper.NID = @NID) is not null begin
    print N'you already have this newspaper'
  end
  else if (select [count] from Newspaper where Newspaper.NID = @NID) <= 0 begin
    print N'we dont have enough book'
  end
  else begin
    insert into CNewspaper(customer_id, NID) values(@id, @NID)
    update Newspaper set [count] = [count] - 1 where Newspaper.NID = @NID
  end
end
go
Exec add_newspaper 'C001', 'NID001'
```

(f) add magazines to customers magazines

```
create proc add_mag
@id nvarchar(30),
@MID nvarchar(30)
as
begin
  if ( select MID from Cmagazine where Cmagazine.customer_id = @id and Cmagazine.MID = @MID) is not null begin
    print N'you already have this magazine'
  end
  else if (select [count] from Magazine where Magazine.MID = @MID) <= 0 begin
    print N'we dont have enough book'
  end
  else begin
    insert into Cmagazine values(@id, @MID)
    update Magazine set[count] = [count] - 1 where MID = @MID
  end
end
go
Exec add_mag 'C001', 'MID002'
```

customers can delete their books, magazines, newspapers.

(g) **delete book from customers books**

```
create proc delete_book
@id nvarchar(30),
@ISBN nvarchar(30)
as
begin
  if ( select @ISBN from CBook where CBook.customer_id = @id and CBook.ISBN = @ISBN) is not null begin
    delete from CBook where CBook.customer_id = @id and CBook.ISBN = @ISBN
    update Book set [count] = [count] + 1 where ISBN = @ISBN
  end
  else begin
    print N'you dont have this book'
  end
end
go
Exec delete_book 'C001', 'ISBN002'
```

(h) **delete newspaper from customers newspaper**

```
create proc delete_newspaper
@id nvarchar(30),
@NID nvarchar(30)
as
begin
  if ( select @NID from CNewspaper where CNewspaper.customer_id = @id and CNewspaper.NID = @NID) is not null begin
    delete from CNewspaper where CNewspaper.customer_id = @id and CNewspaper.NID = @NID
    update Newspaper set [count] = [count] + 1 where NID = @NID
  end
  else begin
    print N'you dont have this book'
  end
end
go
Exec delete_newspaper 'C001', 'ISBN002'
```

(i) **delete magazines from customers magazines**

```
create proc delete_mag
@id nvarchar(30),
@MID nvarchar(30)
as
begin
  if ( select MID from Cmagazine where Cmagazine.customer_id = @id and Cmagazine.MID = @MID) is not null begin
    delete from Cmagazine where Cmagazine.customer_id = @id and Cmagazine.MID = @MID
    update Magazine set [count] = [count] + 1 where MID = @MID
  end
  else begin
    print N'you dont have this magazine'
  end
end
go
Exec delete_mag 'C001', 'MID001'
```

Employees and manager can add or delete Author and Publisher.

(j) add Publisher

```
create proc add_publisher
@publisher_id nvarchar(30),
@Pname nvarchar(30),
@Ppassword nvarchar(30),
@address nvarchar(30),
@phone nvarchar(30)
as
begin
  if not exists(select publisher_id from Publisher where publisher_id = @publisher_id) begin
    insert into Publisher (publisher_id, Pname, Ppassword, [address], phone)
    values(@publisher_id, @Pname, @Ppassword, @address, @phone)
  end
  else begin
    print N'there is already a publisher with this ID'
  end
end

Exec add_publisher 'P001','فارسی انتشارات', '123456', 'پرچ کوچه اصلی، خیابان تهران،', '09123456789'
```

(k) add Author

```
create proc add_author
@author_id nvarchar(30),
@Aname nvarchar(30),
@Apassword nvarchar(30),
@address nvarchar(30),
@phone nvarchar(30)
as
begin
  if not exists(select Author.author_id from Author where author_id = @author_id) begin
    insert into Author(author_id, Aname, Apassword, [address], phone)
    values(@author_id, @Aname, @Apassword, @address, @phone)
  end
  else begin
    print N'there is already a author with this ID'
  end
end
Exec add_author'A001','۱ نویسنده', '123456', 'پرچ کوچه اصلی، خیابان تهران،', '09123456789'
```

(l) **delete Publisher**

```
create proc delete_publisher
@publisher_id nvarchar(30)
as
begin
  if exists(select Publisher.publisher_id from Publisher where publisher_id = @publisher_id) begin
    delete from Publisher where publisher_id = @publisher_id
  end
  else begin
    print N'there is no publisher with this ID'
  end
end

Exec delete_publisher 'A001','۱ نویسنده', '123456', 'پرچ کوچه اصلی، خیابان تهران', '09123456789'
```

(m) **delete Author**

```
create proc delete_author
@author_id nvarchar(30)
as
begin
  if exists(select Author.author_id from Author where author_id = @author_id) begin
    delete from Author where author_id = @author_id
  end
  else begin
    print N'there is no one with this ID'
  end
end

