**Final Project: Library Database Management System**

Andrei Fedotov, Kimberly Hoang, Mary Tess Joseph

Ira A. Fulton Schools of Engineering, Arizona State University

IFT 530: Advanced Database Management Systems

Professor Robert Rucker

April 30, 2021

**Final Project: Library Database Management System**

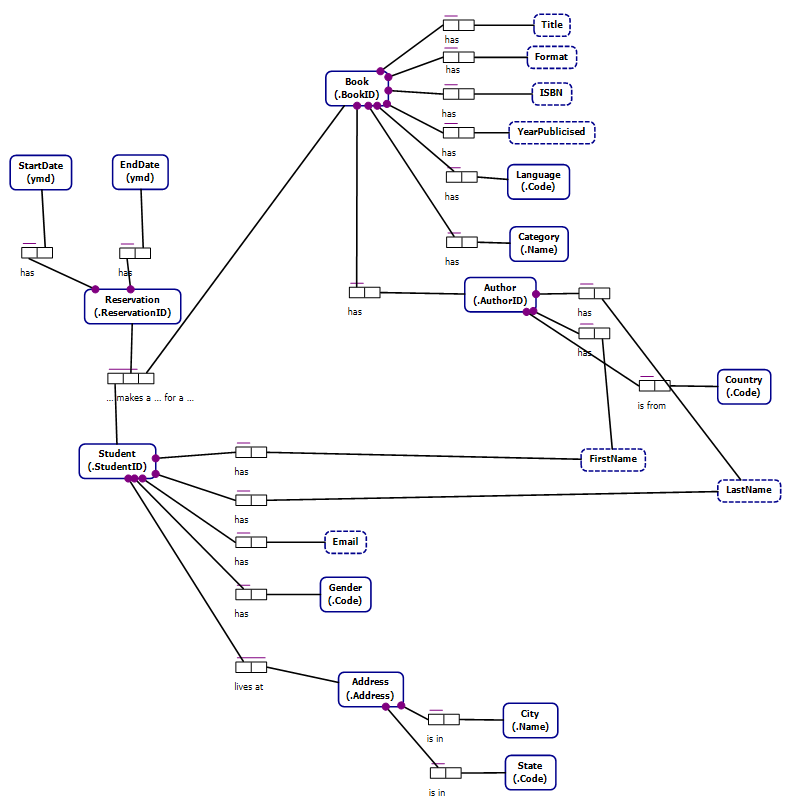
**Introduction**

A Library Management System (LMS) is a software designed to manage a library's primary housekeeping functions. Libraries use library management systems to keep in sight of their asset collections and relationships with their patrons. Our topic is on a library database management system because we want to keep track of the inventory of the books that are available, been checked out, and limit the amount of books students or faculty can check out. Our library database will include each book’s title, ID, author, year, category, description, quantity, location, book reservation, and book reservation archive, also we will have a unique ID for each student/library member along with their name, address, email which will help us to track the person who holds the book currently. We will be focusing on the requirements of the system as follows:

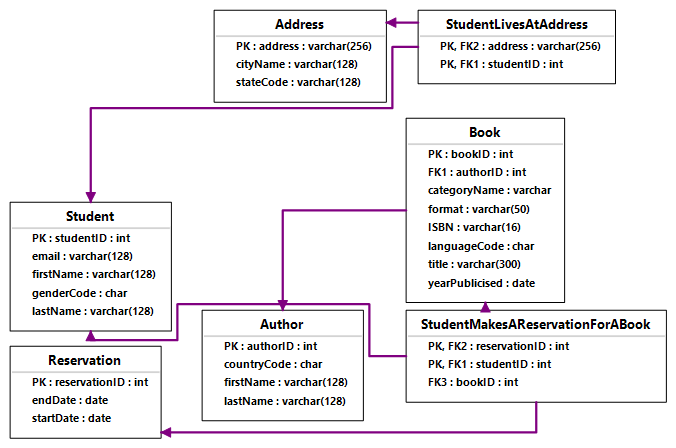
* Any library member can search a book with its Title, Book ID, Author.
* Each book will have a unique ID and a Category ID which will help to locate the book.
* There will be a maximum limit of 5 books a person can check out at a time.
* There will be a maximum limit of 15 days a person can keep the book.
* People will be able to reserve the book beforehand or when it's not available.
* The system would be able to retrieve information like who took a particular book or what are the books checked-out by a specific person.

This system will serve as an asset management system, which will also make us able to retrieve certain statistical inferences.

**ORM**

****

**Relational Schema**

****

**Code**

CREATE SCHEMA ORMModel1

GO

GO

CREATE TABLE ORMModel1.Book

(

bookID int NOT NULL,

authorID int NOT NULL,

categoryName nvarchar(max) NOT NULL,

format nvarchar(50) NOT NULL,

ISBN nvarchar(16) NOT NULL,

languageCode nchar(4000) NOT NULL,

title nvarchar(300) NOT NULL,

yearPublicised date NOT NULL,

CONSTRAINT Book\_PK PRIMARY KEY(bookID)

)

GO

CREATE TABLE ORMModel1.Author

(

authorID int NOT NULL,

countryCode nchar(4000) NOT NULL,

firstName nvarchar(128) NOT NULL,

lastName nvarchar(128) NOT NULL,

CONSTRAINT Author\_PK PRIMARY KEY(authorID)

)

