Laboratorio di Basi di Dati: *Progetto "Collectors"*

Gruppo di lavoro:

Matricola	Nome	Cognome	Contributo al progetto
278661	Luca Francesco	Macera	Diagramma ER, diagramma ER ristrutturato, modello relazionale, realizzazione del database, query 1 3 4 5 6, diversi trigger, implementazione interfaccia
279660	Michael	Piccirilli	Diagramma ER, diagramma ER ristrutturato, realizzazione del database, documentazione, query 10 11,trigger, implementazione interfaccia
278844	Calogero	Carlino	Diagramma ER, diagramma ER ristrutturato, modello relazionale, realizzazione del database, query 2 7 8 9 12, implementazione interfaccia

Data di consegna del progetto: 13/06/2023

Analisi dei requisiti

Il caso d'uso rappresenta una situazione in cui si vuole tenere traccia delle **collezioni** relative ad un **collezionista**. Tali collezioni rappresentano un gruppo di **dischi** , i quali presentano ognuno una tracklist formata di **tracce,** delle **immagini** e ulteriori informazioni quali, ad esempio, gli **autori**.

Dominio:

- **Collezionista**: persona identificata univocamente da un nickname e da un indirizzo email che crea delle collezioni che possono essere anche condivise con altri collezionisti.
- **Collezione:** gruppo di dischi che può, o meno, essere condivisa(visualizzabile) con altri collezionisti.
- **Disco**: rappresenta in tutto e per tutto il concetto di disco, con i suoi autori, un suo titolo, l'anno di uscita, l'etichetta, dei generi, lo stato di conservazione, il formato, il barcode (il quale potrebbe essere anche assente).
- **Immagine:** relativa ad un disco, può essere la copertina, il retro o ulteriori tipologie.
- **Etichetta**: rappresenta l'azienda che distribuisce e pubblicizza un disco, identificata da una partita IVA e da un nome
- **Traccia:** o canzone, presente all'interno del disco, con un suo titolo, la durata ed altre informazioni come gli autori e/o gli esecutori e/o i compositori
- **Autore:** rappresenta colui che compone e/o esegue una traccia o, per quanto riguarda il disco, colui che lo ha pubblicato

Assunzioni

Assumiamo che ogni disco fisico possegga un barcode univoco e diverso da tutti gli altri dischi fisici. Specificatamente:

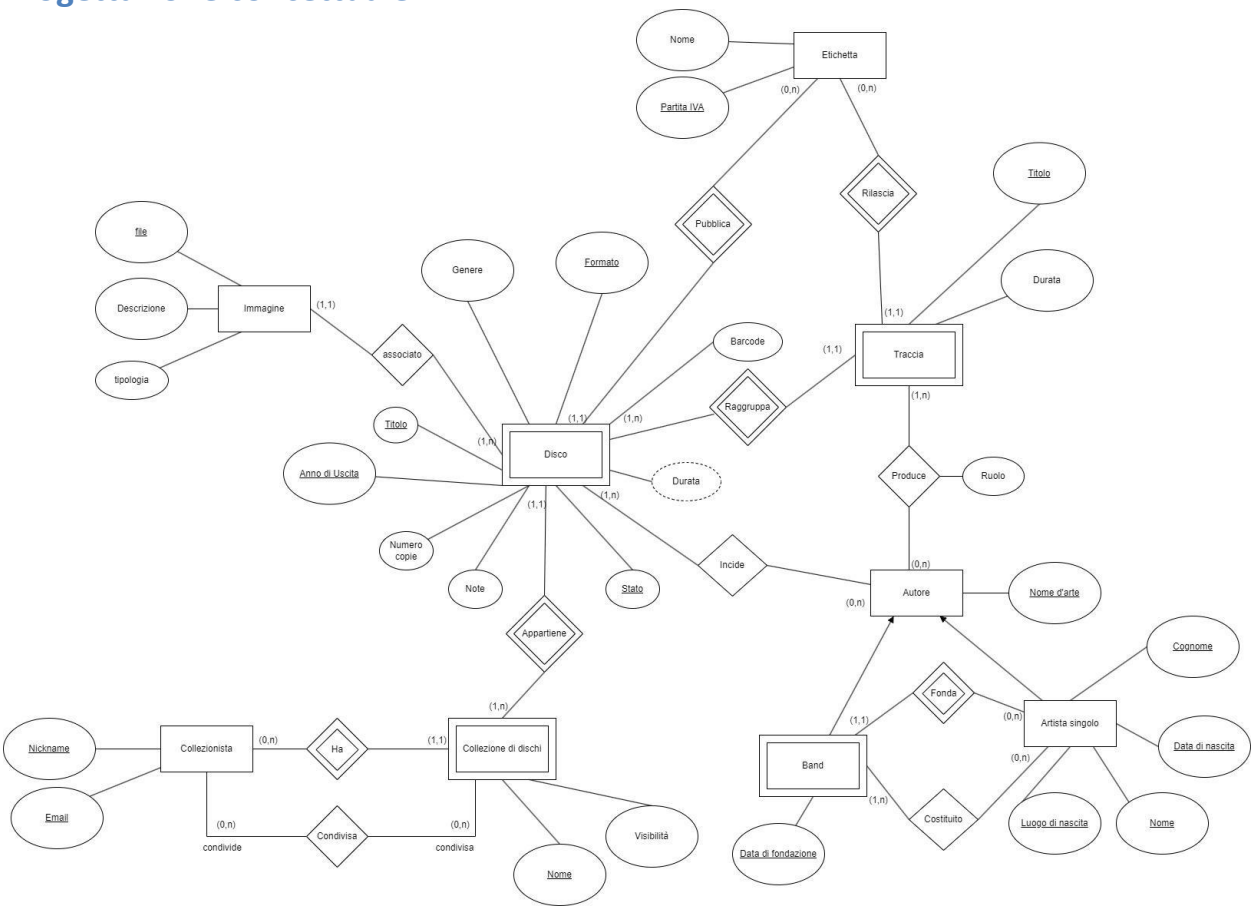
Due dischi che hanno stessi attributi quali titolo, anno di uscita, etichetta, generi, stato di conservazione ecc. hanno comunque barcode diverso:

Es.

Due copie fisiche dello stesso disco hanno barcode diverso

Titolo	**Data**	**Stato**	**Formato**	**Etichetta**	**Generi**	**Barcode**	**Collezione**
Aida	15-04-1993	Ottimo	Vinile	Sony Music	Cantautorato	783948200214	Successi mondiali
Aida	15-04-1993	Ottimo	Vinile	Sony Music	Cantautorato	784205093842	Grandi Hit

Progettazione concettuale



- **Genere** rappresenta un attributo multivalore dal momento che un disco può avere più di un genere.
- **Durata** (di un disco) è un attributo calcolato sulla base della somma delle durate di ogni traccia presente nel disco stesso.
- La **visibilità** rappresenta i due stati in cui può trovarsi una collezione:
    1. *privato* - ciò significa che una determinata collezione che appartiene ad un collezionista è visibile solo a lui o eventualmente anche agli altri collezionisti con cui è condivisa
    2. *pubblica* - ciò significa che la collezione è visibile a tutti i collezionisti

