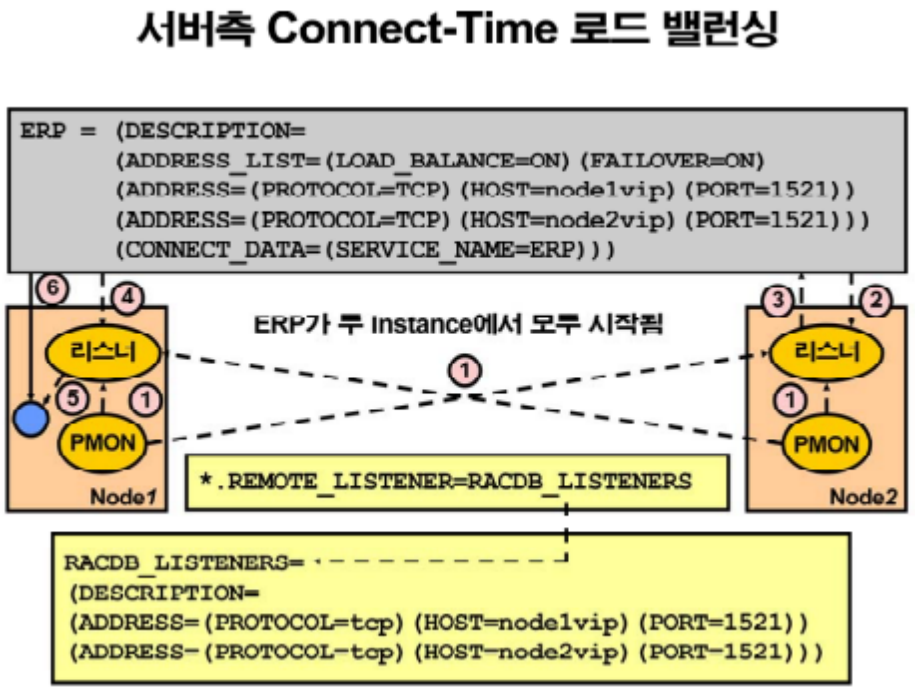
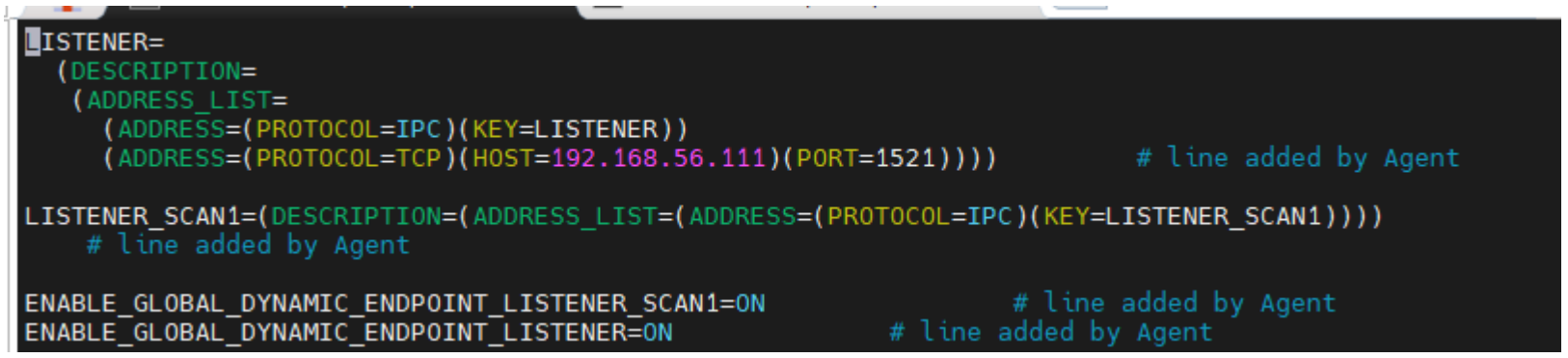
73. Server Load Balancing




```
Server Load Balancing: 서버의 부하에 따라서 로드를 분산하는 기능
=> 각각의 노드에 PMON이 자기 인스턴스에 얼마나 부하가 있는지를 내 노드의 리스너와 상대편 노드의
     리스너에게 알려준다. 리스너들은 노드의 부하가 각각 얼마나 일어나고 있는지 알고 있다.

Server Load Balancing 기능을 이용하기 위해서 설정해야 할 2가지 사항
1. spfile에 remote_listener 파라미터 설정을 해야 한다. 여기에 리스너들의 주소를 넣어야 한다.
     tnsnames.ora에 그 주소를 등록하면 된다.

2. 리스너의 서비스 등록 방법을 동적 서비스 등록 방법으로 설정해야 한다.

실습 1.
1. remote_listener 파라미터가 설정 되어 있는지 확인한다.
show parameter remote_listener

=> rac-scan은 scan listener ip 주소의 병칭이다.
=> 리스너의 주소를 알고 있으므로 PMON이 리스너에게 인스턴스가 부하가 있는지 없는지 실시간으로
     알려준다.
```

