# Lab 3
# Data Definition and Data Manipulation

CSE 4308

DATABASE MANAGEMENT SYSTEMS LAB

SEPTEMBER 1, 2022

# 1 Data Definition

## 1.1 Creating Tables

The general syntax for creating a table with constraints is given below:

```
CREATE TABLE table_name
(
attribute1 datatype [ NULL | NOT NULL ],
attribute2 datatype [ NULL | NOT NULL ],
...,
[CONSTRAINT constraint_name] PRIMARY KEY (primary_attribute1, ...),
[CONSTRAINT constraint_name] FOREIGN KEY (foreign_attribute1, ...)
REFERENCES reference_table_name [ON DELETE CASCADE],
[CONSTRAINT constraint_name] CHECK condition
);
```

Primary key is a special column that is able to uniquely identify each record. For example, if we want to create a table called `COURSE`, it might have the `COURSE_ID` as the primary key. However, considering multiple departments can offer the same course with the same course ID, we can consider both `COURSE_ID` and `DEPT_NAME` as primary keys.
Foreign keys are used to restrict the domain of columns of one table to the values of other table. For example, the `DEPT_NAME` column of the `COURSE` table can only have values of the departments that are available in the `DEPARTMENT` table.
We can also add additional constraints, such as, the `CREDITS` for each course cannot be a negative value.

```
CREATE TABLE COURSE
(
  COURSE_ID VARCHAR2(8),
  TITLE VARCHAR2(50),
  DEPT_NAME VARCHAR2(20),
  CREDITS NUMBER,
  CONSTRAINT PK_COURSE PRIMARY KEY(COURSE_ID),
  CONSTRAINT FK_COURSE_DEPARTMENT FOREIGN KEY(DEPT_NAME)
   REFERENCES DEPARTMENT(DEPT_NAME),
  CONSTRAINT POSITIVE_CREDIT CHECK (CREDITS > 0)
);
```

We can also create table by copying the schema and data from another table:

```
CREATE TABLE new_table_name AS
(
SELECT * FROM old_table_name
);
```

## 1.2 Deleting Tables

To delete both the schema and data from a table, we use:

```
DROP TABLE table_name;
```

To only delete the data from a table, we use:

```
DELETE FROM table_name [WHERE condition];
```

### 1.3 Altering Tables

To add a new attribute to the table, we use:

```
ALTER TABLE table_name ADD attribute_name datatype;
```

We can even add multiple attributes at the same time:

```
ALTER TABLE table_name ADD (attribute1 datatype, ...);
```

To delete an attribute from a table, we use:

```
ALTER TABLE table_name DROP COLUMN attribute_name;
```

We can even delete multiple attributes at the same time:

```
ALTER TABLE table_name DROP COLUMN (attribute1, ...);
```

To modify the data type of an attribute, we need to ensure that the column is empty. Then we can execute:

```
ALTER TABLE table_name MODIFY attribute_name new_datatype;
```

To rename an attribute, we use:

```
ALTER TABLE table_name RENAME COLUMN old_attribute_name TO new_attribute_name;
```

To rename a table, we use:

```
ALTER TABLE table_name RENAME TO new_table_name;
```

To add constraints to a table, we use:

```
ALTER TABLE table_name ADD CONSTRAINT constraint_name constraint;
```

To delete a constraint from a table, we use:

```
ALTER TABLE table_name DROP CONSTRAINT constraint_name;
```

## 2 Data Manipulation

Remember the basic query structure looks like:

```
SELECT a1, a2, ..., an
FROM r1, r2, ..., rm
WHERE p;
```

We can apply arithmetic operations in the SELECT clause:

```
SELECT ID, NAME, SALARY/12
  FROM INSTRUCTOR;
```

We can perform Cartesian Product between tables:

```
SELECT *
  FROM INSTRUCTOR, DEPARTMENT;
```

We can perform natural join as well:

```
SELECT *
  FROM INSTRUCTOR NATURAL JOIN DEPARTMENT;
```

# 3 Lab Task

You have to write all SQL statements in an editor first and save them with .sql extention. Then execute the SQL script.

1. Write SQL statements to create the following tables with the given specifications:

   (a) ACCOUNT

| | | |
|---|---|---|
| ACCOUNT_NO | CHAR(5) | (e.g.: A-101) Primary Key |
| BALANCE | NUMBER | Not Null |

   (b) CUSTOMER

| | | |
|---|---|---|
| CUSTOMER_NO | CHAR(5) | (e.g.: C-101) Primary Key |
| CUSTOMER_NAME | VARCHAR2(20) | Not Null |
| CUSTOMER_CITY | VARCHAR2(10) | (e.g.: DHK, KHL, etc.) |

   (c) DEPOSITOR

| | | |
|---|---|---|
| ACCOUNT_NO | CHAR(5) | (e.g.: A-101) |
| CUSTOMER_NO | CHAR(5) | (e.g.: C-101) |
| | | Primary Key(ACCOUNT_NO, CUSTOMER_NO) |

2. Write SQL statements to perform the following alteration operations:

   (a) Add a new attribute 'DATE_OF_BIRTH' (DATE type) in CUSTOMER table.

   (b) Modify the data type of BALANCE from NUMBER to NUMBER(12, 2).

   (c) Rename the attribute ACCOUNT_NO, CUSTOMER_NO from DEPOSITOR table to A_NO and C_NO, respectively.

   (d) Rename the table DEPOSITOR to DEPOSITOR_INFO.

   (e) Add two foreign key constraints FK_DEPOSITOR_ACCOUNT and FK_DEPOSITOR_CUSTOMER that identifies A_NO and C_NO as foreign keys.

3. Write SQL statements to answer the following queries:

   (a) Find all account number with balance less than 100000.

   (b) Find all customer names who live in 'KHL' city.

   (c) Find all customer number whose name contains 'A'.

   (d) Find distinct account numbers from DEPOSITOR_INFO table.

   (e) Show the result of Cartesian Product between ACCOUNT and DEPOSITOR_INFO table.

   (f) Show the result of Natural Join between CUSTOMER and DEPOSITOR_INFO table.

   (g) Find all customer names and their city who have an account.

   (h) Find all customer related information who have balance greater than 1000.

   (i) Find all accounts related information where balance is in between 5000 and 10000 or their depositor lives in 'DHK' city.