Exec delete_author'A001','۱ نویسنده', '123456', 'پرچ کوچه اصلی، خیابان تهران', '09123456789'
```

Employees, Managers, Authors and Publishers can update their information data. such as: 1. changing account name, 2. changing password, 3. changing phone, 4. changing address.

(n) **update publisher information data**

```
create proc update_publisher_info
@publisher_id nvarchar(30),
@Pname nvarchar(30),
@newPass nvarchar(30),
@newPhone nvarchar(30),
@newAdd nvarchar(30)
as
begin
if @Pname is not null begin
  update Publisher set Pname = @Pname where Publisher.publisher_id = @publisher_id
end
if @newPass is not null begin
  update Publisher set Ppassword = @newPass where Publisher.publisher_id = @publisher_id
end
if @newPhone is not null begin
  update Publisher set phone = @newPhone where Publisher.publisher_id = @publisher_id
end
if @newAdd is not null begin
  update Publisher set [address] = @newAdd where Publisher.publisher_id = @publisher_id
end
end
go
Exec update_manager_info 'E001', N'کتابخانه مدیر',null ,null , null
```

(o) **update author information data**

```
create proc update_author_info
@author_id nvarchar(30),
@Aname nvarchar(30),
@newPass nvarchar(30),
@newPhone nvarchar(30),
@newAdd nvarchar(30)
as
begin
if @Aname is not null begin
  update Author set Aname = @Aname where Author.author_id = @author_id
end
if @newPass is not null begin
  update Author set Apassword = @newPass where Author.author_id = @author_id
end
if @newPhone is not null begin
  update Author set phone = @newPhone where Author.author_id = @author_id
end
if @newAdd is not null begin
  update Author set [address] = @newAdd where Author.author_id = @author_id
end
end
go
Exec update_customer_info 'A001', N'۵ نویسنده',null ,null , null, null
```

(p) update employee information data

```
create proc update_employee_info
@id nvarchar(30),
@Ename nvarchar(30),
@newPass nvarchar(30),
@newPhone nvarchar(30),
@newAdd nvarchar(30)
as
begin
if @Ename is not null begin
  update Employee set Ename = @Ename where Employee.employee_id = @id
end
if @newPass is not null begin
  update Employee set Epassword = @newPass where Employee.employee_id = @id
end
if @newPhone is not null begin
  update Employee set phone = @newPhone where Employee.employee_id = @id
end
if @newAdd is not null begin
  update Employee set [address] = @newAdd where Employee.employee_id = @id
end
end
go
Exec update_employee_info '1001', N'ا کارمند',null ,null , null
```

(q) update manager information data

```
create proc update_manager_info
@id nvarchar(30),
@mname nvarchar(30),
@newPass nvarchar(30),
@newPhone nvarchar(30),
@newAdd nvarchar(30)
as
begin
if @mname is not null begin
  update Library_manager set Mname = @mname where Library_manager.employee_id = @id
end
if @newPass is not null begin
  update Library_manager set Mpassword = @newPass where Library_manager.employee_id = @id
end
if @newPhone is not null begin
  update Library_manager set phone = @newPhone where Library_manager.employee_id = @id
end
if @newAdd is not null begin
  update Library_manager set [address] = @newAdd where Library_manager.employee_id = @id
end
end
go
Exec update_manager_info 'E001', N'کتابخانه مدیر',null ,null , null
```

Manager can add or delete an employee. On add, first we check if an employee with this id exist or not? is it exists, manager cannot add the employee. Otherwise, manager can add the employee.On delete, first we have to check that the employee exists or not? if yes, Manager can remove the employee. Otherwise, manager cannot remove it.

(r) **add employee**

```
create proc add_employee
@employee_id nvarchar(30),
@Ename nvarchar(30),
@Epassword nvarchar(30),
@address nvarchar(30),
@phone nvarchar(30),
@position nvarchar(30)
as
begin
  if not exists(select Employee.employee_id from Employee where employee_id = @employee_id) begin
    insert into Employee(employee_id, Ename, Epassword, [address], phone, position)
    values(@employee_id, @Ename, @Epassword, @address, @phone, @position)
  end
  else begin
    print N'there is already a Employee with this ID'
  end
end
Exec add_author'A001','ا نویسنده', '123456', 'برج کوچه اصلی، خیابان تهران', '09123456789'
```

(s) **delete employee**

```
create proc delete_employee
@employee_id nvarchar(30)
as
begin
  if exists(select Employee.employee_id from Employee where employee_id = @employee_id) begin
    delete from Employee where employee_id = @employee_id
  end
  else begin
    print N'there is no employee with this ID'
  end
end

Exec delete_employee 'A001','ا نویسنده', '123456', 'برج کوچه اصلی، خیابان تهران', '09123456789'
```

Manager and Employee can add and delete books, newspapers and magazines. On add if the book, newspaper, magazine exists, just count up Otherwise we create it and add it to the related table. On delete if the book, newspaper and magazine exists we count down. Other wise "There is no (book, magazine, newpaper) to delete" is displayed.

(t) **add Book to total**

