RELAZIONE PROGETTO BASI DI DATI,
CREAZIONE DI UN DATABASE IN SQL PER LA GESTIONE DEL SISTEMA BIBLIOTECARIO DELL'UNIVERSITÀ DI FERRARA

A.A. 2019/2020, a cura di Alfonso Esposito e Francesco Penna

1. Definizione del Problema

Nei tempi odierni risulta necessaria la creazione di uno strumento informatico in grado di gestire le informazioni generate da parte di un sistema bibliotecario, come ad esempio quelle relative ai libri, gli utenti iscritti e i prestiti. L'università degli studi di Ferrara ha richiesto la creazione di questo sistema per la gestione della biblioteca di ateneo.
In particolare, l'Università di Ferrara è divisa in dipartimenti, ognuno con una sede fisica diversa, quindi a monte del problema va presa in considerazione la presenza di diversi sedi in cui possono essere presenti i libri. Di questi ultimi è necessario gestire alcune informazioni a loro relative, ovvero il titolo, l'ISBN, l'editore, l'anno di pubblicazione e la lingua in cui è stato scritto, ricordando che, a prescindere dalla lingua, il titolo del libro è registrato in inglese per facilitarne la ricerca. Per quanto riguarda le informazioni sull'editore, queste vengono memorizzate a loro volta registrandone nome, codice, indirizzo e numero di telefono. Si considera poi di dover registrare gli autori che hanno contribuito alla scrittura del libro, e anch'essi vengono memorizzati tenendo traccia di nome, data di nascita e luogo di nascita.
I libri all'interno della biblioteca di ateneo sono concessi in prestito solamente agli studenti registrati all'Università di Ferrara, quindi per poterne tenere traccia vengono registrati come dati il nome, il cognome, numero di telefono, domicilio e il numero di matricola. La presenza dei libri e degli utenti porta quindi alla generazione di diversi prestiti da gestire in tutte le sedi delle biblioteche dell'Università, in particolare per questi ultimi si vuole tenere traccia delle seguenti informazioni: data di uscita del libro dalla biblioteca, quale libro è stato preso in prestito e da quale utente, la succursale in cui è avvenuto il prestito e infine la data di restituzione del libro. Per quanto riguarda quest'ultimo dato si considera come tempo di restituzione standard trenta giorni, ma, vista l'eventuale presenza di libri in sola consultazione, o il caso di richiesta di una proroga del prestito, si tiene in considerazione che questa data è modificabile manualmente. Le succursali della biblioteca sono divise per dipartimenti dell'Università, e tenendo conto di questo particolare si vuole tenere traccia dell'indirizzo della sede e del numero di telefono, per fare in modo che queste siano facilmente raggiungibili o contattabili dagli utenti o dagli stessi operatori. A questo proposito si specifica che lo strumento informatico è concepito per questi ultimi, e conterrà quindi informazioni non accessibili al pubblico, non è stato richiesto quindi di implementare una schermata di login per la verifica delle credenziali in quanto l'indirizzo di accesso è raggiungibile solo dal personale autorizzato.
Infine, per quanto riguarda la realizzazione, lo strumento informatico creato consiste in un sito web creato con l'utilizzo del linguaggio HTML, e ampliato da CSS. I dati sopra descritti vengono registrati in una Base di Dati gestita attraverso il linguaggio SQL che comunica con il sito web grazie a degli script in linguaggio php.

2. Modello Entità Relazione

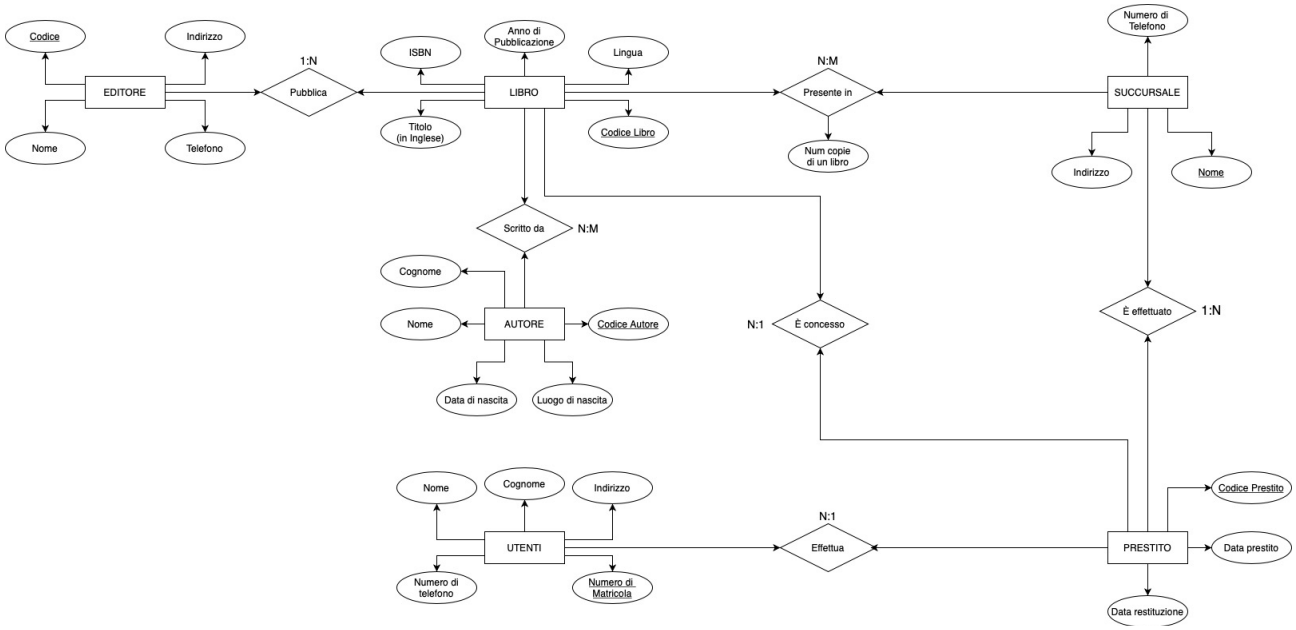
2.1 Creazione di Entità, Attributi e Relazioni

