

**Experiment No.1**

**Title:** Execution of Parallel Database queries.

**Batch:A2** **Roll No.:16010421073 Experiment No.: 1**

**Aim: To execute Parallel Database queries.**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Resources needed:** PostgreSQL 9.3

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

**Theory**

A parallel database system seeks to improve performance through parallelization of various

operations, such as loading data, building indexes and evaluating queries. Although data may

be stored in a distributed fashion, the distribution is governed solely by performance

considerations. Parallel databases improve processing and input/output speeds by using

multiple CPUs and disks in parallel. Centralized and client–server database systems are not

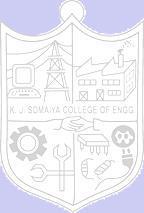
powerful enough to handle such applications. In parallel processing, many operations are

performed simultaneously, as opposed to serial processing, in which the computational steps

are performed sequentially.

Types of parallelism :

• Interquery parallelism: Execution of multiple queries in parallel



* Interoperation parallelism: Execution of single queries that may consist of more than one operations to be performed.
  + Independent Parallelism - Execution of each operation individually in different

processors only if they can be executed independent of each other. For example, if we need to join four tables, then two can be joined at one

processor and the other two can be joined at another processor. Final join can be done later.

* + Pipe-lined parallelism - Execution of different operations in pipe-lined

fashion. For example, if we need to join three tables, one processor may join two tables and send the result set records as and when they are produced to the

other processor. In the other processor the third table can be joined with the incoming records and the final result can be produced.

* Intraoperation parallelism Execution of single complex or large operations in parallel in

multiple processors. For example, ORDER BY clause of a query that tries to execute on millions of records can be parallelized on multiple processors.

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

**Procedure:**

**Parallel queries provide parallel execution of sequential scans, joins, and aggregates etc.**

Parallel queries provide parallel execution of sequential scans, joins, and aggregates. To make the performance gains need a lot of data

create table ledger (

id serial primary key,

date date not null,

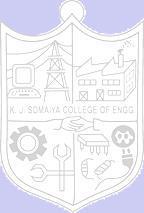
amount decimal(12,2) not null

);

insert into ledger (date, amount)

select current\_date - (random() \* 3650)::integer, (random() \* 1000000)::decimal(12,2) - 50000 from generate\_series(1,50000000);

**explain analyze select sum(amount) from ledger;**



Reading the output, we can see that Postgres has chosen to run this query sequentially.

Parallel queries are not enabled by default. To turn them on, we need to increase a config param called max\_parallel\_workers\_per\_gather.

**show max\_parallel\_workers\_per\_gather;**

Let’s raise it to four, which happens to be the number of cores on this workstation.

**set max\_parallel\_workers\_per\_gather to 4;**

Explaining the query again, we can see that Postgres is now choosing a parallel query. And it’s about four times faster.

**explain analyze select sum(amount) from ledger;**

**The planner does not always consider a parallel sequential scan to be the best option. If a query is not selective enough and there are many tuples to transfer from worker to worker, it may prefer a “classic” sequential scan.PostgreSQL optimises the number of workers according to size of the table and the min\_parallel\_relation\_size.**

Similar ways we can execute join operation and check parallel execution of sequential join.

**explain analyse select library1.id,library1.quantity,library2.location from library2,library1 where library1.id=library2.id;**

**SET max\_parallel\_workers\_per\_gather TO 3;**

**explain analyse select library1.id,library1.quantity,library2.location from library2,library1 where library1.id=library2.id;**

Page No:

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

**Questions:**

1. **Explain the parallelism achieved in the experiment you performed.**

**Ans:**

* By breaking down each query into its component components that operate in parallel, parallelism in a query enables us to execute numerous inquiries simultaneously.
* Shared-nothing architecture can be used to achieve this.
* As more and more resources, such processors and storage, are made available, parallelism is also used to speed up the process of a query execution.
* We can achieve parallelism in a query by the following methods :

I/O parallelism

Intra-query parallelism

Inter-query parallelism

Intra-operation parallelism

Inter-operation parallelism

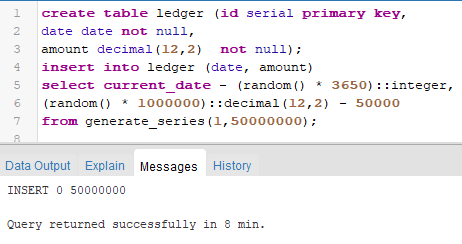
1. **With comparison of the results explain how degree of parallelism ( no of parallel processors) affect the operation conducted.**

