Bookshelf Manager

Bilotta Antonio

December 19, 2025

Contents

1 Idea del progetto

Il concetto di *possedere* una vasta collezione di libri per la maggior parte non letti è un qualcosa di già esplorato da grandi autori quali **Umbero Eco**.
Una volta realizzato questo sistema, un utente più che possedere fisicamente dei titoli, avrà a disposizione un catalogo di volumi per tenere traccia di ciò che ha letto o meno. Una piattaforma del genere vuole essere d'aiuto per quei tipi di persone, come il sottoscritto, che hanno bisogno di un "percorso" chiaro e ben definito, per mantere e praticare con costanza un hobby quale può essere la lettura.

2 Considerazioni sulla realtà d'interesse

Ogni **Persona** sarà specializzata, con possibilità di overlapping, in **Utente**, o in **Admin**.
O gni **Utente** avrà delle informazioni anagrafiche legate al proprio profilo, come *nome, cognome, data di nascita*, oltre alla coppia *email-password* che consentirà a questi di effettuare il login.
Gli **Admin**, utenti speciali, avranno accesso a delle aree speciali per caricare nuovi libri che saranno poi fruibili agli utenti.

Ogni **Libro** avrà una sezione principale, che conserverà *codice univoco, titolo originale, data prima edizione, voto medio*, a cui le varie **Edizioni**, contenenti *titolo tradotto, lingua, immagine di copertina, numero pagine, data di pubblicazione* e una breve *didascalia*, faranno riferimento, in modo che agli utenti un libro risulti letto in tutte le sue edizioni, nonostante abbiano usufruito solo di una.
Ogni **Edizione** avrà inoltre una **Prefazione**, composta da *titolo e data* scritta da un **Autore**.

Il sistema conserverà anche informazioni su **Autori**, che siano di libri o di prefazioni di questi, tra queste informazioni sulla vita quali una piccola biografia, data di nascita e (in caso) di morte. Ogni libro sarà *classificato* in **categorie** diverse (conservate con multivalore), queste rappresentano generi e temi analizzati.

Gli utenti, possono, dopo aver letto un libro, valutare la lettura, infatti effettueranno (non obbligatoriamente) delle **Recensioni**, divise in voto numerico e commento, i libri così, potranno avere una valutazione generale data dalla media dei voti attribuiti dai lettori.

Gli utenti avranno un numero totale di libri con il quale hanno interagito, poichè questi saranno divisi in due liste: libri letti, e da leggere in futuro. Per ogni utente sarà consevarto (opzionalmente) il libro che stanno attualmente leggendo. Vogliamo memorizzare anche gli **Editori**, con *codice, nome, nazione, sede* che stampano i vari **libri**.

3 Specifiche Realtà d'interesse

Per semplificare l'implementazione possiamo, successivamente alla Progettazione Concettuale, discutere alcuni casi.

Gli **Admin**, a livello implementativo, possono essere dei campi *booleani* all'interno dell'entità **Utente**, vero se all'utente sono concessi dei permessi speciali, falso altrimenti.

Inoltre le immagini dei libri voglio essere idealmente compresse e memorizzate fisicamente su un server con il quale l'applicazione comunicherà, memorizzeremo quindi il *percorso relativo* di queste all'interno del server che le memorizza, in particolare il namespacing sarà il seguente: $[codiceUnivicoLibro]img.jpeg$.

