Basic SQL Practice

VLDB Lab.

Professor Sangwon Lee

Contents

- **Basic SQL using LiveSQL**
- **Export CSV File - LiveSQL**
- **Import CSV File – Postgres**
- **Basic SQL using Postgres (Create/Drop Table, Insert/Update/Delete Tuples)**

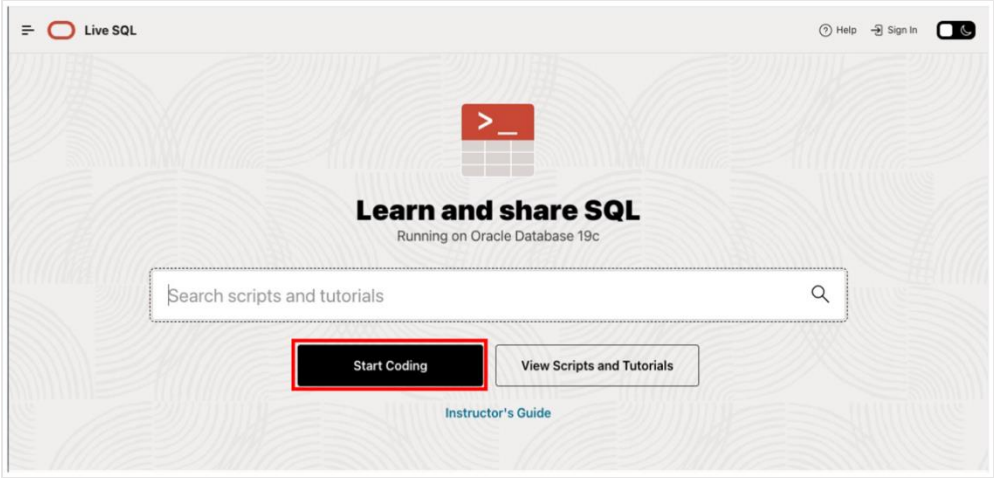
Scott Schema

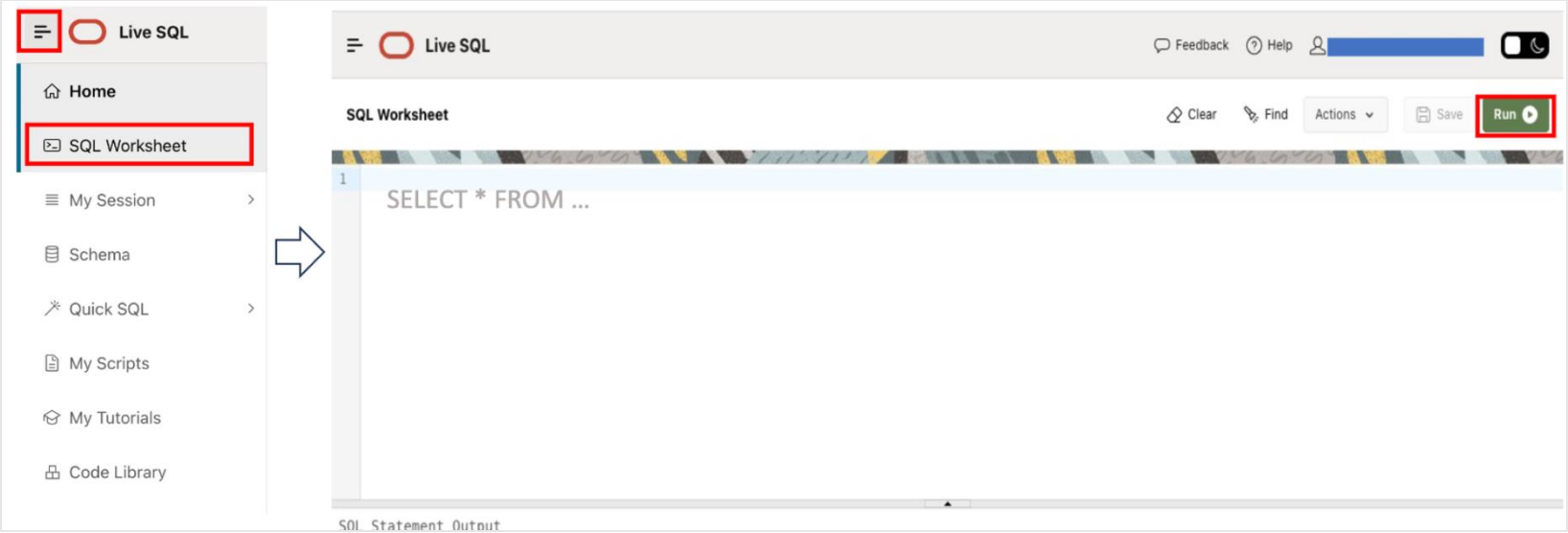
EMP	
empno	Number(4)
ename	Varchar2(10)
job	Varchar2(9)
mgr	Number(4)
hiredate	date
sal	Number(7,2)
comm	Number(7,2)
deptno	Number(2)

DEPT	
deptno	Number(2)
dname	Varchar2(14)
loc	Varchar2(13)

Basic SQL using LiveSQL

1. **Sign In to LiveSQL. (https://livesql.oracle.com/)**





Basic SQL using LiveSQL

1. **Refer to the github link.** (https://github.com/snu-vldb-ta/SNU-BigData-Fintech-2025-1H/tree/main)

SNU-BigData-Fintech-2025-1H / 1 / 1.1 Introduction to DB (LiveSQL).md

Section 1.5.1

```
desc emp;
desc dept;
desc salgrade;

select * from emp;
select * from dept;
```

Section 1.6

```
select empno, ename from emp;
select * from emp where deptno = 20;
select * from emp where deptno = 20 and sal >= 2000;
select deptno, count(*) from emp group by deptno;
select deptno, avg(sal) from emp group by deptno;
select ename, dname, loc from emp e, dept d where e.deptno = d.deptno;

-- + advanced SQL (chater 25)
SELECT ENAME, SAL, PERCENT_RANK() OVER (ORDER BY SAL DESC) as PR FROM emp;
```

