

IF2240 - Basis Data

Praktikum 2

Operasi SQL Lanjut

9 April 2025

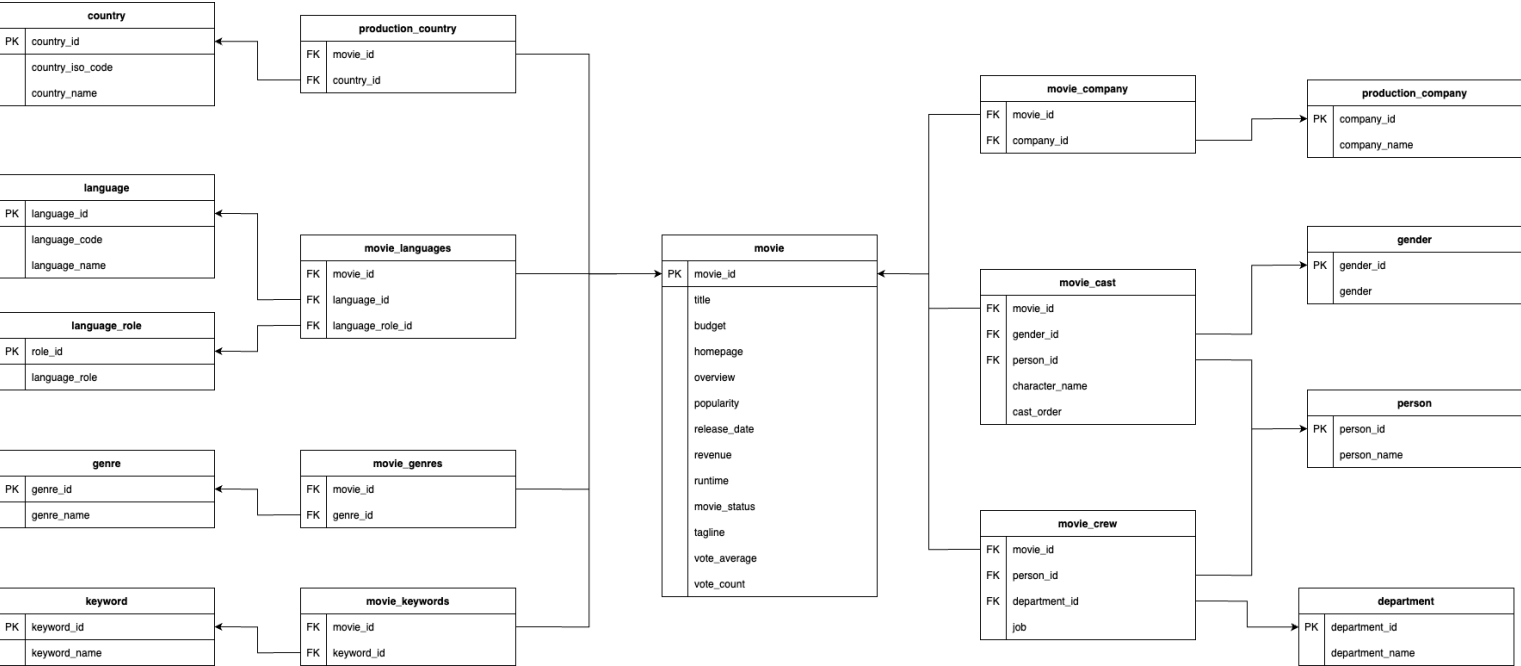


Jethro Jens Norbert Simatupang - 13523081

Muhammad Iqbal Haidar - 13523111

**Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung
2025**

IMDb (IMDatabase)



Legendasel

No	Nama Tabel	Keterangan
1	movie	Informasi setiap film.
2	production_country	Negara tempat film diproduksi. Tabel pemetaan negara dengan film.
3	movie_languages	Bahasa yang digunakan pada film. Tabel pemetaan bahasa dengan film.
4	movie_genres	Genre film. Tabel pemetaan genre dengan film.
5	movie_keywords	Keyword film. Tabel pemetaan kata kunci dengan film.
6	movie_company	Perusahaan produksi film. Tabel pemetaan perusahaan produksi dengan film.
7	movie_cast	Pemeran film. Tabel pemetaan pemeran dengan film.
8	movie_crew	Kru film. Tabel pemetaan kru pekerja dengan film.
9	country	Negara.
10	language	Bahasa.
11	language_role	Jenis bahasa yang digunakan (Original atau Spoken).
12	genre	Genre.
13	keyword	Keyword.
14	production_company	Perusahaan pemroduksi.
15	gender	Gender.
16	person	Orang.
17	department	Departemen dari para kru.

