SUCCEED WE MUST

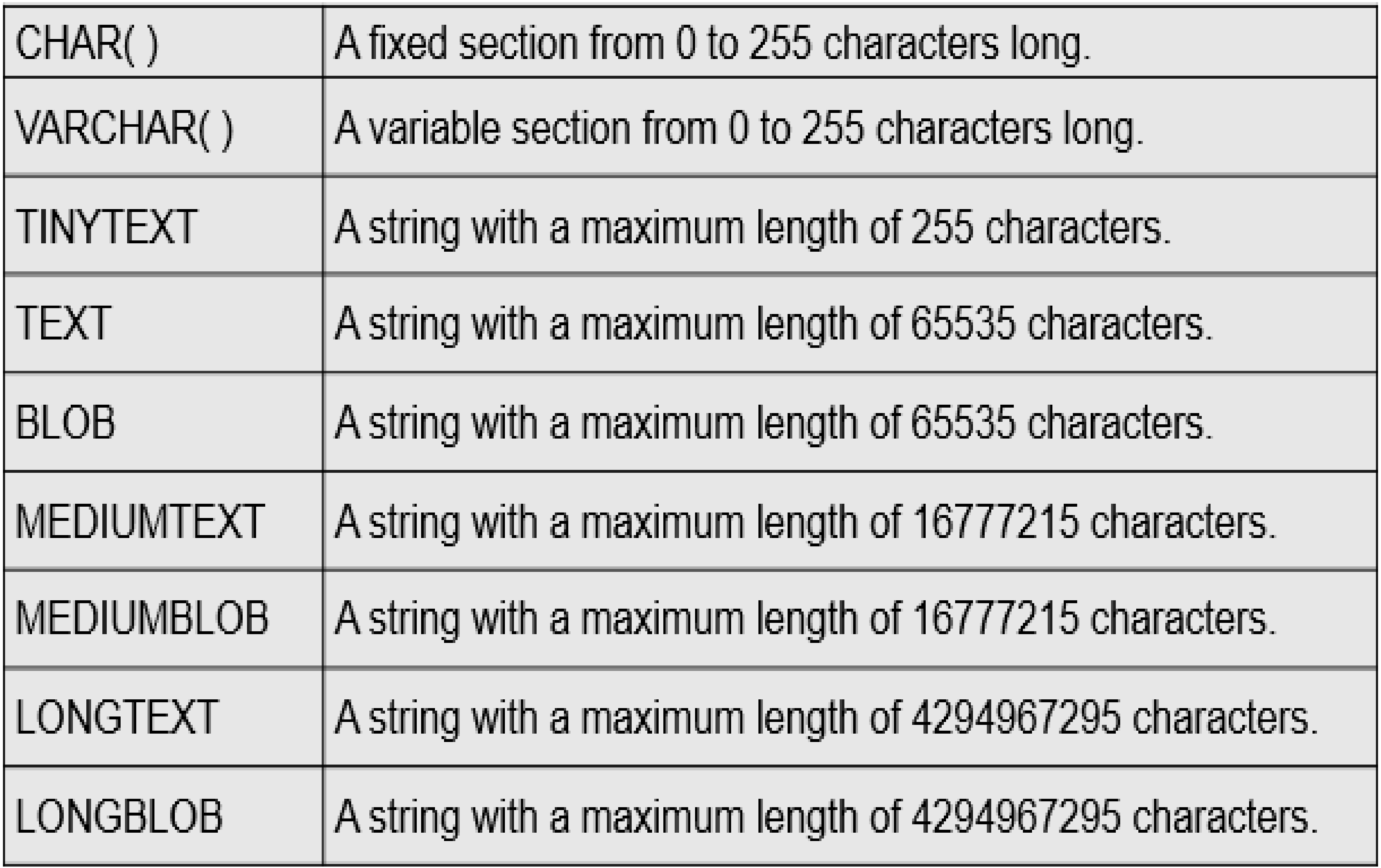
DATABASE PROGRAMING BIT-2212

Lecture 2

SUCCEED WE MUST

Mwavu Rogers
Email: mwavurogers@gmail.com
Phone 0773426328 and 0700497421
INSTITUTE OF COMPUTER SCIENCE
DEPARTMENT OF INFORMATION TECHNOLOGY

