

**LAPORAN DESAIN DAN IMPLEMENTASI DATABASE
LAYANAN STREAMING MUSIK PADA APLIKASI SPOTITIP V2**



Disusun Oleh :

Kelompok 10

Muh Adhim Rahman Rusdi

D121221095

Raihan Miftah Andara Rum

D121221101

Faisol Akbar

T202310151

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS HASANUDDIN

2023/2024

DAFTAR ISI

JUDUL.....	0
DAFTAR ISI.....	1
BAB I : NARASI.....	2
A. Latar Belakang.....	2
BAB II : MEMBUAT ERD.....	3
A. Penjelasan.....	3
B. Entity Relationship Diagram.....	3
BAB III : HASIL PEMERIKSAAN ERD TIM LAIN.....	4
BAB IV : KAMUS DATA.....	6
BAB V : TABEL DAN POPULASINYA.....	7
BAB VI : QUERY DAN ALJABAR RELASINYA.....	9
A. Query Pembuatan Tabel.....	9
B. Query Pengisian Data pada Tabel Menggunakan File.....	13
C. Query Pengecekan Data Tabel.....	13
D. Query Aljabar Relasional pada Tabel.....	14
BAB VII : QUERY OPTIMASI.....	15
1. Apa itu optimasi query?.....	15
2. Apa saja tujuan optimasi query?.....	15
3. Bagaimana contoh teknik optimasi query?.....	15
4. Berikan contoh optimasi query di MySQL.....	15
5. Implementasikan lima dari teknik tersebut dalam database MySQL.....	18

BAB I

NARASI

A. Latar Belakang

Aplikasi Spotitip adalah sebuah platform layanan streaming musik yang bersifat online, terdapat hubungan timbal balik secara tidak langsung antara pendengar atau dengan kata lain sebagai pengguna aplikasi atau layanan tersebut dengan artis yang bersangkutan. Aplikasi Spotitip menawarkan pengalaman mendengarkan musik yang interaktif dan memungkinkan pengguna untuk terhubung dengan artis, album, playlist, dan lagu-lagu favorit mereka.

Mengakses aplikasi ini cukup mudah dimana pengguna hanya perlu melakukan instalasi. Setelah mendownload aplikasi Spotitip, pengguna diminta untuk login jika sudah memiliki akun sebelumnya dan membuat akun jika belum memiliki akun. Setelah memiliki akun, pengguna dapat mengakses berbagai fitur dan fungsi yang ditawarkan oleh aplikasi Spotitip. Dimana pengguna dapat mengakses dan memutar lagu di seluruh dunia kapanpun dan dimanapun. Pengguna juga dapat menyukai lagu-lagu yang disukai.

Lagu-lagu yang disukai pengguna dapat diakses oleh pengguna pada section bernama Suka. Bukan hanya lagu, pengguna dapat mengikuti artis-artis atau penyanyi kesukaannya agar tidak ketinggalan info terbaru dari artis favorit. Setelah memiliki artis dan lagu yang disukai, pengguna dapat menyimpannya ke dalam playlist. Dimana playlist juga harus dibuat oleh pengguna terlebih dahulu.

Dengan begitu, pengguna memiliki banyak opsi untuk mengakses lagu-lagu atau artis dari lagu yang ingin pengguna dengarkan dimanapun dan kapanpun. User memiliki preferensi mereka masing-masing, dan aplikasi Spotitip akan menyesuaikan preferensi user berdasarkan artis yang mereka ikuti, koneksi user dengan user lainnya, dan lagu yang mereka dengarkan. Masing-masing lagu memiliki genre yang berbeda juga. Oleh karena itu, aplikasi Spotitip memberikan konten sesuai keinginan User.

Artis juga memiliki langkah pembuatan akun yang sama dengan pengguna biasa. Fitur yang bisa diakses oleh pengguna bisa juga diakses oleh Artis. Tetapi, artis bisa mempublikasikan lagu kemudian memasukkannya ke dalam album. Lagu-lagu tersebut bisa didengarkan dan disukai oleh pengguna lainnya. Lagu-lagu tersebut menjadi komponen di dalam playlist yang dibuat oleh pengguna.

BAB II

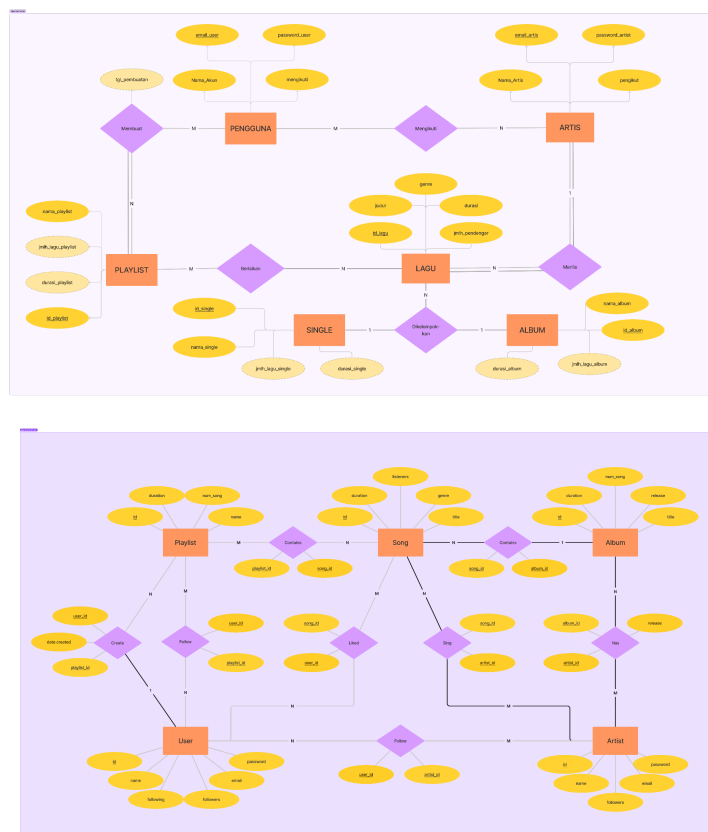
MEMBUAT ERD

A. Penjelasan

Sesuai dengan narasi di atas dan gambar di bawah, Entitas utama yang terdapat dalam ERD ini adalah "User", "Artist", "Album", "Song", dan "Playlist". Masing-masing entitas memiliki peran dan fungsi penting dalam merepresentasikan informasi dan interaksi dalam basis data tersebut. Berikut adalah daftar entitas dari ERD aplikasi Spotitip.