Basic SQL using LiveSQL

1. **Refer to the github link.** (https://github.com/snu-vldb-ta/SNU-BigData-Fintech-2025-1H/tree/main)

SNU-BigData-Fintech-2025-1H / 1 / 1.2 relational model (LiveSQL).md Section 3.3

```sql
CREATE TABLE DEPT2
       (DEPTNO NUMBER(2) CONSTRAINT PK_DEPT2 PRIMARY KEY,
        DNAME VARCHAR2(14) ,
        LOC VARCHAR2(13) ) ;
CREATE TABLE EMP2
       (EMPNO NUMBER(4) CONSTRAINT PK_EMP2 PRIMARY KEY,
        ENAME VARCHAR2(10),
        JOB VARCHAR2(9),
        MGR NUMBER(4) CONSTRAINT FK_MGR REFERENCES EMP2 ON DELETE CASCADE, -- Oracle Options: NO ACTION, CASCADE, SE
        HIREDATE DATE,
        SAL NUMBER(7,2),
        COMM NUMBER(7,2),
        DEPTNO NUMBER(2) CONSTRAINT FK_DEPTNO2 REFERENCES DEPT2 ON DELETE CASCADE);

INSERT INTO DEPT2 VALUES (10,'ACCOUNTING','NEW YORK');
INSERT INTO DEPT2 VALUES (20,'RESEARCH','DALLAS');
INSERT INTO DEPT2 VALUES (30,'SALES','CHICAGO');
INSERT INTO DEPT2 VALUES (40,'OPERATIONS','BOSTON');

INSERT INTO EMP2 VALUES (7839,'KING','PRESIDENT',NULL,to_date('17-11-1981','dd-mm-yyyy'),5000,NULL,10);
INSERT INTO EMP2 VALUES (7566,'JONES','MANAGER',7839,to_date('2-4-1981','dd-mm-yyyy'),2975,NULL,20);
INSERT INTO EMP2 VALUES (7698,'BLAKE','MANAGER',7839,to_date('1-5-1981','dd-mm-yyyy'),2850,NULL,30);
INSERT INTO EMP2 VALUES (7782,'CLARK','MANAGER',7839,to_date('9-6-1981','dd-mm-yyyy'),2450,NULL,10);
INSERT INTO EMP2 VALUES (7788,'SCOTT','ANALYST',7566,to_date('13-7-87','dd-mm-rr')-85,3000,NULL,20);
INSERT INTO EMP2 VALUES (7902,'FORD','ANALYST',7566,to_date('3-12-1981','dd-mm-yyyy'),3000,NULL,20);
INSERT INTO EMP2 VALUES (7369,'SMITH','CLERK',7902,to_date('17-12-1980','dd-mm-yyyy'),800,NULL,20);
INSERT INTO EMP2 VALUES (7499,'ALLEN','SALESMAN',7698,to_date('20-2-1981','dd-mm-yyyy'),1600,300,30);
INSERT INTO EMP2 VALUES (7521,'WARD','SALESMAN',7698,to_date('22-2-1981','dd-mm-yyyy'),1250,500,30);
INSERT INTO EMP2 VALUES (7900,'JAMES','CLERK',7698,to_date('3-12-1981','dd-mm-yyyy'),950,NULL,30);
INSERT INTO EMP2 VALUES (7654,'MARTIN','SALESMAN',7698,to_date('28-9-1981','dd-mm-yyyy'),1250,1400,30);
INSERT INTO EMP2 VALUES (7844,'TURNER','SALESMAN',7698,to_date('8-9-1981','dd-mm-yyyy'),1500,0,30);
INSERT INTO EMP2 VALUES (7876,'ADAMS','CLERK',7788,to_date('13-7-87', 'dd-mm-rr')-51,1100,NULL,20);
INSERT INTO EMP2 VALUES (7934,'MILLER','CLERK',7782,to_date('23-1-1982','dd-mm-yyyy'),1300,NULL,10);
commit;

DELETE FROM DEPT2 WHERE DEPTNO = 30;      -- Check Blake, Allen, Ward, James and Martin from EMP2 table
DELETE FROM EMP2  WHERE ENAME = 'JONES';  -- Check Scott, Ford from EMP2 table
DELETE FROM DEPT2 WHERE DEPTNO = 10;      -- Check EMP2 table. What about DEPT2 table?

DROP TABLE EMP2;
DROP TABLE DEPT2;
```

Basic SQL using LiveSQL

1. **Refer to the github link.** (https://github.com/snu-vldb-ta/SNU-BigData-Fintech-2025-1H/tree/main)

SNU-BigData-Fintech-2025-1H / 2 / 2.1 relational algebra.md

Section 4.2

```
---------------
-- projection -
---------------

select ename, sal from emp;
-- input relation schema vs. output result (relation) schema
select sal, ename from emp;
-- Note 1: The column orders in emp schema and projection list are different!
-- Note 2: Two results from the above two queries are same.
--         The order of columns does not matter.
select job from emp;
-- note: duplication in the result table (It is not SET but Multi-set or Bag
-- Relational algebra: set semantics vs. SQL: bag semantics

-- How to eliminate duplicate tuples from the result:
select distinct job from emp;

--------------
-- selection -
--------------

select * from where deptno = 30;

select * from emp where not(deptno = 30);
-- where deptno != 30; or where deptno <> 30;

select * from emp where deptno = 30 and sal > 2000;

select * from emp where deptno = 30 and (job = 'CLERK' or sal > 2000);

-- selection, then projection
select ename, sal from emp
where deptno = 30;
```

Basic SQL using LiveSQL

1. **Refer to the github link.** (https://github.com/snu-vldb-ta/SNU-BigData-Fintech-2025-1H/tree/main)

