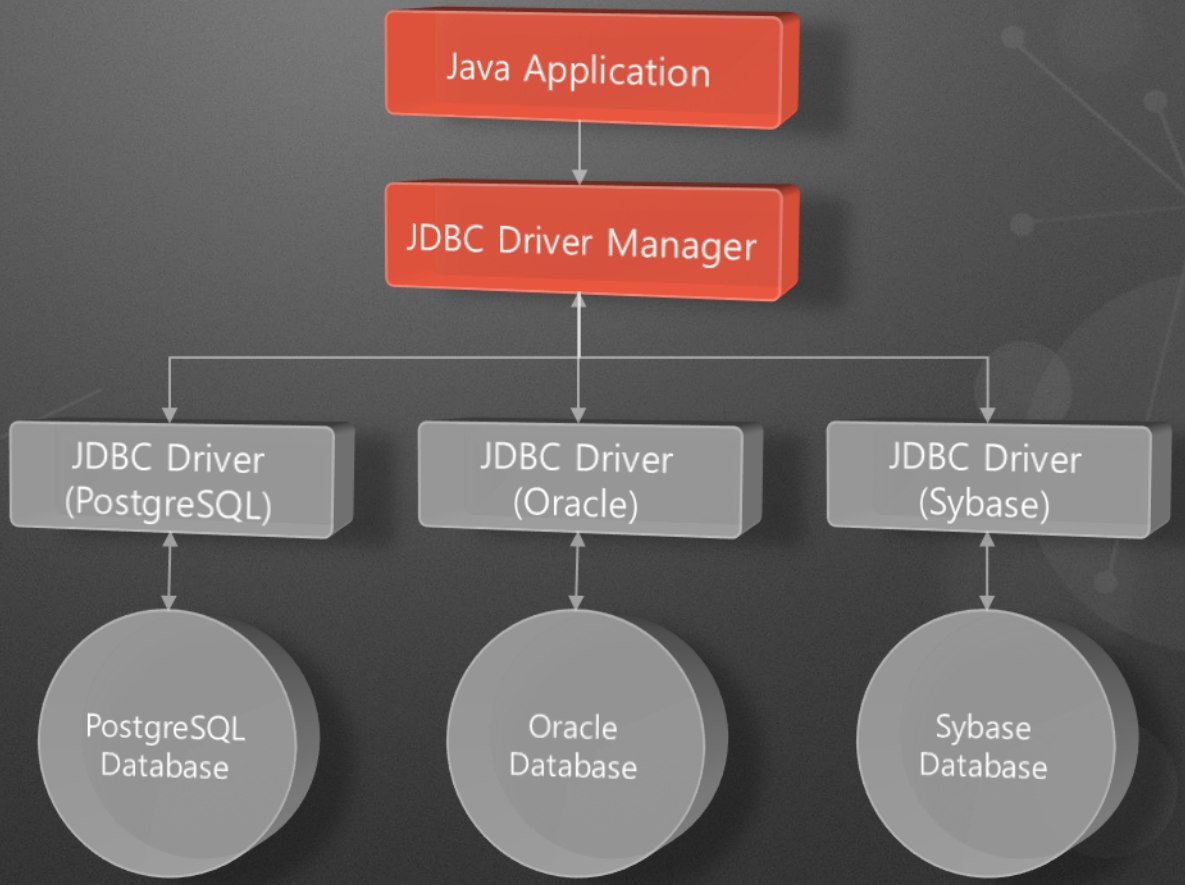

opennaru
DB & JDBC 장애 패턴
