

# Oracle Process Architecture 개요

Oracle 19c Database Concepts 공식 문서 기반

## 1. 프로세스란 무엇인가

Oracle에서 **프로세스(Process)**는 OS 레벨에서 Oracle 코드를 실행하는 단위다. 운영체제에 따라 구현이 다르다:

- **Linux/UNIX:** OS 프로세스 또는 OS 프로세스 내의 스레드
- **Windows:** 프로세스 내의 스레드

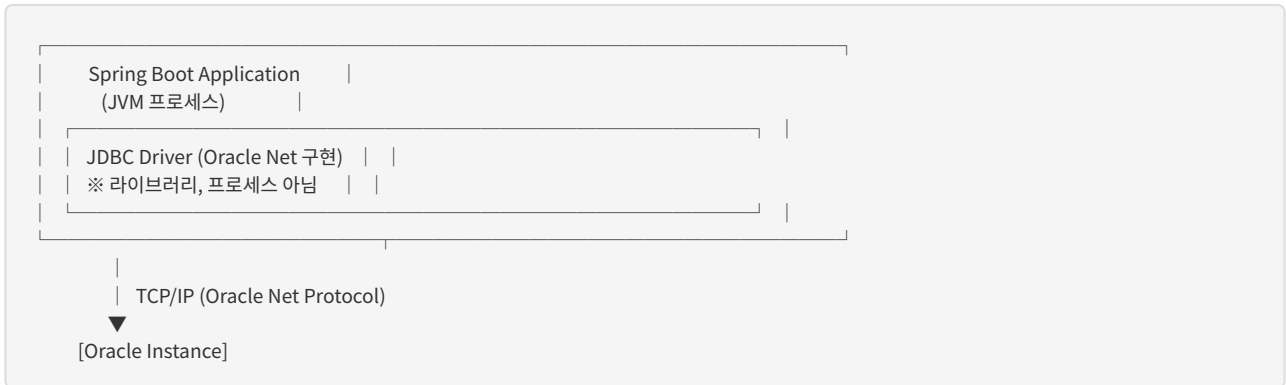
Spring Boot 애플리케이션이 Oracle에 연결하면, **Oracle 측에서 프로세스가 생성된다는 것**이 핵심이다. 단순히 “소켓 연결”이 아니라 **OS 레벨의 리소스 할당**이 발생한다.

## 2. Oracle 프로세스의 세 가지 유형



## 2.1 Client Process

Client Process는 **Oracle 외부**에서 실행되는 프로세스다. Spring Boot 애플리케이션(JVM 프로세스), SQL\*Plus 등이 여기에 해당한다.



**JDBC Driver의 정확한 위치:** JDBC Driver는 독립된 프로세스가 아니라, Client Process(JVM) **내부에서 실행되는 라이브러리**다. Oracle Net Protocol을 구현하여 Server와 통신하는 코드일 뿐, OS가 자원을 할당하는 실행 단위가 아니다.

Client Process는 Oracle 서버의 리소스를 직접 소비하지 않는다. 네트워크를 통해 **요청을 전달**할 뿐이다.

## 2.2 Server Process

Server Process는 **Client의 요청을 실제로 처리**하는 Oracle 측 프로세스다. 이것이 Connection 생성 비용의 핵심이다.

Server Process가 수행하는 작업:

1. SQL 문장 파싱 (Syntax/Semantic Check)
2. 파싱된 SQL과 실행 계획을 **Shared Pool 내의 Library Cache**에 캐싱
3. 쿼리 실행
4. Buffer Cache 또는 디스크에서 데이터 읽기
5. 결과를 Client에게 반환

**Library Cache 명시 이유:** Shared Pool은 여러 하위 구조를 포함하는 상위 개념이다. 실행 계획이 저장되는 정확한 위치는 Library Cache이며, Shared Pool에는 이 외에도 Data Dictionary Cache, Reserved Pool 등이 포함된다. Memory Architecture 이해 시 이 계층 구조 구분이 필수적이다.

**핵심:** Server Process는 **자신만의 메모리 영역(PGA)**을 갖는다.

## 2.3 Background Process

Background Process는 **Instance가 시작될 때 함께 시작**되어 유지보수 작업을 수행한다. Client 요청과 무관하게 항상 실행된다.

