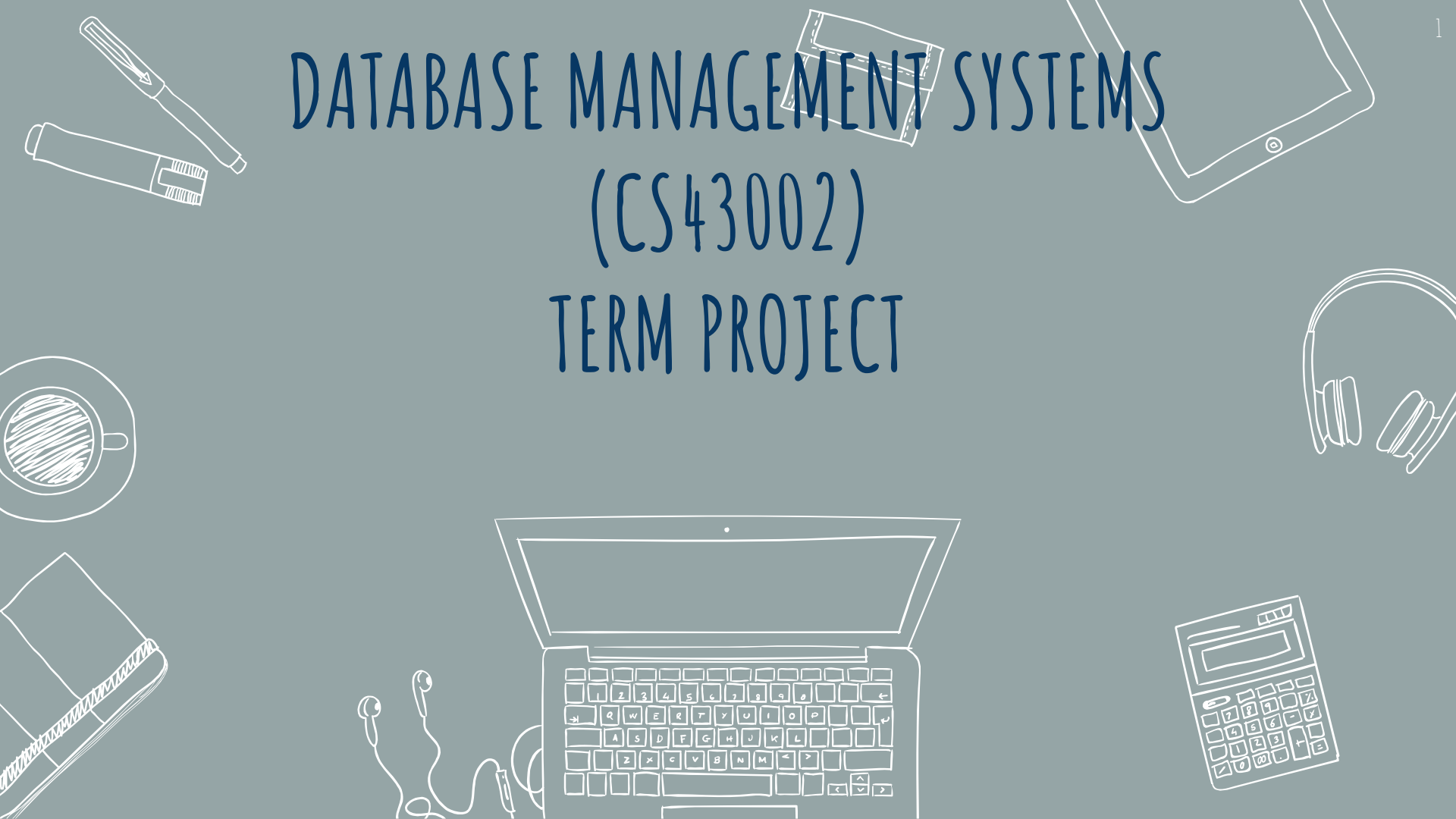
DATABASE MANAGEMENT SYSTEMS (CS43002) TERM PROJECT



Query Performance Analysis and Optimization through Real-time Metrics Collection in Postgres

Group members (AARVY):

1. Aditya Choudhary (20CS10005)
2. Astitva (20CS30007)
3. Rishi Raj (20CS30040)
4. Vikas Vijaykumar Bastewad (20CS10073)
5. Yashraj Singh (20CS10079)

WHY DO WE NEED THIS ?

- Finding bottlenecks in a query processing system.
- Making system reliable.
- Building an alerting mechanism to prevent database damage.
- Analyze query performance and optimize resources.

