Homework - SQL Programming

1. Use Google Colabortory and do your homework.
2. (In Google colaboratory) Before you submit your homework, restart kernel and run every cell!
3. Save (File->Save) the file
4. Submit your homework (this file) in Google classroom
5. **Don't forget to click "제출" button** ("Submit", "완료로 표시", 또는 "제출" 버턴을 누르지 않으면 제출된 것이 아님)
6. No late homeworks will be accepted for any reason!

To edit this cell, double click here

```
이름:이준용
학번:201904458
학과:컴퓨터전자시스템공학과
제출일:20211205
```

```
%load_ext sql
```

```
!pip install pymysql
```

```
Collecting pymysql
  Downloading PyMySQL-1.0.2-py3-none-any.whl (43 kB)
     |████████████████████████████████| 43 kB 757 kB/s
Installing collected packages: pymysql
Successfully installed pymysql-1.0.2
```

Use your host, id, password, and database

```
import getpass
user = 's201904458'
password = getpass.getpass()
host='dm.hufs.ac.kr'
database = 's201904458db'
connection_string = f'mysql+pymysql://{user}:{password}@{host}:3306/{database}'

%sql $connection_string
```

```
··········
'Connected: s201904458@s201904458db'
```

Warning: Your companydb state MUST be clean as initial state

- Refer to jupyter notebook on "SQL Lab on AWS RDS"

```
%%sql

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
```

```
DROP TABLE IF EXISTS DEPENDENT;
DROP TABLE IF EXISTS WORKS_ON;
DROP TABLE IF EXISTS PROJECT;
DROP TABLE IF EXISTS DEPT_LOCATIONS;
DROP TABLE IF EXISTS DEPARTMENT;
DROP TABLE IF EXISTS EMPLOYEE;

CREATE TABLE EMPLOYEE
(       Fname VARCHAR(15) NOT NULL,
        Minit CHAR,
        Lname VARCHAR(15) NOT NULL,
        Ssn CHAR(9) NOT NULL,
        Bdate DATE,
        Address VARCHAR(30),
        Sex CHAR,
        Salary DECIMAL(10, 2),
        Superssn CHAR(9),
        Dno INT,
        PRIMARY KEY (Ssn),
        FOREIGN KEY (Superssn)
                REFERENCES EMPLOYEE(Ssn)
                ON DELETE SET NULL
                ON UPDATE CASCADE,
        FOREIGN KEY (Dno)
                REFERENCES DEPARTMENT(Dnumber)
                ON DELETE SET NULL
                ON UPDATE CASCADE
) ENGINE=InnoDB;

CREATE TABLE DEPARTMENT
(       Dname VARCHAR(15) NOT NULL,
        Dnumber INT NOT NULL,
        Mgrssn CHAR(9),
        Mgrstartdate DATE,
        PRIMARY KEY (Dnumber),
        UNIQUE (Dname),
        FOREIGN KEY (Mgrssn)
                REFERENCES EMPLOYEE(Ssn)
                ON DELETE SET NULL
                ON UPDATE CASCADE
) ENGINE=InnoDB;


CREATE TABLE DEPT_LOCATIONS
(       Dnumber INT NOT NULL,
        Dlocation VARCHAR(15) NOT NULL,
        PRIMARY KEY (Dnumber, Dlocation),
        FOREIGN KEY (Dnumber)
                REFERENCES DEPARTMENT(Dnumber)
                ON DELETE CASCADE
                ON UPDATE CASCADE
) ENGINE=InnoDB;


CREATE TABLE PROJECT
(       Pname VARCHAR(15) NOT NULL,
        Pnumber INT NOT NULL,
        Plocation VARCHAR(15),
        Dnum INT,
        PRIMARY KEY (Pnumber),
        UNIQUE (Pname),
        FOREIGN KEY (Dnum)
                REFERENCES DEPARTMENT(Dnumber)
                ON DELETE SET NULL
                ON UPDATE CASCADE
) ENGINE=InnoDB;


CREATE TABLE WORKS_ON
(       Essn CHAR(9) NOT NULL,
        Pno INT NOT NULL,
        Hours DECIMAL(3, 1) ,
        PRIMARY KEY (Essn, Pno),
        FOREIGN KEY (Essn)
                REFERENCES EMPLOYEE(Ssn)
                ON DELETE CASCADE
                ON UPDATE CASCADE,
        FOREIGN KEY (Pno)
                REFERENCES PROJECT(Pnumber)
                ON DELETE CASCADE
                ON UPDATE CASCADE
) ENGINE=InnoDB;


CREATE TABLE DEPENDENT
(       Essn CHAR(9) NOT NULL,
        Dependent_name VARCHAR(15) NOT NULL,
        Sex CHAR,
        Bdate DATE,
        Relationship VARCHAR(8),
        PRIMARY KEY (Essn, Dependent_name),
        FOREIGN KEY (Essn)
                REFERENCES EMPLOYEE(Ssn)
                ON DELETE CASCADE
                ON UPDATE CASCADE
) ENGINE=InnoDB;


insert into EMPLOYEE values
        ("John", "B", "Smith", "123456789","1965-01-09", "731-Fondren-Houston-TX", "M", 30000, "333445555", 5),
        ("Franklin", "T", "Wong", "333445555","1955-12-08", "638-Voss-Houston-TX", "M", 40000, "888665555", 5),
        ("Alicia", "J", "Zelaya", "999887777", "1968-01-19", "3321-Castle-Spring-TX", "F", 25000, "987654321", 4),
        ("Jennifer", "S", "Wallace", "987654321", "1941-06-20", "291-Berry-Bellaire-TX", "F", 43000, "888665555", 4),
        ("Ramesh", "K", "Narayan", "666884444", "1962-09-15", "975-Fire-Oak-Humble-TX", "M", 38000, "333445555", 5),
        ("Joyce", "A", "English", "453453453", "1972-07-31", "5631-Rice-Houston-TX", "F", 25000, "333445555", 5),
        ("Ahmad", "V", "Jabbar", "987987987", "1969-03-29", "980-Dallas-Houston-TX", "M", 25000, "987654321", 4),
        ("James", "E", "Borg", "888665555", "1937-11-10", "450-Stone-Houston-TX", "M", 55000, NULL, 1);


insert into DEPENDENT  values
        (333445555, "Alice", "F","1986-04-05", "Daughter"),
        (333445555, "Theodore", "M","1983-10-25", "Son"),
        (333445555, "Joy", "F","1958-05-03", "Spouse"),
        (987654321, "Abner", "M","1942-02-28", "Spouse"),
        (123456789, "Michael", "M","1988-01-04", "Son"),
        (123456789, "Alice", "F", "1988-12-30", "Daughter"),
        (123456789, "Elizabeth", "F", "1967-05-05", "Spouse");


insert into DEPARTMENT values
        ("Research", 5, 333445555,"1988-05-22"),
        ("Administration", 4, 987654321,"1995-01-01"),
        ("Headquarters", 1, 888665555,"1981-06-19");

insert into DEPT_LOCATIONS values
        (1, "Houston"),
        (4, "Stafford"),
        (5, "Bellaire"),
        (5, "Sugarland"),
        (5, "Houston");


insert into PROJECT values
        ("ProductX", 1, "Bellaire", 5),
        ("ProductY", 2, "Sugarland", 5),
        ("ProductZ", 3, "Houston", 5),
        ("Computerization", 10, "Stafford", 4),
        ("Reorganization", 20, "Houston", 1),
        ("Newbenefits", 30, "Stafford", 4);


insert into WORKS_ON values
        (123456789, 1, 32.5),
        (123456789, 2, 7.5),
        (666884444, 3, 40.0),
        (453453453, 1, 20.0),
        (453453453, 2, 20.0),
        (333445555, 2, 10.0),
        (333445555, 3, 10.0),
        (333445555, 10, 10.0),
        (333445555, 20, 10.0),
        (999887777, 30, 30.0),
        (999887777, 10, 10.0),
        (987987987, 10, 35.0),
        (987987987, 30, 5.0),
        (987654321, 30, 20.0),
        (987654321, 20, 15.0),
        (888665555, 20, NULL);

SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
```

```
 * mysql+pymysql://s201904458:***@dm.hufs.ac.kr:3306/s201904458db
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
8 rows affected.
7 rows affected.
3 rows affected.
5 rows affected.
6 rows affected.
16 rows affected.
0 rows affected.
0 rows affected.
[]
```

