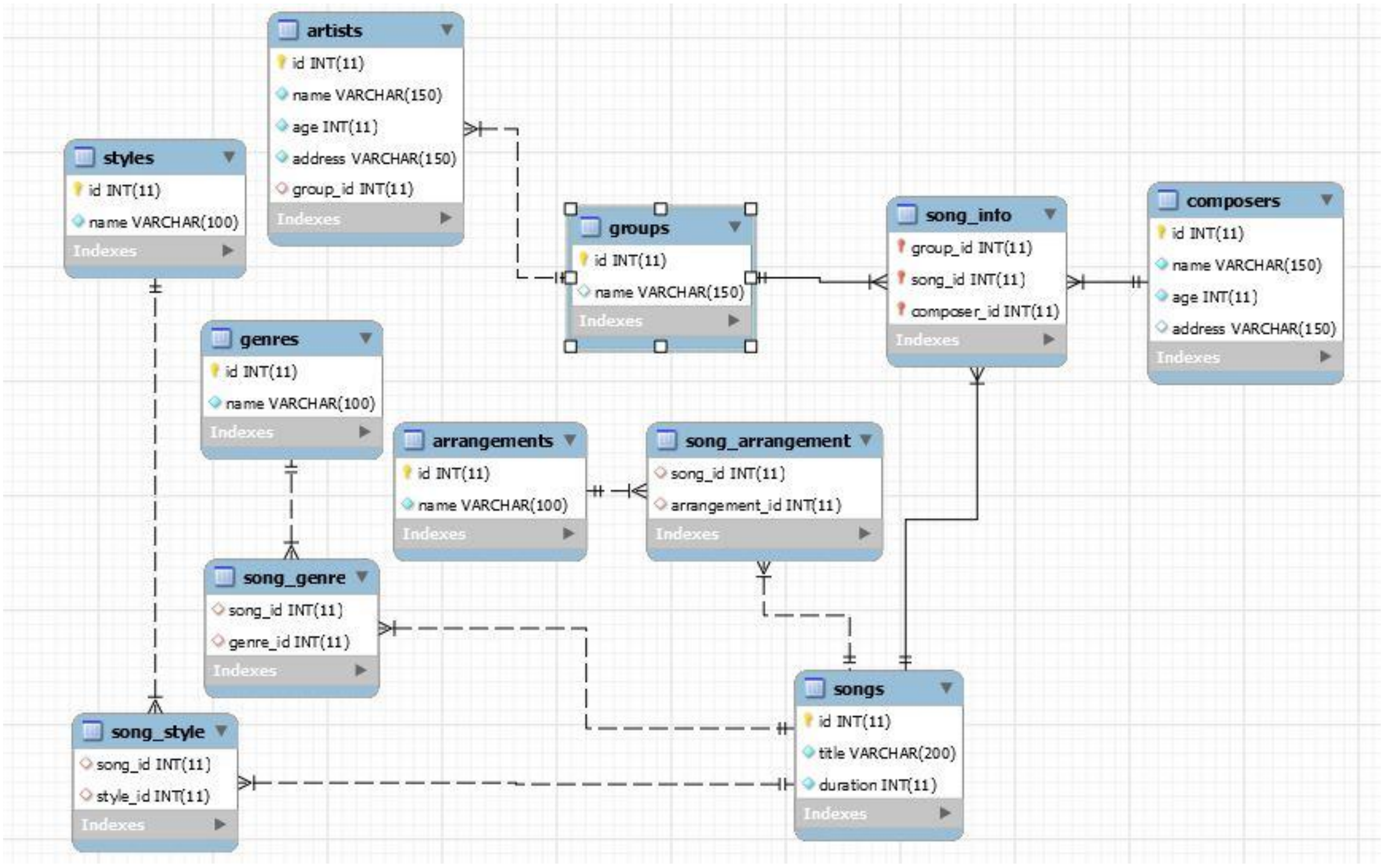
Решения

За решение на задачата ще използваме езика **MySQL**, който се изучава по време на лабораторните упражнения по дисциплината. За проектирането на базата в задача 1 ще използваме модела **ER**-диаграма (Entity Relationship Diagram).

