

Sistema di una Biblioteca Digitale

Gennaro Nappo - Giovanni Riccio
N86004294 - N86004316

Anno accademico 2022-2023



UNIVERSITÀ DEGLI STUDI
DI NAPOLI FEDERICO II

Indice

1 Progettazione concettuale	3
1.1 Analisi dei requisiti	3
1.2 Schema concettuale	5
2 Ristrutturazione del modello concettuale	6
2.1 Analisi delle ridondanze	6
2.2 Analisi delle generalizzazioni	6
2.3 Eliminazione degli attributi multivалore	6
2.4 Eliminazione degli attributi strutturati	6
2.5 Analisi di Entità e Associazioni	7
2.6 Identificazione delle chiavi primarie	7
2.7 Schema ristrutturato ER e UML	7
3 Dizionari	9
3.1 Dizionario delle Entità	9
3.2 Dizionario delle Relazioni	12
3.3 Dizionario dei Vincoli	15
4 Schema Logico e descrizioni	22
4.1 Schema Logico	22
4.2 Descrizione di Trigger	24
4.3 Descrizioni di Funzioni e Procedure	27

1 Progettazione concettuale

1.1 Analisi dei requisiti

In questa sezione sono individuate le informazioni rilevanti per poter soddisfare le richieste informative e funzionali del DB, in particolare sono individuate le *Entità*, le *Relazioni*, i *Vincoli* e le operazioni che gli utenti possono effettuare più frequentemente.

"Si sviluppi un sistema informativo, composto da una base di dati relazionale per la gestione di una biblioteca digitale"

Come da richiesta, consideriamo di gestire un'unica grande biblioteca digitale, cioè un catalogo di elementi che gli utenti possono consultare, quindi non necessiteremo di un'entità *Biblioteca* per poter rappresentare la biblioteca del mondo reale.

"Gli elementi che possono essere inclusi nella biblioteca digitale sono di due tipi: articoli scientifici e libri (didattici o romanzi). Per ognuno di essi, devono essere specificati il titolo, l'anno di pubblicazione, gli autori, l'editore e definire le modalità di fruizione (cartaceo, digitale o audiolibro)."

Abbiamo la necessità di dover rappresentare come entità gli elementi inclusi nella biblioteca, cioè gli articoli scientifici e i libri, quindi avremo le entità *articolo_scientifico* e *libro*.

Alcuni articoli scientifici saranno introdotti in dei fascicoli, che a loro volta faranno parte di riviste, quindi avremo bisogno delle entità *rivista* e *fascicolo*, quest'ultima sarà un'entità debole in quanto non potrà esistere se non ci sono delle riviste e degli articoli scientifici associati.

Ogni libreria sarà gestita da un utente che possiede una partita IVA, quindi avremo bisogno di un'entità *venditore*, ovvero una specializzazione dell'entità *utente*, che avrà un attributo *PartitaIVA* e quindi potrà gestire una libreria. Le librerie saranno rappresentate dall'entità *libreria* e gli elementi che possiedono, sono rappresentati dalle seguenti sottoclassi dell'entità *elemento*:

- *fascicoli*;
- *serie*;
- *libri*.

Ognuno di queste entità avrà il titolo, la data di pubblicazione, gli autori e l'editore come attributi. Inoltre la modalità di fruizione sarà definita come attributo della relazione *possesso*, che mette in relazione le entità *libreria* ed *elemento*, questa relazione avrà anche un attributo *Quantita* per avere traccia della quantità di elementi disponibili di ogni libreria .

"Per i libri è importante definire una eventuale presentazione. Un libro può anche far parte di una collana, la quale può raggruppare tutti libri che condividono una determinata caratteristica."

Si richiede di gestire eventuali presentazioni dei libri, ed eventuali collane alle quali apparterranno dei libri, quindi necessiteremo delle entità *presentazione* e *collana*, le quali saranno deboli in quanto non potranno esistere se non saranno associate a dei libri.

Per le pubblicazioni, andare a definire in quale rivista (nome, argomento, anno di pubblicazione, responsabile della rivista) o in quale conferenza (luogo della conferenza, data di inizio e data fine conferenza, struttura organizzatrice) è stato pubblicato.

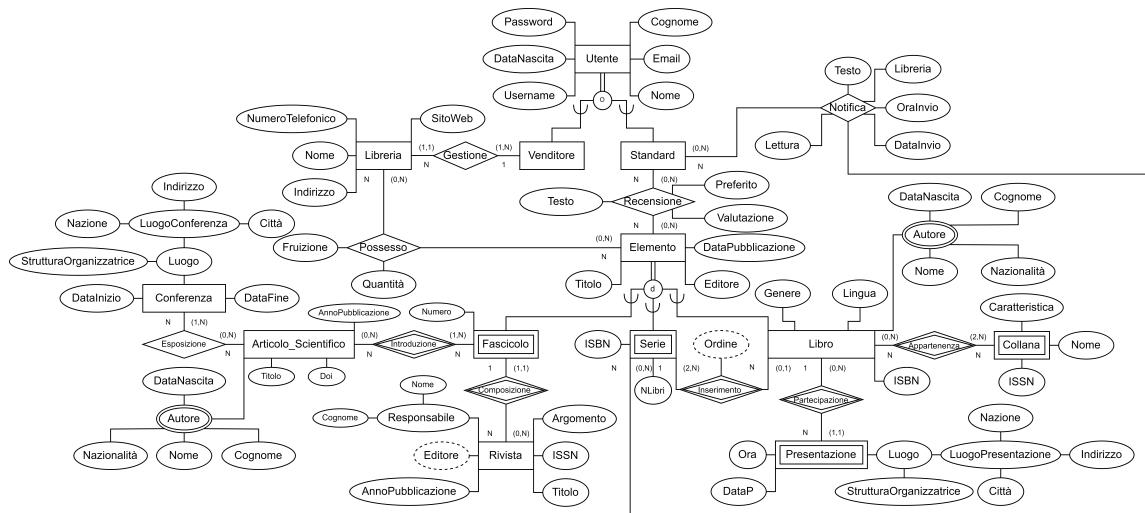
Per gli articoli scientifici è importante avere traccia in quale rivista sono introdotti, ovvero in quale rivista è contenuto il fascicolo in cui è stato pubblicato l'articolo, e in quale conferenza sono stati esposti, quindi avremo bisogno dell'entità *rivista* che avrà i seguenti attributi: Titolo; Argomento; Anno di pubblicazione e Responsabile, quest'entità sarà in relazione con l'entità *fascicolo*, che a sua volta sarà in relazione con l'entità *articolo_scientifico*. Inoltre necessiteremo anche dell'entità *conferenza* con i seguenti attributi: Luogo, che sarà formato dall'indirizzo e dalla struttura organizzatrice; Data di Inizio e Data di Fine. Questa entità sarà in relazione con l'entità *articolo_scientifico*.

per ogni libro, è necessario specificare dove può essere acquistato. Un romanzo può avere anche uno o più seguiti. In tal caso, è importante prevedere un'interrogazione che permette di recuperare tutte le librerie dai quali è possibile acquistare l'intera serie dei libri. Non appena una serie sarà disponibile per l'acquisto da almeno una libreria, il sistema notificherà la disponibilità all'utente.