KNOWLEDGE

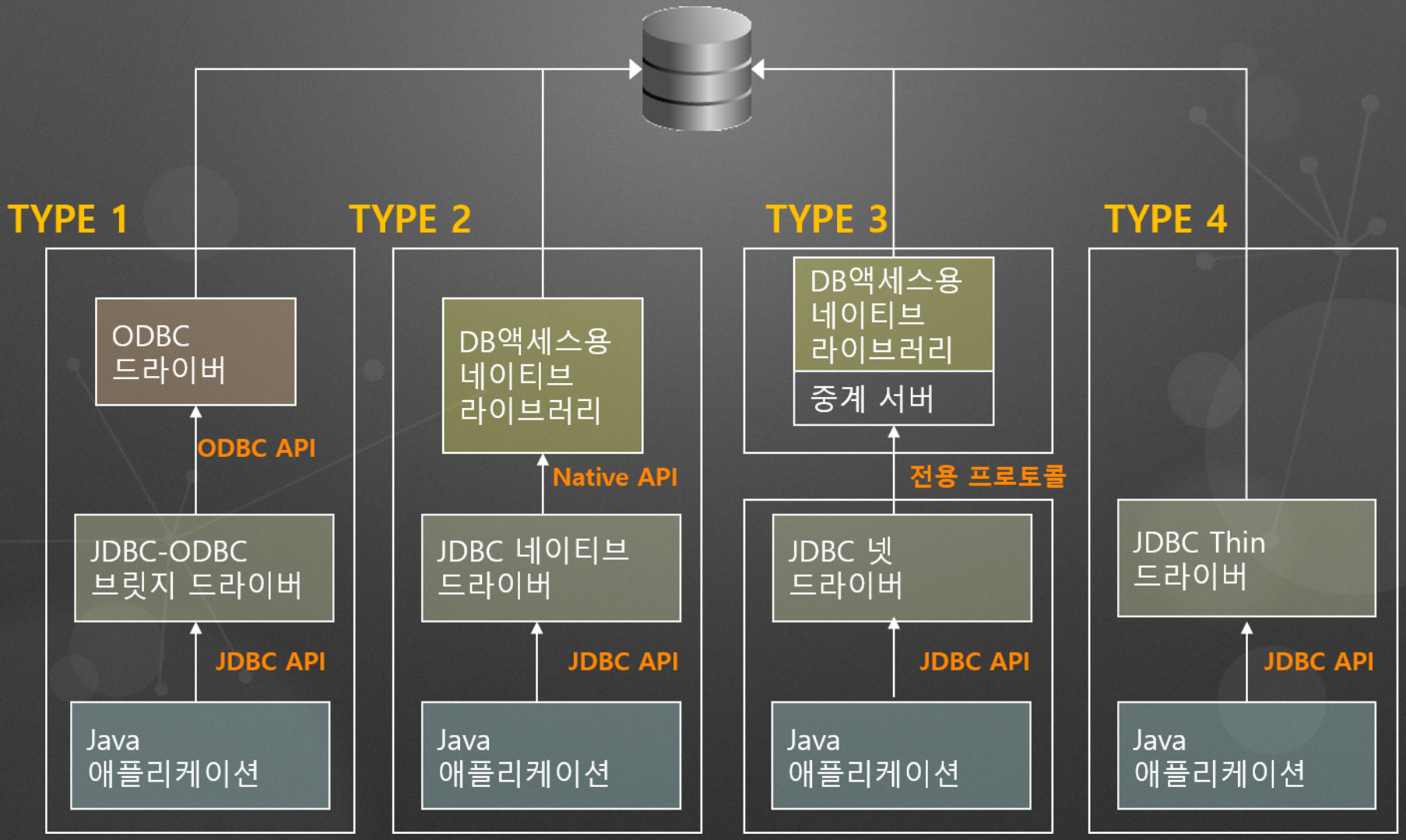
JDBC란?

