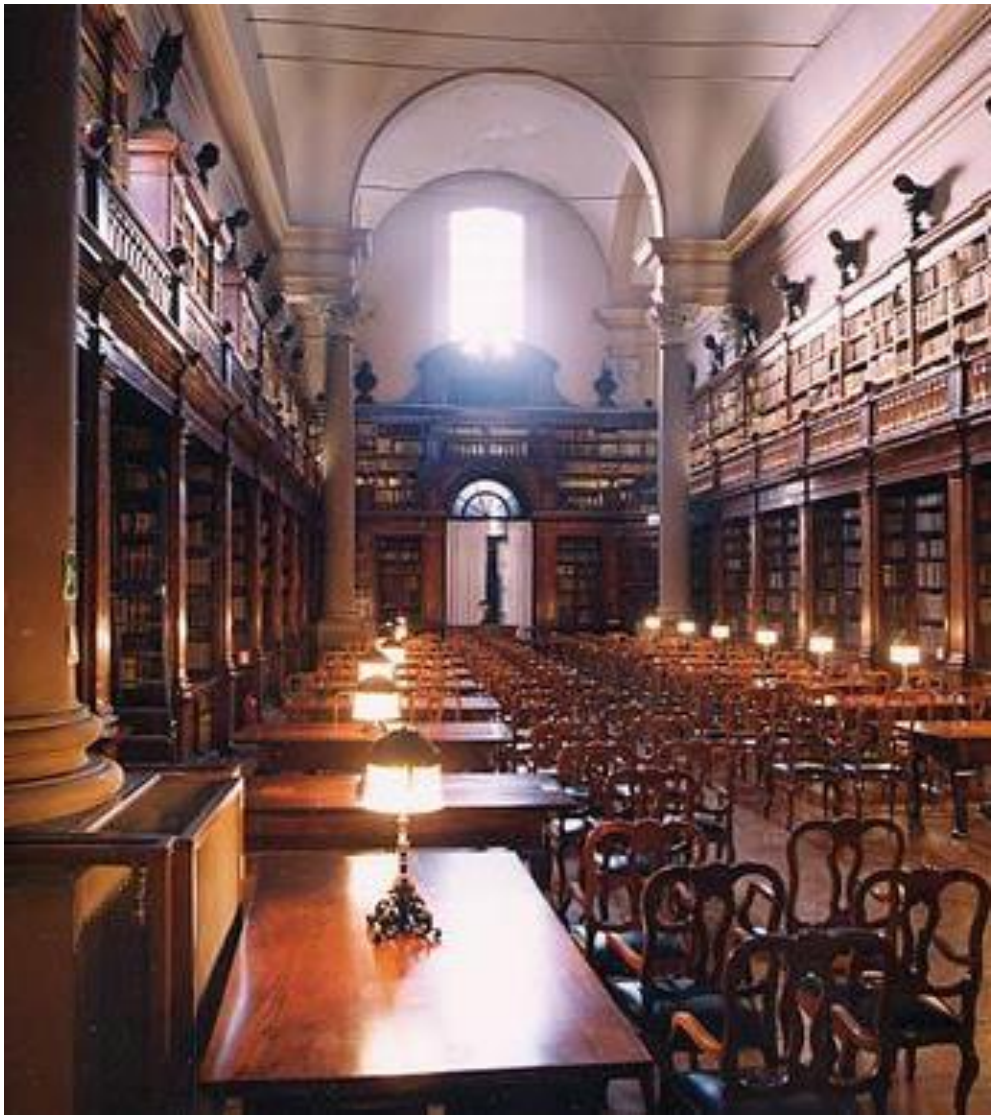


EBIBLIO

Progetto Basi di dati

Fantini Omar | 0000840120
Stefani Flavio | 0000883331
Tabanelli Simone | 0000890230

1. Raccolta / Analisi dei requisiti

La piattaforma EBIBLIO deve gestire i dati delle biblioteche UNIBO.
Ogni biblioteca dispone di un nome (univoco), un indirizzo, un'email, un sito web, delle coordinate (latitudine/longitudine), uno o più recapiti telefonici, un campo di testo relativo alle note storiche.
Ogni biblioteca può disporre di una galleria di immagini (una o più).
Inoltre, ogni biblioteca dispone di un numero limitato di posti lettura: ogni posto lettura dispone di un numero progressivo (univoco, ma solo all'interno di una biblioteca), dell'indicazione se dotato di presa di corrente o meno (campo booleano) e dell'indicazione se dotato di presa di rete Ethernet o meno (campo booleano).
La biblioteca mette a disposizione del pubblico l'accesso ad i propri libri: ogni libro dispone di un codice (univoco a livello UNIBO), un titolo, una lista degli autori, un anno di pubblicazione, un nome dell'edizione, un genere. I libri possono appartenere a due categorie (e solo a quelli): libri cartacei o ebook.
Nel primo caso, si vogliono memorizzare anche stato di conservazione, numero di pagine, numero scaffale, e stato del prestito. Lo stato di conservazione può assumere solo quattro valori: Ottimo, Buono, Non Buono, Scadente.
Lo stato del prestito può essere: Disponibile, Prenotato, Consegnato.
Nel caso degli e-book, si vogliono memorizzare anche la dimensione, il numero di accessi alla scheda e il PDF del documento.
Alla piattaforma EBIBLIO possono accedere tre categorie di utenti: Amministratori, Volontari e Utilizzatori.
Ogni utente dispone di email, password, nome, cognome, data di nascita, luogo di nascita, recapito telefonico. Gli utenti possono appartenere a tre categorie: amministratori di biblioteca, volontari e utilizzatori.
Gli amministratori (dipendenti UNIBO) dispongono anche del campo qualifica (testo, massimo 10 caratteri); ogni amministratore è responsabile di una sola biblioteca UNIBO. Una biblioteca UNIBO può avere più amministratori.
Gli utenti volontari hanno un campo mezzo di trasporto (piedi, bici, auto).
Gli utenti utilizzatori dispongono di un campo aggiuntivo relativo alla data di creazione dell'account, un campo professione, ed un campo relativo allo stato dell'account (quest'ultimo può assumere solo due valori: Attivo o Sospeso).
Gli utilizzatori possono prenotare un posto lettura presso una biblioteca UNIBO; ogni prenotazione dispone di un campo data, ora inizio ed ora fine.
La prenotazione di un posto lettura è possibile solo a condizione che la biblioteca abbia effettivamente posti lettura disponibili per la data/orario richiesto.
Gli utilizzatori possono accedere liberamente agli e-book disponibili: tuttavia, il sistema tiene traccia dello storico degli accessi agli e-book (o meglio alle loro schede) effettuati da ciascun utente. In maniera simile, gli utilizzatori possono prenotare un libro cartaceo, a patto che il testo sia nello stato Disponibile, e che lo stato di conversazione non sia pari a Scadente.
La prenotazione dispone di un codice (univoco), una data di avvio e una data di fine

(automaticamente settata a +15gg a partire dalla data di consegna, vedi sotto).
Gli utenti volontari si fanno carico di consegnare i libri prenotati agli utilizzatori: a tal proposito si vogliono gestire gli eventi di consegna: ogni evento di consegna è inserito da un utente volontario, fa riferimento ad una prenotazione di testo cartaceo, può essere di tipo "Restituzione" o "Affidamento" e dispone di campo data e note (massimo 200 caratteri).
Infine, è prevista la possibilità di gestire messaggi nella piattaforma.
Ogni messaggio è inserito da un amministratore ed è destinato ad un utente utilizzatore, e dispone di un titolo (es."Libro non disponibile"), un campo testo ed una data. Infine, gli amministratori possono inviare segnalazioni per comportamenti non corretti da parte di utilizzatori; ogni segnalazione dispone di data ed eventuale nota di testo, è inserita da un amministratore e diretta verso un utilizzatore.
Nel caso in cui un utilizzatore riceva cumulativamente più di tre segnalazioni (anche da amministratori di biblioteche diverse), lo stato dell'account viene settato a "Sospeso" (impedendo qualsiasi accesso alla piattaforma da parte dell'utente sanzionato).

