

## BAB III

### PELAKSANAAN KERJA MAGANG

#### 3.1 Kedudukan dan Koordinasi

Pelaksanaan kerja magang yang dilakukan oleh mahasiswa pada PT. Inovasi Informatika Indonesia pada bulan Juli 2022 mahasiswa akan melaksanakan *training* terlebih dahulu sebelum akhirnya di masukkan ke dalam divisi yang sesuai dengan peminatan dan kemampuan mahasiswa. Hingga pada Agustus 2022, mahasiswa ditempatkan pada divisi *Data Management* sebagai seorang *consultant*. Divisi *Data Management* memiliki beberapa proyek yang sudah berjalan yaitu *training*, *maintenance*, *managed service*, serta *hardening*.

Pada kerja magang ini, mahasiswa di tempatkan ke dalam divisi *Data Management* dengan proyek *managed service* di bawah naungan Bapak Farhan A. M. Muhammad. Dimana proyek *managed service* telah memiliki proyek dengan masa kontrak yaitu 1 tahun, mahasiswa di berikan kesempatan untuk ikut serta dalam membantu *managed service database* klien pada perusahaan XYZ. Tugas utama dari seorang konsultan *Data Management* terlebih pada proyek *managed service* adalah memonitor serta memastikan semua *database* klien dapat berjalan dan dalam kondisi yang baik atau normal. Apabila pada *database* klien terdapat *issue* maka seorang *managed service* dapat membantu untuk memberikan laporan terkait identifikasi penyebab *issue*, serta solusi atas *issue* yang terjadi. Pada gambar 3.1.1 merupakan alur kerja yang dilakukan oleh divisi *Data Management* terkhusus pada proyek *managed service* pada PT. Inovasi Informatika Indonesia.

Seorang yang bertanggung jawab pada proyek *managed service* akan selalu melakukan *checking* terhadap *database* klien setiap harinya, dimana dari hasil pemeriksaan yang dilakukan tersebut akan terlihat apakah *database* klien terdapat *issue* atau tidak. Apabila *database* memiliki *issue* maka divisi *managed service* akan mengidentifikasi *issue* tersebut, dimana identifikasi

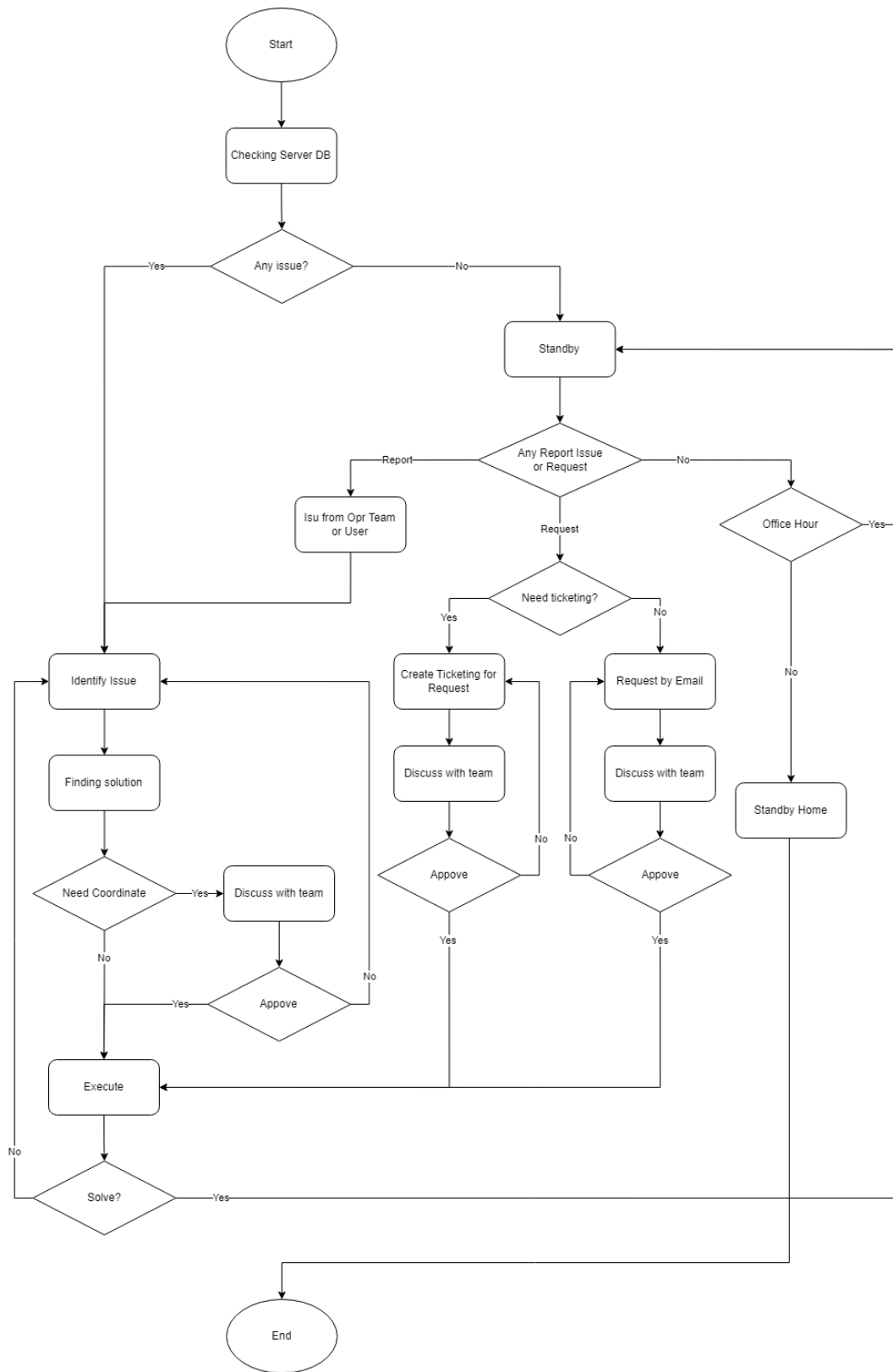
*issue* dibagi menjadi 3 yaitu *high*, *medium*, dan *low issue*. Setelah *issue* diidentifikasi ke dalam kategori yang tepat, maka akan dicari solusi atas masalah atau *issue* yang dialami. Jika solusi telah ditemukan, apakah solusi tersebut memerlukan koordinasi untuk mengkonfirmasi solusi tersebut dari divisi lain atau *user*. Jika iya, maka akan dilakukan diskusi terlebih dahulu bersama dengan divisi atau *user* yang berwenang, dari hasil diskusi apabila sudah disetujui maka akan dilakukan eksekusi terhadap *issue* dan apabila hasil diskusi tidak disetujui maka akan dilakukan identifikasi ulang terhadap *issue* yang ada untuk pemeriksaan kembali. Dari eksekusi solusi yang dilakukan apakah *issue* terhadap *database* telah terpecahkan, apabila belum maka akan dilakukan identifikasi *issue* kembali. Jika *issue* telah terpecahkan maka seorang yang bertanggung jawab pada proyek *managed service* akan tetap *standby* untuk berjaga-jaga apabila *database* terdapat masalah atau *issue* baru.

Apabila *database* klien tidak terdapat *issue* maka seorang yang bertanggung jawab pada proyek *managed service* akan melakukan *standby*, dimana *issue* tetap dapat didapatkan diluar dari *monitoring* yang telah dilakukan. Apabila terdapat laporan *issue* dari divisi lain selain *Database* seperti adanya *user* yang tidak dapat mengakses *database*, terdapat *blocking*, *database* lambat, atau lainnya maka akan dilakukan identifikasi terhadap *issue* yang dilaporkan tersebut dan dilanjutkan dengan pencarian solusi, serta pengimplementasian solusi. Selain itu, adapun *request* yang dapat dilakukan oleh *user* atas *database*, seperti *request* untuk mendapatkan IP Address pada *pg\_hba* agar dapat mengakses *database*, *request install database* baru, penambahan *user* baru, dan lainnya. *Request* yang dilakukan oleh *user* nantinya akan diidentifikasi apakah membutuhkan *ticket* atau tidak, *ticket* disini memiliki pengertian yaitu sebagai antrian kebutuhan yang dibagi menjadi *high*, *medium*, dan *low*. Apabila dari hasil identifikasi *request* memerlukan *ticket*, sebelum dieksekusi maka akan didiskusikan terlebih dahulu dengan tim, dan jika sudah mendapatkan persetujuan baru akan dieksekusi, apabila dari hasil diskusi tidak disetujui maka *user* perlu

melakukan *request* ulang dan *ticket* akan diidentifikasi kembali. Jika tidak memerlukan *ticketing* maka *request* dapat dikirimkan via email, dan akan didiskusikan terlebih dahulu, apabila dari hasil diskusi disetujui maka akan langsung dieksekusi, apabila tidak maka *user* perlu melakukan *request* ulang melalui email.

Ketika tidak terdapat *request* ataupun laporan *issue* mengenai *database*, jika waktu masih menunjukkan *office hour* maka akan tetap dilakukan *standby* terhadap *database* yang dapat dilakukan secara *remote* atau *standby at home*. Apabila sudah diluar dari *office hour* maka pekerjaan untuk *managed service* dihari tersebut telah selesai.





Gambar 3.1.1 Alur Kerja *Managed Service*

Selain menjadi salah satu anggota *managed service*, mahasiswa diberi kesempatan untuk terlibat dalam *Project Internal Data Management Internship* pada PT. Inovasi Informatika Indonesia. Dalam pengerjaan *project* ini dikoordinasi oleh Muhammad Fahmi selaku PIC. *Project* ini beranggotakan seluruh mahasiswa magang pada divisi *Data Management*, pada tabel 3.1.1 merupakan *task list* yang terdapat untuk *Project Internal* yang dilakukan oleh *internship Data Management*, dimana pada kesempatan ini mahasiswa diberikan pernah yaitu *benchmarking* dan *alert and monitoring*.

Tabel 3.1. 1 *Task List Project Internal Data Management Internship*

<b><i>Task List</i></b>
<i>Streaming replication using RepMgr</i>
<i>Tuning</i>
<i>Hardening</i>
<i>Load balancer and Connection Pooling</i>
<i>Benchmarking</i>
<i>Alert and Monitoring</i>

### **3.2 Tugas dan Uraian Kerja Magang**

Selama program kerja magang yang dilakukan mahasiswa pada PT. Inovasi Informatika Indonesia, mahasiswa tentunya diberikan tugas selama menjadi salah satu anggota *internship* pada divisi *Data* khususnya proyek *managed service*. Seperti yang diketahui PT. Inovasi Informatika Indonesia merupakan perusahaan yang menyediakan *service* untuk kebutuhan terkait dengan Teknologi Informasi. Sebelum mahasiswa diberikan tugas dan tanggung jawab untuk dapat berkontribusi langsung terhadap *project* yang terdapat pada PT. Inovasi Informatika Indonesia, mahasiswa diberikan pembekalan *training* terlebih dahulu.

Pembekalan *training* yang didapatkan oleh mahasiswa dilakukan selama 2 minggu dihitung dari pertama kali mahasiswa bergabung menjadi *internship*. Tujuan dilakukannya *training* ini adalah untuk mendapatkan pengetahuan dasar mengenai RedHat. Modul yang dipelajari adalah modul RH124 dan

RH134, dimana modul RH124 mempelajari mengenai *basic* Linux dan modul RH134 merupakan modul lanjutan untuk mendalami konsep-konsep dari sistem Linux. *Training* ini dilakukan secara *offline* di ruangan *Training* yang disediakan oleh PT. Inovasi Informatika Indonesia. Adapun pada tabel 3.2.1 merupakan uraian dari materi yang didapatkan paska pembekalan:

Tabel 3.2. 1 Uraian Materi *Training*

<b>Minggu Ke-</b>	<b>Training</b>	<b>Materi</b>
1	RedHat 124	<p>Sumber pembekalan yaitu Modul RedHat 124 dengan total yaitu 16 bab, yang mempelajari materi sebagai berikut:</p> <p>Bab 1: <i>Getting Started with Red Hat Enterprise Linux</i></p> <p>Bab 2: <i>Accessing the Command Line</i></p> <p>Bab 3: <i>Managing Files from the Command Line</i></p> <p>Bab 4: <i>Getting Help in Red Hat Enterprise Linux</i></p> <p>Bab 5: <i>Creating, Viewing, and Editing Text Files</i></p> <p>Bab 6: <i>Managing Local Users and Groups</i></p> <p>Bab 7: <i>Controlling Access to Files</i></p> <p>Bab 8: <i>Monitoring and Managing Linux Processes</i></p> <p>Bab 9: <i>Controlling Services and Daemons</i></p> <p>Bab 10: <i>Configuring and Securing ssh</i></p> <p>Bab 11: <i>Analyzing and Storing Logs</i></p> <p>Bab 12: <i>Managing Networking</i></p> <p>Bab 13: <i>Archiving and Transferring Files</i></p> <p>Bab 14: <i>Installing and Updating Software Packages</i></p> <p>Bab 15: <i>Accessing Linux File System</i></p> <p>Bab 16: <i>Analyzing Servers and Getting Supports</i></p>
2	RedHat 134	<p>Sumber pembekalan yaitu Modul RedHat 134 dengan total bab yaitu 13 bab, yang mempelajari materi sebagai berikut:</p> <p>Bab 1: <i>Improving Command:Line Productivity</i></p> <p>Bab 2: <i>Scheduling Future Tasks</i></p> <p>Bab 3: <i>Tuning System Performance</i></p> <p>Bab 4: <i>Controlling Access to Files with</i></p>

<b>Minggu Ke-</b>	<b>Training</b>	<b>Materi</b>
		<i>Acls</i> Bab 5: <i>Managing Selinux Security</i> Bab 6: <i>Managing Basic Storage</i> Bab 7: <i>Managing Local Volumes</i> Bab 8: <i>Implementing Advanced Storage Features</i> Bab 9: <i>Accessing Network Attached Storage</i> Bab 10: <i>Controlling the Boot Process</i> Bab 11: <i>Managing Network Security</i> Bab 12: <i>Installing Red Hat Enterprise Linux</i> Bab 13: <i>Running Containers</i>

Pada minggu ketiga kegiatan program kerja magang dilakukan pembekalan *Product Knowledge*, dimana pada kegiatan tersebut mahasiswa akan diberikan materi terkait dengan *product* yang ada dan diharapkan mahasiswa lebih mengenal seluruh produk atau jasa yang ditawarkan oleh PT. Inovasi Informatika Indonesia kepada klien. Adapun pada tabel 3.2.2 merupakan uraian dari kegiatan *Product Knowledge* yang dilakukan:

Tabel 3.2. 2 Uraian *Product Knowledge*

<b>Minggu Ke-</b>	<b>Product Knowledge</b>	<b>Materi</b>
3	<i>Middleware</i>	Mempelajari SSO, Jboss EAP, 3Scale API Management, Apache Camel, dan AMQ Messaging System.
4	<i>Security</i>	Mempelajari <i>Vulnerability Assessment</i> , <i>Penetration Testing</i> , dan DVWA ( <i>Brute Force</i> , <i>Command Injection</i> , <i>SQL Injection</i> , dan XSS)
5	<i>Cloud</i>	Mempelajari secara umum bagaimana <i>Cloud Computing</i> .
	<i>RnD</i>	Mempelajari Fushion dan dikenalkan kepada <i>product i3gis</i> .
6	<i>Data</i>	Mempelajari PostgreSQL terkait instalasi, <i>upgrade</i> , dan replikasi <i>database</i> .
	<i>DevOps</i>	Mempelajari <i>Container</i> , <i>Docker Compose</i> , <i>Pipeline</i> , dan JenKins.

Pada minggu keenam kegiatan program kerja magang, mahasiswa ditempatkan pada divisi *Data Management* dan diberikan bekal *training*

PostgreSQL dan EDB selama 6 minggu, dimana untuk 2 minggu pertama mempelajari PostgreSQL dan pada minggu ke-7 program kerja magang dimulai *training* untuk EDB, dimana EDB sendiri merupakan versi *Enterprise* dari PostgreSQL. Adapun pada tabel 3.2.3 merupakan uraian dari pembekalan yang telah dilakukan:

Tabel 3.2. 3 Training Divisi *Data Management*

<b>Minggu Ke-</b>	<b>Training</b>	<b>Materi</b>
6 dan 7	PostgreSQL	Mempelajari dasar-dasar PostgreSQL, bagaimana cara melakukan instalasi, <i>upgrade minor</i> dan <i>major</i> , replikasi, <i>pgbench</i> , <i>vacuum</i> , <i>indexing</i> , dan <i>blocking</i> pada <i>database</i> .
7 s/d 11	EDB	Mempelajari Modul EDB yang telah disediakan pada website resmi EDB.
		Mempelajari PEM yang disediakan oleh EDB untuk <i>monitoring database</i> . Pembelajaran mencakup instalasi dan konfigurasi PEM Server.
		Mempelajari PEM Agent yang berguna untuk mengumpulkan data-data pada <i>database</i> untuk di tampilkan pada PEM Server untuk keperluan monitoring. Pembelajar mencakup instalasi dan konfigurasi PEM Agent.

