

# Oracle Database Architecture

경북산업직업전문학교 [1-SEP-2022]  
- 무단배포를 금지합니다 -

# Agenda

---

- 1 History of Oracle Database
- 2 Oracle Database Server Architecture
- 3 Oracle Database – Storage
- 4 Oracle Instance, RAC, Multitenant, MAA
- 5 SQL
- 6 Transaction, Instance Recovery

# History of Oracle Database

1977: Software Development Laboratories



```
$ sqlplus scott/tiger
```

```
SQL*Plus: Release 11.1.0.6.0 - Production on Sun Oct 21 13:08:36 2007  
Copyright (c) 1982, 2007, Oracle. All rights reserved.
```

```
Connected to:  
Oracle Database 11g Enterprise Edition Release 11.1.0.6.0 - Production  
With the Partitioning, OLAP, Data Mining and Real Application Testing options
```

```
SQL>
```

# History of Oracle Database

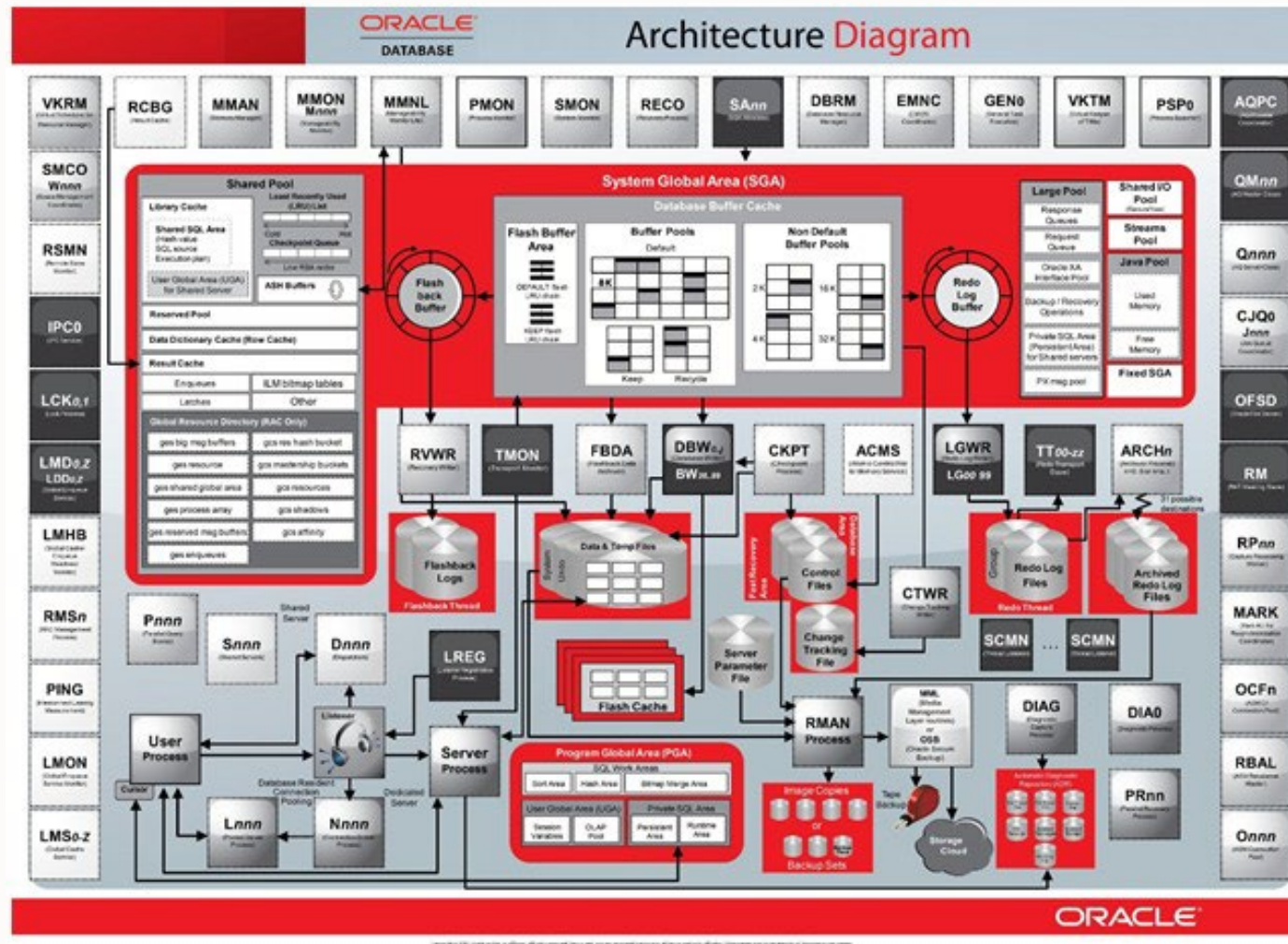
Version / Release / Year			Marquee Features
v2	2.3	1979	<b>First commercially available SQL-based RDBMS</b> implementing some basic SQL queries and simple joins
v3	3.1.3	1983	Concurrency control, data distribution, and scalability
v4	4.1.4.0	1984	<b>Multiversion read consistency</b> . First version available for MS-DOS.
v5	5.0.22	1985	Support for client/server computing and distributed database systems. First version available for OS/2.
v6	6.0.17	1988	<b>Row-level locking</b> , scalability, <b>online backup and recovery</b> , PL/SQL. First version available for Novell Netware 386.
6.2	6.2.0		<b>Oracle Parallel Server</b>
		1989	<b>한국오라클 설립</b>
7	7.0.12	1992	PL/SQL stored procedures, Triggers, Distributed 2-phase commit, Shared Cursors, <b>Cost Based Optimizer</b>
7.1	7.1.0	1994	<b>Parallel SQL Execution</b> . First version available for Windows NT.
7.2	7.2.0	1995	Shared Server, XA Transactions, <b>Transparent Application Failover</b>
7.3	7.3.0	1996	<b>Object-relational database</b>
8	8.0.3	1997	Recovery Manager, <b>Partitioning</b> . First version available for Linux.
8i	8.1.5.0	1998	Native <b>internet</b> protocols and Java, Virtual Private Database
9i	9.0.1.0	2001	<b>Oracle Real Application Clusters (RAC)</b> , Oracle XML DB
9iR2	9.2.0.1	2002	Advanced Queuing, <b>Data Mining</b> , Streams, Logical Standby
10gR1	10.1.0.2	2003	Automated Database Management, Automatic Database Diagnostic Monitor, <b>Grid infrastructure</b> , <b>Oracle ASM</b> , Flashback Database
10gR2	10.2.0.1	2005	<b>Real Application Testing</b> , Database Vault, Online Indexing, <b>Advanced Compression</b> , Data Guard Fast-Start Failover, <b>Transparent Data Encryption</b>
11gR1	11.1.0.6	2007	<b>Active Data Guard</b> , Secure Files, <b>Exadata</b>
11gR2	11.2.0.1	2009	Edition Based Redefinition, Data Redaction, <b>Hybrid Columnar Compression</b> , Cluster File System, Golden Gate Replication, Database Appliance
12cR1	12.1.0.1	2013	<b>Multitenant</b> architecture, In-Memory Column Store, Native JSON, SQL Pattern Matching, <b>Database Cloud Service</b>
12cR2	12.2.0.1	2016	Native Sharding, Zero Data Loss Recovery Appliance, <b>Exadata Cloud Service</b> , <b>Cloud at Customer</b>
18c	18.1.0 // 12.2.0.2	2018	Polymorphic Table Functions, Active Directory Integration
19c	19.1.0 // 12.2.0.3	2019	Active Data Guard DML Redirection, Automatic Index Creation, Real-Time Statistics Maintenance, SQL Queries on Object Stores, In-Memory for IoT Data Streams, and many more.

# Agenda

---

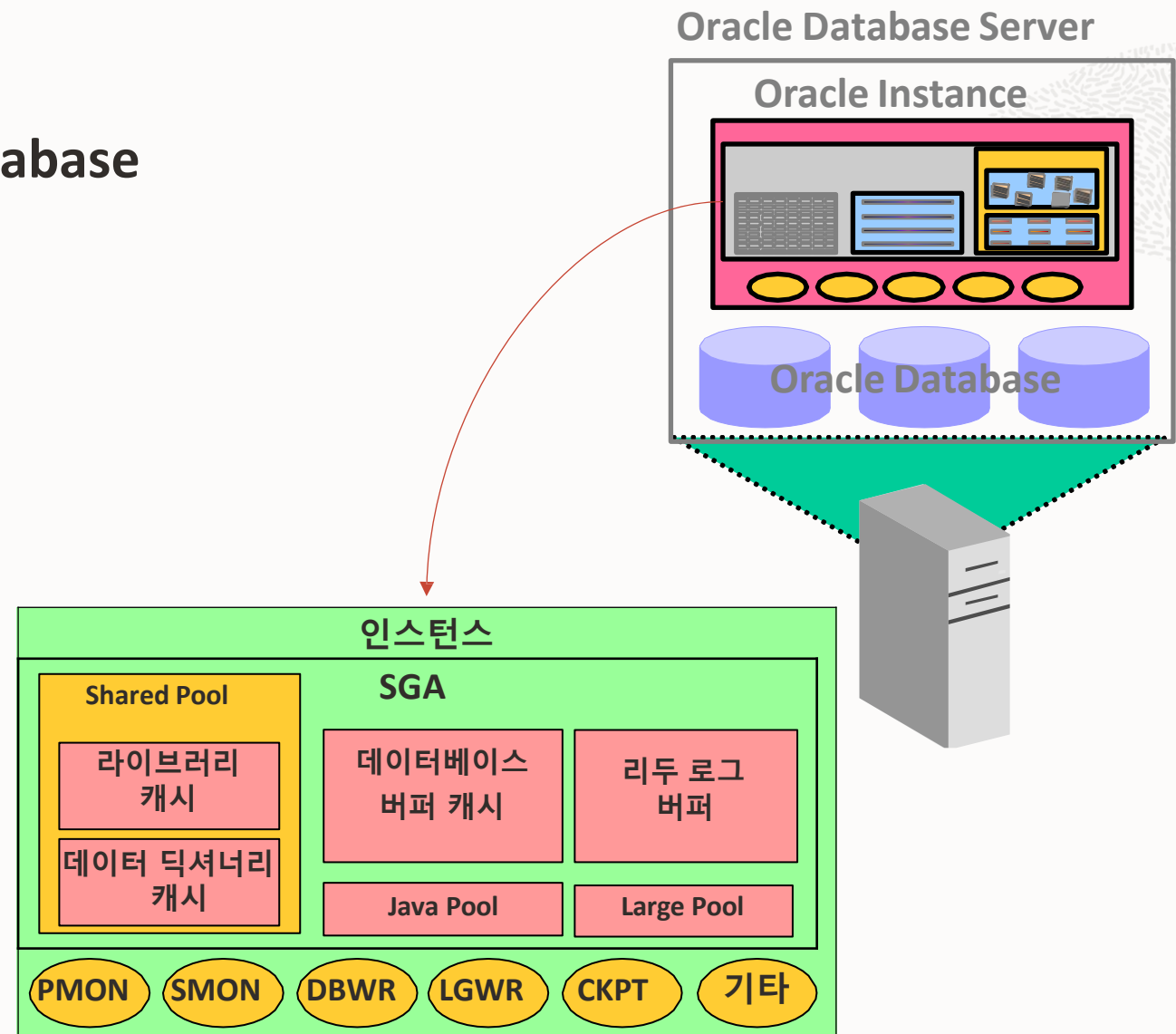
- 1 History of Oracle Database
- 2 Oracle Database Server Architecture
- 3 Oracle Database – Storage
- 4 Oracle Instance, RAC, Multitenant, MAA
- 5 SQL
- 6 Transaction, Instance Recovery
- 7 Summary

# Oracle Database Server Architecture



# Oracle Database Server / Instance / Database

- Oracle Database Server?
  - 정보를 개방적이고 통합적으로 관리할 수 있는 “오라클의 데이터베이스 관리 시스템” (DBMS)
  - “인스턴스”와 “데이터베이스”로 구성
- Oracle Instance?
  - 데이터베이스를 액세스하는 수단
  - SGA(System Global Area) 메모리와 백그라운드 프로세스 구조로 구성
- Oracle Database?
  - 하나의 단위로 취급되는 데이터 모음
  - Data file, redolog file, control file로 구성  
(참고) 오라클의 DBMS 제품명이기도 함