Operazioni sui dati

Operazioni che riguardano tutti gli utenti:

- Autenticazione alla piattaforma
- Visualizzazione delle biblioteche presenti
- Visualizzazione dei posti lettura presenti in ogni biblioteca
- Visualizzazione dei libri disponibili in ogni biblioteca
- Visualizzazione delle statistiche

Operazioni che riguardano SOLO gli utenti UTILIZZATORI:

- Registrazione alla piattaforma
- Prenotazione di un posto lettura
- Prenotazione di un libro cartaceo
- Visualizzazione delle proprie prenotazioni
- Visualizzazione di un E-BOOK
- Visualizzazione propri eventi di consegna

Operazioni che riguardano SOLO gli utenti VOLONTARI:

- Visualizzazione di tutte le prenotazioni inserite sulla piattaforma
- Inserimento di un nuovo evento di consegna
- Aggiornamento di un evento di consegna

Operazioni che riguardano SOLO gli utenti AMMINISTRATORI:

- Inserimento/Cancellazione/Aggiornamento di un libro presso la biblioteca gestita
- Visualizzazione di tutte le prenotazioni presso la biblioteca gestita
- Inserimento di un messaggio rivolto ad un utente utilizzatore
- Inserimento di una segnalazione di comportamento non corretto

- Rimuovere tutte le segnalazioni di un utente, riportandone lo stato ad Attivo

Statistiche:

- Visualizzare la classifica delle biblioteche con postazioni letture meno utilizzate (in percentuale rispetto al numero di posti letture disponibili)
- Visualizzare la classifica dei volontari che hanno effettuato più consegne
- Visualizzare la classifica dei libri cartacei più prenotati
- Visualizzare la classifica degli e-book più acceduti

Vincoli sull'implementazione:

- Implementare tutte le operazioni sui dati (ove possibile) attraverso stored procedure.
- Utilizzare un trigger per implementare l'operazione cambio di stato di libro cartaceo, da Disponibile a Prenotato nel momento in cui un utente utilizzatore ha inserito una prenotazione per quel libro.
- Utilizzare un trigger per implementare l'operazione cambio di stato di libro cartaceo, da Prenotato a Consegnato nel momento in cui un utente volontario inserisce un evento di consegna di quel libro, di tipo ""Affidamento".
- Utilizzare un trigger per implementare l'operazione cambio di stato di libro cartaceo, da Consegnato a Disponibile nel momento in cui un utente volontario inserisce un evento di consegna di quel libro, di tipo ""Restituzione".
- Utilizzare un trigger per implementare l'operazione di passaggio di stato (da Attivo a Sospeso) di un account di un utente utilizzatore, ogni qual volta viene inserita la terza segnalazione da parte di un amministratore.

Tabella dei volumi

α (peso operazioni scrittura) = 2
wI (peso operazioni interattive)=1
wB (peso operazioni batch)=0.5

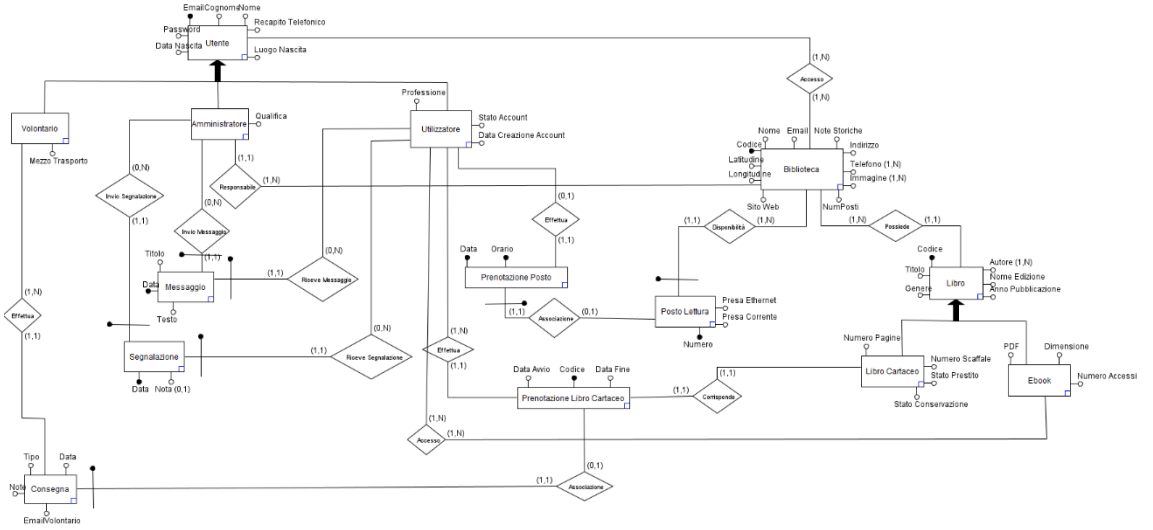
Totale ebook = 348
Accessi medio di ogni ebook = 55

Glossario dei dati

- Utente è una persona che è registrata alla piattaforma EBIBLIO
- Utente_Utilizzatore è una persona che si registra come ruolo di utilizzatore
- Utente_Amministratore è una persona che si registra come ruolo di amministratore
- Utente_Volontario è una persona che si registra come ruolo di volontario

2. Progettazione Concettuale

Diagramma E-R



Dizionario delle entità

Entità	Descrizione	Attributi	Identificatori
Biblioteca	Struttura di una biblioteca	Codice, Nome, Email, Note Storiche, Indirizzo, Latitudine, Longitudine, Sito Web,NumPosti	Codice
Immagine	Immagini della biblioteca	Nome, CodiceBiblioteca	Nome, CodiceBiblioteca
Telefono	Numeri di telefono della biblioteca	Numero, CodiceBiblioteca	Numero, CodiceBiblioteca
Posto Lettura	Posto Lettura	Numero, Presa Ethernet, PresaCorrente, CodiceBiblioteca	Numero, CodiceBiblioteca
Libro	Libro	Codice,Titolo, Genere, CodiceBiblioteca, AnnoPubblicazione , NomeEdizione	Codice
Autore	Autori di un libro	NumeroAutore, CodiceLibro	NumeroAutore

