CREATE Questions (with Properties)

1.**Create user nodes: Alice, Bob, Charlie with emails.**

```
CREATE
  (a:User {name: 'Alice', email: 'alice@example.com'}),
  (b:User {name: 'Bob', email: 'bob@example.com'}),
  (c:User {name: 'Charlie', email: 'charlie@example.com'});
```

2**. Create product nodes: Product1 (Electronics, 49.99), Product2 (Books, 29.99), Product3 (Clothing, 39.99).**

```
CREATE
  (p1:Product {name: 'Product1', category: 'Electronics', price: 49.99}),
  (p2:Product {name: 'Product2', category: 'Books', price: 29.99}),
  (p3:Product {name: 'Product3', category: 'Clothing', price: 39.99});
```

3. **Create category nodes: Electronics, Books, Clothing.**

```
CREATE
  (c1:Category {name: 'Electronics'}),
  (c2:Category {name: 'Books'}),
  (c3:Category {name: 'Clothing'});
```

4. **Create brand nodes: BrandA, BrandB, BrandC.**

```
CREATE
  (b1:Brand {name: 'BrandA'}),
  (b2:Brand {name: 'BrandB'}),
  (b3:Brand {name: 'BrandC'});
```

5. **Create review nodes: Review1 (rating 5), Review2 (rating 4), Review3 (rating 3).**

```
CREATE
  (r1:Review {name: 'Review1', rating: 5}),
  (r2:Review {name: 'Review2', rating: 4}),
  (r3:Review {name: 'Review3', rating: 3});
```

6. **Create BOUGHT relationship: Alice bought Product1 on 2025-10-14, quantity 2, price_paid 99.98.**

```
MATCH (a:User {name: 'Alice'}), (p:Product {name: 'Product1'})
CREATE (a)-[:BOUGHT {date: date('2025-10-14'), quantity: 2, price_paid: 99.98}]->(p);
```

7**. Create BOUGHT relationship: Bob bought Product2 on 2025-10-13, quantity 1, price_paid 29.99.**

```
MATCH (b:User {name: 'Bob'}), (p:Product {name: 'Product2'})
CREATE (b)-[:BOUGHT {date: date('2025-10-13'), quantity: 1, price_paid: 29.99}]->(p);
```

8. Create VIEWED relationship: Charlie viewed Product3 on 2025-10-12 for 120 seconds on mobile.

```
MATCH (c:User {name: 'Charlie'}), (p:Product {name: 'Product3'})
CREATE (c)-[:VIEWED {date: date('2025-10-12'), duration_seconds: 120, device: 'mobile'}]->(p);
```

9. Create BELONGS_TO relationship: Product1 belongs to Electronics, added_date 2025-01-01.

```
MATCH (p:Product {name: 'Product1'}), (c:Category {name: 'Electronics'})
CREATE (p)-[:BELONGS_TO {added_date: date('2025-01-01')}]->(c);
```

10. Create MADE_BY relationship: Product1 made by BrandA, launch_year 2024.

```
MATCH (p:Product {name: 'Product1'}), (b:Brand {name: 'BrandA'})
CREATE (p)-[:MADE_BY {launch_year: 2024}]->(b);
```

11. Create RATED relationship: Alice rated Review1 on 2025-10-14; Review1 reviews Product1.

```
MATCH (a:User {name: 'Alice'}), (r:Review {name: 'Review1'}), (p:Product {name: 'Product1'})
CREATE (a)-[:RATED {rating: 5, review_date: date('2025-10-14')}]->(r),
    (r)-[:REVIEWS]->(p);
```

12. Create SIMILAR_TO relationship: Product1 is similar to Product3, similarity_score 0.85.

```
MATCH (p1:Product {name: 'Product1'}), (p3:Product {name: 'Product3'})
CREATE (p1)-[:SIMILAR_TO {similarity_score: 0.85}]->(p3);
```

13. Create FRIENDS_WITH relationship: Alice friends with Bob since 2023-01-01, interaction_count 15.

```
MATCH (a:User {name: 'Alice'}), (b:User {name: 'Bob'})
CREATE (a)-[:FRIENDS_WITH {since: date('2023-01-01'), interaction_count: 15}]->(b);
```

14. Create BOUGHT relationship: Charlie bought Product2 on 2025-10-14, quantity 1, price_paid 29.99.

```
MATCH (c:User {name: 'Charlie'}), (p:Product {name: 'Product2'})
CREATE (c)-[:BOUGHT {date: date('2025-10-14'), quantity: 1, price_paid: 29.99}]->(p);
```

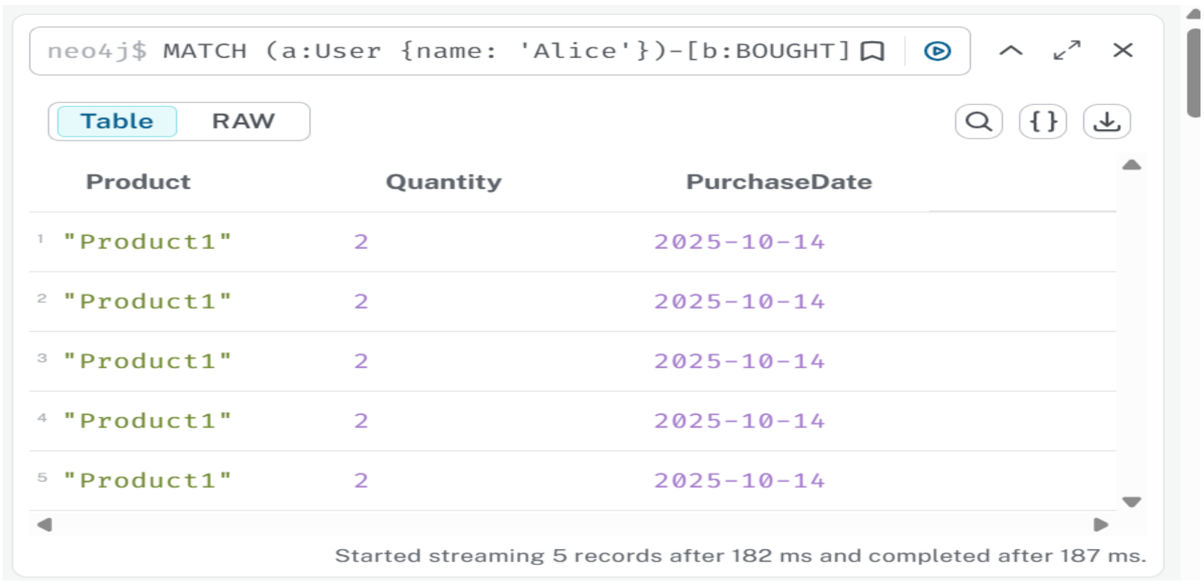
15. Create VIEWED relationship: Alice viewed Product2 on 2025-10-13 for 45 seconds on desktop.

```
MATCH (a:User {name: 'Alice'}), (p:Product {name: 'Product2'})
CREATE (a)-[:VIEWED {date: date('2025-10-13'), duration_seconds: 45, device: 'desktop'}]->(p);
```

QUERY Questions (with Properties)

16. Find all products purchased by Alice, including quantity and date.

```
MATCH (a:User {name: 'Alice'})-[b:BOUGHT]->(p:Product)
RETURN p.name AS Product, b.quantity AS Quantity, b.date AS PurchaseDate;
```



```
neo4j$ MATCH (a:User {name: 'Alice'})-[b:BOUGHT]
```

	Product	Quantity	PurchaseDate
1	"Product1"	2	2025-10-14
2	"Product1"	2	2025-10-14
3	"Product1"	2	2025-10-14
4	"Product1"	2	2025-10-14
5	"Product1"	2	2025-10-14

```
Started streaming 5 records after 182 ms and completed after 187 ms.
```

17. Recommend products for Alice based on products Bob bought in the last month.

MATCH (a:User {name: 'Alice'}), (b:User {name: 'Bob'})-[:BOUGHT]->(p:Product)

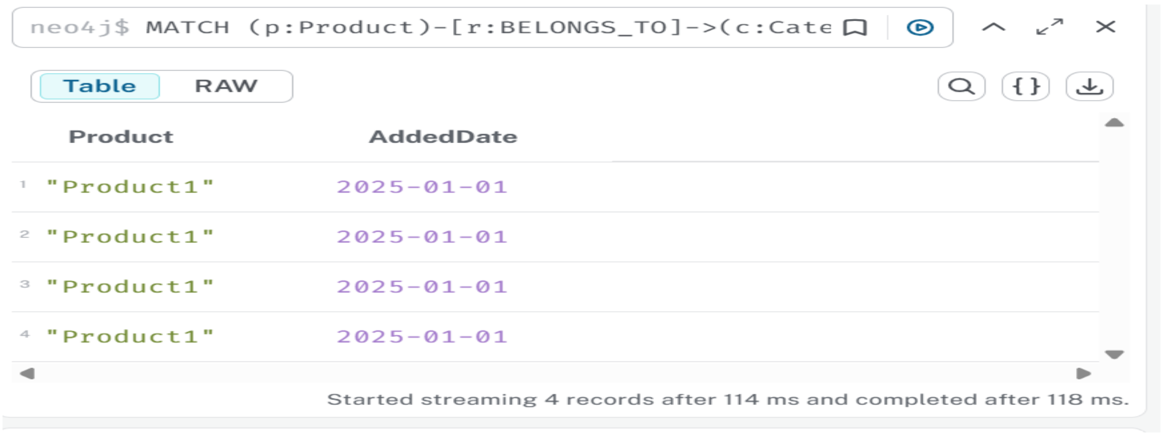
WHERE b<>a AND b.date >= date() - duration('P30D')

RETURN DISTINCT p.name AS RecommendedProducts;

18. List all products in the Electronics category and show when they were added.

MATCH (p:Product)-[r:BELONGS_TO]->(c:Category {name: 'Electronics'})

RETURN p.name AS Product, r.added_date AS AddedDate;



```
neo4j$ MATCH (p:Product)-[r:BELONGS_TO]->(c:Cate
```

	Product	AddedDate
1	"Product1"	2025-01-01
2	"Product1"	2025-01-01
3	"Product1"	2025-01-01
4	"Product1"	2025-01-01

```
Started streaming 4 records after 114 ms and completed after 118 ms.
```

19. Find top 5 products by average rating, including the latest review date.

MATCH (u:User)-[r:RATED]->(rev:Review)-[:REVIEWS]->(p:Product)

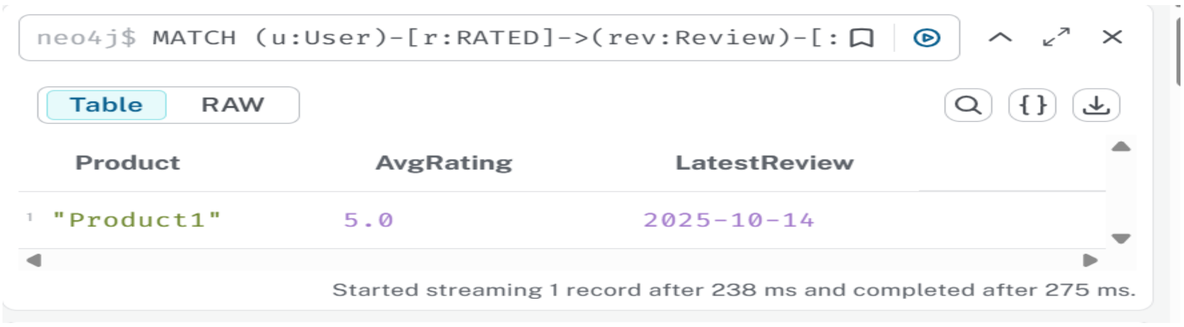
RETURN p.name AS Product,

AVG(r.rating) AS AvgRating,

MAX(r.review_date) AS LatestReview

ORDER BY AvgRating DESC

LIMIT 5;



```
neo4j$ MATCH (u:User)-[r:RATED]->(rev:Review)-[:
```

Table RAW

	Product	AvgRating	LatestReview
1	"Product1"	5.0	2025-10-14

Started streaming 1 record after 238 ms and completed after 275 ms.