\* 제품명 변경: Oracle 7 Server → Oracle Database 10g ~ Oracle Database 19c

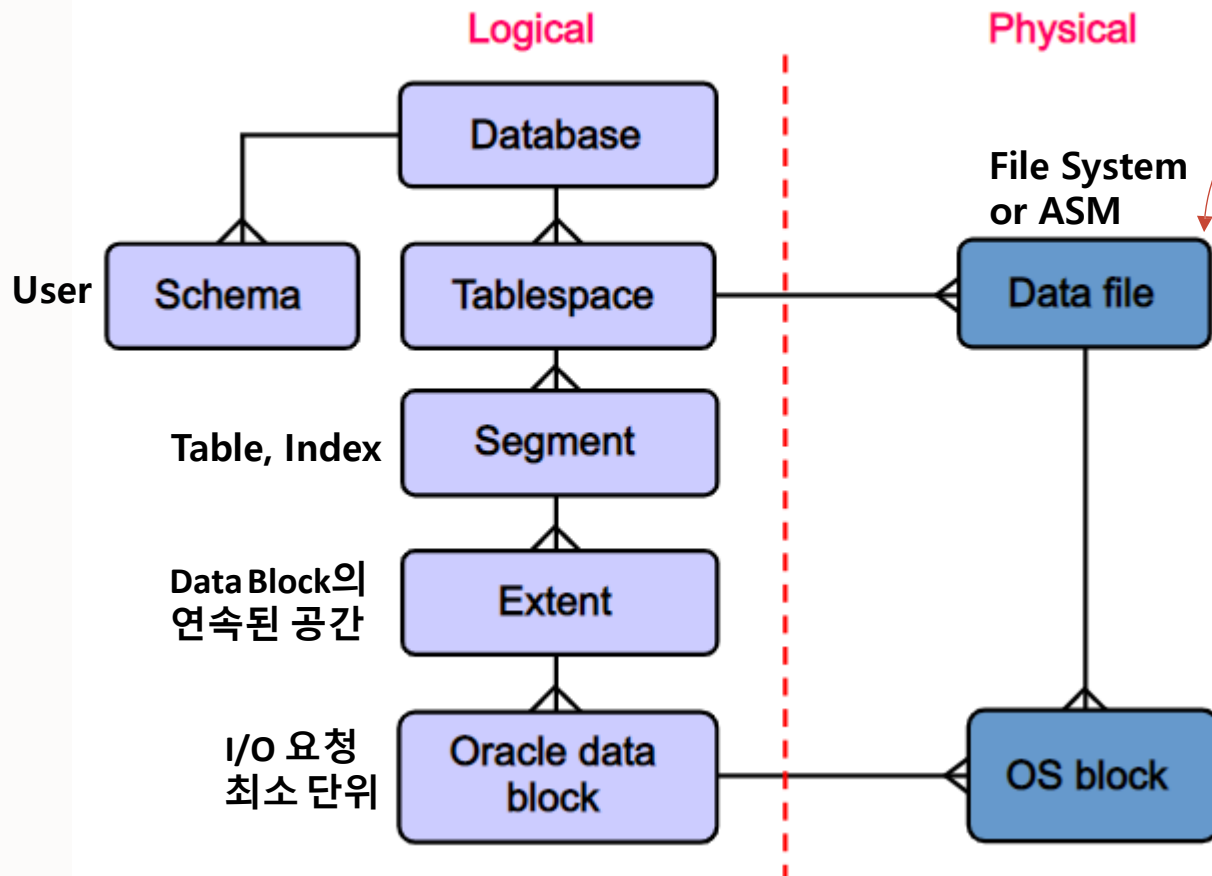
# Agenda

---

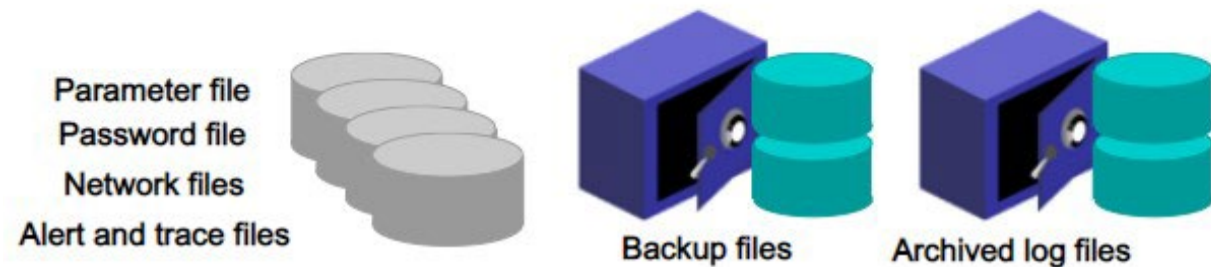
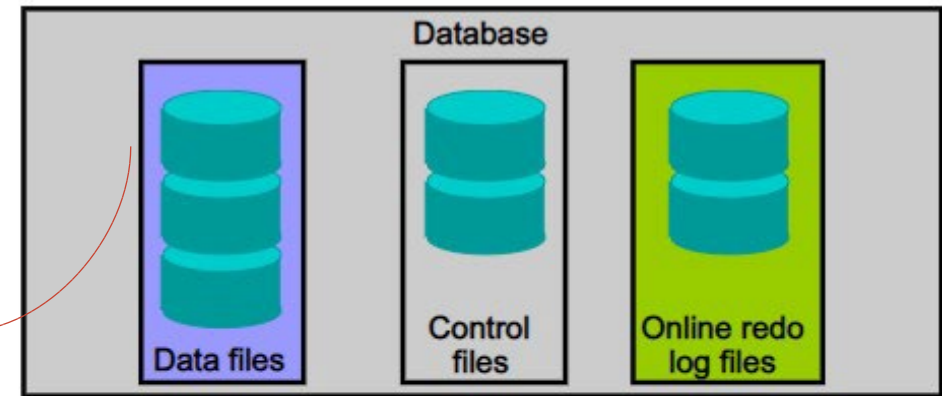
- 1 History of Oracle Database
- 2 Oracle Database Server Architecture
- 3 Oracle Database – Storage
- 4 Oracle Instance, RAC, Multitenant, MAA
- 5 SQL
- 6 Transaction, Instance Recovery
- 7 Summary



# Oracle Database 데이터 저장 구조



## Database Storage Architecture

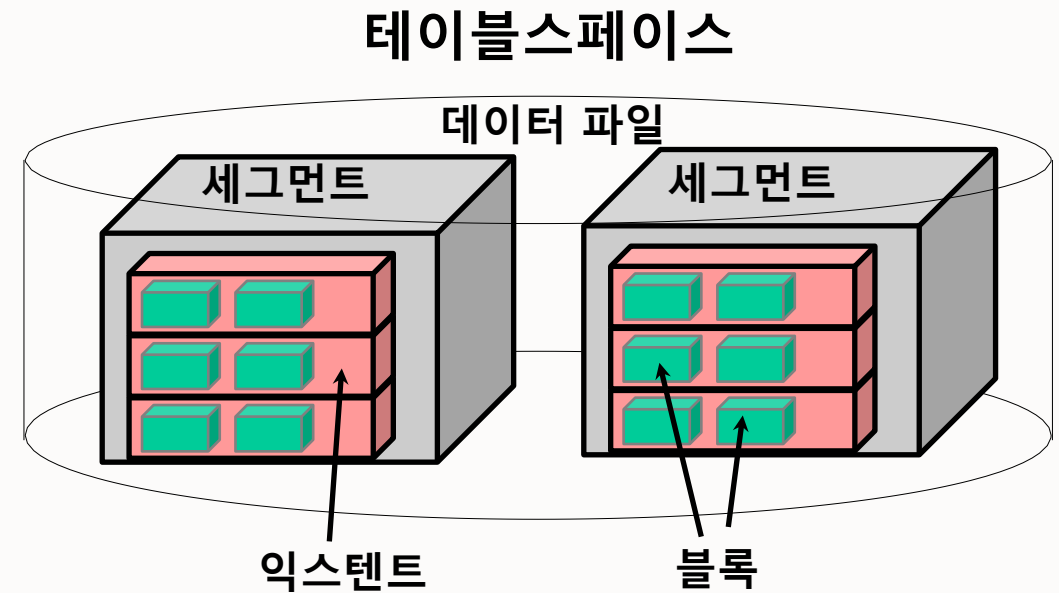
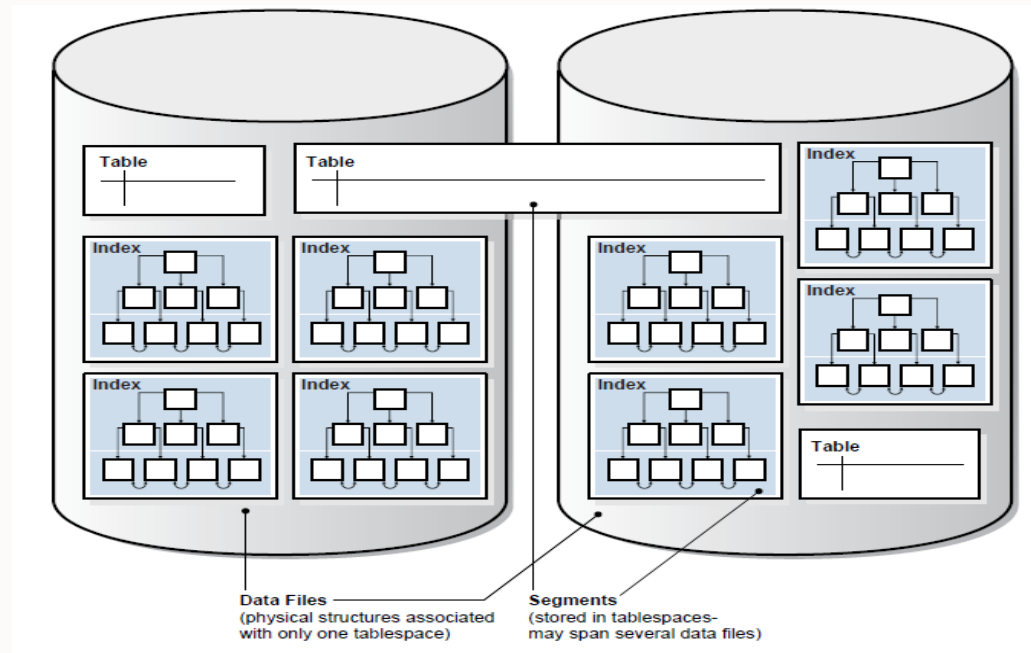


데이터베이스 공간 부족 시 :

- Logical : 추가 공간이 요구되는 세그먼트가 들어있는 테이블스페이스의 공간 확보가 필요
- Physical : 테이블스페이스를 구성하는 데이터 파일의 크기 resize 혹은 추가적인 데이터 파일 추가 → 추가할 공간이 없을 시는 새로운 disk 할당이 필요

## Tablespace, data file, Segment

- Oracle은 데이터를 논리적으로는 테이블스페이스에 저장하고 물리적으로는 데이터 파일에 저장



# Partitioning

- Table, Index 등을 작은 단위로 나눈 단위
  - Benefits : Performance, Availability, Manageability, Cost, etc.
  - Strategies : Range, List, Hash, Interval, Composite



```
CREATE TABLE time_range_sales
( prod_id      NUMBER(6)
, cust_id      NUMBER
, time_id      DATE
, channel_id   CHAR(1)
, promo_id     NUMBER(6)
, quantity_sold NUMBER(3)
, amount_sold  NUMBER(10,2)
)
PARTITION BY RANGE (time_id)
(PARTITION SALES_1998 VALUES LESS THAN (TO_DATE('01-JAN-1999', 'DD-MON-YYYY')),
PARTITION SALES_1999 VALUES LESS THAN (TO_DATE('01-JAN-2000', 'DD-MON-YYYY')),
PARTITION SALES_2000 VALUES LESS THAN (TO_DATE('01-JAN-2001', 'DD-MON-YYYY')),
PARTITION SALES_2001 VALUES LESS THAN (MAXVALUE))
;
```

Table Partition SALES_1998						
PROD_ID	CUST_ID	TIME_ID	CHANNEL_ID	PROMO_ID	QUANTITY_SOLD	AMOUNT_SOLD
40	100530	30-NOV-98	9	33	1	44.99
125	9417	04-FEB-98	3	999	1	16.86
45	9491	28-AUG-98	4	350	1	47.45