Petunjuk

1. Praktikum ini wajib dikerjakan di dalam Laboratorium Teknik Informatika ITB.
2. Tidak ada restriksi penggunaan keyword pada praktikum ini. Anda dapat menggunakan keyword apapun untuk menjawab soal praktikum.
3. Aturan No 2 tidak berlaku untuk *join* relasi. Anda harus menggunakan *keyword join* yang bersesuaian, tidak boleh melakukan *join* secara manual (seperti pada praktikum 1).
4. Terdapat 2 query yang digunakan pada soal 8 - 10,
 - a. Query jawaban
Seluruh query yang Anda tulis pada bagian ini merupakan jawaban Anda untuk persoalan terkait. Query ini akan menjadi komponen penilaian dari praktikum ini.
 - b. Query testing
Query yang digunakan untuk membuktikan jawaban Anda. Query ini sudah disediakan oleh Tim Asisten. Anda hanya perlu menjalankannya pada perangkat Laboratorium untuk mendapatkan tangkapan layar sebelum dan sesudah Anda menjalankan query jawaban Anda.

1. Skena Total!

Sudah lama tidak bertemu! Sebelumnya, terima kasih karena sudah membantu Kelkelkelkelkel Cicerokel Mokel Ambatokel dalam menjalankan ibadah puasanya. Berkat Anda, dia berhasil melewati puasa dengan hanya *mokel* sebanyak 25.3 kali. Semenjak selesai puasa, menonton film telah menjadi hobi barunya. Sekarang, ia ingin mendalami hobi baru tersebut dengan menonton daftar film baru. Film tersebut merupakan film yang menurutnya tergolong *skena*. Ia ingin menonton film-film skena agar dapat mencantumkan film-film tersebut ke akun Letterboxd miliknya (yang tidak memiliki *followers* satupun).

Berikut adalah kriteria film skena yang memenuhi standar Kelkel:

- Kelkel merasa bahwa film yang *skena* adalah film yang tidak bisa dijelaskan dengan kata-kata. Oleh karena itu, **film yang dihasilkan hanya film yang tidak memiliki keyword apapun.**
- Kelkel juga merasa bahwa semakin terkenal suatu film, semakin buruk pula kualitas film tersebut. Oleh karena itu, *popularity* dari film tersebut harus kurang dari tahun rilis film tersebut dikali 0,00001.

Bantulah Kelkel untuk menampilkan ***title*** dan ***popularity*** dari film yang memenuhi kriteria tersebut!

Anda diwajibkan untuk memanfaatkan salah satu keyword OUTER JOIN pada soal ini.

[HINT] Gunakan fungsi dasar SQL untuk mengekstrak bagian dari sebuah tanggal seperti MONTH(), YEAR(), WEEK().

Query	<pre>SELECT m.title, m.popularity FROM movie m WHERE m.popularity < YEAR(m.release_date)*0.00001 EXCEPT SELECT m.title,</pre>
-------	--

	m.popularity FROM movie m NATURAL RIGHT JOIN movie_keywords mk
SS Query dan Hasil	
<pre> +-----+-----+ title popularity +-----+-----+ The Blood of My Brother: A Story of Death in Iraq 0.005256 Hav Plenty 0.003142 The Legend of God's Gun 0.003582 Alien Zone 0.000372 Four Single Fathers 0.008263 Nothing But a Man 0.008453 Sex With Strangers 0.014406 Down & Out With The Dolls 0.002386 Smiling Fish & Goat On Fire 0.007340 Arnolds Park 0.006069 Short Cut to Nirvana: Kumbh Mela 0.004998 The Work and The Story 0.002388 Harrison Montgomery 0.006943 An American in Hollywood 0.015697 Death Calls 0.005883 UnDivided 0.010607 Hum To Mohabbat Karega 0.001186 Light from the Darkroom 0.012942 Amidst the Devil's Wings 0.018087 Midnight Cabaret 0.001389 +-----+-----+ 20 rows in set (0.029 sec) </pre>	

2. Favorite Writer 🙌🙌🙌

Bu Kanny sedang maraton ulang semua film Marvel dari awal hingga terbaru. Di tengah-tengah tontonan, ia mulai merasa ada pola aneh yang terus berulang.