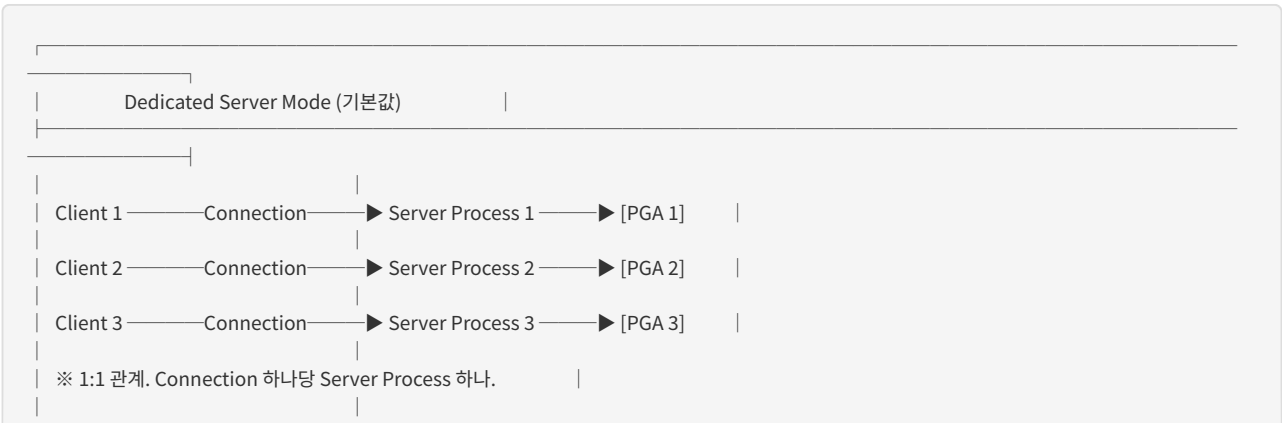
백엔드 개발자가 알아야 할 주요 Background Process:

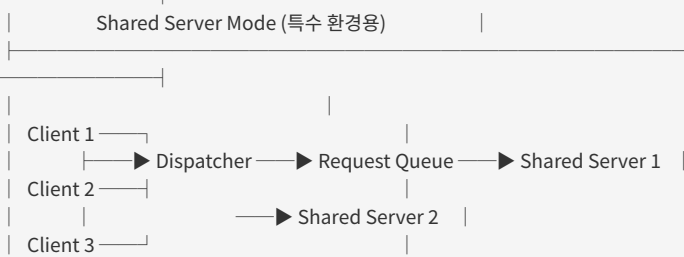
Process	역할	개발자 연관성
<b>LGWR</b> (Log Writer)	Redo Log Buffer → Redo Log Files 기록	커밋 시 sync write 발생. DBWn 기록 전 선행 필수 (WAL 원칙)
<b>DBWn</b> (Database Writer)	Buffer Cache → Data Files 기록	dirty buffer 임계값, 체크포인트 등 다중 조건으로 트리거
<b>PMON</b> (Process Monitor)	비정상 종료된 프로세스 정리	Connection이 비정상 종료되면 PMON이 정리
<b>SMON</b> (System Monitor)	인스턴스 복구, 임시 세그먼트 정리	인스턴스 크래시 후 재시작 시 복구 수행. 시작 지연 원인

**LGWR와 DBWn의 의존 관계:** DBWn이 dirty buffer를 datafile에 기록하려면, 해당 변경에 대한 redo 레코드가 **먼저** redo log file에 기록되어야 한다. 이것이 WAL(Write-Ahead Logging) 원칙이며, 장애 복구 시 redo log를 재생하여 커밋된 트랜잭션을 복구할 수 있는 기반이다.

## 3. Dedicated Server vs Shared Server

Oracle은 두 가지 Server Process 모드를 지원한다. **Spring Boot + HikariCP 환경에서는 거의 100% Dedicated Server 모드를** 사용한다.