PREVIOUS WORK IN THIS DOMAIN

- pg_stat_statements module in Postgres, which collects statistics on the execution of SQL statements.
- Sys.dm_exec_query_stats in Microsoft SQL Server, which collects query performance metrics such as execution count, total execution time, and query plan analysis
- None of them provide real-time metrics collection during query execution.

OUR SOLUTION

- Build wrapper that can collect and report query processing metrics in real time.
- The collected metrics will be reported in a user-friendly format

INBUILT-POSTGRES STATISTICS

PostgreSQL offers a built-in statistics collector that automatically aggregates key metrics.

- pg_stat_database
- pg_stat_user_tables
- pg_stat_user_indexes
- pg_stat_bgwriter
- pg_statio_user_tables

QueryPilotX Statistics

- **get_stats_disk_usage_for_database:**
  - Gives the size of each database in bytes
- **get_stats_index_hit_rates:**
  - Gives indexes vs sequential scan through the table, helping us understand effectiveness of out indexing strategies.
- **get_stats_tx_rate_for_database:**
  - Gives the rate of transactions executed per second as well as the rate of roll backs executed per second.
- **get_stats_seconds_since_last_vacuum_per_table:**
  - Gives the amount of time passed in seconds since the last vacuum was performed on each table in the database.

SOME REAL-TIME METRICS

- **Query Duration:**
  - Reveals the time duration for which the query has been running.
- **CPU usage:**
  - Gives the percentage of CPU time consumed by each active query.
- **Memory Usage:**
  - Reveals the amount of memory consumed by each active query.
- **Disk I/O:**
  - Shows the number of bytes being read or written by each active query.

INTERFACES DEVELOPED BY US

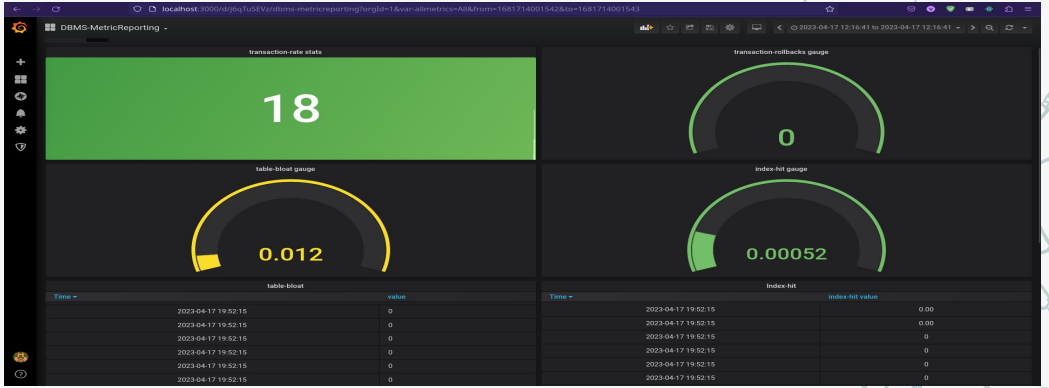
We have made three tools to provide different kinds of analytics

1. **Python CLI's app integrated with Grafana**
2. **Python GUI App**
3. **Go CLI app to provide in-depth table and overall database analysis**

Python CLI app integrated with Grafana

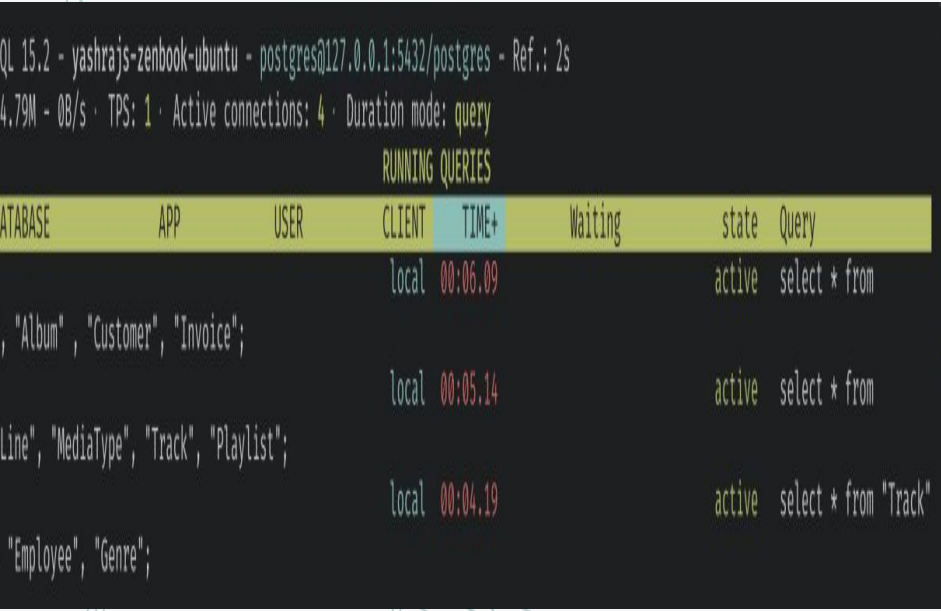
Python is used to connect to the database and fetch different metrics. These fetched raw data are then written in a log file, which can then be used by users to integrate with different monitoring tools. currently, we have integrated it with Grafana.