Per permettere al sistema di specificare dove può essere acquistato ogni elemento posseduto da una libreria (fascicolo di una rivista, libro e serie di libri) avremo bisogno della relazione *possesso* tra le entità *libreria* ed *elemento*. Una serie sarà formata da un certo numero di libri, quindi anche l'entità *serie* sarà debole perché non potrà esistere se non contiene libri. Le entità *serie* e *libro* saranno in relazione tramite l'associazione *inserimento*. Per inviare la notifica a gli utenti interessati ad una serie, utilizzeremo:

- Una relazione *recensione* tra *utente* ed *elemento*: questa relazione permetterà di avere traccia di una valutazione e una recensione (presenti rispettivamente negli attributi Valutazione e Testo) fatte dall'utente. Inoltre avrà un attributo Preferito che indica gli utenti che hanno tra i preferiti degli elementi, e quindi indicherà anche se l'utente, che partecipa a questa relazione, vuole ricevere una notifica quando la serie associata è completamente disponibile in qualche libreria;
- Una relazione *notifica* tra le entità *utente* e *serie*: che conterrà tutte le notifiche inviate agli utenti che partecipano alla relazione *recensione* e che hanno l'attributo Preferito a 'true' (e quindi hanno attivato le notifiche per una certa serie), ogni volta che una serie associata a tali utenti è disponibile presso qualche libreria.
- Trigger e Funzioni che si occuperanno di inviare automaticamente le notifiche quando:
 - Un'utente attiva le notifiche per una serie già disponibile, cioè nella relazione *recensione* avviene un inserimento o un modifica in cui l'attributo Preferito assume il valore 'true' e l'elemento associato a tale utente è una serie;
 - Una libreria inizia a possedere una serie per la quale un utente ha attivato le notifiche, cioè nella relazione *possesso* avviene un inserimento o una modifica in cui l'attributo Quantità ha un valore maggiore di 0 oppure la modalità di fruizione è 'Digitale' o 'AudioLibro' (quindi non è necessario specificare la quantità disponibile).

1.2 Schema concettuale



2 Ristrutturazione del modello concettuale

Prima di poter passare allo schema logico è necessario ristrutturare il diagramma delle classi.

2.1 Analisi delle ridondanze

L'editore di una rivista può essere ricavato consultando l'attributo 'Editore' dell'entità *fascicolo*, quindi non è necessario inserire tale attributo nell'entità *rivista*. L'attributo 'Ordine' della relazione *Inserimento* può essere calcolato contando il numero di libri associati a una serie, quindi anche questo attributo non verrà inserito nella sua relazione.

In questo modo avremo una minore occupazione di spazio e degli aggiornamenti meno pesanti.

2.2 Analisi delle generalizzazioni

Per rimuovere la generalizzazione totale overlapping "utente", accorpiamo le entità figlie *venditore* e *standard* nell'entità padre *utente*. Per quanto riguarda la generalizzazione disgiunta totale "elemento" procediamo accorpando l'entità padre *elemento* nelle entità figlie *fascicolo*, *libro* e *serie*.

2.3 Eliminazione degli attributi multivalore

Nelle entità *articolo_scientifico* e *libro* abbiamo un attributo multivalore (e strutturato) *Autore* che ci permette di rappresentare gli autori che hanno scritto un libro e/o un articolo scientifico, quindi abbiamo creato delle apposite entità esterne.

2.4 Eliminazione degli attributi strutturati

L'entità *conferenza* contiene l'attributo composto *Luogo* che è formato da *Struttura Organizzatrice* e *Luogo Conferenza*, quest'ultimo a sua volta è composto da *Nazione*, *Città* e *Indirizzo*. Quindi trascuriamo la struttura dell'attributo *Luogo Conferenza* rendendolo un unico attributo, poi estraiamo gli attributi *Luogo Conferenza* e *Struttura Organizzatrice* inserendo nell'entità *conferenza* i corrispondenti attributi *Luogo* e *Struttura organizzatrice*.

Procediamo in modo analogo con l'attributo composto *Luogo* dell'entità *presentazione*, in questo modo avremo gli attributi *Luogo* e *Struttura* nell'entità *presentazione*.

Un'altro attributo composto è *Responsabile* nell'entità *rivista*, composto dagli attributi *Nome* e *Cognome*. Anche in questo caso procediamo estraendo questi attributi e introduciamo nell'entità *rivista* i corrispondenti attributi *Nome R* e *Cognome R*.

Infine anche gli attributi *autore* delle entità *articolo_scientifico* e *libro* (citati alla sezione 2.3) sono strutturati (e multivalore) con i seguenti attributi:

- *Nome*;
- *Cognome*;
- *Data di Nascita*;
- *Nazionalità*;

quindi le entità esterne create appositamente nella sezione 2.3 avranno gli attributi sopraelencati.

2.5 Analisi di Entità e Associazioni

Sarà utile rendere tutte le entità deboli in entità forti, in modo tale che possiamo accedere direttamente alle informazioni di tali entità, senza dover prima 'passare' per altre entità, ma potremo farlo semplicemente conoscendo la chiavi primarie di queste entità.

Le entità introdotte in precedenza per tenere traccia degli autori dei libri e degli articoli scientifici, contengono entrambe le stesse informazioni, quindi accorpiamo queste entità in un'unica entità *autore*, la quale sarà in relazione con le entità *libro* e *articolo_scientifico*.

2.6 Identificazione delle chiavi primarie

Per rendere alcune delle entità deboli in entità forti, inseriamo degli appositi codici come chiavi primarie nelle rispettive entità, come mostrato nel seguente elenco:

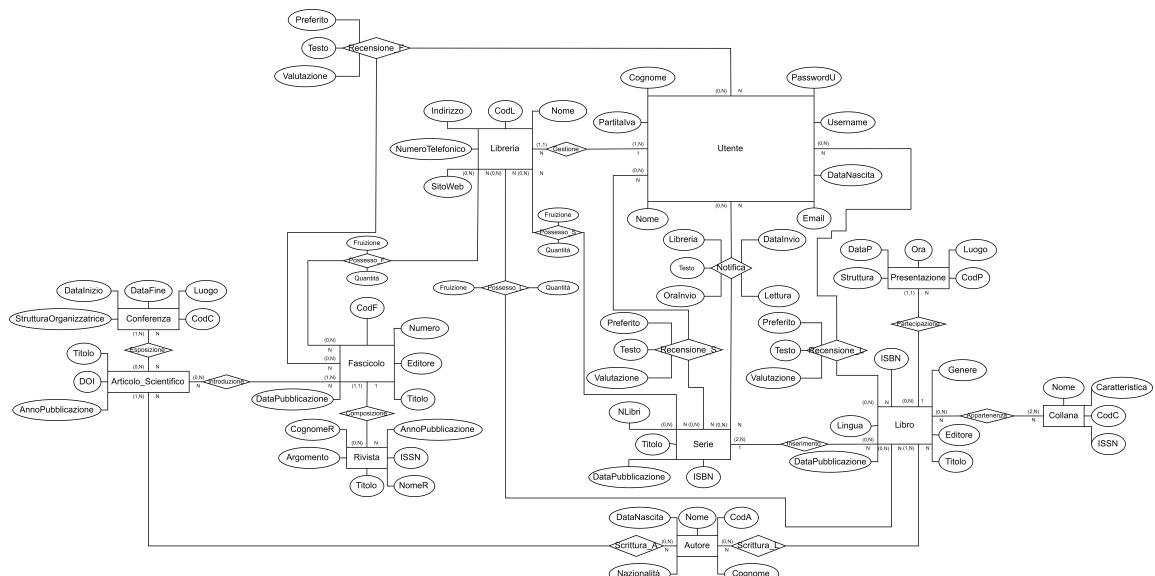
- *CodF* nell'entità *fascicolo*;
- *CodP* nell'entità *presentazione*;
- *CodC* nell'entità *collana*, l'attributo *ISSN* identifica solo le collane che vengono pubblicate periodicamente, quindi non può essere utilizzato come chiave primaria di questa entità.

