**Manipulation**

**Introduction to SQL**

3 min

Preview: Docs Loading link description

[SQL](https://www.codecademy.com/resources/docs/sql/about-sql)

, **S**tructured **Q**uery **L**anguage, is a programming language designed to manage data stored in

Preview: Docs A relational database is a store of data organized in tables made of columns and rows, and the relationships between those tables.

[relational databases](https://www.codecademy.com/resources/docs/general/database/relational-database)

. SQL operates through simple, declarative statements. This keeps data accurate and secure, and helps maintain the integrity of

Preview: Docs Loading link description

[databases](https://www.codecademy.com/resources/docs/general/database?page_ref=catalog)

, regardless of size.

The SQL language is widely used today across web frameworks and database applications. Knowing SQL gives you the freedom to explore your data, and the power to make better decisions. By learning SQL, you will also learn concepts that apply to nearly every data storage system.

The statements covered in this course use SQLite Relational Database Management System [(RDBMS)](https://www.codecademy.com/articles/what-is-rdbms-sql). You can also access a glossary of all the [SQL commands](https://www.codecademy.com/articles/sql-commands) taught in this course.

**Instructions**

1. Checkpoint 1 Passed

**1.**

Let’s begin by entering a SQL command.

In the code editor, type:

**SELECT** \* **FROM** celebs;

Copy to Clipboard

You will run all of your SQL commands in this course by pressing the Run button at the bottom of the code editor.

Press Run.

Hint

Common errors:

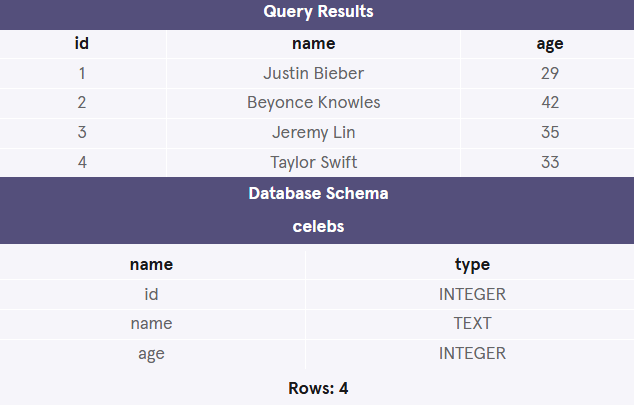
* + Did you forget the \*?
  + Did you forget the ;?

The table should show four rows of data and three columns (id, name, and age).

We will explain what this line means in a bit!

**test.sqlite**

**SELECT** \* **FROM** celebs;



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Relational Databases**

2 min

Nice work! In one line of code, you returned information from a relational database.

**SELECT** \* **FROM** celebs;

Copy to Clipboard

We’ll take a look at what this code means soon, for now, let’s focus on what relational databases are and how they are organized.

A

Preview: Docs Loading link description

[*relational database*](https://www.codecademy.com/resources/docs/general/relational-database?page_ref=catalog)

 is a

Preview: Docs Loading link description

[database](https://www.codecademy.com/resources/docs/general/database?page_ref=catalog)

 that organizes information into one or more tables. Here, the relational database contains one table.

A *table* is a collection of data organized into rows and columns. Tables are sometimes referred to as *relations*. Here the table is celebs.

A *column* is a set of data values of a particular type. Here, id, name, and age are the columns.

A *row* is a single record in a table. The first row in the celebs table has:

* An id of 1
* A name of Justin Bieber
* An age of 29

All data stored in a relational database is of a certain

Preview: Docs Loading link description

[data type](https://www.codecademy.com/resources/docs/sql/data-types?page_ref=catalog)

. Some of the most common data types are:

* INTEGER, a positive or negative whole number
* TEXT, a text string
* DATE, the date formatted as YYYY-MM-DD
* REAL, a decimal value

