Introduction to SQL

Creating Tables

Jon Flanders
@jonflanders

pluralsight
hardcore developer training

What you will learn...

What you will learn...

- How to create and modify Tables

Creating Things with SQL

Creating Things with SQL

- There is a whole set of SQL commands that relate to creating and modifying constructs in a database

Creating Things with SQL

- There is a whole set of SQL commands that relate to creating and modifying constructs in a database
- These are classified as DDL

Creating Things with SQL

- **There is a whole set of SQL commands that relate to creating and modifying constructs in a database**
- **These are classified as DDL**
  - Data Definition Language

Creating Things with SQL

- **There is a whole set of SQL commands that relate to creating and modifying constructs in a database**
- **These are classified as DDL**
  - Data Definition Language
  - Formally, it is a subset of SQL

Creating Things with SQL

- **There is a whole set of SQL commands that relate to creating and modifying constructs in a database**
- **These are classified as DDL**
  - Data Definition Language
  - Formally, it is a subset of SQL
- **Many RDMS products have visual tools to help you create Tables and relations**

Creating Things with SQL

- **There is a whole set of SQL commands that relate to creating and modifying constructs in a database**
- **These are classified as DDL**
  - Data Definition Language
  - Formally, it is a subset of SQL
- **Many RDMS products have visual tools to help you create Tables and relations**
  - So you may never need to do DDL by hand, understanding it is a good foundation for using those tools

Creating the Database

Creating the Database

```
--this is not ANSI SQL
--but is supported by most vendors
CREATE DATABASE Contacts;
USE DATABASE Contact;
```

CREATE TABLE

CREATE TABLE

- CREATE TABLE is part of the ANSI standard

CREATE TABLE

- CREATE TABLE is part of the ANSI standard
- CREATE TABLE takes a Table name as input

CREATE TABLE

- CREATE TABLE is part of the ANSI standard
- CREATE TABLE takes a Table name as input
- A list of column definitions follows the Table name

CREATE TABLE

- **CREATE TABLE is part of the ANSI standard**
- **CREATE TABLE takes a Table name as input**
- **A list of column definitions follows the Table name**
  - Inside of parens ( )

CREATE TABLE

- **CREATE TABLE is part of the ANSI standard**
- **CREATE TABLE takes a Table name as input**
- **A list of column definitions follows the Table name**
  - Inside of parens ( )
  - Each column definition is separated by a comma

CREATE TABLE

- **CREATE TABLE is part of the ANSI standard**
- **CREATE TABLE takes a Table name as input**
- **A list of column definitions follows the Table name**
  - Inside of parens ( )
  - Each column definition is separated by a comma
- **Each column definition contains the following**

CREATE TABLE

- **CREATE TABLE is part of the ANSI standard**
- **CREATE TABLE takes a Table name as input**
- **A list of column definitions follows the Table name**
  - Inside of parens ( )
  - Each column definition is separated by a comma
- **Each column definition contains the following**
  - A column name

CREATE TABLE

- **CREATE TABLE is part of the ANSI standard**
- **CREATE TABLE takes a Table name as input**
- **A list of column definitions follows the Table name**
  - Inside of parens ( )
  - Each column definition is separated by a comma
- **Each column definition contains the following**
  - A column name
  - A column type definition

Column Definition

Column Definition

- Starts with a name

Column Definition

- Starts with a name
- Followed by a data type

Column Definition

- **Starts with a name**
- **Followed by a data type**

Data Type	Value Space
CHARACTER	Can hold N character values – set to N statically
CHARACTER VARYING	Can hold N character values – set to N dynamically – storage can be less than N
BINARY	Hexadecimal data
SMALLINT	-2^15 (-32,768) to 2^15-1 (32,767)
INTEGER	-2^31 (-2,147,483,648) to 2^31-1 (2,147,483,647)
BIGINT	-2^63 (-9,223,372,036,854,775,808) to 2^63-1 (9,223,372,036,854,775,807)
BOOLEAN	TRUE or FALSE
DATE	YEAR, MONTH, and DAY in the format YYYY-MM-DD
TIME	HOUR, MINUTE, and SECOND in the format HH:MM:SS[.sF] where F is the fractional part of the SECOND value
TIMESTAMP	Both DATE and TIME

