پروژه پایگاه داده

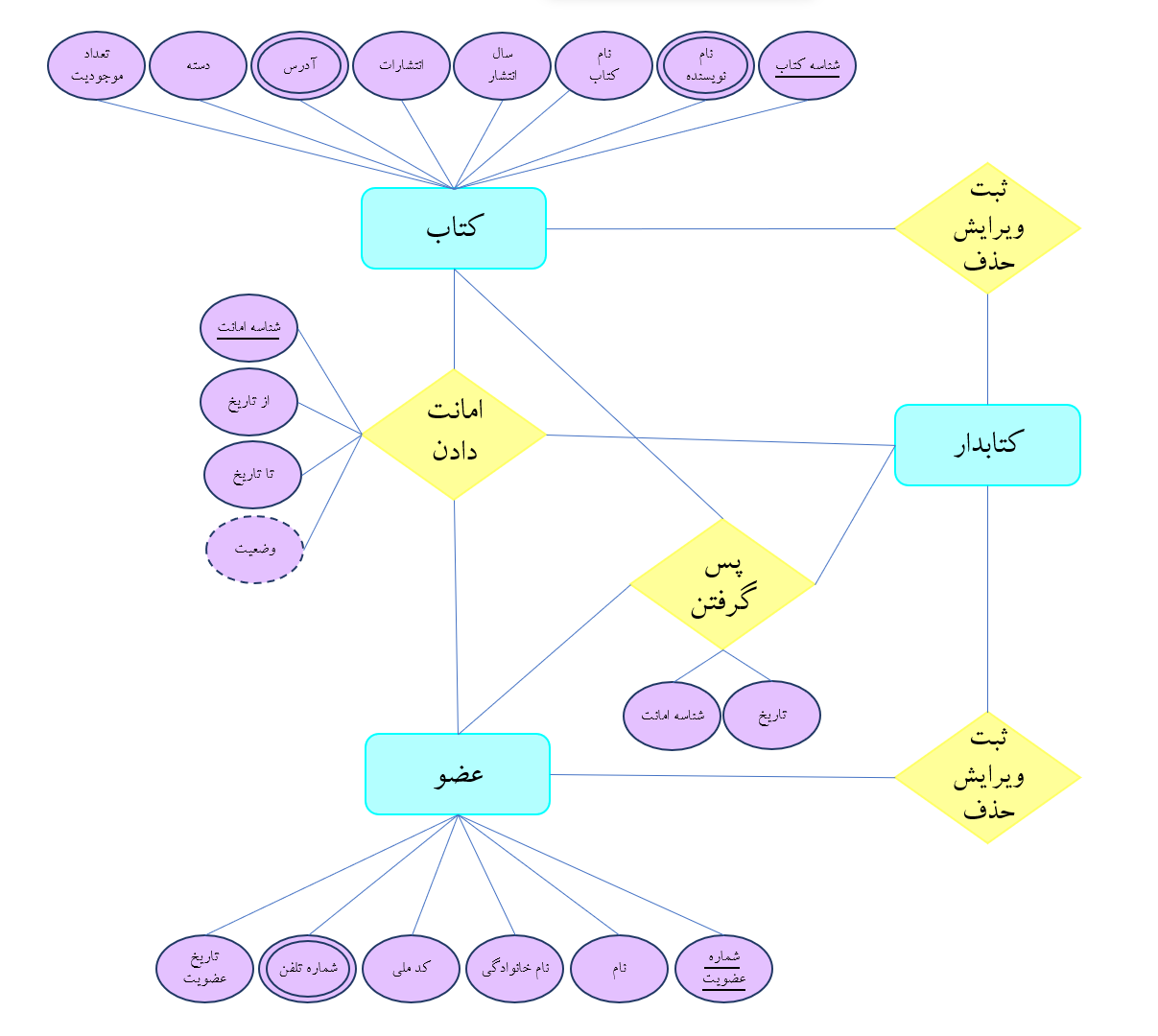
معصومه شیخان - 990122680038

گلاره صداقت - 990122680029

گام اول در طراحی دیتابیس کتابخانه، شناخت موجودیت های کتابخانه است. این موجودیت ها وابسته به درخواست و نیازهای درخواست کننده دارد. لازم است بدانیم که چه اطلاعاتی باید ذخیره سازی شوند.

