Structured Query Language (SQL) 1: Data Definition Language (DDL)

SECD2523 Database

Semester 1 2021/2022

Learning Objective

At the end of this module, students should be able to:

- Perform basic operations in a DBMS software.
- Construct SQL statements to:
  - Create and delete tables
  - Perform changes to table structure
  - Include integrity constraints to tables

Introduction to DBMS

- DBMS:
  - A software system that enables users to **define**, **create**, and maintain the database and that provides controlled access to this database.
  - DBMS perform the tasks through commands (a.k.a queries) written in Structured Query Language (SQL)

Structured Query Language

- A database language that allows users to:
  - Create database and relation structures.
  - Perform basic data management tasks (insertion, modification and deletion of data from relations)
  - Perform simple and complex queries.
- Two major components:
  - **Data Definition Language (DDL)**
  - **Data Manipulation Language (DML)**

Data Definition Language (DDL)

- Definition:
  - A language that allows DBA or user to **describe** and **name** the entities, attributes, and relationships required for the application, together with any associated **integrity** and **security** constraints.
- DDL allows database objects (schemas, domains, tables, views, etc) to be **created**, **modified** or **deleted**. Examples of DDL statements:
  - CREATE SCHEMA …   DROP SCHEMA …
  - CREATE TABLE …   DROP TABLE …
  - ALTER TABLE …

Integrity Enhancement Feature

- SQL provides some facilities for integrity control to protect the database from becoming inconsistent.
  - Required Data
    - Using the **NOT NULL** constraint for column to ensure the column must contain a valid value.
  - Entity Integrity
    - Using **PRIMARY KEY** constraint for a column with unique, non-null value. Only ONE primary key per table.
    - Use **UNIQUE** constraint for column with unique values. Allow to be NULL but can assign to multiple columns.
  - Referential Integrity
    - Using **FOREIGN KEY** constraint to link to parent table that containing the matching attribute. A table can have multiple foreign keys.

