DIC 587 Project2 Part2 Report

Prepared by:

- **Anuj Rastogi**

UBPerson# 50134324

Email: anujrast@buffalo.edu

- **Nalin Kumar**

UBPerson# 50170479

Email: nalinkum@buffalo.edu

Question1

What is the seat utilization percentage trend with respect to a particular year for the last ten years?

Class – SeatUtilizationYear.java

Jar-suy.jar

Output: Que1o

Objective

In this problem, we tried to visualize the seat utilization percentage for the last 10 years i.e. from 2006-2016 individually. Seat utilization is defined as what percentage of capacity is the enrolment for a particular course. The results of this problem will give us an interesting observation that how the seat utilization percentage is varying since the last 10 years. This observation will be useful for course schedulers so that they may take necessary actions to increase the seat utilization and make some assumptions regarding the trend in change of seat utilization percentage in the past couple of years

Code Implementation

To achieve this objective, we implemented 2 map reduce stages. In the initial stage, we did data cleaning and also tried to eliminate redundant data. For data cleaning, we cleaned up certain columns such as hallName which were possessing certain unidentifiable values. Also, we considered only those rows for which enrolment is lesser than or equal to capacity since our seat utilization percentage change trend would have given incorrect results for the case when enrolment is greater than capacity value. We would also like to highlight that we didn't included the year 2017 since we felt that the data was insufficient for this year and we would not have been able to correctly compare results for this particular year as compared to other years. For eliminating redundant data, we uniquely identified a course in the first map reduce stage using the key attributes as semester, hallname, courseId, coursename, capacity, enrolment and emitted the value 1 corresponding to all such keys. This cleaned up the data in the sense that we will not be taking the contributions from a specific course entry more than once for a particular semester. In the second mapper stage, we emitted year as the key and a string containing space separated capacity and enrolment values as the value. In the second reducer stage, we calculated what percentage of capacity is enrolment for a particular year.

Results

The result file clearly states two columns namely year and respective seat utilization percentage values for all the courses that occurred in that semester corresponding to a particular year. For example, we can use the results to visualize that the seat utilization percentage was maximum in the year 2014 with the value 48.02%. Based on this result, course schedulers and event planners can use these results to take some necessary steps to increase the seat utilization percentage for courses occurring in those specific halls where seat utilization percentage value is lower than some threshold value set by the course schedulers. For example, they might decrease the capacity of such courses for which seat utilization percentage is lesser than 40% for a particular year.

Question2

What is the seat utilization percentage trend with respect to a particular year and hallname for the last three years?

Class – SeatUtilizationHall.java

Jar – suh.jar

Output: Que2o

Objective

In this problem, we tried to visualize the seat utilization percentage for the last 3 years i.e. from 2014-2016 for a particular hall individually. Seat utilization is defined as what percentage of capacity is the enrolment for a particular course. The results of this problem will give us an interesting observation that how the seat utilization percentage is varying for a particular hall since the last 3 years. This observation will be useful for course schedulers so that they may take necessary actions to increase the seat utilization for certain halls and make some assumptions regarding the trend in change of seat utilization percentage in the past couple of years with respect to a particular hall

Code Implementation

To achieve this objective, we implemented 2 map reduce stages. In the initial stage, we did data cleaning and also tried to eliminate redundant data. For data cleaning, we cleaned up certain columns such as hallName which were possessing certain unidentifiable values. Also, we considered only those rows for which enrolment is lesser than or equal to capacity since our seat utilization percentage change trend would have given incorrect results for the case when enrolment is greater than capacity value. We would also like to highlight that we didn't included the year 2017 since we felt that the data was insufficient for this year and we would not have been able to correctly compare results for this particular year as compared to other years. For eliminating redundant data, we uniquely identified a course in the first map reduce stage using the key attributes as semester, hallname, courseId, coursename, capacity, enrolment and emitted the value 1 corresponding to all such keys. This cleaned up the data in the sense that we will not be taking the contributions from a specific course entry more than once for a particular semester. In the second mapper stage, we emitted year and hallname as the key and a string containing space separated capacity and enrolment values as the value. In the second reducer stage, we calculated what percentage of capacity is enrolment for a specific combination of year and hallname.

Results

The result file clearly states three columns namely year, hallname and respective seat utilization percentage values for all the courses that occurred in that semester corresponding to a particular hallname. Based on this result, course schedulers and event planners can use these results to take some necessary steps to increase the seat utilization percentage for courses occurring in those specific halls where seat utilization percentage value is lower than some threshold value set by the course schedulers. For example, they might decrease the capacity of such courses for which seat utilization percentage is lesser than 40% for a particular hall

Question3

What is the seat utilization percentage trend with respect to a particular year, course id and course name for the last three years?