- JDBC가 하는일
  - 데이터베이스 연결
  - SQL문장 전송
  - 결과 처리



JDBC 드라이버의 타입

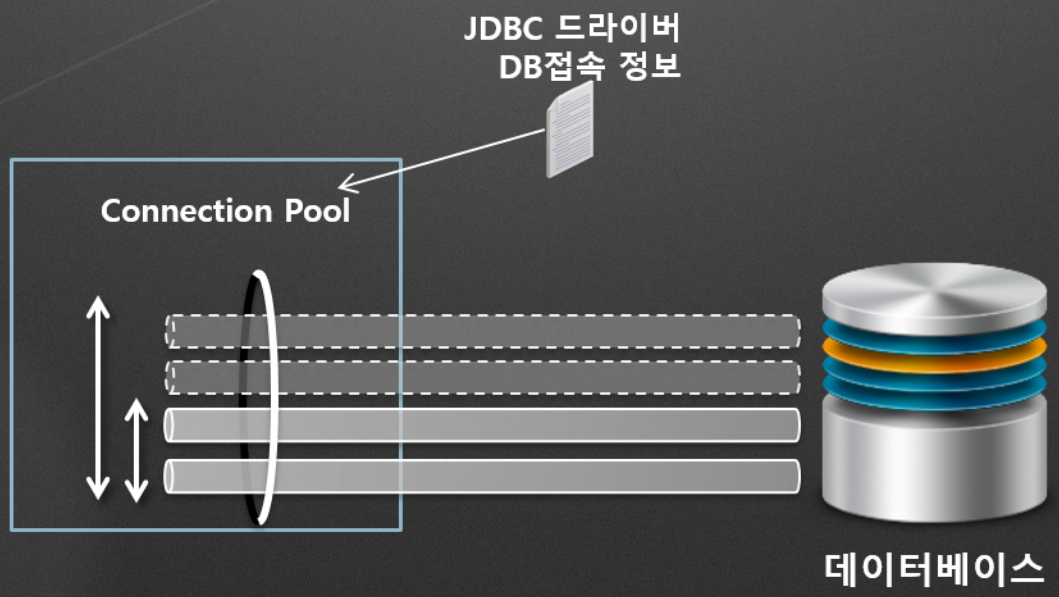
opennaru



Connection Pool 관리

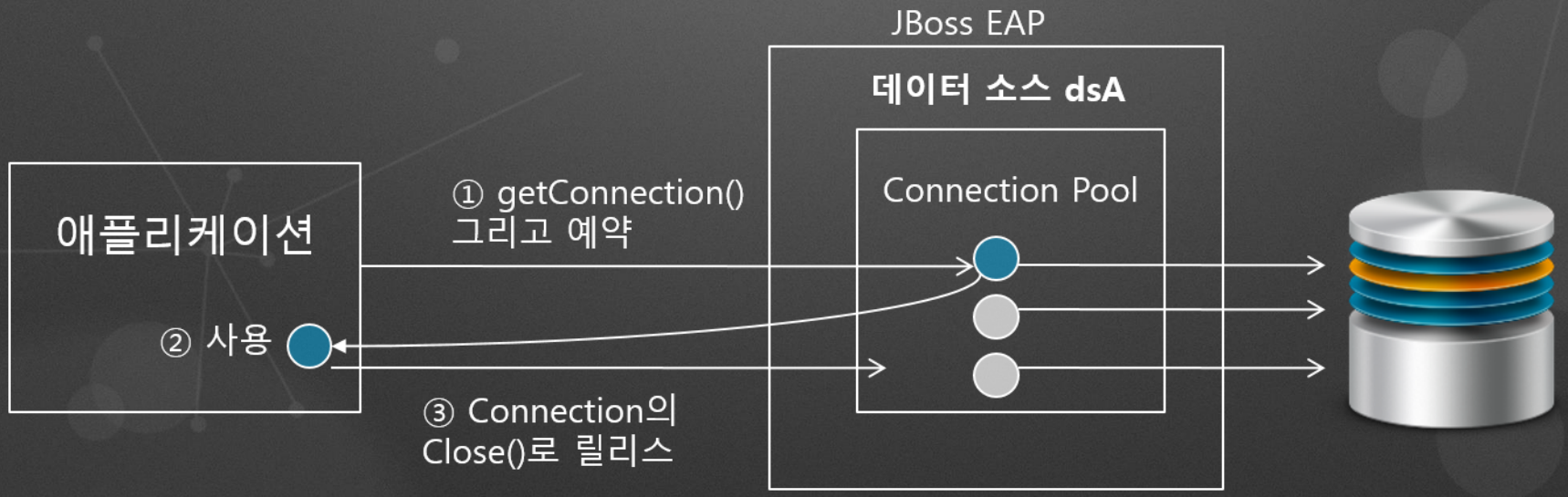
opennaru

- 커넥션 풀은 DB와의 커넥션을 풀로 관리해 애플리케이션에서 DB접속이 요구되었을 때 신속한 접속을 제공
- 커넥션 풀에 대하여 다음과 같이 설정
  - 최대 접속 수(max-pool-size) = 최소 접속 수(min-pool-size)
  - 초기 커넥션 연결 (pool-prefill)
  - 과부하 시 최소 커넥션 보장(pool-use-strict-min)
  - Idle 커넥션 자동 삭제(idle-timeout-minutes)