1. ssn이 '888665555'인 직원과 그 모든 직간접 부하직원의 이름과 '888665555'로 부터 부하직원에 이르는 경로를 출력한다. (s학번db(companydb) DB 사용, mysql recursive CTE 참조, Googling) (20점)

```
+---------------------+-------------------------------------------------------+
| ename               | path                                                  |
+---------------------+-------------------------------------------------------+
| James E. Borg       | James E. Borg                                         |
| Franklin T. Wong    | James E. Borg > Franklin T. Wong                      |
| John B. Smith       | James E. Borg > Franklin T. Wong > John B. Smith      |
| Joyce A. English    | James E. Borg > Franklin T. Wong > Joyce A. English   |
| Ramesh K. Narayan   | James E. Borg > Franklin T. Wong > Ramesh K. Narayan  |
| Jennifer S. Wallace | James E. Borg > Jennifer S. Wallace                   |
| Ahmad V. Jabbar     | James E. Borg > Jennifer S. Wallace > Ahmad V. Jabbar |
| Alicia J. Zelaya    | James E. Borg > Jennifer S. Wallace > Alicia J. Zelaya |
+---------------------+-------------------------------------------------------+
```

```
%%sql

WITH RECURSIVE Emorder (ssn,ename,path) AS (
  SELECT e.Ssn As ssn,CONCAT(e.Fname,' ',e.Minit,'.',' ', e.Lname) AS ename, CAST(CONCAT(e.Fname,' ',e.Minit,'.',' ',
  FROM EMPLOYEE e
  WHERE e.Ssn='888665555'
  UNION ALL
  SELECT a.Ssn, CONCAT(a.Fname,' ',a.Minit,'.',' ', a.Lname), CONCAT(d.path,'>',CONCAT(a.Fname,' ',a.Minit,'.',' ', a
  FROM EMPLOYEE AS a JOIN Emorder AS d ON a.Superssn= d.Ssn
)
SELECT ename,path FROM Emorder;
```

```
 * mysql+pymysql://s201904458:***@dm.hufs.ac.kr:3306/s201904458db
8 rows affected.
```

ename	path
James E. Borg	James E. Borg
Franklin T. Wong	James E. Borg>Franklin T. Wong
Jennifer S. Wallace	James E. Borg>Jennifer S. Wallace
John B. Smith	James E. Borg>Franklin T. Wong>John B. Smith
Joyce A. English	James E. Borg>Franklin T. Wong>Joyce A. English
Ramesh K. Narayan	James E. Borg>Franklin T. Wong>Ramesh K. Narayan
Ahmad V. Jabbar	James E. Borg>Jennifer S. Wallace>Ahmad V. Jabbar
Alicia J. Zelaya	James E. Borg>Jennifer S. Wallace>Alicia J. Zelaya

temporary table 나 혼자만 계산 하기 위해서 쓰는 테이플 애플리케이션이 끝나면 사라짐 메인 메모리에만 상주, 디스크 까지 x recursion x