20. Find all products made by BrandB and launch year.

MATCH (p:Product)-[m:MADE_BY]->(b:Brand {name: 'BrandB'})

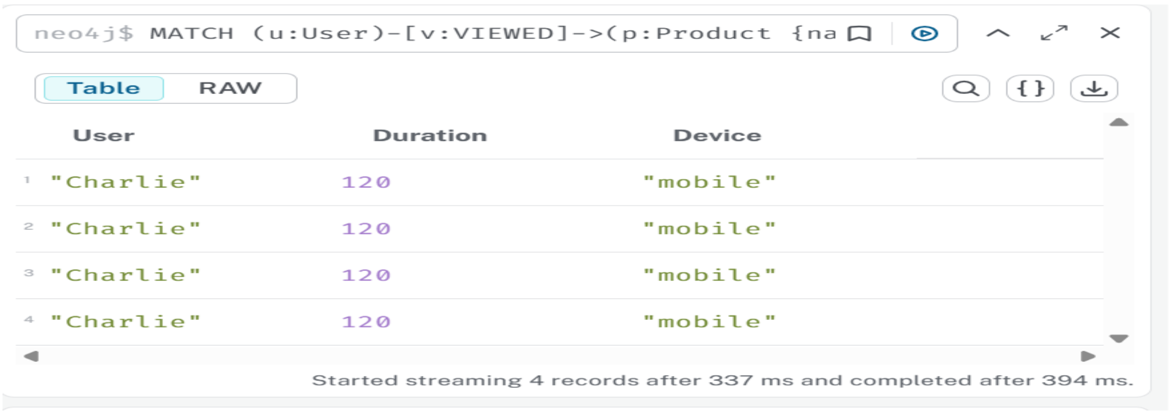
RETURN p.name AS Product, m.launch_year AS LaunchYear;

21. Find users who viewed Product3 but did not buy it; show viewing duration and device.

MATCH (u:User)-[v:VIEWED]->(p:Product {name: 'Product3'})

WHERE NOT (u)-[:BOUGHT]->(p)

RETURN u.name AS User, v.duration_seconds AS Duration, v.device AS Device;



```
neo4j$ MATCH (u:User)-[v:VIEWED]->(p:Product {na
```

Table RAW

	User	Duration	Device
1	"Charlie"	120	"mobile"
2	"Charlie"	120	"mobile"
3	"Charlie"	120	"mobile"
4	"Charlie"	120	"mobile"

Started streaming 4 records after 337 ms and completed after 394 ms.

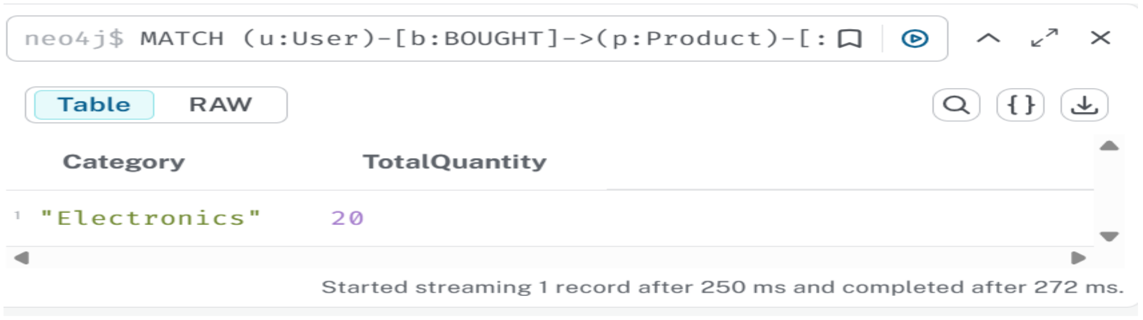
22. Find the category with the highest total purchased quantity.

MATCH (u:User)-[b:BOUGHT]->(p:Product)-[:BELONGS_TO]->(c:Category)

RETURN c.name AS Category, SUM(b.quantity) AS TotalQuantity

ORDER BY TotalQuantity DESC

LIMIT 1;



```
neo4j$ MATCH (u:User)-[b:BOUGHT]->(p:Product)-[:
```

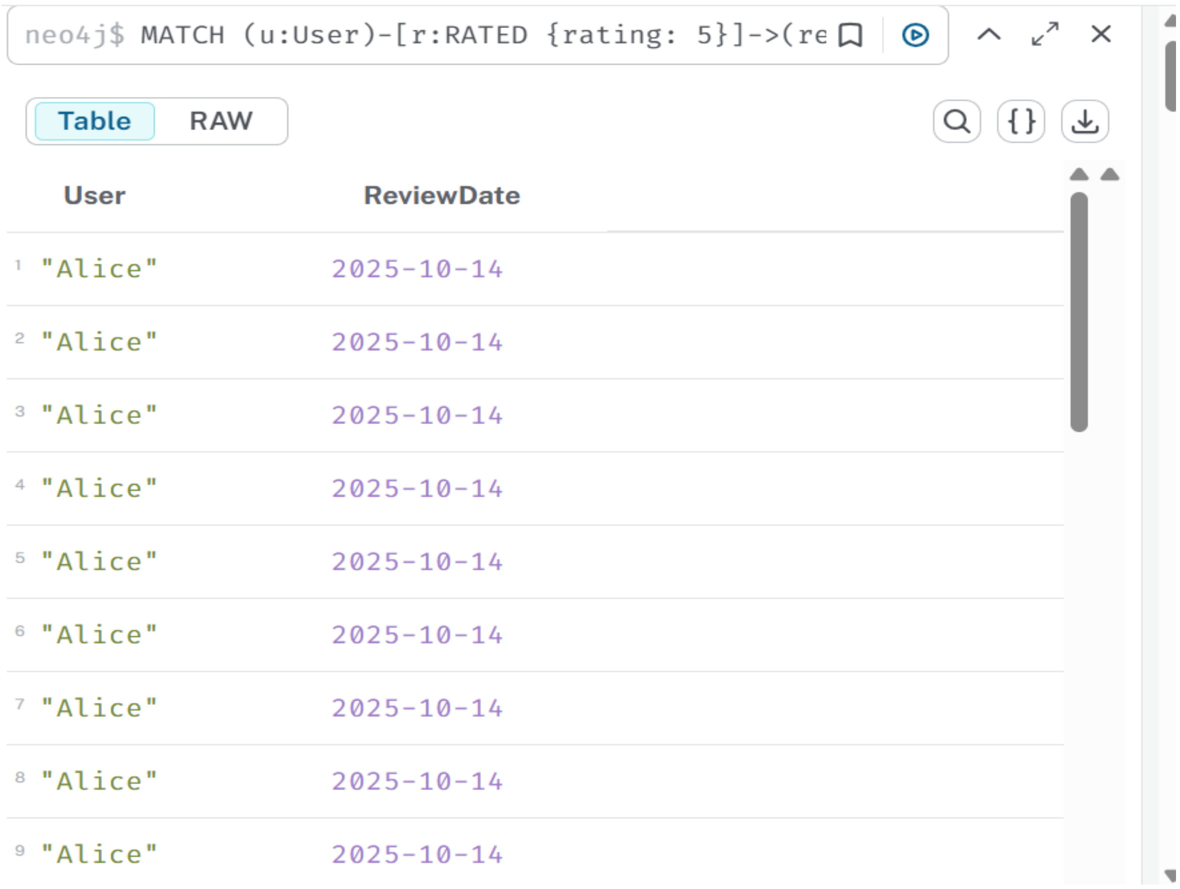
```
  Category          TotalQuantity
1 "Electronics"     20
```

```
Started streaming 1 record after 250 ms and completed after 272 ms.
```

23. Find all users who rated Product1 with 5 stars, including review date.

MATCH (u:User)-[r:RATED {rating: 5}]->(rev:Review)-[:REVIEWS]->(p:Product {name: 'Product1'})

RETURN u.name AS User, r.review_date AS ReviewDate;



```
neo4j$ MATCH (u:User)-[r:RATED {rating: 5}]->(re
```

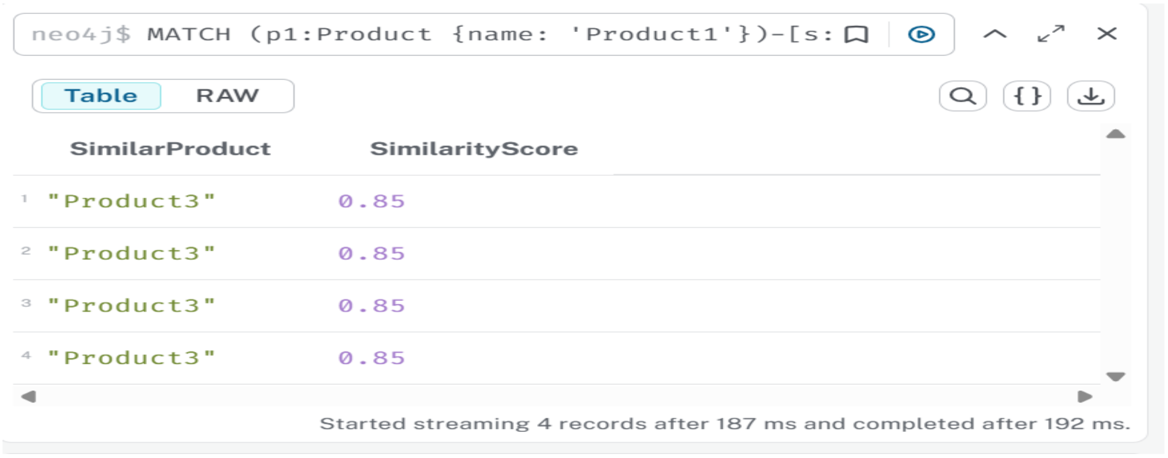
```
  User       ReviewDate
1 "Alice"    2025-10-14
2 "Alice"    2025-10-14
3 "Alice"    2025-10-14
4 "Alice"    2025-10-14
5 "Alice"    2025-10-14
6 "Alice"    2025-10-14
7 "Alice"    2025-10-14
8 "Alice"    2025-10-14
9 "Alice"    2025-10-14
```

24. Suggest products similar to Product1 with similarity_score > 0.8.

MATCH (p1:Product {name: 'Product1'})-[s:SIMILAR_TO]->(p2:Product)

WHERE s.similarity_score > 0.8

RETURN p2.name AS SimilarProduct, s.similarity_score AS SimilarityScore;



```
neo4j$ MATCH (p1:Product {name: 'Product1'})-[s:
```

