Shishir Rajendrakumar Deshpande Cypher Queries and Results

Query 1	Query 2	Query 3	Query 4
Creating the	// 2. Execute the	// 3. Execute the	// 4. Execute the following
database	following Cypher	following Cypher code to	Cypher code to get the list
	code to get the list of	get the list of employees:	of customers:
	retailers:		
		MATCH (e:Employee)	MATCH (c:Customer)
	MATCH (r:Retailer)	RETURN (e);	RETURN (c);
	RETURN (r);		
Query 5	Query 6	Query 7	Query 8
// 5. Execute the	// 6. Write the	// 7. Write the Cypher	// 8. Write the Cypher code
following Cypher	Cypher code to get	code to get the number	to get the number of
code to get the list	the number of	of disputed transactions	disputed transactions for
of all Disputed	disputed transactions	and the list of	every
transactions	for every	customer names for	customer that has more
MATCH	retailer	these disputed	than one disputed
(customer:Custome		transactions for every	transaction
r)-	MATCH (customer:Cu	retailer	
[transaction:SHOPP	stomer)-		MATCH (customer:Custome
ED_AT]->(retailer)	[transaction:SHOPPE	MATCH (customer:Custo	r)-
WHERE	D_AT]->(retailer)	mer)-	[transaction:SHOPPED_AT]-
transaction.status =	WHERE transaction.st	[transaction:SHOPPED_A	>(retailer)
"Disputed"	atus = "Disputed"	T]->(retailer)	WHERE transaction.status =
RETURN	RETURN retailer.nam	WHERE transaction.statu	"Disputed"
customer.name AS `Customer Name`,	e AS `Retailer Name`, COUNT(transacti	s = "Disputed" RETURN retailer.name AS	WITH customer, COUNT(tra nsaction) AS numDisputedT
retailer.name AS	on) AS `Number of Di	`Retailer Name`,	ransactions
`Retailer Name`,	sputed Transactions`	COUNT(transaction)	WHERE numDisputedTrans
transaction.amount	ORDER BY `Number o	AS `Number of Disputed	actions > 1
AS `Transaction	f Disputed Transactio	Transactions`,	RETURN customer.name AS
Amount`,	ns` DESC	COLLECT(DISTINCT cu	`Customer Name`,
transaction.date AS	5250	stomer.name) AS `Custo	numDisputedTransactio
`Transaction date`		mer Names`	ns AS `Number of Disputed
ORDER BY		ORDER BY `Number of Di	Transactions`
`Transaction date`		sputed Transactions` DES	ORDER BY numDisputedTra
DESC		C	nsactions DESC

Query 9	Query 10	Query 11	Query 12
// 9. Write the Cypher	// 10. Write the Cypher code to	// Tutorial 6. Get a	// Tutorial 7. Get the
code to get the list of	get the list of Employees who	list of Approved	number of times we
stores on LaSalle	work in at least 2 stores	transactions	see each Retailer in
street that have	where disputed transactions	occurred before the	disputed transactions
disputed transactions	reported in these retailers	disputed transaction	MATCH
and the number of		MATCH	(customer:Customer)-
disputed transactions	MATCH (customer)-	(customer:Customer	[transaction1:SHOPPE
for every store;	[transaction : SHOPPED_AT]-)-	D_AT]->(retailer1)
the store list must be	>(retailer)	[transaction1:SHOPP	WHERE
sorted by store name	WHERE transaction.status =	ED_AT]->(retailer1)	transaction1.status =
in ascending order.	"Disputed"	WHERE	"Disputed"
	MATCH	transaction1.status =	MATCH (customer)-
MATCH (customer:Cus	(employee1:Employee)-	"Disputed"	[transaction2:SHOPPE
tomer)-	[:WORKS_AT]->(retailer1)	MATCH (customer)-	D_AT]->(retailer2)
[transaction:SHOPPED	WHERE retailer.name =	[transaction2:SHOPP	WHERE
_AT]->	retailer1.name	ED_AT]->(retailer2)	transaction2.status =
(retailer:Retailer)	MATCH	WHERE	"Approved" AND
WHERE transaction.sta	(employee2:Employee)-	transaction2.status =	transaction2.date <
tus = "Disputed" AND r	[:WORKS_AT]->(retailer2)	"Approved" AND	transaction1.date
etailer.street CONTAIN	WHERE employee1.name =	transaction2.date <	WITH customer,
S "LaSalle"	employee2.name AND	transaction1.date	retailer2, transaction2
WITH retailer, COUNT(retailer1.name <>	WITH customer,	ORDER BY
transaction) AS numDi	retailer2.name	retailer2,	transaction2.date
sputedTransactions	WITH employee1, retailer1,	transaction2 ORDER	DESC
RETURN retailer.name	retailer2, customer, retailer	BY transaction2.date	RETURN DISTINCT
AS `Retailer Name`,	Return employee1.name AS	DESC	retailer2.name AS
retailer.street AS `	`Employee`, retailer1.name AS	RETURN	`Retailer`,
Retailer Address`,	`Retailer1`,	customer.name AS	count(DISTINCT
numDisputedTrans	collect(DISTINCT	`Customer Name`,	transaction2)
actions AS `Number of	retailer2.name) AS `Retailer2`,	retailer2.name AS	AS Count,
Disputed Transactions`	customer.name AS `Customer	`Retailer	collect(DISTINCT
ORDER BY `Retailer Na	Name`,	Name`,	customer.name) AS
me` ASC	retailer.name AS `Retailer	transaction2.amount	Customer
	Name`	AS Amount,	ORDER BY Count
		transaction2.date AS	DESC
		`Transaction date`	
		ORDER BY	
		`Transaction date`	
		DESC	