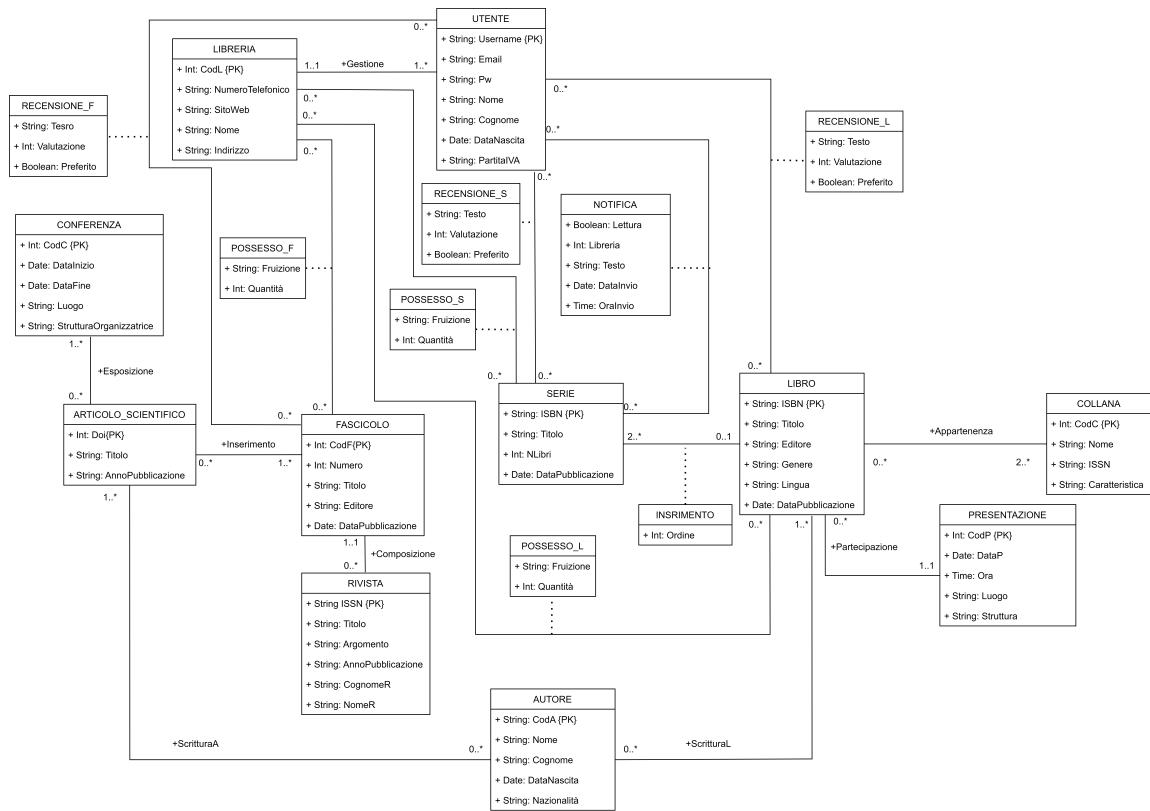
L'entità *serie* sarà identificata tramite l'attributo *ISBN*.

Inoltre introduciamo anche i codici *CodL*, *CodC*, *CodA*, questi codici identificheranno rispettivamente le entità *libreria*, *conferenza* e *autore*.

L'entità *utente* sarà identificata dall'attributo *username*, mentre l'entità *rivista* sarà individuata da *ISSN* e infine l'entità *articolo_scientifico* sarà riconosciuta da un attributo *DOI*.

2.7 Schema ristrutturato ER e UML





3 Dizionari

3.1 Dizionario delle Entità

Classi	Descrizione	Attributi
UTENTE	Utente registrato al sistema.	Username (String): Indica l'username con il quale si identifica l'utente; Email (String): Indirizzo Email dell'utente; PasswordU (String): Password d'accesso dell'utente; PartitaIVA (String): Partita IVA dell'utente che vuole registrare la propria libreria nel sistema; Nome (String): Nome dell'utente; Cognome (String): Cognome dell'utente.
LIBRERIA	Libreria gestista da un utente con la partita IVA.	CodL (Int): Codice che identifica la libreria; NumeroTelefonico (String): Numero di telefono della libreria; SitoWeb (String): Indirizzo Web della libreria; Nome (String): Nome della libreria; Indirizzo (String): Indirizzo della libreria; Gestore (String): Username dell'utente che gestisce la libreria.
RIVISTA	Rivista contenente i fascicoli.	ISSN (String): Identifica univocamente la rivista; Argomento (String): Argomento principale trattato nella rivista; CognomeR (String): Cognome del responsabile della rivista; NomeR (String): Nome del responsabile della rivista; Titolo (String): Titolo della rivista; AnnoPubblicazione (Int): Anno di pubblicazione della rivista.

Classi	Descrizione	Attributi
ARTICOLO_SCIENTIFICO	Articolo scientifico (o pubblicazione) presente in un fascicolo di una rivista.	DOI (String): Identifica univocamente un articolo scientifico; Titolo (String): Titolo dell'articolo scientifico; AnnoPubblicazione (INT): Anno di pubblicazione dell'articolo scientifico.
FASCICOLO	Fascicolo di una rivista contenente degli articoli scientifici.	CodF (Int): Codice che identifica univocamente il fascicolo; Numero (Int): Numero del fascicolo in una rivista; Editore (String): Editore del fascicolo; DataPubblicazione (Date): Data di pubblicazione del fascicolo; ISSN (String): Rivista che contiene il fascicolo.
CONFERENZA	Conferenza in cui sono stati pubblicati degli articoli scientifici.	CodC (Int): Codice che identifica univocamente la conferenza; Luogo (String): Indirizzo in cui è avvenuta la conferenza; StrutturaOrganizzatrice (String): Struttura che ha organizzato la conferenza; DataInizio (Date): Data in cui è iniziata la conferenza; DataFine (Date): Data in cui è finita la conferenza.
SERIE	Serie di libri.	ISBN (String): Identifica univocamente la serie; Titolo (String): Titolo della serie; DataPubblicazione (Date): Data di pubblicazione della serie; NLibri (Int): Numero di libri che compongono la serie.

Classi	Descrizione	Attributi
LIBRO	Libro registrato nel sistema.	ISBN(String): Identifica univocamente il libro; Titolo(String): Titolo del libro; Genere(String): Genere del libro (didattico o romanzo); Lingua(String): Lingua in cui è scritto il libro; Editore(String): Editore del libro; DataPubblicazione(Date): Data di pubblicazione del libro.
PRESENTAZIONE	Presentazione in cui è stato esposto un libro.	CodP(Int): Codice che identifica univocamente la presentazione; Luogo(String): Luogo in cui è avvenuta la presentazione; Struttura(String): Struttura in cui è stata organizzata la presentazione; DataP(Date): Data in cui è stata organizzata la presentazione; ISBN(String): Libro esposto durante la presentazione.
COLLANA	Collana di libri che condividono una determinata caratteristica.	CodC(Int): Codice che identifica univocamente la collana; Nome(String): Nome della collana; ISSN(String): Identifica univocamente le collane che vengono pubblicate periodicamente; Caratteristica(String): Caratteristica condivisa dai libri della collana.
AUTORE	Autore di libri e/o articoli scientifici.	CodA(Int): Codice che identifica univocamente l'autore; Nome(String): Nome dell'autore; Cognome(String): Cognome dell'autore; Nazionalita(String): Nazionalità dell'autore; DataNascita(Date): Data di nascita dell'autore.