2. 입력으로 주어진 ssn의 모든 직간접 부하직원을 찾으려 한다. 이를 수행하는 Java JDBC 코드를 작성한다. (60점)

방법: Common table supervisees에 해당하는 temporary table 'tempssn' (using MEMORY engine)을 사용한다. tempssn에 루프의 전 단계에서 삽입된 ssn과 EMPLOYEE 테이블을 조인하여 다음 단계의 부하직원을 찾는다. 다음 단계의 직원이 한 명도 없다면 루프를 끝낸다. (**반드시 이 방법을 사용하여여 한다.**)

- temporary table (using memory engine)를 사용하지 않으면 0점
- SQL 문장에서 "insert into ... select ..."를 사용하면 0점
- recursion을 사용하면 0점
- truncate를 사용하면 0점
- code의 indentation이 맞지 않으면 감점

테스트하기전에 반드시 아래 데이터를 삽입한다.

```
insert into EMPLOYEE(fname, lname, ssn, superssn) values
    ('UUU', 'UUU', '000000001', '999887777'),
    ('VVV', 'VVV', '000000002', '000000001'),
    ('WWW', 'WWW', '000000003', '000000002'),
    ('XXX', 'XXX', '000000004', '000000003'),
    ('YYY', 'YYY', '000000005', '000000004'),
    ('ZZZ', 'ZZZ', '000000006', '000000005')
```

```
%%sql

insert into EMPLOYEE(fname, lname, ssn, superssn) values
    ('UUU', 'UUU', '000000001', '999887777'),
    ('VVV', 'VVV', '000000002', '000000001'),
    ('WWW', 'WWW', '000000003', '000000002'),
    ('XXX', 'XXX', '000000004', '000000003'),
    ('YYY', 'YYY', '000000005', '000000004'),
    ('ZZZ', 'ZZZ', '000000006', '000000005')
```

```
 * mysql+pymysql://s201904458:***@dm.hufs.ac.kr:3306/s201904458db
6 rows affected.
[]
```

제출문:

1. 자바 코드
2. 아래의 각 테스트 케이스에 해당하는 이클립스 **전체** 스크린샷(ALT+PrintScreen).

CHEATING 발견시 모든 제출자의 점수를 "-만점"으로 처리함. (예 20점 만점이면 -20점)

Your Java Code

- To edit this cell, double click here
- `java ...`부분을 제거하지 말 것!!!

```java //copy&paste your java code here import java.sql.Connection; import java.sql.DriverManager; import java.sql.PreparedStatement; import java.sql.ResultSet; import java.sql.SQLException; import java.util.ArrayList; import java.util.HashSet; import java.util.Scanner; import java.sql.*; import javax.swing.JOptionPane; import javax.swing.JPasswordField; import java.io.*;*

public class problem2 { // problem2클래스에 위에 import한 java class들을 private으로 명명 private static Scanner scn;
private static Connection con; private static PreparedStatement psmt; private static PreparedStatement psmt1; private static ResultSet rset; private static ResultSet rset1; public static void main(String args[]) { scn = new Scanner(System.in); //main 함수에서 problem_2()함수 돌림 problem_2(); scn.close(); // 함수 main에서 실행 후 닫음 메인메모리에서 데이터 누수 방지 }

private static String readEntry(String prompt) { try { StringBuffer buffer = new StringBuffer(); System.out.print(prompt); System.out.flush(); int c = System.in.read(); while (c != '\n' && c != -1) { buffer.append((char)c); c = System.in.read(); } return buffer.toString().trim(); } catch (IOException e) { return ""; } } private static String getPassword() { final String password, message = "Enter password"; if(System.console() == null) { final JPasswordField pf = new JPasswordField(); password = JOptionPane.showConfirmDialog(null, pf,
```

message, JOptionPane.OK_CANCEL_OPTION, JOptionPane.QUESTION_MESSAGE) == JOptionPane.OK_OPTION ? new String(pf.getPassword()) : ""; } else password = new String(System.console().readPassword("%s> ", message));

```
return password;
}
```

