

BASI DI DATI 2020-2021



# E-BIBLIO



UN SISTEMA DI  
GESTIONE DELLE  
BIBLIOTECHE UNIBO

BUONA LETTURA



[elisa.ancarani4@studio.unibo.it](mailto:elisa.ancarani4@studio.unibo.it)

[giorgia.castelli2@studio.unibo.it](mailto:giorgia.castelli2@studio.unibo.it)

[lorenzo.pisano@studio.unibo.it](mailto:lorenzo.pisano@studio.unibo.it)

# INDICE

## Sommario

<b><i>ANALISI DEI REQUISITI</i></b> .....	<b>2</b>
Tavola dei volumi.....	5
Glossario dei dati .....	8
Lista delle operazioni.....	9
<b><i>Progettazione Concettuale</i></b> .....	<b>10</b>
Il modello ER .....	10
Dizionario delle entità .....	11
Dizionario delle relazioni.....	13
Tabella delle Business Rules .....	15
<b><i>Progettazione Logica</i></b> .....	<b>16</b>
Lista delle tabelle con i vincoli delle chiavi.....	16
Commenti a seguito della creazione delle tabelle.....	18
Analisi delle ridondanze.....	19
<b><i>Le funzionalità a livello implementativo</i></b> .....	<b>22</b>
<b><i>Le funzionalità di Ebiblio</i></b> .....	<b>23</b>

## DESCRIZIONE GENERALE DEL PROGETTO

EBIBLIO è una piattaforma multiutente che permette la gestione delle biblioteche dell'Università di Bologna, permettendo agli studenti Unibo di registrarsi e fruire di un servizio completo. Gli studenti possono, infatti, prenotare i libri di cui necessitano e, grazie alla presenza di volontari, è stata possibile la realizzazione di un servizio di consegna per ricevere i libri prenotati comodamente a casa. Ebiblio elimina il disagio di sale studio eccessivamente affollate e posti lettura introvabili, poiché uno studente, grazie al servizio di prenotazione, può garantirsi una postazione in cui studiare nella biblioteca che preferisce. Se viene fatto un uso improprio della piattaforma e dei servizi che essa offre, lo studente utilizzatore può essere segnalato da un amministratore della biblioteca, in caso di 3 segnalazioni, l'utente viene sospeso dal servizio. Credendo nell'utilità del servizio e nel suo corretto uso da parte degli studenti, auguriamo a tutti una buona permanenza in Ebiblio!

Con la seguente relazione, si presenta l'analisi dei requisiti effettuata dal gruppo su cui poi si baserà l'intero sviluppo della piattaforma, descrivendo le relative modalità di implementazione e organizzazione.

## 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. Ii 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). Infine, si vuole tenere traccia di tutti gli eventi che occorrono nella piattaforma, relativamente all'inserimento di nuovi dati (es. nuovi utenti, nuovi libri cartacei, etc). Tali eventi vanno inseriti, sotto forma di messaggi di testo, all'interno di un log, implementato in un'apposita collezione MongoDB.

---

Dopo una multipla e attenta lettura del documento di specifica, siamo partiti da una sua iniziale decomposizione:

### **Frasi Relative Alla Biblioteca:**

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. La biblioteca mette a disposizione del pubblico l'accesso ad i propri libri.

### **Frasi Relative Ai Posti Lettura**

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.

### **Frasi Relative Agli Utenti**

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).

### **Frasi Relative Alle Prenotazione Posti Lettura:**

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.

### **Frasi Relative Alla Prenotazione Libro Cartaceo:**

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).

### **Frasi Relative Ai Messaggi:**

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.

### **Frasi Relative Alle Segnalazioni:**

Ogni segnalazione dispone di data ed eventuale nota di testo, è inserita da un amministratore e diretta verso un utilizzatore.

**Tavola dei volumi.**

La tavola dei volumi stima il numero delle istanze che bisogna gestire nel modello.

CONCETTO	TIPO	VOLUME
Biblioteca	Entità	100
Foto	Entità	100
Posto Lettura	Entità	1000
Utente	Entità	15.180
Amministratore	Entità	100
Volontario	Entità	80
Utilizzatore	Entità	15000
Messaggio	Entità	200
Libro	Entità	510.000
Libro Cartaceo	Entità	500.000
Ebook	Entità	10.000
Prenotazione	Entità	5000
Consegna	Entità	5000
Recapito	Entità	100
Autore	Entità	5000
Segnalazione	Entità	50
Prenotazione Posto Lettura	Entità	1000

Considerando che in unibo ci sono 96 biblioteche presenti (le fonti sono ricavate dal sito <https://sba.unibo.it/it/biblioteche?SearchableText=&typebib=&campus=&showas=list> mettiamo una media di volumi pari a 100.

Il numero di utenti utilizzatori è stimato a partire dagli studenti presenti quest'anno all'interno di unibo: 18590, supponiamo anche il fatto che non tutti gli studenti presenti in unibo siano iscritti al servizio, per questo abbiamo scelto una media dei volumi pari a 15000.

E' bene che la tabella dei volumi includa anche le *relazioni*, poiché anche queste devono essere opportunamente tradotte nel modello di traduzione. Dunque qui sotto presentiamo una tabella dei volumi che stima il numero medio di occorrenze di una relazione.

CONCETTO	TIPO	VOLUME
Galleria	Relazione	100
Utenza	Relazione	100
Libri Disponibili	Relazione	500,000
Scrittori	Relazione	1,020,000
Libri Prenotati	Relazione	1,000,000
Possiede Posto Lettura	Relazione	2000
Effettua Prenotazione	Relazione	15000
Riceve Messaggio	Relazione	15000
Inserisce Messaggio	Relazione	250
Inserisce Segnalazione	Relazione	25
Riceve Segnalazione	Relazione	15000
Aggiunge Consegna	Relazione	1600
Partecipa Consegna	Relazione	30,000
Riferimento Consegna	Relazione	5000
Accede Ebook	Relazione	500
Responsabile	Relazione	100