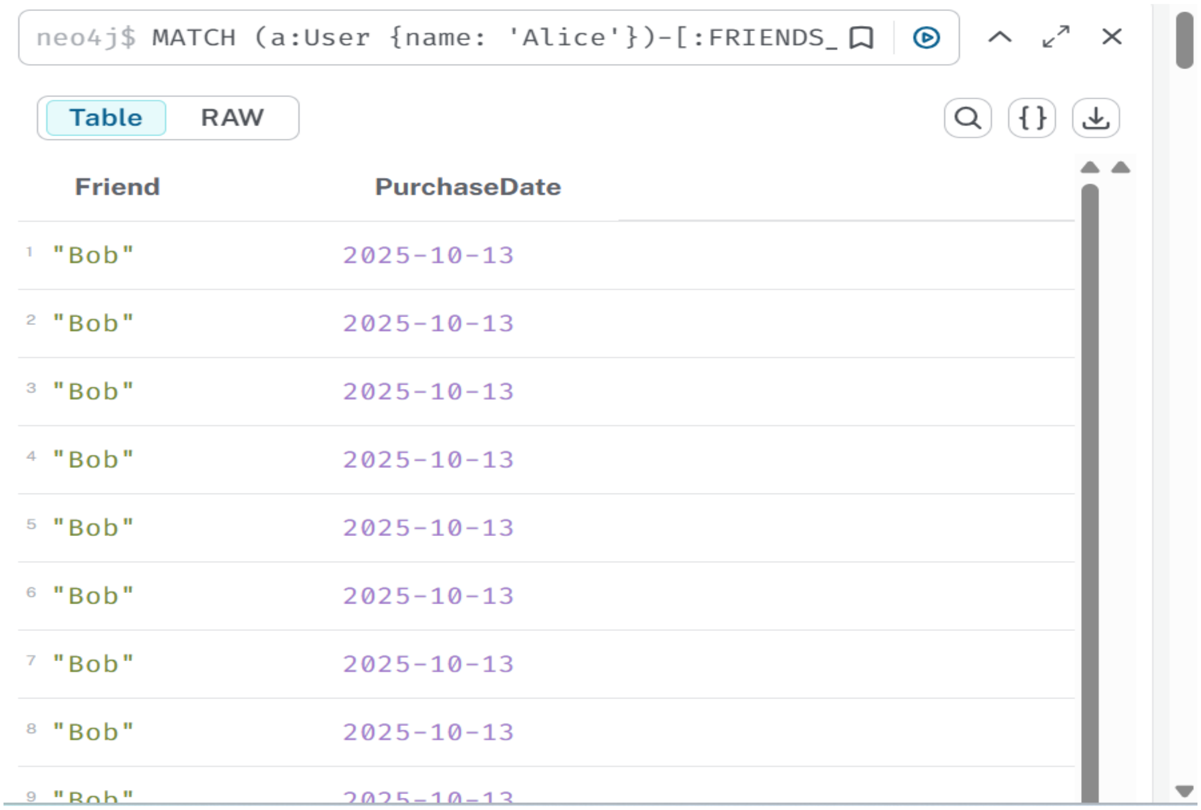
	SimilarProduct	SimilarityScore
1	"Product3"	0.85
2	"Product3"	0.85
3	"Product3"	0.85
4	"Product3"	0.85

Started streaming 4 records after 187 ms and completed after 192 ms.

25. Find friends of Alice who purchased Product2, showing purchase date.

MATCH (a:User {name: 'Alice'})-[:FRIENDS_WITH]->(f:User)-[b:BOUGHT]->(p:Product {name: 'Product2'})

RETURN f.name AS Friend, b.date AS PurchaseDate;



```
neo4j$ MATCH (a:User {name: 'Alice'})-[:FRIENDS_
```

	Friend	PurchaseDate
1	"Bob"	2025-10-13
2	"Bob"	2025-10-13
3	"Bob"	2025-10-13
4	"Bob"	2025-10-13
5	"Bob"	2025-10-13
6	"Bob"	2025-10-13
7	"Bob"	2025-10-13
8	"Bob"	2025-10-13
9	"Bob"	2025-10-13

26. List all reviews for Product2 along with user and rating.

MATCH (u:User)-[r:RATED]->(rev:Review)-[:REVIEWS]->(p:Product {name: 'Product2'})

RETURN u.name AS User, rev.name AS Review, r.rating AS Rating;

27. Find users who bought products from multiple categories, showing product names and categories.

MATCH (u:User)-[:BOUGHT]->(p:Product)-[:BELONGS_TO]->(c:Category)

WITH u, COLLECT(DISTINCT c.name) AS Categories, COLLECT(DISTINCT p.name) AS Products

WHERE SIZE(Categories) > 1

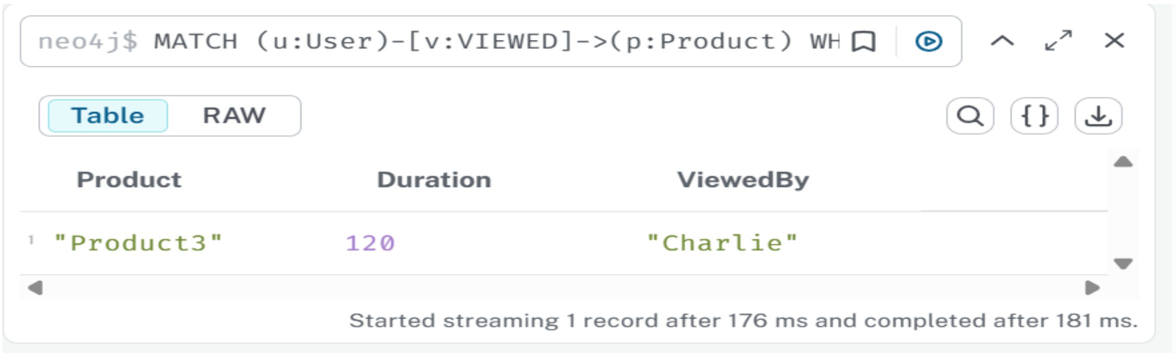
RETURN u.name AS User, Products, Categories;

28. Find products viewed more than 100 seconds by any user.

MATCH (u:User)-[v:VIEWED]->(p:Product)

WHERE v.duration_seconds > 100

RETURN DISTINCT p.name AS Product, v.duration_seconds AS Duration, u.name AS ViewedBy;



```
neo4j$ MATCH (u:User)-[v:VIEWED]->(p:Product) WH
```

	Product	Duration	ViewedBy
1	"Product3"	120	"Charlie"

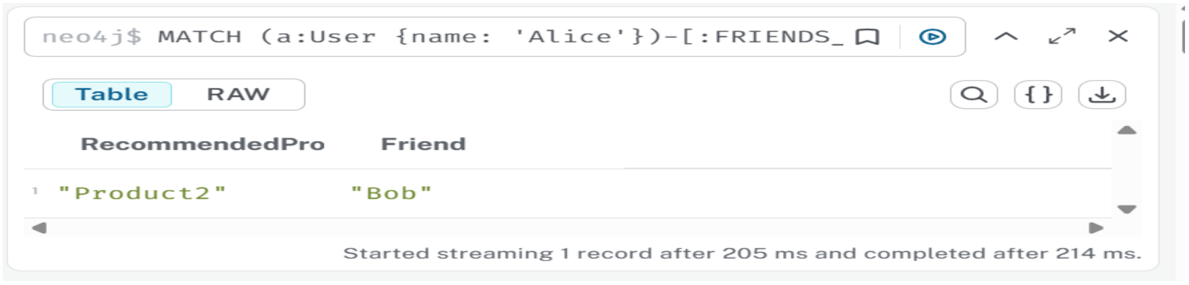
Started streaming 1 record after 176 ms and completed after 181 ms.

29. Recommend products based on friends' purchases in the last 30 days.

MATCH (a:User {name: 'Alice'})-[:FRIENDS_WITH]->(f:User)-[b:BOUGHT]->(p:Product)

WHERE b.date >= date() - duration('P30D')

RETURN DISTINCT p.name AS RecommendedProduct, f.name AS Friend;



```
neo4j$ MATCH (a:User {name: 'Alice'})-[:FRIENDS_
```

	RecommendedPro	Friend
1	"Product2"	"Bob"

Started streaming 1 record after 205 ms and completed after 214 ms.

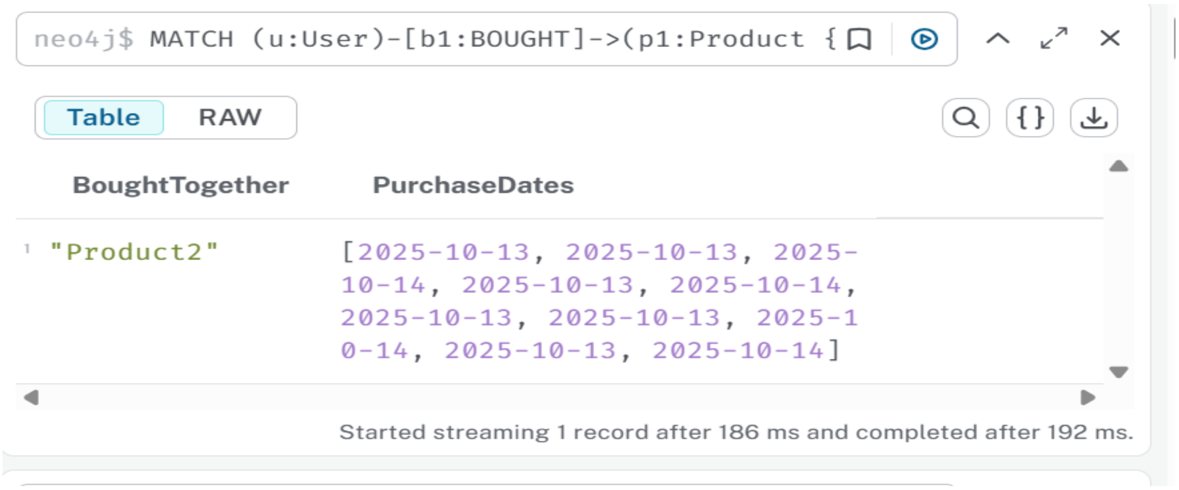
30. Find products frequently bought together with Product2, including purchase dates.

MATCH (u:User)-[b1:BOUGHT]->(p1:Product {name: 'Product2'}),

(u)-[b2:BOUGHT]->(p2:Product)

WHERE p1 <> p2

RETURN p2.name AS BoughtTogether, COLLECT(b2.date) AS PurchaseDates;



```
neo4j$ MATCH (u:User)-[b1:BOUGHT]->(p1:Product {

Table  RAW

BoughtTogether    PurchaseDates
1 "Product2"      [2025-10-13, 2025-10-13, 2025-
                  10-14, 2025-10-13, 2025-10-14,
                  2025-10-13, 2025-10-13, 2025-1
                  0-14, 2025-10-13, 2025-10-14]

Started streaming 1 record after 186 ms and completed after 192 ms.
```

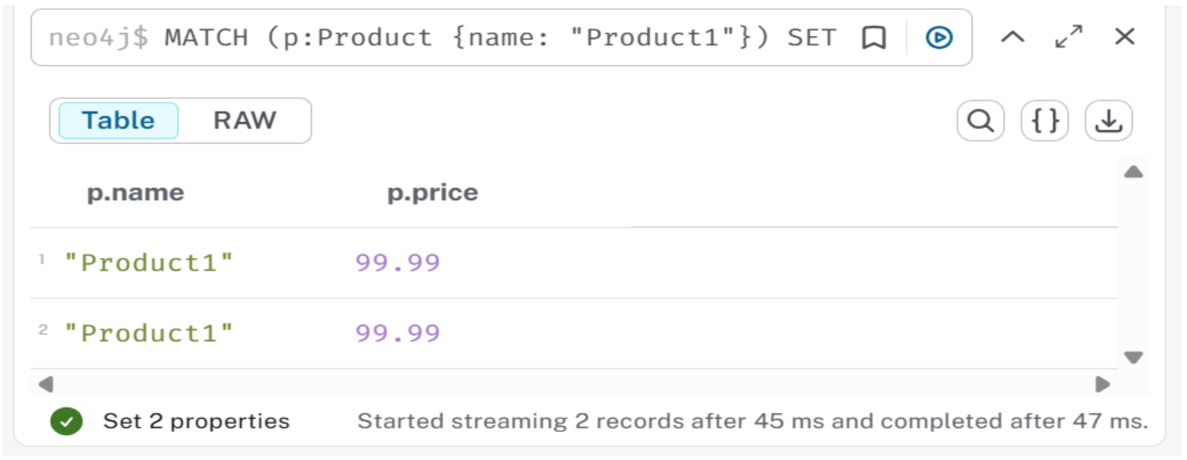
UPDATE Questions (with Properties)