Ia mencurigai bahwa Marvel Studios sengaja memakai penulis yang sama berulang kali, terutama dalam film-film garapan sutradara favorit mereka, untuk menjaga konsistensi cerita dan meningkatkan penghasilan box office.


Untuk membuktikan teorinya, Bu Kanzy ingin mengetahui siapa saja penulis naskah yang:

- Terlibat di departemen Writing
- Sering bekerja sama dengan sutradara (department Directing) Anthony Russo
- Dalam film produksi Marvel Studios
- Dengan pendapatan di atas 300 juta USD
- Dan pernah menulis lebih dari satu film seperti itu

Bantu Bu Kanzy membuat query untuk menampilkan nama penulis dan jumlah filmnya, urutkan dari yang paling sering bekerja sama.

[HINT] Gunakan INNER/OUTER/NATURAL join sesuai dengan kebutuhan soal

Query	<pre>WITH writer(movie_id, person_id) AS (SELECT mc.movie_id, mc.person_id FROM movie_crew mc NATURAL JOIN department d WHERE d.department_name = 'Writing' AND mc.job = 'Writer') , russoMarvelHitMovies(movie_id) AS (SELECT m.movie_id FROM movie m NATURAL JOIN movie_crew mcr NATURAL JOIN department d NATURAL JOIN person p NATURAL JOIN movie_company mco NATURAL JOIN production_company pc WHERE d.department_name = 'Directing' AND p.person_name = 'Anthony Russo' AND pc.company_name = 'Marvel Studios' AND m.revenue > 300000000 AND mcr.job = 'Director') , eligibleWriter(person_id, count) AS (</pre>
-------	---

	<pre> SELECT person_id, COUNT(movie_id) FROM writer w NATURAL JOIN russoMarvelHitMovies r GROUP BY person_id HAVING COUNT(movie_id) > 1) SELECT p.person_name, e.count FROM eligibleWriter e NATURAL JOIN person p ORDER BY e.count DESC; </pre>
SS Query dan Hasil	
	

3. Duolinganguliguliguli Guacha

Setelah lebaran, Kelkelkelkelkel Cicerokel Mokel Ambatokel juga ingin memperluas *skillset* yang dimilikinya dengan mempelajari bahasa baru. Ia sudah menemukan aplikasi yang menurutnya cocok untuk belajar bahasa baru yang bernama Duolinganguliguliguli Guacha. Akan tetapi, ia sedang bingung menentukan bahasa apa yang sebaiknya ia pelajari. Untuk membantunya memilih bahasa yang cocok, Anda ditugaskan untuk membuat *query* dengan ketentuan sebagai berikut:

Untuk setiap pasangan bahasa dan *role*, ia ingin melihat *average vote* yang didapatkan pasangan tersebut. Agar memperkecil pilihan, bahasa yang ditampilkan harus memenuhi **setidaknya salah satu dari dua syarat berikut**:

- Film yang dimasukkan dalam hitungan *average vote* adalah film yang menghasilkan keuntungan setidaknya 199,8%. Persentase keuntungan dilihat dari *budget* membuat film serta *revenue* yang dihasilkan.
- Bahasa yang ditampilkan pada hasil *query* hanya bahasa yang rata-rata filmnya menghasilkan keuntungan setidaknya 2045%.

Untuk setiap pasangan bahasa dan *role*, tampilkan bahasa, *role*, *average vote* film pada pasangan tersebut, serta rata-rata keuntungan yang didapatkan setiap pasangan. Urutkan berdasarkan *average vote* tertinggi.