Class-SeatUtilizationCourse.java

Jar-suc.jar

Output: Que3o

Objective

In this problem, we tried to visualize the seat utilization percentage for the last 3 years i.e. from 2014-2016 for a particular course individually. Seat utilization is defined as what percentage of capacity is the enrolment for a particular course. The results of this problem will give us an interesting observation that how the seat utilization percentage is varying for a particular course since the last 3 years. This observation will be useful for course schedulers so that they may take necessary actions to increase the seat utilization for certain courses and make some assumptions regarding the trend in change of seat utilization percentage in the past couple of years with respect to a particular course

Code Implementation

To achieve this objective, we implemented 2 map reduce stages. In the initial stage, we did data cleaning and also tried to eliminate redundant data. For data cleaning, we cleaned up data coming from column such as hallName which were possessing certain unidentifiable values. Also, we considered only those rows for which enrolment is lesser than or equal to capacity since our seat utilization percentage change trend would have given incorrect results for the case when enrolment is greater than capacity value. We would also like to highlight that we didn't included the year 2017 since we felt that the data was insufficient for this year and we would not have been able to correctly compare results for this particular year as compared to other years. For eliminating redundant data, we uniquely identified a course in the first map reduce stage using the key attributes as semester, hallname, courseId, coursename, capacity, enrolment and emitted the value 1 corresponding to all such keys. This cleaned up the data in the sense that we will not be taking the contributions from a specific course entry more than once for a particular semester. In the second mapper stage, we emitted year, courseId and coursename as the key and a string containing space separated capacity and enrolment values as the value. In the second reducer stage, we calculated what percentage of capacity is enrolment for a specific combination of year, courseId and coursename.

Results

The result file clearly states four columns namely year, courseId, coursename and respective seat utilization percentage values for all the courses that occurred in that semester corresponding to a particular combination of courseId and coursename. For example, the seat utilization percentage for

courseId=4875 and coursename=Data Intensive Computing in 2016 was 21.78%. Based on this result, course schedulers and event planners can use these results to take some necessary steps to increase the seat utilization percentage for those courses where seat utilization percentage value is lower than some threshold value set by the course schedulers. For example, they might decrease the capacity of such courses for which seat utilization percentage is lesser than 40% for a particular course

Question4

What is the seat utilization percentage trend with respect to a particular year and time slot for the last three years?

Class-TimeSlotUtilization.java

Jar-tsu.jar

Output: Que4o

Objective

In this problem, we tried to visualize the seat utilization percentage for the last 3 years i.e. from 2014-2016 for a particular time slot individually. Seat utilization is defined as what percentage of capacity is the enrolment for a particular course. Time slot is a specific combination of day and time mentioned in the input csv file. The results of this problem will give us an interesting observation that how the seat utilization percentage is varying for a particular time slot since the last 3 years. This observation will be useful for course schedulers so that they may take necessary actions to increase the seat utilization for certain time slots and make some assumptions regarding the trend in change of seat utilization percentage in the past couple of years with respect to a particular time slot

Code Implementation

To achieve this objective, we implemented 2 map reduce stages. In the initial stage, we did data cleaning and also tried to eliminate redundant data. For data cleaning, we cleaned up data coming from certain columns such as hallName, day and time which were possessing certain unidentifiable values. Also, we considered only those rows for which enrolment is lesser than or equal to capacity since our seat utilization percentage change trend would have given incorrect results for the case when enrolment is greater than capacity value. We would also like to highlight that we didn't included the year 2017 since we felt that the data was insufficient for this year and we would not have been able to correctly compare results for this particular year as compared to other years. For eliminating redundant data, we uniquely identified a course in the first map reduce stage using the key attributes as semester, hallname, courseId, coursename, capacity, enrolment, day, time and emitted the value 1 corresponding to all such keys. This cleaned up the data in the sense that we will not be taking the contributions from a specific course entry more than once for a particular time slot in a particular semester. In the second mapper stage, we emitted year, day and time as the key and a string containing space separated capacity and enrolment values as the value. In the second reducer stage, we calculated what percentage of capacity is enrolment for a specific combination of year, day and time

Results

The result file clearly states four columns namely year, day, time and respective seat utilization percentage values for all the courses that occurred in that semester corresponding to a particular timeslot. For example, we can use the results to visualize that the seat utilization percentage value for time slot MWF 5-5:59 pm in 2016 was 4.24%. Based on this result, course schedulers and event planners can use these results to take some necessary steps to increase the seat utilization percentage for courses occurring in those specific time slots where seat utilization percentage value is lower than some threshold value set by the course schedulers. For example, they might decrease the capacity of such courses for which seat utilization percentage is lesser than 40% for a particular time slot.

Question5

List all classroom with the capacity greater than 200 which are free on 4th July 2016 between 6 and 7 PM?

Class: Que5.java

Jar: q5.jar

Output: Que5o

Reason for Question:

This question is of importance since on July 4th USA has its independence day. So in case of an occasion we wanted to find out a hall with capacity more than 200 which is free on this day. It is a Monday on July 4th. So we will find out the class rooms which are free on July 4th Monday.

Solution and Explanations:

The solution is available in source code "Que5.java". The jar file is "q5.jar".

In the first MR stage we filtered the data and removed the impurities. The focus was to get a data wherein we could have found out when in spring 2016 each hall is busy. Hence, the key was <HallName>_<Day>_<time> and the value was its capacity. If there were multiple entries corresponding to a Hall name at the same time and day, then we tried to find the maximum value of the capacity for each hall according to the data set. By the end of this stage we got each hall with different time engagements and capacity. Also, we removed those halls whose capacity was less than 200.

