# Shishir Rajendrakumar Deshpande

Cypher Queries and Results

| Query 1 | Query 2 | Query 3 | Query 4 |
|---|---|---|---|
| Creating the database | // 2. Execute the following Cypher code to get the list of retailers:<br><br>MATCH (r:Retailer)<br>RETURN (r); | // 3. Execute the following Cypher code to get the list of employees:<br><br>MATCH (e:Employee)<br>RETURN (e); | // 4. Execute the following Cypher code to get the list of customers:<br><br>MATCH (c:Customer)<br>RETURN (c); |
| **Query 5** | **Query 6** | **Query 7** | **Query 8** |
| // 5. Execute the following Cypher code to get the list of all Disputed transactions<br>MATCH (customer:Customer)-[transaction:SHOPPED_AT]->(retailer)<br>WHERE transaction.status = "Disputed"<br>RETURN customer.name AS `Customer Name`, retailer.name AS `Retailer Name`, transaction.amount AS `Transaction Amount`, transaction.date AS `Transaction date`<br>ORDER BY `Transaction date` DESC | // 6. Write the Cypher code to get the number of disputed transactions for every retailer<br><br>MATCH (customer:Customer)-[transaction:SHOPPED_AT]->(retailer)<br>WHERE transaction.status = "Disputed"<br>RETURN retailer.name AS `Retailer Name`,<br>COUNT(transaction) AS `Number of Disputed Transactions`<br>ORDER BY `Number of Disputed Transactions` DESC | // 7. Write the Cypher code to get the number of disputed transactions and the list of customer names for these disputed transactions for every retailer<br><br>MATCH (customer:Customer)-[transaction:SHOPPED_AT]->(retailer)<br>WHERE transaction.status = "Disputed"<br>RETURN retailer.name AS `Retailer Name`,<br>COUNT(transaction) AS `Number of Disputed Transactions`,<br>COLLECT(DISTINCT customer.name) AS `Customer Names`<br>ORDER BY `Number of Disputed Transactions` DESC | // 8. Write the Cypher code to get the number of disputed transactions for every customer that has more than one disputed transaction<br><br>MATCH (customer:Customer)-[transaction:SHOPPED_AT]->(retailer)<br>WHERE transaction.status = "Disputed"<br>WITH customer, COUNT(transaction) AS numDisputedTransactions<br>WHERE numDisputedTransactions > 1<br>RETURN customer.name AS `Customer Name`,<br>numDisputedTransactions AS `Number of Disputed Transactions`<br>ORDER BY numDisputedTransactions DESC |

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

| Query 13 | Query 14 | Query 15 | Query 16 |
|---|---|---|---|
| 8. Get the number of disputed transactions for every store<br><br>MATCH (customer:Customer)-[transaction:SHOPPED_AT]->(retailer)<br>WHERE transaction.status = "Disputed"<br>WITH customer, retailer, transaction<br>RETURN DISTINCT retailer.name AS `Retailer`, count(DISTINCT transaction)<br>AS Count<br>ORDER BY Count DESC | 9. Get the number of disputed transactions and the list of customer names for these disputed transactions for EVERY STORE<br><br>MATCH (customer:Customer)-[transaction:SHOPPED_AT]->(retailer)<br>WHERE transaction.status = "Disputed"<br>WITH customer, retailer, transaction<br>RETURN retailer.name AS `Fraud Store`, count(transaction)<br>AS Count, collect(customer.name) AS Customers<br>ORDER BY Count DESC | // Tutorial 10. Get the number of disputed transactions for EVERY CUSTOMER that has more<br>that one disputed transaction<br><br>MATCH (customer:Customer)-[transaction:SHOPPED_AT]->(retailer)<br>WHERE transaction.status = "Disputed"<br>WITH customer, count(*) AS number_of_disputed_transactions<br>WHERE number_of_disputed_transactions > 1<br>RETURN customer.name AS Customer, number_of_disputed_transactions | // Tutorial 11. Get the top 3 customers that reported the maximum number of disputed<br>transactions<br>MATCH (customer:Customer)-[transaction:SHOPPED_AT]->(retailer)<br>WHERE transaction.status = "Disputed"<br>WITH customer, count(*) AS number_of_disputed_transactionsWHERE number_of_disputed_transactions > 0<br>WITH customer, number_of_disputed_transactions<br>ORDER BY number_of_disputed_transactions DESC LIMIT 3<br>RETURN customer.name AS Customer, number_of_disputed_transactions |