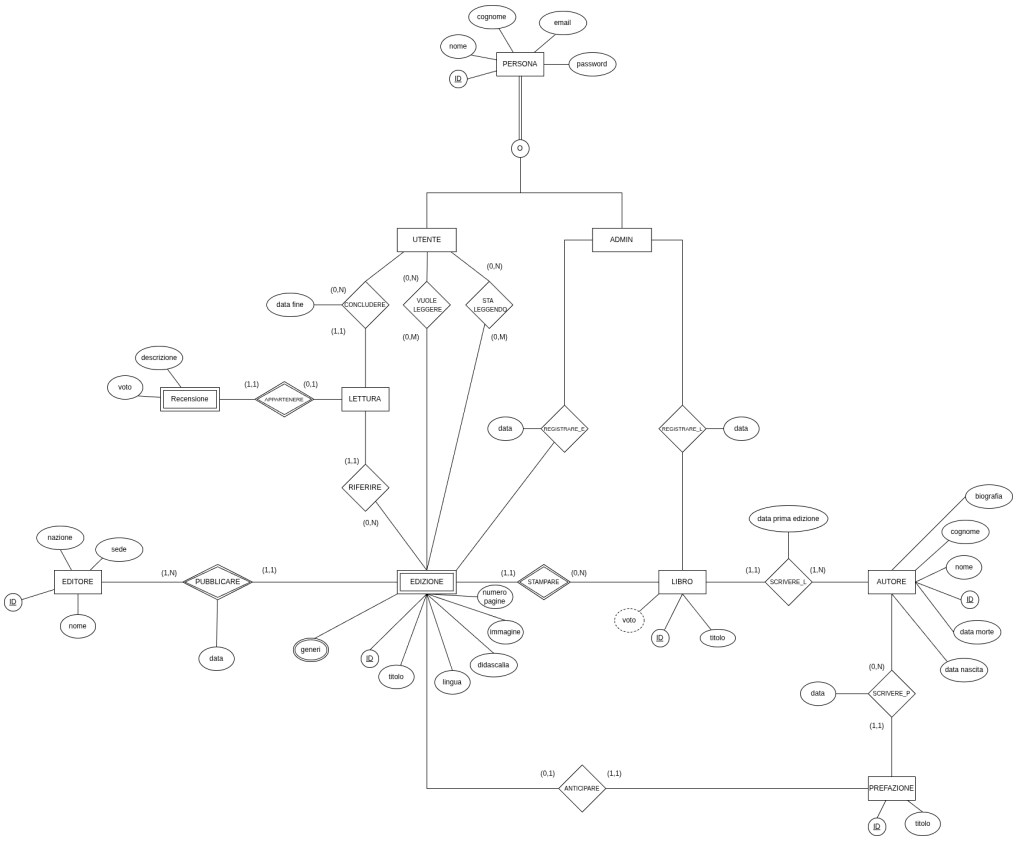
Le interazioni di ogni utente con i libri possono essere ridotte a un'interazione generica che conserva un flag di stato, 0 se letto, 1 se leggendo, 2 se finito. questa conserva anche (in maniera opzionale) un riferimento ad una **Recensione** in caso il libro sia stato letto.

4 Glossario dei termini

Termine	**Significato**
Persona	Insieme di dati anagrafici che qualificano un individo, che si specializza poi in utente
Admin	Super utente che può aggiungere nuovi libri alla piattaforma
Utente	Account di una persona che interagisce con libri e altri utenti
Autore	Scrittore di libri o di Prefazioni di questi
Prefazione	Scritto, lungo o meno, che funge da premessa di un libro.
Libro	Titolo originale, non fisico
Edizione	Libro specifico, esiste fisicamente, poichè stampato da qualcuno
Recensione	Pensiero e valutazione lasciato dagli utenti nei confronti di una lettura
Editore	Casa editrice che si occupa di redigere e mandare in stampa libri

5 Progettazione Concettuale

Procedendo con la progettazione concettuale della base di dati, si ottiene il seguente **schema EER**:



5.1 Dizionario delle Entità

Legenda: Sotto-entità, attributo multivalore, attributo ridondante, entità debole, chiave candidata

Termine	Significato	Attributi	Chiave
Persona	Individuo che interagisce con la piattaforma	• ID • Nome • Cognome • Email • Password	ID
Utente	Account di una persona per interagire con libri e altri utenti	/	/
Admin	Utente con permessi speciali per caricare libri	/	/
Libro	Titolo originale, concettuale	• ID • titolo • voto	ID
Edizione	Versione fisica di un libro	• titolo • lingua • didascalia • copertina • numero pagine • generi	/

Autore	Scrittore di libri o prefazioni	• ID • nome • cognome • biografia • data nascita • data morte	ID
Prefazione	Premessa di un libro	• ID • titolo	ID
Lettura	Completamento di una edizione da parte di un utente	/	/
Recensione	Valutazione di un libro	• voto • descrizione	/
Editore	Casa editrice	• ID • nome • nazione • sede	ID