مهم ترین و اساسی ترین موجودیت های کتابخانه عبارتند از کتاب و اعضای کتابخانه. به واسته این دو موجودیت کمکی و صفت های آنها، تعدادی موجودیت دیگر تعریف و کشف میشوند.

برای مدیریت یک کتابخانه و امانات آن و اعضای آن و کتب آن، با توجه به اینکه کتابدار از این سیستم استفاده میکند، نمودار ER و روابط بین این موجودیت ها، به صورت زیر قابل بیان هستند.



بنابراین برای ثبت اطلاعات به 3 جدول اصلی اعضا، کتاب ها و امانات نیاز داریم.

1.جدول اعضا :

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **id** | **نام** | **نام خانوادگی** | **کد ملی** | **شماره تلفن** | **تاریخ عضویت** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

2.جدول کتاب ها :

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **id** | **نام** | **نویسنده** | **انتشارات** | **جلد** | **سال انتشار** | **دسته** | **آدرس** | **تعداد موجودیت** |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |

3.جدول امانات :

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **id** | **کتاب** | **عضو** | **از تاریخ** | **تا تاریخ** | **تاریخ برگشت** | **وضعیت** |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

اما ذخیره اطلاعات به این صورت در دیتابیس، موجب افزونگی و بروز خطا میشود. بنابراین لازم است این سه جدول را به نرم نرمال 3NF تبدیل کنیم. که برای این نرمال سازی باید ابتدا در فرم نرمال 1NF و 2NF قرار گرفت.

پس از نرمال سازی جداول به صورت زیر خواهند بود :

**1.جداول اعضا :**

1.1.جدول اعضا :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **id** | **نام** | **نام خانوادگی** | **کد ملی** | **تاریخ عضویت** |
|  |  |  |  |  |
|  |  |  |  |  |

1.2.جدول شماره تلفن ها :

|  |  |
| --- | --- |
| **id** | **شماره تلفن** |
|  |  |
|  |  |

**2. جداول کتاب ها :**

2.1.جدول کتاب ها :

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **id** | **نام** | **id انتشارات** | **جلد** | **سال انتشار** | **Id دسته** | **Id آدرس** | **تعداد موجودیت** |
|  |  |  |  |  |  |  |  |

2.2.جدول انتشارات ها :

|  |  |
| --- | --- |
| **id** | **انتشارات** |
|  |  |

2.3.جدول نویسنده ها :

|  |  |
| --- | --- |
| **id** | **نویسنده** |
|  |  |

2.4.جدول نویسنده ی کتاب ها :

|  |  |
| --- | --- |
| **Id کتاب** | **Id نویسنده** |
|  |  |

2.5.جدول دسته ها :

