**// Loading Data**

CREATE CONSTRAINT ON (route:Route) ASSERT route.route\_id IS UNIQUE

CREATE CONSTRAINT ON (trip:Trip) ASSERT trip.trip\_id IS UNIQUE

CREATE CONSTRAINT ON (stop:Stop) ASSERT stop.stop\_id IS UNIQUE

LOAD CSV WITH HEADERS FROM "file:///shapes.csv" AS row

CREATE (shape:Shape {

    shape\_id: row.shape\_id,

    shape\_pt\_lat: row.shape\_pt\_lat,

    shape\_pt\_lon: row.shape\_pt\_lon,

    shape\_pt\_sequence: row.shape\_pt\_sequence,

    shape\_dist\_traveled: row.shape\_dist\_traveled

    })

LOAD CSV WITH HEADERS FROM "file:///routes.csv" AS row

CREATE (route:Route {

    route\_id: row.route\_id,

    agency\_id: row.agency\_id,

    route\_short\_name: row.route\_short\_name,

    route\_long\_name: row.route\_long\_name,

    route\_type: row.route\_type,

    route\_color: row.route\_color

    })

LOAD CSV WITH HEADERS FROM "file:///trips.csv" AS row

MATCH(route:Route{route\_id:toString(row.route\_id)})

CREATE (trip:Trip {

    service\_id: row.service\_id,

    trip\_id: row.trip\_id,

    trip\_headsign: row.trip\_headsign,

    direction\_id: row.direction\_id,

    shape\_id:row.shape\_id,

    block\_id: row.block\_id,

    wheelchair\_accessible: row.wheelchair\_accessible,

    bikes\_allowed: row.bikes\_allowed

    })

CREATE (trip)-[r2:FOLLOWS\_ROUTE]->(route)

LOAD CSV WITH HEADERS FROM "file:///trips.csv" AS row

MATCH(shape:Shape{shape\_id:toString(row.shape\_id)}),(trip:Trip{trip\_id:toString(row.trip\_id)})

CREATE (shape)-[r1:COVERS\_TRIP]->(trip)

LOAD CSV WITH HEADERS FROM "file:///stops.csv" AS row

CREATE (stop:Stop {

    stop\_id: row.stop\_id,

    stop\_code: row.stop\_code,

    stop\_name: row.stop\_name,

    stop\_desc: row.stop\_desc,

    stop\_lat: row.stop\_lat,

    stop\_lon: row.stop\_lon,

    zone\_id: row.zone\_id

    })

LOAD CSV WITH HEADERS FROM "file:///stop\_times.csv" AS row

MATCH(trip:Trip{trip\_id:toString(row.trip\_id)}), (stop:Stop{stop\_id:toString(row.stop\_id)})

CREATE (st:StopTime {

    arrival\_time: row.arrival\_time,

    departure\_time: row.departure\_time,

    stop\_sequence: row.stop\_sequence,

    stop\_headsign: row.stop\_headsign,

    pickup\_type:row.pickup\_type,

    drop\_off\_type: row.drop\_off\_type,

    shape\_dist\_traveled: row.shape\_dist\_traveled

    })

CREATE (trip)-[r1:HAS\_STOP\_TIME]->(st)-[r2:OCCURS\_AT\_STOP]->(stop)

**// Returning Complete schedule of ttc bus which has route\_id ="66509" and which starts at "5:44:00" from its base station**

match (stop:Stop)<-[r1:OCCURS\_AT\_STOP]-(st:StopTime)<-[r2:HAS\_STOP\_TIME]-(trip:Trip)-[r3:FOLLOWS\_ROUTE]->(route:Route)

where route.route\_id="66509" and trip.trip\_id="44088175"

return  stop.stop\_name as Stop, stop.stop\_id as Stope\_id, st.arrival\_time as Arrival\_Time

order by Arrival\_Time

or

match (stop:Stop)<-[r1:OCCURS\_AT\_STOP]-(st:StopTime)<-[r2:HAS\_STOP\_TIME]-(trip:Trip)-[r3:FOLLOWS\_ROUTE]->(route:Route)