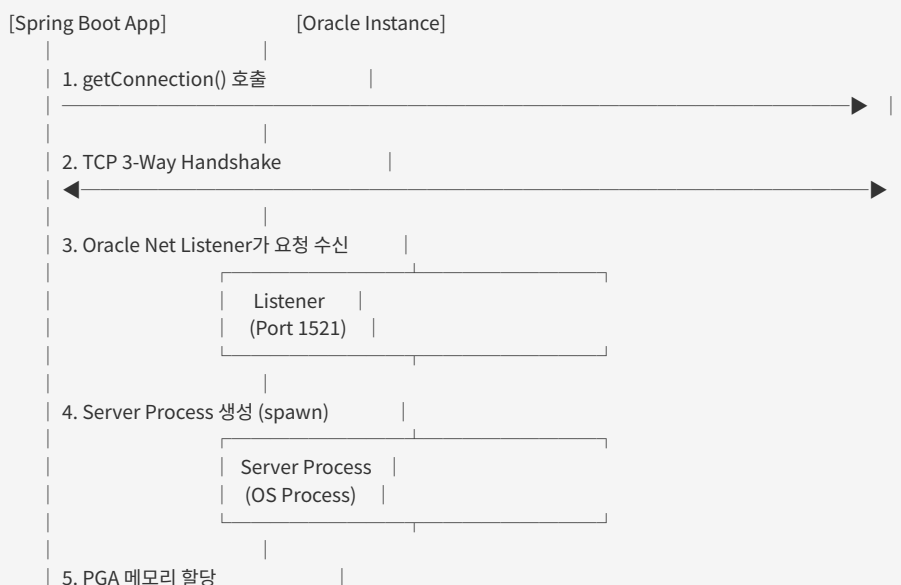


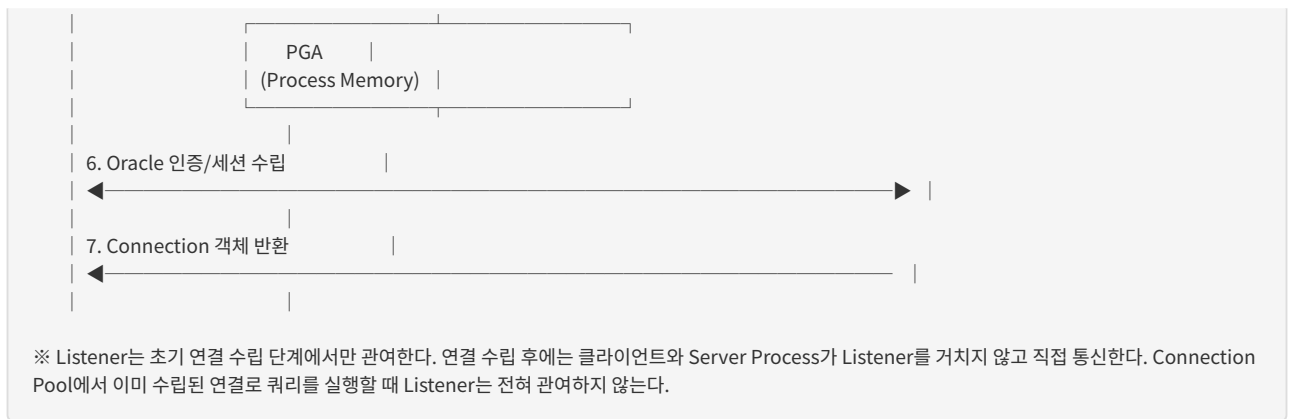
- ※ N:M 관계. 여러 Client가 적은 수의 Server Process를 공유.
- ※ 수천 개의 동시 연결이 필요한 특수 환경에서만 사용.

## 왜 Dedicated Server가 기본인가?

- Shared Server는 세션 상태(UGA)를 SGA의 Large Pool에 저장하여 메모리 압박 발생 가능
- JDBC Thin 드라이버의 일부 기능(LOB 처리, Scrollable ResultSet 등)이 Shared Server에서 제한되거나 비효율적
- 요청마다 세션 상태를 복원해야 하므로 latency-sensitive 작업에 불리
- 대부분의 웹 애플리케이션은 Connection Pool로 연결 수를 제한하므로 Dedicated로 충분
- HikariCP의 `maximumPoolSize=10` 이면 최대 10개의 Server Process만 생성된다

## 4. Connection 생성 시 발생하는 일 (Dedicated Server 기준)





**4단계 “spawn”에 대한 기술적 명확화:** Listener가 Server Process를 생성하는 방식은 Unix의 `fork()` 시스템 콜과 다르다. `fork()` 는 부모 프로세스의 메모리 공간을 복제하여 자식을 생성하는 것인데, Server Process는 Listener의 복제본이 아니다. 실제 동작은 `oracle` 실행 파일의 새로운 인스턴스를 **spawn(생성)**하고, 연결을 새 프로세스에 **handoff**하거나 클라이언트에게 새 주소를 **redirect**하는 방식이다.

**비용 발생 지점:**

단계	비용 유형	비고
TCP Handshake	네트워크 I/O	네트워크 지연에 비례
Server Process 생성	<b>OS 레벨 프로세스 생성</b>	OS, 하드웨어 사양에 따라 변동
PGA 할당	<b>메모리 할당</b>	PGA_AGGREGATE_TARGET 설정에 영향
인증/세션 수립	CPU + I/O	인증 방식에 따라 변동

**수치 추정에 대한 주의사항:** Connection 생성에 수십~수백 밀리초가 소요된다는 것은 일반적으로 알려진 사실이나, 구체적인 밀리초 단위 수치는 서버 사양, Oracle 버전, OS 종류(Linux의 copy-on-write 최적화 등), 네트워크 환경에 따라 크게 변동한다. 정확한 수치가 필요한 경우 해당 환경에서 AWR 리포트나 직접 측정을 통해 확인해야 한다.