In the second MR stage once we have the hall names along with timings and Days when they were engaged, we found out if in any key corresponding to a Hall, there was Monday with the time desired as 6 to 7 PM. If so, then we emitted the key as <HallName> and the value as <Capacity>_<Y/N>. Where Y indicates that there is Monday and N indicating that there is no Monday. Post this stage before Reducer2 we get Hall Name clubbed with values specifying if the hall is free or not on Monday. The shuffle stage after Mapper 2 will look something like:

<HallName> [60_Y, 60_Y, 60_N].

Hence, in the reducer we just check the list corresponding to each hall and find if there was N in the list. This indicates that one of the entries have N and therefore the hall is not free. If there was no N then we emitted that Hall along with capacity.

Interpretations:

The result shows the classes which are free on Monday i.e. July 4. From the data we can schedule a meeting or a gathering in any of these halls to celebrate Independence Day.

Question6

List all the classroom with the capacity greater than 60 which are free on either Monday, Wednesday or Friday in spring 2016?

Class: Que4.java

Jar: q6.jar

Output: Que6o

Reason for Question:

Here we wanted to do an interesting scenario supposing if an instructor asks us that he/she wants to schedule a class on either of Monday , Wednesday or Friday between 5:00PM to 6:00 PM, then what all halls can she consider having capacity greater than 60. Hence we tried to find out an answer for this question. It is a complicated question since the task will be to club the data for same halls together and then find when they are free from the clubbed data analysis.

Solution and Explanation:

The solution is available in source code "Que4.java" and the jar name is "q6.jar".

The solution involves 4 MR stages. In the below section we describe what each MR stage does for us:

In the first MR stage we filtered the data and removed the impurities. The focus was to get a data wherein we could have found out when in spring 2016 each hall is busy. Hence, the key was <HallName>_<time>_<Day> and the value was its capacity. If there were multiple entries corresponding to a Hall name at the same time and day, then we tried to find the average value of the capacity for each hall according to the data set. By the end of this stage we got each hall with different time engagements and average capacity.

In the second MR stage, we tried to filter those halls whose capacity was more than 60. Since this was our requirement. Hence, by the end we reached to the halls, along with their engagements, where in capacity was more than 60.

In the third MR Stage, for each hall we tried to find out when it was free from the various entries corresponding to that hall. It is worth noting that the key of our output from the previous results contain Hall name along with its timings when it is engaged. Hence, in this stage from different entries which specified different timings when the hall was busy we tried to find out the timings when the hall was free between 5 to 6 PM. For example if an entry said that the hall "X" was busy at 5 to 6 PM on "M" we can have an output that it is free on "W" and "F". Again if an entry says that the same hall is busy on "MW" we can have an output that it is free on "F". In the next stage we will club this data corresponding to a hall. The clubbed data will help us find the common timing when the hall is free. Output of this stage gives us different times for a hall when it is free but the data is still scattered for a hall which will involve another stage. The output key is <HallName>_<FreeTime> and the value is capacity

In the fourth MR stage, we tried to club the data of a hall. We split the key from <HallName>_<FreeTime> to <HallName> and we ouput the value as <FreeTime>_<Capacity>. Now, it is worth noting that corresponding to a hall, we will get all the data specifying different times, when it is free. This will be input to reducer. Based on what the data says i.e. when the hall is free, we can find out the common timings when the hall is really free. Hence the output key is <HallName>_<TimFree> and value is capacity. "X" in key indicates that it is not free on that particular day. Eg: ABC_XWF means hall is free on W and F but now on M.

Interpretations:

This knowledge is really of a lot of use as through this, we can get halls which are free between 5 to 6 PM and have capacity more than 60. With such a knowledge we can schedule a class demanding capacity more than 60 and wants timings between 5 to 6 PM. In fact the same can be extended to different time slots and days to find out the availability. **The result show the rooms which are free and XWF means they are free on Wednesday and Friday but not Monday. Also, it present the capacity each hall has.**

Question7

Which is the most favorite subject (in terms of no. of students enrolled) in each of the last spring and falls since 2011?

Class: Que6.java (Please do not get confused with the names, they are correct)

Jar: q7.jar

Output: Que7o

Reason for Question:

The question was done to see the trend in the choices of people since, 2011. **The result shows the various subjects in which the enrollment is more than 300 for each semester.** This analysis will help us understand the subjects that are popular in spring and the ones that are popular in fall.

Solution and Explanation:

The source code is "Que6.java" and the jar file is "q7.jar".

The question involves 2 MR Stages. In the first MR Stage, we tried to filter the data and clean it. Here we also filter the data to take the data corresponding to the years between 2011 and 2016. One

thing to understand is the fact that there can be multiple entries for the same subject and under the same subject code. So we want to take only two entries for the course which correspond to two different subject codes, in a semester. Hence, our key is <SemName>_<CourseName>_<CourseId>. If there are multiple entries corresponding to this key then we take that data as redundant and hence, we consider the entry with the maximum enrollment because the enrollment cannot be less than that. This is done in Reducer1. Post this stage we will have only 2 entries for each subject per semester, with 2 different course IDs.

In the second MR Stage, we will have the 2 entries corresponding to a subject in each semester added to give the total enrollment for that subject in that semester. The output key is <SemName>_<CourseName> and the value is the total capacity. The output shows the various subjects in a semester having enrollment more than 300. This data can be visualized to figure out the subjects with the maximum enrollment in each of the semesters since 2011.

