Basic data types

As you already know, SQL is a language used for working with different types of data organized into a table. Usually, data values from the same column in a table have the same meaning and type

We see that values in the manufacture yearcolumn are integer numbers, values in price are decimal, and values in electricity are boolean.

SQL databases usually require that each column in a database table has a name and a **data type**. The columndata type restricts the set of values that can be stored in the column and defines a set of possible operations on them.

manufacture_year	color	horsepower	price	electricity	...
2018	red	283	34.990	TRUE	...
2019	black	313	50.000	FALSE	...
...	...	...	...	...	...

ANSI standard defines a pretty complex set of data types. Besides, database vendors usually add their non-standard data options.

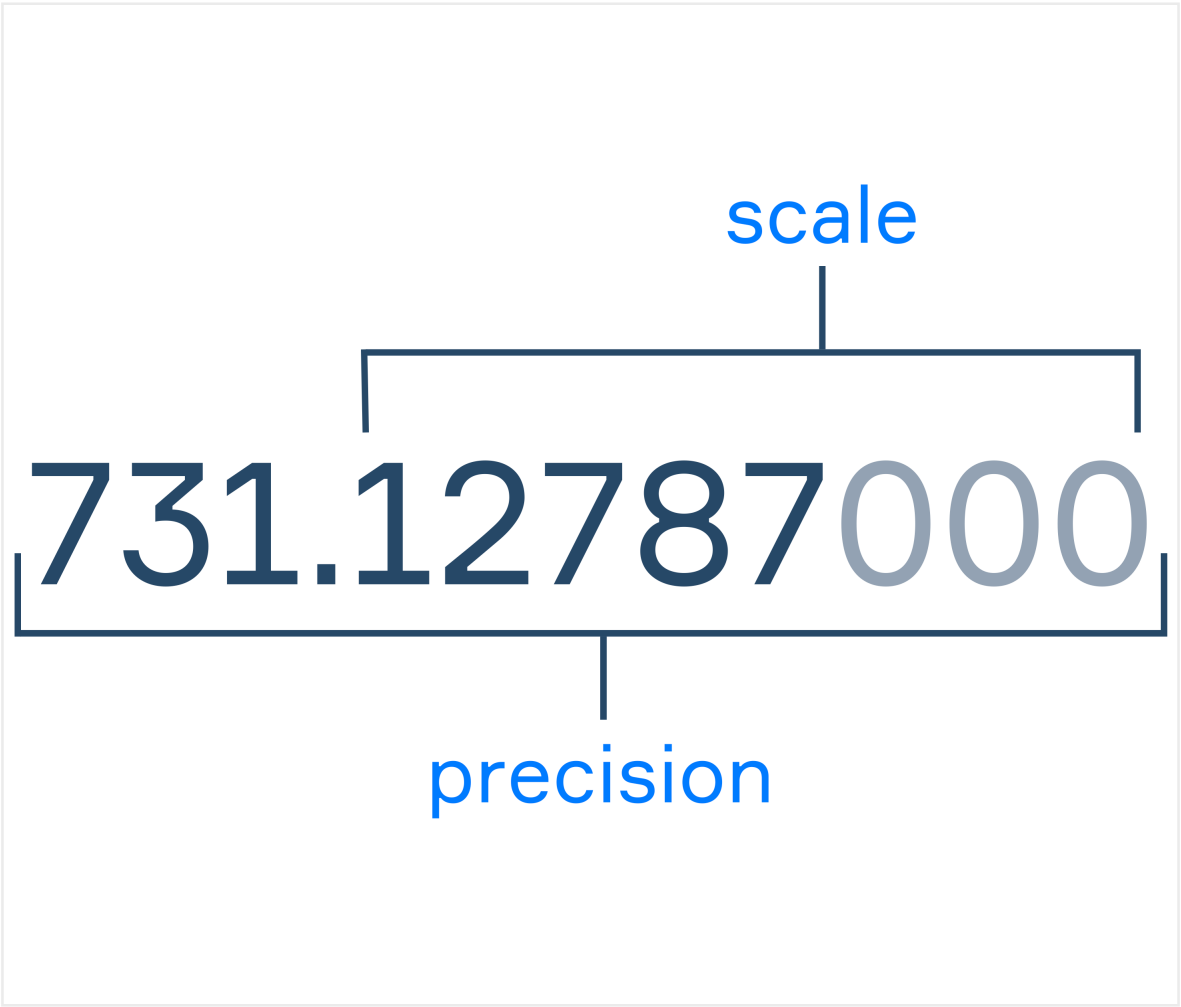
Numerical data types

INTEGER is a numeric data type that represents some range of mathematical integers (usually from -2147483648 to +2147483647).INTEGER type is good for counters, numeric identifiers, and any integer business value you can imagine that fits the scale range.

In everyday life, we usually face decimal numbers quite a lot: for example, when measuring body temperature (36.6 degrees Celsius) or counting our precious finances ($103050.79). SQL supports a special data type for such values – DECIMAL(precision, scale).

This type has two parameters: precision and scale.

- **Scale** is the count of digits to the right of the decimal point.
- **Precision** is the total count of digits in the number.



The FLOAT data type is an approximate numeric data type used for floating-point numbers.

With the FLOAT data type, we can store very large or very small numbers. Also, it is used for calculations that require fast processing. The FLOAT data type has an optional parameter **n** that specifies the precision and storage size (from 1 to 53).

By the way, sometimes in SQL, you can encounter the REAL data type. And so REAL is FLOAT(24), or FLOAT of certain accuracy.

In SQL, the FLOAT data type represents a floating-point number with a specified precision. The precision determines the number of significant digits that can be stored for the floating-point number. The storage size of a FLOAT value depends on its precision.
The precision parameter n for the FLOAT data type specifies the number of bits used to store the mantissa (significant digits) of the floating-point number. The valid range for the precision parameter n is from 1 to 53.
Here's how the precision parameter affects the storage size of a FLOAT value:

- For n less than or equal to 24, a FLOAT value requires 4 bytes of storage.
- For n between 25 and 53 (inclusive), a FLOAT value requires 8 bytes of

storage.

Text
Of course, you may want to process something other than numeric data, and SQL supports a family of data types designed to represent text data. Let's consider one of them, quite universal and basic – VARCHAR(n).

This type represents a string of symbols of varying lengths not longer than n. For example, one can insert the strings apple, plum, and peach into a column with the type VARCHAR(5). The strings orange and banana will exceed the length restriction and the system will either truncate them or generate an error if one tries to insert such long values.

Boolean
The BOOLEAN type represents boolean logic values: either TRUE or FALSE. This simple data type can be utilized for any attributes with flag semantics, for example, whether a client has visited a competitor's site.

Who defines types and how?
As a database user, you should just know the types of table columns you utilize to be able to process them correctly. However, as a software engineer, you should also know how to create a table and define the column types.

Let's consider an example of an SQL query that defines a table census with five columns: id of type INTEGER, name of
type VARCHAR(20), birth_place_latitude of type REAL, year_income of
type DECIMAL(20,2), and is_parent of type BOOLEAN.

```
CREATE TABLE census (
    id INTEGER,
    name VARCHAR(20),
    birth_place_latitude REAL,
    year_income DECIMAL(20,2),
    is_parent BOOLEAN
);
```

One may see the following pattern:

```
CREATE TABLE table_name (
    column_name_1 column_type_1,
    ...,
    column_name_n column_type_n
);
```