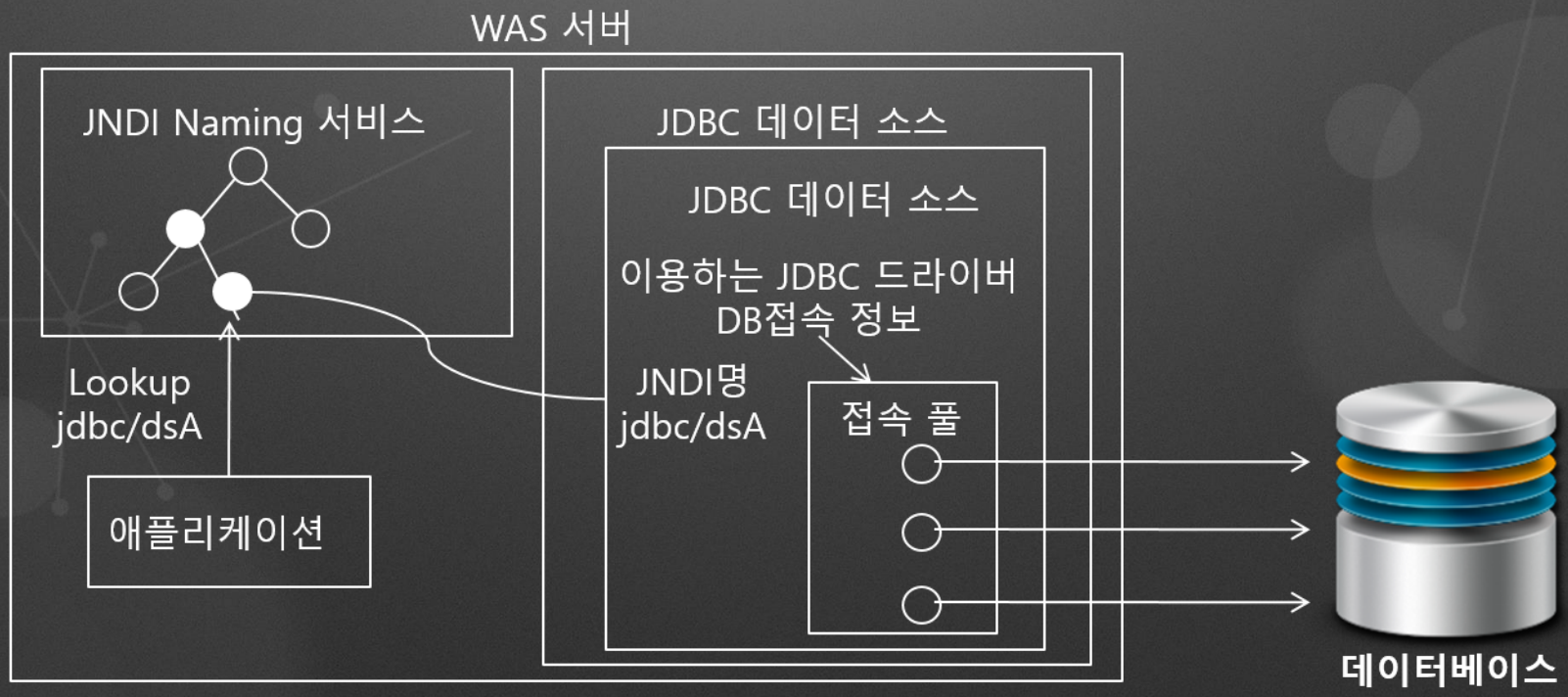
Connection Pool 사용 방법

- 애플리케이션이 데이터소스를 이용해 Connection을 요청하면, 데이터 소스의 접속 풀에 사용하지 않는 Connection이 있으면, 그것을 애플리케이션에서 가져와 사용한다.
- 애플리케이션은 사용 후 Connection을 Release(Close)하면, Connection 풀에 다시 넣는다.



JDBC 데이터 소스란

- 애플리케이션 실행 환경(WAS)에서 애플리케이션에 데이터베이스 접속 서비스를 제공하는 기능
- 애플리케이션은, DB접속에 필요한 물리적인 정보(DB 호스트명, DB유저 ID나 패스워드 등)를 의식하지 않고 데이터베이스에 접속하여 쿼리를 실행
- 접속 풀을 활용하는 것으로 DB접속, 종료 처리의 오버헤드를 없애 성능 향상 가능



PreparedStatement 캐시

opennaru

- JDBC로, 조건치만 변화하는 같은 SQL를 반복해 실행하는 경우는, 일반적으로 PreparedStatement를 사용한다. 그 경우, DB측의 해석 처리 횟수가 줄어들기 때문에, 반복 실행하는 경우는 Statement 사용시부터 성능 향상을 기대할 수 있다.
- JBoss의 JDBC 데이터 소스에서는, 이 PrepareStatement나 CallableStatement를 캐시하는 기능을 제공한다.
- PrepareStatement를 사용하는 애플리케이션에서는 성능 향상을 기대할 수 있다.