**Ans:**

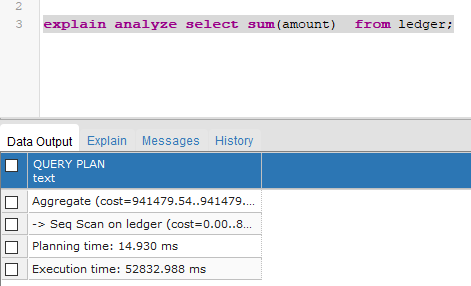
* The degree of parallelism (DOP) is a metric which indicates how many operations <- "can be" or -> are being simultaneously executed -> by a computer.not processor.
* A program running on a parallel computer may utilize different numbers of processors at different times.
* As we used **Setting max\_parallel\_workers\_per\_gather to 4 the** query **Analyzing time taken for select sum taken less time to execute in comparison to executing select sum beforemax parallel workers.**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Results: (Program printout with output)**

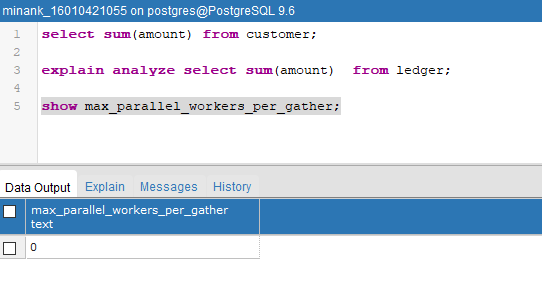
**Creating table and insertion:**

****

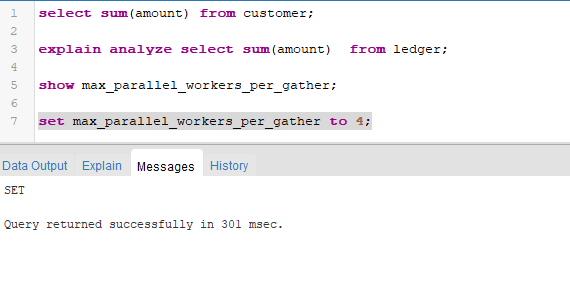
**Analyzing time taken for select sum:**

****

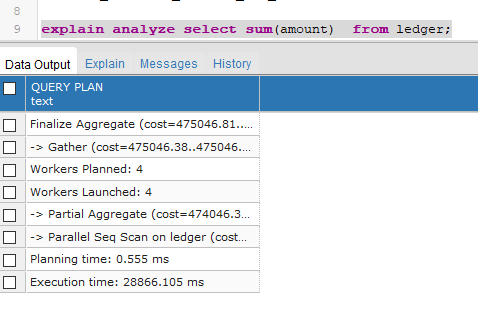
**Checking the max\_parallel\_workers\_per\_gather:**

****

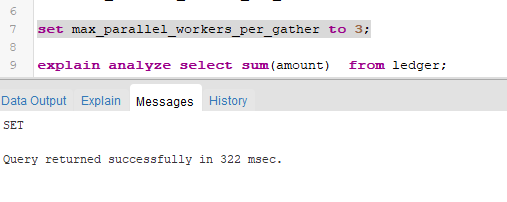
**Setting max\_parallel\_workers\_per\_gather to 4**

****

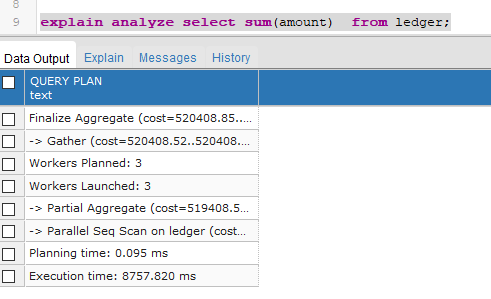
**Analyzing time taken for select sum:**

****

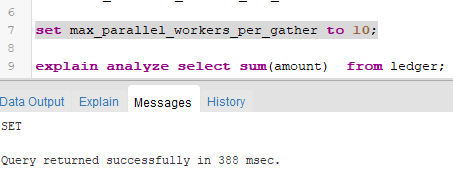
**Setting max\_parallel\_workers\_per\_gather to 3:**

****

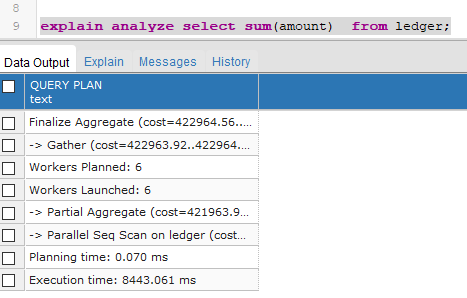
**Analyzing time taken for select sum:**

