Homework - SQL2

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: 20211203
```

- 권고사항: 모든 SQL select 문에 tuple variable를 사용할 것.
- SQL 작성시, 문제에 주어지지 않은 상수를 사용하는 경우 (cheating), **마이너스 점수**를 받는다.
- **점수: 각 10점, 18번 20점, 총200점**

```
%load_ext sql
```

```
The sql extension is already loaded. To reload it, use:
  %reload_ext sql
```

```
!pip install pymysql
```

```
Requirement already satisfied: pymysql in /usr/local/lib/python3.7/dist-packages (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"

When you list the name of employees, the name must be the following format:

- Output Schema and tuple format:

```
+------------------+
| Employee name    |
+------------------+
| John B. Smith    |
...
+------------------+
```

- Use MySQL concat 함수 사용, Refer to MySQL manual or Googling

```
%%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.
[]
```

1-13번에 대해 SQL select 문을 작성하고 MySQL에서 실행한 결과물을 제출하시오.

1. Retrieve the names of employees in department 5 who work more than 10 hours per week on the 'ProductX' project. (single SELECT 사용, MySQL concat 함수 사용)

```
%%sql

SELECT CONCAT(e.Fname,' ',e.Minit,'.',' ', e.Lname) AS 'Employee name'
FROM EMPLOYEE e, PROJECT p, WORKS_ON w
WHERE e.Dno = 5 AND e.Ssn = w.Essn AND p.Pnumber = w.Pno AND w.Hours >10;
```

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

Employee name
John B. Smith
Joyce A. English
Joyce A. English
Ramesh K. Narayan

2. List the names of employees who have a dependent with the same sex as themselves. (EXISTS 사용)

```
%%sql

SELECT CONCAT(e.Fname,' ',e.Minit,'.',' ', e.Lname) AS 'Employee name'
FROM EMPLOYEE e
WHERE EXISTS (
 SELECT *
 FROM DEPENDENT d
 WHERE e.Sex=d.Sex AND e.Ssn=d.Essn );
```

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

Employee name
John B. Smith
Franklin T. Wong

3. Find the names of employees that are directly supervised by 'Franklin Wong'. (EXISTS 사용)

```
%%sql

SELECT CONCAT(e.Fname,' ',e.Minit,'.',' ', e.Lname) AS 'Employee name'
FROM EMPLOYEE e
WHERE EXISTS(
  SELECT *
  FROM EMPLOYEE m
  WHERE m.ssn = e.Superssn AND m.Fname='Franklin' AND m.Lname='Wong'
);
```

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

Employee name
John B. Smith
Joyce A. English
Ramesh K. Narayan

4. For each project, list the project name and the total hours per week (by all employees) spent on that project.

Output Schema:

```
+-----------------+-------------+
| Project name    | Total hours |
+-----------------+-------------+
```

```
%%sql

SELECT p.Pname AS 'Project name', SUM(wo.Hours) AS 'Total hours'
FROM PROJECT p, WORKS_ON wo
WHERE p.Pnumber = wo.Pno
GROUP BY p.Pname;
```

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

Project name	Total hours
Computerization	55.0
Newbenefits	55.0
ProductX	52.5
ProductY	37.5
ProductZ	50.0
Reorganization	25.0

5. Retrieve the names of employees who work on every project managed by 'Administration' department.

```
%%sql
```

```
select CONCAT(e.Fname,' ',e.Minit,'.',' ', e.Lname) AS 'Employee name'
from EMPLOYEE e
where not exists (
  select *
  from PROJECT p, DEPARTMENT d
  where d.Dnumber=p.Dnum and d.Dname='Administration' and
  not exists (
    select *
    from WORKS_ON w
    where e.ssn = w.essn and p.pnumber = w.pno
  )
)
```

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

Employee name
Ahmad V. Jabbar
Alicia J. Zelaya

6. Retrieve the names of employees who do not work on any project which is located in 'Houston'. (NOT EXISTS 사용)

```
%%sql

select concat(e.Fname,' ',e.Minit,'.',' ', e.Lname) as 'Employee name'
from EMPLOYEE e
where not exists (
  select *
  from PROJECT p
  where p.Plocation='Houston' and
  exists (
    select *
    from WORKS_ON w
    where e.ssn = w.essn and p.pnumber = w.pno
  )
);
```

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

Employee name
John B. Smith
Joyce A. English
Ahmad V. Jabbar
Alicia J. Zelaya

7. For each department, retrieve the department name, and the average salary of employees working in that department.(소수점 이하 2자리까지만(버림) 출력, Refer to MySQL manual or Googling)

Output schema

```
+-----------------+----------------+
| Department name | Average salary |
+-----------------+----------------+
```

```
%%sql

select concat(d.Dname) as 'Department name' , concat(truncate(avg(salary),2)) as 'Average salary'
from DEPARTMENT d, EMPLOYEE e
where d.Dnumber=e.Dno
group by d.Dname;
```