Setelah proses *training* yang dilakukan oleh mahasiswa, mahasiswa mendapatkan kesempatan untuk masuk kedalam proyek *managed service* dan selesainya *training* yang dilakukan oleh mahasiswa maka mahasiswa dirasa cukup untuk dapat terjun langsung ke dalam salah satu *project* yang terdapat pada PT. Inovasi Informatika Indonesia. Dimana pada program kerja magang ini mahasiswa diberi kesempatan untuk terjun langsung ke dalam *project* yang telah dikontrak sebelumnya selama 1 tahun dihitung dari awal tahun 2022 oleh perusahaan XYZ yang merupakan perusahaan *finance*. Pada minggu kedua belas, mahasiswa mendapatkan tugas-tugas untuk *project* pada perusahaan XYZ, tugas tersebut merupakan keperluan *managed service database* klien. Selain itu, pada minggu kedua puluh satu mahasiswa secara *parallel* masuk kedalam *Project Internal Data Management Internship*.



Berikut merupakan uraian kegiatan program kerja magang yang dilakukan oleh mahasiswa pada PT. Inovasi Informatika Indonesia:

### **3.2.1. *MANAGED SERVICE* (Minggu 12 s/d 26)**

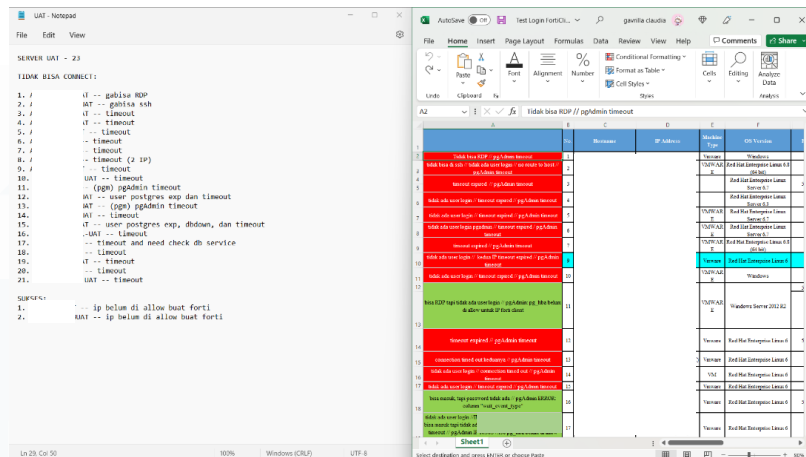
Pada program kerja magang pada PT. Inovasi Informatika Indonesia, mahasiswa diberi kesempatan untuk terjun langsung ke dalam *project* divisi *Data Management* yaitu *managed service* untuk perusahaan XYZ. Pada kesempatan yang diberikan mahasiswa awalnya hanya melakukan *shadowing* pada perusahaan XYZ, dan diberikan tugas-tugas dengan identifikasi *issue* yaitu *low*. Tugas yang nantinya sudah dikerjakan akan di *submit* kepada mentor yang bertanggung jawab, dimana tugas tersebut akan dicek terlebih dahulu sebelum akhirnya diteruskan kepada klien.

#### **1. *Daily Monitoring***

Pada kesempatan mengikuti *managed service project*, mahasiswa awalnya melakukan *shadowing* untuk perusahaan XYZ, dimana tujuan dilakukannya *shadowing* ini adalah agar mahasiswa mengerti bagaimana sistem kerja seorang *managed service*. Pada proyek *managed service* rutin dilakukan *daily monitoring* pada server *database* klien.

##### **a. *Pengecekan Server Database***

*Daily monitoring* ini termasuk ke dalam kategori *low*, sehingga tugas ini dipercayakan mentor kepada mahasiswa magang, dimana mentor menjelaskan terkait dengan pengecekan server dan bagaimana melihat kondisi atau terdapat masalah ketika ingin mengakses ke dalam server. Pada gambar 3.2.1.1 merupakan salah satu contoh *daily monitoring* yang dilakukan oleh mahasiswa:



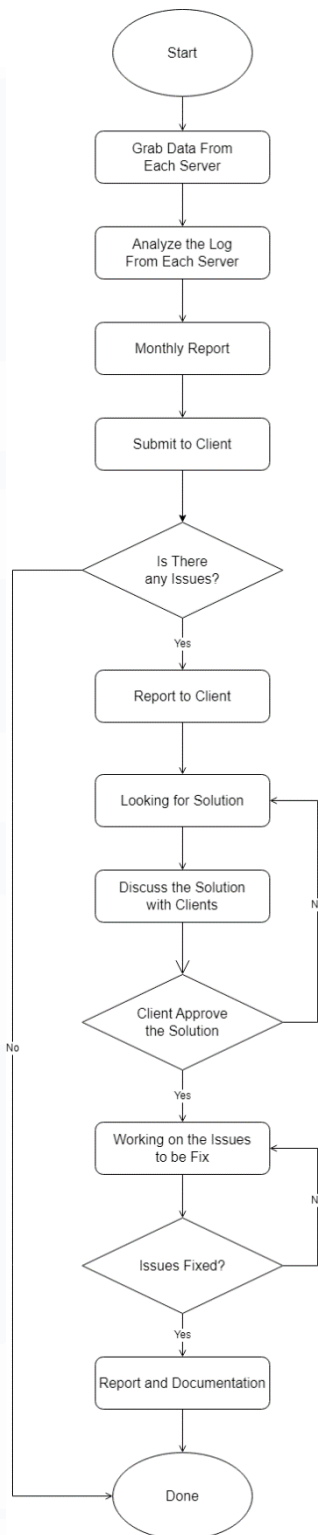
Gambar 3.2.1. 1 *Daily Monitoring: Checking Server UAT*

Pada *daily monitoring* tersebut mahasiswa diberikan tugas untuk mengecek seluruh Server UAT pada perusahaan XYZ dengan cara melakukan ‘ssh’ atau akses menggunakan FortiClient terhadap seluruh Server UAT dan memberikan laporan terkait akses yang dilakukan apakah berhasil atau terdapat kendala.

## 2. *Monthly Monitoring*

Selain dari pada *daily monitoring*, mahasiswa diberi tugas untuk melakukan *monthly monitoring*. *Monthly monitoring* dilakukan setiap 1 bulan sekali dengan mengecek seluruh *database* server klien dimana total terdapat 10 server *database* yang perlu dicek dan dianalisis *log*-nya setiap bulannya. Berikut merupakan alur kerja *monthly monitoring* yang dilakukan oleh mahasiswa:

UNIVERSITAS  
MULTIMEDIA  
NUSANTARA



Gambar 3.2.1. 2 Alur Kerja *Monthly Monitoring*

Dimana pada Gambar 3.2.1.2 merupakan skenario *managed service* yang dilakukan dalam 1 bulan sekali, dimana mahasiswa yang diberi kesempatan dalam proyek *managed service* akan mengambil data-data yang diperlukan pada setiap server *database* milik klien. Setelah data yang diperlukan terkumpul, maka akan dilakukan analisis terhadap *log-log* pada setiap server *database* tersebut. Hasil analisa terhadap *log* dan data yang sudah terkumpul sebelumnya akan dimasukkan ke dalam *Monthly Report* yang akan di *submit* kepada klien untuk menjadi laporan bulanan terkait dengan server *database* mereka.

Dari laporan tersebut seorang yang bertanggung jawab pada proyek *managed service* mengetahui bagaimana keadaan dari server *database* klien, apabila terdapat *issue* pada *database* server klien maka seorang yang bertanggung jawab pada proyek *managed service* akan memberikan informasi tersebut kepada klien. seorang yang bertanggung jawab pada proyek *managed service* akan mencari solusi terkait dengan *issue* yang ada, apabila ditemukan beberapa solusi-solusi yang dapat menyelesaikan permasalahan yang terjadi terhadap server *database* klien maka solusi tersebut akan di diskusikan terlebih dahulu kepada klien. Apabila klien menyetujui solusi yang ditawarkan oleh seorang yang bertanggung jawab pada proyek *managed service* maka akan dilakukan pengimplementasian terhadap solusi tersebut, namun jika tidak maka seorang yang bertanggung jawab pada proyek *managed service* akan mencari solusi lain yang dapat menyelesaikan *issue* yang ada. Apabila solusi yang diimplementasikan telah berhasil maka seorang yang bertanggung jawab pada proyek *managed service* akan membuat laporan terkait penyelesaian *issue* serta dokumentasi agar dapat berguna di waktu mendatang apabila mengalami *issue* serupa. Namun apabila solusi yang diimplementasikan belum dapat menyelesaikan *issue* yang ada maka seorang yang bertanggung

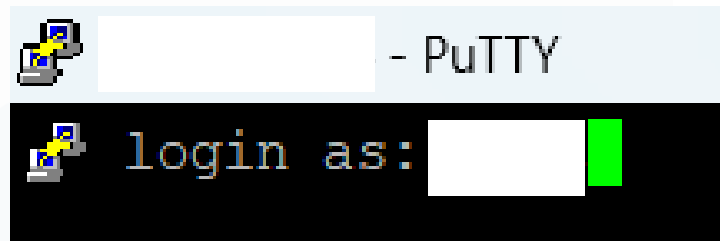
jawab pada projek *managed service* akan mencoba kembali solusi yang ada hingga *issue* yang terjadi pada server *database* klien terpecahkan.

Berikut merupakan *monthly monitoring* yang dilakukan pada bulan November 2022 untuk satu server *master* dan satu server *slave* pada perusahaan XYZ, dimana pengambilan data ini dilakukan pada minggu kedua puluh tiga:

**a. Pengambilan Data: Server Master – Partner Production 01 (XX-dbp\*\*\*\*\*01)**

**1) Login**

Dalam *monthly monitoring* yang dilakukan untuk satu server *master* yaitu server Partner Production 01 (XX-dbp\*\*\*\*\*01) pada perusahaan XYZ, dengan menggunakan *tools* PuTTY untuk mengecek server *database* klien, dimana untuk mengakses server diperlukan *IP Address*, *key password*, dan *user*, dimana data-data tersebut diberikan oleh klien dan bersifat rahasia.



Gambar 3.2.1. 3 Login Root Server Partner Production 01

**2) Cek dan meng-update password**

Setelah masuk kedalam server *database master* klien maka mahasiswa akan masuk menggunakan *user* postgres, lalu dilakukan pengecekan terhadap *password*. Pada gambar 3.2.1.3 dapat terlihat *password* untuk *user* postgres sudah *expired*. Maka dari itu diperlukan pembaharuan *password* untuk *user* postgres. Perubahan *password* berguna untuk

meningkatkan keamanan server dan terjaga dari *hacker* yang berusaha untuk membobol server perusahaan.

```
[postgres@dbp-01 HealthCheck_dbp-01_November]$ chage -l postgres
Last password change      : Nov 02, 2022
Password expires         : Dec 02, 2022
Password inactive        : never
Account expires          : never
Minimum number of days between password change : 0
Maximum number of days between password change : 30
Number of days of warning before password expires : 7
[postgres@dbp-01 HealthCheck_dbp-01_November]$ exit
logout.
[root@dbp-01 ~]# passwd postgres
Changing password for user postgres.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
[root@dbp-01 ~]# su - postgres
Last login: Wed Dec  7 13:48:57 WIB 2022 on pts/0
[postgres@dbp-01 ~]$ chage -l postgres
Last password change      : Dec 07, 2022
Password expires         : Jan 06, 2023
Password inactive        : never
Account expires          : never
Minimum number of days between password change : 0
Maximum number of days between password change : 30
Number of days of warning before password expires : 7
```

Gambar 3.2.1. 4 *Checking Password* untuk User Postgres pada Server Partner Production 01

### 3) Proses pengambilan data

Sebelum melakukan pengambilan data maka diperlukan membuat direktori baru menggunakan *command* *mkdir*. Pembuatan direktori baru ini agar lebih mempermudah untuk pengambilan data kedepannya, karena data-data tersimpan dalam direktori-direktori yang dikategorikan setiap bulannya.

```
[postgres@dbp-01 ~]$ mkdir HealthCheck_dbp-01_November
```

Gambar 3.2.1. 5 Membuat Direktori Baru untuk Bulan November 2022 pada Server Partner Production 01

Agar mempermudah dalam memantau seluruh data yang diperlukan maka digunakan *command* *cp -r* untuk meng-copy file yang terdapat pada direktori sebelumnya dan disalin kepada direktori baru.

```
[postgres@dbp-01 ~]$ cp -r HealthCheck_dbp-01_Oktober/* HealthCheck_dbp-01_November/
```

Gambar 3.2.1. 6 Meng-copy file pada Direktori Bulan Oktober

Setelah direktori yang baru dimasukkan file salinan dari direktori bulan lalu, maka mahasiswa akan berpindah direktori ke direktori yang baru dengan *command* *cd* `HealthCheck_XX-dbp*****01_November/` dan melihat

terlebih dahulu *file* apa saja yang sudah terdapat pada direktori yang telah dibuat dengan menggunakan *command* *ls* -lah. Dengan *command* tersebut kita dapat melihat seluruh *file* yang terdapat pada direktori, *size* pada *file*, serta waktu perubahan terakhir.

```
[postgres@dba ~]$ cd Healthcheck/
[postgres@dba ~]$ ls
database_size.txt  db_status.txt  du-sh.txt  lshk.txt  pg_hba.conf  postgresql.conf  sar
database_table_size.txt  dead_life_tuples.txt  free.txt  lscpu.txt  pg_ident.conf  replication.conf  session.txt
database_uptime.txt  df-h.txt  log  os_version.txt  postgresql.auto.conf  replication_status.txt  uptime.txt
[postgres@dba ~]$ ls -lah
total 124K
drwxr-xr-x 4 postgres dba 4.0K Dec 5 14:06 .
drwx----- 7 postgres dba 4.0K Dec 5 14:05 ..
-rw-r--r-- 1 postgres dba 143 Dec 5 14:06 database_size.txt
-rw-r--r-- 1 postgres dba 138K Dec 5 14:06 database_table_size.txt
-rw-r--r-- 1 postgres dba 89 Dec 5 14:06 database_uptime.txt
-rw-r--r-- 1 postgres dba 4.3K Dec 5 14:06 db_status.txt
-rw-r--r-- 1 postgres dba 1.4K Dec 5 14:06 dead_life_tuples.txt
-rw-r--r-- 1 postgres dba 953 Dec 5 14:06 df-h.txt
-rw-r--r-- 1 postgres dba 1.4K Dec 5 14:06 du-sh.txt
-rw-r--r-- 1 postgres dba 204 Dec 5 14:06 free.txt
drwxr-xr-x 2 postgres dba 4.0K Dec 5 14:06 log
-rw-r--r-- 1 postgres dba 344 Dec 5 14:06 lshk.txt
-rw-r--r-- 1 postgres dba 697 Dec 5 14:06 lscpu.txt
-rw-r--r-- 1 postgres dba 38 Dec 5 14:06 os_version.txt
-rw----- 1 postgres dba 6.2K Dec 5 14:06 pg_hba.conf
-rw----- 1 postgres dba 1.6K Dec 5 14:06 pg_ident.conf
-rw----- 1 postgres dba 88 Dec 5 14:06 postgresql.auto.conf
-rw----- 1 postgres dba 27K Dec 5 14:06 postgresql.conf
-rw-r--r-- 1 postgres dba 131 Dec 5 14:06 replication.conf
-rw-r--r-- 1 postgres dba 62 Dec 5 14:06 replication_status.txt
drwxr-xr-x 2 postgres dba 4.0K Dec 5 14:06 sar
-rw-r--r-- 1 postgres dba 730 Dec 5 14:06 session.txt
-rw-r--r-- 1 postgres dba 72 Dec 5 14:06 uptime.txt
```

Gambar 3.2.1. 7 Masuk ke dalam Direktori Bulan November dan *List File*

#### 4) Data *uptime*

Pada pengambilan data untuk keperluan analisis, pengambilan data dilakukan pada 2 skenario yaitu pada server dan *database* *psql*. Pengambilan data untuk *database uptime* diambil pada *database* dengan menggunakan *command*,

```
select current_timestamp - pg_postmaster_start_time()
as uptime;
```

dimana *command* tersebut digunakan untuk mendapatkan data *uptime database*. Data tersebut dimasukkan kedalam *file database\_uptime.txt*.

```
postgres=# select current_timestamp - pg_postmaster_start_time() as uptime;
uptime
-----
281 days 16:59:31.165128
(1 row)

postgres=# \o database_uptime.txt
postgres=# select current_timestamp - pg_postmaster_start_time() as uptime;
postgres=# \o
postgres=# \q
```