private static void problem_2() { try { Class.forName("com.mysql.cj.jdbc.Driver"); String host = "dm.hufs.ac.kr:3306/"; String db= "s201904458db"; String user = "s201904458"; String password = "01053847822";//getPassword(); // String password = "01053847822"; Connection con = DriverManager.getConnection("jdbc:mysql://" + host + db + "?useSSL=false&serverTimezone=Asia/Seoul", user, password); String drp = "DROP TEMPORARY TABLE IF EXISTS tempssn;"; psmt = con.prepareStatement(drp); psmt.executeUpdate(); String query = " CREATE TEMPORARY TABLE tempssn(" // tempssn temporary table 만들어서 + " essn char(9)" + ")ENGINE=MEMORY;"; psmt = con.prepareStatement(query); int ret = psmt.executeUpdate(); // System.out.println(ret); // tempssn 안에 들어있는 튜플 개수 출력 String inst="INSERT INTO tempssn values (?)"; // tempssn 갱신 psmt = con.prepareStatement(inst); System.out.print("Enter a ssn: "); // 처음 입력 받기 String ssn = scn.next(); // TEST CASE : 888665555 , 453453453 , 999887777, 333445555 psmt.setString(1,ssn); ret = psmt.executeUpdate(); // System.out.println(ret); // 입력 받은후 tempssn update되고 난후 튜플개수 출력 for(int i=1;i<9; i++) { String rst = "SELECT essn FROM tempssn;"; psmt = con.prepareStatement(rst); rset = psmt.executeQuery(); //rset에는 tempssn에 들어있는 튜플이 리턴됨 int cnt =0; while(rset.next()) { cnt = rset.getInt(1); // System.out.println("cnt "+cnt); String query1 = "SELECT a.Ssn" + " FROM EMPLOYEE a join tempssn ON a.Superssn=?"; psmt1 = con.prepareStatement(query1); psmt1.setString(1,rset.getString(1)); rset1 = psmt1.executeQuery();
int num =0; while(rset1.next()) {//rset1이 있을 경우 tempssn에 튜플이 있을경우 num = rset1.getInt(1); // System.out.println("num "+num); if(num>0) { System.out.println(rset1.getString(1) + " at level "+ i); }else { } psmt = con.prepareStatement(drp); psmt.executeUpdate(); psmt = con.prepareStatement(query); psmt.executeUpdate(); String ins = "INSERT into tempssn values (?)"; psmt = con.prepareStatement(ins); psmt.setString(1, rset.getString(1)); psmt.executeUpdate(); String upd= "UPDATE tempssn SET essn=? WHERE essn=?"; psmt = con.prepareStatement(upd); psmt.setString(1,rset1.getString(1)); psmt.setString(2, rset.getString(1)); psmt.executeUpdate(); } } } System.out.println("END OF LIST"); con.close(); rset.close(); } catch (SQLException ex) { System.out.println("SQLException" + ex); } catch (Exception ex) { System.out.println("Exception:" + ex); } } }

Test Case SELF-CHECK

- 아래 CELL을 수정하여 성공한 경우,

  Test Case 1: SUCCESS

- 실패한 경우,

  Test Case 1: FAIL

  로 표시한다.

- 다른 Testcase에 대해서도 같은 방식으로 표기한다.

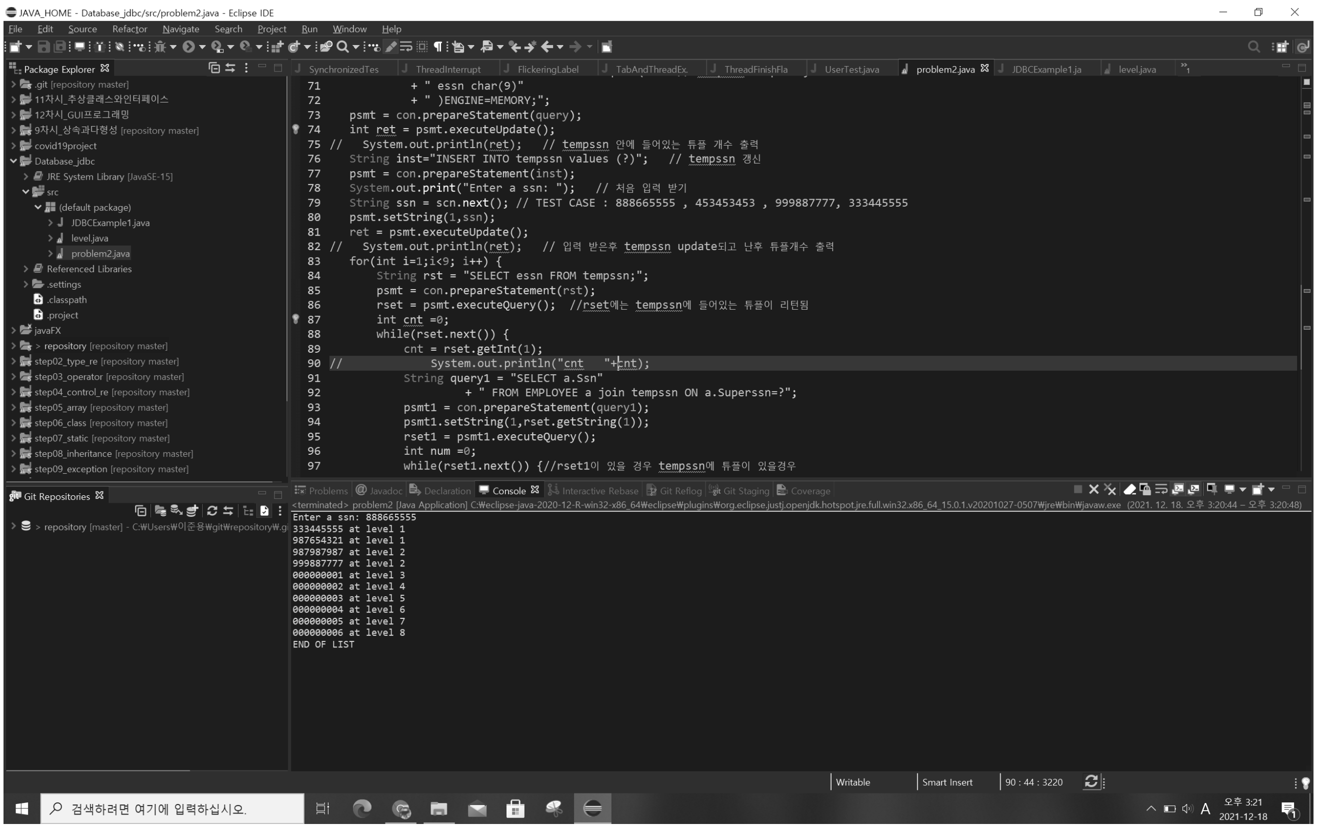
Test Case 1: FAIL

```
Enter a ssn: 888665555
333445555 at level 1
987654321 at level 1
```

```
123456789 at level 2
453453453 at level 2
666884444 at level 2
987987987 at level 2
999887777 at level 2
000000001 at level 3
000000002 at level 4
000000003 at level 5
000000004 at level 6
000000005 at level 7
000000006 at level 8
END OF LIST
```