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



Създаваме таблица groups:

```
CREATE TABLE groups (
id INT AUTO_INCREMENT PRIMARY KEY,
name VARCHAR(150)
);
```

Вкарваме примерни данни за groups:

```
INSERT INTO groups (name) values ('50 Cent'); -- ID = 1
INSERT INTO groups (name) values ('The White Stripes'); -- ID = 2
INSERT INTO groups (name) values ('Arctic Monkeys'); -- ID = 3
```

Създаваме таблица artists:

```
CREATE TABLE artists (
id INT AUTO_INCREMENT PRIMARY KEY,
name VARCHAR(150) NOT NULL,
age INT NOT NULL,
address VARCHAR(150) NOT NULL,
group_id INT,
CONSTRAINT FOREIGN KEY (group_id) REFERENCES groups(id)
);
```

Вкарваме примерни данни за artists:

```
/* инсерт заявки за артисти*/
INSERT INTO artists (name, age, address, group_id) values('50 Cent', 42, '50 Cent address', 1);
INSERT INTO artists (name, age, address, group_id) values('Jack White', 42, 'Jack White address', 1);
INSERT INTO artists (name, age, address, group_id) values('Meg White', 43, 'Meg White address', 2);
INSERT INTO artists (name, age, address, group_id) values('Alex Turner', 32, 'Alex Turner address', 3);
INSERT INTO artists (name, age, address, group_id) values('Matt Helders', 32, 'Matt Helders address',
3);
INSERT INTO artists (name, age, address, group_id) values('Jamie Cook', 32, 'Jamie Cook address', 3);
INSERT INTO artists (name, age, address, group_id) values('Nick O\'Malley', 32, 'Nick O\'Malley
address', 3);
```

Създаваме таблица composers:

```
CREATE TABLE composers (
id INT AUTO_INCREMENT PRIMARY KEY,
name VARCHAR(150) NOT NULL,
age INT NOT NULL,
address VARCHAR(150)
);
```

Вкарваме примерни данни за artists:

```
/* инсерт заявки за композитори*/

INSERT INTO composers (name, age, address) values('Lars Winther', 38, 'Lars Winther address');

INSERT INTO composers (name, age, address) values('Jack White', 42, 'Jack White address');

INSERT INTO composers (name, age, address) values('Alex Turner', 32, 'Alex Turner address');
```

Създаваме таблица songs:

```
CREATE TABLE songs (
id INT AUTO_INCREMENT PRIMARY KEY,
title VARCHAR(200) NOT NULL,
duration INT NOT NULL
);
```

Вкарваме примерни данни за songs:

```
/*инсерт заявки за песни*/

INSERT INTO songs (title, duration) VALUES ('Pilot', 182); -- ID = 1

INSERT INTO songs (title, duration) VALUES ('Animal Ambition', 200); -- ID = 2

INSERT INTO songs (title, duration) VALUES ('Seven Nation Army', 240); -- ID = 3

INSERT INTO songs (title, duration) VALUES ('Do I Wanna Know?', 272); -- ID = 4

INSERT INTO songs (title, duration) VALUES ('R U Mine?', 200); -- ID = 5

INSERT INTO songs (title, duration) VALUES ('One for the Road', 206); -- ID = 6

INSERT INTO songs (title, duration) VALUES ('Arabella', 207); -- ID = 7

INSERT INTO songs (title, duration) VALUES ('I Want It All', 184); -- ID = 8

INSERT INTO songs (title, duration) VALUES ('No.1 Party Anthem', 243); -- ID = 9

INSERT INTO songs (title, duration) VALUES ('Mad Sounds', 215); -- ID = 10

INSERT INTO songs (title, duration) VALUES ('Fireside', 181); -- ID = 11

INSERT INTO songs (title, duration) VALUES ('Why\'d You Only Call Me When You\'re High?', 222); --
ID = 12

INSERT INTO songs (title, duration) VALUES ('Snap Out of It', 357); -- ID = 13

INSERT INTO songs (title, duration) VALUES ('Knee Socks', 272); -- ID = 14

INSERT INTO songs (title, duration) VALUES ('I Wanna Be Yours', 184); -- ID = 15
```

Създаваме таблица song_info:

```
CREATE TABLE song_info (
group_id INT,
song_id INT,
composer_id INT,
CONSTRAINT FOREIGN KEY (group_id) REFERENCES groups(id),
CONSTRAINT FOREIGN KEY (song_id) REFERENCES songs(id),
CONSTRAINT FOREIGN KEY (composer_id) REFERENCES composers(id),
);
```