Gambar 3.2.1. 8 Pengambilan Data *Uptime* pada *Database* Server Partner Production 01

Selain itu untuk memastikan apabila data yang diambil sudah tersimpan ke dalam *file*, dilakukan pengecekan dan terlihat dari waktunya sudah mengalami perubahan.

```

[postgres@dbp-01 HealthCheck ~]$ ls -lah
total 124K
drwxr-xr-x 4 postgres dba 4.0K Dec 5 14:06 .
drwx----- 7 postgres dba 4.0K Dec 5 14:05 ..
-rw-r--r-- 1 postgres dba 143 Dec 5 14:06 database_size.txt
-rw-r--r-- 1 postgres dba 1.3K Dec 5 14:06 database_table_size.txt
-rw-r--r-- 1 postgres dba 89 Dec 5 14:12 database_uptime.txt
-rw-r--r-- 1 postgres dba 4.3K Dec 5 14:06 db_status.txt
-rw-r--r-- 1 postgres dba 1.4K Dec 5 14:06 dead_life_tuples.txt
-rw-r--r-- 1 postgres dba 555 Dec 5 14:06 df-h.txt
-rw-r--r-- 1 postgres dba 1.4K Dec 5 14:06 du-sh.txt
-rw-r--r-- 1 postgres dba 204 Dec 5 14:06 free.txt
drwxr-xr-x 2 postgres dba 4.0K Dec 5 14:06 log
-rw-r--r-- 1 postgres dba 344 Dec 5 14:06 lsblk.txt
-rw-r--r-- 1 postgres dba 697 Dec 5 14:06 lspcu.txt
-rw-r--r-- 1 postgres dba 38 Dec 5 14:06 os_version.txt
-rw----- 1 postgres dba 6.2K Dec 5 14:06 pg_hba.conf
-rw----- 1 postgres dba 1.6K Dec 5 14:06 pg_ident.conf
-rw----- 1 postgres dba 88 Dec 5 14:06 postgresql.auto.conf
-rw----- 1 postgres dba 27K Dec 5 14:06 postgresql.conf
-rw-r--r-- 1 postgres dba 131 Dec 5 14:06 replication.conf
-rw-r--r-- 1 postgres dba 62 Dec 5 14:06 replication_status.txt
drwxr-xr-x 2 postgres dba 4.0K Dec 5 14:06 sar
-rw-r--r-- 1 postgres dba 730 Dec 5 14:06 session.txt
-rw-r--r-- 1 postgres dba 72 Dec 5 14:06 uptime.txt

```

Gambar 3.2.1. 9 Pengecekan *File* uptime.txt Mengalami Perubahan

## 5) Data disk free

Selanjutnya, dilakukan pengambilan data pada bagian server dijalankan *command* berikut,

```
df -h > df-h.txt
```

```
df -h /data/partner/data/ >> df-h
```

*command* tersebut merupakan *command* untuk melihat *disk free* pada server *database* Partner Production 01 (XX-dbp\*\*\*\*\*01) yang dimasukkan kedalam *file* df-h.txt.

```

[postgres@dbp-01 HealthCheck ~]$ cat df-h.txt
Filesystem      Size  Used Avail Use% Mounted on
/dev/vda1       50G  7.2G  40G  16% /
devtmpfs        7.7G   0  7.7G   0% /dev
tmpfs           7.7G  240K  7.7G   1% /dev/shm
tmpfs           7.7G  456K  7.7G   1% /run
tmpfs           7.7G   0  7.7G   0% /sys/fs/cgroup
/dev/vdb1       99G  2.2G  92G   3% /data
/dev/vdc1       99G  2.2G  92G   3% /backup
/dev/vdd1       99G   93M  94G   1% /archive
tmpfs           1.6G   0  1.6G   0% /run/user/0
Filesystem      Size  Used Avail Use% Mounted on
/dev/vdb1       99G  2.2G  92G   3% /data
[postgres@dbp-01 HealthCheck ~]$ df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/vda1       50G  7.3G  40G  16% /
devtmpfs        7.7G   0  7.7G   0% /dev
tmpfs           7.7G  240K  7.7G   1% /dev/shm
tmpfs           7.7G  428K  7.7G   1% /run
tmpfs           7.7G   0  7.7G   0% /sys/fs/cgroup
/dev/vdb1       99G  2.4G  91G   3% /data
/dev/vdc1       99G  1.1G  93G   2% /backup
/dev/vdd1       99G  125M  94G   1% /archive
tmpfs           1.6G   0  1.6G   0% /run/user/0
[postgres@dbp-01 HealthCheck ~]$ df -h > df-h.txt
[postgres@dbp-01 HealthCheck ~]$ df -h /data/partner/data/ >> d
df-h.txt

```

Gambar 3.2.1. 10 Pengambilan Data *Disk Free* pada Server Partner Production 01

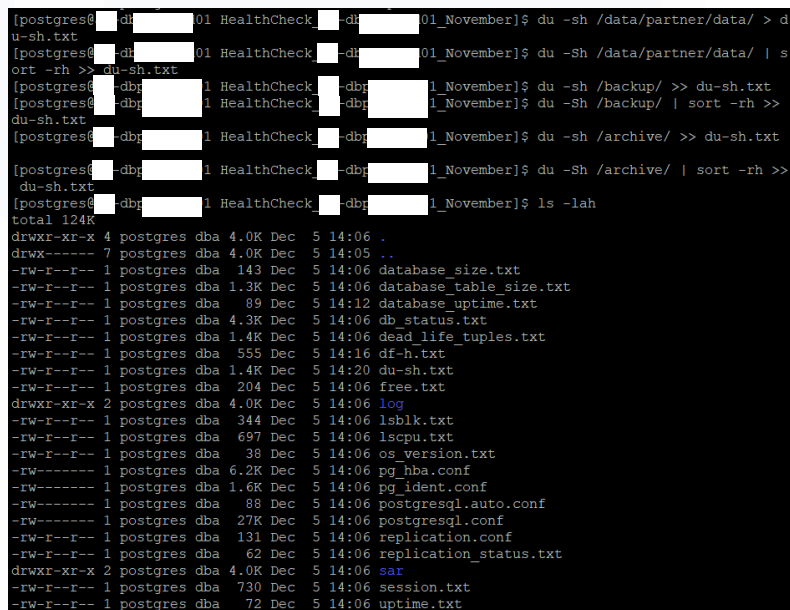


## 6) Data disk usage

Data yang diambil pada bagian server selanjutnya adalah data untuk melihat penggunaan *disk* atau *disk usage* pada server Partner Production 01 (XX-dbp\*\*\*\*\*01). Untuk pengambilan data tersebut menggunakan *command*:

```
du -sh /data/partner/data/ > du-sh.txt
du -Sh /data/partner/data | sort -rh >> du-sh.txt
```

Dari *command* tersebut data *disk usage* akan dimasukkan ke dalam *file* du-sh.txt, dimana akan dilakukan *sort* juga terhadap *disk usage* yang digunakan pada server Partner Production 01 (XX-dbp\*\*\*\*\*01).



```
[postgres@dbp*****01 HealthCheck] du -sh /data/partner/data/ > du-sh.txt
[postgres@dbp*****01 HealthCheck] du -Sh /data/partner/data | sort -rh >> du-sh.txt
[postgres@dbp*****01 HealthCheck] du -sh /backup/ >> du-sh.txt
[postgres@dbp*****01 HealthCheck] du -Sh /backup/ | sort -rh >> du-sh.txt
[postgres@dbp*****01 HealthCheck] du -sh /archive/ >> du-sh.txt
[postgres@dbp*****01 HealthCheck] du -Sh /archive/ | sort -rh >> du-sh.txt
[postgres@dbp*****01 HealthCheck] ls -lah
total 124K
drwxr-xr-x 4 postgres dba 4.0K Dec  5 14:06 .
drwx----- 7 postgres dba 4.0K Dec  5 14:05 ..
-rw-r--r-- 1 postgres dba 143 Dec  5 14:06 database_size.txt
-rw-r--r-- 1 postgres dba 1.3K Dec  5 14:06 database_table_size.txt
-rw-r--r-- 1 postgres dba  89 Dec  5 14:12 database_uptime.txt
-rw-r--r-- 1 postgres dba 4.3K Dec  5 14:06 db_status.txt
-rw-r--r-- 1 postgres dba 1.4K Dec  5 14:06 dead_life_tuples.txt
-rw-r--r-- 1 postgres dba 555 Dec  5 14:16 df-h.txt
-rw-r--r-- 1 postgres dba 1.4K Dec  5 14:20 du-sh.txt
-rw-r--r-- 1 postgres dba 204 Dec  5 14:06 free.txt
drwxr-xr-x 2 postgres dba 4.0K Dec  5 14:06 log
-rw-r--r-- 1 postgres dba 344 Dec  5 14:06 lsbk.txt
-rw-r--r-- 1 postgres dba 697 Dec  5 14:06 lscpu.txt
-rw-r--r-- 1 postgres dba  38 Dec  5 14:06 os_version.txt
-rw----- 1 postgres dba 6.2K Dec  5 14:06 pg_hba.conf
-rw----- 1 postgres dba 1.6K Dec  5 14:06 pg_ident.conf
-rw----- 1 postgres dba  88 Dec  5 14:06 postgresql.auto.conf
-rw----- 1 postgres dba 27K Dec  5 14:06 postgresql.conf
-rw-r--r-- 1 postgres dba 131 Dec  5 14:06 replication.conf
-rw-r--r-- 1 postgres dba  62 Dec  5 14:06 replication_status.txt
drwxr-xr-x 2 postgres dba 4.0K Dec  5 14:06 sar
-rw-r--r-- 1 postgres dba 730 Dec  5 14:06 session.txt
-rw-r--r-- 1 postgres dba  72 Dec  5 14:06 uptime.txt
```

Gambar 3.2.1. 11 Pengambilan Data *Disk Usage* pada Server Partner Production 01

## 7) Data free memory

Selanjutnya dilakukan pengambilan data menggunakan *command* free -h, dimana *command* tersebut berguna untuk melihat total penggunaan yang *free* dan *used* oleh *physical* dan *swap* memori pada sistem atau biasa disebut dengan *buffer* dengan pada server *database* Partner Production 01

(XX-dbp\*\*\*\*\*01). Data tersebut dimasukkan kedalam *file* free.txt.

```
[postgres@XX-dbp*****01 HealthCheck]# free -h > free.txt
[postgres@XX-dbp*****01 HealthCheck]# ls -lah
total 124K
drwxr-xr-x 4 postgres dba 4.0K Dec  5 14:06 .
drwx----- 7 postgres dba 4.0K Dec  5 14:05 ..
-rw-r--r-- 1 postgres dba 143 Dec  5 14:06 database_size.txt
-rw-r--r-- 1 postgres dba 1.3K Dec  5 14:06 database_table_size.txt
-rw-r--r-- 1 postgres dba 89 Dec  5 14:12 database_uptime.txt
-rw-r--r-- 1 postgres dba 4.3K Dec  5 14:06 db_status.txt
-rw-r--r-- 1 postgres dba 1.4K Dec  5 14:06 dead_life_tuples.txt
-rw-r--r-- 1 postgres dba 555 Dec  5 14:16 df-h.txt
-rw-r--r-- 1 postgres dba 1.4K Dec  5 14:20 du-sh.txt
-rw-r--r-- 1 postgres dba 204 Dec  5 14:22 free.txt
drwxr-xr-x 2 postgres dba 4.0K Dec  5 14:06 log
-rw-r--r-- 1 postgres dba 344 Dec  5 14:06 lsblk.txt
-rw-r--r-- 1 postgres dba 697 Dec  5 14:06 lscpu.txt
-rw-r--r-- 1 postgres dba 38 Dec  5 14:06 os_version.txt
-rw----- 1 postgres dba 6.2K Dec  5 14:06 pg_hba.conf
-rw----- 1 postgres dba 1.6K Dec  5 14:06 pg_ident.conf
-rw----- 1 postgres dba 88 Dec  5 14:06 postgresql.auto.conf
-rw----- 1 postgres dba 27K Dec  5 14:06 postgresql.conf
-rw-r--r-- 1 postgres dba 131 Dec  5 14:06 replication.conf
-rw-r--r-- 1 postgres dba 62 Dec  5 14:06 replication_status.txt
drwxr-xr-x 2 postgres dba 4.0K Dec  5 14:06 sar
-rw-r--r-- 1 postgres dba 730 Dec  5 14:06 session.txt
-rw-r--r-- 1 postgres dba 72 Dec  5 14:06 uptime.txt
[postgres@af-dbpartord01 HealthCheck]#
```

Gambar 3.2.1. 12 Pengambilan Data *Free Memory* pada Server Partner Production 01

## 8) Data *block device*

Untuk keperluan *monthly monitoring* diperlukan pengambilan data untuk *block device* yang ada pada sistem *database* server Partner Production 01 (XX-dbp\*\*\*\*\*01), yang dimana data tersebut dimasukkan kedalam *file* lsblk.txt. Untuk mengambil data tersebut menggunakan *command* lsblk.

```
[postgres@XX-dbp*****01 HealthCheck]# lsblk > lsblk.txt
[postgres@XX-dbp*****01 HealthCheck]# ls -lah
total 124K
drwxr-xr-x 4 postgres dba 4.0K Dec  5 14:06 .
drwx----- 7 postgres dba 4.0K Dec  5 14:05 ..
-rw-r--r-- 1 postgres dba 143 Dec  5 14:06 database_size.txt
-rw-r--r-- 1 postgres dba 1.3K Dec  5 14:06 database_table_size.txt
-rw-r--r-- 1 postgres dba 89 Dec  5 14:12 database_uptime.txt
-rw-r--r-- 1 postgres dba 4.3K Dec  5 14:06 db_status.txt
-rw-r--r-- 1 postgres dba 1.4K Dec  5 14:06 dead_life_tuples.txt
-rw-r--r-- 1 postgres dba 555 Dec  5 14:16 df-h.txt
-rw-r--r-- 1 postgres dba 1.4K Dec  5 14:20 du-sh.txt
-rw-r--r-- 1 postgres dba 204 Dec  5 14:22 free.txt
drwxr-xr-x 2 postgres dba 4.0K Dec  5 14:06 log
-rw-r--r-- 1 postgres dba 344 Dec  5 14:23 lsblk.txt
-rw-r--r-- 1 postgres dba 697 Dec  5 14:06 lscpu.txt
-rw-r--r-- 1 postgres dba 38 Dec  5 14:06 os_version.txt
-rw----- 1 postgres dba 6.2K Dec  5 14:06 pg_hba.conf
-rw----- 1 postgres dba 1.6K Dec  5 14:06 pg_ident.conf
-rw----- 1 postgres dba 88 Dec  5 14:06 postgresql.auto.conf
-rw----- 1 postgres dba 27K Dec  5 14:06 postgresql.conf
-rw-r--r-- 1 postgres dba 131 Dec  5 14:06 replication.conf
-rw-r--r-- 1 postgres dba 62 Dec  5 14:06 replication_status.txt
drwxr-xr-x 2 postgres dba 4.0K Dec  5 14:06 sar
-rw-r--r-- 1 postgres dba 730 Dec  5 14:06 session.txt
-rw-r--r-- 1 postgres dba 72 Dec  5 14:06 uptime.txt
```

Gambar 3.2.1. 13 Pengambilan Data *Block Device* pada Server Partner Production 01