더블클릭 또는 Enter 키를 눌러 수정

- To edit this cell, double click here
- copy&paste your eclipse screenshot here



```
            + " essn char(9)"
            + " )ENGINE=MEMORY;";
  psmt = con.prepareStatement(query);
  int ret = psmt.executeUpdate();
//   System.out.println(ret);   // tempssn 안에 들어있는 튜플 개수 출력
  String inst="INSERT INTO tempssn values (?)";   // tempssn 갱신
  psmt = con.prepareStatement(inst);
  System.out.print("Enter a ssn: ");   // 처음 입력 받기
  String ssn = scn.next(); // TEST CASE : 888665555 , 453453453 , 999887777, 333445555
  psmt.setString(1,ssn);
  ret = psmt.executeUpdate();
//   System.out.println(ret);   // 입력 받은후 tempssn update되고 난후 튜플개수 출력
  for(int i=1;i<9; i++) {
      String rst = "SELECT essn FROM tempssn;";
      psmt = con.prepareStatement(rst);
      rset = psmt.executeQuery();  //rset에는 tempssn에 들어있는 튜플이 리턴됨
      int cnt =0;
      while(rset.next()) {
          cnt = rset.getInt(1);
//              System.out.println("cnt   "+cnt);
          String query1 = "SELECT a.Ssn"
                  + " FROM EMPLOYEE a join tempssn ON a.Superssn=?";
          psmt1 = con.prepareStatement(query1);
          psmt1.setString(1,rset.getString(1));
          rset1 = psmt1.executeQuery();
          int num =0;
          while(rset1.next()) {//rset1이 있을 경우 tempssn에 튜플이 있을경우
```

```
<terminated> problem2 [Java Application] C:₩eclipse-java-2020-12-R-win32-x86_64₩eclipse₩plugins₩org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_15.0.1.v20201027-0507₩jre₩bin₩javaw.exe (2021. 12. 18. 오후 3:20:44 – 오후 3:20:48)
Enter a ssn: 888665555
333445555 at level 1
987654321 at level 1
987987987 at level 2
999887777 at level 2
000000001 at level 3
000000002 at level 4
000000003 at level 5
000000004 at level 6
000000005 at level 7
000000006 at level 8
END OF LIST
```