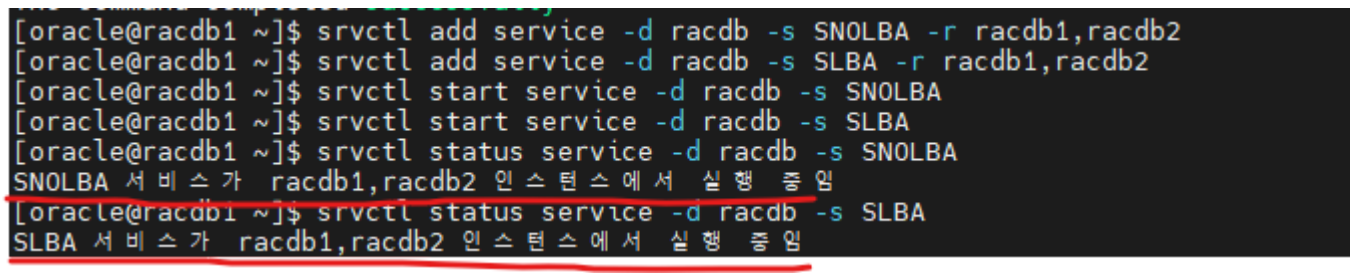
```
racdb1(SYS) >  show parameter remote_listener

NAME                                 TYPE
------------------------------------ ---------------
VALUE
------------------------------
remote_listener                      string
rac-scan:1521
```

```
2. 리스너의 서비스 등록 방법이 동적인지 정적인지 확인하시오.
=> 동적 서비스 등록 (오라클 권장): 오라클이 알아서 등록하기
     11g PMON -> listener
     12c LPEG -> listener

=> 정적 서비스 등록: listener.ora 파일 안에 사람이 직접 서비스를 등록
cd $ORACLE_HOME/network/admin
vi listener.ora
=> 열어 봤을 때 안에 sid name이나 service name이 없으면 동적 서비스 등록이다.
     => (아래 그림) 없음
```



실습 2. Server Load Balancing Test
1. 아래 2개 서비스를 시작시킨다.
(1번 노드)

```
srvctl add service -d racdb -s SNOLBA -r racdb1,racdb2
srvctl add service -d racdb -s SLBA -r racdb1,racdb2

srvctl start service -d racdb -s SNOLBA
srvctl start service -d racdb -s SLBA

srvctl status service -d racdb -s SNOLBA
srvctl status service -d racdb -s SLBA
```



2. 아래의 SQL 을 수행하여 서비스 속성을 변경한다.sys 유저에서 수정한다.

```
exec DBMS_SERVICE.MODIFY_SERVICE (                -
      'SNOLBA',                                    -
      goal     => DBMS_SERVICE.GOAL_NONE,          -
      clb_goal => DBMS_SERVICE.CLB_GOAL_LONG);

 exec DBMS_SERVICE.MODIFY_SERVICE (                -
      'SLBA',                                      -
      goal     => DBMS_SERVICE.GOAL_SERVICE_TIME,  -
      clb_goal => DBMS_SERVICE.CLB_GOAL_SHORT);

=> '-' 싸인은 이어 붙이겠다는 뜻으로 쓰인다.

※ goal 옵션 :
 1. goal     => DBMS_SERVICE.GOAL_NONE : 서버 로드 밸랜싱 안하겠다.
 2. goal     => DBMS_SERVICE.GOAL_SERVICE_TIME : 서버 로드 밸랜싱 하겠다.

※ clb_goal 옵션 :
1. clb_goal => DBMS_SERVICE.CLB_GOAL_LONG : 세션의 갯수로 로드 밸런싱
2. clb_goal => DBMS_SERVICE.CLB_GOAL_SHORT : 서비스 친화도로 로드 밸런싱

1번 노드가 부하가 있는데도 불구하고 무조건 균등하게 세션을 2개의 노드에 각각 분할 해준다면
1번 노드에 접속한 세션들은 성능 저하를 경험하게 됩니다.
 1번 노드가 busy 하면 2번 노드로 접속할 수 있게 해줘야합니다
```

