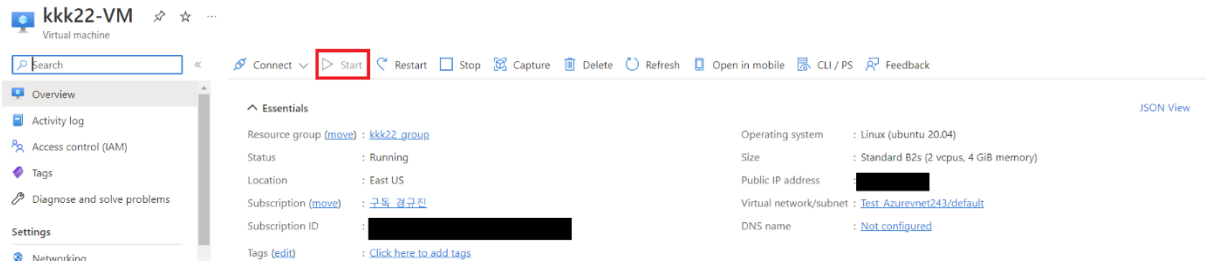
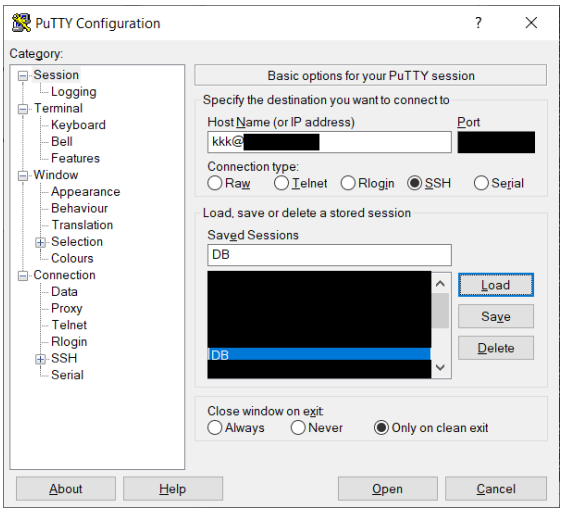
1. 프로그램 실행 및 사용 방법

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.

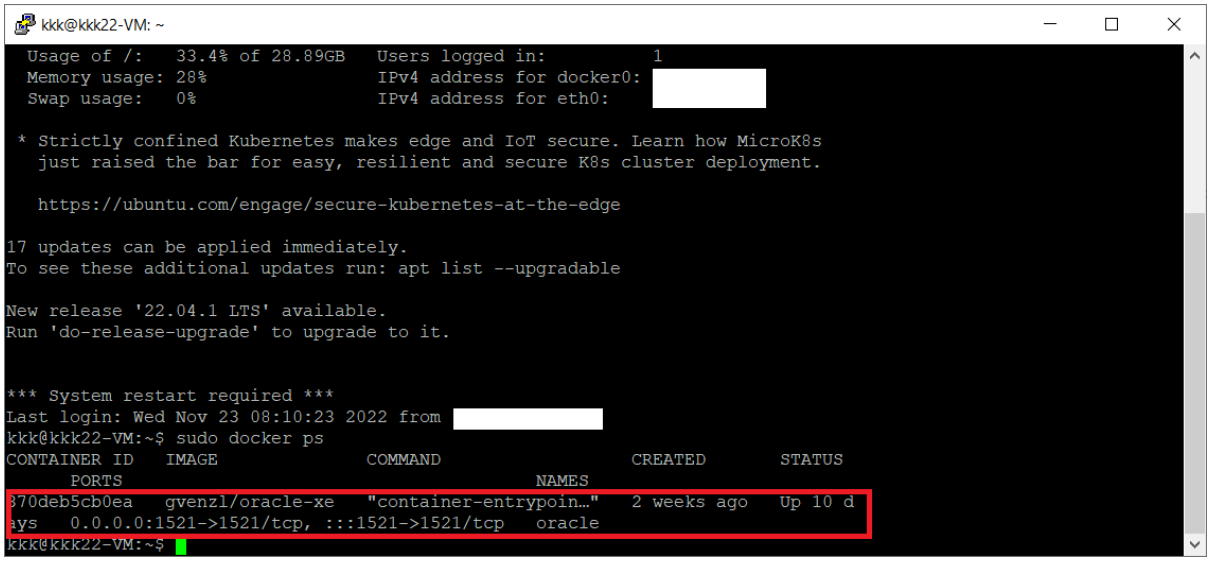
>> sudo docker ps

보통의 경우, 컨테이너를 재기동 하여 Oracle DB 서비스를 지원해야하기 때문에

>> sudo docker restart [컨테이너 명]