Interpretations:

The results from this are interesting in a way that they help us know about the subjects that the students are interested for in either spring or fall. Also, from these results university administrators can find out which way the trend of studying is moving towards and can also decide what classrooms to be allocated to these courses since their capacity is huge.

Question8

Which is the most favorite subject in terms of the number of students enrolled since the last 2 years?

Class: Que61.java

Jar: q8.jar

Output: Que8o

Reason for Question:

UB is a university with a lot of courses and halls. So it may be somewhat hard for the authorities to allocate buildings to classes and schedule them. Hence, we wanted to find the most pupolar subject in the university right now, the one with the maximum enrollment since the last couple of years. This may help authorities to figure out which class to allocate to this subject. **The result in fact contains the enrollment in different subjects whose enrollment exceeds 500**. From here we can see the subject with the maximum enrollment.

Solution and Explanation:

The solution is available in source code "Que61.java". The jar file is "q8.jar".

The source code runs 2 MR stages. In the first MR stage the idea is to filter the data so that entries such as "Arr" or "Unknown" are removed. Also the data corresponding to only the years between 2015 and 2016 should come.

In the first MR stage we tried to get only two entries corresponding to each subject for each semester. This was a challenge as in the dataset provided we had multiple entries under the same subject name and the subject code, for the same Semester. Hence, for one semester we wanted to

take only 2 entries for one subject which correspond to 2 different subject codes. The key was like <SemName>_<CourseName>_<CourseId> and the output value was the enrollment under that subject name and code. If in case there are two keys with the same name, there is a redundancy, so we took the value specifying the maximum enrollment capacity because the enrollment cannot be less than this. This was done in reducer 1. Post this stage we will get only 2 entries for each subject, for one semester, which correspond to different subject codes.

In the second MR stage we tried to sum up the entries with the same subject name. Since, the data provided was only for 2 years hence, if one takes the key as <CourseName> it will accumulate all the entries under one subject corresponding to different subject codes across semesters. Post this in reducer 2 one just needs to sum up all the values for one subject getting the total enrollment for a subject since the last 2 years. We also put a filter to get only those subjects whose enrollment exceeded 500.

Interpretations:

With the knowledge of the popular courses a university can not only schedule them in appropriate places but also in appropriate timings. Also, this helps the university figure out the trend of studies of students and the direction in which the society in the university is heading towards. Also, it helps university to figure out that what it is known for. In the later part of the report we have also found out what is the most favorite slot for the most favorite subject, obtained from this part.

Question9

Track the trend in growth of seats per year in building (KNOX OBRIAN BALDY JACOBS) in the last 4 years?

Class: Que9.java

Jar: q9.jar

Output: Que9o

Reason for question:

We wanted to track the growth in the number of seats in all these halls, since these are some popular halls in North campus. The results from these will help us understand the growth in the university infrastructure which in turn will help us understand the performance of the university and hence, the likeness of the university among students. For example, if the capacity increases, that means that, more people are getting enrolled in university showing increasing popularity. **The results obtained show the capacities held by the halls since the last 4 years.** These can be analyzed to see if there is a rising or a declining or a steady trend in growth.

Solution and Explanation:

The source code for the question is in "Que9.java". The jar file is "q9.jar".

In the first MR stage we filter the data and purify it. Post this, we check if the data belongs to only 4 years i.e. between "2013" and "2016". Once purified we take the key as <SemName>_<HallName>. The key here is that each hall will be listed many times in a semester. Therefore, we want to take

that entry wherein the capacity of the hall is more than the rest. This is so because, we cannot have a capacity less than that value. Hence, the output of the first stage is <SemName>_<HallName> key and the capacity corresponding to each hall in that semester.

In the second MR Stage, we calculate the cumulative capacity of the hall combining all the classes in a hall and adding them to get the total capacity of the aforementioned halls. Eg: if Knox has three halls then the total capacity is the sum of those three halls. Hence the output is something like <SemName>_<Hall> and the value is capacity.

In the third MR stage we need to compute the total capacity of each hall for the entire year. Hence we want to sum up the data corresponding to a hall from fall, spring and summer semester. This will find the total capacity served by a particular hall throughout the year. Hence the output key is <Year>_<Hall> and the value is capacity. From this data we can visualize the growth in capacity served by these halls since the last 4 years.

Interpretations:

From the results we can summarize if the university infrastructure is expanding which is a clear indication of the increase in University likeness among the students and a symbol of its popularity growth.

Question10

Which hall has held the maximum capacity since the last three years?

Class-MaxHallCapacity.java

Jar-mhc.jar

Output: Que10o

Objective

In this problem, we tried to visualize the maximum hall capacity for the last 3 years i.e. from 2014-2016 individually. For this, our output file contained the results for halls and their respective total capacities in specific years. The results of this problem will give us an interesting observation that which hall has served the maximum capacity since the last 3 years. This observation will be useful for course schedulers so that they may take necessary actions to avoid scheduling courses in those halls which are already over-utilized in terms of total capacity and devote more proportion of students to other halls which have available space and are under-utilized in terms of total capacity. Course schedulers and event planners can also use these results to schedule courses and events such as exams in such halls and can also visualize trends related to changes in utilization of these halls in terms of total course capacity scheduled over the last three years.