Setiap entitas tersebut dihubungkan melalui beberapa relasi. Berikut adalah relasi yang ada. Relasi "Follow" dan "Create" menjalin koneksi antara entitas "User" dan "Playlist", Entitas "User" memiliki relasi "Follow" yang menghubungkan ke entitas "Artist" dan relasi "Liked" yang menghubungkan ke entitas "Song". Entitas "Artist" memiliki relasi "Sing" yang menghubungkan ke entitas "Song" dan relasi "Has" yang menghubungkan ke entitas "Album". Kemudian entitas "Album" memiliki relasi "Contains" yang menghubungkan ke entitas "Song". Terakhir yaitu relasi "Contains" yang menghubungkan entitas "Song" dan entitas "Playlist".

B. Entity Relationship Diagram



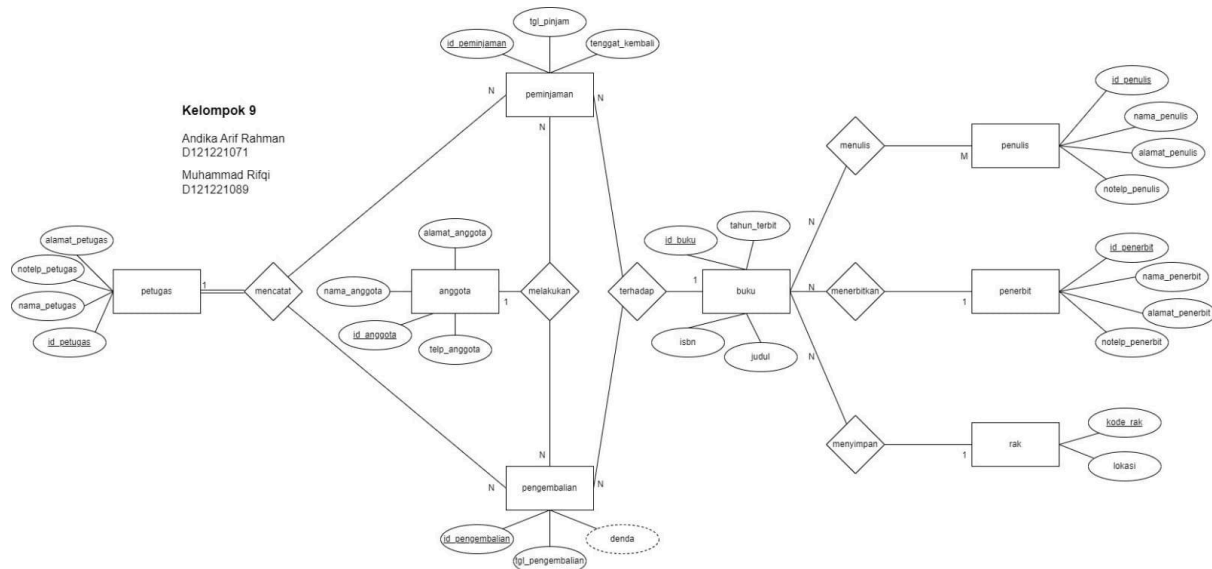
[LINK ERD SPOTITIP V1](#) & [LINK ERD SPOTITIP V2](#)

BAB III

HASIL PEMERIKSAAN ERD TIM LAIN

ERD dari Kelompok 9 : Muhammad Rifqi (D121221089)

Penilaian dari Kelompok 11 : Muh. Adhim Rahman Rusdi (D121221095)



1. Apakah semua entitas sudah ada?

= Menurut saya entitas yang tersedia sudah lengkap, dimana terdapat entitas **anggota** sebagai main character, **petugas** sebagai jembatan **anggota** dalam melakukan **peminjaman** maupun **pengembalian** berupa **buku** di perpustakaan.

2. Apakah atribut entitas sudah benar?

= Menurut saya atribut pada entitas sudah benar, dimana setiap atribut telah **ditempatkan** pada entitasnya dengan tepat. Tidak hanya penempatan, **penamaan** atribut dan **penentuan unique key** juga sudah benar.

3. Apakah semua relasi (antar entitas) sudah ada?

= Semua hubungan antar entitas (relasi) sudah ada, dimana setiap hubungan antar dua entitas memiliki satu relasi diantaranya.

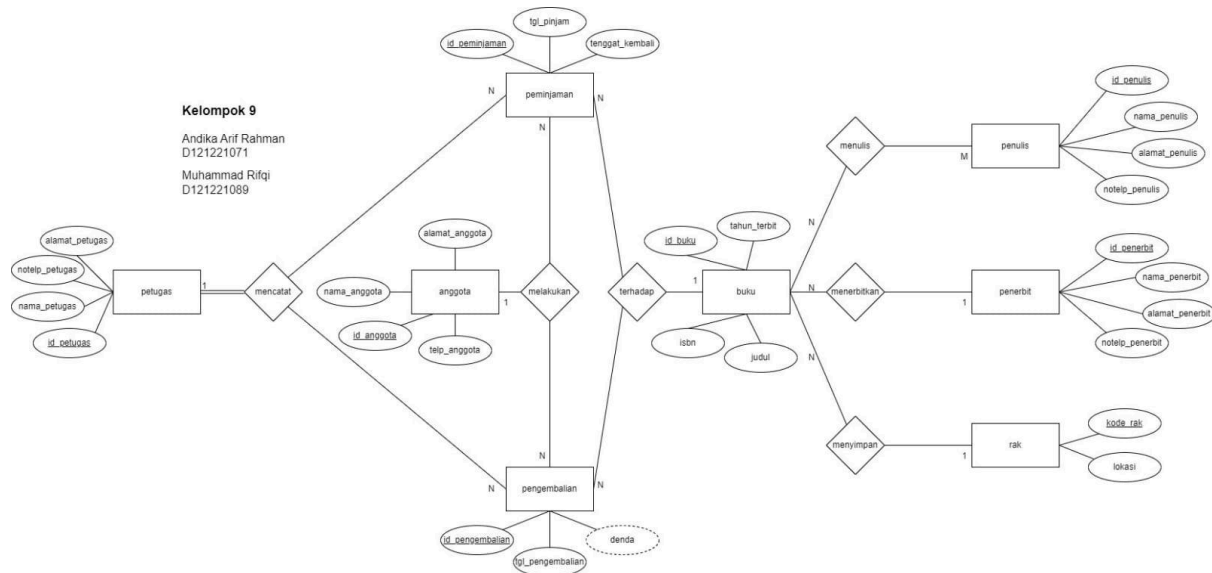
4. Apakah atribut relasi sudah ada / benar?

