

PROJECT SQL QUERY

FEBRUARY 2

Authored by: FERDYANSYAH PERMANA PUTRA



Pada Dataset **dvdrental**

Mendapatkan **nama lengkap** customers dan waktu pengembalian untuk semua transaksi yang melebihi batas waktu pengembalian (7 hari)

Jawab:

Query dan Output →

-- 1. Dapatkan nama lengkap customers dan waktu pengembalian untuk semua transaksi yang melebihi batas waktu pengembalian (7 hari)

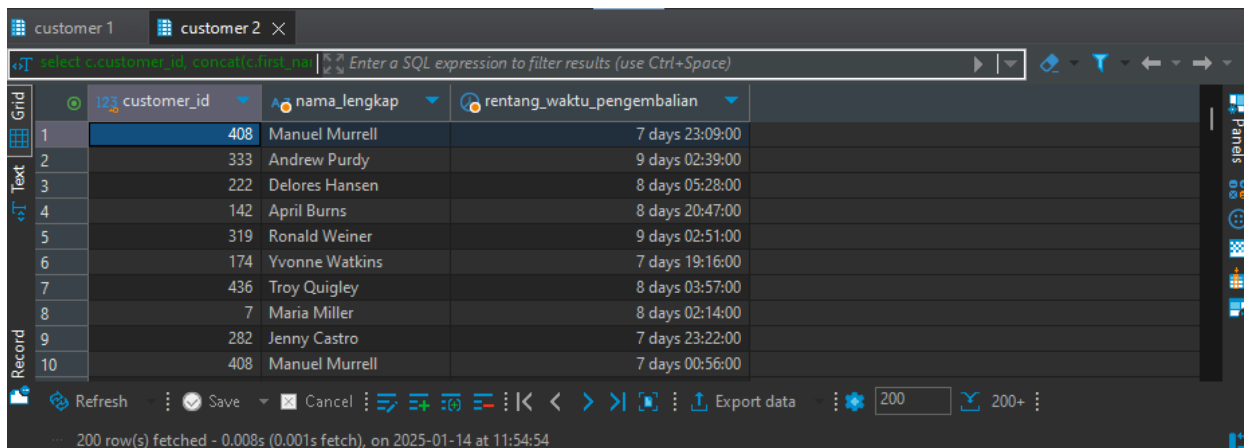
-- cek data

```
select * from customer;
```

```
select * from rental;
```

--## query

```
select
    c.customer_id,
    concat(c.first_name, ' ', c.last_name) as nama_lengkap,
    r.return_date - r.rental_date as pengembalian
from
    customer c
inner join
    rental r on c.customer_id = r.customer_id
where
    r.return_date - r.rental_date > '7 days' ;
```



Grid	customer_id	nama_lengkap	rentang_waktu_pengembalian
1	408	Manuel Murrell	7 days 23:09:00
2	333	Andrew Purdy	9 days 02:39:00
3	222	Delores Hansen	8 days 05:28:00
4	142	April Burns	8 days 20:47:00
5	319	Ronald Weiner	9 days 02:51:00
6	174	Yvonne Watkins	7 days 19:16:00
7	436	Troy Quigley	8 days 03:57:00
8	7	Maria Miller	8 days 02:14:00
9	282	Jenny Castro	7 days 23:22:00
10	408	Manuel Murrell	7 days 00:56:00

Menampilkan nama pelanggan yang melakukan transaksi peminjaman lebih dari sekali pada hari Senin!

Gunakan CTE!

Jawab:

Query dan Output →

--2. Tampilkan nama pelanggan yang melakukan transaksi peminjaman lebih dari sekali pada hari Senin! Gunakan CTE!

