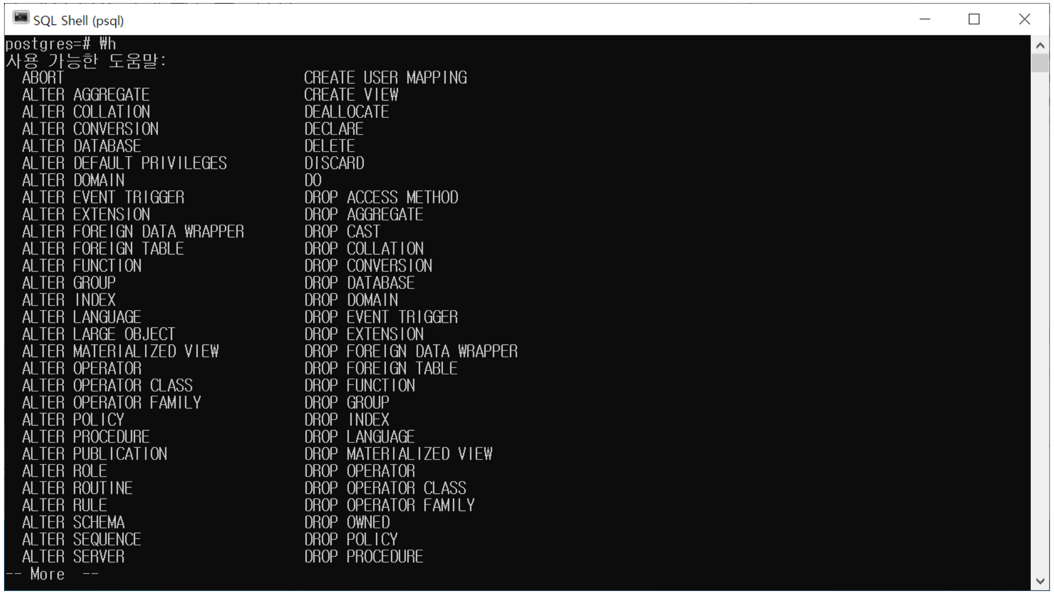
Lab

Chapter 4 – Intermediate SQL

Useful PostgreSQL Commands

- \h: help, \h *command*: help on the command
- \d: list tables, \d *table_name*: describe table
- \i *file_name*: import SQL script
- \c *database_name*: connect to the database
- \l : list databases
- \q: quit PostgreSQL
- History 기능 제공 (위, 아래 화살표 사용)



```
postgres=# ₩h
사용 가능한 도움말:
  ABORT                          CREATE USER MAPPING
  ALTER AGGREGATE                CREATE VIEW
  ALTER COLLATION                DEALLOCATE
  ALTER CONVERSION               DECLARE
  ALTER DATABASE                 DELETE
  ALTER DEFAULT PRIVILEGES       DISCARD
  ALTER DOMAIN                   DO
  ALTER EVENT TRIGGER            DROP ACCESS METHOD
  ALTER EXTENSION                DROP AGGREGATE
  ALTER FOREIGN DATA WRAPPER     DROP CAST
  ALTER FOREIGN TABLE            DROP COLLATION
  ALTER FUNCTION                 DROP CONVERSION
  ALTER GROUP                    DROP DATABASE
  ALTER INDEX                    DROP DOMAIN
  ALTER LANGUAGE                 DROP EVENT TRIGGER
  ALTER LARGE OBJECT             DROP EXTENSION
  ALTER MATERIALIZED VIEW        DROP FOREIGN DATA WRAPPER
  ALTER OPERATOR                 DROP FOREIGN TABLE
  ALTER OPERATOR CLASS           DROP FUNCTION
  ALTER OPERATOR FAMILY          DROP GROUP
  ALTER POLICY                   DROP INDEX
  ALTER PROCEDURE                DROP LANGUAGE
  ALTER PUBLICATION              DROP MATERIALIZED VIEW
  ALTER ROLE                     DROP OPERATOR
  ALTER ROUTINE                  DROP OPERATOR CLASS
  ALTER RULE                     DROP OPERATOR FAMILY
  ALTER SCHEMA                   DROP OWNED
  ALTER SEQUENCE                 DROP POLICY
  ALTER SERVER                   DROP PROCEDURE
-- More  --
```

