

Project report

프로젝트 명	ORACLE 11g RAC(Real Application Clusters)
개발 기간	2018.07.25 ~ 2018.07.28
참여인원	1명
개발 환경	Oracle Linux
사용 도구	Oracle Vm ware
사용 기술	Oracle 11g

Content

Real Application Clusters 2 page

Configuration & Feature 3 page

Clusterware

CRS

ASM

RAC(+cache fusion)

Installation 9 page

Grid Infrastructure 설치

RAC DB설치

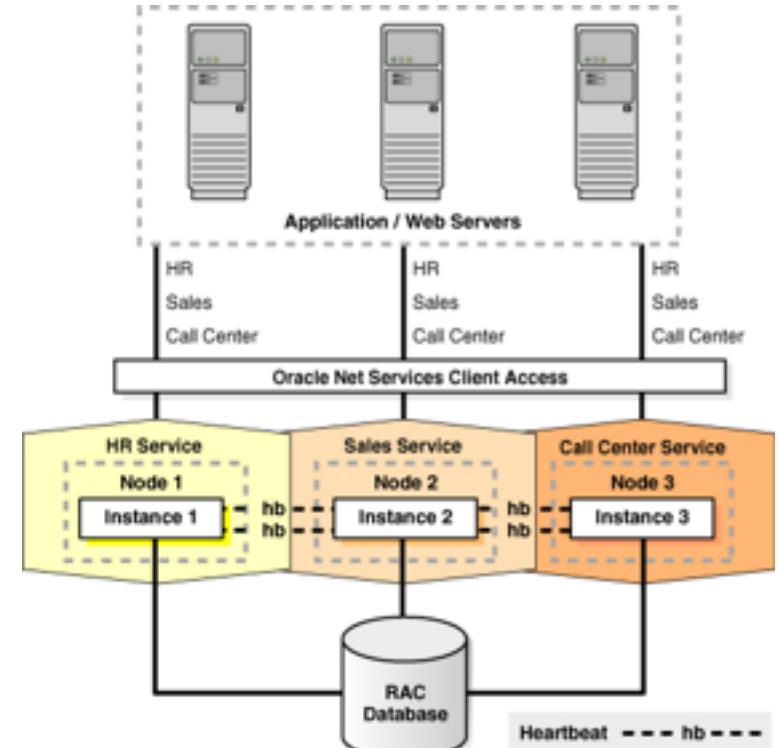
Cache Fusion 확인 28 page

■ Real Application Clusters

Real Application Clusters(이하 RAC)는 오라클의 대표적인 instance High availability option으로 하나의 Database에 2개이상의 Instance node를 구성하여 무정지 시스템으로 사용하는 것이다.

RAC를 구성하였을 때 얻어지는 이점은

1) Instance Fail Over : 임의의 노드의 instance에 장애가 발생하여 instance가 down되었을 때 정상 node의 instance의 정보를 가지고 recovery 가능



2) Load Balancing : 동시에 여러 session이 Database에 connection을 맺을 때 instance들에 균등하게 session을 분배

■ Configuration & Feature

1. Clusterware

Cluster란 노드(단일 CPU, Storage, Memory 보유)들을 그룹으로 묶은 것이고 Clusterware는 Cluster 구축 및 관리를 제공하는 소프트웨어이다.

Grid Infrastructure를 설치하여 Oracle Clusterware Architecture를 구성하게 되면 얻는 이점으로 클러스터관리, 노드 모니터링, 이벤트 서비스(cluster의 변경 사항을 인식), 노드들의 시간정보 동기화, High availability, private IP 이중화(HAIP/1~4개 까지 생성됨)

1.

2.RAC DATABASE를 구성하기 전 Oracle Clusterware인 Grid Infrastructure를 setup하여 CRS(Cluster Ready Service)와 ASM(Automatic Storage Management)를 구성한다

3.

2. CRS

4.CRS에 DB instance, ASM instance, listener, disk group... Oracle resource들의 정보가 기록된다.

5. CRS의 가장 큰 장점은 CRS에 기록된 모든 resource에 대하여 Auto re-startup을 지원, OCSSD가 Private IP를 통해 OCR에 등록된 resource들의 vital sign을 체크하고 비정상적이면 auto re-start를 수행, 그 결과를 Voting Disk file에 저장한다.

3. ASM

6.OracleDB에서 사용하는 파일들을 Storage에 자동으로 저장하고 관리하는 것으로

7.#Raid0 - Striping -- file_size/disk_num 균등하게 가져감

- Coarse- Striping - 1, 2, 4, 8, 16, 32 MB / Default

- Fine-Grained Striping - 좀더 작은 단위로 조각 128KB씩 8조각 낼 수 있음(OLTP)

*AU - allocation Unit - Raid0(Striping 에서 조각단위)1,2,4,8,16,32 MB

8.#Raid1 – Mirroring option

1-way:External (no mirroring)

2-way:Normal

3-way:High

* 최대 63개 그룹까지 생성 가능 * 최대 10000개의 디스크 사용 가능

4. RAC

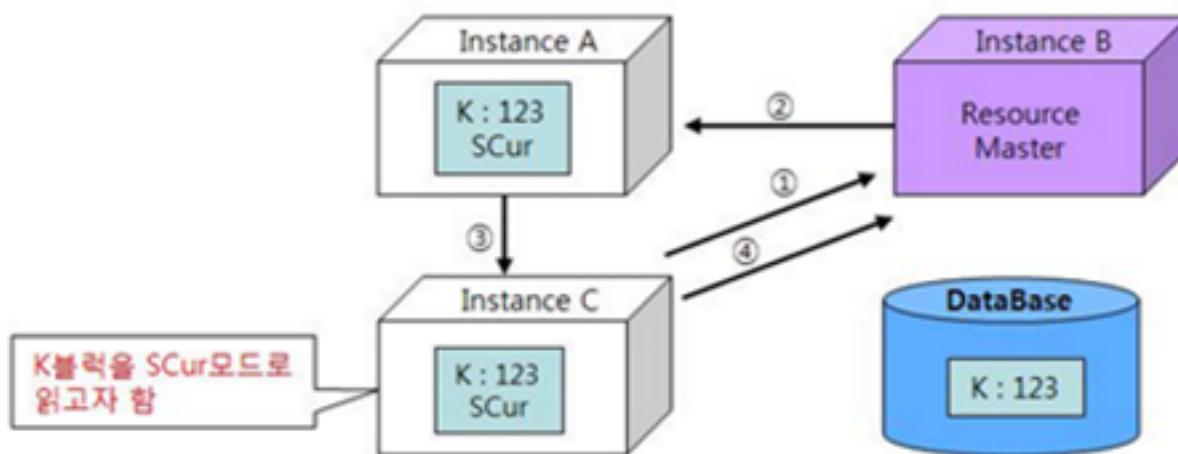
9.

- TAF(Transparent Application Failover) : 1번째 instance node에서 장애발생 시 수행중이던 Transaction의 정보가 2번째 instance node로 옮겨져 작업을 이어서 수행
- CTF : 장애가 있는 Instance로 접속 요청이 오면 다른 instance로 연결해주는 기능
- OPS : Oracle Parallel Sever. RAC의 Cache Fusion을 사용하기 이전에 사용되던 기술. 하나의 storage에 2개 이상의 instance가 연결되어 있는 구성으로, 사용자가 각각 다른 instance에 접속을 해도 storage가 하나이므로 같은 데이터를 조회, 변경할 수 있다.
 - 장점
하나의 instance에서 장애가 발생할 경우 남아 있는 다른 instance에 접속을 해도 storage에 접근 가능하기 때문에 서비스가 중단되는 경우를 예방할 수 있다.
 - 단점
RAC ping : 다른 instance의 데이터를 가져오기 위해서는 우선 디스크에 저장을 한 후 데이터를 사용하려고 하는 instance로 복사해서 작업해야 한다. 즉, 디스크를 사용해야 하므로 시간이 오래 걸린다.
- Cache Fusion : OSP구성의 단점인 RAC ping문제가 개선되어 성능이 크게 향상된 것으로, oracle 9i버전부터는 서로 다른 instance에서 변경된 데이터를 디스크를 거치지 않고 바로 instance로 가져올 수 있는 기능이다. block단위로 일어나며 private network(interconnect)를 통해 전송된다.
 - 장점
logical read이므로 성능에 좋고, 경합(Contention)을 해결하는데 도움을 준다.
 - 단점
로컬 버퍼 캐시 (또는 다른 SGA구성 요소)와 달리 물리적으로 안전한 메모리 구성 요소가 아니다.
- Cache Fusion 관련 Deamon & Background Process
 - GRD : Global Resource Directory. RAC 에서 Buffer 와 LCO/Library Cache Object), RCO(Row Cache Object) 등의 모든 리소스의 최신 상태를 관리하는 곳. 각 instance의 Shared Pool내에 존재한다. GES와 GCS로 관리, Dynamic Remastering를 담당한다. (from 9i)
 - GCS : Global Cache Service. 공통 버퍼 캐시에 대한 액세스 관리 기능. GCS는 Cache Fusion 알고리즘을 통해 캐시 일관성을 구현한다.
 - GES : 클러스터에서 Enqueue관리, 모든 instance간 리소스 운영 담당, 클러스터간 Deadlock감지 기능
 - Global cache : 클러스터링 되어 있는 모든 인스턴스의 버퍼 캐시를 하나의 버퍼 캐시로 간주한다. 모든 데이터 블록에 대해서 마스터 노드가 각각 정해져 있고, 그 노드를 통해서 global cache에 캐싱되어 있는 블록의 상태와 Lock정보를 관리한다.
 - 관련 백그라운드 프로세스
 - LMON : Global Enqueue Service Monitor. 각 instance에 떠있는 GRD를 계속 refresh한다. (각 GRD의 싱크를 맞춰준다)
 - LMSn : Global Cache Service Process. GCS전담 프로세스로 ownership을 가지고 있는 instance에 요청하는 역할을 한다.
 - LMD : GES전담 프로세스다.

- 블록 모드 종류
 - Null(N) 모드 : 해당 블록을 사용중인 사용자가 없다는 것을 뜻한다.
 - Share(S) 모드 : 해당 블록을 select하고 있는 세션이 있다는 것을 뜻한다.
여러 instance에서 동시에 select할 수 있다.
 - Exclusive(X) 모드 : 해당 블록을 특정 사용자가 변경하고 있다는 것을 뜻한다.
반드시 1개의 instance에서만 변경할 수 있다.
- Current block :
 - 디스크로부터 읽혀진 후 사용자의 간접사항이 반영된 최종 상태의 원본 블록을 말한다.
 - Shared Current(SCur) : select 작업하는 CR block (read용도). 동시에 여러 노드에 캐싱될 수 있다
 - Exclusive Current(XCur) : update 작업하는 CR block (write용도). 단 하나의 노드에만 존재 할 수 있다.
 - 자주 읽히는 데이터 블록은 각 노드가 SCur 모드로 캐싱하고 있을 때 가장 효율적이며, 그 중 한 노드가 XCur 모드로 업그레이드 요청하는 순간 다른 노드에 캐싱되어 있던 SCur 블록들은 모두 NULL 모드로 다운그레이드 된다.
- 주요 buffer states 정리
(V\$BH.STATUS에서 상태 확인 가능 - read로 빌려갔는지 write로 빌려갔는지)
 - Shared Current : SCur
 - Exclusive Current : XCur
 - Consistent Read : CR
 - update중에 commit하기 전 상태
 - cache fusion으로 전송된다.
 - SCur 또는 PI로부터 CR블록이 만들어진다.
 - Past Image : PI (이전 이미지)
 - 블록이 디스크에 기록되기 전에 데이터 블록의 복사본을 말한다.
 - RAC의 한 instance가 블록을 요청하면 버퍼 캐시에 그 블록의 PI를 유지하여 블록을 요청한 instance로 보낸다. (디스크에 쓰지 않고)
 - XCur이미지에서 write되면 CR로 변경된다.
 - 전체 instance내에서 한 번에 둘 이상의 PI가 있을 수도 있다.
- Request Node : 필요한 특정 블록을 Master Node의 LMSn에 요청한다.
Master Node : 요청 받은 블록의 Holder Node를 파악한 후 LMSn에게
 - Request Node로의 버퍼 전송을 요청한다. 블록이 없을 시에는 Request Node에게 디스크에서 읽어 들이는 권한을 부여한다.
 - Holder Node : 블록전송이 가능한 경우, Request Node로 블록을 전송한다.

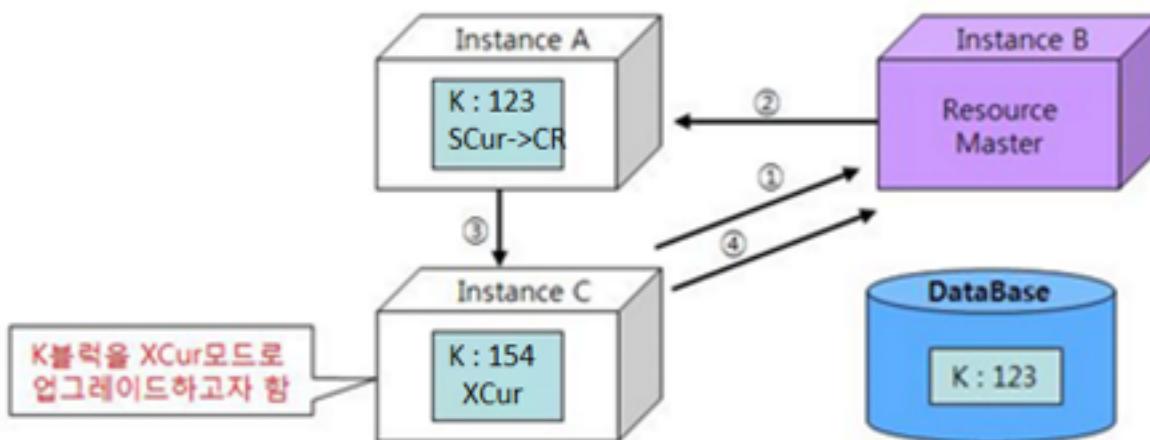
1) Cache Fusion Type

- Read – Read



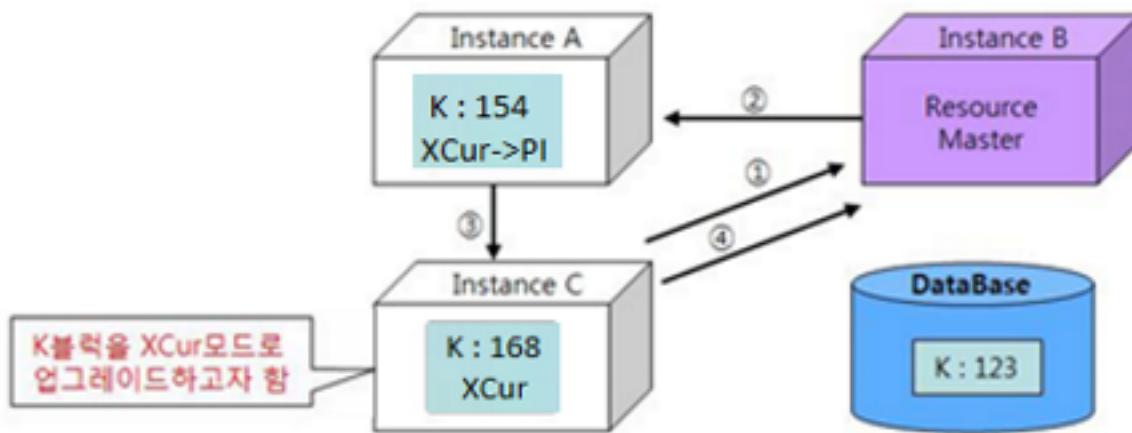
- ① C노드는 리소스 마스터인 B노드에게 K블록에 대한 전송 요청을 보낸다.
- ② B노드는 현재 K블록을 A노드가 캐싱하고 있음을 확인하고 C노드에 블록을 전송해주도록 A 노드에게 지시한다.
- ③ A노드는 C노드에게 블록을 전송해 준다.
- ④ C노드는 블록을 성공적으로 전송 받아 SCur 모드로 캐싱하게 되었음을 알리는 메시지를 마스터 노드인 B에게 보낸다.

