**This project was built with Neo4J**

**Create the database and use it**

CREATE DATABASE ONLMS;

USE ONLMS;

**Store data nodes and establish the relationships that exist between the collections (tables):**

//Provinces

CREATE (province1:Province {provinceCode: 'GP', name: 'Gauteng'})

CREATE (province2:Province {provinceCode: 'WC’, name: 'Western Cape'});

//Municipality

CREATE (municipality1:Municipality {munCode: 'JHB', name: 'City of Johannesburg', averagePopulation: 4949347, provinceCode: 'GP'})

CREATE (municipality2:Municipality {munCode: 'CPT', name: 'City of Cape Town', averagePopulation: 4005016, provinceCode: 'WC'});

//Facility

CREATE (facility1:Facility {facilityId: 'FAC001', name: 'Modise Cultural Facility', address: '123 Cultual St', capacity: 500, munCode: 'JHB'})

CREATE (facility2:Facility {facilityId: 'FAC002', name: 'Ubuntu Arts Centre', address: '456 Ubuntu Ave', capacity: 300, munCode: 'CPT'});

//Room

CREATE (room1:Room {roomNo: 'R001', description: 'Theatre', facilityId: 'FAC001'})

CREATE (room2:Room {roomNo: 'R002', description: 'Cinema', facilityId: 'FAC001'})

CREATE (room3:Room {roomNo: 'R003', description: 'Exhibition Hall', facilityId: 'FAC002'})

CREATE (room4:Room {roomNo: 'R004', description: 'Workshop Room', facilityId: 'FAC002'});

//Activity

CREATE (activity1:Activity {activityRef: 'ACT001', name: 'IsiZulu Dance Competition'})

CREATE (activity2:Activity {activityRef: 'ACT002', name: 'Art Exhibition'})

CREATE (activity3:Activity {activityRef: 'ACT003', name: 'Music Concert'})

CREATE (activity4:Activity {activityRef: 'ACT004', name: 'Theatre Performance'});

//Uses

CREATE (use1:Uses {facilityId: 'FAC001', activityRef: 'ACT001', Date: '2025-05-20'})

CREATE (use2:Uses {facilityId: 'FAC001', activityRef: 'ACT002', Date: '2025-06-20'})

CREATE (use3:Uses {facilityId: 'FAC002', activityRef: 'ACT003', Date: '2025-10-20'})

CREATE (use4:Uses {facilityId: 'FAC002', activityRef: 'ACT004', Date: '2025-12-12'});

//Create relationship between province and municipality

MATCH (municipality1:Municipality {munCode: 'JHB'}), (province1:Province {provinceCode: 'GP'})

CREATE (municipality1)-[r:HAS\_MUNICIPALITY]->(province1);

MATCH (municipality2:Municipality {munCode: 'CPT'}), (province2:Province {provinceCode: 'WC'})

CREATE (municipality2)-[r:HAS\_MUNICIPALITY]->(province2);

//Create relationships between facilities and municipality

MATCH (municipality1:Municipality {munCode: 'JHB'}), (facility1:Facility {facilityId: 'FAC001'})

CREATE (municipality1)-[r:HAS\_FACILITY]->(facility1);

MATCH (municipality2:Municipality {munCode: 'CPT'}), (facility2:Facility {facilityId: 'FAC002'})

CREATE (municipality2)-[r:HAS\_FACILITY]->(facility2);

//Create relationships between rooms and facilities

MATCH (facility1:Facility {facilityId: 'FAC001'}), (room1:Room {roomNo: 'R001'})

CREATE (facility1)-[r:HAS\_ROOM]->(room1);

MATCH (facility1:Facility {facilityId: 'FAC001'}), (room2:Room {roomNo: 'R002'})

CREATE (facility1)-[r:HAS\_ROOM]->(room2);

MATCH (facility2:Facility {facilityId: 'FAC002'}), (room3:Room {roomNo: 'R003'})

CREATE (facility2)-[r:HAS\_ROOM]->(room3);

MATCH (facility2:Facility {facilityId: 'FAC002'}), (room4:Room {roomNo: 'R004'})

CREATE (facility2)-[r:HAS\_ROOM]->(room4);

//Create relationships between activities and facilities

MATCH (facility1:Facility {facilityId: 'FAC001'}), (activity1:Activity {activityRef: 'ACT001'})

CREATE (facility1)-[r:HOSTS\_ON\_2025\_05\_20]->(activity1);

MATCH (facility1:Facility {facilityId: 'FAC001'}), (activity2:Activity {activityRef: 'ACT002'})

CREATE (facility1)-[r:HOSTS\_ON\_2025\_06\_20]->(activity2);

MATCH (facility2:Facility {facilityId: 'FAC002'}), (activity3:Activity {activityRef: 'ACT003'})

CREATE (facility2)-[r:HOSTS\_ON\_2025\_10\_20]->(activity3);

MATCH (facility2:Facility {facilityId: 'FAC002'}), (activity4:Activity {activityRef: 'ACT004'})

CREATE (facility1)-[r:HOSTS\_ON\_2025\_12\_12]->(activity4);

**Using query, display all the created collections (nodes) and their relationships in a graph data model.**

MATCH p=()-->() RETURN p LIMIT 25;

**Create a script to display the total capacity of facilities within each municipality.**

MATCH (municipality:Municipality)-[:HAS\_FACILITY]->(facility:Facility)

RETURN

    municipality.name AS MunicipalityName,

    SUM(facility.capacity) AS TotalCapacity

;

**Retrieve and display a list of all facilities along with their municipalities and provinces.**

MATCH (facility:Facility)<-[:HAS\_FACILITY]->(municipality:Municipality)<-[:HAS\_MUNICIPALITY]->(province:Province)

RETURN

    facility.name AS FacilityName,

    municipality.name AS MunicipalityName,

    province.name AS ProvinceName

;

**Write a query to determine and display all activities along with the number of days until they occur from today.**

MATCH (facility:Facility)-[h]->(activity:Activity)

WHERE type(h) STARTS WITH "HOSTS\_ON\_"

WITH

    activity.name AS ActivityName,

    facility.name AS FacilityName,

    substring(type(h), size("HOSTS\_ON\_")) AS DateAsStringUnderscores

WITH

    ActivityName,

    FacilityName,

    trim(replace(DateAsStringUnderscores, '\_', '-')) AS FormattedDate

WITH

    ActivityName,

    FacilityName,

    FormattedDate,

    date(FormattedDate) AS ActivityDate

WITH

    ActivityName,

    FacilityName,

    ActivityDate,

    duration.inDays(date(), ActivityDate).days AS DaysUntilActivity

RETURN

    ActivityName,

    FacilityName,

    ActivityDate,

    DaysUntilActivity

ORDER BY

    DaysUntilActivity ASC;

;

**Write a query to retrieve and display a list of municipalities with an average population of 4,000,000 (4 million) or more.**

MATCH (municipality:Municipality)

WHERE municipality.averagePopulation >=4000000

RETURN

    municipality.munCode AS MunicipalityCode,

    municipality.name AS MunicipalityName,

    municipality.averagePopulation AS AveragePopulation

ORDER BY

    AveragePopulation DESC

;