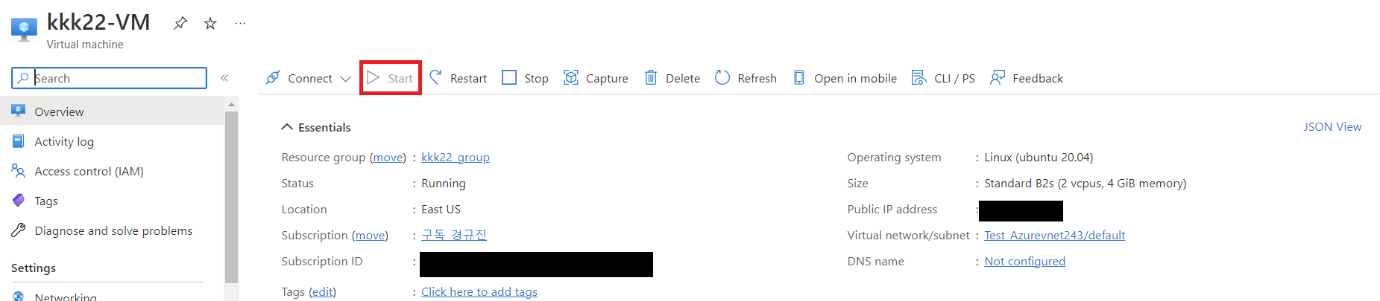
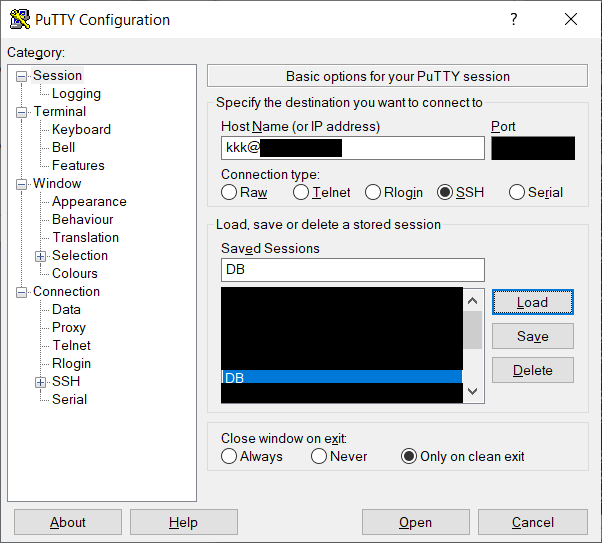
1. 프로그램 실행 및 사용 방법

1) Azure Platform에서 VM 실행.



2. VM에 SSH를 통해 연결



* Window의 경우, PuTTY 혹은 MobaXterm 프로그램을 통해 SSH 연결하는 것을 추천한다.
* MAC의 경우, 기본 터미널에서 SSH 연결 명령어 (ssh kkk@ip\_주소) 를 통해 해당 서버에 접속한다.

3. VM 내 명령어 (참고 사항)

================================================================

컨테이너를 새로 올리는 경우,

>> sudo docker run -dit --name oracle -e ORACLE\_PASSWORD=[PASSWD] -p 1521:1521 gvenzl/oracle-xe

이전 컨테이너를 재기동하는 경우,

>> sudo docker restart [컨테이너 명]

이전 컨테이너를 중지 후 삭제한는 경우,

>> sudo docker stop [컨테이너 명]

>> sudo docker rm [컨테이너 명]

컨테이너 동작 유무 확인.

