1. CLIENT WME-PG

DBParams.ini

```
[DB]
POSTGRES=1
[POSTGRESQL]
NumeServerul1=19x.xx.xx.11
NumeServerul2=19x.xx.xx.22
NumeServerul3=19x.xx.xx.33
DBName=postgres
ServerName=NumeServerul1
```

- NumeServerul1,2,3 sunt serverele la care vreti sa va logati din WME;
- ServerName=NumeServerul1 este serverul curent pe care lucrati;
- IP-urile trebuie setate la cele N aliasuri de server si trebuie sa fie IP server de Postgres dorit.

Important: DBName="postgres" numele bazei de date ESTE CASE-SENSITIVE.

Nota: Pentru cei care vor sa testeze pe acelasi server versiuni diferite de PostGreSQL se pot instala si configura pe porturi diferite ca de ex:
Serverul1=19x.xx.xx.11**:5432** *port default*
Serverul2=19x.xx.xx.11**:5433** *port custom*
Acelasi IP la server dar porturi diferite conform datelor din instalarile de versiuni diferite de PostGreSQL, de ex *PGv.14.0* pe 19x.xx.xx.11**:5432** si *PGv.14.1* pe 19x.xx.xx.11**:5433**.
Pentru acest caz trebuie doua directoare pentru instalare de ***Server*WME**, fiecare cu dbparams.ini configurat corespunzator serverului de PostGreSQL.

Deoarece in POSTGRES nu exista ceva similar cu TNSNames.ORA, toate datele de logare sunt in acest fisier INI.
DBParams.ini se pune in folderul cu WMEnterprise.

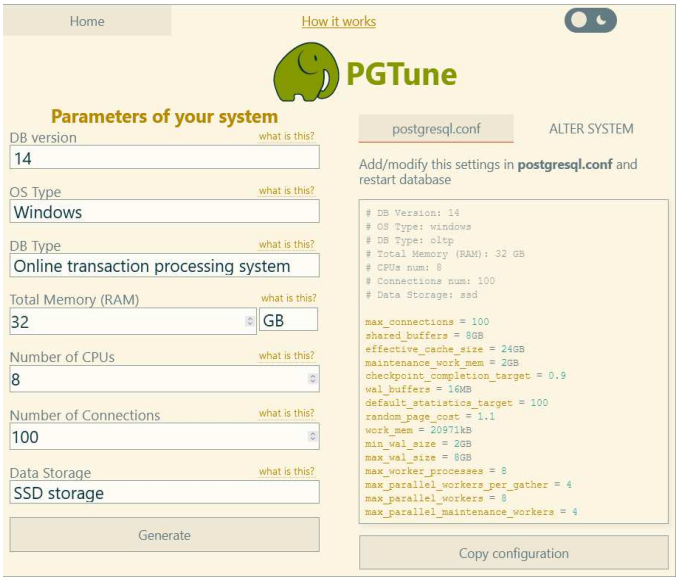
2. SERVER WME-PG

Postgresql.conf este fisierul de parametri de pornire Postgres si este numai pentru server.
Fisierul text *postgresql.conf din kit* contine setarile pentru buna functionare a Postres, cu observatia ca
datele referitoare la configuratie hardware a serverului trebuie insa descrisa dupa instalare (si la fiecare modificare de configuratie).

Accesati link **https://pgtune.leopard.in.ua/#/** in ideea de a obtine parametrii specifici serverului vostru.

Apasa butonul „Generate”. În partea dreapta vor apărea câțiva parametri și valorile propuse.
Recomandarea initiala ar fi generarea configurarea cu optiunea DBType = Online Transaction Processing (oltp).
De asemenea pot fi luate in calcul si optiunile Mixed type of application (server comun pentru BD si Aplicatie) sau Data Warehouse (dw) (in functie de modul de folosire al aplicatiei, predominant

rapoarte, etc).