Вкарваме примерни данни за song_info:

```
/* INSERT заявки за свърване на песен към даден изпълнител и композитор */

INSERT INTO song_info VALUES (1, 1, 1);
INSERT INTO song_info VALUES (1, 2, 1);
INSERT INTO song_info VALUES (2, 3, 2);
INSERT INTO song_info VALUES (3, 4, 3);
INSERT INTO song_info VALUES (3, 5, 3);
INSERT INTO song_info VALUES (3, 6, 3);
INSERT INTO song_info VALUES (3, 7, 3);
INSERT INTO song_info VALUES (3, 8, 3);
INSERT INTO song_info VALUES (3, 9, 3);
INSERT INTO song_info VALUES (3, 10, 3);
INSERT INTO song_info VALUES (3, 11, 3);
INSERT INTO song_info VALUES (3, 12, 3);
INSERT INTO song_info VALUES (3, 13, 3);
INSERT INTO song_info VALUES (3, 14, 3);
INSERT INTO song_info VALUES (3, 15, 3);
```

Създаваме таблица arrangements:

```
CREATE TABLE arrangements (
id INT AUTO_INCREMENT PRIMARY KEY,
name VARCHAR(100) NOT NULL UNIQUE
);
```

Вкарваме примерни данни за arrangements:

```
/* INSERT заявки за добавяне на аранжименти */

INSERT INTO arrangements (name) VALUES ('Electronic'); -- ID = 1

INSERT INTO arrangements (name) VALUES ('Jazz'); -- ID = 2

INSERT INTO arrangements (name) VALUES ('Rock'); -- ID = 3
```

Създаваме таблица song_arrangement:

```
CREATE TABLE song_arrangement (
song_id INT,
arrangement_id INT,
CONSTRAINT FOREIGN KEY (song_id) REFERENCES songs(id),
CONSTRAINT FOREIGN KEY (arrangement_id) REFERENCES arrangements(id)
);
```

Вкарваме примерни данни за song_arrangement:

```
INSERT INTO song_arrangement (song_id, arrangement_id) VALUES (1, 1);
INSERT INTO song_arrangement (song_id, arrangement_id) VALUES (2, 2);
INSERT INTO song_arrangement (song_id, arrangement_id) VALUES (3, 3);
INSERT INTO song_arrangement (song_id, arrangement_id) VALUES (4, 3);
INSERT INTO song_arrangement (song_id, arrangement_id) VALUES (5, 3);
INSERT INTO song_arrangement (song_id, arrangement_id) VALUES (6, 3);
INSERT INTO song_arrangement (song_id, arrangement_id) VALUES (7, 3);
INSERT INTO song_arrangement (song_id, arrangement_id) VALUES (8, 3);
INSERT INTO song_arrangement (song_id, arrangement_id) VALUES (9, 3);
INSERT INTO song_arrangement (song_id, arrangement_id) VALUES (10, 3);
INSERT INTO song_arrangement (song_id, arrangement_id) VALUES (11, 3);
INSERT INTO song_arrangement (song_id, arrangement_id) VALUES (12, 3);
INSERT INTO song_arrangement (song_id, arrangement_id) VALUES (13, 3);
INSERT INTO song_arrangement (song_id, arrangement_id) VALUES (14, 3);
INSERT INTO song_arrangement (song_id, arrangement_id) VALUES (15, 3);
```

Създаваме таблица genres:

```
CREATE TABLE genres (
id INT AUTO_INCREMENT PRIMARY KEY,
name VARCHAR(100) NOT NULL UNIQUE
);
```

Вкарваме примерни данни за genres:

```
/* INSERT заявки за добавяне на жанрове */
INSERT INTO genres (name) VALUES ('Hip-Hop'); -- ID = 1
INSERT INTO genres (name) VALUES('Pop'); -- ID = 2
INSERT INTO genres (name) VALUES ('Rock'); -- ID = 3
```

Създаваме таблица song_genre:

```
CREATE TABLE song_genre (
song_id INT,
genre_id INT,
CONSTRAINT FOREIGN KEY (song_id) REFERENCES songs(id),
CONSTRAINT FOREIGN KEY (genre_id) REFERENCES genres(id)
);
```

