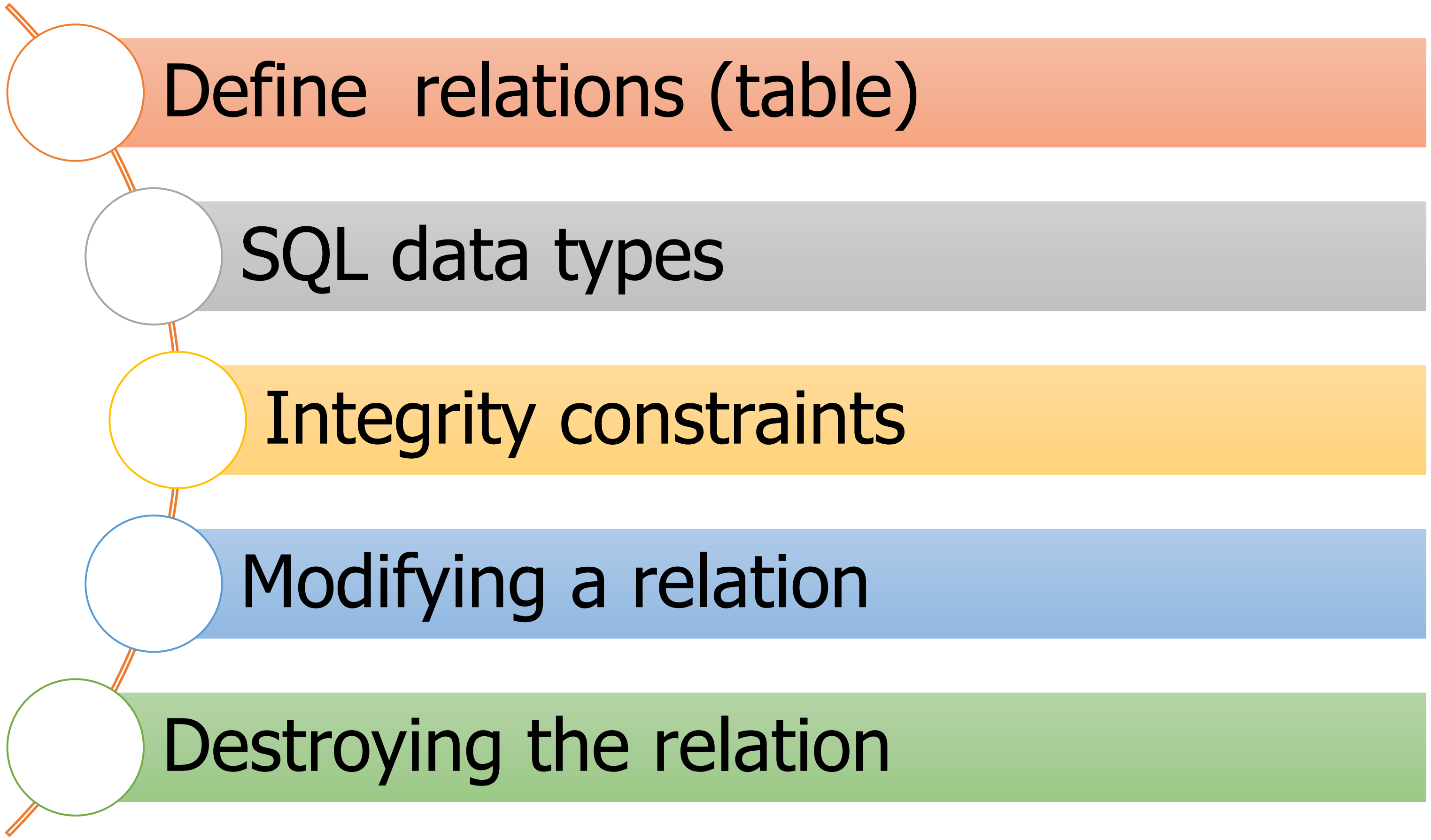
DBMS

Lab-2

Things we will complete today



Define relations (table)

SQL data types

Integrity constraints

Modifying a relation

Destroying the relation

Define A Table

Let’s Define a table in plain English

Create a table named department. The table should have the following information: department name, name of the building where the department is located, it’s budget and the numbers of stuff the department have.