Statement 예제

```
Statement stmt = conn.createStatement();
for ( int id = 0 ; id < 10000 ; id++ ) {
    String sql = "SELECT ENAME FROM EMP WHERE EMPNO = " + id;
    ResultSet rs = stmt.executeQuery(sql);
    while ( rs.next() ) {
      // ResultSet 처리
    }
}
```

PreparedStatement 예제

```
String sql = "SELECT ENAME FROM EMP WHERE EMPNO = ?";
PreparedStatement    ps = conn.prepareStatement(sql);
for ( int id = 0 ; id < 10000 ; id++ ) {
    ps.setInt(1, id);
    ResultSet rs = ps.executeQuery();
    while ( rs.next() ) {
      // ResultSet 처리
    }
}
```

JDBC 코드 – Select

opennaru

```
import java.sql. *;
...
String sql = "select * from emp";  //실행하는 SQL문
DriverManager.registerDriver(new oracle.jdbc.driver.OracleDriver());
String url = "jdbc:oracle:thin:hostname:1521:SID";
Connection conn =DriverManager.getConnection(url, "SCOTT" , "TIGER");  //connection 연결
Statement stmt = conn.createStatement();   //Statement 생성
ResultSet rset = stmt.executeQuery(sql);   //Resultset 생성
...
rset.close();   // ResultSet close 처리
stmt.close();   //Statement close 처리
conn.close();   //connection close 처리
```

데이터 소스를 사용한 DB액세스에 필요한 패키지 임포트

JDBC URL
접속할 데이터베이스 지정

1. JDBC Driver 등록

2. Connection 생성

3. Statement 생성

4. SQL문(SELECT문) 실행

5. 쿼리 결과 처리

6. 접속 종료

※ JDBC의 사용중에 에러가 발생했을 경우, 데이터베이스에 액세스하는 모든 메소드는 SQLException를 Throw 합니다.

JDBC를 사용한 SELECT문 실행

opennaru

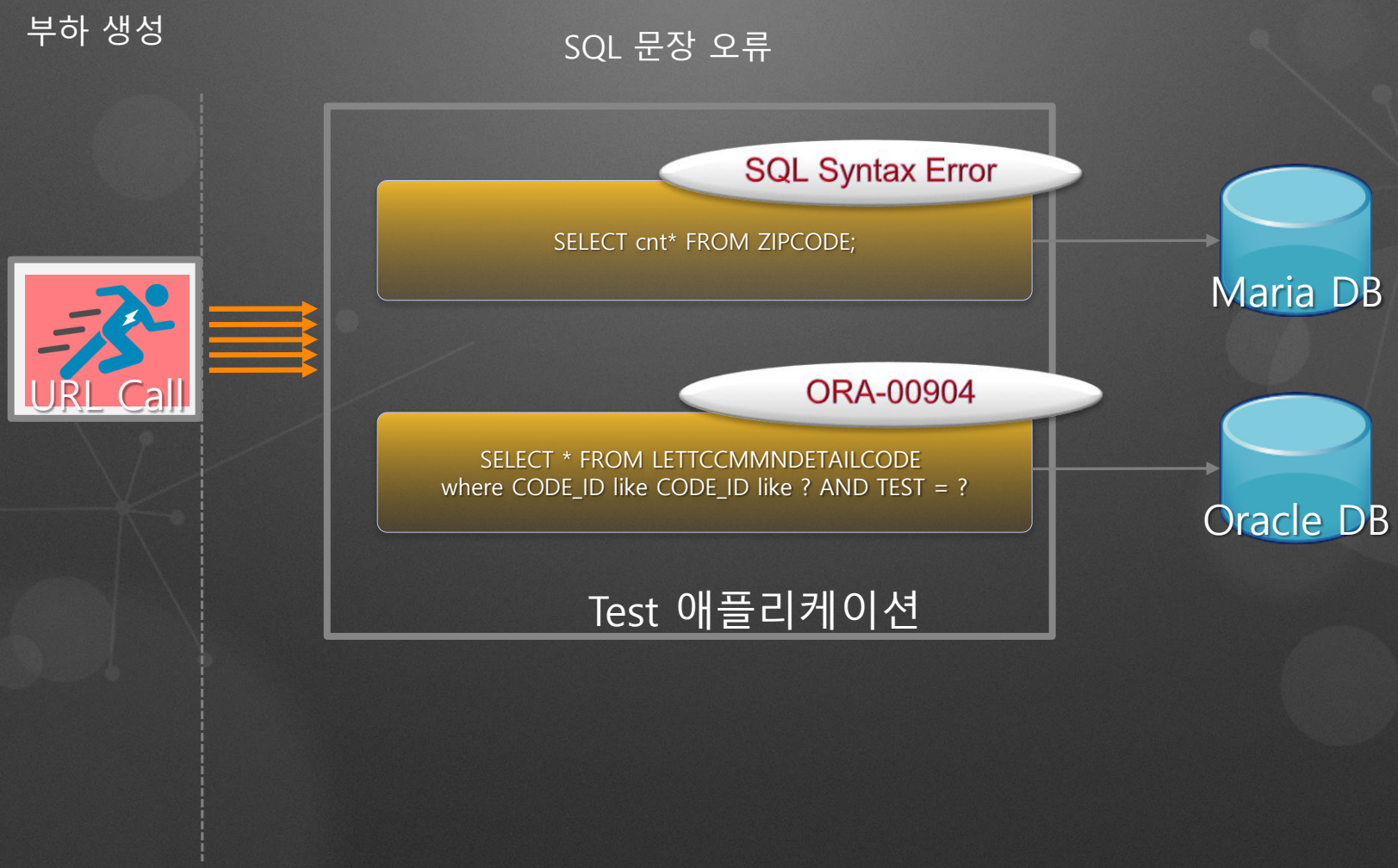
0. JDBC Driver 준비	접속할 데이터베이스에 대한 JDBC Driver를 준비한다
1. JDBC Driver 등록	JDBC Driver를 등록한다.
2. Connection 생성	Connection 객체를 얻는다. (DriverManager를 사용)
3. Statement 오브젝트 생성	Statement 객체를 작성합니다. (Connection 객체를 사용)
4. SQL문(SELECT문) 실행	쿼리를 위해서 executeQuery() 메서드를 실행합니다. (Statement 객체를 사용)
5. 쿼리 결과 처리	쿼리를 실행해 ResultSet 객체를 얻는 경우, ResultSet를 반복 처리해, 각 행의 데이터를 처리합니다.
6. 접속 종료	종료 후 ResultSet, Statement와 Connection 객체를 close한다.