= Semua relasi pada ERD di atas tidak memiliki atribut, karena berbeda dengan entitas, relasi sendiri tidak memiliki keharusan dalam memiliki atribut tergantung dengan konteks dan

kebutuhannya. Tetapi menurut pengamatan saya mengapa tidak terdapat atribut pada relasi di atas, karena terdapat 2 entitas yang biasanya dijadikan sebagai relasi yaitu “pengembalian” dan “peminjaman” yang dijadikan sebagai entitas dan itu bukanlah sebuah masalah.

ERD dari Kelompok 9 : Andika Arif Rahman (D121221089)

Penilaian dari Kelompok 11 : Raihan Miftah Andara Rum (D121221095)



1. Apakah semua entitas sudah ada?

= Ya, semua entitas sudah ada, yaitu “Penulis”, “Penerbit”, “Rak”, “Buku”, “Pengembalian”, “Peminjaman”, “Anggota”, Dan Petugas”.

2. Apakah atribut entitas sudah benar?

= Ya, atribut entitas sudah benar. Atribut “denda” pada entitas “Pengembalian” merupakan atribut derivatif yang dihasilkan dari “tgl_pengembalian” dan berarti denda akan diterapkan jika tanggal pengembalian buku melewati tenggat kembali peminjaman. Selain itu, atribut kunci pada semua entitas sudah benar.

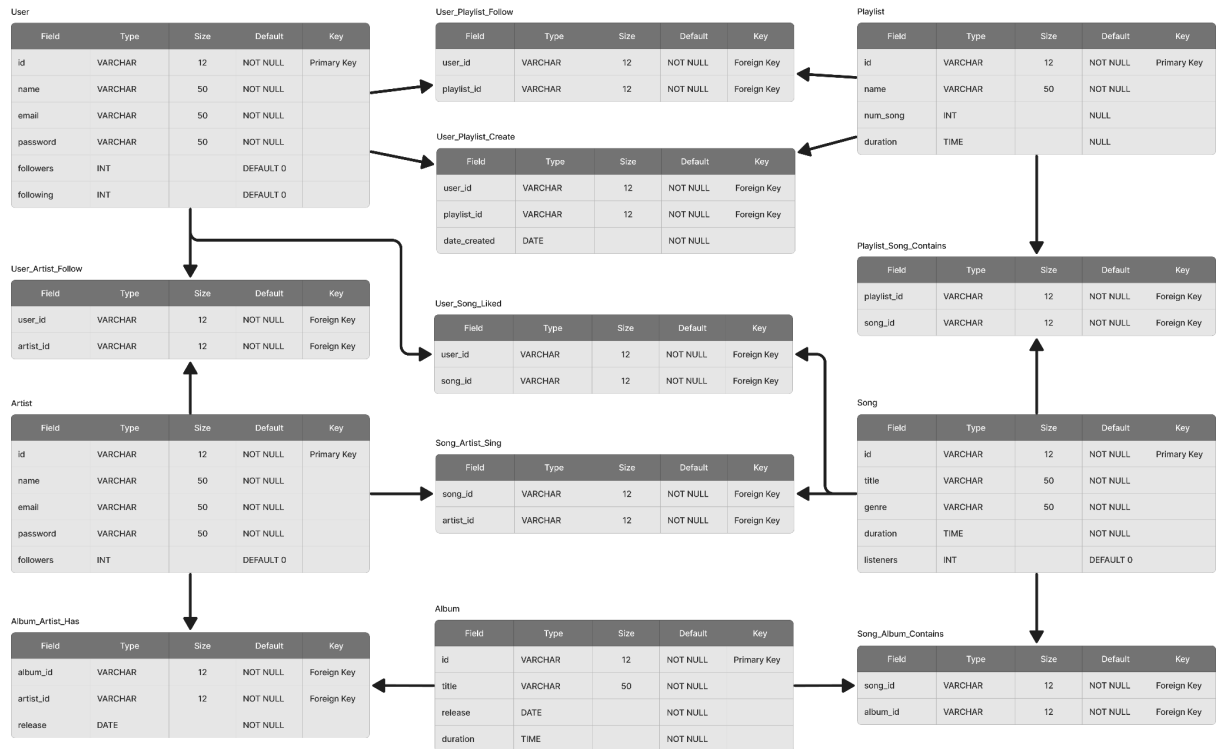
3. Apakah semua relasi (antar entitas) sudah ada?

= Ya, semua relasi sudah ada dan lengkap. Misalnya, entitas “petugas” memiliki relasi ‘mencatat’ ke entitas “pengembalian” dan “peminjaman”

4. Apakah atribut relasi sudah ada / benar?

= Tidak, atribut relasi tidak ada sama sekali pada ERD ini.

BAB IV KAMUS DATA



[LINK KAMUS DATA](#) / [LINK KAMUS DATA](#)

BAB V

TABEL DAN POPULASINYA

user

spotitip v2.user: 500 rows total (approximately) Next

id	name	email	password	followers	following
US1	Makenzie Pfeffer	harvey.hanna@hotmail.com	c035a5bfb14e0483fe7c957f9a743a6a8e77ad4a	778	892
US10	Ashly Kerluke	larkin.tad@hotmail.com	ac1080eb68e6324a21e8ef9de5079bdc10327703	35	196
US100	Lazaro Emmerich II	mckenzie.blaze@yahoo.com	3ad33fa12319db06f5b50622634bc4cb2735395d	95	233
US101	Lonzo Upton DVM	tristin.beahan@gmail.com	434bca73e66c65788db0295ace8fba30e64e22c1	514	56
US102	Maritza Runte	wboyle@hotmail.com	fef1b74011d596d6492d412d9ff80c4bdb877a69	293	123
US103	Christa Ortiz	destiney33@hotmail.com	edd9c96c75624819413ef47ffa251b8b9a3ecb8a	852	858

artist

spotitip v2.artist: 500 rows total (approximately)

id	name	email	password	followers
AR1	Berta Reichel MD	chloe32@yahoo.com	1d05cb7fe340b13d28a8e93f4a588672317ed8a6	636
AR10	Dorris Mertz	bbruen@hotmail.com	2f078027dba78329d26ef922125c0ae7051dd122	841
AR100	Aleen Macejkovic	pfunk@yahoo.com	54df4d51269299e3be4fe69c646a1ed747faab99	638
AR101	Seth Mayert	dandre61@gmail.com	bd4ba307a3c306af6b9ec766073e05ae7cfcd3e4	14
AR102	Amy Torphy	goodwin.lessie@gmail.com	13bfb09c10db3b9ceb0a46b532903f667508a10d	353

song

spotitip v2.song: 500 rows total (approximately)

