607 Week12 NoSQL Migration Process

# Overview

This assignment explores the differences of SQL and NoSQL databases. This is done by migrating a sample of records from an SQL database to a NoSQL database. The migration will be from MySQL to Neo4j.

# Assumptions

This process makes the following assumptions:

1. MySQL software environment is installed and configured
2. The “flights” database is loaded in that SQL environment
3. Neo4j software is installed and configured

# Process

1. **Create a subset of records from the flights database in a new schema named flights2**
   1. Limit to only 20 flights
   2. Limit to airlines and airports connected to those flights
   3. See appendix A for the SQL scripts used in this process
2. **Export each table in the flights2 schema as a .csv file**
3. **Create a new Neo4j graph named “Flights”**
   1. Open Neo4j
   2. Create a new project
   3. Within the project click “Add Graph”
   4. Click “Create Local Graph”
   5. Enter name “Flights”
   6. Enter password of your choosing
4. **Connect to the new Neo4j Flights database**
   1. Click on the “Start” button within the Flights graphic tile
5. **Open the Neo4j Browser**
   1. Click on the Neo4j browser tile at the top of the project space
6. **Use cypher (Neo4j) scripts to upload the csv data to the Neo4j graph**
   1. Create “Nodes” for flights, airports, and airlines (See Appendix C)
   2. Create “relationships” between flights-airports and flights-airlines (See Appendix D)
7. **Explore the Neo4j database by returning simple relationship graphs**
   1. For comprehensibility, only return two of the three node types at a time
   2. See Appendix E

# Conclusions

Both the relational (MySQL) and NoSQL (Neo4j) databases have advantages and disadvantages.

Relational databases strengths:

* The structure of the data (schemas) is visible as well as the data itself
* The data are stored more efficiently
* The query syntax is closer to common English and therefore easier to pick up

Relational databases weaknesses

* The data must conform to rigid structures
* Relationships between data are implicit (only visible in meta data; i.e., schema graphics)

Neo4j strengths

* The relationships of data points are themselves explicitly stored as data points
* Querying on relationships is faster
* The data can more easily be visualized

Neo4j weakness

* The storage of data is less efficient
* The query language is further from common English and therefore harder to acquire

# Appendix A – Create Flights2

SQL script used to create flights2 schema (subset of records from flights schema)

-- get flights (base)

DROP TABLE IF EXISTS flights2.flights;  
CREATE TABLE flights2.flights  
SELECT \* FROM flights.flights LIMIT 20;

-- limit to airports connected to the flights sample

DROP TABLE IF EXISTS flights2.airports;

CREATE TABLE flights2.airports

SELECT DISTINCT airports.\*

FROM flights.airports airports

INNER JOIN flights2.flights flights

ON flights.origin = airports.faa

OR flights.dest = airports.faa;

-- limit to airlines connected to the flights sample

DROP TABLE IF EXISTS flights2.airlines;

CREATE TABLE flights2.airlines

SELECT DISTINCT airlines.\*

FROM flights.airlines airlines

INNER JOIN flights2.flights flights

ON flights.carrier = airlines.carrier;

# Appendix B (Export Flights2 tables to CSV)

SQL script used to export the flights2 (subset of flights) tables to csv files.

USE flights2;

-- get the folder where files must be uploaded/downloaded under MySQLs new security settings

-- SHOW VARIABLES LIKE "secure\_file\_priv";

-- export tables

-- flights.airlines

SELECT 'Carrier', 'Name'

UNION SELECT \*

INTO OUTFILE

'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/Data\_607/flights\_export/airlines.csv'

FIELDS TERMINATED BY ','

OPTIONALLY ENCLOSED BY '"'

ESCAPED BY '\\'

LINES TERMINATED BY '\n'

FROM airlines;

-- flights.airports

SELECT 'FAA', 'Name', 'Lat', 'Lon', 'Alt', 'TZ', 'Dst'

UNION SELECT \*

INTO OUTFILE

'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/Data\_607/flights\_export/airports.csv'

FIELDS TERMINATED BY ','

OPTIONALLY ENCLOSED BY '"'

ESCAPED BY '\\'

LINES TERMINATED BY '\n'

FROM airports;

-- flights.flights

SELECT 'Year', 'Month', 'Day', 'Dep\_Time', 'Dep\_Delay', 'Arr\_Time', 'arr\_Delay', 'Carrier',

'TailNum', 'Flight', 'Origin', 'Dest', 'Air\_Time', 'Distance', 'Hour', 'Minute'

UNION SELECT \*

INTO OUTFILE

'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/Data\_607/flights\_export/flights.csv'

FIELDS TERMINATED BY ','

OPTIONALLY ENCLOSED BY '"'

ESCAPED BY '\\'

LINES TERMINATED BY '\n'

FROM flights;

