Roster many-to-many Database

Course — logical key: course_title, primary key: course_id

course_title	course_id
Kendrick Lamar	1
Billie Eilish	2

belongs-to

Member — logical key: member_role, primary key: course_id

member_role	member_id	course_id	account_id
Damn	1	1	1
Hit Me Hard and Sof	2	2	2

belongs-to

User — logical key: user_name, primary key: account_id

user_name	account_id
DNA	1
Loyalty	2
Humble	3
Love	4
Bad Guy	5
You Should See Me in a Crown	6

CREATE DATABASE Roster
        DEFAULT CHARACTER SET UTF8MB4;

CREATE TABLE User (
        user_id INTEGER NOT NULL AUTO_INCREMENT,
        name VARCHAR (128),
        PRIMARY KEY (user_id)
) ENGINE = InnoDB CHARACTER SET UTF8MB4;

CREATE TABLE Course (
        course_id INTEGER NOT NULL AUTO_INCREMENT,
        title VARCHAR (128) UNIQUE,
        PRIMARY KEY (course_id)
) ENGINE = InnoDB CHARACTER SET UTF8MB4;

CREATE TABLE Member (
        role_id INTEGER),

```
        role VARCHAR (128),
        user_id INTEGER,
        course_id INTEGER,
        CONSTRAINT FOREIGN KEY (user_id) REFERENCES User (user_id)
                ON DELETE CASCADE ON UPDATE CASCADE,
        CONSTRAINT FOREIGN KEY (course_id) REFERENCES Course (course_id)
                ON DELETE CASCADE ON UPDATE CASCADE,
        PRIMARY KEY (user_id, course_id),
        INDEX USING BTREE (role)
) ENGINE = InnoDB CHARACTER SET UTF8MB4;

INSERT INTO User (name) VALUES ('Ramanas');
INSERT INTO User (name) VALUES ('Annie');
INSERT INTO User (name) VALUES ('Cedar');
INSERT INTO User (name) VALUES ('Cejay');
INSERT INTO User (name) VALUES ('Trassa');
INSERT INTO User (name) VALUES ('Trinity');
INSERT INTO User (name) VALUES ('Amberlouise');
INSERT INTO User (name) VALUES ('Hcen');
INSERT INTO User (name) VALUES ('Latif');
INSERT INTO User (name) VALUES ('Madison');
INSERT INTO User (name) VALUES ('Nirvana');
INSERT INTO User (name) VALUES ('Leigha');
INSERT INTO User (name) VALUES ('Nader');
INSERT INTO User (name) VALUES ('Prithvi');
INSERT INTO User (name) VALUES ('Reynold');

INSERT INTO Course (title) VALUES ('si106');
INSERT INTO Course (title) VALUES ('si110');
INSERT INTO Course (title) VALUES ('si206');

INSERT INTO Member (role_id, role) VALUES (0, 'student')
INSERT INTO Member (role_id, role) VALUES (1, 'instructor')

INSERT INTO Member (user_id, course_id, role) VALUES ('1', '1', '1');
INSERT INTO Member (user_id, course_id, role) VALUES ('2', '1', '0');
INSERT INTO Member (user_id, course_id, role) VALUES ('3', '1', '0');
INSERT INTO Member (user_id, course_id, role) VALUES ('4', '1', '0');
INSERT INTO Member (user_id, course_id, role) VALUES ('5', '1', '0');
INSERT INTO Member (user_id, course_id, role) VALUES ('6', '2', '1');
INSERT INTO Member (user_id, course_id, role) VALUES ('7', '2', '0');
```

```sql
INSERT INTO Member (user_id, course_id, role) VALUES ('8', '2', '0');
INSERT INTO Member (user_id, course_id, role) VALUES ('9', '2', '0');
INSERT INTO Member (user_id, course_id, role) VALUES ('10', '2', '0');
INSERT INTO Member (user_id, course_id, role) VALUES ('11', '3', '1');
INSERT INTO Member (user_id, course_id, role) VALUES ('12', '3', '0');
INSERT INTO Member (user_id, course_id, role) VALUES ('13', '3', '0');
INSERT INTO Member (user_id, course_id, role) VALUES ('14', '3', '0');
INSERT INTO Member (user_id, course_id, role) VALUES ('15', '3', '0');

SELECT User.name, Course.title, Member.role
        FROM User  JOIN Member JOIN Course
        ON User.user_id = Member.user_id
        AND Member.course_id = Course.course_id
        ORDER BY Course.title, Member.role, User.name
```

SELECT Track.track_title, Track.track_count, Track.track_time, Track.track_rating, Album.album_title, Artist.artist_name, Genre.genre_id
    FROM Track JOIN Genre JOIN Album JOIN Artist ON Track.genre_id = Genre.genre_id
    AND Track.album_id = Album.album_id AND Album.artist_ID = Artist.artist_id
    ORDER BY Album.album_title ASC, Track.track_count ASC;

SELECT DISTINCT Track.track_title, Track.track_count, Track.track_time, Track.track_rating, Album.album_title, Artist.artist_name, Genre.genre_name
    FROM Track JOIN Genre JOIN Album JOIN Artist ON Track.genre_id = Genre.genre_id
    AND Track.album_id = Album.album_id AND Album.artist_ID = Artist.artist_id
    WHERE Artist.artist_name = 'Kendrick Lamar'
    ORDER BY Album.album_title ASC, Track.track_count ASC;