ТЕХНИЧЕСКИ УНИВЕРСИТЕТ – СОФИЯ

Катедра „Компютърни системи”

КУРСОВ ПРОЕКТ

ПО БАЗИ ОТ ДАННИ

Студент: Стефан Моллов

ФАК. № 121221042 **Група:** 44б

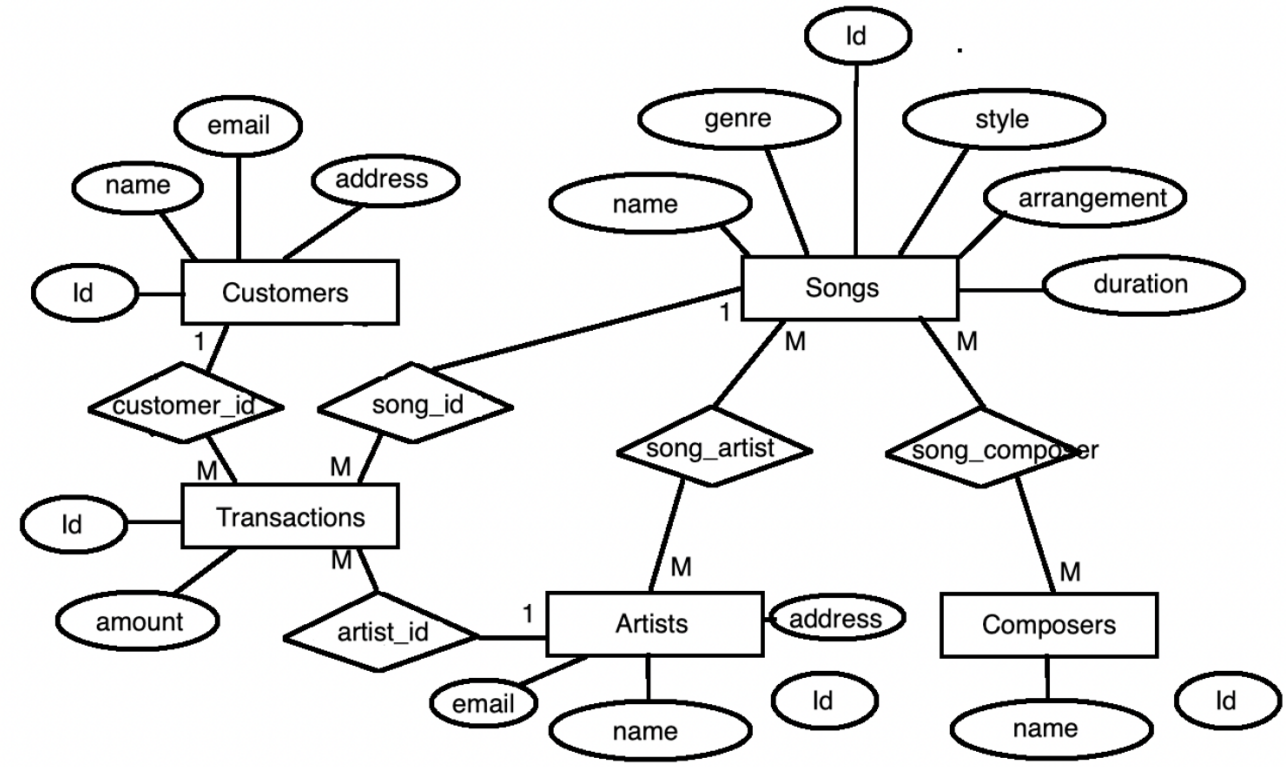
Тема № 25
Напишете база данни за сайт за продажба на песни. Да се пази основна информация за композитора, изпълнителя и песента – жанр, стил, аранжимент, продължителност. Една песен може да бъде написана от повече от един композитор и изпълнявана от повече от един изпълнител. Допълнете таблиците с необходима информация по ваш избор.

1. Да се проектира база от данни и да се представи ER диаграма със съответни CREATE TABLE заявки за средата MySQL.
2. Напишете заявка, в която демонстрирате SELECT с ограничаващо условие по избор.
3. Напишете заявка, в която използвате агрегатна функция и GROUP BY по ваш избор.
4. Напишете заявка, в която демонстрирате INNER JOIN по ваш избор.
5. Напишете заявка, в която демонстрирате OUTER JOIN по ваш избор.
6. Напишете заявка, в която демонстрирате вложен SELECT по ваш избор.
7. Напишете заявка, в която демонстрирате едновременно JOIN и агрегатна функция.
8. Създайте тригер по ваш избор.
9. Създайте процедура, в която демонстрирате използване на курсор.