Libro_Libro_Cartaceo	Tipo di libro cartaceo	CodiceLibro, Numero Pagine, Numero Scaffale Stato Prestito, Stato Conservazione	CodiceLibro
Libro_Ebook	Tipo di libro ebook	CodiceLibro, PDF, Dimensione, Numero Accessi	CodiceLibro
Prenotazione Libro Cartaceo	Prenotazione di un libro cartaceo	Codice, Data Avvio, Data Fine, EmailUtilizzatore, CodiceLibro	Codice
Prenotazione Posto	Prenotazione posto in un posto lettura	Data, Orario, EmailUtilizzatore, NumeroPostoLettura, CodiceBiblioteca	Data, Orario, NumeroPostoLettura, CodiceBiblioteca
Utente	Utente	Email, Nome, Cognome, Password, Recapito Telefonico, Luogo Nascita, Data Nascita	Email
Utente_Volontario	Tipo di Utente	EmailUtente, Mezzo Trasporto	EmailUtente
Utente_Amministratore	Tipo di Utente	EmailUtente, Qualifica, CodiceBiblioteca	EmailUtente
Utente_Utilizzatore	Tipo di Utente	EmailUtente, Professione, Stato Account, DataCreazioneAccount	EmailUtente
Consegna	Consegna effettuata dal volontario	EmailUtente, Data, Note, Tipo, CodicePrenotazione, EmailVolontario	CodicePrenotazione
Messaggio	Messaggio inviato dall'amministratore o utilizzatore	Titolo, Data, Testo, EmailAmministratore, EmailUtilizzatore	EmailAmministratore, Data, EmailUtilizzatore
Segnalazione	Segnalazione inviata dall'amministratore all'utilizzatore	Data, Nota, EmailAmministratore, EmailUtilizzatore	Data, EmailAmministratore, EmailUtilizzatore

Dizionario delle relazioni

Relazioni	Descrizione	Componenti	Attributi
Accesso	Accesso utente alla biblioteca	Utente, Biblioteca	
Disponibilità	Verifica la disponibilità di un posto lettura	Biblioteca, Posto Lettura	
Possiede	Libro posseduti dalla biblioteca	Biblioteca, Libro	
Corrisponde	Corrispondenza tra libro cartaceo e la sua prenotazione	Libro Cartaceo, Prenotazione Libro Cartaceo	
Accesso	Accesso di un utente utilizzatore ad un ebook	Ebook, Utilizzatore	
Associazione	Associazione tra la prenotazione del libro cartaceo e la sua consegna	Prenotazione Libro Cartaceo, Consegna	
Effettua	L'utilizzatore può effettuare una prenotazione del posto lettura	Utilizzatore, Prenotazione Posto	
Associazione	Associazione tra la prenotazione del posto e la sua sala lettura	Posto lettura, Prenotazione Posto	
Effettua	L'utilizzatore può effettuare una prenotazione di un libro cartaceo	Prenotazione Libro Cartaceo, Utilizzatore	
Invio Segnalazione	Invio della segnalazione dall'amministratore	Amministratore, Segnalazione	
Riceve Segnalazione	Ricezione di una segnalazione dall'utilizzatore	Utilizzatore, Segnalazione	
Invio Messaggio	Invio di un messaggio dall'amministratore	Amministratore, Messaggio	

Riceve Messaggio	Ricezione di un messaggio dall'utilizzatore	Utilizzatore, Messaggio	
Responsabile	L'amministratore è responsabile di gestire biblioteche	Amministratore, Biblioteca	
Effettua	Effettuazione della consegna da parte del volontario	Volontario, Consegna	

Tabella delle business rules

Il numero dei posti lettura è un numero progressivo
Il campo presa corrente del posto lettura è un campo booleano
Il campo presa di rete ethernet del posto lettura è un campo booleano
Lo stato di conservazione può assumere solo quattro valori: Ottimo, Buono, Non Buono, Scadente
Lo stato del prestito può essere: Disponibile, Prenotato, Consegnato
Il campo qualifica di ogni amministratore è un testo di massimo 10 caratteri
Gli utenti volontari hanno un campo mezzo di trasporto che può essere soltanto piedi, bici, auto
Gli utenti utilizzatori hanno un campo stato dell'account che può assumere solo due valori: Attivo o Sospeso
La prenotazione di un posto lettura è possibile solo a condizione che la biblioteca abbia effettivamente posti lettura disponibili per la data/orario richiesto.
Il sistema tiene traccia dello storico degli accessi agli e-book effettuati da ciascun utente
Gli utenti utilizzatori possono prenotare un libro cartaceo, a patto che il testo sia nello stato Disponibile, e che lo stato di conversazione non sia pari a Scadente.
La data di fine prestito del libro è automaticamente settata a +15gg a partire dalla data di consegna
Il tipo della consegna di un libro può assumere solo i valori testuali di "Restituzione" o "Affidamento"
Il campo note della consegna può avere massimo 200 caratteri
Nel caso in cui un utilizzatore riceva cumulativamente più di tre segnalazioni (anche da amministratori di biblioteche diverse), lo stato dell'account viene settato a "Sospeso"

3. Progettazione Logica

Ristrutturazione dello schema concettuale

Dallo schema concettuale siamo partiti dalle generalizzazioni, che abbiamo scelto di sostituire con apposite entità e relazioni sia per i genitori sia per le figlie, definendo i loro vincoli di integrità.
Abbiamo poi eliminato gli attributi multi-valore in *Biblioteca* e *Autore* separandoli con opportune entità.
Le relazioni tra le varie entità le abbiamo gestite inglobandole nell'entità con partecipazione obbligatoria (dove possibile), mentre le due relazioni "*Accesso_Utente_Biblioteca*" e "Accesso_Utilizzatore_Ebook" le abbiamo trasformate in apposite tabelle.

Analisi delle ridondanze

Valutare se la seguente ridondanza: campo *NUMERO_ACCESSI* relativo ad ogni E-BOOK debba essere tenuta o eliminata, sulla base delle seguenti operazioni:

- Inserire un nuovo e-book (3 volte/mese, interattiva)
- Rimuovere un e-book (3 volte/mese, interattiva)
- Contare il numero di accessi di ogni e-book (2 volte/mese, interattiva)

Data la formula per calcolare il costo di un'operazione:
c(x) = frequenza operazione * peso tipo di operazione * (α * numero accessi in scrittura + numero accessi in lettura

Considerando la tabella dei volumi già definita:
α (peso operazioni scrittura) = 2, wI (peso operazioni interattive)=1, wB (peso operazioni batch)=0.5, Totale ebook = 348, Accessi medio di ogni ebook = 55

Calcolo prima il costo delle operazioni nel caso di mantenimento del campo ridondante.

1) Per calcolare il costo di questa operazione considero due accessi in scrittura in *Libri* e in *Ebook*.

   C(op1R) = 3 * 1 * (2*(1+1)+ 0) = 12

2) Per questa operazione considero un'operazione di scrittura ciascuna per *Ebook* e *Libro*.

   C(op2R) = 3 * 1 * (2*(1+1) + 0) = 12

3) In questa operazione considero 55 operazioni di lettura moltiplicate per il numero medio di accessi nel campo *NUMERO_ACCESSI* di *Ebook*.