31. Update the price of Product1 to 99.99

MATCH (p:Product {name: "Product1"})

SET p.price = 99.99

RETURN p.name, p.price;



```
neo4j$ MATCH (p:Product {name: "Product1"}) SET

Table  RAW

p.name        p.price
1 "Product1"  99.99
2 "Product1"  99.99

Set 2 properties    Started streaming 2 records after 45 ms and completed after 47 ms.
```

32. Add Product1 to a new category “Gadgets” with added_date 2025-10-14

MERGE (c:Category {name: "Gadgets"})

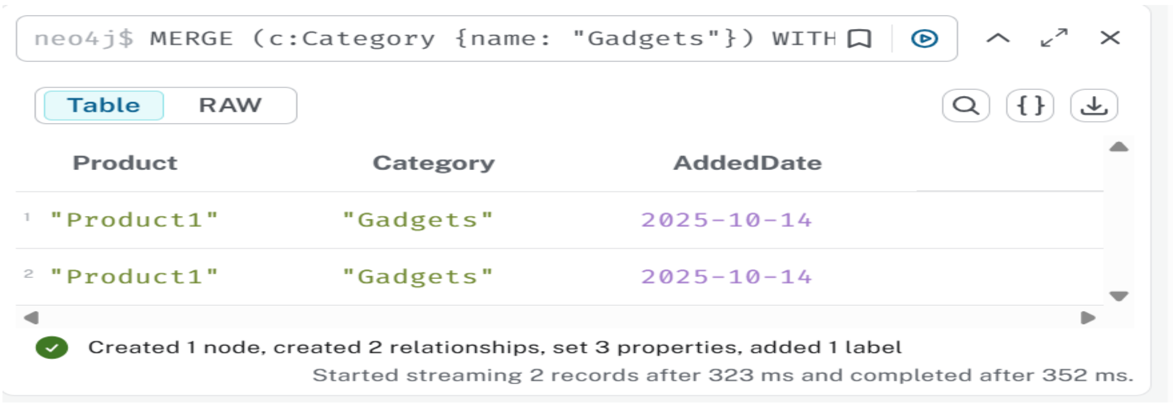
WITH c

MATCH (p:Product {name: "Product1"})

MERGE (p)-[r:BELONGS_TO]->(c)

SET r.added_date = date("2025-10-14")

RETURN p.name AS Product, c.name AS Category, r.added_date AS AddedDate;



```
neo4j$ MERGE (c:Category {name: "Gadgets"}) WITH
```

	Product	Category	AddedDate
1	"Product1"	"Gadgets"	2025-10-14
2	"Product1"	"Gadgets"	2025-10-14

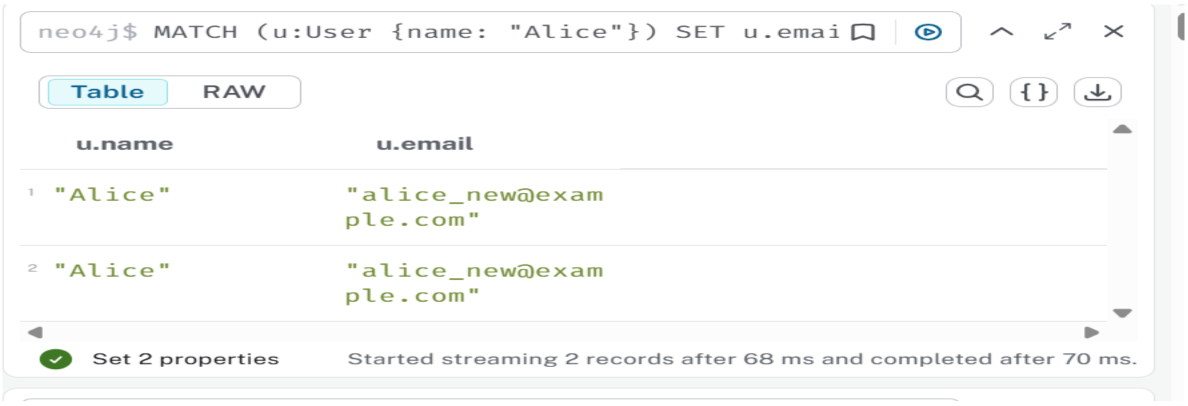
```
Created 1 node, created 2 relationships, set 3 properties, added 1 label
Started streaming 2 records after 323 ms and completed after 352 ms.
```

33. Update Alice's email

MATCH (u:User {name: "Alice"})

SET u.email = "alice_new@example.com"

RETURN u.name, u.email;



```
neo4j$ MATCH (u:User {name: "Alice"}) SET u.emai
```

	u.name	u.email
1	"Alice"	"alice_new@example.com"
2	"Alice"	"alice_new@example.com"

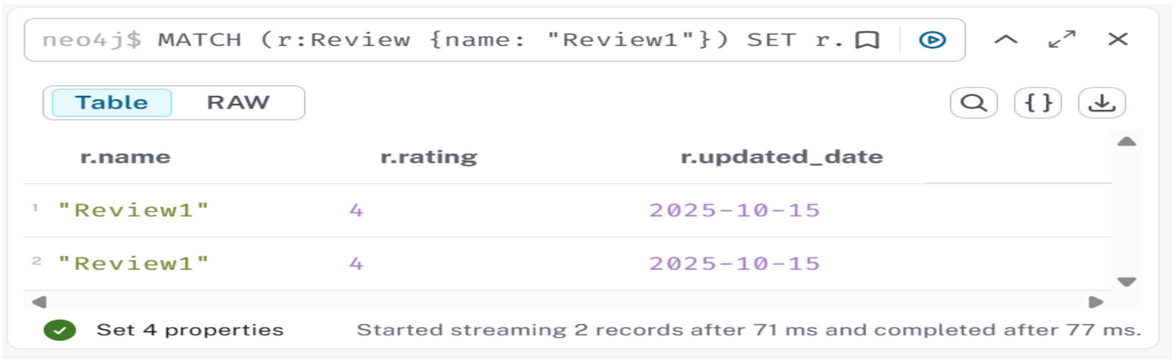
```
Set 2 properties    Started streaming 2 records after 68 ms and completed after 70 ms.
```

34. Update Review1 rating and date

MATCH (r:Review {name: "Review1"})

SET r.rating = 4, r.updated_date = date("2025-10-15")

RETURN r.name, r.rating, r.updated_date;



```
neo4j$ MATCH (r:Review {name: "Review1"}) SET r.
```

Table RAW

```
  r.name      r.rating   r.updated_date
1 "Review1"   4          2025-10-15
2 "Review1"   4          2025-10-15
```

Set 4 properties   Started streaming 2 records after 71 ms and completed after 77 ms.

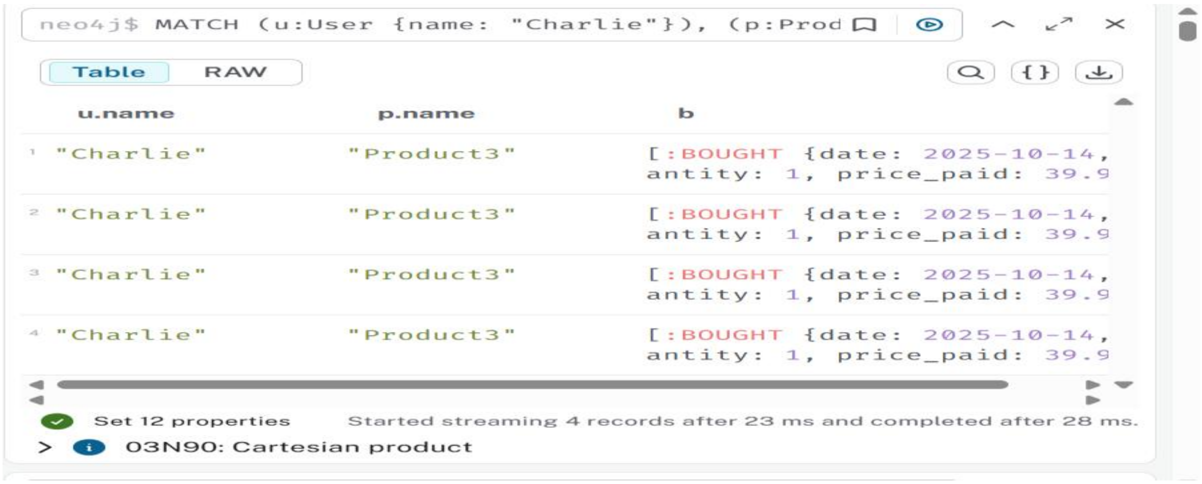
35. Add a BOUGHT relationship: Charlie bought Product3

MATCH (u:User {name: "Charlie"}), (p:Product {name: "Product3"})

MERGE (u)-[b:BOUGHT]->(p)

SET b.date = date("2025-10-14"), b.quantity = 1, b.price_paid = 39.99

RETURN u.name, p.name, b;



```
neo4j$ MATCH (u:User {name: "Charlie"}), (p:Prod
```

Table RAW

```
  u.name      p.name       b
1 "Charlie"   "Product3"   [:BOUGHT {date: 2025-10-14,
                           antity: 1, price_paid: 39.9
2 "Charlie"   "Product3"   [:BOUGHT {date: 2025-10-14,
                           antity: 1, price_paid: 39.9
3 "Charlie"   "Product3"   [:BOUGHT {date: 2025-10-14,
                           antity: 1, price_paid: 39.9
4 "Charlie"   "Product3"   [:BOUGHT {date: 2025-10-14,
                           antity: 1, price_paid: 39.9
```

Set 12 properties   Started streaming 4 records after 23 ms and completed after 28 ms.