Вашата работа трябва да включва: задание, ER-диаграма, CREATE TABLE заявки, всички останали заявки, решения на задачите от 2 до 9 и резултатите от тях.

1. Да се проектира база от данни и да се представи ER диаграма със съответни CREATE TABLE заявки за средата MySQL:

Основните обекти, за които трябва да съхраняваме информация, според заданието са: Songs, Composers, Artists. В таблица Customers се записва информацията за купувачите на песни. В таблица Transactions отразяваме всяка покупка направена в сайта. В нея ще се съдържа информация за сумата и номера на транзакцията, отбелязваща покупка от клиент на артист. За проектирането на базата ще използваме модела ER-диаграма (Entity Relationship Diagram):



Заявките, с които създаваме базата данни и таблиците:

```
DROP DATABASE IF EXISTS kursova;
CREATE DATABASE kursova;
USE kursova;

CREATE TABLE Composers (
id INT PRIMARY KEY,
name VARCHAR(255)
);
```

```
CREATE TABLE Artists (
id INT PRIMARY KEY,
name VARCHAR(255),
email VARCHAR(255),
address VARCHAR(255)
);

CREATE TABLE Customers (
id INT PRIMARY KEY,
name VARCHAR(255),
email VARCHAR(255),
address VARCHAR(255)
);
CREATE TABLE Songs (
id INT PRIMARY KEY,
name VARCHAR(255),
genre VARCHAR(255),
style VARCHAR(255),
arrangement VARCHAR(255),
duration INT
);

CREATE TABLE song_composer (
composer_id INT,
song_id INT,
FOREIGN KEY (composer_id) REFERENCES Composers(id),
FOREIGN KEY (song_id) REFERENCES Songs(id)
);

CREATE TABLE song_artist (
artist_id INT,
song_id INT,
FOREIGN KEY (artist_id) REFERENCES Artists(id),
FOREIGN KEY (song_id) REFERENCES Songs(id)
);

CREATE TABLE Transactions (
id INT PRIMARY KEY,
customer_id INT,
song_id INT,
artist_id INT,
amount DECIMAL(10, 2),
FOREIGN KEY (customer_id) REFERENCES Customers(id),
FOREIGN KEY (song_id) REFERENCES Songs(id),
FOREIGN KEY (artist_id) REFERENCES Artists(id)
);
```

Добавяме тестови данни в таблиците:

```
INSERT INTO Composers (id, name)
VALUES  (1, 'Иван Иванов'),
        (2, 'Андрей Ястребов'),
        (3, 'David Lee');

INSERT INTO Customers (id, name, email, address)
VALUES  (1, 'Мария Георгиева', 'maria@mail.com', 'ул. Липа 123'),
        (2, 'Петър Петров', 'peter@mail.com', 'ул. Бук 456'),
        (3, 'Анна Иванова', 'anna@mail.com', 'ул. Геран 789'),
        (4, 'Николай Николов', 'nikolay@mail.com', 'ул. Морава 321'),
        (5, 'Ивелина Георгиева', 'iveli@mail.com', 'ул. Топола 654'),
        (6, 'Георги Георгиев', 'georgi@mail.com', 'ул. Кестен 987'),
        (7, 'Мария Петрова', 'maria2@mail.com', 'ул. Бор 111'),
        (8, 'Павел Иванов', 'pavel@mail.com', 'ул. Туя 222'),
        (9, 'Елена Николова', 'elena@mail.com', 'ул. Бряст 3'),
        (10, 'Стефан Стефанов', 'stefan@mail.com', 'ул. Лира 444');

INSERT INTO Artists (id, name, email, address)
VALUES  (1, 'Георги Иванов', 'georgi@mail.com', 'ул. Иглика 555'),
        (2, 'Мария Павлова', 'maria@mail.com', 'ул. Бяла роза 666'),
        (3, 'Иван Николов', 'ivan@mail.com', 'ул. Синя вода 777'),
        (4, 'Елена Георгиева', 'elena@mail.com', 'ул. Зелен лес 888'),
        (5, 'Николай Петров', 'nikolay@mail.com', 'ул. Червен цвят 999'),
        (6, 'Анна Маринова', 'anna@mail.com', 'ул. Късмет 123'),
        (7, 'Димитър Николов', 'dimitar@mail.com', 'ул. Радост 456'),
        (8, 'Петър Иванов', 'peter@mail.com', 'ул. Щастие 789'),
        (9, 'Стефан Стоянов', 'stefan@mail.com', 'ул. Мечта 321'),
        (10, 'Ивелина Петрова', 'iveli@mail.com', 'ул. Магия 654');

INSERT INTO Songs (id, name, genre, style, arrangement, duration)
VALUES (1, 'Song D', 'Rock', 'Emotional', 'Original', 270),
        (2, 'Song E', 'Pop', 'Emotional', 'Original', 190),
        (3, 'Song F', 'Hip Hop', 'Energetic', 'Cover', 220),
        (4, 'Song G', 'Country', 'Uplifting', 'Original', 180),
        (5, 'Song H', 'Electronic', 'Upbeat', 'Original', 250),
        (6, 'Song I', 'R&B', 'Sensual', 'Cover', 210),
         (7, 'Song J', 'Indie', 'Melancholic', 'Original', 290),
        (8, 'Song K', 'Pop', 'Upbeat', 'Original', 200),
         (9, 'Song L', 'Rock', 'Energetic', 'Cover', 240),
        (10, 'Song M', 'Reggae', 'Relaxing', 'Original', 170);

INSERT INTO Transactions (id, customer_id, song_id, artist_id, amount)
VALUES (1, 3, 2, 2, 2.49),
        (2, 1, 2, 2, 4.49),
        (3, 6, 1, 6, 6.49),
        (4, 9, 9, 10, 1.49),
        (5, 3, 10, 10, 2.49),
        (6, 1, 7, 7, 2.49),
        (7, 6, 3, 3, 7.49),
        (8, 9, 4, 4, 1.49);
```