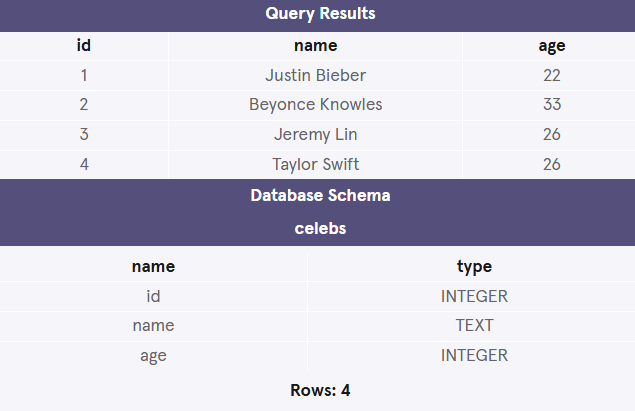
**Instructions**

Now that you have an understanding of what relational databases are, let’s take a closer look at SQL syntax.

Click Next to continue.

**test.sqlite**

**SELECT** \* **FROM** celebs;



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Statements**

4 min

The code below is a SQL statement. A *statement* is text that the database recognizes as a valid command. Statements always end in a semicolon ;.

**CREATE TABLE** table\_name (  
   column\_1 data\_type,   
   column\_2 data\_type,   
   column\_3 data\_type  
);

Copy to Clipboard

Let’s break down the components of a statement:

1. Preview: Docs Loading link description

[CREATE TABLE](https://www.codecademy.com/resources/docs/sql/commands/create-table?page_ref=catalog)

 is a *clause*. Clauses perform specific tasks in SQL. By convention, clauses are written in capital letters. Clauses can also be referred to as commands.

1. table\_name refers to the name of the table that the command is applied to.
2. (column\_1 data\_type, column\_2 data\_type, column\_3 data\_type) is a *parameter*. A parameter is a list of columns,

Preview: Docs Loading link description

[data types](https://www.codecademy.com/resources/docs/sql/data-types?page_ref=catalog)

, or values that are passed to a clause as an argument. Here, the parameter is a list of column names and the associated data type.

The structure of SQL statements vary. The number of lines used does not matter. A statement can be written all on one line, or split up across multiple lines if it makes it easier to read. In this course, you will become familiar with the structure of common statements.

**Instructions**

1. Checkpoint 1 Passed

**1.**

Let’s take a closer look at the statement we wrote before. In the code editor, type:

**SELECT** \* **FROM** celebs;

Copy to Clipboard

Run the code to observe the results, and ask yourself:

* + Which parts of the statement are the *clauses*?
  + What table are we applying the command to?

Uncover the hint to view the answers, and then proceed to the next exercise.

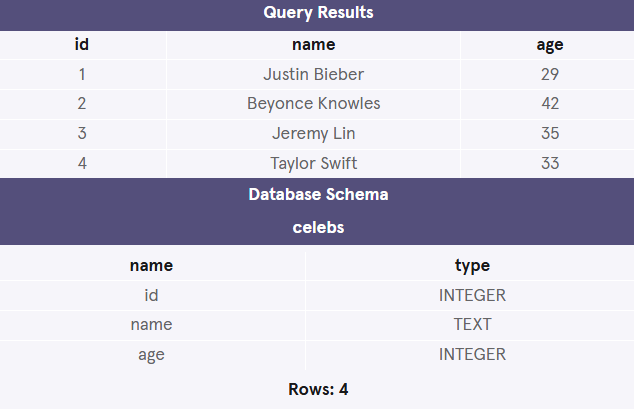
Hint

* + SELECT and FROM are the clauses here.
  + We are applying the command to the celebs table.

Soon you will learn more about each part of this statement!

**test.sqlite**

**SELECT** \* **FROM** celebs;



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Create**

4 min

CREATE statements allow us to create a new table in the database. You can use the CREATE statement anytime you want to create a new table from scratch. The statement below creates a new table named celebs.

**CREATE TABLE** celebs (  
   id INTEGER,   
   name TEXT,   
   age INTEGER  
);

Copy to Clipboard

1.