Let’s Define a table in SQL

Create a **table** named **department**. The table should have the following information: **department name**, **name of the building** where the department is located, it’s **budget** and the **numbers of stuff** the department have.

```
CREATE  TABLE  department (
      dept_name       varchar(100),
      building     varchar(120),
      budget       numeric(12,2),
      num_staff    int
)
```

SQL DDL

The set of relations in a database must be specified to the system by means of a data-definition language (DDL). The SQL DDL allows specification of not only a set of relations, but also information about each relation, including:

- The names of each attribute of the schema.
- The types of values associated with each attribute.
- The integrity constraints.
- The set of indices to be maintained for each relation.
- The security and authorization information for each relation.
- The physical storage structure of each relation on disk

SQL Data types

char(n)

A fixed-length character string with user-specified length n. The full form, **character**, can also be used instead.

varchar(n)

A variable-length character string with user-specified maximum length n.

int

An integer. The full form, **integer**, is equivalent.

smallint

A small integer (a machine-dependent subset of the integer type).

SQL Data types

numeric(p,d)

A fixed-point number with user-specified precision. The number consists of p digits (plus a sign), and d of the p digits are to the right of
the decimal point. Thus, numeric(3,1) allows 44.5 to be stored, but neither 444.5 or 0.32 can be stored exactly in a field of this type.