SNU-BigData-Fintech-2025-1H / 2 / 2.2 basic sql.md

Section 5.1

```
SELECT *
FROM EMP
ORDER BY ename;
-- or, {depno, hiredate}
```

Section 5.2

```
------------------------------
--====== 5.2 JOIN ===========
------------------------------

-- equi-join: join condition
select ename, loc
from emp, dept
where emp.deptno = dept.deptno;

-- retrieve deptno of all departments with at least one employee.
select d.deptno
from dept d, emp e
where d.deptno = e.deptno;
-- What if Distinct keyword is used in SELECT?
```

Basic SQL using LiveSQL

1. **Refer to the github link.** (https://github.com/snu-vldb-ta/SNU-BigData-Fintech-2025-1H/tree/main)

SNU-BigData-Fintech-2025-1H / 2 / 2.2 basic sql.md

```
-- keyword in SQL is NOT case-sensitive: SELECT, select, Select ...
-- String value is CASE-SENSITIVE:       'BOB' != 'Bob' != 'bob'
SELECT *
From emp
where ename like 'S%';

SELECT *
FROM emp
WHERE ename like 'S_I%';
```

Section 5.4

```
---------------------------------------------
--====== 5.4 Nested Query or SUBQUERY =======
---------------------------------------------

SELECT ename
FROM   emp
WHERE  sal > (SELECT sal
              FROM   emp
              WHERE  empno=7566);

-- This query can be rewritten to the following join query.
-- (Query Transformation! pioneered by Prof. Won Kim)

SELECT e1.ename
FROM emp e1, emp e2
WHERE e1.sal > e2.sal and e2.empno = 7566;

SELECT empno
FROM   emp
WHERE  sal > some (SELECT sal
              FROM   emp
              WHERE  deptno = 30)

SELECT e1.empno
FROM emp e1, emp e2
WHERE e1.sal > e2.sal and e2.deptno = 30;
```

Basic SQL using LiveSQL

1. **Refer to the github link.** (https://github.com/snu-vldb-ta/SNU-BigData-Fintech-2025-1H/tree/main)

SNU-BigData-Fintech-2025-1H / 2 / 2.2 basic sql.md

Section 5.4

```
---------------------------------------------
--====== 5.4 Nested Query or SUBQUERY =======
---------------------------------------------

SELECT ename
FROM   emp
WHERE  sal > (SELECT sal
              FROM   emp
              WHERE  empno=7566);

-- This query can be rewritten to the following join query.
-- (Query Transformation! pioneered by Prof. Won Kim)

SELECT e1.ename
FROM emp e1, emp e2
WHERE e1.sal > e2.sal and e2.empno = 7566;

SELECT empno
FROM   emp
WHERE  sal > some (SELECT sal
              FROM   emp
              WHERE  deptno = 30)

SELECT e1.empno
FROM emp e1, emp e2
WHERE e1.sal > e2.sal and e2.deptno = 30;
```

Basic SQL using LiveSQL

1. **Refer to the github link.** (https://github.com/snu-vldb-ta/SNU-BigData-Fintech-2025-1H/tree/main)

SNU-BigData-Fintech-2025-1H / 2 / 2.2 basic sql.md

```
-------------------------------------------------------------
-- MULTIPLE-ROW subquery:                                   --
-- set comparison operators: EXISTS, IN, ANY(or SOME), ALL --
-- UNIQUE is not supported in Oracle.                       --
-------------------------------------------------------------

SELECT dname
FROM   dept d
WHERE  EXISTS ( SELECT *
                FROM   emp e
                WHERE  d.deptno = e.deptno);

SELECT  empno, ename, job
FROM    emp
WHERE   sal < ANY (SELECT sal
                   FROM   emp
                   WHERE  job = 'CLERK')
AND     job <> 'CLERK';

SELECT  empno, ename, job
FROM    emp
WHERE   sal < ANY (SELECT sal
                   FROM   emp
                   WHERE  job = 'NOJOB')
AND     job <> 'CLERK';
-- What if subquery returns empty relation?

SELECT  empno, ename, job
FROM    emp
WHERE   sal > ALL (SELECT       avg(sal)
                   FROM         emp
                   GROUP BY     deptno);
```

Basic SQL using LiveSQL

1. **Refer to the github link.** (https://github.com/snu-vldb-ta/SNU-BigData-Fintech-2025-1H/tree/main)

SNU-BigData-Fintech-2025-1H / 2 / 2.2 basic sql.md

Section 5.5

```
----------------------------
--====== 5.5 GROUP BY ======
----------------------------

-- Maximum salary
select max(sal)
from emp;

-- count of employees, max/min/avg/sum of salary
select count(*), max(sal), min(sal), avg(sal), sum(sal)
from emp;

-- List the name and salary of employee with greatest salary
-- The following query is ILLEGAL!
select ename, max(sal)
from emp;


-- One solution
select ename, sal
from emp
where sal = ( select max(sal)
              from emp );

-- Max salary in each dept
select deptno, max(sal)
from emp
group by deptno;

-- Avg salary by dept, job
select deptno, job, avg(sal)
from emp
group by deptno, job;

select job, avg(sal)  /* attribute list is a subset of the grouping-list */
from emp
group by deptno, job;
```

Basic SQL using LiveSQL

1. **Refer to the github link.** (https://github.com/snu-vldb-ta/SNU-BigData-Fintech-2025-1H/tree/main)

