**RMIT University**

**COSC2406/2407 – Database Systems**

*Assignment #1*

Creating Derby and MongoDB databases and implementing a heap file in Java

**Task 1: Derby**

***Summary***

I created a Java app that creates derby database, reads a file that contains the data to be imported, creates statements and sends them to the database. At first I was going to create 2 tables: one having all the info about the business and second to have ABNs as help file that was provided with the dataset specifies that there could be more than one. And then link the tables. However after inspecting the data, I found that maximum number of ABNs provided is 1. The structure of the database is:

create table businessNames (id integer not null generated always as identity (start with 1, increment by 1),

name varchar(200) not null,

status varchar(20) not null,

registerDate date,

cancelDate date,

renewDate date,

stateNumber varchar(10),

state varchar(3),

abn varchar(20),

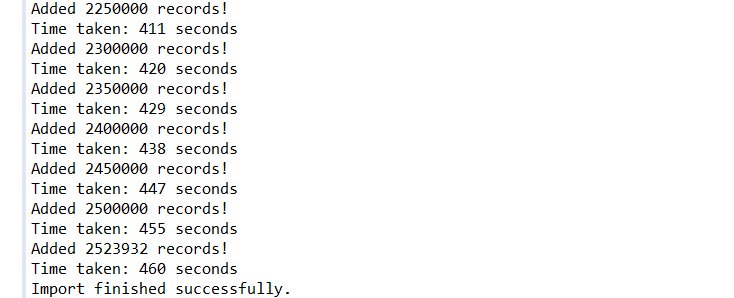
primary key(id))

I faced a problem of the app being super slow because once it read 1 line from the file it would send insert command to the database. So I decided to collect 100 record (maximum size for insert command) and add them to the database in one go.

***Timing***

At first it would take over 25 minutes to finish but after the changes stated above on average it would take 450-490 seconds (7.5 – 8.16 minutes) to import all 2523932 records to the database.

Here’s the screenshot from the app:



**Task 2: MongoDB**

***Summary***

Mongoimport was a very useful command for importing this kind of data into the database. Firstly I used the command like:

mongoimport --db MyDB --collection business --type tsv --columnsHaveTypes --headerline --file dataset.csv

However, I wanted the dates to be in date format to make it easier in the future to perform a reasonable search. I had 2 choices to do that: create a file that provides the fields’ names and their types or change the header line to provide relevant(specifying the date format to covert with no errors) field types as well. I went with the second choice and changed the first line to:

REGISTER\_NAME.string() Name.string() Status.string() RegistrationDate.date\_ms(dd/MM/yyyy) CancelationDate.date\_ms(dd/MM/yyyy) RenewDate.date\_ms(dd/MM/yyyy) StateNumber.string() State.string() ABN.string()

Also I’ve added –ignoreBlanks flag to the command to skip the empty fields without creating them in the database. So new command looked like this:

mongoimport --db MyDB --collection business --type tsv --columnsHaveTypes --headerline --file text10.csv --ignoreBlanks

After importing the data I decided to get rid of the first field for all the records as it’s the same for all 2523932 records in the file.

db.business.update({}, {$unset: {words:1}}, false, true);

***Timing***

I ran the import batch file on my pc at home and on my laptop to get the average time it takes to import the data which was 24-25 seconds. Here is a snapshot of the progress and successful import.

2018-03-31T13:25:48.809+1100 connected to: localhost

2018-03-31T13:25:50.747+1100 [##......................] MyDB.business 18.5MB/215MB (8.6%)

2018-03-31T13:25:53.746+1100 [#####...................] MyDB.business 44.9MB/215MB (20.9%)

2018-03-31T13:25:56.747+1100 [#######.................] MyDB.business 70.6MB/215MB (32.8%)

2018-03-31T13:25:59.747+1100 [##########..............] MyDB.business 97.1MB/215MB (45.1%)

2018-03-31T13:26:02.748+1100 [#############...........] MyDB.business 122MB/215MB (56.8%)

2018-03-31T13:26:05.747+1100 [################........] MyDB.business 147MB/215MB (68.6%)

2018-03-31T13:26:08.746+1100 [###################.....] MyDB.business 173MB/215MB (80.5%)

2018-03-31T13:26:11.746+1100 [######################..] MyDB.business 199MB/215MB (92.7%)

2018-03-31T13:26:13.650+1100 [########################] MyDB.business 215MB/215MB (100.0%)

2018-03-31T13:26:13.651+1100 imported 2523932 documents