   C(op3R) = 2 * 1 * (2*(0)+ 55*348) = 38.280

Nel caso in cui elimino il campo *NUMERO_ACCESSI* da *Ebook*, il costo della prima operazione risulterebbe uguale a quello precedente, ossia C(op1R), ma cambiano gli altri costi.

Nella seconda operazione considero tre operazioni di scrittura, una ciascuna per *Ebook* e *Libro* e l'eliminazione degli accessi del singolo ebook in Accesso. Abbiamo quindi :

C(op2) = 3 * 1 * (2*(1+1+55*1) + 0) = 342

Il costo della terza operazione aumenta rispetto al caso precedente, perché conto il numero di accessi di ogni e-book moltiplicato per il numero medio di accessi sia per *Ebook* sia per *Accessi*. Risulta quindi:

C(op3) = 2 * 1 * (2*(0)+ 55*348+55*348) = 76.560

Confrontando i due casi, le somme delle operazioni sono di 38.304 nel primo caso e di 76.914 nel secondo: in quest'ultimo il costo sarà dunque il doppio rispetto al primo. L'utilizzo della ridondanza semplifica il peso per il sistema di gestire le operazioni, quindi possiamo mantenerla nello schema.

Tabelle risultanti con vincoli di chiave

Utente (Email, Cognome, Nome, RecapitoTelefonico, Password, DataNascita, LuogoNascita)

Volontario (MezzoTrasporto, EmailUtente)

Amministratore (Qualifica, EmailUtente, CodiceBiblioteca)

Utilizzatore (Professione, StatoAccount, DataCreazioneAccount, EmailUtente)

Consegna (Data, Tipo, Note, EmailVolontario, CodicePrenotazioneLibro)

Segnalazione (Data, EmailAmministratore, EmailUtilizzatore, Nota)

Messaggio (Titolo, Testo, Data, EmailAmministratore, EmailUtilizzatore)

Biblioteca (Codice, Nome, Email, Note Storiche, Indirizzo, Longitudine, Latitudine, Sito Web)

Immagine (Nome, CodiceBiblioteca)

Telefono (Numero, CodiceBiblioteca)

Posto Lettura (Numero, PresaEthernet, PresaCorrente, CodiceBiblioteca)

Libro (Codice, Titolo, Genere, AnnoPubblicazione, NomeEdizione, CodiceBiblioteca)

Autore (NomeAutore, CodiceLibro)

Libro_Cartaceo (NumeroPagine, NumeroScaffale, StatoPrestito, StatoConservazione, CodiceLibro)

Ebook (PDF, Dimensione, NumeroAccessi, CodiceLibro)

Prenotazione_Libro_Cartaceo (DataAvvio, Codice, DataFine, EmailUtilizzatore, CodiceLibro)

Prenotazione_Posto (Data, Orario, NumeroPostoLettura, EmailUtilizzatore, CodiceBiblioteca)

Accesso_Utente_Biblioteca(EmailUtente, CodiceBiblioteca)

Accesso_Utilizzatore_Ebook (CodiceEbook, EmailUtilizzatore)

Vincoli inter-relazionali

Volontario. EmailUtente → Utente. Email
Amministratore. EmailUtente → Utente. Email
Amministratore. CodiceBiblioteca → Biblioteca. Codice
Utilizzatore. EmailUtente → Utente. Email
Consegna. EmailVolontario → Utente. Email
Consegna. CodicePrenotazioneLibro → Prenotazione_Libro_Cartaceo.Codice
Segnalazione. EmailAmministratore → Utente. Email
Segnalazione. EmailUtilizzatore → Utente. Email
Messaggio. EmailAmministratore → Utente. Email
Messaggio. EmailUtilizzatore → Utente. Email
Immagine. CodiceBiblioteca → Biblioteca. Codice
Telefono. CodiceBiblioteca → Biblioteca. Codice
PostoLettura. CodiceBiblioteca → Biblioteca. Codice
Libro. CodiceBiblioteca → Biblioteca. Codice
Autore. CodiceLibro → Libro. Codice
Libro_Cartaceo. CodiceLibro → Libro. Codice
Ebook. CodiceLibro → Libro. Codice
Prenotazione_Libro_Cartaceo. EmailUtilizzatore → Utente. Email
Prenotazione_Libro_Cartaceo. CodiceLibro → Libro. Codice
Prenotazione_Posto. NumeroPostoLettura → PostoLettura. Numero
Prenotazione_Posto. EmailUtilizzatore → Utente. Email
Prenotazione_Posto. CodiceLibro → Libro. Codice
Accesso_Utente_Biblioteca. EmailUtente → Utente. Email
Accesso_Utente_Biblioteca. CodiceBiblioteca → Biblioteca. Codice
Accesso_Utilizzatore_Ebook. CodiceEbook → Libro. Codice
Accesso_Utilizzatore_Ebook. EmailUtilizzatore → Utente. Email

4. Normalizzazione

Non è necessaria una normalizzazione delle tabelle ottenute, perché già ben strutturate e non vi sono ridondanze da eliminare.

5. Descrizione delle funzionalità dell'applicazione Web

Per riuscire a far eseguire correttamente il programma bisogna prima di tutto utilizzare *Xamp*, poi copiare la cartella "UploadFile" nella cartella di root.
Successivamente eseguire direttamente in SQL il file "Ebiblio.sql" e poi "ScriptUploadData.sql".

L'applicazione Web da noi sviluppata offre la possibilità, come da specifiche fornitoci, di poter accedere e registrarsi ad un sistema di biblioteche e poter effettuare diverse operazioni in base al tipo di utente registrato.

Nell'homepage dà la possibilità di: registrarsi o autenticarsi alla piattaforma; visualizzare tutte le biblioteche presenti, i loro libri, i loro posti lettura e poterle localizzare in una mappa.
Si ha anche la possibilità di vedere alcune statistiche delle biblioteche, una classifica delle biblioteche con posti di lettura meno utilizzati, i titoli dei libri ed ebook più richiesti e la classifica dei volontari con maggior numero di consegne effettuate.

Dopo aver effettuato l'accesso in base all'utente possiamo svolgere diverse azioni.
Esistono tre tipi diversi di utente: amministratore, utilizzatore e volontario.

L'amministratore è una persona che si occupa di gestire una biblioteca a lui associata e può gestire completamente la disponibilità dei libri e vedere tutte le prenotazioni esistenti al momento. Inoltre, può inviare e visualizzare messaggi e segnalazioni agli utenti utilizzatori per notificare degli eventi e/o comportamenti non corretti.

Un utilizzatore è un utente lettore che può prenotarsi un posto lettura definendo la data, l'orario e la biblioteca e vedere le sue prenotazioni. Può navigare tra i vari titoli editoriali e gli ebook presenti nelle biblioteche e prenotarli. In caso prenotasse un ebook, l'utente può leggerlo direttamente dal proprio dispositivo. Per il libro cartaceo si attiva l'evento di consegna da parte di utente volontario che effettuerà il recapito presso la sua abitazione, di cui si può visualizzare lo stato di consegna del suo libro.
Oltre a questo, l'utilizzatore può ricevere e visualizzare i messaggi da parte degli amministratori riguardanti eventi delle sue prenotazioni o segnalazioni di comportamenti scorretti.