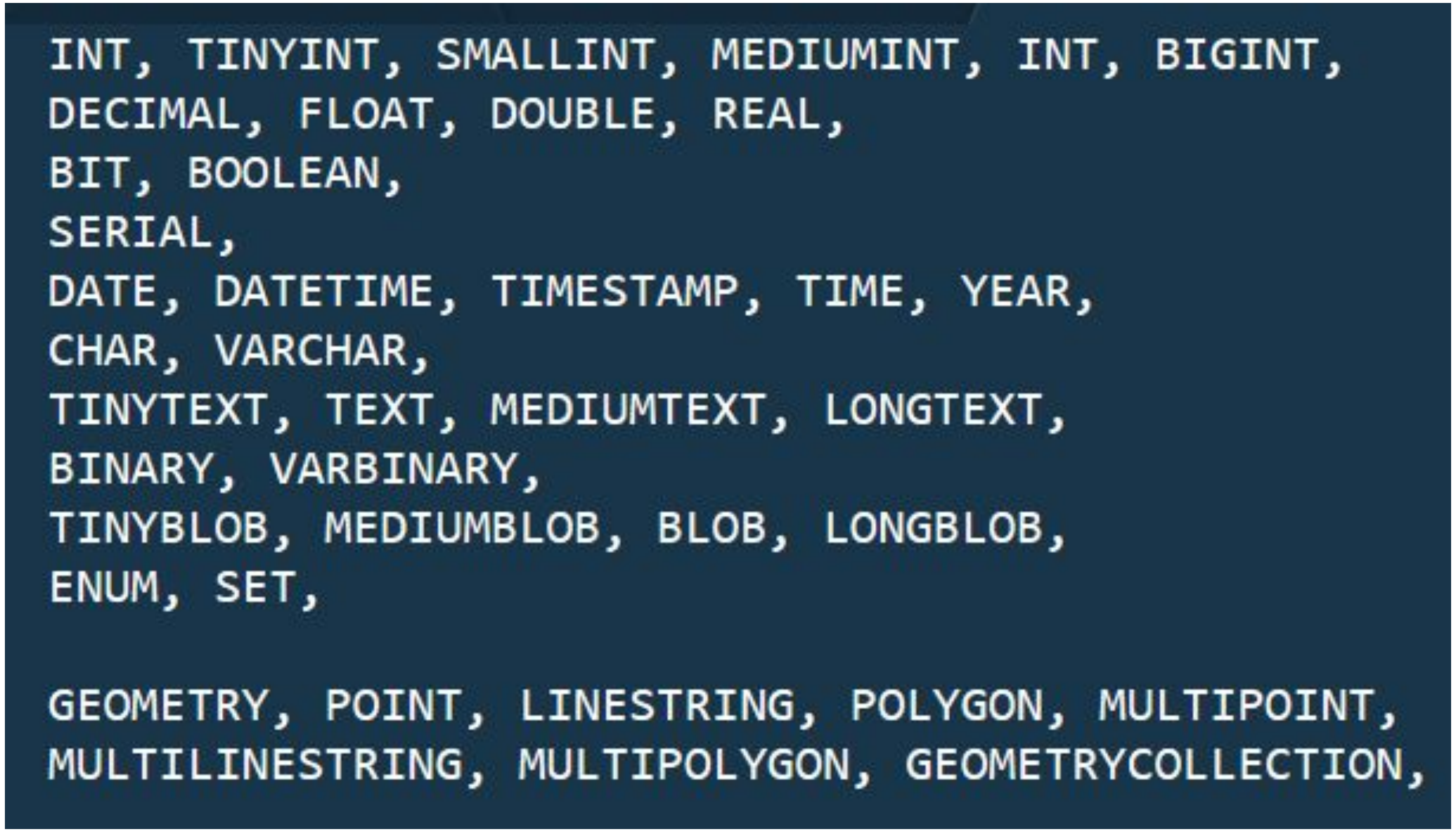
real, **double precision**

Floating-point and double-precision floating-point numbers with machine-dependent precision.

float(n)

A floating-point number, with precision of at least n digits.

MY SQL Data types



```
INT, TINYINT, SMALLINT, MEDIUMINT, INT, BIGINT,
DECIMAL, FLOAT, DOUBLE, REAL,
BIT, BOOLEAN,
SERIAL,
DATE, DATETIME, TIMESTAMP, TIME, YEAR,
CHAR, VARCHAR,
TINYTEXT, TEXT, MEDIUMTEXT, LONGTEXT,
BINARY, VARBINARY,
TINYBLOB, MEDIUMBLOB, BLOB, LONGBLOB,
ENUM, SET,

GEOMETRY, POINT, LINESTRING, POLYGON, MULTIPOINT,
MULTILINESTRING, MULTIPOLYGON, GEOMETRYCOLLECTION,
```