Preview: Docs Loading link description

[CREATE TABLE](https://www.codecademy.com/resources/docs/sql/commands/create-table?page_ref=catalog)

 is a clause that tells SQL you want to create a new table.

2. celebs is the name of the table.

3. (id INTEGER, name TEXT, age INTEGER) is a list of parameters defining each column, or attribute in the table and its

Preview: Docs Loading link description

[data type](https://www.codecademy.com/resources/docs/sql/data-types?page_ref=catalog)

:

* id is the first column in the table. It stores values of data type INTEGER
* name is the second column in the table. It stores values of data type TEXT
* age is the third column in the table. It stores values of data type INTEGER

**Instructions**

1. Checkpoint 1 Passed

**1.**

Now that you have a good understanding of SQL syntax, we can create a new table. We’ve cleared the database from the previous exercises. Confirm no celebs table exists by entering the following in the code editor:

**SELECT** \* **FROM** celebs;

Copy to Clipboard

Look at the results. The database should be empty!

1. Checkpoint 2 Passed

**2.**

Now that we know the database is empty, let’s create a new celebs table.

In the code editor, type:

**CREATE TABLE** celebs (  
   id INTEGER,   
   name TEXT,   
   age INTEGER  
);

Copy to Clipboard

We will learn how to view this table in a later exercise after we have added some data to it.

Hint

We want to create a new table named celebs. Make sure you:

* + Insert commas after the first two columns are declared.
  + End your statement with a ;.

**test.sqlite**

**SELECT** \* **FROM** celebs;

**CREATE TABLE** celebs (

  id INTEGER,

  name TEXT,

  age INTEGER

);

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Insert**

7 min

The INSERT statement inserts a new row into a table.

We can use the INSERT statement when you want to add new records. The statement below enters a record for Justin Bieber into the celebs table.

INSERT INTO celebs (id, name, age)   
VALUES (1, 'Justin Bieber', 29);

Copy to Clipboard

* Preview: Docs Loading link description

[INSERT INTO](https://www.codecademy.com/resources/docs/sql/commands/insert-into?page_ref=catalog)

 is a clause that adds the specified row or rows.

* celebs is the table the row is added to.
* (id, name, age) is a parameter identifying the columns that data will be inserted into.
* VALUES is a clause that indicates the data being inserted.
* (1, 'Justin Bieber', 29) is a parameter identifying the values being inserted.
  + 1: an integer that will be added to id column
  + 'Justin Bieber': text that will be added to name column
  + 29: an integer that will be added to age column

**Instructions**

1. Checkpoint 1 Passed

**1.**

Add a row to the table. In the code editor, type:

**INSERT INTO** celebs (id, name, age)   
**VALUES** (1, 'Justin Bieber', 29);

Copy to Clipboard

Look at the Database Schema. How many rows are in the celebs table?

Hint

Make sure there is a set of parentheses around the column names and values to be inserted into the table!

Notice the single quotes around Justin Bieber. This is because text strings require quotes around them, while numbers don’t.

There should be 1 row in the celebs table now.

1. Checkpoint 2 Passed

**2.**

Add three more celebrities to the table. Beneath your previous INSERT statement type:

**INSERT INTO** celebs (id, name, age)   
**VALUES** (2, 'Beyonce Knowles', 42);   
  
**INSERT INTO** celebs (id, name, age)   
**VALUES** (3, 'Jeremy Lin', 35);   
  
**INSERT INTO** celebs (id, name, age)   
**VALUES** (4, 'Taylor Swift', 33);

Copy to Clipboard

Look at the Database Schema. How many rows are in the celebs table now?

Hint

Make sure to enter the three new INSERT statements beneath the first INSERT statement.

There should be 4 rows in the celebs table now.

How do we see what information is stored in these rows? Head to the next exercise to find out!