Table Partition SALES_1999						
PROD_ID	CUST_ID	TIME_ID	CHANNEL_ID	PROMO_ID	QUANTITY_SOLD	AMOUNT_SOLD
116	11393	05-JUN-99	2	999	1	12.18
36	4523	27-JAN-99	3	999	1	53.89
24	11899	26-JUN-99	4	999	1	43.04

Table Partition SALES_2000						
PROD_ID	CUST_ID	TIME_ID	CHANNEL_ID	PROMO_ID	QUANTITY_SOLD	AMOUNT_SOLD
133	9450	01-DEC-00	2	999	1	31.28
35	2606	17-FEB-00	3	999	1	54.94

Table Partition SALES_2001						
PROD_ID	CUST_ID	TIME_ID	CHANNEL_ID	PROMO_ID	QUANTITY_SOLD	AMOUNT_SOLD
116	133	06-JUN-01	2	999	1	17.12
30	170	23-FEB-01	2	999	1	8.8

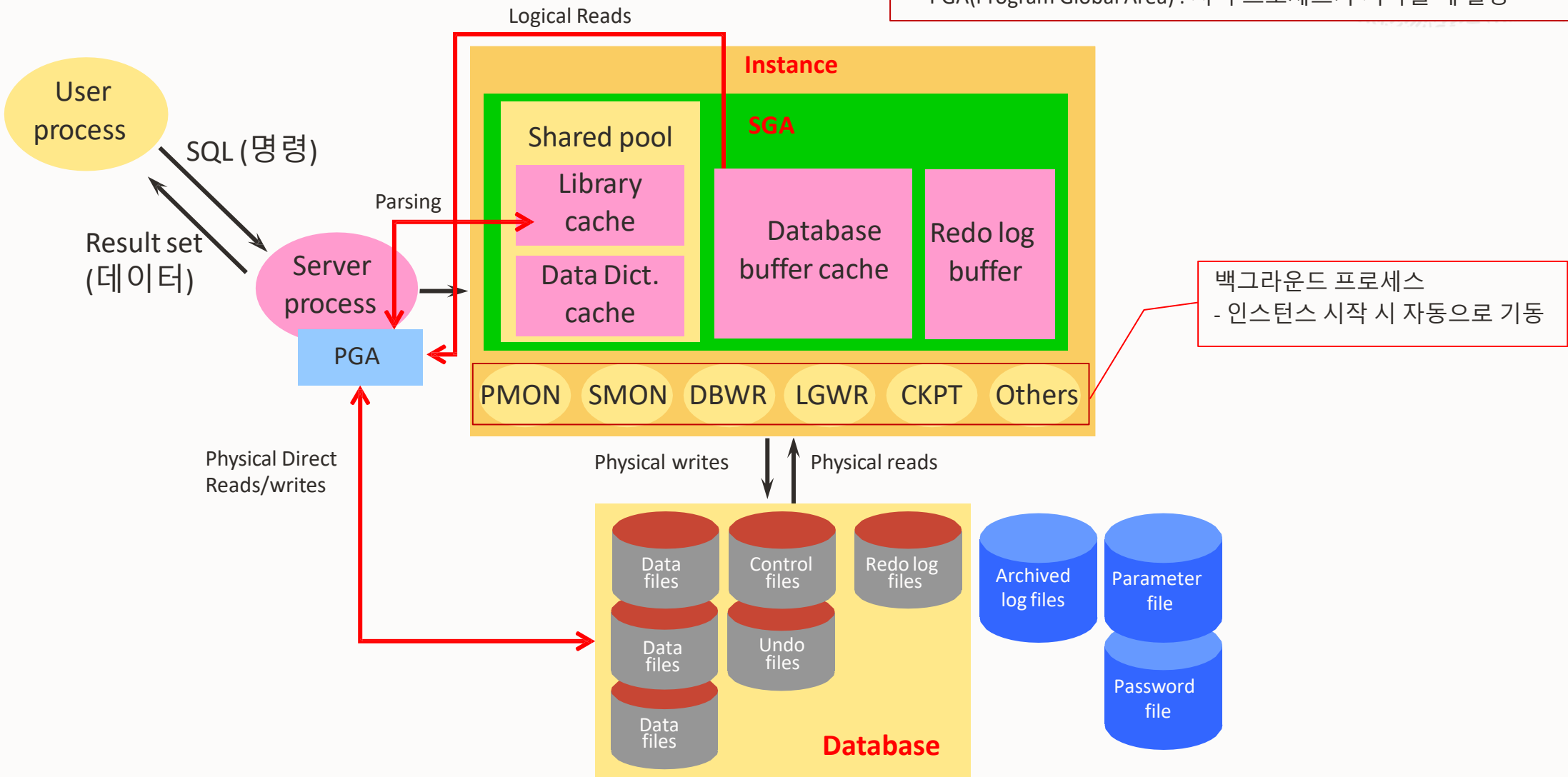
# Agenda

---

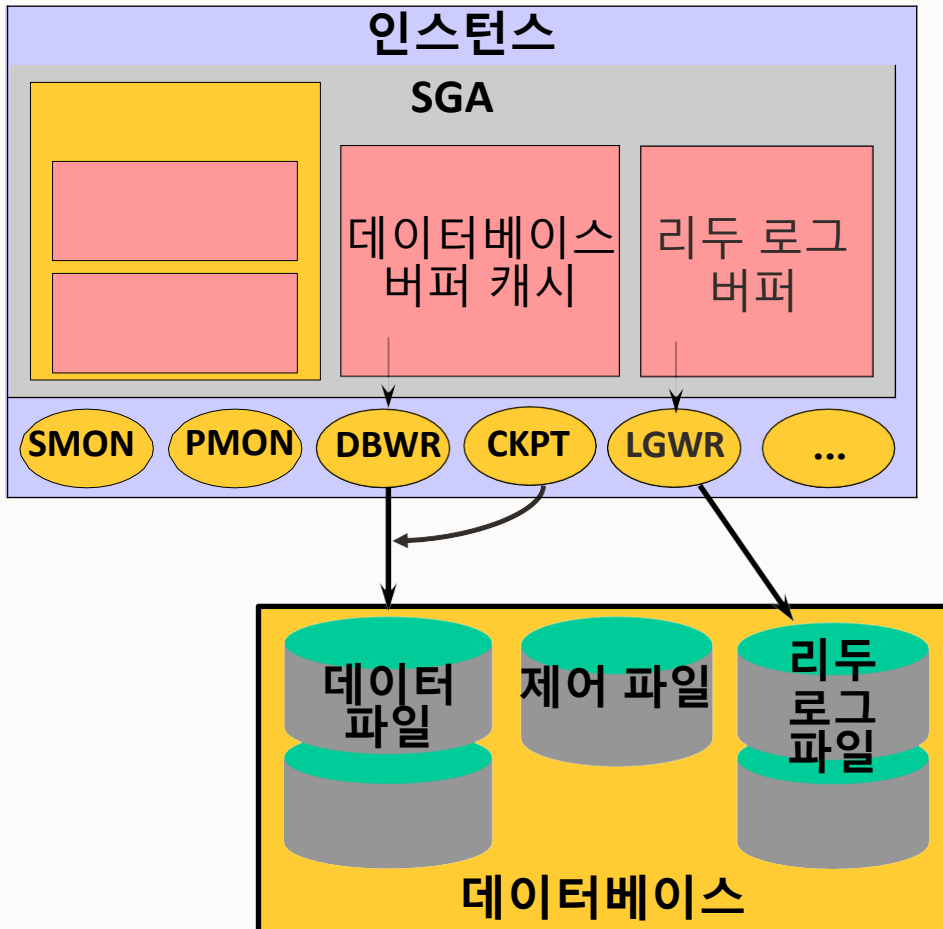
- 1 History of Oracle Database
- 2 Oracle Database Server Architecture
- 3 Oracle Database – Storage
- 4 Oracle Instance, RAC, Multitenant, MAA
- 5 SQL
- 6 Transaction, Instance Recovery
- 7 Summary

# Oracle Database Server Architecture

- Oracle의 메모리 구조는 다음 두 가지 메모리 영역으로 구성
- SGA(System Global Area) : 인스턴스가 시작될 때 할당되며 Oracle 인스턴스의 기본적인 구성 요소
  - PGA(Program Global Area) : 서버 프로세스가 시작될 때 할당



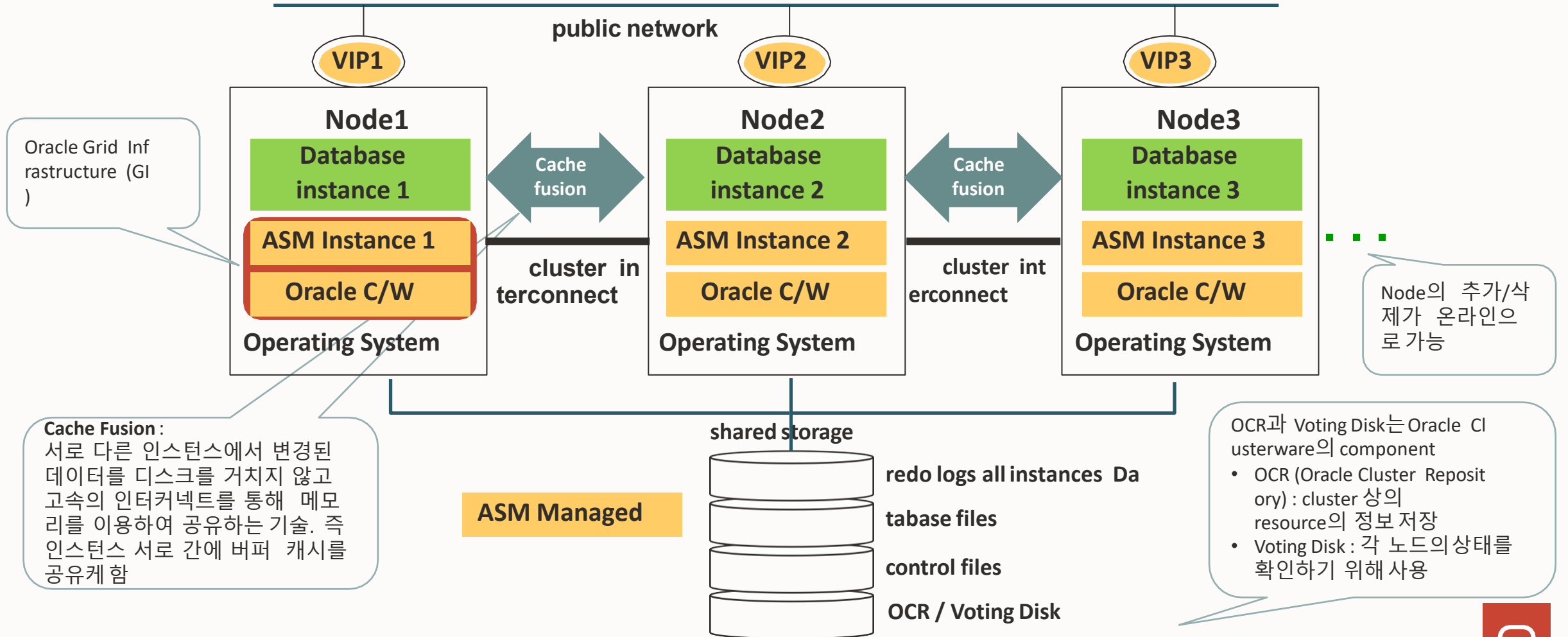
## 주요 Background Process



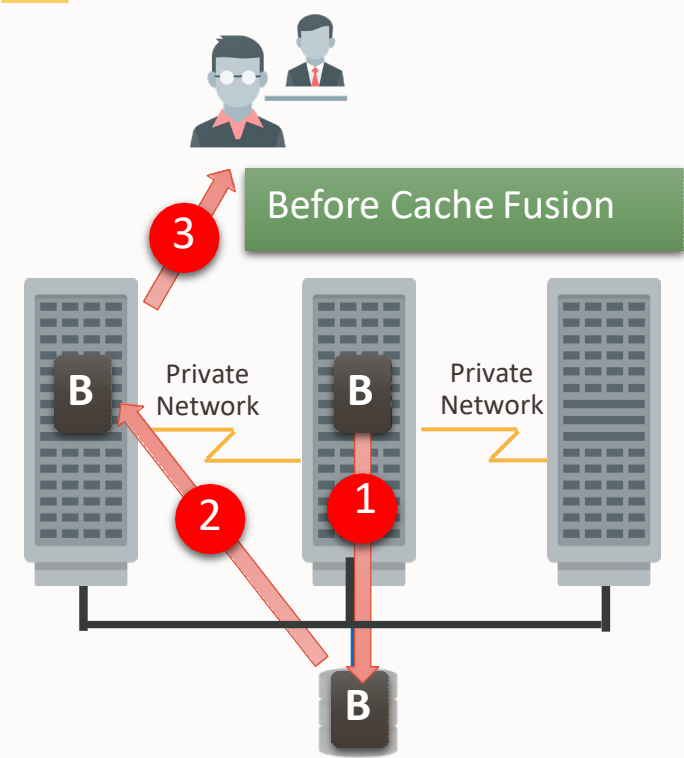
- **SMOM** : 시스템 모니터는 오라클 인스턴스를 관리하는 프로세스. 인스턴스 fail 시 인스턴스를 복구하는 역할
- **PMON** : 오라클 서버에서 사용되는 각 프로세스들을 감시하는 프로세스. 비정상 종료된 데이터베이스의 접속을 정리
- **LGWR** : 데이터베이스에 발생한 모든 변경을 기록하는 역할 (Write Ahead Logging). 트랜잭션이 완료되었을 때 LGWR가 리두로그 버퍼의 내용을 온라인 리두로그 파일에 기록
- **DBWR** : 버퍼 캐시에 있는 수정된(Dirty) 버퍼의 내용을 데이터 파일에 기록 (Deferred 방식)
- **CKPT** : 변화된 데이터 블록의 개수 또는 일정 시간 간격으로 DBWR 프로세스가 데이터베이스 버퍼(Dirty Buffer)를 데이터 파일에 저장하도록 명령