Un utente di tipo volontario è una persona che effettua le consegne dei libri appena questi vengono prenotati dai lettori e li consegna presso i loro domicili. Esso si occupa inoltre del

ritiro dei libri alla scadenza della prenotazione.
Ha la possibilità tramite la sua area personale di vedere tutte le prenotazioni della piattaforma, inserire un nuovo evento di consegna e aggiornarne lo stato.

6. Codice SQL completo dello schema

```
DROP DATABASE IF EXISTS EBIBLIO;
CREATE DATABASE EBIBLIO;

USE EBIBLIO;

CREATE TABLE `Utente` (
        Email VARCHAR(50) PRIMARY KEY,
        Cognome VARCHAR(50),
   Nome VARCHAR(50),
   RecapitoTelefonico VARCHAR(50),
   Password VARCHAR(50),
   DataNascita DATE,
   LuogoNascita VARCHAR(50)
)
ENGINE=InnoDB
;

CREATE TABLE `Volontario` (
        EmailVolontario VARCHAR(50) PRIMARY KEY REFERENCES Utente(Email),
        MezzoTrasporto VARCHAR(50)
)
ENGINE=InnoDB
;

CREATE TABLE `Amministratore` (
        EmailAmministratore VARCHAR(50) PRIMARY KEY REFERENCES Utente(Email),
        CodiceBiblioteca VARCHAR(50) REFERENCES Biblioteca(Codice),
        Qualifica VARCHAR(50)
)
ENGINE=InnoDB
;

CREATE TABLE `Utilizzatore` (
        EmailUtilizzatore VARCHAR(100) PRIMARY KEY REFERENCES Utente(Email),
        Professione VARCHAR(50),
   StatoAccount VARCHAR(50),
   DataCreazioneAccount DATE
)
ENGINE=InnoDB
;
```

```
CREATE TABLE `Consegna` (
        Data DATE,
   Tipo VARCHAR(50),
   Note VARCHAR(50),
        EmailVolontario VARCHAR(50) REFERENCES Volontario(EmailVolontario),
   CodicePrenotazioneLibro INT REFERENCES prenotazione_libro_cartaceo(Codice),
   PRIMARY KEY (CodicePrenotazioneLibro)
)
ENGINE=InnoDB
;

CREATE TABLE `Segnalazione` (
        Data DATE,
   Nota VARCHAR(50),
   EmailAmministratore VARCHAR(50) REFERENCES
Amministratore(EmailAmministratore),
   EmailUtilizzatore VARCHAR(50) REFERENCES Utilizzatore(EmailUtilizzatore),
   PRIMARY KEY (Data , EmailAmministratore, EmailUtilizzatore)
)
ENGINE=InnoDB
;

CREATE TABLE `Messaggio` (
        Titolo VARCHAR(50),
   Testo VARCHAR(50),
   Data DATE,
   EmailAmministratore VARCHAR(50) REFERENCES
Amministratore(EmailAmministratore),
   EmailUtilizzatore VARCHAR(50) REFERENCES Utilizzatore(EmailUtilizzatore),
   PRIMARY KEY (Data , EmailAmministratore, EmailUtilizzatore)
)
ENGINE=InnoDB
;

CREATE TABLE `Biblioteca` (
        Codice VARCHAR(50) NOT NULL,
        Nome VARCHAR(200),
   Email VARCHAR(50),
   NoteStoriche VARCHAR(50),
   Indirizzo VARCHAR(100),
   Longitudine VARCHAR(50),
   Latitudine VARCHAR(50),
```

```
   SitoWeb VARCHAR(100),
   PRIMARY KEY (Codice),
   NumPosti Int
)
ENGINE=InnoDB
;

CREATE TABLE `Immagine` (
        Nome VARCHAR(50) REFERENCES Amministratore(EmailAmministratore),
   CodiceBiblioteca VARCHAR(50) REFERENCES Biblioteca(Codice),
   PRIMARY KEY (Nome, CodiceBiblioteca)
)
ENGINE=InnoDB
;

CREATE TABLE `Telefono` (
        Numero INT,
   CodiceBiblioteca VARCHAR(50) REFERENCES Biblioteca(Codice),
   PRIMARY KEY (Numero, CodiceBiblioteca)
)
ENGINE=InnoDB
;

CREATE TABLE `PostoLettura` (
        CodiceBiblioteca VARCHAR(50) REFERENCES Biblioteca(Codice),
        Numero INT,
   PresaEthernet BOOL,
   PresaCorrente BOOL,
   PRIMARY KEY (CodiceBiblioteca)
)
ENGINE=InnoDB;

CREATE TABLE `Libro` (
        Codice INT NOT NULL AUTO_INCREMENT,
   Titolo VARCHAR(200),
   Genere VARCHAR(50),
   AnnoPubblicazione INT,
   NomeEdizione VARCHAR(50),
   CodiceBiblioteca VARCHAR(50) REFERENCES Biblioteca(Codice),
   PRIMARY KEY (Codice)
)
ENGINE=InnoDB
```

```
;

CREATE TABLE `Autore` (
        CodiceLibro INT PRIMARY KEY REFERENCES Libro(Codice),
   NomeAutore VARCHAR(50)
)
ENGINE=InnoDB
;

CREATE TABLE `LibroCartaceo` (
        NumeroPagine INT,
   NumeroScaffale INT,
   StatoPrestito VARCHAR(50),
   StatoConservazione VARCHAR(50),
   CodiceLibro INT PRIMARY KEY REFERENCES Libro(Codice)
)
ENGINE=InnoDB
;

CREATE TABLE `Ebook` (
        PDF VARCHAR(50),
   Dimensione VARCHAR(50),
   NumeroAccessi INT,
   CodiceLibro INT PRIMARY KEY REFERENCES Libro(Codice)
)
ENGINE=InnoDB
;

CREATE TABLE `Prenotazione_libro_cartaceo` (
        DataAvvio DATE,
   Codice INT NOT NULL AUTO_INCREMENT,
   DataFine DATE,
   EmailUtilizzatore VARCHAR(50) REFERENCES Utilizzatore(EmailUtilizzatore),
   CodiceLibro INT REFERENCES Libro(Codice),
        PRIMARY KEY (Codice)
)
ENGINE=InnoDB
;

CREATE TABLE `Prenotazione_Posto` (
        Data DATE,
   Orario INT,
```

```
   NumeroPosti INT,
   EmailUtilizzatore VARCHAR(50) REFERENCES Utilizzatore(EmailUtilizzatore),
   CodiceBiblioteca VARCHAR(50) REFERENCES Biblioteca(Codice),
   PRIMARY KEY (Data, Orario, CodiceBiblioteca, NumeroPosti)
)
ENGINE=InnoDB
;

CREATE TABLE `Accesso_Utente_Biblioteca` (
        CodiceBiblioteca VARCHAR(50) REFERENCES Biblioteca(Codice),
   EmailUtente VARCHAR(50) REFERENCES Utilizzatore(EmailUtilizzatore),
   PRIMARY KEY (EmailUtente, CodiceBiblioteca)
)
ENGINE=InnoDB
;

CREATE TABLE `Accesso_Utilizzatore_Ebook` (
        CodiceEbook INT REFERENCES Ebook(CodiceLibro),
   EmailUtilizzatore VARCHAR(50) REFERENCES Utilizzatore(EmailUtilizzatore),
   Codice INT NOT NULL AUTO_INCREMENT,
   PRIMARY KEY (Codice)
)
ENGINE=InnoDB
;




##########################################################################
############################
#Stored Procedure
# REGISTRAZIONE UTILIZZATORE
DELIMITER $
CREATE PROCEDURE RegistrazioneUtilizzatore(IN Email VARCHAR(50),IN Cognome
VARCHAR(50),IN Nome VARCHAR(50), IN Telefono VARCHAR(14), IN Password