- Read – Write



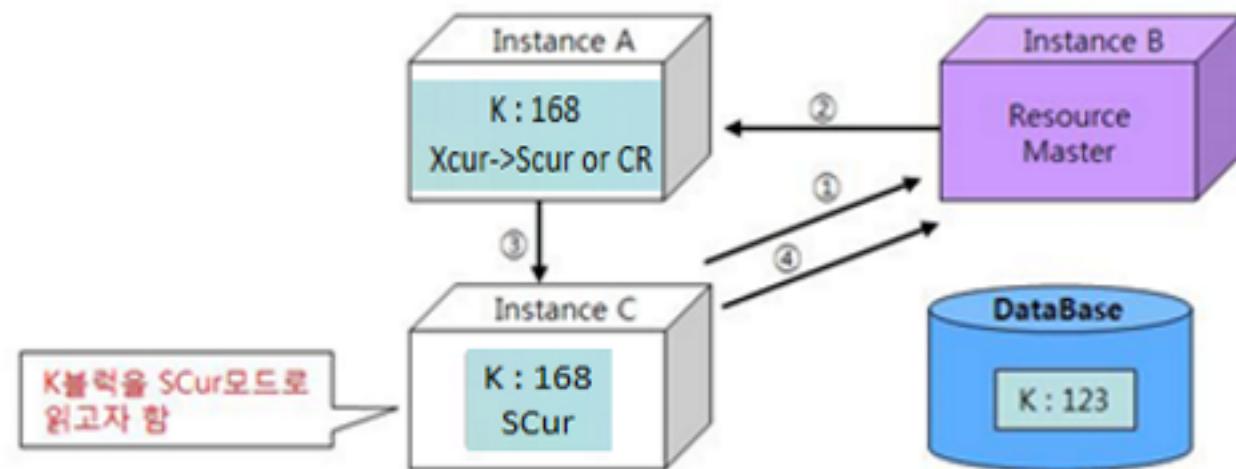
- ① C노드는 마스터 노드인 B에게 K블록을 XCur 모드로 업그레이드하겠다고 요청한다.
- ② B노드는 현재 K블록을 A노드도 캐싱하고 있음을 확인하고 Null모드(CR상태)로 다운그레이드 하도록 지시한다.
- ③ A노드는 C노드에게 Null 모드로 다운그레이드 했음을 알려준다.
- ④ C노드는 K블록을 XCur모드로 업그레이드하고 그 결과를 마스터 노드인 B에게 알려준다. 이 때 A노드의 K블록이 Null 모드로 다운그레이드 된 것까지 함께 알려준다. (C노드의 K블록 SCN이 123 -> 154로 변경된다)

- Write – Write



- ① C노드는 마스터 노드인 B에게 K블록을 XCur 모드로 요청한다.
 - ② B노드는 현재 K블록을 A노드가 XCur 모드로 캐싱하고 있음을 확인하고 C노드에게 보내주도록 지시한다.
 - ③ A노드는 C노드에게 블록을 전송하고 자신이 갖고 있던 블록은 Null 모드로 다운그레이드(PI 상태)하여 전송한다.
 - ④ C노드는 K블록을 XCur 모드로 캐싱하게 되었음을 B노드에게 알려준다.
- (A노드의 K블록 SCN이 154 -> 168로 변경된다.)

- Write – Read



- ① C노드는 마스터 노드인 B에게 K블록을 SCur 모드로 요청한다.
- ② B노드는 현재 K블록을 A노드가 XCur 모드로 캐싱하고 있음을 확인하고 C노드에게 보내주도록 지시한다.
- ③ A노드는 C노드에게 블록을 전송하고 자신이 가지고 있던 블록을 SCur모드로 다운그레이드 한다.
- ④ C노드는 K블록을 SCur모드로 캐싱하게 되었음을 B노드에게 알려준다. 이때 A노드에 캐싱되어 있던 블록이 SCur모드로 다운그레이드된 사실까지 함께 알려준다.

■ Installation

1) Grid Infrastructure 설치

A. 초기 환경 설정

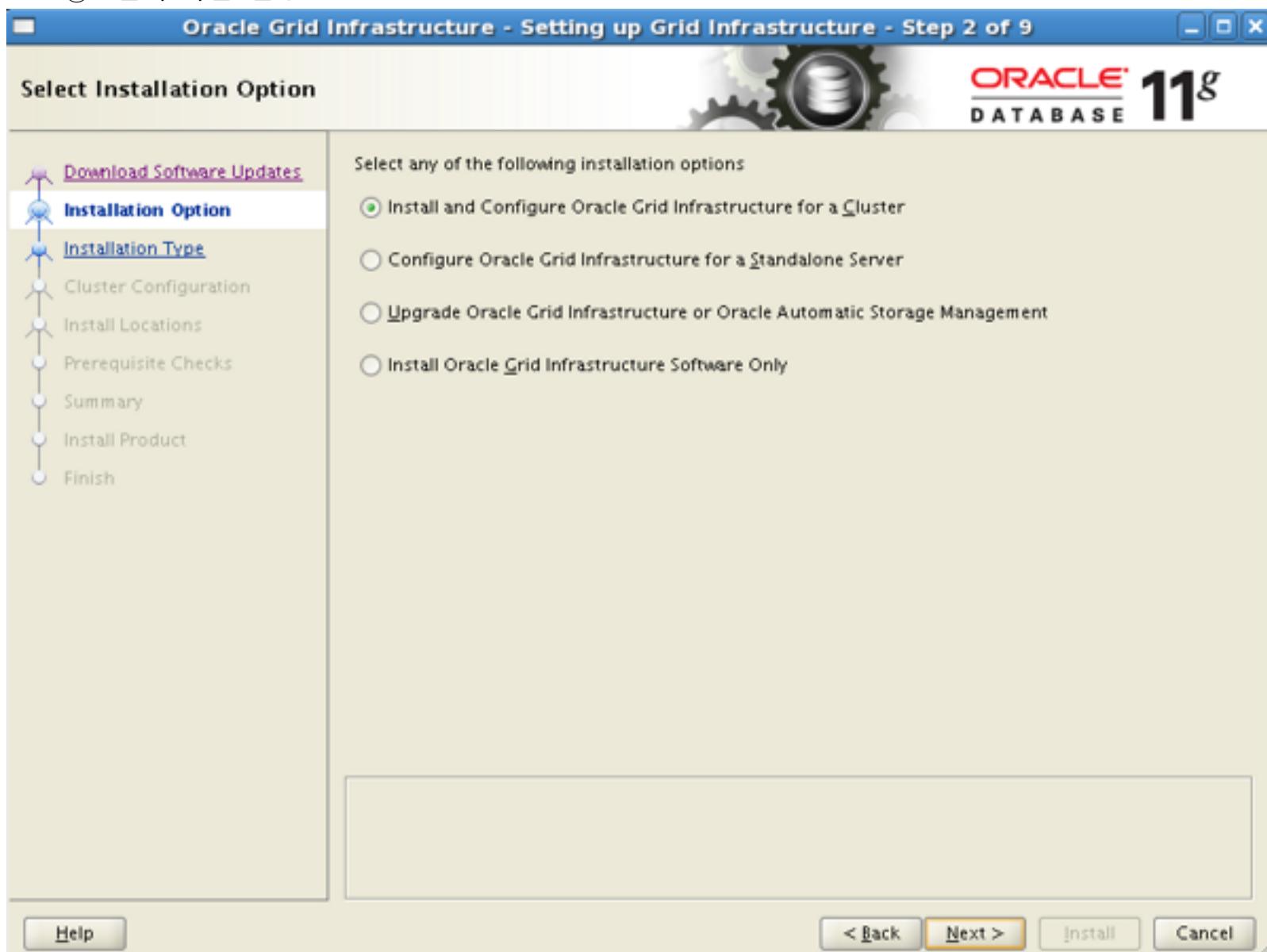
- 3개의 노드 구성 : host01, host02, host03
- 각 노드의 grid 계정 : Grid Infrastructure의 owner
- 각 노드의 oracle 계정 : RAC DB Software의 owner
- oracle, grid계정의 base directory 설정

B. Grid Infrastructure 설치

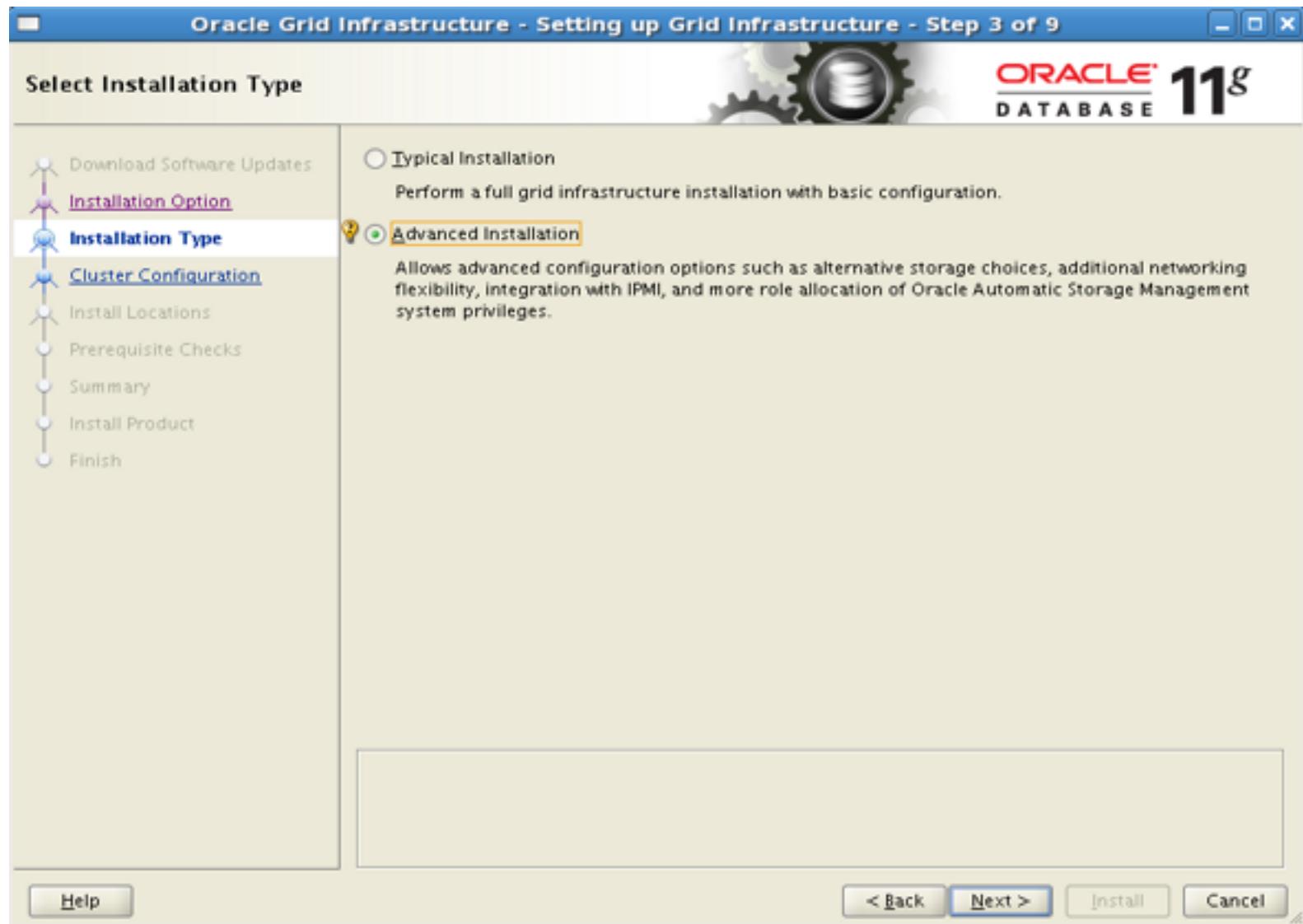
host01의 grid 계정으로 들어가서 runInstaller로 Grid Infrastructure 설치.

```
[grid@host01 ~]$ id  
uid=502(grid) gid=501(oinstall)  
groups=501(oinstall),504(asmadmin),505(asmdba),506(asmoper)  
  
[grid@host01 ~]$ cd /stage/grid  
  
[grid@host01 ~]$ ./runInstaller
```

① 설치 타입 설정



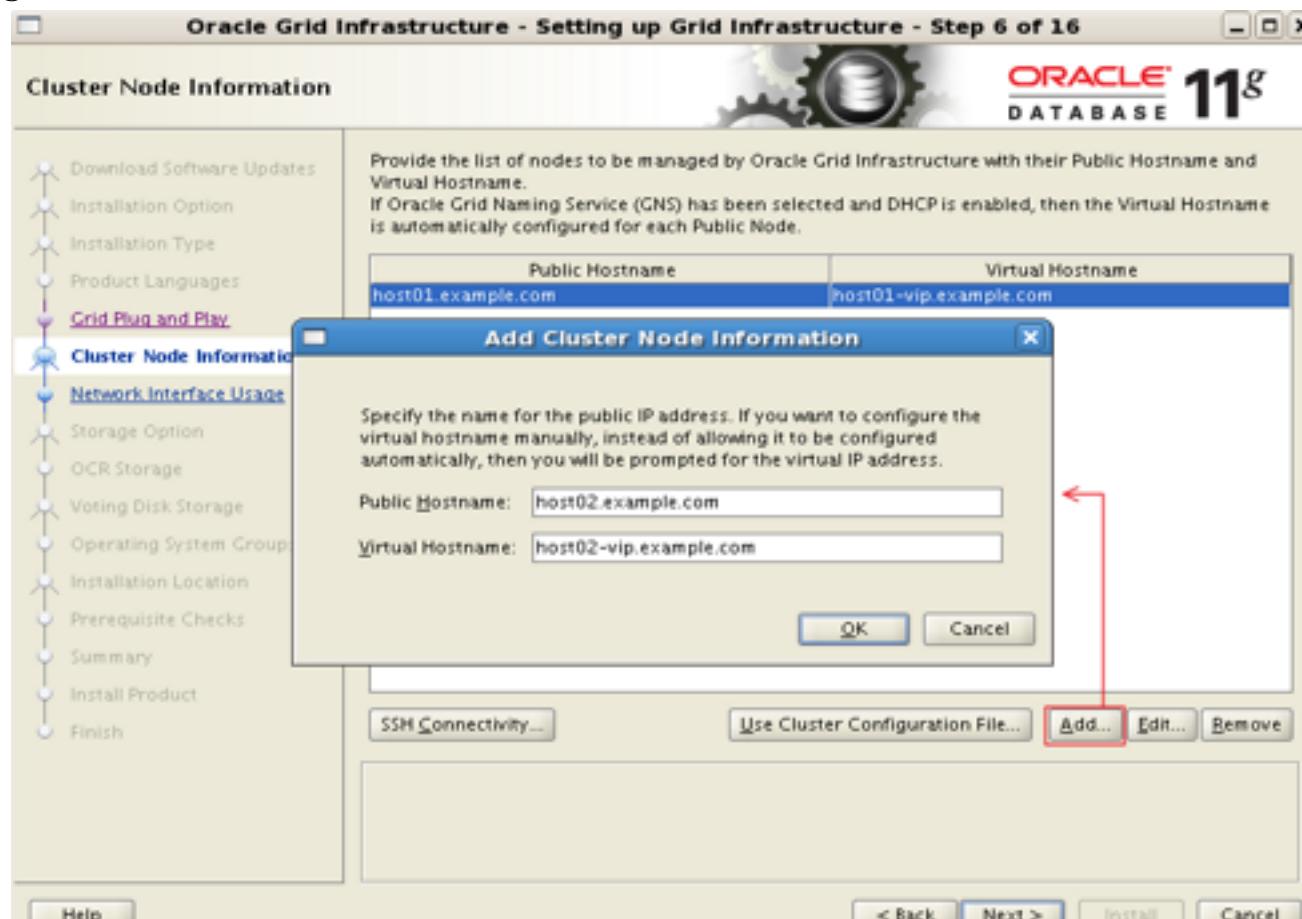
Cluster를 위한 Grid Infrastructure를 설치해야 하므로 Install and Configure Oracle Grid Infrastructure for a Cluster를 선택한다.