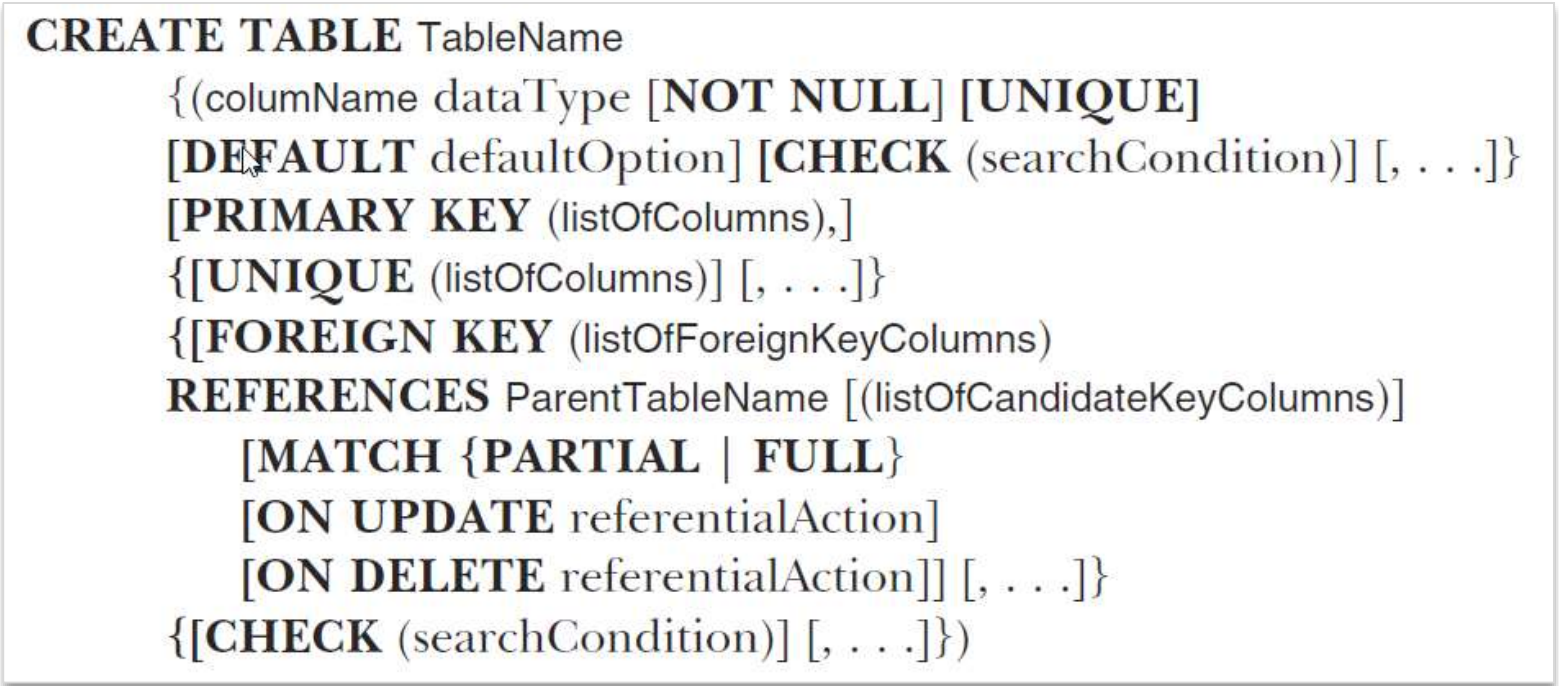
Dealing with Tables

- Define database structure and controlling access to data

1. **CREATE TABLE**:
   - to create table.
2. **ALTER TABLE**:
   - to modify the structure of the existing tables.
3. **DROP TABLE**:
   - to delete the existing tables.

Creating Tables

- General SQL syntax for creating a table:



```
CREATE TABLE TableName
        {(columName dataType [NOT NULL] [UNIQUE]
        [DEFAULT defaultOption] [CHECK (searchCondition)] [, . . .]}
        [PRIMARY KEY (listOfColumns),]
        {[UNIQUE (listOfColumns)] [, . . .]}
        {[FOREIGN KEY (listOfForeignKeyColumns)
        REFERENCES ParentTableName [(listOfCandidateKeyColumns)]
            [MATCH {PARTIAL | FULL}
            [ON UPDATE referentialAction]
            [ON DELETE referentialAction]] [, . . .]}
        {[CHECK (searchCondition)] [, . . .]})
```

Source: Connolly, 2015

[ ] : optional

CREATE TABLE

- Creating a table **without constraint**.

```
CREATE TABLE tableName (
columnName dataType [ DEFAULT value ]
[, column2Name datatype [ DEFAULT value]]);
```

[ ] : optional

CREATE TABLE

- Creating a table **with constraints**.
- Constraints at **column level**.

```
CREATE TABLE tableName
(columnName dataType [CONSTRAINT constraintName]
constraintType [DEFAULT value]
[, column2Name datatype [CONSTRAINT constraintName]
constraintType [ DEFAULT value]]);
```

[ ] : optional

CREATE TABLE

- Creating a table **with constraints**.
- Constraints at **table level**.

```
CREATE TABLE tableName
(columnName dataType [DEFAULT value]
[, column2Name datatype[ DEFAULT value]]
[, CONSTRAINT constraintName constraintType
(columnName,…)]);
```

[ ] : optional

Common Data Type

- Common Data Type (in Oracle)
  - Characters or String: CHAR(size), VARCHAR(size)
  - Dates: DATE
  - Numeric:
    - Integer: INTEGER, NUMBER(p) p: precision
    - Fixed number: NUMBER(p,s) p: precision, s: scale
    - Floating-number: NUMBER, FLOAT

