HW 03

Creating MusicVideo Table

```
DROP TABLE IF EXISTS MusicVideo;
CREATE TABLE MusicVideo (
        TrackId INTEGER PRIMARY KEY,
        VideoDirector TEXT,
        FOREIGN KEY (TrackId) REFERENCES tracks(TrackID)
);
```

Insert 10 videos

```
INSERT INTO MusicVideo (TrackId, VideoDirector) VALUES (1, "Director 001");
INSERT INTO MusicVideo (TrackId, VideoDirector) VALUES (2, "Director 002");
INSERT INTO MusicVideo (TrackId, VideoDirector) VALUES (3, "Director 003");
INSERT INTO MusicVideo (TrackId, VideoDirector) VALUES (4, "Director 004");
INSERT INTO MusicVideo (TrackId, VideoDirector) VALUES (5, "Director 005");

INSERT INTO MusicVideo (TrackId, VideoDirector) VALUES (6, "Director 006");
INSERT INTO MusicVideo (TrackId, VideoDirector) VALUES (7, "Director 007");
INSERT INTO MusicVideo (TrackId, VideoDirector) VALUES (8, "Director 008");
INSERT INTO MusicVideo (TrackId, VideoDirector) VALUES (9, "Director 009");
INSERT INTO MusicVideo (TrackId, VideoDirector) VALUES (10, "Director 010");
```

	TrackId	VideoDirector
	Filter	Filter
1	1	Director 001
2	2	Director 002
3	3	Director 003
4	4	Director 004
5	5	Director 005
6	6	Director 006
7	7	Director 007
8	8	Director 008
9	9	Director 009
10	10	Director 010

Insert Voodoo to MusicVideo without using TrackID

There is an existing Voodoo tracks by using

```
SELECT TrackId FROM tracks WHERE Name = "Voodoo";
```

```
INSERT INTO MusicVideo (TrackId, VideoDirector)
        SELECT TrackId,  "Director Voodoo"
        FROM  tracks
        WHERE Name == "Voodoo";
```

	TrackId	VideoDirector
	Filter	Filter
1	1	Director 001
2	2	Director 002
3	3	Director 003
4	4	Director 004
5	5	Director 005
6	6	Director 006
7	7	Director 007
8	8	Director 008
9	9	Director 009
10	10	Director 010
11	175	Director Voodoo

Write a query that lists all the customers that listen to longer-than-average tracks, excluding the tracks that are longer than 15 minutes.

1. Calculate the avg duration

```
SELECT AVG(Milliseconds) AS avg_length
FROM tracks;
```

2. Create a intermediate view to filter the tracks that is either shorter than the average or longer than 15 minutes.

```
DROP VIEW IF EXISTS FilteredTracks;
CREATE VIEW FilteredTracks AS
SELECT DISTINCT TrackId, Milliseconds
FROM tracks
```

```
WHERE Milliseconds > (SELECT AVG(Milliseconds) FROM tracks)
AND Milliseconds <= 60*1000*15;
```

3. Select distinct customers with their first name and last name

```
SELECT DISTINCT
        customers.FirstName AS firstname,
        customers.LastName AS  lastname
FROM FilteredTracks
        JOIN invoice_items USING(TrackId)
        JOIN invoices USING(InvoiceId)
        JOIN customers USING(CustomerId);
```

	firstname	lastname
1	Mark	Philips
2	Ladislav	Kovács
3	John	Gordon
4	Leonie	Köhler
5	Michelle	Brooks
6	Joakim	Johansson
7	Aaron	Mitchell
8	Kathy	Chase
9	Frank	Ralston
10	Hannah	Schneider
11	Enrique	Muñoz
12	Alexandre	Rocha
13	Martha	Silk
14	Camille	Bernard
15	Dan	Miller
16	Puja	Srivastava
17	Eduardo	Martins
18	Stanisław	Wójcik
19	Niklas	Schröder
20	Marc	Dubois
21	Heather	Leacock
22	Julia	Barnett
23	Dominique	Lefebvre

```
Execution finished without errors.
Result: 56 rows returned in 20ms
At line 42:
-- 3. Select distinct customers with their first name and last name
SELECT DISTINCT
        customers.FirstName AS firstname,
        customers.LastName AS  lastname
FROM FilteredTracks
```

Write a query that lists all the tracks that are not in one of the top 5 genres with longer duration in the database.