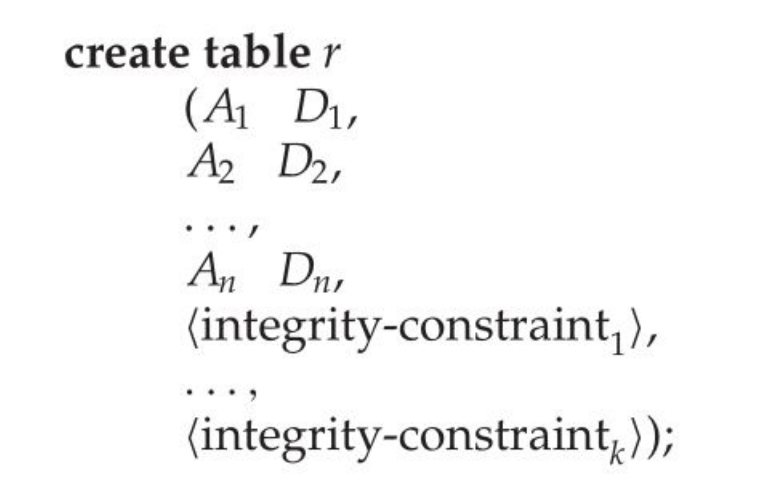
For More About My SQL Datatypes

https://www.tutorialspoint.com/mysql/mysql-data-types.htm

Integrity Constraints in Create Table

- **not null**
- **primary key (A1, …, An)**
- **Many more ….**

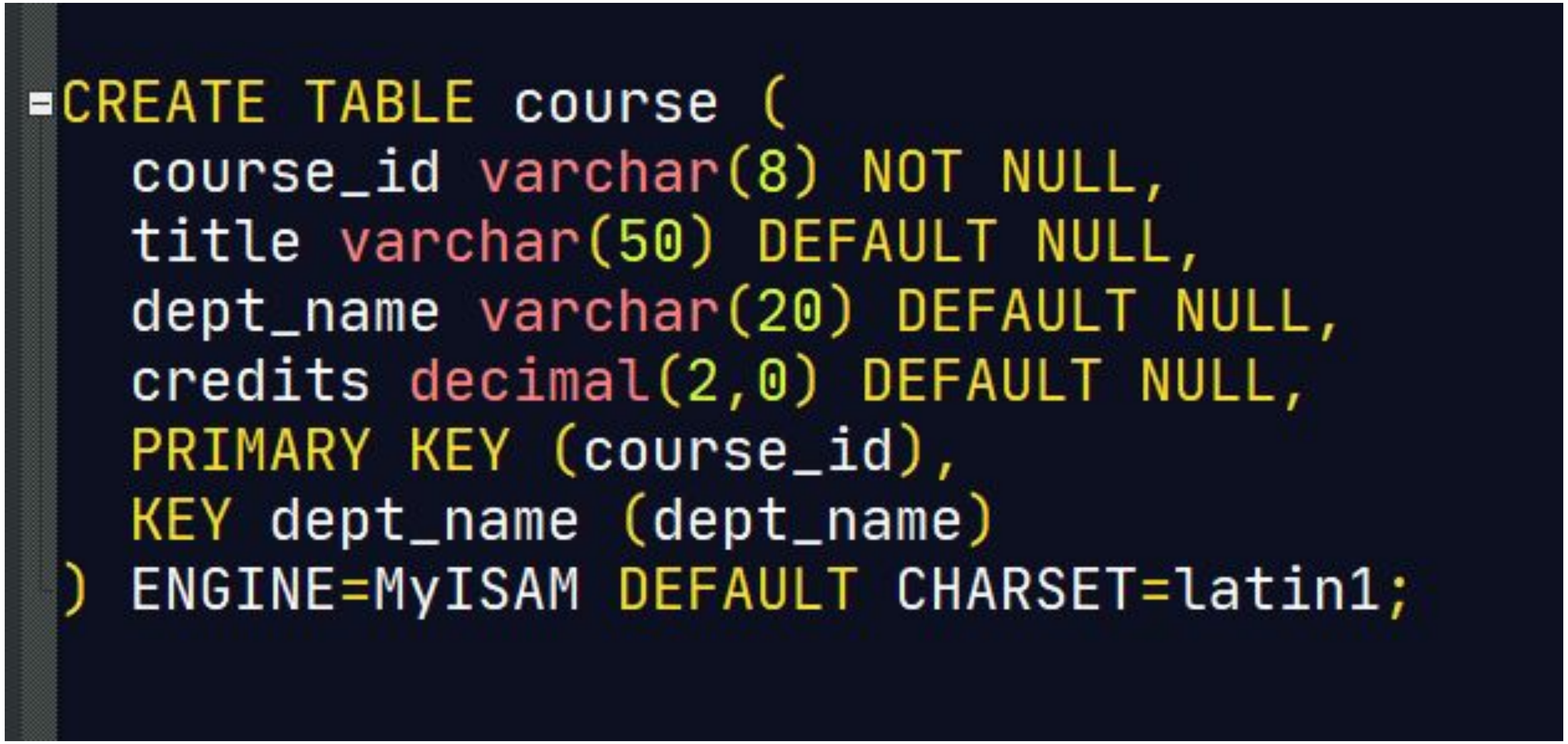
```
CREATE  TABLE  department (
      dept_name       varchar(100),
      building      varchar(120) not null,
      budget        numeric(12,2) not null,
      num_staff     int,
      primary key (        dept_name)
)
```



create table *r*
(A_1 D_1,
A_2 D_2,
…,
A_n D_n,
⟨integrity-constraint$_1$⟩,
…,
⟨integrity-constraint$_k$⟩);

primary key declaration on an attribute automatically ensures **not null** in SQL-92 onwards, needs to be explicitly stated in SQL-89