# Oracle RAC Architecture

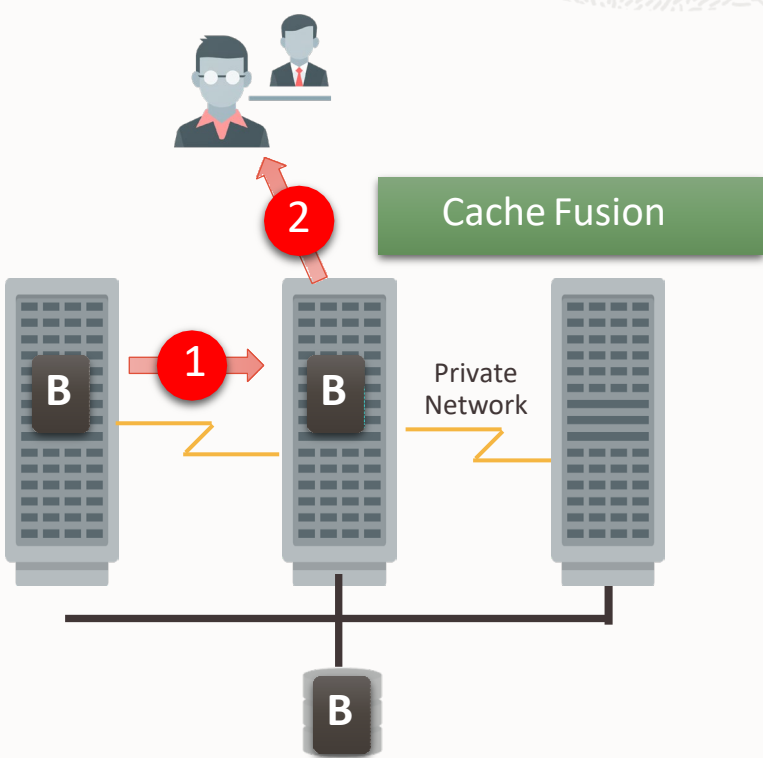
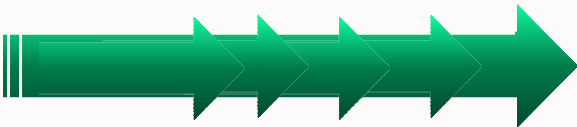
- RAC Architecture : 가용성과 확장성을 충족시키는 멀티 인스턴스 아키텍처
  - 가용성 : 한 노드 failure 시, 다른 노드를 통한 서비스 연속성 확보
  - 확장성 : 노드를 추가할수록 처리능력은 더 커짐
- Oracle 9i 부터 Vendor Cluster S/W 대신 Oracle Clusterware 제공
- Oracle 10g 부터 Volume Manager 역할의 ASM 제공



# Cache Fusion: A long Journey



Oracle Parallel Server  
(OPS)



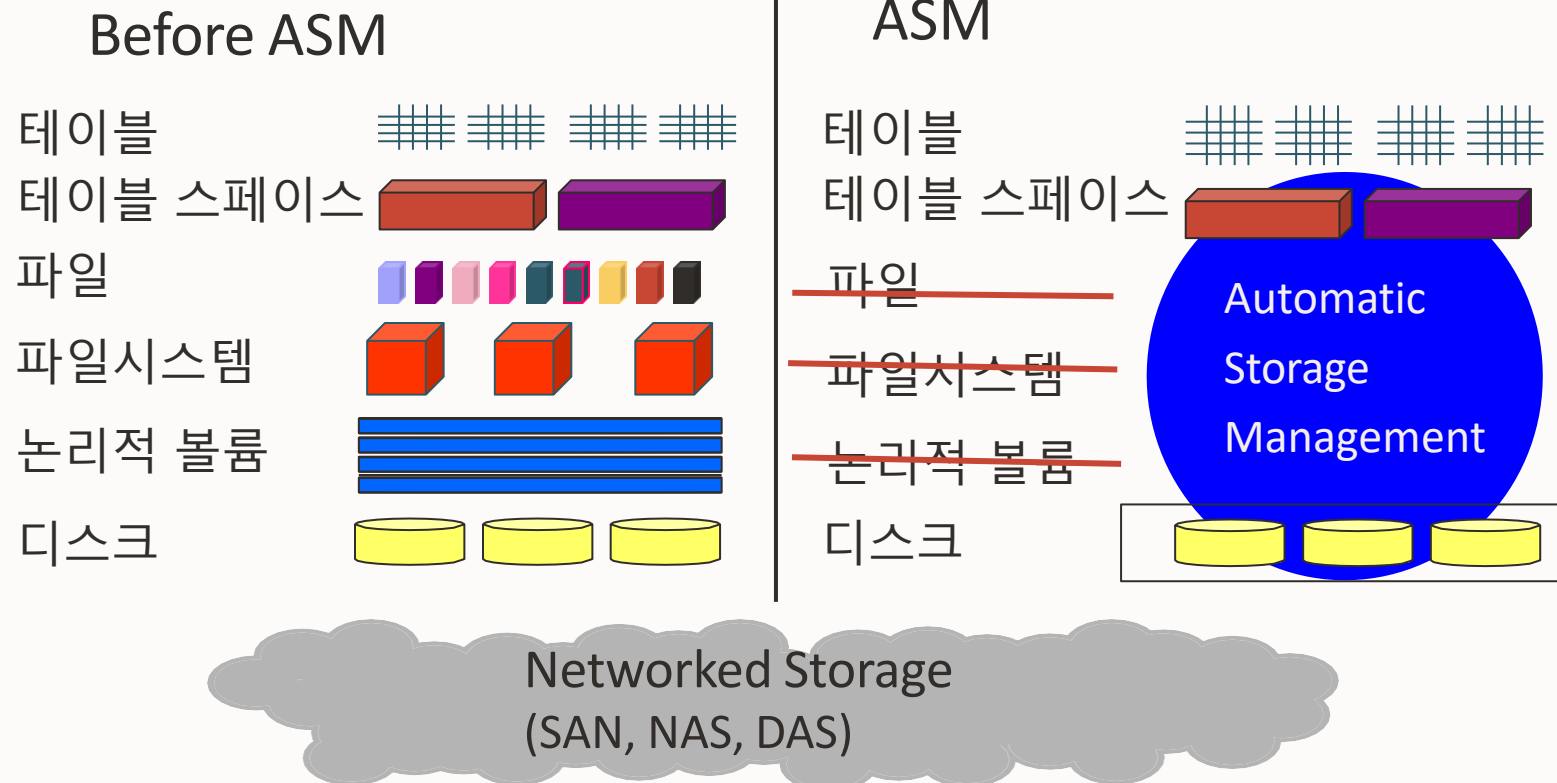
Real Application Clusters  
(RAC)



# Automatic Storage Management (ASM)

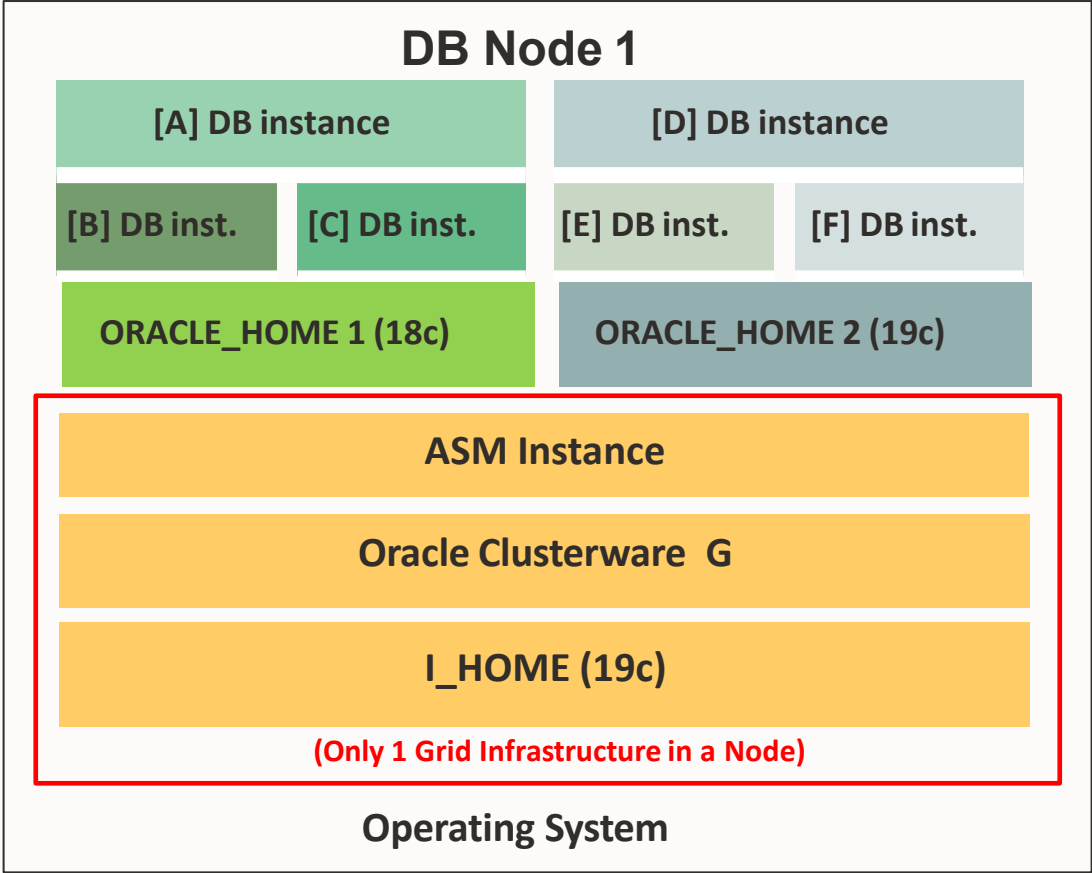
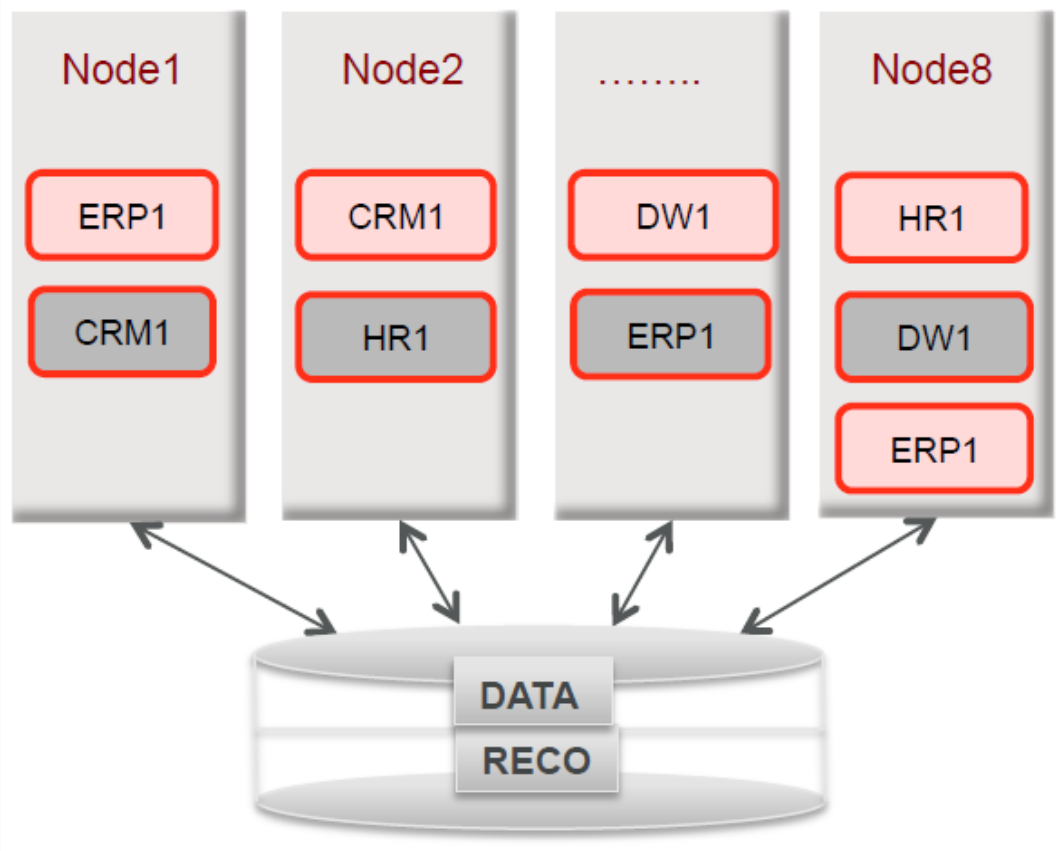
ASM 은 오라클 데이터베이스 파일을 위한 볼륨 매니저