VARCHAR(50), IN DataNascita DATE,   IN LuogoNascita VARCHAR(50), IN Professione
VARCHAR(250), IN StatoAccount VARCHAR(30), IN DataCreazioneAccount DATE)
BEGIN
        START TRANSACTION;
                INSERT INTO utente (Email,
Cognome,Nome,RecapitoTelefonico,Password,DataNascita,LuogoNascita) VAlUES
(Email,Cognome, Nome,Telefono, Password,DataNascita,  LuogoNascita   );
```

```
      INSERT INTO utilizzatore (EmailUtilizzatore, Professione, StatoAccount,
DataCreazioneAccount) VAlUES (Email, Professione, StatoAccount,
DataCreazioneAccount);
  COMMIT;
END $
DELIMITER ;
#_____________________________________________________________

# REGISTRAZIONE VOLOTNARIO
DELIMITER $
CREATE PROCEDURE RegistrazioneVolontario(IN Email VARCHAR(50),IN Cognome
VARCHAR(50),IN Nome VARCHAR(50), IN Telefono VARCHAR(14), IN Password
VARCHAR(50), IN DataNascita DATE,   IN LuogoNascita VARCHAR(50), IN Mezzo
VARCHAR(250))
BEGIN
          START TRANSACTION;
                INSERT INTO utente (Email,
Cognome,Nome,RecapitoTelefonico,Password,DataNascita,LuogoNascita) VAlUES
(Email,Cognome, Nome,Telefono, Password,DataNascita,  LuogoNascita   );
      INSERT INTO volontario (EmailVolontario,MezzoTrasporto) VAlUES (Email,
Mezzo);
  COMMIT;
END $
DELIMITER ;
#_____________________________________________________________

# REGISTRAZIONE AMMINISTRATORE
DELIMITER $
CREATE PROCEDURE RegistrazioneAmministratore(IN Email VARCHAR(50),IN
Cognome VARCHAR(50),IN Nome VARCHAR(50), IN Telefono VARCHAR(14), IN
Password VARCHAR(50), IN DataNascita DATE,   IN LuogoNascita VARCHAR(50), IN
CodiceBiblioteca VARCHAR(50), IN Qualifica VARCHAR(50))
BEGIN
          START TRANSACTION;
                INSERT INTO utente VAlUES (Email,Cognome, Nome,Telefono,
Password,DataNascita,  LuogoNascita   );
      INSERT INTO amministratore  VAlUES (Email, CodiceBiblioteca, Qualifica);
  COMMIT;
END $
DELIMITER ;
#_____________________________________________________________
```

```
# Visualizzazione delle biblioteche presenti
DELIMITER $
CREATE PROCEDURE VisualizzaBibliotechePresenti()
BEGIN
        SELECT Nome,Codice
   FROM biblioteca;
END $
DELIMITER ;
#_____________________________________________________________

# Visualizzazione dei posti lettura presenti in ogni biblioteca
DELIMITER $
CREATE PROCEDURE VisualizzaPostiLettura()
BEGIN
        SELECT Nome as NomeBiblioteca, Codice, NumPosti
   FROM biblioteca INNER JOIN postolettura ON Codice=CodiceBiblioteca   ;
END $
DELIMITER ;
#_____________________________________________________________

# Visualizzazione dei libri disponibili
DELIMITER $
CREATE PROCEDURE VisualizzaLibriDisponibili(In codiceBiblioteca VARCHAR(50))
BEGIN
        SELECT Codice,Titolo,Genere,AnnoPubblicazione,NomeEdizione
   FROM libro INNER JOIN librocartaceo ON Codice=CodiceLibro
   WHERE StatoPrestito="Disponibile" and codiceBiblioteca=libro.CodiceBiblioteca;
END $
DELIMITER ;
#_____________________________________________________________


# Visualizzazione classifica delle biblioteche con postazioni letture meno utilizzate
DELIMITER $
CREATE PROCEDURE BiblioConPostLetturaMenoUtilizzate()
BEGIN
select CodiceBiblioteca, (ConteggioP/NumPosti) as Percentuale
        from biblioteca inner join (
   select CodiceBiblioteca, Count(*) as ConteggioP from prenotazione_posto
                group by CodiceBiblioteca
                order by ConteggioP desc) as t on Codice=CodiceBiblioteca
                order by Percentuale;
```

```
END $
DELIMITER ;
#_____________________________________________________________

# Visualizzazione dei volontari che che hanno effettuato più consegne
DELIMITER $
CREATE PROCEDURE VisualizzaVolontariConPiuConsegne()
BEGIN
	Select volontario.EmailVolontario, count(*) as NumConsegne
   FROM volontario inner join consegna on consegna.EmailVolontario =
volontario.EmailVolontario
   group by volontario.EmailVolontario
   order by NumConsegne DESC
   limit 5;
END $
DELIMITER ;
#_____________________________________________________________

# Visualizzazione della classifica dei libri cartacei più prenotati
DELIMITER $
CREATE PROCEDURE VisualizzaLibriCartPiuPrenotati()
BEGIN
	Select libro.Codice,Titolo, count(*) as NumPrenotazioni
   FROM (libro INNER JOIN librocartaceo ON libro.Codice = librocartaceo.CodiceLibro)
		inner join prenotazione_libro_cartaceo on
prenotazione_libro_cartaceo.CodiceLibro= libro.Codice
	group by libro.Codice
   order by NumPrenotazioni DESC
   limit 10;
END $
DELIMITER ;
#_____________________________________________________________

#Visualizzare la classifica degli e-book più acceduti
DELIMITER $
CREATE PROCEDURE EbookPiuAcceduti()
BEGIN
	Select CodiceLibro,Titolo, NumeroAccessi
   FROM libro INNER JOIN ebook ON Codice = CodiceLibro
   order by NumeroAccessi DESC
   limit 10;
```

```
END $
#_____________________________________________________________

#Prenotazione Posto Lettura
DELIMITER $
CREATE PROCEDURE PrenotaPosto(IN Giorno DATE,IN Orario INT, IN NumeroPosti
INT,IN EmailUtilizzatore VARCHAR(250), IN CodiceBiblioteca VARCHAR (50) )
#CREATE PROCEDURE PrenotaPosto(IN Inizio TIME)
BEGIN
        START TRANSACTION;
    INSERT INTO prenotazione_posto VALUES (Giorno, Orario, NumeroPosti,
EmailUtilizzatore, CodiceBiblioteca);
    #INSERT INTO prenotazione_posto (OraInizio) VALUES (OraInizio);
  COMMIT;
END $
DELIMITER ;
#_____________________________________________________________

#Prenotazione Libro Cartaceo
DELIMITER $
CREATE PROCEDURE PrenotaLibroCartaceo(IN GiornoInizio DATE, IN DataFine DATE,