Вкарваме примерни данни за song_genre:

```
/* INSERT заявки за свързване на песен към даден жанр */
INSERT INTO song_genre VALUES (1, 1);
INSERT INTO song_genre VALUES (2, 1);
INSERT INTO song_genre VALUES (3, 2);
INSERT INTO song_genre VALUES (4, 3);
INSERT INTO song_genre VALUES (5, 3);
INSERT INTO song_genre VALUES (6, 3);
INSERT INTO song_genre VALUES (7, 3);
INSERT INTO song_genre VALUES (8, 3);
INSERT INTO song_genre VALUES (9, 3);
INSERT INTO song_genre VALUES (10, 3);
INSERT INTO song_genre VALUES (11, 3);
INSERT INTO song_genre VALUES (12, 3);
INSERT INTO song_genre VALUES (13, 3);
INSERT INTO song_genre VALUES (14, 3);
INSERT INTO song_genre VALUES (15, 3);
```

Създаваме таблица styles:

```
CREATE TABLE styles (
id INT AUTO_INCREMENT PRIMARY KEY,
name VARCHAR(100) NOT NULL UNIQUE
);
```

Вкарваме примерни данни за styles:

```
INSERT INTO styles (name) VALUES ('Dance'); -- ID = 1
INSERT INTO styles (name) VALUES('Sad'); -- ID = 2
INSERT INTO styles (name) VALUES('Funky'); -- ID = 3
INSERT INTO styles (name) VALUES('Deep'); -- ID = 4
```

Създаваме таблица song_style:

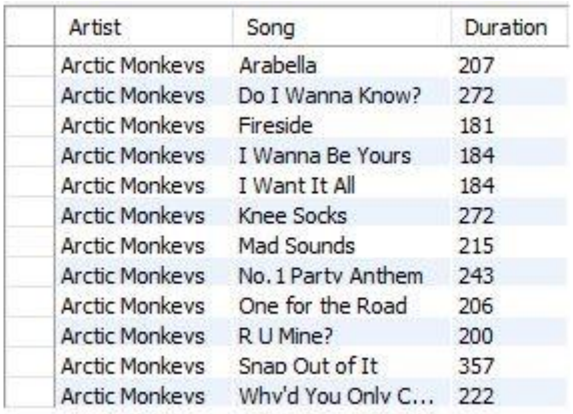
```
CREATE TABLE song_style (
song_id INT,
style_id INT,
CONSTRAINT FOREIGN KEY (song_id) REFERENCES songs(id),
CONSTRAINT FOREIGN KEY (style_id) REFERENCES styles(id)
);
```

Вкарваме примерни данни за song_style:

```
/* INSERT заявки за свързване на песен към даден стил */
INSERT INTO song_style VALUES (1, 1);
INSERT INTO song_style VALUES (2, 1);
INSERT INTO song_style VALUES (3, 3);
INSERT INTO song_style VALUES (4, 4);
INSERT INTO song_style VALUES (5, 1);
INSERT INTO song_style VALUES (6, 2);
INSERT INTO song_style VALUES (7, 3);
INSERT INTO song_style VALUES (8, 2);
INSERT INTO song_style VALUES (9, 3);
INSERT INTO song_style VALUES (10, 4);
INSERT INTO song_style VALUES (11, 4);
INSERT INTO song_style VALUES (12, 4);
INSERT INTO song_style VALUES (13, 3);
INSERT INTO song_style VALUES (14, 1);
INSERT INTO song_style VALUES (15, 3);
```

Задача 2: Заявката извежда имената на групите, имената на песните и продължителността им. Ограничаващото условие е групата да е с име "Arctic Monkeys".

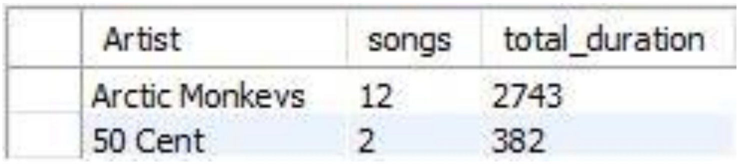
```
SELECT groups.name AS Artist, songs.title AS Song, songs.duration AS Duration
FROM groups
JOIN songs ON groups.id IN (
	SELECT group_id
	FROM song_info
	WHERE song_id = songs.id)
WHERE groups.name LIKE '%Arctic Monkeys%'
ORDER BY songs.title;
```