> 03N90: Cartesian product

36. Change brand of Product3 from BrandC to BrandD

MATCH (p:Product {name: "Product3"})-[old:MADE_BY]->(oldBrand:Brand {name: "BrandC"})

DELETE old

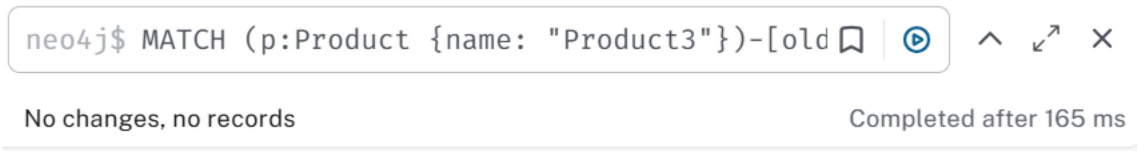
WITH p

MERGE (newBrand:Brand {name: "BrandD"})

MERGE (p)-[r:MADE_BY]->(newBrand)

SET r.launch_year = 2025

RETURN p.name, newBrand.name, r.launch_year;



```
neo4j$ MATCH (p:Product {name: "Product3"})-[old
```

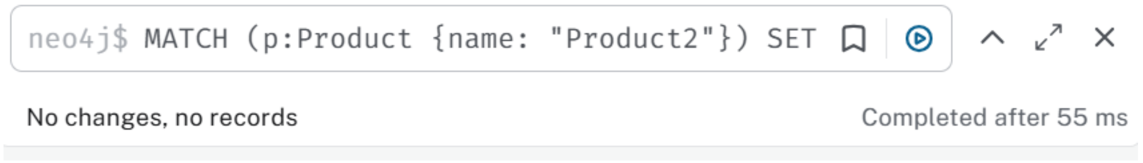
No changes, no records   Completed after 165 ms

37. Rename Product2 to “BookMaster 2025”

MATCH (p:Product {name: "Product2"})

SET p.name = "BookMaster 2025"

RETURN p.name;



```
neo4j$ MATCH (p:Product {name: "Product2"}) SET
```

No changes, no records   Completed after 55 ms

38. Update user names: Bob → Robert, Charlie → Charles

MATCH (u:User)

WHERE u.name IN ["Bob", "Charlie"]

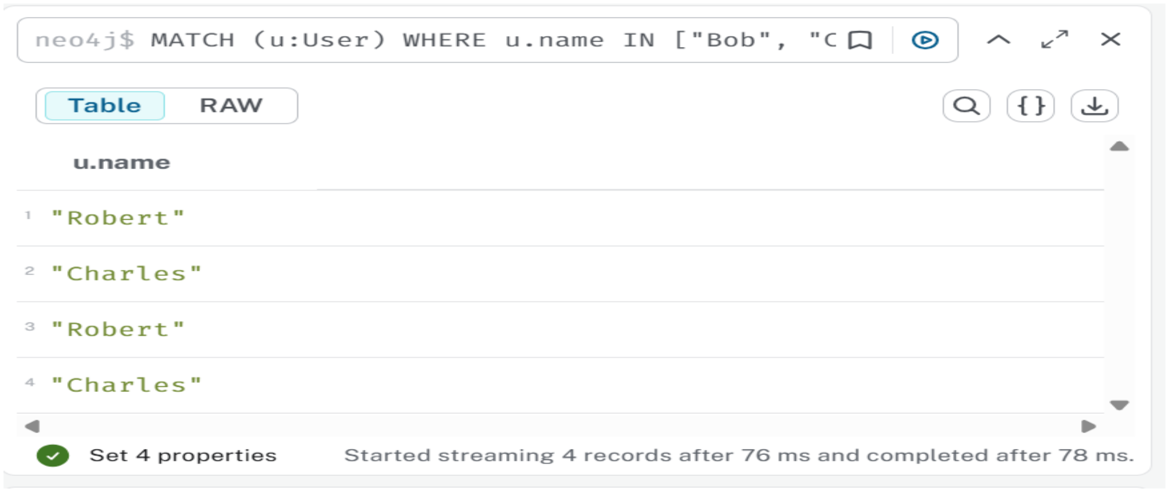
SET u.name = CASE u.name

  WHEN "Bob" THEN "Robert"

  WHEN "Charlie" THEN "Charles"

END

RETURN u.name;



```
neo4j$ MATCH (u:User) WHERE u.name IN ["Bob", "C
```

```
u.name
1 "Robert"
2 "Charles"
3 "Robert"
4 "Charles"
```

Set 4 properties   Started streaming 4 records after 76 ms and completed after 78 ms.

39. Change category of Product3 from Clothing to Sports

MATCH (p:Product {name: "Product3"})-[rel:BELONGS_TO]->(:Category {name: "Clothing"})

DELETE rel

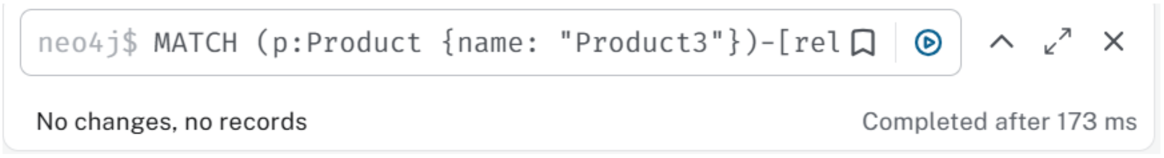
WITH p

MERGE (c:Category {name: "Sports"})

MERGE (p)-[r:BELONGS_TO]->(c)

SET r.added_date = date("2025-10-14")

RETURN p.name, c.name, r.added_date;



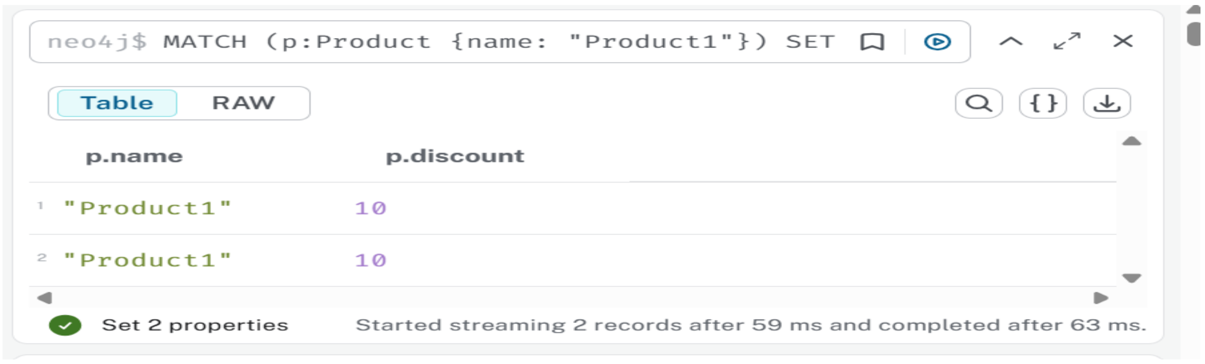
```
neo4j$ MATCH (p:Product {name: "Product3"})-[rel
No changes, no records                                  Completed after 173 ms
```

40. Add new attribute discount = 10% to Product1

MATCH (p:Product {name: "Product1"})

SET p.discount = 10

RETURN p.name, p.discount;



```
neo4j$ MATCH (p:Product {name: "Product1"}) SET
Table   RAW

   p.name          p.discount
1  "Product1"      10
2  "Product1"      10

Set 2 properties    Started streaming 2 records after 59 ms and completed after 63 ms.
```

DELETE Questions (with Properties)

41. Delete Product50

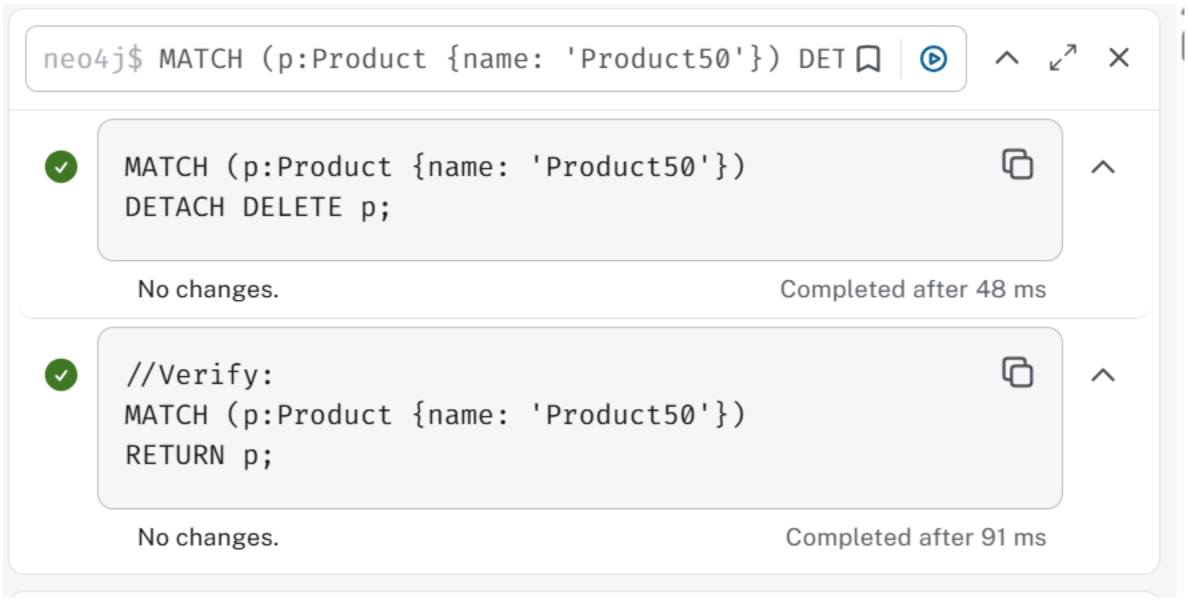
MATCH (p:Product {name: 'Product50'})

DETACH DELETE p;

//Verify:

MATCH (p:Product {name: 'Product50'})

RETURN p;



```
neo4j$ MATCH (p:Product {name: 'Product50'}) DET

MATCH (p:Product {name: 'Product50'})
DETACH DELETE p;
No changes.                                    Completed after 48 ms

//Verify:
MATCH (p:Product {name: 'Product50'})
RETURN p;
No changes.                                    Completed after 91 ms
```

42. Delete Review5 with rating 2

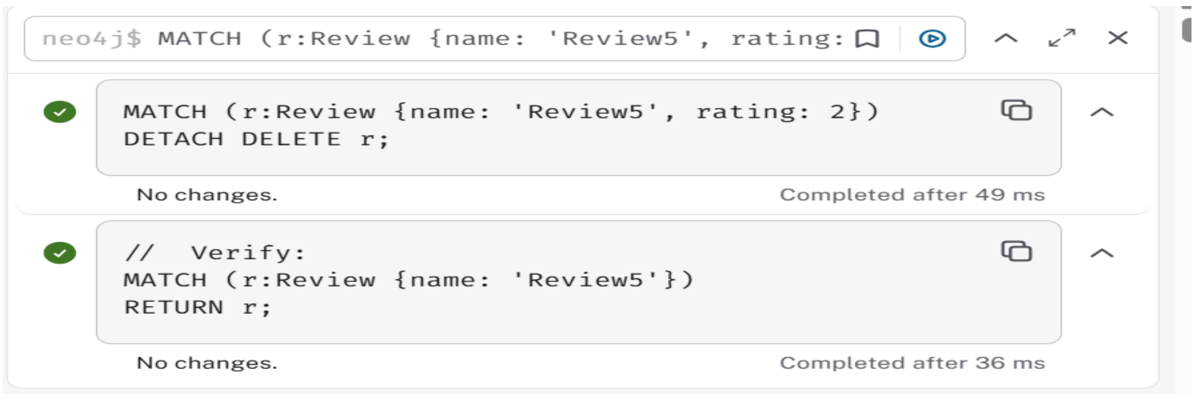
MATCH (r:Review {name: 'Review5', rating: 2})

DETACH DELETE r;

// Verify:

MATCH (r:Review {name: 'Review5'})

RETURN r;



```
neo4j$ MATCH (r:Review {name: 'Review5', rating:

MATCH (r:Review {name: 'Review5', rating: 2})
DETACH DELETE r;
No changes.                                    Completed after 49 ms

//  Verify:
MATCH (r:Review {name: 'Review5'})
RETURN r;
No changes.                                    Completed after 36 ms
```