-- cek data

```
select * from customer;
```

```
select * from rental;
```

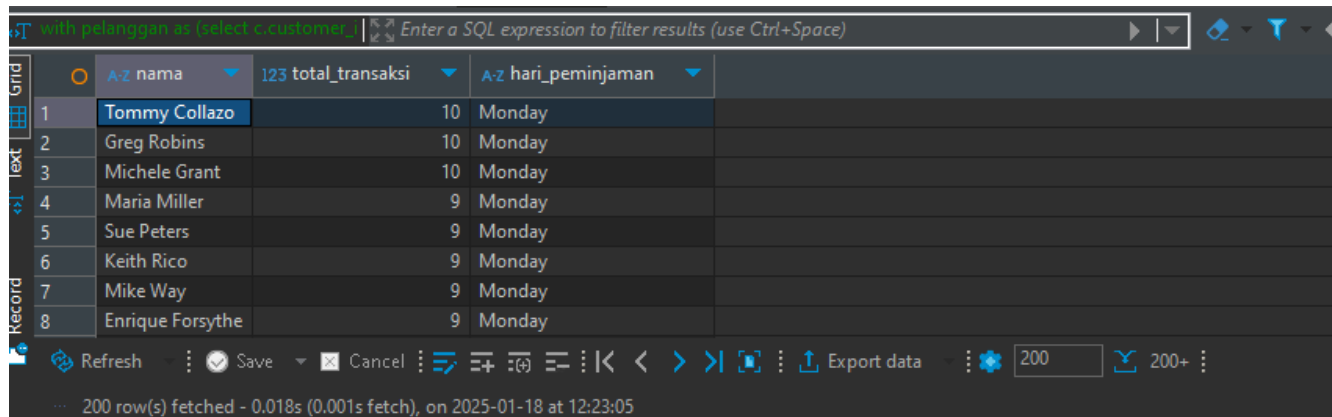
-- query

```
with pelanggan as (select c.customer_id,
    concat(c.first_name, ' ', c.last_name) as Nama,
    count(r.rental_id) as Total_transaksi,
    to_char(r.rental_date, 'Day') as Hari_Peminjaman
from
```

```

        customer c
    inner join rental r
    on c.customer_id = r.customer_id
    where trim(to_char(r.rental_date, 'Day')) = 'Monday'
    group by c.customer_id, to_char(r.rental_date, 'Day')
)
select Nama, Total_transaksi, Hari_Peminjaman
from pelanggan
where Total_transaksi > 1
order by Total_transaksi desc;

```



The screenshot shows a SQL query execution interface. The query is: `with pelanggan as (select c.customer_id, c.first_name || ' ' || c.last_name as Nama, count(r.rental_id) as Total_transaksi, to_char(r.rental_date, 'Day') as Hari_Peminjaman from customer c inner join rental r on c.customer_id = r.customer_id where trim(to_char(r.rental_date, 'Day')) = 'Monday' group by c.customer_id, to_char(r.rental_date, 'Day')) select Nama, Total_transaksi, Hari_Peminjaman from pelanggan where Total_transaksi > 1 order by Total_transaksi desc;` The results are displayed in a table with 8 rows. The first row is highlighted.

	A-z nama	123 total_transaksi	A-z hari_peminjaman
1	Tommy Collazo	10	Monday
2	Greg Robins	10	Monday
3	Michele Grant	10	Monday
4	Maria Miller	9	Monday
5	Sue Peters	9	Monday
6	Keith Rico	9	Monday
7	Mike Way	9	Monday
8	Enrique Forsythe	9	Monday

200 row(s) fetched - 0.018s (0.001s fetch), on 2025-01-18 at 12:23:05

Menemukan nama aktor dan jumlah film yang dimainkan, serta peringkat aktor berdasarkan jumlah film. Urutkan berdasarkan peringkat secara **ascending**. Gunakan **RANK!**

Query dan Output →

-- 3. Temukan nama aktor dan jumlah film yang dimainkan, serta peringkat aktor berdasarkan jumlah film. Urutkan berdasarkan peringkat secara ascending. Gunakan RANK!
 -- cek data

```

select * from actor;
select * from film;
select * from film_actor;

-- query
select a.actor_id, concat(a.first_name, ' ', a.last_name) as Nama, count(f.film_id) as
Total_Film,
       rank() over (order by count(f.film_id) desc) as ranked
from actor a
inner join film_actor ff
on ff.actor_id = a.actor_id
inner join film f
on ff.film_id = f.film_id
group by a.actor_id, nama
order by ranked asc;

```

The screenshot shows a SQL query editor with a table of actor data. The table has columns for actor_id, nama (name), total_film, and ranked. The data is sorted by total_film in descending order. The first row shows Gina Degeneres with 42 films and a rank of 1. The second row shows Walter Torn with 41 films and a rank of 2. The third row shows Mary Keitel with 40 films and a rank of 3. The fourth row shows Matthew Carrey with 39 films and a rank of 4. The fifth row shows Sandra Kilmer with 37 films and a rank of 5. The sixth row shows Scarlett Damon with 36 films and a rank of 6. The seventh row shows Henry Berry with 35 films and a rank of 7. The eighth row shows Vivien Basinger with 35 films and a rank of 7. The ninth row shows Groucho Dunst with 35 films and a rank of 7. The tenth row shows Uma Wood with 35 films and a rank of 7. The eleventh row shows Angela Witherspoc with 35 films and a rank of 7.

	actor_id	nama	total_film	ranked
1	107	Gina Degeneres	42	1
2	102	Walter Torn	41	2
3	198	Mary Keitel	40	3
4	181	Matthew Carrey	39	4
5	23	Sandra Kilmer	37	5
6	81	Scarlett Damon	36	6
7	60	Henry Berry	35	7
8	158	Vivien Basinger	35	7
9	106	Groucho Dunst	35	7
10	13	Uma Wood	35	7
11	144	Angela Witherspoc	35	7

Note*: ini memakai rank() dimana jika ada dua actor atau lebih yang memiliki jumlah film yang sama maka memiliki peringkat yang sama dari actor-aktor tersebut, tetapi peringkat berikutnya akan terlewat.