Formalizzazione dei vincoli non esprimibili nel modello ER

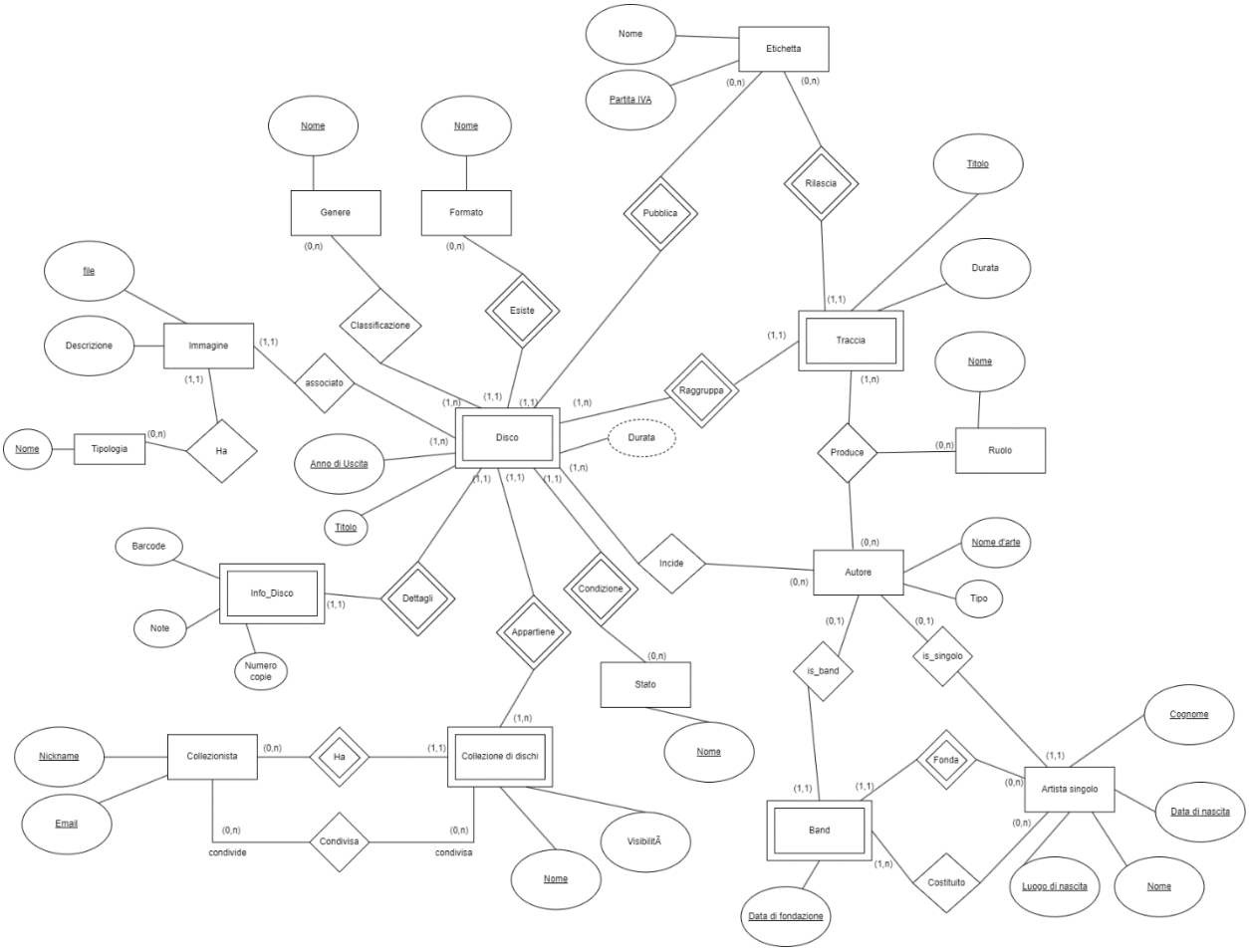
- **Collezionista:** *nickname* ed *email* sono chiavi primarie dunque devono essere **UNIQUE** e **NOT NULL.**
- **Collezioni di dischi:** *nome* e gli attributi della relazione *ha* (collezionista) sono chiave primaria dunque devono essere **UNIQUE** e **NOT NULL.** Il flag *visibilità* non può essere **NULL**, può avere solo *privata/pubblica (0 o 1)* come valori.
- **Disco:** *titolo*, *anno di uscita, formato, stato* e la relazione *appartiene*(collezione) e *pubblica* (etichetta) sono la chiave primaria dunque devono essere **UNIQUE** e **NOT**

NULL. Barcode è **UNIQUE** dal momento che deve essere univoco. Il numero di copie non può essere **NULL**(di **DEFAULT** è 1).

- **Tracce:** *titolo* e gli attributi della relazione *rilascia*(etichetta) e *raggruppa*(disco) sono la chiave primaria dunque devono essere **UNIQUE** e **NOT NULL.** *Durata* non può essere **NULL.**
- **Immagine:** *file* è chiave primaria dunque deve essere **UNIQUE** e **NOT NULL.** L'attributo *tipologia* non può essere **NULL.**
- **Etichetta:** *Partita_IVA* è chiave primaria dunque deve essere **UNIQUE** e **NOT NULL.**
- **Band:** *Data di fondazione* e gli attributi della relazione *fonda*(artista singolo) sono chiave primaria dunque devono essere **UNIQUE** e **NOT NULL.**
- **Artista Singolo:** *nome*, *cognome*, *luogo di nascita*, *data di nascita* sono chiave primaria dunque devono essere **UNIQUE** e **NOT NULL.**
- **Autore:** *nome d'arte* è chiave primaria, dunque deve essere **UNIQUE** e **NOT NULL.**

Progettazione logica

Ristrutturazione ed ottimizzazione del modello ER



Si è scelto di dividere l'entità Disco in due entità separate (**Disco** ed **Info_Disco**) per fare in modo che questa non fosse troppo carica di attributi. Inoltre, tale divisione è avvenuta in virtù di quegli attributi meno importanti come, appunto, le *note***,** il *barcode* e il *numero di copie*.

Le generalizzazioni presenti nel modello ER sono state eliminate in favore di due nuove relazioni: **is_band** e **is_singolo** tra **Autore**, **Band** e **Artista Singolo** per fare in modo di tenere separate le tre entità senza sovraccaricare le entità figlie o padre. Inoltre, si è scelto di aggiungere un attributo *tipo* ad **Autore** in modo da ricavare subito se i dati di un autore salvato riferiscono ad un artista singolo o ad una band. Questo potrebbe anche essere esteso nel caso in cui si volessero modellare altri tipi di autore oltre ai due citati precedentemente.

Sono state aggiunte nuove entità: **Genere, Formato, Tipologia, Stato, Ruolo.** Queste nuove entità servono per poter salvare dati selezionabili da liste preimpostate , come i generi di un disco, il formato in cui essi sono salvati o il ruolo di un autore nella produzione di un brano. L'uso di queste entità permette maggiore flessibilità della base di dati nel caso in cui fosse necessaria la manipolazione di questi dati.

Traduzione del modello ER nel modello relazionale

Collezionista(ID, nickname, email)

Collezione_di_dischi(ID , nome,visibilità, *ID_collezionista*)

Disco(ID, titolo, anno_di_uscita, nome_formato, nome_stato, *ID_etichetta*, *ID_collezione_di_dischi*)

Info_disco(*ID_disco*, barcode, note, numero_copie)

Etichetta(ID, nome, partita_IVA)

Genere(nome)

Formato(nome)

Ruolo(nome)

Stato(nome)

Tipologia(nome)

Immagine(ID, file, descrizione, nome_tipologia, *ID_disco*)

Traccia(ID, titolo, durata, *ID_etichetta*, *ID_disco*)

Autore(ID, nome_darte, tipo)

Artista_Singolo(ID, nome, cognome, data_di_nascita, luogo_di_nascita, *ID_autore*)

Band(ID, data_di_fondazione, *ID_fondatore*, *ID_autore*)
Condivisa(*ID_collezione_di_dischi*, *ID_collezionista*)

Classificazione(*ID_disco*, *nome_genere*)

Produce(*ID_traccia*, *ID_autore*, *nome_ruolo*)

Costituito(*ID_band*, *ID_membro*)

Incide(*ID_disco*, *ID_autore*)