id	title	genre	duration	listeners
S0381	Cupiditate rerum.	Aut.	00:06:00	798,233
S0397	Non numquam rerum.	Nihil.	00:06:00	8,244
S0398	Alias quam illo.	Officiis.	00:06:00	902,605
S0399	Velit cum.	Quaerat voluptas.	00:06:00	495,356

album

spotitip v2.album: 500 rows total (approximately)

id	title	release	num_song	duration
AL1	Possimus numquam minima.	2015-08-06	5	00:30:00
AL10	Qui molestiae.	1985-06-20	14	01:24:00
AL100	Tempore quia adipisci.	1971-09-21	15	01:30:00

playlist

spotitip v2.playlist: 500 rows total (approximately)

id	name	num_song	duration
PL1	Animi id.	38	03:48:00
PL10	Asperiores id ducimus.	68	06:48:00
PL100	Assumenda harum.	25	02:30:00
PL101	Est quibusdam aut.	56	05:36:00
PL102	Suscipit molestiae.	74	07:24:00

album_artist_has

playlist_song_contains

song_album_contains

song_artist_sing

user_artist_follow

user_playlist_create

user_playlist_follow

user_song_liked

[LINK TABEL](#)

BAB VI

QUERY DAN ALJABAR RELASINYA

A. Query Pembuatan Tabel

```
CREATE DATABASE IF NOT EXISTS `spotitip v2`;
USE `spotitip v2`;

CREATE TABLE IF NOT EXISTS `artist` (
  `id` varchar(12) NOT NULL, `name` varchar(50) NOT NULL, `email` varchar(50) NOT NULL,
  `password` varchar(50) NOT NULL, `followers` int(11) DEFAULT 0,
  PRIMARY KEY (`id`)
);

CREATE TABLE IF NOT EXISTS `song` (
  `id` varchar(12) NOT NULL, `title` varchar(50) NOT NULL, `genre` varchar(50) NOT NULL,
  `duration` TIME NOT NULL, `listeners` int(11) DEFAULT 0,
  PRIMARY KEY (`id`)
);

CREATE TABLE IF NOT EXISTS `album` (
  `id` varchar(12) NOT NULL, `title` varchar(50) NOT NULL, `release` date NOT NULL,
  `num_song` int(11) NOT NULL, `duration` TIME NOT NULL,
  PRIMARY KEY (`id`)
);

CREATE TABLE IF NOT EXISTS `album_artist_has` (
  `album_id` varchar(12) NOT NULL, `artist_id` varchar(12) NOT NULL, `release` date NOT NULL,
  KEY `album_id_has` (`album_id`), KEY `artist_id_has` (`artist_id`),
  CONSTRAINT `album_id_has` FOREIGN KEY (`album_id`) REFERENCES `album` (`id`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `artist_id_has` FOREIGN KEY (`artist_id`) REFERENCES `artist` (`id`) ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE IF NOT EXISTS `playlist` (
  `id` varchar(12) NOT NULL, `name` varchar(50) NOT NULL,
  `num_song` int(11) DEFAULT NULL, `duration` TIME DEFAULT NULL,
  PRIMARY KEY (`id`)
);

CREATE TABLE IF NOT EXISTS `playlist_song_contains` (
  `playlist_id` varchar(12) NOT NULL, `song_id` varchar(12) NOT NULL,
  KEY `playlist_id_contains` (`playlist_id`), KEY `song_id_contains_pl` (`song_id`),
  CONSTRAINT `playlist_id_contains` FOREIGN KEY (`playlist_id`) REFERENCES `playlist` (`id`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `song_id_contains_pl` FOREIGN KEY (`song_id`) REFERENCES `song` (`id`) ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE IF NOT EXISTS `song_album_contains` (
  `song_id` varchar(12) NOT NULL, `album_id` varchar(12) NOT NULL,
  KEY `song_id_contains` (`song_id`), KEY `album_id_contains_al` (`album_id`),
  CONSTRAINT `album_id_contains` FOREIGN KEY (`album_id`) REFERENCES `album` (`id`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `song_id_contains_al` FOREIGN KEY (`song_id`) REFERENCES `song` (`id`) ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE IF NOT EXISTS `song_artist_sing` (
  `song_id` varchar(12) NOT NULL, `artist_id` varchar(12) NOT NULL,
  KEY `song_id_sing` (`song_id`), KEY `artist_id_sing` (`artist_id`),
  CONSTRAINT `artist_id_sing` FOREIGN KEY (`artist_id`) REFERENCES `artist` (`id`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `song_id_sing` FOREIGN KEY (`song_id`) REFERENCES `song` (`id`) ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE IF NOT EXISTS `user` (
  `id` varchar(12) NOT NULL, `name` varchar(50) NOT NULL, `email` varchar(50) NOT NULL,
  `password` varchar(50) NOT NULL, `followers` int(11) DEFAULT 0, `following` int(11) DEFAULT 0,
  PRIMARY KEY (`id`)
);


CREATE TABLE IF NOT EXISTS `user_artist_follow` (
  `user_id` varchar(12) NOT NULL, `artist_id` varchar(12) NOT NULL,
  KEY `user_id_follow_ar` (`user_id`), KEY `artist_id_follow` (`artist_id`),
  CONSTRAINT `artist_id_follow` FOREIGN KEY (`artist_id`) REFERENCES `artist` (`id`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `user_id_follow_ar` FOREIGN KEY (`user_id`) REFERENCES `user` (`id`) ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE IF NOT EXISTS `user_playlist_create` (
  `user_id` varchar(12) NOT NULL, `playlist_id` varchar(12) NOT NULL, `date_created` date NOT NULL,
  KEY `user_id_create` (`user_id`), KEY `playlist_id_create` (`playlist_id`),
  CONSTRAINT `playlist_id_create` FOREIGN KEY (`playlist_id`) REFERENCES `playlist` (`id`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `user_id_create` FOREIGN KEY (`user_id`) REFERENCES `user` (`id`) ON DELETE CASCADE ON UPDATE CASCADE
);


CREATE TABLE IF NOT EXISTS `user_playlist_follow` (
  `user_id` varchar(12) NOT NULL, `playlist_id` varchar(12) NOT NULL,
  KEY `playlist_id_follow` (`playlist_id`), KEY `user_id_follow_pl` (`user_id`),
  CONSTRAINT `playlist_id_follow` FOREIGN KEY (`playlist_id`) REFERENCES `playlist` (`id`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `user_id_follow_pl` FOREIGN KEY (`user_id`) REFERENCES `user` (`id`) ON DELETE CASCADE ON UPDATE CASCADE
);
```