Assunzione 1: ogni biblioteca ha in media una foto

Assunzione 2: ogni biblioteca ha in media un recapito

Assunzione 3: ogni biblioteca ha in media 5000 libri disponibili

Assunzione 4: ogni libro ha in media 2 scrittori

Assunzione 5: un libro cartaceo ha in media 2 prenotazioni

Assunzione 6: una biblioteca ha in media 20 posti lettura

Assunzione 7: ogni utilizzatore effettua in media 10 prenotazioni

Assunzione 8: ogni utilizzatore riceve in media 1 messaggio

Assunzione 9: ogni amministratore inserisce in media 50 messaggi

Assunzione 10: ogni amministratore inserisce in media 5 segnalazioni

Assunzione 11: ogni utente utilizzatore mediamente riceve 1 segnalazione

Assunzione 12: un volontario inserisce mediamente 20 consegne

Assunzione 13: un utilizzatore partecipa in media a 2 consegne

Assunzione 14: una consegna si riferisce a una prenotazione

Assunzione 15: ogni amministratore è responsabile di una biblioteca

## Glossario dei dati

TERMINE	DESCRIZIONE	SINONIMI	COLLEGAMENTI
Biblioteca	Biblioteca del sistema da gestire		Amministratore, Libro, Posto Lettura, Foto, Recapito
Posto Lettura	Posti lettura messi a disposizione dalle biblioteche		Biblioteca, Prenotazione Posto Lettura
Galleria Immagini	Foto relative alle biblioteche		Biblioteca
Recapito	Recapiti relativi alle biblioteche		Biblioteca
Libro	Libri presenti nelle biblioteche, si distinguono in libri cartacei ed ebook		Biblioteca, Autore, Utente utilizzatore, Prenotazione
Autore	Autore del libro		Libro
Prenotazione	Prenotazione di un libro cartaceo		Libro Cartaceo, Consegnna, Utilizzatore
Utenti	Utenti registrati alla piattaforma, distinti tra amministratori, volontari e utilizzatori	Amministratori (dipendenti unibo)	Biblioteca, Messaggio, Segnalazione, Consegnna, Prenotazione Posto Lettura, Libro,
Messaggio	Messaggi scambiati tra utenti della piattaforma (amministratori mandano messaggi a utilizzatori)		Utente amministratore, utente utilizzatore
Consegnna	Consegne inserite da utenti volontari che si riferiscono alla prenotazione di libri cartacei		Utente volontario, Utente utilizzatore, Prenotazione
Segnalazione	Le segnalazioni sono effettuate da un amministratore in caso di comportamento scorretto da parte di un utilizzatore		Utente utilizzatore, Utente amministratore
Prenotazione Posto Lettura	Prenotazione di un posto lettura da parte di un utilizzatore		Utente utilizzatore, Posto lettura

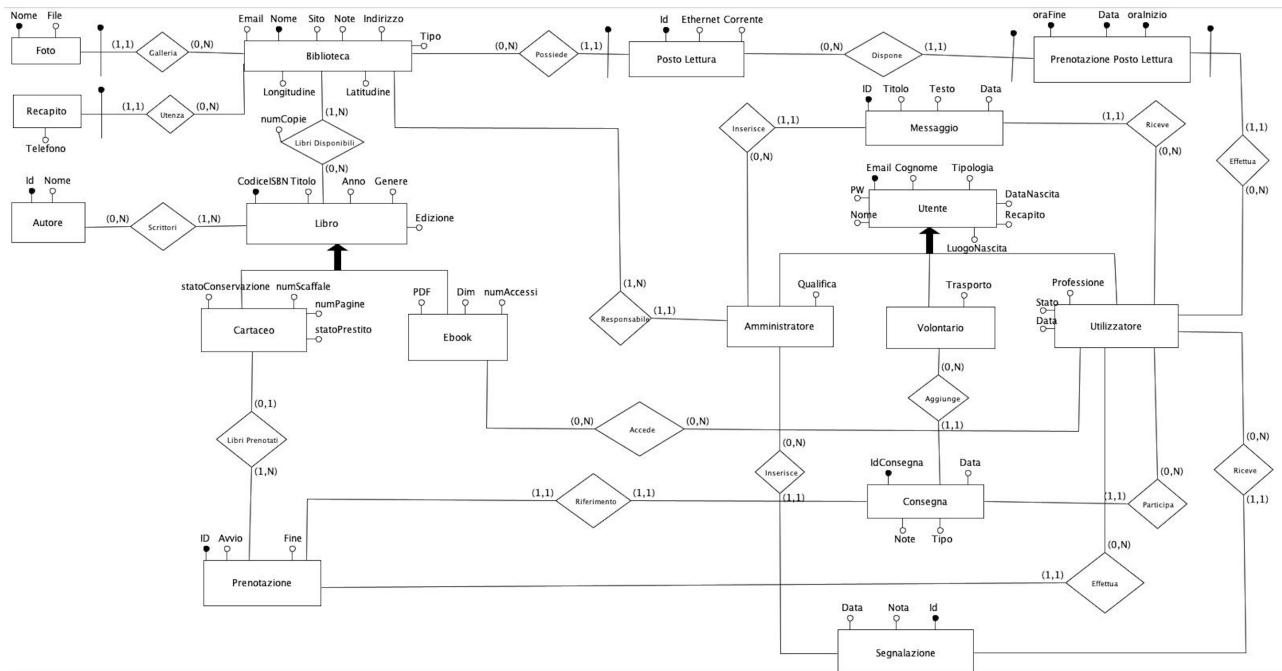
**Lista delle operazioni**

1. Registrazione alla piattaforma da parte degli utenti utilizzatori
2. Aggiornamento profilo utente utilizzatore
3. Inserimento immagini relative a una biblioteca
4. Inserimento di una biblioteca
5. Inserimento/cancellazione/aggiornamento di un libro sia cartaceo che ebook da parte di un amministratore
6. Inserimento di una biblioteca
7. Inserimento/cancellazione/aggiornamento di un autore da parte di un amministratore
8. Prenotazione di un posto lettura da parte di un utilizzatore
9. Prenotazione di un libro cartaceo da parte di un utilizzatore
10. Visualizzazione di un ebook da parte di un utilizzatore
11. Inserimento/cancellazione/aggiornamento di un evento di consegna
12. Inserimento di un messaggio da parte di un amministratore
13. Visualizzazione di un messaggio da parte di un utilizzatore
14. Inserimento/cancellazione di una segnalazione da parte di un amministratore
15. Visualizzazione di una segnalazione da parte di un utilizzatore
16. Visualizzazione lista biblioteche
17. Visualizzazione statistiche
18. Visualizzazione evento di consegna
19. Inserimento di un evento di consegna da parte di un volontario
20. Visualizzazione dettaglio profilo di un utente