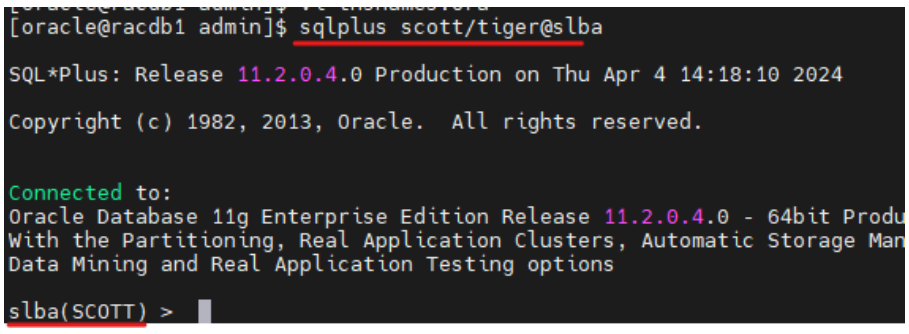
3. 위의 두개의 서비스로 접속할 수 있는 tns 정보가 1번 노드의 tnsnames.ora 에 있는지 확인한다.

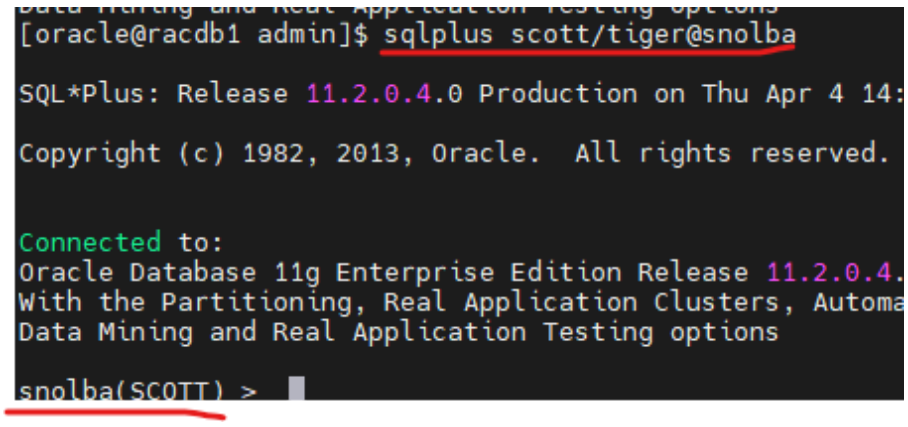
```
cd $ORACLE_HOME/network/admin
vi tnsnames.ora

SNOLBA =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = rac1-vip)(PORT = 1521))
    (ADDRESS = (PROTOCOL = TCP)(HOST = rac2-vip)(PORT = 1521))
    (LOAD_BALANCE = yes)
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = SNOLBA)
    )
  )


SLBA =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = rac1-vip)(PORT = 1521))
    (ADDRESS = (PROTOCOL = TCP)(HOST = rac2-vip)(PORT = 1521))
    (LOAD_BALANCE = yes)
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = SLBA)
    )
  )
```

=> 두 서비스의 tns alias 가 설정되어져 있다.(접속 테스트)

```
sqlplus scott/tiger@slba

sqlplus scott/tiger@snolba
```

=> 두개 모두 접속 완료





지금 부터는 서버쪽 로드 밸런스 실습 자료다. 아래 7개의 파일을 모바텀에다가 올린다.
그 다음에 잘 작동되는지 확인한다. (1,2번 노드 모두 실행)

```
./primes
```