*****Le primary key sono marcate in underline mentre le foreign key in *italic*

Progettazione fisica

Implementazione del modello relazionale

Script sql per la creazione della struttura della base di dati estratto dal file "DDL.sql"

```
#Inizializzazione database
drop database if exists collectors;
create database collectors;
use collectors;
#Tabelle
drop table if exists incide;
drop table if exists classificazione;
drop table if exists produce;
drop table if exists costituito;
drop table if exists condivisa;
drop table if exists band;
drop table if exists artista_singolo;
drop table if exists traccia;
drop table if exists immagine;
drop table if exists info_disco;
drop table if exists disco;
drop table if exists collezione_di_dischi;
drop table if exists collezionista;
drop table if exists autore;
drop table if exists etichetta;
drop table if exists tipologia;
drop table if exists stato;
drop table if exists ruolo;
drop table if exists formato;
drop table if exists genere;

create table genere(
nome varchar(50),
constraint NOME_genere primary key(nome)
);
create table formato(
nome varchar(50),
constraint NOME_formato primary key(nome)
);
create table ruolo(
nome varchar(50),
constraint NOME_ruolo primary key(nome)
);
create table stato(
nome varchar(50),
constraint NOME_stato primary key(nome)
);
create table tipologia(
nome varchar(50),
constraint NOME_tipologia primary key(nome)
```

```
);
create table etichetta(
id integer unsigned auto_increment,
nome varchar(200) not null,
partitaIVA varchar(11) not null,
constraint ID_etichetta primary key(id),
constraint unique_partitaIVA unique(partitaIVA)
);
create table autore(
id integer unsigned auto_increment,
nome_darte varchar(100) not null,
tipo boolean not null,
constraint ID_autore primary key(id),
constraint unique_autore unique(nome_darte)
);
create table collezionista(
id integer unsigned auto_increment,
nickname varchar(100) not null,
email varchar(200) not null,
constraint ID_collezionista primary key(id),
constraint unique_collezionista unique(nickname, email)
);
create table collezione_di_dischi(
id integer unsigned auto_increment,
nome varchar(200) not null,
visibilita boolean not null,
id_collezionista integer unsigned not null,
constraint ID_collezione primary key(id),
constraint collezione_collezionista foreign key(id_collezionista) references
collezionista(id) on delete cascade,
constraint unique_collezione unique(id_collezionista,nome)
);
create table disco(
id integer unsigned auto_increment,
titolo varchar(50) not null,
anno_di_uscita date not null,
nome_formato varchar(50) not null,
nome_stato varchar(50) not null,
id_etichetta integer unsigned not null,
id_collezione_di_dischi integer unsigned not null,
constraint ID_disco primary key(id),
constraint disco_formato foreign key(nome_formato) references formato(nome) on
delete restrict on update cascade,
constraint disco_etichetta foreign key(id_etichetta) references etichetta(id) on
delete restrict,
```

```
constraint disco_stato foreign key(nome_stato) references stato(nome) on delete
restrict on update cascade,
constraint disco_collezione foreign key(id_collezione_di_dischi) references
collezione_di_dischi(id) on delete cascade,
constraint unique_disco unique(titolo,anno_di_uscita,nome_formato,nome_stato,
id_etichetta,id_collezione_di_dischi)
);
create table info_disco(
id_disco integer unsigned,
barcode varchar(200),
note text,
numero_copie integer unsigned default 1,
constraint ID_info_disco primary key(id_disco),
constraint unique_barcode unique(barcode),
constraint info_disco_disco foreign key(id_disco) references disco(id) on delete
cascade
);
create table immagine(
id integer unsigned auto_increment,
path varchar(100) not null,
descrizione text,
id_disco integer unsigned not null,
nome_tipologia varchar(50) not null,
constraint ID_immagine primary key(id),
constraint unique_immagine unique(id_disco,path,nome_tipologia),
constraint immagine_disco foreign key(id_disco) references disco(id) on delete
cascade,
constraint immagine_tipologia foreign key(nome_tipologia) references
tipologia(nome) on delete restrict on update cascade
);
create table traccia(
id integer unsigned auto_increment,
titolo varchar(50) not null,
durata decimal(10,2) unsigned default 0.0,
id_etichetta integer unsigned not null,
id_disco integer unsigned not null,
constraint ID_traccia primary key(id),
constraint traccia_etichetta foreign key(id_etichetta) references etichetta(id)
on delete restrict,
constraint traccia_disco foreign key(id_disco) references disco(id) on delete
cascade,
constraint unique_traccia unique(titolo,id_etichetta,id_disco)
);
create table artista_singolo(
id integer unsigned auto_increment,
```

```
nome varchar(50) not null,
cognome varchar(50) not null,
data_nascita date not null,
luogo_nascita varchar(50) not null,
id_autore integer unsigned not null,
constraint ID_artista primary key(id),
constraint artista_singolo_autore foreign key(id_autore) references autore(id) on
delete cascade,
constraint unique_artista_singolo
unique(nome,cognome,data_nascita,luogo_nascita),
constraint unique_autore_artista_singolo unique(id_autore)
);
create table band(
id integer unsigned auto_increment,
id_fondatore integer unsigned not null,
data_fondazione date not null,
id_autore integer unsigned not null,
constraint ID_band primary key(id),
constraint band_artista_singolo foreign key(id_fondatore) references
artista_singolo(id) on delete restrict,
constraint band_autore foreign key(id_autore) references autore(id) on delete
cascade,
constraint unique_band unique(id_fondatore,data_fondazione),
constraint unique_autore_band unique(id_autore)
);
# Relazioni
create table condivisa(
id_collezionista integer unsigned not null,
id_collezione integer unsigned not null,
constraint primary_condivisione primary key(id_collezionista,id_collezione),
constraint condivisa_collezionista foreign key(id_collezionista) references
collezionista(id) on delete cascade,
constraint condivisa_collezione foreign key(id_collezione) references
collezione_di_dischi(id) on delete cascade
);
create table costituito(
 id_band integer unsigned not null,
 id_artista integer unsigned not null,
 constraint primary_costituito primary key(id_band,id_artista),
 constraint costituito_band foreign key(id_band) references band(id) on delete
cascade,
 constraint costituito_artista_singolo foreign key(id_artista) references
artista_singolo(id) on delete restrict
);
create table produce(
```

```
id_traccia integer unsigned not null,
id_autore integer unsigned not null,
nome_ruolo varchar(50) not null,
constraint primary_produce primary key(id_traccia,id_autore,nome_ruolo),
constraint produce_traccia foreign key(id_traccia) references traccia(id) on
delete cascade,
constraint produce_autore foreign key(id_autore) references autore(id) on delete
restrict,
constraint produce_ruolo foreign key(nome_ruolo) references ruolo(nome) on delete
restrict on update cascade
);
create table classificazione(
nome_genere varchar(50) not null,
id_disco integer unsigned not null,
constraint primary_classificazione primary key(nome_genere,id_disco),
constraint classificazione_genere foreign key(nome_genere) references
genere(nome) on delete restrict on update cascade,
constraint classificazione_disco foreign key(id_disco) references disco(id) on
delete cascade
);
create table incide(
id_disco integer unsigned not null,
id_autore integer unsigned not null,
constraint primary_raggruppamento primary key(id_disco,id_autore),
constraint incide_disco foreign key(id_disco) references disco(id) on delete
cascade,
constraint incide_autore foreign key(id_autore) references autore(id) on delete
restrict
);
```

È presente anche uno script per il popolamento delle tabelle chiamato "data.sql"

Implementazione dei vincoli

Trigger che controlla l'inserimento di un artista singolo.

```
DROP TRIGGER IF EXISTS check_artista;

DELIMITER $

CREATE TRIGGER check_artista BEFORE INSERT ON artista_singolo FOR EACH ROW
BEGIN

  DECLARE tipo_autore boolean;

  SET tipo_autore = (
      SELECT tipo
      FROM autore
      WHERE id = new.id_autore);

      /* Controlla che il tipo dichiarato nell'inserimento dell'autore combaci
      con quello dell'artista che si sta inserendo (artista singolo o band)*/
      IF(tipo_autore = true) THEN
            SIGNAL SQLSTATE "45000" SET MESSAGE_TEXT="Il tipo dell'autore non
            combacia.";
      END IF;

      /* Controllo sulla data di nascita affinchè non ecceda quella corrente*/
      IF(new.data_nascita > date(now())) THEN
        SIGNAL SQLSTATE "45000" SET MESSAGE_TEXT="Errore nella data di nascita.";
      END IF;
END$
```

Trigger che controlla l'inserimento di una band.

```
DROP TRIGGER IF EXISTS check_band;

DELIMITER $

CREATE TRIGGER check_band BEFORE INSERT ON band FOR EACH ROW
BEGIN

  DECLARE tipo_autore boolean;

  SET tipo_autore = (
      SELECT tipo
      FROM autore
      WHERE id = new.id_autore);

  /* Controlla che il tipo dichiarato nell'inserimento dell'autore combaci con
  quello dell'artista che si sta inserendo (artista singolo o band)*/
  IF(tipo_autore = false) THEN
    SIGNAL SQLSTATE "45000" SET MESSAGE_TEXT="Il tipo dell'autore non combacia";
  END IF;

  /* Controllo sulla data di nascita affinchè non ecceda quella corrente*/
  IF(new.data_fondazione > date(now())) THEN
    SIGNAL SQLSTATE "45000" SET MESSAGE_TEXT="Errore nella data di nascita.";
  END IF;

END$

DELIMITER ;
```

Trigger che controlla che ogni disco abbia associato almeno un genere

```
DROP TRIGGER IF EXISTS controlla_generi;

DELIMITER $

CREATE TRIGGER controlla_generi BEFORE DELETE ON classificazione FOR EACH ROW
BEGIN

    DECLARE numero_generi integer;
    /* Controlla che un disco non rimanga mai con meno di 1 genere */

    SET numero_generi = (
        SELECT count(distinct c.nome_genere)
        FROM classificazione c
        WHERE c.id_disco=old.id_disco);

    IF(numero_generi <= 1) THEN
        SIGNAL SQLSTATE "45000" SET MESSAGE_TEXT="Operazione vietata!";
    END IF;

END$
DELIMITER ;
```

Trigger che controlla che la visibilità di una collezione non rimanga in uno stato inconsistente (non possono esistere collezioni pubbliche e condivise allo stesso momento)

```
DROP TRIGGER IF EXISTS on_update_visibilita;

DELIMITER $

CREATE TRIGGER on_update_visibilita BEFORE UPDATE ON collezione_di_dischi FOR
EACH ROW
BEGIN

    IF(old.visibilita = false AND new.visibilita = true) THEN
        DELETE FROM condivisa WHERE id_collezione = old.id;
  END IF;

END$

DELIMITER ;
```

Implementazione funzionalità richieste

Funzionalità 1

Inserimento di una nuova collezione.

```
USE collectors;
DROP PROCEDURE IF EXISTS insert_collezione;

DELIMITER $

CREATE PROCEDURE insert_collezione(
    in id_collezionista integer unsigned,
    in nome_collezione varchar(200),
    in visibilita boolean)
BEGIN
    INSERT INTO collezione_di_dischi(nome,visibilita,id_collezionista)
    VALUES(nome_collezione,visibilita,id_collezionista);
END$

DELIMITER ;
```