|  |  |
| --- | --- |
| **id** | **دسته** |
|  |  |

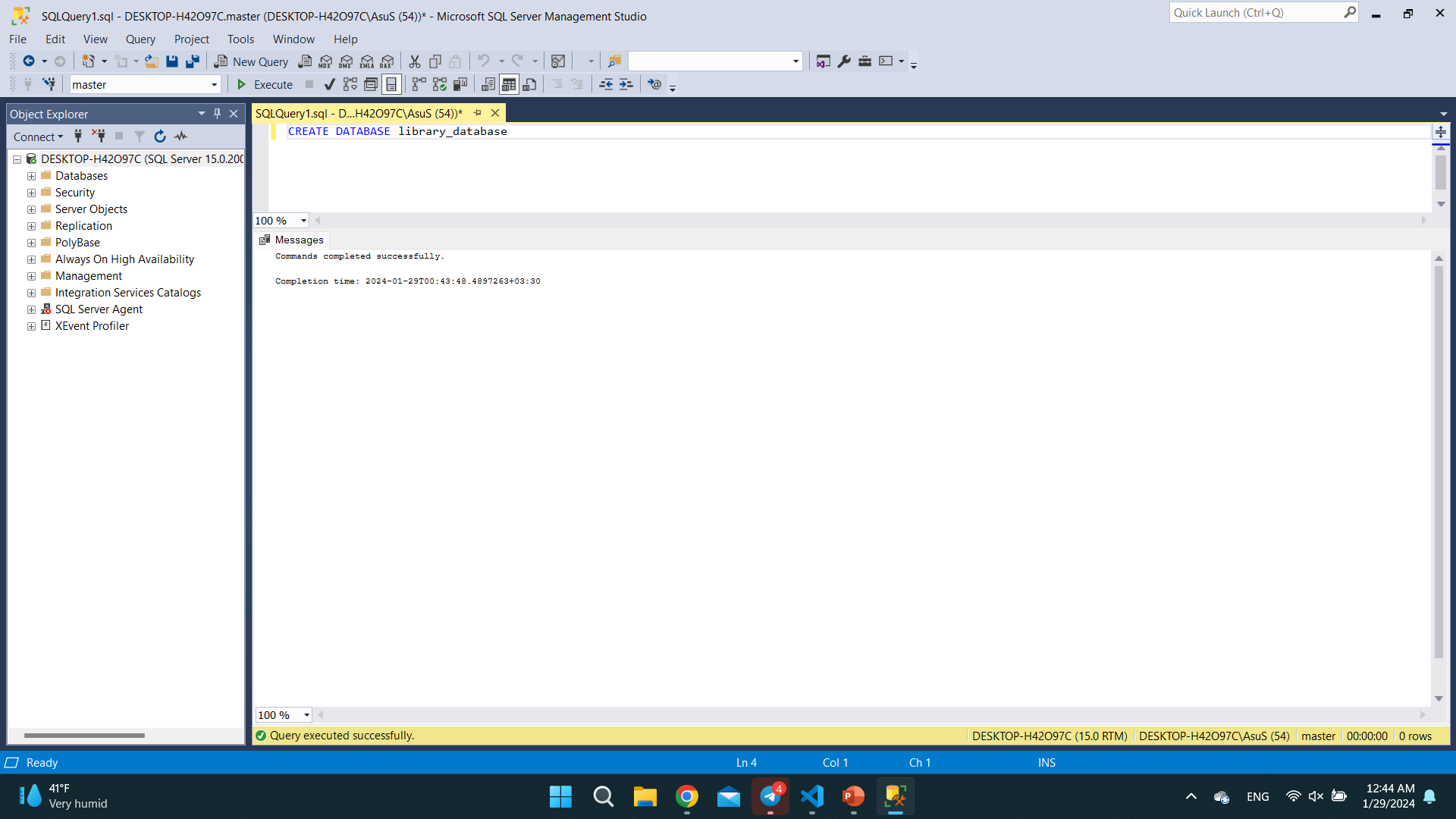
2.6.جدول آدرس ها :

|  |  |  |  |
| --- | --- | --- | --- |
| **id** | **طبقه** | **راهرو** | **قفسه** |
|  |  |  |  |