- Before ASM은 일반적으로 많이 사용하는 Veritas Volume Manager 환경이라고 이해
- ASM에서는 logical volume & 파일시스템 layer가 필요 없음

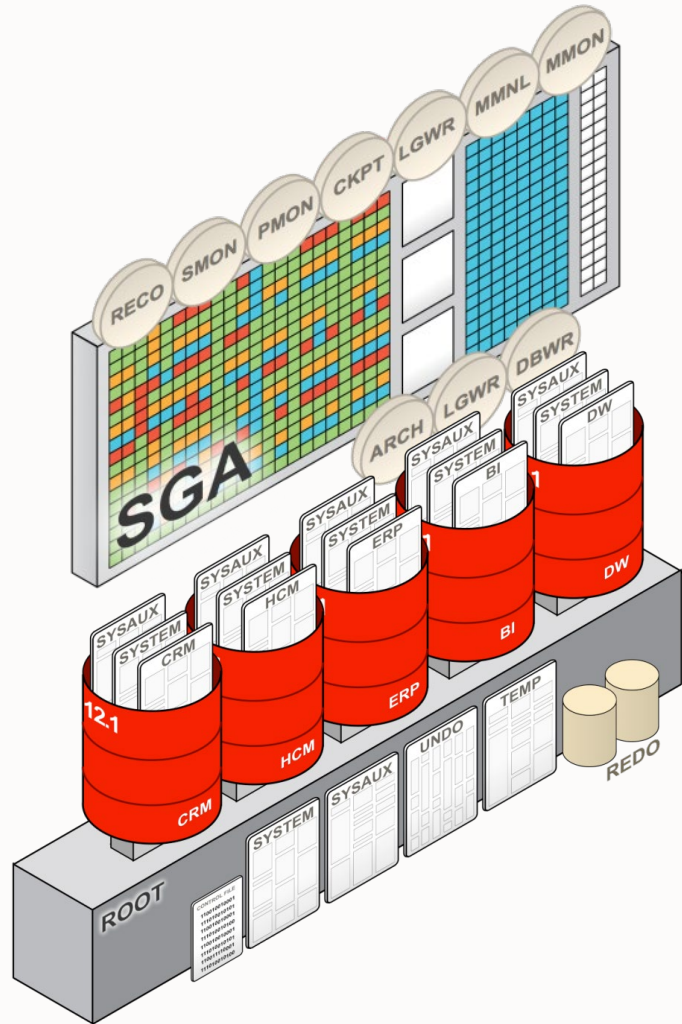


# Consolidation Architecture

- 한 노드에 여러 개의 ORACLE\_HOME(DBMS S/W)을 가질 수 있는 반면, GI\_HOME은 한 개만 가능
- Multi Version의 Multi HOME인 경우 상위 Version의 Grid S/W 설치 필요

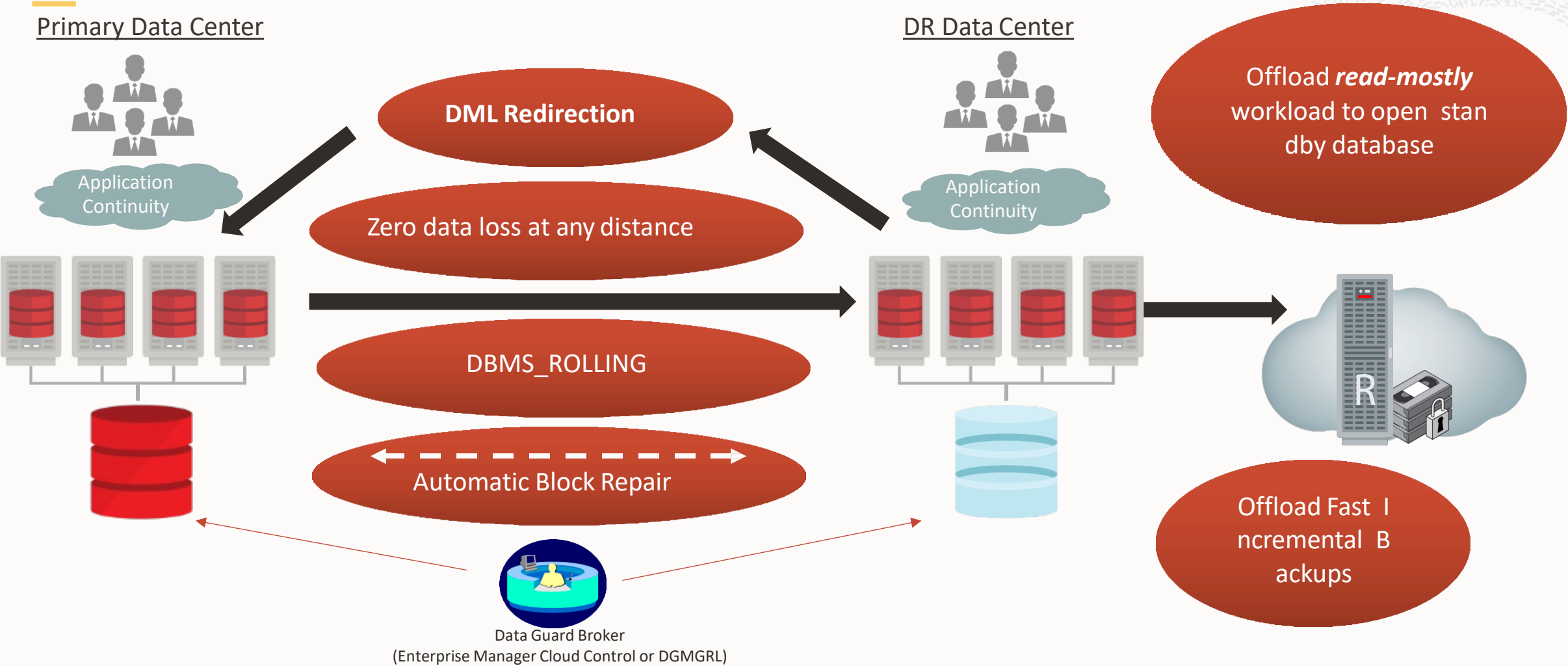


# Multitenant Architecture



- PDB들은 SGA, Background Processes, Undo, Redo, Control Files, spfile 등을 공유
- PDB는 각자의 SYSTEM, SYSAUX, 사용자 Tablespace가 있고 Temp를 선택적으로 소유할 수 있음
- 각각의 데이터베이스에(PDBs)에 연결된 고객 세션 (Foreground sessions)들은 연결된 데이터베이스의 데이터만 접근 가능

# Active Data Guard (ADG) & Maximum Availability Architecture (MAA)



# Agenda

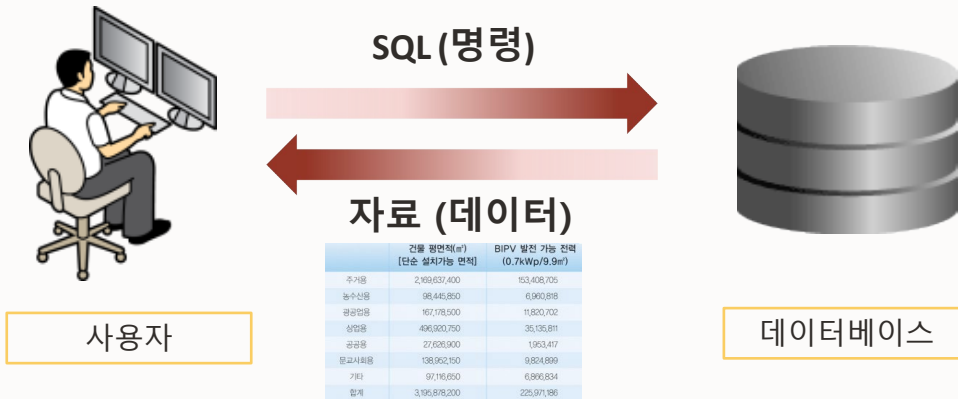
---

- 1 History of Oracle Database
- 2 Oracle Database Server Architecture
- 3 Oracle Database – Storage
- 4 Oracle Instance, RAC, Multitenant, MAA
- 5 **SQL**
- 6 Transaction, Instance Recovery
- 7 Summary

# SQL (Structured Query Language)

## 데이터베이스의 언어 SQL은 ?

- 관계형 데이터베이스에 접근(생성, 변경, 삭제, 조회)하기 위한 비절차적 표준언어



### ◆ SQL 언어의 특징

- ✓ 이해하기 쉬운 형태로 표현
- ✓ 대화식 질의어로 사용가능
- ✓ 데이터 정의, 데이터 조작, 제어기능 제공
- ✓ 레코드 집합단위로 처리
- ✓ 비절차적 언어

# SQL문의 유형

SQL 튜닝 대상

명령어의 종류	명령어	설명
QUERY (데이터 검색, 조회)	SELECT	데이터를 조회하거나 검색하기 위한 명령어
DML (데이터 조작어)	INSERT UPDATE DELETE	테이블의 데이터를 조작(새로운 데이터 입력, 수정, 삭제)하는 명령어 : Data Manipulation Language
DDL (데이터 정의어)	CREATE, DROP ALTER, RENAME TRUNCATE, COMMENT	테이블과 같은 데이터의 구조를 정의(생성, 변경, 제거 등)하는데 사용되는 명령어 : Data Definition Language
TCL (트랜잭션 제어어)	COMMIT, ROLLBACK SAVEPOINT	논리적인 작업의 단위를 묶어서 DML에 의해 조작된 결과를 작업 단위별로 제어하는 명령어 : Transaction Control Language
DCL (데이터 제어어)	GRANT REVOKE	Object들을 사용하도록 권한을 주고 받는 명령어 : Data Control Language