## 9) Data informasi CPU

Selain *block device*, diperlukan pengambilan data untuk melihat informasi mengenai arsitektur CPU yang digunakan pada server *database* Partner Production 01 (XX-dbp\*\*\*\*\*01), dimana *file* untuk informasi tersebut disimpan ke dalam *file* *lscpu.txt*. Untuk mendapatkan data tersebut menggunakan *command* *lscpu*.

```
[postgres@XX-dbp*****01 HealthCheck_XX-dbp*****01_November]$ lscpu > lscpu.txt
[postgres@XX-dbp*****01 HealthCheck_XX-dbp*****01_November]$ ls -lah
total 124K
drwxr-xr-x 4 postgres dba 4.0K Dec  5 14:06 .
drwx----- 7 postgres dba 4.0K Dec  5 14:05 ..
-rw-r--r-- 1 postgres dba 143 Dec  5 14:06 database_size.txt
-rw-r--r-- 1 postgres dba 1.3K Dec  5 14:06 database_table_size.txt
-rw-r--r-- 1 postgres dba 89 Dec  5 14:12 database_uptime.txt
-rw-r--r-- 1 postgres dba 4.3K Dec  5 14:06 db_status.txt
-rw-r--r-- 1 postgres dba 1.4K Dec  5 14:06 dead_life_tuples.txt
-rw-r--r-- 1 postgres dba 555 Dec  5 14:16 df-h.txt
-rw-r--r-- 1 postgres dba 1.4K Dec  5 14:20 du-sh.txt
-rw-r--r-- 1 postgres dba 204 Dec  5 14:22 free.txt
drwxr-xr-x 2 postgres dba 4.0K Dec  5 14:06 log
-rw-r--r-- 1 postgres dba 344 Dec  5 14:23 lsblk.txt
-rw-r--r-- 1 postgres dba 697 Dec  5 14:24 lscpu.txt
-rw-r--r-- 1 postgres dba 38 Dec  5 14:06 os_version.txt
-rw----- 1 postgres dba 6.2K Dec  5 14:06 pg_hba.conf
-rw----- 1 postgres dba 1.6K Dec  5 14:06 pg_ident.conf
-rw----- 1 postgres dba 88 Dec  5 14:06 postgresql.auto.conf
-rw----- 1 postgres dba 27K Dec  5 14:06 postgresql.conf
-rw-r--r-- 1 postgres dba 131 Dec  5 14:06 replication.conf
-rw-r--r-- 1 postgres dba 62 Dec  5 14:06 replication_status.txt
drwxr-xr-x 2 postgres dba 4.0K Dec  5 14:06 sar
-rw-r--r-- 1 postgres dba 730 Dec  5 14:06 session.txt
-rw-r--r-- 1 postgres dba 72 Dec  5 14:06 uptime.txt
```

Gambar 3.2.1. 14 Pengambilan Data Arsitektur CPU pada Server Partner Production 01

## 10) Data status replikasi

Selanjutnya dilakukan pengambilan data pada *database*, menggunakan *command* berikut,

```
select * from pg_stat_replication
select pg_is_in_recovery();
```

*command* tersebut digunakan untuk melihat bagaimana status replikasi dan bagaimana status *recovery* dari server *database* Partner Production 01 (XX-dbp\*\*\*\*\*01), dimana data tersebut masuk ke dalam *file* *replication\_status.txt*. Untuk melihat data yang diambil sudah masuk ke dalam *file* juga dapat menggunakan *command* *cat*.

```

postgres=# \o replication_status.txt
postgres=# \x
Expanded display is on.
postgres=# select * from pg_stat_replication;
postgres=# \x
Expanded display is off.
postgres=# select pg_is_in_recovery();
postgres=# \o
postgres=# \o
[postgres@XX-XXXXXXXXXX01 HealthCheck-XXXXXXXXXXdb:XXXXXXXXXX01_November]$ cat replication_status.txt
-[ RECORD 1 ]-----+-----
pid                | 25521
usesysid           | 17438
username           | replicator
application_name   | walreceiver
client_addr        | 10.91.1.14
client_hostname    |
client_port        | 50878
backend_start      | 2022-11-23 08:08:42.670662+07
backend_xmin       |
state              | streaming
sent_lsn           | 0/AB28B6F0
write_lsn          | 0/AB28B6F0
flush_lsn          | 0/AB28B6F0
replay_lsn         | 0/AB28B6F0
write_lag          | 00:00:00.015723
flush_lag          | 00:00:00.016355
replay_lag         | 00:00:00.016562
sync_priority      | 0
sync_state         | async
reply_time         | 2022-12-05 14:26:45.825873+07

pg_is_in_recovery
-----
f
(1 row)

```

Gambar 3.2.1. 15 Pengambilan Data Status Replikasi pada *Database* Server Partner Production 01

## 11) Data session

Selanjutnya data diambil pada bagian *database* server Partner Production 01 (XX-dbp\*\*\*\*\*01) dengan menggunakan *command*

```

select count(*),state from pg_stat_activity group by 2;
select count(*),username,datname,state,client_addr from
pg_stat_activity group by 2,3,4,5;

```

*command* tersebut berguna untuk mengambil data *session* untuk melihat jumlah aktivitas yang sedang berlangsung pada setiap proses yang terdapat pada *database* server. Data tersebut masuk ke dalam *file* session.txt.

```

postgres=# \o session.txt
postgres=# select count(*),state from pg_stat_activity group by 2;
postgres=# \o
postgres=# \q
[postgres@XX-dbp-*****01 HealthCheck XX-dbp-*****01_November]$ ls -lah
total 124K
drwxr-xr-x 4 postgres dba 4.0K Dec 5 14:06 .
drwx----- 7 postgres dba 4.0K Dec 5 14:05 ..
-rw-r--r-- 1 postgres dba 143 Dec 5 14:06 database_size.txt
-rw-r--r-- 1 postgres dba 1.3K Dec 5 14:06 database_table_size.txt
-rw-r--r-- 1 postgres dba 89 Dec 5 14:12 database_uptime.txt
-rw-r--r-- 1 postgres dba 4.3K Dec 5 14:06 db_status.txt
-rw-r--r-- 1 postgres dba 1.4K Dec 5 14:06 dead_life_tuples.txt
-rw-r--r-- 1 postgres dba 555 Dec 5 14:16 df-h.txt
-rw-r--r-- 1 postgres dba 1.4K Dec 5 14:20 du-sh.txt
-rw-r--r-- 1 postgres dba 204 Dec 5 14:22 free.txt
drwxr-xr-x 2 postgres dba 4.0K Dec 5 14:06 log
-rw-r--r-- 1 postgres dba 344 Dec 5 14:23 lsblk.txt
-rw-r--r-- 1 postgres dba 697 Dec 5 14:24 lscpu.txt
-rw-r--r-- 1 postgres dba 38 Dec 5 14:06 os_version.txt
-rw----- 1 postgres dba 6.2K Dec 5 14:06 pg_hba.conf
-rw----- 1 postgres dba 1.6K Dec 5 14:06 pg_ident.conf
-rw----- 1 postgres dba 88 Dec 5 14:06 postgresql.auto.conf
-rw----- 1 postgres dba 27K Dec 5 14:06 postgresql.conf
-rw-r--r-- 1 postgres dba 131 Dec 5 14:06 replication.conf
-rw-r--r-- 1 postgres dba 706 Dec 5 14:27 replication_status.txt
drwxr-xr-x 2 postgres dba 4.0K Dec 5 14:06 sar
-rw-r--r-- 1 postgres dba 973 Dec 5 14:51 session.txt
-rw-r--r-- 1 postgres dba 72 Dec 5 14:06 uptime.txt

```

Gambar 3.2.1. 16 Pengambilan Data *Session* pada *Database Server Partner Production 01*

## 12) Data *database size*

Pengambilan data masih dilakukan pada *database* yaitu untuk melihat *database size* pada server *database Partner Production 01 (XX-dbp\*\*\*\*\*01)*, data yang diambil tersebut dimasukkan ke dalam *file database\_size.txt*.

```

postgres=# \o database_size.txt
postgres=# SELECT
pg_database.datname,
pg_size_pretty(pg_database_size(pg_database.datname)) AS size
FROM pg_database;
postgres=# \o
postgres=# \q
[postgres@XX-dbp-*****01 HealthCheck XX-dbp-*****01_November]$ ls -lah
total 124K
drwxr-xr-x 4 postgres dba 4.0K Dec 5 14:06 .
drwx----- 7 postgres dba 4.0K Dec 5 14:05 ..
-rw-r--r-- 1 postgres dba 143 Dec 5 14:53 database_size.txt
-rw-r--r-- 1 postgres dba 1.3K Dec 5 14:06 database_table_size.txt
-rw-r--r-- 1 postgres dba 89 Dec 5 14:12 database_uptime.txt
-rw-r--r-- 1 postgres dba 4.3K Dec 5 14:06 db_status.txt
-rw-r--r-- 1 postgres dba 1.4K Dec 5 14:06 dead_life_tuples.txt
-rw-r--r-- 1 postgres dba 555 Dec 5 14:16 df-h.txt
-rw-r--r-- 1 postgres dba 1.4K Dec 5 14:20 du-sh.txt
-rw-r--r-- 1 postgres dba 204 Dec 5 14:22 free.txt
drwxr-xr-x 2 postgres dba 4.0K Dec 5 14:06 log
-rw-r--r-- 1 postgres dba 344 Dec 5 14:23 lsblk.txt
-rw-r--r-- 1 postgres dba 697 Dec 5 14:24 lscpu.txt
-rw-r--r-- 1 postgres dba 38 Dec 5 14:06 os_version.txt
-rw----- 1 postgres dba 6.2K Dec 5 14:06 pg_hba.conf
-rw----- 1 postgres dba 1.6K Dec 5 14:06 pg_ident.conf
-rw----- 1 postgres dba 88 Dec 5 14:06 postgresql.auto.conf
-rw----- 1 postgres dba 27K Dec 5 14:06 postgresql.conf
-rw-r--r-- 1 postgres dba 131 Dec 5 14:06 replication.conf
-rw-r--r-- 1 postgres dba 706 Dec 5 14:27 replication_status.txt
drwxr-xr-x 2 postgres dba 4.0K Dec 5 14:06 sar
-rw-r--r-- 1 postgres dba 973 Dec 5 14:51 session.txt
-rw-r--r-- 1 postgres dba 72 Dec 5 14:06 uptime.txt

```

Gambar 3.2.1. 17 Pengambilan Data *Database Size* pada *Database Server Partner Production 01*

### 13) Data *table size*

Setelah mengambil data untuk melihat informasi terkait *database size*, dilakukan pengambilan data untuk melihat informasi terkait *database table size*, setiap *table* pada tiap-tiap *database* pada server. Data untuk mengambil *database table size* dilakukan untuk setiap *database* yang terdapat pada server Partner Production 01 (XX-dbp\*\*\*\*\*01). Data tersebut akan masuk ke dalam file *database\_table\_size.txt*.

```
postgres=# \o database_table_size.txt
postgres=# \t
postgres=# select 'postgres';
postgres=# \t
postgres=# SELECT relkind, nspname || '.' || relname AS "relation", pg_size_pretty(pg_total_relation_size(C.oid)) AS "total_size" FROM pg_class
LEFT JOIN pg_namespace N ON (N.oid = C.relnamespace)
WHERE nspname NOT IN ('pg_catalog', 'information_schema')
AND C.relkind <> 'i'
AND nspname != 'pg_toast'
ORDER BY pg_total_relation_size(C.oid) DESC;
postgres=# \o [redacted]
You are now connected to database "[redacted]" as user "postgres".
[redacted]=# \t
[redacted]=# select 'postgres';
[redacted]=# \t
[redacted]=# SELECT relkind, nspname || '.' || relname AS "relation", pg_size_pretty(pg_total_relation_size(C.oid)) AS "total_size" FROM pg_class
LEFT JOIN pg_namespace N ON (N.oid = C.relnamespace)
WHERE nspname NOT IN ('pg_catalog', 'information_schema')
AND C.relkind <> 'i'
AND nspname != 'pg_toast'
ORDER BY pg_total_relation_size(C.oid) DESC;
[redacted]=# \o
[redacted]=# \t
```

Gambar 3.2.1. 18 Pengambilan Data *Database Table Size* pada *Database Server Partner Production 01*

### 14) Data *dead life tuples*

Serupa dengan pengambilan *database table size*, dilakukan juga pengambilan data untuk untuk mendapatkan data *dead life tuples* seluruh *table* pada seluruh *database* yang terdapat pada server *database* Partner Production 01 (XX-dbp\*\*\*\*\*01).

```
postgres=# \o dead_life_tuples.txt
postgres=# \t
postgres=# select 'postgres';
postgres=# \t
postgres=# SELECT schemaname, relname AS TableName, n_live_tup AS LiveTuples, n_dead_tup AS DeadTuple FROM pg_stat_user_tables;
postgres=# \o [redacted]
You are now connected to database "[redacted] Partner" as user "postgres".
[redacted]=# \t
[redacted]=# select 'postgres';
[redacted]=# \t
[redacted]=# SELECT schemaname, relname AS TableName, n_live_tup AS LiveTuples, n_dead_tup AS DeadTuple FROM pg_stat_user_tables;
[redacted]=# \o
[redacted]=# \t
```

Gambar 3.2.1. 19 Pengambilan Data *Dead Life Tuples* pada *Database Server Partner Production 01*

## 15) Data log dan SAR

Setelah semua data-data pada server dan *database* sudah diambil maka akan dilakukan pengambilan data untuk *log* dan SAR selama bulan November, dimana untuk pengambilan data *log* dan SAR mahasiswa diharapkan untuk berhati-hati agar tidak salah menghapus *file* karena terdapat *command* untuk menghapus *file* pada server. Sebelum menghapus *file* bulan lalu maka dilakukan *list* terlebih dahulu terhadap *folder log* dengan menggunakan *command ls*, setelah dirasa *list* yang terlihat tidak berbahaya untuk dihapus maka akan dilakukan penghapusan seluruh *file* pada *folder log*, menggunakan *command rm -rf*.

```
postgres=# \! healthcheck -d - November 11 log/
total 784
drwxr-xr-x 1 postgres dba 2339 Dec  3 14:06 postgresql-2022-10-01_000000_log
drwxr-xr-x 1 postgres dba 3513 Dec  3 14:06 postgresql-2022-10-02_000000_log
drwxr-xr-x 1 postgres dba 4339 Dec  3 14:06 postgresql-2022-10-03_000000_log
drwxr-xr-x 1 postgres dba 13059 Dec  3 14:06 postgresql-2022-10-04_000000_log
drwxr-xr-x 1 postgres dba 18369 Dec  3 14:06 postgresql-2022-10-05_000000_log
drwxr-xr-x 1 postgres dba 18553 Dec  3 14:06 postgresql-2022-10-06_000000_log
drwxr-xr-x 1 postgres dba 23329 Dec  3 14:06 postgresql-2022-10-07_000000_log
drwxr-xr-x 1 postgres dba 4693 Dec  3 14:06 postgresql-2022-10-08_000000_log
drwxr-xr-x 1 postgres dba 9 Dec  3 14:06 postgresql-2022-10-09_000000_log
drwxr-xr-x 1 postgres dba 6301 Dec  3 14:06 postgresql-2022-10-10_000000_log
drwxr-xr-x 1 postgres dba 15569 Dec  3 14:06 postgresql-2022-10-11_000000_log
drwxr-xr-x 1 postgres dba 1544 Dec  3 14:06 postgresql-2022-10-12_000000_log
drwxr-xr-x 1 postgres dba 7140 Dec  3 14:06 postgresql-2022-10-13_000000_log
drwxr-xr-x 1 postgres dba 25264 Dec  3 14:06 postgresql-2022-10-14_000000_log
drwxr-xr-x 1 postgres dba 9 Dec  3 14:06 postgresql-2022-10-15_000000_log
drwxr-xr-x 1 postgres dba 1374 Dec  3 14:06 postgresql-2022-10-16_000000_log
drwxr-xr-x 1 postgres dba 11020 Dec  3 14:06 postgresql-2022-10-17_000000_log
drwxr-xr-x 1 postgres dba 1481 Dec  3 14:06 postgresql-2022-10-18_000000_log
drwxr-xr-x 1 postgres dba 7521 Dec  3 14:06 postgresql-2022-10-19_000000_log
drwxr-xr-x 1 postgres dba 174 Dec  3 14:06 postgresql-2022-10-20_000000_log
drwxr-xr-x 1 postgres dba 26839 Dec  3 14:06 postgresql-2022-10-21_000000_log
drwxr-xr-x 1 postgres dba 2629 Dec  3 14:06 postgresql-2022-10-22_000000_log
drwxr-xr-x 1 postgres dba 7009 Dec  3 14:06 postgresql-2022-10-23_000000_log
drwxr-xr-x 1 postgres dba 1644 Dec  3 14:06 postgresql-2022-10-24_000000_log
drwxr-xr-x 1 postgres dba 9794 Dec  3 14:06 postgresql-2022-10-25_000000_log
drwxr-xr-x 1 postgres dba 6481 Dec  3 14:06 postgresql-2022-10-26_000000_log
drwxr-xr-x 1 postgres dba 8696 Dec  3 14:06 postgresql-2022-10-27_000000_log
drwxr-xr-x 1 postgres dba 41790 Dec  3 14:06 postgresql-2022-10-28_000000_log
drwxr-xr-x 1 postgres dba 42848 Dec  3 14:06 postgresql-2022-10-29_000000_log
drwxr-xr-x 1 postgres dba 9368 Dec  3 14:06 postgresql-2022-10-30_000000_log
drwxr-xr-x 1 postgres dba 3774 Dec  3 14:06 postgresql-2022-10-31_000000_log
postgres=# \! healthcheck -d - November 11 log/postgresql-2022-10-
log/postgresql-2022-10-01_000000_log log/postgresql-2022-10-15_000000_log log/postgresql-2022-10-29_000000_log
log/postgresql-2022-10-02_000000_log log/postgresql-2022-10-09_000000_log log/postgresql-2022-10-16_000000_log log/postgresql-2022-10-21_000000_log
log/postgresql-2022-10-03_000000_log log/postgresql-2022-10-12_000000_log log/postgresql-2022-10-19_000000_log log/postgresql-2022-10-24_000000_log
log/postgresql-2022-10-04_000000_log log/postgresql-2022-10-11_000000_log log/postgresql-2022-10-18_000000_log log/postgresql-2022-10-25_000000_log
log/postgresql-2022-10-05_000000_log log/postgresql-2022-10-13_000000_log log/postgresql-2022-10-20_000000_log log/postgresql-2022-10-26_000000_log
log/postgresql-2022-10-06_000000_log log/postgresql-2022-10-14_000000_log log/postgresql-2022-10-21_000000_log log/postgresql-2022-10-27_000000_log
log/postgresql-2022-10-07_000000_log log/postgresql-2022-10-17_000000_log log/postgresql-2022-10-22_000000_log log/postgresql-2022-10-28_000000_log
log/postgresql-2022-10-08_000000_log log/postgresql-2022-10-18_000000_log log/postgresql-2022-10-23_000000_log log/postgresql-2022-10-30_000000_log
postgres=# \! healthcheck -d - November 11 sar/
sar/
messages-20221009 messages-20221016 sar/
sar/0 sar/1 sar/2 sar/3 sar/4 sar/5 sar/6 sar/7 sar/8 sar/9 sar/10 sar/11 sar/12 sar/13 sar/14 sar/15 sar/16 sar/17 sar/18 sar/19 sar/20 sar/21 sar/22 sar/23 sar/24 sar/25 sar/26 sar/27 sar/28 sar/29 sar/30
messages-20221024 sar/
sar/0 sar/1 sar/2 sar/3 sar/4 sar/5 sar/6 sar/7 sar/8 sar/9 sar/10 sar/11 sar/12 sar/13 sar/14 sar/15 sar/16 sar/17 sar/18 sar/19 sar/20 sar/21 sar/22 sar/23 sar/24 sar/25 sar/26 sar/27 sar/28 sar/29 sar/30
postgres=# \! healthcheck -d - November 11 sar/sar/
sar/sar/
sar/messages-20221009 sar/messages-20221016 sar/sar/0 sar/sar/1 sar/sar/2 sar/sar/3 sar/sar/4 sar/sar/5 sar/sar/6 sar/sar/7 sar/sar/8 sar/sar/9 sar/sar/10 sar/sar/11 sar/sar/12 sar/sar/13 sar/sar/14 sar/sar/15 sar/sar/16 sar/sar/17 sar/sar/18 sar/sar/19 sar/sar/20 sar/sar/21 sar/sar/22 sar/sar/23 sar/sar/24 sar/sar/25 sar/sar/26 sar/sar/27 sar/sar/28 sar/sar/29 sar/sar/30
sar/messages-20221024 sar/sar/0 sar/sar/1 sar/sar/2 sar/sar/3 sar/sar/4 sar/sar/5 sar/sar/6 sar/sar/7 sar/sar/8 sar/sar/9 sar/sar/10 sar/sar/11 sar/sar/12 sar/sar/13 sar/sar/14 sar/sar/15 sar/sar/16 sar/sar/17 sar/sar/18 sar/sar/19 sar/sar/20 sar/sar/21 sar/sar/22 sar/sar/23 sar/sar/24 sar/sar/25 sar/sar/26 sar/sar/27 sar/sar/28 sar/sar/29 sar/sar/30
postgres=# \! healthcheck -d - November 11 rm -r sar/sar/
rm -r sar/sar/
rm -r sar/messages*
```