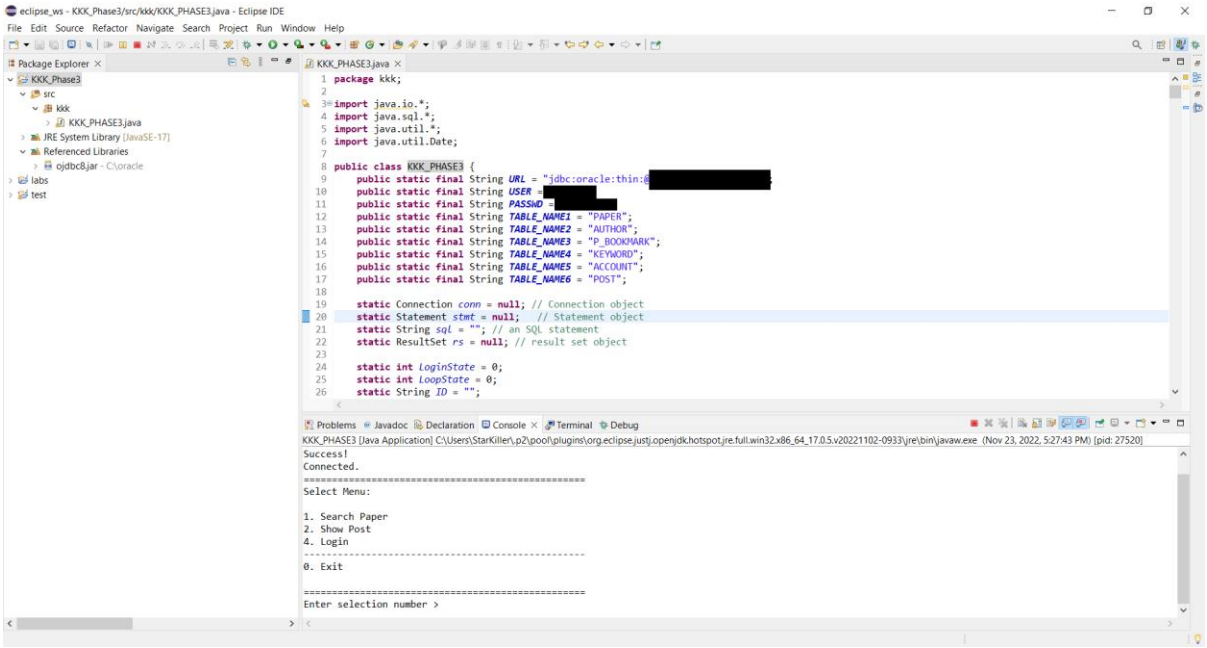
를 실행시키면 1~2분 후 KKK팀 Oracle DB에 접속할 수 있다.