La creazione di una base di dati di tipo relazionale prevede come primo passo la creazione di un modello Entità Relazione, che viene poi presentato come un diagramma ER.
Si è scelto di partire dalla creazione di un'entità LIBRO, che viene descritta dagli attributi ISBN, Anno di Pubblicazione, Lingua, Titolo. Per la scelta dell'attributo chiave, identificativo del libro, la scelta non è ricaduta sul codice ISBN del libro, in quanto è stata considerata la possibilità che un libro scritto in diverse lingue possieda un codice ISBN diverso per ogni lingua in cui è stato scritto. Si è quindi provveduto a creare un attributo chiave Codice Libro, per poter rappresentare in modo univoco ogni libro.
Considerando poi che il libro viene scritto da qualcuno viene creata una seconda Entità, AUTORE, in cui vengono registrati gli attributi Nome, Cognome, Data e Luogo di nascita, e, vista l'assenza di un attributo chiave, che non possono essere Nome e Cognome considerando il caso di un'omonimia tra autori, viene aggiunto come attributo chiave un Codice Autore, per procedere ad una corretta identificazione.
Come accennato sopra si viene logicamente a creare una relazione nel modello tra le entità LIBRO e AUTORE chiamata *Scritto Da*.
Si considera poi l'editore che ha pubblicato il libro, creando così un'entità EDITORE, i cui attributi sono il Codice Editore, che è l'attributo chiave, il nome dell'Editore, il numero di telefono e l'indirizzo. Tra EDITORE e LIBRO si instaura una relazione chiamata *Pubblica*.
È stato mostrato nella definizione del problema che si considera il fatto che un libro sia presente in più succursali della biblioteca dell'università, considerando questo viene creata l'entità SUCCURSALE, definita dagli attributi Nome, che è l'attributo chiave in quanto si considera che non ci siano casi di omonimia tra i nomi delle succursali dell'Università, Indirizzo e Numero di Telefono, quest'ultimo registrato col fine di permettere agli operatori di comunicare più velocemente con la succursale desiderata.
SUCCURSALE e LIBRO vengono connesse attraverso la relazione *Presente In*, che che possiede un attributo, ovvero il numero di copie di un libro presenti in una data succursale, questo perché come attributo non descrive né LIBRO né SUCCURSALE.
Nella gestione della biblioteca uno dei dati fondamentali da gestire è quello riguardante i libri concessi in prestito a degli utenti, e considerando questo rapporto vengono create due entità: PRESTITO ed UTENTE. La prima è caratterizzata dalla data di inizio prestito e dalla data di restituzione del libro, e siccome nessuno di questi due può essere considerato un attributo univoco dell'entità, considerando ad esempio che due prestiti possono essere effettuati nello stesso giorno e lo stesso può accadere per la restituzione dei libri, come attributo chiave viene creato un Codice

Prestito. UTENTE invece rappresenta solo studenti iscritti all'Università degli studi di Ferrara, e visto questo fattore viene scelto come attributo chiave il Numero di Matricola, in quanto questo è univoco per ogni studente iscritto, inoltre vengono scelti come attributi Nome, Cognome, Indirizzo di Domicilio e Numero di Telefono. PRESTITO in quanto entità va a costruire tre relazioni: *Effettua, È concesso* ed *È effettuato*. La prima è una relazione instaurata tra UTENTE e PRESTITO, considerando che un utente usufruisce dei servizi della biblioteca effettuando un prestito, la seconda invece tra LIBRO e PRESTITO, in quanto si suppone che quando un utente effettua un prestito venga concesso un libro. Infine, *È effettuato* è la relazione che lega PRESTITO con SUCCURSALE poiché come è stato spiegato, un prestito è effettuato fisicamente all'interno di una delle succursali della biblioteca di Ferrara.

2.2 Studio delle cardinalità delle relazioni

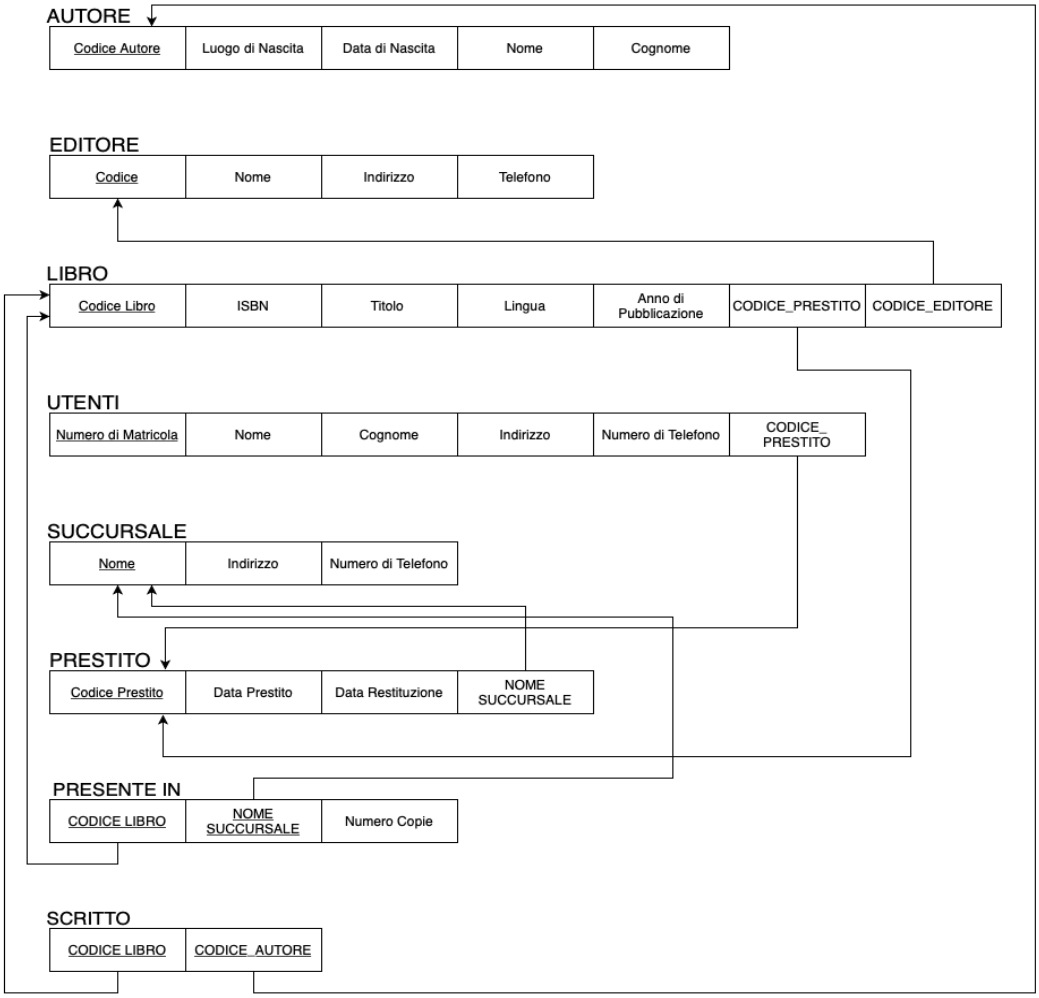
Tutte quelle viste finora nel modello ER sono relazioni di tipo binario poiché per ognuna vi partecipano due Entità, e per le relazioni di tipo binario si considerano le cardinalità, che corrispondono al numero massimo di istanze di relazione a cui può partecipare un'entità. Si procede quindi allo studio delle cardinalità delle relazioni evidenziate in precedenza.
Partendo da *Scritto da,* questa è una relazione di cardinalità N:M, in quanto si evince che un libro può essere scritto da più autori e allo stesso tempo un autore può scrivere diversi libri. La relazione *Pubblica* invece lega EDITORE a LIBRO, e in questo caso si considera che un editore pubblica diversi libri, mentre, considerate le condizioni di partenza, nel nostro problema si evince che un libro è pubblicato da un solo editore, quindi avremo una relazione di tipo N:1. Infine, l'entità LIBRO si lega a SUCCURSALE con la relazione *Presente In*, questa è una di tipo N:M in quanto di un libro è presente potenzialmente in diverse succursali, e in una di queste ultime sono conservati diversi libri.
Passando poi alle relazioni relative all'entità PRESTITO si evidenzia come la relazione *Effettua* sia di cardinalità N:1, in quanto un utente ha la possibilità di effettuare più di un prestito ma questo quando viene registrato è univoco per l'utente che lo effettua. Un discorso simile è applicabile alla relazione *È concesso*, dove un libro può essere concesso in prestito diverse volte ma il prestito generato per un dato libro è unico ed univoco, quindi avrà una cardinalità N:1. Rimane la relazione *È effettuato* che unisce SUCCURSALE con PRESTITO, secondo cui in una succursale vengono registrati diversi prestiti, ma il prestito è univoco per la succursale in cui viene registrato, andando a creare una relazione di tipo 1:N.
Andando quindi a presentare ciò che è stato esposto finora si ottiene il seguente diagramma ER.



