Parallel Sort and Parallel Join

Submit Assignment

Due Apr 13 by 11:59pm   **Points** 20
Submitting   a text entry box or a file upload
Available   Mar 30 at 12am - Apr 13 at 11:59pm 15 days

The required task is to build a generic parallel sort and parallel join algorithm.

1. Implement a Python function **ParallelSort()** that takes as input: (1) **InputTable** stored in a PostgreSQL database, (2) **SortingColumnName** the name of the column used to order the tuples by. **ParallelSort()** then sorts all tuples (using five parallelized threads) and stores the sorted tuples for in a table named **OutputTable** (the output table name is passed to the function). The **OutputTable** contains all the tuple present in **InputTable** sorted in ascending order.

For more information on Psycopg, click here.   **(http://initd.org/psycopg/docs/)**

Function Interface:

ParallelSort (InputTable, SortingColumnName, OutputTable, openconnection)
InputTable – Name of the table on which sorting needs to be done.
SortingColumnName – Name of the column on which sorting needs to be done, would be either of type integer or real or float. Basically Numeric format. Will be Sorted in Ascending order.

OutputTable – Name of the table where the output needs to be stored.

openconnection – connection to the database.

2. Implement a Python function **ParallelJoin()** that takes as input: (1) **InputTable1** and **InputTable2** table stored in a PostgreSQL database, (2) **Table1JoinColumn** and **Table2JoinColumn** that represent the join key in each input table respectively. **ParallelJoin()** then joins both **InputTable1** and **InputTable2** (using five parallelized threads) and stores the resulting joined tuples in a table named **OutputTable** (the output

table name is passed to the function). The schema of **OutputTable** should be similar to the schema of both **InputTable1** and **InputTable2** combined.

Function Interface:

ParallelJoin (InputTable1, InputTable2, Table1JoinColumn, Table2JoinColumn, OutputTable, openconnection)

InputTable1 – Name of the first table on which you need to perform join.

InputTable2 – Name of the second table on which you need to perform join.
Table1JoinColumn – Name of the column from first table i.e. join key for first table.
Table2JoinColumn – Name of the column from second table i.e. join key for second table.

OutputTable - Name of the table where the output needs to be stored.

openconnection – connection to the database.

Please use the function signature exactly same as mentioned in Assignment3_Interface.py. You will notice that in top of Assignment3_Interface.py, following things are declared

Example

Let's use an example to understand how those functions are called.

FIRST_TABLE_NAME = 'table1'

SECOND_TABLE_NAME = 'table2'

SORT_COLUMN_NAME_FIRST_TABLE = 'column1'
SORT_COLUMN_NAME_SECOND_TABLE = 'column2'
JOIN_COLUMN_NAME_FIRST_TABLE = 'column1'
JOIN_COLUMN_NAME_SECOND_TABLE = 'column2'

Once your implementation is complete, you will have to create two tables in database manually. Name them *MovieRating* and *MovieBoxOfficeCollection*. Suppose, you want to sort MovieRating by column Rating and MovieBoxOfficeCollection by column Collection. You also want to join MovieRating and MovieBoxOfficeCollection by column MovieID. Then, you would define the variables mentioned above as : FIRST_TABLE_NAME = 'MovieRating'

SECOND_TABLE_NAME = 'MovieBoxOfficeCollection'
SORT_COLUMN_NAME_FIRST_TABLE = 'Rating'
SORT_COLUMN_NAME_SECOND_TABLE = 'Collection'
JOIN_COLUMN_NAME_FIRST_TABLE = 'MovieID'
JOIN_COLUMN_NAME_SECOND_TABLE = 'MovieID'

Naming Convention to be followed strictly:

Database name - dds_assignment

Postgres User name - postgres

Postgres password - 1234

Instructions on how this will be tested:

Please follow these instructions closely.

1. Two tables will be created in the database manually.

2. The created tables will contain at least an integer field, which would be used for both

Parallel Sorting and Parallel Joining.

3. Then, the ParallelSort() and ParallelJoin() Function will be called to check the correctness of the program.

4. Your code should use **5 threads** for both ParallelSort() as well as ParallelJoin().

5. Your code should be able to handle table irrespective of its schema.

6. Do not make your code dependent on any particular table; it should be able to work on any table.

Assignment Tips!

Please follow these instructions closely else marks will be deducted.

1. Please follow the function signature as provided in the Assignment3_Interfacy.py.

2. Please use the same database name, table name, user name and password as provided in the assignment to keep it consistent.

3. Please make sure to run the file before submitting and make sure there is no indentation error. In case of any compilation error, 0 marks will be given.

4. Do not modify any function signature in Assignment3_Interface.py. In case any modification is needed, please post the same on discussion board.

Submission Instructions:

Upload the Assignment3_Interface.py file. **Do not upload *.zip files**.

Note: -

Failure to follow the instructions provided in the document will result in the loss of the points.

We provide a virtual machine that has the testing environment and an installed PostgreSQL.

OS username: user

OS password: user

Postgres username: postgres

Posgres password: 1234

You can download it and use any VM software such as **VirtualBox (https://www.virtualbox.org/)** to import it:

https://drive.google.com/file/d/1EBlmGZmqDQ8XGTuiHPoqP7XE2ZykAXkX/view? usp=sharing
(https://drive.google.com/file/d/1EBlmGZmqDQ8XGTuiHPoqP7XE2ZykAXkX/view)

You will need the following files in your assignment.

1. Assignment3_Interface.py: You need to implement your code inside this file.

- **Assignment3_Interface.py** ↓ **(https://canvas.asu.edu/courses/75440/files/26129891/download?download_frd=1)**

2. tester.py: You can use this tester to test your code.

- **tester.py** ↓ **(https://canvas.asu.edu/courses/75440/files/26129890/download?download_frd=1)**

3. movies.txt and ratings.txt: some test data

- **movies.txt** ↓ **(https://canvas.asu.edu/courses/75440/files/26129935/download?download_frd=1)**
- **ratings.txt** ↓ **(https://canvas.asu.edu/courses/75440/files/26129931/download?download_frd=1)**