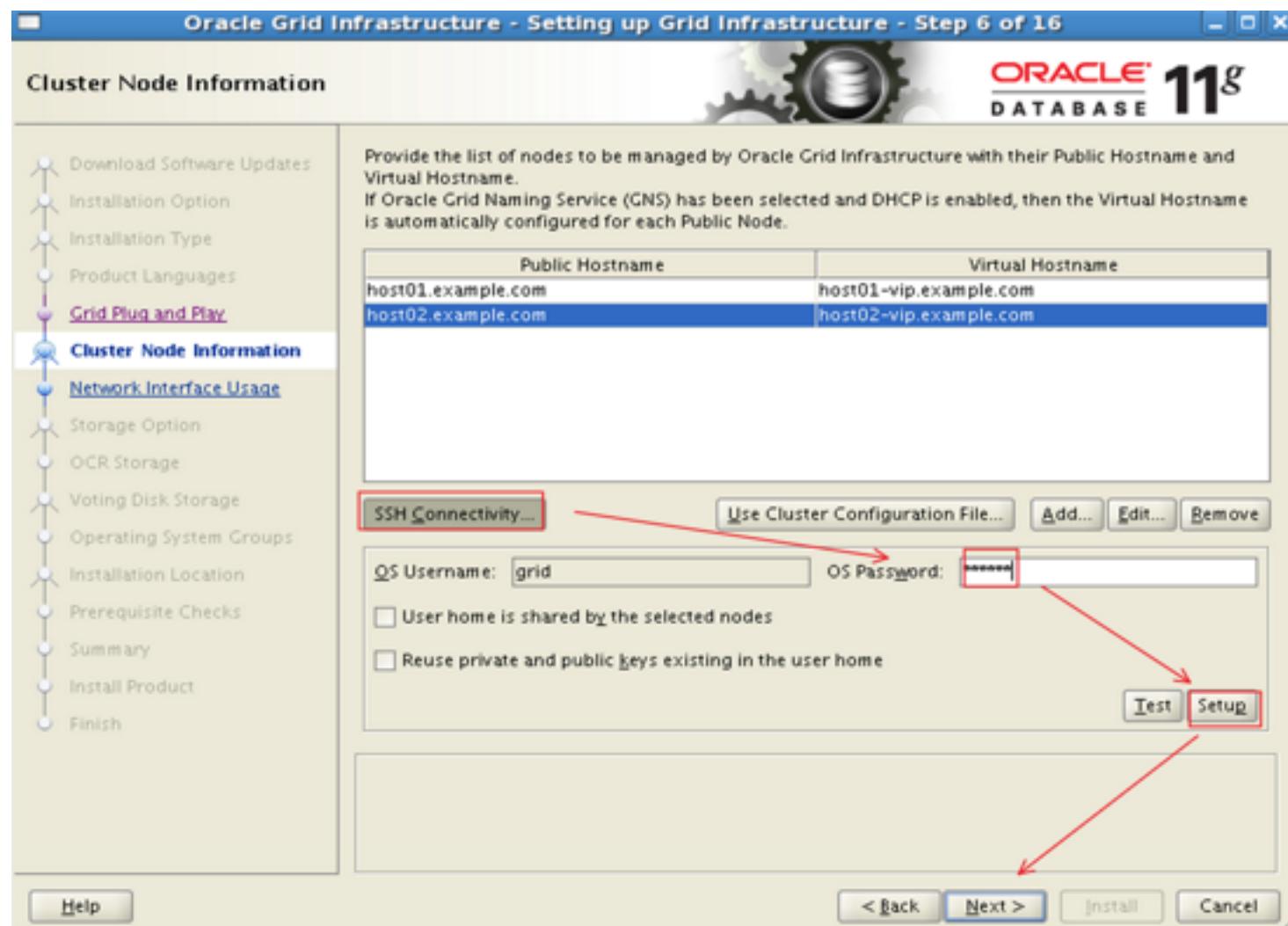
Network, IPMI, ASM의 구성을 사용자 설정에 맞게 바꿀 수 있도록 Advanced Installation을 선택

(만약, Basic한 환경설정을 원한다면 Typical Installation 선택한다)

- ② 클러스터 노드 추가 : 새로운 Node의 Hostname을 입력하여 추가한다.

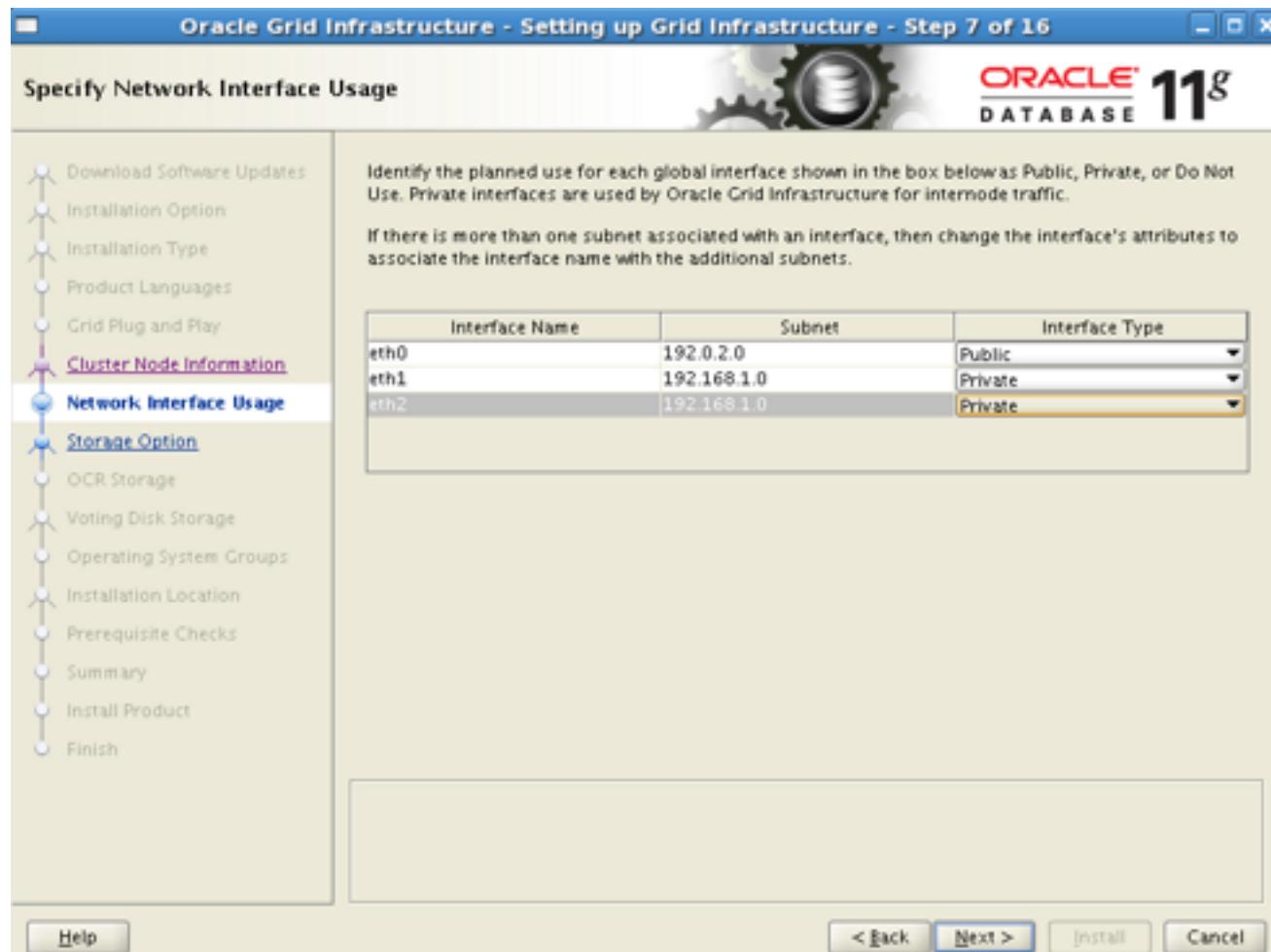


- ③ SSH 설정



SSH(Secure Shell)는 네트워크 보안도구 중 하나로 원격접속을 안전하게 할 수 있게 해주는 프로토콜이다. SSH Connectivity를 누른 후, 해당 노드의 Password를 입력한 뒤 Setup버튼을 눌러 SSH를 Setting한다.