Artist	Song	Duration
Arctic Monkeys	Arabella	207
Arctic Monkeys	Do I Wanna Know?	272
Arctic Monkeys	Fireside	181
Arctic Monkeys	I Wanna Be Yours	184
Arctic Monkeys	I Want It All	184
Arctic Monkeys	Knee Socks	272
Arctic Monkeys	Mad Sounds	215
Arctic Monkeys	No. 1 Party Anthem	243
Arctic Monkeys	One for the Road	206
Arctic Monkeys	R U Mine?	200
Arctic Monkeys	Snap Out of It	357
Arctic Monkeys	Why'd You Only C...	222

Задача 3: Заявката извежда имената на групите, броя на песните на всяка една от тях и сумира продължителността им, ако броя на песните им е повече от една. Резултатите са подредени по максимален брой песни.

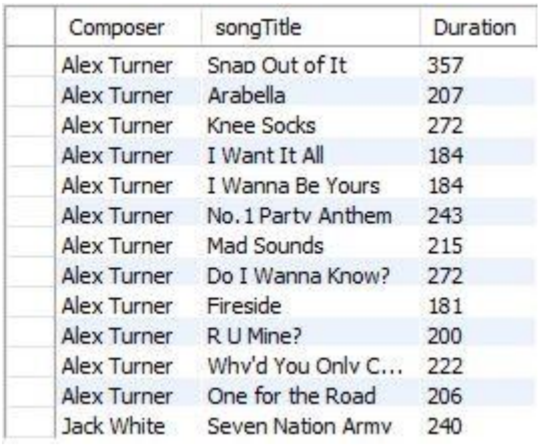
```
SELECT groups.name AS Artist, COUNT(songs.id) AS songs, SUM(songs.duration) AS total_duration
FROM groups
JOIN songs ON groups.id IN (
	SELECT group_id
	FROM song_info
	WHERE song_id = songs.id)
GROUP by groups.name
HAVING songs > 1
ORDER BY songs DESC;
```



Artist	songs	total_duration
Arctic Monkeys	12	2743
50 Cent	2	382

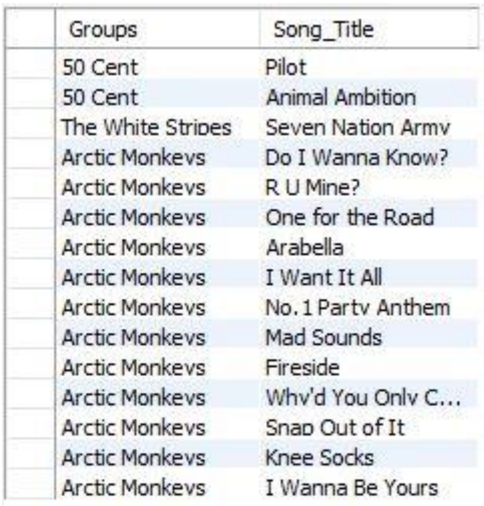
Задача 4: Заявката извежда имена на групите, композиторите и имената на песните, които те изпълняват/композират.

```
SELECT composers.name AS Composer, songs.title AS songTitle, songs.duration AS Duration
FROM composers
INNER JOIN song_info ON composers.id = song_info.composer_id
INNER JOIN songs ON songs.id = song_info.song_id
INNER JOIN groups ON groups.id = song_info.group_id
INNER JOIN artists ON artists.group_id = groups.id
WHERE composers.name = artists.name
ORDER BY Composer ASC;
```



Composer	songTitle	Duration
Alex Turner	Snap Out of It	357
Alex Turner	Arabella	207
Alex Turner	Knee Socks	272
Alex Turner	I Want It All	184
Alex Turner	I Wanna Be Yours	184
Alex Turner	No. 1 Party Anthem	243
Alex Turner	Mad Sounds	215
Alex Turner	Do I Wanna Know?	272
Alex Turner	Fireside	181
Alex Turner	R U Mine?	200
Alex Turner	Why'd You Only C...	222
Alex Turner	One for the Road	206
Jack White	Seven Nation Army	240

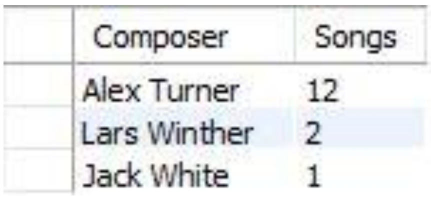
```
SELECT groups.name AS Groups, songs.title AS Song_Title
FROM groups
LEFt JOIN songs ON groups.id IN (
        SELECT group_id
        FROM song_info
        WHERE song_id = songs.id)
```