3. Modello Relazionale

3.1 Passaggio dallo schema ER allo schema Relazionale

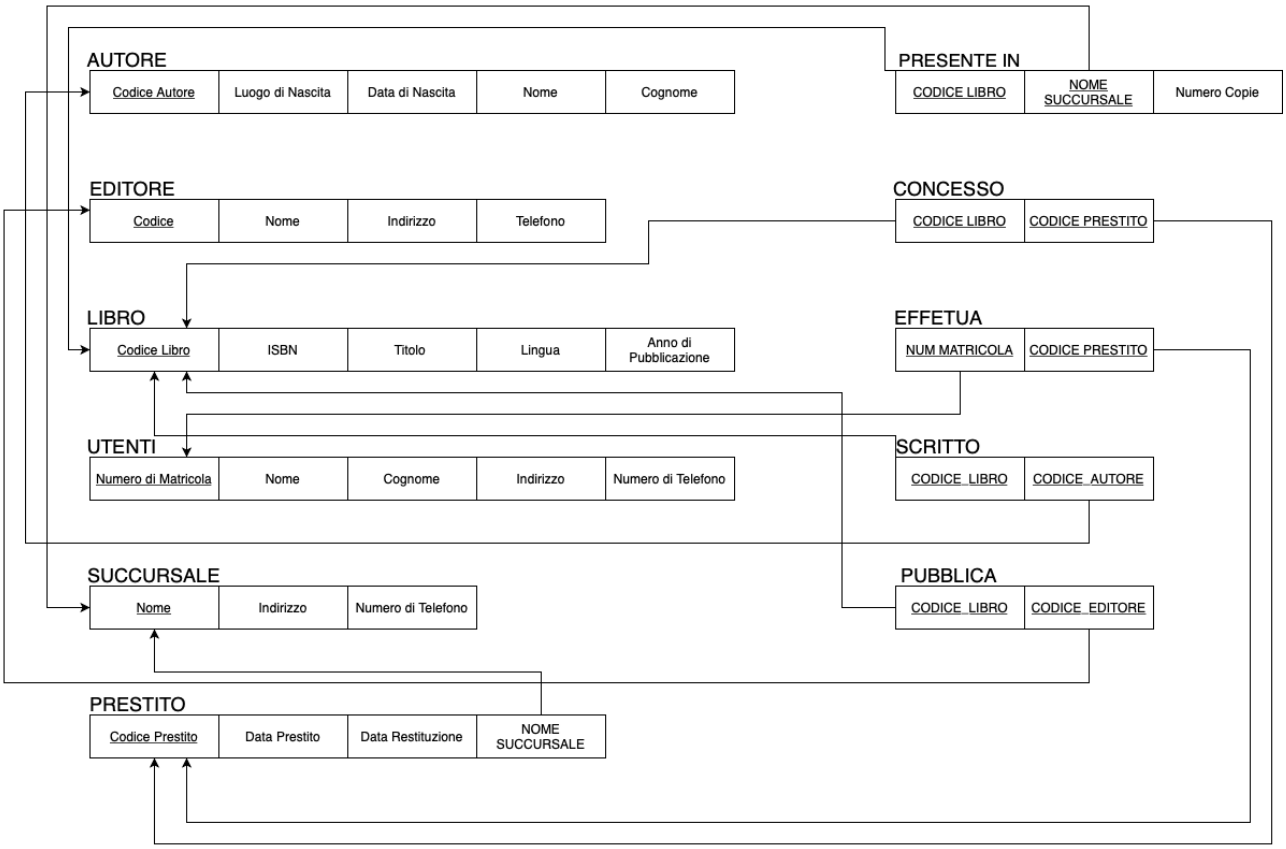
Dopo aver espresso la base di dati attraverso il diagramma Entità Relazione il passo successivo consiste nel trasformarlo in uno schema Relazionale, ovvero passare da un modello basato su Entità che comunicano attraverso Relazioni ad un una collezione di Relazioni più simile a tabelle di valori. Per poter arrivare a questo nuovo schema ci si avvalora di un algoritmo di mappatura ER-Relazionale.
Il primo passaggio consiste quindi nel creare per ogni Entità presente all'interno del diagramma ER un'Entità nello schema Relazionale. Saranno quindi presenti le entità: Autore, Editore, Libro, Utenti, Succursale, Prestito. Si procede poi a inserire all'interno di ogni Relazione i suoi attributi già descritti nel modello ER, in aggiunta l'attributo chiave diventa la chiave primaria della Relazione, ovvero la chiave che identifica una tupla (ovvero la riga della tabella).
Per quanto riguarda le Relazioni del diagramma ER queste possono essere sostituite in due modi: per le relazioni 1:N, o N:1, come il caso di *Pubblica*, *È concesso*, *È effettuato* ed *Effettua*, si inserisce come chiave esterna la chiave primaria della relazione di lato uno. Infine, per le relazioni N:M viste, come *Scritto da* e *Presente In* si vanno a creare due ulteriori Relazioni, con, come attributi, le chiavi primarie di entrambe le relazioni. In aggiunta a questi due attributi la relazione *Presente In* possiede come attributo il Numero di Copie di un libro.
Ad un primo passaggio della trasformazione da schema ER a schema Relazionale si ottiene quindi il seguente diagramma.



