

# Defining Relational Schema in SQL

Hui Chen <sup>a</sup>

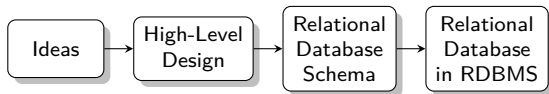
<sup>a</sup>CUNY Brooklyn College, Brooklyn, NY, USA

February 22, 2022

# Outline

- 1 Introduction to SQL
- 2 Data Types in SQL
- 3 Table Declaration
- 4 Dealing with “Regrets”
- 5 Getting Help
- 6 Assignment

# Overview



# Outline

- 1 Introduction to SQL
- 2 Data Types in SQL
- 3 Table Declaration
- 4 Dealing with “Regrets”
- 5 Getting Help
- 6 Assignment

# Introduction to SQL

SQL (pronounced as “sequel”) is the principal language used to describe and manipulate relational database, and has several aspects:

- ▶ Data definition language (DDL).
  - ▶ SQL includes commands to create database objects such as tables, indexes, and views, as well as commands to define access rights to those database objects.
  - ▶ Topics of this lecture: SQL commands to create database tables (relations)
- ▶ Data manipulation language (DML).
  - ▶ SQL includes commands to insert, update, delete, and retrieve data within the database tables.
- ▶ Transaction control language (TCL).
  - ▶ The DML commands in SQL are executed within the context of a transaction.
- ▶ Data control language (DCL).
  - ▶ Data control commands are used to control access to data objects.

# Relations in SQL

SQL makes a distinction between three kinds of relations:

- ▶ Stored relations (also called tables) – a relation that exists in the database and that can be modified by changing its tuples, as well as queried.
- ▶ Views – relations defined by a computation, they are not stored, but are constructed, in whole or in part, when needed.
- ▶ Temporary tables – constructed by the SQL language processor when it performs its job of executing queries and data modifications, and then thrown away and not stored.

Stored relations or tables are today's topic.

# Outline

- 1 Introduction to SQL
- 2 Data Types in SQL**
- 3 Table Declaration
- 4 Dealing with “Regrets”
- 5 Getting Help
- 6 Assignment

# Data Types

All attributes must have data types

- ▶ Character types
- ▶ Numeric types
- ▶ Datetime types
- ▶ Binary types
- ▶ National character types
- ▶ Interval type
- ▶ Boolean
- ▶ XML
- ▶ JSON



# Character Types

## Character or string types

SQL Data Type	Description
CHAR(n)	a fixed-length string of up to n characters, a typical implementation pads a short string to n characters
VARCHAR(n)	a string of up to n characters, a typical implementation may have end-marker or length, but occupies n characters
VARCHAR2(n)	no unused spaces, some RDBMS treats (converts) VARCHAR(n) to code VARCHAR2(n)
NCHAR(n)	like CHAR(n), in unicode
NVARCHAR(n)	like VARCHAR(n), in unicode
NVARCHAR2(n)	like VARCHAR2(n), in unicode
TEXT	a string (text) of up to $2^{16} - 1 = 65,535$ characters, non-standard with wide support

# Numeric Types

SQL Data Type	Description
INTEGER or INT	whole numbers, typical implementations are signed 32-bit integers
SMALLINT	whole numbers, typical implementations are signed 16-bit integers
DECIMAL(L,D)	A packed “exact” fixed-point number. L is the total number of digits and D is the number of digits after the decimal point.
NUMBER(L,D)	synonym to DECIMAL(L,D)
NUMERIC(L,D)	synonym to DECIMAL(L,D)
FLOAT	a single-precision floating point number
DOUBLE PRECISION	a double-precision floating point number
DOUBLE	synonym to DOUBLE PRECISION
REAL	a floating point number, some implementations treat it as a synonym to DOUBLE

# Date and Time Types

SQL Data Type	Description
Date	a date, i.e., typically YYYY-MM-DD
Time	a time, i.e., typically HH:MM:SS.ssssss
DATETIME	A date and time combination
TIMESTAMP	a timestamp, i.e., typically, YYYY-MM-DD HH:MM:SS.ssssss

# Boolean and Bit String Types

SQL Data Type	Description
BOOLEAN	TRUE, FALSE
BIT	Bit strings are strings of 1's and 0's

# Data Types in Example RDBMS

Check out

- ▶ MariaDB
- ▶ PostgreSQL
- ▶ Microsoft SQL Server
- ▶ Oracle DBMS
- ▶ IBM DB2

and

- ▶ SQLite3
- ▶ Microsoft Access

# Outline

- 1 Introduction to SQL
- 2 Data Types in SQL
- 3 Table Declaration**
- 4 Dealing with “Regrets”
- 5 Getting Help
- 6 Assignment

# Table Declaration in SQL

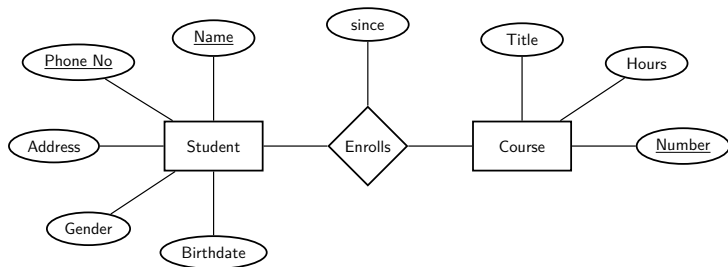
Use CREATE TABLE command

Syntax

```
CREATE TABLE tablename (  
column1 datatype [constraint] [,  
column2 datatype [constraint] ] [,  
PRIMARY KEY (column1 [, column2]) ] [,  
FOREIGN KEY (column1 [, column2]) REFERENCES tablename]  
CONSTRAINT constraint ] );
```