Funzionalità 2

Aggiunta di dischi a una collezione e di tracce a un disco.

```
USE collectors;
DROP PROCEDURE IF EXISTS aggiungi_disco_a_collezione;
DROP PROCEDURE IF EXISTS aggiugi_traccia_a_disco;

DELIMITER $

CREATE PROCEDURE aggiungi_disco_a_collezione(
    in titolo varchar(50),
    in anno_di_uscita date,
    in nome_formato varchar(50),
    in nome_stato varchar(50),
    in id_etichetta integer unsigned,
    in id_collezione_di_dischi integer unsigned,
    in barcode varchar(200),
    in note text,
    in numero_copie integer unsigned)
BEGIN

    DECLARE data_invalida CONDITION FOR SQLSTATE '45000';
    DECLARE disco_gia_inserito CONDITION FOR SQLSTATE '23000';

    # Controlla che la data corretta suia valida
    DECLARE EXIT HANDLER FOR data_invalida
    BEGIN
        RESIGNAL SET MESSAGE_TEXT='Errore! Data invalida' ;
    END;

    # Controlla se il disco è gia presente, se si dà errore
    DECLARE EXIT HANDLER FOR disco_gia_inserito
    BEGIN
        RESIGNAL SET MESSAGE_TEXT="Errore! Il disco è gia presente";
    END;

    IF(anno_di_uscita > now()) THEN
        SIGNAL data_invalida;
    ELSE
        INSERT INTO
disco(titolo,anno_di_uscita,nome_formato,nome_stato,id_etichetta,id_collezione_di
_dischi)
```

```
        VALUES
(titolo,anno_di_uscita,nome_formato,nome_stato,id_etichetta,id_collezione_di_disc
hi);
        INSERT INTO info_disco VALUES
(last_insert_id(),barcode,note,numero_copie);
    END IF;
END$

CREATE PROCEDURE aggiungi_traccia_a_disco(
    in titolo varchar(50),
    in durata decimal(10,2),
    in id_etichetta integer unsigned,
    in id_disco integer unsigned)
BEGIN
    INSERT INTO traccia(titolo,durata,id_etichetta,id_disco)
    VALUES (titolo,durata,id_etichetta,id_disco);
END$

DELIMITER ;
```

Funzionalità 3

Modifica dello stato di pubblicazione di una collezione (da privata a pubblica e viceversa) e aggiunta di nuove condivisioni a una collezione.

```
DROP PROCEDURE IF EXISTS inserisci_condivisione;
DROP PROCEDURE IF EXISTS cambia_visibilita;

DELIMITER $

CREATE PROCEDURE inserisci_condivisione(
    in nickname varchar(100),
    in email varchar(200),
    in id_collezione integer unsigned)
BEGIN
    DECLARE id_coll integer unsigned;
    DECLARE visibilita boolean;
    DECLARE id_condivide integer unsigned;

    DECLARE EXIT HANDLER FOR SQLSTATE "45000" BEGIN RESIGNAL; END;

    SELECT c.id FROM collezionista c WHERE c.nickname = nickname AND c.email =
email INTO id_coll;
    SELECT c.id_collezionista,c.visibilita FROM collezione_di_dischi c WHERE c.id
= id_collezione INTO id_condivide,visibilita;

    IF(id_coll = id_condivide)
    THEN
        SIGNAL SQLSTATE "45000" SET MESSAGE_TEXT = "Non si può condividere con se
stessi";
    END IF;
    IF(visibilita = false)
    THEN
        INSERT INTO condivisa(id_collezionista,id_collezione) VALUES
(id_coll,id_collezione);
    ELSE
        SIGNAL SQLSTATE "45000" SET MESSAGE_TEXT = "La collezione è gia
pubblica";
    END IF;
END$

CREATE PROCEDURE cambia_visibilita(in id_collezione integer unsigned)
BEGIN
    DECLARE newVis boolean;

    SELECT visibilita FROM collezione_di_dischi WHERE id = id_collezione INTO
newVis;
```

```
    UPDATE collezione_di_dischi SET visibilita = not newVis WHERE
id=id_collezione;
END$

DELIMITER ;
```

Funzionalità 4

Rimozione di un disco da una collezione.

```
DROP PROCEDURE IF EXISTS rimuovi_disco;

DELIMITER $

CREATE PROCEDURE rimuovi_disco(in id_disco integer unsigned)
BEGIN
    DELETE FROM disco WHERE id = id_disco;
END$

DELIMITER ;
```

Funzionalità 5

Rimozione di una collezione.

```
DROP PROCEDURE IF EXISTS rimuovi_collezione;

DELIMITER $

CREATE PROCEDURE rimuovi_collezione(in id_collezione integer unsigned)
BEGIN
    DELETE FROM collezione_di_dischi WHERE id= id_collezione;
END$

DELIMITER ;
```

Funzionalità 6

Lista di tutti i dischi in una collezione.

```
DROP PROCEDURE IF EXISTS dischi_in_collezione;
DROP FUNCTION IF EXISTS generi_disco;

DELIMITER $

CREATE PROCEDURE dischi_in_collezione(in id_collezione integer unsigned)
BEGIN
    SELECT d.id as "ID",
           d.titolo as "Titolo",
           d.anno_di_uscita as "Anno di uscita",
           d.nome_stato as "Stato",
           d.nome_formato as "Formato",
           e.nome as "Etichetta",
           generi_disco(d.id) as "Generi", # Funzione per ricavare tutti i generi
in un unica riga
           inf.barcode as "Barcode",
           inf.note as "Note",
           inf.numero_copie as "Copie"
    from disco d
    join etichetta e on d.id_etichetta = e.id
    join info_disco inf on inf.id_disco = d.id
    where d.id_collezione_di_dischi = id_collezione;
END$

#Funzione che ritorna tutti i generi di un disco in un unica concatenazione da
usare in altre query
CREATE FUNCTION generi_disco(id_disco integer unsigned)
RETURNS varchar(200) DETERMINISTIC
BEGIN
    RETURN (SELECT group_concat(g.nome separator ", ")
            FROM classificazione c
            JOIN genere g ON c.nome_genere = g.nome
            WHERE c.id_disco = id_disco);
END$

DELIMITER ;
```

Funzionalità 7

Track list di un disco.

```
DROP PROCEDURE IF EXISTS track_list_disco;

DELIMITER $

CREATE PROCEDURE track_list_disco(id_disco integer unsigned)
BEGIN
    SELECT t.titolo as "Titolo traccia",t.durata as "Durata" FROM traccia t WHERE
t.id_disco=id_disco;
END$

DELIMITER ;
```