# Progettazione Concettuale

## Il modello ER

Per tradurre il documento di specifica nel corrispondente diagramma ER siamo partiti utilizzando la strategia di progettazione inside out. Questa strategia consiste nell'identificare entità e relazioni raffinandole aggiungendo dettagli ancor prima di collegarle attraverso relazioni. Ovvero si individuano i concetti più importanti e a partire da questi si procede verso concetti correlati con un'estensione “a macchia d'olio”. Abbiamo scelto questa strategia a discapito delle altre (mista, top-down, bottom-up) perché riteniamo che segua un approccio più intuitivo ed efficiente per il nostro studio di analisi dei requisiti.



Per una migliore visualizzazione del diagramma, si mette in condivisione la cartella drive:

[https://liveunibo-my.sharepoint.com/:f/g/personal/elisa\\_ancarani4\\_studio\\_unibo\\_it/EIVok4jl\\_iINqWa5kVc2OrsBh6SRDweZOy4kgYXQ5-fsWw?e=BmsXGF](https://liveunibo-my.sharepoint.com/:f/g/personal/elisa_ancarani4_studio_unibo_it/EIVok4jl_iINqWa5kVc2OrsBh6SRDweZOy4kgYXQ5-fsWw?e=BmsXGF)

**Dizionario delle entità**

Entità	Descrizione	Attributi	Identificatore
Biblioteca	Biblioteche unibo	Email, Latitudine, Longitudine, Indirizzo, Nome, Sito, Note, Recapito	Nome
Foto	Foto appartenenti alla galleria	Nome, File	Nome
Recapito	Recapito delle biblioteche unibo	Telefono	NomeBiblioteca
Posto Lettura	Posto lettura all'interno della Biblioteca	ID, Ethernet, Corrente	ID, NomeBiblioteca
Prenotazione Posto Lettura	Prenotazione posti lettura all'interno di una biblioteca	Id, Oralnizio, OraFine, Data	IdPostoLettura, Oralnizio, OraFine, Data, EmailUtilizzatore
Libro	Libro presente nella biblioteca	CodiceISBN, Titolo, Anno, Genere, Nome Edizione	CodiceISBN
L Cartaceo	Tipo di Libro	statoConservazione, statoPrestito, numeroPagine, numeroScaffale	CodiceISBN
L E-Book	Tipo di Libro	PDF, Dimensione, Numero di Accessi	CodiceISBN
Autore	Autore di un libro	Id, Nome	Id
Utente	Utente che interagisce con la piattaforma Ebiblio	Email, Password, Nome, Cognome, Data di Nascita, Luogo di Nascita, Recapito	Email
U Amministratore	Tipo di Utente	Qualifica	EmailUtente
U Volontario	Tipo di Utente	Trasporto	EmailUtente

U Utilizzatore	Tipo di Utente	Data iscrizione, Stato, Professione	EmailUtente
Messaggio	Messaggio inviato/ricevuto su Ebiblio	ID, Data, Titolo, Testo	ID
Segnalazione	Segnalazione inviata/ricevuta su Ebiblio	ID, Nota, Data	ID
Prenotazione	Prenotazione libro cartaceo	ID, Avvio, Fine	ID
Consegna	Consegna libro cartaceo	IDConsegna, Note, Tipo, Data	IDConsegna

**Dizionario delle relazioni**

Relazione	Descrizione	Tipo	Componenti	Attributi
Galleria	associa la Biblioteca alle sue Foto	0 a N	Foto, Biblioteca	
Possiede	Associa la biblioteca ai posti lettura	1 a N	Biblioteca, Posto Lettura	
LibriDisponibili	Associa Biblioteca a Libro	1 a N	Biblioteca, Libro	numeroCopie
Utenza	Associa Biblioteca ai recapiti telefonici che possiede	0 a N	Biblioteca, Recapito	
Dispone	Associa i posti lettura alle relative prenotazioni	0 a N	Posto Lettura, Prenotazione Posto Lettura	
Responsabile	Associa un Amministratore alla biblioteca	1 a N	Amministratore, Biblioteca	
Inserisce	Associa un Amministratore ai Messaggi	0 a N	Amministratore, Messaggio	
Inserisce	Associa un Amministratore alla Segnalazione da lui effettuata	0 a N	Amministratore, Segnalazione	
Riceve	Associa l'Utilizzatore ai Messaggi ricevuti	0 a N	Utilizzatore, Messaggio	

Riceve	Associa L'utilizzatore alle Segnalazioni ricevute	0 a N	Utilizzatore, Segnalazione	
Accede	Associa l'Utilizzatore all'E-Book a cui accede	0 a N	Utilizzatore, E-Book	
Prenota	Associa l'Utilizzatore al Posto Lettura prenotato	0 a N	Utilizzatore, Posto Lettura	
Partecipa	Associa l'Utilizzatore alla Consegnna del Libro Cartaceo	0 a N	Utilizzatore, Consegnna	
Aggiunge	Associa il Volontario che aggiunge un evento di Consegnna	0 a N	Volontario, Consegnna	
Riferimento	Associa una Prenotazione alla Consegnna	1 a 1	Prenotazione, Consegnna	
LibriPrenotati	Associa il Libro Cartaceo alla Prenotazione	1 a N	Cartaceo, Prenotazione	
Effettua	Associa l'Utilizzatore alla Prenotazione	0 a N	Utilizzatore, Prenotazione	
Scrittori	Associa il libro agli autori	1 a N	Libro, Autore	

Ci sono alcuni vincoli non modellabili tramite il diagramma ER, utilizziamo per questo una tabella di business rules. Per preparare una tabella delle business rules completa è fondamentale fare una distinzione iniziale tra vincoli di integrità e vincoli di derivazione. I vincoli di integrità sono quelle affermazioni che devono essere sempre verificate nella nostra base di dati, mentre i vincoli di derivazione specificano le operazioni (aritmetiche, logiche) grazie alle quali possiamo ottenere un concetto derivato.