opennaru
Extreme Demo
KHAN [a p m]

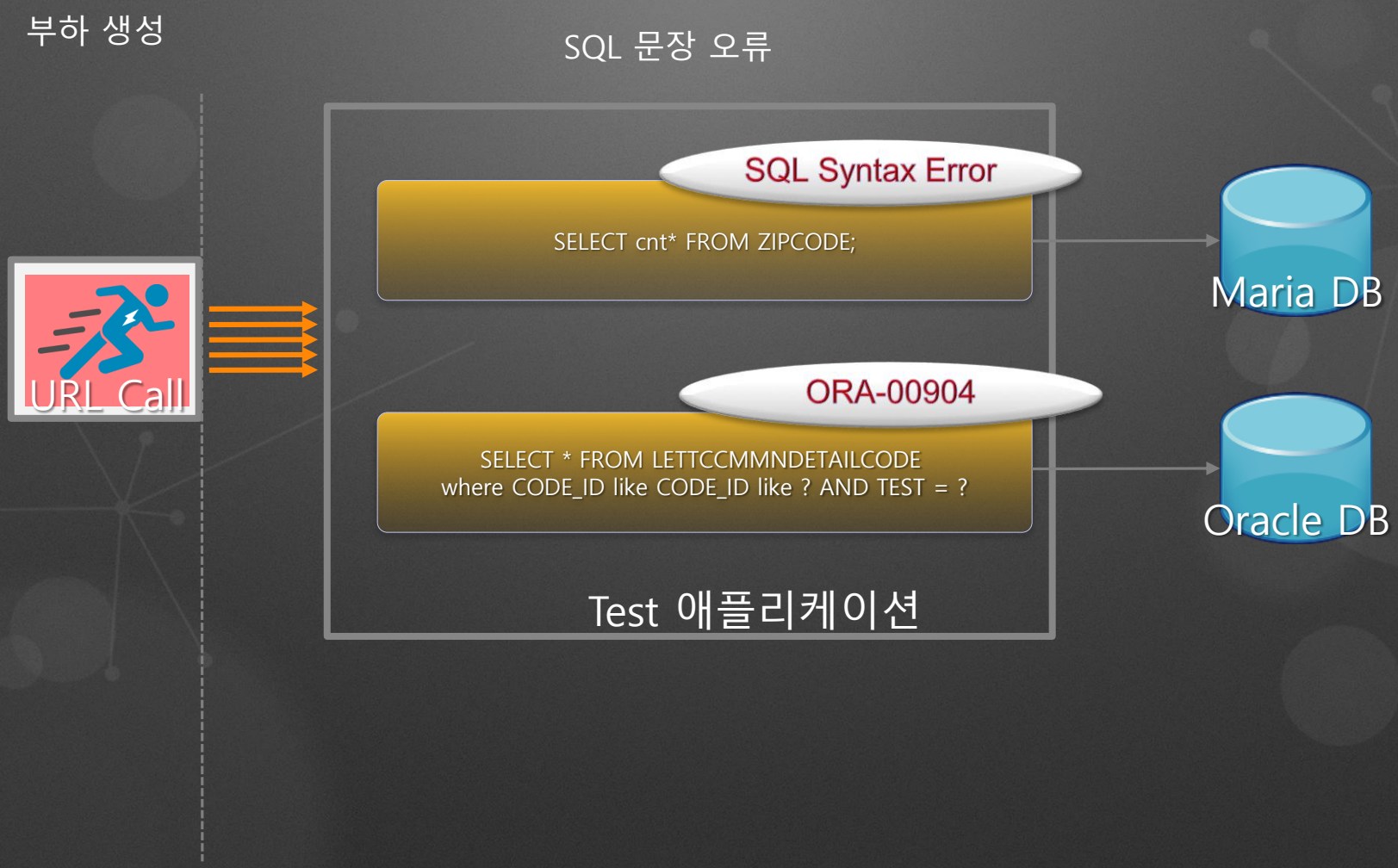
대규모 SQL Fetch

DB Connection Leak

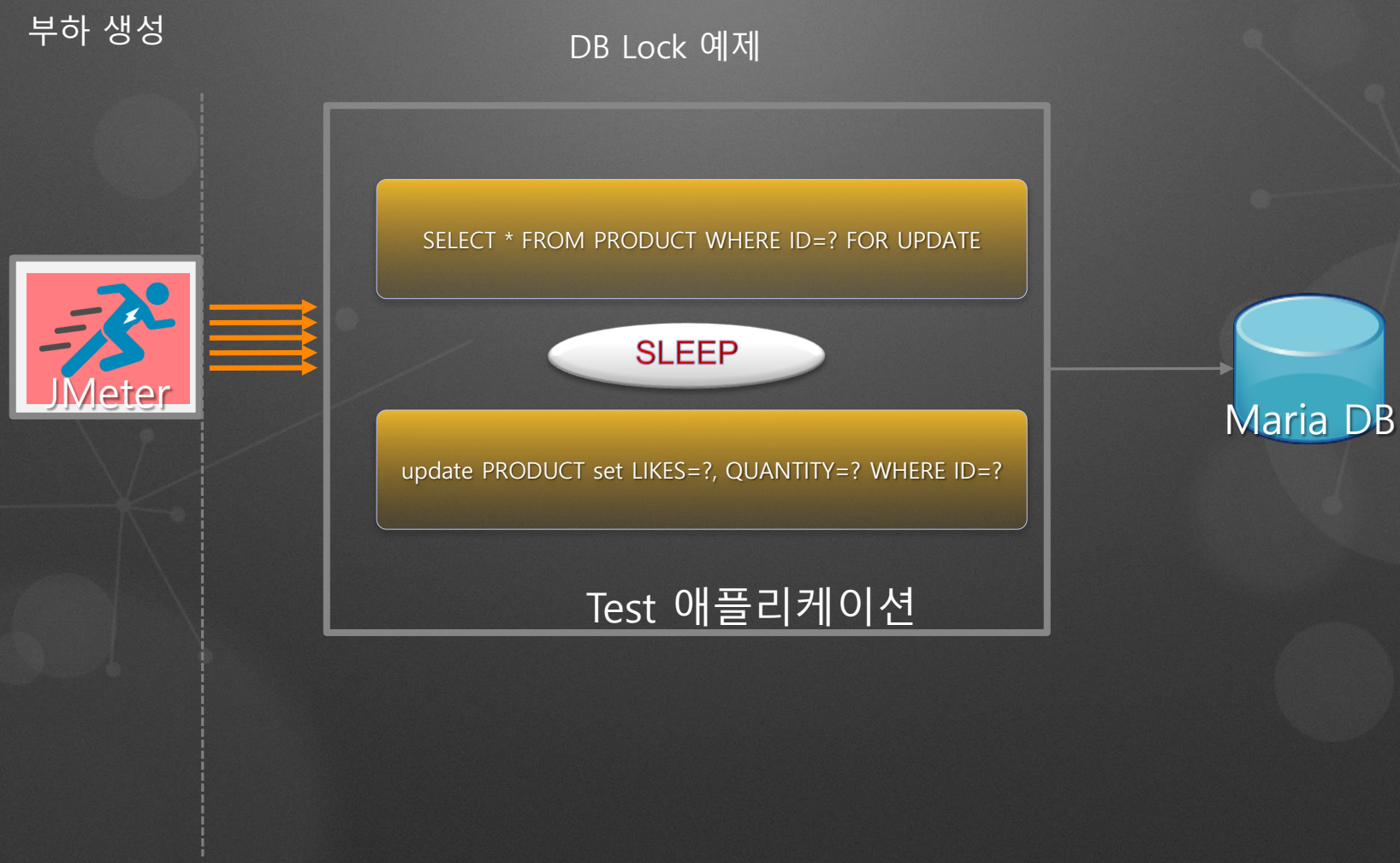