**test.sqlite**

**INSERT INTO** celebs (id, name, age)

**VALUES** (1, 'Justin Bieber', 29);

**INSERT INTO** celebs (id, name, age)

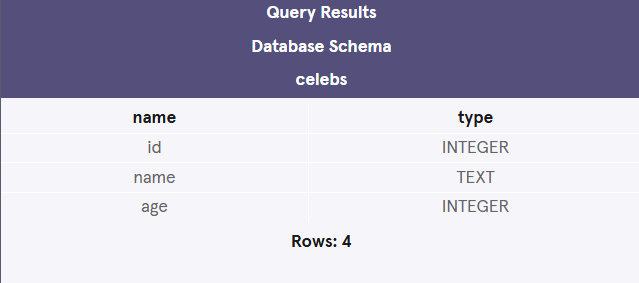
**VALUES** (2, 'Beyonce Knowles', 42);

**INSERT INTO** celebs (id, name, age)

**VALUES** (3, 'Jeremy Lin', 35);

**INSERT INTO** celebs (id, name, age)

**VALUES** (4, 'Taylor Swift', 33);



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Select**

3 min

SELECT statements are used to fetch data from a database. In the statement below, SELECT returns all data in the name column of the celebs table.

SELECT name FROM celebs;

Copy to Clipboard

1.

Preview: Docs Loading link description

[SELECT](https://www.codecademy.com/resources/docs/sql/commands/select?page_ref=catalog)

 is a clause that indicates that the statement is a query. You will use SELECT every time you query data from a database.

2. name specifies the column to query data from.

3. FROM celebs specifies the name of the table to query data from. In this statement, data is queried from the celebs table.

You can also query data from all columns in a table with SELECT.

**SELECT** \* **FROM** celebs;

Copy to Clipboard

\* is a special wildcard character that we have been using. It allows you to select every column in a table without having to name each one individually. Here, the result set contains every column in the celebs table.

SELECT statements always return a new table called the *result set*.

**Instructions**

1. Checkpoint 1 Passed

**1.**

Let’s take a closer look at SELECT and retrieve all the names in the celebs table. In the code editor, type:

**SELECT** name **FROM** celebs;

Copy to Clipboard

Hint

Don’t forget to include the FROM clause and the name of the table which we are selecting the data from!

The result should only have a single column (name).

**test.sqlite**

**SELECT** name **FROM** celebs;

**2.**

Delete your previous SELECT statement from the code editor.

To SELECT *all* the data in the celebs table, enter the following statement in the code editor using the \* wildcard character:

**SELECT** \* **FROM** celebs;

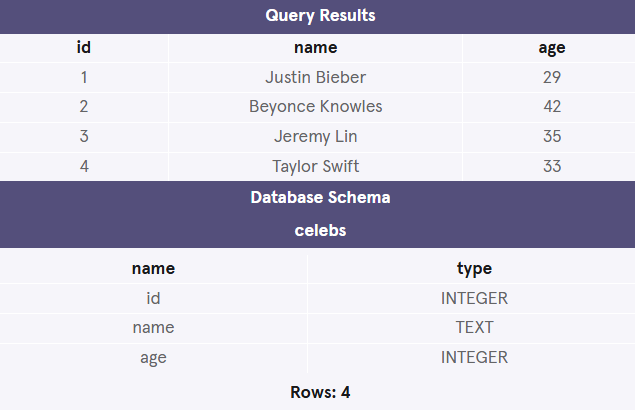
Copy to Clipboard

Hint

Did you delete the first SELECT statement from the code editor?

**test.sqlite**

**SELECT** \* **FROM** celebs;



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Alter**

3 min

The ALTER TABLE statement adds a new column to a table. You can use this command when you want to add columns to a table. The statement below adds a new column twitter\_handle to the celebs table.

**ALTER TABLE** celebs   
**ADD COLUMN** twitter\_handle TEXT;

Copy to Clipboard

1.

Preview: Docs Adds, deletes, modifies or changes the data type of a column in a table.