Common SQL Data Type

- Overview

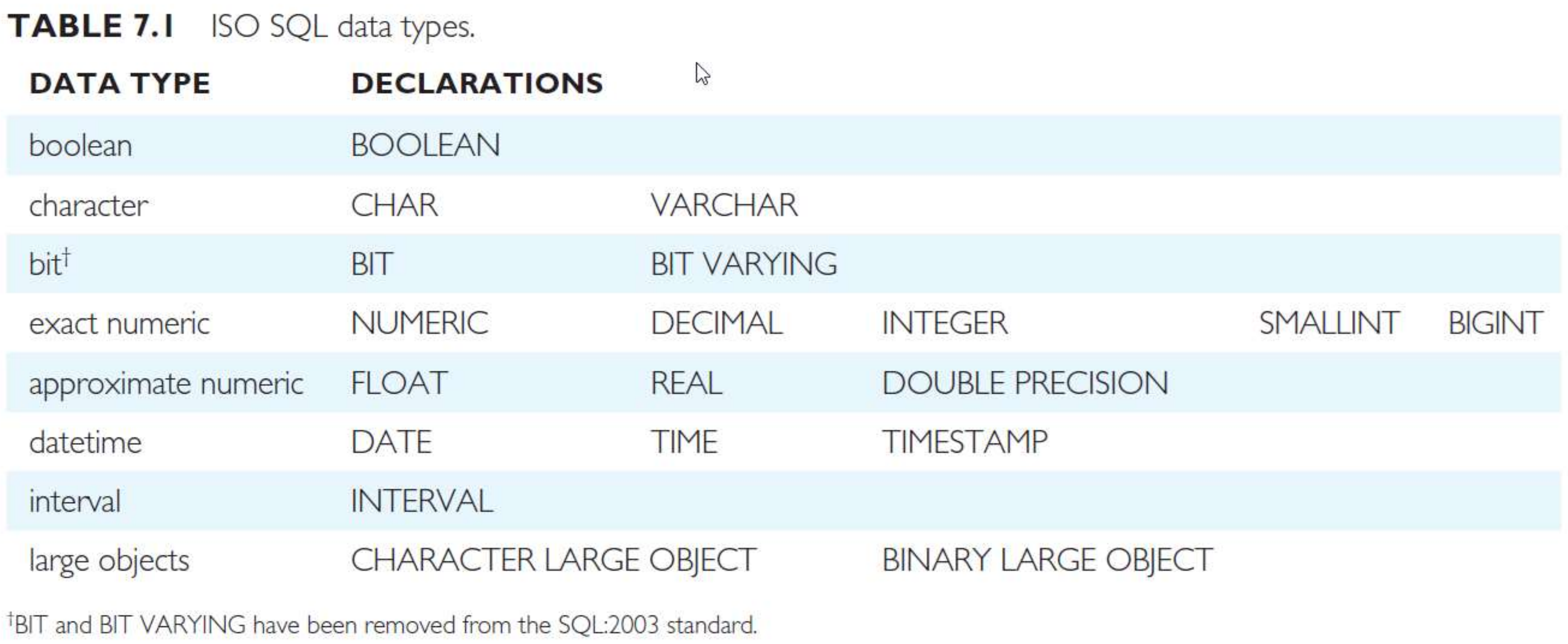


TABLE 7.1 ISO SQL data types.

DATA TYPE	DECLARATIONS				
boolean	BOOLEAN				
character	CHAR	VARCHAR			
bit[†]	BIT	BIT VARYING			
exact numeric	NUMERIC	DECIMAL	INTEGER	SMALLINT	BIGINT
approximate numeric	FLOAT	REAL	DOUBLE PRECISION		
datetime	DATE	TIME	TIMESTAMP		
interval	INTERVAL				
large objects	CHARACTER LARGE OBJECT		BINARY LARGE OBJECT		

[†]BIT and BIT VARYING have been removed from the SQL:2003 standard.

Source: Connolly, 2015

Constraints

- Types of constraints:
  - **PRIMARY KEY**
  - **FOREIGN KEY**
  - **UNIQUE**
    - Ensure all data values stored in the column are unique.
    - Differ from PRIMARY KEY since it allows NULL values.
  - **CHECK**
    - Checks a specific condition during the execution of the query.
  - **NOT NULL**

Examples (CREATE TABLE)

- Given the structure of the Department table as follows:

Department

Attribute	Datatype
deptNo (Primary Key)	Number – precision(5)
deptName – requires values	Varchar(20)
Address	Varchar(30)
City	Varchar(15)

Employees

Attribute	Datatype
empID (Primary Key)	Number
fName – requires values	Varchar(20)
lName – requires values	Varchar(30)
deptNo (Foreign key) – deptNo is an attribute in Department relation	Number(5)

Examples (CREATE TABLE)

- Create table without constraint:

```
CREATE TABLE Department (
        deptNo NUMBER(5),
        deptName VARCHAR(20),
        address VARCHAR(30),
        city VARCHAR(15)
);
```

```
CREATE TABLE Employees (
        empID NUMBER,
        fName VARCHAR(10),
        lName VARCHAR(20),
        deptNo NUMBER(5)
);
```

- In this case, constraints (e.g.: primary key) can be added using **ALTER TABLE** command.