OUTPUT


user

#	Name	Datatype	Length/Set	Unsigned	Allow NULL	Zerofill	Default
 1	id	VARCHAR	12	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default
2	name	VARCHAR	50	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default
3	email	VARCHAR	50	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default
4	password	VARCHAR	50	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default
5	followers	INT	11	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	'0'
6	following	INT	11	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	'0'


artist

#	Name	Datatype	Length/Set	Unsigned	Allow NULL	Zerofill	Default
 1	id	VARCHAR	12	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default
2	name	VARCHAR	50	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default
3	email	VARCHAR	50	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default
4	password	VARCHAR	50	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default
5	followers	INT	11	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	'0'


song

#	Name	Datatype	Length/Set	Unsigned	Allow NULL	Zerofill	Default
 1	id	VARCHAR	12	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default
2	title	VARCHAR	50	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default
3	genre	VARCHAR	50	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default
4	duration	TIME		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	'00:00:00'
5	listeners	INT	11	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	'0'



playlist



#	Name	Datatype	Length/Set	Unsigned	Allow NULL	Zerofill	Default
 1	id	VARCHAR	12	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default
2	name	VARCHAR	50	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default
3	num_song	INT	11	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL
4	duration	TIME		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL

album



#	Name	Datatype	Length/Set	Unsigned	Allow NULL	Zerofill	Default
 1	id	VARCHAR	12	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default
2	title	VARCHAR	50	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default
3	release	DATE		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default
4	num_song	INT	11	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default
5	duration	TIME		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default



user_playlist_follow

#	Name	Datatype	Length/Set	Unsigned	Allow NULL	Zerofill	Default
 1	user_id	VARCHAR	12	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default
 2	playlist_id	VARCHAR	12	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default



Key name	Columns	Reference table	Foreign columns	On UPDATE	On DELETE
 playlist_id_follow	playlist_id	spotitip v2.playlist	id	CASCADE	CASCADE
 user_id_follow	user_id	spotitip v2.user	id	CASCADE	CASCADE



user_playlist_create

#	Name	Datatype	Length/Set	Unsigned	Allow NULL	Zerofill	Default
 1	user_id	VARCHAR	12	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	No default
 2	playlist_id	VARCHAR	12	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	No default
3	date_created	DATE		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	No default



Key name	Columns	Reference table	Foreign columns	On UPDATE	On DELETE
 playlist_id_create	playlist_id	spotifyip v2.playlist	id	CASCADE	CASCADE
 user_id_create	user_id	spotifyip v2.user	id	CASCADE	CASCADE



user_artist_follow

#	Name	Datatype	Length/Set	Unsigned	Allow NULL	Zerofill	Default
 1	user_id	VARCHAR	12	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	No default
 2	artist_id	VARCHAR	12	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	No default



Key name	Columns	Reference table	Foreign columns	On UPDATE	On DELETE
 artist_id	artist_id	spotifyip v2.artist	id	CASCADE	CASCADE
 user_id	user_id	spotifyip v2.user	id	CASCADE	CASCADE



user_song_liked

#	Name	Datatype	Length/Set	Unsigned	Allow NULL	Zerofill	Default
 1	user_id	VARCHAR	12	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	No default
 2	song_id	VARCHAR	12	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	No default



Key name	Columns	Reference table	Foreign columns	On UPDATE	On DELETE
 song_id_liked	song_id	spotifyip v2.song	id	CASCADE	CASCADE
 user_id_liked	user_id	spotifyip v2.user	id	CASCADE	CASCADE



song_artist_sing

#	Name	Datatype	Length/Set	Unsigned	Allow NULL	Zerofill	Default
 1	song_id	VARCHAR	12	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	No default
 2	artist_id	VARCHAR	12	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	No default



Key name	Columns	Reference table	Foreign columns	On UPDATE	On DELETE
 artist_id_sing	artist_id	spotifyip v2.artist	id	CASCADE	CASCADE
 song_id_sing	song_id	spotifyip v2.song	id	CASCADE	CASCADE



song_album_contains

#	Name	Datatype	Length/Set	Unsigned	Allow NULL	Zerofill	Default
 1	song_id	VARCHAR	12	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	No default
 2	album_id	VARCHAR	12	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	No default

Key name	Columns	Reference table	Foreign columns	On UPDATE	On DELETE
 album_id	album_id	spotifyip v2.album	id	CASCADE	CASCADE
 song_id	song_id	spotifyip v2.song	id	CASCADE	CASCADE

playlist_song_contains

#	Name	Datatype	Length/Set	Unsigned	Allow NULL	Zerofill	Default
 1	playlist_id	VARCHAR	12	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	No default
 2	song_id	VARCHAR	12	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	No default

Key name	Columns	Reference table	Foreign columns	On UPDATE	On DELETE
 playlist_id_contains	playlist_id	spotifyip v2.playlist	id	CASCADE	CASCADE
 song_id_contains	song_id	spotifyip v2.song	id	CASCADE	CASCADE

album_artist_has

#	Name	Datatype	Length/Set	Unsigned	Allow NULL	Zerofill	Default
1	album_id	VARCHAR	12	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	No default
2	artist_id	VARCHAR	12	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	No default
3	release	DATE		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	No default

Key name	Columns	Reference table	Foreign columns	On UPDATE	On DELETE
album_id_has	album_id	spotify v2.album	id	CASCADE	CASCADE
artist_id_has	artist_id	spotify v2.artist	id	CASCADE	CASCADE

Tampilan Tabel Keseluruhan (Tanpa Data)

Name ▲	Rows	Size
album	0	16.0 KiB
album_artist_has	0	48.0 KiB
artist	0	16.0 KiB
playlist	0	16.0 KiB
playlist_song_contains	0	48.0 KiB
song	0	16.0 KiB
song_album_contains	0	48.0 KiB
song_artist_sing	0	48.0 KiB
user	0	16.0 KiB
user_artist_follow	0	48.0 KiB
user_playlist_create	0	48.0 KiB
user_playlist_follow	0	48.0 KiB
user_song_liked	0	48.0 KiB

Name ▲	Rows	Size
album	500	80.0 KiB
album_artist_has	500	80.0 KiB
artist	500	96.0 KiB
playlist	500	64.0 KiB
playlist_song_contains	500	80.0 KiB
song	500	64.0 KiB
song_album_contains	500	80.0 KiB
song_artist_sing	500	80.0 KiB
user	500	96.0 KiB
user_artist_follow	500	80.0 KiB
user_playlist_create	500	80.0 KiB
user_playlist_follow	500	80.0 KiB
user_song_liked	500	80.0 KiB

[QUERY CREATE TABLE](#)