```
create proc add_books_total
@ISBN nvarchar(30),
@Bauthor nvarchar(30),
@Bcategory nvarchar(30),
@Bname nvarchar(30),
@shelf_number int,
@row_number int,
@rental_price int,
@price int,
@status bit,
@count_ int,
@libID int
as
begin
    if ( select Book.[count] from Book where Book.ISBN = @ISBN  ) > 0 begin
        update Book set [count] = [count] + 1 where Book.ISBN = @ISBN
    end
    else begin
        insert into Book(ISBN, Bauthor, Bcategory, Bname, shelf_number, [row_number], rental_price, price, [status], [count])
        values(@ISBN, @Bauthor, @Bcategory, @Bname, @shelf_number, @row_number, @rental_price, @price, @status, @count_)
    end
end
go
Exec add_books_total 'ISBN007', 'A001', 'داستان', 'کتاب',  1, 1, 10000, 50000, 1, 10, 1
```

(u) **delete Book from total**

```
create proc delete_books_total
@ISBN nvarchar(30)
as
begin
    if ( select Book.[count] from Book where Book.ISBN = @ISBN  ) > 0 begin
        update Book set [count] = [count] - 1 where Book.ISBN = @ISBN
    end
    else begin
        print N'there is not enough book to delete'
    end
end
go
Exec delete_books_total 'ISBN001'
```

(v) **add newspaper to total**

```
create proc add_newspaper_total
@NID nvarchar(30),
@Ncategory nvarchar(30),
@Nname nvarchar(30),
@shelf_number int,
@row_number int,
@rental_price int,
@price int,
@status bit,
@count_ int,
@libID int
as
begin
    if ( select Magazine.[count] from Magazine where Magazine.MID = @NID    ) > 0 begin
        update Magazine set [count] = [count] + 1 where Magazine.MID = @NID
    end
    else begin
        insert into Magazine(MID, Mcategory, Mname, shelf_number, [row_number], rental_price, price, [status], [count], libID)
        values(@NID, @Ncategory, @Nname, @shelf_number, @row_number, @rental_price, @price, @status, @count_, @libID)
    end
end
go
Exec add_newspaper_total 'MID007', 'مجله', 'Magazine \', 2, 1, 10000, 50000, 1, 10, 1
```

(w) **delete newspaper from total**

```
create proc delete_newspaper_total
@NID nvarchar(30)
as
begin
    if ( select Newspaper.[count] from Newspaper where Newspaper.NID = @NID ) > 0 begin
        update Newspaper set [count] = [count] - 1 where Newspaper.NID = @NID
    end
    else begin
        print N'there is not enough newpaper to delete'
    end
end
go
Exec delete_newspaper_total 'NID001'
```

(x) **add magazine to total**

```
create proc add_magazine_total
@MID nvarchar(30),
@Mcategory nvarchar(30),
@Mname nvarchar(30),
@shelf_number int,
@row_number int,
@rental_price int,
@price int,
@status bit,
@count_ int,
@libID int
as
begin
    if ( select Magazine.[count] from Magazine where Magazine.MID = @MID    ) > 0 begin
        update Magazine set [count] = [count] + 1 where Magazine.MID = @MID
    end
    else begin
        insert into Magazine(MID, Mcategory, Mname, shelf_number, [row_number], rental_price, price, [status], [count], libID)
        values(@MID, @Mcategory, @Mname, @shelf_number, @row_number, @rental_price, @price, @status, @count_, @libID)
    end
end
go
Exec add_magazine_total 'MID007', 'مجله', 'Magazine \', 2, 1, 10000, 50000, 1, 10, 1
```

(y) **delete magazine from total**

```
create proc delete_magazine_total
@MID nvarchar(30)
as
begin
    if ( select Magazine.[count] from Magazine where Magazine.MID = @MID    ) > 0 begin
        update Magazine set [count] = [count] - 1 where Magazine.MID = @MID
    end
    else begin
        print N'there is not enough magazine to delete'
    end
end
go
Exec delete_magazine_total 'MID001'
```

4- Triggers

(a) **Book numbers Trigger**

Whenever a book is added to a customer's books (i.e. a customer borrows a new book), the alpha table changes. Using this Trigger, the number of books borrowed by the customer is displayed.

```
create trigger Book_Num_Trigger
ON CBook
After Insert
as
begin
DECLARE @count int
SELECT @count = Count(*)FROM CBook WHERE customer_id IN (SELECT customer_id FROM inserted)
print(@count)
end
```

This is also the case for one customer's magazines and newspapers. That is, when a customer borrows a new magazine or newspaper, using the triggers below the number of magazines, the customer's newspapers up to this point are displayed.

(b) **Magazines and newspaper numbers Trigger**

```
Create trigger Magazine_Num_Trigger
ON Cmagazine
After Insert
as
begin
DECLARE @count int
SELECT @count = Count(*)FROM Cmagazine WHERE customer_id IN (SELECT customer_id FROM inserted)
print(@count)
end
```

```
Create trigger Newspaper_Num_Trigger
ON CNewspaper
After Insert
as
begin
DECLARE @count int
SELECT @count = Count(*)FROM CNewspaper WHERE customer_id IN (SELECT customer_id FROM inserted)
print(@count)
end
```