Gambar 3.2.1. 20 Penghapusan Data Log Bulan Oktober pada Database Server Partner Production 01

Sama dengan *log* dilakukan hal yang sama untuk *folder SAR*, yaitu dengan melakukan *list* terlebih dahulu lalu menghapus seluruh *file* pada *folder SAR*.

```
postgre=# \! healthcheck -d - November 11 sar/
messages-20221009 messages-20221016 sar/
sar/0 sar/1 sar/2 sar/3 sar/4 sar/5 sar/6 sar/7 sar/8 sar/9 sar/10 sar/11 sar/12 sar/13 sar/14 sar/15 sar/16 sar/17 sar/18 sar/19 sar/20 sar/21 sar/22 sar/23 sar/24 sar/25 sar/26 sar/27 sar/28 sar/29 sar/30
messages-20221024 sar/
sar/0 sar/1 sar/2 sar/3 sar/4 sar/5 sar/6 sar/7 sar/8 sar/9 sar/10 sar/11 sar/12 sar/13 sar/14 sar/15 sar/16 sar/17 sar/18 sar/19 sar/20 sar/21 sar/22 sar/23 sar/24 sar/25 sar/26 sar/27 sar/28 sar/29 sar/30
postgres=# \! healthcheck -d - November 11 sar/sar/
sar/sar/
sar/messages-20221009 sar/messages-20221016 sar/sar/0 sar/sar/1 sar/sar/2 sar/sar/3 sar/sar/4 sar/sar/5 sar/sar/6 sar/sar/7 sar/sar/8 sar/sar/9 sar/sar/10 sar/sar/11 sar/sar/12 sar/sar/13 sar/sar/14 sar/sar/15 sar/sar/16 sar/sar/17 sar/sar/18 sar/sar/19 sar/sar/20 sar/sar/21 sar/sar/22 sar/sar/23 sar/sar/24 sar/sar/25 sar/sar/26 sar/sar/27 sar/sar/28 sar/sar/29 sar/sar/30
sar/messages-20221024 sar/sar/0 sar/sar/1 sar/sar/2 sar/sar/3 sar/sar/4 sar/sar/5 sar/sar/6 sar/sar/7 sar/sar/8 sar/sar/9 sar/sar/10 sar/sar/11 sar/sar/12 sar/sar/13 sar/sar/14 sar/sar/15 sar/sar/16 sar/sar/17 sar/sar/18 sar/sar/19 sar/sar/20 sar/sar/21 sar/sar/22 sar/sar/23 sar/sar/24 sar/sar/25 sar/sar/26 sar/sar/27 sar/sar/28 sar/sar/29 sar/sar/30
postgres=# \! healthcheck -d - November 11 rm -r sar/sar/
rm -r sar/sar/
rm -r sar/messages*
```

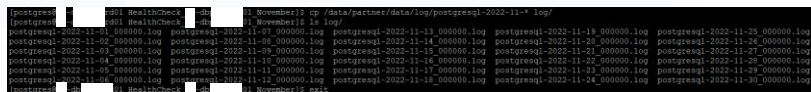
Gambar 3.2.1. 21 Penghapusan Data SAR Bulan Oktober pada Database Server Partner Production 01

Setelah semua *folder log* dan SAR sudah kosong, maka akan dilakukan pengambilan data *log* yang terdapat pada server yang terdapat pada direktori



/data/partner/data/log/postgresql-2022-11-\*, dimana *command* untuk pengambilan data tersebut adalah:

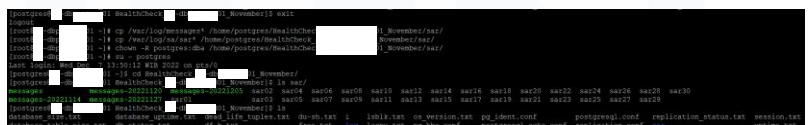
`cp /data/partner/data/log/postgresql-2022-11-* log/`  
*command* tersebut merupakan *command* untuk meng-copy seluruh *file log* pada bulan November (11) dan disalin ke dalam *folder log* yang pada direktori yang sudah dibuat sebelumnya.



```
postgres@db1:~$ cp /data/partner/data/log/postgresql-2022-11-* log/
postgres@db1:~$ ls -la log/
total 108
-rw-r--r-- 1 postgres postgres 11-19-000000.log
-rw-r--r-- 1 postgres postgres 11-20-000000.log
-rw-r--r-- 1 postgres postgres 11-21-000000.log
-rw-r--r-- 1 postgres postgres 11-22-000000.log
-rw-r--r-- 1 postgres postgres 11-23-000000.log
-rw-r--r-- 1 postgres postgres 11-24-000000.log
-rw-r--r-- 1 postgres postgres 11-25-000000.log
-rw-r--r-- 1 postgres postgres 11-26-000000.log
-rw-r--r-- 1 postgres postgres 11-27-000000.log
-rw-r--r-- 1 postgres postgres 11-28-000000.log
-rw-r--r-- 1 postgres postgres 11-29-000000.log
-rw-r--r-- 1 postgres postgres 11-30-000000.log
```

Gambar 3.2.1. 22 Pengambilan Data Log pada Database Server Partner Production 01

Selanjutnya untuk data SAR didapatkan dengan meng-copy *file* pada direktori /var/log/messages\* dan /var/log/sa/sar\* ke dalam *folder SAR* pada direktori yang sudah dibuat sebelumnya. Setelah *file SAR* berhasil disalin maka diperlukan *command chown* untuk memberikan akses kepada *user postgres* pada *group dba* untuk mengakses *file* sistem. Setelah semua data *file* untuk *folder SAR* dan *log* telah diambil, maka dilakukan pengecekan dengan menggunakan *command ls*.



```
postgres@db1:~$ cp /var/log/messages* /home/postgres/SAR/
postgres@db1:~$ cp /var/log/sa/sar* /home/postgres/SAR/
postgres@db1:~$ chown -R postgres:dba /home/postgres/SAR/
postgres@db1:~$ ls -la /home/postgres/SAR/
total 12
-rw-r--r-- 1 postgres postgres 11-19-000000.log
-rw-r--r-- 1 postgres postgres 11-20-000000.log
-rw-r--r-- 1 postgres postgres 11-21-000000.log
-rw-r--r-- 1 postgres postgres 11-22-000000.log
-rw-r--r-- 1 postgres postgres 11-23-000000.log
-rw-r--r-- 1 postgres postgres 11-24-000000.log
-rw-r--r-- 1 postgres postgres 11-25-000000.log
-rw-r--r-- 1 postgres postgres 11-26-000000.log
-rw-r--r-- 1 postgres postgres 11-27-000000.log
-rw-r--r-- 1 postgres postgres 11-28-000000.log
-rw-r--r-- 1 postgres postgres 11-29-000000.log
-rw-r--r-- 1 postgres postgres 11-30-000000.log
```

Gambar 3.2.1. 23 Pengambilan SAR Uptime pada Database Server Partner Production 01

## 16) Data dimasukkan kedalam file .tar

Data-data yang telah diambil sebelumnya akan dimasukkan ke dalam satu *file .tar* sehingga lebih mudah untuk ditarik nantinya.



```

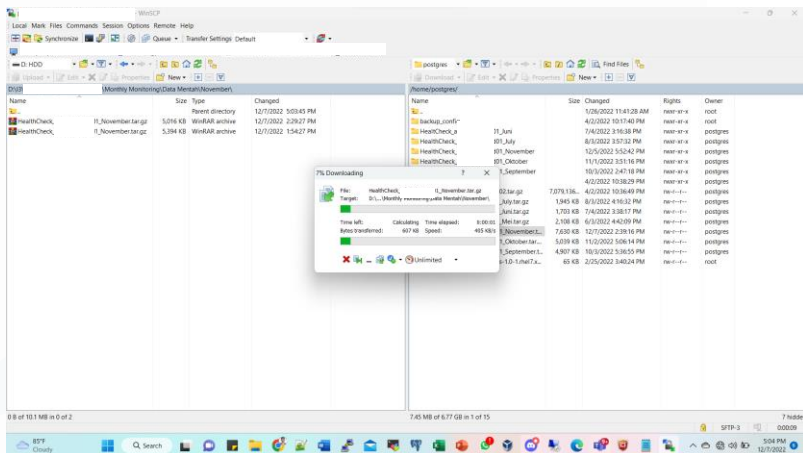
[postgres@h01 ~]$ tar -czvf HealthCheck_01_11_November.tar.gz HealthCheck_01_11_November/
HealthCheck_01_11_November/
HealthCheck_01_11_November/os_version.txt
HealthCheck_01_11_November/db_status.txt
HealthCheck_01_11_November/replication.conf
HealthCheck_01_11_November/database_size.txt
HealthCheck_01_11_November/pg_ident.conf
HealthCheck_01_11_November/psql.txt
HealthCheck_01_11_November/database_uptime.txt
HealthCheck_01_11_November/postgresql.auto.conf
HealthCheck_01_11_November/uptime.txt
HealthCheck_01_11_November/pg_hba.conf
HealthCheck_01_11_November/log/
HealthCheck_01_11_November/log/postgresql-2022-11-02_000000.log
HealthCheck_01_11_November/log/postgresql-2022-11-16_000000.log
HealthCheck_01_11_November/log/postgresql-2022-11-07_000000.log
HealthCheck_01_11_November/log/postgresql-2022-11-05_000000.log
HealthCheck_01_11_November/log/postgresql-2022-11-20_000000.log
HealthCheck_01_11_November/log/postgresql-2022-11-30_000000.log
HealthCheck_01_11_November/log/postgresql-2022-11-07_000000.log
HealthCheck_01_11_November/log/postgresql-2022-11-25_000000.log
HealthCheck_01_11_November/log/postgresql-2022-11-23_000000.log
HealthCheck_01_11_November/log/postgresql-2022-11-29_000000.log
HealthCheck_01_11_November/log/postgresql-2022-11-08_000000.log
HealthCheck_01_11_November/log/postgresql-2022-11-19_000000.log
HealthCheck_01_11_November/log/postgresql-2022-11-26_000000.log
HealthCheck_01_11_November/log/postgresql-2022-11-04_000000.log
HealthCheck_01_11_November/log/postgresql-2022-11-11_000000.log
HealthCheck_01_11_November/log/postgresql-2022-11-21_000000.log
HealthCheck_01_11_November/log/postgresql-2022-11-24_000000.log
HealthCheck_01_11_November/log/postgresql-2022-11-06_000000.log
HealthCheck_01_11_November/log/postgresql-2022-11-01_000000.log
HealthCheck_01_11_November/log/postgresql-2022-11-09_000000.log
HealthCheck_01_11_November/log/postgresql-2022-11-14_000000.log
HealthCheck_01_11_November/log/postgresql-2022-11-03_000000.log
HealthCheck_01_11_November/log/postgresql-2022-11-15_000000.log
HealthCheck_01_11_November/log/postgresql-2022-11-13_000000.log
HealthCheck_01_11_November/log/postgresql-2022-11-17_000000.log
HealthCheck_01_11_November/log/postgresql-2022-11-22_000000.log
HealthCheck_01_11_November/log/postgresql-2022-11-18_000000.log
HealthCheck_01_11_November/log/postgresql-2022-11-28_000000.log
HealthCheck_01_11_November/log/postgresql-2022-11-27_000000.log
HealthCheck_01_11_November/log/postgresql-2022-11-12_000000.log
HealthCheck_01_11_November/database_table_size.txt
HealthCheck_01_11_November/postgresql.conf

```

Gambar 3.2.1. 24 Seluruh Data yang telah diambil pada Database Server Partner Production 01 dijadikan ke dalam 1 File .tar

### 17) Penarikan data

Setelah semua data pada server Partner Production 01 (XX-dbp\*\*\*\*\*01) sudah masuk ke dalam satu file .tar, maka penarikan data akan dilakukan menggunakan WinSCP.