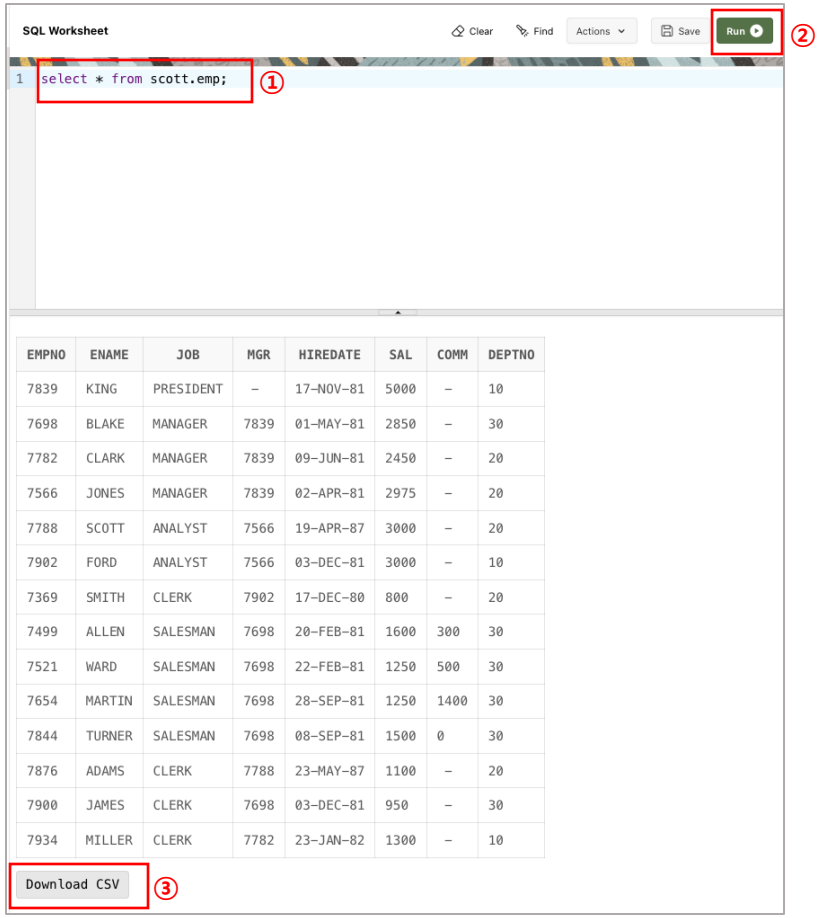
SNU-BigData-Fintech-2025-1H / 2 / 2.2 basic sql.md

Section 5.6

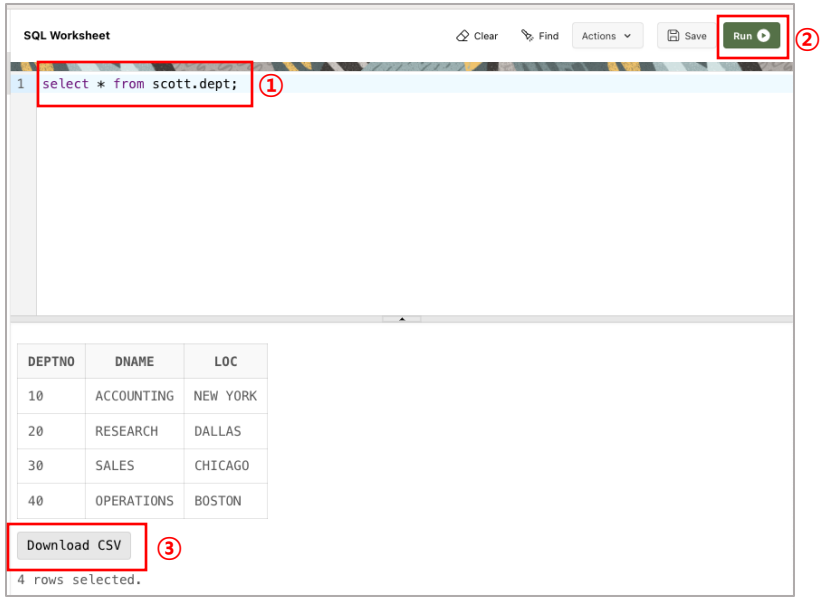
```
-------------------------------
--====== 5.6 NULL Values ======
-------------------------------

select * from emp;


-- 5.6.1 Comparison using NULL Values

-- IS NULL: check whether column value is null
-- If NULL, then TRUE
select ename
from emp
where comm is null;

-- NVL function: if a column value is NULL, set its value to the given one.
select ename, nvl(comm, 0)
from emp;
```

Export CSV File

1. Sign In to LiveSQL. (https://livesql.oracle.com/)
2. Select Data from emp table and dept table. Then, download csv file of each table.



EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7839	KING	PRESIDENT	-	17-NOV-81	5000	-	10
7698	BLAKE	MANAGER	7839	01-MAY-81	2850	-	30
7782	CLARK	MANAGER	7839	09-JUN-81	2450	-	20
7566	JONES	MANAGER	7839	02-APR-81	2975	-	20
7788	SCOTT	ANALYST	7566	19-APR-87	3000	-	20
7902	FORD	ANALYST	7566	03-DEC-81	3000	-	10
7369	SMITH	CLERK	7902	17-DEC-80	800	-	20
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30
7844	TURNER	SALESMAN	7698	08-SEP-81	1500	0	30
7876	ADAMS	CLERK	7788	23-MAY-87	1100	-	20
7900	JAMES	CLERK	7698	03-DEC-81	950	-	30
7934	MILLER	CLERK	7782	23-JAN-82	1300	-	10

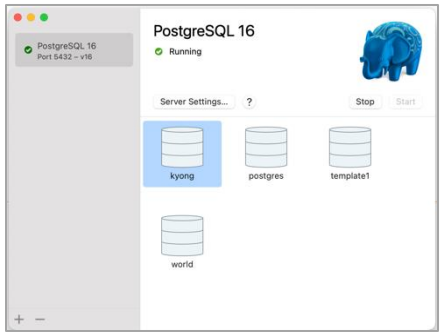


DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

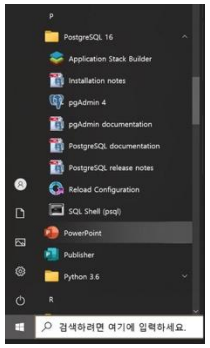
Start Postgres

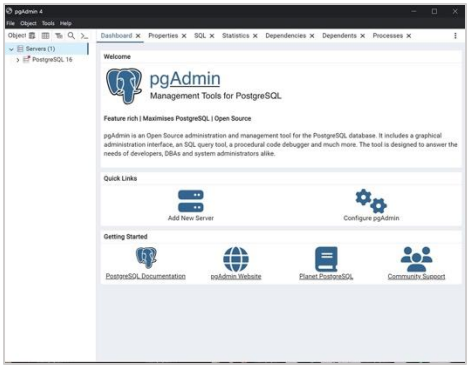
1. Start Postgres.