# SQL Processing & Optimizer

Figure 7-3 Stages of **SQL Processing**

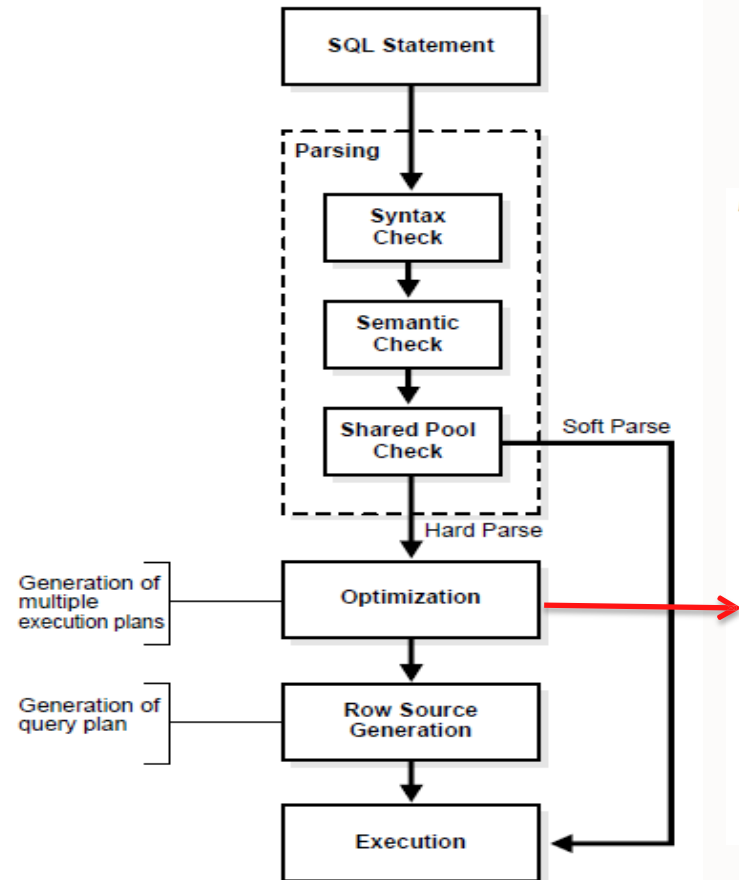
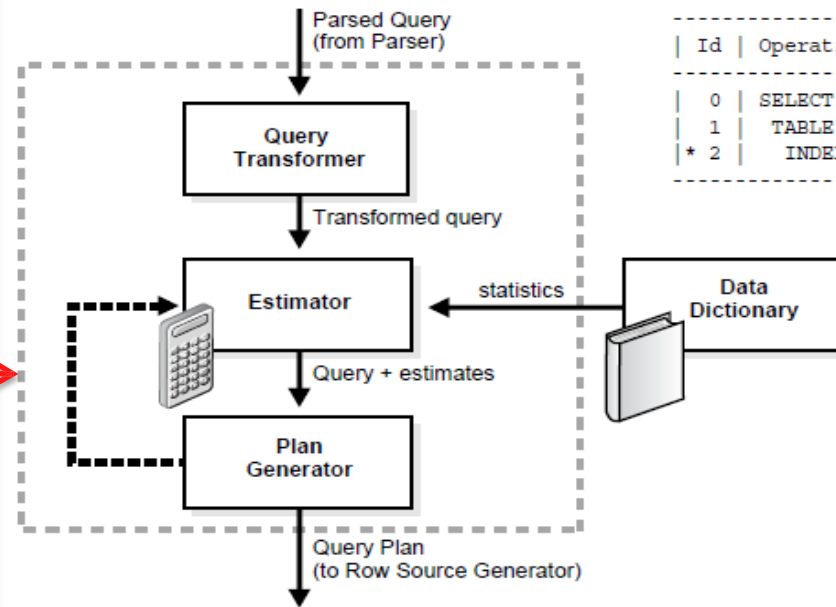


Figure 7-2 **Optimizer** Components



Example 7-4 **Execution Plan** for SELECT with FIRST\_ROWS Hint

```
SELECT /*+ FIRST_ROWS(25) */ employee_id, department_id
FROM   hr.employees
WHERE  department_id > 50;
```

Id	Operation	Name	Rows	Bytes
0	SELECT STATEMENT		26	182
1	TABLE ACCESS BY INDEX ROWID	EMPLOYEES	26	182
* 2	INDEX RANGE SCAN	EMP_DEPARTMENT_IX		



# Optimizer

사용자 요청 SQL을 가장 효율적이고 빠르게 수행할 수 있는 최적(최저비용)의 처리경로를 선택해주는 DBMS의 핵심기능

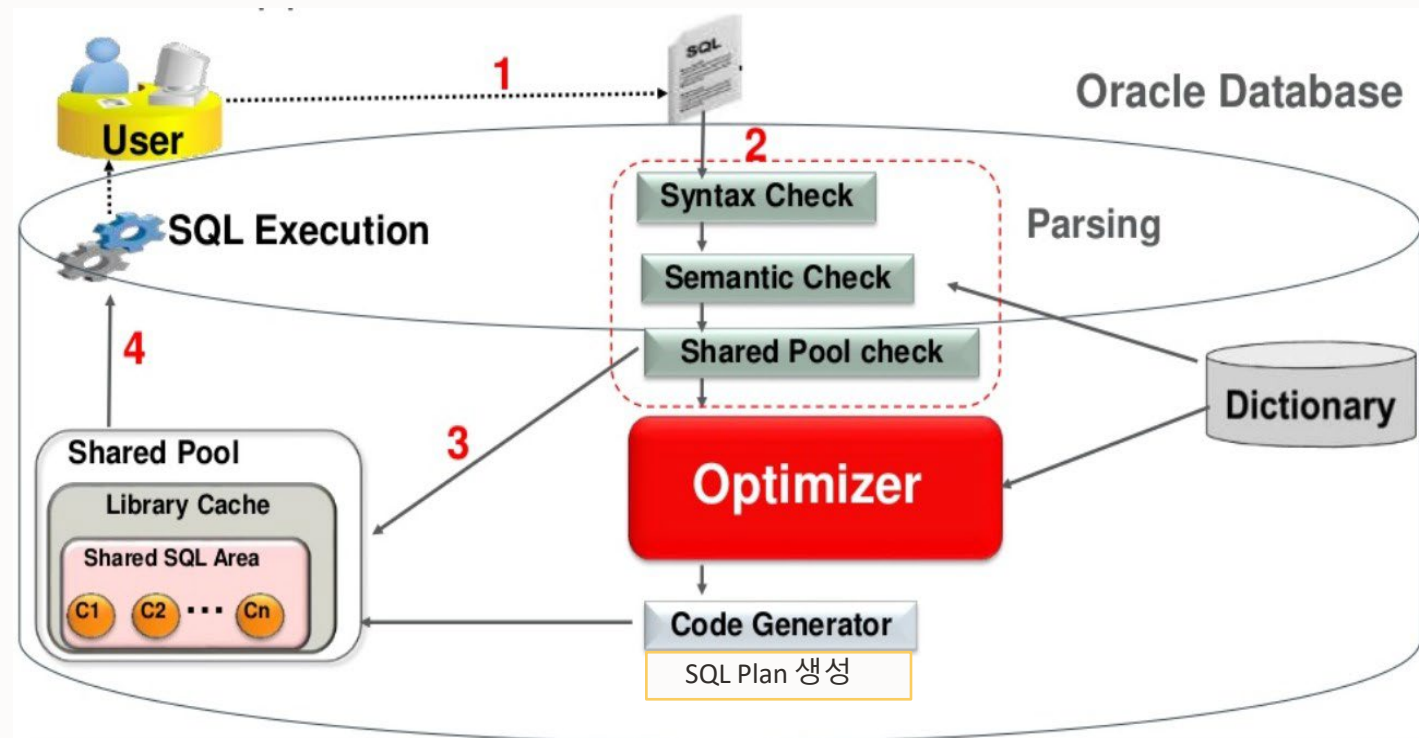
- 통계정보를 기반으로 실질적인 비용을 계산하여 수행 (Cost-Based Optimizer = CBO)

- 실행계획(Execution Plan)에 영향을 미치는 요소

- ✓ 통계 및 Parameter 정보
- ✓ 인덱스, 테이블 구조
- ✓ SQL의 형태 및 사용 컬럼, 연산자 형태
- ✓ 데이터베이스 환경

- 실제 실행계획을 확인 & 검토 필요성 존재

- ✓ 통계정보 누락 및 수집된 통계정보가 부정확
- ✓ 버전업 될수록 알고리즘이 향상되지만 검토 필요
- ✓ **제한된 시간내에서의 수행**





## SQL Processing & Optimizer 예시

```
> ALTER SESSION SET STATISTICS_LEVEL=ALL;  
  
> SELECT * FROM HR.EMPLOYEES WHERE EMPLOYEE_ID LIKE '2%';  
  
> SELECT * FROM TABLE(DBMS_XPLAN.DISPLAY_CURSOR(null,null,'ALLSTATS LAST'));
```

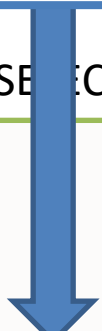
user has no select privilege on v\$sql\_plan 또는 user has no select privilege on v\$session 에러가 나올 때

SYS로 접속 후

```
> GRANT SELECT ON v_$session TO 유저명;  
> GRANT SELECT ON v_$sql_plan_statistics_all TO 유저명;  
> GRANT SELECT ON v_$sql_plan TO 유저명;  
> GRANT SELECT ON v_$sql TO 유저명;
```

## SQL Processing & Optimizer 예시

```
> ALTER SESSION SET STATISTICS_LEVEL=ALL;  
  
> SELECT * FROM HR.EMPLOYEES WHERE EMPLOYEE_ID LIKE '2%';  
  
> SELECT * FROM TABLE(DBMS_XPLAN.DISPLAY_CURSOR(null,null,'ALLSTATS LAST'));
```



	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
1	201	Michael	Hartstein	MHARTSTE	515.123.5555	04/02/17	MK_MAN	13000	(null)	100	20
2	202	Pat	Fay	PFAY	603.123.6666	05/08/17	MK_REP	6000	(null)	201	20
3	203	Susan	Mavris	SMAVRIS	515.123.7777	02/06/07	HR_REP	6500	(null)	101	40
4	204	Hermann	Baer	HBAER	515.123.8888	02/06/07	PR_REP	10000	(null)	101	70
5	205	Shelley	Higgins	SHIGGINS	515.123.8080	02/06/07	AC_MGR	12008	(null)	101	110
6	206	William	Gietz	WGIEZT	515.123.8181	02/06/07	AC_ACCOUNT	8300	(null)	205	110

# SQL Processing & Optimizer 예시

```
> SELECT * FROM TABLE(DBMS_XPLAN.DISPLAY_CURSOR(null,null,'ALLSTATS LAST'));
```

```

PLAN_TABLE_OUTPUT
SQL_ID d0k4tu2ctp3ka, child number 0
-----
SELECT * FROM HR.employees WHERE EMPLOYEE_ID LIKE '2%'
Plan hash value: 1445457117
-----
| Id | Operation | Name | Starts | E-Rows | A-Rows | A-Time | Buffers |
-----|-----|-----|-----|-----|-----|-----|-----|
| 0 | SELECT STATEMENT | | 1 | | 7 | 100:00:00.01 | 6 |
|* 1 | TABLE ACCESS FULL | EMPLOYEES | 1 | 5 | 7 | 100:00:00.01 | 6 |
-----
Predicate Information (identified by operation id):
-----
1 - filter(TO_CHAR("EMPLOYEE_ID") LIKE '2%')

```

	INDEX_OWNER	INDEX_NAME	UNIQUENESS	STATUS	INDEX_TYPE	TEMPORARY	PARTITIONED	FUNCIDX_STATUS	JOIN_INDEX	COLUMNS
1	HR	EMP_JOB_IX	NONUNIQUE	VALID	NORMAL	N	NO	(null)	NO	JOB_ID
2	HR	EMP_NAME_IX	NONUNIQUE	VALID	NORMAL	N	NO	(null)	NO	LAST_NAME, FIRST_NAME
3	HR	EMP_EMAIL_UK	UNIQUE	VALID	NORMAL	N	NO	(null)	NO	EMAIL
4	HR	EMP_EMP_ID_PK	UNIQUE	VALID	NORMAL	N	NO	(null)	NO	<u>EMPLOYEE_ID</u>
5	HR	EMP_MANAGER_IX	NONUNIQUE	VALID	NORMAL	N	NO	(null)	NO	MANAGER_ID
6	HR	EMP_DEPARTMENT_IX	NONUNIQUE	VALID	NORMAL	N	NO	(null)	NO	DEPARTMENT_ID

Index를 다른  
데이터 형태로  
변경하면 안됨

# SQL Processing & Optimizer 예시

## Starts

오퍼레이션을 수행한 횟수를 의미한다. Starts \* E-Rows 의 값이 A-Rows 값과 비슷하다면, 통계정보의 예측 Row 수와 실제 실행 결과에 따른 실제 Row 수가 유사함을 알 수 있다. 만약 값에 큰 차이가 있다면 통계정보가 실제의 정보를 제대로 반영하지 못했다고 생각할 수 있다. 이로 인해 오라클의 Optimizer가 잘못된 실행 계획을 수립할 수도 있음을 염두에 두어야 한다.