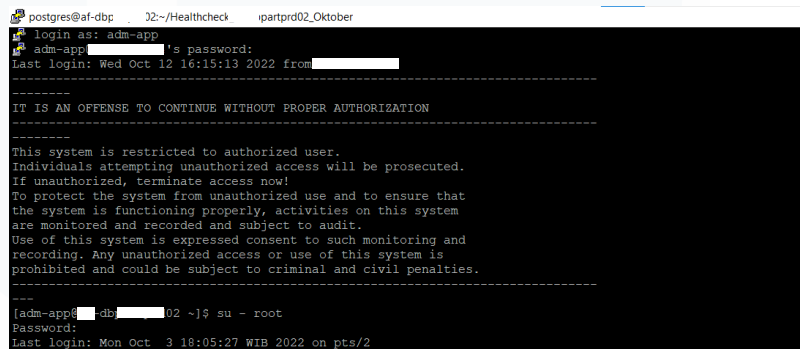


Gambar 3.2.1. 25 Penarikan Data pada Database Server Partner Production 01 Menggunakan WinSCP

## b. Pengambilan Data: Server *Slave* – Partner Production 02 (XX-dbp\*\*\*\*\*02)

### 1) *Login*

Dalam *monthly monitoring* yang dilakukan untuk satu server *slave* yaitu server Partner Production 02 (XX-dbp\*\*\*\*\*02) pada perusahaan XYZ, digunakan PuTTY untuk mengecek server *database* klien, dimana untuk mengakses diperlukan IP Address dan juga *password* untuk user adm-app.



```
postgres@af-dbp: ~2~/Healthcheck ipartprd02_Oktober
login as: adm-app
adm-app@XX-dbp*****02:~$ su - root
Last login: Wed Oct 12 16:15:13 2022 from [redacted]
-----
IT IS AN OFFENSE TO CONTINUE WITHOUT PROPER AUTHORIZATION
-----
This system is restricted to authorized user.
Individuals attempting unauthorized access will be prosecuted.
If unauthorized, terminate access now!
To protect the system from unauthorized use and to ensure that
the system is functioning properly, activities on this system
are monitored and recorded and subject to audit.
Use of this system is expressed consent to such monitoring and
recording. Any unauthorized access or use of this system is
prohibited and could be subject to criminal and civil penalties.
-----
[adm-app@XX-dbp*****02 ~]$ su - root
Password:
Last login: Mon Oct  3 18:05:27 WIB 2022 on pts/2
```

Gambar 3.2.1. 26 Login adm-app pada Server Partner Production 02

### 2) *Cek dan meng-update password*

Setelah masuk ke dalam *database* menggunakan user adm-app, maka perlu masuk ke dalam user postgres, dan mengecek *password* untuk user postgres. Dimana *password* untuk user postgres pada server *slave* Partner Production 02 (XX-dbp\*\*\*\*\*02) sudah *expired*, sehingga diperlukan pembaharuan terhadap *password*-nya.

```

[root@-dbp-02 ~]# su - postgres
Last login: Mon Dec 5 15:21:03 WIB 2022 on pts/0
[postgres@-dbp-02 ~]$ chage -l postgres
Last password change           : Nov 03, 2022
Password expires               : Dec 03, 2022
Password inactive              : never
Account expires                : never
Minimum number of days between password change : 0
Maximum number of days between password change : 30
Number of days of warning before password expires : 7
[postgres@-dbp-02 ~]$ exit
logout
[root@-dbp-02 ~]# passwd postgres
Changing password for user postgres.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
[root@-dbp-02 ~]# su - postgres
Last login: Wed Dec 7 14:01:18 WIB 2022 on pts/0
[postgres@-dbp-02 ~]$ chage -l postgres
Last password change           : Dec 07, 2022
Password expires               : Jan 06, 2023
Password inactive              : never
Account expires                : never
Minimum number of days between password change : 0
Maximum number of days between password change : 30
Number of days of warning before password expires : 7
[postgres@-dbp-02 ~]$

```

Gambar 3.2.1. 27 Checking Password untuk User Postgres pada Server Partner Production 02

### 3) Proses pengambilan data

Pada proses pengambilan data untuk *server slave*, sama seperti dengan server *master* yang dilakukan sebelumnya, dimana diperlukan pembuatan direktori baru terlebih dahulu sebelum memulai melakukan *monthly monitoring*, hal ini bertujuan agar tidak terjadinya kejadian yang tidak diinginkan seperti terhapusnya *file* atau data-data di bulan sebelumnya menjadi hilang. Selain itu agar data-data dapat disortir sesuai dengan bulan, sehingga lebih mudah dalam pencarian data.

```

[root@-dbp-02 ~]# su - postgres
Last login: Mon Oct 3 18:07:26 WIB 2022 on pts/2
[postgres@-dbp-02 ~]$ ls
Healthcheck_af      tprd02_July
Healthcheck_af      tprd02_July.tar.gz
Healthcheck_af      tprd02_Juni
Healthcheck_af      tprd02_Juni.tar.gz
Healthcheck_af      tprd02_Mei.tar.gz
Healthcheck_af      tprd02_September
Healthcheck_af      tprd02_September.tar.gz
[postgres@-dbp-02 ~]$ mkdir Healthcheck_-dbp-02_Oktober/
[postgres@-dbp-02 ~]$ cp -r Healthcheck_-dbp-02_September/* Healthcheck_-dbp-02_Oktober/
[postgres@-dbp-02 ~]$ cd Healthcheck_-dbp-02_Oktober/

```

Gambar 3.2.1. 28 Membuat Direktori Baru untuk Bulan November 2022 pada Server Partner Production 02

Setelah direktori yang baru sudah dibuat maka, maka perlu merubah direktori ke dalam direktori baru untuk

memulai pengambilan data untuk server *slave*, *command* yang digunakan untuk berpindah direktori adalah *cd*.

#### 4) Pengambilan data

Dalam pengambilan data yang dilakukan oleh mahasiswa untuk server *slave* Partner Production 02 (XX-dbp\*\*\*\*\*02), langkah-langkah yang dilakukan serupa dengan langkah-langkah yang telah dilakukan sebelumnya pada server *master* Partner Production 01 (XX-dbp\*\*\*\*\*01). Data-data yang diambil dari server *slave* untuk kebutuhan analisis adalah sebagai berikut:

- a) Uptime
- b) *Disk free*
- c) *Disk usage*
- d) *Free memory*
- e) *Block device*
- f) Arsitektur CPU
- g) *Replication status*
- h) *Session*

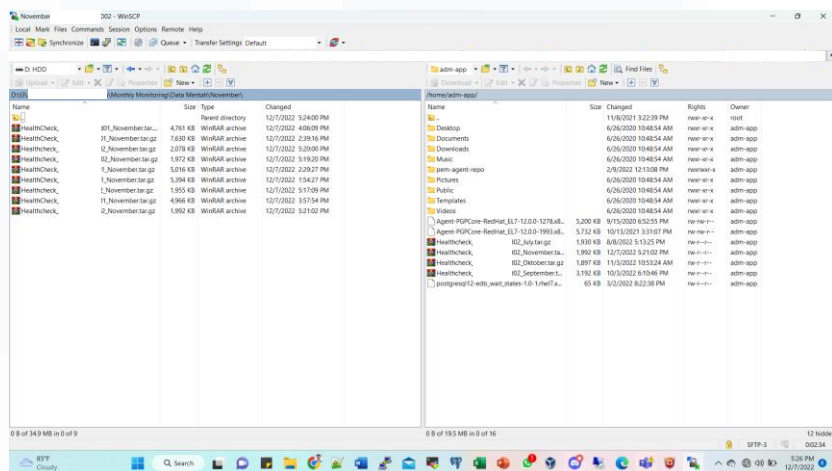
Setelah pengambilan data-data dari server *slave* telah dilakukan maka langkah selanjutnya adalah dengan menjadikan seluruh *file* tersebut kedalam satu *file* .tar, setelah dijadikan satu *file* maka akan dilakukan penarikan data. Namun terdapat perbedaan untuk penarikan data *file* .tar, hal ini dikarenakan dalam mengakses server *slave* tidak dapat menggunakan *super user* melainkan menggunakan *user* adm-app. Maka dari itu, diperlukan penyalinan *file* /home/postgres/Healthchekch\_XX-dbp\*\*\*\*\*02\_November.tar.gz ke dalam direktori /home/adm-app/

Setelah *file* tersalin ke dalam direktori untuk *user* adm-app, maka diperlukan *grant* akses agar *file* dapat di akses oleh *user* adm-app.

```
[root@db-2 ~]# cp /home/postgres/healthcheck-xx-dbp-11-November.tar.gz /home/adm-app/
[root@db-2 ~]# chown adm-app /home/adm-app/healthcheck-xx-dbp-11-November.tar.gz
```

Gambar 3.2.1. 29 Memberikan Akses *File* Untuk *User* adm-app

Setelah akses diberikan maka dapat dilakukan penarikan data menggunakan WinSCP, dimana data yang ditarik akan terdapat pada direktori /home/adm-app/.



Gambar 3.2.1. 30 Penarikan Data pada *Database* Server Partner Production 02 Menggunakan WinSCP

**c. Hasil pengambilan data pada server *master* (Partner Production 01 (XX-dbp\*\*\*\*\*01))**

Data yang sudah di ambil akan dimasukkan ke dalam *report* *monthly monitoring*. Dimana *report* tersebut memuat seluruh informasi penting dari 5 server *database* *master* yang dimiliki oleh perusahaan XYZ. Berikut merupakan hasil pengambilan data bulan November yang telah dilakukan sebelumnya untuk server Partner Production 01 (XX-dbp\*\*\*\*\*01):

## 1) *Memory server*

	total	used	free	shared	buff/cache	available
Mem:	15G	621M	3.9G	302M	10G	14G
Swap:	0B	0B	0B			

Gambar 3.2.1. 31 Hasil *Memory Server* untuk Server Partner Production 01

Pada Gambar 3.2.1.31 dapat terlihat untuk *memory* pada server Partner Production 01 (XX-dbp\*\*\*\*\*01), dimana hasil tersebut didapatkan dari pengambilan data menggunakan *command free -h* yang terdapat pada *file free.txt*. Dapat terlihat pada gambar tersebut *free memory* untuk server *master* Partner Production 01 adalah 3.9G.

## 2) *Mount point*

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/vda1	50G	7.3G	40G	16%	/
devtmpfs	7.7G	0	7.7G	0%	/dev
tmpfs	7.7G	240K	7.7G	1%	/dev/shm
tmpfs	7.7G	428K	7.7G	1%	/run
tmpfs	7.7G	0	7.7G	0%	/sys/fs/cgroup
/dev/vdb1	99G	2.4G	91G	3%	/data
/dev/vdc1	99G	1.1G	93G	2%	/backup
/dev/vdd1	99G	125M	94G	1%	/archive
tmpfs	1.6G	0	1.6G	0%	/run/user/0
Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/vdb1	99G	2.4G	91G	3%	/data

Gambar 3.2.1. 32 Hasil Mount Point untuk Server Partner Production 01

Pada Gambar 3.2.1.32 dapat terlihat untuk *mount point* pada server Partner Production 01 (XX-dbp\*\*\*\*\*01), dimana hasil tersebut didapatkan dari pengambilan data menggunakan *command df -h* yang terdapat pada *file df-h.txt*. Dapat terlihat pada gambar tersebut *mount point* untuk server *master* Partner Production 01 masih terpantau aman dan belum melebihi batas.

### 3) Status replikasi

```
pg_is_in_recovery
-----
f
(1 row)
```

Gambar 3.2.1. 33 Hasil Status Replikasi untuk Server Partner Production 01

Pada Gambar 3.2.1.33 dapat terlihat untuk status replikasi pada server Partner Production 01 (XX-dbp\*\*\*\*\*01), dimana hasil tersebut didapatkan dari pengambilan data menggunakan *command* `select pg_is_in_recovery()` pada *database* server, yang dimana data disimpan ke dalam *file* `replication_status.txt`. Dapat terlihat pada gambar tersebut *recovery* untuk server *master* Partner Production 01 adalah *false*.

### 4) Database size

Table 3.2.1. 1 Hasil Database Size untuk Server Partner Production 01

Database Name	Size Oktober	Size November	Growth
postgres	7953 kB	7953 kB	0%
template1	7953 kB	7953 kB	0%
template0	7809 kB	7809 kB	0%
Ad1P*****	117 MB	124 MB	5.982%

Pada Tabel 3.2.1.1 dapat terlihat untuk *database size* pada server Partner Production 01 (XX-dbp\*\*\*\*\*01), dimana hasil tersebut didapatkan pada *file* `database_size.txt`. Pada hasil yang didapatkan, dilakukan perbandingan antara hasil pada bulan November dengan bulan sebelumnya yaitu Oktober untuk melihat perkembangan pada *size database*-nya. Adapun rumus yang digunakan untuk melihat perkembangan tersebut adalah sebagai berikut:

$$\text{database size} = \frac{\text{db size saat ini} - \text{db size bulan lalu}}{\text{db size bulan saat ini}}$$

dapat terlihat pada Tabel tersebut terdapat perubahan pada *database* Ad1P\*\*\*\*\* yaitu sebesar 5.982%.

### 5) *Session*

count	state			
6				
2	active			
21	idle			
(3 rows)				
count	username	datname	state	client_addr
4				
1	it_opr	Ad1Partner	idle	10.61.26.68
1	adira_kafkon	Ad1Partner	active	10.50.5.34
1	integration2022	Ad1Partner	idle	10.61.49.148
1	replicator		active	10.91.1.14
5	adira_partner	Ad1Partner	idle	10.162.17.173
5	adira_kafkon	Ad1Partner	idle	10.50.5.34
2	pem	postgres	idle	127.0.0.1
1	postgres	postgres		
1	postgres	postgres	idle	10.161.14.34
1	postgres			
5	adira_partner	Ad1Partner	idle	10.163.16.150
1	postgres	postgres	active	
(13 rows)				

Gambar 3.2.1. 34 Hasil *Session* untuk Server Partner Production 01

Pada Gambar 3.2.1.34 dapat terlihat untuk *session* pada server Partner Production 01 (XX-dbp\*\*\*\*\*01), dimana hasil tersebut didapatkan dari pengambilan data yang terdapat pada *file* session.txt. Dapat terlihat pada gambar tersebut merupakan jumlah *session* yang berjalan pada *database* server.

### 6) *Dead life tuples*

Pada Tabel 3.2.1.2 dapat terlihat untuk *dead life tuples* pada server Partner Production 01 (XX-dbp\*\*\*\*\*01), dimana hasil tersebut didapatkan dari pengambilan data yang terdapat pada *file* dead\_life\_tuples.txt.



Table 3.2.1. 2 Hasil *Database Dead Life Tuples* untuk Server Partner Production 01

XX1P*****					
<i>Schemaname</i>	<i>Tablename</i>	<i>Live Tuples</i>	<i>Dead Tuples</i>	<i>Total</i>	<i>Persentase</i>
public	tbl_data_user_inject_history	1211	3	1214	0,25%
public	tbl_status_order_detail	85454	7078	92532	7,65%
public	tbl_data_user_inject	1	31	32	96,88%
public	tbl_order_customer	24143	1741	25884	6,73%
public	tbl_submit_order	24173	3575	27748	12,88%
public	tbl_order_unit	23061	403	23464	1,72%
public	tbl_order_domisili	23100	819	23919	3,42%
public	tbl_info_app_detail	1	0	1	0,00%
public	databasechangelock	1	38	39	97,44%
public	tbl_doc_dtl	24044	25	24069	0,10%
public	databasechangelog	20	0	20	0,00%
public	tbl_list_url_partner	22	0	22	0,00%
public	login_user_detail	1059	119	1178	10,10%
public	para_user_detail	1206	102	1308	7,80%
public	para_seq_generateid	5	6	11	54,55%
public	tbl_submit_order_temp_051022	23738	0	23738	0,00%

(16 rows)

Dapat terlihat pada tabel tersebut terdapat informasi terkait *live tuples*, *dead tuples*, *total tuples*, dan persentase. Dapat diketahui untuk *tablename* *tbl\_data\_user\_inject* pada skema *public* memiliki persentase tinggi, hal ini dikarenakan *dead tuples* pada tabel tersebut sangat besar sehingga diperlukan tindakan lanjut agar persentase kembali normal.

7) *Database table size*

Table 3.2.1. 3 Hasil *Database Table Size* untuk Server Partner Production 01

<b>XX1p*****</b>		
<b>Relkind</b>	<b>Relation</b>	<b>Total Size</b>
r	public.tbl_order_customer	43 MB
r	public.tbl_status_order_detail	19 MB
r	public.tbl_submit_order	11 MB
r	public.tbl_doc_dtl	11 MB
r	public.tbl_order_domisili	11 MB
r	public.tbl_order_unit	9976 kB
r	public.tbl_submit_order_temp_051022	2256 kB
r	public.para_user_detail	528 kB
r	public.tbl_data_user_inject_history	344 kB
r	public.login_user_detail	224 kB
r	public.tbl_data_user_inject	120 kB
r	public.databasechangelock	56 kB
r	public.para_seq_generateid	48 kB
r	public.tbl_info_app_detail	32 kB
r	public.tbl_list_url_partner	32 kB
r	public.databasechangelog	16 kB
S	public.user_id_seq	8192 bytes
S	public.lud_seq_gen	8192 bytes

(18 rows)

Pada Tabel 3.2.1.3 dapat terlihat untuk *database table size* pada server Partner Production 01 (XX-dbp\*\*\*\*\*01), dimana hasil tersebut didapatkan dari pengambilan data yang terdapat pada *file database\_table\_size.txt*. Dapat terlihat pada tabel tersebut seluruh *size* untuk setiap tabel pada setiap *database* yang terdapat pada server Partner Production 01 (XX-dbp\*\*\*\*\*01).