Funzionalità 8

Ricerca di dischi in base a nomi di autori/compositori/interpreti e/o titoli. Si potrà decidere di includere nella ricerca le collezioni di un certo collezionista e/o quelle condivise con lo stesso collezionista e/o quelle pubbliche. (Suggerimento: potete realizzare diverse query in base alle varie combinazioni di criteri di ricerca. Usate la UNION per unire i risultati delle ricerche effettuate sulle collezioni private, condivise e pubbliche)

```
 /* La seguente query si articola così:

 1. Ricerca in collezioni personale, condivise e pubbliche
    1.1 Ricerca con nome d'arte e con titolo
    1.2 Ricerca con titolo
    1.3 Ricerca con nome d'arte
 2. Ricerca in collezioni personali e condivise
    2.1 Ricerca con nome d'arte e con titolo
    2.2 Ricerca con titolo
    2.3 Ricerca con nome d'arte
 3. Ricerca in collezioni personali e pubbliche
    3.1 Ricerca con nome d'arte e con titolo
    3.2 Ricerca con titolo
    3.3 Ricerca con nome d'arte
 4. Ricerca in collezioni pubbliche e condivise
    4.1 Ricerca con nome d'arte e con titolo
    4.2 Ricerca con titolo
    4.3 Ricerca con nome d'arte
 5. Ricerca in collezioni personali
    5.1 Ricerca con nome d'arte e con titolo
    5.2 Ricerca con titolo
    5.3 Ricerca con nome d'arte
 6. Ricerca in collezioni condivise
    6.1 Ricerca con nome d'arte e con titolo
    6.2 Ricerca con titolo
    6.3 Ricerca con nome d'arte
 7. Ricerca in collezioni pubbliche
    7.1 Ricerca con nome d'arte e con titolo
    7.2 Ricerca con titolo
    7.3 Ricerca con nome d'arte

  In questo modo vengono coperte tutte le possibili combinazioni di ricerca
*/

DROP PROCEDURE IF EXISTS ricerca_di_dischi_con_autore_eo_titolo;

DELIMITER $
```

```
CREATE PROCEDURE ricerca_di_dischi_con_autore_eo_titolo(
    in nomedarte varchar(100),
    in titolo varchar(50),
    in id_collezionista integer unsigned,
    in collezioni boolean,
    in condivise boolean,
    in pubbliche boolean)
BEGIN
    IF(pubbliche AND condivise AND collezioni) THEN

        /* Se il nomedarte e il titolo sono entrambi not null*/
        IF(titolo is not null and nomedarte is not null) THEN
            ( /* Ricerca di un disco in collezioni personali dato l'id del
                collezionista, il nome d'arte dell'autore e il titolo*/
                SELECT d.titolo as "Titolo",
                       d.anno_di_uscita as "Anno di uscita",
                       d.nome_formato as "Formato",
                       d.nome_stato as "Condizioni",
                       cd.nome as "Collezione",
                       co.nickname as "Proprietario"
                FROM disco d
                JOIN collezione_di_dischi cd ON d.id_collezione_di_dischi=cd.id
                JOIN incide i ON d.id=i.id_disco
                JOIN autore a ON i.id_autore=a.id
                JOIN collezionista co ON cd.id_collezionista=co.id
                WHERE cd.id_collezionista = id_collezionista
                AND a.nome_darte LIKE nomedarte
                AND d.titolo LIKE titolo
            ) UNION (/* Ricerca di un disco in collezioni pubbliche dato l'id del
                        collezionista, il nome d'arte dell'autore e il titolo */
                SELECT d.titolo as "Titolo",
                       d.anno_di_uscita as "Anno di uscita",
                       d.nome_formato as "Formato",
                       d.nome_stato as "Condizioni",
                       cd.nome as "Collezione",
                       co.nickname as "Proprietario"
                FROM disco d
                JOIN collezione_di_dischi cd ON d.id_collezione_di_dischi=cd.id
                JOIN incide i ON d.id=i.id_disco
                JOIN autore a ON i.id_autore=a.id
                JOIN collezionista co ON cd.id_collezionista=co.id
                WHERE cd.visibilita=true
                AND a.nome_darte LIKE nomedarte
                AND d.titolo LIKE titolo
```

```
            ) UNION (/* Ricerca di un disco in collezioni condivise con il
collezionista
                        dato l'id del collezionista, il nome d'arte dell'autore e
il titolo */
                SELECT d.titolo as "Titolo",
                       d.anno_di_uscita as "Anno di uscita",
                       d.nome_formato as "Formato",
                       d.nome_stato as "Condizioni",
                       cd.nome as "Collezione",
                       co.nickname as "Proprietario"
                FROM condivisa c
                JOIN collezione_di_dischi cd ON c.id_collezione=cd.id
                JOIN disco d ON d.id_collezione_di_dischi=cd.id
                JOIN incide i ON i.id_disco=d.id
                JOIN autore a ON i.id_autore=a.id
                JOIN collezionista co ON cd.id_collezionista=co.id
                WHERE c.id_collezionista=id_collezionista
                AND a.nome_darte LIKE nomedarte
                AND d.titolo LIKE titolo
            );
        ELSE
        /* Se il titolo è null */
        IF(titolo is null) THEN
            ( /* Ricerca di un disco in collezioni personali dato l'id del
                collezionista e il nome d'arte dell'autore*/
                SELECT d.titolo as "Titolo",
                       d.anno_di_uscita as "Anno di uscita",
                       d.nome_formato as "Formato",
                       d.nome_stato as "Condizioni",
                       cd.nome as "Collezione",
                       co.nickname as "Proprietario"
                FROM disco d
                JOIN collezione_di_dischi cd ON d.id_collezione_di_dischi=cd.id
                JOIN incide i ON d.id=i.id_disco
                JOIN autore a ON i.id_autore=a.id
                JOIN collezionista co ON cd.id_collezionista=co.id
                WHERE cd.id_collezionista = id_collezionista
                AND a.nome_darte LIKE nomedarte
            ) UNION ( /* Ricerca di un disco in collezioni pubbliche dato l'id
del
                    collezionista e il nome d'arte dell'autore */
                SELECT d.titolo as "Titolo",
                       d.anno_di_uscita as "Anno di uscita",
                       d.nome_formato as "Formato",
                       d.nome_stato as "Condizioni",
```

```
                    cd.nome as "Collezione",
                    co.nickname as "Proprietario"
                FROM disco d
                JOIN collezione_di_dischi cd ON d.id_collezione_di_dischi=cd.id
                JOIN incide i ON d.id=i.id_disco
                JOIN autore a ON i.id_autore=a.id
                JOIN collezionista co ON cd.id_collezionista=co.id
                WHERE cd.visibilita=true
                AND a.nome_darte LIKE nomedarte
            ) UNION ( /* Ricerca di un disco in collezioni condivise con il
collezionista dato
                    l'id del collezionista e il nome d'arte dell'autore */
                SELECT d.titolo as "Titolo",
                    d.anno_di_uscita as "Anno di uscita",
                    d.nome_formato as "Formato",
                    d.nome_stato as "Condizioni",
                    cd.nome as "Collezione",
                    co.nickname as "Proprietario"
                FROM condivisa c
                JOIN collezione_di_dischi cd ON c.id_collezione=cd.id
                JOIN disco d ON d.id_collezione_di_dischi=cd.id
                JOIN incide i ON i.id_disco=d.id
                JOIN autore a ON i.id_autore=a.id
                JOIN collezionista co ON cd.id_collezionista=co.id
                WHERE c.id_collezionista=id_collezionista
                AND a.nome_darte LIKE nomedarte
            );
        ELSE

        /* Se il nomedarte è null */
        IF(nomedarte is null) THEN
            ( /* Ricerca di un disco in collezioni personali dato l'id del
                collezionista e il nome d'arte dell'autore*/
                SELECT d.titolo as "Titolo",
                    d.anno_di_uscita as "Anno di uscita",
                    d.nome_formato as "Formato",
                    d.nome_stato as "Condizioni",
                    cd.nome as "Collezione",
                    co.nickname as "Proprietario"
                FROM disco d
                JOIN collezione_di_dischi cd ON d.id_collezione_di_dischi=cd.id
                JOIN incide i ON d.id=i.id_disco
                JOIN autore a ON i.id_autore=a.id
                JOIN collezionista co ON cd.id_collezionista=co.id
                WHERE cd.id_collezionista = id_collezionista
```

```
                AND d.titolo LIKE titolo
            ) UNION ( /* Ricerca di un disco in collezioni pubbliche dato l'id
del
                    collezionista e il nome d'arte dell'autore */
                SELECT d.titolo as "Titolo",
                       d.anno_di_uscita as "Anno di uscita",
                       d.nome_formato as "Formato",
                       d.nome_stato as "Condizioni",
                       cd.nome as "Collezione",
                       co.nickname as "Proprietario"
                FROM disco d
                JOIN collezione_di_dischi cd ON d.id_collezione_di_dischi=cd.id
                JOIN incide i ON d.id=i.id_disco
                JOIN autore a ON i.id_autore=a.id
                JOIN collezionista co ON cd.id_collezionista=co.id
                WHERE cd.visibilita=true
                AND d.titolo LIKE titolo
            ) UNION ( /* Ricerca di un disco in collezioni condivise con il
collezionista dato
                    l'id del collezionista e il nome d'arte dell'autore */
                SELECT d.titolo as "Titolo",
                       d.anno_di_uscita as "Anno di uscita",
                       d.nome_formato as "Formato",
                       d.nome_stato as "Condizioni",
                       cd.nome as "Collezione",
                       co.nickname as "Proprietario"
                FROM condivisa c
                JOIN collezione_di_dischi cd ON c.id_collezione=cd.id
                JOIN disco d ON d.id_collezione_di_dischi=cd.id
                JOIN incide i ON i.id_disco=d.id
                JOIN autore a ON i.id_autore=a.id
                JOIN collezionista co ON cd.id_collezionista=co.id
                WHERE c.id_collezionista=id_collezionista
                AND d.titolo LIKE titolo
            );
        END IF;
        END IF;
        END IF;

    ELSE IF(condivise and collezioni) THEN

        /* Se il nomedarte e il titolo sono entrambi not null*/
        IF(titolo is not null and nomedarte is not null) THEN
            ( /* Ricerca di dischi in collezioni personali dato l'id del
collezionista
```

```
                    e il nome dell'autore e il titolo del disco*/
                    SELECT d.titolo as "Titolo",
                           d.anno_di_uscita as "Anno di uscita",
                           d.nome_formato as "Formato",
                           d.nome_stato as "Condizioni",
                           cd.nome as "Collezione",
                           co.nickname as "Proprietario"
                    FROM disco d
                    JOIN collezione_di_dischi cd ON
d.id_collezione_di_dischi=cd.id
                    JOIN incide i on d.id=i.id_disco
                    JOIN autore a on i.id_autore=a.id
                    JOIN collezionista co on cd.id_collezionista=co.id
                    WHERE cd.id_collezionista = id_collezionista
                    AND a.nome_darte LIKE nomedarte
                    AND d.titolo LIKE titolo
                ) UNION (/* Ricerca di dischi in collezioni condivise con il
collezionista dato l'id
                             del collezionista e il nome dell'autore e il titolo
del disco*/
                    SELECT d.titolo as "Titolo",
                           d.anno_di_uscita as "Anno di uscita",
                           d.nome_formato as "Formato",
                           d.nome_stato as "Condizioni",