43. Remove BOUGHT relationship between Alice and Product2

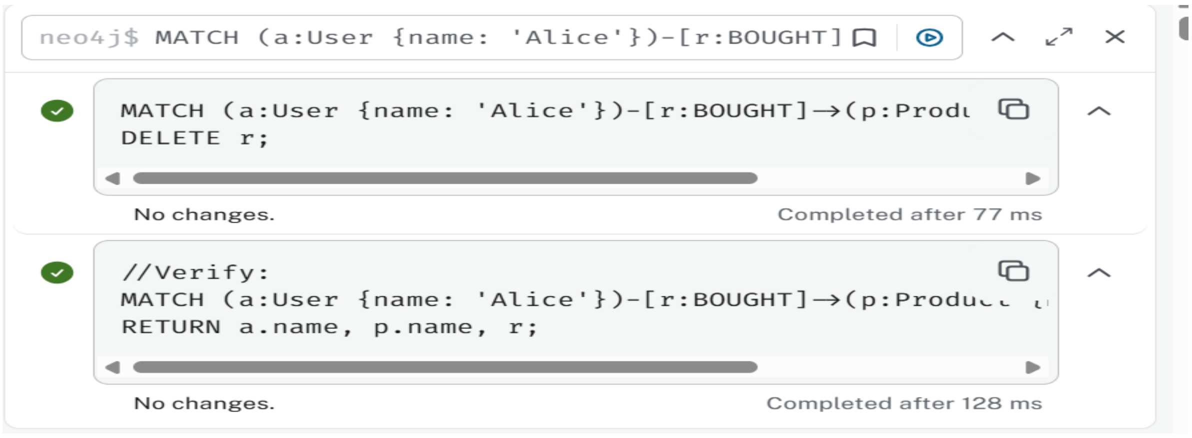
MATCH (a:User {name: 'Alice'})-[r:BOUGHT]->(p:Product {name: 'Product2'})

DELETE r;

//Verify:

MATCH (a:User {name: 'Alice'})-[r:BOUGHT]->(p:Product {name: 'Product2'})

RETURN a.name, p.name, r;



```
neo4j$ MATCH (a:User {name: 'Alice'})-[r:BOUGHT]
```

```
MATCH (a:User {name: 'Alice'})-[r:BOUGHT]→(p:Produ
DELETE r;
```

No changes. Completed after 77 ms

```
//Verify:
MATCH (a:User {name: 'Alice'})-[r:BOUGHT]→(p:Produc
RETURN a.name, p.name, r;
```

No changes. Completed after 128 ms

44. Delete category "OldCategory"

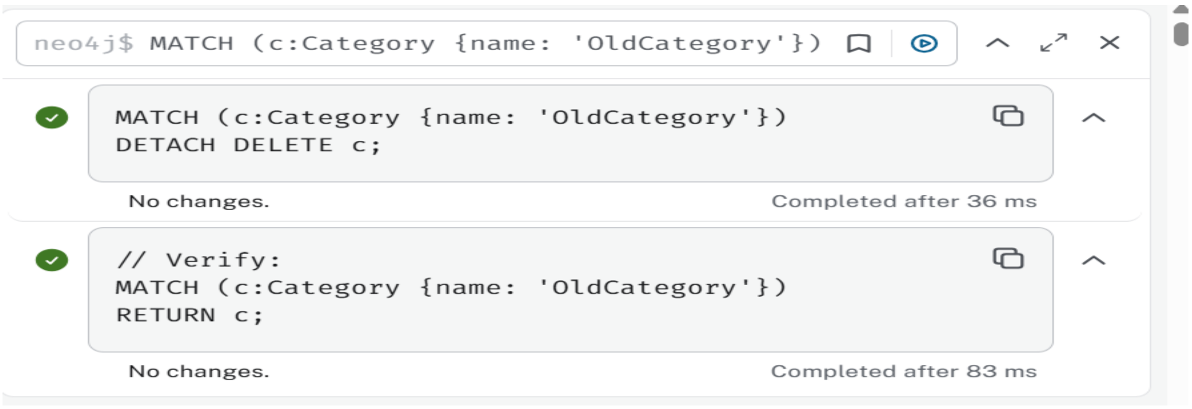
MATCH (c:Category {name: 'OldCategory'})

DETACH DELETE c;

// Verify:

MATCH (c:Category {name: 'OldCategory'})

RETURN c;



```
neo4j$ MATCH (c:Category {name: 'OldCategory'})
```

```
MATCH (c:Category {name: 'OldCategory'})
DETACH DELETE c;
```

No changes. Completed after 36 ms

```
// Verify:
MATCH (c:Category {name: 'OldCategory'})
RETURN c;
```

No changes. Completed after 83 ms

45. Delete user Tina

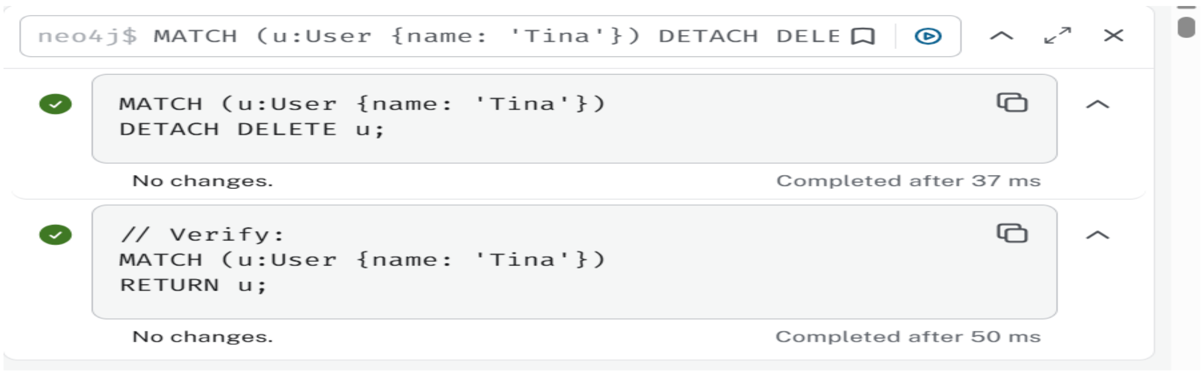
MATCH (u:User {name: 'Tina'})

DETACH DELETE u;

// Verify:

MATCH (u:User {name: 'Tina'})

RETURN u;



```
neo4j$ MATCH (u:User {name: 'Tina'}) DETACH DELE
```

```
MATCH (u:User {name: 'Tina'})
DETACH DELETE u;
```

No changes. Completed after 37 ms

```
// Verify:
MATCH (u:User {name: 'Tina'})
RETURN u;
```

No changes. Completed after 50 ms

46. Remove SIMILAR_TO relationship between Product1 and Product3

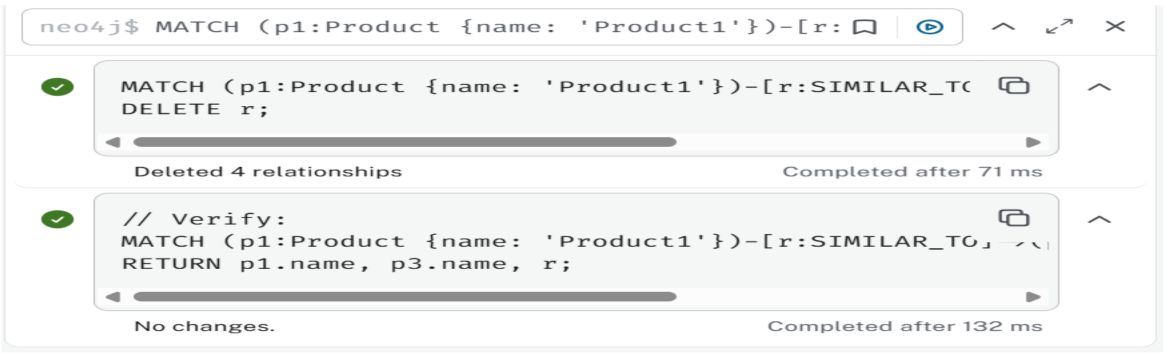
MATCH (p1:Product {name: 'Product1'})-[r:SIMILAR_TO]->(p3:Product {name: 'Product3'})

DELETE r;

// Verify:

MATCH (p1:Product {name: 'Product1'})-[r:SIMILAR_TO]->(p3:Product {name: 'Product3'})

RETURN p1.name, p3.name, r;



```
neo4j$ MATCH (p1:Product {name: 'Product1'})-[r:
```

```
MATCH (p1:Product {name: 'Product1'})-[r:SIMILAR_TO
DELETE r;
```

Deleted 4 relationships Completed after 71 ms

```
// Verify:
MATCH (p1:Product {name: 'Product1'})-[r:SIMILAR_TO]
RETURN p1.name, p3.name, r;
```

No changes. Completed after 132 ms

47. Delete all products made by BrandJ

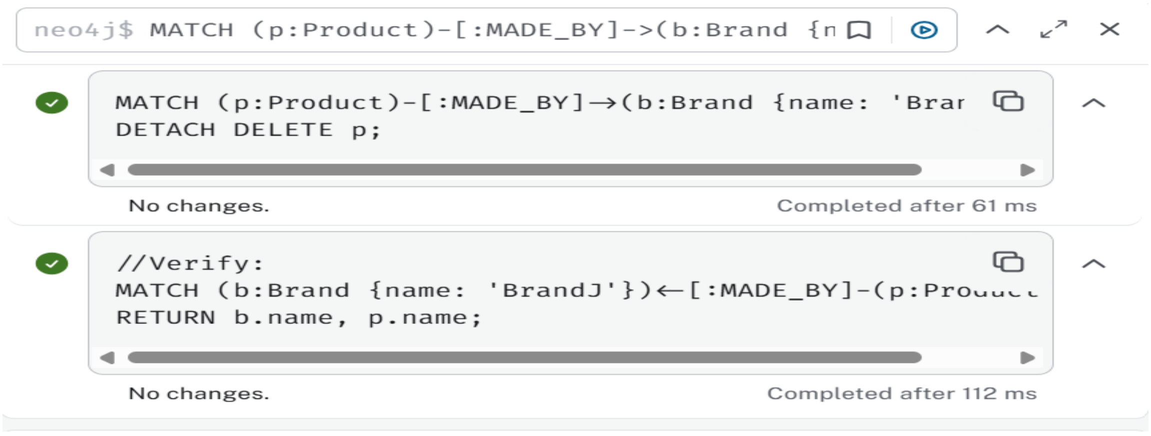
MATCH (p:Product)-[:MADE_BY]->(b:Brand {name: 'BrandJ'})

DETACH DELETE p;

//Verify:

MATCH (b:Brand {name: 'BrandJ'})<-[:MADE_BY]-(p:Product)

RETURN b.name, p.name;



```
neo4j$ MATCH (p:Product)-[:MADE_BY]->(b:Brand {n
```

```
MATCH (p:Product)-[:MADE_BY]→(b:Brand {name: 'Brar
DETACH DELETE p;
```

```
No changes.                                    Completed after 61 ms
```

```
//Verify:
MATCH (b:Brand {name: 'BrandJ'})←[:MADE_BY]-(p:Prouuc
RETURN b.name, p.name;
```

```
No changes.                                    Completed after 112 ms
```

48. Delete all VIEWED relationships for Product3 before 2025-01-01

MATCH (u:User)-[v:VIEWED]->(p:Product {name: 'Product3'})