## E-Rows (Estimated Rows)

통계정보에 근거한 예측 Row 수를 의미한다. 통계정보를 갱신할수록 값이 매번 다를 수 있으며, 대부분의 DB 운영에서는 통계정보를 수시로 갱신하지 않으므로 해당 값에 큰 의미를 둘 필요는 없다. 하지만 E-Rows 값과 A-Rows 값이 현격하게 차이가 있다면 오라클이 잘못된 실행 계획을 세울 수도 있음을 인지해야 하며 통계정보 생성을 검토해 보아야 한다.

## A-Rows (Actual Rows)

쿼리 실행 결과에 따른 실제 Row 수를 의미한다.

## A-Time (Actual Elapsed Time)

쿼리 실행 결과에 따른 실제 수행 시간을 의미한다. 하지만 실행 시점의 여러 상황이 늘 가변적이고 또한 메모리에 올라온 Block의 수에 따라서 수행 시간이 달라지므로 해당 값에 큰 의미를 둘 필요는 없다.

## Buffers (Logical Reads)

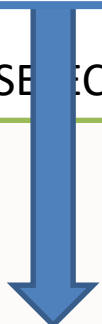
논리적인 Get Block 수를 의미한다. 해당 값은 오라클 옵티마이저가 일한 총량을 의미하므로, 튜닝을 진행할 때 가장 중요하게 생각하는 요소 중 하나다.

## SQL Processing & Optimizer 예시

```
> ALTER SESSION SET STATISTICS_LEVEL=ALL;
```

```
> SELECT * FROM HR.employees WHERE EMPLOYEE_ID > 200;
```

```
> SELECT * FROM TABLE(DBMS_XPLAN.DISPLAY_CURSOR(null,null,'ALLSTATS LAST'));
```



	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
1	201	Michael	Hartstein	MHARTSTE	515.123.5555	04/02/17	MK_MAN	13000	(null)	100	20
2	202	Pat	Fay	PFAY	603.123.6666	05/08/17	MK_REP	6000	(null)	201	20
3	203	Susan	Mavris	SMAVRIS	515.123.7777	02/06/07	HR_REP	6500	(null)	101	40
4	204	Hermann	Baer	HBAER	515.123.8888	02/06/07	PR_REP	10000	(null)	101	70
5	205	Shelley	Higgins	SHIGGINS	515.123.8080	02/06/07	AC_MGR	12008	(null)	101	110
6	206	William	Gietz	WGIETZ	515.123.8181	02/06/07	AC_ACCOUNT	8300	(null)	205	110



## SQL Processing & Optimizer 예시

### PLAN\_TABLE\_OUTPUT

```
1 SQL_ID 3dqpynkuh45uy, child number 0
2 -----
3 SELECT * FROM HR.employees WHERE EMPLOYEE_ID > 200
4
5 Plan hash value: 1781021061
6
7 -----
8 | Id | Operation | Name | Starts | E-Rows | A-Rows | A-Time | Buffers |
9 -----
10 | 0 | SELECT STATEMENT | | 1 | | 6 | 00:00:00.01 | 2 |
11 | 1 | TABLE ACCESS BY INDEX ROWID BATCHED | EMPLOYEES | 1 | 6 | 6 | 00:00:00.01 | 2 |
12 |* 2 | INDEX RANGE SCAN | EMP_EMP_ID_PK | 1 | 6 | 6 | 00:00:00.01 | 1 |
13 -----
14
15 Predicate Information (identified by operation id):
16 -----
17
18 2 - access("EMPLOYEE_ID">200)
19
```



```
> SELECT * FROM TABLE(DBMS_XPLAN.DISPLAY_CURSOR(null,null,'ALLSTATS LAST'));
```

7	-----							
8	Id	Operation	Name	Starts	E-Rows	A-Rows	A-Time	Buffers
9	-----							
10	0	SELECT STATEMENT		1		7	00:00:00.01	1
11	* 1	INDEX FULL SCAN	EMP_EMP_ID_PK	1	5	7	00:00:00.01	1
12	-----							
13								
14	Predicate Information (identified by operation id):							
15	-----							
16								
17	1 - filter(TO_CHAR("EMPLOYEE_ID") LIKE '2%')							

# SQL Processing & Optimizer 예시

## 실행계획 읽는 법

1. 위에서 아래로 읽어 내려가면서 제일 먼저 읽을 스텝 찾기
2. 내려가는 과정에서 같은 들여쓰기가 존재하면, 무조건 위에서 아래 순으로 읽기
3. 읽고자 하는 스텝보다 들여쓰기가 된 하위 스텝이 존재한다면, 가장 안쪽으로 들여쓰기 된 스텝을 시작으로 하여 한 단계씩 상위 스텝으로 읽어 나오기

ID	PID	OPERATION	
0		SELECT STATEMENT	
1	0	TABLE ACCESS BY INDEX ROWID	APIPLIST
2	1	NESTED LOOPS	
3	2	NESTED LOOPS	
4	3	VIEW	
5	4	SORT GROUP BY	
6	5	INDEX RANGE SCAN	SQ2DHIST_PK
7	3	TABLE ACCESS BY INDEX ROWID	APPATBAT
8	7	INDEX UNIQUE SCAN	APPATBAT_PK
9	2	INDEX RANGE SCAN	APIPLIST_PK

실행순서: 6 → 5 → 4 → 8 → 7 → 3 → 9 → 2 → 1

## SQL Processing & Optimizer 예시

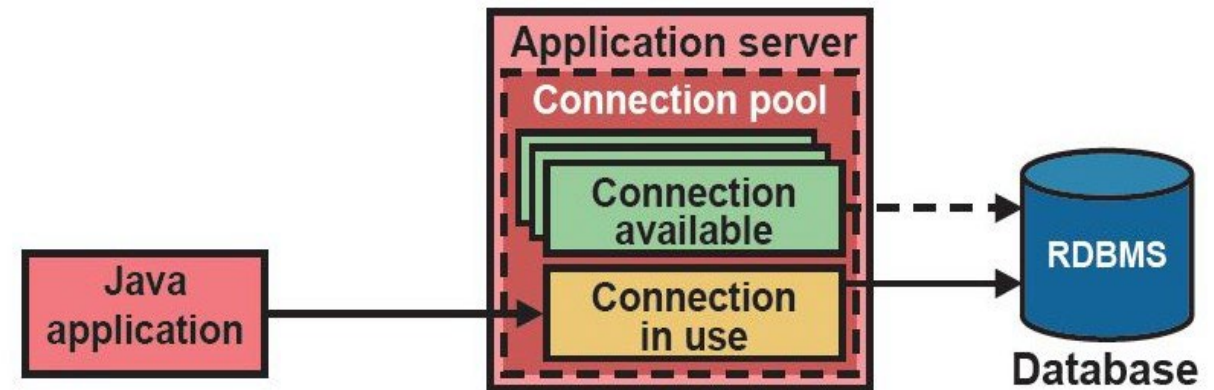
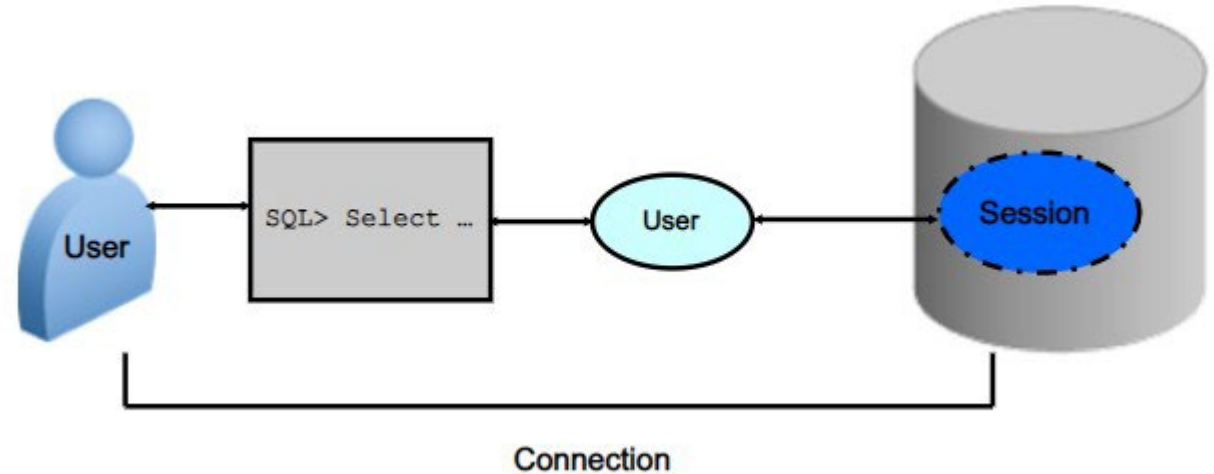
(퀴즈)  
실행계획 순서는?

ID	PID	OPERATION
0		SELECT STATEMENT
1	0	SORT GROUP BY
2	1	HASH JOIN
3	2	TABLE ACCESS BY INDEX ROWID
4	3	NESTED LOOPS
5	4	NESTED LOOPS
6	5	TABLE ACCESS BY INDEX ROWID
7	6	INDEX RANGE SCAN
8	5	TABLE ACCESS BY INDEX ROWID
9	8	INDEX RANGE SCAN
10	4	INDEX RANGE SCAN
11	2	VIEW
12	11	UNION-ALL
13	12	TABLE ACCESS BY INDEX ROWID
14	13	INDEX RANGE SCAN
15	12	TABLE ACCESS BY INDEX ROWID
16	15	INDEX RANGE SCAN

정답

## Connection & Running a SQL

- 다음을 사용하여 인스턴스에 접속(connection)
  - 사용자 프로세스
  - 서버 프로세스
- 사용하는 Oracle 서버 구성 요소는 SQL문 유형에 따라 다름
  - Query는 Parse → Execute → Fetch 3단계 과정을 거쳐 결과 행을 반환
  - DML 문은 변경 사항을 기록 (No fetch)
  - Commit은 트랜잭션을 보장
  - Rollback은 변경사항 이전으로 되돌림 (예: commit 전, user connection의 비정상 종료 시)



Connection pool 이용 시 미리 생성된 DB와의 connection을 사용하게 되므로 훨씬 효율적

# Agenda

---

- 1 History of Oracle Database
- 2 Oracle Database Server Architecture
- 3 Oracle Database – Storage
- 4 Oracle Instance, RAC, Multitenant, MAA
- 5 SQL
- 6 Transaction, Instance Recovery
- 7 Summary

# TRANSACTION

- DBMS에서 사용되는 쪼갤 수 없는 업무처리의 단위 (Atomicity = All or Nothing )

[100만원 계좌이체 할 경우]



트랜잭션은 DBMS에서 데이터를 다루는 논리적인 작업의 단위가 된다. 예를 들어 A계좌에서 B계좌로 돈을 이체하는 경우에 이 업무는 A에서 돈을 빼고 B에 돈을 더하는 2가지의 Update문으로 나뉘게 된다. 그리고 이것들은 개별적으로 수행되는 것이 아니라 하나의 트랜잭션으로 묶이게 되며 하나의 트랜잭션이 실행될 때 이 2개의 SQL문이 연속적으로 실행되게 된다. 그러므로 2개의 UPDATE문이 하나의 트랜잭션으로 묶여있다고 가정할 때 1개의 SQL만 실행되는 상황은 발생하지 않고 이를 All or Nothing 이라고 함

## Transaction Control

실행 가능한 SQL문장이 제일 처음 실행될 때 시작



# Transaction Control

## COMMIT과 ROLLBACK전후

Rollback/비정상종료

- 모든 변경 취소
- 변경행 LOCK해제
- SAVEPOINT 제거

Commit 전

- 데이터 변경
- 이전의 상태로 취소 가능
- 현재 사용자는 변경내용 select 가능
- 다른 사용자는 변경내용 참조 불가
- 변경된 행은 LOCK이 설정 되어서 다른 사용자가 동시에 변경할 수 없음