3.2 Dizionario delle Relazioni

Relazioni	Descrizione	Attributi
GESTIONE	Associazione: 'una-a-molti' che mette in relazione le librerie con l'utente che le gestisce.	
RECENSIONE_F	Associazione: 'molti-a-molti' che mette in relazione i fascicoli con gli utenti che li hanno valutati, recensiti e/o inseriti nei preferiti.	Username (String): Username dell'utente che ha valutato, recensito e/o inserito il fascicolo nei preferiti; CodF (Int): Codice identificativo del fascicolo che è stato valutato, recensito, e/o inserito nei preferiti dall'utente; Testo (String): Testo di un'eventuale recensione scritta dall'utente; Valutazione (Int) Valore di un'eventuale valutazione fatta dall'utente; Preferito (Boolean): Indica se il fascicolo è stato inserito nei preferiti dall'utente.
RECENSIONE_S	Associazione: 'molti-a-molti' che mette in relazione le serie di libri con gli utenti che li hanno valutate, recensite e/o inserite nei preferiti (attivando le notifiche).	Username (String): Username dell'utente che ha valutato, recensito e/o inserito la serie nei preferiti; ISBN (String) Serie che è stata valutata, recensita, e/o inserita nei preferiti dall'utente; Testo (String) Testo di un'eventuale recensione scritta dall'utente; Valutazione (Int) Valore di un'eventuale valutazione fatta dall'utente; Preferito (Boolean): Indica se l'utente ha inserito tra i preferiti la serie, e quindi vuole ricevere una notifica quando è disponibile presso una libreria.

Relazioni	Descrizione	Attributi
RECENSIONE_L	Associazione: 'molti-a-molti' che mette in relazione i libri con gli utenti che li hanno valutati, recensiti e/o inseriti nei preferiti.	Username (String): Username dell'utente che ha valutato, recensito e/o inserito il libro nei preferiti; ISBN (String) Libro che è stato valutato, recensito, e/o inserito nei preferiti dall'utente; Testo (String) Testo di un'eventuale recensione scritta dall'utente; Valutazione (Int) Valore di un'eventuale valutazione fatta dall'utente; Preferito (Boolean): Indica se il libro è stato inserito nei preferiti dall'utente.
POSSESSO_F	Associazione: 'molti-a-molti' che mette in relazione i fascicoli con le librerie che li possiedono	Fruizione (String): Modalità di fruizione (Cartaceo, Digitale o AudioLibro) in cui è disponibile il fascicolo; Quantità (Int): Numero di fascicoli disponibili.
POSSESSO_S	Associazione: 'molti-a-molti' che mette in relazione le serie con le librerie che le possiedono	Fruizione (String): Modalità di fruizione (Cartaceo, Digitale o AudioLibro) in cui è disponibile la serie; Quantità (Int): Numero di serie disponibili.
POSSESSO_L	Associazione: 'molti-a-molti' che mette in relazione i libri con le librerie che li possiedono	Fruizione (String): Modalità di fruizione (Cartaceo, Digitale o AudioLibro) in cui è disponibile il fascicolo; Quantità (Int): Numero di libri disponibili.

Relazioni	Descrizione	Attributi
NOTIFICA	Associazione: 'molti-a-molti' che contiene tutte le notifiche inviate	Lettura (Boolean): Indica se la notifica inviata è stata letta dall'utente che l'ha ricevuta; Libreria (Int): Codice identificativo della libreria che possiede tutta la serie; Testo (String): Testo della notifica inviata; DataInvio (Date): Data in cui è stata inviata la notifica, OraInvio (Orario): Ora in cui è stata inviata la notifica.
COMPOSIZIONE	Associazione: 'una-a-molti' che mette in relazione i fascicoli con la rivista che li contiene.	
INTRODUZIONE	Associazione: 'molti-a-molti' che mette in relazione gli articoli scientifici con i fascicoli in cui sono introdotti.	
ESPOSIZIONE	Associazione: 'molti-a-molti' che mette in relazione gli articoli scientifici con le conferenze in cui sono stati esposti.	
INSERIMENTO	Associazione: 'una-a-molti' che mette in relazione i libri con la serie in cui sono stati inseriti.	
PARTECIPAZIONE	Associazione: 'una-a-molti' che mette in relazione le presentazioni con il libro che è stato presentato.	
APPARTENENZA	Associazione: 'molti-a-molti' che mette in relazione i libri con le collane nelle quali sono stati inseriti.	

Relazioni	Descrizione	Attributi
SCRITTURA_L	Associazione: 'molti-a-molti' che mette in relazione gli autori con i libri che hanno scritto.	
SCRITTURA_A	Associazione: 'molti-a-molti' che mette in relazione gli autori con gli articoli scientifici che hanno scritto.	

3.3 Dizionario dei Vincoli

In questa sezione sono descritti tutti i vincoli necessari per garantire l'integrità semantica della base di dati.

- **ASSERTION A1:** Garantisce che non esistano librerie con un gestore che non ha una partita IVA, evitando che vengano registrate librerie inesistenti.

```
CREATE ASSERTION A1
CHECK(
    NOT EXISTS(
        SELECT *
        FROM UTENTE AS U JOIN LIBRERIA AS L ON U.Username=L.Gestore
        WHERE U.PartitaIVA IS NULL
    )
);
```

Questo vincolo sarà imposto dai seguenti trigger:

- T_inserimentoLibreria
- T_modificaGestore
- T_chiusura_partitaIVA

- **ASSERTION A2:** Garantisce che non esistano articoli scientifici che sono stati pubblicati dopo i fascicoli in cui sono stati inseriti, evitando che vengano inseriti in dei fascicoli degli articoli scientifici quando questi ultimi non sono ancora stati pubblicati.

```
CREATE ASSERTION A2
CHECK(
    NOT EXISTS(
        SELECT *
        FROM ((ARTICOLO_SCIENTIFICO AS AR JOIN INTRODUZIONE AS I ON
               AR.DOI=I.DOI) JOIN FASCICOLO AS F ON F.CodF=I.CodF)
        WHERE AR.AnnoPubblicazione>EXTRACT(YEAR FROM F.DataPubblicazione)
    )
);
```

Questo vincolo sarà imposto dai seguenti trigger:

- T_inserimentoIntroduzione
- T_modificaIntroduzione
- T_modifica_pubblicazioneFascicolo

- **ASSERTION A3:** Garantisce che non esistano dei fascicoli che sono stati pubblicati prima delle loro riviste, evitando che vengano inseriti dei fascicoli in delle riviste non ancora pubblicate.

```
CREATE ASSERTION A3
CHECK(
    NOT EXISTS(
        SELECT *
        FROM FASCICOLO AS F JOIN RIVISTA AS R ON F.ISSN=R.ISSN
        WHERE EXTRACT(YEAR FROM F.DataPubblicazione)<R.AnnoPubblicazione
    )
);
```

Questo vincolo sarà imposto dai seguenti trigger:

- T_inserimentoFascicolo
- T_modificaFascicolo
- T_modificaRivista

- **CONSTRAINT C1:** Garantisce che non vengano registrate conferenze con la data di inizio successiva a quella di fine.

```
ALTER TABLE CONFERENZA
ADD CONSTRAINT C1
CHECK(
    DataInizio <= DataFine
);
```

- **CONSTRAINT C2:** Viene utilizzato dal dominio 'isbn' garantendo che gli ISBN vengano scritti nel formato giusto.

```
CONSTRAINT C2
CHECK (VALUE LIKE '978-%-%-_');
```

- **ASSERTION A4:** Garantisce che non esistano dei libri inseriti in una serie con l'ordine maggiore del numero di libri che contiene la serie in cui sono stati introdotti, evitando che vengano inseriti dei libri in una serie quando quest'ultima è stata già completata.

```

CREATE ASSERTION A4
CHECK(
    NOT EXISTS(
        SELECT *
        FROM INSERIMENTO AS I JOIN SERIE AS S on I.Serie=S.isbn
        WHERE S.NLibri<(
            SELECT COUNT(Libro)
            FROM INSERIMENTO
            WHERE Serie=S.isbn
        )
    )
);

```

Questo vincolo sarà imposto dal seguente trigger:

- T_inserimentoLibroSerie

- **CONSTRAINT C3:** Viene utilizzato dal dominio 'fruizione' garantendo che la modalità di fruizione degli elementi venduti dalle librerie sia 'Cartaceo', 'Digitale' o 'AudioLibro'.

```

CONSTRAINT C3
CHECK (VALUE IN ('Cartaceo', 'Digitale', 'AudioLibro'));