SQL Exception

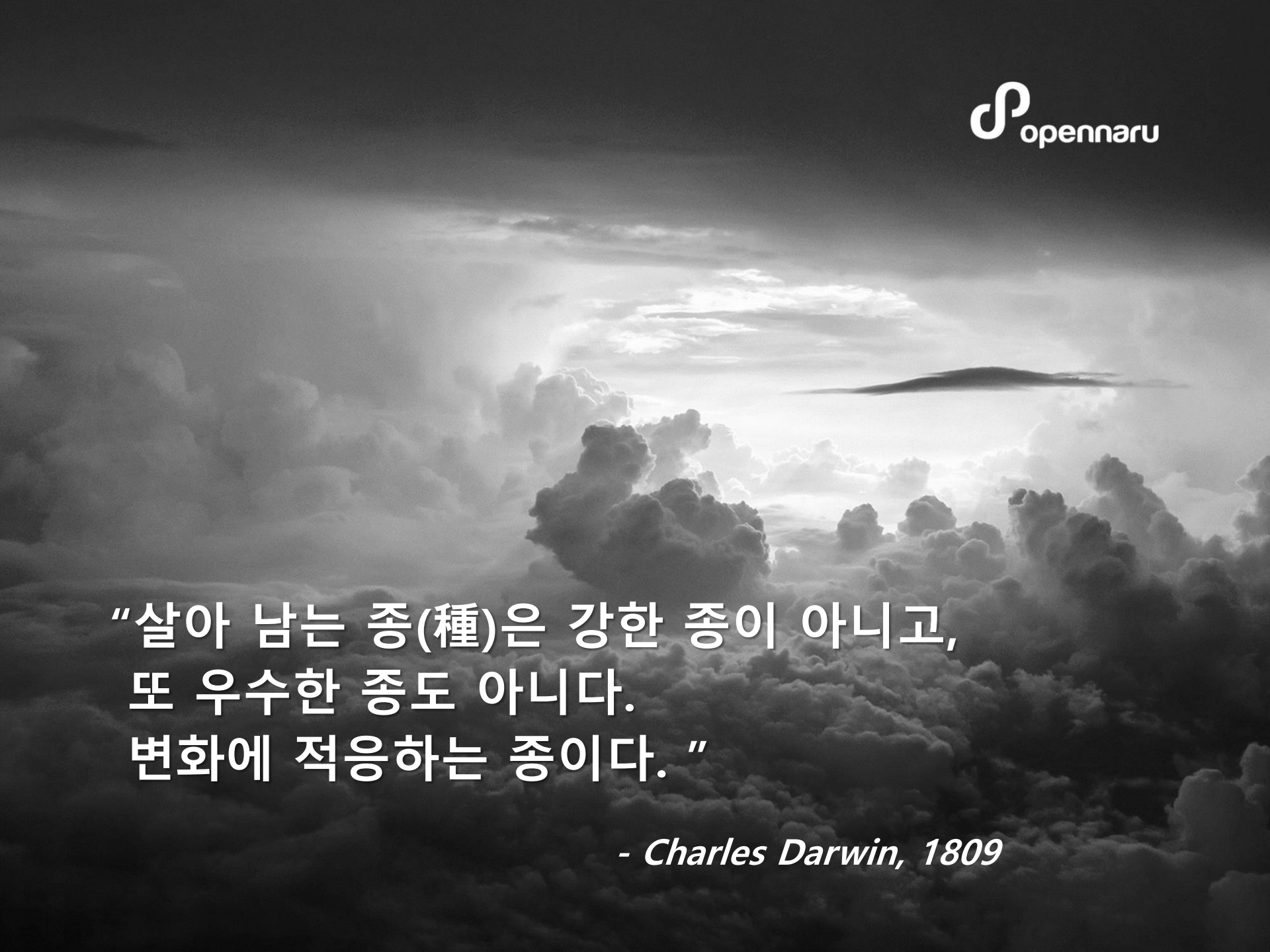


SLOW Query



DB Lock




opennaru
“살아 남는 종(種)은 강한 종이 아니고,
또 우수한 종도 아니다.
변화에 적응하는 종이다. ”
- Charles Darwin, 1809

opennaru

감사합니다.

제품이나 서비스에 관한 문의

콜 센터 : 02-469-5426 (휴대폰 : 010-2243-3394)

전자 메일 : sales@opennaru.com