Chapter 17

How to use SQL to create a database

Objectives

Applied

1. Given the design for a database, create a SQL script that will create the database, including all tables, primary keys, foreign key constraints, and indexes.
2. Use SQL statements to create users and assign privileges to the users.
3. Load data into a database table from a text file.
4. Dump a database to a SQL script.

Objectives (continued)

Knowledge

1. Describe the use of the DDL statements for creating, altering, and dropping databases, tables, and indexes.
2. Describe the column definitions for a table in terms of data types and these attributes: unique, not null, default, primary key, auto-increment, and references. Also, describe a table-level definition for a primary key and a foreign key constraint.
3. Describe the use of the DDL statements for creating, renaming, and dropping users and for assigning and revoking privileges.
4. Describe the process of loading data from a text file into a table, and the process of dumping a database to a SQL script.
5. Describe the use of a script for creating a database.

How to create a database

```
CREATE DATABASE my_guitar_shop2;
```

How to create a database only if it does not exist

```
CREATE DATABASE IF NOT EXISTS my_guitar_shop2;
```

How to select a database

```
USE my_guitar_shop2;
```

How to drop a database

```
DROP DATABASE my_guitar_shop2;
```

How to drop a database only if it exists

```
DROP DATABASE IF EXISTS my_guitar_shop2;
```


Common numeric data types

```
INT[(size)]
TINYINT[(size)]
DECIMAL[(p[,s])]
```

Common string data types

```
VARCHAR(size)
CHAR[(size)]
TEXT
```

Common date and time data types

```
DATE
TIME
DATETIME
```


The syntax of the CREATE TABLE statement

```
CREATE TABLE [IF NOT EXISTS] tableName
(
     columnName1  dataType  [columnAttributes][,
     columnName2  dataType  [columnAttributes]][,
     columnName3  dataType  [columnAttributes]]...
)
```

Three common column attributes

```
UNIQUE
NOT NULL
DEFAULT default_value
```

A table without column attributes

```
CREATE TABLE customers
(
  customerID  INT,
  firstName   VARCHAR(60),
  lastName    VARCHAR(60)
);
```

A table with column attributes

```
CREATE TABLE customers
(
  customerID  INT           NOT NULL   UNIQUE,
  firstName   VARCHAR(60)   NOT NULL,
  lastName    VARCHAR(60)   NOT NULL
);
```


Another table with column attributes

```
CREATE TABLE orders
(
  orderID      INT           NOT NULL   UNIQUE,
  customerID   INT           NOT NULL,
  orderNumber  VARCHAR(50)   NOT NULL,
  orderDate    DATE          NOT NULL,
  orderTotal   DECIMAL(9,2)  NOT NULL,
  paymentTotal DECIMAL(9,2)             DEFAULT 0
);
```


Two column attributes for working with a primary key

```
PRIMARY KEY
AUTO_INCREMENT
```

A table with a column-level primary key

```
CREATE TABLE customers (
  customerID    INT           NOT NULL   PRIMARY KEY
                                         AUTO_INCREMENT,
  emailAddress  VARCHAR(255)  NOT NULL   UNIQUE
);
```

A table with a table-level primary key

```
CREATE TABLE customers (
  customerID    INT           NOT NULL   AUTO_INCREMENT,
  emailAddress  VARCHAR(255)  NOT NULL   UNIQUE,

  PRIMARY KEY (customerID)
);
```


A table with a two-column primary key

```
CREATE TABLE orderItems (
  orderID         INT            NOT NULL,
  productID       INT            NOT NULL,
  itemPrice       DECIMAL(10,2)  NOT NULL,
  discountAmount  DECIMAL(10,2)  NOT NULL,
  quantity        INT            NOT NULL,

  PRIMARY KEY (orderID, productID)
);
```


Three attributes for working with a foreign key

```
CONSTRAINT
FOREIGN KEY
REFERENCES
```

A table with a column-level foreign key constraint

```
CREATE TABLE orders
(
  orderID      INT        PRIMARY KEY,
  customerID   INT        NOT NULL
                          REFERENCES customers (customerID),
  orderDate    DATETIME   NOT NULL
)
```


A table with a table-level foreign key constraint

```
CREATE TABLE orders
(
  orderID       INT          PRIMARY KEY,
  customerID    INT          NOT NULL,
  orderDate     DATETIME     NOT NULL,
  CONSTRAINT ordersFkCustomers
    FOREIGN KEY (customerID)
      REFERENCES customers (customerID)
)
```

A constraint that uses the ON DELETE clause

```
CONSTRAINT ordersFkCustomers
  FOREIGN KEY (customerID) REFERENCES customers (customerID)
  ON DELETE CASCADE
```