IN CodiceLibro INT,IN EmailUtilizzatore VARCHAR(250))
BEGIN
            if exists(Select 1 From librocartaceo Where librocartaceo.CodiceLibro =
CodiceLibro AND librocartaceo.StatoPrestito="Disponibile" AND
librocartaceo.StatoConservazione !="Scadente") THEN
    INSERT INTO prenotazione_libro_cartaceo
(DataAvvio,CodiceLibro,DataFine,EmailUtilizzatore) VALUES (GiornoInizio,CodiceLibro,
DataFine,EmailUtilizzatore);
    #UPDATE librocartaceo SET StatoPrestito = "Prenotato" WHERE
librocartaceo.CodiceLibro = CodiceLibro;
    END IF;
END $
DELIMITER ;
#_____________________________________________________________

#Visualizzazione Proprie Prenotazioni di libri
DELIMITER $
CREATE PROCEDURE VisualizzaProprie PrenotazioniDiLibri(IN EmailUtilizzatore
VARCHAR(50))
BEGIN
```

```
        Select prenotazione_libro_cartaceo.Codice as CodicePrenotazione, CodiceLibro,
Titolo, DataAvvio, DataFine
   FROM prenotazione_libro_cartaceo inner join libro on
libro.Codice=prenotazione_libro_cartaceo.CodiceLibro
   where EmailUtilizzatore=prenotazione_libro_cartaceo.EmailUtilizzatore;
END $
DELIMITER ;
#_____________________________________________________________

#Visualizzazione Proprie Prenotazioni di posti lettura
DELIMITER $
CREATE PROCEDURE VisualizzaProprie PrenotazioniPostiLettura(IN EmailUtilizzatore
VARCHAR(50))
BEGIN
        Select Data, Orario, Nome,CodiceBiblioteca, NumeroPosti as NumeroPosto
   FROM prenotazione_posto inner join biblioteca on
prenotazione_posto.CodiceBiblioteca=biblioteca.Codice
   where EmailUtilizzatore=prenotazione_posto.EmailUtilizzatore;
END $
DELIMITER ;
#_____________________________________________________________

#Visualizza Ebook
DELIMITER $
CREATE PROCEDURE VisualizzaEbook(IN Codice VARCHAR(50), IN EmailUtilizzatore
VARCHAR(50))

BEGIN
        IF EXISTS (Select * FROM ebook WHERE ebook.codicelibro = Codice)
   THEN
   UPDATE EBOOK SET numeroaccessi = numeroaccessi + 1 where Codice=
ebook.CodiceLibro;
   INSERT INTO accesso_utilizzatore_ebook(CodiceEbook, EmailUtilizzatore) VALUES
(Codice, EmailUtilizzatore);
   SELECT PDF FROM ebook WHERE ebook.codicelibro = Codice;
   END IF;
END $
DELIMITER ;
#_____________________________________________________________

#Visualizza eventi consegna
DELIMITER $
```

```
CREATE PROCEDURE VisualizzaEventiConsegna(IN EmailUtilizzatore VARCHAR(250))
BEGIN
	Select Data,Tipo,Note,EmailVolontario,CodicePrenotazioneLibro
   FROM (consegna INNER JOIN prenotazione_libro_cartaceo ON
CodicePrenotazioneLibro = Codice )
		inner join utilizzatore on prenotazione_libro_cartaceo.EmailUtilizzatore =
utilizzatore.EmailUtilizzatore
	Where EmailUtilizzatore = utilizzatore.EmailUtilizzatore;
END $
DELIMITER ;
#_____________________________________________________________

#Visualizza tutte le prenotazioni della piattaforma
DELIMITER $
CREATE PROCEDURE VisualizzaTuttePrenotazioni()
BEGIN
	Select prenotazione_libro_cartaceo.Codice,CodiceLibro,biblioteca.Nome as
NomeBiblioteca, EmailUtilizzatore, DataAvvio, DataFine
   FROM (prenotazione_libro_cartaceo inner join libro on CodiceLibro=libro.Codice)
inner join biblioteca on libro.CodiceBiblioteca=biblioteca.Codice;
END $
DELIMITER ;
#_____________________________________________________________

#Inserimento Evento Consegna
DELIMITER $
CREATE PROCEDURE InserisciEventoConsegna(IN Data DATE,IN Tipo
VARCHAR(100),IN Note VARCHAR(250),IN EmailVolontario VARCHAR(100), IN
CodPrenotazioneLibro INT, IN DataFinale DATE)
BEGIN
	#if exists(Select 1 From prenotazione_libro_cartaceo  Where
prenotazione_libro_cartaceo.Codice = CodicePrenotazioneLibro ) THEN
	INSERT INTO consegna VALUES
(Data,Tipo,Note,EmailVolontario,CodPrenotazioneLibro);
	update prenotazione_libro_cartaceo set prenotazione_libro_cartaceo.DataFine =
DataFinale where  prenotazione_libro_cartaceo.Codice = CodPrenotazioneLibro;
	#END IF;
END $
DELIMITER ;
#_____________________________________________________________

#Aggiornamento Evento Consegna
```

```
DELIMITER $
CREATE PROCEDURE AggiornaEventoConsegna(IN CodicePrenotazioneLibro_old
INT,IN Data DATE,IN Tipo VARCHAR(100),IN Note VARCHAR(250),IN EmailVolontario
VARCHAR(100))
BEGIN
        START TRANSACTION;
    UPDATE consegna
                SET
consegna.Data=Data,consegna.Tipo=Tipo,consegna.Note=Note,consegna.EmailVolontario
=EmailVolontario
                WHERE CodicePrenotazioneLibro_old =
consegna.CodicePrenotazioneLibro ;
    update consegna set consegna.tipo = "Restituzione" where CodicePrenotazioneLibro
= CodicePrenotazioneLibro_old;
  COMMIT;
END $
DELIMITER ;

#_____________________________________________________________

#Elimina Libro
DELIMITER $
CREATE PROCEDURE EliminaLibro(IN CodiceLibro INT)
BEGIN
        START TRANSACTION;
    DELETE FROM libro
                WHERE libro.Codice=CodiceLibro;
    DELETE FROM librocartaceo
                WHERE librocartaceo.CodiceLibro=CodiceLibro;
  COMMIT;
END $
DELIMITER ;
#_____________________________________________________________

#Inserimento Libro
DELIMITER $
CREATE PROCEDURE InserisciLibro(IN Titolo VARCHAR(100),IN Genere
VARCHAR(100),IN AnnoPubblicazione VARCHAR(100),IN NomeEdizione
VARCHAR(100), IN CodiceBiblioteca VARCHAR(50),IN NumeroPagine INT,IN
NumeroScaffale INT,IN StatoPrestito VARCHAR(50),IN StatoConservazione
VARCHAR(50))
BEGIN
```

```
        START TRANSACTION;
      INSERT INTO libro
(Titolo,Genere,AnnoPubblicazione,NomeEdizione,CodiceBiblioteca) VALUES
(Titolo,Genere,AnnoPubblicazione,NomeEdizione,CodiceBiblioteca);
      SELECT @cod := Codice FROM libro WHERE libro.Titolo = Titolo;