Code Implementation

To achieve this objective, we implemented 2 map reduce stages. In the initial stage, we did data cleaning and also tried to eliminate redundant data. For data cleaning, we cleaned up data coming from certain column such as hallName which were possessing certain unidentifiable values. We

would also like to highlight that we didn't included the year 2017 since we felt that the data was insufficient for this year and we would not have been able to correctly compare results for this particular year as compared to other years. For eliminating redundant data, we uniquely identified a course in the first map reduce stage using the key attributes as semester, hallname, courseId, coursename, capacity, enrolment and emitted the value 1 corresponding to all such keys. This cleaned up the data in the sense that we will not be taking the contributions from a specific course entry more than once in a particular semester. In the second mapper stage, we emitted year and hallname as the key and course capacity as the value. In the second reducer stage, we summed up the values of capacities for all the courses which belonged to a specific combination of year and hallname and emitted the final results

Results

The result file clearly states three columns namely year, hallname and capacity values. For example, we can use the results to visualize that the Obrian hall has the maximum capacity of 2658 in 2014. Based on this result, course schedulers and event planners can use these results to take some necessary steps to take advantage of the large capacity of Obrian hall. That is, if any new course is introduced in the future, then they should try first to accommodate that newly introduced course in Obrian.

Question11

Which is the busiest time slot for a hall with the maximum capacity in previous question in the last three years?

Class-BusiestTimings.java

Jar-bt.jar

Output: Que11o

Objective

In this problem, we tried to visualize the busiest time slot for a hall with the maximum capacity computed in previous problem since the last 3 years i.e. from 2014-2016 individually. The results of this problem will give us an interesting observation that which is the busiest time slot in a hall which has served the maximum capacity since the last 3 years. This observation will be useful for course schedulers so that they may take necessary actions to refrain from scheduling courses in those time slots in the busiest hall in terms of total course capacity and instead use other time slots which are under-utilized in terms of total number of courses scheduled in those time slots for a particular year and hall possessing maximum capacity. Course schedulers and event planners can also use these results to schedule courses and events such as exams in such halls and can also visualize trends related to changes in utilization of time slots in hall with the maximum capacity over the last three years.

Code Implementation

To achieve this objective, we implemented 2 map reduce stages. In the initial stage, we did data cleaning and also tried to eliminate redundant data. For data cleaning, we cleaned up data coming from certain columns such as hallName, day and time which were possessing certain unidentifiable values. We would also like to highlight that we didn't included the year 2017 since we felt that the data was insufficient for this year and we would not have been able to correctly compare results for this particular year as compared to other years. For eliminating redundant data, we uniquely identified a course in the first map reduce stage using the key attributes as semester, hallname, courseId, coursename, capacity, enrolment, day, time and emitted the value 1 corresponding to all such keys. This cleaned up the data in the sense that we will not be taking the contributions from a specific course entry more than once for a particular time slot in a particular semester. In the second mapper stage, we emitted year, hallname, day and time as the key and 1 as the value. In the second reducer stage, we summed up the contributions from all the courses with respect to a particular combination of year, hallname, day and time which is our key in this case.

Results

We did our analysis based on the results of previous problem. The hall with the maximum capacity was Obrian based on the results of the previous problem. Henceforth, we evaluated the busiest timing only for Obrian in this problem. The result file clearly states four columns namely year, hallname, day and time. For example, we can use the results to visualize that only 2 courses are scheduled in the time slot MWF 5:00pm in Obrian hall in 2015. Based on this result, course schedulers and event planners can use these results to take some necessary steps to take advantage of the availability of this particular timeslot. That is, if any new course is introduced in the future, then they should try first to accommodate that newly introduced course in Obrian hall in the timeslot with minimum number of courses scheduled.

Question12

Which is the busiest day for a particular hall in a specific year in the last three years?

Class-BusiestDay.java

Jar-bd.jar

Output: Que12o

Objective

In this problem, we tried to visualize the busiest day for a hall since the last 3 years i.e. from 2014-2016 individually. Busiest day is computed in terms of total number of unique course entries for a particular day and hallname in a specific year. The results of this problem will give us an interesting observation that which has been the busiest day in a hall since the last 3 years. This observation will be useful for course schedulers so that they may take necessary actions to refrain from scheduling courses in the busiest days in a particular hall and instead use other days to schedule courses which are under-utilized in terms of total number of unique course entries for a particular hall and year. Course schedulers and event planners can also use these results to schedule courses and events such as exams and can also visualize trends related to changes in utilization of certain days for a particular hall over the last three years.

Code Implementation

To achieve this objective, we implemented 2 map reduce stages. In the initial stage, we did data cleaning and also tried to eliminate redundant data. For data cleaning, we cleaned up data coming from certain columns such as hallName and day which were possessing certain unidentifiable values. We would also like to highlight that we didn't included the year 2017 since we felt that the data was insufficient for this year and we would not have been able to correctly compare results for this particular year as compared to other years. For eliminating redundant data, we uniquely identified a course scheduled on a particular day in the first map reduce stage using the key attributes as semester, hallname, courseId, coursename, day and emitted the value 1 corresponding to all such keys. This cleaned up the data in the sense that we will not be taking the contributions from a specific course entry more than once for a particular day in a particular semester. In the second mapper stage, we emitted year, hallname and day as the key and 1 as the value. In the second reducer stage, we summed up the contributions from all the courses with respect to a particular combination of year, hallname and day which is our key in this case.