                           cd.nome as "Collezione",
                           co.nickname as "Proprietario"
                    FROM condivisa c
                    JOIN collezione_di_dischi cd ON c.id_collezione=cd.id
                    JOIN disco d ON d.id_collezione_di_dischi=cd.id
                    JOIN incide i ON i.id_disco=d.id
                    JOIN autore a ON i.id_autore=a.id
                    JOIN collezionista co ON cd.id_collezionista=co.id
                    WHERE c.id_collezionista=id_collezionista
                    AND a.nome_darte LIKE nomedarte
                    AND d.titolo LIKE titolo
                );
        ELSE
        /* Se il titolo è null */
        IF(titolo is null) THEN
            ( /* Ricerca di dischi in collezioni personali dato l'id del
collezionista
                e il nome dell'autore */
                SELECT d.titolo as "Titolo",
                       d.anno_di_uscita as "Anno di uscita",
                       d.nome_formato as "Formato",
```

```
                        d.nome_stato as "Condizioni",
                        cd.nome as "Collezione",
                        co.nickname as "Proprietario"
                FROM disco d
                JOIN collezione_di_dischi cd on d.id_collezione_di_dischi=cd.id
                JOIN incide i on d.id=i.id_disco
                JOIN autore a on i.id_autore=a.id
                JOIN collezionista co on cd.id_collezionista=co.id
                WHERE cd.id_collezionista = id_collezionista
                AND a.nome_darte LIKE nomedarte
            ) UNION ( /* Ricerca di dischi in collezioni condivise con il
collezionista dato l'id
                         del collezionista e il nome dell'autore*/
                SELECT d.titolo as "Titolo",
                        d.anno_di_uscita as "Anno di uscita",
                        d.nome_formato as "Formato",
                        d.nome_stato as "Condizioni",
                        cd.nome as "Collezione",
                        co.nickname as "Proprietario"
                FROM condivisa c
                JOIN collezione_di_dischi cd ON c.id_collezione=cd.id
                JOIN disco d ON d.id_collezione_di_dischi=cd.id
                JOIN incide i ON i.id_disco=d.id
                JOIN autore a ON i.id_autore=a.id
                JOIN collezionista co ON cd.id_collezionista=co.id
                WHERE c.id_collezionista=id_collezionista
                AND a.nome_darte LIKE nomedarte
            );
        ELSE

        /* Se il nomedarte è null */
        IF(nomedarte is null) THEN
            ( /* Ricerca di dischi in collezioni personali dato l'id del
collezionista
                e il nome dell'autore */
                SELECT d.titolo as "Titolo",
                        d.anno_di_uscita as "Anno di uscita",
                        d.nome_formato as "Formato",
                        d.nome_stato as "Condizioni",
                        cd.nome as "Collezione",
                        co.nickname as "Proprietario"
                FROM disco d
                JOIN collezione_di_dischi cd on d.id_collezione_di_dischi=cd.id
                JOIN incide i on d.id=i.id_disco
                JOIN autore a on i.id_autore=a.id
```

```
                JOIN collezionista co on cd.id_collezionista=co.id
                WHERE cd.id_collezionista = id_collezionista
                AND d.titolo LIKE titolo
            ) UNION ( /* Ricerca di dischi in collezioni condivise con il
collezionista dato l'id
                        del collezionista e il nome dell'autore*/
                SELECT d.titolo as "Titolo",
                       d.anno_di_uscita as "Anno di uscita",
                       d.nome_formato as "Formato",
                       d.nome_stato as "Condizioni",
                       cd.nome as "Collezione",
                       co.nickname as "Proprietario"
                FROM condivisa c
                JOIN collezione_di_dischi cd ON c.id_collezione=cd.id
                JOIN disco d ON d.id_collezione_di_dischi=cd.id
                JOIN incide i ON i.id_disco=d.id
                JOIN autore a ON i.id_autore=a.id
                JOIN collezionista co ON cd.id_collezionista=co.id
                WHERE c.id_collezionista=id_collezionista
                AND d.titolo LIKE titolo
            );
        END IF;
        END IF;
        END IF;

    ELSE IF(pubbliche AND collezioni) THEN

        /* Se il nomedarte e il titolo sono entrambi not null*/
        IF(titolo is not null and nomedarte is not null) THEN
            ( /* Ricerca di dischi in collezioni personali dato il nome d'arte
dell'autore e il
                titolo del disco */
                SELECT d.titolo as "Titolo",
                       d.anno_di_uscita as "Anno di uscita",
                       d.nome_formato as "Formato",
                       d.nome_stato as "Condizioni",
                       cd.nome as "Collezione",
                       co.nickname as "Proprietario"
                FROM disco d
                JOIN collezione_di_dischi cd ON d.id_collezione_di_dischi=cd.id
                JOIN incide i ON d.id=i.id_disco
                JOIN autore a ON i.id_autore=a.id
                JOIN collezionista co ON cd.id_collezionista=co.id
                WHERE cd.id_collezionista = id_collezionista
                AND a.nome_darte LIKE nomedarte
```

```
                AND d.titolo LIKE titolo
            ) UNION ( /* Ricerca di dischi in collezioni pubbliche dato il nome
d'arte dell'autore e il
                        titolo del disco */
                SELECT d.titolo as "Titolo",
                    d.anno_di_uscita as "Anno di uscita",
                    d.nome_formato as "Formato",
                    d.nome_stato as "Condizioni",
                    cd.nome as "Collezione",
                    co.nickname as "Proprietario"
                FROM disco d
                JOIN collezione_di_dischi cd ON d.id_collezione_di_dischi=cd.id
                JOIN incide i ON d.id=i.id_disco
                JOIN autore a ON i.id_autore=a.id
                JOIN collezionista as co ON cd.id_collezionista=co.id
                WHERE cd.visibilita=true
                AND a.nome_darte LIKE nomedarte
                AND d.titolo LIKE titolo
            );
        ELSE

        /* Se il titolo è null */
        IF(titolo is null) THEN
            ( /* Ricerca di dischi in collezioni personali dato il nome d'arte
dell'autore*/
                SELECT d.titolo as "Titolo",
                       d.anno_di_uscita as "Anno di uscita",
                       d.nome_formato as "Formato",
                       d.nome_stato as "Condizioni",
                       cd.nome as "Collezione",
                       co.nickname as "Proprietario"
                FROM disco d
                JOIN collezione_di_dischi cd ON d.id_collezione_di_dischi=cd.id
                JOIN incide i ON d.id=i.id_disco
                JOIN autore a ON i.id_autore=a.id
                JOIN collezionista co ON cd.id_collezionista=co.id
                WHERE cd.id_collezionista = id_collezionista
                AND a.nome_darte LIKE nomedarte
            ) UNION ( /* Ricerca di dischi in collezioni pubbliche dato il nome
d'arte dell'autore */
                SELECT d.titolo as "Titolo",
                       d.anno_di_uscita as "Anno di uscita",
                       d.nome_formato as "Formato",
                       d.nome_stato as "Condizioni",
                       cd.nome as "Collezione",
```

```
                co.nickname as "Proprietario"
            FROM disco d
            JOIN collezione_di_dischi cd ON d.id_collezione_di_dischi=cd.id
            JOIN incide i ON d.id=i.id_disco
            JOIN autore a ON i.id_autore=a.id
            JOIN collezionista as co ON cd.id_collezionista=co.id
            WHERE cd.visibilita=true
            AND a.nome_darte LIKE nomedarte
        );
    ELSE

    /* Se il nomedarte è null */
    IF(nomedarte is null) THEN
        ( /* Ricerca di dischi in collezioni personali dato il nome d'arte
dell'autore*/
            SELECT d.titolo as "Titolo",
                d.anno_di_uscita as "Anno di uscita",
                d.nome_formato as "Formato",
                d.nome_stato as "Condizioni",
                cd.nome as "Collezione",
                co.nickname as "Proprietario"
            FROM disco d
            JOIN collezione_di_dischi cd ON d.id_collezione_di_dischi=cd.id
            JOIN incide i ON d.id=i.id_disco
            JOIN autore a ON i.id_autore=a.id
            JOIN collezionista co ON cd.id_collezionista=co.id
            WHERE cd.id_collezionista = id_collezionista
            AND d.titolo LIKE titolo
        ) UNION ( /* Ricerca di dischi in collezioni pubbliche dato il nome
d'arte dell'autore */
            SELECT d.titolo as "Titolo",
                d.anno_di_uscita as "Anno di uscita",
                d.nome_formato as "Formato",
                d.nome_stato as "Condizioni",
                cd.nome as "Collezione",
                co.nickname as "Proprietario"
            FROM disco d
            JOIN collezione_di_dischi cd ON d.id_collezione_di_dischi=cd.id
            JOIN incide i ON d.id=i.id_disco
            JOIN autore a ON i.id_autore=a.id
            JOIN collezionista as co ON cd.id_collezionista=co.id
            WHERE cd.visibilita=true
            AND d.titolo LIKE titolo
        );
    END IF;
```

```
        END IF;
        END IF;

    ELSE IF(pubbliche AND condivise) THEN

        /* Se il nomedarte e il titolo sono entrambi not null*/
        IF(titolo is not null and nomedarte is not null) THEN
            ( /* Ricerca di dischi in collezioni pubbliche dato il nome d'arte
dell'autore e il
                titolo del disco */
                SELECT d.titolo as "Titolo",
                       d.anno_di_uscita as "Anno di uscita",
                       d.nome_formato as "Formato",
                       d.nome_stato as "Condizioni",
                       cd.nome as "Collezione",
                       co.nickname as "Proprietario"
                FROM disco d
                JOIN collezione_di_dischi cd on d.id_collezione_di_dischi=cd.id
                JOIN incide i on d.id=i.id_disco
                JOIN autore a on i.id_autore=a.id
                JOIN collezionista co on cd.id_collezionista=co.id
                WHERE cd.visibilita=true
                AND a.nome_darte LIKE nomedarte
                AND d.titolo LIKE titolo
            ) UNION  (/* Ricerca di dischi condivisi con il collezionista dato il
nome d'arte
                        dell'autore e il titolo del disco*/
                SELECT d.titolo as "Titolo",
                       d.anno_di_uscita as "Anno di uscita",
                       d.nome_formato as "Formato",
                       d.nome_stato as "Condizioni",
                       cd.nome as "Collezione",
                       co.nickname as "Proprietario"
                FROM condivisa c
                JOIN collezione_di_dischi cd ON c.id_collezione=cd.id
                JOIN disco d ON d.id_collezione_di_dischi=cd.id
                JOIN incide i ON i.id_disco=d.id
                JOIN autore a ON i.id_autore=a.id
                JOIN collezionista co ON cd.id_collezionista=co.id
                WHERE c.id_collezionista=id_collezionista
                AND a.nome_darte LIKE nomedarte
                AND d.titolo LIKE titolo
            );
        ELSE
```

```
        /* Se il titolo è null */
        IF(titolo is null) THEN
            ( /* Ricerca di dischi in collezioni pubbliche dato il nome d'arte
dell'autore */
                SELECT d.titolo as "Titolo",
                       d.anno_di_uscita as "Anno di uscita",
                       d.nome_formato as "Formato",
                       d.nome_stato as "Condizioni",