An insert statement that fails because a related row doesn't exist

```
INSERT INTO orders
VALUES (1, 999, '2017-08-03')
```

The response from the system

```
Error Code: 1452. Cannot add or update a child row: a
foreign key constraint fails ('ex'.'orders', CONSTRAINT
'ordersFkCustomers' FOREIGN KEY ('customerID') REFERENCES
'customers' ('customerID'))
```


A statement that renames a table

```
ALTER TABLE products RENAME TO product;
```

A statement that adds a new column at the end of the table

```
ALTER TABLE customers ADD lastTransactionDate DATE;
```

A statement that adds a new column after a specified column

```
ALTER TABLE customers ADD lastTransactionDate DATE
AFTER emailAddress;
```

A statement that drops a column

```
ALTER TABLE customers DROP lastTransactionDate;
```

A statement that renames a column

```
ALTER TABLE customers
CHANGE emailAddress email VARCHAR(255) NOT NULL UNIQUE;
```


A statement that changes a column definition

```
ALTER TABLE customers MODIFY firstName VARCHAR(100) NOT NULL
```

A statement that changes a column's data type

```
ALTER TABLE customers MODIFY firstName CHAR(100) NOT NULL;
```

A statement that may cause data to be lost

```
ALTER TABLE customers MODIFY firstName VARCHAR(8);
```

A statement that sets a column's default value

```
ALTER TABLE customers ALTER firstName SET DEFAULT '';
```

A statement that drops a column's default value

```
ALTER TABLE customers ALTER firstName DROP DEFAULT;
```

Warning

- You should never alter a table or other database object in a production database without first consulting the DBA.

A statement that drops a table

```
DROP TABLE customers;
```

A statement that drops a table if it exists

```
DROP TABLE IF EXISTS customers;
```

Warning

- You should never drop a table in a production database without first consulting the DBA, but you probably won't have the privileges for doing that.

The syntax of the CREATE INDEX statement

```
CREATE [UNIQUE] INDEX|KEY indexName
ON tableName (columnName1 [ASC|DESC]
              [, columnName2 [ASC|DESC]]...)
```

A statement that…

Creates an index based on a single column

```
CREATE INDEX customerID
ON orders (customerID);
```

Creates a unique index

```
CREATE UNIQUE INDEX emailAddress
ON customers (emailAddress);
```

Creates an index based on two columns

```
CREATE UNIQUE INDEX customerIDorderNumber
ON orders (customerID, orderNumber);
```

Creates an index that's sorted in descending order

```
CREATE INDEX orderTotal
ON orders (orderTotal DESC);
```


A CREATE TABLE statement that creates indexes

```
CREATE TABLE customers (
  customerID    INT           NOT NULL   AUTO_INCREMENT,
  emailAddress  VARCHAR(255)  NOT NULL,
  firstName     VARCHAR(60)   NOT NULL,

  PRIMARY KEY (customerID),
  UNIQUE INDEX emailAddress (emailAddress),
  INDEX firstName (firstName)
);
```

A DROP INDEX statement that drops an index

```
DROP INDEX firstName ON customers;
```

Privileges for working with data

- SELECT
- INSERT
- UPDATE
- DELETE

Privileges for modifying the database structure

- CREATE
- ALTER
- DROP
- INDEX

Other privileges

- CREATE USER
- ALL [PRIVILEGES]
- GRANT OPTION
- USAGE

The four privilege levels

Level	Example
Global	*.*
Database	music_db.*
Table	music_db.products
Column	(listPrice) music_db.products

How to create a user from a specific host

```
CREATE USER joel@localhost IDENTIFIED BY 'sesame';
```

How to create a user from any host

```
CREATE USER dba IDENTIFIED BY 'sesame';
```

How to rename a user from a specific host

```
RENAME USER joel@localhost TO joelmurach@localhost;
```

How to change a user's password

```
GRANT USAGE ON *.*
TO joelmurach@localhost
IDENTIFIED BY 'newpassword';
```

How to drop a user from a specific host

```
DROP USER joelmurach@localhost;
```

How to drop a user from any host

```
DROP USER dba;
```


The syntax of the GRANT statement

```
GRANT privilegeList
ON [dbName.]table
TO userName1 [IDENTIFIED BY 'password1']
[, userName2 [IDENTIFIED BY 'password2'] ...]
[WITH GRANT OPTION]
```

A statement that uses the current database (no database name is specified)

```
GRANT SELECT, INSERT, UPDATE, DELETE
ON customers TO joel@localhost;
```


A statement that creates a user with no privileges

```
GRANT USAGE
ON *.*
TO joel@localhost IDENTIFIED BY 'sesame';
```