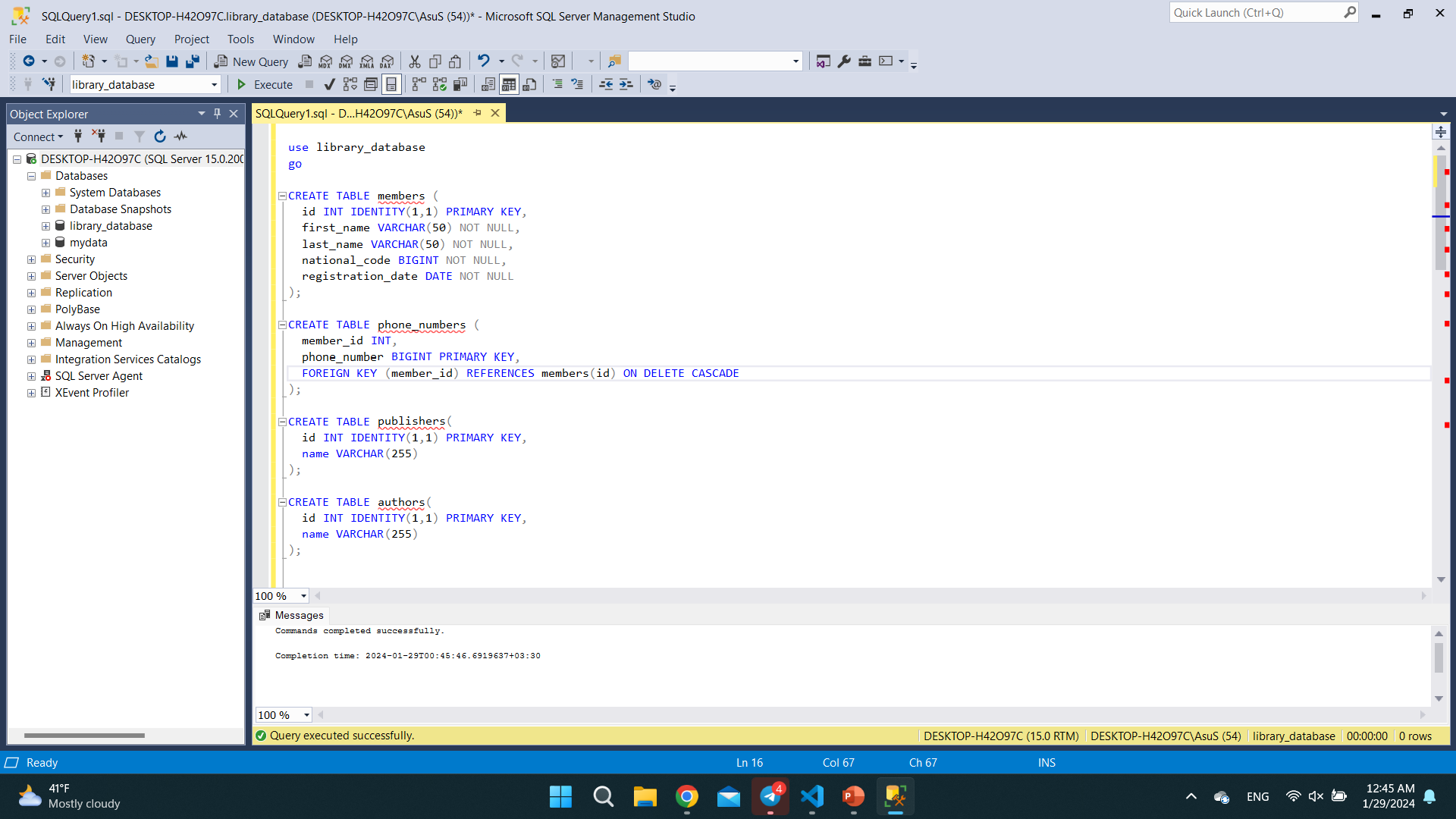
3.جدول امانات :

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **id** | **کتاب** | **عضو** | **از تاریخ** | **تا تاریخ** | **تاریخ برگشت** | **وضعیت** |
|  |  |  |  |  |  |  |

برای ساخت این جداول در SQL Server ابتدا دیتابیس خود را با دستور زیر میسازیم.



سپس جداول تعریف شده را در دیتابیس اضافه میکنیم.



جداول تعریف شده به صورت زیر هستند.

* 1. جدول اعضا، شامل صفات متعلق به اعضا است.

CREATE TABLE members (

id INT IDENTITY(1,1) PRIMARY KEY,

first\_name VARCHAR(50) NOT NULL,

last\_name VARCHAR(50) NOT NULL,

national\_code BIGINT NOT NULL,

registration\_date DATE NOT NULL

);

* 1. در جدول شماره تلفن ها، شماره تلفن متناظر با هر عضو، ثبت شده. هر فرد میتواند بیش از یک شماره تلفن داشته باشد اما هر شماره تلفن میتواند تنها متعلق به یک نفر باشد.

CREATE TABLE phone\_numbers (

member\_id INT,

phone\_number BIGINT PRIMARY KEY,

FOREIGN KEY (member\_id) REFERENCES members(id) ON DELETE CASCADE

);

2.1. جدول کتاب ها شامل اطلاعات مستقیم و فرعی با شناسه مورد نظر.

CREATE TABLE books (

id INT IDENTITY(1,1) PRIMARY KEY,

title VARCHAR(255) NOT NULL,

publisher\_id INT,

volume INT,

publication\_year INT,

category\_id INT,

address\_id INT,

quantity INT,

FOREIGN KEY (publisher\_id) REFERENCES publishers(id),

FOREIGN KEY (address\_id) REFERENCES addresses(id),

FOREIGN KEY (category\_id) REFERENCES categories(id)

);

2.2. جدول انتشارات ها و شناسه انها.