Database Setup

1. Download the following sql file from blackboard
   - University.sql (which we used in Chapter3 lab)
2. Make university schema and insert the data into relations using sql files
   a. Execute PostgreSQL SQL Shell(psql)
   b. Create a new database using ‘CREATE DATABASE chapter4;’ command
   c. Run ‘\c chapter4’ // connection to database ‘chapter4’
   d. Run ‘\i [filepath]/University.sql’ (Don’t use whitespace or backslash ‘\’ in the filepath)
      - \i 'C:\\Users\\account\\한글 폴더\\University.sql’ (double backslash wrapped in single quotation marks)
      - 문제가 있으면 파일을 조건에 맞는 디렉토리로 옮겨서 사용

DDL Statements for University Database “university.sql”

- create table classroom(building varchar(15), room_number varchar(7), capacity numeric(4,0), primary key (building, room_number));
- create table department(dept_name varchar(20), building varchar(15), budget numeric(12,2) check (budget > 0), primary key (dept_name));
- create table course(course_id varchar(8), title varchar(50), dept_name varchar(20), credits numeric(2,0) check (credits > 0), primary key (course_id), foreign key (dept_name) references department (dept_name) on delete set null);
- create table instructor(ID varchar(5), name varchar(20) not null, dept_name varchar(20), salary numeric(8,2) check (salary > 9000), primary key (ID), foreign key (dept_name) references department (dept_name) on delete set null);
- create table section(course_id varchar(8), sec_id varchar(8), semester varchar(6) check (semester in ('Fall', 'Winter', 'Spring', 'Summer’)), year numeric(4,0) check (year > 1701 and year < 2100), building varchar(15), room_number varchar(7), time_slot_id varchar(4), primary key (course_id, sec_id, semester, year), foreign key (course_id) references course (course_id) on delete cascade, foreign key (building, room_number) references classroom (building, room_number) on delete set null);
- create table teaches(ID varchar(5), course_id varchar(8), sec_id varchar(8), semester varchar(6), year numeric(4,0), primary key (ID, course_id, sec_id, semester, year), foreign key (course_id, sec_id, semester, year) references section (course_id, sec_id, semester, year) on delete cascade, foreign key (ID) references instructor (ID) on delete cascade);
- create table student(ID varchar(5), name varchar(20) not null, dept_name varchar(20), tot_cred numeric(3,0) check (tot_cred >= 0), primary key (ID), foreign key (dept_name) references department (dept_name) on delete set null);
- create table takes(ID varchar(5), course_id varchar(8), sec_id varchar(8), semester varchar(6), year numeric(4,0), grade varchar(2), primary key (ID, course_id, sec_id, semester, year), foreign key (course_id, sec_id, semester, year) references section (course_id, sec_id, semester, year) on delete cascade, foreign key (ID) references student (ID) on delete cascade);
- create table advisor(s_ID varchar(5), i_ID varchar(5), primary key (s_ID), foreign key (i_ID) references instructor (ID) on delete set null, foreign key (s_ID) references student (ID) on delete cascade);
- create table time_slot(time_slot_id varchar(4), day varchar(1), start_hr numeric(2) check (start_hr >= 0 and start_hr < 24), start_min numeric(2) check (start_min >= 0 and start_min < 60), end_hr numeric(2) check (end_hr >= 0 and end_hr < 24), end_min numeric(2) check (end_min >= 0 and end_min < 60), primary key (time_slot_id, day, start_hr, start_min));
- create table prereq(course_id varchar(8), prereq_id varchar(8), primary key (course_id, prereq_id), foreign key (course_id) references course (course_id) on delete cascade, foreign key (prereq_id) references course (course_id));

Exercise

- Make examples for the followings
  1. The university database schema contains various integrity constraints. Execute some SQL statements violating them.
     - Primary key constraints, foreign key constraints, not null constraints, etc.
     - Hint: \d table_name
  2. Make two or more concurrently executed transactions, and show they are executed in an isolated manner.
     - Hint: Make two (or more) windows (i.e., terminals) and use 'begin transaction' commands
  3. Make users and set up a few authorization rules; Show some non-authorized accesses.
     - Hint: \h create user, \h grant, \h revoke
       - ***create user*** *guest* ***login password*** *'1234' ;*
     - Hint: Use two or more windows for different users
  4. Create some views and show how view maintenance works and how view update is processed.
     - Update of source relation vs. update of view.

Homework

- Complete today's practice exercise
- Take some screenshots containing the execution results
- Submit your report on blackboard
  - 10:29:59, 2024/05/14
  - **Only PDF files** are accepted
  - **No late submission**

End of Lab