==========



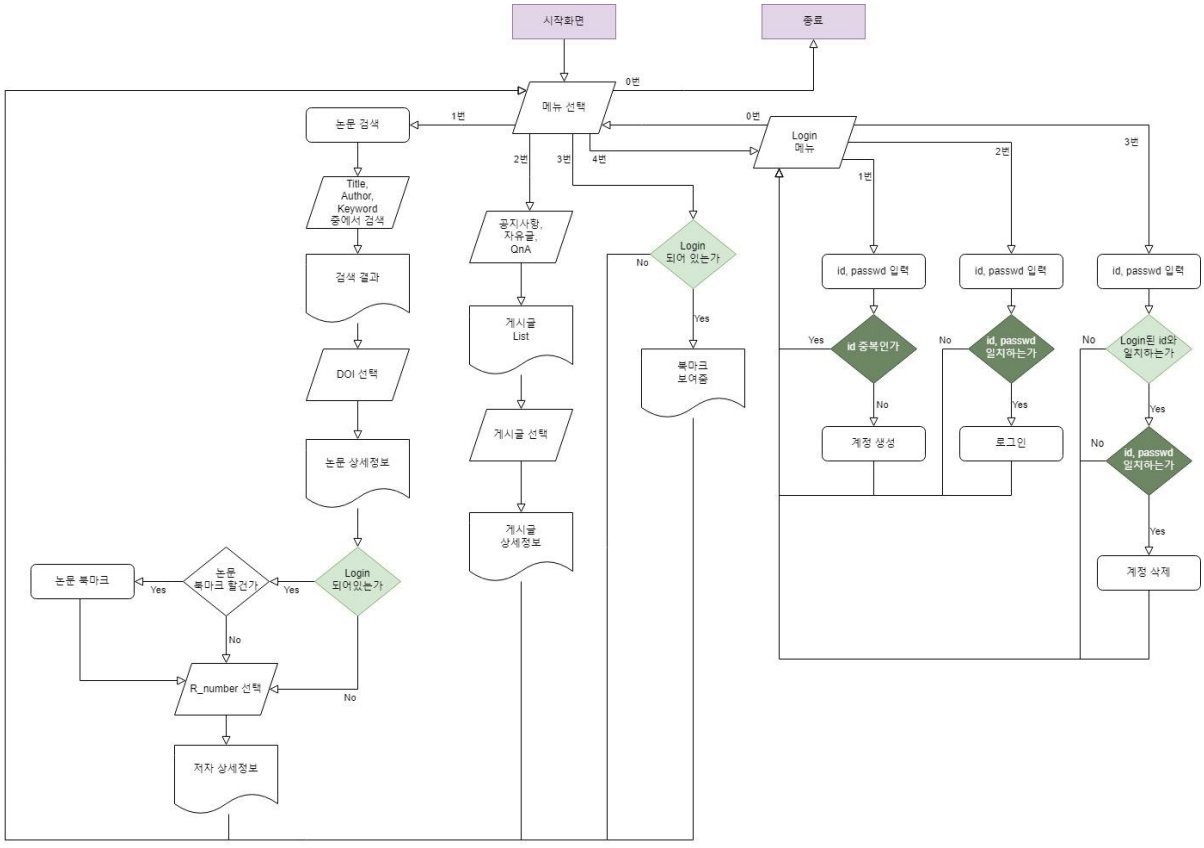
- 정상적으로 컨테이너가 운영되는 상황이라면 다음 화면과 같이 docker ps 명령어 수행 시 컨테이너가 작동함을 확인할 수 있다.

4. ZIP 파일에서 “KKK_Phase3” 폴더를 eclipse 작업환경에 올린 후 KKK_PHASE3.java 실행.



- 정상적인 프로그램 동작화면: 따로 설정할 필요가 없다.

2. 프로그램 순서도



[프로그램 시나리오]

--비로그인 시--

메인화면 메뉴

1.Search Paper, 2.Show Post, 4.Login, 0.Exit

0번 메뉴 선택 시

1) 프로그램을 종료한다.

1번 메뉴 선택 시

1) 1.Title, 2.Author, 3.Keyword 중에 검색하고 싶은 메뉴를 선택한다.

2) 선택한 메뉴에 맞게 검색할 '검색어'를 입력한다.

3) 입력한 '검색어'와 관련된 논문들의 정보가 출력된다.

4) 자세한 논문정보를 원하는 논문번호(DOI)를 입력한다.

5) 입력한 논문번호(DOI)와 일치하는 논문의 상세정보를 출력한다.

6) 관련 저자 중에서 자세한 저자정보를 원하는 저자번호(Rnum)을 입력한다.

7) 입력한 저자번호(Rnumber)와 일치하는 저자의 상세정보를 출력한다.

8) 메인화면 메뉴로 다시 돌아온다.

2번 메뉴 선택 시

1) 1.공지, 2.커뮤니티, 3.Q&A 중에 보고싶은 글의 카테고리를 선택한다.

