ADVANCED DATABASE
Individual Journal

Name: Amoga Varsha
Day 1
Date and Time: 29th March 11:00 am to 4:pm

Today we started our project. We came up with a plan to work with CSV files. Our database would include the user's queries, such as create table <table_name> and the column headings, which would then create the table with the respective details. Each CSV file would be only a table of data.

Personally, I implemented the create table functionality. This function works with the tokens that were split by Kushal tokenizing function to understand the action the user wants to perform. Since we are performing something close to a query, we have to tokenize. This is how the query looks like create table <table_name> (id, name, age). This function will create a new csv file that is stored on the disk, which is a representation of the table format we are going for. Since the query is also taking the column heading, we know what kind of data the CSV file requires. I also implemented the check counter function, which understands what action the user is taking, such as create or insert. Which mentions if there is an error or incorrect format in the query.

Day 2
Date and Time : 5th April 9:00 am to 2:00 pm

In todays work , my role was to work on the update query, which I was partially successful in. This function would take in a query such as " update <csv_table_name> <column_name> <new_value> where <condition_column> = <condition_value>" and would perform the necessary changes in the main dbms file. The issue arose when we were trying to reflect those changes in the index files. Kushal was working on the delete function and had a similar issue. But we were done with all the crud operations in the main file and storage. The only issue is indexing.

Day 3
Date and Time: 11th April, 12:00 pm to 6:00 pm

By now, we had a conversation with our professor and realised that creating index files for each column is a complicated and memory heavy task. Initially, we were planning to index files and then do another form of indexing using B trees to sort them, which was unnecessary. We changed our approach and started a new code that first only works with B trees. I was in charge of

creating B trees for each column and inserting values into them. Which we were successfully able to do. I also figured out how to export the data into a csv file.

Day 4
Date and Time : April 5th, 9:00 am to 2:00 pm

Today we were finishing up by re implementing our initial query inputs from the user. As I had worked with create table and update records in the initial code, I worked on the same to work with our new b tree structured code. I also made a new query type for the export data in csv. Just to reiterate all the formats.

- Format 1: create table <table_name> <column_headings>"
- "Format 3: update <csv_table_name> <column_name> <new_value> where <condition_column> = <condition_value>"
- "Format 6: export to <csv_file_name> from <csv_table_name>"

The update function takes in the arguments of which column to change and what change to make based on what criteria. The create table creates a b tree with the columns mentioned with the column name which will be used in the future for further use.

Additionally, we developed a display function to showcase all changes made within the system. Furthermore, we added functionality to print column headings every time, eliminating the need to repeatedly enter ".schema.".