3.2 Verifica dello Schema Relazionale e riduzione alla terza forma normale

Ad una prima visione di questo schema ci si rende conto del problema, ricorrente in alcune relazioni, della ridondanza di dati nelle tuple: in particolare si può riscontrare nella relazione LIBRO e in UTENTI. Partendo da LIBRO questo fenomeno è osservabile in particolare nelle due chiavi esterne CODICE_PRESTITO e CODICE_EDITORE, poiché, per il modo in cui questo schema è organizzato, bisognerebbe registrare, ogni volta che viene preso in prestito un libro, tutti i suoi dati e l'editore che pubblica il libro, e questo porterebbe ad una cattiva gestione della memoria. Per motivi quindi di correttezza si è provveduto a spezzare la relazione LIBRO in tre relazioni: LIBRO stesso, che rimane senza le due chiavi esterne, CONCESSO, relazione formata da CODICE_LIBRO e CODICE_PRESTITO, che permette di rintracciare i dati del prestito di un determinato libro all'interno della base di dati, e PUBBLICA, relazione formata da CODICE_LIBRO e CODICE_EDITORE per tenere traccia delle informazioni riguardante l'editore che ha curato la pubblicazione del libro.

Lo stesso modus operandi viene applicato alla relazione UTENTI, in quanto la presenza di CODICE_PRESTITO portava ad una ridondanza di informazioni, considerando che per ogni prestito effettuato da un dato utente si sarebbero dovute registrare tutte le informazioni dello stesso. Quindi si va a creare una relazione di nome EFFETTUA, che contiene il NUM_MATRICOLA (numero di matricola) dello studente che effettua il prestito e CODICE_PRESTITO per poter risalire alle informazioni riguardanti lo stesso.
Concluse queste modifiche si arriva allo Schema Relazionale utilizzato per la realizzazione della base di dati.



Si procede poi alla verifica che lo schema sia in terza forma normale. Lo schema rispetta la prima forma normale in quanto si può verificare che in primo luogo non sono presenti attributi multivalore, inoltre non sono presenti relazioni nidificate. La seconda forma normale viene a sua volta rispettata poiché, nel caso della relazione PRESENTE IN, l'attributo Numero Copie dipende sia da CODICE_LIBRO, per poter sapere quali siano le informazioni del libro che viene preso in considerazione, che da NOME_SUCCURSALE, da cui si ricava l'informazione sulla posizione fisica delle copie del libro. La terza forma normale è rispettata poiché non vi è presenza di una dipendenza transitiva all'interno delle relazioni prese in considerazione: infatti tutti gli attributi presenti fanno riferimento a informazioni riguardanti solo la relazione di cui fanno parte, dipendendo quindi dalla chiave primaria della loro relazione.

3.3 Vincoli dello Schema Relazionale

Andando a studiare i vincoli relativi allo Schema Relazionale si parte dal Vincolo di integrità dell'Entità, che specifica che nessun valore di chiave primaria può essere nullo. I valori Codice Libro, Codice Autore, Codice Editore, Numero di Matricola Nome Succursale, Codice Prestito non potranno mai essere nulli in quanto andrebbero a violare, o per loro stessi, o nei casi in cui sono chiavi esterne, il vincolo di integrità dell'Entità, rischiando l'impossibilità di distinguere le tuple create all'interno della base di dati. Come accennato, gli attributi chiave primaria sopra descritti diventano in diverse relazioni chiavi esterne. Questo accade in PRESENTE IN, la cui chiave è formata da CODICE LIBRO e NOME SUCCURSALE, entrambi chiave esterne che si riferiscono a LIBRO e SUCCURSALE, in CONCESSO, con CODICE LIBRO e CODICE PRESTITO chiave e il secondo attributo chiave esterna di PRESTITO, in EFFETTUA con NUM MATRICOLA e CODICE PRESTITO chiave, e dove il primo attributo è chiave esterna di UTENTI, infine SCRITTO con CODICE LIBRO e CODICE AUTORE, dove il secondo si riferisce ad AUTORE e PUBBLICA con CODICE LIBRO e CODICE EDITORE, dove il secondo attributo si riferisce ad EDITORE. Si verifica un vincolo di chiave esterna anche nella relazione PRESTITO, dove è presente l'attributo NOME SUCCURSALE, che si riferisce a Nome all'interno della relazione SUCCURSALE.
Si passa ora a parlare delle operazioni di inserimento, modifica e cancellazione, e dei vincoli che rischiano di infrangere, ricordando che queste sono state richieste per la gestione di utenti e prestiti.
Partendo dall'inserimento di un utente, questo rischia di violare il vincolo di integrità dell'entità, vista la possibilità che un operatore inserisca un numero di matricola nullo, questo viene evitato a livello di codice php effettuando un controllo sul parametro inserito. L'eliminazione invece rischia di infrangere il vincolo di integrità referenziale, in quanto eliminando un numero di matricola dalla relazione UTENTI bisogna avere cura di rimuovere i suoi riferimenti all'interno di EFFETTUA. Anche questo viene evitato a livello di codice php, poiché se un utente ha un riferimento nella relazione EFFETTUA vuol dire che presenta dei prestiti ancora in sospeso, e non si può permettere ad un utente, con in possesso libri della biblioteca, di essere eliminato dalla base di dati, rischiando di perdere i libri. Si verifica prima che l'utente non abbia in prestito dei libri, e se questo è il caso si procede all'eliminazione. Come è solito che accada la modifica non tocca in questo caso la chiave primaria Numero di Matricola non dando quindi origine a problemi con i vincoli.
Per quanto riguarda invece la gestione dell'inserimento di un prestito vengono effettuati nel codice i seguenti controlli per evitare che ci siano infrangimenti dei vincoli: viene richiesto il Numero di Matricola dell'utente che intende registrare il prestito e si procede al controllo che sia registrato per evitare problemi sul vincolo di integrità referenziale. Si procede quindi allo stesso modo, attraverso il codice del libro, a verificare che anche il libro sia registrato per evitare problemi sul vincolo di integrità referenziale. In un secondo momento viene ricercato il numero di copie dalla relazione PRESENTE IN per il libro richiesto nella succursale desiderata, e si