B. Query Pengisian Data pada Tabel Menggunakan File

```
LOAD DATA INFILE "D:/adhim/SEMESTER 4/Manajemen Basis Data/Spotitip V2/csv/user.csv" INTO TABLE user FIELDS TERMINATED BY ',' IGNORE 1 LINES;
LOAD DATA INFILE "D:/adhim/SEMESTER 4/Manajemen Basis Data/Spotitip V2/csv/artist.csv" INTO TABLE artist FIELDS TERMINATED BY ',' IGNORE 1 LINES;
LOAD DATA INFILE "D:/adhim/SEMESTER 4/Manajemen Basis Data/Spotitip V2/csv/song.csv" INTO TABLE song FIELDS TERMINATED BY ',' IGNORE 1 LINES;
LOAD DATA INFILE "D:/adhim/SEMESTER 4/Manajemen Basis Data/Spotitip V2/csv/playlist.csv" INTO TABLE playlist FIELDS TERMINATED BY ',' IGNORE 1 LINES;
LOAD DATA INFILE "D:/adhim/SEMESTER 4/Manajemen Basis Data/Spotitip V2/csv/album.csv" INTO TABLE album FIELDS TERMINATED BY ',' IGNORE 1 LINES;

LOAD DATA INFILE "D:/adhim/SEMESTER 4/Manajemen Basis Data/Spotitip V2/csv/user_artist_follow.csv" INTO TABLE user_artist_follow FIELDS TERMINATED BY ',' IGNORE 1 LINES;
LOAD DATA INFILE "D:/adhim/SEMESTER 4/Manajemen Basis Data/Spotitip V2/csv/user_playlist_create.csv" INTO TABLE user_playlist_create FIELDS TERMINATED BY ',' IGNORE 1 LINES;
LOAD DATA INFILE "D:/adhim/SEMESTER 4/Manajemen Basis Data/Spotitip V2/csv/user_playlist_follow.csv" INTO TABLE user_playlist_follow FIELDS TERMINATED BY ',' IGNORE 1 LINES;
LOAD DATA INFILE "D:/adhim/SEMESTER 4/Manajemen Basis Data/Spotitip V2/csv/user_song_liked.csv" INTO TABLE user_song_liked FIELDS TERMINATED BY ',' IGNORE 1 LINES;
LOAD DATA INFILE "D:/adhim/SEMESTER 4/Manajemen Basis Data/Spotitip V2/csv/song_album_contains.csv" INTO TABLE song_album_contains FIELDS TERMINATED BY ',' IGNORE 1 LINES;
LOAD DATA INFILE "D:/adhim/SEMESTER 4/Manajemen Basis Data/Spotitip V2/csv/song_artist_sing.csv" INTO TABLE song_artist_sing FIELDS TERMINATED BY ',' IGNORE 1 LINES;
LOAD DATA INFILE "D:/adhim/SEMESTER 4/Manajemen Basis Data/Spotitip V2/csv/album_artist_has.csv" INTO TABLE album_artist_has FIELDS TERMINATED BY ',' IGNORE 1 LINES;
LOAD DATA INFILE "D:/adhim/SEMESTER 4/Manajemen Basis Data/Spotitip V2/csv/playlist_song_contains.csv" INTO TABLE playlist_song_contains FIELDS TERMINATED BY ',' IGNORE 1 LINES;
```

C. Query Pengecekan Data Tabel

```
You, 1 second ago | 1 author (You) | ▶ Run on active connection |
-- entitas
select * from user;
select * from artist;
select * from song;
select * from album;
select * from playlist;

-- relasi
select * from user_artist_follow;
select * from user_playlist_create;
select * from user_playlist_follow;
select * from user_song_liked;
select * from song_album_contains;
select * from song_artist_sing;
select * from album_artist_has;
select * from playlist_song_contains;
```

D. Query Aljabar Relasional pada Tabel

a. SELECTION

- Tampilkan lagu dengan jumlah pendengar diatas seratus ribu pada Aplikasi Spotitip
- $\sigma_{jmlh_pendengar > 100000} (LAGU)$
- SELECT * FROM lagu WHERE jmlh_pendengar > 100000;

b. PROJECTION

- Tampilkan judul lagu, genre musik, dan jumlah pendengar dari Aplikasi Spotitip
- $\pi_{judul, genre, jmlh_pendengar} (LAGU)$
- SELECT judul, genre, jmlh_pendengar FROM lagu;

c. UNION

- Tampilkan semua email, nama, dan password dari akun artis dan akun pengguna
- $\rho_{nama} (\pi_{email_artis, nama_artis, password_artis} (ARTIS) \cup \pi_{email_user, nama_akun, password_user} (USER))$
- SELECT email_artis AS email, nama_artis AS nama, password_artis AS password
FROM artis UNION SELECT email_user AS email, nama_akun AS nama,
password_user AS password FROM pengguna;

d. CARTESIAN PRODUCT

- Tampilkan daftar pengguna apabila pengguna mendengarkan semua playlist dan tampilkan juga playlistnya.
- $\rho_{daftar} (\pi_{email, name, password} (USER) \times \pi_{nama_playlist, jmlh_lagu_playlist} (PLAYLIST))$
- SELECT * FROM pengguna CROSS JOIN (SELECT nama_playlist, jmlh_lagu_playlist
FROM playlist) AS daftar;

e. Intersection

- Tampilkan lagu-lagu yang dikelompokkan ke dalam single
- $\pi_{lagu.id_lagu, lagu.judul} (\sigma_{lagu.id_lagu = dikelompokkan.id_lagu} (LAGU \cap Dikelompokkan \cap Single))$
- SELECT lagu.id_lagu, lagu.judul, single.id_single, single.nama_single FROM lagu INNER
JOIN dikelompokkan ON lagu.id_lagu = dikelompokkan.id_lagu INNER JOIN single ON
dikelompokkan.id_single = single.id_single;

f. DIFFERENCE

- Tampilkan semua lagu yang tidak dikelompokkan kedalam single maupun album
- $\pi_{id_lagu} (LAGU) - \pi_{id_lagu} (Dikelompokkan)$
- SELECT id_lagu FROM lagu WHERE id_lagu NOT IN (SELECT id_lagu FROM dikelompokkan

BAB VII

QUERY OPTIMASI

1. Apa itu optimasi *query*?

= Optimasi *query* pada MySQL adalah proses meningkatkan kinerja dan efisiensi eksekusi *query* dalam database. Ini melibatkan pemahaman terhadap bagaimana struktur tabel, indeks, dan cara untuk menulis *query* yang lebih efisien.