GO

CREATE TABLE ORMModel1.Student

(

studentID int NOT NULL,

email nvarchar(128) NOT NULL,

firstName nvarchar(128) NOT NULL,

genderCode nchar(4000) NOT NULL,

lastName nvarchar(128) NOT NULL,

CONSTRAINT Student\_PK PRIMARY KEY(studentID)

)

GO

CREATE TABLE ORMModel1.Address

(

address nvarchar(256) NOT NULL,

cityName nvarchar(128) NOT NULL,

stateCode nvarchar(128) NOT NULL,

CONSTRAINT Address\_PK PRIMARY KEY(address)

)

GO

CREATE TABLE ORMModel1.Reservation

(

reservationID int NOT NULL,

endDate date NOT NULL,

startDate date NOT NULL,

CONSTRAINT Reservation\_PK PRIMARY KEY(reservationID)

)

GO

CREATE TABLE ORMModel1.StudentLivesAtAddress

(

address nvarchar(256) NOT NULL,

studentID int NOT NULL,

CONSTRAINT StudentLivesAtAddress\_PK PRIMARY KEY(address, studentID)

)

GO

CREATE TABLE ORMModel1.StudentMakesAReservationForABook

(

reservationID int NOT NULL,

studentID int NOT NULL,

bookID int NOT NULL,

CONSTRAINT StudentMakesAReservationForABook\_PK PRIMARY KEY(reservationID, studentID)

)

GO

ALTER TABLE ORMModel1.Book ADD CONSTRAINT Book\_FK FOREIGN KEY (authorID) REFERENCES ORMModel1.Author (authorID) ON DELETE NO ACTION ON UPDATE NO ACTION

GO

ALTER TABLE ORMModel1.StudentLivesAtAddress ADD CONSTRAINT StudentLivesAtAddress\_FK1 FOREIGN KEY (studentID) REFERENCES ORMModel1.Student (studentID) ON DELETE NO ACTION ON UPDATE NO ACTION

GO

ALTER TABLE ORMModel1.StudentLivesAtAddress ADD CONSTRAINT StudentLivesAtAddress\_FK2 FOREIGN KEY (address) REFERENCES ORMModel1.Address (address) ON DELETE NO ACTION ON UPDATE NO ACTION

GO

ALTER TABLE ORMModel1.StudentMakesAReservationForABook ADD CONSTRAINT StudentMakesAReservationForABook\_FK1 FOREIGN KEY (studentID) REFERENCES ORMModel1.Student (studentID) ON DELETE NO ACTION ON UPDATE NO ACTION

GO

ALTER TABLE ORMModel1.StudentMakesAReservationForABook ADD CONSTRAINT StudentMakesAReservationForABook\_FK2 FOREIGN KEY (reservationID) REFERENCES ORMModel1.Reservation (reservationID) ON DELETE NO ACTION ON UPDATE NO ACTION

GO

ALTER TABLE ORMModel1.StudentMakesAReservationForABook ADD CONSTRAINT StudentMakesAReservationForABook\_FK3 FOREIGN KEY (bookID) REFERENCES ORMModel1.Book (bookID) ON DELETE NO ACTION ON UPDATE NO ACTION