2) 선택한 카테고리에 있는 글들의 목록을 출력한다.

3) 글의 내용을 보고싶은 글번호(P_NO)를 입력한다.

4) 입력한 글번호(P_NO)의 글 내용, 작성자, 댓글 등의 상세정보를 출력한다.

5) 메인화면 메뉴로 다시 돌아온다.

4번 Login 메뉴

1.Create Account, 2.Login Account, 0.Exit

1번 메뉴 선택 시

1) 생성할 ID와 passwd를 입력받는다.

2) 생성할 ID가 DB의 다른 ID와 중복된건 아닌지 확인한다.

3) 중복된 ID가 아닐 시 입력받는 ID와 passwd로 계정을 생성하여 DB에 저장한다.

4) Login 메뉴로 다시 돌아온다.

2번 메뉴 선택 시

1) 로그인할 ID와 passwd를 입력받는다.

2) ID가 DB에 존재하는지, passwd가 일치하는지 확인한다.

3) 전부 확인될 경우 그 ID로 로그인에 성공한다.

4) Login 메뉴로 다시 돌아온다.

0번 메뉴 선택 시

1) 메인화면 메뉴로 돌아간다.

--로그인 시--

메인화면 메뉴

Login as “account”

1.Search Paper, 2.Show Post, 3.Show BookMark 4.Login, 0.Exit

1번 메뉴 선택 시

1) 1.Title, 2.Author, 3.Keyword 중에 검색하고 싶은 메뉴를 선택한다.

2) 선택한 메뉴에 맞게 검색할 '검색어'를 입력한다.

3) 입력한 '검색어'와 관련된 논문들의 정보가 출력된다.

4) 자세한 논문정보를 원하는 논문번호(DOI)를 입력한다.

5) 입력한 논문번호(DOI)와 일치하는 논문의 상세정보를 출력한다.

6) 선택한 논문을 북마크에 저장할 것인지 물어본다. "BookMark (y/n)?"

7) y를 입력하면 북마크에 저장하고, n을 입력하면 넘어간다.

8) 관련 저자 중에서 자세한 저자정보를 원하는 저자번호(Rnum)을 입력한다.

9) 입력한 저자번호(Rnumber)와 일치하는 저자의 상세정보를 출력한다.

10) 메인화면 메뉴로 다시 돌아온다.

2번 메뉴 선택 시

1) 1.공지, 2.커뮤니티, 3.Q&A 중에 보고싶은 글의 카테고리를 선택한다.

2) 선택한 카테고리에 있는 글들의 목록을 출력한다.

3) 글의 내용을 보고싶은 글번호(P_NO)를 입력한다.

4) 입력한 글번호(P_NO)의 글 내용, 작성자, 댓글 등의 상세정보를 출력한다.

5) 메인화면 메뉴로 다시 돌아온다.

3번 메뉴 선택 시

1) 로그인한 계정의 북마크 정보를 보여준다.

2) 차례대로 Keyword, Author, Paper BookMark 정보를 출력한다.

3) 메인화면 메뉴로 다시 돌아온다.

4번 Login 메뉴

1.Create Account, 2.Login Account, 3.Delete Account, 0.Exit

1번 메뉴 선택 시

1) 생성할 ID와 passwd를 입력받는다.

2) 생성할 ID가 DB의 다른 ID와 중복된건 아닌지 확인한다.

3) 중복된 ID가 아닐 시 입력받는 ID와 passwd로 계정을 생성하여 DB에 저장한다.

4) Login 메뉴로 다시 돌아온다.

2번 메뉴 선택 시

1) 로그인할 ID와 passwd를 입력받는다.

2) ID가 DB에 존재하는지, passwd가 일치하는지 확인한다.

3) 전부 확인될 경우 입력했던 ID로 로그인에 성공한다.

4) Login 메뉴로 다시 돌아온다.

3번 메뉴 선택 시

1) 삭제할 ID와 passwd를 입력받는다.

2) 입력받은 ID가 현재 로그인하고 있는 ID와 동일한지 확인한다.

3-1) 동일하지 않다면 “삭제할 수 없다”는 알람을 띄우고 Login 메뉴로 돌아간다.