5.2 Dizionario delle Relazioni

Relazione	**Descrizione**	**Entità coinvolte**	**Attributi**
Sta leggendo	Un utente è impegnato nella lettura di uno o più libri	Utente(0,N) Edizione(0,M)	/
Vuole leggere	Un utente salva una lista di libri da leggere	Utente(0,N) Edizione(0,M)	/
Concludere	Un Utente ha concluso delle letture	Utente(0,N) Lettura(1,1)	data fine

Appartenere	Una recensione appartiene a una lettura	Recensione(1,1) Lettura(0,1)	/
Riferire	Una lettura si riferisce ad un'edizione	Lettura(1,1) Edizione(0,N)	/
Pubblicare	Un editore pubblica un'edizione di un libro	Editore(1,N) Edizione(1,1)	data
RegistrareE	Admin registra un'edizione	Edizione(1,1) Admin(0,N)	data
RegistrareL	Admin registra un libro	Libro(1,1) Admin(0,N)	data
Stampare	Un libro è stampato in diverse edizioni	Edizione(1,1) Libro(0,N)	/
ScrivereL	Un autore scrive un libro	Autore(1,N) Libro(1,1)	data prima edizione
ScrivereP	Un autore scrive una prefazione	Autore(0,N) Prefazione(1,1)	data
Anticipare	Una prefazione anticipa un'edizione di un libro	Prefazione(1,1) Edizione(0,1)	data

5.3 Vincoli non esprimibili

- L'attributo **voto** di un libro e delle recensioni deve essere un numero $n \pm 0.5$, con $0.5 \leq n \leq 9.5$
- L'attributo data di morte di un autore è opzionale.
- L'attributo email di un utente è univoco.

6 Tavole

6.1 Tavole dei volumi

È riportata di seguito la tavola dei volumi della base di dati

Concetto	**Tipo**	**Carico Applicativo**
Persona	E	100
Utente	E	95
Admin	E	5

Concludere	R	300
Vuole leggere	R	400
Sta Leggendo	R	50
Lettura	E	300
Appartenere	R	150
Recensione	E	150
Riferire	R	250
Edizione	E	300
RegistrareE	R	300
RegistrareL	R	100
Pubblicare	R	100
Editore	E	5
Stampare	R	300
Libro	E	100
ScrivereL	R	100
ScrivereP	R	30
Prefazione	E	30
Autore	E	20
Anticipare	R	30

6.2 Tavole delle operazioni

Operazione	**Tipo**	**Frequenza**
Registrazione nuovo utente	I	10/gg
Login utente	I	40/gg
Aggiunta libro da parte di un admin (RegistrareL)	I	2/gg
Aggiunta edizione da parte di un admin (RegistrareE)	I	5/gg
Inserimento di una lettura completata (Concludere)	I	15/gg
Inserimento di una recensione associata a una lettura (Appartenere)	I	10/gg
Aggiunta libro alla lista “Vuole leggere”	I	20/gg
Aggiornamento stato lettura	I	10/gg
Generazione del voto medio di un libro	B	10/gg
Import massivo nuovi editori	B	1/settimana

7 Ristrutturazione

7.1 Analisi delle ridondanze

Il dato ridondante è l'attributo "voto" dell'entità Libro. Infatti sarebbe possibile ottenere la media dei voti attraversando le letture che fanno riferimento a tale libro, per accedere ai voti delle recensioni, effettuando, poi, una media aritmetica.
Supponendo che l'attributo pesi **4 byte**, essendo un normale intero, e considerando il carico applicativo dell'entità libro pari a 100 allora occupiamo circa **400 byte**.
Calcoliamo, con le tavole degli accessi se conviene mantenere o meno tale attributo.