                       cd.nome as "Collezione",
                       co.nickname as "Proprietario"
                FROM disco d
                JOIN collezione_di_dischi cd ON d.id_collezione_di_dischi=cd.id
                JOIN incide i ON d.id=i.id_disco
                JOIN autore a ON i.id_autore=a.id
                JOIN collezionista co ON cd.id_collezionista=co.id
                WHERE cd.visibilita=true
                AND a.nome_darte LIKE nomedarte
            ) UNION ( /* Ricerca di dischi in collezioni condivise con il
collezionista dato il
                         nome d'arte dell'autore */
                SELECT d.titolo as "Titolo",
                       d.anno_di_uscita as "Anno di uscita",
                       d.nome_formato as "Formato",
                       d.nome_stato as "Condizioni",
                       cd.nome as "Collezione",
                       co.nickname as "Proprietario"
                FROM condivisa c
                JOIN collezione_di_dischi cd ON c.id_collezione=cd.id
                JOIN disco d ON d.id_collezione_di_dischi=cd.id
                JOIN incide i ON i.id_disco=d.id
                JOIN autore a ON i.id_autore=a.id
                JOIN collezionista co ON cd.id_collezionista=co.id
                WHERE c.id_collezionista=id_collezionista
                AND a.nome_darte LIKE nomedarte
            );
        ELSE

        /* Se il domedarte è null */
        IF(nomedarte is null) THEN
            ( /* Ricerca di dischi in collezioni pubbliche dato il nome d'arte
dell'autore */
                SELECT d.titolo as "Titolo",
                       d.anno_di_uscita as "Anno di uscita",
                       d.nome_formato as "Formato",
                       d.nome_stato as "Condizioni",
```

```
                    cd.nome as "Collezione",
                    co.nickname as "Proprietario"
                FROM disco d
                JOIN collezione_di_dischi cd ON d.id_collezione_di_dischi=cd.id
                JOIN incide i ON d.id=i.id_disco
                JOIN autore a ON i.id_autore=a.id
                JOIN collezionista co ON cd.id_collezionista=co.id
                WHERE cd.visibilita=true
                AND d.titolo LIKE titolo
            ) UNION ( /* Ricerca di dischi in collezioni condivise con il
collezionista dato il
                     nome d'arte dell'autore */
                SELECT d.titolo as "Titolo",
                    d.anno_di_uscita as "Anno di uscita",
                    d.nome_formato as "Formato",
                    d.nome_stato as "Condizioni",
                    cd.nome as "Collezione",
                    co.nickname as "Proprietario"
                FROM condivisa c
                JOIN collezione_di_dischi cd ON c.id_collezione=cd.id
                JOIN disco d ON d.id_collezione_di_dischi=cd.id
                JOIN incide i ON i.id_disco=d.id
                JOIN autore a ON i.id_autore=a.id
                JOIN collezionista co ON cd.id_collezionista=co.id
                WHERE c.id_collezionista=id_collezionista
                AND d.titolo LIKE titolo
            );
        END IF;
        END IF;
        END IF;

    ELSE IF(collezioni) THEN

        /* Se il nomedarte e il titolo sono entrambi not null*/
        IF(titolo is not null and nomedarte is not null) THEN
            ( /* Ricerca di dischi in collezioni personali dato il nome d'arte
dell'autore
                e il titolo*/
                SELECT d.titolo as "Titolo",
                    d.anno_di_uscita as "Anno di uscita",
                    d.nome_formato as "Formato",
                    d.nome_stato as "Condizioni",
                    cd.nome as "Collezione",
                    co.nickname as "Proprietario"
                FROM disco d
```

```
                JOIN collezione_di_dischi cd ON d.id_collezione_di_dischi=cd.id
                JOIN incide i ON d.id=i.id_disco
                JOIN autore a ON i.id_autore=a.id
                JOIN collezionista co ON cd.id_collezionista=co.id
                WHERE cd.id_collezionista = id_collezionista
                AND a.nome_darte LIKE nomedarte
                AND d.titolo LIKE titolo
            );
        ELSE

        /* Se il titolo è null */
        IF(titolo is null) THEN
            ( /* Ricerca di dischi in collezioni personali dato il nome d'arte
dell'autore*/
                SELECT d.titolo as "Titolo",
                       d.anno_di_uscita as "Anno di uscita",
                       d.nome_formato as "Formato",
                       d.nome_stato as "Condizioni",
                       cd.nome as "Collezione",
                       co.nickname as "Proprietario"
                FROM disco d
                JOIN collezione_di_dischi cd ON d.id_collezione_di_dischi=cd.id
                JOIN incide i ON d.id=i.id_disco
                JOIN autore a ON i.id_autore=a.id
                JOIN collezionista co ON cd.id_collezionista=co.id
                WHERE cd.id_collezionista = id_collezionista
                AND a.nome_darte LIKE nomedarte
            );
        ELSE

        /* Se il nomedarte è null */
        IF(nomedarte is null) THEN
            ( /* Ricerca di dischi in collezioni personali dato il nome d'arte
dell'autore*/
                SELECT d.titolo as "Titolo",
                       d.anno_di_uscita as "Anno di uscita",
                       d.nome_formato as "Formato",
                       d.nome_stato as "Condizioni",
                       cd.nome as "Collezione",
                       co.nickname as "Proprietario"
                FROM disco d
                JOIN collezione_di_dischi cd ON d.id_collezione_di_dischi=cd.id
                JOIN incide i ON d.id=i.id_disco
                JOIN autore a ON i.id_autore=a.id
                JOIN collezionista co ON cd.id_collezionista=co.id
```

```
                WHERE cd.id_collezionista = id_collezionista
                AND d.titolo LIKE titolo
            );
        END IF;
        END IF;
        END IF;

    ELSE IF(condivise) THEN

        /* Se il nomedarte e il titolo sono entrambi not null*/
        IF(titolo is not null and nomedarte is not null) THEN
            ( /* Ricerca di dischi in collezioni condivise con il collezionista
dato
                il nome dell'autore e il titolo*/
                SELECT d.titolo as "Titolo",
                d.anno_di_uscita as "Anno di uscita",
                d.nome_formato as "Formato",
                d.nome_stato as "Condizioni",
                cd.nome as "Collezione",
                co.nickname as "Proprietario"
                FROM condivisa c
                JOIN collezione_di_dischi cd ON c.id_collezione=cd.id
                JOIN disco d ON d.id_collezione_di_dischi=cd.id
                JOIN incide i ON i.id_disco=d.id
                JOIN autore a ON i.id_autore=a.id
                JOIN collezionista co ON cd.id_collezionista=co.id
                WHERE c.id_collezionista=id_collezionista
                AND a.nome_darte LIKE nomedarte
                AND d.titolo LIKE titolo
            );
        ELSE

        /* Se il titolo è null */
        IF(titolo is null) THEN
            ( /* Ricerca di dischi in collezioni condivise con il collezionista
dato
                il nome dell'autore*/
                SELECT d.titolo as "Titolo",
                d.anno_di_uscita as "Anno di uscita",
                d.nome_formato as "Formato",
                d.nome_stato as "Condizioni",
                cd.nome as "Collezione",
                co.nickname as "Proprietario"
                FROM condivisa c
                JOIN collezione_di_dischi cd ON c.id_collezione=cd.id
```

```
                JOIN disco d ON d.id_collezione_di_dischi=cd.id
                JOIN incide i ON i.id_disco=d.id
                JOIN autore a ON i.id_autore=a.id
                JOIN collezionista co ON cd.id_collezionista=co.id
                WHERE c.id_collezionista=id_collezionista
                AND a.nome_darte LIKE nomedarte
            );

        ELSE

        /* Se il nomedarte è null */
        IF(nomedarte is null) THEN
            ( /* Ricerca di dischi in collezioni condivise con il collezionista
dato
                il nome dell'autore*/
                SELECT d.titolo as "Titolo",
                d.anno_di_uscita as "Anno di uscita",
                d.nome_formato as "Formato",
                d.nome_stato as "Condizioni",
                cd.nome as "Collezione",
                co.nickname as "Proprietario"
                FROM condivisa c
                JOIN collezione_di_dischi cd ON c.id_collezione=cd.id
                JOIN disco d ON d.id_collezione_di_dischi=cd.id
                JOIN incide i ON i.id_disco=d.id
                JOIN autore a ON i.id_autore=a.id
                JOIN collezionista co ON cd.id_collezionista=co.id
                WHERE c.id_collezionista=id_collezionista
                AND d.titolo LIKE titolo
            );

        END IF;
        END IF;
        END IF;

    ELSE IF(pubbliche) THEN

        /* Se il nomedarte e il titolo sono entrambi not null*/
        IF(titolo is not null and nomedarte is not null) THEN
            ( /* Ricerca di dischi in collezioni pubbliche dato il nome d'arte
dell'autore
                e il titolo del disco*/
                SELECT d.titolo as "Titolo",
                d.anno_di_uscita as "Anno di uscita",
                d.nome_formato as "Formato",
```

```
                d.nome_stato as "Condizioni",
                cd.nome as "Collezione",
                co.nickname as "Proprietario"
                FROM disco d
                JOIN collezione_di_dischi cd ON d.id_collezione_di_dischi=cd.id
                JOIN incide i ON d.id=i.id_disco
                JOIN autore a ON i.id_autore=a.id
                JOIN collezionista co ON cd.id_collezionista=co.id
                WHERE cd.visibilita=true
                AND a.nome_darte LIKE nomedarte
                AND d.titolo LIKE titolo
            );
        ELSE

        /* Se il titolo è null */
        IF(titolo is null) THEN
            ( /* Ricerca di dischi in collezioni pubbliche dato il nome d'arte
dell'autore*/
                SELECT d.titolo as "Titolo",
                d.anno_di_uscita as "Anno di uscita",
                d.nome_formato as "Formato",
                d.nome_stato as "Condizioni",
                cd.nome as "Collezione",
                co.nickname as "Proprietario"
                FROM disco d
                JOIN collezione_di_dischi cd ON d.id_collezione_di_dischi=cd.id
                JOIN incide i ON d.id=i.id_disco
                JOIN autore a ON i.id_autore=a.id
                JOIN collezionista co ON cd.id_collezionista=co.id
                WHERE cd.visibilita=true
                AND a.nome_darte LIKE nomedarte
            );
        ELSE

        /* Se il nomedarte è null */
        IF(nomedarte is null) THEN
            ( /* Ricerca di dischi in collezioni pubbliche dato il nome d'arte
dell'autore*/
                SELECT d.titolo as "Titolo",
                d.anno_di_uscita as "Anno di uscita",
                d.nome_formato as "Formato",
                d.nome_stato as "Condizioni",
                cd.nome as "Collezione",
                co.nickname as "Proprietario"
                FROM disco d
```

```
                JOIN collezione_di_dischi cd ON d.id_collezione_di_dischi=cd.id
                JOIN incide i ON d.id=i.id_disco
                JOIN autore a ON i.id_autore=a.id
                JOIN collezionista co ON cd.id_collezionista=co.id
                WHERE cd.visibilita=true
                AND d.titolo LIKE titolo
            );

        END IF;
        END IF;
        END IF;

    END IF;
    END IF;
    END IF;
    END IF;
    END IF;
    END IF;
    END IF;
END$

DELIMITER ;
```