[Mac]

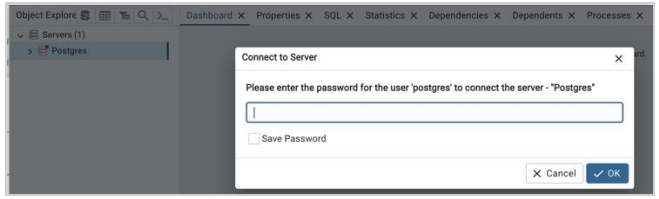


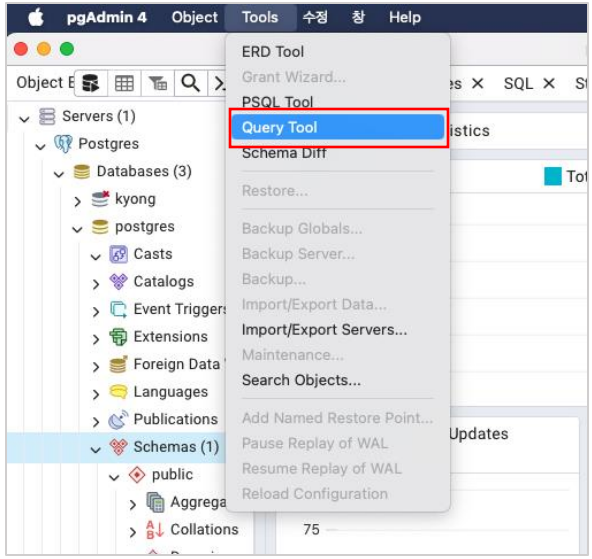
[Windows]

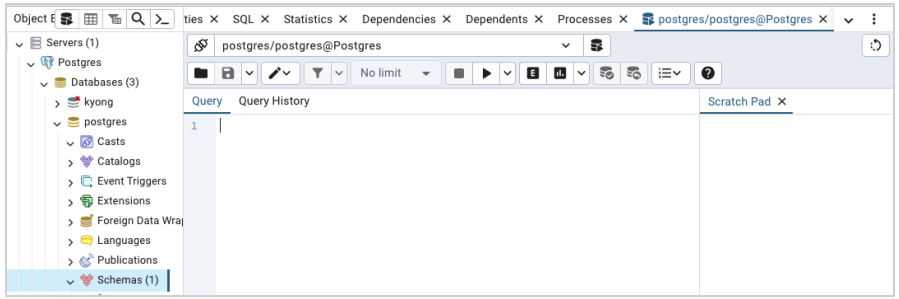




2. Start pgAdmin and connect postgres server.



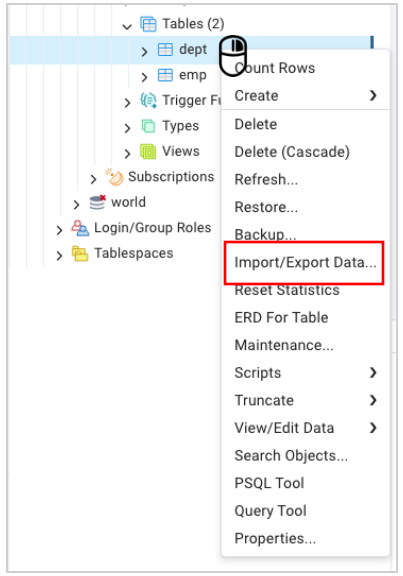


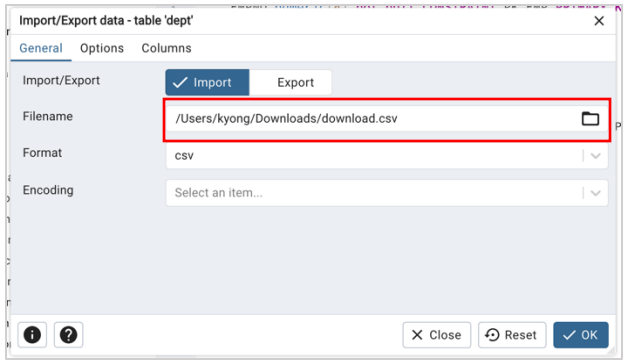


Now you can write sql query!