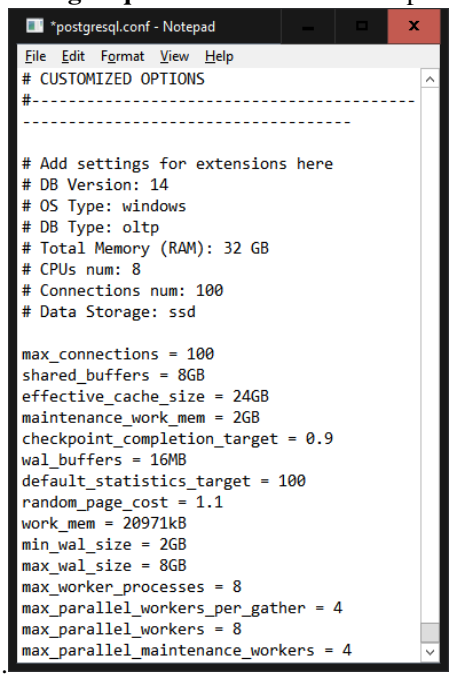


Nota:

La parametrul **Number of Connections** se punea 100 default: "max_connections=100" (se da restart la servicii PostGre) dar pentru un tunning mai bun acest paramentru se poate schimba in functie de nr. de utilizatori WME-PG, astfel:

a. Intre 1 si 5 utilizatori: max_connections=40
b. Intre 5 si 10 utilizatori: max_connections=60
c. Intre 10 si 20 utilizatori: max_connections=80
d. Intre 20 si 50 utilizatori: max_connections=100
e. Peste 50 utilizatori: max_connections=100 si se va stabili de la caz la caz, in functie de nr. de utilizatori peste 50.

Copy Configuration si editati **Postgresql.conf** la final de lista de parametri, dupa linia *"# Add settings for extensions here"*.



Acești parametri vor fi actualizati în fișierul de configurare *postgresql.conf* dat de noi pe care tocmai l-ati copiati peste cel creat by default la instalarea Postgres.

Nota:

La "Total Memory (RAM)" **daca** aveti pus 32GB va aparea sugestia la parametrul: maintenance_work_mem = 2GB.

Schimbati acest parametru maintenance_work_mem = 2047MB

Indiferent de cat RAM aveti pe server si vi se sugereaza la acest paramentru, acesta nu poate depasi 2047MB.

Nota AUTOVACUUM:

In PostgreSQL, managementul tabelelor temporare este diferit de Oracle, in sensul ca acestea sunt create pentru fiecare sesiune noua. Din acest motiv tabela de sistem pg_attribute si indecsii acesteia, in anumite cazuri cresc intr-un mod alarmant. Aceasta crestere a acestei tabele afecteaza in timp performanta bazei de date PostgreSQL.

Pentru a ajuta la menținerea performanței bazei de date și a eficienței spațiului prin eliminarea rândurilor care nu mai sunt necesare trebuie facuta o mentenanta periodica a BD numita VACUUM.

Trebuie precizat exista AUTOVACUUM care este activat implicit în PostgreSQL și poate fi configurat folosind mai mulți parametri în fișierul postgresql.conf.

1. autovacuum: activează sau dezactivează procesul de fundal de autovacuum.

2. autovacuum_vacuum_threshold: determină numărul minim de rânduri moarte care trebuie să fie prezente într-un tabel înainte de a fi curatat. Valoarea implicită este 50.

3. autovacuum_analyze_threshold: determină numărul minim de rânduri active care trebuie să fie prezente într-un tabel înainte de a fi analizat. Valoarea implicită este 50.

4. autovacuum_vacuum_scale_factor: un multiplicator care determină câte rânduri moarte sunt necesare pentru a declanșa un vacuum în funcție de dimensiunea tabelului. Valoarea implicită este 0,2.

5. autovacuum_analyze_scale_factor: multiplicator care determină câte rânduri active sunt necesare pentru a declanșa o analiză în funcție de dimensiunea tabelului. Valoarea implicită este 0,1.

6. autovacuum_vacuum_cost_delay: timpul (în milisecunde) în care AUTOVACUUM îl va aștepta înainte de a începe o operație de VACUUM. Valoarea implicită este 20.

