CSC430/530 – Database Management Systems

Lab 0 – Creating & populating the database.

Part A – Creating DB and tables, defining domains of attributes & entity integrity constraints (primary keys).

Create database and define schemas for every relation (table).

- First, open phpMyAdmin and click on "SQL".
- To create the database named “company”, put the following in the SQL editor:
  `CREATE DATABASE company;`
- Click the "Go" button to run the query (or press ctrl+enter).
- Click on the created "company" database in the left panel
  - if using command line, this is the same as entering: `use company;`
- Next, we will create a schema and define domain & entity integrity constraints for every relation:
- To do this, in the SQL editor for the company database, enter the following one at a time:
  - Employee relation:

```
DROP TABLE IF EXISTS employee;
CREATE TABLE employee (
     fname VARCHAR(15) NOT NULL,
     minit CHAR,
     lname VARCHAR(15) NOT NULL,
     ssn VARCHAR(9) NOT NULL,
     bdate DATE,
     address VARCHAR(50),
     sex CHAR,
     salary DECIMAL(10, 2) CHECK (salary > 0),
     super_ssn VARCHAR(9),
     dno INTEGER DEFAULT 1,
     CONSTRAINT emp_pk PRIMARY KEY (ssn)
);
```

  - Dependent relation:

```
DROP TABLE IF EXISTS dependent;
CREATE TABLE dependent (
     essn VARCHAR(9) NOT NULL,
     dependent_name VARCHAR(15) NOT NULL,
     sex CHAR,
     bdate DATE, relationship VARCHAR(8),
     CONSTRAINT dependent_pk PRIMARY KEY (essn,dependent_name)
);
```

- Department relation:

```
DROP TABLE IF EXISTS department;
CREATE TABLE department (
    dname VARCHAR(25) NOT NULL,
    dnumber INTEGER NOT NULL,
    mgr_ssn VARCHAR(9),
    mgr_start_date DATE,
    CONSTRAINT dept_pk PRIMARY KEY (dnumber),
    CONSTRAINT dept_unique UNIQUE (dname)
);
```

- Department locations relation:

```
DROP TABLE IF EXISTS dept_locations;
CREATE TABLE dept_locations (
    dnumber INTEGER NOT NULL,
    dlocation VARCHAR(15) NOT NULL,
    CONSTRAINT dept_loc_pk PRIMARY KEY (dnumber, dlocation)
);
```

- Project relation:

```
DROP TABLE IF EXISTS project;
CREATE TABLE project (
    pname VARCHAR(25) NOT NULL,
    pnumber INTEGER NOT NULL,
    plocation VARCHAR(15),
    dnum INTEGER,
    CONSTRAINT project_pk PRIMARY KEY (pnumber),
    CONSTRAINT project_unique UNIQUE (pname)
);
```

- Works on relation:

```
DROP TABLE IF EXISTS works_on;
CREATE TABLE works_on (
    essn VARCHAR(9) NOT NULL,
    pno INTEGER NOT NULL,
    hours DECIMAL(4,1),
    CONSTRAINT works_on_pk PRIMARY KEY (essn,pno)
);
```

Part B – Defining referential integrity constraints.

Once the schemas with domain and entity integrity constraints are defined, we can define referential integrity constraints (foreign keys).

- In SQL query editor:
  - Employee relation:

```
ALTER TABLE employee
ADD CONSTRAINT emp_super_fk
    FOREIGN KEY (super_ssn) REFERENCES employee(ssn)
        ON DELETE SET NULL
        ON UPDATE CASCADE,
ADD CONSTRAINT emp_dept_fk
    FOREIGN KEY (Dno) REFERENCES department(dnumber)
        ON DELETE SET NULL
        ON UPDATE CASCADE;
```

  - Dependent relation:

```
ALTER TABLE dependent
ADD CONSTRAINT dependent_fk
    FOREIGN KEY (essn) REFERENCES employee(ssn)
        ON DELETE RESTRICT
        ON UPDATE CASCADE;
```

  - Department relation:

```
ALTER TABLE department
ADD CONSTRAINT dept_mgr_fk
    FOREIGN KEY (mgr_ssn) REFERENCES employee(ssn)
        ON DELETE SET NULL
        ON UPDATE CASCADE;
```

  - Department locations relation:

```
ALTER TABLE dept_locations
ADD CONSTRAINT dept_loc_fk
    FOREIGN KEY (dnumber) REFERENCES department(dnumber)
        ON DELETE RESTRICT
        ON UPDATE CASCADE;
```

  - Project relation:

```
ALTER TABLE project
ADD CONSTRAINT project_fk
    FOREIGN KEY (dnum) REFERENCES department(dnumber)
        ON DELETE RESTRICT
        ON UPDATE CASCADE;
```

- Works on relation:

```
ALTER TABLE works_on
ADD CONSTRAINT works_on_ssn_fk
    FOREIGN KEY (essn) REFERENCES employee(ssn)
        ON DELETE RESTRICT
        ON UPDATE CASCADE,
ADD CONSTRAINT works_on_pno_fk
    FOREIGN KEY (pno) REFERENCES project(pnumber)
        ON DELETE RESTRICT
        ON UPDATE CASCADE;
```

Part C – Populating the DB.

Parts A & B conclude the definition of the database schema. Next step is to create database state, i.e. populate the database with data.

Database can be populated in three ways:

- Manually, using `INSERT` DML command.
- Through “Import” (at the top navigation after selecting a table).
  - Accepts data in multiple formats.
- Dump data through command line (shell), using `LOAD DATA INFILE` command.
  - Accepts data in any text file format.

We will use the import option. Do the following for each table:

- Click on a table in the database
- Go to the Import page (click on "Import" at the top).
- Under "File to import", click on "Choose File" and find the .dat file for that table. These files can be found on Canvas in a zip file for Lab 0.
- Scroll down to "Other options" and uncheck "Enable foreign key checks".
- Change the "Format" to "CSV".
- Click "Import" button at the bottom.
- Repeat this for all 6 tables.