Funzionalità 9

Verifica della visibilità di una collezione da parte di un collezionista. (Suggerimento: una collezione è visibile a un collezionista se è sua, condivisa con lui o pubblica)

```
DROP PROCEDURE IF EXISTS verifica_visibilita_collezione;

DELIMITER $

CREATE PROCEDURE verifica_visibilita_collezione(
    in id_collezione_di_dischi integer unsigned,
    in id_collezionista integer unsigned)
BEGIN
    (/* Verifica della visibiltà per collezioni personali */
        SELECT cd.id as "Visibilità"
        FROM collezione_di_dischi cd
        WHERE cd.id=id_collezione_di_dischi
        AND cd.id_collezionista=id_collezionista
    ) UNION (/* Verifica della visibilità per collezioni pubbliche */
        select cd.id as "Visibilità"
        from collezione_di_dischi as cd
        where cd.id=id_collezione_di_dischi
            and cd.visibilita=true
    ) UNION (/* Verifica della visibilità per collezioni condivise */
        SELECT c.id_collezione as "Visibilità"
        FROM condivisa c
        WHERE c.id_collezione=id_collezione_di_dischi
        AND c.id_collezionista=id_collezionista
    );

END$

DELIMITER ;
```

Funzionalità 10

Numero dei brani (tracce di dischi) distinti di un certo autore (compositore, musicista) presenti nelle collezioni pubbliche.

```
DROP FUNCTION IF EXISTS conta_brani;

DELIMITER $

CREATE FUNCTION conta_brani(nome_darte varchar(200))
RETURNS integer unsigned DETERMINISTIC
BEGIN

    DECLARE numero_brani integer unsigned;

    SET numero_brani = (
        SELECT count(distinct t.id)
        FROM autore a
        JOIN produce p ON p.id_autore=a.id
        JOIN traccia t ON t.id=p.id_traccia
        JOIN disco d ON d.id=t.id_disco
        JOIN collezione_di_dischi c ON c.id=d.id_collezione_di_dischi
        WHERE c.visibilita=true
        AND a.nome_darte LIKE nome_darte
        GROUP BY a.id);

    # Se numer_brani è null, ritorna 0, altrimenti il numero di brani
    IF(numero_brani is null) THEN
        RETURN 0;
    ELSE
        RETURN numero_brani;
    END IF;

END$

DELIMITER ;
```

Funzionalità 11

Minuti totali di musica riferibili a un certo autore (compositore, musicista) memorizzati nelle collezioni pubbliche.

```
DROP FUNCTION IF EXISTS conta_minuti_autore;

DELIMITER $

CREATE FUNCTION conta_minuti_autore(nome_darte varchar(100))
RETURNS decimal(10,2) DETERMINISTIC
BEGIN

    DECLARE minutaggio_totale decimal(10,2);

    SET minutaggio_totale = (
        SELECT sum(distinct t.durata)
        FROM autore a
        JOIN produce p ON p.id_autore=a.id
        JOIN traccia t ON t.id=p.id_traccia
        JOIN disco d ON d.id=t.id_disco
        JOIN collezione_di_dischi c ON c.id=d.id_collezione_di_dischi
        WHERE c.visibilita=true
        AND a.nome_darte=nome_darte
        GROUP BY a.id);

    RETURN minutaggio_totale;

END$

DELIMITER ;
```

Funzionalità 12

Statistiche (una query per ciascun valore): numero di collezioni di ciascun collezionista, numero di dischi per genere nel sistema.

```
DROP PROCEDURE IF EXISTS  conta_collezioni;
DROP PROCEDURE IF EXISTS conta_dischi_per_genere;

DELIMITER $

CREATE PROCEDURE conta_collezioni()
BEGIN
    SELECT c.nickname as "Collezionista" ,
           count(cd.id) as "Numero di Collezioni"
    FROM collezione_di_dischi cd
    RIGHT JOIN collezionista c ON cd.id_collezionista=c.id
    GROUP BY c.nickname
    ORDER BY Collezionista ASC;

END$


CREATE PROCEDURE conta_dischi_per_genere()
BEGIN

    SELECT c.nome_genere as "Genere" ,
           count(c.id_disco) as "Numero di Dischi"
    FROM classificazione c
    GROUP BY c.nome_genere;

END$

DELIMITER ;
```

Funzionalità 13

Opzionalmente, dati un numero di barcode, un titolo e il nome di un autore, individuare tutti i dischi presenti nelle collezioni che sono più coerenti con questi dati (funzionalità utile, ad esempio, per individuare un disco già presente nel sistema prima di inserirne un doppione). L'idea è che il barcode è univoco, quindi i dischi con lo stesso barcode sono senz'altro molto coerenti, dopodichè è possibile cercare dischi con titolo simile e/o con l'autore dato, assegnando maggior punteggio di somiglianza a quelli che hanno più corrispondenze. Questa operazione può essere svolta con una stored procedure o implementata nell'interfaccia Java/PHP.

Implementata nell'interfaccia(vedere file "Query_JDBC")

Interfaccia verso il database

Si è scelto di costruire un'interfaccia scritta in Java con le librerie di JavaFX e progetto Maven, tutte le query sono state implementate nell'interfaccia.

Dichiarare come VM Arguments la seguente stringa:

--module-path "C:\java\javafx-sdk-18.0.1\lib" --add-modules javafx.controls,javafx.fxml --add-modules javafx.controls

Sostituire il path presente all'interno della stringa con il path della cartella dove si è scelto di salvare le librerie di JavaFX.

Il codice sorgente è allegato alla seguente documentazione nella cartella project

---

Raccomandazioni finali

- Questo documento è un modello che spero possa esservi utile per scrivere la documentazione finale del vostro progetto di Laboratorio di Basi di Dati.
- Cercate di includere tutto il codice SQL nella documentazione, come indicato in questo modello, per facilitarne la correzione. Potete comunque allegare alla documentazione anche il *dump* del vostro database o qualsiasi altro elemento che ritenete utile ai fini della valutazione.
- Ricordate che la documentazione deve essere consegnata, anche per email, almeno *una settimana prima* della data prevista per l'appello d'esame. Eventuali eccezioni a questa regola potranno essere concordate col docente.