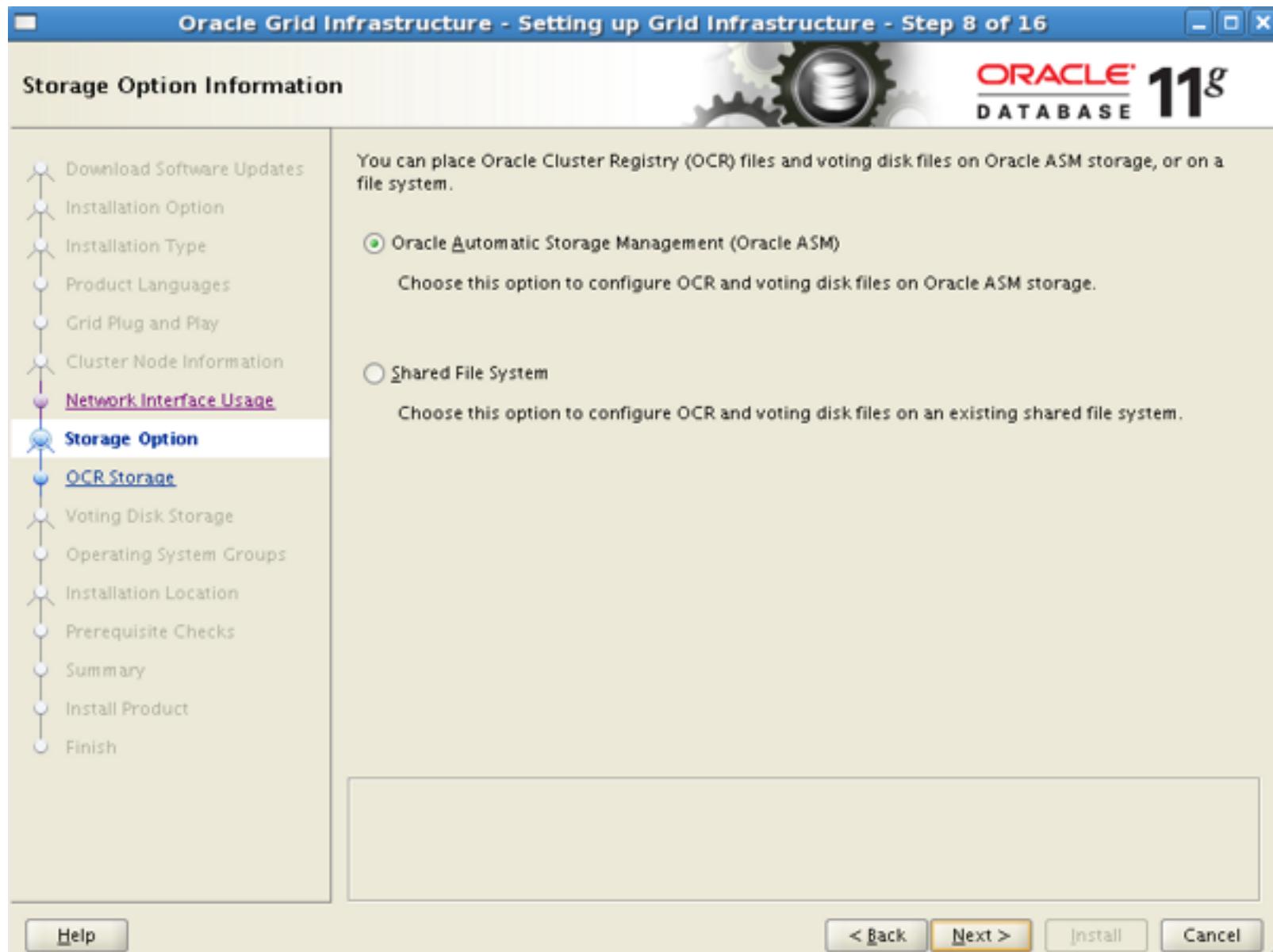
④ 네트워크 설정



eth0 인터페이스는 호스트의 기본 네트워크 인터페이스이므로 Public으로 선택하고

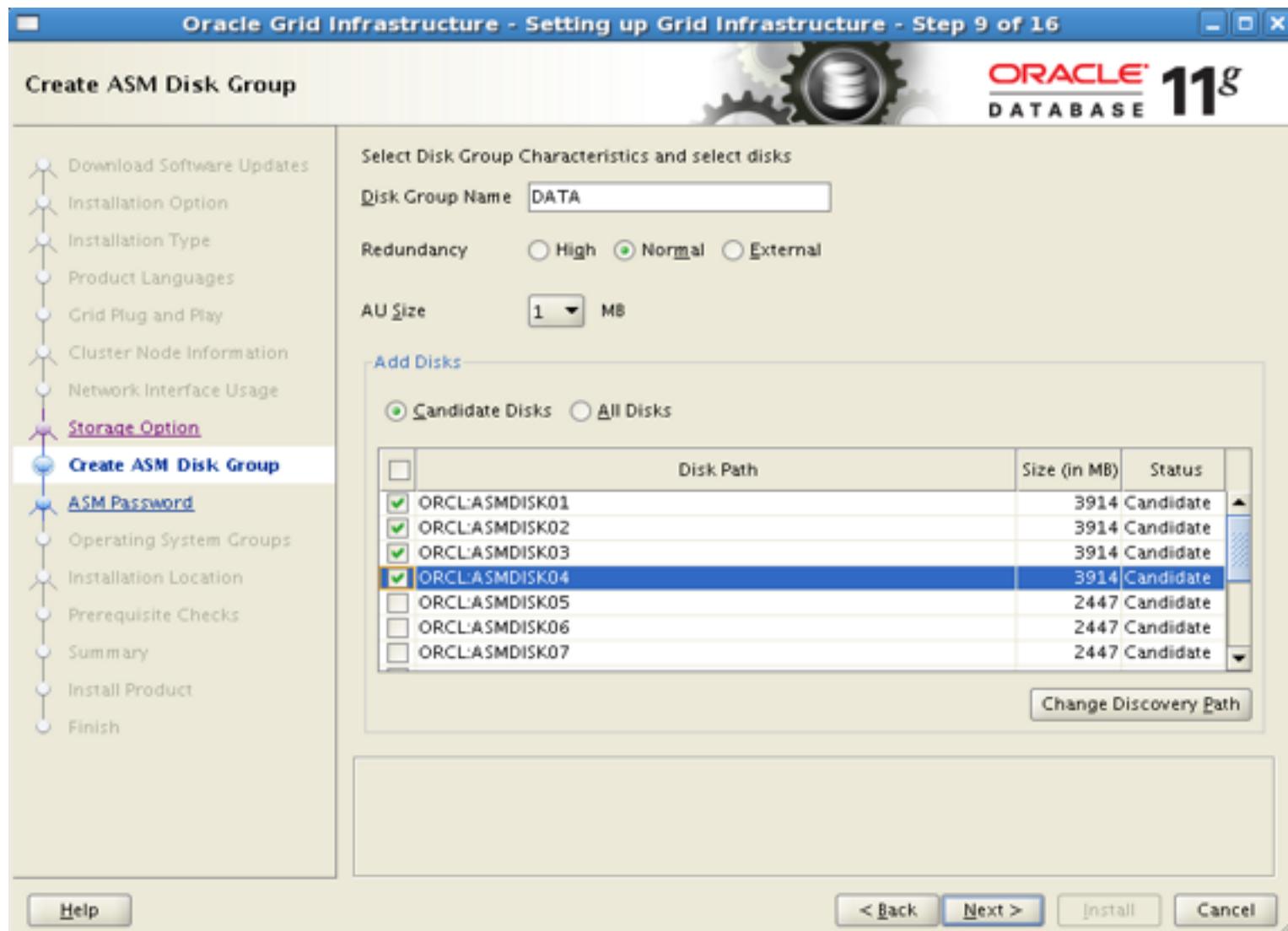
eth1, eth2 인터페이스는 노드 간의 인터페이스이므로 Private를 선택한다.

⑤ 클러스터 파일 저장 방법 설정



OCR과 Voting disk를 사용하기 위해 Automatic Storage Management(ASM)를 선택한다.

⑥ Disk Group 설정



미러링은 하지 않을 것이기 때문에 Redundancy를 External을 체크하고, Disk Group DATA의 구성 disk를 4개를 선택한다.

⑦ System Group 권한 설정 : Oracle ASM Group에게 알맞은 권한을 부여한다.

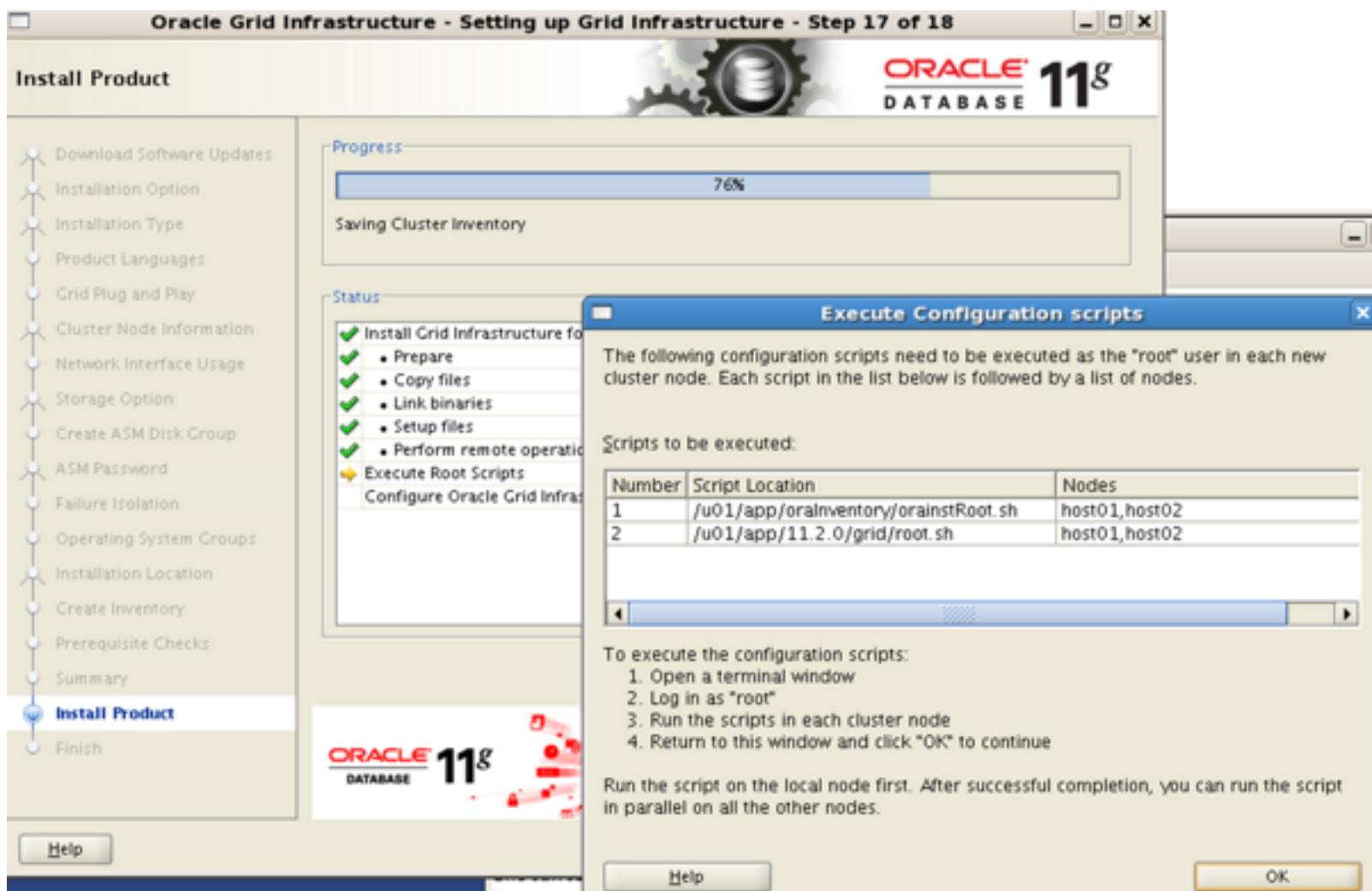


⑧ Oracle Grid Infrastructure base path, Inventory directory 설정



미리 설정해둔 grid계정의 base directory로 설정, Inventory Directory는 default로 설정한다.

⑨ 설치 마무리



위 메시지가 뜨면 각 노드에서 orainstRoot.sh와 root.sh를 실행해 준다. (OK누르지 말 것)

```
[root@host01 ~]# /u01/app/oraInventory/orainstRoot.sh
Changing permissions of /u01/app/oraInventory.
Adding read,write permissions for group.
Removing read,write,execute permissions for world.

Changing groupname of /u01/app/oraInventory to oinstall.
The execution of the script is complete.
[root@host01 ~]# /u01/app/11.2.0/grid/root.sh
Performing root user operation for Oracle 11g
```

```
The following environment variables are set as:
ORACLE_OWNER= grid
ORACLE_HOME= /u01/app/11.2.0/grid
```

```
Enter the full pathname of the local bin directory: [/usr/local/bin]:
Copying dbhome to /usr/local/bin ...
Copying oraenv to /usr/local/bin ...
```

... (중략) ...

```
## STATE File Universal Id File Name Disk group
-- -----
1. ONLINE 619044086c794f63bf06934349ad73b9 (ORCL:ASMDISK01) [DATA]
2. ONLINE 55263ee067854fecbf60088ed34daa95 (ORCL:ASMDISK02) [DATA]
3. ONLINE 26fc9dal1f3864f84bf679db518c9elf3 (ORCL:ASMDISK03) [DATA]
Located 3 voting disk(s).
CRS-2672: Attempting to start 'ora.asm' on 'host01'
CRS-2676: Start of 'ora.asm' on 'host01' succeeded
CRS-2672: Attempting to start 'ora.DATA.dg' on 'host01'
CRS-2676: Start of 'ora.DATA.dg' on 'host01' succeeded
CRS-2672: Attempting to start 'ora.registry.acfs' on 'host01'
CRS-2676: Start of 'ora.registry.acfs' on 'host01' succeeded
Configure Oracle Grid Infrastructure for a Cluster ... succeeded
[root@host01 ~]#
```

단, 한 노드에서의 실행이 끝난 뒤 다음 노드에서 실행해야 한다. (host01이후 host02에서 실행)

실행이 끝나면 메시지창의 OK를 눌러 창을 닫는다.

⑩ 설치 확인

```
[grid@host01 ~]$ /u01/app/11.2.0/grid/bin/crsctl stat res -t
```

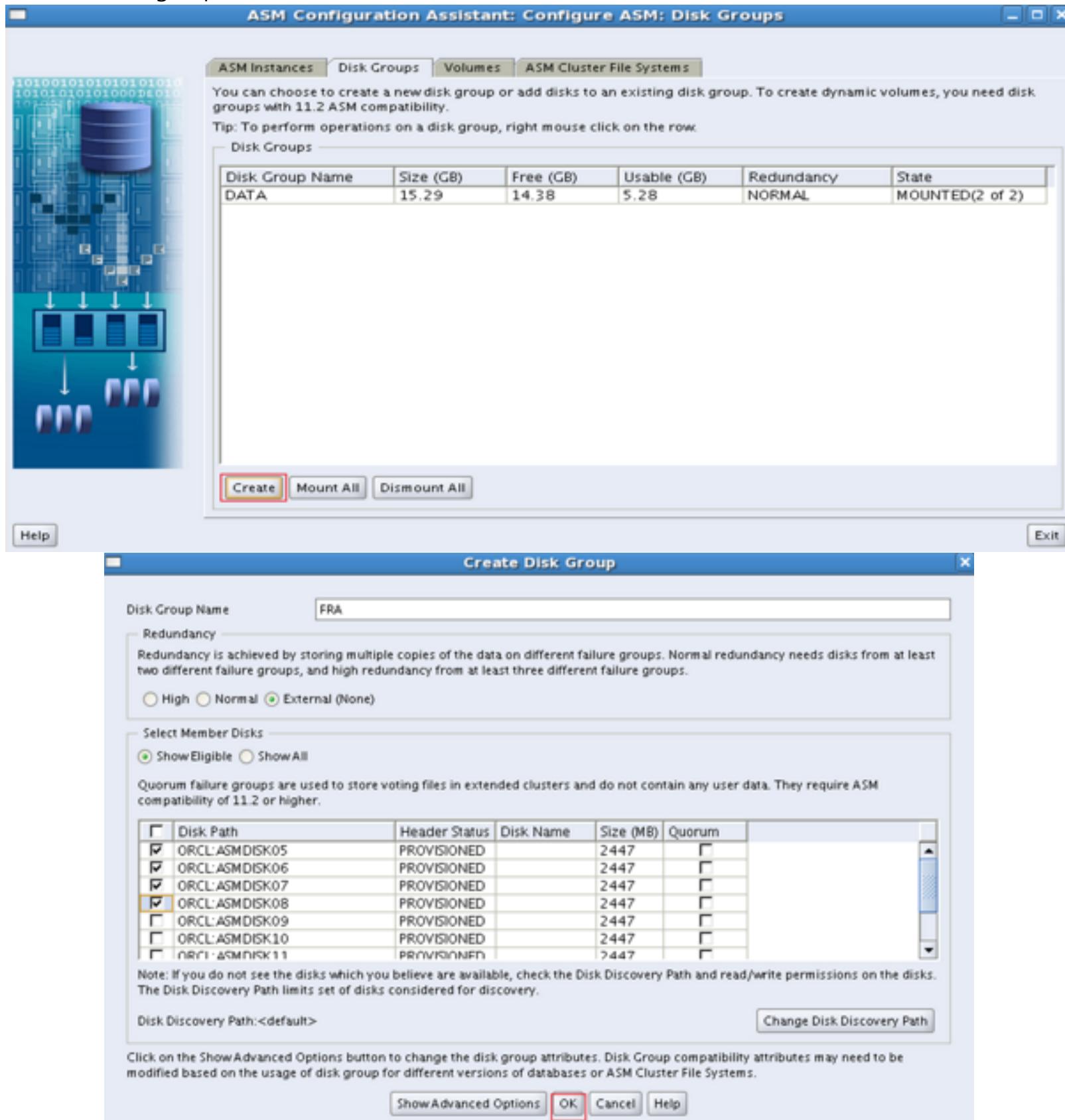
NAME	TARGET	STATE	SERVER	STATE_DETAILS
<hr/>				
Local Resources				
ora.DATA.dg				
	ONLINE	ONLINE	host01	
	ONLINE	ONLINE	host02	
ora.LISTENER.lsnr				
	ONLINE	ONLINE	host01	
	ONLINE	ONLINE	host02	
ora.asm				
	ONLINE	ONLINE	host01	Started
	ONLINE	ONLINE	host02	Started
ora.gsd				
	OFFLINE	OFFLINE	host01	
	OFFLINE	OFFLINE	host02	
ora.net1.network				
	ONLINE	ONLINE	host01	
	ONLINE	ONLINE	host02	
ora.ons				
	ONLINE	ONLINE	host01	
	ONLINE	ONLINE	host02	
ora.registry.acfs				
	ONLINE	ONLINE	host01	
	ONLINE	ONLINE	host02	
<hr/>				
Cluster Resources				
ora.LISTENER_SCAN1.lsnr				
1	ONLINE	ONLINE	host02	
ora.LISTENER_SCAN2.lsnr				
1	ONLINE	ONLINE	host01	
ora.LISTENER_SCAN3.lsnr				
1	ONLINE	ONLINE	host01	
ora.cvu				
1	ONLINE	ONLINE	host01	
ora.host01.vip				
1	ONLINE	ONLINE	host01	
ora.host02.vip				
1	ONLINE	ONLINE	host02	
ora.oc4j				
1	ONLINE	ONLINE	host01	
ora.scan1.vip				
1	ONLINE	ONLINE	host02	
ora.scan2.vip				
1	ONLINE	ONLINE	host01	
ora.scan3.vip				
1	ONLINE	ONLINE	host01	
grid@host01 ~%				