procede alla verifica che il primo sia presente all'interno della sede, per evitare problemi sul vincolo di integrità referenziale, e, se il numero di copie è pari a zero allora non è possibile effettuare il prestito, se è maggiore di zero allora si diminuisce il numero di copie per tenere traccia del numero di copie di un libro presenti in una succursale della biblioteca. Infine, per registrare tutti i cambiamenti avvenuti con la generazione del prestito, l'aggiornamento della matricola che lo ha effettuato e del libro concesso vengono aggiornate le relazioni EFFETTUA, CONCESSO e PRESTITO per evitare di rompere vincoli di integrità referenziale e integrità dell'entità.
L'eliminazione di un prestito dalla base di dati, ovvero la restituzione di un libro da parte di un utente in una determinata sede, viene effettuato, come per l'inserimento, un controllo per ricavare il numero di copie del libro nella succursale desiderata. Il codice del libro viene ricavato dal codice prestito, e col numero di copie viene poi aggiornato aumentandolo di uno, per confermare il rientro del libro nella succursale della biblioteca. Infine, si procede all'eliminazione dei dati dalle relazioni EFFETTUA, CONCESSO e PRESTITO attraverso Codice Prestito per evitare problemi con il vincolo di integrità referenziale.
La modifica del prestito non viola potenzialmente vincoli, come descritto sopra, ma nella modifica del prestito è stata disposta la possibilità di modificare anche, oltre le date del prestito, il codice del libro, nel caso ci fosse stato un errore nel registrare il prestito da parte di un operatore, e la succursale di rientro del libro. Se infatti un utente è impossibilitato a restituire un libro nella succursale in cui lo ha richiesto ha la possibilità di richiedere la modifica del prestito per poterlo restituire in un'altra succursale. Perché questo sia possibile senza infrangere il vincolo di chiave, ovvero correndo il rischio di creare duplicati all'interno della base di dati, si effettua un controllo per verificare quindi che il libro in questione non sia già registrato nella succursale nuova del rientro, se non è registrato viene inserito con numero di copie pari a zero, in alternativa si modifica il valore. Si procede infine alla modifica delle relazioni EFFETTUA, CONCESSO e PRESTITO, per evitare problemi con il vincolo di integrità referenziale.

4. Interrogazioni e Statistiche di Base

4.1 Scrittura in Algebra Relazionale delle interrogazioni della sezione “Statistiche di Base”

Al fine di progettare e strutturare l’applicativo web, è stata richiesta l’implementazione di una sezione in grado di fornire gli strumenti per effettuare la ricerca di un utente e di un libro, in modo da ricavare determinate informazioni, espresse in algebra relazionale. Da questa ricerca si deve ricavare: l’elenco di tutti i libri dati in prestito ad un utente, l’elenco di tutti gli utenti che hanno in prestito un determinato libro, e l’elenco di alcune statistiche basilari, ovvero determinare, all’interno del catalogo dei libri della biblioteca, le cinque lingue più comuni, l’autore che ha scritto più libri e l’editore che ha pubblicato più libri.
Le operazioni dell’algebra relazionale consentono di costruire le interrogazioni fondamentali in termini di espressioni dell’algebra relazionale. L’utilizzo, dunque, delle regole e delle espressioni che caratterizzano l’algebra relazionale è da intendersi come obiettivo principale per creare una base che ha il fine di implementare ed ottimizzare le interrogazioni nei moduli di gestione della base di dati.
Verranno presentate di seguito le espressioni in algebra relazionale delle query implementate.

1. L’elenco di tutti i libri dati in prestito ad un utente

LIBRO_CONCESSO ← **Π** (CODICE_LIBRO, ISBN, TITOLO, CODICE_PRESTITO_CONCESSO)(LIBRO*(CODICE_LIBRO=CODICE_LIBRO _CONCESSO) CONCESSO);

UTENTI_EFFETTUA ← **Π** (N_MATRICOLA, NOME, COGNOME, CODICE_PRESTITO_EFFETTUA)(UTENTI*(N_MATRICOLA=N_MATRICOLA _EFFETTUA) EFFETTUA);

RIS ← **Π** (ISBN, titolo)(**σ**(N_MATRICOLA="matricola" || NOME="nome" || COGNOME="cognome")(LIBRO_CONCESSO*(CODICE_PRESTITO_CONCESS O=CODICE_PRESTITO_EFFETTUA) UTENTI_EFFETTUA));

LIBRO_CONCESSO è la nuova relazione che ospita tutti i libri dati in prestito ad un utente. Per ottenere ciò viene eseguita una natural join tra la relazione LIBRO e CONCESSO, con la condizione che il codice del libro, chiave primaria della relazione LIBRO, sia uguale al codice del libro, chiave esterna della relazione CONCESSO.

UTENTI_EFFETTUA invece è una nuova relazione che contiene al suo interno tutti gli utenti che hanno effettuato un prestito. A tal fine viene eseguita una natural join tra la relazione UTENTI e la relazione EFFETTUA, con la condizione che il numero di matricola, chiave primaria della relazione UTENTI sia uguale al numero di matricola, chiave esterna della relazione EFFETTUA.
Il passaggio successivo porta, con la combinazione delle due relazioni, alla creazione di una nuova relazione, che ci mostra per ogni libro concesso in prestito, l'utente che lo ha preso, grazie all'operazione di join tra le due relazioni con la condizione che la chiave primaria CODICE_PRESTITO_CONCESSO sia uguale alla chiave primaria CODICE_PRESTITO_EFFETTUA. La select e la project identificano quindi, attraverso le selezioni, i dati che interessano all'utente e che soddisfano la ricerca.

2. L'elenco di tutti gli utenti che hanno in prestito un determinato libro

LIBRO_CONCESSO ← **Π** (CODICE_LIBRO, ISBN, TITOLO, CODICE_PRESTITO)(LIBRO*(CODICE_LIBRO = CODICE_LIBRO_CONCESSO) CONCESSO);

UTENTI_EFFETTUA ←
Π (N_MATRICOLA, NOME, COGNOME, CODICE_PRESTITO)(UTENTI* (N_MATRICOLA=N_MATRICOLA_EFFETTUA) EFFETTUA);

UTENTE_LIBRO ←
LIBRO_CONCESSO*(CODICE_PRESTITO_CONCESSO=CODICE_PRESTITO_EFFETTUA) UTENTI_EFFETTUA;

RIS ←
Π(N_MATRICOLA,NOME,COGNOME)(**σ**(TITOLO="titolo")(UTENTE_LIBRO));

Come descritto sopra, LIBRO_CONCESSO è la nuova relazione che ospita tutti i libri dati in prestito ad un utente. Per ottenere ciò viene eseguita una natural join tra la relazione LIBRO e CONCESSO, con la condizione che il codice del libro, chiave primaria della relazione LIBRO, sia uguale al codice del libro, chiave esterna della relazione CONCESSO.
Allo stesso modo, UTENTI_EFFETTUA è una nuova relazione che contiene al suo interno tutti gli utenti che hanno effettuato un prestito. A tal fine viene eseguita una natural join tra la relazione UTENTI e la relazione EFFETTUA, con la condizione che il numero di matricola, chiave primaria della relazione UTENTI sia uguale al numero di matricola, chiave esterna della relazione EFFETTUA.