match (stop1:Stop)<-[r11:OCCURS\_AT\_STOP]-(st1:StopTime)<-[r21:HAS\_STOP\_TIME]-(trip1:Trip)-[r31:FOLLOWS\_ROUTE]->(route1:Route)

where stop1.stop\_name = "Petrolia Rd at Steeles Ave West South Side" and st1.arrival\_time = "5:44:00"

and route.route\_id="66509" and trip.trip\_id = trip1.trip\_id

return  stop.stop\_name as Stop, stop.stop\_id as Stope\_id, st.arrival\_time as Arrival\_Time

order by Arrival\_Time

**// Jay is new in city and looking to explore the city. He is waiting for the bus at bus stop. He wants to know what kind of buses would come at that bus stop**

match (stop:Stop)<-[r1:OCCURS\_AT\_STOP]-(st:StopTime)<-[r2:HAS\_STOP\_TIME]-(trip:Trip)-[r3:FOLLOWS\_ROUTE]->(route:Route)

where stop.stop\_id="685" and st.arrival\_time>"5:51:46"

return  route.route\_long\_name as Route, trip.trip\_headsign as Trip, st.arrival\_time as Arrival\_Time

order by Arrival\_Time

LIMIT 5

**// Which shapes(or buses/subways) covers a particular trip with a given route ?**

**// trip = SOUTH - 107F YORK UNIVERSITY HEIGHTS towards WILSON STATION**

**// route= YORK UNIVERSITY HEIGHTS**

match (shape:Shape)-[r1:COVERS\_TRIP]->(trip:Trip)-[r2:FOLLOWS\_ROUTE]->(route:Route)

where  route.route\_long\_name ="YORK UNIVERSITY HEIGHTS" and trip.trip\_headsign="SOUTH - 107F YORK UNIVERSITY HEIGHTS towards WILSON STATION"

return distinct shape.shape\_id as Shapes

**// Now time is 14:00:00. Harshit lives near Keele St and Finch Ave West intersection. He is in his apartment and he is ready to leave for the college. He wants to know what all buses can help him to reach the college before 15:00:00. College is situtated near Sheppard West Station**

match (stop:Stop)<-[r1:OCCURS\_AT\_STOP]-(st:StopTime)<-[r2:HAS\_STOP\_TIME]-(trip:Trip)-[r3:FOLLOWS\_ROUTE]->(route:Route)

match (stop1:Stop)<-[r11:OCCURS\_AT\_STOP]-(st1:StopTime)<-[r21:HAS\_STOP\_TIME]-(trip:Trip)

where stop.stop\_name="Sheppard West Station" and st.arrival\_time < "13:30:00" and stop1.stop\_name="Keele St at Finch Ave West" and st1.arrival\_time > "12:00:00"

return  route.route\_long\_name as Route, trip.trip\_headsign as Trip, st1.arrival\_time as Departure\_Time, st.arrival\_time as Arrival\_Time

order by Arrival\_Time

**// Looking for all diiferent types of trips for given route YORK UNIVERSITY HEIGHTS**

match(trip:Trip)-[r1:FOLLOWS\_ROUTE]->(route:Route)

where route.route\_long\_name = "YORK UNIVERSITY HEIGHTS"

return trip, route

match(trip:Trip)-[r1:FOLLOWS\_ROUTE]->(route:Route)

where route.route\_long\_name = "YORK UNIVERSITY HEIGHTS"

return distinct trip.trip\_headsign

**// Top 5 longest routes (w.r.t. most number of stops)**

match (stop:Stop)<-[r1:OCCURS\_AT\_STOP]-(st:StopTime)<-[r2:HAS\_STOP\_TIME]-(trip:Trip)

with trip, count(stop) as Number\_Of\_Stops

return  distinct trip.trip\_headsign, Number\_Of\_Stops

order by Number\_Of\_Stops desc

Limit 5

**// Shortest path between two stops (not on the basis of distance or time, but on the basis of number of hops)**

MATCH p=shortestPath(

(stop:Stop {stop\_id:"685"})-[\*]-(stop1:Stop {stop\_id:"7337"})

)

RETURN distinct p