[ALTER TABLE](https://www.codecademy.com/resources/docs/sql/commands/alter-table?page_ref=catalog)

 is a clause that lets you make the specified changes.

2. celebs is the name of the table that is being changed.

3. ADD COLUMN is a clause that lets you add a new column to a table:

* twitter\_handle is the name of the new column being added
* TEXT is the data type for the new column

4. NULL is a special value in SQL that represents missing or unknown data. Here, the rows that existed before the column was added have NULL (∅) values for twitter\_handle.

**Instructions**

1. Checkpoint 1 Passed

**1.**

Add a new column to the table. In the code editor, type:

**ALTER TABLE** celebs **ADD COLUMN** twitter\_handle TEXT;   
  
**SELECT** \* **FROM** celebs;

Copy to Clipboard

Hint

Common errors:

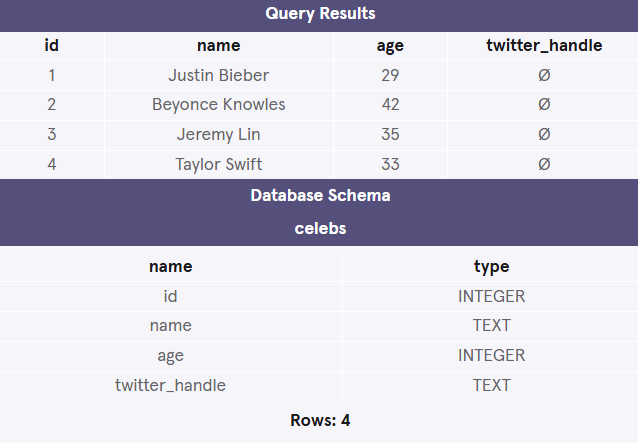
* + Missing the complete ADD COLUMN syntax.
  + Missing the data type for the new column.

**test.sqlite**

**ALTER TABLE** celebs

**ADD COLUMN** twitter\_handle TEXT;

**SELECT** \* **FROM** celebs;



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Update**

3 min

The UPDATE statement edits a row in a table. You can use the UPDATE statement when you want to change existing records. The statement below updates the record with an id value of 4 to have the twitter\_handle @taylorswift13.

**UPDATE** celebs   
**SET** twitter\_handle = '@taylorswift13'   
**WHERE** id = 4;

Copy to Clipboard

1.

Preview: Docs Loading link description

[UPDATE](https://www.codecademy.com/resources/docs/sql/commands/update?page_ref=catalog)

 is a clause that edits a row in the table.

2. celebs is the name of the table.

3. SET is a clause that indicates the column to edit.

* twitter\_handle is the name of the column that is going to be updated
* @taylorswift13 is the new value that is going to be inserted into the twitter\_handle column.

4. WHERE is a clause that indicates which row(s) to update with the new column value. Here the row with a 4 in the id column is the row that will have the twitter\_handle updated to @taylorswift13.

**Instructions**

1. Checkpoint 1 Passed

**1.**

Update the table to include Taylor Swift’s [twitter handle](https://twitter.com/taylorswift13). In the code editor, type:

**UPDATE** celebs   
**SET** twitter\_handle = '@taylorswift13'   
**WHERE** id = 4;   
  
**SELECT** \* **FROM** celebs;

Copy to Clipboard

Hint

Double-check your statement character by character:

* + Did you include the underscore in twitter\_handle?
  + Did you include the @ in Taylor’s twitter\_handle?

Notice the single quotes around @taylorswift13. This is because text strings require quotes around them, while numbers don’t.