Examples (CREATE TABLE)

- Create table with constraints:

```
CREATE TABLE Department (
        deptNo NUMBER(5) PRIMARY KEY,
        deptName VARCHAR(20) NOT NULL,
        address VARCHAR(30),
        city VARCHAR(15)
);
```

```
CREATE TABLE Employees (
        empID NUMBER PRIMARY KEY,
        fName VARCHAR(10) NOT NULL,
        lName VARCHAR(20) NOT NULL,
        deptNo NUMBER(5) REFERENCES Department(deptNo)
);
```

Altering Tables

- To change the structure of a table once it has been created.
- General Syntax:

```
ALTER TABLE tableName
ADD columnName dataType );
```

Add new column

```
ALTER TABLE tableName
ADD CONSTRAINT constraintName
constraintType(columnName);
```

Add new constraint

```
ALTER TABLE tableName
MODIFY columnName newdataType/size/defaultvalue;
```

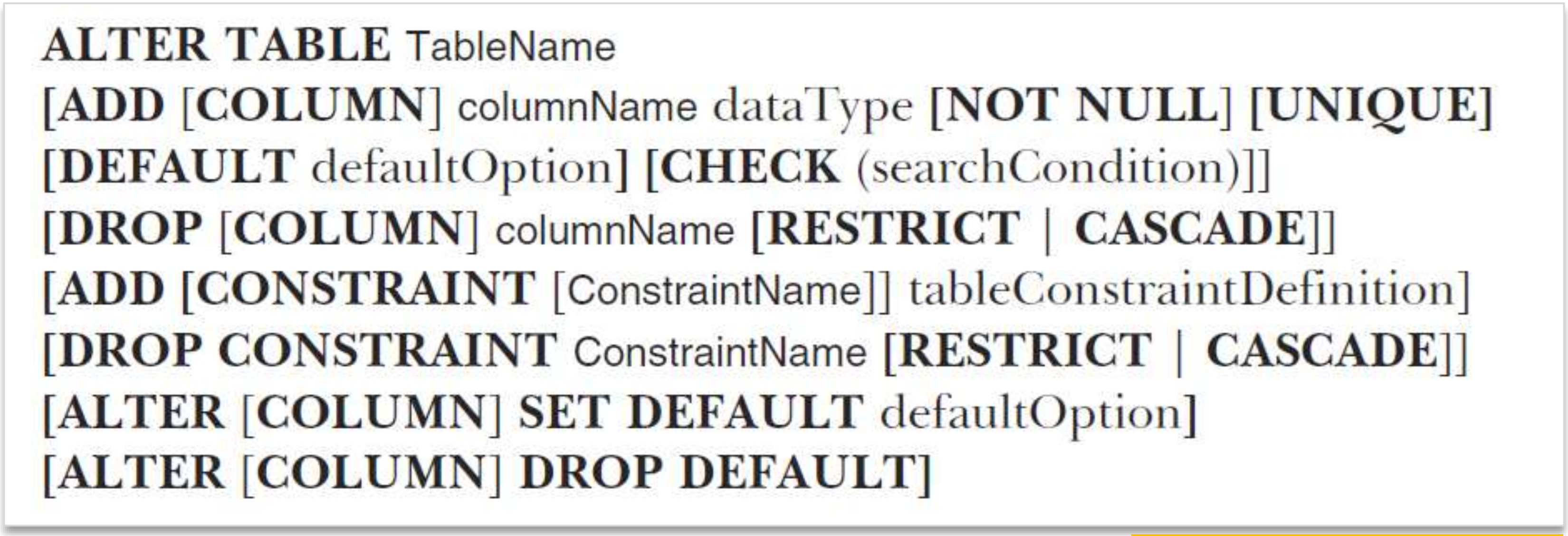
Change data type, data size, default values, constraints

```
ALTER TABLE tableName
DROP COLUMN columnName;
```