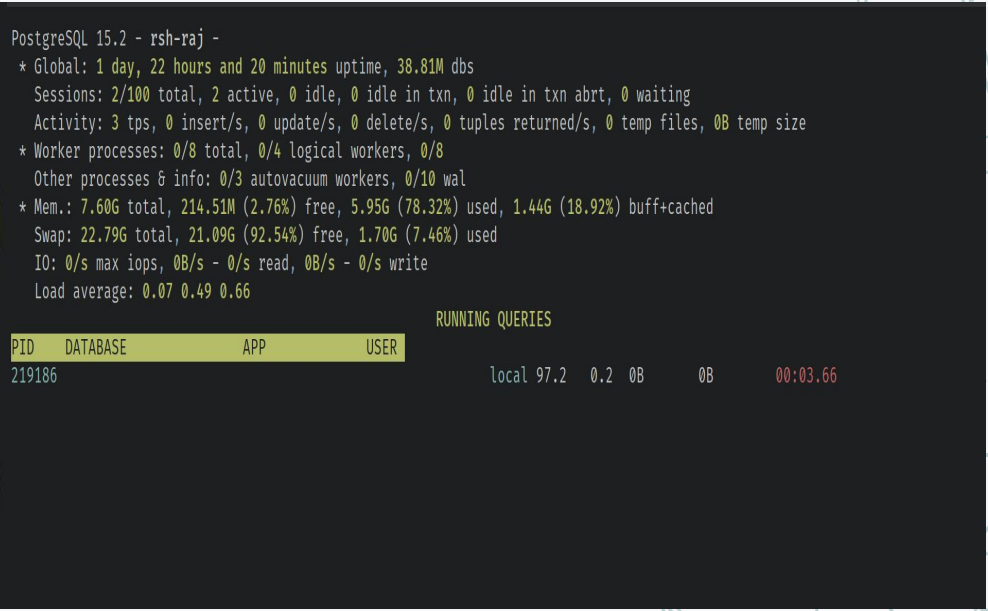
```
term-project/postgresql_metrics/postgresql_metrics via  v3.10.10 took 5s
➜ python3 metrics_logic.py all
postgresql-metrics.yml
[2023-04-16 14:34:42.335172] INFO: postgresql-metrics: open database connection to 127.0.0.1:5432, user 'postgres', database 'test'
# sleep 5 s to get diffs on derivative metrics
{'type': 'metric', 'key': 'postgresql', 'value': 7, 'attributes': {'what': 'client-connections', 'unit': 'connection'}}
{'type': 'metric', 'key': 'postgresql', 'value': 1, 'attributes': {'what': 'locks_granted', 'type': 'locks', 'locktype': 'relation', 'unit': 'lock'}}
{'type': 'metric', 'key': 'postgresql', 'value': 0, 'attributes': {'what': 'locks_waiting', 'type': 'locks', 'locktype': 'relation', 'unit': 'lock'}}
{'type': 'metric', 'key': 'postgresql', 'value': 1, 'attributes': {'what': 'locks_granted', 'type': 'locks', 'locktype': 'virtualxid', 'unit': 'lock'}}
{'type': 'metric', 'key': 'postgresql', 'value': 0, 'attributes': {'what': 'locks_waiting', 'type': 'locks', 'locktype': 'virtualxid', 'unit': 'lock'}}
{'type': 'metric', 'key': 'postgresql', 'value': 2, 'attributes': {'what': 'locks_granted', 'type': 'locks', 'locktype': 'total', 'unit': 'lock'}}
{'type': 'metric', 'key': 'postgresql', 'value': 0, 'attributes': {'what': 'locks_waiting', 'type': 'locks', 'locktype': 'total', 'unit': 'lock'}}
{'type': 'metric', 'key': 'postgresql', 'value': 0, 'attributes': {'what': 'wal-file-amount', 'unit': 'file'}}
{'type': 'metric', 'key': 'postgresql', 'value': 100.0, 'attributes': {'what': 'xid-remaining', 'unit': '%'}}
{'type': 'metric', 'key': 'postgresql', 'value': 7738159, 'attributes': {'what': 'database-size', 'database': 'test', 'unit': 'B'}}
{'type': 'metric', 'key': 'postgresql', 'value': 2.8, 'attributes': {'what': 'transaction-rate', 'type': 'transactions', 'database': 'test', 'unit': 'transaction'}}
{'type': 'metric', 'key': 'postgresql', 'value': 0.0, 'attributes': {'what': 'transaction-rollbacks', 'type': 'transactions', 'database': 'test', 'unit': 'transaction'}}
{'type': 'metric', 'key': 'postgresql', 'value': 0, 'attributes': {'what': 'sec-since-oldest-xact-start', 'database': 'test', 'unit': 's'}}
```