Import CSV File

1. Right Click dept table and click ‘Import/Export Data…’



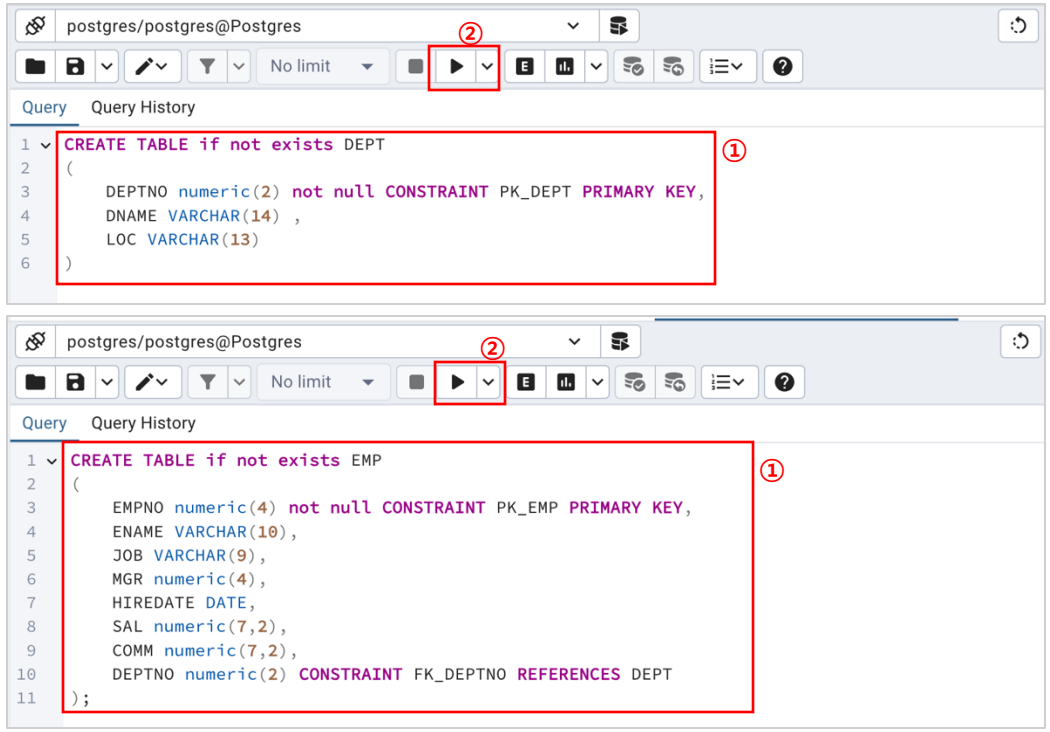


※ 만약 윈도우에서 import가 되지 않는다면 github에 올린 emp, dept 파일을 사용해주세요!

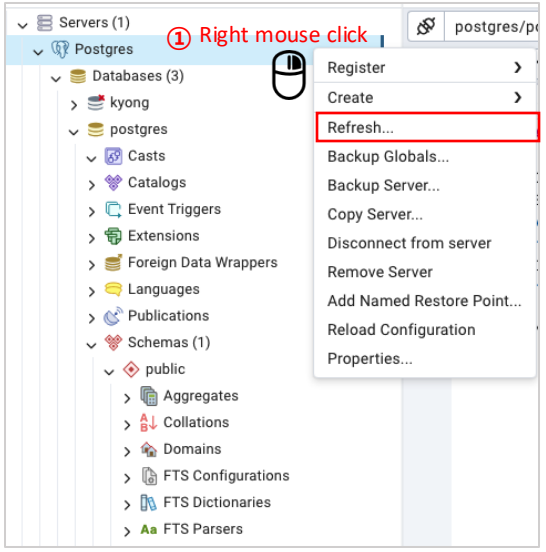
2. Repeat process 1. for the emp table.

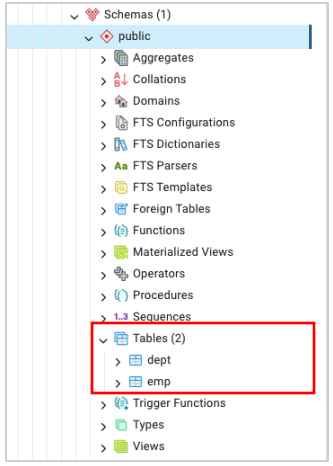
CREATE TABLE

1. Create dept, emp table in postgres.



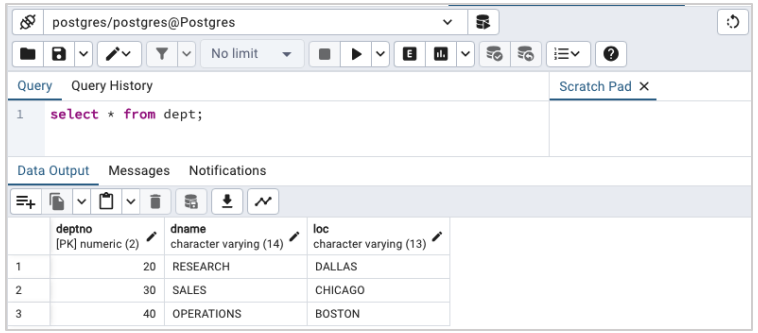
If you refresh postgres server, you can now see two tables.