CREATE TABLE Example

CREATE TABLE Example

```
CREATE TABLE email_address
( email_address_id INTEGER,
email_address_person_id INTEGER,
   email_address VARCHAR(55));
```

NULL or NOT NULL

NULL or NOT NULL

- After the column type, you can specify NULL or NOT NULL

NULL or NOT NULL

- After the column type, you can specify NULL or NOT NULL
- NULL is the default

NULL or NOT NULL

- **After the column type, you can specify NULL or NOT NULL**
- **NULL is the default**
  - In fact, ANSI SQL disallows the default to be specified (most DBs allow it however)

NULL or NOT NULL

- **After the column type, you can specify NULL or NOT NULL**
- **NULL is the default**
  - In fact, ANSI SQL disallows the default to be specified (most DBs allow it however)
  - Means that NULL values are valid values for that column in a particular row

NULL or NOT NULL

- **After the column type, you can specify NULL or NOT NULL**
- **NULL is the default**
  - In fact, ANSI SQL disallows the default to be specified (most DBs allow it however)
  - Means that NULL values are valid values for that column in a particular row
- **NOT NULL means that a valid value is required for that column**

NULL or NOT NULL

- **After the column type, you can specify NULL or NOT NULL**
- **NULL is the default**
  - In fact, ANSI SQL disallows the default to be specified (most DBs allow it however)
  - Means that NULL values are valid values for that column in a particular row
- **NOT NULL means that a valid value is required for that column**
  - An attempt to insert a NULL value will fail

NULL or NOT NULL

- **After the column type, you can specify NULL or NOT NULL**
- **NULL is the default**
  - In fact, ANSI SQL disallows the default to be specified (most DBs allow it however)
  - Means that NULL values are valid values for that column in a particular row
- **NOT NULL means that a valid value is required for that column**
  - An attempt to insert a NULL value will fail
  - An attempt to use INSERT without specifying a value for that column will fail

NULL or NOT NULL

- **After the column type, you can specify NULL or NOT NULL**
- **NULL is the default**
  - In fact, ANSI SQL disallows the default to be specified (most DBs allow it however)
  - Means that NULL values are valid values for that column in a particular row
- **NOT NULL means that a valid value is required for that column**
  - An attempt to insert a NULL value will fail
  - An attempt to use INSERT without specifying a value for that column will fail

```
CREATE TABLE email_address
( email_address_id INTEGER NOT NULL,
  email_address_person_id INTEGER,
email_address VARCHAR(55) NOT NULL);
```

PRIMARY KEY

PRIMARY KEY

- After the data type you can add the PRIMARY KEY constraint instead of NULL or NOT NULL

PRIMARY KEY

- **After the data type you can add the PRIMARY KEY constraint instead of NULL or NOT NULL**
  - PRIMARY KEY columns are implicitly NOT NULL

PRIMARY KEY

- **After the data type you can add the PRIMARY KEY constraint instead of NULL or NOT NULL**
  - PRIMARY KEY columns are implicitly NOT NULL
- **It is possible for more than one column to form together as a compound PRIMARY KEY**

PRIMARY KEY

- **After the data type you can add the PRIMARY KEY constraint instead of NULL or NOT NULL**
  - PRIMARY KEY columns are implicitly NOT NULL
- **It is possible for more than one column to form together as a compound PRIMARY KEY**

```
CREATE TABLE email_address
( email_address_id INTEGER PRIMARY KEY,
  email_address_person_id INTEGER,
  email_address VARCHAR(55) NOT NULL);
```

CONSTRAINT

CONSTRAINT

- You can add a CONSTRAINT clause at the end of your column definitions

CONSTRAINT

- **You can add a CONSTRAINT clause at the end of your column definitions**
  - You can add the PRIMARY KEY constraint this way if you want to

CONSTRAINT

- **You can add a CONSTRAINT clause at the end of your column definitions**
  - You can add the PRIMARY KEY constraint this way if you want to
  - You can add FOREIGN KEY constraints

CONSTRAINT

- **You can add a CONSTRAINT clause at the end of your column definitions**
  - You can add the PRIMARY KEY constraint this way if you want to
  - You can add FOREIGN KEY constraints

```
CREATE TABLE phone_number
( phone_number_id INTEGER NOT NULL,
phone_number_person_id INTEGER NOT  NULL,
    phone_number VARCHAR(55) NOT NULL,
CONSTRAINT PK_phone_number PRIMARY KEY
(phone_number_id));
```