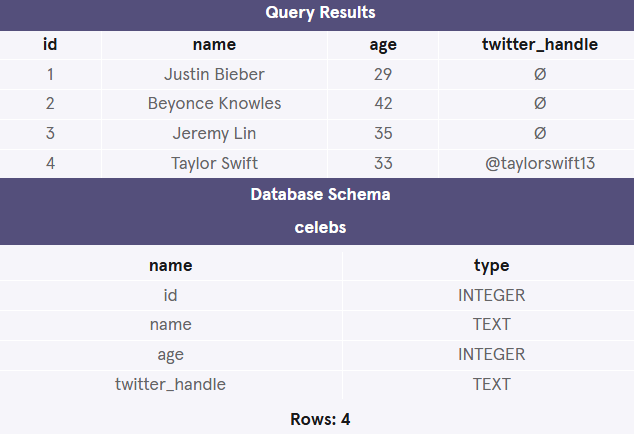
**test.sqlite**

**UPDATE** celebs

**SET** twitter\_handle = '@taylorswift13'

**WHERE** id = 4;

**SELECT** \* **FROM** celebs;



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Delete**

2 min

The DELETE FROM statement deletes one or more rows from a table. You can use the statement when you want to delete existing records. The statement below deletes all records in the celebs table with no twitter\_handle:

**DELETE FROM** celebs   
**WHERE** twitter\_handle **IS NULL**;

Copy to Clipboard

1. Preview: Docs Loading link description

[DELETE FROM](https://www.codecademy.com/resources/docs/sql/commands/delete?page_ref=catalog)

 is a clause that lets you delete rows from a table.

1. celebs is the name of the table we want to delete rows from.
2. Preview: Docs Loading link description

[WHERE](https://www.codecademy.com/resources/docs/sql/commands/where)

 is a clause that lets you select which rows you want to delete. Here we want to delete all of the rows where the twitter\_handle column IS NULL.

1. IS NULL is a condition in SQL that returns true when the value is NULL and false otherwise.

**Instructions**

1. Checkpoint 1 Passed

**1.**

Delete all of the rows that have a NULL value in the twitter handle column. In the code editor, type the following:

**DELETE FROM** celebs   
**WHERE** twitter\_handle **IS NULL**;  
  
**SELECT** \* **FROM** celebs;

Copy to Clipboard

How many rows exist in the celebs table now?

Hint

Did you type:

**SELECT** \* **FROM** celebs;

Copy to Clipboard

after your deletion statement?

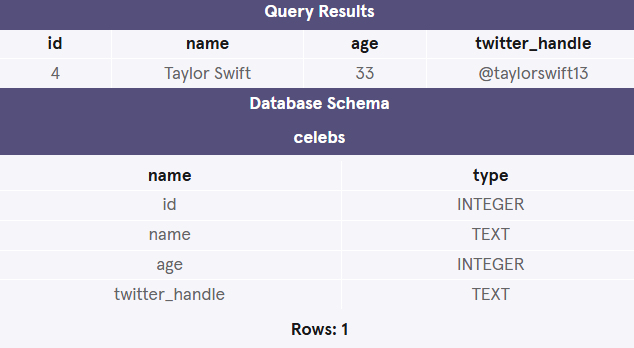
There should only be 1 row left.

**test.sqlite**

**DELETE FROM** celebs

**WHERE** twitter\_handle **IS NULL**;

**SELECT** \* **FROM** celebs;



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Constraints**

6 min

Preview: Docs Loading link description

[*Constraints*](https://www.codecademy.com/resources/docs/sql/constraints?page_ref=catalog)

 that add information about how a column can be used are invoked after specifying the data type for a column. They can be used to tell the database to reject inserted data that does not adhere to a certain restriction. The statement below sets *constraints* on the celebs table.

**CREATE TABLE** celebs (  
   id **INTEGER PRIMARY KEY**,   
   name **TEXT UNIQUE**,  
   date\_of\_birth **TEXT NOT NULL**,  
   date\_of\_death **TEXT DEFAULT** 'Not Applicable'  
);

Copy to Clipboard

1. PRIMARY KEY columns can be used to uniquely identify the row. Attempts to insert a row with an identical value to a row already in the table will result in a *constraint violation* which will not allow you to insert the new row.

2. UNIQUE columns have a different value for every row. This is similar to PRIMARY KEY except a table can have many different UNIQUE columns.