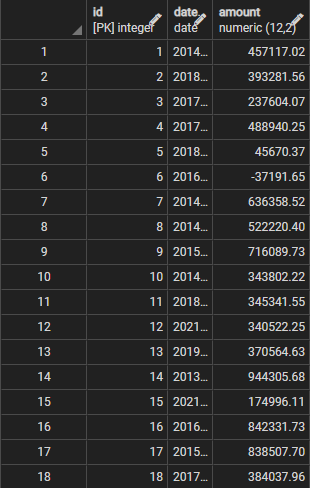
****

**Setting max\_parallel\_workers\_per\_gather to 10:**

****

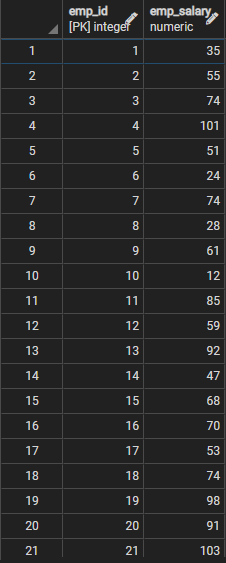
**Analyzing time taken for select sum:**

****

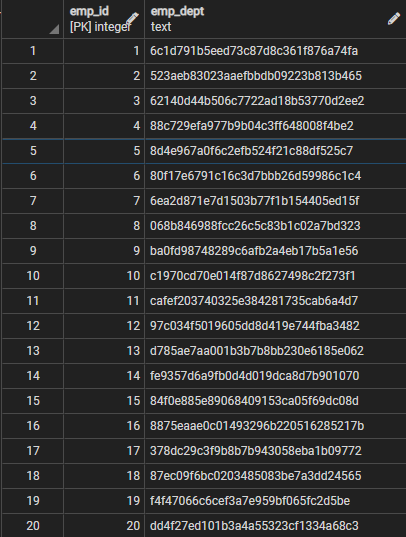




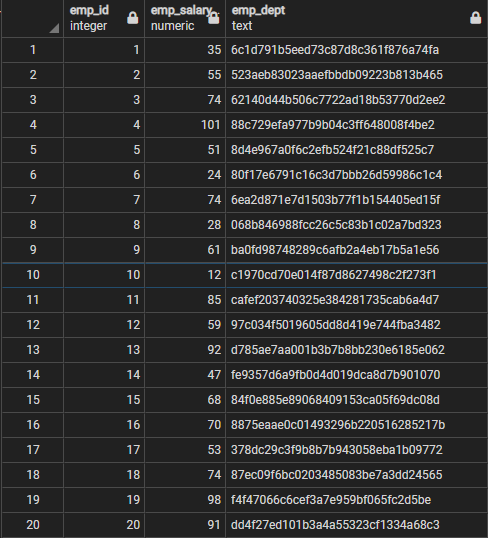
**Employee1 table**



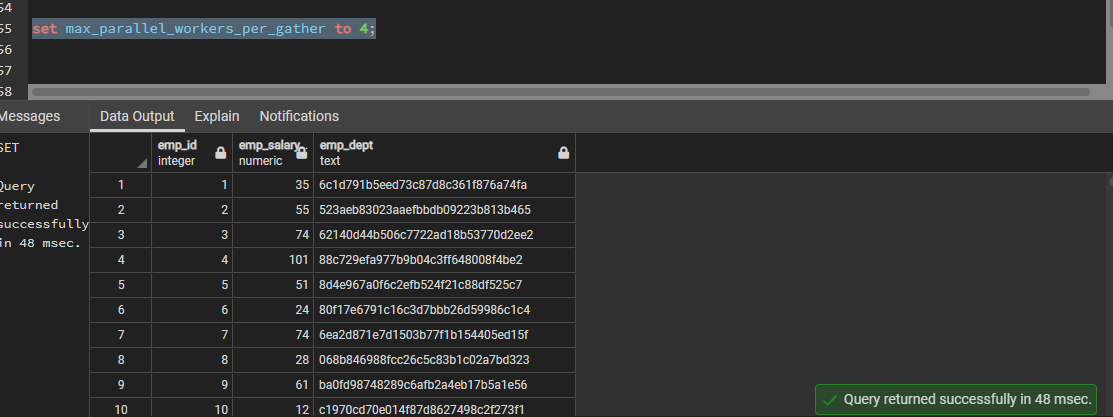
**Employee2 table**

****

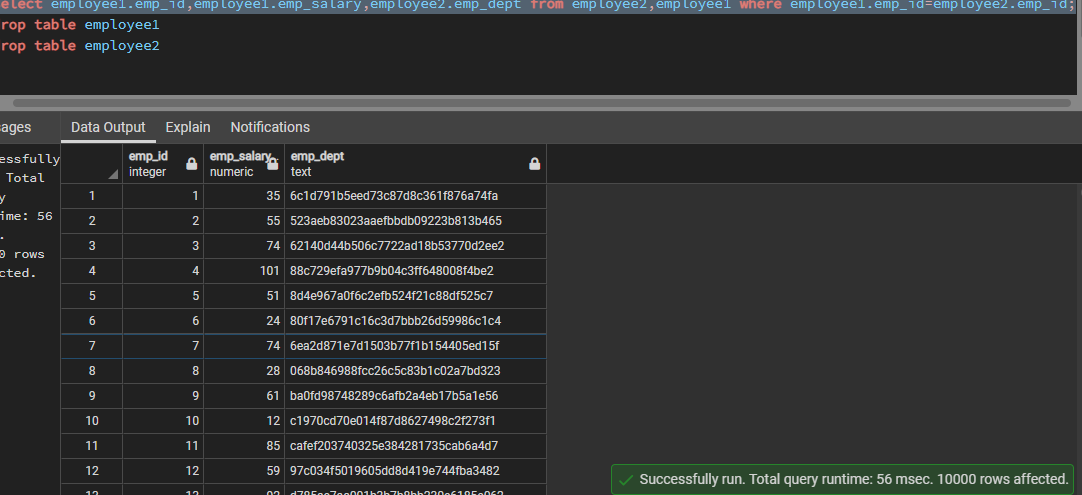
**Merging to tables employee1 and employee2 table**

****

**set max\_parallel\_workers\_per\_gather to 4;**

****

**set max\_parallel\_workers\_per\_gather to 1024;**

****

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