A statement that creates a user with database privileges

```
GRANT SELECT, INSERT, UPDATE, DELETE
ON my_guitar_shop2.*
TO mgs_user@localhost IDENTIFIED BY 'pa55word';
```

A statement that creates a user with global privileges

```
GRANT ALL
ON *.*
TO dba IDENTIFIED BY 'supersecret'
WITH GRANT OPTION;
```

A statement that grants table privileges to a user

```
GRANT SELECT, INSERT, UPDATE
ON my_guitar_shop2.products TO joel@localhost;
```

A statement that grants database privileges to a user

```
GRANT SELECT, INSERT, UPDATE
ON my_guitar_shop2.* TO joel@localhost;
```

A statement that grants global privileges to a user

```
GRANT SELECT, INSERT, UPDATE
ON *.* TO joel@localhost;
```

A statement that grants column privileges to a user

```
GRANT SELECT (productCode, productName, listPrice),
      UPDATE (description)
ON my_guitar_shop2.products TO joel@localhost
```


The syntax of the REVOKE statement for all privileges

```
REVOKE ALL[ PRIVILEGES], GRANT OPTION
FROM user [, user]
```

A statement that revokes all privileges from a user

```
REVOKE ALL, GRANT OPTION
FROM dba;
```

A statement that revokes all privileges from multiple users

```
REVOKE ALL, GRANT OPTION
FROM dba, joel@localhost;
```


The syntax of the REVOKE statement for specific privileges

```
REVOKE privilegeList
ON [dbName.]table
FROM user [, user]
```

A statement that revokes specific privileges from a user

```
REVOKE UPDATE, DELETE
ON my_guitar_shop2.customers FROM joel@localhost
```


A statement that lists all users

```
SELECT User, Host from mysql.user;
```

The result set

User	Host
dba	%
joel	localhost
mgs_tester	localhost
mgs_user	localhost
pma	localhost
root	localhost

The syntax of the SHOW GRANTS statement

```
SHOW GRANTS [FOR user]
```

A statement that shows the privileges for a user from any host

```
SHOW GRANTS FOR dba;
```

The result set

Grants for dba@%
GRANT ALL PRIVILEGES ON *.* TO 'dba'@'%' IDENTIFIED BY PASSWORD '*90BA3AC0BFDE07AE334CA523CB27167AE33825B9' WITH GRANT OPTION

A statement that shows the privileges for a user from a specific host

```
SHOW GRANTS FOR mgs_user@localhost;
```

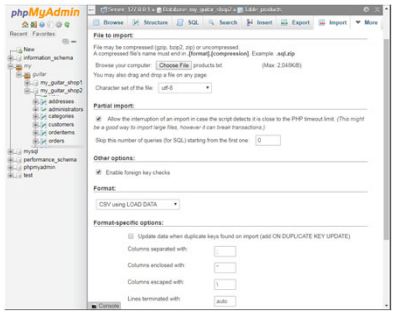
The result set

Grants for mgs_user@localhost
GRANT USAGE ON *.* TO 'mgs_user'@'localhost' IDENTIFIED BY PASSWORD '*F71B0AF6B232C58021B6AC63A29FCF13A4E46E59'
GRANT SELECT, INSERT, UPDATE, DELETE ON 'my_guitar_shop1'.* TO 'mgs_user'@'localhost'
GRANT SELECT, INSERT, UPDATE, DELETE ON 'my_guitar_shop2'.* TO 'mgs_user'@'localhost'

A statement that shows the privileges for the current user

```
SHOW GRANTS;
```

The Import tab for a table named products



A tab-delimited text file that's stored in users.txt

```
1   John      Smith       jsmith@gmail.com
2   Andrea    Steelman    andi@murach.com
3   Joel      Murach      joelmurach@yahoo.com
```

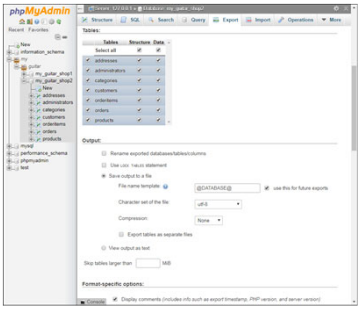
Using phpMyAdmin to load data from a text file

1. Start phpMyAdmin, select the database, select the table, and click on the Import tab.
2. Select the file to import.
3. Set the options for the import and click the Go button. If you get an error, you can modify the import options and try again.

How to use the Windows command prompt to load data from a text file

```
cd \xampp\mysql\bin
mysql -u root -p
Enter password: ******
use my_guitar_shop2;
load data local infile "c:/murach/products.txt"
into table products;
exit;
```


The Export tab for my_guitar_shop2



How to use phpMyAdmin to dump a database

1. Start phpMyAdmin, select the database, and click on the Export tab.
2. Set the options for the SQL script file.
3. Click on the Go button and save the file.