=> 권한이 없다고 나오면 권한을 올려준다.

```
chmod 777 primes
chmod 777 *.sql
chmod 777 *.sh
```

primes

stopfanload.sh

startfanload.sh

fan.sh

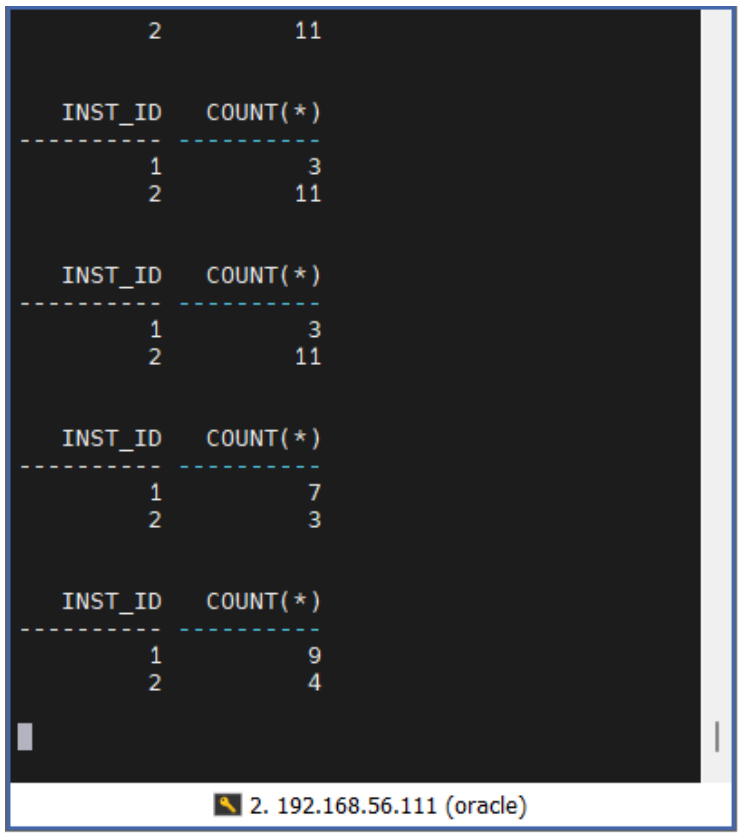
kill_tafb.sql

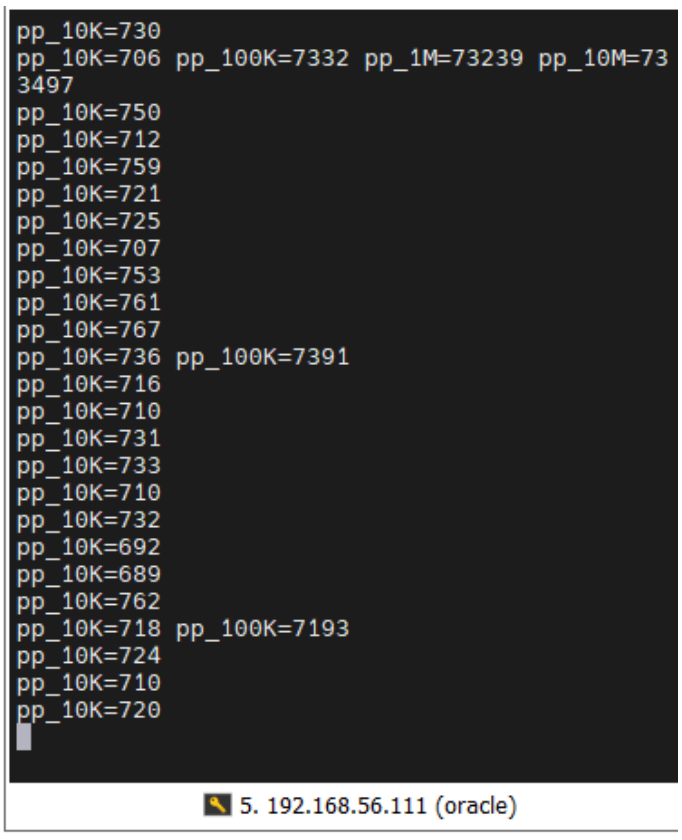
kill_fan.sql

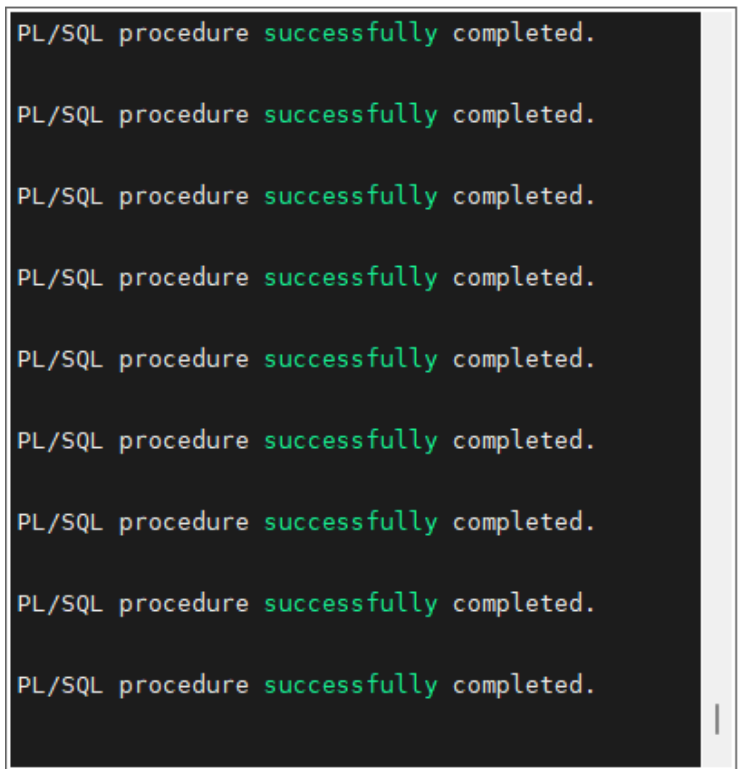
fan.sql

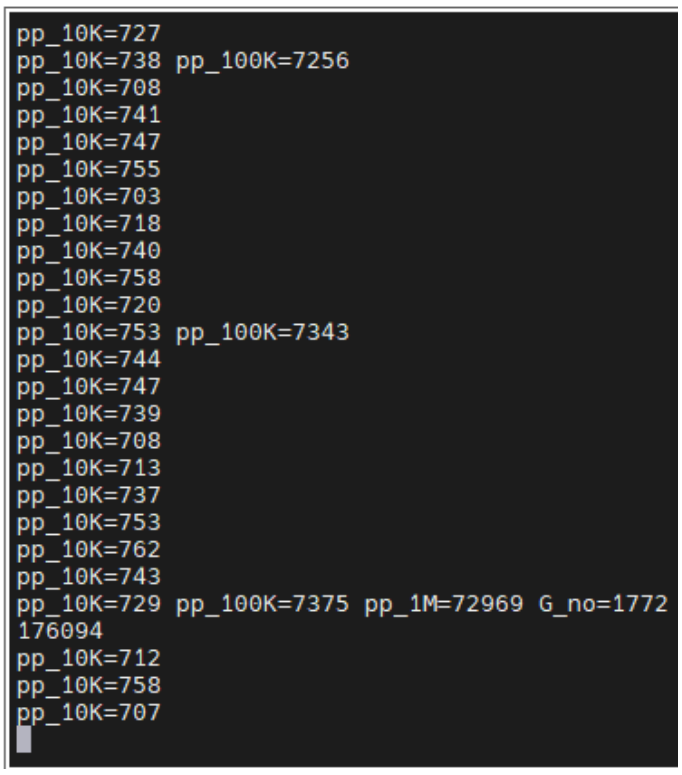
```
4. sys 유저에서 아래의 유저를 생성하고 권한 부여

create user JFV
identified by jfv;

grant connect, resource, dba to JFV;

connect jfv/jfv

create table fan(c number);

5. 별도의 터미널 창을 열고 아래의 SQL 이 반복되게 쉘 스크립트 작성
vi jfv.sh
---------------------------
#!/bin/bash
while true
do
sqlplus -s / as sysdba <<EOF

select inst_id, count(*)
  from gv\$session
  where username='JFV'
  group by inst_id
  order by inst_id;

EOF
sleep  2;
done
-----------------------------
```

```