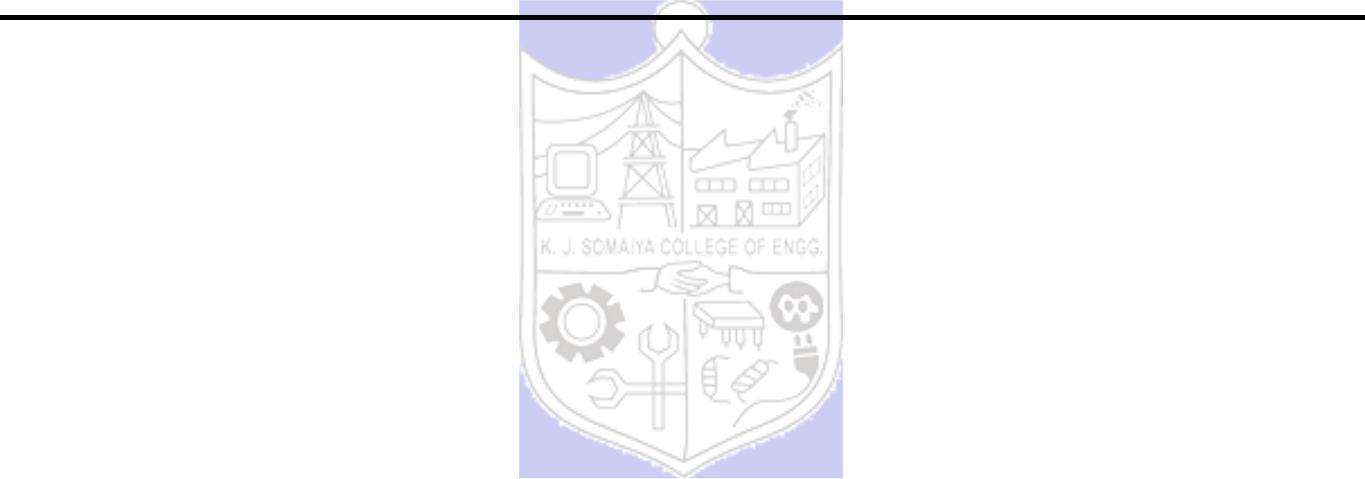
**Outcomes:**

**CO1 : Design advanced database systems using Parallel, Distributed and In-memory**

**databases and its implementation.**

**Conclusion: (Conclusion to be based on the outcomes achieved)**

**Hence we successfully implemented the execution of parallel databases queries.**



**Grade: AA / AB / BB / BC / CC / CD /DD**

**Signature of faculty in-charge with date**

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

**References:**

**Books/ Journals/ Websites:**

1. Elmasri and Navathe, “Fundamentals of Database Systems”, Pearson Education
2. https://www.postgresql.org/docs/