### **Tabella delle Business Rules**

#### **Vincoli di integrità:**

- Campo qualifica degli amministratori è un campo di testo di massimo 10 caratteri
- Il campo note dell'entità prenotazione possono avere al massimo 200 caratteri

#### **Vincoli di derivazione:**

- Quando l'utilizzatore riceve cumulativamente più di tre segnalazioni lo stato viene sospeso.

## Progettazione Logica

---

### **Lista delle tabelle con i vincoli delle chiavi**

Biblioteca(**Nome**, Indirizzo, Email, Sito, Note, Latitudine, Longitudine)

Foto(**Nome**, nomeBiblioteca)

Recapito(**NomeBiblioteca**, Telefono)

PostoLettura(**Id**, nomeBiblioteca,, Ethernet, Corrente)

Libro(**CodiceISBN**, Titolo, Anno, Genere, nomeEdizione, Tipologia)

LibriDisponibili(**nomeBiblioteca**, CodiceISBN\_Libro, numCopie)

Autore(**Id**, NomeAutore)

Scrittori(**idAutore**, CodiceISBN\_Libro)

Cartaceo(**CodiceISBN**, statoConservazione, statoPrestito, numeroPagine, numScaffale)

Autore(**Id**, NomeAutore)

Scrittori(**idAutore**, CodiceISBN\_Libro)

Cartaceo(**CodiceISBN**,, statoConservazione, statoPrestito, numeroPagine, numeroScaffale)

Ebook(**CodiceISBN**, PDF, Dimensione, Numero\_accessi)

PrenotazioneCartaceo(**Id\_prenotazione**, CodiceISBN\_cartaceo, Avvio\_prenotazione, FinePrenotazione, EmailUtilizzatore)

Consegna(**IdPrenotazioneCartaceo**, EmailVolontario, EmailUtilizzatore, Note, Tipo, Data)

AccessoEbook(**CodiceISBN\_ebook**, **Email\_utilizzatore**)

Utente(**Email**, Nome, Cognome, Password, DataNascita, LuogoNascita, Recapito, TipoUtente)

Amministratore(**EmailUtente**, Nome\_Biblioteca, Qualifica)

Volontario(**EmailUtente**, Mezzo\_di\_trasporto)

Utilizzatore(**EmailUtente**, Professione, Stato, DataAccount)

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

Segnalazione(**Id**, EmailUtilizzatore, EmailAmministratore, Data, Nota)

PrenotazionePostoLettura(**IdPostolettura**, **EmailUtilizzatore**, **OraInizio**, **OraFine**, Data)

Foto.nomeBiblioteca -> Biblioteca.Nome

Recapito.nomeBiblioteca -> Biblioteca.Nome

PostoLettura.nomeBiblioteca -> Biblioteca.Nome

LibriDisponibili.nomeBiblioteca -> Biblioteca.Nome

LibriDisponibili.CodiceISBN\_Libro -> Libro.CodiceISBN

Cartaceo.CodiceISBN\_cartaceo -> Libro.CodiceISBN

Ebook.CodiceISBN\_ebook -> Libro.CodiceISBN

Scrittori.CodiceISBN\_Libro -> Libro.CodiceISBN

Scrittori.IdAutore -> Autore.Id

PrenotazioneCartaceo.CodiceISBNCartaceo -> Cartaceo.CodiceISBN

PrenotazioneCartaceo.EmailUtilizzatore -> Utilizzatore.EmailUtente

Consegna.EmailVolontario -> Volontario.EmailUtente

Consegna.EmailUtilizzatore -> Utilizzatore.EmailUtente

AccessoEbook.EmailUtilizzatore -> Utilizzatore.EmailUtente

AccessoEbook.CodiceISBN -> Ebook.CodiceISBN

Amministratore.EmailUtente -> Utente.Email

Volontario.EmailUtente -> Utente.Email

Utilizzatore.EmailUtente -> Utente.Email

Messaggio.EmailAmministratore -> Amministratore.EmailUtente

Messaggio.EmailUtilizzatore -> Utilizzatore.EmailUtente

Segnalazione.EmailAmministratore -> Amministratore.EmailUtente

Segnalazione.EmailUtilizzatore -> Utilizzatore.EmailUtente

PrenotazionePostoLettura.EmailUtilizzatore -> Utilizzatore.EmailUtente

PrebotazionePostoLettura.idPostoLettura -> PostoLettura.id

### Commenti a seguito della creazione delle tabelle.

La decisione di implementare una tabella PrenotazionePostoLettura, prendendo come chiavi primarie gli attributi: **Id\_posto\_lettura**, **Email\_utilizzatore**, **Ora\_inizio**, **Ora\_fine**, **Data**, nasce per consentire ad un utente utilizzatore di prenotarsi più volte, a orari differenti durante il corso della giornata. **ora\_inizio** e **ora\_fine** sono anch'esse chiavi primarie per evitare di “accavallare” gli orari, dunque una persona non può, per esempio, prenotarsi due volte dalle 10-12 o dalle 11-12 ma può prenotare il suo posto lettura nella biblioteca d'interesse dalle 10-11 e dalle 12-13.

Il campo numeroCopie nasce dall'esigenza di gestire il caso in cui una biblioteca metta a disposizione più copie dello stesso libro cartaceo. Inizialmente l'idea era ricaduta sul fatto di aggiungere un id auto incrementale per consentire a un amministratore di aggiungere più volte lo stesso libro in caso di copie multiple, proposta che abbiamo subito scartato poiché oltre che ad essere scomodo per un utente aggiungere più righe per lo stesso libro, avrebbe introdotto un problema ancora maggiore: una ridondanza di tipo logico. Per questo è stato introdotto il campo numero copie nella relazione LibriDisponibili\*. Problema che non si è presentato con l'ebook, la cui modalità è spiegata nel paragrafo sottostante.

Nella relazione LibriDisponibili sono presenti unicamente le coppie biblioteca-libro, con il relativo numero delle copie del libro, escludendo le copie libro-ebook

L'ebook è stato pensato dal team di progetto in maniera differente dal libro cartaceo. Quando una biblioteca rende disponibile un ebook, lo mette a disposizione dell'intera piattaforma. Ad un utente utilizzatore basta cercare l'ebook di interesse e, se lo trova, può tranquillamente leggerlo. Dunque non importa a quale biblioteca appartiene l'ebook poiché è come se, una volta caricato, appartenesse a tutte le biblioteche di ebiblio.