6. 부하를 주는 스크립트를 1번 노드에만 2개를 띄운다.
(1번 노드 1)
./primes

(1번 노드 2)
./primes

7. fan.sh 스크립트를 아래의 내용으로 편집하시오.
=> fan.sh 쉘 스크립트는 jfv 유저로 접속을 120번 하게 하는 스크립트인데

vi fan.sh
----------------------------------
#!/bin/ksh
#
# fan.sh
#
users=120
x=1
y=$users
UNPW="jfv/jfv@$1"
ORACLE_HOME="/u01/app/oracle/product/11.2.0/db_1"
export ORACLE_HOME

while [ $x -le $y ]
do
    /u01/app/oracle/product/11.2.0/db_1/bin/sqlplus -s $UNPW @fan.sql
    x=$((x + 1))
done
----------------------------------
※ 주의 사항!
fan.sh에 ^M 를 다 지워야 한다. 다음과 같이 실행하여 fan.sh를 열고 ^M을 다 지운다.
vi -b fan.sh

8. fan.sh 스크립트를 호출하는 startfanload.sh를 다음과 같이 수행하여 load balancing을 안
     했을 때 테스트를 해보시오.
mkdir solutions
cp fan.sh ./solutions/fan.sh

./startfanload.sh SNOLBA

새로운 창 열어서
sh jfv.sh

또 새로운 창 열어서
./primes

또 새로운 창 열어서
./primes
```