Results

The result file clearly states four columns namely year, hallname, day and number of courses scheduled on that particular day. For example, we can use the results to visualize that maximum number of courses were scheduled on Mondays and Thursdays in Obrian hall in 2014. Based on this result, course schedulers and event planners can use these results to refrain from scheduling any new courses in Obrian hall on Monday and Thursday in the first place.

Question13

How many courses are scheduled for a particular hall in a specific year in the last three years?

Class-CoursesScheduled.java

Jar-cs.jar

Output: Que13o

Objective

In this problem, we tried to visualize how many courses are scheduled for a particular hall in a specific year since the last 3 years i.e. from 2014-2016 individually. The results of this problem will give us an interesting observation that which has been the busiest hall in terms of toal number of courses scheduled since the last 3 years. This observation will be useful for course schedulers so that they may take necessary actions to refrain from scheduling courses in the busiest halls and instead use other halls to schedule courses which have available space and are under-utilized in terms of total number of courses scheduled for a particular year. Course schedulers and event planners can also use these results to schedule courses and events such as exams in such halls and can also visualize trends related to changes in utilization of these halls in terms of total number of courses scheduled over the last three years.

Code Implementation

To achieve this objective, we implemented 2 map reduce stages. In the initial stage, we did data cleaning and also tried to eliminate redundant data. For data cleaning, we cleaned up data coming from certain column such as hallName which were possessing certain unidentifiable values. We would also like to highlight that we didn't included the year 2017 since we felt that the data was insufficient for this year and we would not have been able to correctly compare results for this particular year as compared to other years. For eliminating redundant data, we uniquely identified a course entry in the first map reduce stage using the key attributes as semester, hallname, courseId, coursename, capacity, enrolment and emitted the value 1 corresponding to all such keys. This cleaned up the data in the sense that we will not be taking the contributions from a specific course entry more than once in a particular semester. In the second mapper stage, we emitted year and hallname as the key and 1 as the value. In the second reducer stage, we summed up the contributions from all the courses with respect to a particular combination of year and hallname which is our key in this case.

Results

The result file clearly states three columns namely year, hallname and number of courses scheduled in that particular hall in that year. For example, we can use the results to visualize that maximum number of courses were scheduled in Baldy in 2015 and the number is 1156. Based on this result, course schedulers and event planners can use these results to refrain from scheduling any new courses in Baldy in the first place.

Question14

What is the increase in building space over the last two years?

Class-BuildingSpaceIncrease.java

Jar-bsi.jar

Output: Que14o

Objective

In this problem, we tried to visualize what is the increase in building space over the last 3 years i.e. from 2014-2016 individually. The results of this problem will give us an interesting observation related to what is the actual trend in space increase for different buildings in specific year range and how can it be correlated to course enrolment increase over the same year range. We can extend this problem to part3 of Project2. This observation will be useful for course schedulers so that they may take necessary actions to refrain from scheduling courses in those buildings where space has not increased significantly over the past couple of years and instead use other halls to schedule courses which have available space and larger increase in building space. However, course enrolment increase in those buildings should also be taken into consideration before scheduling any kind of activities. Course schedulers and event planners can also use these results to schedule courses and events such as exams in such halls and can also visualize trends related to changes in building space increase over the last two years.

Code Implementation

To achieve this objective, we implemented 3 map reduce stages. In the initial stage, we did data cleaning and also tried to eliminate redundant data. For data cleaning, we cleaned up data coming from certain column such as hallName which were possessing certain unidentifiable values. We would also like to highlight that we didn't included the year 2017 since we felt that the data was insufficient for this year and we would not have been able to correctly compare results for this particular year as compared to other years. For eliminating redundant data, we uniquely identified a course entry in the first map reduce stage using the key attributes as semester, hallname, courseId, coursename, capacity, enrolment and emitted the value 1 corresponding to all such keys. This cleaned up the data in the sense that we will not be taking the contributions from a specific course entry more than once in a particular semester. In the second mapper stage, we emitted year and hallname as the key and course capacity as the value. In the second reducer stage, we summed up the contributions of course capacities from all the courses with respect to a particular combination of year and hallname which is our key in this case. In the third mapper stage, we constructed two keys for year range based on year value coming from input and emitted two key value pairs for a single year entry i.e. two year ranges concatenated with year and hallname as the key and course capacity as the value. In the final reducer stage, we emitted the year ranges and their respective building space increase values for a particular year and hallname

Results

The result file clearly states three columns namely year range, hallname and total course capacity increase corresponding to that particular hall in the specified year range. Based on this result, course schedulers and event planners can use these results to focus on scheduling new courses in those buildings/halls where course capacity has increased to the maximum extent in the last 2 years.

Question15

What is the increase in course enrolment over the last two years?