Query	<pre> WITH language_pair(language_name, language_role, average_vote, revenue, budget, profit) AS (SELECT l.language_name, lr.language_role, m.vote_average, m.revenue, m.budget, m.revenue/m.budget FROM movie m NATURAL JOIN language l NATURAL JOIN language_role lr WHERE m.revenue/m.budget > 1.998) SELECT language_name, language_role, avg(average_vote), avg(profit) FROM language_pair GROUP BY language_name, language_role HAVING avg(profit) > 20.45; </pre>
-------	--

SS Query dan Hasil

```

MariaDB [movies]> WITH language_pair(language_name, language_role, average_vote, revenue, budget, profit) AS (SELECT l.language_name, lr.language_role, m.vote_average, m.revenue, m.budget, m.revenue/m.budget FROM movie m NATURAL JOIN language l NATURAL JOIN language_role lr WHERE m.revenue/m.budget > 1.998)
->
-> SELECT language_name, language_role, avg(average_vote), avg(profit) FROM language_pair GROUP BY language_name, language_role HAVING avg(profit) > 20.45;
+-----+-----+-----+-----+
| language_name | language_role | avg(average_vote) | avg(profit) |
+-----+-----+-----+-----+
| Afrikaans     | Original      | 6.514309          | 5270.61480901 |
| Afrikaans     | Spoken        | 6.514309          | 5270.61480901 |
| Amharic       | Original      | 6.514309          | 5270.61480901 |
| Amharic       | Spoken        | 6.514309          | 5270.61480901 |
| Arabic        | Original      | 6.514309          | 5270.61480901 |
| Arabic        | Spoken        | 6.514309          | 5270.61480901 |
| Bahasa Indonesia | Original      | 6.514309          | 5270.61480901 |
| Bahasa Indonesia | Spoken        | 6.514309          | 5270.61480901 |
| Bamanankan    | Original      | 6.514309          | 5270.61480901 |
| Bamanankan    | Spoken        | 6.514309          | 5270.61480901 |
| Bangla        | Original      | 6.514309          | 5270.61480901 |
| Bangla        | Spoken        | 6.514309          | 5270.61480901 |
| Bod Yig       | Original      | 6.514309          | 5270.61480901 |
| Bod Yig       | Spoken        | 6.514309          | 5270.61480901 |
| Ukrainskyi    | Original      | 6.514309          | 5270.61480901 |
| Ukrainskyi    | Spoken        | 6.514309          | 5270.61480901 |
| Urdu          | Original      | 6.514309          | 5270.61480901 |
| Urdu          | Spoken        | 6.514309          | 5270.61480901 |
| Wolof         | Original      | 6.514309          | 5270.61480901 |
| Wolof         | Spoken        | 6.514309          | 5270.61480901 |
| Yiddish       | Original      | 6.514309          | 5270.61480901 |
| Yiddish       | Spoken        | 6.514309          | 5270.61480901 |
| Zimbabwean    | Original      | 6.514309          | 5270.61480901 |
| Zimbabwean    | Spoken        | 6.514309          | 5270.61480901 |
| Zyanese       | Original      | 6.514309          | 5270.61480901 |
| Zyanese       | Spoken        | 6.514309          | 5270.61480901 |
+-----+-----+-----+-----+
176 rows in set, 1037 warnings (0.212 sec)

```

4. Pemain yang

Seorang *database administrator*, John, ingin mencari nama actor/actress dari movie yang memiliki **votes_avg** diantara 7.2 dan 7.5, jumlah pemain Female lebih banyak dari jumlah pemain Male, dan movie tersebut harus diproduksi oleh negara dengan rata-rata *revenue* dari movie-movie yang diproduksi oleh negara tersebut lebih besar dari rata-rata *budget*-nya, selain negara United States of America. Tampilkan movie id, nama negara, votes_avg, and nama actor/actress. Urutkan hasil berdasarkan votes_avg secara *descending*. Anda diwajibkan menerapkan *subquery* untuk persoalan ini. **Keyword subquery yang diperbolehkan adalah IN** (tidak boleh menggunakan keyword subquery lain).

[HINT] Gunakan AS untuk melakukan penamaan ulang atribut yang ditampilkan.