오래동안 돌면 안 꺼진다. 스크립트를 중지 시킬려면 Ctrl + C 를 누르고 빨리 ./stopfanload.sh 수행.

stopfanload.sh에도 ^M 를 다 지운다.
vi -b stopfanload.sh

jfv 유저가 fan 테이블에 데이터를 insert를 하는 일을 수행했다. 그러면서 테이블 스페이스도 full이 났을 거고 archive log file도 꽉 차서 공간 부족하다고 문제가 생겼다.

이를 해결하기 위해서 2가지를 하겠다.
1. jfv 유저로 접속해서 fan 테이블을 truncate 한다.
connect jfv/jfv

truncate table fan;

2. rman으로 접속해서 archive log file을 전부 지운다.
(1번 노드)
rman target sys/oracle nocatalog

delete copy;
delete backup;

backup database;

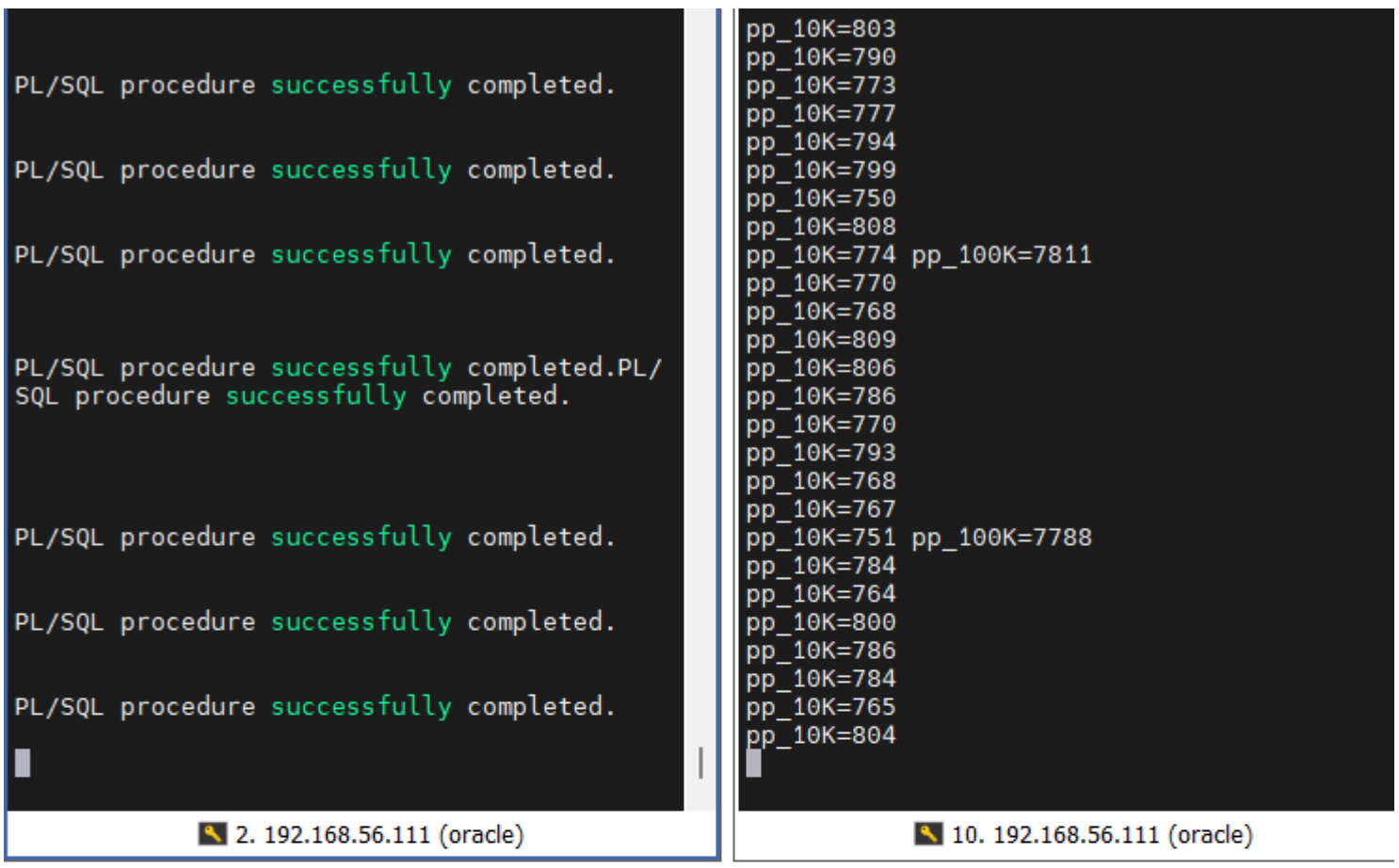
```
=> 위의 실습은 load balancing을 안 하는 SNOLBA 서비스를 이용해서 jfv 유저로 1번 노드와 2번
     노드에 계속해서 접속하게 하고 fan 테이블에 데이터를 입력하게 하는 실습을 했다. 결과적으로 load
     balancing을 안하므로 1번으로 접속을 많이 하든 2번으로 접속을 많이 하든 균등하게 나오든 크게
     상관 없다.
```