1. Calculate the average duration of each genre

```
DROP VIEW IF EXISTS genre_track_length_ordered;
CREATE VIEW genre_track_length_ordered AS
        SELECT genres.Name AS name,  genres.GenreId AS gid,
AVG(tracks.Milliseconds) AS avg_length
```

```
        FROM tracks
                JOIN genres USING(GenreId)
        GROUP by genres.GenreId;
```

2: Identify the top 5 genres with the longest average duration

```
DROP VIEW IF EXISTS top_5_longest_genres;
CREATE VIEW top_5_longest_genres AS
        SELECT gid
        FROM genre_track_length_ordered
        ORDER by avg_length DESC
        LIMIT 5;
```

3: Filter out tracks that belong to one of these top 5 genres

```
SELECT tracks.TrackId, tracks.Name, tracks.Milliseconds
FROM tracks
WHERE tracks.GenreId NOT IN (SELECT gid FROM top_5_longest_genres)
```

	TrackId	Name	Milliseconds
1	1	For Those About To Rock (We Salute You)	343719
2	2	Balls to the Wall	342562
3	3	Fast As a Shark	230619
4	4	Restless and Wild	252051
5	5	Princess of the Dawn	375418
6	6	Put The Finger On You	205662
7	7	Let's Get It Up	233926
8	8	Inject The Venom	210834
9	9	Snowballed	203102
10	10	Evil Walks	263497
11	11	C.O.D.	199836
12	12	Breaking The Rules	263288
13	13	Night Of The Long Knives	205688
14	14	Spellbound	270863
15	15	Go Down	331180
16	16	Dog Eat Dog	215196
17	17	Let There Be Rock	366654
18	18	Bad Boy Boogie	267728
19	19	Problem Child	325041
20	20	Overdose	369319
21	21	Hell Ain't A Bad Place To Be	254380
22	22	Whole Lotta Rosie	323761
23	23	Walk On Water	295680

```
Execution finished without errors.
Result: 3290 rows returned in 41ms
At line 68:
-- 3: Filter out tracks that belong to one of these top 5 genres
SELECT tracks.TrackId, tracks.Name, tracks.Milliseconds
FROM tracks
```

Define an insightful query on your own that involves at least three tables

Write a query to get a list of tracks that are part of invoices with a total billing amount over $5.

1. Calculate the average track length in the database

2. Identify invoices with a total billing amount over 5

```sql
SELECT
    invoices.Total AS BillingAmount,
    tracks.Name AS TrackName,
    tracks.Milliseconds * 1.0 / (1000 * 60) AS TrackLengthInMinutes
FROM invoices
JOIN invoice_items USING(InvoiceId)
JOIN tracks USING (TrackId)
WHERE invoices.Total > 5
    AND tracks.Milliseconds < (SELECT AVG(Milliseconds) FROM tracks)
    AND tracks.Milliseconds > 3 * 60 * 1000
ORDER BY TrackLengthInMinutes;
```

	BillingAmount	TrackName	TrackLengthInMinutes
1	8.91	Promises	3.00668333333333
2	13.86	On Fire	3.0106
3	13.86	Fascinação	3.01321666666667
4	13.86	Cafezinho	3.0154
5	8.91	Ponto De Interrogação	3.01583333333333
6	13.86	Can't Stand Losing You	3.01931666666667
7	8.91	Comportamento Geral	3.02628333333333
8	13.86	Comportamento Geral	3.02628333333333
9	13.86	Um Love	3.02671666666667
10	8.91	Basket Case	3.02715
11	5.94	Carolina	3.0302
12	13.86	Pinball Wizard	3.0315
13	16.86	Atomic Punk	3.03455
14	9.91	Put Your Head Out	3.03716666666667
15	5.94	Minha Historia	3.0376
16	13.86	No Futuro	3.03846666666667
17	5.94	My Time After Awhile	3.04151666666667
18	13.86	We Are The Champions	3.04805
19	13.86	Tempos Modernos	3.0511
20	5.94	Sweetest Thing	3.0511
21	8.91	Sweetest Thing	3.0511
22	10.91	Caras Como Eu	3.05153333333333
23	8.91	Mensagem De Amor (2000)	3.0598

```
Execution finished without errors.
Result: 1235 rows returned in 13ms
At line 1:
--
-- DROP TABLE IF EXISTS MusicVideo;
-- CREATE TABLE MusicVideo (
--             TrackId INTEGER PRIMARY KEY,
```