How to use the Windows command prompt to dump a database to a SQL script

```
cd \xampp\mysql\bin
mysqldump -u root -p my_guitar_shop2 > my_guitar_shop2.sql
Enter password: ******
```


The SQL script that creates the my_guitar_shop2 database

```
-- create and select the database
DROP DATABASE IF EXISTS my_guitar_shop2;
CREATE DATABASE my_guitar_shop2;
USE my_guitar_shop2;

-- create the tables for the database
CREATE TABLE customers (
  customerID        INT             NOT NULL  AUTO_INCREMENT,
  emailAddress      VARCHAR(255)    NOT NULL,
  password          VARCHAR(60)     NOT NULL,
  firstName         VARCHAR(60)     NOT NULL,
  lastName          VARCHAR(60)     NOT NULL,
  shipAddressID     INT                       DEFAULT NULL,
  billingAddressID  INT                       DEFAULT NULL,
  PRIMARY KEY (customerID),
  UNIQUE INDEX emailAddress (emailAddress)
);
```

The SQL script that creates the database (cont.)

```
CREATE TABLE addresses (
  addressID         INT             NOT NULL  AUTO_INCREMENT,
  customerID        INT             NOT NULL,
  line1             VARCHAR(60)     NOT NULL,
  line2             VARCHAR(60)               DEFAULT NULL,
  city              VARCHAR(40)     NOT NULL,
  state             VARCHAR(2)      NOT NULL,
  zipCode           VARCHAR(10)     NOT NULL,
  phone             VARCHAR(12)     NOT NULL,
  disabled          TINYINT(1)      NOT NULL  DEFAULT 0,
  PRIMARY KEY (addressID),
  INDEX customerID (customerID)
);
```


The SQL script that creates the database (cont.)

```
CREATE TABLE orders (
  orderID           INT             NOT NULL  AUTO_INCREMENT,
  customerID        INT             NOT NULL,
  orderDate         DATETIME        NOT NULL,
  shipAmount        DECIMAL(10,2)   NOT NULL,
  taxAmount         DECIMAL(10,2)   NOT NULL,
  shipDate          DATETIME                  DEFAULT NULL,
  shipAddressID     INT             NOT NULL,
  cardType          INT             NOT NULL,
  cardNumber        CHAR(16)        NOT NULL,
  cardExpires       CHAR(7)         NOT NULL,
  billingAddressID  INT             NOT NULL,
  PRIMARY KEY (orderID),
  INDEX customerID (customerID)
);
```


The SQL script that creates the database (cont.)

```
CREATE TABLE orderItems (
  itemID            INT             NOT NULL  AUTO_INCREMENT,
  orderID           INT             NOT NULL,
  productID         INT             NOT NULL,
  itemPrice         DECIMAL(10,2)   NOT NULL,
  discountAmount    DECIMAL(10,2)   NOT NULL,
  quantity          INT NOT NULL,
  PRIMARY KEY (itemID),
  INDEX orderID (orderID),
  INDEX productID (productID)
);
```


The SQL script that creates the database (cont.)

```
CREATE TABLE products (
  productID         INT             NOT NULL  AUTO_INCREMENT,
  categoryID        INT             NOT NULL,
  productCode       VARCHAR(10)     NOT NULL,
  productName       VARCHAR(255)    NOT NULL,
  description       TEXT            NOT NULL,
  listPrice         DECIMAL(10,2)   NOT NULL,
  discountPercent   DECIMAL(10,2)   NOT NULL  DEFAULT 0.00,
  dateAdded         DATETIME        NOT NULL,
  PRIMARY KEY (productID),
  INDEX categoryID (categoryID),
  UNIQUE INDEX productCode (productCode)
);

CREATE TABLE categories (
  categoryID        INT             NOT NULL  AUTO_INCREMENT,
  categoryName      VARCHAR(255)    NOT NULL,
  PRIMARY KEY (categoryID)
);
```


The SQL script that creates the database (cont.)

```
CREATE TABLE administrators (
  adminID           INT             NOT NULL  AUTO_INCREMENT,
  emailAddress      VARCHAR(255)    NOT NULL,
  password          VARCHAR(255)    NOT NULL,
  firstName         VARCHAR(255)    NOT NULL,
  lastName          VARCHAR(255)    NOT NULL,
  PRIMARY KEY (adminID)
);

-- Create a user and grant privileges to that user
GRANT SELECT, INSERT, UPDATE, DELETE
ON *
TO mgs_user@localhost
IDENTIFIED BY 'pa55word';
```