## Rollback of Transactions

데이터 변경사항이 취소되어 데이터를 이전 상태로 되돌리는 것

① SELECT \* FROM 직원; [데이터 건수 10건]

② DELETE \* FROM 직원;

③ SELECT \* FROM 직원; [데이터 건수 0건]

④ ROLLBACK;

⑤ SELECT \* FROM 직원; [데이터 건수 10건]

시간의 흐름

## Commit of Transactions

입력, 수정 및 삭제한 변경사항을 데이터베이스에 영구히 반영하는 것

세션 1

① SELECT \* FROM 직원; [데이터 건수 10건]

② INSERT INTO 직원 VALUES('11','XXX');  
[데이터 추가 1건]

③ SELECT \* FROM 직원; [데이터 건수 11건]

⑤ COMMIT;

세션 2

④ SELECT \* FROM 직원 : [데이터 건수 10건]

⑥ SELECT \* FROM 직원 : [데이터 건수 11건]

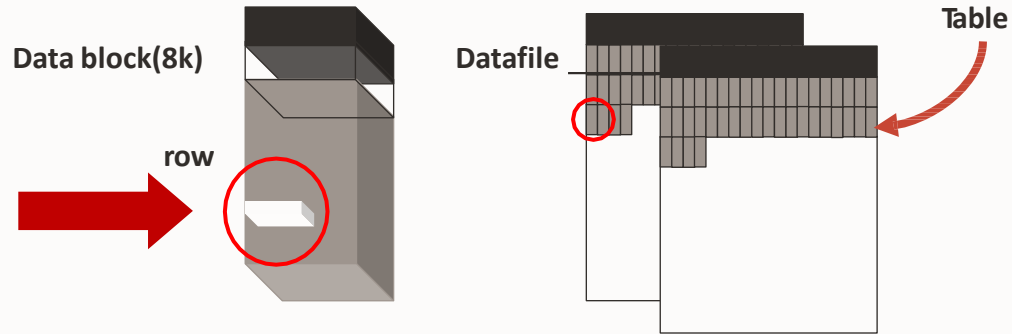
시간의 흐름

- Dirty Read 방지
- Writer가 Reader를 막지 않음
- 오라클은 UNDO를 이용한 다중 버전 읽기 일관성 모델(MVRC)을 채택함으로써 Dirty Read를 피해 일관성 있는 데이터 읽기가 가능
- Read Committed

# 데이터 동시성 및 일관성

- Oracle Database Locking Mechanism의 장점

✓ Row level Lock  
해당 row 만 lock 설정 contention 최소화



Writer      Block      Writer

Writer      Reader

Reader      Writer

✓ 아무리 많은 row 를 변경해도 절대 lock escalation 하지 않음

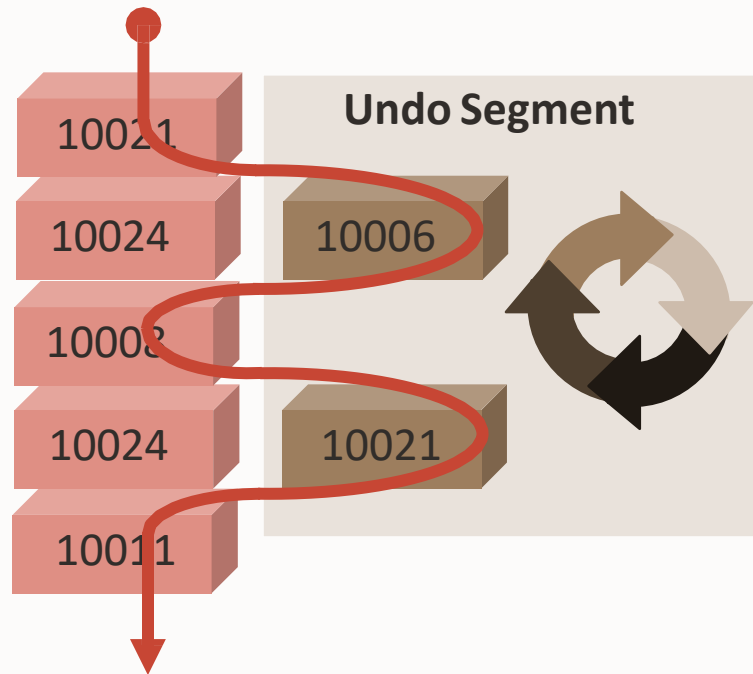
✓ Multi-version Read Consistency Model

✓ Select 시 exclusive lock 설정하지 않음

# 데이터 동시성 및 일관성

## Oracle Multi-version Read Consistency (MVRC) / Multi-version Consistency Control (MVCC)

Select SCN 10023



SCN + Undo  
데이터를 활용

높은 동시성  
읽기 일관성

Rollback  
Recovery  
Flashback

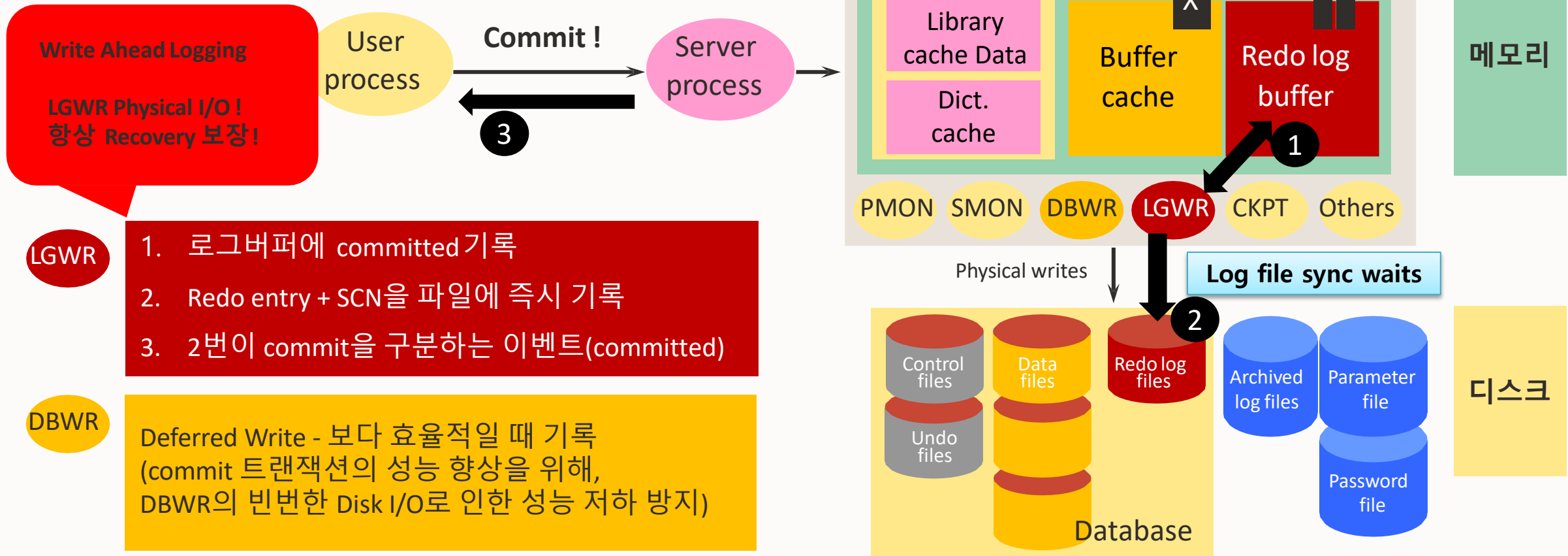
## (참고) Lock Escalation

### Oracle에서는 발생하지 않는 Lock Escalation이란?

- Transaction 내에서 많은 row의 변경을 처리할 때 Row 레벨에서 Block 또는 Table 레벨로 Lock이 전이 되는 현상
- 오라클은 별도의 Lock 매니저 없이 해당 row가 있는 데이터 블록에 Lock 정보를 저장하지만, Lock 정보를 메모리에서 관리하는 DBMS에서는 공통적으로 발생
- Lock Escalation이 발생할 경우 Transaction의 동시성은 급격히 감소하며 전반적인 성능 문제 유발

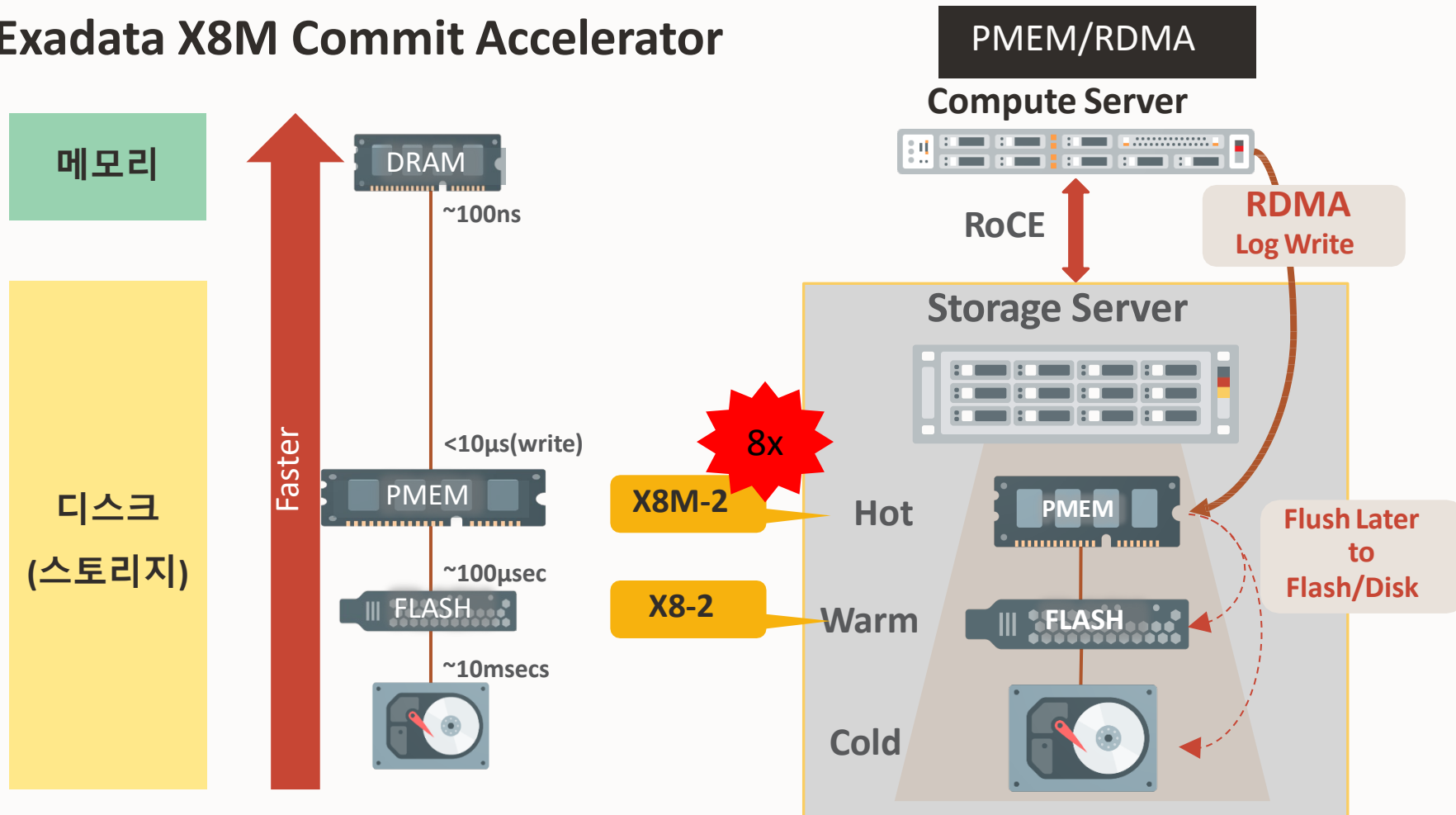
# Commit of Transactions

## Fast Commit Mechanism



# Commit of Transactions

## Exadata X8M Commit Accelerator



# Instance Recovery

## Instance Recovery Phases

- First phase – cache recovery (rolling forward) : online redo log에 기록된 모든 변경 내용을 data file에 적용
- Second phase – rolling back (transaction recovery) : Undo block을 적용하여 commit 되지 않은 변경 내용을 rollback

Figure 13–6 Basic Instance Recovery Steps: Rolling Forward and Rolling Back