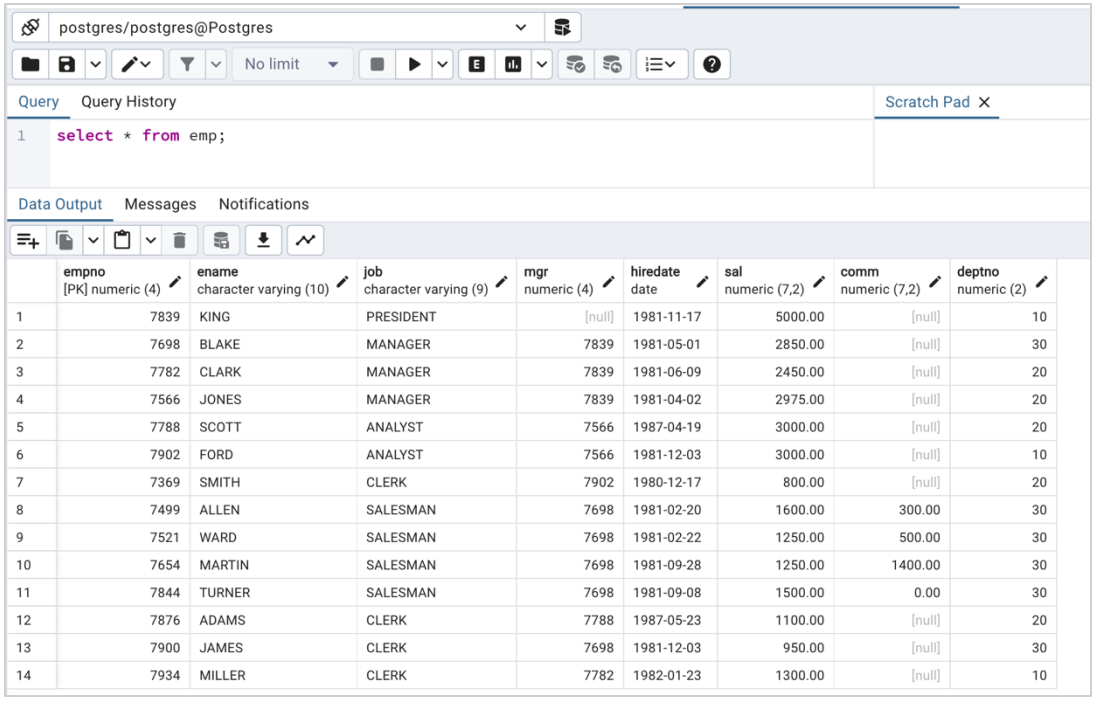


SELECT statement

1. **Now, you can search data within the Scott account table using the SELECT statement.**



	deptno [PK] numeric (2)	dname character varying (14)	loc character varying (13)
1	20	RESEARCH	DALLAS
2	30	SALES	CHICAGO
3	40	OPERATIONS	BOSTON



	empno [PK] numeric (4)	ename character varying (10)	job character varying (9)	mgr numeric (4)	hiredate date	sal numeric (7,2)	comm numeric (7,2)	deptno numeric (2)
1	7839	KING	PRESIDENT	[null]	1981-11-17	5000.00	[null]	10
2	7698	BLAKE	MANAGER	7839	1981-05-01	2850.00	[null]	30
3	7782	CLARK	MANAGER	7839	1981-06-09	2450.00	[null]	20
4	7566	JONES	MANAGER	7839	1981-04-02	2975.00	[null]	20
5	7788	SCOTT	ANALYST	7566	1987-04-19	3000.00	[null]	20
6	7902	FORD	ANALYST	7566	1981-12-03	3000.00	[null]	10
7	7369	SMITH	CLERK	7902	1980-12-17	800.00	[null]	20
8	7499	ALLEN	SALESMAN	7698	1981-02-20	1600.00	300.00	30
9	7521	WARD	SALESMAN	7698	1981-02-22	1250.00	500.00	30
10	7654	MARTIN	SALESMAN	7698	1981-09-28	1250.00	1400.00	30
11	7844	TURNER	SALESMAN	7698	1981-09-08	1500.00	0.00	30
12	7876	ADAMS	CLERK	7788	1987-05-23	1100.00	[null]	20
13	7900	JAMES	CLERK	7698	1981-12-03	950.00	[null]	30
14	7934	MILLER	CLERK	7782	1982-01-23	1300.00	[null]	10

SELECT statement

2. **You can also execute advanced queries.**

Query Query History

```
select * from emp where job not in('PRESIDENT','MANAGER');
```

Query Query History

```
select max(sal) from emp;
```

Query Query History

```
select * from emp where deptno in (
    select deptno from emp where ename='JAMES'
    )
```

Query Query History

```
select emp.ename, emp.deptno, dept.loc from emp
    join dept on emp.deptno = dept.deptno
    where emp.ename = 'KING'
```

INSERT statement

- **Insert new tuple in the emp table.**

[Syntax]

```
INSERT INTO table_name(column1, column2, ...)
VALUES (value1, value2, ...);
```

[Example]

postgres/postgres@Postgres

No limit

Query | Query History

```
INSERT INTO emp VALUES
(1234, 'BKMS1', 'MANAGER', null, '2024-03-26', 3000.00, null, 10);
```

postgres/postgres@Postgres

No limit

Query | Query History

```
select * from emp;
```

Data Output | Messages | Notifications

	empno [PK] numeric (4)	ename character varying (10)	job character varying (9)	mgr numeric (4)	hiredate date	sal numeric (7,2)	comm numeric (7,2)	deptno numeric (2)
1	7839	KING	PRESIDENT	[null]	1981-11-17	5000.00	[null]	10
2	7698	BLAKE	MANAGER	7839	1981-05-01	2850.00	[null]	30
3	7782	CLARK	MANAGER	7839	1981-06-09	2450.00	[null]	20
4	7566	JONES	MANAGER	7839	1981-04-02	2975.00	[null]	20
5	7788	SCOTT	ANALYST	7566	1987-04-19	3000.00	[null]	20
6	7902	FORD	ANALYST	7566	1981-12-03	3000.00	[null]	10
7	7369	SMITH	CLERK	7902	1980-12-17	800.00	[null]	20
8	7499	ALLEN	SALESMAN	7698	1981-02-20	1600.00	300.00	30
9	7521	WARD	SALESMAN	7698	1981-02-22	1250.00	500.00	30
10	7654	MARTIN	SALESMAN	7698	1981-09-28	1250.00	1400.00	30
11	7844	TURNER	SALESMAN	7698	1981-09-08	1500.00	0.00	30
12	7876	ADAMS	CLERK	7788	1987-05-23	1100.00	[null]	20
13	7900	JAMES	CLERK	7698	1981-12-03	950.00	[null]	30
14	7934	MILLER	CLERK	7782	1982-01-23	1300.00	[null]	10
15	1234	BKMS1	MANAGER	[null]	2024-03-26	3000.00	[null]	10

Update statement

- **Insert new tuple in the emp table.**

[Syntax]

```
UPDATE table
SET column1 = value1, column2 = value2, ...
WHERE condition;
```