Menggunakan Versi dense_rank() -> meskipun jumlah film dari setiap actor sama, maka peringkat berikutnya tidak akan terlewat.

```
-- query
select a.actor_id, concat(a.first_name, ' ', a.last_name) as Nama, count(f.film_id) as
Total_Film,
       dense_rank() over (order by count(f.film_id) desc) as ranked
from actor a
inner join film_actor ff
on ff.actor_id = a.actor_id
inner join film f
on ff.film_id = f.film_id
group by a.actor_id, nama
order by ranked asc;
```

SQL: `select a.actor_id, concat(a.first_name, ' ' || a.last_name) as nama, a.total_films as total_film, a.rank as ranked`

	actor_id	nama	total_film	ranked
1	107	Gina Degeneres	42	1
2	102	Walter Torn	41	2
3	198	Mary Keitel	40	3
4	181	Matthew Carrey	39	4
5	23	Sandra Kilmer	37	5
6	81	Scarlett Damon	36	6
7	60	Henry Berry	35	7
8	158	Vivien Basinger	35	7
9	106	Groucho Dunst	35	7
10	13	Uma Wood	35	7
11	144	Angela Witherspoc	35	7

200 row(s) fetched: 0.016s (0.016s fetch) on 2025-01-18 at 12:50:27

*Peringkat berikutnya tidak hilang, ketika dua actor atau lebih memiliki jumlah total film yang sama.

Pada dataset DS Salaries

Menampilkan job_title yang memiliki salary_in_usd lebih besar dari rata-rata salary dari seluruh job_title. Namun, tampilkan hanya company_size = S. **Gunakan Subquery!**

Query dan Output →

-- 4. Tampilkan job_title yang memiliki salary_in_usd lebih besar dari rata-rata salary dari seluruh job_title. Namun, tampilkan hanya company_size = S. Gunakan Subquery!

-- cek data

```
select * from ds_salaries ds ;
```

-- query

```
select job_title, salary_in_usd, company_size
from ds_salaries ds
```

```
where salary_in_usd > (select avg(salary_in_usd) from ds_salaries ds) and company_size = 'S'
```

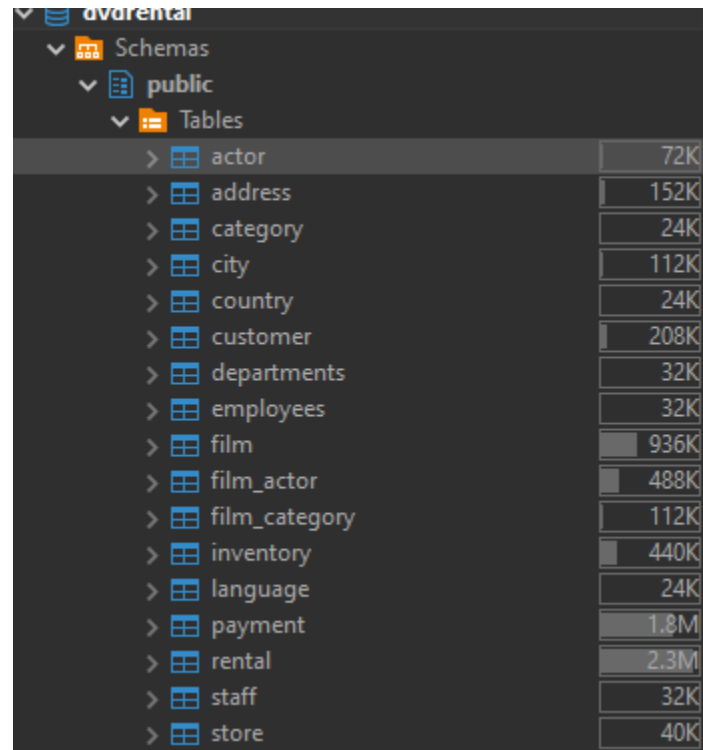
```
order by salary_in_usd desc;
```

SQL: `select job_title, salary_in_usd, company_size`

	job_title	salary_in_usd	company_size
1	Principal Data Scientist	416,000	S
2	Machine Learning Scientist	260,000	S
3	ML Engineer	256,000	S
4	Lead Data Scientist	190,000	S
5	Director of Data Science	168,000	S
6	Lead Data Engineer	160,000	S
7	Cloud Data Engineer	160,000	S
8	Computer Vision Software Engineer	150,000	S
9	Machine Learning Engineer	138,000	S
10	Lead Data Engineer	125,000	S
11	Machine Learning Engineer	125,000	S