Query 13Query 14Query 15Query 168. Get the number of disputed transactions for every store9. Get the number of disputed transactions for every store// Tutorial 10. Get the number of disputed transactions for every store// Tutorial 11. Get the top 3 customers that reported the mumber of disputed transactions for EVERY CUSTOMER that has more transactions for EVERY CUSTOMER that has more transactions// Tutorial 11. Get the top 3 customers that reported the maximum number of disputed transactions for EVERY CUSTOMER that has more transactionsMATCH (customer:Custome r)- [transactions for EVERY STORE]WATCH (customer:Customer)- [transaction:SHOPPED_AT]->(retailer)WHEREMATCH (customer:Customer)- [transaction:SHOPPED_AT]->(retailer)WHERE transaction.Status = "Disputed""Disputed"WHEREWHERE transaction.Status = "Disputed"WHEREWHEREWHERE transaction.Status = "Disputed"WHERE"Disputed"WITH customer, count(*) AS number_of_disputed_transactions > 0RETURN DISTINCT retailer, transactionWHERE transactions > 0WITH customer, number_of_disputed_transactions > 0RETURN OSTOMER BY Count DESCRETURN Customer.name AS 'Fraud Store', count(flashed) and part of the part of transaction and part of transactionsRETURN Customer.name AS Customer, number_of_disputed_transactionsAS Count ame) AS Customers ORDER BY CountAS Coustomers on the part of transaction and part of transactionsRETURN customer.name AS Customer, number_of_disputed_transactions
DESC

Query 17	Query 18	Query 19	Query 20
// Tutorial 12. Get the list of	// Tutorial 13. Get the list	// Tutorial 14. Get the list of	// Tutorial 15. Get the list
stores on LaSalle street that	of customers and stores	customers who have	of customers who have
have disputed tranasctions	that have disputed	MEMBERSHIP in those stores	online accounts and have
and the	transactions where	but have	disputed
number of disputed	Jonathan reported a	disputed transactions.	transaction
transactions for every store	disputed transaction	MATCH	MATCH
// The store list must be	MATCH	(customer:Customer)-	(customer:Customer)-
sorted by store name in	(customer:Customer)-	[transaction:SHOPPED_AT]-	[transaction:SHOPPED_A
ascending order.	[transaction:SHOPPED_AT]-	>(retailer)WHERE	T]->(retailer)
MATCH	>(retailer)	transaction.status =	WHERE transaction.status
(customer:Customer)-	WHERE transaction.status =	"Disputed"	= "Disputed"
[transaction:SHOPPED_AT]-	"Disputed" AND	MATCH (customer)-	MATCH (customer)-
>(retailer)	customer.name =	[membership:HAS_MEMBER	[used_account:USES_ACC
WHERE transaction.status =	"Jonathan	SHIP_AT]->(retailer)	OUNT]->(account)
"Disputed" AND	Rinka"	WITH customer,	WITH customer, account,
retailer.street =~ '.*	MATCH (customer2)-	membership, retailer	used_account, retailer,
LaSalle.*'	[transaction2:SHOPPED_AT	RETURN customer.name AS	transaction
WITH retailer, count(*) AS]->(retailer2)	`Customer Name`,	RETURN customer.name
number_of_disputed_transa	WHERE transaction2.status	membership.reward_points	AS `Customer`,
ctions	= "Disputed" AND	AS `Reward Points`,	account.login_id AS
WHERE	retailer.name =	retailer.name AS `Retailer	`Account
number_of_disputed_transa	retailer2.name AND	Name`	Used`,
ctions > 0	customer2.name <>		used_account.last_date_
WITH retailer,	"Jonathan Rinka"		accessed AS `Account
number_of_disputed_transa	WITH customer,		Accesed On`,
ctions	customer2, retailer,		retailer.name AS `Retailer
ORDER BY retailer.name ASC	retailer2		Name`, transaction.date
RETURN retailer.name AS	RETURN customer.name AS		AS `Disputed
Retailer ,	`Customer`,		Transaction Date`
number_of_disputed_transa	customer2.name AS `Other		
ctions	Customer`, retailer2.name		
	AS `Retailer`		

Query 21

// Tutorial 16. Get the list of Employees who work in at least 2 stores where disputed transactions reported these stores

MATCH (customer)-[transaction:SHOPPED_AT]->(retailer)

WHERE transaction.status = "Disputed"

MATCH (employee1:Employee)-[:WORKS_AT]->(retailer1)

WHERE retailer.name = retailer1.name

MATCH (employee2:Employee)-[:WORKS_AT]->(retailer2)

WHERE employee1.name = employee2.name AND retailer1.name <>

retailer2.name

WITH employee1, retailer1, retailer2, customer, retailer

Return employee1.name AS `Employee`, retailer1.name AS `Retailer1`,

collect(DISTINCT retailer2.name) AS `Retailer2`, customer.name,

retailer.name