Il passaggio successivo porta, con la combinazione delle due relazioni, alla creazione di una nuova relazione, che ci mostra per ogni libro concesso in prestito, l'utente che lo ha preso, grazie all'operazione di join tra le due relazioni con la condizione che la chiave primaria CODICE_PRESTITO_CONCESSO sia uguale alla chiave primaria CODICE_PRESTITO_EFFETTUA. Il risultato finale mostra i dati degli utenti che hanno preso in prestito un determinato libro.

3. Determinare le cinque lingue più comuni in cui sono stati scritti i libri presenti nella base di dati

ρ (Lingua, Quantità)(Lingue) ← LINGUA $\mathscr{F}$ COUNT $_{\text{LINGUA}}$ (LIBRO)

Utilizzando la funzione di aggregazione COUNT all'interno della relazione LIBRO, che contiene tutti i libri registrati nella base di dati, è possibile contare per ogni lingua quanti libri sono stati scritti.

4. Determinare l'autore che ha scritto più libri

ρ (Codice_autore, contatore)(n_autori) ←
CODICE_AUTORE_SCRITTO $\mathscr{F}$ COUNT $_{\text{CODICE\_LIBRO}}$ (SCRITTO);

LISTA_AUTORI ←
Π (nome, cognome, contatore)(n_autori*****(codice_autore_scritto=codice_autore) AUTORE);

MAX ← $\mathscr{F}$ MAXIMUM $_{\text{CONTATORE}}$ (LISTA_AUTORI)

RIS ←**Π** (NOME, COGNOME)(**σ**(CONTATORE=MAX)(LISTA_AUTORI));

Inizialmente viene creata la relazione n_autori al cui interno risiedono gli attributi codice autore e contatore. Utilizzando la funzione COUNT, viene aggiornato, ogni volta che compare, per ogni codice autore nella relazione SCRITTO il numero complessivo di libri scritti per ogni autore.
Dato che è nell'interesse della richiesta conoscere il nome ed il cognome dell'autore si crea una relazione temporanea che ospita gli attributi Nome Cognome e contatore. Posso risalire al nome e al cognome dell'autore tramite il codice dell'autore, chiave primaria che identifica la tupla, tenendo quindi traccia non più del codice ma dei dati generali dell'autore, ai quali è sempre associato il numero totale di libri scritti. Grazie alla tupla MAX si salva il numero massimo di libri scritti dell'autore che soddisfa la richiesta, e, con l'opportuna select, si ottiene infine il risultato.

5. Determinare l’editore che ha pubblicato più libri

ρ (CODICE_EDITORE_PUBBLICA, contatore)(EDITORI_LIBRI) ← CODICE_EDITORE_PUBBLICA $\mathscr{F}$ COUNT Codice_libro(PUBBLICA);

EDITORI_COMPLETO ←
Π (Nome, Indirizzo, contatore)
(EDITORI_LIBRI*(CODICE_EDITORE_PUBBLICA = Codice_Editore) EDITORE);

MAX ← $\mathscr{F}$ MAXIMUM $_{\text{contatore}}$;

RIS ← **Π** (Nome, Indirizzo)(**σ**(contatore=MAX)(EDITORI_COMPLETO));

EDITORI_LIBRI è una relazione che contiene il risultato della funzione aggregata COUNT che conta, per ogni codice editore, quanti libri sono stati pubblicati dallo stesso.
EDITORI_COMPLETO è invece la relazione che racchiude tutte le informazioni relative all'editore, ovvero identifica, per ogni codice editore, il rispettivo nome della casa editrice e l’indirizzo della sede, grazie alla natural join tra la relazione EDITORE e la relazione EDITORI_LIBRI, con la condizione che la chiave primaria Codice_editore di EDITORE sia uguale alla chiave primaria CODICE_EDITORE_PUBBLICA di EDITORI_COMPLETO.
Con la funzione di aggregazione MAXIMUM si ottiene il valore massimo dell’attributo contatore.
Infine, si mostra il risultato dell'operazione di proiezione, che prende il nome e l'indirizzo dell'editore per il quale contatore è uguale al massimo, e restituisce i valori richiesti dalla query, ottenendo quindi l’editore che ha pubblicato più libri.

4.2 Scrittura in sintassi SQL delle interrogazioni della sezione “Statistiche di Base”

La logica di funzionamento delle query che verranno presentate ora in linguaggio SQL è la stessa di quella della sezione 4.1. Per questo motivo si procederà a mostrare i codici sviluppati per il funzionamento effettivo dell’applicativo web.
Per mantenere la corretta indentazione delle query, che mostra visivamente come vengono innestate è stato preferito mostrare delle immagini prese dall’applicazione MySQL Workbench.

1. L’elenco di tutti i libri dati in prestito ad un utente

```
SELECT CODICE_LIBRO,ISBN,TITOLO, LINGUA, ANNO_PUBBLICAZIONE
FROM UTENTI AS U,EFFETTUA AS E,LIBRO AS L,CONCESSO AS C
WHERE L.CODICE_LIBRO=C.CODICE_LIBRO_CONCESSO
AND E.CODICE_PRESTITO_EFFETTUA=C.CODICE_PRESTITO_CONCESSO
AND E.N_MATRICOLA_EFFETTUA=U.N_MATRICOLA
AND (U.N_MATRICOLA = '$N_MATRICOLA' OR U.NOME_UTENTE='$NOME_UTENTE' OR U.COGNOME_UTENTE='$COGNOME_UTENTE')
ORDER BY CODICE_LIBRO ASC;
```

2. L’elenco di tutti gli utenti che hanno in prestito un determinato libro

```
SELECT N_MATRICOLA, NOME_UTENTE, COGNOME_UTENTE, N_TELEFONO
FROM UTENTI AS U,EFFETTUA AS E,LIBRO AS L,CONCESSO AS C
WHERE L.CODICE_LIBRO=C.CODICE_LIBRO_CONCESSO
AND E.CODICE_PRESTITO_EFFETTUA=C.CODICE_PRESTITO_CONCESSO
AND E.N_MATRICOLA_EFFETTUA=U.N_MATRICOLA
AND (L.CODICE_LIBRO='$CODICE_LIBRO' OR L.TITOLO LIKE '$TITOLO' OR L.ISBN='$ISBN')
ORDER BY U.N_MATRICOLA ASC;
```