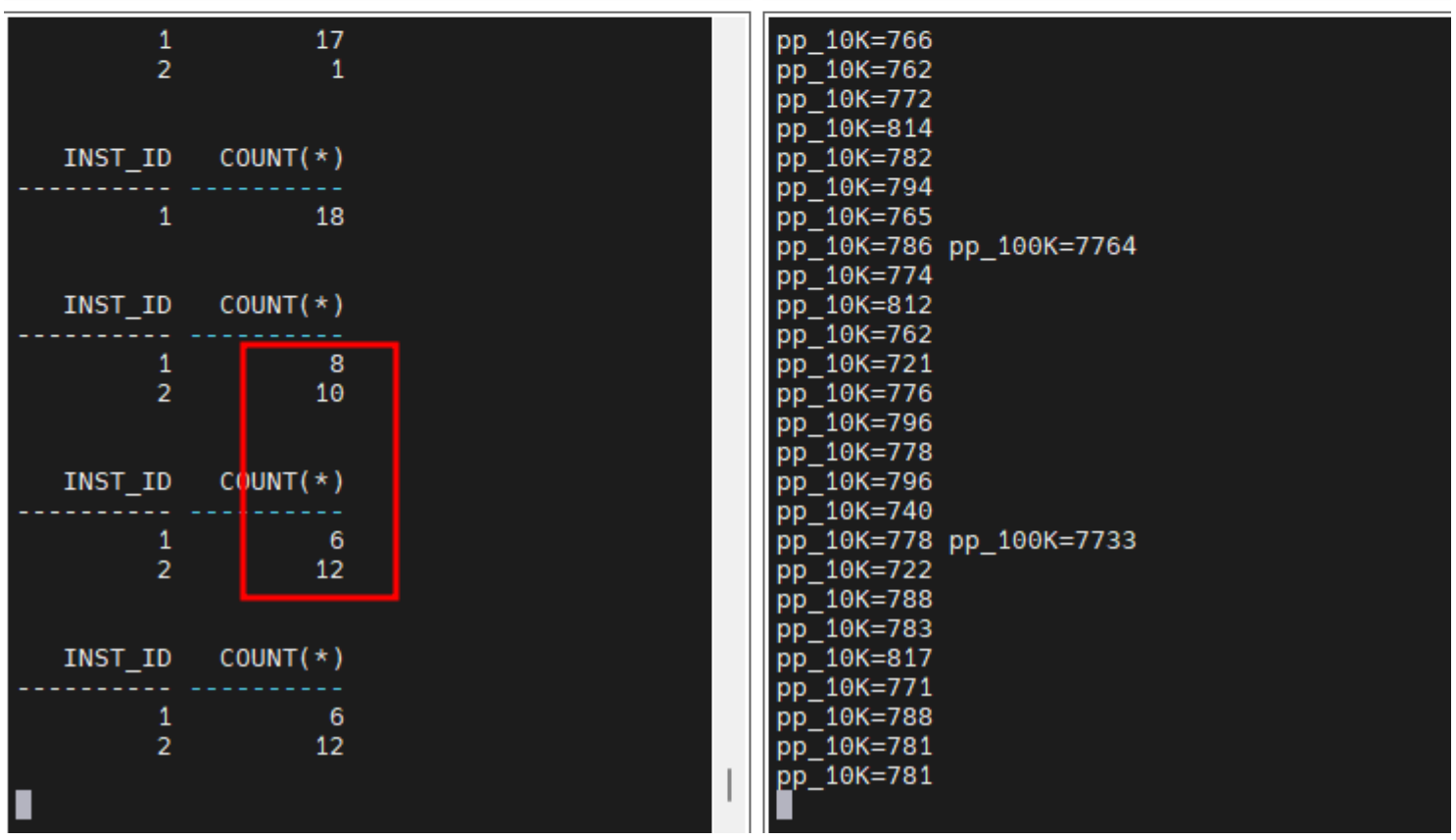
```
문제 1. 다음과 같이 서버쪽 load balancing을 하겠다고 SLBA 서비스를 수정하는데 세션의 개수로
load balancing을 하게 설정하시오.
(sql 1 sys)
 exec DBMS_SERVICE.MODIFY_SERVICE (                -
      'SLBA',                                      -
      goal     => DBMS_SERVICE.GOAL_SERVICE_TIME, -
      clb_goal => DBMS_SERVICE.CLB_GOAL_LONG);

=> 변경하고 나서 터미널 창 2개를 준비한다. 1개는 startfanload SLBA를 수행해서 jfv 유저로
     접속하게 하는 창이고 다른 1개는 jfv 유저가 각각의 노드에 몇개나 접속 되었는지 확인하는
     sh jfv.sh를 수행한다.

(1번 노드 1)
./startfanload.sh SLBA

(1번 노드 2)
sh jfv.sh

exec DBMS_SERVICE.MODIFY_SERVICE (               -
      'SNOLBA',                                   -
      goal     => DBMS_SERVICE.GOAL_NONE,         -
      clb_goal => DBMS_SERVICE.CLB_GOAL_LONG);
```

```
PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.


PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.
PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.



PL/SQL procedure successfully completed.PL/SQL procedure successfull
pleted.


PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.
[oracle@racdb1 ~]$
```

```
         1         14
         2          4


   INST_ID   COUNT(*)
---------- ----------
         1         14
         2          4


   INST_ID   COUNT(*)