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

Department name	Average salary
Research	33250.00
Headquarters	55000.00
Administration	31000.00

8. Retrieve the average salary of all female employees. (소수점 이하 2자리까지만(버림) 출력, Refer to MySQL manual or Googling)

Output Schema:

```
+----------------+
| Average salary |
+----------------+
```

```
%%sql

select concat(truncate(avg(salary),2)) as 'Average salary'
from EMPLOYEE e
where e.sex='F';
```

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

Average salary
31000.00

9. Find the names and addresses of employees who work on at least one project located in Houston but whose department has no location in Houston. (EXISTS, NOT EXISTS 사용)

Output schema:

```
+--------------------+-------------------------+
| Employee name      | Address                 |
+--------------------+-------------------------+
```

```
%%sql

select concat(e.Fname,' ',e.Minit,'.',' ', e.Lname) AS 'Employee name',concat(e.Address) as 'Address'
from EMPLOYEE e
where exists (
  select *
  from WORKS_ON w, PROJECT p
  where w.Pno=p.Pnumber and p.Plocation='Houston' and e.ssn=w.Essn
) and not exists (
```

```
  select *
  from DEPT_LOCATIONS de
  where e.Dno=de.Dnumber and de.Dlocation='Houston'
);
```

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

Employee name	Address
Jennifer S. Wallace	291-Berry-Bellaire-TX

10. List the names of department managers who have no dependents. (EXISTS, NOT EXISTS 사용)

```
%%sql

select concat(e.Fname,' ',e.Minit,'.',' ', e.Lname) AS 'Employee name'
from EMPLOYEE e
where exists (
  select *
  from DEPARTMENT d
  where e.Ssn=d.Mgrssn
) and not exists (
  select *
  from DEPENDENT de
  where e.Ssn=de.Essn
)
```

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

Employee name
James E. Borg

11. Retrieve the names of all employees who work in the department that has the employee with the highest salary among all employees.

```
%%sql

select concat(e.Fname,' ',e.Minit,'.',' ', e.Lname) AS 'Employee name'
from EMPLOYEE e
where e.Dno = (
  select em.Dno
    from EMPLOYEE em
    where em.Salary = (select MAX(Salary)from EMPLOYEE))
```

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

Employee name
James E. Borg

12-1. Retrieve the names of all employees whose supervisor's supervisor is James Borg. (EXISTS 안에 EXISTS 사용)

```
%%sql
```

```
select concat(e.Fname,' ',e.Minit,'.',' ', e.Lname) AS 'Employee name'
from EMPLOYEE e
where exists (
    select *
    from EMPLOYEE j
    where e.superssn = j.ssn and j.lname = 'Borg' and j.fname = 'James'
)
```

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

Employee name
Franklin T. Wong
Jennifer S. Wallace

12-2. Retrieve the names of all employees whose supervisor's supervisor is James Borg. (single SELECT 사용)

```
%%sql

select concat(e.Fname,' ',e.Minit,'.',' ', e.Lname) AS 'Employee name'
from EMPLOYEE e, EMPLOYEE j
where j.lname = 'Borg' and j.fname = 'James' and j.ssn = e.superssn

union all

select  concat(k.Fname,' ',k.Minit,'.',' ', k.Lname) AS 'Employee name'
from EMPLOYEE e, EMPLOYEE j, EMPLOYEE k
where e.lname = 'Borg' and e.fname = 'James' and e.ssn = j.superssn and j.ssn = k.superssn;
```

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

Employee name
Franklin T. Wong
Jennifer S. Wallace
John B. Smith
Joyce A. English
Ramesh K. Narayan
Ahmad V. Jabbar
Alicia J. Zelaya

13. Retrieve the names of employees who make at least 10,000 dollars more than the employee who is paid the least in the company.

```
%%sql

select concat(e.Fname,' ',e.Minit,'.',' ', e.Lname) AS 'Employee name'
from EMPLOYEE e
where e.salary >=all (select (min(salary)+10000) from EMPLOYEE);
```

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

Employee name

14-17번에 대해 SQL view문을 작성하고 MySQL에서 “select * from ”을 실행한 결과물을 제출하시오.