7.2 Tavole degli accessi

Dalla tavola delle operazioni, risultano coinvolte nell'utilizzo del voto medio solo le seguenti operazioni:

Operazione 6: Inserimento di una recensione (frequenza: 10/gg)

Ridondanza

Entità/Relazioni	Tipo	Accessi	Tipo di accesso
Recensione	E	1	S
Appartenere	R	1	S
Lettura	E	1	S
Riferire	R	1	S
Edizione	E	3	S
Stampare	R	3	S
Libro	E	1	L
Libro	E	1	S

$Totale = [2TOT_S + TOT_L] \cdot FREQUENZA = [2 \cdot 11 + 1] \cdot 10/gg = 230$ accessi/gg

No Ridondanza

Entità/Relazioni	Tipo	Accessi	Tipo di accesso
Recensione	E	1	S
Appartenere	R	1	S

$Totale = [2TOT_S + TOT_L] \cdot FREQUENZA = [2 \cdot 2 + 0] \cdot 10/gg = 40$ accessi/gg

---

Operazione 9: Generazione del voto medio di un libro (operazione batch, 10/gg)

Ridondanza

Entità/Relazioni	Tipo	Accessi	Tipo di accesso
Libro	E	1	L

$Totale = [2TOT_S + TOT_L] \cdot FREQUENZA = [0 + 1] \cdot 10/gg = 10$ accessi/gg

No Ridondanza

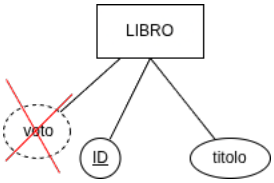
Entità/Relazioni	Tipo	Accessi	Tipo di accesso
Libro	E	1	L
Stampare	R	3	L
Edizione	E	3	L
Riferire	R	1	L
Lettura	E	1	L
Appartenere	R	1	L
Recensione	E	1	L

$Totale = [2TOT_S + TOT_L] \cdot FREQUENZA = [0 + 11] \cdot 10/gg = 110$ accessi/gg

Conclusione

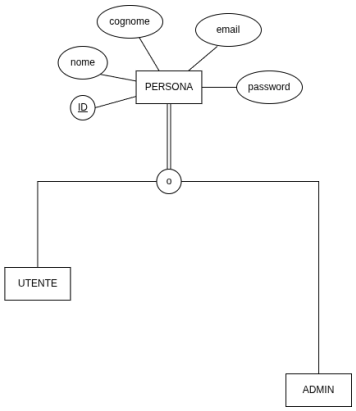
Totale accessi con ridondanza = $(230 + 10)\ a/gg + 400byte$
Totale accessi senza ridondanza = $(110 + 40)\ a/gg$
Conviene eliminare il dato ridondante.

7.3 Eliminazione Ridondanza

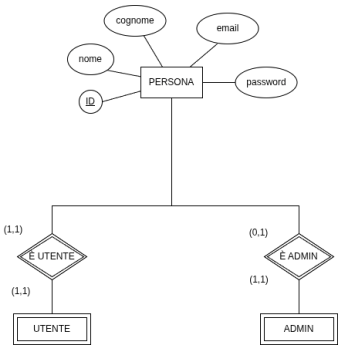


7.4 Eliminazione Gerarchie

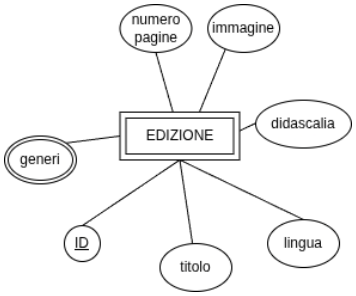
Lo schema inizialmente presenta la seguente specializzazione dell'entità **Persona**.



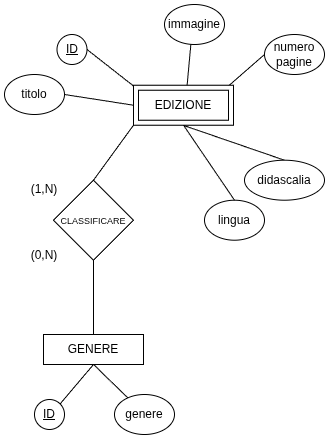
Procediamo sostituendo la gerarchia con due nuove relazioni che rendono le entità figlie due nuove entità deboli.



