Creating and Dropping Tables

The complete grammar for the CREATE TABLE expression is 160 lines long and can be found at https://dev.mysql.com/doc/refman/8.0/en/create-table.html. We omit much of the grammar here as many of the options are beyond the scope of this course. For us, the pertinent parts of the grammar are below.

```
CREATE TABLE [IF NOT EXISTS] tbl_name
   (create_definition, ... )

create_definition: {
     col_name column_definition
     | PRIMARY KEY (key_part,...)
     | FOREIGN KEY (col_name,...)
                    reference_definition
}

column_definition: {
     data_type
     [NOT NULL | NULL]
     [DEFAULT {literal | (expr)} ]
              [AUTO_INCREMENT]
     [UNIQUE [KEY]]
     [[PRIMARY] KEY]
}

key_part: {col_name | (expr)}

reference_definition:
   REFERENCES tbl_name (key_part,...)
     [ON DELETE reference_option]
     [ON UPDATE reference_option]

reference_option:
   RESTRICT | CASCADE | SET NULL | NO ACTION | SET DEFAULT

data_type:
   (see Chapter 11, Data Types)
```

The grammar for DROP TABLE is below.

```
DROP TABLE [IF EXISTS]
   tbl_name [, tbl_name] ...
   [RESTRICT | CASCADE]
```

Using CREATE TABLE

The following steps are used to create a table in a database.

1. Determine the table name.
2. Determine the column names and definitions (types and attributes)
3. Specify multi-column primary keys (if any)
4. Specify foreign keys (if any)

Determine the Table Name

- Table names must be unique within the database.
- Capitalize the table name and use camelCasing
- By convention we include _T as a suffix to the table name.

Determine the Column Names and Definitions (Types and Attributes)

Names

- No two columns in the same table can have the same name.
- By convention we do not capitalize column names, but do use cameCasing for multi-word names.

Types

Determine the appropriate type for each column. See Chapter 11, Data Types.

Attributes

[NOT NULL | NULL]

Specify NOT NULL, If the column value in a row cannot be NULL.

- Conceptually, NULL means “a missing or unknown value”.
- Null does not equal 0! So you may enter a value of zero in a column that specifies NOT NULL.
- NULL values are allowed by default.

[DEFAULT {literal | (expr)}]

Include a DEFAULT clause if a value is automatically inserted in a column when a row is inserted without a value for the column.

[AUTO_INCREMENT]

Include an AUTO_INCREMENT clause if the value in the column should be inserted automatically when a row is added to the table.

- Can only be applied to integer and floating point types.
- When inserting a row, specify NULL for the value of the column.
- There can be only one column per table that is auto incremented and it cannot have a default value.
- See 3.6.9 Using AUTO_INCREMENT for more details.

[UNIQUE [KEY]]

Specify the UNIQUE attribute if each row should have a unique value for the column.

- Key is a synonym for index and is optional.
- An error occurs if you try to add a new row with a key value that matches an existing row.
- A UNIQUE index permits multiple NULL values for columns that can contain NULL.

[[PRIMARY] KEY]

If the table's primary key consists of a single column, you can identify the column as the primary key by using the PRIMARY KEY attribute in the column definition.

- If the table's primary key consists of multiple columns, you *must* use a PRIMARY KEY clause (see below).

Example

```
CREATE TABLE Users_T(
    userId INT AUTO_INCREMENT PRIMARY KEY,
    userName VARCHAR(40),
    careerSales INT DEFAULT 0
);
```

PRIMARY KEY clause

A PRIMARY KEY is a column or a set of columns that uniquely identifies each row in the table. The following are the rules that you must follow when you define a primary key for a table:

- A primary key must contain unique values.
- If the primary key consists of multiple columns, the combination of values in these columns must be unique.
- A primary key column cannot contain NULL values but you do not need to declare this. It is implied.
- A primary key column **should** be an INT or floating point type.
- If you do not specify a primary key, MySQL will return the first UNIQUE index (column) that has no NULL columns as the PRIMARY KEY.

A primary key column often has AUTO_INCREMENT attribute that generates a unique sequence for the key automatically. The primary key of the next row is one greater than the previously inserted row.

Below we create a table using the PRIMARY KEY column attribute.

```
CREATE TABLE Users_T(
    userId INT AUTO_INCREMENT PRIMARY KEY,
    userName VARCHAR(40)
);
```

The example below demonstrates the user of the PRIMARY KEY clause. PRIMARY KEY clauses are specified *after* the column definitions.

```
CREATE TABLE Users_T(
    userId INT AUTO_INCREMENT,
    userName VARCHAR(40),
    PRIMARY KEY(userId)
);
```

FOREIGN KEY Clause

A foreign key is a column in a child table that is part of the primary key in a parent table.

We use reference definitions to specify the the column in parent table that the data in the foreign key column comes from.

```
CREATE TABLE UserRoles_T(
    userId INT NOT NULL,
    roleId INT NOT NULL,
    PRIMARY KEY(userId, roleId),
    FOREIGN KEY(userId) REFERENCES Users_T(userId),
    FOREIGN KEY(roleId) REFERENCES Position_T(roleId)
);
```

Foreign key constraints help keep data consistent between child tables and parent tables. A foreign key constraint is defined on the child table.

MySQL allows foreign key constraints on delete and update operations.

```
CREATE TABLE Parent_T (
   id INT NOT NULL,
   PRIMARY KEY (id)
);
```

```
CREATE TABLE Child_T (
   id INT,
   parentId INT,
     FOREIGN KEY (parentId)
       REFERENCES Parent_T(id)
       ON DELETE CASCADE
);
```

When an UPDATE (row) or DELETE (row) **operation on the parent table** affects a key value that has matching rows in the child table, the success of the operation depends on the ON UPDATE or ON DELETE subclauses of the FOREIGN KEY clause listed in the child table definition.

The possible actions are defined in the reference_options definition:

RESTRICT | CASCADE | SET NULL | NO ACTION

RESTRICT: Rejects the delete or update operation on the parent table.

CASCADE: Delete or update the row from the parent table, and automatically delete or update the matching rows in the child table.

SET NULL: Delete or update the row from the parent table, and set the foreign key column or columns in the child table to NULL.

For storage engines supporting foreign keys, MySQL rejects any INSERT or UPDATE operation that attempts to create a foreign key value in a child table if there is no a matching candidate key value in the parent table. In other words we have to make sure an entry exists in the parent table first, then add an entry in the child table.

See 13.1.20.5 FORIGN KEY Constraints for more information.