CREATE TABLE publishers(

id INT IDENTITY(1,1) PRIMARY KEY,

name VARCHAR(255)

);

2.3.جدول نویسنده ها و شناسه آنها.

CREATE TABLE authors(

id INT IDENTITY(1,1) PRIMARY KEY,

name VARCHAR(255)

);

2.4.جدول نویسنده ی کتاب ها. زیرا هر کتاب میتواند بیش از یک نویسنده داشته باشد و هر نویسنده ممکن است بیش از یک کتاب داشته باشد.

CREATE TABLE book\_authors (

author\_id INT,

book\_id INT,

PRIMARY KEY (author\_id, book\_id),

FOREIGN KEY (author\_id) REFERENCES authors(id),

FOREIGN KEY (book\_id) REFERENCES books(id) ON DELETE CASCADE

);

2.5.جدول دسته ها.

CREATE TABLE categories(

id INT IDENTITY(1,1) PRIMARY KEY,

name VARCHAR(255)

);

2.6. جدول آدرس ها که در این کتابخانه به صورت پیش فرض 3 طبقه و در هر طبقه 4 راهرو و در هر راهرو 10 قفسه وجود دارد و برای دسترسی آسان کتابدار به کتاب هاست. این جدول به صورت دیفالت دارای 120 آدرس خواهد بود که برای جلوگیری از افزونگی داده ها آنها را ثبت خواهیم کرد.

CREATE TABLE addresses(

id INT IDENTITY(1,1) PRIMARY KEY,

floor\_number INT,

corridor\_letter CHAR(1),

shelf\_number INT

);

3.جدول امانات شامل کتاب مد نظر و عضو مورد نظر و تاریخ های مهم.

CREATE TABLE loans (

id INT IDENTITY(1,1) PRIMARY KEY,

book\_id INT ,

member\_id INT ,

loan\_date DATE NOT NULL,

due\_date DATE NOT NULL,

return\_date DATE,

FOREIGN KEY (member\_id) REFERENCES members(id) ON DELETE SET DEFAULT,

FOREIGN KEY (book\_id) REFERENCES books(id) ON DELETE SET DEFAULT

);

پس از ساخت جداول، کد برنامه با زبان Python و تکنولوژی های Flask و کد فرانت اند برنامه با html , css پیاده سازی شده است.

ابتدا با کد پایتون به دیتابیس متصل میشویم.

*def* connect\_to\_database():

    connection\_string = (

        "Driver={ODBC Driver 17 for SQL Server};"

        "Server=DESKTOP-H42O97C;"

        "Database=library\_database;"

        "Trusted\_Connection=yes;"

    )

    connection = pyodbc.connect(connection\_string)

    return connection

*def* create\_cursor():

    connection = connect\_to\_database()

    cursor = connection.cursor()

    return cursor

سپس برای اجرای هر کوئری دیتابیس، کافیست از تابع create\_cursor استفاده کنیم و با ساخت یک نشانگر جدید، کوئری مورد نظر را در دیتابیس اجرا کنیم.

هر صفحه از برنامه، مجموعه اکشن هایی دارد که به فراخوانی هر url، تابع مربوط به آن در بک اند فراخوانی میشود. در این تابع اطلاعاتی که صفحه باید نمایش دهد را با دستورات کوئری از دیتابیس استخراج میکند و به فرانت اند ارسال میکند. همچنان برخی از صفحات درخوایت های post ارسال میکنند. این توابع درخواست را دریافت میکنند و بر اساس فیلد های دریافتی، کوئری مد نظر را اجرا میکند. و سپس صفحه مورد نظر نمایش داده میشود.

چالش های پروژه :

1. برای نمایش برخی از اطلاعات مربوط به کتاب ها، به دلیل تعداد زیاد جداول مربوط به کتب، لازم به join کردن تعداد زیادی از جداول بود.
2. همچنین برای حذف یک عضو، لازم است توجه شود که شماره های مربوط به آن کاربر نیز از دیتابیس حذف شود که این مورد را با دیتور delete on cascade کنترل کردیم.
3. برخی از ورودی های جدول دیتابیس با تگ NOT NULL تعریف شده اند. لازم است این مورد در فرانت بررسی شود که ورودی های آن required باشد و فیلد خالی ارسال نشود.