---------- ----------
         1         14
         2          4


   INST_ID   COUNT(*)
---------- ----------
         1         14
         2          4


   INST_ID   COUNT(*)
---------- ----------
         1         16
         2          2


   INST_ID   COUNT(*)
---------- ----------
         1         16
         2          2


   INST_ID   COUNT(*)
---------- ----------
         1         16
         2          2


   INST_ID   COUNT(*)
---------- ----------
         1         17
         2          1


   INST_ID   COUNT(*)
---------- ----------
         1         17
         2          1


   INST_ID   COUNT(*)
---------- ----------
         1         11
         2          7
```

```
위와 같이 실행하고 1번에 부하를 많이 주게 되면 2번으로 세션들이 더 많이 접속 되는지 확인하시오.
(1번 노드 3)
./primes

(1번 노드 4)
./primes

=> 이렇게 걸고 20분 정도 지나면 2번 노드로 접속이 많이 되는걸 확인할 수 있다.
```






```
connect jfv/jfv

truncate table fan;

rman target sys/oracle nocatalog

delete copy;
delete backup;

backup database;
```

문제 2. client load balancing test를 해서 제대로 1번과 2번 노드가 랜덤으로 접속이 되는지 확인하시오.

```
cd $ORACLE_HOME/network/admin
vi tnsnames.ora
=> racdb_taf 있는지 확인
```

```
sqlplus scott/tiger@racdb_taf

select instance_name from v$instance;

=> 1번 노드와 2번 노드 둘 다 나온다.
```