GO

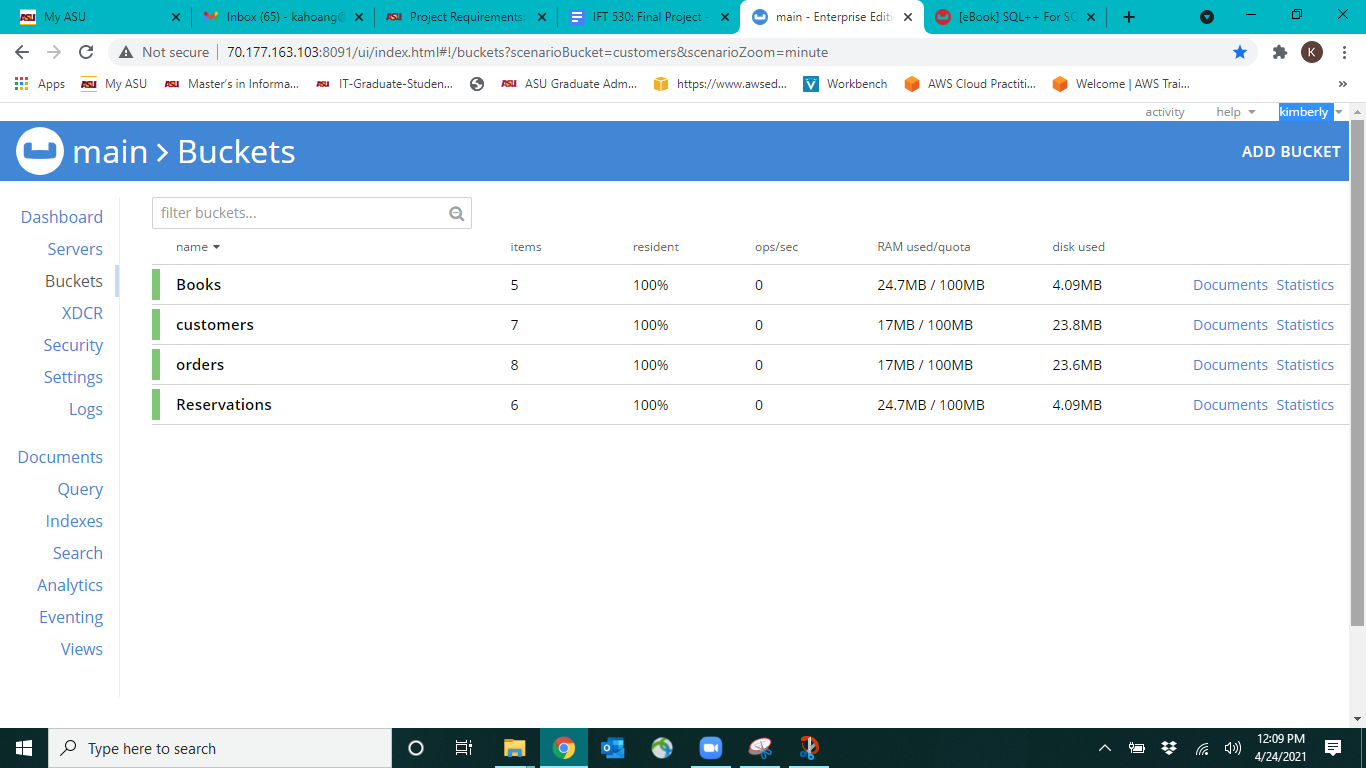
GO

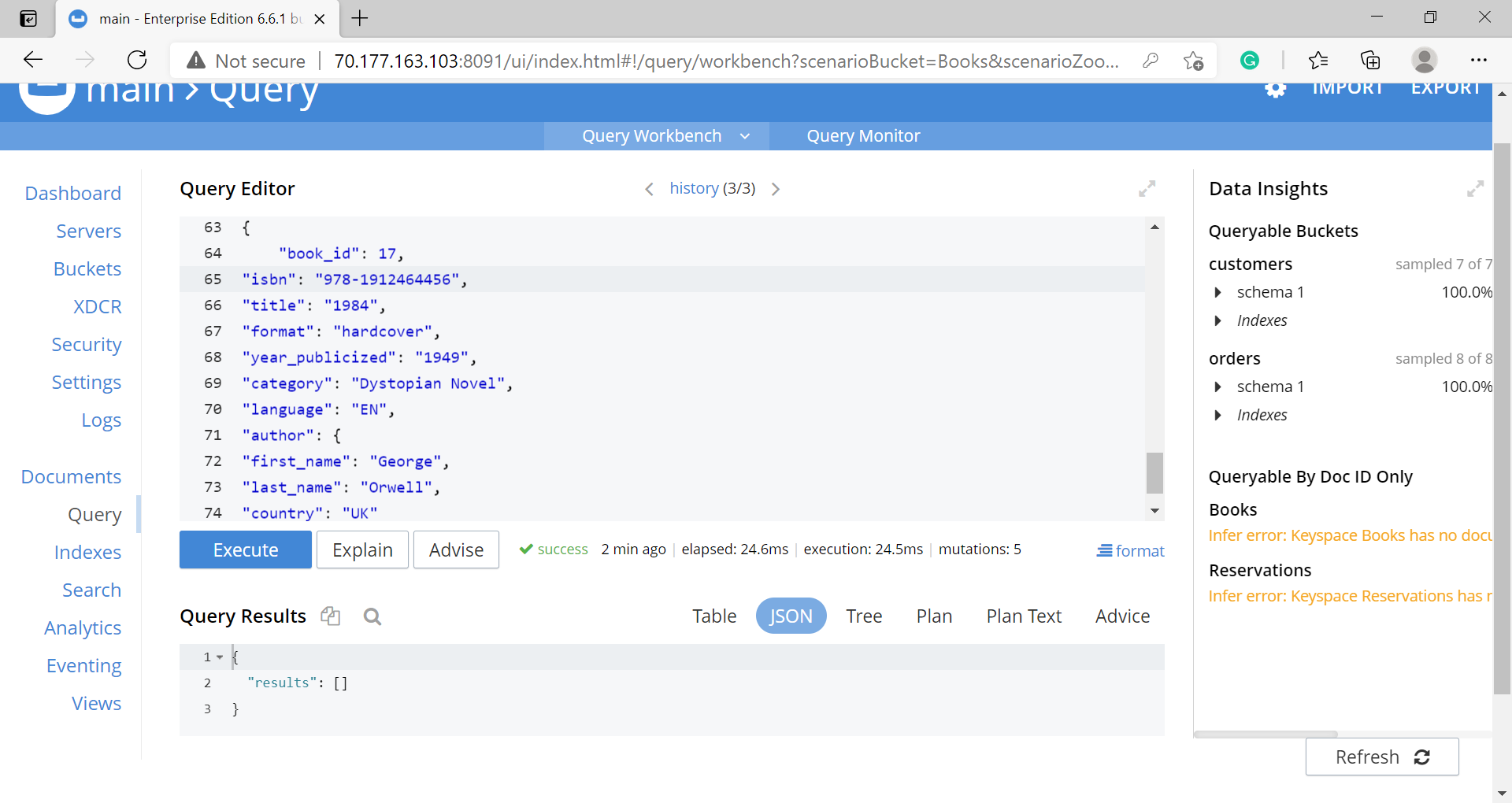
**Questions**

1. When did the student or faculty check out / return the book?
2. How many books do we have in stock?
3. How many books have been checked out?
4. How many times has a specific book been checked out?
5. Whether the X book is available or not?
6. How many copies of a particular book are available?
7. Where can I find X book from X category?
8. Which book is the most popular? (Most checked out)