```

- **CONSTRAINT C4, C5 e C6:** Garantiscono che le quantità di elementi disponibili (rispettivamente fascicoli, serie e libri) presso qualche libreria abbiano come valore NULL solo quando sono disponibili in formato Digitale o AudioLibro.

```

ALTER TABLE POSSESSO_F
ADD CONSTRAINT C4
CHECK(NOT(Quantita IS NULL AND Fruizione='Cartaceo'));

ALTER TABLE POSSESSO_S
ADD CONSTRAINT C5
CHECK(NOT(Quantita IS NULL AND Fruizione='Cartaceo'));

ALTER TABLE POSSESSO_L
ADD CONSTRAINT C6
CHECK(NOT(Quantita IS NULL AND Fruizione='Cartaceo'));

```

- **CONSTRAINT C7, C8 e C9:** Garantiscono che le valutazioni agli elementi che possono essere posseduti dalle librerie (rispettivamente fascicoli, serie e libri) abbiano un valore compreso in [1,5].

```

ALTER TABLE RECENSIONE_F
ADD CONSTRAINT C7
CHECK(Valutazione>=0 AND Valutazione<=5);

ALTER TABLE RECENSIONE_S
ADD CONSTRAINT C8
CHECK(Valutazione>=0 AND Valutazione<=5);

ALTER TABLE RECENSIONE_L
ADD CONSTRAINT C9
CHECK(Valutazione>=0 AND Valutazione<=5);

```

- **CONSTRAINT C10:** Viene utilizzato dal dominio 'issn' garantendo che gli ISSN vengano scritti nel formato giusto.

```

CONSTRAINT C10
CHECK (VALUE LIKE '---- - ----');

```

- **CONSTRAINT C10:** Viene utilizzato dal dominio 'doi' garantendo che i DOI vengano scritti nel formato giusto.

```

CONSTRAINT C11
CHECK (VALUE LIKE '10-%');

```

- **CONSTRAINT C12:** Garantisce che nella relazione 'RECENSIONE_F' non ci siano dei fascicoli associati ad un utente che non l'ha valutato, recensito o inserito nei preferiti.

```

ALTER TABLE RECENSIONE_F
ADD CONSTRAINT C12
CHECK(NOT(Testo IS NULL AND Valutazione IS NULL AND Preferito=false));

```

- **CONSTRAINT C13:** Garantisce che nella relazione 'RECENSIONE_S' non ci siano delle serie associate ad un utente che non l'ha valutata, recensita o inserita nei preferiti, e quindi non ha attivato le notifiche.

```

ALTER TABLE RECENSIONE_S
ADD CONSTRAINT C13
CHECK(NOT(Testo IS NULL AND Valutazione IS NULL AND Preferito=false));

```

- **CONSTRAINT C14:** Garantisce che nella relazione 'RECENSIONE_L' non ci siano dei libri associati ad un utente che non l'ha valutato, recensito o inserito nei preferiti.

```

ALTER TABLE RECENSIONE_L
ADD CONSTRAINT C14
CHECK(NOT(Testo IS NULL AND Valutazione IS NULL AND Preferito=false));

```

- **CONSTRAINT C15:** Viene utilizzato dal dominio 'partitaiva' garantendo che le partite IVA dei gestori delle librerie vengano scritte nel formato giusto.

```

CONSTRAINT C15
CHECK (VALUE LIKE '_____');

```

- **CONSTRAINT C16:** Garantisce che non vengano registrate delle librerie che non abbiano nè un sito web e nè un indirizzo.

```

ALTER TABLE LIBRERIA
ADD CONSTRAINT C16
CHECK(NOT(SitoWeb IS NULL AND Indirizzo IS NULL));

```

- **ASSERTION A5:** Garantisce che non esistano delle conferenze che iniziano prima della pubblicazione degli articoli scientifici esposti, evitando che vengano esposti in delle conferenze degli articoli che non sono ancora stati pubblicati.

```

CREATE ASSERTION A5
CHECK(
    NOT EXISTS(
        SELECT *
        FROM ((CONFERENZA AS CO NATURAL JOIN ESPOSIZIONE AS E)
              NATURAL JOIN ARTICOLO_SCIENTIFICO AS AR)
        WHERE EXTRACT(YEAR FROM CO.DataInizio)<AR.AnnoPubblicazione
    )
);

```

Questo vincolo sarà imposto dai seguenti trigger:

- T_inserimentoEsposizione
 - T_modificaEsposizione
 - T_modifica_pubblicazioneArticolo
 - T_modifica_inizioConferenza
- **ASSERTION A6:** Garantisce che non esistano delle presentazioni che iniziano prima della pubblicazione del libro presentato, evitando che vengano presentati dei libri che non sono ancora stati pubblicati.

```

CREATE ASSERTION A6
CHECK(
    NOT EXISTS(
        SELECT *
        FROM LIBRO AS L NATURAL JOIN PRESENTAZIONE AS P
        WHERE P.DataP < L.DataPubblicazione
    )
);

```

Questo vincolo sarà imposto dai seguenti trigger:

- T_inserimentoPresentazione
- T_modificaPresentazione
- T_modificaLibro

- **ASSERTION A7:** Garantisce che nell'associazione, che mette in relazione tutte le librerie con tutte le serie che possiedono, ovvero 'POSSESSO_S', avviene un inserimento solo quando una libreria possiede tutti i libri che sono inseriti in una serie nella stessa modalità di fruizione. Inoltre garantisce che la modalità di fruizione in 'POSSESSO_S' sia la stessa di quella dei libri posseduti dalla libreria (cioè presenti in 'POSSESSO_L') e che la quantità disponibile di una serie sia uguale al minor numero di libri della serie disponibile nella libreria, solo quando la modalità di fruizione è cartacea.

```

CREATE ASSERTION A7
CHECK(
    NOT EXISTS(
        SELECT COUNT(*),
        FROM (((POSSESSO_S AS PS JOIN SERIE AS S ON PS.ISBN = S.ISBN)
            JOIN INSERIMENTO AS I ON S.ISBN = I.Serie) JOIN
            LIBRO AS L ON L.ISBN = I.Libro) JOIN POSSESSO_L AS PL ON
            PL.ISBN=L.ISBN )
        WHERE ((PL.Quantita>0 AND PL.Fruizione='Cartaceo') OR
            PL.Quantita IS NULL) AND PL.CodL = PS.CodL
        GROUP BY PL.CodL, PL.Fruizione, S.NLibri, PS.Fruizione,
            PS.Quantita, S.ISBN
        HAVING COUNT(*)<>S.NLibri OR PL.Fruizione<>PS.Fruizione OR
            (PS.Quantita <>(SELECT MIN(Quantita)
                FROM POSSESSO_L AS PL
                WHERE PL.Fruizione='Cartaceo' AND ISBN IN(
                    SELECT Libro
                    FROM INSERIMENTO AS I
                    WHERE I.Serie=S.ISBN
                )
            ) AND PS.Quantita IS NOT NULL)
    )
);

```

Questo vincolo sarà imposto dai seguenti trigger:

