Table Constraints

by Sophia Tutorial

WHAT'S COVERED

This tutorial explores the different table constraints that can be applied in three parts:

1. The PRIMARY KEY, NOT NULL, and UNIQUE Constraints
2. The FOREIGN KEY Constraint
3. The DEFAULT and CHECK Constraints

1. The PRIMARY KEY, NOT NULL, and UNIQUE Constraints

In the prior tutorial, we made use of the PRIMARY KEY constraint. The PRIMARY KEY constraint helps to uniquely identify a row within a table by enforcing entity integrity. In order to do so, it applies two other constraints: the NOT NULL constraint and the UNIQUE constraint.

The NOT NULL constraint ensures that a value has to exist for this column. When it is crucial for us to have the data, the NOT NULL constraint will not allow the attribute to not have a value. For instance, in our artist table, the name of the artist is a column that we would not want to have any empty values for. Otherwise, we could not identify who the artist is.

In the employee table, many of the columns could be set up to be required values using the NOT NULL constraint. For example, the hire_date would be one value that should always exist, as the hire_date would need to exist for any employee hired into the company. To add a NOT NULL constraint to a column, we would list it beside the data type like this:

```
CREATE TABLE contact(
    contact_id int PRIMARY KEY,
    username VARCHAR(50) NOT NULL,
    password VARCHAR(50) NOT NULL
);
```

The NOT NULL constraint in this example will not permit the username and password to have missing values in the table.

The other constraint that the PRIMARY KEY constraint uses is the UNIQUE constraint. Similar to the NOT NULL constraint, the UNIQUE constraint can be set up on its own. The UNIQUE constraint creates a unique index on

the column. We can use the UNIQUE constraint to ensure that we do not have duplicate values in the column. The exception is that an attribute in the column could be empty. For example:

CREATE TABLE newsletter(email VARCHAR(50) UNIQUE);
In the example above, the email must be unique, but it potentially permits no value to be inserted. This is not ideal, as it is the only attribute in the table.

The primary key constraint combines these two constraints so that with NOT NULL, the value cannot be empty, and with UNIQUE, the value also must be distinct. By doing so, we can then use the primary key to uniquely identify any row within the table. In the same example of the newsletter, having the email be the primary key would avoid us having empty values.

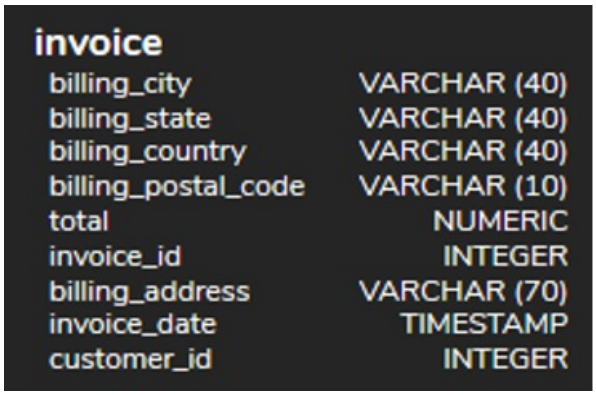
CREATE TABLE newsletter(email VARCHAR(50) PRIMARY KEY);
A primary key does not have to be for a single column. It can combine multiple columns together as a special type of primary key called a composite key. A composite key takes the combination of two or more columns together to uniquely identify a row within a table.

---

2. The FOREIGN KEY Constraint

The FOREIGN KEY constraint is an important one to help maintain referential integrity. It is used to link two tables together. The foreign key in one table refers to the primary key in another table. For example, in the customer table, the customer_id is the primary key to uniquely identify each customer:

The invoice table contains a foreign key (customer_id) to the customer table:



From a coding standpoint, the foreign key would look like this in the invoice table creation:

CONSTRAINT invoice_customer_id_fkey FOREIGN KEY (customer_id) REFERENCES customer (customer_id);
To break it down, the invoice_customer_id_fkey is the constraint name. The column name is then identified with what column the foreign key is applied to. Then we define the table that it is referencing along with the parent key or primary key of that table.

What this means is that we cannot delete a customer from the customer table if at least one invoice row references that customer. This is the default behavior of a foreign key. If we tried to delete from the table, we would get an error like:

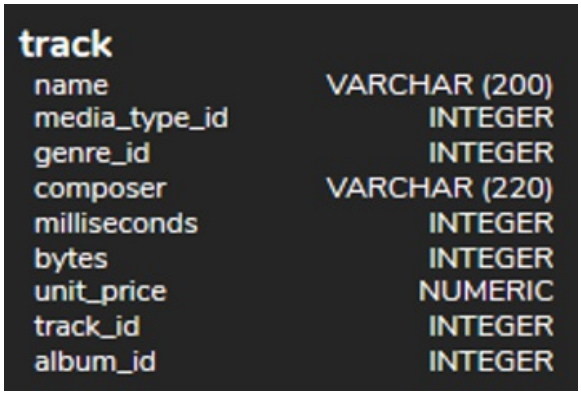
In addition, if we tried to insert or update the customer_id to a value that didn't exist in the customer table, we would get an error like this:

The foreign key enforces referential integrity to ensure that any value for an attribute/column must exist in the referenced table.

---

3. The DEFAULT and CHECK Constraints

The DEFAULT constraint assigns a value to an attribute whenever a new row is added to a table if a value is not set for it. This can be useful to set a base value for an attribute. For example, in our track table:



We can set the unit_price default value to be 0.99 so that there is a default price if none was set.

The CHECK constraint can be used to validate data when an attribute is entered. For example, we could do checks of items such as:

- Check that the unit_price in the track table has a value of >=0 as there should be no negative price.
- Check that the hire_date in the employee table is greater than January 01, 2000, as that was the date that the company opened.
- Check that the customer's email has a standard email format.

Video Transcription

[MUSIC PLAYING] There are many different types of constraints that can be applied to a table. One of them is the primary key that we've discussed previously. With the primary key the value is unique across all the different rows in the table, as well as it can't be null, meaning that it can't be empty. You can also set individual different constraints, including unique and not null.

You can set it up just having not null or just unique. You can also have checks to ensure that the data is within a specific range or criteria. You can also set default, so if you're not passing any values to it, it automatically send it that value as well as foreign keys to link different tables together.

So with this foreign key, based on this referral ID, it's going to reference the customer table referencing the customer ID within that table. So if we're inserting a row into this table, you'll check first that, that value exists first, if it doesn't, then it'll error out. All these will have that check on the data before any data can be inserted into the tables.

[MUSIC PLAYING]

TRY IT

Your turn! Open the SQL tool by clicking on the LAUNCH DATABASE button below. Then enter in one of the examples above and see how it works. Next, try your own choices for which columns you want the query to provide.

SUMMARY

There are many different table constraints that can be applied to a table to help validate the data that is being inserted into the table.

Source: Authored by Vincent Tran