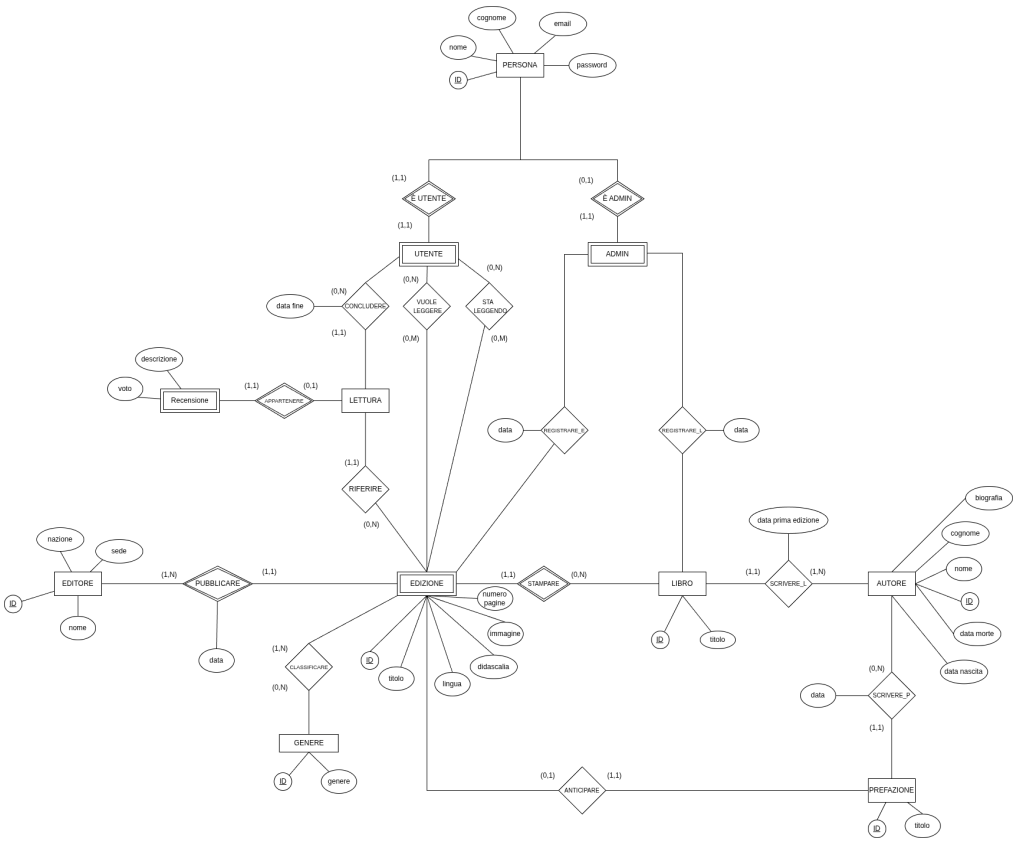
7.5 Eliminazione Attributi Multivalore



Sostituiamo all'attiributo multivalore **generi** l'entità **Genere** che sarà in relazione con un'**Edizione** con molteplicità $N : 1$



7.6 Schema EER ristrutturato



8 Progettazione Logica

8.1 Schema relazionale

Lo schema EER ristrutturato viene mappato nel seguente schema logico, implementiamo anche ciò scritto nelle **Considerazioni sulla realta d'interesse**. Rinominiamo le chiavi primarie del tipo ID in modo che siano nel formato [nomeEntità]ID, per una maggiore chiarezza.

UTENTE(utenteID, nome, cognome, email, password, isAdmin)

Libro(libroID, titolo, voto, dataPrimaEdizione, autoreID↑)

Genere(genereID, genere)

Editore(editoreID, nome, sede, nazione)

Edizione(edizioneID, editoreID↑, libroID↑, titolo, lingua, didascalia, numeroPagine)

Autore(autoreID, nome, cognome, biografia, dataNascita, dataMorte)

Prefazione(prefazioneID, titoloEdizione, autoreID↑)

CLASSIFICARE(EdizioneID↑, genereID↑)

LETTURA(utenteID↑, EdizioneID↑, status, dataFine, voto, mezzo, descrizione)