2. Apa saja tujuan optimasi *query*?

- a. Meningkatkan kinerja dan waktu respon dari sistem basis data
- b. Mempercepat waktu eksekusi
- c. Mengurangi penggunaan memory dan kerja CPU
- d. Meningkatkan respon yang lebih cepat.

3. Bagaimana contoh teknik optimasi *query*?

- a. Menulis *query* yang efisien dengan mempertimbangkan penggunaan indeks dan menghindari subquery yang berlebihan.
- b. Melakukan pengindeksan yang tepat dengan membuat indeks pada kolom yang sering digunakan.
- c. Memastikan struktur tabel sesuai dengan kebutuhan *query* dan penggunaan tipe data yang sesuai.
- d. Menggunakan INSERT IGNORE INTO dalam memasukkan data.
 - Dapat menyisipkan data ke dalam tabel tanpa harus memeriksa terlebih dahulu apakah data tersebut sudah ada di tabel atau tidak
 - Proses penyisipan data menjadi lebih lancar dan efisien

4. Berikan contoh optimasi *query* di MySQL.

1. Gunakan Indeks Secara Efektif:

- Pastikan kolom yang terlibat dalam klausa WHERE dan kondisi JOIN diindeks.
- Contoh: Jika Anda memiliki *query* yang difilter berdasarkan `user_id`, buat indeks pada kolom itu: `CREATE INDEX idx_user_id ON users(user_id);`

2. Hindari hanya menggunakan `SELECT *`:

- Hanya pilih kolom yang Anda perlukan alih-alih menggunakan `SELECT *`.
- Contoh: Dari pada `SELECT * FROM orders`, gunakan `SELECT order_id, customer_id, order_date FROM orders.`

3. Optimalkan *JOIN*:

- Gunakan INNER JOIN ketika Anda hanya membutuhkan baris yang cocok, dan LEFT JOIN ketika Anda ingin semua baris dari tabel kiri.
- Contoh: `SELECT users.name, orders.order_id FROM users INNER JOIN orders ON users.user_id = orders.user_id;`

4. Gunakan LIMIT pada query:

- Jika Anda tidak membutuhkan semua hasil, gunakanlah **LIMIT** untuk membatasi jumlah baris yang dikembalikan.
- Contoh: `SELECT * FROM products LIMIT 10;`

5. Hindari *Subquery*:

- Tulis ulang *subquery* sebagai JOIN bila memungkinkan untuk meningkatkan kinerja.
- Contoh: Konversi `SELECT name FROM products WHERE category_id = (SELECT category_id FROM categories WHERE name = 'Electronics')` ke sebuah JOIN.

6. Gunakan UNION Daripada OR:

- Ganti beberapa **OR** kondisi dengan **UNION** untuk *query* yang lebih efisien.
- Contoh: Perubahan `SELECT * FROM products WHERE price > 100 OR category = 'Electronics'` ke sebuah *query* **UNION**.

7. Hindari Penggunaan Wildcard di Awal Kueri LIKE:

- Memulai sebuah pola **LIKE** dengan % tidak dapat menggunakan indeks. Hindari jika memungkinkan.
- Contoh: Gunakan `name LIKE 'app%'` alih-alih `name LIKE '%app%'`.

8. Penggunaan INSERT INTO dan INSERT IGNORE INTO secara efisien:

- Saat menyisipkan, menambahkan atau memperbarui beberapa baris data, gunakan INSERT INTO untuk mencegah duplikasi data pada tabel tapi akan menggunakan waktu sedikit lebih lama karena akan dilakukan perbandingan data yang ingin dimasukkan dan data yang sudah ada sebelumnya untuk mencegah terjadinya duplikasi.
- Gunakan INSERT IGNORE INTO ketika sebuah tabel tidak mempermasalahkan terjadinya duplikasi data sehingga data akan dimasukkan tanpa perlu memeriksa baris-baris yang sudah ada sebelumnya tentunya cara ini akan menjadi pilihan yang tepat.

9. Hindari Penggunaan Fungsi di WHERE:

- Menerapkan fungsi ke kolom dalam klausa WHERE dapat mencegah penggunaan indeks.
- Contoh: Daripada `WHERE YEAR(order_date) = 2023`, gunakan `WHERE order_date >= '2023-01-01' AND order_date < '2024-01-01'`.

10. Gunakan EXPLAIN untuk Menganalisis Query:

- Memanfaatkan *query* `EXPLAIN` untuk menganalisis rencana eksekusi kueri dan mengoptimalkannya.
- Contoh: `EXPLAIN SELECT * FROM customers WHERE country = 'USA' ;`

11. Normalisasi Data:

- Normalisasikan database Anda untuk mengurangi redundansi dan meningkatkan efisiensi kueri.
- Contoh: Daripada menyimpan data berulang seperti nama negara bagian dalam beberapa baris, gunakan tabel negara bagian terpisah dan tautkan dengan kunci asing.

12. Hindari ORDER BY RAND():

- Menggunakan `ORDER BY RAND()` bisa sangat lambat pada kumpulan data besar. Pertimbangkan metode alternatif untuk mengacak hasil.
- Contoh: Daripada `SELECT * FROM products ORDER BY RAND() LIMIT 10`, gunakan teknik pengacakan yang lebih efisien.

13. Agregasi Cache:

- Cache sering kali menggunakan data agregat untuk mengurangi kebutuhan penghitungan yang mahal.
- Contoh: Simpan total penjualan harian dalam tabel terpisah dan perbarui secara berkala.

14. Menggunakan LOAD DATA, bukan INSERT INTO:

- Dalam memasukkan data dengan jumlah yang besar, menggunakan LOAD DATA melakukan proses eksekusi lebih cepat daripada penggunaan INSERT INTO ataupun INSERT IGNORE INTO.
- LOAD DATA adalah sintaks MySQL yang sudah dioptimalkan karena langsung membaca satu file sedangkan sintaks INSERT melakukannya dengan memproses

satu baris data pada satu waktu yang dapat menjadi lambat untuk volume data yang besar

15. Partisi Tabel Besar:

- Untuk tabel besar, pertimbangkan mempartisi untuk meningkatkan kinerja kueri.
- Contoh: Partisi tabel berdasarkan tanggal, pisahkan data menjadi partisi bulanan atau tahunan untuk pengambilan data lebih cepat.