Query	<pre>SELECT m.movie_id, c.country_name, m.votes_avg, p.person_name FROM movie m NATURAL JOIN production_country pc NATURAL JOIN country c NATURAL JOIN movie_crew mc NATURAL JOIN person p WHERE m.movie_id IN (SELECT movie_id FROM (SELECT mc.movie_id, COUNT(mc.gender_id) FROM movie_cast mc WHERE mc.gender_id = 1 GROUP BY mc.movie_id) f NATURAL JOIN (SELECT mc.movie_id, COUNT(mc.gender_id) FROM movie_cast mc WHERE mc.gender_id = 2 GROUP BY mc.movie_id) m NATURAL JOIN movie mov WHERE f.countFemale > m.countMale AND mov.votes_avg BETWEEN 7.2 AND 7.5) AND</pre>
-------	---

	<pre> pc.country_id IN (SELECT pc.country_id, AVG(m.revenue), AVG(m.budget) FROM production_country pc NATURAL JOIN movie m GROUP BY pc.country_id HAVING AVG(m.revenue) > AVG(m.budget)) AND c.country_name NOT IN ('United States of America'); </pre>
SS Query dan Hasil	

5. Dokumenter Garis Keras

Pak Red, seorang kurator film independen yang dikenal jeli memburu talenta tersembunyi, sedang mengerjakan riset bertajuk “Aktor yang Setia pada Dokumenter.” Ia tertarik mengungkap nama-nama aktor atau aktris yang **benar-benar hanya membintangi film bergenre Documentary**, tanpa pernah menyeberang ke genre lain sepanjang karier filmnya.

Namun bagi Pak Red, loyalitas genre saja belum cukup. Ia juga menetapkan kriteria tambahan:

- Film-film tersebut **harus diproduksi oleh satu negara saja**, tak boleh ada produksi lintas negara.
- Dan, **minimal satu** dari film-film itu **memiliki bahasa tambahan** selain bahasa utama, misalnya dalam bentuk subtitle atau dubbing.

Keyword subquery yang diperbolehkan adalah EXISTS (tidak boleh menggunakan *keyword subquery* lain)

Query	<pre> SELECT p.person_name FROM movie m NATURAL JOIN movie_cast mc NATURAL JOIN person p </pre>
-------	---

	WHERE NOT EXIST (SELECT p.person_name FROM movie m NATURAL JOIN movie_cast mc NATURAL JOIN person p EXCEPT SELECT p.person_name FROM movie m, movie_cast mc, person p WHERE ...
SS Query dan Hasil	

6. Pantheon of the Shackled

D, seorang historian yang sering mengonsumsi media modern sedang tertarik dengan film-film yang bertema sejarah perbudakan. D ingin menonton film-film berkualitas tinggi dengan tema ini dan mencari rekomendasi film yang diproduksi oleh beberapa negara berbeda.

Bantulah D dengan membuat query SQL yang menghasilkan daftar:

1. Judul film
2. Tahun rilis
3. Vote rata-rata
4. Negara-negara produksi (sebagai string yang dipisahkan koma)

Tabel harus menampilkan film-film yang memiliki keyword yang mengandung "**slavery**", diproduksi oleh minimal **2 negara berbeda** dan memiliki rating (vote rata - rata) **di atas** rata-rata rating film lainnya (yang mengandung keyword "slavery"). Urutkan hasil berdasarkan peringkat tertinggi (berdasarkan vote) agar D dapat mulai mengkritik film terbaik terlebih dahulu.

Clue:

1. Gunakan fungsi YEAR(tanggal) untuk mendapatkan tahun dari tanggal, contoh: YEAR(release_date) akan menghasilkan tahun dari tanggal releasedate.

2. Fungsi GROUP_CONCAT(kolom SEPARATOR 'pemisah') dapat digunakan untuk menggabungkan beberapa nilai dari baris yang berbeda menjadi satu string, contoh: GROUP_CONCAT(c.country_name SEPARATOR ', ') akan menggabungkan nama-nama negara yang berbeda menjadi satu string dengan pemisah koma dan spasi (,).

3. keyword CROSS JOIN dapat juga digunakan untuk melakukan cartesian product

Anda diwajibkan untuk menggunakan keyword **WITH** pada soal ini.