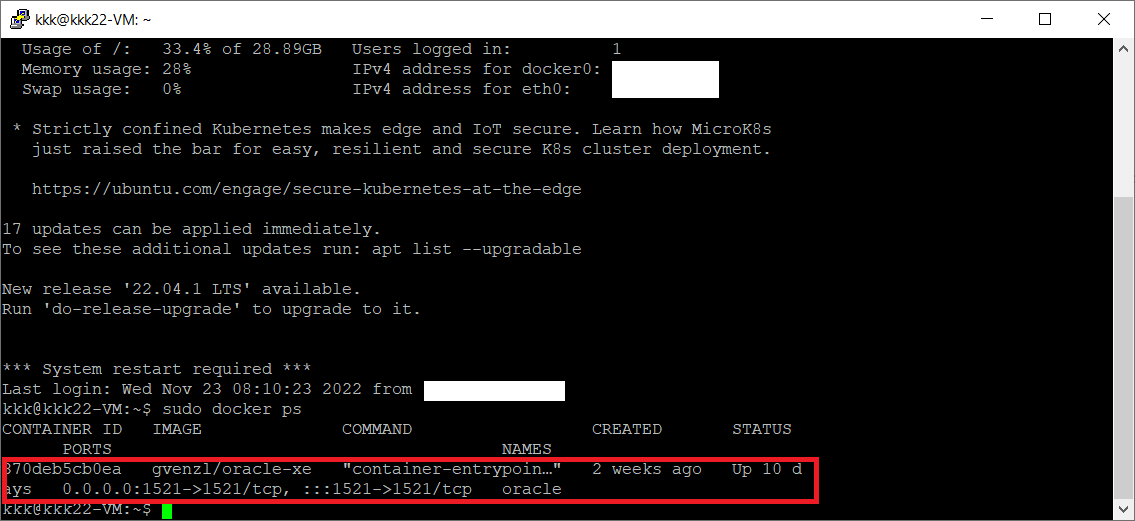
>> sudo docker ps

보통의 경우, 컨테이너를 재기동 하여 Oracle DB 서비스를 지원해야하기 때문에

>> sudo docker restart [컨테이너 명]

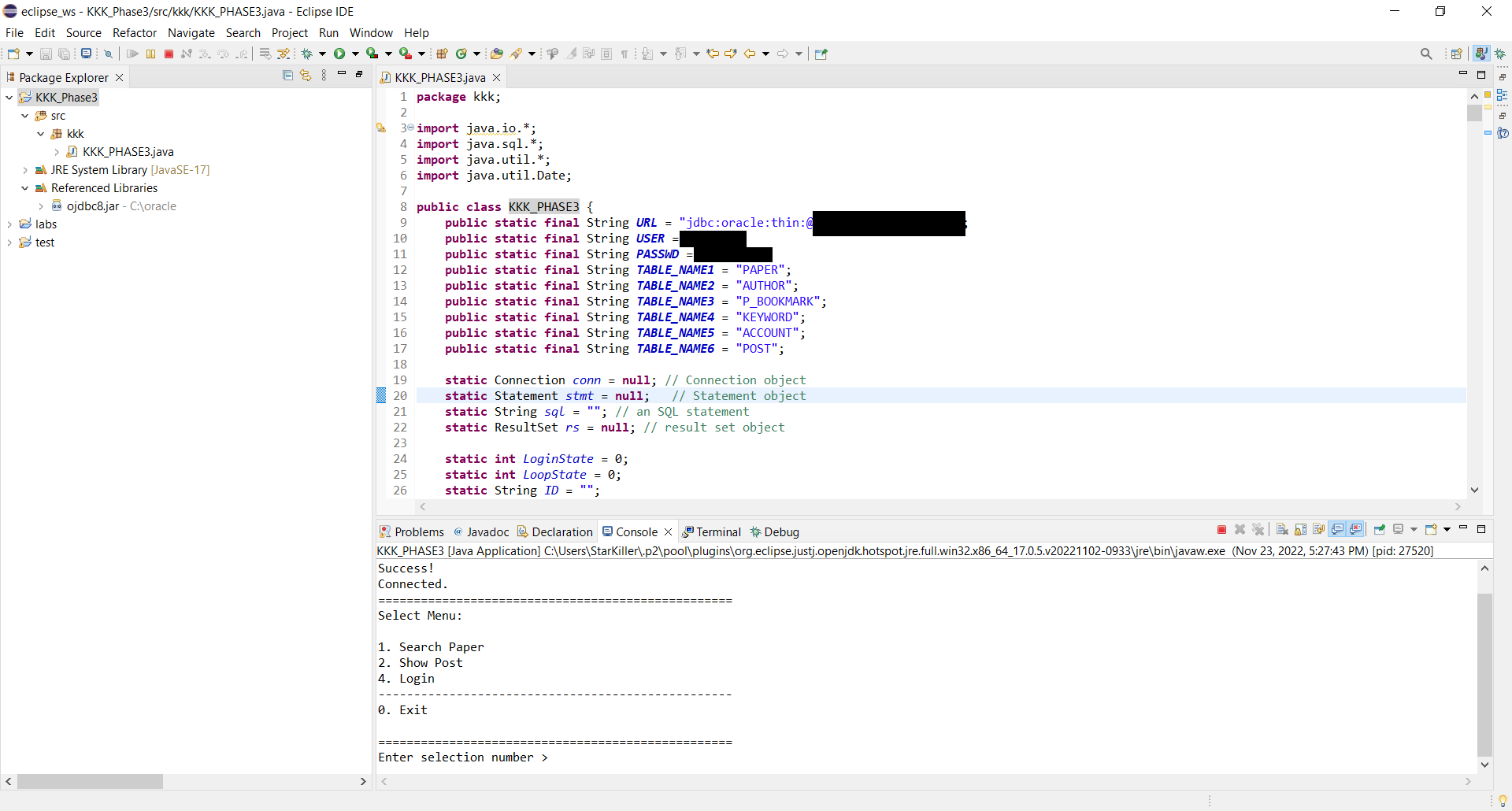
를 실행시키면 1~2분 후 KKK팀 Oracle DB에 접속할 수 있다.