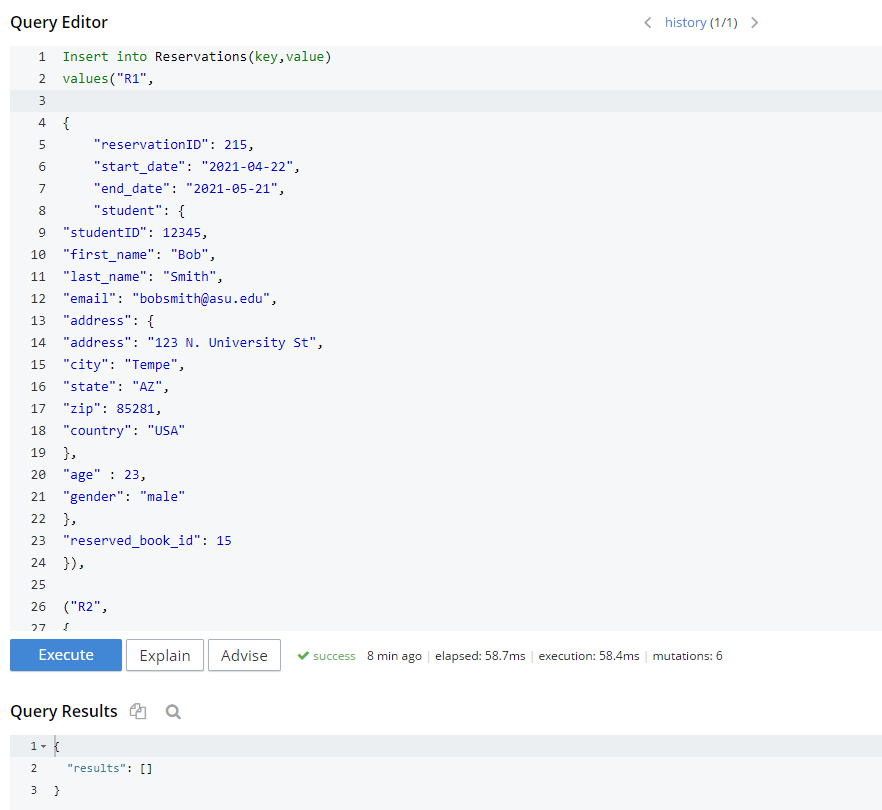
**Results in Couchbase**

**Created two buckets: Books & Reservations**

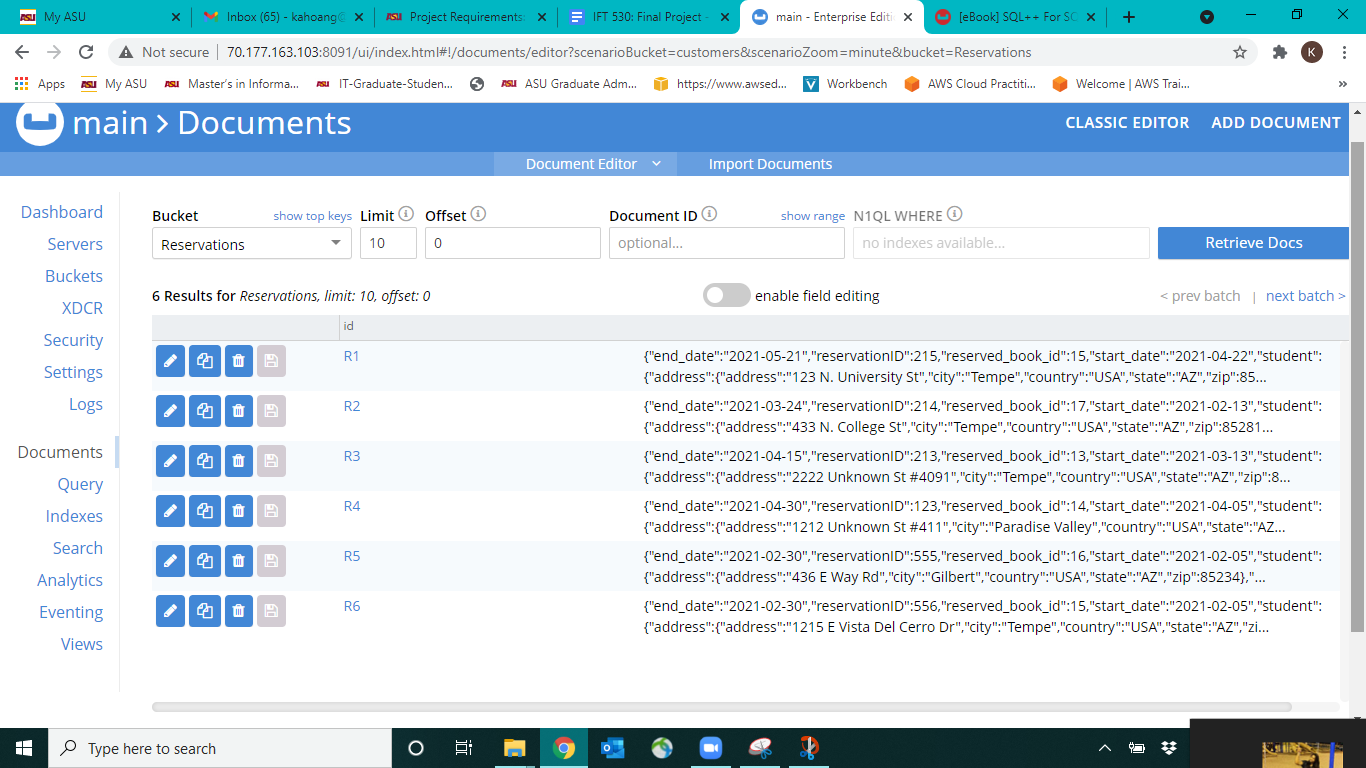
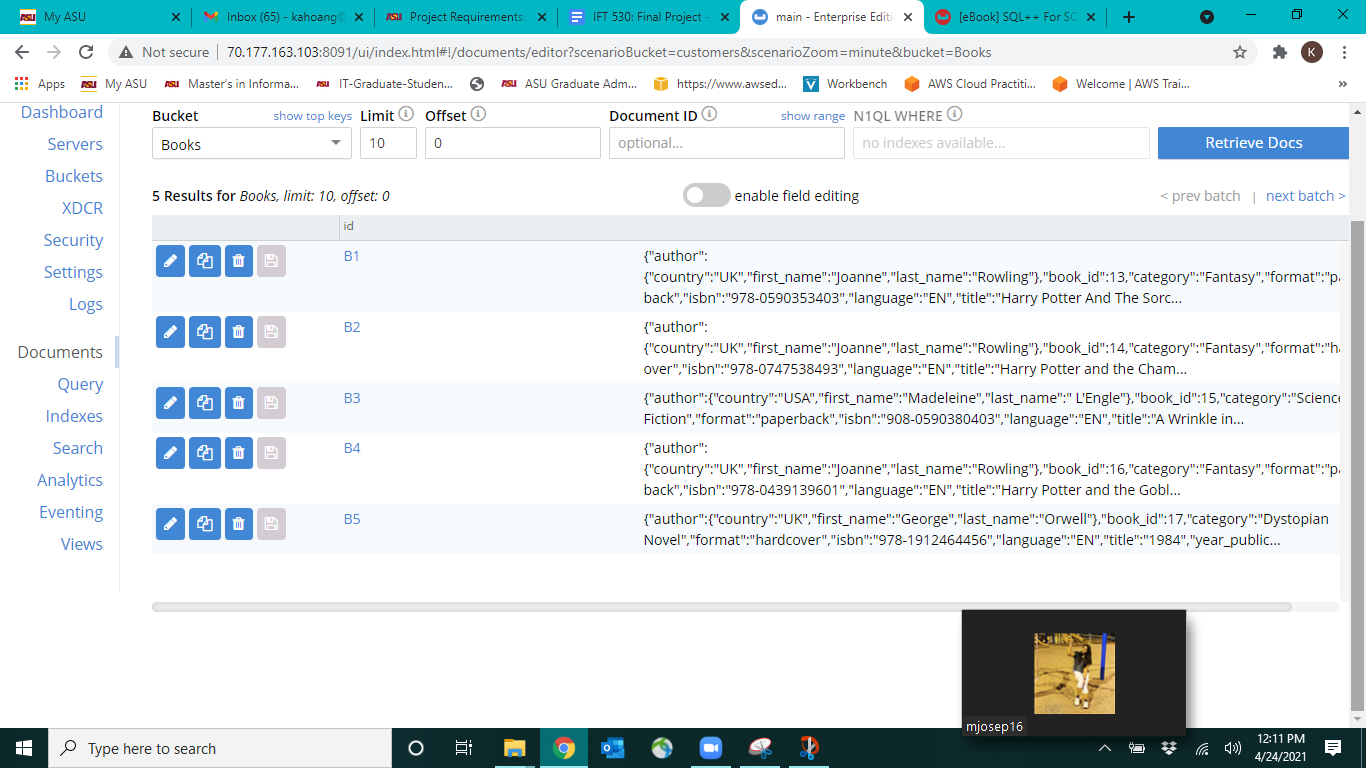
****

**Data Insertion in Book Bucket  
  
**

**Data Insertion in Reservations Bucket**

****

**Documents**

****

**Insertion Query for Books Bucket**

Insert into Books (key, value)

values

(“B1”,

{

“book\_id”: 13,

“isbn”: “978-0590353403”,

“title”: “Harry Potter And The Sorcerer's Stone”,

“format”: “paperback”,

“year\_publicized”: 1997,

“category”: “Fantasy”,

“language”: “EN”,

“author”: {

“first\_name”: “Joanne”,

“last\_name”: “Rowling”,

“country”: “UK”

}

}),

(“B2”,

{

“book\_id”: 14,

“isbn”: “978-0747538493”,

“title”: “Harry Potter and the Chamber of Secrets”,

“format”: “hardcover”,

“year\_publicized”: 1998,

“category”: “Fantasy”,

“language”: “EN”,

“author”: {

“first\_name”: “Joanne”,

“last\_name”: “Rowling”,

“country”: “UK”

}

}),

**(**“B3”,

{

“book\_id”: 15,

“isbn”: “908-0590380403”,

“title”: “A Wrinkle in Time”, “format”: “paperback”,

“year\_publicized”: 1962,

“category”: “Science Fiction”,

“language”: “EN”,

“author”: {

“first\_name”: “Madeleine”,

“last\_name”: “L'Engle”,

“country”: “USA”

}

}),

(“B4”,

{

“book\_id”: 16,

“isbn”: “978-0439139601”,

“title”: “Harry Potter and the Goblet of Fire”,

“format”: “paperback”,

“year\_publicized”: 2000,

“category”: “Fantasy”,

“language”: “EN”,

“author”: {

“first\_name”: “Joanne”,

“last\_name”: “Rowling”,

“country”: “UK”

}

}),

(“B5”,

{

“book\_id”: 17,

“isbn”: “978-1912464456”,

“title”: “1984”,

“format”: “hardcover”,

“year\_publicized”: “1949”,

“category”: “Dystopian Novel”,

“language”: “EN”,

“author”: {

“first\_name”: “George”,

“last\_name”: “Orwell”,

“country”: “UK”