# Appendix C – Scripts to load Nodes

Cypher (Neo4j) script used to load the flights, airports, and airline nodes

## // Load Flights

USING PERIODIC COMMIT LOAD CSV WITH HEADERS FROM "https://raw.githubusercontent.com/ChadRyanBailey/607-Data-Acquisition-and-Management/master/607-Week12-Assignment/03\_flights2\_exports/flights.csv" AS flights CREATE (n:Flights {number: flights.Flight, departTime: flights.Dep\_Time, arrivalTime: flights.Arr\_Time, distance: flights.Distance, airTime: flights.Air\_Time});

## // Load Airlines

USING PERIODIC COMMIT LOAD CSV WITH HEADERS FROM "https://raw.githubusercontent.com/ChadRyanBailey/607-Data-Acquisition-and-Management/master/607-Week12-Assignment/03\_flights2\_exports/airlines.csv" AS airlines CREATE (:Airlines {airlineCode: airlines.Carrier, airlineName: airlines.Name});

## // Load Airports

USING PERIODIC COMMIT LOAD CSV WITH HEADERS FROM "https://raw.githubusercontent.com/ChadRyanBailey/607-Data-Acquisition-and-Management/master/607-Week12-Assignment/03\_flights2\_exports/airports.csv" AS airports CREATE (:Airports {airportCode: airports.FAA, airportName: airports.Name});

# Appendix D – Load Relationships

Cypher (Neo4j) scripts to load the flights-airports and flights-airline relationships

## // Load Flights to Departure Airport

USING PERIODIC COMMIT LOAD CSV WITH HEADERS FROM "https://raw.githubusercontent.com/ChadRyanBailey/607-Data-Acquisition-and-Management/master/607-Week12-Assignment/03\_flights2\_exports/flights.csv" AS flights MATCH (a:Flights {number: flights.Flight}), (b:Airports {airportCode: flights.Origin}) MERGE (a)-[r:Departs]->(b);

## // Load Flights to Arrival Airport

USING PERIODIC COMMIT LOAD CSV WITH HEADERS FROM "https://raw.githubusercontent.com/ChadRyanBailey/607-Data-Acquisition-and-Management/master/607-Week12-Assignment/03\_flights2\_exports/flights.csv" AS flights MATCH (a:Flights {number: flights.Flight}), (b:Airports {airportCode: flights.Dest}) MERGE (a)-[r:Arrives]->(b);

## // Load Flights to Airline

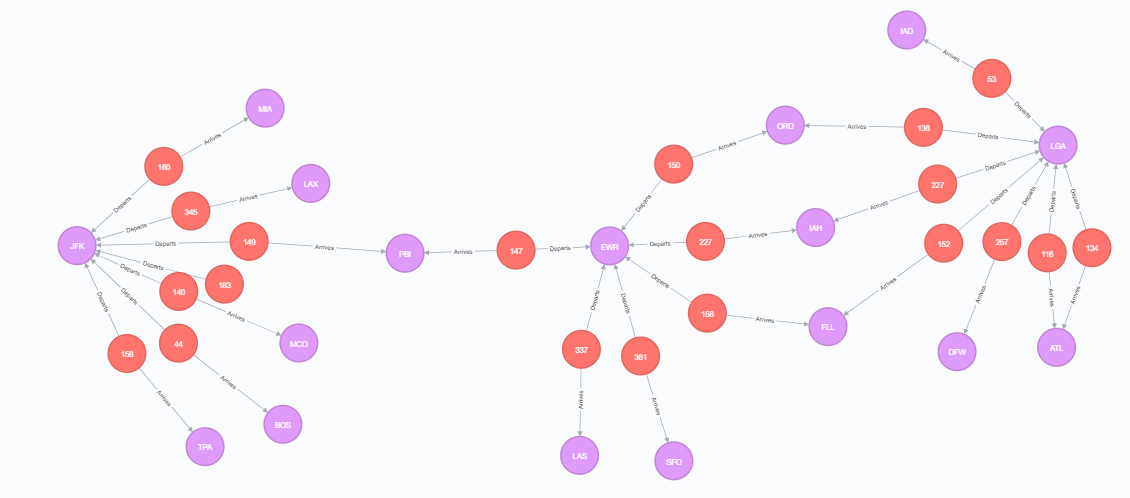
USING PERIODIC COMMIT LOAD CSV WITH HEADERS FROM "https://raw.githubusercontent.com/ChadRyanBailey/607-Data-Acquisition-and-Management/master/607-Week12-Assignment/03\_flights2\_exports/flights.csv" AS flights MATCH (a:Flights {number: flights.Flight}), (b:Airlines {airlineCode: flights.Carrier}) MERGE (a)-[r:Flown\_By]->(b);

# Appendix E – Explore the Loaded Graph

This provides the Cypher (Neo4j) queries used to return the exploratory graphs.

## Flight to Airport relationship graph

MATCH (n:Flights), (a:Airports) RETURN (n), (a)



# Flight to Airline relationship graph

MATCH (n:Flights), (b:Airlines) RETURN (n), (b)