      INSERT INTO librocartaceo
(NumeroPagine,NumeroScaffale,StatoPrestito,StatoConservazione,CodiceLibro) VALUES
(NumeroPagine,NumeroScaffale,StatoPrestito,StatoConservazione,@cod);
  COMMIT;
END $
DELIMITER ;
#_____________________________________________________________

#modifica Libro
DELIMITER $
CREATE PROCEDURE ModificaLibro(IN Codice INT,IN Titolo VARCHAR(100),IN Genere
VARCHAR(100),IN AnnoPubblicazione VARCHAR(100),IN NomeEdizione
VARCHAR(100), IN CodiceBiblioteca VARCHAR(50),IN NumeroPagine INT,IN
NumeroScaffale INT,IN StatoPrestito VARCHAR(50),IN StatoConservazione
VARCHAR(50))
BEGIN
        START TRANSACTION;
      UPDATE libro
            SET libro.Titolo=Titolo,libro.Genere=Genere, libro.AnnoPubblicazione=
AnnoPubblicazione,libro.NomeEdizione= NomeEdizione,libro.CodiceBiblioteca=
CodiceBiblioteca
            WHERE Codice = libro.Codice;
  COMMIT;
END $
DELIMITER ;
#_____________________________________________________________

#Visualizza tutte le prenotazioni della piattaforma in una biblioteca
DELIMITER $
CREATE PROCEDURE VisualizzaTuttePrenotazioniDiUnaBiblio(IN CodiceBiblioteca
VARCHAR(50) )
BEGIN
        Select prenotazione_libro_cartaceo.Codice as
CodicePrenotazione,CodiceLibro,EmailUtilizzatore,DataAvvio,DataFine
   FROM (prenotazione_libro_cartaceo inner join libro on CodiceLibro=libro.Codice)
inner join biblioteca on libro.CodiceBiblioteca=biblioteca.Codice
   where CodiceBiblioteca =biblioteca.Codice;
```

```
END $
DELIMITER ;
#_____________________________________________________________

#Inserimento Messaggio per utilizzatore
DELIMITER $
CREATE PROCEDURE InviaMessaggio(IN Titolo VARCHAR(50),IN Testo
VARCHAR(200),IN EmailAmministratore VARCHAR(100),IN EmailUtilizzatore
VARCHAR(100))
BEGIN
        START TRANSACTION;
    INSERT INTO messaggio VALUES
(Titolo,Testo,CURDATE(),EmailAmministratore,EmailUtilizzatore);
  COMMIT;
END $
DELIMITER ;
#_____________________________________________________________

#Inserimento SEgnalazione comportamento
DELIMITER $
CREATE PROCEDURE InsSegnalazione(IN Nota VARCHAR(100),IN EmailAmministratore
VARCHAR(100),IN EmailUtilizzatore VARCHAR(100))
BEGIN
        START TRANSACTION;
    INSERT INTO segnalazione VALUES
(CURDATE(),Nota,EmailAmministratore,EmailUtilizzatore);
  COMMIT;
END $
DELIMITER ;
#_____________________________________________________________

#Rimuovi segnalazioni
DELIMITER $
CREATE PROCEDURE EliminaSegnalazioni(IN EmailUtilizzatore VARCHAR(100))
BEGIN
        START TRANSACTION;
    DELETE FROM segnalazione
            WHERE EmailUtilizzatore = segnalazione.EmailUtilizzatore;
    UPDATE utilizzatore
            SET utilizzatore.StatoAccount="Attivo"
            WHERE EmailUtilizzatore = utilizzatore.EmailUtilizzatore;
```

```
  COMMIT;
END $
DELIMITER ;
#_____________________________________________________________

#Visualizza i messaggi ricevuti per ogni utente
DELIMITER $
CREATE PROCEDURE VisualizzaMessaggiRicevuti(IN EmailUtente VARCHAR(100) )
BEGIN
        Select Titolo,Testo,Data,EmailAmministratore
   FROM messaggio
   where  messaggio.EmailUtilizzatore= EmailUtente;
END $
DELIMITER ;
#_____________________________________________________________

#Visualizza Segnalazioni ricevute
DELIMITER $
CREATE PROCEDURE VisualizzaSegnalazioni(IN EmailUtente VARCHAR(100) )
BEGIN
        Select Data,Nota,EmailAmministratore
   FROM segnalazione
   where EmailUtente = messaggio.EmailUtilizzatore;
END $
DELIMITER ;
#_____________________________________________________________

#Visualizza Numero Posti Biblioteca
DELIMITER $
CREATE PROCEDURE VisualizzaTotalePosti(IN CodiceBiblioteca INT )
BEGIN
        Select NumPosti
   FROM Biblioteca
   where codice = codiceBiblioteca;
END $
DELIMITER ;
#_____________________________________________________________

#Visualizza Indirizzo Biblioteca
DELIMITER $
CREATE PROCEDURE VisualizzaCodiceBiblioteca(IN CodiceBiblioteca INT )
BEGIN
```

```
        Select StatoConservazione
    FROM librocartaceo
    where librocartaceo.CodiceLibro = CodiceLibro;
END $
DELIMITER ;
#_____________________________________________________________


#TRIGGER


# TRIGGER per libro da disponibile a prenotato
CREATE TRIGGER DiventaPrenotato
AFTER INSERT ON prenotazione_libro_cartaceo
FOR EACH ROW
        UPDATE librocartaceo SET StatoPrestito = 'Prenotato'
    where NEW.CodiceLibro= librocartaceo.CodiceLibro ;

# TRIGGER per libro da prenotato a Consegnato
CREATE TRIGGER DiventaConsegnato
AFTER INSERT ON consegna
FOR EACH ROW
        UPDATE librocartaceo SET StatoPrestito = 'Consegnato'
        where librocartaceo.CodiceLibro In (
                select CodiceLibro
    from prenotazione_libro_cartaceo
    where Codice = New.CodicePrenotazioneLibro);


# TRIGGER per libro da Consegnato a Disponibile
CREATE TRIGGER DiventaDisponibile
AFTER UPDATE ON consegna
FOR EACH ROW
        UPDATE librocartaceo SET StatoPrestito = 'Disponibile'
        where librocartaceo.CodiceLibro In (
                select CodiceLibro
    from prenotazione_libro_cartaceo
    where new.Tipo="Restituzione");

#TRIGGER PER LA SOSPENSIONE DELLO STATO DELL'ACCOUNT
DELL'UTILIZZATORE
CREATE TRIGGER DisattivazioneAccount
```

```
AFTER INSERT ON segnalazione
FOR EACH ROW
        UPDATE utilizzatore SET StatoAccount = 'Sospeso'
   WHERE EmailUtilizzatore IN(SELECT NEW.EmailUtilizzatore
                                    FROM segnalazione
            GROUP BY EmailUtilizzatore
                                    HAVING COUNT(*)>= 3);
```