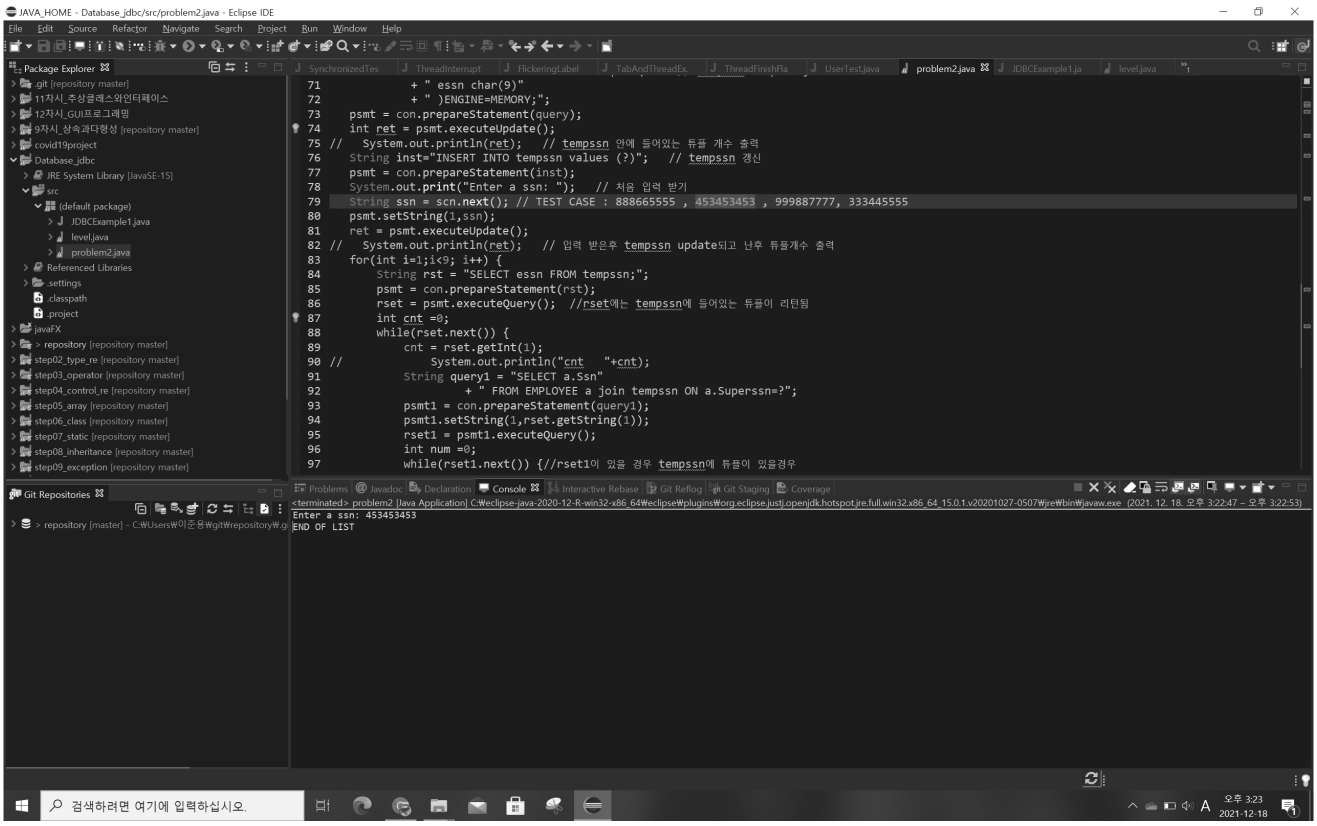
Test Case 2: SUCCESS

1. 항목 추가
2. 항목 추가

```
Enter a ssn: 453453453
END OF LIST
```

- To edit this cell, double click here

- copy&paste your eclipse screenshot here


JAVA_HOME - Database_jdbc/src/problem2.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
problem2.java
Problems Javadoc Declaration Console Interactive Rebase Git Reflog Git Staging Coverage
Git Repositories

```
                + " essn char(9)"
                + " )ENGINE=MEMORY;";
    psmt = con.prepareStatement(query);
    int ret = psmt.executeUpdate();
//   System.out.println(ret);   // tempssn 안에 들어있는 튜플 개수 출력
    String inst="INSERT INTO tempssn values (?)";   // tempssn 갱신
    psmt = con.prepareStatement(inst);
    System.out.print("Enter a ssn: ");   // 처음 입력 받기
    String ssn = scn.next(); // TEST CASE : 888665555 , 453453453 , 999887777, 333445555
    psmt.setString(1,ssn);
    ret = psmt.executeUpdate();
//   System.out.println(ret);   // 입력 받은후 tempssn update되고 난후 튜플개수 출력
    for(int i=1;i<9; i++) {
        String rst = "SELECT essn FROM tempssn;";
        psmt = con.prepareStatement(rst);
        rset = psmt.executeQuery();  //rset에는 tempssn에 들어있는 튜플이 리턴됨
        int cnt =0;
        while(rset.next()) {
            cnt = rset.getInt(1);
//              System.out.println("cnt   "+cnt);
            String query1 = "SELECT a.Ssn"
                    + " FROM EMPLOYEE a join tempssn ON a.Superssn=?";
            psmt1 = con.prepareStatement(query1);
            psmt1.setString(1,rset.getString(1));
            rset1 = psmt1.executeQuery();
            int num =0;
            while(rset1.next()) {//rset1이 있을 경우 tempssn에 튜플이 있을경우
```