C. FRA를 위한 추가적인 ASM Disk Groups 생성

- ① host01노드의 grid계정으로 가서 환경변수를 ASM로 설정한 후 asmca실행

```
[root@host01 ~]# ssh -X grid@host01
grid@host01's password:
Last login: Wed Apr  4 05:12:32 2018 from host01.example.com
[grid@host01 ~]$
[grid@host01 ~]$ . oraenv
ORACLE_SID = [grid] ? +ASM1
The Oracle base has been set to /u01/app/grid
[grid@host01 ~]$ asmca
```

- ② FRA disk group생성



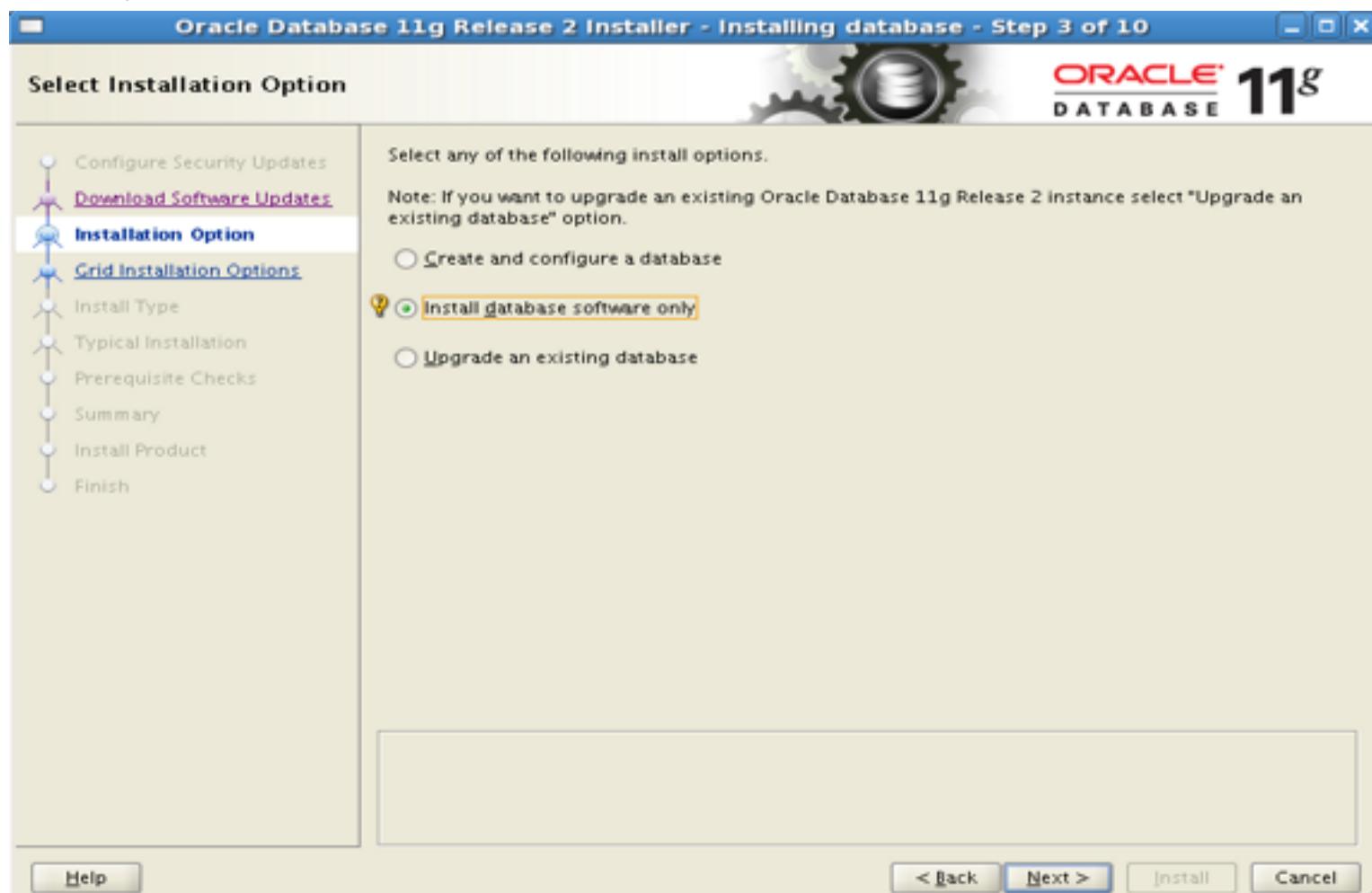
- ③ 확인

```
[grid@host01 grid]$ crsctl stat res -t
-----
NAME          TARGET  STATE        SERVER          STATE_DETAILS
-----
Local Resources
-----
ora.DATA.dg
    ONLINE  ONLINE   host01
    ONLINE  ONLINE   host02
ora.FRA.dg
    ONLINE  ONLINE   host01
    ONLINE  ONLINE   host02
ora.LISTENER.lsnr
    ONLINE  ONLINE   host01
    ONLINE  ONLINE   host02
ora.asm
    ONLINE  ONLINE   host01          Started
    ONLINE  ONLINE   host02          Started
ora.gsd
    OFFLINE OFFLINE  host01
    OFFLINE OFFLINE  host02
```

2) RAC DB설치

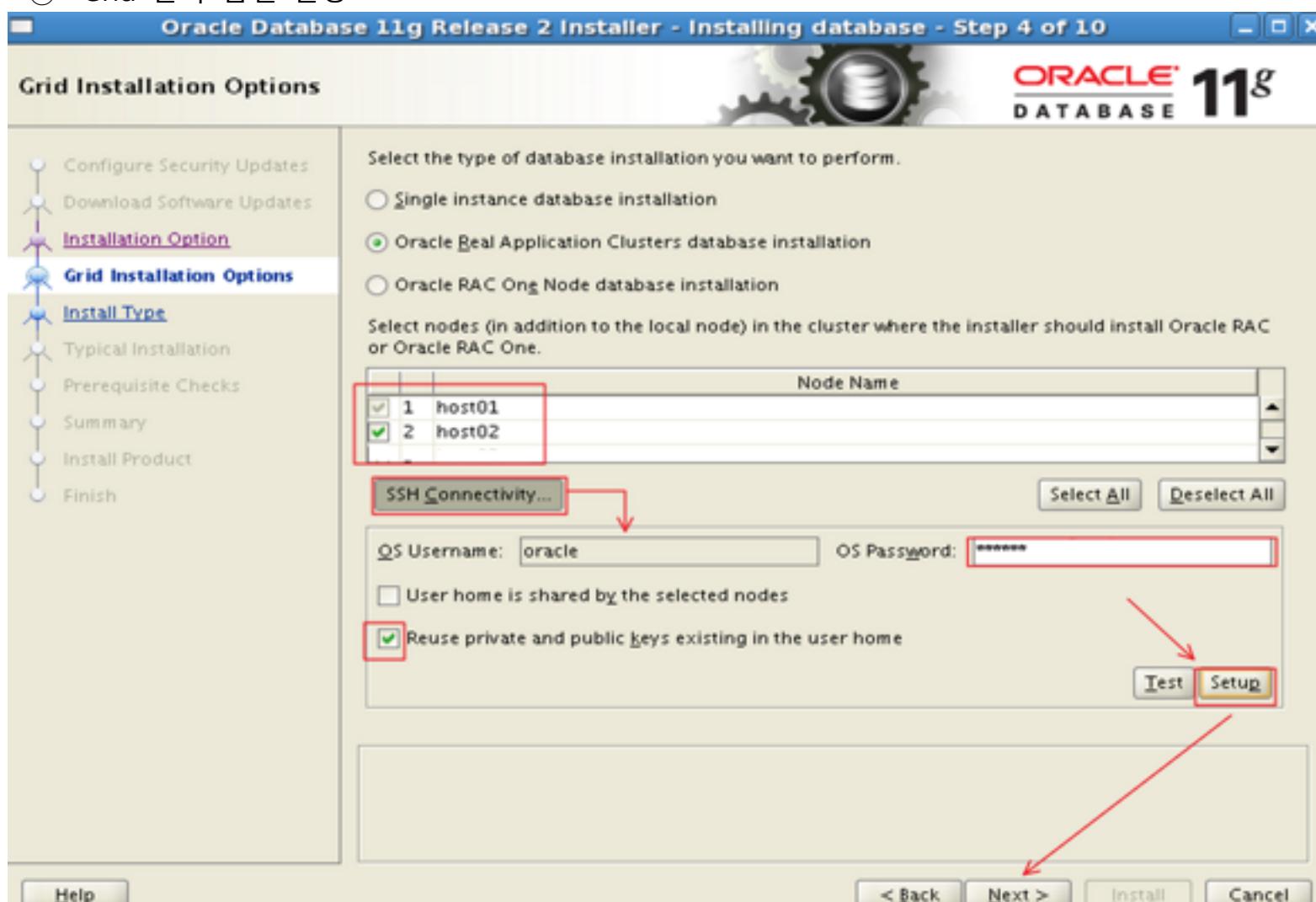
A. Oracle Database Software설치

- ① host01의 oracle계정으로 들어가서 Database설치파일이 있는 위치로 이동하여 runInstaller를 실행 한다.



Database Software만 설치할 것이기 때문에 Install database software only를 체크한다.

- ② Grid 설치 옵션 설정



③ Oracle base directory path 설정

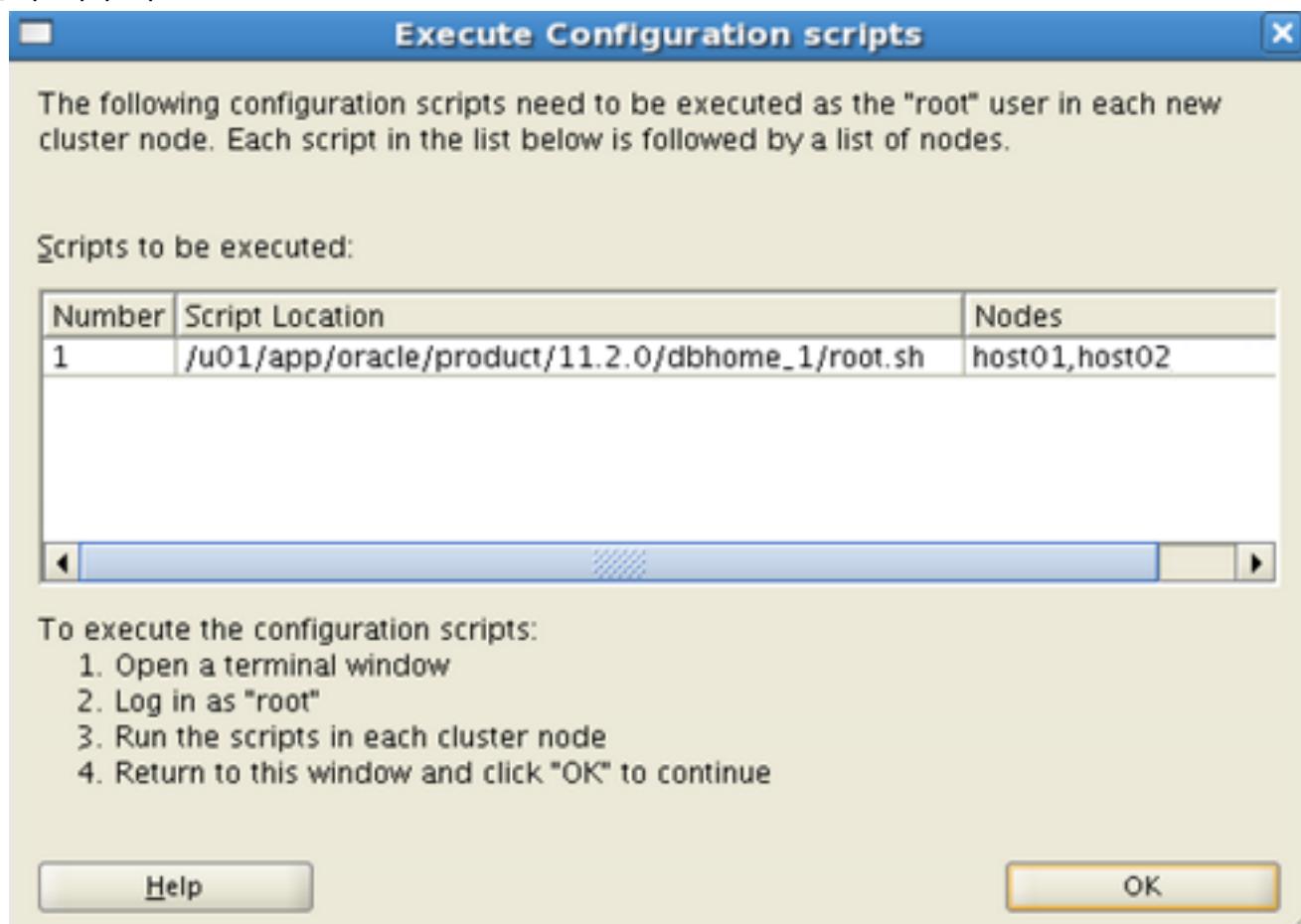


미리 설정해둔 oracle계정의 base directory로 설정한다.

④ System Group 권한 설정 : OSDBA와 OSOPER Group에 알맞은 권한을 부여한다.



⑤ 설치 마무리

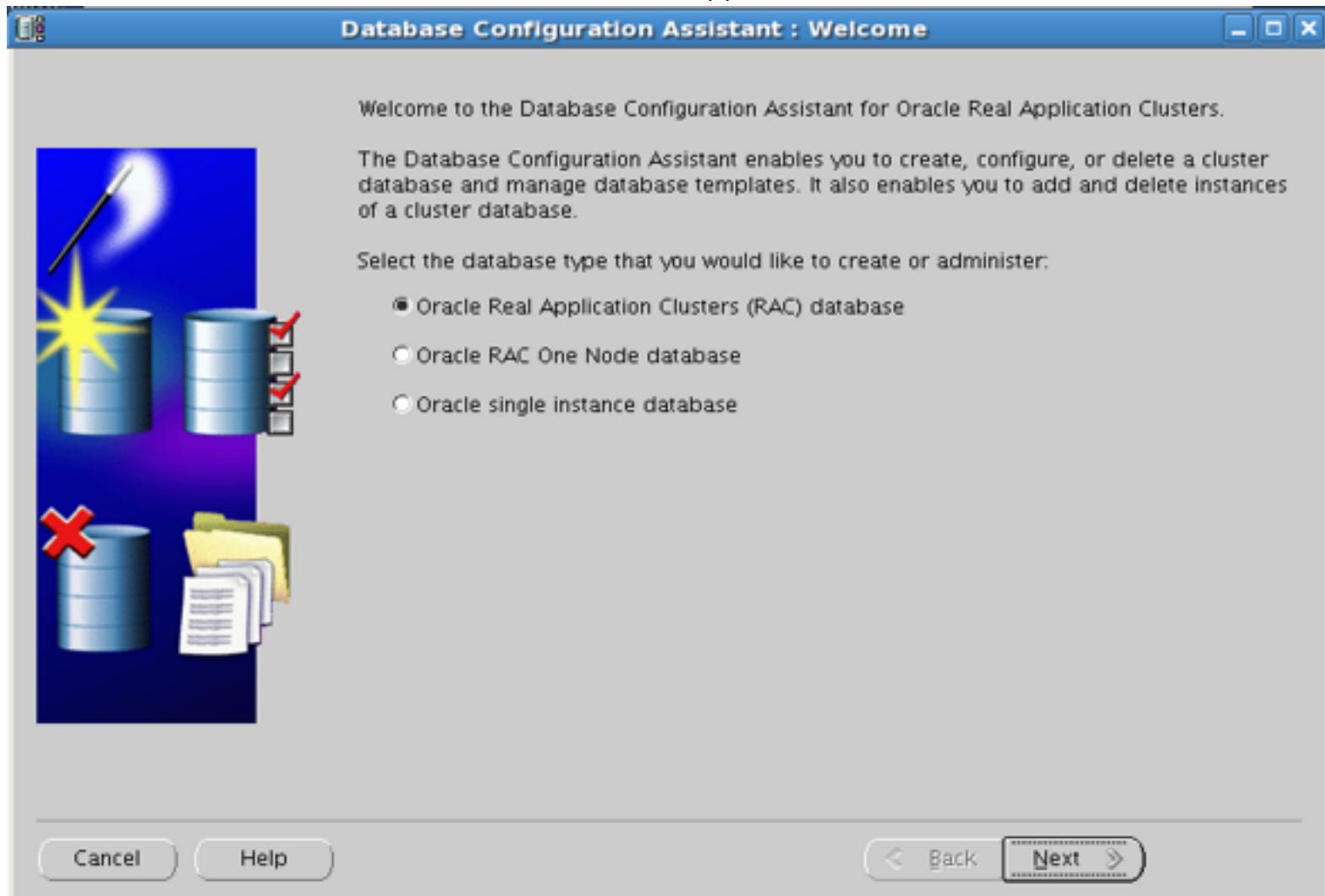


Root Scripts를 실행하라는 메시지가 뜨면 각 노드에서 Root계정으로 root.sh를 실행한다.