- T_inserimentoPossesso_L
- T_modificaPossesso_L
- T_eliminazionePossesso_L

4 Schema Logico e descrizioni

4.1 Schema Logico

- **UTENTE**(Username, Email, PasswordU, PartitaIVA, Nome, Cognome)
- **LIBRERIA**(CodL, NumeroTelefonico, SitoWeb, Nome, Indirizzo, Gestore)
 - LIBRERIA.Gestore → UTENTE.Username
- **RIVISTA**(ISSN, Editore, Argomento, CognomeR, NomeR, Titolo, AnnoPubblicazione)
- **ARTICOLO_SCIENTIFICO**(DOI, Titolo, AnnoPubblicazione)
- **FASCICOLO**(CodF, Numero, Editore, Titolo, DataPubblicazione, ISSN)
 - FASCICOLO.ISSN→RIVISTA.ISSN
- **INTRODUZIONE**(CodF, DOI)
 - INTRODUZIONE.CodF→FASCICOLO.CodF
 - INTRODUZIONE.DOI→ARTICOLO_SCIENTIFICO.DOI
- **CONFERENZA**(CodC, Luogo, StrutturaOrganizzatrice, DataInizio, DataFine)
- **ESPOSIZIONE**(DOI, CodC)
 - ESPOSIZIONE.DOI→ARTICOLO_SCIENTIFICO.DOI
 - ESPOSIZIONE.CodC→CONFERENZA.CodC
- **SERIE**(ISBN, Titolo, ISSN, DataPubblicazione, NLibri)
- **LIBRO**(ISBN, Titolo, Genere, Lingua, Editore, DataPubblicazione)
- **INSERIMENTO**(Libro, Serie)
 - INSERIMENTO.Libro→LIBRO.ISBN
 - INSERIMENTO.Serie→SERIE.ISBN
- **POSSESSO_F**(CodL, CodF, Fruizione, Quantità)
 - POSSESSO_F.CodL→Libreria.CodL
 - POSSESSO_F.CodF→FASCICOLO.CodF
- **RECENSIONE_F**(Username, CodF, Testo, Valutazione, Preferito)
 - RECENSIONE_F.Username→UTENTE.Username
 - RECENSIONE_F.CodF→FASCICOLO.CodF
- **POSSESSO_S**(CodL, ISBN, Fruizione, Quantità)
 - POSSESSO_S.CodL→Libreria.CodL

- POSSESSO_S.ISBN→SERIE.ISBN
- RECENSIONE_S(Username, ISBN, Testo, Valutazione, Preferito)
 - PREFERITI.S.Username→UTENTE.Username
 - PREFERITI.S.ISBN→SERIE.ISBN
- POSSESSO_L(CodL, ISBN, Fruizione, Quantità)
 - POSSESSO_L.Username→UTENTE.Username
 - POSSESSO_L.CodL→LIBRERIA.CodL
- RECENSIONE_L(Username, ISBN, Testo, Valutazione, Preferito)
 - PREFERITI.L.Username→UTENTE.Username
 - PREFERITI.L.ISBN→LIBRO.ISBN
- NOTIFICA(Username, ISBN, Libreria, OraInvio, DataInvio, Lettura, Testo)
 - NOTIFICA.Username→UTENTE.Username
 - NOTIFICA.ISBN→SERIE.ISBN
- PRESENTAZIONE(CodP, Luogo, Struttura, DataP, Ora, ISBN)
 - PRESENTAZIONE.ISBN→LIBRO.ISBN
- COLLANA(CodC, Nome, ISSN)
- APPARTENENZA(ISBN, CodC, Caretteristica)
 - APPARTENENZA.ISBN→LIBRO.ISBN
 - APPARTENENZA.CodC→COLLANA.CodC
- AUTORE(CodA, Nome, Cognome, Nazionalità, DataNascita)
- SCRITTURA_A(DOI, CodA)
 - SCRITTURA_A.DOI→ARTICOLO_SCIENTIFICO.DOI
 - SCRITTURA_A.CodA→AUTORE.CodA
- SCRITTURA_L(ISBN, CodA)
 - SCRITTURA_L.ISBN→LIBRO.ISBN
 - SCRITTURA_L.CodA→AUTORE.CodA

4.2 Descrizione di Trigger

Di seguito sono elencati tutti i Trigger utilizzati per la relizzazione della base di dati del sistema.

- **T_inserimentoLibreria:** il trigger si attiva dopo l'inserimento di una libreria e utilizza la funzione 'controllo_inserimentoLibreria' per garantire che i gestori delle librerie abbiano una partita IVA, e quindi che il vincolo dell'assertion A1 non sia violato.
- **T_modificaGestore:** il trigger si attiva dopo un aggiornamento del campo 'Gestore' della tabella 'Libreria' e utilizza la funzione 'controllo_modificaLibreria' per garantire che il nuovo gestore della libreria abbia una partita IVA, e quindi che il vincolo dell'assertion A1 non sia violato.
- **T_chiusura_partitaIVA:** il trigger si attiva quando il valore del campo 'partitaIVA' della tabella utente viene messo a NULL e utilizza la funzione 'chiusuraLibreria' per garantire che la libreria gestita dall'utente che ha chiuso la partita IVA venga eliminata, e quindi che il vincolo dell'assertion A1 non sia violato.
- **T_inserimentoIntroduzione:** il trigger si attiva dopo un inserimento nella tabella 'Introduzione' e utilizza la funzione 'controllo_introduzioneArticolo' per garantire che in ogni fascicolo non ci siano degli articoli scientifici pubblicati dopo al fascicolo, e quindi che il vincolo dell'assertion A2 non sia violato.
- **T_modificaIntroduzione:** il trigger si attiva dopo un aggiornamento nella tabella 'Introduzione' e utilizza la funzione 'controllo_modificaIntroduzione' per garantire che nella tupla aggiornata non ci sia un articolo scientifico che è stato pubblicato dopo al fascicolo nel quale è stato introdotto, e quindi che il vincolo dell'assertion A2 non sia violato.
- **T_modifica_pubblicazioneFascicolo:** il trigger si attiva quando la data di pubblicazione di un fascicolo viene aggiornata con una data precedente a quella presente prima della modifica e utilizza la funzione 'controllo_modifica_DataPubblicazioneFascicolo' per garantire che la nuova data del fascicolo non sia precedente a quella degli articoli che contiene il fascicolo modificato, e quindi che il vincolo dell'assertion A2 non sia violato.
- **T_inserimentoFascicolo:** il trigger si attiva dopo l'inserimento di un fascicolo e utilizza la funzione 'controllo_inserimentoFascicolo' per garantire che il fascicolo appena inserito faccia parte di una rivista pubblicata prima del fascicolo, e quindi che il vincolo dell'assertion A3 non sia violato.
- **T_modificaFascicolo:** il trigger si attiva dopo un aggiornamento del campo 'DataPubblicazione' e/o 'ISSN' della tabella 'Fascicolo' e utilizza la funzione 'controllo_modificaFascicolo' garantendo che il fascicolo sia stato pubblicato dopo la rivista di cui fa parte, e quindi che il vincolo dell'assertion A3 non sia violato.
- **T_modificaRivista:** il trigger si attiva quando l'anno di pubblicazione di una rivista viene aggiornata con una data più recente e utilizza la funzione 'controllo_modificaRivista' per garantire che la nuova data della rivista non sia successiva a quella dei fascicoli che contiene, e quindi che il vincolo dell'assertion A3 non sia violato.