# Table Declaration for a Database in SQL: Example

Consider the following database model



whose relational database schemas are,

```
Students(name:string, phone:string, address:string,
gender:string, birthdate:date)
```

```
Courses(name:string, title:string, number, hours:integer)
```

```
Enrollment(sname:string, sphone:string, cnumber:string,
since:datetime)
```



## Declarations in SQL: Example

```
CREATE TABLE Students (  
    name CHAR(30),  
    phone CHAR(10),  
    address VARCHAR(255),  
    gender CHAR(1),  
    birthdate DATE,  
    -- primary key constraint  
    PRIMARY KEY (name, phone)  
);  
  
CREATE TABLE Courses (  
    title VARCHAR(255),  
    -- also primary key constraint  
    idnum CHAR(5) PRIMARY KEY,  
    hours integer  
);
```

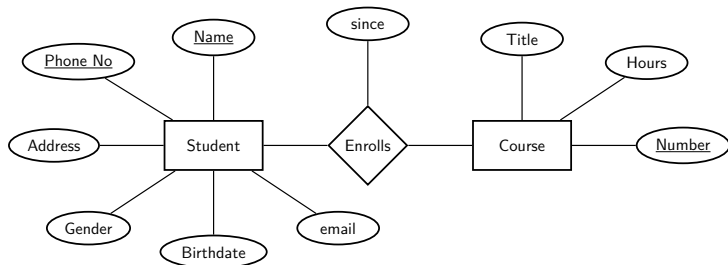
## Declarations in SQL: Example

```
CREATE TABLE Enrollment(  
  sname CHAR(30),  
  sphone CHAR(10),  
  -- foreign key constraint  
  cidnum CHAR(5) REFERENCES Courses ( idnum ),  
  since DATETIME,  
  -- primary key constraint  
  PRIMARY KEY(sname, sphone, cidnum),  
  -- also foreign key constraint  
  FOREIGN KEY(sname, sphone)  
    REFERENCES Students (name, phone));
```

In SQL, -- leads a comment

## Table Declaration for a Database in SQL: Revisiting the Example

Let's revisit our example database and consider that we add an attribute to Students entity set.



where the business rule indicates that email addresses must be unique.

# Declaration of Students in SQL: Example

```
CREATE TABLE Students (  
  name CHAR(30),  
  phone CHAR(10),  
  address VARCHAR(255),  
  gender CHAR(1),  
  email VARCHAR(255),  
  birthdate DATE,  
  -- primary key constraint  
  PRIMARY KEY (name, phone),  
  UNIQUE (email)  
);
```

## Declaration of Students in SQL: Example

Since the alternate key has a single attribute, alternatively we can declare it as,

```
CREATE TABLE Students (  
  name CHAR(30),  
  phone CHAR(10),  
  address VARCHAR(255),  
  gender CHAR(1),  
  email VARCHAR(255) UNIQUE,  
  birthdate DATE,  
  -- primary key constraint  
PRIMARY KEY (name, phone)  
);
```

## Describing Table Definition

This is MariaDB specific. To show a table definition, use “DESCRIBE table\_name,” for example,

```
DESCRIBE Students;
```

## Constraints in the Example

- ▶ Primary Key constraint
  - ▶ Single-attribute key?
  - ▶ Multi-attribute key?
- ▶ Unique constraint (alternate key)
  - ▶ Single-attribute key?
  - ▶ Multi-attribute key?
- ▶ Foreign Key constraint
  - ▶ Single-attribute key?
  - ▶ Multi-attribute key?

# Outline

- 1 Introduction to SQL
- 2 Data Types in SQL
- 3 Table Declaration
- 4 Dealing with “Regrets”**
- 5 Getting Help
- 6 Assignment



## Removing Tables

To remove tables and the data, use "DROP TABLE table\_name."

```
DROP TABLE Enrollment;
```

What would happen if we attempted to remove Students? Why?

## Change Structure of Existing Tables

To change the structure of an existing table, use "ALTER TABLE ..."

- ▶ Add or delete columns
- ▶ Change the type of existing columns
- ▶ Rename columns or the table itself
- ▶ More ...

## Adding or Deleting Columns

To delete a column, use "ALTER TABLE table\_name DROP column\_name."

```
ALTER TABLE Students DROP gender;
```

To add a column, use "ALTER TABLE table\_name ADD column\_name data\_type."

```
ALTER TABLE Students ADD gender CHAR(1);
```

## Changing Type of Existing Columns

To change the type of an existing column, use "ALTER TABLE table\_name MODIFY column\_name data\_type"

```
ALTER TABLE Students MODIFY gender VARCHAR(10);
```

## Renaming Existing Columns

To rename an existing column, use "ALTER TABLE table\_name CHANGE old\_column\_name new\_column\_name data\_type"

```
ALTER TABLE Students CHANGE birthdate birthday DATE;
```

## Renaming Existing Tables

To rename an existing table, use "ALTER TABLE table\_name RENAME TO new\_table\_name"

```
ALTER TABLE Students RENAME TO Students2;
```

# Outline

- 1 Introduction to SQL
- 2 Data Types in SQL
- 3 Table Declaration
- 4 Dealing with “Regrets”
- 5 Getting Help**
- 6 Assignment

## Getting Help

Other than looking up the documentation online or from a book, we can use the “help” command – this is MariaDB specific.

```
HELP ALTER TABLE ;
```

```
HELP CREATE TABLE ;
```



# Outline

- 1 Introduction to SQL
- 2 Data Types in SQL
- 3 Table Declaration
- 4 Dealing with “Regrets”
- 5 Getting Help
- 6 Assignment**

# Assignment

Let's work on an assignment using paper and pencil/pen ...