Chapter 3 : Advanced SQL

Aviral Janveja

1 Data Types

Each column in a database table is required to have a name and a data type. The data type of a column defines what value the column can hold : integer, character, date-time, boolean and it also identifies how SQL will interact with the stored data.

It is certainly best for data to be stored in its optimal format from the beginning, but if it isn't, you can always change it in your query. It is particularly common for dates or numbers, for example, to be stored as strings. This becomes problematic when you want to sum a column and you get an error because SQL is reading numbers as strings. When this happens, you can use **CAST** or **CONVERT** to change the data type to a numeric one that will allow you to perform the sum.

You can actually achieve this with two different types of syntax, both of which produce the same result :

```
CAST(column_name AS integer)
or
column_name::integer
```

2 Date Format

Most relational databases format dates as **YYYY-MM-DD**. This format makes a lot of sense because it sorts in chronological order even when the date field is stored as a string.

2.1 Date Subtraction

The following query uses date subtraction to determine how long it took companies to be acquired. Unacquired companies and those without dates entered were filtered out. Note that because the date column is stored as a string, it must be cast as a timestamp before it can be subtracted from another timestamp :

```
SELECT companies_name,
founded_date,
acquired_date,
acquired_date - founded_date::timestamp AS
time_to_acquisition
```

In the example above, the time_to_acquisition column is an interval, not another date. An interval refers to an integer representing a period of time.

2.2 Interval Function

You can introduce intervals using the interval function as well, for example :

```
SELECT companies_name,
founded_date::timestamp + INTERVAL '1 week' AS
plus_one_week
```

The interval is defined using plain-English terms as shown above, like **10 seconds** or **5 months**. Also note that adding or subtracting a date column and an interval column results in another date column.

2.3 Now Function

You can add the current time at which you run the query, into your code using the now function. For example :

```
SELECT companies_name,
   founded_date,
   NOW() - founded_date::timestamp AS founded_time_ago
```

3 Data Cleaning