Il team ha deciso di mantenere l'entità libro aggiungendo un campo tipologia Libro. La tipologia del libro può essere di tre tipi: “cartaceo”, “ebook” ma anche “entrambi”. Data la differente gestione delle due entità figlie cartaceo e ebook, ci è sembrato opportuno aggiungere una tipologia libro per poter gestire al meglio e in maniera più “pulita” le differenze spiegate qui sopra.

Il campo tipologia utente è stato pensato per facilitare le operazioni di ricerca. Questo campo rende più leggera ed efficiente la stesura del codice, portando a una diminuzione del costo computazionale. Un'analisi più approfondita sul campo tipologia utente viene presentata nel paragrafo sottostante “Analisi delle ridondanze”.

### Analisi delle ridondanze.

Viene presa in considerazione l'ipotesi di un campo 'NUMERO\_ACCESSI' relativo ad ogni E-BOOK. Il mantenimento di questa variabile viene valutato in base alla lista di operazioni presenti in specifica;

- Inserire un nuovo ebook (3 volte/mese, **interattiva**)
- Rimuovere un ebook (3 volte/mese, **batch**)
- Contare il numero degli accessi relativo ad ogni e-book (2 volte/mese, interattiva)

Costanti:

- $\alpha$  (peso operazioni scrittura) = 2
- $wI$  (peso operazioni interattive) = 1
- $wB$  (peso operazioni batch) = 0,5

La formula per calcolare il costo delle operazioni è la seguente:

$$C(Ot) = f(Ot) * wt * (\alpha * NCwrite + NCread)$$

### Inserimento di un nuovo ebook:

L'inserimento di un ebook non comporta l'associazione diretta con una biblioteca, per cui il numero di accessi è pari a 1.

CONCETTO	COSTRUTTO	ACCESSI	TIPO
Ebook	Entità	1	Writing

$$C(op1) = 3 * 1 * (2 * 1 + 0) = 6$$

### Rimozione di un ebook:

In maniera analoga all'inserimento, è possibile rimuovere una riga dalla tabella libro, dove le modifiche si propagano a cascata verso la tabella ebook.

CONCETTO	COSTRUTTO	ACCESSI	TIPO
Ebook	Entità	1	Writing

$$C(op2) = 3 * 0,5 * (2 * 1 + 0) = 3$$

### Conteggio del numero di accessi di ogni ebook:

Per contare il numero di accessi agli ebook da parte degli utenti utilizzatori, senza il campo numero accessi, bisogna rifarsi alla tabella dei volumi. Nella tabella dei volumi, il numero degli accessi è stato ipotizzato a 500, per cui l'operazione avrà costo:

$$c(op3) = 2 * 1 * (2 * 0 + 500) = 1000$$

Il **costo totale** dell'operazioni senza considerare la ridondanza numero accessi è:

$$C(tot) = 6+3+1000 = 1009$$


---

Prendiamo ora in considerazione la ridondanza concettuale 'NUMERO\_ACCESSI'. I costi di inserimento e rimozione delle prime due operazioni rimangono invariati. Nell'ultima operazione si ha, invece, un cambiamento. Non dobbiamo appellarcia alla tabella dei volumi, ma il numero degli accessi è gestito direttamente dalla variabile numero accessi, rendendo necessaria una sola operazione di lettura.

CONCETTO	COSTRUTTO	ACCESSI	TIPO
Ebook	Entità	1	Reading

$$c(op3) = 2 * 1 * (2 * 0 + 1) = 4$$

Il **costo totale** dell'operazione mantenendo la ridondanza concettuale numero\_accessi è:

$$C(tot) = 6 + 3 + 4 = 13$$


---

Per calcolare la memoria si deve prima considerare quanto costa la memorizzazione del campo aggiuntivo numeroAccessi.

Considerando dalla tabella dei volumi 15.000 utenti e ogni utente effettua una media di 5 accessi

$$15.000 * 5 = 75.000 = 2^{17} \rightarrow 17 \text{ bit per la memorizzazione del campo, dunque 3 byte.}$$

**Memoria occupata:**  $X + 5 * 3 = X + 15$  Bytes  $\rightarrow$  15 bytes di overhead di memoria

**Speed Up:**  $1000 / 13 = 76$

Quando lo Speed Up > 1 il costo operazionale viene ridotto attraverso la ridondanza, in questo caso abbiamo uno Speed Up di 76 dunque abbiamo un'ottima riduzione a fronte di una maggior occupazione di memoria di 15 bytes. Tuttavia 15 bytes in più in una base di dati sono una quantità di memoria occupata per la ridondanza veramente irrisoria. In conclusione, mantenere la ridondanza concettuale numeroAccessi conviene.

### Analisi delle ridondanze sul campo tipologia Utente

Il campo tipologia Utente viene in aiuto soprattutto durante il controllo dei permessi. Senza questa ridondanza nella tabella padre utente, ogni volta che si effettua un check sui permessi bisognerebbe fare 3 controlli, uno per ogni entità figlia. Questo, oltre che a risultare particolarmente scomodo a livello di programmazione, si ripercuote anche sul costo computazionale. Si dimostra grazie all'analisi delle ridondanze.

- Ricercare un utente (3 volte/mese, **interattiva**)

Ricerca di un utente *con* ridondanza concettuale tipologia utente:

$$C(op) = 3 * 1 * (2^0 + 1) = 3$$

Ricerca di un utente *senza* ridondanza concettuale tipologia utente:

$$C(op) = 3 * 1 * (2^0 + 3) = 6$$

# DESCRIZIONI FEATURES IMPLEMENTATE

## Le funzionalità a livello implementativo

### Il database:

Il database è composto interamente dalle tabelle analizzate a pagina 16 e da trigger che scattano ogni volta che si verifica un determinato evento. Il modello di database utilizzato è il modello relazionale che ci ha permesso di garantire una corretta gestione dei dati creando una rappresentazione logica e solida della base di dati Ebiblio.