}

});

**Insertion Query for Reservations Bucket**

Insert into Reservations(key,value)  
values(“R1”,

{

“reservationID”: 215,

“start\_date”: “2021-04-22”,

“end\_date”: “2021-05-21”,

“student”: {

"studentID": 12345,

"first\_name": "Bob",

“last\_name”: “Smith”,

“email”: ”[bobsmith@asu.edu](mailto:bob@asu.edu)”,

"address": {

“address”: "123 N. University St",

“city”: "Tempe",

“state”: "AZ",

“zip”: 85281,

“country”: "USA"

},

"age" : 23,

“gender”: “male”

},

“reserved\_book\_id”: 15

}),

(“R2”,

{

“reservationID”: 214,

“start\_date”: “2021-02-13”,

“end\_date”: “2021-03-24”,

“student”: {

"studentID": 67891,

"first\_name": "Joe",

“last\_name”: “Lee”,

“email”: ”joelee@asu.edu”,

"address": {

“address”: "433 N. College St",

“city”: "Tempe",

“state”: "AZ",

“zip”: 85281,

“country”: "USA"

},

"age" : 20,

“gender”: “male”

},

“reserved\_book\_id”: 17

}),

(“R3”,

{

“reservationID”: 213,

“start\_date”: “2021-03-13”,

“end\_date”: “2021-04-15”,

“student”: {

"studentID": 55555,

"first\_name": "Andrei",

“last\_name”: “Fedotov”,

“email”: ”afedotov@asu.edu”,

"address": {

“address”: "2222 Unknown St #4091",

“city”: "Tempe",

“state”: "AZ",

“zip”: 85281,

“country”: "USA"

},

"age" : 23,

“gender”: “male”

},

“reserved\_book\_id”: 13

}),

(“R4”,

{

“reservationID”: 123,

“start\_date”: “2021-04-05”,

“end\_date”: “2021-04-30”,

“student”: {

"studentID": 66666,

"first\_name": "Ali",

“last\_name”: “Oliphant”,

“email”: ”aoliph@asu.edu”,

"address": {

“address”: "1212 Unknown St #411",

“city”: "Paradise Valley",

“state”: "AZ",

“zip”: 85253,

“country”: "USA"

},

"age" : 21,

“gender”: “female”

},

“reserved\_book\_id”: 14

}),

(“R5”,

{

“reservationID”: 555,

“start\_date”: “2021-02-05”,

“end\_date”: “2021-02-30”,

“student”: {

"studentID": 12857,

"first\_name": "Kimberly",

“last\_name”: “Hoang”,

“email”: ”kahoang@asu.edu”,

"address": {

“address”: "436 E Way Rd",

“city”: "Gilbert",

“state”: "AZ",

“zip”: 85234,

“country”: "USA"

},

"age" : 22,

“gender”: “female”

},

“reserved\_book\_id”: 16

}),

(“R6”,

{

“reservationID”: 556,

“start\_date”: “2021-02-05”,

“end\_date”: “2021-02-30”,

“student”: {

"studentID": 12136,

"first\_name": "Tess",

“last\_name”: “Joseph”,

“email”: ”tjosepr@asu.edu”,

"address": {

“address”: "1215 E Vista Del Cerro Dr",

“city”: "Tempe",

“state”: "AZ",

“zip”: 85281,

“country”: "USA"

} ,

"age" : 23,

“gender”: “female”

},

“reserved\_book\_id”: 15

});

**Query #1**

Display

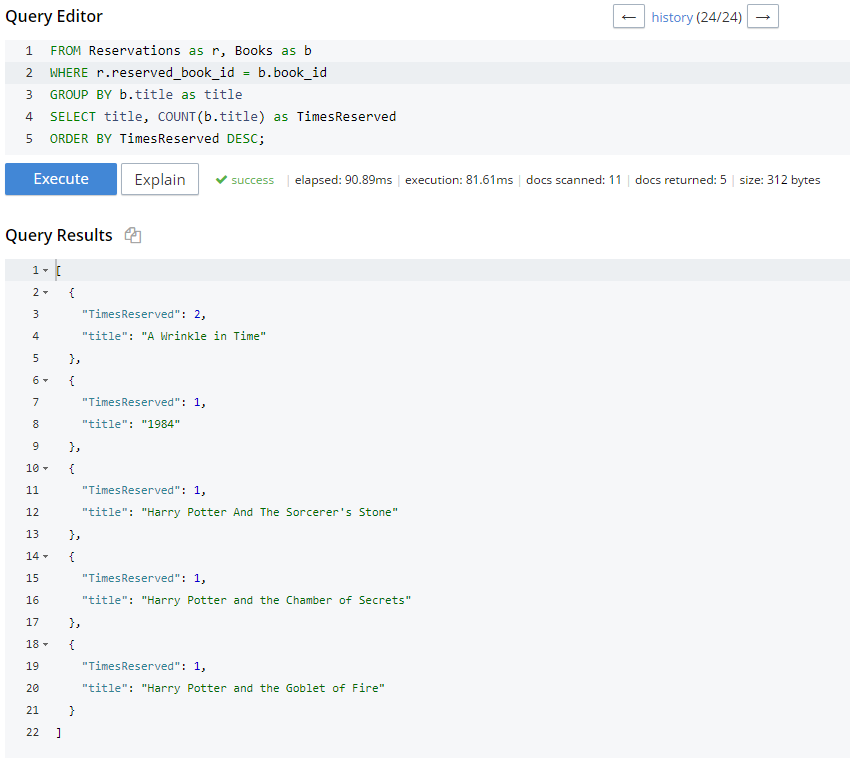
FROM Reservations as r, Books as b

WHERE r.reserved\_book\_id = b.book\_id

GROUP BY b.title as title

SELECT title, COUNT(b.title) as TimesReserved

ORDER BY TimesReserved DESC;