200 row(s) fetched: 0.016s (0.016s fetch) on 2025-01-18 at 12:50:27

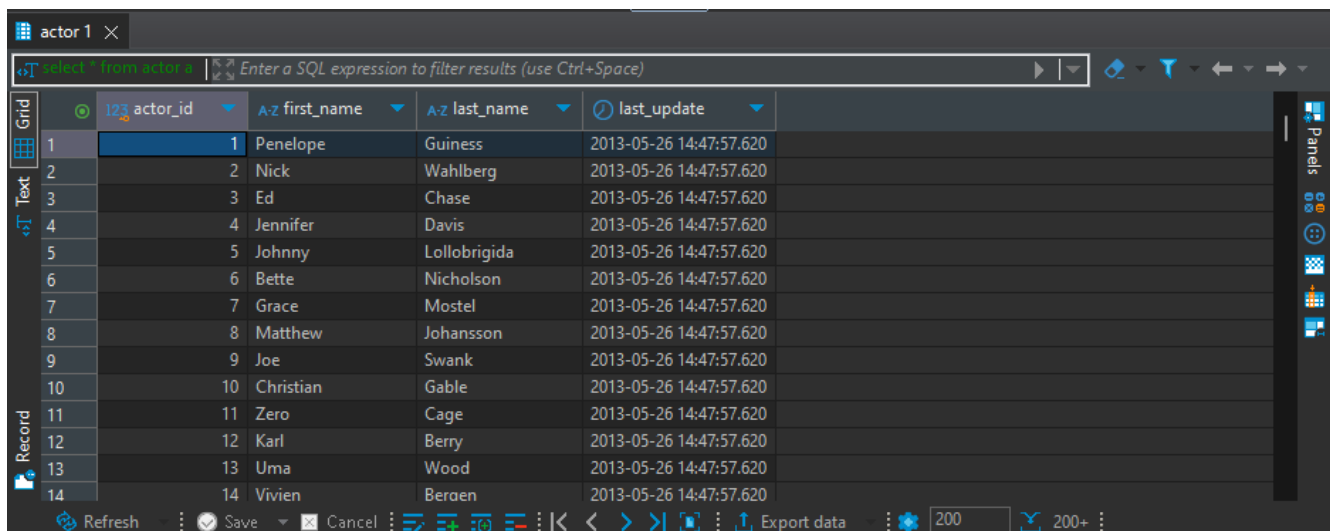
Dataset dvdrental



dvdrental	
Schemas	
public	
Tables	
actor	72K
address	152K
category	24K
city	112K
country	24K
customer	208K
departments	32K
employees	32K
film	936K
film_actor	488K
film_category	112K
inventory	440K
language	24K
payment	1.8M
rental	2.3M
staff	32K
store	40K

Salah satu tabel dari dvdrental →

Tabel actor



actor 1 X				
select * from actor a				
	actor_id	first_name	last_name	last_update
1	1	Penelope	Guinness	2013-05-26 14:47:57.620
2	2	Nick	Wahlberg	2013-05-26 14:47:57.620
3	3	Ed	Chase	2013-05-26 14:47:57.620
4	4	Jennifer	Davis	2013-05-26 14:47:57.620
5	5	Johnny	Lollobrigida	2013-05-26 14:47:57.620
6	6	Bette	Nicholson	2013-05-26 14:47:57.620
7	7	Grace	Mostel	2013-05-26 14:47:57.620
8	8	Matthew	Johansson	2013-05-26 14:47:57.620
9	9	Joe	Swank	2013-05-26 14:47:57.620
10	10	Christian	Gable	2013-05-26 14:47:57.620
11	11	Zero	Cage	2013-05-26 14:47:57.620
12	12	Karl	Berry	2013-05-26 14:47:57.620
13	13	Uma	Wood	2013-05-26 14:47:57.620
14	14	Vivien	Beraen	2013-05-26 14:47:57.620

Dataset DS Salaries

ds_salaries 1

select * from ds_salaries ds

Enter a SQL expression to filter results (use Ctrl+Space)