3-2) 동일하다면, passwd가 일치하는지 확인한다.

4) 일치한다면 계정을 DB에서 삭제하고 로그아웃 상태로 돌아간다.

5) Login 메뉴로 다시 돌아온다.

0번 메뉴 선택 시

1) 메인화면 메뉴로 돌아간다.

3. 프로그램 주요 기능

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

파라미터	타입	설명
type	Int	<1(논문명), 2(저자명), 3(키워드명)>을 입력 받음. 입력에 따라 각각에 해당하는 검색 쿼리문 생성.
name	String	검색 쿼리문에 들어가는 검색어를 입력 받음.

- TYPE(논문명, 저자명, 키워드명)에 따라 논문 검색한다.

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

파라미터	타입	설명
UserID	String	북마크 해 놓을 계정의 ID를 입력 받음.
DOI	Int	북마크 하고자 하는 논문의 DOI를 입력 받음.

- 특정 논문을 계정에 대해 북마크 한다.

- 북마크에 대한 중복 검사 시행

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

파라미터	타입	설명
DOI	Int	자세한 내용을 볼 문서에 대한 DOI를 입력으로 받음.

- 특정 논문에 대한 세부 내용을 보여준다.

-

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

파라미터	타입	설명
Rnum	Int	자세한 내용을 볼 저자에 대한 Rnum을 입력 받음.

- 특정 저자에 대한 세부 내용을 보여준다.

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

파라미터	타입	설명
kind	Int	보고자 하는 글에 대한 종류를 입력 받음. 글 종류: <1(공지사항), 2(자유글), 3(QnA)>

- KIND(공지사항, 자유글, QnA)에 따라 글 리스트를 보여준다.

- 게시물 제목 / 게시물 내용 / 게시물에 대한 COMMENT 등을 보여준다.

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

파라미터	타입	설명
PID	Int	자세한 내용을 볼 글에 대한 PID를 입력 받음.

- 특정 게시물에 대한 세부 내용을 보여준다.

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

파라미터	타입	설명
UserID	String	북마크를 보고자 하는 계정에 대한 아이디를 받음.

- 특정 계정에 대해 특정 논문을 북마크 한다.

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

파라미터	타입	설명
UserID	String	생성할 계정에 대한 아이디를 입력 받음.
UserPW	String	생성할 계정에 대한 비밀번호를 입력 받음.

- 계정에 대한 아이디 / 비밀번호를 받은 후 DB 상에 계정을 새로 생성한다.

- DB 내에 같은 아이디 / 비밀번호가 있는지 중복검사를 시행한다.

- 성공 시 “계정 생성 성공” 출력, 실패 시 “계정 생성 실패” 출력

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

파라미터	타입	설명
UserID	String	로그인할 계정에 대한 아이디를 입력 받음.
UserPW	String	로그인할 계정에 대한 비밀번호를 입력 받음.

- 계정에 대한 아이디 / 비밀번호를 받은 후 DB 상에 계정이 존재하는지 확인한다.

- 성공 시 “계정 접속 성공” 출력, 실패 시 “계정 접속 실패” 출력

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

파라미터	타입	설명
UserID	String	삭제할 계정에 대한 아이디를 입력 받음.
UserPW	String	삭제할 계정에 대한 비밀번호를 입력 받음.

- 계정에 대한 아이디 / 비밀번호를 받은 후 DB 상에 계정이 존재하는지 확인한다.

- 이미 로그인 된 계정과 입력한 계정의 아이디 / 비밀번호가 일치한지 확인한다.