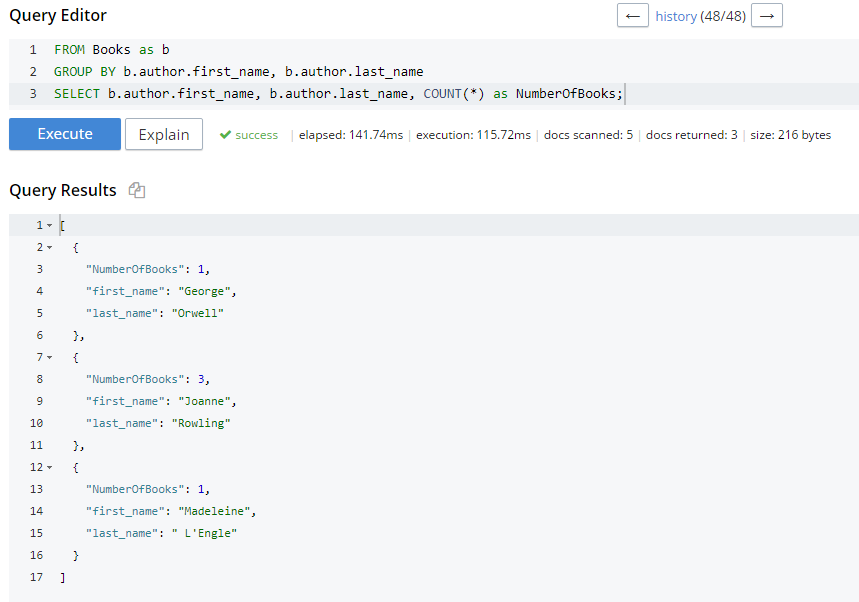


**Query #2**

FROM Books as b

GROUP BY b.author.first\_name, b.author.last\_name

SELECT b.author.first\_name, b.author.last\_name, COUNT(\*) as NumberOfBooks;

****

**Stored Procedure**

CREATE PROCEDURE findBookTitleByISBN

(

@isbn Varchar(16)= NULL

)

AS

IF @ISBN IS NULL

PRINT 'ISBN is NULL'

IF NOT EXISTS (SELECT ISBN FROM Book WHERE ISBN = @isbn)

PRINT 'Book does not exist in the database'

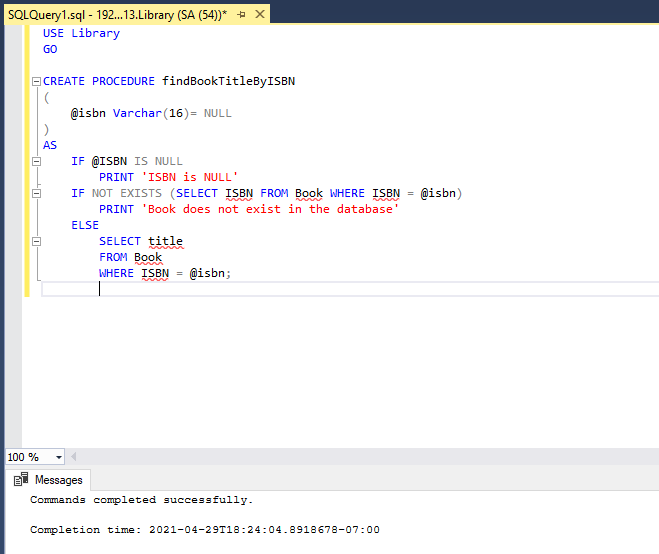
ELSE

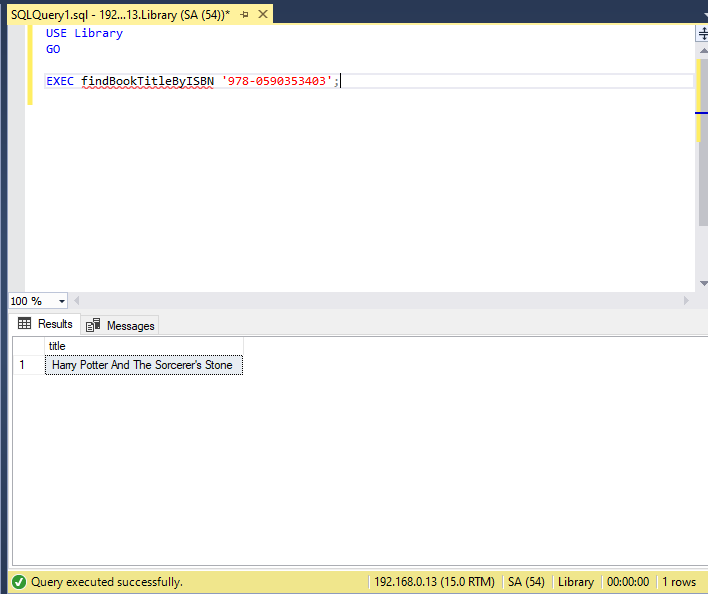
SELECT title

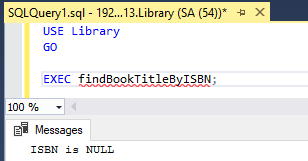
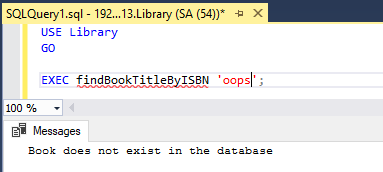
FROM Book