### HTML e Php:

La parte relativa alla grafica e alla user experience è stata implementata in HTML, con l'integrazione di CSS per migliorare ulteriormente la grafica a livello utente e parti di javascript. All'interno dell'html viene importato bootstrap, un toolkit che fornisce un solido supporto alla gestione dell'html e al css, per poter creare pagine maggiormente user-friendly. La parte di javascript, è stata di fondamentale importanza per la creazione di open street map, fornendo così una visualizzazione su mappa delle biblioteche presenti all'interno di ebiblio.

Mentre HTML, Css o JavaScript sono interpretati prima dal browser quando la web page viene aperta, il codice PHP è già eseguito sul web server, infatti i file php consentono di comunicare col database. All'interno dei file php sono implementate le query che permettono di effettuare operazioni sulla base di dati ebiblio.

### Log:

In una piattaforma non può mancare la gestione dei log, questa è stata effettuata instaurando una connessione tra MongoDB e Php. Grazie ai file di log si registrano tutte le attività che avvengono all'interno di Ebiblio. I file di log in generale svolgono un ruolo fondamentale, per esempio, permettono di rilevare accessi non autorizzati, eventuali anomalie o transazioni fallite.

## Le funzionalità di Ebiblio

---

La piattaforma viene ideata per essere utilizzata sia dall'utente registrato (amministratore, utilizzatore, volontario) sia dall'utente che si interfaccia per la prima volta con il sito. Come si può dedurre, quest'ultimo avrà funzionalità estremamente limitate rispetto agli altri utilizzatori di cui si conosce l'identità.

La piattaforma si presenta dunque con un'interfaccia che permette di ricercare un libro in qualsiasi biblioteca presente in UNIBO. Se questo esiste, verranno mostrati i dettagli relativi, altrimenti, sarà stampato a schermo un messaggio di errore. Questa è la principale funzionalità che è disponibile anche per gli utenti non iscritti, oltre alla visualizzazione della mappa delle biblioteche e alle statistiche di Ebiblio.

Entrando nel vivo del progetto, è presente una parte dedicata al login. Qui, è possibile distinguere il tipo di utente e presentargli una dashboard con le operazioni che gli è permesso svolgere.

Riassumendo:

- Per l'utente volontario sarà possibile creare un nuovo evento di consegna, specificando l'utilizzatore e l'id della prenotazione con tutti i suoi dettagli.
- Per l'utente utilizzatore sarà possibile:
  - prenotare un libro cartaceo, se questo è disponibile
  - accedere a un ebook, se questo è disponibile
  - prenotare un posto lettura all'interno di una biblioteca UNIBO
  - ricevere un messaggio da parte di un amministratore
  - ricevere una segnalazione da parte di un amministratore
  - ricevere una partecipazione ad un evento di consegna.
- Per l'utente amministratore sarà possibile (per la biblioteca di cui è responsabile)
  - Inserire/cancellare/aggiornare un nuovo libro
  - inserire una segnalazione per un utente utilizzatore
  - inviare un messaggio diretto ad un utente utilizzatore
  - modificare i dati della biblioteca
  - inserire foto e recapiti di una biblioteca

Successivamente, viene prevista una parte riservata agli amministratori del gestionale. Questi, sono gli unici ad avere accesso ad un account esclusivo. Grazie a questa, è possibile aggiungere Biblioteche, Autori e Libri. In conclusione, viene previsto un servizio di manutenzione da quest'ultima categoria di utenti, per questo motivo gli viene data la possibilità di modificare, aggiungere, eliminare libri, modificare informazioni relative a biblioteche ed aggiungere, eliminare autori.

## Data Mining - Weka

---

Per implementare l'algoritmo di clustering abbiamo utilizzato il tool di Weka. Il team ha instaurato una connessione con workbench e, per segmentare gli utenti utilizzatori sulla base della professione, età e numero di prestiti cartacei effettuati, è stata inserita nell'apposita sezione weka la seguente query:

```
SELECT Professione, YEAR(CURRENT_TIMESTAMP) - YEAR(DataNascita) -  
(RIGHT(CURRENT_TIMESTAMP, 5) < RIGHT(DataNascita, 5)) as age, count(*) AS  
NumPrenotazioni  
FROM Utente, Utilizzatore, prenotazionecartaceo  
where Email = EmailUtente AND EmailUtente = EmailUtilizzatore AND TipoUtente =  
"Utilizzatore"  
Group By Professione, Age;
```

Da qui l'algoritmo del tool ci ha permesso di effettuare un'analisi sui dati presenti a DB.

## Ebiblio – Il codice

### La gestione dei permessi:

```

if($_SESSION['TipoUtente']=="Amministratore"){
    echo "<script> alert('Non possiedi le credenziali per accedere a questa pagina');
    window.location.href='../../home/adminHome.php'</script>";
} else if($_SESSION['TipoUtente']=="Utilizzatore"){
    echo "<script> alert('Non possiedi le credenziali per accedere a questa pagina');
    window.location.href='../../home/myHome.php'</script>";
} else if($_SESSION['TipoUtente']=="Volontario"){
    echo "<script> alert('Non possiedi le credenziali per accedere a questa pagina');
    window.location.href='../../home/volHome.php'</script>";
} else if($_SESSION['TipoUtente']==""){
    echo "<script> alert('Non possiedi le credenziali per accedere a questa pagina');
    window.location.href='../../home/home.php'</script>";
}

```

### Un esempio di inserimento:

```

if(isset($_POST['submit'])){
    $nomeUtente= $_POST['nomeUtente'];
    $cognomeUtente = $_POST['cognomeUtente'];
    $emailUtente = $_POST['emailUtente'];
    $password = $_POST['password'];
    $password = md5($password);
    $dataNascita= $_POST['dataNascita'];
    $luogoNascita = $_POST['luogoNascita'];
    $recapito = $_POST['recapito'];
    $nomeEncode = $_POST['biblio'];
    $qualifica = $_POST['qualifica'];

    $amministratore = "Amministratore";

    $sql = $pdo->prepare("INSERT INTO Utente VALUES(?, ?, ?, ?, ?, ?, ?, ?, ?)");

    $sql->bindParam(1, $emailUtente, PDO::PARAM_STR);
    $sql->bindParam(2, $nomeUtente, PDO::PARAM_STR);
    $sql->bindParam(3, $cognomeUtente, PDO::PARAM_STR);
    $sql->bindParam(4, $password, PDO::PARAM_STR);
    $sql->bindParam(5, $dataNascita, PDO::PARAM_STR);
    $sql->bindParam(6, $luogoNascita, PDO::PARAM_STR);
    $sql->bindParam(7, $recapito, PDO::PARAM_STR);
    $sql->bindParam(8, $amministratore, PDO::PARAM_STR);

    $res = $sql->execute();
}