Python GUI Application

We have made an app that provides real-time query analysis and system resource usage in GUI.





Go CLI Application

This app provides in-depth table and overall database analysis in the form of two reports, whole DB analysis, and individual table analysis.

```
PostgreSQL Cluster:
    Name:                15/main
    Server Version:      15.2 (Ubuntu 15.2-1.pgdg22.04+1)
    Server Started:      10 Apr 2023 5:55:01 PM (6 days ago)
    System Identifier:   7194855233783590104
    Timeline:            1
    Last Checkpoint:     10 Apr 2023 5:55:01 PM (6 days ago)
    REDO LSN:            0/2433AD0
    Checkpoint LSN:      0/2433AD0 (0 B since REDO)
    Transaction IDs:     oldest = 716, next = 1165, range = 449
    Notification Queue:  0.0% used
    Active Backends:     1 (max 100)
    Recovery Mode?       no

System Information:
    Hostname:            yashrajs-zenbook-ubuntu
    CPU Cores:           16 x Intel(R) Core(TM) i9-10980HK CPU @ 2.40GHz
    Load Average:        0.37
    Memory:              used=4.6 GiB, free=11 GiB, buff=1.4 GiB, cache=13 GiB
    Swap:                used=0 B, free=3.8 GiB
    +-----------------------------------+-----------------+
    |                           Setting |           Value |
    +-----------------------------------+-----------------+
    |                    shared_buffers | 16384 (128 MiB) |
    |                         work_mem |  4096 (4.0 MiB) |
    |               maintenance_work_mem |  65536 (64 MiB) |
    |                       temp_buffers |  1024 (8.0 MiB) |
    |                autovacuum_work_mem |              -1 |
    |                     temp_file_limit |              -1 |
    |                max_worker_processes |               8 |
    |            autovacuum_max_workers |               3 |
    |   max_parallel_workers_per_gather |               2 |
    |          effective_io_concurrency |               1 |
    +-----------------------------------+-----------------+
```

```
Database #4:
    Name:                hms
    Owner:               postgres
    Tablespace:          pg_default
    Connections:         1 (no max limit)
    Frozen Xid Age:      449
    Transactions:        1621 (100.0%) commits, 0 (0.0%) rollbacks
    Cache Hits:          99.8%
    Rows Changed:        ins 0.0%, upd 0.0%, del 0.0%
    Total Temp:          0 B in 0 files
    Problems:            0 deadlocks, 0 conflicts
    Totals Since:
    Size:                7.8 MiB
    Installed Extensions:
      +---------+---------+-------------------------------+
      |    Name | Version |                       Comment |
      +---------+---------+-------------------------------+
      | plpgsql |     1.0 | PL/pgSQL procedural language |
      +---------+---------+-------------------------------+

Table #1 in "hms":
    Name:                hms.public.users
    Columns:             8
    Manual Vacuums:      never
    Manual Analyze:      never
    Auto Vacuums:        never
    Auto Analyze:        never
    Post-Analyze:        0.0% est. rows modified
    Row Estimate:        0.0% live of total 0
    Rows Changed:        ins 0.0%, upd 0.0%, del 0.0%
    HOT Updates:         0.0% of all updates
    Seq Scans:           0, 0.0 rows/scan
    Idx Scans:           0, 0.0 rows/scan
    Cache Hits:          0.0% (idx=0.0%)
    Size:                16 KiB
    +-----------+-------+--------+-------+------------+-------+-----------------+-----------------+
    |     Index |  Type |   Size | Bloat | Cache Hits | Scans | Rows Read/Scan | Rows Fetched/Scan |
    +-----------+-------+--------+-------+------------+-------+-----------------+-----------------+
    | users_pkey | btree | 16 KiB |       |            |     0 |             0.0 |             0.0 |
    +-----------+-------+--------+-------+------------+-------+-----------------+-----------------+

Table #2 in "hms":
    Name:                hms.public.doctors
    Columns:             4
```

THANK YOU