# NBD Assignment 6 – Neo4J

The assignment should be submitted AS a text file WITH numbered queries + result files IN text and image format (e.g. SVG and JSON – available export options change quite often) named resultX.svg and resultX.json (X – query number, extensions depend on available file formats, export image files WHERE relevant). Export result files using web interface.

**Part 1 – Mountain trips**

Import data running task2.cypher. Write the following queries:

1. Shortest (least stages) routes from Darjeeling to Sandakphu that can be used IN winter

*MATCH p=shortestPath((t:town {name:"Darjeeling"})-[\*]-(peak:peak {name:"Sandakphu"}))*

*WHERE ALL (r IN relationships(p) WHERE r.winter='true')*

*RETURN p*

1. All routes you can use to get from Darjeeling to Sandakphu, sorted by the distance you need to travel

*MATCH paths=(t:town {name:"Darjeeling"})-[r\*1..6]-(p:peak {name:"Sandakphu"}) WHERE ALL(n IN NODES(paths) WHERE SINGLE(nonDuplicatedNode IN NODES(paths) WHERE n=nonDuplicatedNode ))*

*RETURN distinct(paths),reduce(total=0, n IN relationships(paths)|total+n.distance) AS totalDistance*

*order by totalDistance*

1. All places you can get from Darjeeling using a twowheeler IN the summer

*MATCH paths=(t:town {name:"Darjeeling"})-[:twowheeler\*]-(place)*

*WHERE ALL (r IN relationships(paths) WHERE r.summer='true')*

*AND ALL(n IN NODES(paths) WHERE SINGLE(nonDuplicatedNode IN NODES(paths) WHERE n=nonDuplicatedNode ))*

*RETURN distinct place*

**Part 2 - Airlines**

Import data running task3.cypher. Write the following queries:

1. List of all airports sorted by number of flights starting IN them

*MATCH (f:Flight)-[:ORIGIN]->(a:Airport) RETURN a,count(f) AS flights ORDER BY flights*

1. List of all airports you can reach (directly or indirectly) from LAX spending less than 3000

*MATCH paths = (origin:Airport { name:"LAX" })<-[r:ORIGIN|DESTINATION\*1..6]->(destination:Airport) WHERE ALL(n IN NODES(paths) WHERE SINGLE(nonDuplicatedNode IN NODES(paths) WHERE n=nonDuplicatedNode )) AND ALL(notRepeatLastRelationship IN RELATIONSHIPS(paths) WHERE TYPE(notRepeatLastRelationship)<>TYPE( RELATIONSHIPS(paths)[[i IN RANGE(0, SIZE(RELATIONSHIPS(paths))-1) WHERE RELATIONSHIPS(paths)[i] = notRepeatLastRelationship][0]-1] ))*

*WITH paths,REDUCE(total = 0, n IN filter(x IN NODES(paths) WHERE 'Flight' IN LABELS(x)) | total + reduce(minPrice = 99999, price IN [(n)<-[:ASSIGN]-(ticket) | ticket.price]| CASE WHEN minPrice < price THEN minPrice ELSE price END)*

*) AS totalPrice WHERE totalPrice< 3000*

*RETURN paths,totalPrice*

1. Sort connections you can use to go from LAX to DAY by connection price

*MATCH paths = (origin:Airport { name:"LAX" })<-[r:ORIGIN|DESTINATION\*4..10]->(destination:Airport{name:"DAY" }) WHERE ALL(n IN NODES(paths) WHERE SINGLE(nonDuplicatedNode IN NODES(paths) WHERE n=nonDuplicatedNode )) AND ALL(notRepeatLastRelationship IN RELATIONSHIPS(paths) WHERE TYPE(notRepeatLastRelationship)<>TYPE( RELATIONSHIPS(paths)[[i IN RANGE(0, SIZE(RELATIONSHIPS(paths))-1) WHERE RELATIONSHIPS(paths)[i] = notRepeatLastRelationship][0]-1] ))*

*WITH paths, REDUCE(total = 0, n IN filter(x IN NODES(paths) WHERE 'Flight' IN LABELS(x)) | total + reduce(minPrice = 99999, price IN [(n)<-[:ASSIGN]-(ticket) | ticket.price]| CASE WHEN minPrice < price THEN minPrice ELSE price END)) AS totalPrice*

*RETURN paths,totalPrice ORDER BY totalPrice*

1. Cheapest connection from LAX to DAY

*MATCH paths = (origin:Airport { name:"LAX" })<-[r:ORIGIN|DESTINATION\*4..10]->(destination:Airport{name:"DAY" }) WHERE* *ALL(n IN NODES(paths) WHERE SINGLE(nonDuplicatedNode IN NODES(paths) WHERE n=nonDuplicatedNode )) AND ALL(notRepeatLastRelationship IN RELATIONSHIPS(paths) WHERE TYPE(notRepeatLastRelationship)<>TYPE( RELATIONSHIPS(paths)[[i IN RANGE(0, SIZE(RELATIONSHIPS(paths))-1) WHERE RELATIONSHIPS(paths)[i] = notRepeatLastRelationship][0]-1] ))*

*WITH MIN(REDUCE(total = 0, n IN filter(x IN NODES(paths) WHERE 'Flight' IN LABELS(x)) | total + reduce(minPrice = 99999, price IN [(n)<-[:ASSIGN]-(ticket) | ticket.price]| CASE WHEN minPrice < price THEN minPrice ELSE price END))) AS totalPrice*

