CSC430/530 – Database Management Systems

Lab 0 – Creating & populating the database.

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

Open MySQL Workbench.

On the Welcome page, click on "Local instance MySQL80".

When prompted type in user password that you have set up for the database.

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

First, create the database named "company".

In the SQL editor: `CREATE DATABASE company;` -or- `CREATE SCHEMA company;`

Then, start using the database.

In the SQL editor: `USE company;`

Next, create a schema and define domain & entity integrity constraints for every relation.

In the SQL editor:

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 MySQL 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 "Table Data Import Wizard" utility in MySQL Workbench.
  Accepts data in CSV or JSON format.
- Dump data through command line (shell), using **`LOAD DATA INFILE`** command.
  Accepts data in any text file format.

IMPORANT: before going through the next steps, make sure your PATH environment variable contains a path to the MySQL binary folder (e.g. "C:\Program Files\MySQL\MySQL Server 8.0\bin").
Use this link for reference - dev.mysql.com/doc/refman/8.0/en/mysql-installation-windows-path.html

Open command line and connect to your local MySQL database.

```
mysql -u root -p --local_infile
```

Once prompted, type in the password you created for the database.

Select database you are going to use.

```
USE company;
```

Next, allow the database to accept data dump from files located in your filesystem.

```
SET GLOBAL local_infile = 1;
```

Then, turn off automatic checks for database state updates and referential integrity constraint violations. This is needed to prevent database from complaining when the data is dumped.

```
SET SQL_SAFE_UPDATES = 0;
SET FOREIGN_KEY_CHECKS = 0;
```

Finally, populate the database relations with tuples. Data files are provided on Moodle.

Important: You will have to provide full address of a data file location in your system.

E.g. 'C:/Users/andtimo/Desktop/CSC430/company_data/employee.dat'

Employee relation:

```
LOAD DATA LOCAL INFILE 'employee.dat'
REPLACE INTO TABLE employee
FIELDS TERMINATED BY ','
OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY '\n';
```

Dependent relation:

```
LOAD DATA LOCAL INFILE 'dependent.dat'
REPLACE INTO TABLE dependent
FIELDS TERMINATED BY ','
OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY '\n';
```

Department relation:

```
LOAD DATA LOCAL INFILE 'department.dat'
REPLACE INTO TABLE department
FIELDS TERMINATED BY ','
OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY '\n';
```

Department locations relation:

```
LOAD DATA LOCAL INFILE 'dept_locations.dat'
REPLACE INTO TABLE dept_locations
FIELDS TERMINATED BY ','
OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY '\n';
```

Project relation:

```
LOAD DATA LOCAL INFILE 'project.dat'
REPLACE INTO TABLE project
FIELDS TERMINATED BY ','
OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY '\n';
```

Works on relation:

```
LOAD DATA LOCAL INFILE 'works_on.dat'
REPLACE INTO TABLE works_on
FIELDS TERMINATED BY ','
OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY '\n';
```

Last step is to turn back on automatic checks for database state updates and referential integrity constraint violations, so any new updates to the database do not violate database state.

```
SET SQL_SAFE_UPDATES = 1;
SET FOREIGN_KEY_CHECKS = 1;
exit
```