| Query 17 | Query 18 | Query 19 | Query 20 |
|---|---|---|---|
| // Tutorial 12. Get the list of stores on LaSalle street that have disputed tranasctions and the<br>number of disputed transactions for every store<br>// The store list must be sorted by store name in ascending order.<br>MATCH (customer:Customer)-[transaction:SHOPPED_AT]->(retailer)<br>WHERE transaction.status = "Disputed" AND retailer.street =~ '.* LaSalle.*'<br>WITH retailer, count(*) AS number_of_disputed_transactions<br>WHERE number_of_disputed_transactions > 0<br>WITH retailer, number_of_disputed_transactions<br>ORDER BY retailer.name ASC<br>RETURN retailer.name AS Retailer , number_of_disputed_transactions | // Tutorial 13. Get the list of customers and stores that have disputed transactions where Jonathan reported a disputed transaction<br>MATCH (customer:Customer)-[transaction:SHOPPED_AT]->(retailer)<br>WHERE transaction.status = "Disputed" AND customer.name = "Jonathan Rinka"<br>MATCH (customer2)-[transaction2:SHOPPED_AT]->(retailer2)<br>WHERE transaction2.status = "Disputed" AND retailer.name = retailer2.name AND customer2.name <> "Jonathan Rinka"<br>WITH customer, customer2, retailer, retailer2<br>RETURN customer.name AS `Customer`, customer2.name AS `Other Customer`, retailer2.name AS `Retailer` | // Tutorial 14. Get the list of customers who have MEMBERSHIP in those stores but have<br>disputed transactions.<br>MATCH (customer:Customer)-[transaction:SHOPPED_AT]->(retailer)WHERE transaction.status = "Disputed"<br>MATCH (customer)-[membership:HAS_MEMBERSHIP_AT]->(retailer)<br>WITH customer, membership, retailer<br>RETURN customer.name AS `Customer Name`, membership.reward_points AS `Reward Points`, retailer.name AS `Retailer Name` | // Tutorial 15. Get the list of customers who have online accounts and have disputed<br>transaction<br>MATCH (customer:Customer)-[transaction:SHOPPED_AT]->(retailer)<br>WHERE transaction.status = "Disputed"<br>MATCH (customer)-[used_account:USES_ACCOUNT]->(account)<br>WITH customer, account, used_account, retailer, transaction<br>RETURN customer.name AS `Customer`, account.login_id AS `Account<br>Used`, used_account.last_date_accessed AS `Account Accesed On`, retailer.name AS `Retailer Name`, transaction.date AS `Disputed Transaction Date` |

| Query 21 |
|---|
| // Tutorial 16. Get the list of Employees who work in at least 2 stores where disputed transactions reported these stores<br>MATCH (customer)-[transaction:SHOPPED_AT]->(retailer)<br>WHERE transaction.status = "Disputed"<br>MATCH (employee1:Employee)-[:WORKS_AT]->(retailer1)<br>WHERE retailer.name = retailer1.name<br>MATCH (employee2:Employee)-[:WORKS_AT]->(retailer2)<br>WHERE employee1.name = employee2.name AND retailer1.name <> retailer2.name<br>WITH employee1, retailer1, retailer2, customer, retailer<br>Return employee1.name AS `Employee`, retailer1.name AS `Retailer1`, collect(DISTINCT retailer2.name) AS `Retailer2`, customer.name, retailer.name |