Задача 2. Напишете заявка, в която демонстрирате SELECT с ограничаващо условие по избор – Извеждаме информация само за песни, които са кавъри.

```
SELECT * FROM Song WHERE arrangement = 'Cover';
```

id	name	genre	style	arrangement	duration
3	Song C	Hip Hop	Energetic	Cover	220
6	Song F	R&B	Sensual	Cover	210
9	Song I	Rock	Energetic	Cover	240

Задача 3. Напишете заявка, в която използвате агрегатна функция и GROUP BY по ваш избор – Ще изведем броя на песните във всеки жанр, заедно със средната им продължителност.

```
SELECT genre, COUNT(*) AS song_count, AVG(duration) AS average_duration
FROM Songs
GROUP BY genre;
```

genre	song_count	average_duration
Rock	2	255.0000
Pop	2	195.0000
Hip Hop	1	220.0000
Country	1	180.0000
Electronic	1	250.0000
R&B	1	210.0000
Indie	1	290.0000
Reggae	1	170.0000

Задача 4. Напишете заявка, в която демонстрирате INNER JOIN по ваш избор – Ивеждаме всички песни заедно с имената на техните композитори.

```
SELECT Songs.name, Composers.name AS composer_name
FROM Songs
INNER JOIN song_composer ON Songs.id = song_composer.song_id
INNER JOIN Composers ON song_composer.composer_id = Composers.id
```

name	composer_name
Song A	Иван Иванов
Song C	Иван Иванов
Song F	Иван Иванов
Song I	Иван Иванов
Song E	Иван Иванов
Song G	Иван Иванов
Song H	Иван Иванов
Song F	Андрей Ястребов
Song C	Андрей Ястребов
Song G	Андрей Ястребов
Song J	Андрей Ястребов
Song F	Андрей Ястребов
Song D	Андрей Ястребов
Song E	Андрей Ястребов
Song D	David Lee
Song E	David Lee
Song H	David Lee
Song A	David Lee
Song I	David Lee
Song J	David Lee

Задача 5. Напишете заявка, в която демонстрирате OUTER JOIN по ваш избор – Извеждаме всички песни и информацията за тях, заедно с техните изпълнители, дори и все още да нямат обозначен изпълнител.

```
SELECT Artists.name, Songs.name AS song_name,
Songs.genre,
Songs.style,
Songs.arrangement,
Songs.duration
FROM Artists
RIGHT OUTER JOIN song_artist ON Artists.id = song_artist.artist_id
RIGHT OUTER JOIN Songs ON song_artist.song_id = Songs.id;
```

name	song_name	genre	style	arrangement	duration
Георги Иванов	Song A	Rock	Emotional	Original	270
Анна Маринова	Song A	Rock	Emotional	Original	270
Мария Павлова	Song B	Pop	Emotional	Original	190
Иван Николов	Song C	Hip Hop	Energetic	Cover	220
Елена Георгиева	Song D	Country	Uplifting	Original	180
Николай Петров	Song E	Electronic	Upbeat	Original	250
NULL	Song F	R&B	Sensual	Cover	210
Димитър Николов	Song G	Indie	Melancholic	Original	290
Петър Иванов	Song H	Pop	Upbeat	Original	200
Стефан Стоянов	Song I	Rock	Energetic	Cover	240
Ивелина Петрова	Song I	Rock	Energetic	Cover	240
Ивелина Петрова	Song J	Reggae	Relaxing	Original	170