================================================================



* 정상적으로 컨테이너가 운영되는 상황이라면 다음 화면과 같이 docker ps 명령어 수행 시 컨테이너가 작동함을 확인할 수 있다.

4. ZIP 파일에서 “KKK\_Phase3” 폴더를 eclipse 작업환경에 올린 후 KKK\_PHASE3.java 실행.



* 정상적인 프로그램 동작화면: 따로 설정할 필요가 없다.

2. 프로그램 순서도

Diagram, schematic

Description automatically generated

3. 프로그램 주요 기능

1) **public** **static** **void** SearchPaper(**int** type, String name)

|  |  |  |
| --- | --- | --- |
| 파라미터 | 타입 | 설명 |
| type | Int | <0(논문명), 1(저자명), 2(키워드명)>을 입력으로 받음.  입력에 따라 각각에 해당하는 검색 쿼리문 생성. |
| name | String | 검색 쿼리문에 들어가는 검색어를 입력으로 받음. |

2) **public** **static** **void** BookMarkPaper(String UserID, **int** DOI)

|  |  |  |
| --- | --- | --- |
| 파라미터 | 타입 | 설명 |
| UserID | String |  |
| DOI | Int |  |

3) **public** **static** **void** ShowPaperDetail(**int** DOI)

|  |  |  |
| --- | --- | --- |
| 파라미터 | 타입 | 설명 |
| DOI | Int |  |

4) **public** **static** **void** ShowAuthorDetail(**int** Rnum)

|  |  |  |
| --- | --- | --- |
| 파라미터 | 타입 | 설명 |
| Rnum | Int |  |

5) **public** **static** **void** ShowPostAll(**int** kind)

|  |  |  |
| --- | --- | --- |
| 파라미터 | 타입 | 설명 |
| kind | Int |  |

6) **public** **static** **void** ShowPostDetail(**int** PID)

|  |  |  |
| --- | --- | --- |
| 파라미터 | 타입 | 설명 |
| PID | Int |  |

7) **public** **static** **void** ShowBookMark(String UserID)

|  |  |  |
| --- | --- | --- |
| 파라미터 | 타입 | 설명 |
| UserID | String |  |

8) **public** **static** **void** CreateAccount(String UserId, String UserPW)

|  |  |  |
| --- | --- | --- |
| 파라미터 | 타입 | 설명 |
| UserID | String |  |
| UserPW | String |  |

9) **public** **static** **void** LoginAccount(String UserID, String UserPW)

|  |  |  |
| --- | --- | --- |
| 파라미터 | 타입 | 설명 |
| UserID | String |  |
| UserPW | String |  |

10) **public** **static** **void** DeleteAccount(String UserId, String UserPW)

|  |  |  |
| --- | --- | --- |
| 파라미터 | 타입 | 설명 |
| UserID | String |  |
| UserPW | String |  |

4. 유의 사항

#1) 로컬 환경에서 DB를 구축하는 경우,

1. SQL\_Phase3 폴더의 “[KKK팀] Phase3-(CREATE).sql” 내 쿼리 실행.
2. SQL\_Phase3 폴더의 “[KKK팀] Phase3-(INSERT).sql” 내 쿼리 실행.
3. SQL\_Phase3 폴더의 “[KKK팀] Phase3-(ALTER).sql” 내 쿼리 실행.
4. KKK\_Phase3 폴더의 “KKK\_PHASE3.java”에서 9~11번째 라인 코드  
   **public** **static** **final** String ***URL*** = "jdbc:oracle:thin:@ ~ ";  
   **public** **static** **final** String ***USER*** =" ~ ";  
   **public** **static** **final** String ***PASSWD*** =" ~ ";  
   사용자 로컬 환경에 맞게 설정.

#2) **쿼리 변경 사항**

|  |  |  |
| --- | --- | --- |
| 순번 | 변경전 | 변경후 |
| #1 |  |  |
| #2 |  |  |
| #3 |  |  |

5. Application 개발 환경

|  |  |
| --- | --- |
| **Item** | **Version** |
| Eclipse IDE for Java Developers | 2022-09 (4.25.0) |
| JDBC | Oracle DBMS 19c |
| Java Runtime Environment | Java SE 17 |
| JDBC Driver | ojdbc8.jar |