3. NOT NULL columns must have a value. Attempts to insert a row without a value for a NOT NULL column will result in a constraint violation and the new row will not be inserted.

4. DEFAULT columns take an additional argument that will be the assumed value for an inserted row if the new row does not specify a value for that column.

**Instructions**

1. Checkpoint 1 Passed

**1.**

**Create a new table with constraints on the values. In the code editor type:**

**CREATE TABLE** awards (  
   id **INTEGER PRIMARY KEY**,  
   recipient **TEXT NOT NULL**,  
   award\_name **TEXT DEFAULT** 'Grammy'  
);

Copy to Clipboard

How many tables do you see in the database schema on the right?

Hint

Common errors:

* + Missing the commas after the first and second column declarations.
  + Missing the data type or constraints of each column.

**test.sqlite**

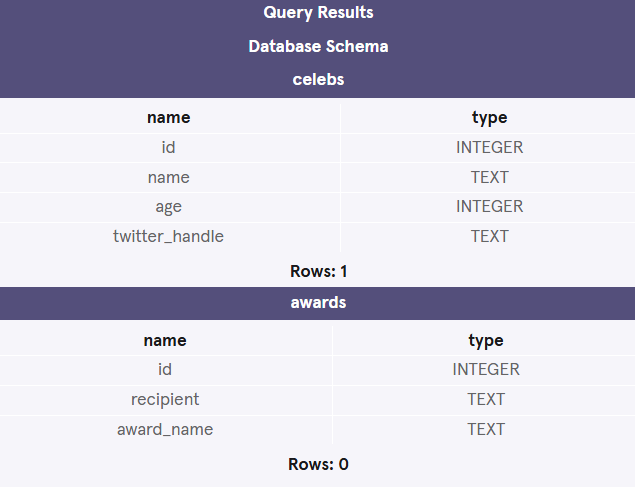
**CREATE TABLE** awards (

  id **INTEGER PRIMARY KEY**,

  recipient **TEXT NOT NULL**,

  award\_name **TEXT DEFAULT** 'Grammy'

)



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Review**

1 min

Congratulations! We’ve learned six commands commonly used to manage data stored in a relational database and how to set constraints on such data. What can we generalize so far?

Preview: Docs Loading link description

[SQL](https://www.codecademy.com/resources/docs/sql?page_ref=catalog)

 is a programming language designed to manipulate and manage data stored in relational databases.

* A *relational database* is a database that organizes information into one or more tables.
* A *table* is a collection of data organized into rows and columns.

A *statement* is a string of characters that the database recognizes as a valid command.

* Preview: Docs Loading link description

[CREATE TABLE](https://www.codecademy.com/resources/docs/sql/commands/create-table?page_ref=catalog)

 creates a new table.

* Preview: Docs Loading link description

[INSERT INTO](https://www.codecademy.com/resources/docs/sql/commands/insert-into?page_ref=catalog)

 adds a new row to a table.

* Preview: Docs Loading link description

[SELECT](https://www.codecademy.com/resources/docs/sql/commands/select?page_ref=catalog)

 queries data from a table.

* Preview: Docs Loading link description

[ALTER TABLE](https://www.codecademy.com/resources/docs/sql/commands/alter-table?page_ref=catalog)

 changes an existing table.

* Preview: Docs Loading link description

[UPDATE](https://www.codecademy.com/resources/docs/sql/commands/update?page_ref=catalog)

 edits a row in a table.

* Preview: Docs Loading link description

[DELETE FROM](https://www.codecademy.com/resources/docs/sql/commands/delete?page_ref=catalog)

 deletes rows from a table.

Preview: Docs Loading link description

[*Constraints*](https://www.codecademy.com/resources/docs/sql/constraints?page_ref=catalog)

 add information about how a column can be used.

**Instructions**

In this lesson, we have learned SQL statements that create, edit, and delete data. In the upcoming lessons, we will learn how to use SQL to retrieve information from a database!