Class-CourseEnrollmentIncrease.java

Jar-cei.jar

Output: Que15o

Objective

In this problem, we tried to visualize what is the increase in course enrolment over the last 3 years i.e. from 2014-2016 individually. The results of this problem will give us an interesting observation related to what is the actual trend in enrolment increase for different courses in specific year range and how can it be correlated to course capacity increase over the same year range. We can extend this problem to part3 of Project2. This observation will be useful for course schedulers so that they may take necessary actions to refrain from scheduling courses with lesser capacities vis a vis course enrolment increase. That is, if there is an enrolment increase in a specific course and if that course was already full, then there should be an increase in course capacity corresponding to an increase in enrolment for that course in a specific year range. Course schedulers can also visualize trends related to changes in course enrolment increase over the last two years and judge based on the

results which course has recently attracted a lot of attention from the students. If the course is in heavy demand, then the event planners can also use this information to schedule certain events such as exams and recitations since a bigger space needs to be allocated to such courses as compared to the previous years

Code Implementation

To achieve this objective, we implemented 3 map reduce stages. In the initial stage, we did data cleaning and also tried to eliminate redundant data. For data cleaning, we cleaned up data coming from certain column such as hallName which were possessing certain unidentifiable values. We would also like to highlight that we didn't included the year 2017 since we felt that the data was insufficient for this year and we would not have been able to correctly compare results for this particular year as compared to other years. For eliminating redundant data, we uniquely identified a course entry in the first map reduce stage using the key attributes as semester, hallname, courseId, coursename, capacity, enrolment and emitted the value 1 corresponding to all such keys. This cleaned up the data in the sense that we will not be taking the contributions from a specific course entry more than once in a particular semester. In the second mapper stage, we emitted year, courseId and coursename as the key and course enrolment as the value. In the second reducer stage, we summed up the contributions of course enrolments from all the courses with respect to a particular combination of year, courseId and coursename which is our key in this case. In the third mapper stage, we constructed two keys for year range based on year value coming from input and emitted two key value pairs for a single year entry i.e. two year ranges concatenated with year, courseId and coursename as the key and course enrolment as the value. In the final reducer stage, we emitted the year ranges and their respective course enrolment increase values for a particular year, courseId and coursename

Results

The result file clearly states four columns namely year range, courseId, coursename and total course enrolment increase corresponding to that particular course in the specified year range. Based on this result, course schedulers and event planners can use these results to focus on increasing the capacity of such courses if the capacity is not enough for those courses or they can focus on accommodating such courses in halls of larger capacity in the future after looking at the current trend of course enrolment increase.

Question16

What is the increase in course capacity over the last two years?

Class-CourseCapacityIncrease.java

Jar-cci.jar

Output: Que16o

Objective

In this problem, we tried to visualize what is the increase in course capacity over the last 2 years i.e. from 2014-2016 individually. The results of this problem will give us an interesting observation related to what is the actual trend in capacity increase for different courses in specific year range

and how can it be correlated to course enrolment increase over the same year range which is computed in previous problem. We can extend this analysis to part3 of Project2. This observation will be useful for course schedulers so that they may take necessary actions to refrain from scheduling courses with lesser capacities vis a vis course enrolment increase. That is, if there is an enrolment increase in a specific course and if that course was already full, then there should be an increase in course capacity corresponding to an increase in enrolment for that course in a specific year range. Course schedulers can also visualize trends related to changes in course capacity increase over the last two years and judge based on the results which course has recently attracted a lot of attention from the students. If the course is in heavy demand, then the event planners can also use this information to schedule certain events such as exams and recitations since a bigger space needs to be allocated to such courses as compared to the previous years

Code Implementation

To achieve this objective, we implemented 3 map reduce stages. In the initial stage, we did data cleaning and also tried to eliminate redundant data. For data cleaning, we cleaned up data coming from certain column such as hallName which were possessing certain unidentifiable values. We would also like to highlight that we didn't included the year 2017 since we felt that the data was insufficient for this year and we would not have been able to correctly compare results for this particular year as compared to other years. For eliminating redundant data, we uniquely identified a course entry in the first map reduce stage using the key attributes as semester, hallname, courseId, coursename, capacity, enrolment and emitted the value 1 corresponding to all such keys. This cleaned up the data in the sense that we will not be taking the contributions from a specific course entry more than once in a particular semester. In the second mapper stage, we emitted year, courseId and coursename as the key and course capacity as the value. In the second reducer stage, we summed up the contributions of course capacities from all the courses with respect to a particular combination of year, courseId and coursename which is our key in this case. In the third mapper stage, we constructed two keys for year range based on year value coming from input and emitted two key value pairs for a single year entry i.e. two year ranges concatenated with year, courseId and coursename as the key and course capacity as the value. In the final reducer stage, we emitted the year ranges and their respective course capacity increase values for a particular year, courseId and coursename

Results

The result file clearly states four columns namely year range, courseId, coursename and total course capacity increase corresponding to that particular course in the specified year range. Based on this result, course schedulers and event planners can use these results to focus on scheduling courses whose capacity increase has been the maximum in those halls which are having sufficient capacity to serve these courses

Question17

Since last 2 years in the best course (the one with maximum enrolment in last 2 years) what are the preferred time slots?

Class: **Que15.java**

Jar: **q17.java**

Output: Que17o

Reason for Question:

In Question 8, we found out the favourite subject in the last 2 years. Here we want to figure out corresponding to this course what is the preferred time. It is a fairly straight forward question. With this analysis we can schedule the favourite subject in a time which is preferred by the students. **The result present different times in which the subject was scheduled since the last 2 years and the enrolments in each time.**

Solution and Explanation:

The source code for the question is "Que15.java". The jar file is "q17.java".