14. A view that has the department name, manager name, and manager salary for every department

```
%%sql
DROP view IF EXISTS DEPT_VIEW;

create view DEPT_VIEW
as select d.Dname,concat(e.Fname,' ',e.Minit,'.',' ', e.Lname) AS 'Employee name', e.Salary
   from DEPARTMENT d, EMPLOYEE e
   where d.Mgrssn = e.Ssn
```

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

```
%sql select * from DEPT_VIEW;
```

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

Dname	Employee name	Salary
Headquarters	James E. Borg	55000.00
Administration	Jennifer S. Wallace	43000.00
Research	Franklin T. Wong	40000.00

15. A view that has the employee name, supervisor name, and employee salary for each employee who works in the ‘Research’ department

```
%%sql

DROP view IF EXISTS RESEARCH_DEPT_VIEW;

create view RESEARCH_DEPT_VIEW  as
select concat(e.Fname,' ',e.Minit,'.',' ', e.Lname) AS 'employee name',concat(em.Fname,' ',em.Minit,'.',' ', em.Lname
from EMPLOYEE e, EMPLOYEE em
where e.Superssn = em.Ssn
```

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

```
%sql select * from RESEARCH_DEPT_VIEW;
```

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

employee name	supervisor name	employee salary
John B. Smith	Franklin T. Wong	30000.00
Franklin T. Wong	James E. Borg	40000.00
Joyce A. English	Franklin T. Wong	25000.00

16. A view that has the project name, controlling department name, number of employees, and total hours worked per week on the project for each project

```
%%sql
DROP view IF EXISTS PROJECT_VIEW;

create view PROJECT_VIEW as
select p.pname as 'project_name', d.dname as 'department_name', count(*) as 'numOfEmployees', avg(wo.Hours) as 'hours
from EMPLOYEE e, DEPARTMENT d, WORKS_ON wo, PROJECT p
where e.dno = d.dnumber and wo.pno=p.pnumber and wo.essn=e.ssn and p.dnum = d.dnumber
group by p.pname
```

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

```
%sql select * from PROJECT_VIEW;
```

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

project_name	department_name	numOfEmployees	hoursWorkedPerWeek
Computerization	Administration	2	22.50000
Newbenefits	Administration	3	18.33333
Reorganization	Headquarters	1	None
ProductX	Research	2	26.25000
ProductY	Research	3	12.50000
ProductZ	Research	2	25.00000

17. A view that has the project name, controlling department name, number of employees, and total hours worked per week on the project for each project with more than two employees working on it

```
%%sql
DROP view IF EXISTS PROJECT_VIEW_GT2;

create view PROJECT_VIEW_GT2 as
select p.Pname as 'project_name', d.Dname as 'department_name', count(*) as 'numOfEmployees', avg(wo.Hours) as 'hours
from EMPLOYEE e, DEPARTMENT d, WORKS_ON wo, PROJECT p
where e.Dno = d.Dnumber and wo.Pno=p.Pnumber and wo.Essn=e.Ssn and p.Dnum = d.Dnumber
group by p.Pname
having count(*) >= 2
```

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

```
%sql select * from PROJECT_VIEW_GT2;
```

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

project_name	department_name	numOfEmployees	hoursWorkedPerWeek
Computerization	Administration	2	22.50000
Newbenefits	Administration	3	18.33333
ProductX	Research	2	26.25000
ProductY	Research	3	12.50000
ProductZ	Research	2	25.00000

18. EMPLOYEE 테이블의 salary의 변경사항이 있을 때 마다, 변경사항을 기록하는 테이블 salary_audit(ssn, before_salary, after_salary, u_datetime)을 만들고, update trigger를 사용하여 변경사항을 기록하시오. 5번 부서에 속하는 직원의 salary를 100% 상향 조정 후, “select * from salary_audit”를 통해 결과를 확인하시오. udatetime에는 수정이 실행된 DATETIME(SQL Datatype, 날짜와 시각)을 기록한다. 현재 시각값은 now() 함수를 통해 얻을 수 있다.

Warning

- 18번 문제 Update 한 후에 1-17번 문제를 다시 풀면 다른 결과가 나올 수 있음
- 1-17번 문제는 반드시 companydb의 초기 상태에서 풀어야 함

```
%%sql
drop table if exists salary_audit;

create table salary_audit (
  essn char(9),
  before_salary decimal(10,2),
  after_salary decimal(10,2),
  udatetime datetime
);
```

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

```
%%sql

drop trigger if exists salary_audit_trig;

create trigger salary_audit_trig
after update
on EMPLOYEE
for each row
begin
if new.salary <> old.salary then
    insert into salary_audit values (new.ssn, old.salary, new.salary, now());
end if;
end
```

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

```
%sql select * from salary_audit;
```

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

essn	before_salary	after_salary	udatetime

```
%sql update EMPLOYEE set salary = 2 * salary where dno = 5;
```

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

```
%sql select * from salary_audit;
```

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

essn	before_salary	after_salary	udatetime
123456789	30000.00	60000.00	2021-11-24 01:42:02
333445555	40000.00	80000.00	2021-11-24 01:42:02
453453453	25000.00	50000.00	2021-11-24 01:42:02
666884444	38000.00	76000.00	2021-11-24 01:42:02