5. Implementasikan lima dari teknik tersebut dalam database MySQL.

- Gunakan limitasi dengan fungsi LIMIT

```
SELECT title, genre, listeners FROM user WHERE listeners
< 100000 LIMIT 20;
```

title	genre	listeners
 Filter...	 Filter...	 Filter...
Atque cumque.	Assumenda qui.	11906
Ad natus.	Deleniti.	99785
Autem nulla error.	Voluptates.	4580
Eveniet voluptas velit.	Consectetur.	73419
Harum quasi laboriosam.	Itaque quae.	94329
Sed ipsam.	Quisquam.	18672
Consectetur alias.	Quos expedita.	85493
Inventore ea.	Sapiente porro.	40859
Natus qui.	Corporis.	66689
Inventore vel adipisci.	Optio.	29938
Quia ut ut.	Provident.	48560

Ea est.	Voluptas.	47867
Praesentium voluptatibus.	Ipsa quod.	88926
Nihil sed blanditiis.	Architecto.	32486
Est neque.	Molestiae quia.	6907
Aliquam voluptates.	Veniam sint.	13181
Atque laudantium laborio...	Atque et.	19396
Voluptas beatae.	Corporis et.	71118
Aut commodi.	Nesciunt enim.	39032
Qui molestiae consequatur.	Vel hic.	66895

b. Hindari hanya menggunakan SELECT *

Berikut hasil query dari **SELECT * FROM user;**

id	name	email	password	followers	following
abc Filter...	abc Filter...	abc Filter...	abc Filter...	abc Filter...	abc Filter...
US1	Makenzie Pfeffer	harvey.hanna@hotmail.com	c035a5bfb14e0483fe7c957f9a743a6a8e77ad4a	778	892
US10	Ashly Kerluke	larkin.tad@hotmail.com	ac1080eb68e6324a21e8ef9de5079bdc10327703	35	196
US100	Lazaro Emmerich II	mckenzie.blaze@yahoo.com	3ad33fa12319db06f5b50622634bc4cb2735395d	95	233
US101	Lonzo Upton DVM	tristin.beahan@gmail.com	434bca73e66c65788db0295ace8fba30e64e22c1	514	56
US102	Maritza Runte	wboyle@hotmail.com	fef1b74011d596d6492d412d9ff80c4bdb877a69	293	123
US103	Christa Ortiz	destiney33@hotmail.com	edd9c96c75624819413ef47ffa251b8b9a3ecb8a	852	858
US104	Prof. Eloisa Eichmann ...	herman.karen@gmail.com	05d53526702cdeafc8548994e49644c90f238da0	717	15

akan muncul semua kolom dan data yang ada di dalam tabel user.

Berikut hasil query dari **SELECT name, email, password FROM user;**

name	email	password
abc Filter...	abc Filter...	abc Filter...
Makenzie Pfeffer	harvey.hanna@hotmail.com	c035a5bfb14e0483fe7c957f9a743a6a8e77ad4a
Ashly Kerluke	larkin.tad@hotmail.com	ac1080eb68e6324a21e8ef9de5079bdc10327703
Lazaro Emmerich II	mckenzie.blaze@yahoo.com	3ad33fa12319db06f5b50622634bc4cb2735395d
Lonzo Upton DVM	tristin.beahan@gmail.com	434bca73e66c65788db0295ace8fba30e64e22c1
Maritza Runte	wboyle@hotmail.com	fef1b74011d596d6492d412d9ff80c4bdb877a69
Christa Ortiz	destiney33@hotmail.com	edd9c96c75624819413ef47ffa251b8b9a3ecb8a
Prof. Eloisa Eichmann PhD	herman.karen@gmail.com	05d53526702cdeafc8548994e49644c90f238da0
Gino OHara	jlebsack@yahoo.com	cbb55d10a1a84586e74837faef2936fc248cb382
Cristina Carroll	maeve69@yahoo.com	35756481ac5972014bf588fb4b0c59acec247971
Eva Stanton	zula66@hotmail.com	930264f31f1e1e23c9ba6d3559550e5b24b28907
Mr. Pedro Mayer	ali.jacobson@gmail.com	9a2f6972f51952a8430d125e81ee6023add324eb

- c. Memasukkan data menggunakan LOAD DATA ketimbang INSERT INTO menggunakan LOAD DATA ketika ingin memasukkan data dalam jumlah yang besar dan menggunakan INSERT ketika ingin menyisipkan beberapa data ke dalam tabel.

Query_ID	Duration	Query
abc Filter...	abc Filter...	abc Filter...
1	0.008712	LOAD DATA INFILE "D:/adhim/SEN
4	0.0099215	INSERT IGNORE INTO `user` (`id`,

- d. Menggunakan query INSERT INTO dan INSERT IGNORE INTO
menggunakan query INSERT INTO untuk menghindari duplikasi data ketika ingin menyisipkan data baru ke dalam tabel.

```
INSERT INTO `user` (`id`, `name`, `email`, `password`, `followers`, `following`) VALUES
('US1', 'Makenzie Pfeffer', 'harvey.hanna@hotmail.com', 'c035a5bfb14e0483fe7c957f9a743a6a8e77ad4a', 778, 892),
('US10', 'Ashly Kerluke', 'larkin.tad@hotmail.com', 'ac1080eb68e6324a21e8ef9de5079bdc10327703', 35, 196),
('US100', 'Lazaro Emmerich II', 'mckenzie.blaze@yahoo.com', '3ad33fa12319db06f5b50622634bc4cb2735395d', 95, 233);
```

dan menggunakan query INSERT IGNORE INTO ketika suatu tabel memperbolehkan duplikasi data untuk mempercepat dan efisiensi dalam eksekusi query.

```
INSERT IGNORE INTO `user_artist_follow` (`user_id`, `artist_id`) VALUES
('US1', 'AR464'),,
('US104', 'AR205'),
('US105', 'AR498'),
('US106', 'AR109'),
('US107', 'AR303'),
('US108', 'AR272'),
('US109', 'AR342'),
('US111', 'AR11'),
('US111', 'AR459'),
('US111', 'AR308'),
('US111', 'AR499'),
('US111', 'AR192'),
('US111', 'AR95');
```

- e. Gunakan index
memberikan index key pada satu kolom atau lebih pada suatu tabel yang akan sering mengalami proses pencarian. Menggunakan sintaks:

```
CREATE INDEX `user_idx` ON `user`(`id`) USING BTREE;
CREATE INDEX `username_idx` ON `user`(`name`) USING BTREE;
```

#	Name	Datatype	Length/Set	Unsigned	Allow NULL	Zerofill	Default
1	id	VARCHAR	12	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default
2	name	VARCHAR	50	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default
3	email	VARCHAR	50	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default
4	password	VARCHAR	50	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default
5	followers	INT	11	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	0
6	following	INT	11	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	0

Name	Type / Length	Algorithm
PRIMARY KEY	PRIMARY	BTREE
username_idx	KEY	BTREE
name		
user_idx	KEY	BTREE
id		

[LINK GITHUB SPOTITIP V2](#)