Abbiamo accorpato l'entià debole **Recensione** nella relazione **LETTURA** poichè le chiavi di queste due coincidono, inoltre con il flag status indichiamo se il libro è letto oppure se si sta leggendo, o si pianifica di leggerlo, tenendo, però a mente, che queste ultime due opzioni implicano, che *LETTURA.dataFine, LETTURA.voto, Lettura.mezzo, LETTURA.descrizione* siano NULL. Notiamo che il campo voto sarà un intero da 0 a 10, se assume valore pari a 10, allora il campo booleano mezzo, sarà 0 poichè accettiamo mezzi voti (1.5, 3.5, 9.5, etc) solo se non eccedono il massimo, quindi 10

8.2 Normalizzazione

Lo schema relazionale rispetta le condizioni della **1NF**, poichè gli attributi di qualsiasi relazione sono atomici. Sono rispettate anche le condizioni della **2NF**, poichè ogni attiributo di qualsiasi relazione non dipende parzialmente dalla chiave.
Lo schema è in **3NF** poichè non esistono dipendenze in cui attributi non primi (non parte di chiave) dipendono da attributi non chiave.
L'unica violazione della **BCNF** riguarda il campo *Utente.email*, in quanto questo è univoco, quindi determina gli altri, però non è superchiave. Decidiamo di **mantenere questa violazione**, per avere una maggiore coesione tra gli **ID** e per far sì che siano tutti numerici.

Di conseguenza, non ci sono ridondanze né anomalie di aggiornamento.

9 Progettazione Fisica

9.1 Creazione delle tabelle

```
-- ========================================
-- TABELLA UTENTE
-- ========================================
CREATE TABLE Utente (
    utenteID INT AUTO_INCREMENT PRIMARY KEY,
    nome VARCHAR(50) NOT NULL,
    cognome VARCHAR(50) NOT NULL,
    email VARCHAR(100) NOT NULL UNIQUE,
    password VARCHAR(255) NOT NULL,
    isAdmin BOOLEAN NOT NULL DEFAULT FALSE
);


-- ========================================
-- TABELLA AUTORE
-- ========================================
CREATE TABLE Autore (
    autoreID INT AUTO_INCREMENT PRIMARY KEY,
    nome VARCHAR(50) NOT NULL,
    cognome VARCHAR(50) NOT NULL,
    biografia TEXT,
    dataNascita DATE,
    dataMorte DATE
```

```
);


-- ============================================
-- TABELLA LIBRO
-- (SENZA VOTO      scelta: eliminazione ridondanza)
-- ============================================
CREATE TABLE Libro (
    libroID INT AUTO_INCREMENT PRIMARY KEY,
    titolo VARCHAR(200) NOT NULL,
    dataPrimaEdizione DATE,
    autoreID INT NOT NULL,
    FOREIGN KEY (autoreID) REFERENCES Autore(autoreID)
);


-- ============================================
-- TABELLA GENERE
-- ============================================
CREATE TABLE Genere (
    genereID INT AUTO_INCREMENT PRIMARY KEY,
    genere VARCHAR(50) NOT NULL UNIQUE
);


-- ============================================
-- TABELLA EDITORE
-- ============================================
CREATE TABLE Editore (
    editoreID INT AUTO_INCREMENT PRIMARY KEY,
    nome VARCHAR(100) NOT NULL,
    sede VARCHAR(100),
    nazione VARCHAR(50)
);


-- ============================================
-- TABELLA EDIZIONE
-- ============================================
CREATE TABLE Edizione (
    edizioneID INT AUTO_INCREMENT PRIMARY KEY,
    editoreID INT NOT NULL,
    libroID INT NOT NULL,
    titolo VARCHAR(200) NOT NULL,
    lingua VARCHAR(50),
    didascalia TEXT,
    numeroPagine INT,
    FOREIGN KEY (editoreID) REFERENCES Editore(editoreID),
    FOREIGN KEY (libroID) REFERENCES Libro(libroID)
);


-- ============================================
-- TABELLA PREFAZIONE
-- ============================================
CREATE TABLE Prefazione (
    prefazioneID INT AUTO_INCREMENT PRIMARY KEY,
```

```
    titolo VARCHAR(200) NOT NULL,
    autoreID INT NOT NULL,
    edizioneID INT NOT NULL,
    FOREIGN KEY (autoreID) REFERENCES Autore(autoreID)
    FOREIGN KEY (autoreID) REFERENCES Edizione(edizioneID)
);


-- ============================================
-- RELAZIONE CLASSIFICARE (Edizione      Genere)
-- ============================================
CREATE TABLE CLASSIFICARE (
    edizioneID INT NOT NULL,
    genereID INT NOT NULL,
    PRIMARY KEY (edizioneID, genereID),
    FOREIGN KEY (edizioneID) REFERENCES Edizione(edizioneID),
    FOREIGN KEY (genereID) REFERENCES Genere(genereID)
);


-- ============================================
-- TABELLA LETTURA
-- (integrata con i dati della recensione)
-- ============================================
CREATE TABLE LETTURA (
    utenteID INT NOT NULL,
    edizioneID INT NOT NULL,

    -- status = 'da_leggere' | 'in_lettura' | 'finito'
    status ENUM('da_leggere', 'in_lettura', 'finito') NOT NULL,

    dataFine DATE DEFAULT NULL,

    -- voto = numero intero 0  10
    voto INT CHECK (voto BETWEEN 0 AND 10),

    -- mezzo = 0 se voto intero, 1 se mezzo voto
    mezzo BOOLEAN,

    descrizione TEXT,

    PRIMARY KEY (utenteID, edizioneID),
    FOREIGN KEY (utenteID) REFERENCES Utente(utenteID),
    FOREIGN KEY (edizioneID) REFERENCES Edizione(edizioneID),

    -- vincolo logico: se finito, dataFine non pu  essere NULL
    CHECK (
        (status <> 'finito') OR
        (status = 'finito' AND dataFine IS NOT NULL)
    ),

    -- se voto = 10, mezzo deve essere 0
    CHECK (
        mezzo IN (0,1) AND
        (voto IS NULL OR (voto < 10 OR mezzo = 0))
    )
);
```