Groups	Song_Title
50 Cent	Pilot
50 Cent	Animal Ambition
The White Stripes	Seven Nation Army
Arctic Monkeys	Do I Wanna Know?
Arctic Monkeys	R U Mine?
Arctic Monkeys	One for the Road
Arctic Monkeys	Arabella
Arctic Monkeys	I Want It All
Arctic Monkeys	No. 1 Party Anthem
Arctic Monkeys	Mad Sounds
Arctic Monkeys	Fireside
Arctic Monkeys	Why'd You Only C...
Arctic Monkeys	Snap Out of It
Arctic Monkeys	Knee Socks
Arctic Monkeys	I Wanna Be Yours

Задача 5: Заявката извежда име на композитор и брой песни, които е композирал.

```
SELECT composers.name AS Composer, Count(songs.id) AS Songs
FROM composers
JOIN song_info ON composers.id = song_info.composer_id
JOIN songs ON song_info.song_id = songs.id
GROUP BY composers.name
ORDER BY songs DESC;
```

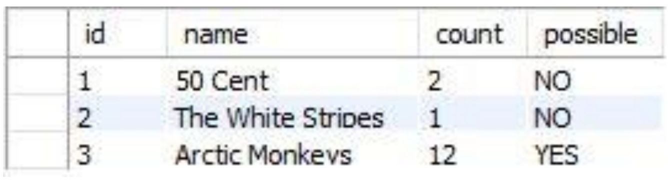


Composer	Songs
Alex Turner	12
Lars Winther	2
Jack White	1

Задача 6: В процедурата декларираме променливи. След това и курсорът, който взима име на група и броя на песните, които има групата. След това се създава празна таблица, в която пазим резултата. Итерираме през данните от курсора и ги записваме във временната таблица, ако условието група да има повече от 11 песни, тя може да създаде албум. Взимаме информацията от временната таблица и я изтриваме.

```
USE song_seller;
DROP PROCEDURE IF EXISTS CursorTask;
DELIMITER $$
CREATE PROCEDURE CursorTask()
BEGIN
DECLARE finished INT;
DECLARE tempGroupName VARCHAR(150);
DECLARE tempSongsCount INT;
DECLARE tempCanHaveAlbum VARCHAR(10);
DECLARE id INT;

DECLARE AlbumCursor CURSOR FOR
SELECT groups.name , COUNT(songs.title)
FROM groups
JOIN songs ON groups.id IN (
            SELECT group_id
   FROM song_info
   WHERE songs.id = song_info.song_id)
GROUP BY groups.id;

DECLARE CONTINUE handler FOR NOT FOUND SET finished =1;

SET finished = 0;
SET id = 0;
```

```
DROP TABLE IF EXISTS TempAlbumInfo;
CREATE TEMPORARY TABLE TempAlbumInfo(
            id INT PRIMARY KEY AUTO_INCREMENT,
   name VARCHAR(150),
   count INT,
   possible VARCHAR(10)
) ENGINE = Memory;

OPEN AlbumCursor;
songs_loop: WHILE(finished=0)
DO
FETCH AlbumCursor INTO tempGroupName, tempSongsCount;
IF(finished =0)
THEN
            SET id = id + 1;
ELSE
            LEAVE songs_loop;
END IF;
IF(tempSongsCount>11)
THEN
            SET tempCanHaveAlbum = 'YES';
            INSERT INTO TempAlbumInfo VALUES (id, tempGroupName, tempSongsCount,
tempCanHaveAlbum);
ELSE
            SET tempCanHaveAlbum = 'NO';
            INSERT INTO TempAlbumInfo VALUES (id, tempGroupName, tempSongsCount,
tempCanHaveAlbum);
 END IF;

 END WHILE;
 CLOSE AlbumCursor;
 SELECT * FROM tempAlbumInfo;
 DROP TABLE tempAlbumInfo;
 END
 $$
 DELIMITER ;

CALL CursorTask();
```



	id	name	count	possible
	1	50 Cent	2	NO
	2	The White Stripes	1	NO
	3	Arctic Monkeys	12	YES