```

### Il login:

```

<?php

require '../../connectionDB/connection.php';

if(isset($_POST['submit'])){
    $emailUtente = $_POST['email'];
    $passwordUtente = $_POST['password'];
    $passwordUtente = md5($passwordUtente);

    try{
        $sql = "SELECT * FROM utente WHERE Email='$emailUtente'";
        $res = $pdo->query($sql);
    }catch(PDOException $e){echo $e->getMessage();}

    if($res->rowCount() > 0){
        $_SESSION['EmailUtente'] = $emailUtente;
        try{
            $sql = "SELECT TipoUtente FROM utente WHERE Email='$emailUtente'";
            $res= $pdo->query($sql);
            $row = $res->fetch();
            $tipoUtente= $row['TipoUtente'];
            $_SESSION['TipoUtente']=$tipoUtente;
        }catch(PDOException $e){echo $e->getMessage();}
        if($tipoUtente=="Amministratore"){
            header('Location: ../home/adminHome.php');
        }else if($tipoUtente == "Utilizzatore"){
            header('Location: ../home/myHome.php');
        }else if($tipoUtente == "Volontario"){
            header('Location: ../home/volHome.php');
        }else if($tipoUtente == "SuperUser"){
            header('Location: ../home/superUserHome.php');
        }
    }

}else
    echo "<script> alert('I dati non risultano corretti, sicuro di esserti registrato?'); window.location.href='login.php'; </script>";
}

?>

```

## Inserimento di una prenotazione di un libro cartaceo:

Controllo della disponibilità:

```

<label> Seleziona il genere del libro </label>
<div class="form-group input-group">
    <select name="Genere" id="Genere" class="form-control" required>
        <?php

            require '../../../../../connectionDB/connection.php';

            try{
                $sql = "SELECT Distinct(Genere) FROM Libro";
                $res = $pdo -> query($sql);
            }catch(PDOException $e){echo $e->getMessage();}

            while ($row = $res->fetch()) {
                echo '<option value=' . $row['Genere'] . '>' . $row['Genere'] . '</option>';
            }

        ?>
    </select>
</div>

<div class="form-group input-group">
    <label> Hai una biblioteca in particolare dove vorresti cercare il libro?</label>
    <select name="Biblioteca" id="Biblioteca" class="form-control" >
        <option value='none'> ----- </option>
        <?php

            try{
                $sql = "SELECT Nome FROM Biblioteca";
                $res = $pdo -> query($sql);
            }catch(PDOException $e){echo $e->getMessage();}

            while ($row = $res->fetch()) {
                echo '<option value=' . urlencode($row['Nome']) . '>' . $row['Nome'] . '</option>';
            }

        ?>
    </select>
</div>
```

Mostra scelta cartaceo:

```

try{
    if($_POST['Biblioteca'] != 'none'){

        $nomeEncode = $_POST['Biblioteca'];
        $nome = urldecode($nomeEncode);
        if($_POST['Isbn'] != ''){

            $isbn = $_POST['Isbn'];

            $sql = "SELECT Libro.CodiceISBN, Titolo, NomeBiblioteca, StatoDiConservazione
                    FROM Libro
                    join libridisponibili on (Libro.CodiceISBN = LibriDisponibili.CodiceISBN)
                    join cartaceo on (Libro.CodiceISBN = Cartaceo.CodiceISBN)
                    WHERE Libro.CodiceISBN = $isbn
                    AND Cartaceo.StatoPrestito = 'Disponibile'
                    AND Libro.Genere = '$genere'
                    AND Libro.Titolo = '$titolo'
                    AND NomeBiblioteca = '$nome';";

        }else{
            $sql = "SELECT Libro.CodiceISBN, Titolo, NomeBiblioteca, StatoDiConservazione
                    FROM Libro
                    join libridisponibili on (Libro.CodiceISBN = LibriDisponibili.CodiceISBN)
                    join cartaceo on (Libro.CodiceISBN = Cartaceo.CodiceISBN)
                    WHERE Cartaceo.StatoPrestito = 'Disponibile'
                    AND Libro.Genere = '$genere'
                    AND Libro.Titolo = '$titolo'
                    AND NomeBiblioteca = '$nome';";
        }
    }else {

        if($_POST['Isbn'] != ''){

            $isbn = $_POST['Isbn'];

            $sql = "SELECT Libro.CodiceISBN, Titolo, NomeBiblioteca, StatoDiConservazione
                    FROM Libro
                    join libridisponibili on (Libro.CodiceISBN = LibriDisponibili.CodiceISBN)
                    join cartaceo on (Libro.CodiceISBN = Cartaceo.CodiceISBN)
                    WHERE Libro.CodiceISBN = $isbn
                    AND Cartaceo.StatoPrestito = 'Disponibile'
                    AND Libro.Genere = '$genere'
                    AND Libro.Titolo = '$titolo';";

        }else{
            $sql = "SELECT Libro.CodiceISBN, Titolo, NomeBiblioteca, StatoDiConservazione
                    FROM Libro
                    join libridisponibili on (Libro.CodiceISBN = LibriDisponibili.CodiceISBN)
                    join cartaceo on (Libro.CodiceISBN = Cartaceo.CodiceISBN)
                    WHERE Cartaceo.StatoPrestito = 'Disponibile'
                    AND Libro.Genere = '$genere'
                    AND Libro.Titolo = '$titolo';";
        }
    }
}
$res = $pdo -> query($sql);