WHERE ISBN = @isbn;

EXEC findBookTitleByISBN ‘978-0590353403’;







**User-Defined Function (UDF): Scalar-Valued Function**

CREATE FUNCTION fnBookID

(@BookTitle varchar(50))

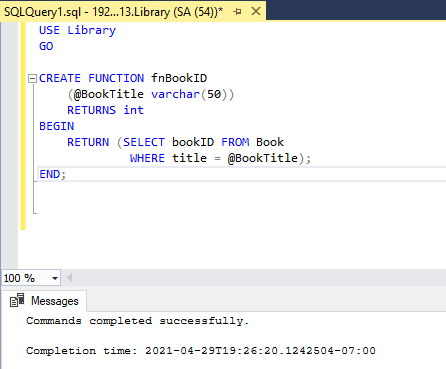
RETURNS int

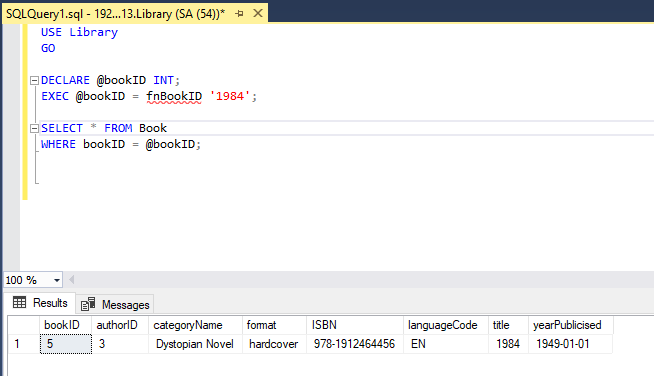
BEGIN

RETURN (SELECT bookID FROM Book

WHERE title = @BookTitle);

END;





**Trigger**CREATE TRIGGER Books\_INSERT\_UPDATE

ON Books

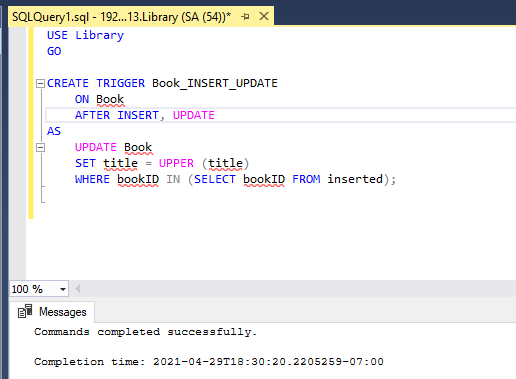
AFTER INSERT, UPDATE

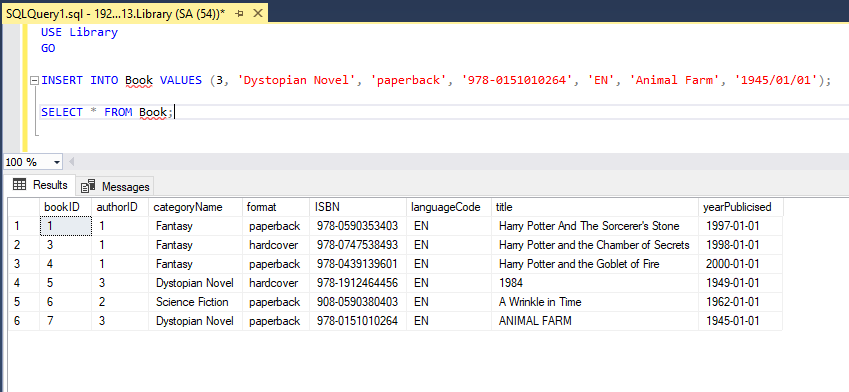
AS

UPDATE Books

SET BookTitle = UPPER (BookTitle)

WHERE BookID IN (SELECT BookID FROM Inserted);

****

****

**Conclusion**

In conclusion, our library management system will allow librarians to keep track of the book inventory, reservations, and the students who reserved the books. The online cloud-based library management system provides easy ways for librarians to create and store their library collections. It enables students and staff to communicate and collaborate through multiple channels. The library management system allows the user to keep track of both the book and the individual. This software allows you to store all of the information about your library. The system's implementation would speed up data entry and provide easily measured reports. The books and students are given a particular unique id number. So that they can be accessed correctly and without errors. Our main aim of the project is to get the correct information about a particular student and books available in the library. It is expected that this project will go a long way in satisfying users requirements. The computerization of the library management will not only improve efficiency but will also reduce human stress thereby indirectly improving human resources.

**References**

<https://www.educative.io/courses/grokking-the-object-oriented-design-interview/RMlM3NgjAyR>

<https://www.slideshare.net/fiu025/library-management-system-in-sql>

<https://www.c-sharpcorner.com/UploadFile/ea3ed6/database-design-for-library-management-system/>

<https://www.sciencedirect.com/topics/computer-science/library-management-system>

<https://www.upgrad.com/blog/dbms-project-ideas-for-beginners/>

Haplin, T (2015) Object-Role Modeling Fundamentals, Technics Publishing

Murach, Joel (2016) SQL Server 2016 for Developers, Murach Publishing