7. autovacuum_vacuum_cost_limit: numărul maxim de rânduri care pot fi curatate într-o singură operațiune de VACUUM. Valoarea implicită este 200.

EXEMPLU:

autovacuum = on

autovacuum_vacuum_threshold = 100

autovacuum_analyze_threshold = 100

autovacuum_vacuum_scale_factor = 0.5

autovacuum_analyze_scale_factor = 0.2

autovacuum_vacuum_cost_delay = 50

autovacuum_vacuum_cost_limit = 500

Este important să configurați aceste setări pentru a vă asigura că vacuum și analiza corect rulează eficient și nu provoacă o încărcare prea mare a bazei de date. De asemenea, este o idee bună să monitorizați activitatea autovacuumului și să faceti VACUUM manual al tabelelor care nu sunt întreținute în mod adecvat de către autovacuum.

VACUUM

În PostgreSQL, ori de câte ori rândurile dintr-un tabel sunt șterse (DELETE), rândul (tuplul) existent este marcat ca mort (nu va fi eliminat fizic) și în timpul unei actualizări (INSERT), marchează tuplul de ieșire corespunzător ca mort și inserează un nou tuplu, deoarece în PostgreSQL operațiunile UPDATE = DELETE + INSERT.

Aceste tupluri moarte consumă spațiu de stocare inutil și, în cele din urmă, aveți o bază de date PostgreSQL umflată.

VACUUM recuperează spațiu și îl face disponibil pentru reutilizare, dar spațiul suplimentar nu este returnat sistemului de operare este doar păstrat disponibil pentru reutilizare în cadrul aceluiași tabel.

Balonarea (umflare) tabelelor cu tuple moarte afectează serios performanța interogărilor PostgreSQL, deaorece tabelele și indecșii sunt stocate ca matrice de pagini cu dimensiune fixă. Ori de câte ori o interogare solicită rânduri, instanța PostgreSQL încarcă aceste pagini în memorie, iar rândurile moarte provoacă ingreuneaza operatiile I/O pe disc în timpul încărcării datelor.

VACUUM FULL rescrie întregul conținut al tabelului într-un fișier de disc nou, fără spațiu suplimentar, permițând ca spațiul neutilizat să fie returnat sistemului de operare.

Acest mod este mult mai lent și necesită o blocare exclusivă pe fiecare tabel în timp ce este procesat.

a. **Pg_hba.conf** este fisierul de autentificare a clientilor PG si contine ip-uri care au acces la serverul de postgre, trebuie editat, de ex:

IPv4 local connections (*0.0.0.0/0* permite oricarui IP sa se conecteze la serverul de PG) :

host all all 0.0.0.0/0 md5

Note:

a. O problema care merita atentie la importul de date in PG, este dimensiunea foarte mare a fisierelor de tip LOG.

b. De retinut ca fisierele .log pot fi sterse manual sau de un job care ruleaza la un anumit interval de timp.

Dar pentru importul datelor sau pentru alte operatiuni care genereaza fisiere de log foarte mari, ar fi indicat ca in fisierul **postgresql.conf** parametrul sa fie pe **off**:

logging_collector = off (necesita restart la BD) , astfel incat sa nu se mai genereaze fisiere tip log.

Dupa terminarea operatiunilor de import date ar fi bine sa se genereze totusi fisiere log, de exemplu cate un fisier pentru fiecare zi a saptamanii cu suprascrierea lor la fiecare 7 zile. Pentru aceasta trebuie configurati in **postgresql.conf** urmatorii parametri astfel:

log_destination = 'stderr'
logging_collector = on
log_directory = 'log'
log_filename = 'postgresql-%a.log'
log_truncate_on_rotation = on

log_rotation_age = 1d
log_rotation_size = 0

Daca cineva considera ca nu sunt necesare fisierele log, atunci se poate seta in **postgresql.conf** doar:

logging_collector = off

c. Atat **Postgresql.conf** cat si **Pg_hba.conf** se configureaza si se pun in dir.: \PostgreSQL**14\data** !

Se restarteaza serviciul de postgres sau restart server, dupa orice modificare a acestor fisiere de config!