- 성공 시 “계정 삭제 성공” 출력, 실패 시 “계정 삭제 실패” 출력

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 (수정된 쿼리 위주로)**

순번	변경 전	변경 후
#1	SELECT p.title FROM PAPER p WHERE p.title like '%Bitcoin%';	SELECT * FROM PAPER p WHERE p.title like '%' || ? || '%'
#2	SELECT p.* FROM PAPER p, AUTHOR a, WRITE w WHERE p.doi = w.doi AND a.r_number = w.r_number AND w.r_number = 12111;	SELECT p.* FROM PAPER p, AUTHOR a, WRITE w WHERE a.name LIKE '%' || ? || '%' AND p.doi = w.doi AND a.r_number = w.r_number
#3	SELECT k.sub FROM KEYWORD k, PAPER p, HAS h WHERE p.doi = 111 AND h.doi = p.doi AND h.k_id = k.k_id;	SELECT p.* FROM PAPER p, KEYWORD k, HAS h WHERE k.sub LIKE '%' || ? || '%' AND p.doi = h.doi AND k.k_id = h.k_id
#4	SELECT TO_CHAR(j.year, 'yyyy') AS YEAR FROM JOURNAL j, PAPER p WHERE p.doi = 6 AND j.j_number = p.j_number;	SELECT j.Publisher, j.Name, j.Vol, j.Issue, TO_CHAR(j.Year, 'yyyy') FROM JOURNAL j, PAPER p WHERE p.DOI = [DOI] AND p.J_number = j.J_number
#5	SELECT k.sub FROM KEYWORD k, PAPER p, HAS h WHERE p.doi = 111 AND h.doi = p.doi AND h.k_id = k.k_id;	SELECT k.sub FROM KEYWORD k, HAS h WHERE h.doi = [DOI] AND h.k_id = k.k_id
#6	SELECT k.sub, count(*) FROM JOURNAL j, PAPER p, HAS h, KEYWORD k WHERE j.ddc = 150 AND p.j_number = j.j_number AND h.doi = p.doi AND k.k_id = h.k_id GROUP BY (k.k_id, k.sub) ORDER BY COUNT(k.k_id) desc;	SELECT COUNT(*) FROM HAS h WHERE h.doi = [DOI]
#7	SELECT f.large, f.middle FROM FIELD f, JOURNAL j, PAPER p WHERE p.doi = 65 AND p.j_number = j.j_number AND j.ddc = f.ddc;	SELECT f.large, f.middle FROM FIELD f, JOURNAL j, PAPER p WHERE p.DOI = [DOI] AND p.J_number = j.J_number AND j.ddc = f.ddc
#8	SELECT COUNT(*) FROM REFERENCE r WHERE r.ed_doi = 622;	SELECT COUNT(*) FROM REFERENCE r WHERE r.ed_doi = [DOI]
#9	SELECT p.doi, p.title FROM PAPER p, REFERENCE r WHERE r.ing_doi = 1 AND p.doi = r.ed_doi ORDER BY p.title asc;	SELECT p.title FROM PAPER p, REFERENCE r WHERE r.ing_doi = [DOI] AND p.doi = r.ed_doi ORDER BY p.title ASC
#10	SELECT a.r_number, COUNT(*) FROM AUTHOR a, PAPER p, WRITE w WHERE a.r_number = w.r_number AND p.doi = w.doi GROUP BY a.r_number;	SELECT COUNT(*) FROM AUTHOR a, WRITE w WHERE a.r_number = [Rnum] AND a.r_number = w.r_number
#11	(SELECT p.title FROM PAPER p WHERE p.title LIKE '%Wireless%') INTERSECT (SELECT p.title FROM PAPER p, AUTHOR a, WRITE w WHERE a.name LIKE '%Alan%' AND p.doi = w.doi AND a.r_number = w.r_number) INTERSECT (SELECT p.title FROM PAPER p, KEYWORD k, HAS h WHERE k.sub LIKE '%Computer%' AND p.doi = h.doi AND k.k_id = h.k_id);	SELECT p.title FROM PAPER p, AUTHOR a, WRITE w WHERE p.doi = w.doi AND a.r_number = w.r_number AND w.r_number = '' + [Rnum]

#12	SELECT k.sub FROM KEYWORD k, K_BOOKMARK kb WHERE kb.id = 'test2' AND k.k_id = kb.k_id;	SELECT k.sub FROM KEYWORD k, K_BOOKMARK kb WHERE kb.id = \'[UserID]\' AND k.k_id = kb.k_id
		SELECT a.name FROM AUTHOR a, A_BOOKMARK ab WHERE ab.id = \'[UserID]\' AND a.r_number = ab.r_number
		SELECT p.title FROM PAPER p, P_BOOKMARK pb WHERE pb.id = \'[UserID]\' AND p.doi = pb.doi

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