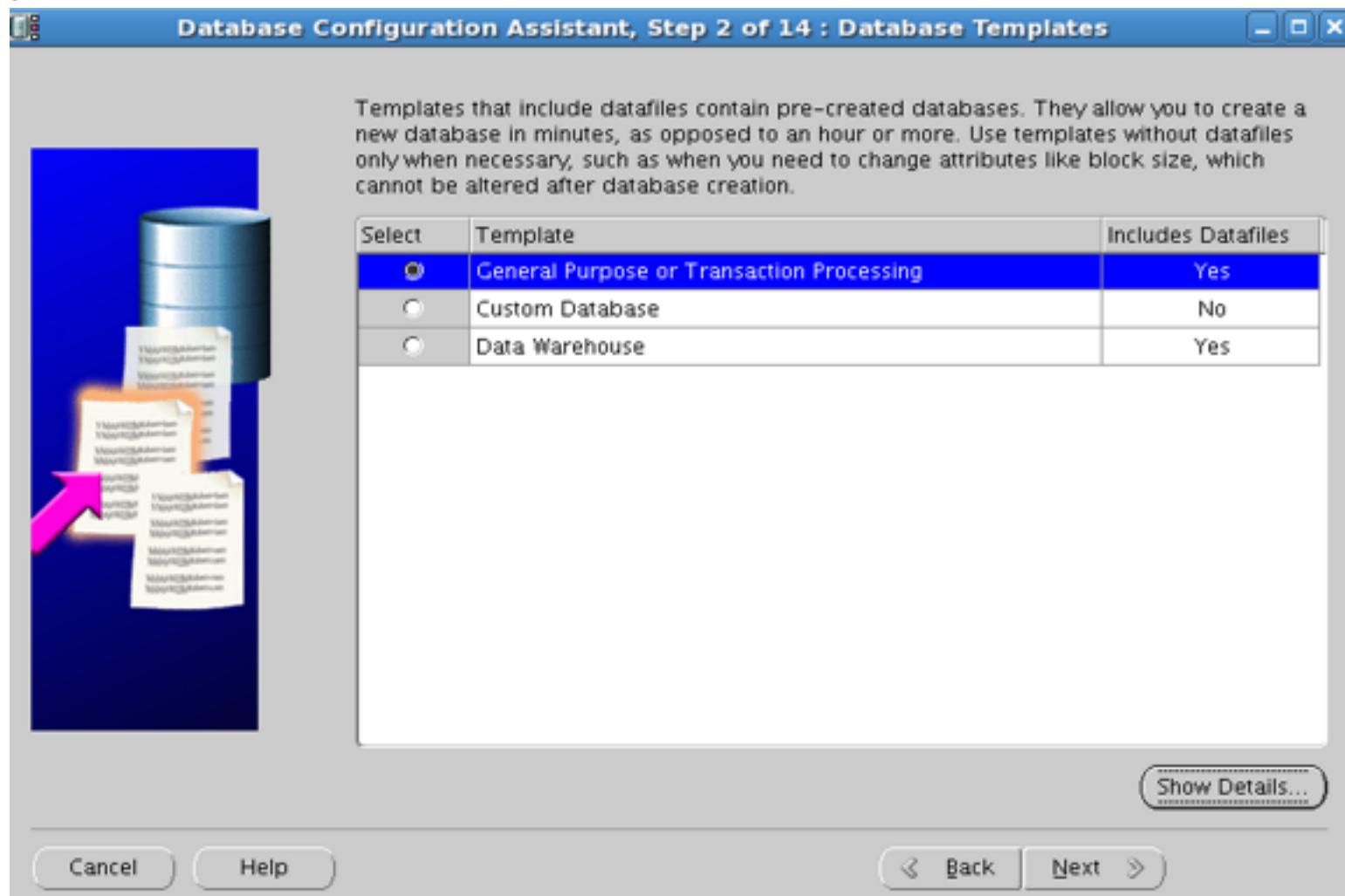
마찬가지로, 한 노드에서의 실행이 끝난 뒤 다음 노드에서 실행해야 한다.

B. DBCA를 사용하여 RAC DB 설치

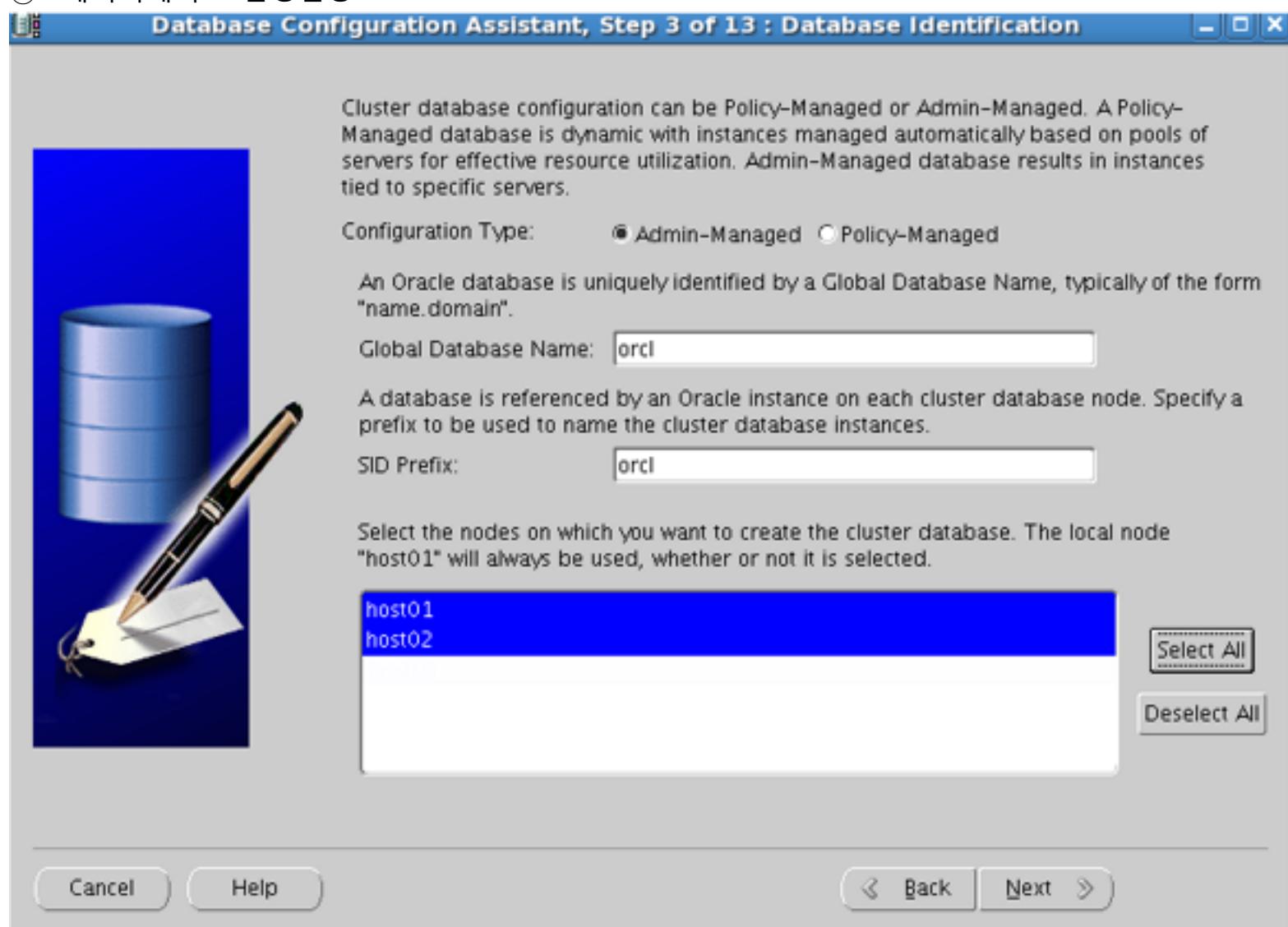
- ① RAC database를 설치 해야 하므로 Oracle Real Application Clusters(RAC) database를 선택한다.