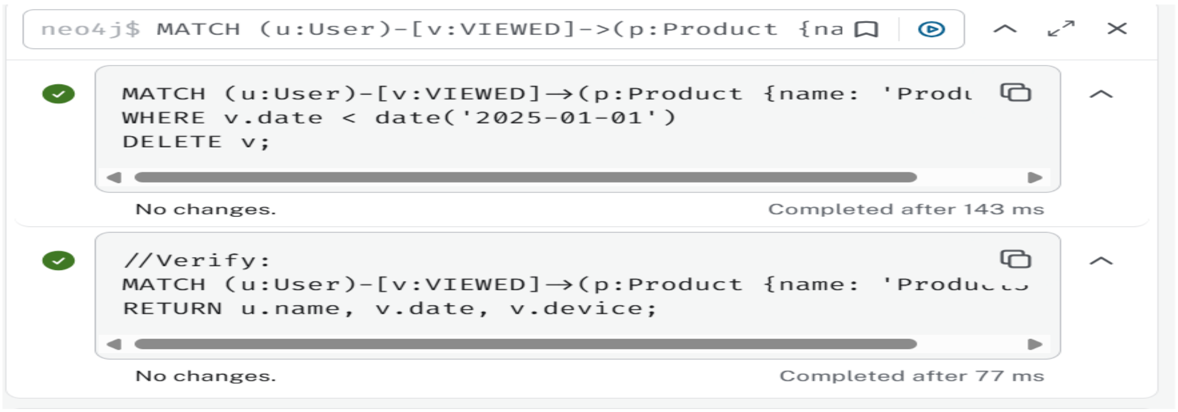
WHERE v.date < date('2025-01-01')

DELETE v;

//Verify:

MATCH (u:User)-[v:VIEWED]->(p:Product {name: 'Product3'})

RETURN u.name, v.date, v.device;



```
neo4j$ MATCH (u:User)-[v:VIEWED]->(p:Product {na
```

```
MATCH (u:User)-[v:VIEWED]→(p:Product {name: 'Prodı
WHERE v.date < date('2025-01-01')
DELETE v;
```

```
No changes.                                    Completed after 143 ms
```

```
//Verify:
MATCH (u:User)-[v:VIEWED]→(p:Product {name: 'Produc
RETURN u.name, v.date, v.device;
```

```
No changes.                                    Completed after 77 ms
```

49. Delete products never bought or reviewed (Product48, Product49)

MATCH (p:Product)

WHERE p.name IN ['Product48', 'Product49']

  AND NOT (()-[:BOUGHT]->(p) OR ()-[:RATED]->()-[:REVIEWS]->(p))

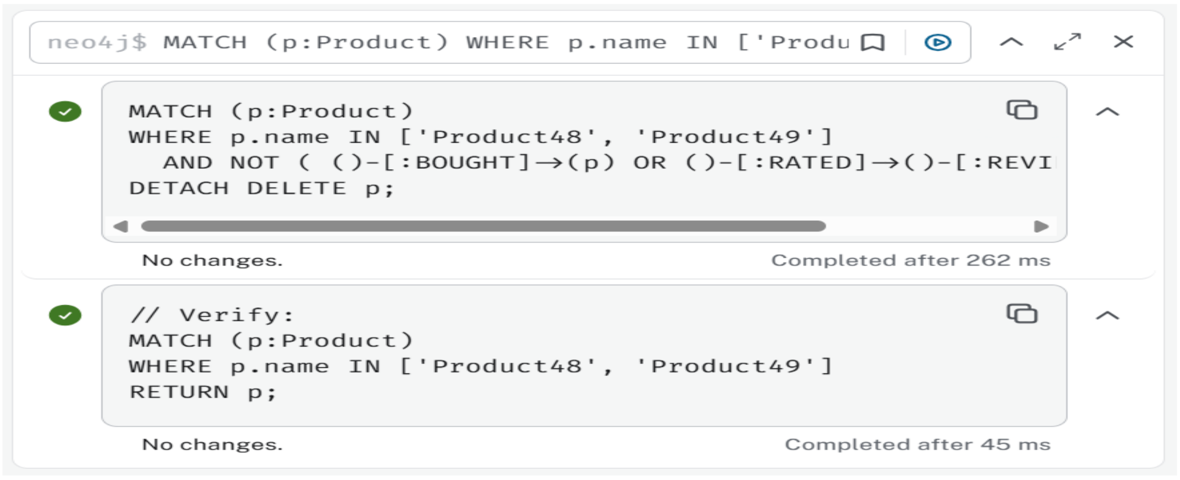
DETACH DELETE p;

// Verify:

MATCH (p:Product)

WHERE p.name IN ['Product48', 'Product49']

RETURN p;



```
neo4j$ MATCH (p:Product) WHERE p.name IN ['Produ
```

```
MATCH (p:Product)
WHERE p.name IN ['Product48', 'Product49']
  AND NOT ( ()-[:BOUGHT]->(p) OR ()-[:RATED]->()-[:REVI
DETACH DELETE p;
```

No changes. Completed after 262 ms

```
// Verify:
MATCH (p:Product)
WHERE p.name IN ['Product48', 'Product49']
RETURN p;
```

No changes. Completed after 45 ms

50. Delete Review3 without deleting Product3

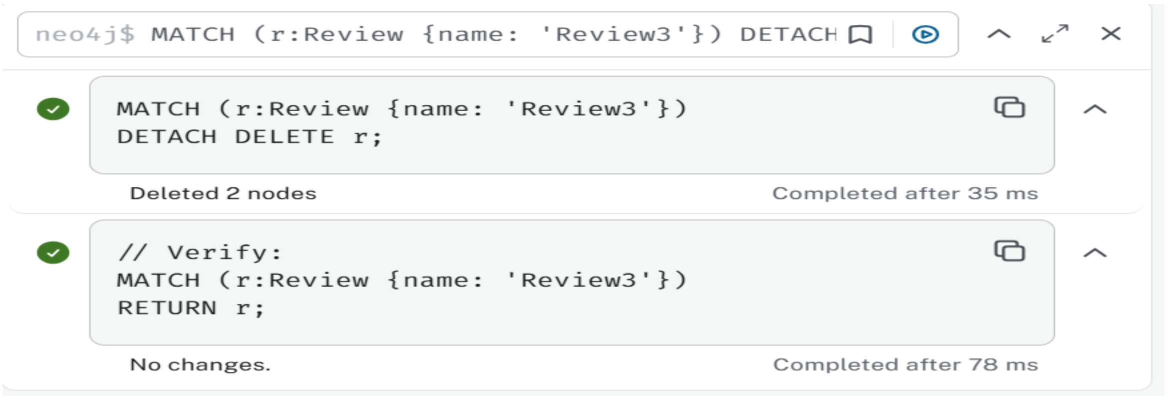
MATCH (r:Review {name: 'Review3'})

DETACH DELETE r;

// Verify:

MATCH (r:Review {name: 'Review3'})

RETURN r;



```
neo4j$ MATCH (r:Review {name: 'Review3'}) DETACH
```

```
MATCH (r:Review {name: 'Review3'})
DETACH DELETE r;
```

Deleted 2 nodes Completed after 35 ms

```
// Verify:
MATCH (r:Review {name: 'Review3'})
RETURN r;
```

No changes. Completed after 78 ms

ANALYTICAL / COMPLEX Query Questions (with Properties)

51. Find top 5 users by purchase quantity in October 2025

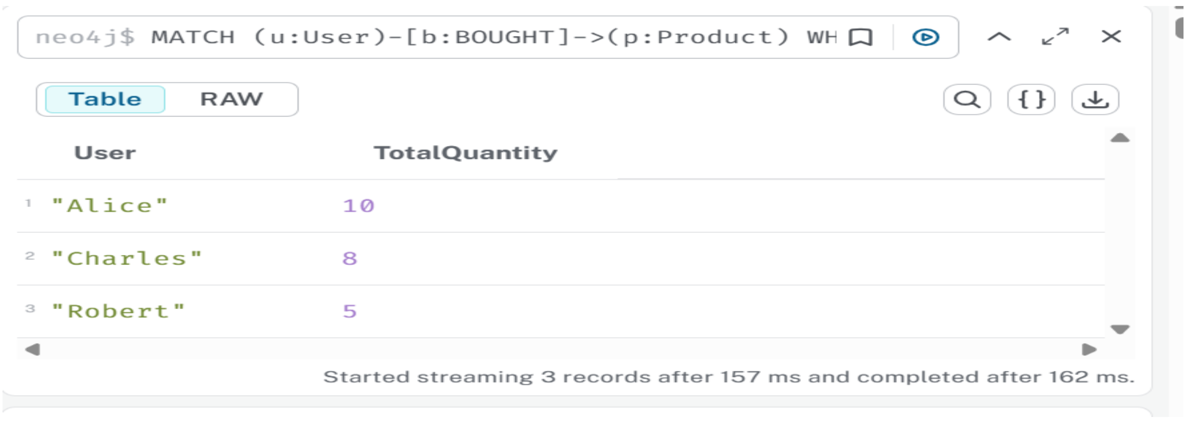
MATCH (u:User)-[b:BOUGHT]->(p:Product)

WHERE b.date >= date('2025-10-01') AND b.date <= date('2025-10-31')

RETURN u.name AS User, SUM(b.quantity) AS TotalQuantity

ORDER BY TotalQuantity DESC

LIMIT 5;



```
neo4j$ MATCH (u:User)-[b:BOUGHT]->(p:Product) WH
```

Table RAW

```
   User          TotalQuantity
1  "Alice"       10
2  "Charles"     8
3  "Robert"      5
```

Started streaming 3 records after 157 ms and completed after 162 ms.

52. Recommend products for Alice based on purchases in Electronics category

MATCH (a:User {name: 'Alice'})-[:BOUGHT]->(:Product)-[:BELONGS_TO]->(c:Category {name: 'Electronics'})

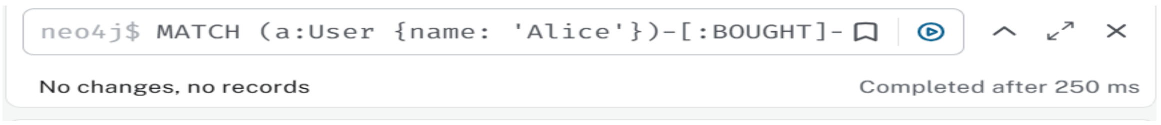
WITH a, c

MATCH (other:User)-[:BOUGHT]->(p:Product)-[:BELONGS_TO]->(c)

WHERE other <> a AND NOT (a)-[:BOUGHT]->(p)

RETURN DISTINCT p.name AS RecommendedProduct, c.name AS Category

LIMIT 10;



```
neo4j$ MATCH (a:User {name: 'Alice'})-[:BOUGHT]-
```

No changes, no records Completed after 250 ms

53. Identify products frequently bought together with Product2 in the last 30 days

MATCH (u:User)-[b1:BOUGHT]->(p1:Product {name: 'Product2'}),

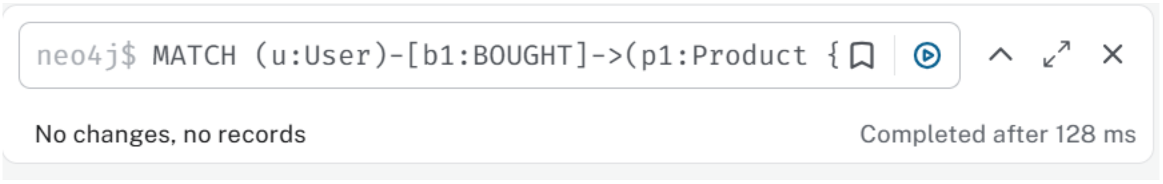
    (u)-[b2:BOUGHT]->(p2:Product)

WHERE p1 <> p2 AND b2.date >= date() - duration({days: 30})