Query	SELECT
SS Query dan Hasil	

7. Haus Penghargaan

Pak Red, seorang kritikus film yang kini aktif menulis di blog pribadi, merasa penting untuk mencatat berbagai penghargaan yang diperoleh film-film yang ia tonton. Karena itu, ia ingin membuat dua tabel tambahan dalam basis data film yang ia kelola.

Tabel pertama dinamakan award_category, dengan spesifikasi atribut sebagai berikut:

- Atribut award_category_id, bertipe integer, merupakan primary key.
- Atribut category_name, bertipe varchar dengan batas maksimum 255 karakter.

Tabel kedua dinamakan `movie_awards`, yang mencatat penghargaan yang diterima oleh film tertentu, dengan spesifikasi:

- Atribut `award_id`, bertipe integer, merupakan primary key.
- Atribut `movie_id`, bertipe integer, tidak boleh NULL.
- Atribut `person_id`, bertipe integer, nullable, karena tidak semua penghargaan diberikan ke individu.
- Atribut `award_category_id`, bertipe integer, tidak boleh NULL.
- Atribut `award_year`, bertipe YEAR, tidak boleh NULL.
- Atribut `won`, bertipe boolean, dengan nilai default FALSE.

Pak Red juga ingin agar integritas data terjaga dengan menambahkan foreign key constraint berikut:

- Atribut `movie_id` harus merujuk ke `movie_id` pada tabel `movie`.
- Atribut `person_id` (jika tidak NULL) harus merujuk ke `person_id` pada tabel `person`.
- Atribut `award_category_id` harus merujuk ke `award_category_id` pada tabel `award_category`.

Setelah selesai membuat kedua tabel tersebut, bantu Pak Red mengecek apakah struktur tabel yang kamu buat sudah benar menggunakan perintah SQL `SHOW TABLES` dan `DESCRIBE`.

[HINT] Gunakan tambahan tag ``auto_increment`` sebagai fungsi initial default value pada primary key

Query	SELECT
-------	--------

Jawaban	
Query Testing (diberikan kepada Praktikan)	<pre>show tables; desc award_category; desc movie_awards;</pre>
SS Query Sebelum Manipulasi Menggunakan Query Testing	
SS Query Saat Manipulasi Menggunakan Query Jawaban	
SS Query Setelah Manipulasi Menggunakan Query Testing	

8. Efisiensi! Efisiensi! Efisiensi!!!

Bu Kanzy adalah direktur keuangan di sebuah perusahaan produksi film yang terkenal ketat soal anggaran. Ia baru saja menemukan bahwa beberapa film yang melibatkan komposer (department 'Sound') ternama Hans Zimmer, terutama yang bergenre Action dan berdurasi panjang (> 120 menit), ternyata menyedot anggaran sangat besar.

Yang membuatnya makin jengkel, film-film ini juga:

- Diproduksi oleh perusahaan produksi kecil yang hanya pernah membuat maksimal 5 film
- Menggunakan bahasa-bahasa langka yang hanya muncul di kurang dari 50 film

Menurut Bu Kanzy, ini kombinasi yang boros dan tidak efisien. Maka tanpa banyak bicara, ia langsung memberi instruksi untuk:

Memotong anggaran (*budget*) sebesar **20%** untuk semua film yang memenuhi kriteria di atas.

Query Jawaban	SELECT
Query Testing (Diberikan kepada Praktikan)	<pre> SELECT m.movie_id, m.title, m.budget, m.runtime FROM movie m JOIN movie_crew mc ON m.movie_id = mc.movie_id JOIN person p ON mc.person_id = p.person_id JOIN movie_genres mg ON m.movie_id = mg.movie_id JOIN genre g ON mg.genre_id = g.genre_id WHERE p.person_name = 'Hans Zimmer' AND g.genre_name = 'Action'; </pre>
SS Query Sebelum Manipulasi Menggunakan Query Testing	
SS Query Saat Manipulasi Menggunakan Query Jawaban	
SS Query Setaah Manipulasi Menggunakan Query Testing	

9. Bersih-bersih

Di tengah program magangnya di sebuah studio perfilman ternama, Raka seorang mahasiswa tingkat akhir jurusan perfilman diberi akses penuh ke sistem manajemen data produksi film. Selama mengevaluasi performa beberapa film, ia merasa resah karena banyak film dengan kualitas buruk ternyata melibatkan kru-kru yang sama berulang kali.