② 데이터베이스 탑업 설정

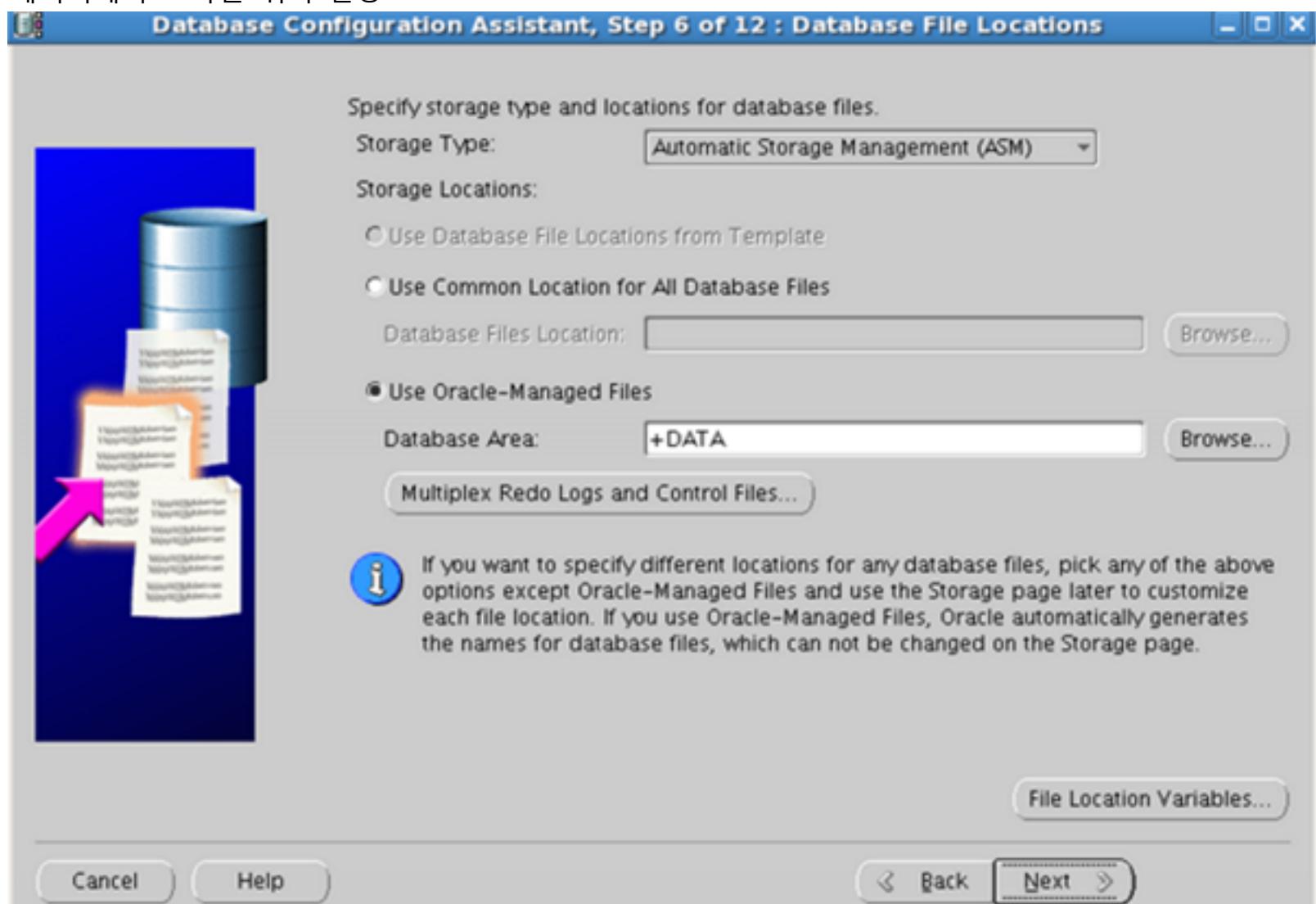


③ 데이터베이스 환경설정



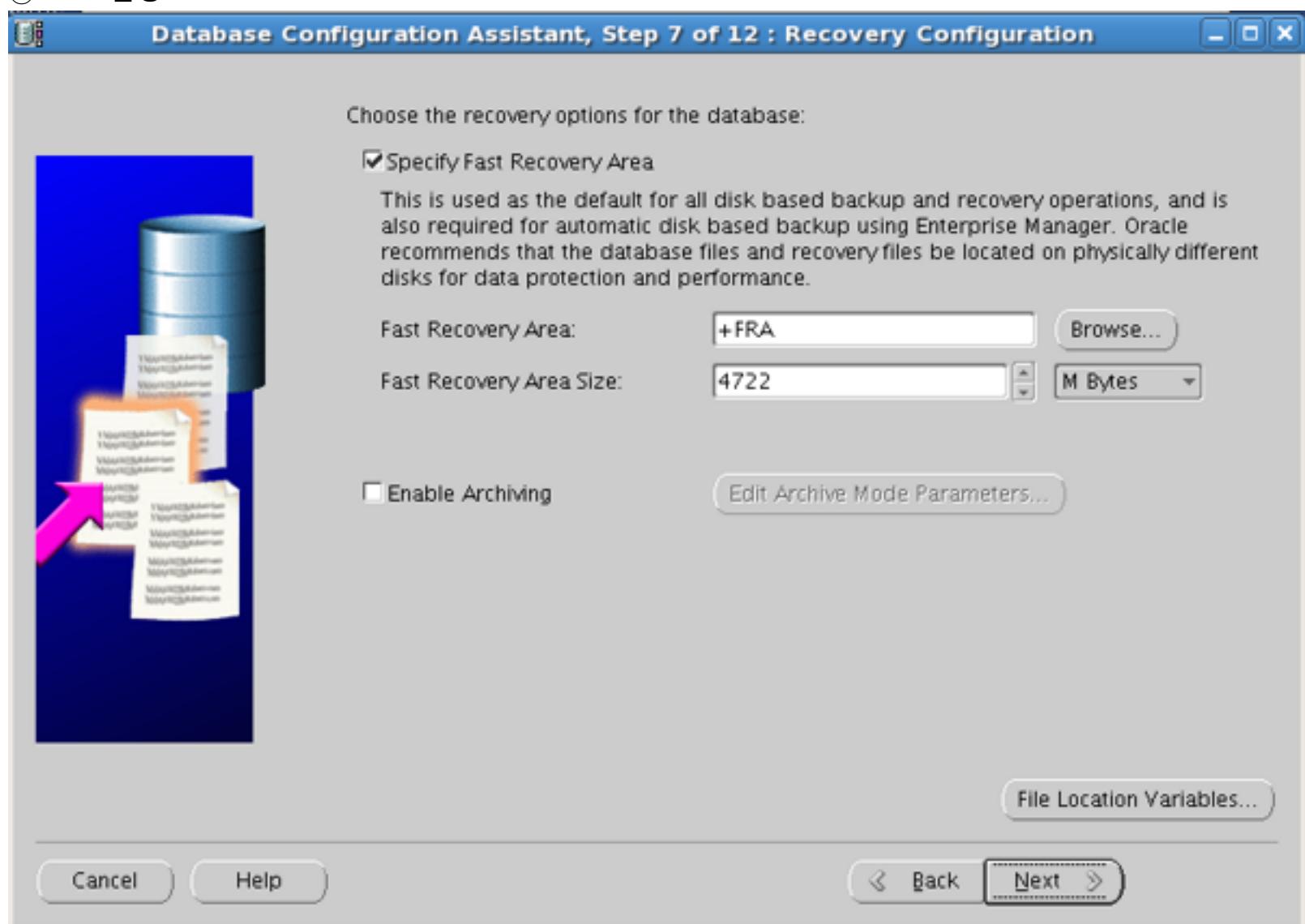
Administrator-managed Type은 RAC instance가 실행 될 클러스터 노드 리스트를 명시하는 Type이다.

④ 데이터베이스 파일 위치 설정

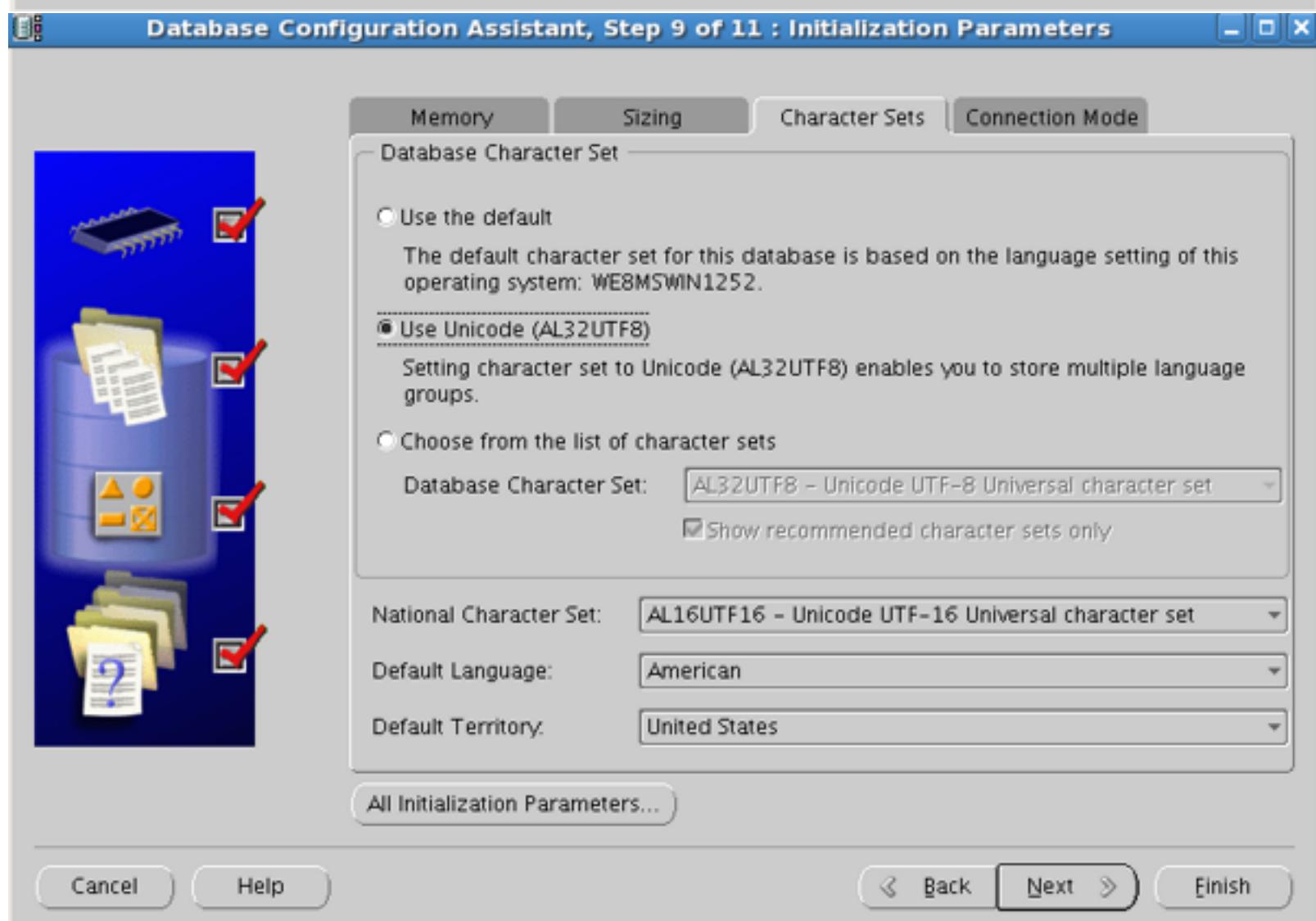
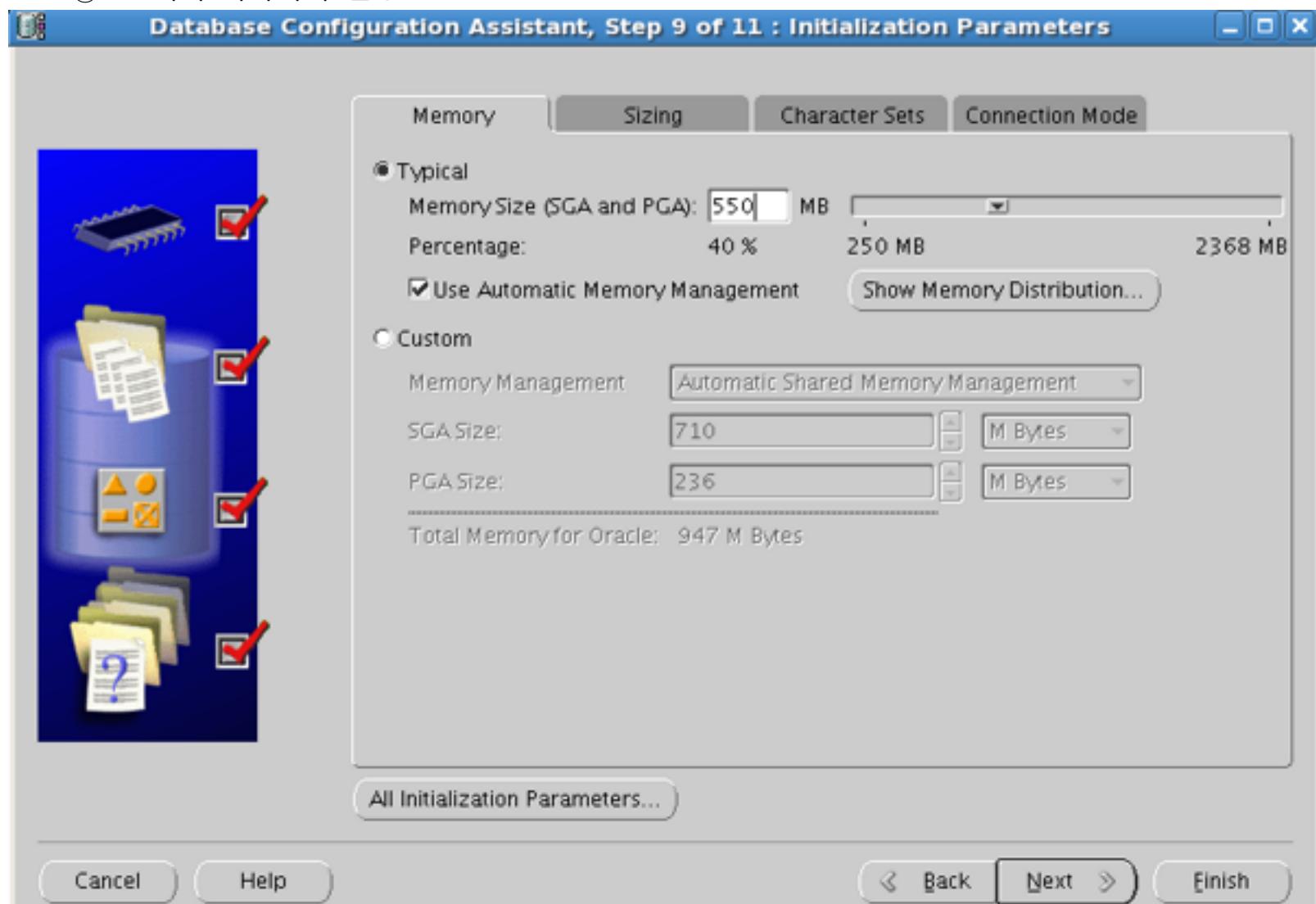


Storage Type은 ASM으로, Storage Location은 OMF방식으로 설정한다.

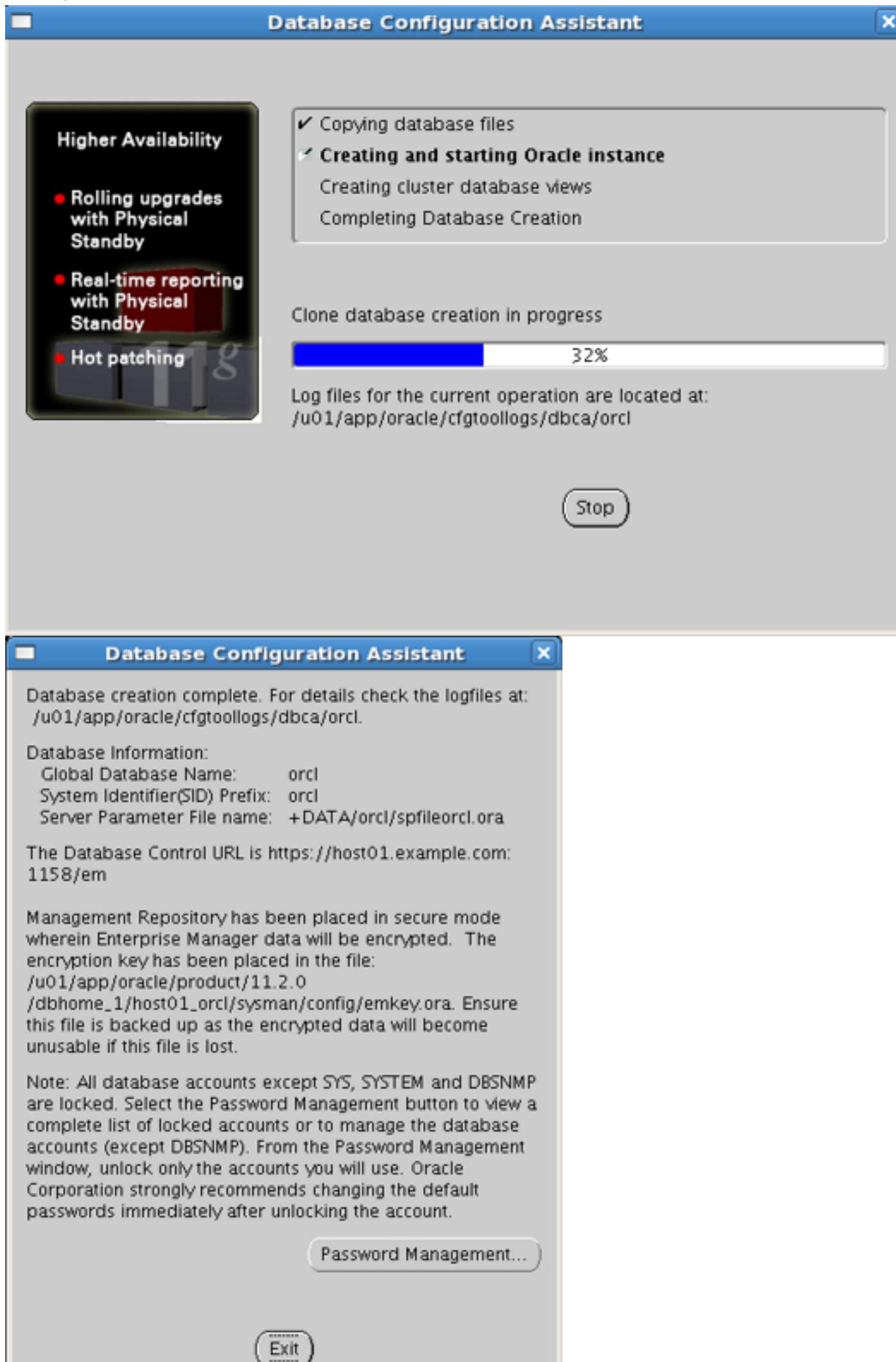
⑤ FRA설정



⑥ 초기화 파라미터 설정



⑦ 설치 마무리



DBCA가 실시간으로 수행하는 특정 태스크에 대해 알려주고, Progress가 100%에 다다르면 설치 로그 파일 위치, 매개 변수 파일 위치 및 Enterprise Manager URL을 보여준다.