**d. Hasil pengambilan data pada server *slave* (Partner Production 02 (XX-dbp\*\*\*\*\*02))**

Data yang sudah di ambil akan dimasukkan ke dalam *report monthly monitoring*. Dimana *report* tersebut memuat seluruh informasi penting dari 4 server *database slave* yang dimiliki oleh perusahaan XYZ. Berikut merupakan hasil pengambilan data bulan November yang telah dilakukan sebelumnya untuk server Partner Production 02 (XX-dbp\*\*\*\*\*02):

**1) *Memory server***

	total	used	free	shared	buff/cache	available
Mem:	15G	681M	10G	1.0G	4.1G	13G
Swap:	15G	0B	15G			

Gambar 3.2.1. 35 Hasil *Memory Server* untuk Server Partner Production 02

Pada Gambar 3.2.1.35 dapat terlihat untuk *memory* pada server Partner Production 02 (XX-dbp\*\*\*\*\*02), dimana dapat terlihat pada gambar tersebut *free memory* untuk server *slave* Partner Production 02 adalah 10G.

## 2) Mount point

Filesystem	Size	Used	Avail	Use%	Mounted on
devtmpfs	7.8G	0	7.8G	0%	/dev
tmpfs	7.8G	240K	7.8G	1%	/dev/shm
tmpfs	7.8G	808M	7.0G	11%	/run
tmpfs	7.8G	0	7.8G	0%	/sys/fs/cgroup
/dev/mapper/vg_os-root	33G	9.3G	24G	29%	/
/dev/sda1	1014M	184M	831M	19%	/boot
/dev/mapper/vg_data-data	100G	1.9G	99G	2%	/data
/dev/mapper/vg_data-backup	100G	33M	100G	1%	/backup
tmpfs	1.6G	12K	1.6G	1%	/run/user/42
tmpfs	1.6G	0	1.6G	0%	/run/user/1000
/dev/mapper/vg_data-archive	100G	33M	100G	1%	/archive
Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/mapper/vg_data-data	100G	1.9G	99G	2%	/data

Gambar 3.2.1. 36 Hasil *Mount Point* untuk Server Partner Production 02

Pada Gambar 3.2.1.36 dapat terlihat untuk *mount point* pada server Partner Production 02 (XX-dbp\*\*\*\*\*02), dimana dapat terlihat pada gambar tersebut *mount point* untuk server *slave* Partner Production 02 masih terpantau aman dan belum melebihi batas.

## 3) Status replikasi

(0 rows)
pg_is_in_recovery
-----
t
(1 row)

Gambar 3.2.1. 37 Hasil Status Replikasi untuk Server Partner Production 02

Pada Gambar 3.2.1.37 dapat terlihat untuk status replikasi pada server Partner Production 02 (XX-dbp\*\*\*\*\*02), dimana dapat terlihat pada gambar tersebut *recovery* untuk server *slave* Partner Production 02 adalah *true*, yang berarti *recovery database* server Partner Production 02 (XX-dbp\*\*\*\*\*02) sedang berlangsung atau berada pada mode *standby*.

#### 4) *Session*

```
count | state
-----+-----
      5 |
      1 | active
      1 | idle
(3 rows)

count | username | datname | state | client_addr
-----+-----+-----+-----+-----
      4 |          |         |      |
      1 | pem      | postgres | idle | 127.0.0.1
      1 | postgres | postgres |      |
      1 | postgres | postgres | active |
(4 rows)
```

Gambar 3.2.1. 38 Hasil *Session* untuk Server Partner Production 02

Pada Gambar 3.2.1.38 dapat terlihat untuk *session* pada server Partner Production 02 (XX-dbp\*\*\*\*\*02), dimana dapat terlihat pada gambar tersebut merupakan jumlah *session* yang berjalan pada *database* server Partner Production 02.

#### e. *Analisa log*

Pada *managed service* juga dilakukan analisis terhadap *log* yang sudah diambil datanya sebelumnya, dimana analisis yang dilakukan adalah dengan melihat beberapa kata kunci penting seperti *error*, *fatal*, *start*, *warning*, dan lainnya. Adanya *log* ini dapat membantu perusahaan dalam memantau server *database*, jika terdapat hal-hal yang mengganjal maka perusahaan juga dapat segera menindak lanjuti sehingga tidak terjadi hal-hal yang tidak diinginkan. Adapun berikut merupakan beberapa analisa *log* yang didapatkan selama periode program kerja magang mahasiswa pada PT. Inovasi Informatika Indonesia sebagai seorang yang bertanggung jawab pada proyek *managed service* pada server-server klien perusahaan XYZ:

Table 3.2.1. 4 Tabel *Log* Analisis

Server	Log
Server Master	Terdapat IP Address yang tidak terdaftar dalam <code>pg_hba.conf</code> , IP Address ingin mencoba mengakses server dengan menggunakan <code>user</code> “itdba”, “it_opr”, “it_dev”, serta “X**** kafkon”.
	Terdapat <code>user</code> yang <code>login</code> dan <code>password</code> yang digunakan salah.
	Terdapat <code>error function upper(bytea) does not exist</code> , hal ini dikarenakan tabel yang dituju tidak terdapat dalam <code>database</code> .
Server Slave	Terdapat IP Address yang tidak terdaftar dalam <code>pg_hba.conf</code> , IP Address ingin mencoba mengakses server dengan menggunakan <code>user</code> “itdba” dan “it_dev”.
	Terdapat <code>FATAL: terminating walreceiver due to timeout ssh</code> , dimana koneksi <code>refused</code> untuk IP Address dan <code>port</code> .

#### f. Issues dan solution

Pada proses kerja magang yang telah dilakukan oleh mahasiswa pada proyek *managed service* terkhusus untuk *monthly monitoring* yang dilakukan oleh mahasiswa pada setiap bulannya, adapun *issues* yang di dapatkan, *issue* dapat dilihat dari analisis *log* dan hasil pengambilan data yang dilakukan. Selain itu adapun solusi yang diberikan untuk menyelesaikan *issue* tersebut. Berikut merupakan *issue* yang pernah terjadi pada server perusahaan XYZ serta solusi yang diberikan:

- 1) Pada bulan Oktober pada server *master* dan *slave* terputus, sehingga dibutuhkan replikasi ulang antara server *master* dan *slave*. Dimana setelah dilakukan replikasi ulang pada bulan November saat *monthly monitoring* dilakukan status replikasi antara server *master* dan *slave* sudah terhubung kembali.
- 2) Pada bulan Oktober dan November *Dead Tuples* ada yang mencapai 50% atau setara dengan 1 juta *dead tuples*, solusi

yang dapat diberikan adalah dilakukan *vacuum* terhadap *table* yang mencapai 50%.

- 3) Pada bulan Oktober dan November untuk server *slave* terdapat *issue* yaitu akses untuk *folder* /backup dan /archive dimana *user* OS postgres tidak dapat mengakses kedua folder tersebut. Solusi yang diberikan adalah dilakukannya penyesuaian *permission* dan *ownership* untuk kedua *folder* agar *user* OS postgres dapat mengaksesnya.
- 4) Pada bulan Oktober untuk server XX-s\*\*db01 penggunaan *buffer memory* terpantau tinggi, hal ini berbahaya karena bisa saja terjadi server melakukan *stop service* secara paksa yang menyebabkan *downtime*. Solusi yang dapat diberikan adalah dilakukannya *stress test* atau *performance test*.
- 5) Server XX-s\*\*db01 belum memiliki replika atau *standby database*, dimana diharapkan server XX-s\*\*db01 dapat memiliki server *standby*.

**g. Saran untuk seluruh *database***

Selain solusi, adapun saran-saran yang diberikan selama *monthly monitoring* berlangsung pada perusahaan XYZ, saran tersebut merupakan masukan diluar dari *issue* yang terjadi agar *database* perusahaan XYZ dapat bekerja secara optimal dan maksimal:

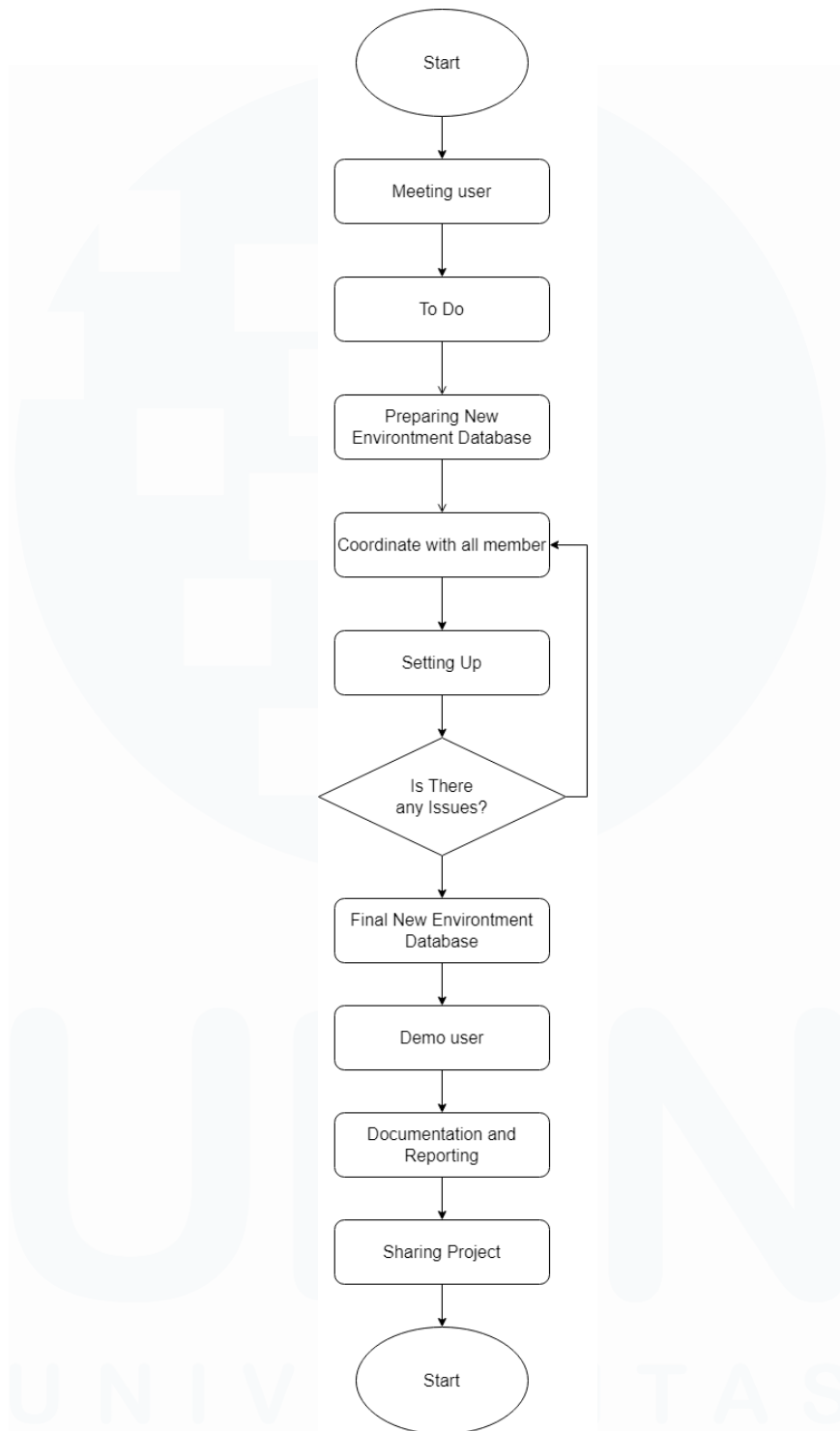
- 1) Melakukan *tuning parameter* agar dapat meningkatkan *performance database*.
- 2) Mengkonfigurasi PEM *Log Manager* hal ini berguna untuk *monitoring* server agar lebih baik lagi.

**3.2.2. Project Internal Data Management Internship (Minggu 21 s/d 26)**

Selain mahasiswa membantu projek *managed service* pada PT. Inovasi Informatika Indonesia, mahasiswa diberi kesempatan untuk terlibat dalam *Project Internal Internship Data Management*. *Project*

*internal* tersebut merupakan proyek yang melibatkan seluruh mahasiswa *internship* dari *Data Management Internship* pada PT. Inovasi Informatika Indonesia. *Project* tersebut merupakan *project* mengenai *Deploying New Environment Database*, tujuan dilakukannya *project* ini adalah agar mahasiswa mendapatkan gambaran bagaimana alur dari pengembangan *database* baru. Adapun alur kerja dari *Project Internal Internship Data Management* seperti yang terdapat pada gambar 3.2.2.1, pada gambar tersebut merupakan skenario kerja pada *project* ini dimulai dari *meeting* dengan *user* untuk mendapatkan kebutuhan *user* terhadap *database*-nya, setelah itu maka seluruh mahasiswa program kerja magang divisi *Data Management* akan mendaftarkan *to-do list* yang perlu dilakukan agar dapat memenuhi kebutuhan *user*. Setelah itu daftar *to-do list* didapatkan maka dilakukan tahap preparasi seluruh kebutuhan yang diperlukan *user*, dalam tahap preparasi tentunya diperlukan juga koordinasi antar anggota agar meminimalisir kesalahan. Dari koordinasi yang dilakukan maka akan dilanjutkan dengan *setting up environment database*-nya. Setelah dilakukan *setting up*, akan dilihat apakah terdapat *issue* atau tidak pada *environment* yang telah disiapkan tersebut. Apabila ada maka akan dilakukan koordinasi ulang dengan anggota untuk menemukan solusi, apabila tidak ada kendala maka akan dilanjutkan kepada tahapan finalisasi *environment database*. Jika *environment* tersebut sudah sesuai dengan kebutuhan yang disebutkan oleh *user* maka akan dilakukan *demo* kepada *user*, untuk memperlihatkan bagaimana cara kerja *environment database* tersebut. Setelah *demo* yang dilakukan maka mahasiswa akan melakukan dokumentasi atas *demo* tersebut dan dari hasil dokumentasi tersebut akan di *share* kepada seluruh *staff* yang berada di PT. Inovasi Informatika Indonesia.





Gambar 3.2.2. 1 Alur Kerja *Project Internal Data Management Internship*

## 1. Meeting User

*Meeting user* dilakukan pada minggu kedua puluh empat, dimana disini mahasiswa diberikan kesempatan untuk memberikan presentasi terkait dengan *project* yaitu persiapan *environment database*. Pada *meeting user* tersebut dapat diketahui beberapa kebutuhan yang diperlukan oleh *user* sebagai berikut:

- a. *User* ingin melakukan migrasi dari *database* Oracle ke *database* lain yang dapat menekan biaya keuangan.
- b. *User* ingin *database* memiliki replikasi secara *real time*.
- c. *User* menginginkan adanya *tools* untuk membatasi koneksi yang masuk ke dalam *database*.
- d. *User* ingin performa *database* dapat meningkat.
- e. *User* ingin adanya *benchmarking* untuk meninjau *database*.
- f. *User* ingin keamanan untuk OS dan *database*.
- g. *User* ingin adanya *monitoring* atas *database*.

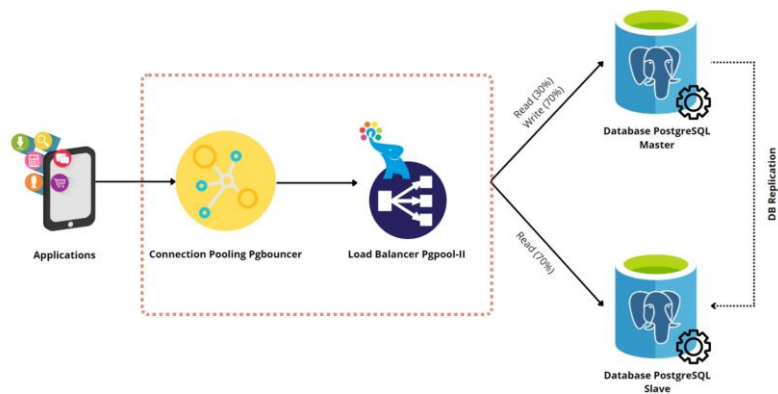
Dari *meeting user* yang dilakukan adapun beberapa hal yang ditawarkan untuk dapat memenuhi kebutuhan tersebut, diantara lain:

- a. Produk yang ditawarkan untuk migrasi adalah PostgreSQL, hal ini dikarenakan PostgreSQL merupakan *open source* dan banyak fitur-fitur yang dapat digunakan serta menekan biaya.
- b. Replikasi dapat dilakukan menggunakan RepMgr, dimana RepMgr merupakan *tools open source* dari EDB, yang dimana EDB merupakan versi *enterprise* dari PostgreSQL.
- c. *Tools* yang digunakan untuk membatasi koneksi yang masuk ke dalam *database* adalah dengan menggunakan *tools* yang disediakan oleh PostgreSQL yaitu *pgbouncer* dan *pgpool*.
- d. Dilakukan *tuning* terhadap parameter *database* agar dapat meningkatkan performa *database*.