[Example]

postgres/postgres@Postgres

No limit

Query | Query History

```
1  update emp set sal=10000.00 where ename='KING'
```

Query | Query History | Scratch Pad

```
1  select * from emp;
```

Data Output | Messages | Notifications

	empno [PK] numeric (4)	ename character varying (10)	job character varying (9)	mgr numeric (4)	hiredate date	sal numeric (7,2)	comm numeric (7,2)	deptno numeric (2)
1	7698	BLAKE	MANAGER	7839	1981-05-01	2850.00	[null]	30
2	7782	CLARK	MANAGER	7839	1981-06-09	2450.00	[null]	20
3	7566	JONES	MANAGER	7839	1981-04-02	2975.00	[null]	20
4	7788	SCOTT	ANALYST	7566	1987-04-19	3000.00	[null]	20
5	7902	FORD	ANALYST	7566	1981-12-03	3000.00	[null]	10
6	7369	SMITH	CLERK	7902	1980-12-17	800.00	[null]	20
7	7499	ALLEN	SALESMAN	7698	1981-02-20	1600.00	300.00	30
8	7521	WARD	SALESMAN	7698	1981-02-22	1250.00	500.00	30
9	7654	MARTIN	SALESMAN	7698	1981-09-28	1250.00	1400.00	30
10	7844	TURNER	SALESMAN	7698	1981-09-08	1500.00	0.00	30
11	7876	ADAMS	CLERK	7788	1987-05-23	1100.00	[null]	20
12	7900	JAMES	CLERK	7698	1981-12-03	950.00	[null]	30
13	7934	MILLER	CLERK	7782	1982-01-23	1300.00	[null]	10
14	7839	KING	PRESIDENT	[null]	1981-11-17	10000.00	[null]	10

DELETE statement

- **Delete the tuple in the table.**

[Syntax]

```
DELETE FROM
table_name
WHERE condition ;
```

[Example]

postgres/postgres@Postgres

No limit

Query | Query History

```
1  delete from emp where ename = 'BKMS1';
```

postgres/postgres@Postgres

No limit

Query | Query History

```
1  select * from emp;
```

Data Output | Messages | Notifications

	empno [PK] numeric (4)	ename character varying (10)	job character varying (9)	mgr numeric (4)	hiredate date	sal numeric (7,2)	comm numeric (7,2)	deptno numeric (2)
1	7839	KING	PRESIDENT	[null]	1981-11-17	5000.00	[null]	10
2	7698	BLAKE	MANAGER	7839	1981-05-01	2850.00	[null]	30
3	7782	CLARK	MANAGER	7839	1981-06-09	2450.00	[null]	20
4	7566	JONES	MANAGER	7839	1981-04-02	2975.00	[null]	20
5	7788	SCOTT	ANALYST	7566	1987-04-19	3000.00	[null]	20
6	7902	FORD	ANALYST	7566	1981-12-03	3000.00	[null]	10
7	7369	SMITH	CLERK	7902	1980-12-17	800.00	[null]	20
8	7499	ALLEN	SALESMAN	7698	1981-02-20	1600.00	300.00	30
9	7521	WARD	SALESMAN	7698	1981-02-22	1250.00	500.00	30
10	7654	MARTIN	SALESMAN	7698	1981-09-28	1250.00	1400.00	30
11	7844	TURNER	SALESMAN	7698	1981-09-08	1500.00	0.00	30
12	7876	ADAMS	CLERK	7788	1987-05-23	1100.00	[null]	20
13	7900	JAMES	CLERK	7698	1981-12-03	950.00	[null]	30
14	7934	MILLER	CLERK	7782	1982-01-23	1300.00	[null]	10

No ‘BKMS1’

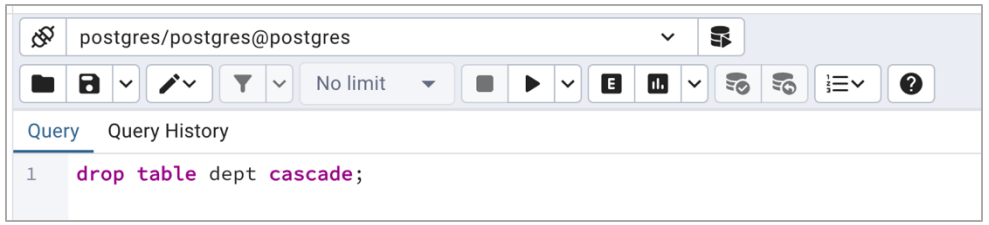
DROP Table

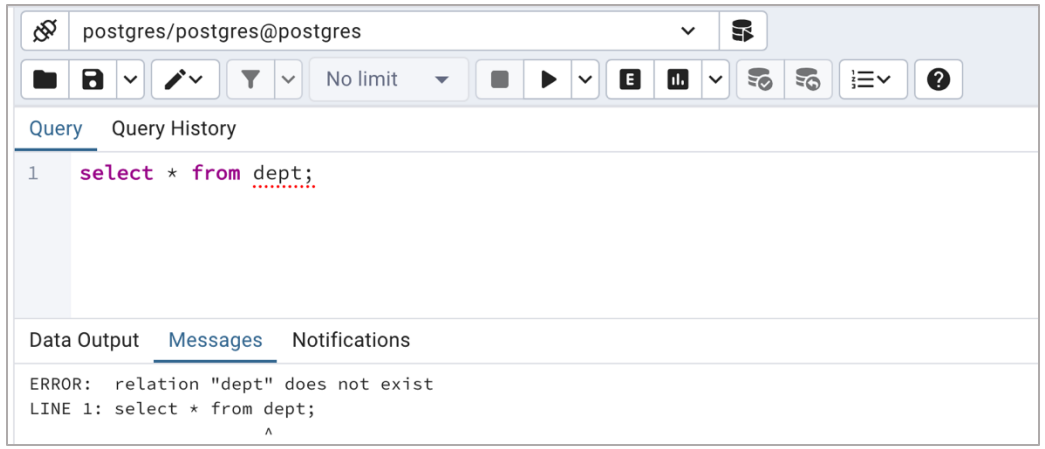
- **Drop table**

[Syntax]

```
DROP TABLE [IF EXISTS]
 table_name [CASCADE | RESTRICT];
```

[Example]





Think : Why does DROP table without cascade condition doesn't work in dept table?

Think : What is the difference between DROP table and TRUNCATE table?