9.2 Definizione delle operazioni

```
-- 1. Registrazione nuovo utente
INSERT INTO UTENTE (utenteID, nome, cognome, email, password, isAdmin)
VALUES (?, ?, ?, ?, ?, FALSE);

-- 2. Login utente
SELECT utenteID, nome, cognome, isAdmin
FROM UTENTE
WHERE email = ? AND password = ?;

-- 3. Aggiunta libro (RegistrareL)
INSERT INTO Libro (libroID, titolo, dataPrimaEdizione, autoreID)
VALUES (?, ?, ?, ?);

-- 4. Aggiunta edizione (RegistrareE)
INSERT INTO Edizione (edizioneID, editoreID, libroID, titolo, lingua, didascalia, numeroPagine
VALUES (?, ?, ?, ?, ?, ?, ?);

-- 5. Inserimento lettura completata (Concludere)
INSERT INTO LETTURA (utenteID, edizioneID, status, dataFine)
VALUES (?, ?, 'finito', CURRENT_DATE);

-- 6. Inserimento recensione collegata alla lettura (Appartenere)
UPDATE LETTURA
SET voto = ?, mezzo = ?, descrizione = ?
WHERE utenteID = ?
  AND edizioneID = ?
  AND status = 'finito';

-- 7. Aggiunta alla lista "Vuole leggere"
INSERT INTO LETTURA (utenteID, edizioneID, status)
VALUES (?, ?, 'da_leggere');

-- 8. Aggiornamento stato lettura
UPDATE LETTURA
SET status = ?
WHERE utenteID = ?
  AND edizioneID = ?;

-- 8b. Aggiornamento per passare a "finito"
UPDATE LETTURA
SET status = 'finito',
    dataFine = CURRENT_DATE
WHERE utenteID = ?
  AND edizioneID = ?;

-- 9. Generazione voto medio di un libro
SELECT AVG(voto + mezzo * 0.5) AS votoMedio
FROM LETTURA L JOIN Edizione E ON L.edizioneID = E.edizioneID
WHERE L.voto IS NOT NULL AND E.libroID = ?;

-- 10. Import massivo editori
INSERT INTO Editore (editoreID, nome, sede, nazione)
SELECT editoreID, nome, sede, nazione
FROM EditoriTemp;
```

10 Applicazione del Database