RETURN p2.name AS CoBoughtProduct, COUNT(*) AS Frequency

ORDER BY Frequency DESC

LIMIT 5;



```
neo4j$ MATCH (u:User)-[b1:BOUGHT]->(p1:Product {
```

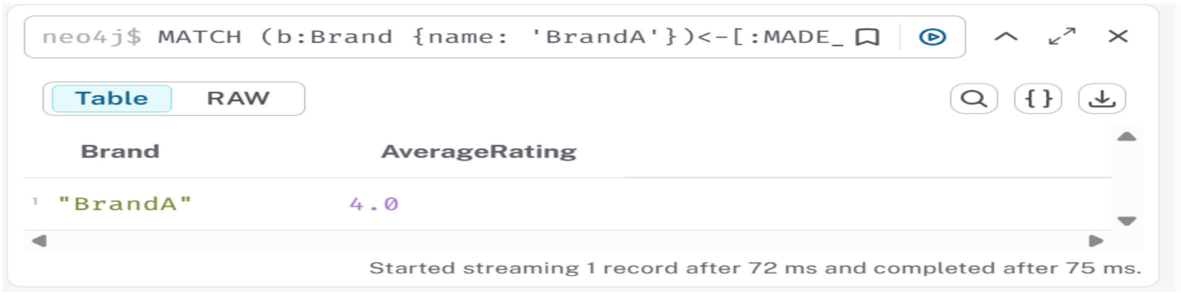
No changes, no records Completed after 128 ms

54. Find average rating for products made by BrandA

MATCH (b:Brand {name: 'BrandA'})<-[:MADE_BY]-(p:Product)<-[:REVIEWS]-(r:Review)

RETURN b.name AS Brand, AVG(r.rating) AS AverageRating

ORDER BY AverageRating DESC;


```
neo4j$ MATCH (b:Brand {name: 'BrandA'})<-[:MADE_
```
	Brand	AverageRating
1	"BrandA"	4.0

```
Started streaming 1 record after 72 ms and completed after 75 ms.
```

55. Suggest products for Alice based on her friends' purchases, including purchase dates

MATCH (a:User {name: 'Alice'})-[:FRIEND_WITH]->(f:User)-[b:BOUGHT]->(p:Product)

WHERE NOT (a)-[:BOUGHT]->(p)

RETURN f.name AS Friend, p.name AS SuggestedProduct, b.date AS PurchaseDate

ORDER BY b.date DESC

LIMIT 10;

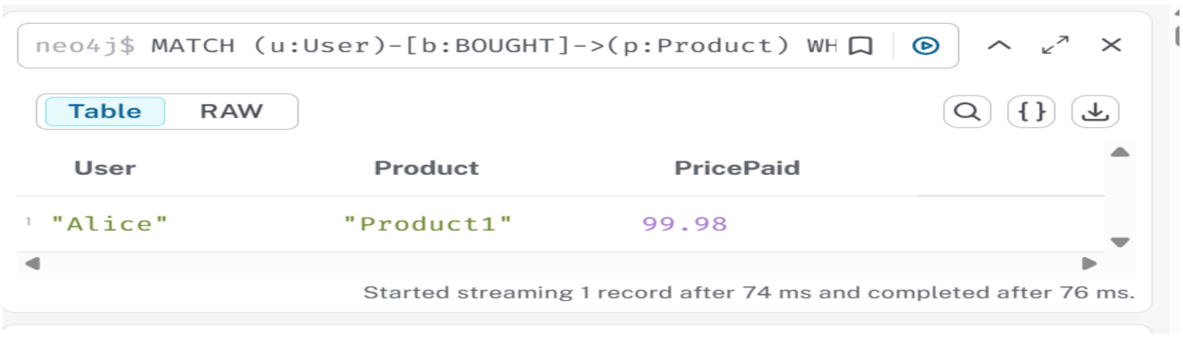
56. Find users who bought products with price_paid > 80

MATCH (u:User)-[b:BOUGHT]->(p:Product)

WHERE b.price_paid > 80

RETURN DISTINCT u.name AS User, p.name AS Product, b.price_paid AS PricePaid

ORDER BY b.price_paid DESC;


```
neo4j$ MATCH (u:User)-[b:BOUGHT]->(p:Product) WH
```
	User	Product	PricePaid
1	"Alice"	"Product1"	99.98

```
Started streaming 1 record after 74 ms and completed after 76 ms.
```

57. Identify categories generating the highest total revenue (Σ price_paid × quantity)

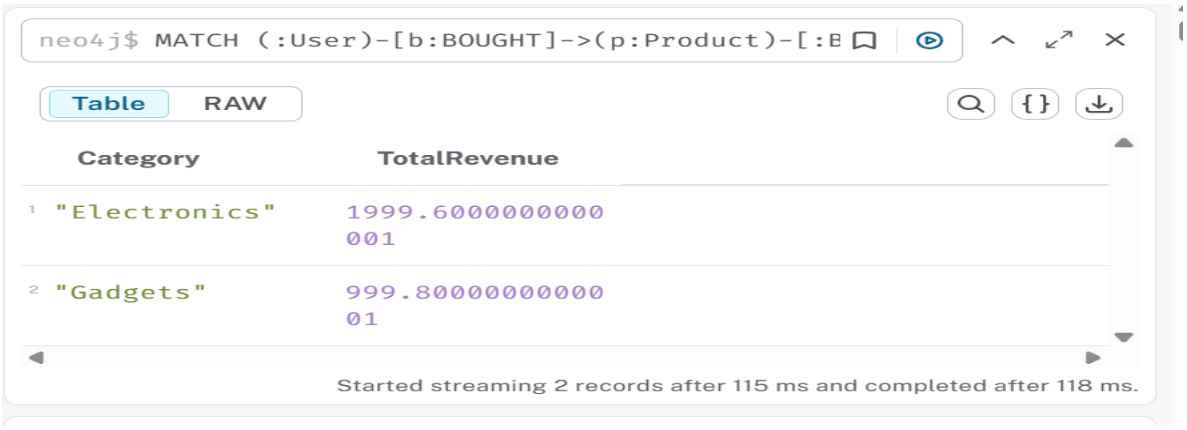
MATCH (:User)-[b:BOUGHT]->(p:Product)-[:BELONGS_TO]->(c:Category)

WITH c.name AS Category, SUM(b.price_paid * b.quantity) AS TotalRevenue

RETURN Category, TotalRevenue

ORDER BY TotalRevenue DESC

LIMIT 5;



```
neo4j$ MATCH (:User)-[b:BOUGHT]->(p:Product)-[:E
```

	Category	TotalRevenue
1	"Electronics"	1999.6000000000001
2	"Gadgets"	999.8000000000001

Started streaming 2 records after 115 ms and completed after 118 ms.

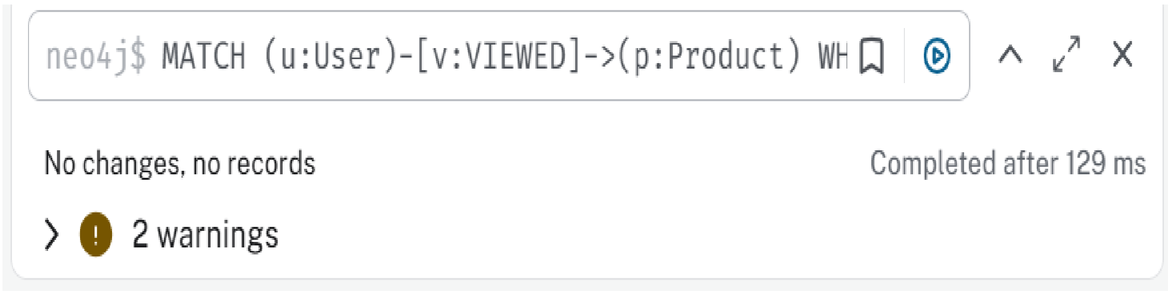
58. Find products viewed >100 seconds but never bought

MATCH (u:User)-[v:VIEWED]->(p:Product)

WHERE v.duration > 100 AND NOT (u)-[:BOUGHT]->(p)

RETURN DISTINCT p.name AS Product, AVG(v.duration) AS AvgViewTime

ORDER BY AvgViewTime DESC;



```
neo4j$ MATCH (u:User)-[v:VIEWED]->(p:Product) WH
```

No changes, no records

Completed after 129 ms

2 warnings

59. List products with review count and average rating

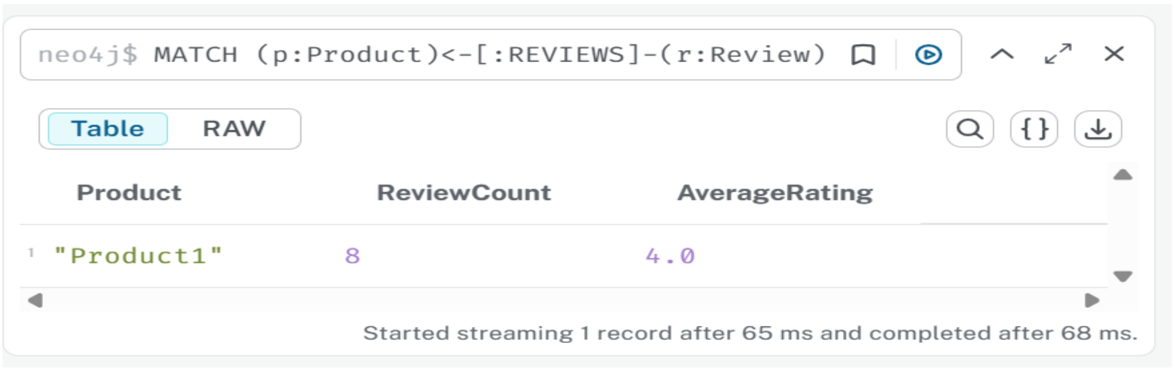
MATCH (p:Product)<-[:REVIEWS]-(r:Review)

RETURN p.name AS Product,

COUNT(r) AS ReviewCount,

ROUND(AVG(r.rating), 2) AS AverageRating

ORDER BY AverageRating DESC;



```
neo4j$ MATCH (p:Product)<-[:REVIEWS]-(r:Review)
```

	Product	ReviewCount	AverageRating
1	"Product1"	8	4.0

Started streaming 1 record after 65 ms and completed after 68 ms.

60. Identify potential bundles of Product1, Product2, Product3 based on co-purchases

MATCH (u:User)-[:BOUGHT]->(p:Product)

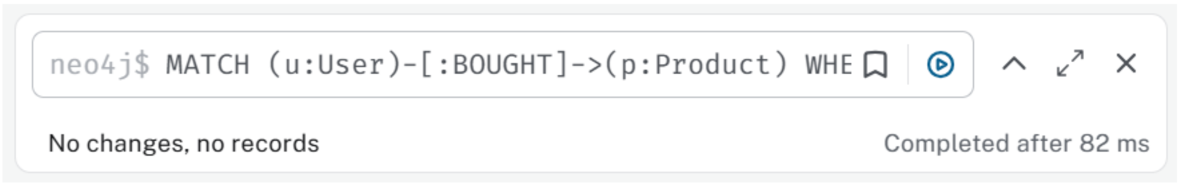
WHERE p.name IN ['Product1', 'Product2', 'Product3']

WITH u, COLLECT(DISTINCT p.name) AS BoughtProducts

WHERE size(BoughtProducts) > 1

RETURN BoughtProducts AS Bundle, COUNT(*) AS Frequency

ORDER BY Frequency DESC;



```
neo4j$ MATCH (u:User)-[:BOUGHT]->(p:Product) WHE
```

```
No changes, no records                                Completed after 82 ms
```