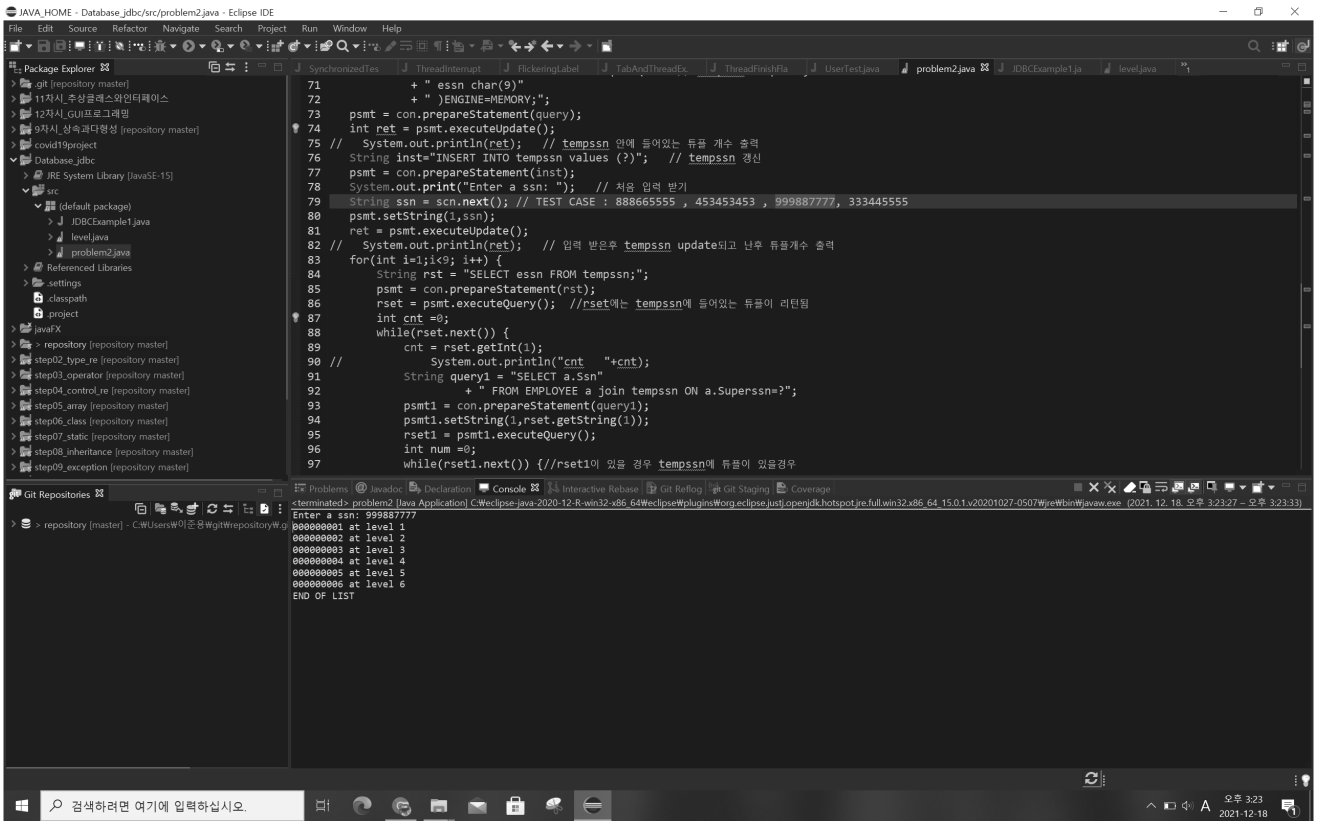
```
<terminated> problem2 [Java Application] C:₩eclipse-java-2020-12-R-win32-x86_64₩eclipse₩plugins₩org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_15.0.1.v20201027-0507₩jre₩bin₩javaw.exe  (2021. 12. 18. 오후 3:22:47 – 오후 3:22:53)
Enter a ssn: 453453453
END OF LIST
```

Test Case 3: SUCCESS

```
Enter a ssn: 999887777
000000001 at level 1
000000002 at level 2
000000003 at level 3
000000004 at level 4
000000005 at level 5
000000006 at level 6
END OF LIST
```

- To edit this cell, double click here

- copy&paste your eclipse screenshot here



```
            + " essn char(9)"
            + " )ENGINE=MEMORY;";
    psmt = con.prepareStatement(query);
    int ret = psmt.executeUpdate();
//  System.out.println(ret);   // tempssn 안에 들어있는 튜플 개수 출력
    String inst="INSERT INTO tempssn values (?)";   // tempssn 갱신
    psmt = con.prepareStatement(inst);
    System.out.print("Enter a ssn: ");   // 처음 입력 받기
    String ssn = scn.next(); // TEST CASE : 888665555 , 453453453 , 999887777, 333445555
    psmt.setString(1,ssn);
    ret = psmt.executeUpdate();
//  System.out.println(ret);   // 입력 받은후 tempssn update되고 난후 튜플개수 출력
    for(int i=1;i<9; i++) {
        String rst = "SELECT essn FROM tempssn;";
        psmt = con.prepareStatement(rst);
        rset = psmt.executeQuery();  //rset에는 tempssn에 들어있는 튜플이 리턴됨
        int cnt =0;
        while(rset.next()) {
            cnt = rset.getInt(1);
//              System.out.println("cnt   "+cnt);
            String query1 = "SELECT a.Ssn"
                    + " FROM EMPLOYEE a join tempssn ON a.Superssn=?";
            psmt1 = con.prepareStatement(query1);
            psmt1.setString(1,rset.getString(1));
            rset1 = psmt1.executeQuery();
            int num =0;
            while(rset1.next()) {//rset1이 있을 경우 tempssn에 튜플이 있을경우
```