Delete column

Altering Tables

- General syntax for altering tables



```
ALTER TABLE TableName
[ADD [COLUMN] columnName dataType [NOT NULL] [UNIQUE]
[DEFAULT defaultOption] [CHECK (searchCondition)]]
[DROP [COLUMN] columnName [RESTRICT | CASCADE]]
[ADD [CONSTRAINT [ConstraintName]] tableConstraintDefinition]
[DROP CONSTRAINT ConstraintName [RESTRICT | CASCADE]]
[ALTER [COLUMN] SET DEFAULT defaultOption]
[ALTER [COLUMN] DROP DEFAULT]
```

Source: Connolly, 2015

[ ] : optional

ALTER TABLE - Examples

- **Add** new column “Address” into Employee table.

```
ALTER TABLE Employee
ADD Address VARCHAR(40);
```

- **Change** datatype of “Address” to VARCHAR(20) and set the column as NOT NULL

```
ALTER TABLE Employee
MODIFY Address VARCHAR(20) NOT NULL;
```

- **Delete** the column “Address”

```
ALTER TABLE Employee
DROP COLUMN Address;
```

ALTER TABLE - Examples

- Adding constraint after a table has been created.
- Set the attribute of "EmployeeID" as the PRIMARY KEY for the Employee table.

```
ALTER TABLE Employee
ADD CONSTRAINT pk_Employee
PRIMARY KEY(EmployeeID);
```

Optional

Examples (ALTER TABLE)

- If both tables are created without constraint.
- Constraints must be added using **ALTER TABLE**

Department

Attribute	Datatype
deptNo (Primary Key)	Number – precision(5)
deptName – requires values	Varchar(20)
Address	Varchar(30)
City	Varchar(15)

Employees

Attribute	Datatype
empID (Primary Key)	Number
fName – requires values	Varchar(20)
lName – requires values	Varchar(30)
deptNo (Foreign key) – deptNo is an attribute in Department relation	Number(5)

Examples (ALTER TABLE)

- For Department table, primary key and the NOT NULL constraint for deptName attribute

```
ALTER TABLE Department
ADD CONSTRAINT pk_Department PRIMARY KEY(deptNo)
MODIFY deptName VARCHAR(20) NOT NULL;
```

- For Employees table, primary key, NOT NULL constraints and foreign key to be added.

```
ALTER TABLE Employees
MODIFY fName VARCHAR(20) NOT NULL
MODIFY lName VARCHAR(30) NOT NULL
ADD CONSTRAINT pk_Employees PRIMARY KEY(empID)
ADD CONSTRAINT fk_EmpDept FOREIGN KEY (deptNo)
REFERENCES Department(deptNo);
```

Deleting Table

- General Syntax:

```
DROP TABLE tableName;
```

- Example: delete the table named “Employee”

```
DROP TABLE Employee;
```

Restore Dropped Table

- Dropped tables are actually placed in a recycle bin and can be restored.
- Syntax:

```
FLASHBACK TABLE tableName TO BEFORE DROP;
```

- Supported only in Oracle.

Other Useful Statements

- Viewing list of tables in a user schema:
- USER_TABLES: oracle’s data dictionary object that stores information of tables created under a specific user (schema)
- To view the list of tables:

```
SELECT TABLE_NAME FROM USER_TABLES;
```

- Supported in Oracle.

Other Useful Statements

- Viewing table structures using **DESC** or **DESCRIBE.**
  - Syntax:
    ```
    DESCRIBE tableName;
    ```
  - Example: View the table structure of Employee table.

```
DESCRIBE Employee;
```

- Rename a table using **RENAME**
  - Syntax:
    ```
    RENAME tableName TO newtableName;
    ```
  - Example: Rename the Employee table to “Workers”

```
RENAME Employee TO Workers;
```

Other Useful Statements

- Create table from existing database tables, using the **AS** clause and subqueries
- Syntax:

```
CREATE TABLE tableName [(columnName,…)]
AS (…subquery…);
```

- Example: create a new table based on the HR Employee table in Oracle.

```
CREATE TABLE NewEmployee
AS (SELECT * FROM hr.Employees);
```

Subquery

Exercise