

Follow Up Questions

Siapa saja pelanggan yang hanya membeli lagu dari satu genre saja, dan genre apa itu?

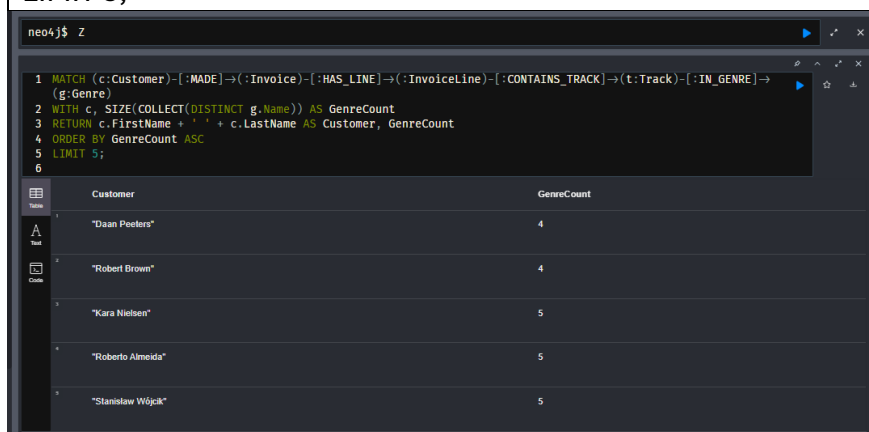
tidak ada pelanggan yang hanya membeli dari satu genre saja, semua pelanggan membeli lagu dari setidaknya 4 genre berbeda.

Query ini digunakan untuk menemukan pelanggan yang membeli lagu dari jumlah genre terbatas, dalam hal ini pelanggan yang membeli lagu dari hanya beberapa genre.

Dengan mengurutkan berdasarkan GenreCount, kita bisa melihat siapa pelanggan yang memiliki keanekaragaman genre terkecil dalam pembelian mereka.

Kita bisa cari pelanggan yang punya jumlah genre paling sedikit:

```
MATCH (c:Customer)-[:MADE]->(i:Invoice)-[:HAS_LINE]->(il:InvoiceLine)-[:CONTAINS_TRACK]->(t:Track)-[:IN_GENRE]->(g:Genre)
WITH c, SIZE(COLLECT(DISTINCT g.Name)) AS GenreCount
RETURN c.FirstName + ' ' + c.LastName AS Customer, GenreCount
ORDER BY GenreCount ASC
LIMIT 5;
```



Customer	GenreCount
"Daan Peeters"	4
"Robert Brown"	4
"Kara Nielsen"	5
"Roberto Almeida"	5
"Stanislaw Wójcik"	5

Komposer mana yang paling banyak menciptakan lagu untuk genre tertentu?

MATCH (t:Track)

→ Ambil semua node Track.

WHERE t.Composer IS NOT NULL

→ Hanya pilih track yang memiliki nama komposer (tidak null).

RETURN t.Composer, COUNT(*)

→ Tampilkan nama komposer dan hitung berapa banyak track yang mereka buat.

ORDER BY COUNT(*) DESC

Urutkan dari yang paling banyak ke yang paling sedikit.

LIMIT 10

→ Tampilkan hanya 10 komposer teratas.

```

MATCH (t:Track)-[:IN_GENRE]->(g:Genre)
WHERE t.Composer IS NOT NULL
WITH g.Name AS Genre, t.Composer AS Composer, COUNT(*) AS TrackCount
ORDER BY Genre, TrackCount DESC
WITH Genre, COLLECT({composer: Composer, count: TrackCount}) AS ComposerList
RETURN Genre, ComposerList[0].composer AS TopComposer,
ComposerList[0].count AS TracksCreated

```

```

1 MATCH (t:Track)
2 WHERE t.Composer IS NOT NULL
3 RETURN t.Composer AS Composer, COUNT(*) AS TotalTracks
4 ORDER BY TotalTracks DESC
5 LIMIT 10
6

```

	Composer	TotalTracks
1	"Steve Harris"	80
2	"U2"	44
3	"JaggerRichards"	35
4	"Billy Corgan"	31
5	"Kurt Cobain"	26
6	"Bill Berry-Peter Dinklage-Mike Mills-Michael Stipe"	25

Siapa pelanggan dengan total pengeluaran paling tinggi?