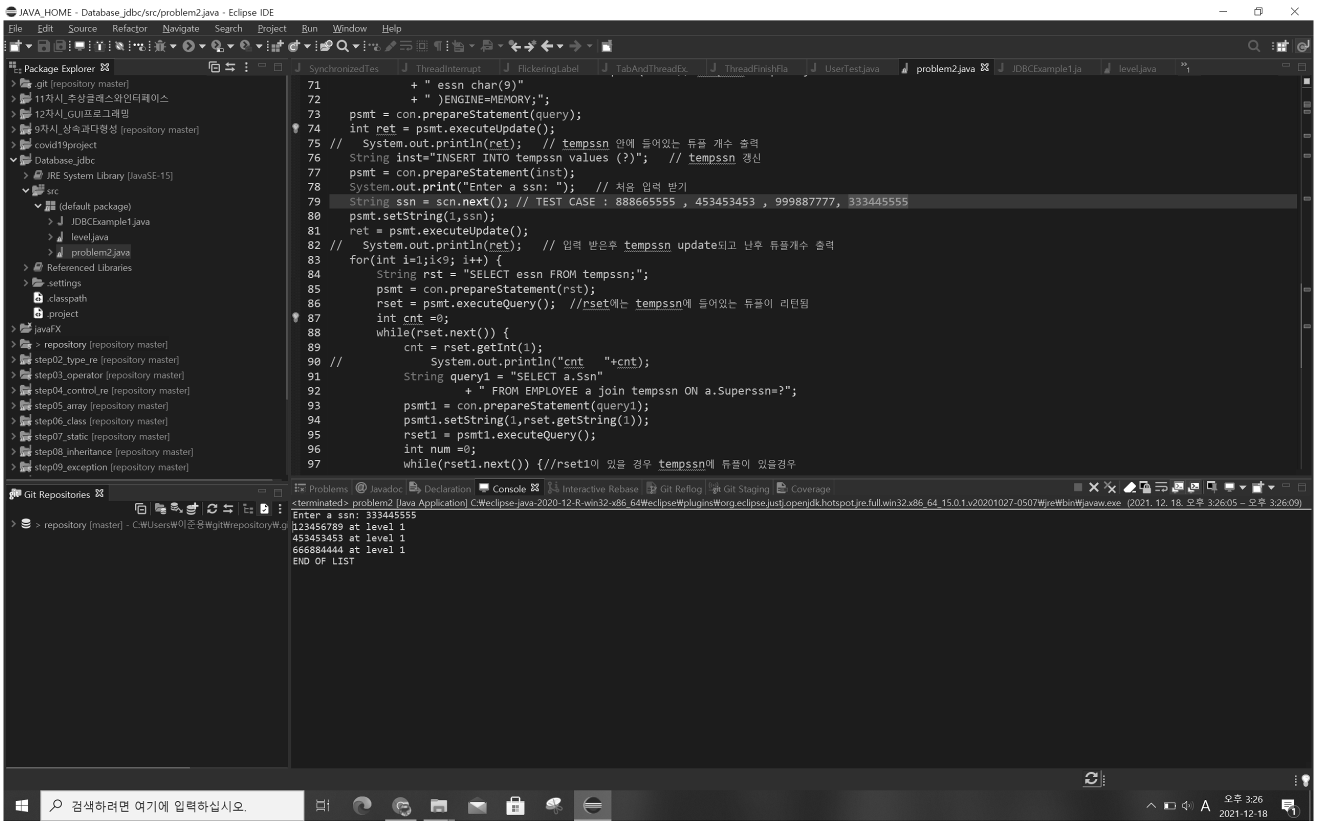
```
<terminated> problem2 [Java Application] C:₩eclipse-java-2020-12-R-win32-x86_64₩eclipse₩plugins₩org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_15.0.1.v20201027-0507₩jre₩bin₩javaw.exe (2021. 12. 18. 오후 3:23:27 – 오후 3:23:33)
Enter a ssn: 999887777
000000001 at level 1
000000002 at level 2
000000003 at level 3
000000004 at level 4
000000005 at level 5
000000006 at level 6
END OF LIST
```

Test Case 4: SUCCESS

```
Enter a ssn: 333445555
123456789 at level 1
453453453 at level 1
666884444 at level 1
END OF LIST
```

- To edit this cell, double click here

- copy&paste your eclipse screenshot here



```
            + " essn char(9)"
            + " )ENGINE=MEMORY;";
    psmt = con.prepareStatement(query);
    int ret = psmt.executeUpdate();
//   System.out.println(ret);   // tempssn 안에 들어있는 튜플 개수 출력
    String inst="INSERT INTO tempssn values (?)";   // tempssn 갱신
    psmt = con.prepareStatement(inst);
    System.out.print("Enter a ssn: ");   // 처음 입력 받기
    String ssn = scn.next(); // TEST CASE : 888665555 , 453453453 , 999887777, 333445555
    psmt.setString(1,ssn);
    ret = psmt.executeUpdate();
//   System.out.println(ret);   // 입력 받은후 tempssn update되고 난후 튜플개수 출력
    for(int i=1;i<9; i++) {
        String rst = "SELECT essn FROM tempssn;";
        psmt = con.prepareStatement(rst);
        rset = psmt.executeQuery();  //rset에는 tempssn에 들어있는 튜플이 리턴됨
        int cnt =0;
        while(rset.next()) {
            cnt = rset.getInt(1);
//              System.out.println("cnt   "+cnt);
            String query1 = "SELECT a.Ssn"
                    + " FROM EMPLOYEE a join tempssn ON a.Superssn=?";
            psmt1 = con.prepareStatement(query1);
            psmt1.setString(1,rset.getString(1));
            rset1 = psmt1.executeQuery();
            int num =0;
            while(rset1.next()) {//rset1이 있을 경우 tempssn에 튜플이 있을경우
```

```
<terminated> problem2 [Java Application] C:₩eclipse-java-2020-12-R-win32-x86_64₩eclipse₩plugins₩org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_15.0.1.v20201027-0507₩jre₩bin₩javaw.exe (2021. 12. 18. 오후 3:26:05 – 오후 3:26:09)
Enter a ssn: 333445555
123456789 at level 1
453453453 at level 1
666884444 at level 1
END OF LIST
```

✓ 0초 오전 11:58에 완료됨