*MATCH path= (origin:Airport { name:"LAX" })<-[r:ORIGIN|DESTINATION\*4..10]->(destination:Airport{name:"DAY" }) WHERE ALL(n IN NODES(path) WHERE SINGLE(nonDuplicatedNode IN NODES(path) WHERE n=nonDuplicatedNode )) AND ALL(notRepeatLastRelationship IN RELATIONSHIPS(paths) WHERE TYPE(notRepeatLastRelationship)<>TYPE( RELATIONSHIPS(paths)[[i IN RANGE(0, SIZE(RELATIONSHIPS(paths))-1) WHERE RELATIONSHIPS(paths)[i] = notRepeatLastRelationship][0]-1] ))*

*AND REDUCE(total = 0, n IN filter(x IN NODES(path) WHERE 'Flight' IN LABELS(x)) | total + reduce(minPrice = 99999, price IN [(n)<-[:ASSIGN]-(ticket) | ticket.price]| CASE WHEN minPrice < price THEN minPrice ELSE price END)) =totalPrice*

*RETURN path,totalPrice*

1. Cheapest connection from LAX to DAY IN business class

*MATCH paths = (origin:Airport { name:"LAX" })<-[r:ORIGIN|DESTINATION\*4..10]->(destination:Airport{name:"DAY" }) WHERE ALL(n IN NODES(paths) WHERE SINGLE(nonDuplicatedNode IN NODES(paths) WHERE n=nonDuplicatedNode )) AND ALL(notRepeatLastRelationship IN RELATIONSHIPS(paths) WHERE TYPE(notRepeatLastRelationship)<>TYPE( RELATIONSHIPS(paths)[[i IN RANGE(0, SIZE(RELATIONSHIPS(paths))-1) WHERE RELATIONSHIPS(paths)[i] = notRepeatLastRelationship][0]-1] ))*

*WITH MIN(REDUCE(total = 0, n IN [x IN NODES(paths) WHERE 'Flight' IN LABELS(x)] | total + [(n)<-[:ASSIGN]-(ticket{class:"business"}) | ticket.price][0])) AS totalPrice*

*MATCH path= (origin:Airport { name:"LAX" })<-[r:ORIGIN|DESTINATION\*4..10]->(destination:Airport{name:"DAY" }) WHERE ALL(n IN NODES(path) WHERE SINGLE(nonDuplicatedNode IN NODES(path) WHERE n=nonDuplicatedNode )) AND ALL(notRepeatLastRelationship IN RELATIONSHIPS(paths) WHERE TYPE(notRepeatLastRelationship)<>TYPE( RELATIONSHIPS(paths)[[i IN RANGE(0, SIZE(RELATIONSHIPS(paths))-1) WHERE RELATIONSHIPS(paths)[i] = notRepeatLastRelationship][0]-1] ))*

*AND REDUCE(total = 0, n IN [x IN NODES(path) WHERE 'Flight' IN LABELS(x)] |* total + [(n)<-[:ASSIGN]-(ticket{class:"business"}) | ticket.price][0]) =totalPrice

RETURN path,totalPrice

1. List of airlines sorted by the number of cities they offer connections to and from (unique cities taking part IN :ORIGIN and :DESTINATION of Flight nodes handled by given airline)

*MATCH (f:Flight) WITH DISTINCT f.airline AS airline*

*MATCH (f1:Flight)-[:ORIGIN]->(a1:Airport),*

*(f2:Flight)-[:DESTINATION]->(a2:Airport)*

*WHERE f1.airline=airline AND f2.airline=airline*

*RETURN airline,count(distinct(a1)) AS NumOriginCity,count(distinct(a2)) AS NumDestCity*

1. Cheapest connection between 3 different airports

*MATCH paths = (origin:Airport)<-[r:ORIGIN|DESTINATION\*4..4]->(destination:Airport) WHERE ALL(n IN NODES(paths) WHERE SINGLE(nonDuplicatedNode IN NODES(paths) WHERE n=nonDuplicatedNode )) AND ALL(notRepeatLastRelationship IN RELATIONSHIPS(paths) WHERE TYPE(notRepeatLastRelationship)<>TYPE( RELATIONSHIPS(paths)[[i IN RANGE(0, SIZE(RELATIONSHIPS(paths))-1) WHERE RELATIONSHIPS(paths)[i] = notRepeatLastRelationship][0]-1] ))*

*WITH MIN(REDUCE(total = 0, n IN [f IN NODES(paths) WHERE 'Flight' IN LABELS(f)] | total + reduce(minPrice = 99999, price IN [(n)<-[:ASSIGN]-(ticket) | ticket.price]| CASE WHEN minPrice < price THEN minPrice ELSE price END))) AS totalPrice*

*MATCH path= (origin:Airport)<-[r:ORIGIN|DESTINATION\*4..4]->(destination:Airport) WHERE ALL(n IN NODES(path) WHERE SINGLE(nonDuplicatedNode IN NODES(path) WHERE n=nonDuplicatedNode )) AND ALL(notRepeatLastRelationship IN RELATIONSHIPS(path) WHERE TYPE(notRepeatLastRelationship)<>TYPE( RELATIONSHIPS(path)[[i IN RANGE(0, SIZE(RELATIONSHIPS(path))-1) WHERE RELATIONSHIPS(path)[i] = notRepeatLastRelationship][0]-1] ))*

*AND REDUCE(total = 0, n IN [ f IN NODES(path) WHERE 'Flight' IN LABELS(f)] | total + reduce(minPrice = 99999, price IN [(n)<-[:ASSIGN]-(ticket) | ticket.price]| CASE WHEN minPrice < price THEN minPrice ELSE price END)) =totalPrice*

*RETURN path,totalPrice*