```

### Inserimento prenotazione:

```

try{
    $sql = $pdo->prepare("INSERT INTO PrenotazioneCartaceo VALUES(?, ?, ?, ?, ?, ?)");
    $sql->bindParam(1, $id, PDO::PARAM_INT);
    $sql->bindParam(2, $IsbnLibro, PDO::PARAM_INT);
    $sql->bindParam(3, $inizio, PDO::PARAM_STR);
    $sql->bindParam(4, $fine, PDO::PARAM_STR);
    $sql->bindParam(5, $email, PDO::PARAM_STR);
    $sql->bindParam(6, $NomeBiblioteca, PDO::PARAM_STR);

    $res = $sql->execute();

}catch(PDOException $e){echo $e->getMessage();}

if($res > 0)
    echo "<script> alert('Prenotazione effettuata correttamente!'); window.location.href='../../home/myHome.php'; </script>";
else
    echo "<script> alert('La prenotazione NON è stata effettuata, riprova!'); window.location.href='controllaDisponibilitaCartaceo.php'; </script>";

```

## Esempio di visualizzazione di una statistica

```
<?php

require '../../../../../connectionDB/connection.php';

try{

    $sql = "Select Titolo, NumeroAccessi
            From Libro, Ebook
            where Libro.CodiceISBN = Ebook.CodiceISBN
            group by Titolo, NumeroAccessi
            order by NumeroAccessi DESC";
    $res = $pdo -> query($sql);

}catch(PDOException $e){echo $e->getMessage();}

echo "
<table>
<thead>
<tr>
<th></th>
<th>Ebook</th>
<th>Numero Accessi</th>
</tr>
</thead>
<tbody>";

while ($row = $res->fetch()) {
    $titolo = $row['Titolo'];
    $accessi = $row['NumeroAccessi'];

    echo "<tr>";

    echo "<td>" .
        "</td>";

    echo "<td>" . $titolo . "</td>";
    echo "<td>" . $accessi . "</td>";
    echo "</tr>";

}
echo "</tbody></table>";
?>
```

## Inserimento messaggio

```
if(isset($_POST['messaggioButton'])){
    $emailAmministratore = $_SESSION['EmailUtente'];
    $emailUtilizzatore = $_POST['emailUtilizzatore'];
    $titolo = $_POST['titolo'];
    $messaggio = $_POST['messaggio'];
    $nota = $_POST['note'];
    $data = date("Y-m-d");

    $sql = $pdo->prepare("INSERT INTO Messaggio (EmailAmministratore, EmailUtilizzatore, DataMessaggio, Titolo, Testo) VALUES (?, ?, ?, ?, ?)");
    $sql->bindParam(1, $emailAmministratore, PDO::PARAM_STR);
    $sql->bindParam(2, $emailUtilizzatore, PDO::PARAM_STR);
    $sql->bindParam(3, $data, PDO::PARAM_STR);
    $sql->bindParam(4, $titolo, PDO::PARAM_STR);
    $sql->bindParam(5, $messaggio, PDO::PARAM_STR);
    $res = $sql->execute();

    if($res > 0){
        $bulk = new MongoDB\Driver\BulkWrite();
        $doc = ['_id' => new MongoDB\BSON\ObjectID(), 'titolo' => 'Messaggio', 'tipoUtente'=>$_SESSION['TipoUtente'], 'emailUtente'=>$_SESSION['EmailUtente'],
        'timestamp'=>date('Y-m-d H:i:s')];
        $bulk->insert($doc);
        $connessioneMongo -> executeBulkWrite('ebiblio.log',$bulk);
        echo "<script> alert('Messaggio inserito correttamente!'); window.location.href='../../home/home.php'; </script>";

    }else
        echo "<script> alert('Il messaggio non è stato inserito correttamente, riprova!'); window.location.href='inserimentoMessaggio.php'; </script>";
}
?>
```

## Log Mongo DB

```

if($sql->execute()) {
    $bulk = new MongoDB\Driver\BulkWrite();
    $doc = ['_id' => new MongoDB\BSON\ObjectId(), 'titolo' => 'PostoLettura', 'tipoUtente'=>$_SESSION['TipoUtente'], 'emailUtente'=>$_SESSION['EmailUtente'],
    'timeStamp'=>date('Y-m-d H:i:s')];
    $bulk -> insert($doc);
    $connessioneMongo -> executeBulkWrite('ebiblio.log',$bulk);
    echo "<script> alert('Posto Lettura inserito correttamente!'); window.location.href='../../home/adminHome.php'; </script>";
}
else
    echo "<script> alert('Il posto lettura non è stato inserito correttamente, riprova!'); window.location.href='inserimentoPostoLettura.php'; </script>";
?>

```

## Cancellazioni

```

try{
    $sql = "DELETE
            FROM libro
            WHERE CodiceISBN = '$isbn'";
    $res = $pdo -> query($sql);

    switch($tipo){
        case 'Cartaceo':
            $sql = "DELETE
                    FROM Cartaceo
                    WHERE CodiceISBN = '$isbn'";
            $res = $pdo -> query($sql);
            break;

        case 'Ebook':
            $sql = "DELETE
                    FROM Ebook
                    WHERE CodiceISBN = '$isbn'";
            $res = $pdo -> query($sql);
            break;

        case 'Entrambi':
            $sql = "DELETE
                    FROM Cartaceo
                    WHERE CodiceISBN = '$isbn'";
            $sql1 = "DELETE
                    FROM Ebook
                    WHERE CodiceISBN = '$isbn'";
            $res = $pdo -> query($sql);
            $res1 = $pdo -> query($sql1);
            break;
    }

}catch(PDOException $e){echo $e->getMessage();}

if($res != 0){
    $bulk = new MongoDB\Driver\BulkWrite();
    $doc = ['_id' => new MongoDB\BSON\ObjectId(), 'titolo' => 'EliminaLibro', 'tipoUtente'=>$_SESSION['TipoUtente'], 'emailUtente'=>$_SESSION['EmailUtente'],
    'timeStamp'=>date('Y-m-d H:i:s')];
    $bulk -> insert($doc);
    $connessioneMongo -> executeBulkWrite('ebiblio.log',$bulk);
    echo "<script> alert('Record eliminato!'); window.location.href='../../visualizzazione/visualizzazioneLibri.php'; </script>";
}
else
    echo "<script> alert('Il record NON è stato eliminato!'); window.location.href='../../visualizzazione/visualizzazioneLibri.php'; </script>";

```