**핵심 결론:** Connection 생성은 단순 소켓 연결이 아니라 **OS 프로세스 생성 + 메모리 할당 + 인증**을 포함하므로 비용이 크다. 이것이 Connection Pool 사용의 아키텍처적 근거다.

## 5. 프로젝트 연결: HikariCP Waiting의 원인

1차 테스트에서 HikariCP Waiting이 최대 9건 발생했다. 이것을 프로세스 아키텍처로 설명하면:

[HikariCP Pool: 10개]

```
graph LR
    subgraph Pool [HikariCP Pool: 10개]
        C1[Connection 1] --> SP1[Server Process 1] --> PGA1[PGA 1]
        C2[Connection 2] --> SP2[Server Process 2] --> PGA2[PGA 2]
        C3[Connection 3] --> SP3[Server Process 3] --> PGA3[PGA 3]
        Dots[...] 
        C10[Connection 10] --> SP10[Server Process 10] --> PGA10[PGA 10]
    end
    PGA1 --> Q[Waiting Queue: 9건]
    PGA2 --> Q
    PGA3 --> Q
    Dots --> Q
    PGA10 --> Q
```

Connection 1 → Server Process 1 → [PGA 1] ← N+1 쿼리 25회 실행 |  
Connection 2 → Server Process 2 → [PGA 2] ← 다른 요청 처리 중 |  
Connection 3 → Server Process 3 → [PGA 3] ← 다른 요청 처리 중 |  
...  
Connection 10 → Server Process 10 → [PGA 10] ← 다른 요청 처리 |

```
graph LR
    subgraph Queue [Waiting Queue: 9건]
        A[요청 A] --> C1[Connection 대기 중]
        B[요청 B] --> C2[Connection 대기 중]
        Dots[...] 
        I[요청 I] --> Cn[Connection 대기 중]
    end
```

[Waiting Queue: 9건]  
요청 A — Connection 대기 중  
요청 B — Connection 대기 중  
...  
요청 I — Connection 대기 중

N+1 패턴으로 25회 쿼리를 순차 실행하는 동안 **하나의 Connection(= 하나의 Server Process)**을 계속 점유한다. 그 동안 다른 요청들은 Pool에서 빈 Connection이 나오기를 기다린다.

## 6. 다음 학습 단계

이제 프로세스 개요를 이해했으므로, 공식 문서에서 다음 순서로 학습하면 된다:

순 서	섹션명	URL 경로	소요 시간
1	Overview of PGA	<a href="https://docs.oracle.com/en/database/oracle/oracle-database/19/cncpt/process-architecture.html#GUID-310F28E9-793A-4816-87CB-FDB54AB68957">https://docs.oracle.com/en/database/oracle/oracle-database/19/cncpt/process-architecture.html#GUID-310F28E9-793A-4816-87CB-FDB54AB68957</a> → Overview of PGA	40분
2	Dedicated Server Processes	<a href="https://docs.oracle.com/en/database/oracle/oracle-database/19/cncpt/process-architecture.html#GUID-8EB10017-7930-45CE-8012-2E1DE8DC4EB2">https://docs.oracle.com/en/database/oracle/oracle-database/19/cncpt/process-architecture.html#GUID-8EB10017-7930-45CE-8012-2E1DE8DC4EB2</a> → Dedicated Server Processes	20분
3	How Oracle Database Creates	<a href="https://docs.oracle.com/en/database/oracle/oracle-database/19/cncpt/process-architecture.html#GUID-13FE4098-61DF-4D76-882D-551A88E0EBB8">https://docs.oracle.com/en/database/oracle/oracle-database/19/cncpt/process-architecture.html#GUID-13FE4098-61DF-4D76-882D-551A88E0EBB8</a> → How Oracle Database Creates Server Processes	30분

순 서	섹션명	URL 경로	소요 시간
	Server Processes		

## 부록: 수정 이력

버전	수정 내용
v3	LGWR, DBWn 트리거 조건 명확화 및 WAL 원칙 설명 추가
v3	SMON 프로세스 추가
v3	Shared Server 제약 사항 보완 (UGA 위치, JDBC 기능 제한)
v3	Listener 역할 범위 명확화 주석 추가
v3	PGA 표기를 “Session Memory”에서 “Process Memory”로 정정
v2	JDBC Driver가 라이브러리를 명확화 (프로세스 아님)
v2	Server Process 생성 방식을 “fork”에서 “spawn”으로 정정, 기술적 차이 설명 추가
v2	“Shared Pool에 배치”를 “Shared Pool 내의 Library Cache에 캐싱”으로 정정
v2	Connection 생성 소요 시간의 구체적 수치 제거, 환경 의존성 명시
v2	ASCII 다이어그램 가로 폭 통일