	123 id	123 work_year	A-z experience_level	A-z employment_type	A-z job_title	123 salary	A-z salary_currency
1	0	2,020	MI	FT	Data Scientist	70,000	EUR
2	1	2,020	SE	FT	Machine Learning Scientist	260,000	USD
3	2	2,020	SE	FT	Big Data Engineer	85,000	GBP
4	3	2,020	MI	FT	Product Data Analyst	20,000	USD
5	4	2,020	SE	FT	Machine Learning Engineer	150,000	USD
6	5	2,020	EN	FT	Data Analyst	72,000	USD
7	6	2,020	SE	FT	Lead Data Scientist	190,000	USD
8	7	2,020	MI	FT	Data Scientist	11,000,000	HUF
9	8	2,020	MI	FT	Business Data Analyst	135,000	USD
10	9	2,020	SE	FT	Lead Data Engineer	125,000	USD
11	10	2,020	EN	FT	Data Scientist	45,000	EUR
12	11	2,020	MI	FT	Data Scientist	3,000,000	INR
13	12	2,020	EN	FT	Data Scientist	35,000	EUR
14	13	2,020	MI	FT	Lead Data Analyst	87,000	USD
15	14	2,020	MI	FT	Data Analyst	85,000	USD
16	15	2,020	MI	FT	Data Analyst	8,000	USD
17	16	2,020	EN	FT	Data Engineer	4,450,000	JPY
18	17	2,020	SE	FT	Big Data Engineer	100,000	EUR
19	18	2,020	EN	FT	Data Science Consultant	423,000	INR
20	19	2,020	MI	FT	Lead Data Engineer	56,000	USD
21	20	2,020	MI	FT	Machine Learning Engineer	299,000	CNY

Refresh Save Cancel Export data 200 200+

ds_salaries 1

select * from ds_salaries ds

Enter a SQL expression to filter results (use Ctrl+Space)

	ent_type	A-z job_title	123 salary	A-z salary_currency	123 salary_in_usd	A-z employee_residence	123 remote
1		Data Scientist	70,000	EUR	79,833	DE	
2		Machine Learning Scientist	260,000	USD	260,000	JP	
3		Big Data Engineer	85,000	GBP	109,024	GB	
4		Product Data Analyst	20,000	USD	20,000	HN	
5		Machine Learning Engineer	150,000	USD	150,000	US	
6		Data Analyst	72,000	USD	72,000	US	
7		Lead Data Scientist	190,000	USD	190,000	US	
8		Data Scientist	11,000,000	HUF	35,735	HU	
9		Business Data Analyst	135,000	USD	135,000	US	
10		Lead Data Engineer	125,000	USD	125,000	NZ	
11		Data Scientist	45,000	EUR	51,321	FR	
12		Data Scientist	3,000,000	INR	40,481	IN	
13		Data Scientist	35,000	EUR	39,916	FR	
14		Lead Data Analyst	87,000	USD	87,000	US	
15		Data Analyst	85,000	USD	85,000	US	
16		Data Analyst	8,000	USD	8,000	PK	
17		Data Engineer	4,450,000	JPY	41,689	JP	
18		Big Data Engineer	100,000	EUR	114,047	PL	
19		Data Science Consultant	423,000	INR	5,707	IN	
20		Lead Data Engineer	56,000	USD	56,000	PT	
21		Machine Learning Engineer	299,000	CNY	43,331	CN	

Refresh Save Cancel Export data 200 200+

ds_salaries 1 ✕

select * from ds_salaries ds Enter a SQL expression to filter results (use Ctrl+Space)

	currency	123 salary_in_usd	A-Z employee_residence	123 remote_ratio	A-Z company_location	A-Z company_size
1		79,833	DE	0	DE	L
2		260,000	JP	0	JP	S
3		109,024	GB	50	GB	M
4		20,000	HN	0	HN	S
5		150,000	US	50	US	L
6		72,000	US	100	US	L
7		190,000	US	100	US	S
8		35,735	HU	50	HU	L
9		135,000	US	100	US	L
10		125,000	NZ	50	NZ	S
11		51,321	FR	0	FR	S
12		40,481	IN	0	IN	L
13		39,916	FR	0	FR	M
14		87,000	US	100	US	L
15		85,000	US	100	US	L
16		8,000	PK	50	PK	L
17		41,689	JP	100	JP	S
18		114,047	PL	100	GB	S
19		5,707	IN	50	IN	M
20		56,000	PT	100	US	M
21		43,331	CN	0	CN	M

Refresh Save Cancel Export data 200 200+