ALTER TABLE

ALTER TABLE

- ALTER TABLE enables you to add or change a column definition or CONSTRAINT on an existing Table

ALTER TABLE

- **ALTER TABLE enables you to add or change a column definition or CONSTRAINT on an existing Table**
  - Often used in database creation scripts to enable correct order of operations

ALTER TABLE

- **ALTER TABLE enables you to add or change a column definition or CONSTRAINT on an existing Table**
  - Often used in database creation scripts to enable correct order of operations
- **Allows you to do anything that you could do in CREATE TABLE definition**

ALTER TABLE

- **ALTER TABLE enables you to add or change a column definition or CONSTRAINT on an existing Table**
  - Often used in database creation scripts to enable correct order of operations
- **Allows you to do anything that you could do in CREATE TABLE definition**
- **Warning – doesn't work if Table already has data in it that conflicts with the change**

ALTER TABLE

- **ALTER TABLE enables you to add or change a column definition or CONSTRAINT on an existing Table**
  - Often used in database creation scripts to enable correct order of operations
- **Allows you to do anything that you could do in CREATE TABLE definition**
- **Warning – doesn't work if Table already has data in it that conflicts with the change**
  - For example, you can't change a column to NOT NULL if it already has NULL data in it

ALTER TABLE

- **ALTER TABLE enables you to add or change a column definition or CONSTRAINT on an existing Table**
  - Often used in database creation scripts to enable correct order of operations
- **Allows you to do anything that you could do in CREATE TABLE definition**
- **Warning – doesn't work if Table already has data in it that conflicts with the change**
  - For example, you can't change a column to NOT NULL if it already has NULL data in it
  - Need to use some sort of temporary Table to hold the original data, change it, ALTER the Table, and then put the data back

ALTER TABLE

- **ALTER TABLE enables you to add or change a column definition or CONSTRAINT on an existing Table**
  - Often used in database creation scripts to enable correct order of operations
- **Allows you to do anything that you could do in CREATE TABLE definition**
- **Warning – doesn't work if Table already has data in it that conflicts with the change**
  - For example, you can't change a column to NOT NULL if it already has NULL data in it
  - Need to use some sort of temporary Table to hold the original data, change it, ALTER the Table, and then put the data back

```
ALTER TABLE email_address
ADD CONSTRAINT FK_email_address_person FOREIGN
KEY(email_address_person_id)
REFERENCES person (person_id);
```

DROP TABLE

DROP TABLE

- DROP TABLE command removes the Table from the database

DROP TABLE

- **DROP TABLE command removes the Table from the database**
  - Also removes all the data

DROP TABLE

- **DROP TABLE command removes the Table from the database**
  - Also removes all the data
- **Use it wisely**

DROP TABLE

- **DROP TABLE command removes the Table from the database**
  - Also removes all the data
- **Use it wisely**
- **You often have to deal with relations**

DROP TABLE

- **DROP TABLE command removes the Table from the database**
  - Also removes all the data
- **Use it wisely**
- **You often have to deal with relations**
  - Can’t delete Table that has a column referenced by another Table as a foreign key

DROP TABLE

- **DROP TABLE command removes the Table from the database**
  - Also removes all the data
- **Use it wisely**
- **You often have to deal with relations**
  - Can't delete Table that has a column referenced by another Table as a foreign key

```
DROP TABLE person;
```

Summary

Summary

- Understanding DDL is a good foundation for working with SQL even if you use it rarely

Summary

- Understanding DDL is a good foundation for working with SQL even if you use it rarely
- CREATE TABLE is the command to configure your columns and relations

Summary

- Understanding DDL is a good foundation for working with SQL even if you use it rarely
- CREATE TABLE is the command to configure your columns and relations
- ALTER TABLE lets you change existing definitions

Summary

- Understanding DDL is a good foundation for working with SQL even if you use it rarely
- CREATE TABLE is the command to configure your columns and relations
- ALTER TABLE lets you change existing definitions
- DROP TABLE removes the table and all its rows from the database