- e. *Benchmarking* untuk kebutuhan perbandingan akan dilakukan dengan menggunakan *tools* yang telah disediakan oleh PostgreSQL yaitu *pgbench*.
- f. Dilakukannya *hardening database* untuk meningkatkan keamanan *database*.
- g. *Monitoring* akan menggunakan PEM yang disediakan oleh EDB, dimana PEM memiliki *alert notification* yang dapat membantu *user* dalam memantau *database*.

## 2. *Preparing*

Dalam *project* ini di ketahui kebutuhan *user* seperti yang sudah disebutkan pada poin sebelumnya, dimana berikut merupakan skenario *environment database* baru. Pada gambar 3.2.2.2 merupakan topologi dari *environment database*, dimana topologi ini telah disesuaikan oleh kebutuhan *user*.



Gambar 3.2.2. 2 Topologi *New Environment Database*

Pada Gambar 3.2.2.2 dapat terlihat 3 server yaitu Server 1 untuk menginstal *pgpool* dan *pgbouncer*, Server 2 yaitu server *master*, dan Server 3 yaitu server *slave*. Pada *project* ini mahasiswa diberi tanggung jawab dalam *to do list benchmarking*

menggunakan *pgbench* serta *setting up monitoring* dan *alert* menggunakan PEM.

#### a. *Benchmarking*

Dalam memenuhi kebutuhan *user* maka dalam *project* ini mahasiswa menawarkan *pgbench* sebagai salah satu *tools* yang berguna untuk *benchmarking*. *Benchmarking* yang dilakukan adalah *benchmarking* terhadap skenario *database* sebelum dan sesudah di *tuning*. Selain itu, *benchmarking* juga dapat dilakukan untuk *testing* skenario alur kerja dari *database* baru.

Untuk dapat melakukan *benchmarking* maka *database* perlu di *set up* dulu sesuai dengan kebutuhan *user*, dimana *database* perlu diinstal dan dikonfigurasi terlebih dahulu *tools* *pgpool* dan *pgbouncer*, setelah itu dilakukan replikasi menggunakan *RepMgr*, dan dilakukan *hardening* terhadap *database*. Setelah itu, maka akan dilakukan *benchmarking* menggunakan *pgbench* sebelum dilakukan *tuning*. Namun, karena *database* belum memiliki data maka untuk kebutuhan *benchmarking* maka mahasiswa membuat *database* kosong pada Server 2: *database master* yang diberi nama yaitu *db\_bench*. Setelah itu, untuk mendapatkan data untuk kebutuhan *testing* diperlukan inisiasi data terlebih dahulu menggunakan *command* berikut:

```
pgbench -i -U postgres -p 5488 -h 10.8.60.243 db_bench
```

Pada *command* tersebut berguna untuk menginisiasi *database* *db\_bench*. Dimana *command -i* berguna untuk menginisiasi *database* agar memiliki data *dummy* yang dapat digunakan untuk kebutuhan *benchmarking*. *Command -U* merupakan spesifikasi *username* yang *login* ke dalam *database*, dimana *user* yang digunakan untuk *login* ke

*database* adalah *user* postgres. Untuk dapat mengakses *database* maka diperlukan *command* -p dan -h, yang dimana -p berguna untuk spesifikasi *port database* dan -h berguna untuk spesifikasi *hostname* dari *database*. Dimana *port* dan *hostname* untuk inisiasi *database* diisi dengan *port* dan *hostname* Server 2: *database master* yaitu dengan *port* 5488 dan *hostname* yaitu 10.8.6.243, setelah spesifikasi *database* dilakukan maka diakhir *command* diberikan spesifikasi nama *database* yang digunakan untuk menampung data *dummy*. Setelah *database* memiliki data *dummy* maka mahasiswa dapat melakukan *benchmarking* untuk melihat bagaimana performa *database* sebelum *tuning*. Berikut merupakan *command* yang digunakan oleh mahasiswa untuk *benchmarking database* sebelum di *tuning*:

```
/usr/pgsql-12/bin/pgbench -p 65488 -U postgres -h  
10.8.60.229 -c 500 -t 100 -n -S db_bench
```

*Benchmarking* dilakukan dengan menjalankan skenario apabila *connection pooling* pada *pgbouncer* dan *pgpool* dihidupkan, serta *load balancer* pada *pgpool* hidup. Dengan skenario tersebut maka spesifikasi *port* adalah pada *port* *pgbouncer*, dengan *user* postgres, dan menggunakan *hostname* yaitu 10.8.6.229, dengan spesifikasi *connection* yang mengakses *database* adalah 500 dengan transaksi perkoneksinya adalah 100. *Command* -n digunakan untuk *no vacuum* agar *database* dengan data *dummy* tidak melakukan *vacuum*, serta *command* -S adalah *command select only*, dimana *command* ini dijalankan agar dapat melihat apakah *load balancer* pada *pgpool* berjalan atau tidak. Dimana *load balancer* merupakan *tools* yang berguna untuk membagi beban pekerjaan antara *database master* dan *slave*. Sehingga berikut merupakan hasil untuk *benchmarking database*

sebelum dilakukannya *tuning* dengan skenario *connection pooling* *pgbouncer* dan *pgpool* hidup serta *load balancer* pada *pgpool* hidup:

```
[root@localhost ~]# /usr/pgsql-12/bin/pgbench -p 65488 -U postgres -h 10.8.60.229 -c 500 -t 100 -n -S db_bench
transaction type: <builtin: select only>
scaling factor: 50
query mode: simple
number of clients: 500
number of threads: 1
number of transactions per client: 100
number of transactions actually processed: 50000/50000
latency average = 98.977 ms
tps = 5051.681512 (including connections establishing)
tps = 5052.359953 (excluding connections establishing)
```

Gambar 3.2.2. 3 *Benchmarking* Menggunakan *pgbench* Sebelum di *Tuning*

Pada Gambar 3.2.2.3 dapat terlihat hasil *benchmarking database* yang belum di *tuning*, dimana dengan koneksi yang masuk pada *database* 500 koneksi dan setiap koneksi melakukan 100 transaksi maka *latency* yang di dapatkan adalah sekitar 98.977 ms, dan *transaction per second* yang didapatkan adalah 5051 transaksi per detik. Untuk dapat meningkatkan kinerja *database* dan menaikkan *tps* maka dilakukan *tuning* terhadap parameter *database*, adapun beberapa parameter yang di *tuning* seperti *max\_connection*, *shared\_buffers*, *work\_mem*, dan lainnya. Dimana dengan dilakukan *tuning* terhadap parameter-parameter tersebut maka dilakukan *benchmarking* ulang dengan skenario alur kerja *database* yang sama, sehingga didapatkan hasil dari *benchmarking* tersebut adalah:

```
[root@localhost ~]# /usr/pgsql-12/bin/pgbench -p 65488 -U postgres -h 10.8.60.229 -c 500 -t 100 -n -S db_bench
transaction type: <builtin: select only>
scaling factor: 50
query mode: simple
number of clients: 500
number of threads: 1
number of transactions per client: 100
number of transactions actually processed: 50000/50000
latency average = 83.419 ms
tps = 5993.844463 (including connections establishing)
tps = 5994.827364 (excluding connections establishing)
```

Gambar 3.2.2. 4 *Benchmarking* Menggunakan *pgbench* Setelah di *Tuning*

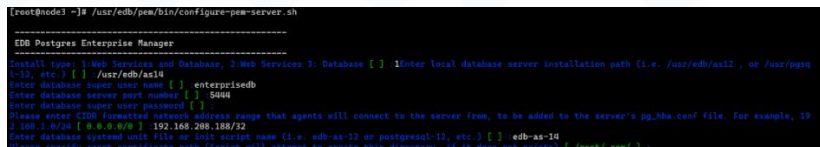
Dengan *tuning database* dapat dilihat bahwa *latency*-nya berubah menjadi 83.419 ms dan *transaction per second*-nya naik menjadi 5993.844 transaksi per detik.

## b. *Setting Up*: PEM Server

Untuk memenuhi kebutuhan *user* dalam *monitoring database*, adapun *tools* yang ditawarkan adalah PEM Server. PEM Server sendiri perlu diinstal dan konfigurasi terlebih dahulu sebelum nantinya dapat digunakan oleh *user*, sehingga mahasiswa diberi kesempatan untuk dapat melakukan *setting* PEM pada *environment database*. Hal yang perlu diperhatikan adalah mahasiswa perlu menginstal repo pem-server terlebih dahulu pada server *database master* dan pada server *pgbouncer* dan *pgpool*, hal ini dikarenakan *database user* akan menggunakan PostgreSQL bukan EDB sehingga repo untuk pem-server perlu di *install* terlebih dahulu, selain itu kedua server tersebut memiliki perannya masing-masing dalam PEM Server, yang dimana server *database master* merupakan *Database*-nya sedangkan server dengan *pgpool* dan *pgbouncer* merupakan *Web Services*. Setelah pem-server berhasil terinstal dalam *database* maka akan dilakukan konfigurasi PEM Server dengan menggunakan *command* berikut:

```
sudo /usr/edb/pem/bin/configure-pem-server.sh
```

Dengan *command* tersebut maka PEM Server akan di konfigurasi, dimana akan muncul beberapa kebutuhan konfigurasi seperti berikut:



```
[root@hdc03 ~]# /usr/edb/pem/bin/configure-pem-server.sh
-----
EDB Postgres Enterprise Manager
-----
Install type: 1) Web Services and Database, 2) Web Services 3) Database [ ] 1)Enter local database server installation path (i.e. /usr/edb/as12 or /usr/local/pgsql-12, etc.) [ ] /usr/edb/as12
Enter database name (e.g. postgres) [ ] enterprisedb
Enter database server port number [ ] 5444
Enter database super user password [ ]
Please enter CIDR formatted network address range that agents will connect to the server from, to be added to the server's pg_hba.conf file. For example, 10.1.10.0/24 [ *.*.*.* ] 192.168.208.188/32
Enter database system user (i.e. edb-as-12 or postgresql-12, etc.) [ ] edb-as-14
Please specify agent certificate path (script will attempt to create this directory, if it does not exist) [ /root/.pem ] :
```

Gambar 3.2.2. 5 Gambaran konfigurasi PEM Server

Dimana Gambar 3.2.2.5 merupakan gambaran untuk melakukan konfigurasi terhadap PEM Server. Sehingga, untuk Server 2 yaitu *database master* perlu mengkonfiguasi

*install type 3* untuk *database*, sedangkan Server 1 untuk server *pgpool* dan *pgbouncer* akan mengkonfigurasi *install type* yaitu 2 untuk dapat mengakses *web services*. Untuk konfigurasi lainnya dapat disesuaikan dengan server-nya, setelah konfigurasi selesai maka nantinya PEM Server dapat diakses melalui link <https://10.8.60.229:8443/pem>, yang dimana agar dapat diakses melalui *device* yang berbeda maka perlu di daftarkan terlebih dahulu IP koneksinya pada *pg\_hba.conf*. Setelah selesai mengkonfigurasi PEM Server, maka diperlukan *enable* untuk service *httpd*, hal ini berguna agar PEM Server dapat diakses tanpa perlu mengkonfigurasi ulang PEM Server. Berikut merupakan dokumentasi untuk PEM Server yang telah di install dan dikonfigurasi sebelumnya, dimana pada Gambar 3.2.2.6 merupakan halaman utama apabila *user* mengakses PEM Server melalui *website browser*. Dimana untuk dapat melihat *monitoring* terhadap *database* dan server maka diperlukan *login* terlebih dahulu.



Gambar 3.2.2. 6 Halaman *Login* PEM Server

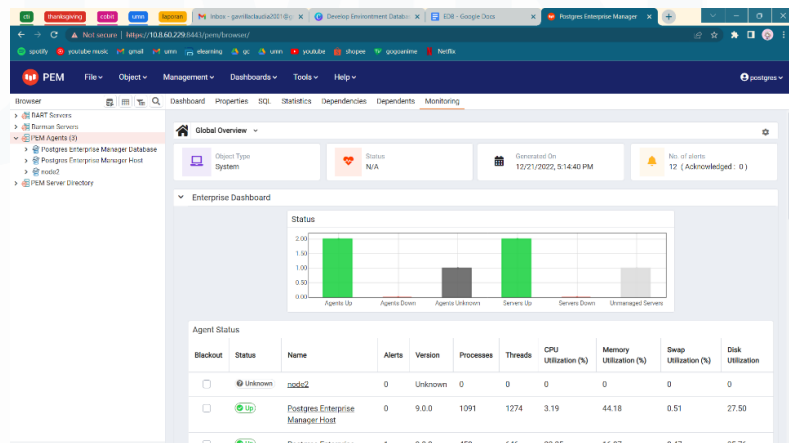


### c. *Setting Up: PEM Agent*

Agar dapat memaksimalkan *monitoring database* maka diperlukan *PEM Agent* yang dapat membantu untuk mengumpulkan informasi-informasi seputar *database*. Untuk dapat menggunakan *PEM Agent*, *PEM Agent* perlu di *set up* terlebih dahulu, dimana setelah mengkonfigurasi *PEM Server* maka secara otomatis terdapat 2 *PEM Agent* yang sudah teregistrasi yaitu pada server *master* dan satu lagi pada Server 1. Dimana kurang 1 *PEM Agent* lagi untuk server *slave*, sehingga perlu dilakukan registrasi terlebih dahulu menggunakan *command* berikut:

```
/usr/edb/pem/agent/bin/pemworker --register-agent --  
pem-server 10.8.60.229 --pem-port 65488 --pem-user  
postgres --allow_server_restart true --allow-batch-  
probes true --batch-script-user postgres
```

dengan *command* tersebut dijalankan, maka nanti *agent* akan teregistrasi dalam *PEM Server*. *PEM Agent* yang sudah di registrasi perlu di tugaskan kepada server yang hendak dipantau, 1 server memiliki 1 *PEM Agent* yang bekerja. Dengan begitu nantinya *user* dapat memantau *database* dengan lebih mudah karena *PEM Agent* membantu dalam pengumpulan *reporting database*.



Gambar 3.2.2. 7 Halaman Utama PEM Server

Pada Gambar 3.2.2.7 merupakan halaman utama dari PEM Server, dimana pada gambar tersebut merupakan *global overview* yang berguna untuk melihat status *agent* atau server yang telah di registrasi. Apabila status *up* maka *agent* atau server aktif, apabila *down* maka *agent* atau server tidak aktif. Pada gambar tersebut juga dapat terlihat untuk *list* dari *agent* dan server yang diregistrasi.

**d. *Sharing Project***

*Sharing project* merupakan kegiatan *sharing* hasil dari *project* yang telah dilakukan, *sharing project* di lakukan pada minggu kedua puluh enam. Pada kegiatan *sharing project* mahasiswa beserta dengan *internship* lain yang terlibat dalam *Project Internal Data Management Internship* membagikan pengalaman dari *project* yang dilakukan.

### **3.3 Kendala yang Ditemukan**

Adapun kendala yang dihadapi oleh mahasiswa selama melaksanakan program kerja magang pada PT. Inovasi Informatika Indonesia sebagai divisi *Data Management* dengan projek yaitu *managed service*, antara lain:

1. Kurangnya sumber untuk memperdalam ilmu terkait dengan PostgreSQL ataupun EDB. Hal ini mempengaruhi kinerja mahasiswa dalam mendalami ilmu.
2. Terdapat kesulitan dalam mengerjakan *Project Internal Data Management Internship* dikarenakan tenggat waktu yang dekat dan sumber dalam mengembangkan kebutuhan *project* masih minim.
3. Terdapat kesulitan dalam penarikan data pada perusahaan klien. Hal ini dikarenakan VPN untuk dapat mengakses server perusahaan sering terputus, dan karena adanya kesulitan koneksi membuat pekerjaan suatu proyek terganggu.

4. Terdapat kesulitan dalam program kerja magang pada PT. Inovasi Informatika Indonesia karena kurangnya perangkat komputer untuk melaksanakan program kerja magang.

### **3.4 Solusi atas Kendala yang Ditemukan**

Adapun solusi yang dapat diimplementasikan guna untuk mengatasi kendala yang dihadapi oleh mahasiswa selama melaksanakan program kerja magang pada PT. Inovasi Informatika Indonesia sebagai divisi *Data Management* dengan proyek yaitu *managed service*, antara lain:

1. Melakukan penelusuran dan pembelajaran mandiri dengan bantuan serta arahan dari pembimbing lapangan, agar mendapatkan sumber ilmu yang lebih akurat.
2. Melakukan penentuan prioritas dalam *project* serta pembagian tugas yang lebih rinci yang ditentukan sebelum dimulainya suatu *project* agar *project* dapat terselesaikan dalam waktu yang telah ditentukan.
3. Mengecek VPN secara berkala agar tidak mengalami gangguan koneksi.
4. Mempersiapkan laptop mandiri dengan spesifikasi standar untuk kebutuhan kerja pada bidang Teknologi Informasi.