Query ini akan mengembalikan satu pelanggan yang memiliki total pengeluaran paling tinggi, bersama dengan nama lengkapnya.

```

MATCH (c:Customer)-[:MADE]->(i:Invoice)
WITH c, SUM(i.Total) AS TotalSpent
RETURN c.FirstName + ' ' + c.LastName AS Customer
ORDER BY TotalSpent DESC
LIMIT 1;

```

```

1 MATCH (c:Customer)-[:MADE]->(i:Invoice)
2 WITH c, SUM(i.Total) AS TotalSpent
3 RETURN c.FirstName + ' ' + c.LastName AS Customer
4 ORDER BY TotalSpent DESC
5 LIMIT 1;
6
7

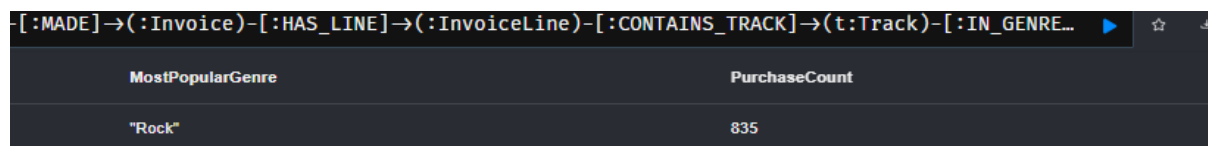
```

	Customer
1	"Luís Gonçalves"

Genre mana yang paling populer di setiap negara?

Query ini bertujuan untuk menganalisis data pembelian musik berdasarkan genre di setiap negara dan menentukan genre musik paling populer di setiap negara berdasarkan jumlah pembelian yang dilakukan oleh pelanggan di sistem.

```
MATCH (c:Customer)-[:MADE]->(:Invoice)-[:HAS_LINE]->(:InvoiceLine)-[:CONTAINS_TRACK]->(t:Track)-[:IN_GENRE]->(g:Genre)
WITH c.Country AS Country, g.Name AS Genre, COUNT(*) AS Count
ORDER BY Country, Count DESC
WITH Country, COLLECT({genre: Genre, count: Count}) AS GenreStats
RETURN Country, GenreStats[0].genre AS MostPopularGenre, GenreStats[0].count AS PurchaseCount
```



The screenshot shows a query result table with two columns: 'MostPopularGenre' and 'PurchaseCount'. The first row of data shows 'Rock' as the most popular genre with a purchase count of 835.

MostPopularGenre	PurchaseCount
"Rock"	835

Rekomendasikan lagu berdasarkan pelanggan yang membeli lagu serupa

Tujuan dari query ini adalah untuk memberikan rekomendasi lagu kepada pelanggan berdasarkan perilaku pembelian pelanggan lain yang memiliki pola serupa. Sistem ini menggunakan pendekatan Collaborative Filtering di mana rekomendasi dibuat berdasarkan kesamaan preferensi antar pelanggan.

```
MATCH (c1:Customer)-[:MADE]->(:Invoice)-[:HAS_LINE]->(:InvoiceLine)-[:CONTAINS_TRACK]->(t:Track)
WITH c1, t
MATCH (c2:Customer)-[:MADE]->(:Invoice)-[:HAS_LINE]->(:InvoiceLine)-[:CONTAINS_TRACK]->(t)
WHERE c1 <> c2
WITH c1, c2, COLLECT(DISTINCT t.TrackId) AS SharedTracks
MATCH (c2)-[:MADE]->(:Invoice)-[:HAS_LINE]->(:InvoiceLine)-[:CONTAINS_TRACK]->(rec:Track)
WHERE NOT ( (c1)-[:MADE]->(:Invoice)-[:HAS_LINE]->(:InvoiceLine)-[:CONTAINS_TRACK]->(rec) )
WITH c1, rec, COUNT(*) AS Score
ORDER BY Score DESC
RETURN DISTINCT c1.FirstName + ' ' + c1.LastName AS Customer, rec.Name AS RecommendedTrack
LIMIT 10
```

```
1 MATCH (c1:Customer)-[:MADE]→(:Invoice)-[:HAS_LINE]→(:InvoiceLine)-[:CONTAINS_TRACK]→(t:Track)
2 WITH c1, t
3 MATCH (c2:Customer)-[:MADE]→(:Invoice)-[:HAS_LINE]→(:InvoiceLine)-[:CONTAINS_TRACK]→(t)
4 WHERE c1 <> c2
5 WITH c1, c2, COLLECT(DISTINCT t.TrackId) AS SharedTracks
6
7 MATCH (c2)-[:MADE]→(:Invoice)-[:HAS_LINE]→(:InvoiceLine)-[:CONTAINS_TRACK]→(rec:Track)
8 WHERE NOT ( (c1)-[:MADE]→(:Invoice)-[:HAS_LINE]→(:InvoiceLine)-[:CONTAINS_TRACK]→(rec) )
9 WITH c1, rec, COUNT(*) AS Score
10 ORDER BY Score DESC
11 RETURN DISTINCT c1.FirstName + ' ' + c1.LastName AS Customer, rec.Name AS RecommendedTrack
12 LIMIT 10
```

	Customer	RecommendedTrack
1	"Daan Peeters"	"Balls to the Wall"
2	"Eduardo Martins"	"Tanto Tempo"
3	"Frank Harris"	"Plot 180"
4	"Frank Harris"	"Big Wave"
5	"Frank Harris"	"Untitled"
6	"Ellie Sullivan"	"Boris The Spider"