⑧ 설치 확인

```
[grid@host01 ~]$ srvctl config database -d orcl
Database unique name: orcl
Database name: orcl
Oracle home: /u01/app/oracle/product/11.2.0/dbhome_1
Oracle user: oracle
Spfile: +DATA/orcl/spfileorcl.ora
Domain:
Start options: open
Stop options: immediate
Database role: PRIMARY
Management policy: AUTOMATIC
Server pools: orcl
Database instances: orcl1,orcl2
Disk Groups: DATA,FRA
Mount point paths:
Services:
Type: RAC
Database is administrator managed
```

5. Cache Fusion 확인

[SQL*Plus로 구현]

- host01에서 두 개의 터미널을 열고 orcl1, orcl2 인스턴스에 각각 접속하여 실행

- Read - Read

Terminal 1

```
[oracle@host01 labs]$ srvctl stop database -d orcl
[oracle@host01 labs]$ srvctl start database -d orcl
[oracle@host01 labs]$ sqlplus sys/oracle_4U@orcl1 as sysdba
```

```
SQL*Plus: Release 11.2.0.4.0 Production on Mon Dec 4 09:39:58 2017
```

```
Copyright (c) 1982, 2013, Oracle. All rights reserved.
```

Connected to:

```
Oracle Database 11g Enterprise Edition Release 11.2.0.4.0 - 64bit Production
With the Partitioning, Real Application Clusters, Automatic Storage Management,
OLAP,
Data Mining and Real Application Testing options
```

```
SYS@orcl1> SELECT inst_id, object_name, statistic_name, value
  FROM gv$segment_statistics
 WHERE object_name = 'EMP'
   AND statistic_name IN ('physical read requests','physical reads'
  , 'gc cr blocks received','gc current blocks received') ; 2      3      4      5
```

no rows selected

```
SYS@orcl1> SELECT empno
  , ename
  , dbms_rowid.rowid_relative_fno(rowid) as fno
  , dbms_rowid.rowid_block_number(rowid) as blkno
  FROM scott.emp
 WHERE empno = 7788 ; 2      3      4      5      6
```

EMPNO	ENAME	FNO	BLKNO
7788	SCOTT	4	151

Elapsed: 00:00:00.06

```
SYS@orcl1> SELECT inst_id, object_name, statistic_name, value
  FROM gv$segment_statistics
 WHERE object_name = 'EMP'
   AND statistic_name IN ('physical read requests','physical reads'
  , 'gc cr blocks received','gc current blocks received') ; 2      3      4      5
```

INST_ID	OBJECT_NAME	STATISTIC_NAME	VALUE
1	EMP	physical reads	6
1	EMP	physical read requests	2
1	EMP	gc cr blocks received	0
1	EMP	gc current blocks received	0

(이 구문 save seg_stat로 저장해두기)

```
SYS@orcl1> SELECT inst_id, file#, block#, status, dirty
  FROM gv$bh
 WHERE file# = 4
   AND block# = 151
   AND inst_id = 1 ; 2 3 4 5
```

INST_ID	FILE#	BLOCK#	STATUS	D
1	4	151	scur	N

(이 구문 save bh1으로 저장해두기)

>> select한 블록을 Disk에서 읽어왔음을 확인 할 수 있고 1번 instance에서 SCur상태임을 확인할 수 있다.

```
[oracle@host01 Desktop]$ sqlplus sys/oracle_4U@orcl2 as sysdba
SQL*Plus: Release 11.2.0.4.0 Production on Mon Dec 4 09:48:57 2017
Copyright (c) 1982, 2013, Oracle. All rights reserved.
```

Connected to:

Oracle Database 11g Enterprise Edition Release 11.2.0.4.0 - 64bit Production
With the Partitioning, Real Application Clusters, Automatic Storage Management, OLAP,
Data Mining and Real Application Testing options

```
SYS@orcl2> select empno, ename, deptno
  2  from scott.emp
  3  where empno = 7788;
```

EMPNO	ENAME	DEPTNO
7788	SCOTT	20

>> 2번 instance에서 동일한 블록을 select하기(read작업)

Terminal 2

```
[oracle@host01 labs]$ sqlplus sys/oracle_4U@orcl2 as sysdba
```

```
SQL*Plus: Release 11.2.0.4.0 Production on Mon Dec 4 09:51:54 2017
```

```
Copyright (c) 1982, 2013, Oracle. All rights reserved.
```

```
Connected to:
```

```
Oracle Database 11g Enterprise Edition Release 11.2.0.4.0 - 64bit Production  
With the Partitioning, Real Application Clusters, Automatic Storage Management, OLAP,  
Data Mining and Real Application Testing options
```

```
SYS@orcl2> @cache
```

INST_ID	FILE#	BLOCK#	STATUS	INST_ID	MODE_HELD
1	4	151	scur	1	1
1	4	151	scur	2	1
2	4	151	scur	2	1

>> 2번 instance에서 동일한 블록을 select하면 둘 모두 Shared Current상태임을 확인할 수 있다.

```
SYS@orcl2> SELECT inst_id, file#, block#, status, dirty  
FROM gv$bh  
WHERE file# = 4  
AND block# = 151  
AND inst_id = 2 ; 2 3 4 5
```

INST_ID	FILE#	BLOCK#	STATUS	D
2	4	151	scur	N

(이 구문 save bh2로 저장)

```
SYS@orcl2> @seg_stat
```

INST_ID	OBJECT_NAME	STATISTIC_NAME	VALUE
1 EMP		physical reads	6
1 EMP		physical read requests	2
1 EMP		gc cr blocks received	0
1 EMP		gc current blocks received	0
2 EMP		physical reads	1
2 EMP		physical read requests	1
2 EMP		gc cr blocks received	0
2 EMP		gc current blocks received	1

```
8 rows selected.
```

>> Cache Fusion 을 통해 Disk I/O 없이 1번 instance에서 가져온 블록 한 개가 저장되어 있음을 확인할 수 있다.

- Read – Write

```
SYS@orcl1> @bh1
```

INST_ID	FILE#	BLOCK#	STATUS	D
1	4	151	scur	N

(1번 instance에서의 현재 블록 상태 확인 : SCur)

```
SYS@orcl1> update scott.emp  
2 set sal = 4000  
3 where empno = 7788 ;
```

```
1 row updated.
```

```
SYS@orcl1> @cache
```

INST_ID	FILE#	BLOCK#	STATUS	INST_ID	MODE_HELD
1	4	151	xcur	1	2
1	4	151	xcur	2	1

```
SYS@orcl1> @bh1
```

INST_ID	FILE#	BLOCK#	STATUS	D
1	4	151	xcur	Y

>> DML작업으로 인해 1번 instance의 블록 상태는 XCur로 변경되었다.

```
SYS@orcl1> @bh2
```

INST_ID	FILE#	BLOCK#	STATUS	D
2	4	151	cr	N

>> 2번 instance에서는 블록이 CR상태로 변경되었다.

- Write – Read (Before commit)

Terminal 2

```
SYS@orcl1> @bh2
```

INST_ID	FILE#	BLOCK#	STATUS	D
2	4	151	cr	N

(2번 instance에서의 현재 블록 상태 확인 : CR)

```
SYS@orcl2> SELECT empno, ename, sal, deptno  
FROM scott.emp  
WHERE empno = 7788 ; 2 3
```

EMPNO	ENAME	SAL	DEPTNO
7788	SCOTT	3000	20

```
SYS@orcl2> @bh2
```

INST_ID	FILE#	BLOCK#	STATUS	D
2	4	151	cr	N
2	4	151	cr	N

```
SYS@orcl2> @cache
```

INST_ID	FILE#	BLOCK#	STATUS	INST_ID	MODE_HELD
1	4	151	xcur	1	2
1	4	151	xcur	2	1

>> 1번 instance에서 CR Block 을 새로 받았으므로 두 개의 CR Block 이 존재한다.

1번 instance에서 수정된 값이 아직 commit 되지 않았으므로 그 이전의 결과를 보여 줘야 한다.

- Write – Read (After commit)

Terminal 1

```
SYS@orcl1> @bh1
```

INST_ID	FILE#	BLOCK#	STATUS	D
1	4	151	xcur	Y
1	4	151	cr	N

>> 1번 instance에서는 동일 버퍼의 재 접근은 없었으나 Cache Fusion 을 통해 2번 instance로의 전송을 위해 Consistent Read 모드의 버퍼가 생성 되었다.

```
SYS@orcl1> commit  
2 ;
```

Commit complete.

```
SYS@orcl1> @cache
```

INST_ID	FILE#	BLOCK#	STATUS	INST_ID	MODE_HELD
1	4	151	xcur	1	2
1	4	151	xcur	2	1

```
SYS@orcl1> @bh1
```

INST_ID	FILE#	BLOCK#	STATUS	D
1	4	151	xcur	N
1	4	151	cr	N

>> 현재 1번 인스턴스에 캐싱되어 있는 버퍼는 commit 이전과 동일하다.

**Dirty Buffer 컬럼은 Y, N 둘 중 하나의 값을 보여 줄 수 있다. commit 은 Dirty Buffer를 Writing 시키지 않으므로 때에 따라 Y, Checkpoint 나 기타 다른 이유에 의해서 내려쳤다면 N을 보여 줄 수도 있다.

```
SYS@orcl1> ALTER SYSTEM CHECKPOINT GLOBAL ;
```

System altered.

```
SYS@orcl1> @bh1
```

INST_ID	FILE#	BLOCK#	STATUS	D
1	4	151	xcur	N
1	4	151	cr	N

Terminal 2

```
SYS@orcl2> SELECT empno, ename, sal, deptno
  FROM scott.emp
 WHERE empno = 7788 ; 2 3
```

EMPNO	ENAME	SAL	DEPTNO
7788	SCOTT	4000	20

>> COMMIT 이후에 실행 된 SELECT 문장이므로 변경 된 4000의 값을 확인 할 수 있다.

```
SYS@orcl2> @cache
```

INST_ID	FILE#	BLOCK#	STATUS	INST_ID	MODE_HELD
1	4	151	scur	1	1
1	4	151	scur	2	1

```
SYS@orcl2> @bh2
```

INST_ID	FILE#	BLOCK#	STATUS	D
2	4	151	cr	N
2	4	151	cr	N
2	4	151	cr	N

>> COMMIT 이 끝난 이미지를 2번 instance에서 select하였으나 2번 instance는 SCur 상태의 블록을 얻지 못하고 CR블록이 추가 되었다.

** XCur모드로 획득한 버퍼에 대해 SCur모드로 전송하는 작업은 일정 횟수 이상 지연하게 한다. 이는 Cache Fusion 의 작업 중 가장 값 비싼 작업을 너무 쉽게 해제 하지 않기 위함이다. (hidden parameter 를 통해서 조절 가능하다.)

```
SYS@orcl2> @hidden
Enter value for parm_name: fairness_threshold
```

Parameter	Value	SES_MODIF	Description
_fairness_threshold	2	false	number of times to CR serve before downgrading lock

```
SYS@orcl2> SELECT empno, ename, sal, deptno
  FROM scott.emp
 WHERE empno = 7788 ; 2 3
```

EMPNO	ENAME	SAL	DEPTNO
7788	SCOTT	4000	20

```
SYS@orcl2> /
```

EMPNO	ENAME	SAL	DEPTNO
7788	SCOTT	4000	20

SYS@orcl2> @cache

INST_ID	FILE#	BLOCK#	STATUS	INST_ID	MODE_HELD
1	4	151	scur	1	1
1	4	151	scur	2	1
2	4	151	scur	1	2
2	4	151	scur	2	1

SYS@orcl2> @bh2

INST_ID	FILE#	BLOCK#	STATUS	D
2	4	151	scur	N
2	4	151	cr	N
2	4	151	cr	N
2	4	151	cr	N
2	4	151	cr	N

>> 최신 버전의 이미지는 1,2 번 instance에 존재하며 2번 instance에도 SCur모드의 버퍼 확인 가능하다. 앞선 작업에서 만들어진 CR모드의 블록은 아직 존재 한다.

- Write – Write

Terminal 1

SYS@orcl1> @bh1

INST_ID	FILE#	BLOCK#	STATUS	D
1	4	151	scur	N
1	4	151	cr	N

>> 1번 instance에도 최신 버전의 SCur모드로 버퍼를 가지고 있다.

** 만약 아직까지 Dirty 버퍼의 상태를 그대로 유지하고 있다면 아래의 명령을 통해 Dirty Buffer 의 상태를 N 으로 변경 한다. 또 다른 DML 수행 시 기존의 SCur모드의 이미지가 Past Image 로 변경 될 수 있기 때문이다.

SYS@orcl1> ALTER SYSTEM ARCHIVE LOG CURRENT ;

System altered.

SYS@orcl1> ALTER SYSTEM CHECKPOINT GLOBAL ;

System altered.

SYS@orcl1> UPDATE scott.emp
SET sal = 3000
WHERE empno = 7788 ; 2 3

1 row updated.

```
SYS@orcl1> @cache
```

INST_ID	FILE#	BLOCK#	STATUS	INST_ID	MODE_HELD
1	4	151	xcur	1	2
1	4	151	xcur	1	2

```
SYS@orcl1> @bh1
```

INST_ID	FILE#	BLOCK#	STATUS	D
1	4	151	xcur	Y
1	4	151	cr	N

>> DML작업으로 인해 XCur상태로 블록을 소유하고 있으며 1번 instance가 최신 버전을 보유 하고 있다.

Terminal 2

SYS@orcl2> @bh2

INST_ID	FILE#	BLOCK#	STATUS	D
2	4	151	cr	N
2	4	151	cr	N
2	4	151	cr	N
2	4	151	cr	N

>> 최신 상태는 1번 instance에 존재하므로 SCur모드의 버퍼는 CR모드로 변경 되었다.

SYS@orcl1> @cache

INST_ID	FILE#	BLOCK#	STATUS	INST_ID	MODE_HELD
1	4	151	pi	1	0
1	4	151	pi	2	1
2	4	151	xcur	1	1
2	4	151	xcur	2	2

SYS@orcl1> @bh2

INST_ID	FILE#	BLOCK#	STATUS	D
2	4	151	xcur	Y
2	4	151	cr	N
2	4	151	cr	N
2	4	151	cr	N
2	4	151	cr	N
2	4	151	cr	N

>> 2번 instance에도 동일 블록의 수정을 위해 XCur상태로 획득 하였고 1번 instance의 이미지는 Past Image로 변경 되었다. Cache Fusion 을 통해 1번 인스턴스의 블록을 전송 받고 과거 이미지는 폐기 할 수 있도록 Past Image 상태로 변경 한다.

Terminal 1

SYS@orcl1> @bh1

INST_ID	FILE#	BLOCK#	STATUS	D
1	4	151	pi	Y
1	4	151	cr	N

>> XCur모드의 버퍼가 Past Image 상태로 변경 된 것을 확인 할 수 있다.

SYS@orcl1> @cache

INST_ID	FILE#	BLOCK#	STATUS	INST_ID	MODE_HELD
1	4	151	pi	1	0
1	4	151	pi	2	1
2	4	151	xcur	1	1
2	4	151	xcur	2	2

SYS@orcl1> @bh1

INST_ID	FILE#	BLOCK#	STATUS	D
1	4	151	pi	Y
1	4	151	cr	N

>> 1번 instance에서 COMMIT 을 진행 하였으나 Dirty 상태는 아직 그대로다. (이 부분은 실습 환경에 따라 다를 수 있다.) 만약 아직 Dirty 상태를 그대로 가지고 있다면 Dirty Buffer 상태의 Past Image 를 Writing 할 수 있는 Checkpoint 명령이 수행 되면 어떻게 될까?

SYS@orcl1> ALTER SYSTEM CHECKPOINT ;

System altered.

SYS@orcl1> @bh1

INST_ID	FILE#	BLOCK#	STATUS	D
1	4	151	cr	N
1	4	151	cr	N

SYS@orcl1> @bh2

INST_ID	FILE#	BLOCK#	STATUS	D
2	4	151	xcur	N
2	4	151	cr	N
2	4	151	cr	N
2	4	151	cr	N
2	4	151	cr	N
2	4	151	cr	N

>> 1번 instance에서 Checkpoint 명령을 진행 했으나 Past Image 의 최신 버전이 대신 Writing 된 것을 확인