(Almost) Full Definition a table in SQL

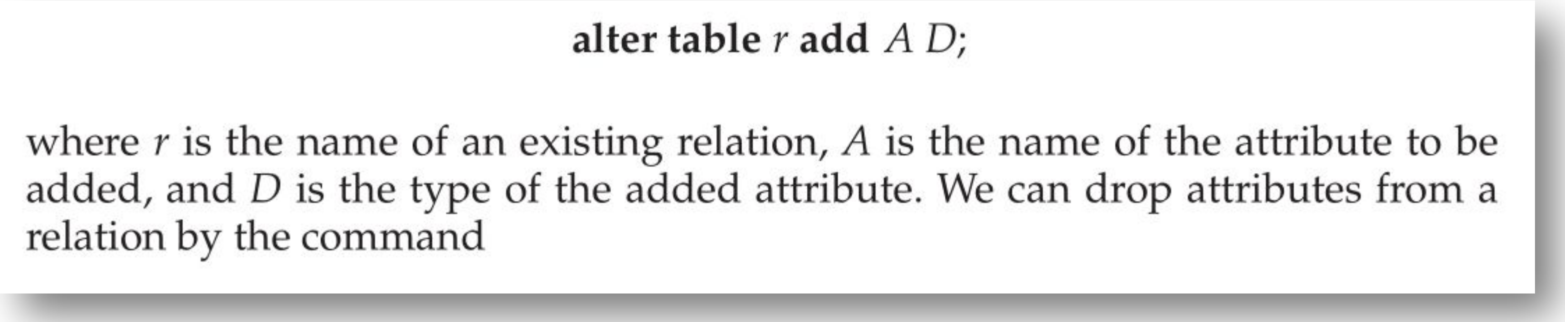


```
CREATE TABLE course (
  course_id varchar(8) NOT NULL,
  title varchar(50) DEFAULT NULL,
  dept_name varchar(20) DEFAULT NULL,
  credits decimal(2,0) DEFAULT NULL,
  PRIMARY KEY (course_id),
  KEY dept_name (dept_name)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;
```

Schema modification

- Adding/ Removing a column
- Modifying column
- Deleting the whole scheme

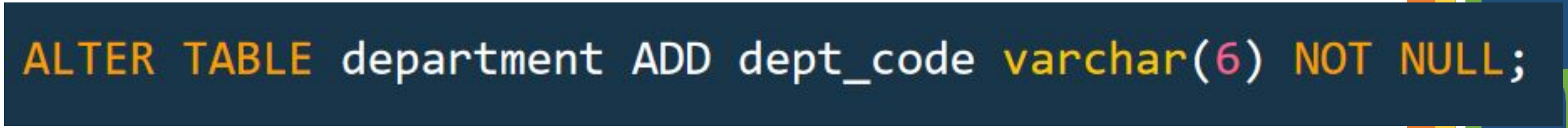
Adding a column



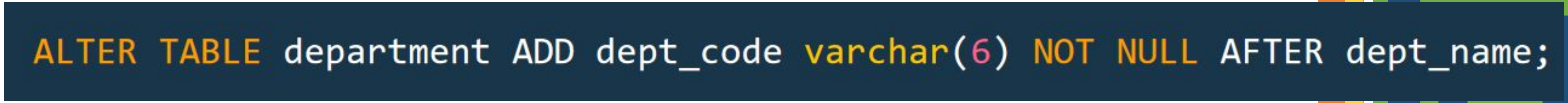
alter table *r* **add** *A D*;

where *r* is the name of an existing relation, *A* is the name of the attribute to be added, and *D* is the type of the added attribute. We can drop attributes from a relation by the command

Add a column for department code in **department** table

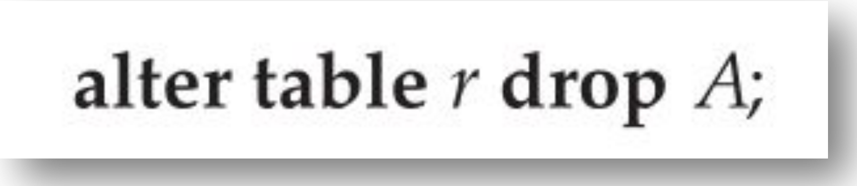


```
ALTER TABLE department ADD dept_code varchar(6) NOT NULL;
```