Ia kemudian menemukan pola yang menarik yaitu film-film dengan genre thriller atau crime, nilai rata-rata vote di bawah 6, dan diproduksi oleh perusahaan yang

tidak mengandung kata 'Indie' dalam namanya, cenderung menampilkan hasil yang mengecewakan. Setelah dicek lebih lanjut, sebagian besar film buruk tersebut melibatkan kru dengan job 'Director' atau yang berasal dari departemen 'Writing'.

Dengan semangat memperbaiki mutu perfilman, Raka memutuskan untuk mulai membersihkan tabel `movie_crew` dari kru-kru yang berperan dalam proyek-proyek dengan karakteristik tersebut.

Bantulah Raka untuk menyusun query penghapusan yang menghapus data kru pada `movie_crew` yang ikut serta dalam film dengan kriteria yang telah dijelaskan.

Query Jawaban	DELETE ...
Query Testing (Diberikan kepada Praktikan)	<pre> SELECT mc.person_id, mc.job, d.department_name, COUNT(DISTINCT m.movie_id) AS total_film_mencurigakan FROM movie_crew mc JOIN department d ON mc.department_id = d.department_id JOIN movie m ON mc.movie_id = m.movie_id JOIN movie_company mcomp ON m.movie_id = mcomp.movie_id JOIN production_company pc ON mcomp.company_id = pc.company_id WHERE m.vote_average < 6.0 AND pc.company_name NOT LIKE '%Indie%' AND (mc.job = 'Director' OR d.department_name = 'Writing') AND EXISTS (SELECT 1 FROM movie_genres mg JOIN genre g ON mg.genre_id = g.genre_id WHERE mg.movie_id = m.movie_id AND g.genre_name IN ('Thriller', 'Crime')) </pre>

	GROUP BY mc.person_id, mc.job, d.department_name HAVING total_film_mencurigakan > 0 ORDER BY total_film_mencurigakan DESC;
SS Query Sebelum Manipulasi Menggunakan Query Testing	
SS Query Saat Manipulasi Menggunakan Query Jawaban	
SS Query Setelah Manipulasi Menggunakan Query Testing	

10. Film Baru

Seorang database administrator, John ingin memasukkan sebuah film lama yang belum terdata di database dengan judul Praktikum 2 dan vote average 10. Bantu John membuat query untuk memasukkan film tersebut ke dalam database. Atribut lain selain yang disebutkan dibebaskan (Jangan NULL).

Setelah film berhasil dimasukkan, masukkan juga para pemain film tersebut ke dalam tabel movie_cast.

- A. Para pemain tersebut adalah pemain-pemain yang didapatkan pada soal nomor 4. Character_name dan cast_order dibebaskan, namun sisanya harus sesuai.
- B. Silahkan insert 2 (3 jika mengerjakan ber 3) pemain baru ke dalam tabel Person dengan ketentuan person_name adalah nama praktikan. Lalu masukkan 3 pemain tersebut sebagai movie_cast untuk film tersebut.

Silahkan pilih salah satu soal A atau B.

Query Jawaban 1	SELECT
Query Jawaban 2	SELECT
Query Testing 1 (diberikan kepada Praktikan)	SELECT * from movie where title = 'Praktikum 2';
Query Testing 2 (diberikan kepada Praktikan)	<p>Untuk soal B jalankan query ini terlebih dahulu</p> <pre> SELECT person_id, person_name FROM person WHERE person_id > (masukkan person id) SELECT movie_id, COUNT(*) AS total_cast FROM movie_cast NATURAL JOIN movie WHERE title = 'Praktikum 2' GROUP BY movie_id; </pre>
SS Query Saat Manipulasi Menggunakan Query Jawaban 1	
SS Query Setelah Manipulasi Menggunakan Query Testing 1	
SS Query Saat Manipulasi Menggunakan Query Jawaban 2	

SS Query Setelah Manipulasi Menggunakan Query Testing 2