3. Determinare le cinque lingue più comuni in cui sono stati scritti i libri presenti nella base di dati

```
SELECT LINGUA, COUNT(LINGUA) AS L
FROM LIBRO
group by LINGUA
ORDER BY L DESC
LIMIT 5;
```

4. Determinare l’autore che ha scritto più libri

```
SELECT CODICE_AUTORE, NOME, COGNOME, N_LIBRI_SCRITTI
FROM AUTORE AS A, (SELECT CODICE_AUTORE_SCRITTO, COUNT(CODICE_AUTORE_SCRITTO) AS N_LIBRI_SCRITTI
                   FROM SCRITTO
                   GROUP BY CODICE_AUTORE_SCRITTO
                   HAVING COUNT(CODICE_AUTORE_SCRITTO) = (SELECT MAX(N)
                                                          FROM (SELECT CODICE_AUTORE_SCRITTO, COUNT(CODICE_AUTORE_SCRITTO) AS N
                                                                FROM SCRITTO
                                                                GROUP BY CODICE_AUTORE_SCRITTO) AS I)) AS N_LIBRI
WHERE A.CODICE_AUTORE = N_LIBRI.CODICE_AUTORE_SCRITTO;
```

5. Determinare l’editore che ha pubblicato più libri

```
SELECT CODICE_EDITORE, NOME_EDITORE, N_LIBRI_PUBBLICATI
FROM EDITORE AS E, (SELECT CODICE_EDITORE_PUBBLICA, COUNT(CODICE_EDITORE_PUBBLICA) AS N_LIBRI_PUBBLICATI
                    FROM PUBBLICA
                    GROUP BY CODICE_EDITORE_PUBBLICA
                    HAVING COUNT(CODICE_EDITORE_PUBBLICA) = (SELECT MAX(N)
                                                             FROM (SELECT CODICE_EDITORE_PUBBLICA, COUNT(CODICE_EDITORE_PUBBLICA) AS N
                                                                   FROM PUBBLICA
                                                                   GROUP BY CODICE_EDITORE_PUBBLICA) AS CASA_EDITRICE)) AS N_LIBRI
WHERE E.CODICE_EDITORE=N_LIBRI.CODICE_EDITORE_PUBBLICA
```

4.3 Scrittura in Algebra Relazionale delle interrogazioni proposte dal gruppo

Oltre alle interrogazioni e alle statistiche richieste sono state proposte dal gruppo ulteriori idee per arricchire l’applicativo web. In particolare, l’obiettivo è stato quello di mettere in evidenza statistiche o richieste che, con maggiore probabilità, potessero risultare utili all’operatore della biblioteca di Ateneo o all’utente.
Considerato questo sono state implementate le seguenti interrogazioni aggiuntive: trovare l’elenco delle sedi dove è presente un determinato libro, verificare presso quale sede è stato effettuato il maggior numero di prestiti, determinare il libro più richiesto in prestito, elencare tutti i libri disponibili o non disponibili per il prestito.

1. Elencare tutte le sedi in cui è presente un determinato libro

RIS ← **Π** (NOME_SUCCURSALE) (**σ** (TITOLO="titolo" || CODICE_LIBRO="Codice_libro" || ISBN=” ISBN”) (PRESENTE_IN*(CODICE_LIBRO=CODICE_LIBRO) LIBRO);

Grazie alla join tra PRESENTE_IN e LIBRO, si ottiene una relazione dove, per ogni libro, è associata la sede dove esso è presente. Viene data l’opzione di inserire un parametro tra il titolo, il codice del libro o l’ISBN per poter selezionare ciò che si desidera. Questa query è stata pensata per offrire un servizio all’utente che, nel caso

non trovi il libro che desidera prendere in prestito, può ottenere una lista di tutte le sedi che ne hanno la disponibilità.
Vista la presenza all'interno dell'applicativo web nella sezione "Ricerca e Statistiche" della funzione "Cerca Libro" è stato pensato di inserire le informazioni ricavate da questa query nella sezione sopracitata, per mostrare più informazioni.

2. Verificare presso quale sede è stato effettuato il maggior numero di prestiti

ρ (Nome_Succursale, contatore)(succursale_prestito) ←
Nome_Succursale $\mathscr{F}$ COUNT Codice_Prestito(PRESTITO);

MAX ← $\mathscr{F}$ MAXIMUM contatore(succursale_prestito);

RIS ← **Π**(Nome_Succursale)(**σ**(contatore=MAX)(succursale_prestito));

Grazie agli attributi presenti all'interno della relazione PRESTITO è possibile tenere traccia di ogni libro in uscita, e la sede presso la quale viene effettuato il prestito. Con la funzione COUNT si ottiene il numero totale di prestiti effettuati presso ogni sede, il valore massimo ottenuto viene salvato nella tupla temporanea MAX che viene poi utilizzata per il confronto necessario per stabilire quale succursale abbia registrato il maggior numero di prestiti.
Questa query è stata pensata per fornire un dato statistico riguardante le sedi, immaginando di tenere traccia di questo dato a favore di una statistica finale, a cadenza annuale, che monitora il numero di prestiti effettuati presso ogni biblioteca di ateneo.

3. Determinare il libro più richiesto in prestito

ρ (Codice_libro_prestiti, contatore)(NUMERO_PRESTITI) ←
Codice_Libro $\mathscr{F}$ COUNT Codice_Prestito (CONCESSO)

ρ (Cod_Libro, N_prestiti) (MAX) ←
$\mathscr{F}$ MAXIMUM contatore (NUMERO_PRESTITI)

NOME_MAX ← MAX * LIBRO (Codice_Libro = codice)

RIS ← **Π** (Codice_Libro, Titolo, ISBN)(NOME_MAX)

In primo luogo, viene creata una relazione NUMERO_PRESTITI che fornisce il numero di volte in cui un libro è stato concesso in prestito, grazie all'utilizzo della funzione COUNT, che per ogni codice libro conta quante volte compare il Codice_Prestito.
A quel punto, attraverso NUMERO_PRESTITI si ricava il massimo, salvato insieme al corrispettivo Codice_Libro all'interno della relazione MAX.

Attraverso la join tra la relazione MAX e la relazione LIBRO si ottiene per il libro richiesto gli attributi codice del libro, Titolo, e l'ISBN. Essendo una sola riga verranno proiettati tutti gli attributi che presenteranno la statistica richiesta.