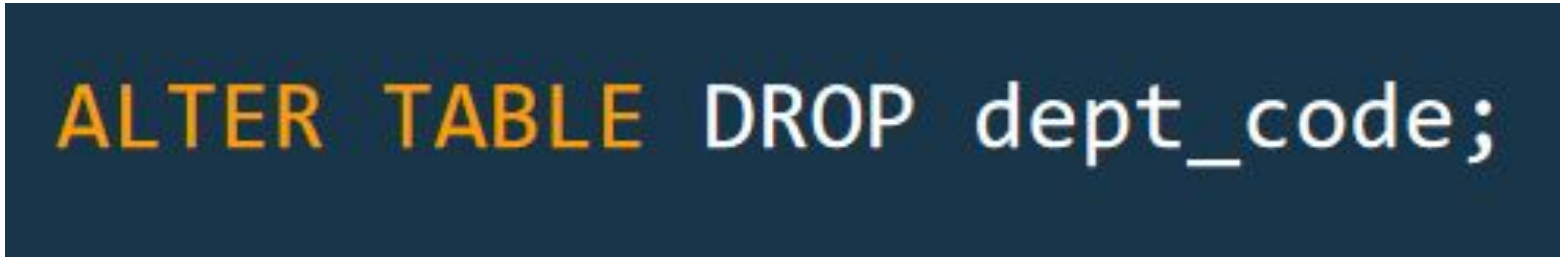
```
ALTER TABLE department ADD dept_code varchar(6) NOT NULL AFTER dept_name;
```

Removing a column



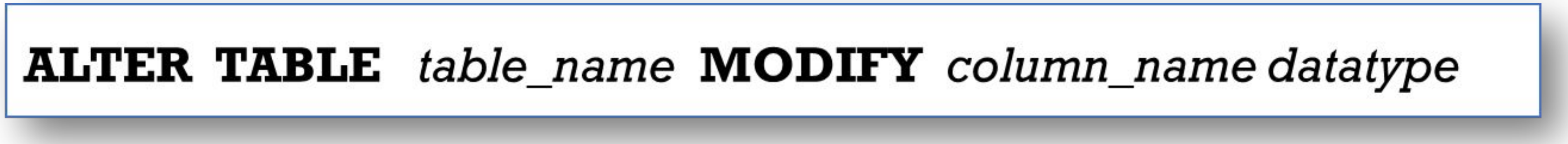
```
alter table r drop A;
```

Remove the column dept_code from **department** table



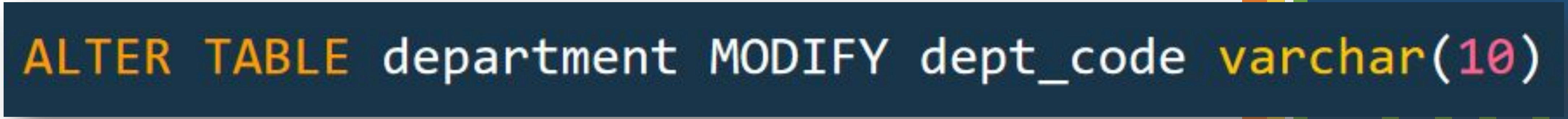
```
ALTER TABLE DROP dept_code;
```

Modifying a column



```
ALTER TABLE table_name MODIFY column_name datatype
```

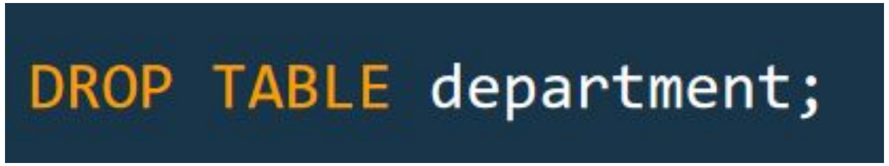
change column length of **dept_code**



```
ALTER TABLE department MODIFY dept_code varchar(10)
```

08/16/2020 13:32

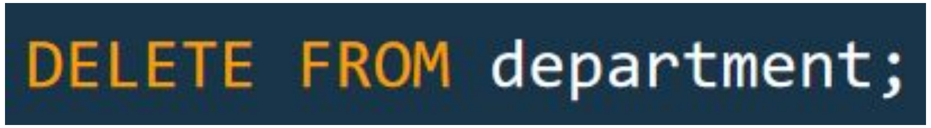
Deleting the WHOLE table



```
DROP TABLE department;
```

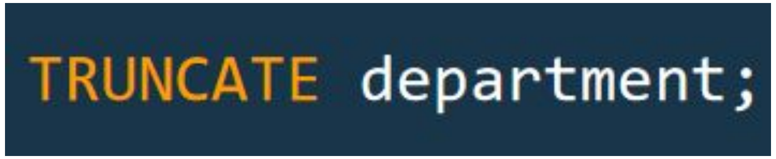
Deleting only data (Keep table structure)

Two option



```
DELETE FROM department;
```

or



```
TRUNCATE department;
```

Demo.

End.