- **T_inserimentoLibroSerie:** il trigger si attiva dopo un inserimento nella tabella 'Inserimento' e utilizza la funzione 'inserimento.LibroSerie' per garantire che in ogni serie non ci sia un numero di libri maggiore di quello indicato nel campo 'NLibri' della tabella 'Serie' e quindi che i vincoli dell'assertion A4 non sia violato.
- **T_inserimentoEsposizione:** il trigger si attiva dopo un inserimento nella tabella 'Esposizione' e utilizza la funzione 'controllo_inserimentoEsposizione' per garantire che nella tupla inserita non ci sia una conferenza di un articolo scientifico che è stato pubblicato dopo la data di inizio della conferenza, e quindi che il vincolo dell'assertion A5 non sia violato.
- **T_modificaEsposizione:** il trigger si attiva dopo un aggiornamento nella tabella 'Esposizione' e utilizza la funzione 'controllo_modificaEsposizione' per garantire che nella tupla aggiornata non ci sia un articolo scientifico che è stato pubblicato dopo l'inizio della conferenza in cui è stato esposto, e quindi che il vincolo dell'assertion A5 non sia violato.
- **T_modifica_pubblicazioneArticolo:** il trigger si attiva quando la data di pubblicazione di un articolo scientifico viene aggiornata con una data più recente e utilizza la funzione 'controllo_modificaArticolo' per garantire che la nuova data aggiornata non sia successiva a quella delle conferenze in cui è stato esposto l'articolo modificato, e quindi che il vincolo dell'assertion A5 non sia violato.
- **T_modifica_inizioConferenza:** il trigger si attiva quando la data di inizio di una conferenza viene aggiornata con una data precedente a quella presente prima della modifica e utilizza la funzione 'controllo_modificaConferenza' per garantire che la nuova data di inizio non sia precedente a quella delle pubblicazioni degli articoli esposti nella conferenza modificata, e quindi che il vincolo dell'assertion A5 non sia violato.
- **T_inserimentoPresentazione:** il trigger si attiva dopo l'inserimento di una presentazione e utilizza la funzione 'controllo_inserimentoPresentazione' per garantire che alle presentazioni vengano esposti dei libri già pubblicati, e quindi che il vincolo dell'assertion A6 non sia violato.
- **T_modificaPresentazione:** il trigger si attiva dopo un aggiornamento del campo 'DataP' e/o 'ISBN' della tabella 'Presentazione' e quando la data viene aggiornata con un'altra che non è più recente di quella presente prima della modifica, il trigger utilizza la funzione 'controllo_modificaPresentazione' per garantire che la data della conferenze non sia precedente a quella della pubblicazione del libro presentato, e quindi che il vincolo dell'assertion A6 non sia violato.
- **T_modificaLibro:** il trigger si attiva quando la data di pubblicazione di un libro viene aggiornata con un'altra più recente e utilizza la funzione 'controllo.Libro' per garantire che la nuova data non sia successiva a quelle delle presentazioni in cui è stato esposto il libro modificato, e quindi che il vincolo dell'assertion A6 non sia violato.
- **T_inserimentoPossesso_L:** il trigger si attiva dopo un'inserimento nella tabella 'Possesso_L', e quindi una librerie inizia a possedere un libro, il trigger utilizza la funzione 'controllo_inserimentoPossesso_L' per far sì che se il libro inserito appartiene a una serie, allora avviene un inserimento nella tabella 'Possesso_S' con la librerie, serie, modalità di fruizione e quantità giuste, e quindi garantisce che il vincolo dell'assertion A7 non sia violato.

- **T_modificaPossesso_L:** il trigger si attiva dopo l'aggiornamento del campo 'Quantita' della tabella 'Possesso_L' e utilizza la funzione 'controllo_modificaPossesso_L' per far sì che se il libro modificato appartiene alla serie e la quantità aggiornata è la minore fra quelle dei libri della serie in cui è stato inserito il libro modificato, allora tale aggiornamento avviene anche nella tabella 'Possesso_S', e quindi garantisce che il vincolo dell'assertion A7 non sia violato.
- **T_eliminazionePossesso_L:** il trigger si attiva dopo un'eliminazione nella tabella 'Possesso_L', quindi una libreria non possiede più un libro, il trigger utilizza la funzione 'controllo_eliminazionePossesso_L', per far sì che se il libro eliminato appartiene a una serie, allora viene eliminata anche la tupla nella tabella 'Possesso_S' in cui sono in relazione la libreria e la serie del libro eliminato, e quindi garantisce che il vincolo dell'assertion A7 non sia violato.
- **T_inserimentoArticolo:** il trigger si attiva dopo l'inserimento di un articolo scientifico e utilizza la funzione 'inserimento_DOIArticolo' per garantire l'univocità dei DOI degli articoli scientifici.
- **T_inserimentoPossesso_S:** il trigger si attiva quando nella tabella 'Possesso_S' viene inserita una tupla con il valore del campo 'Quantita' maggiore di 0, il trigger utilizza la funzione 'invia_notifica_possesso_S' per garantire che tutti gli utenti che hanno attivato le notifiche per la serie inserita, ricevano una notifica, e quindi avviene un inserimento nella tabella 'Notifica' con: gli utenti che hanno la serie inserita nei preferiti; la serie inserita e la libreria che la possiede.
- **T_modificaPossesso_S:** il trigger si attiva quando nella tabella 'Possesso_S' viene aggiornato il valore del campo 'Quantita' da 0 a un valore maggiore di 0, il trigger utilizza la funzione 'invia_notifica_possesso_S' per garantire che tutti gli utenti che hanno attivato le notifiche per la serie nella tupla modificata, ricevano una notifica, e quindi avviene un inserimento nella tabella 'Notifica' con: gli utenti che hanno la serie inserita nei preferiti; la serie inserita e la libreria che la possiede.
- **T_inserimentoPreferiti_S:** il trigger si attiva quando viene inserita una tupla con il campo 'Preferito' a "true" nella tabella 'Recensione_S', e quindi un utente ha attivato le notifiche per la serie inserita nella tupla, il trigger utilizza la funzione 'invia_notifica_preferiti_S' per garantire che l'utente ha attivato le notifiche per una certa serie, riceva una notifica per ogni libreria che possiede la serie, e quindi per ognuna di queste librerie avviene un inserimento nella tabella 'Notifica' con: l'utente che ha attivato le notifiche; la serie inserita dall'utente nei preferiti e una libreria che la possiede.
- **T_modificaPreferiti_S:** il trigger si attiva quando viene aggiornato a "true" il valore del campo 'Preferito' nella tabella 'Recensione_S', e quindi un utente ha attivato le notifiche per la serie inserita nella tupla modificata, il trigger utilizza la funzione 'invia_notifica_preferiti_S' per garantire che l'utente ha attivato le notifiche per una certa serie, riceva una notifica per ogni libreria che possiede la serie, e quindi per ognuna di queste librerie avviene un inserimento nella tabella 'Notifica' con: l'utente che ha attivato le notifiche; la serie inserita dall'utente nei preferiti e una libreria che la possiede.

4.3 Descrizioni di Funzioni e Procedure

Di seguito sono elencati tutte le funzioni e le procedure utilizzate per la relizzazione della base di dati del sistema.

- **controllo_inserimentoLibreria:**

la funzione viene richiamata dal trigger 'T_inserimentoLibreria' e se viene inserita una nuova libreria con un gestore che non ha una partitaIVA, allora la funzione elimina questa libreria.

- **controllo_modificaLibreria:**

la funzione viene richiamata dal trigger 'T_modificaGestore' e se viene aggiornato il gestore di una libreria con un altro che non possiede una partita IVA, allora la funzione annulla l'aggiornamento, inserendo di nuovo il vecchio gestore nella tupla modificata.

- **chiusuraLibreria:**

la funzione viene richiamata dal trigger 'T_chiusura_partitaIVA' quando la partita IVA di un utente viene messa a "NULL", quindi la funzione elimina tutte le librerie dell'utente che ha chiuso la partita IVA.

- **controllo_introduzioneArticolo:**

la funzione viene richiamata dal trigger 'T_inserimentoIntroduzione' e se viene inserita nella tabella 'Introduzione' una tupla in cui l'anno di pubblicazione dell'articolo scientifico è più recente di quello del fascicolo, allora la funzione elimina questa tupla.

- **controllo_modificaIntroduzione:**

la funzione viene richiamata dal trigger 'T_modificaIntroduzione' e se viene aggiornata nella tabella 'Introduzione' una tupla in cui l'anno di pubblicazione dell'articolo scientifico è più recente di quello del fascicolo, allora la funzione annulla la modifica, riportando i valori dei campi della tupla aggiornata a quelli precedenti alla modifica.