This question involves only 1 MR Stage. Here we just make a key <CourseName>_<Timings> and corresponding to this we count the enrolment. The timings with the maximum enrolment is the most preferred time. We found with the analysis of Question 8 that General Chemistry had the maximum enrolment since the last 2 years. So the results here are for General Chemistry.

Interpretations:

The analysis of these results can help us figure out the best time to schedule the most demanded class and hence can provide students an opportunity to study according to their willingness.

Question18

Increase in the Data Intensive Computing Enrollment since 2002?

Class: Que18.java

Jar: q18.jar

Output: Que18o

Reason for Question:

There was an anxiety in us, while viewing the data of the course enrollment, to see how much data science course and especially the one which we are taking has gained in popularity. This was the motivation behind this question.

Solution and Explanation:

The solution for the same is available in "Que18.java" source code. The jar name is "q18.jar". The source code runs 2 MR stages. In the first MR stage the idea is to filter the data so that entries such as "Arr" or "Unknown" are removed. Also the data corresponding to only the year's post 2002 should come.

In the first MR stage we tried to get only two entries corresponding to Data Intensive Computing. This was a challenge as in the dataset provided we had multiple entries for the same subject name and the subject code, for the same Semester. Hence, for one semester we wanted to take only 2 entries for one subject which correspond to 2 different subject codes. The key was like <SemName>_<CourseName>_<CourseId> and the output value was the enrolment in that subject name and code. If in case there are two keys with the same key name, there is a redundancy, so we took the value specifying the maximum enrolment capacity because the enrolment cannot be less

than this. This was done in reducer 1. Post this stage we will get only 2 entries for the subject Data Intensive Computing which correspond to different subject codes.

In the second MR stage, we tried to sum 2 entries of the course corresponding to two different subject codes in a semester and hence, find the total enrolment for one semester in Data Intensive Computing for one semester. The mapper key here was <SemName>_<SubjectName>. The final value was the total enrolment for data intensive computing in each semester since 2002.

The output is provided with the submission.

Interpretations:

The output result show that, since 2014 the numbers in the course have almost doubled with the total enrolment exceeding 100, and in fact the real figure is 164. Also, this trend is continuous now and shows the current popularity of the course. In the output file this can be seen. The trend shows a growth in the inclination towards data science courses and hence, a sign for the university authorities to introduce more data related courses and also increase and schedule data science courses in halls with large capacities.

Question19

What has been the increase in the courses since 1993?

Class: SubjectOffer.java

Jar: q19.jar

Output: Que19o

Reason for Question:

We wanted to see how the institution has grown since the past 20-25 years. **The result gives the number of subjects offered in each year** and seeing this we can see if they have increased or decreased. Such analysis can help us do future predictions and understand the growth curves and popularity rise in the university.

Solution and Explanation:

The input source code is "SubjectOffer.java" and the jar file is "q19.jar".

The question involves 2 MR stages. In the first MR stage we try to purify the data and remove the redundancy by making the key as <SemName_CourseName>. If there is a repetition in the subject within a semester, then in the reducer we neglect all those and output only one value i.e. 1. Hence, the output of this stage gives <SemName_CourseName> as key and value (1) corresponding to each course per semester.

In the second MR Stage, we calculate the number of subjects offered in a year. Hence, we combine the data by making only the <Year> as key and counting the number of subjects offered in that year in the Reducer2.

Interpretations:

These results give an insight on the growth of university as a whole. The increased number of subjects project the rate at which the university has grown and also project the popularity of university among the students.

Question20

Which is the busiest time slot in spring 2016 in terms of the course offered at that period?

Class: Que20.java

Jar: q20.jar

Output: Que20o

Reason for Question:

The question focusses on finding out the busiest time schedule in the university. Such a time schedule can tell the restaurant owners on when the maximum students will be in the college premises. This can help them understand the flow of rush. Also, to instructors who want to schedule their courses this may give information as to when NOT to schedule class.

Solution and Explanation:

In the first MR stage, we clean the data and select only those rows from the dataset where the semester is spring 2016. Also, we want to avoid those redundant set where the same class is listed a couple of times in the same time slot. Hence, the key is <SemName>_<CourseName>_<Timings>_<WeekDays>. If we get a redundant data we just negelect it in Reducer1. Corresponding to each key we take only 1 value.

In the second MR Stage corresponding to each <Time>_<Day> we count the number of entries. Hence, we get the total number of schedules or classes in one time and on one day.

In the third MR stage, we deal with an interesting scenario. In case of the above output we will get time along with Days. But the day format will be like “MWF” or “M-F”. If I want to find out the

number of classes scheduled on Friday between 12 to 1, then in that case “M-F” should also be included and “MWF” should also, be included. Therefore, the third MR stage is dedicated to combining the data in different formats to give separate outputs. Eg: “M-F” is separated as M,T,W,R,F and hence, the results corresponding to all Mondays and all other days will be combined together.

The output displays the <time>_<day> and the number of classes occurring in that time.

Interpretations:

From the analysis of the results, we can find the busiest time slots in the campus in a week and the day when they occur. They can help achieve the objective of this question.

WORK DISTRIBUTION IN TEAMMATES:

50% Anuj Rastogi

50% Nalin Kumar