4. Determinare tutti i libri non disponibili per il prestito

RIS←
Π(Codice_Libro, Titolo, ISBN, Nome_Succursale_Presente_In)
(**σ**(numero_copie=0)
((LIBRO) * (Codice_Libro = Codice_Libro_Presente_In) (PRESENTE_IN)));

5. Determinare tutti i libri disponibili per il prestito

RIS←
Π (Codice_Libro, Titolo, ISBN, Nome_Succursale_Presente_In)
(**σ**(numero_copie>0)
((LIBRO) * (Codice_Libro = Codice_Libro_Presente_In) (PRESENTE_IN)));

La relazione PRESENTE_IN contiene tutte le informazioni relative al libro, alla sede in cui è presente il libro e al numero di copie disponibili presso quella sede. LIBRO, invece, fornisce tutte le informazioni dettagliate in merito ad un determinato libro, identificato univocamente dal suo codice. La natural join tra queste due relazioni dunque, con la condizione che la chiave primaria Codice_Libro di LIBRO e la chiave esterna Codice_Libro_Presente_In di PRESENTE_IN siano uguali, restituisce una relazione che fornisce, oltre che al codice, tutte le informazioni dettagliate in merito al libro.
I dati che interessa restituire ai fini della ricerca sono il codice del libro, il titolo, l'ISBN e il nome della succursale dove il libro è presente, selezionati attraverso l'operatore select. Da un lato, nella query 5, l'interesse è selezionare solo i libri disponibili e dunque dove il numero di copie è maggiore di 0, viceversa nella query 4, il dato di interesse sono i libri non disponibili per il prestito e dunque tutte quelle tuple che presentano nel numero di copie il valore 0.
La Query 4 e la Query 5 nascono dalla volontà di voler determinare, nel modo più veloce possibile, una lista di libri che siano disponibili o non disponibili per un prestito, in quanto la risorsa, nel nostro caso un libro, è limitata al numero di copie presenti, non è quindi garantito che sia disponibile in un determinato momento.

4.4 Scrittura in sintassi SQL delle interrogazioni proposte dal gruppo

La logica dietro le query che verranno ora esposte è stata spiegata nella sezione 4.3, ci si limiterà quindi a mostrarle tradotte in sintassi SQL, come nella sezione 4.2, attraverso immagini riportate dall'applicazione MySQL Workbench, per mostrare la corretta indentazione.

1. Elencare tutte le sedi in cui è presente un determinato libro

```sql
SELECT NOME_SUCCURSALE_PRESENTE_IN
FROM PRESENTE_IN AS P, LIBRO AS L
WHERE L.TITOLO = "$TITOLO" OR L.CODICE_LIBRO = "$CODICE_LIBRO"
AND L.CODICE_LIBRO = P.CODICE_LIBRO_PRESENTE_IN;
```

2. Verificare presso quale sede è stato effettuato il maggior numero di prestiti

```sql
SELECT NOME_SUCCURSALE_PRESTITO,COD_P
FROM (SELECT NOME_SUCCURSALE_PRESTITO, COUNT(CODICE_PRESTITO) AS COD_P
      FROM PRESTITO
      GROUP BY NOME_SUCCURSALE_PRESTITO
      HAVING COUNT(CODICE_PRESTITO)= (SELECT MAX(CP)
                                      FROM(SELECT NOME_SUCCURSALE_PRESTITO, COUNT(CODICE_PRESTITO) AS CP
                                           FROM PRESTITO
                                           GROUP BY NOME_SUCCURSALE_PRESTITO
                                           ORDER BY CP DESC)AS MAX)) AS MAX_PRESTITO
ORDER BY NOME_SUCCURSALE_PRESTITO ASC
```

3. Determinare il libro più richiesto in prestito

```sql
SELECT CODICE_LIBRO,TITOLO,ISBN,COD_P
FROM LIBRO AS LI,(SELECT CODICE_LIBRO_CONCESSO, COD_P
                  FROM (SELECT CODICE_LIBRO_CONCESSO, COUNT(CODICE_PRESTITO_CONCESSO) AS COD_P
                        FROM CONCESSO
                        GROUP BY CODICE_LIBRO_CONCESSO
                        HAVING COD_P=(SELECT MAX(CPC)
                                      FROM(SELECT CODICE_LIBRO_CONCESSO, COUNT(CODICE_PRESTITO_CONCESSO) AS CPC
                                           FROM CONCESSO
                                           GROUP BY CODICE_LIBRO_CONCESSO
                                           ORDER BY CPC DESC)AS MAS_QUERY))AS TAB_COMPLETE) AS L
WHERE LI.CODICE_LIBRO = L.CODICE_LIBRO_CONCESSO
ORDER BY LI.CODICE_LIBRO ASC;
```

4. Determinare tutti i libri non disponibili per il prestito

```
SELECT CODICE_LIBRO,TITOLO,ISBN,NOME_SUCCURSALE_PRESENTE_IN
FROM LIBRO JOIN PRESENTE_IN ON (CODICE_LIBRO=CODICE_LIBRO_PRESENTE_IN)
WHERE NUMERO_COPIE='0'
ORDER BY "CODICE_LIBRO"
```

5. Determinare tutti i libri disponibili per il prestito

```
SELECT CODICE_LIBRO,TITOLO,ISBN,NOME_SUCCURSALE_PRESENTE_IN
FROM LIBRO JOIN PRESENTE_IN ON (CODICE_LIBRO=CODICE_LIBRO_PRESENTE_IN)
WHERE NUMERO_COPIE > 0
ORDER BY CODICE_LIBRO
```

Appendice: Contenuto File ZIP

Per motivi di completezza è stato pensato di spiegare il contenuto del file ZIP utilizzato per la consegna del progetto. Dentro la cartella “Biblioteca” sono presenti tutti i file, sviluppati in HTML e PHP, per la gestione dell’applicativo web.
All’interno di “File Dati” sono presenti tutti i file utilizzati per lo sviluppo della base di dati, in formato CSV, da dove sono stati ricavati i dati in seguito inseriti, eccezione fatta per le tabelle EDITORE e SUCCURSALE, riempite manualmente con codice SQL rintracciabile nel file Biblioteca.sql all’interno di “Biblioteca”.
La cartella “Genera Dati” contiene script in linguaggio PHP sviluppati con il fine di generare casualmente dati per effettuare test sull’applicativo web. Questi file sono stati quindi utilizzati per creare i dati visibili nei file CSV della cartella “File Dati”.
“Query Sql” contiene infine le query implementate in linguaggio SQL e presentate nella sezione 4.2 e 4.4 della relazione.