- **controllo_modifica_DataPubblicazioneFascicolo:**

la funzione viene richiamata dal trigger 'T_modifica_pubblicazioneFascicolo' quando la data di pubblicazione di un fascicolo viene aggiornata con un'altra precedente a quella presente prima della modifica, quindi se il nuovo anno di pubblicazione del fascicolo aggiornato è precedente a quello di un articolo inserito in tale fascicolo, allora la funzione annulla l'aggiornamento, inserendo di nuovo la vecchia data di pubblicazione nella tupla aggiornata.

- **controllo_inserimentoFascicolo:**

la funzione viene richiamata dal trigger 'T_inserimentoFascicolo' e se viene inserito un nuovo fascicolo con una data pubblicazione di cui l'anno è precedente a quello della sua rivista, allora la funzione elimina questo fascicolo.

- **controllo_modificaFascicolo:**

la funzione viene richiamata dal trigger 'T_modificaFascicolo' quando la rivista di un fascicolo viene aggiornata oppure la data di pubblicazione viene aggiornata con un'altra più vecchia, quindi se nella tupla modificata la data di pubblicazione del fascicolo ha un anno precedente a quello della sua rivista, allora la funzione annulla l'aggiornamento, riportando i valori dei campi 'DataPubblicazione' e 'ISSN' della tabella 'Fascicolo' ai valori precedenti alla modifica della tupla.

- **controllo_modificaRivista:**
la funzione viene richiamata dal trigger 'T_modificaRivista' quando l'anno di pubblicazione di una rivista viene aggiornato con un'altro più recente, quindi se il nuovo anno di pubblicazione è più recente di quello di qualche fascicolo contenuto dalla rivista modificata, allora la funzione annulla l'aggiornamento, riportando il valore dell'anno di pubblicazione della rivista a quello precedente alla modifica.
- **inserimento_LibroSerie:**
la funzione viene richiamata dal trigger 'T_inserimentoLibroSerie' e se nella tabella 'Inserimento' sono stati inseriti tutti i libri della serie (e quindi la serie è già stata completata), allora la funzione elimina la tupla inserita.
- **controllo_inserimentoEsposizione:**
la funzione viene richiamata dal trigger 'T_inserimentoEsposizione' e se viene inserita una tupla nella tabella 'Esposizione' in cui l'articolo scientifico ha un anno di pubblicazione successivo a quello dell'inizio della conferenza, allora la funzione elimina questa tupla.
- **controllo_modificaEsposizione:**
la funzione viene richiamata dal trigger 'T_modificaEsposizione' e se viene aggiornata una tupla nella tabella 'Esposizione' in cui l'anno di pubblicazione dell'articolo scietifico è più recente di quello di inizio della conferenza, allora la funzione annulla la modifica, riportando i valori dei campi della tupla aggiornata a quelli precedenti alla modifica.
- **controllo_modificaArticolo:**
la funzione viene richiamata dal trigger 'T_modifica_pubblicazioneArticolo' quando l'anno di pubblicazione di un articolo scientifico viene aggiornato con un'altro più recente, quindi se il nuovo anno di pubblicazione è più recente di quello di inizio di qualche conferenza in cui è stato esposto, allora la funzione annulla la modifica, riportando il valore dell'anno di pubblicazione dell'articolo a quello precedente all'aggiornamento.
- **controllo_modificaConferenza:**
la funzione viene richiamata dal trigger 'T_modifica_inizioConferenza' quando la data di inizio di una conferenza viene aggiornata con una data precedente a quella presente prima della modifica, quindi se l'anno della nuova data di inizio della conferenza è precedete a quello di qualche articolo esposto nella conferenza modificata, allora la funzione annulla l'aggiornamento, inserendo di nuovo la vecchia data di inizio della conferenza nella tupla modificata.
- **controllo_inserimentoPresentazione:**
la funzione viene richiamata dal trigger 'T_inserimentoPresentazione' e se viene inserita una presentazione con una data precedente a quella della pubblicazione del libro, allora la funzione elimina questa presentazione.
- **controllo_modificaPresentazione:**
la funzione viene richiamata dal trigger 'T_modificaPresentazione' quando l'ISBN del libro di una presentazione viene aggiornato oppure la data della presentazione viene aggiornata con un'altra più vecchia, quindi se nella tupla modificata la data della presentazione è precedente a quella della pubblicazione del libro, allora la funzione annulla l'aggiornamento riportando i valori dei campi 'DataP' e 'ISBN' della tabella 'Presentazione' ai valori precedenti alla modifica della tupla.

- **controllo_Libro:**

la funzione viene richiamata dal trigger 'T_modificaLibro' quando la data di pubblicazione di un libro viene aggiornata con un'altra più recente, quindi se la nuova data è successiva a quella di qualche presentazione del libro, allora la funzione annulla la modifica, inserendo di nuovo la vecchia data nella tupla aggiornata.

- **controllo_inserimentoPossesso_L:**

la funzione viene richiamata dal trigger 'T_inserimentoPossesso_L' e se dopo l'inserimento di una tupla nella tabella 'Possesso_L' una libreria inizia a possedere tutti i libri di una serie nello stesso formato, allora la funzione effettua un inserimento nella tabella 'Possesso_S' con:

- il codice della libreria che possiede la serie completa;
- l'ISBN della serie;
- la modalità di fruizione del libro inserito nella nuova tupla di 'Possesso_L';
- e la quantità disponibile con un valore pari a quello minore delle disponibilità dei libri della serie inserita solo quando la modalità di fruizione è 'Cartaceo'.

- **controllo_modificaPossesso_L:**

la funzione viene richiamata dal trigger 'T_modificaPossesso_L' quando la quantità disponibile di un libro presso una libreria viene modificata, e se questo libro appartiene a una serie disponibile presso la stessa libreria, allora la funzione aggiorna la quantità disponibile di tale serie in questa libreria con il numero minore di disponibilità dei libri della serie posseduti dalla libreria.

- **controllo_eliminazionePossesso_L:**

la funzione viene richiamata dal trigger 'T_eliminazionePossesso_L' e se dopo un'eliminazione di una tupla nella tabella 'Possesso_L' una libreria non possiede più tutti i libri di una serie, allora la funzione elimina la tupla della tabella 'Possesso_S' con:

- l'ISBN della serie;
- il codice della libreria che non possiede più il libro;
- la modalità di fruizione del libro eliminato;

- **inserimento_DOIArticolo:**

la funzione viene richiamata dal trigger 'T_inserimentoArticolo' e completa l'inserimento di un articolo scientifico aggiungendo il DOI che identifica univocamente l'articolo appena inserito.

- **invia_notifica:**

la procedura riceve come parametri di input l'username di un utente; l'ISBN di una serie; il codice di una libreria e una modalità di fruizione. La procedura utilizza questi parametri per inviare una notifica all'utente con l'username ricevuto, cioè inserisce nella tabella 'Notifica' una tupla con: l'username dell'utente; l'ISBN della serie; il codice della libreria che possiede la serie; la data e l'ora di invio della notifica e un testo con tutte le informazioni della libreria e della serie disponibile.

- **invia_notifica_possesso_S:**

la funzione viene richiamata dal trigger 'T_inserimentoPossesso_S', quando la quantità disponibile della serie inserita è maggiore di 0, e dal trigger 'T_modificaPossesso_S', quando la

quantità disponibile della serie viene aggiornata da 0 a un valore maggiore di 0, quindi la funzione invia a tutti gli utenti, che hanno attivato le notifiche per la serie inserita o aggiornata, una notifica utilizzando la procedura 'invia_notifica'.

- **invia_notifica_preferiti_S:**

la funzione viene richiamata dai trigger 'T_inserimentoPreferiti_S' e 'T_modificaPreferiti_S' quando nella tabella 'Recensione_S' viene inserita o aggiornata una tupla con il campo 'Preferito' a "true", quindi la funzione invia all'utente, che ha inserito una serie nei preferiti, una notifica per ogni libreria che possiede tale serie utilizzando la procedura 'invia_notifica'.