CMPU4003 Advanced Databases

**TU Dublin TU856/TU857/TU858**

**Advanced Databases**

Setup and Query Data in PostgreSQL, CouchDB, Cassandra

Contents

[Overview 1](#_Toc178180257)

[PART ONE POSTGRESQL 2](#_Toc178180258)

[Step 1 Create and Populate a Schema 2](#_Toc178180259)

[Step 2 Query the data 3](#_Toc178180260)

[PART TWO COUCHDB 4](#_Toc178180261)

[Step 1 Create a New Database 4](#_Toc178180262)

[Step 2 Populate with Data 4](#_Toc178180263)

[Step 3 Create a Query 5](#_Toc178180264)

[Step 4 Run the Query 7](#_Toc178180265)

[PART THREE CASSANDRA 10](#_Toc178180266)

[Step 1 Start the Cassandra Cluster 10](#_Toc178180267)

[Step 2 Create a Keyspace and Table 10](#_Toc178180268)

[Step 3 Populate the table 10](#_Toc178180269)

[Step 4 Query the data 10](#_Toc178180270)

# Overview

You are going to:

* PostgreSQL
  + Set up the database in PostgreSQL(using SQL);
  + Run a Python script to your workspace/project which will populate this database with random data;
  + Create a query to retrieve data from which documents can be created.
  + Export that data to JSON.
* CouchDB
  + Import JSON data into CouchDB.
  + Build a design document in CouchDB.
* Cassandra
  + Import JSON data into Cassandra.
  + Query that data.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

# PART ONE POSTGRESQL

## Step 1 Create and Populate a Schema

* Download the **ExamResultsDBCreate.zip file** from Brightspace.
* Start your postgres docker container.
* Connect to your data source.
* Unzip this file and run the SQL scripts in this order
  + Create the schema and tables:
    - ExamResultsDBCreate.SQL
  + Populate the tables:
    - Departments.sql
    - Courses.sql
    - Exams.sql
    - Students.sql
    - Examresults.sql
* You should now have a schema called **examresultsdb with the following schema**

A screenshot of a computer

Description automatically generated

* Note: anything identified as int(4) in the diagram is just an INT datatype (this was generated from DBeaver).

## Step 2 Query the data

* + You will need to tell postgres you want to use the examresultsdb schema

set search\_path to "examresultsdb";

* + Write the SQL needed to:
    - Retrieve the following details for each exam result:
      * studentid, student firstname, student lastname, examid, examdate, score and grade
      * You will be using a join between examresults and students.

# PART TWO COUCHDB

## Step 1 Create a New Database

* Start your CouchDB docker container if you need to.
* Launch Fauxton.
* Create a database called **examresults**

## Step 2 Populate with Data

* Download **examresults.json** from Brightspace
* Change username and password in the following command to the ones you created for your CouchDB and then execute the command a terminal/command prompt/git bash shell started in the folder where you have stored the json file

curl -X POST http://username:password@127.0.0.1:5984/examresults/\_bulk\_docs -H "Content-type: application/json" -d @examresults.json

* Verify that the documents have been created by viewing them using Fauxton.

## Step 3 Create a Query

* Create a query using a design document to retrieve the student firstname, lastname, examid and grade achieved.
* In Fauxton, Select Design Documents and New View

A screenshot of a computer

Description automatically generated

You will get this dialog:

A screenshot of a computer

Description automatically generated

Complete as follows:

A screenshot of a computer

Description automatically generated

Once you are happy, Create and Build it

Code for what you want to emit

Name of the design view

Name of the design document

## Step 4 Run the Query

* Under Design Documents you should now see your document and the view you created
* If you click on the view it will show you the results – all documents it has found

A screenshot of a computer

Description automatically generated

* If you click on Table it will change the view to show you the fields you have retrieved:
* You can use the drop downs to change the fields displayed.
* You can use the options to change exactly what is being displayed

A screenshot of a computer

Description automatically generated

* By Keys allows you to select individual documents matching values.
* Between Keys allows you to select a range for example to retrieve only those with Firstnames Amanda and Cheryl:

A screenshot of a computer

Description automatically generated

* Your design document has been stored as a document in the database.
* If you go back and look at all the documents, it will appear in the list:



* And you can edit it directly:

A screenshot of a computer

Description automatically generated

* Change the view examgrades to only emit the lastname, examid and grade and rerun the view.

# PART THREE CASSANDRA

## Step 1 Start the Cassandra Cluster

* If you need to, Start your docker Cassandra cluster.
* Connect to it.
* WAIT for the three nodes to be Up and Normal (check using nodetool status)

## Step 2 Create a Keyspace and Table

* Create a keyspace called examresults and use it:

CREATE KEYSPACE examresults WITH replication = {'class':'SimpleStrategy' , 'replication\_factor' : 2};

USE examresults;

* Create a table

CREATE TABLE studentresults (studentid int, firstname text, lastname text, examid int, examdate date, score decimal, grade varchar, PRIMARY KEY(studentid, examid));

## Step 3 Populate the table

* Download the file **CassandraExamResults.cql** from Brightspace.
* Open the file.
* Select all the insert statements and copy them.
* Go to the CQLSH prompt and paste all the commands (as a batch) and hit return
* This will populate your data.

## Step 4 Query the data

* Write a query to find all rows of data for a particular student firstname.
  + You can find help on CQL here <https://www.pythian.com/blog/technical-track/cassandra-cql-cheatsheet>
  + Strings should be enclosed in single quotes
* You will get an error message like this the first time you do it:



* You need to include ALLOW FILTERING at the end of your select statement.
* E.g.

SELECT name from table x where name=’X’ ALLOW FILTERING;