Задача 6. Напишете заявка, в която демонстрирате вложен SELECT по ваш избор - Извеждаме името, имейла и адреса на лица чиито покупки надвишават средната стойност на всички покупки.

```
SELECT name, email, address
FROM Customers
WHERE id IN (
    SELECT customer_id
    FROM Transactions
    WHERE amount > (
        SELECT AVG(amount)
        FROM Transactions
    )
);
```

name	email	address
Мария Георгиева	maria@mail.com	ул. Липа 123
Георги Георгиев	georgi@mail.com	ул. Кестен 987

Задача 7. Напишете заявка, в която демонстрирате едновременно JOIN и агрегатна функция – Извеждаме общата сума на транзакции за всеки клиент, заедно с техните имена:

```
SELECT Customers.name, SUM(Transactions.amount) AS total_amount
FROM Customers
JOIN Transactions ON Customers.id = Transactions.customer_id
GROUP BY Customers.id, Customers.name;
```

name	total_amou...
Анна Иванова	2.49
Мария Георгиева	6.98
Георги Георгиев	13.98
Елена Николова	2.98
Ивелина Георгиева	2.49

Задача 8. Създайте тригер по ваш избор –

Ще създадем един тригер, който открива дали сумата на нововъведена транзакция е над 10. Ако е така, той ще актуализира сумата до 90% от първоначалната стойност (10% отстъпка), преди да вмъкне транзакцията:

```
DELIMITER //
CREATE TRIGGER apply_discount
BEFORE INSERT ON Transactions
FOR EACH ROW
BEGIN
    DECLARE total_amount DECIMAL(10, 2);

    SELECT SUM(amount) INTO total_amount
    FROM Transactions
    WHERE customer_id = NEW.customer_id;
```

```
    IF new.amount > 10 THEN
        SET NEW.amount = NEW.amount * 0.9;
    END IF;
END;
//
DELIMITER ;

-- Тест на тригера
INSERT INTO Transactions (id, customer_id, song_id, artist_id, amount)
VALUES (9, 1, 1, 1, 16.00);
SELECT * FROM Transactions WHERE id=9;
```

id	customer_id	song_id	artist_id	amount
9	1	1	1	14.40

Задача 9. Създайте процедура, в която демонстрирате използване на курсор. - Процедурата GetArtistSongs използва курсор, за да обхожда песните, свързани с конкретен изпълнител. Песните се извличат от таблиците song_artist и Songs с помощта на LEFT JOIN и резултатите се вмъкват във временна таблица, наречена ArtistSongs. Накрая песните за изпълнителя се избират от временната таблица и се извеждат.

```
DELIMITER //

CREATE PROCEDURE GetArtistSongs(IN artistName VARCHAR(255))
BEGIN
    DECLARE done INT DEFAULT FALSE;
    DECLARE artistId INT;
    DECLARE songId INT;
    DECLARE songName VARCHAR(255);
    DECLARE artistCursor CURSOR FOR
        SELECT Artists.id, Songs.id, Songs.name
        FROM Artists
        LEFT JOIN song_artist ON Artists.id = song_artist.artist_id
        LEFT JOIN Songs ON song_artist.song_id = Songs.id
        WHERE Artists.name = artistName;
    DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

    CREATE TEMPORARY TABLE IF NOT EXISTS ArtistSongs (
        artist_id INT,
        song_id INT,
        song_name VARCHAR(255)
    );


    OPEN artistCursor;

    read_loop: LOOP
        FETCH artistCursor INTO artistId, songId, songName;
        IF done THEN
            LEAVE read_loop;
        END IF;
```

```
        INSERT INTO ArtistSongs (artist_id, song_id, song_name)
        VALUES (artistId, songId, songName);
    END LOOP;

    CLOSE artistCursor;

    SELECT *
    FROM ArtistSongs;

    DROP TABLE IF EXISTS ArtistSongs;
END //
DELIMITER ;

-- Извикване на процедура
CALL GetArtistSongs('Ивелина Петрова');
```

artist_id	song_id	song_name
10	10	Song J
10	9	Song I