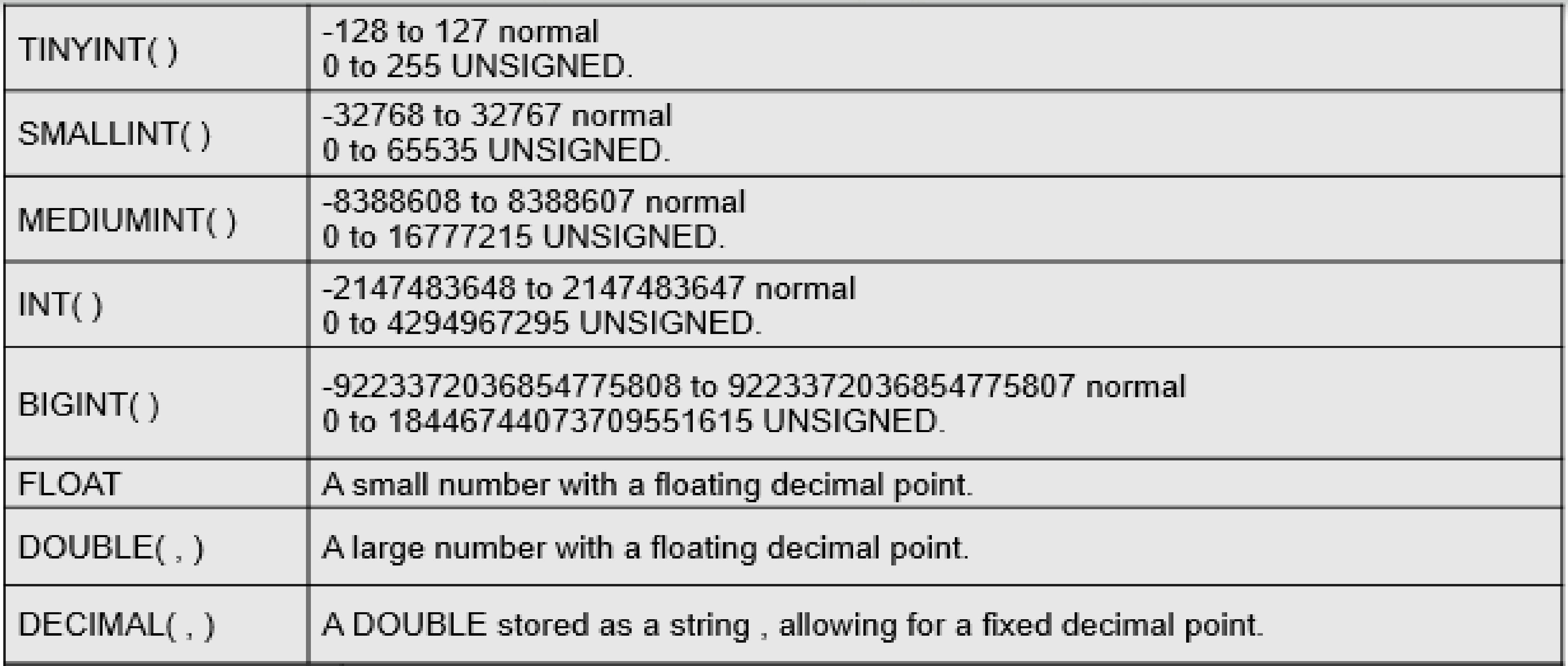
Creating a table: Data Types



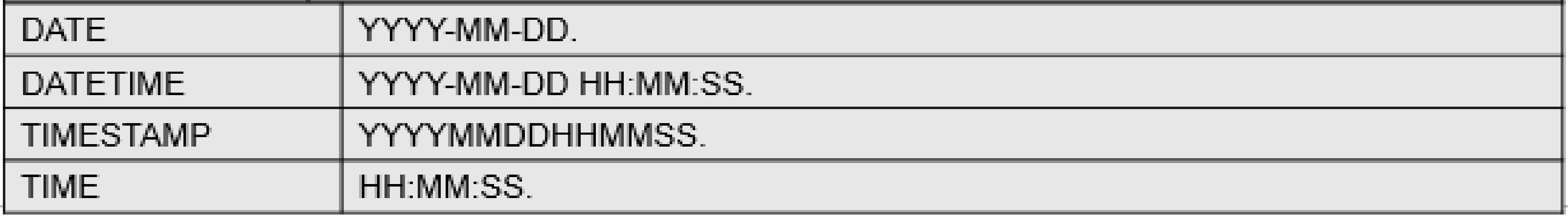
CHAR( )	A fixed section from 0 to 255 characters long.
VARCHAR( )	A variable section from 0 to 255 characters long.
TINYTEXT	A string with a maximum length of 255 characters.
TEXT	A string with a maximum length of 65535 characters.
BLOB	A string with a maximum length of 65535 characters.
MEDIUMTEXT	A string with a maximum length of 16777215 characters.
MEDIUMBLOB	A string with a maximum length of 16777215 characters.
LONGTEXT	A string with a maximum length of 4294967295 characters.
LONGBLOB	A string with a maximum length of 4294967295 characters.

SUCCEED WE MUST

Data Types Number Types & Date Time Types Number Types & DateTime Types



TINYINT( )	-128 to 127 normal 0 to 255 UNSIGNED.
SMALLINT( )	-32768 to 32767 normal 0 to 65535 UNSIGNED.
MEDIUMINT( )	-8388608 to 8388607 normal 0 to 16777215 UNSIGNED.
INT( )	-2147483648 to 2147483647 normal 0 to 4294967295 UNSIGNED.
BIGINT( )	-9223372036854775808 to 9223372036854775807 normal 0 to 18446744073709551615 UNSIGNED.
FLOAT	A small number with a floating decimal point.
DOUBLE(,)	A large number with a floating decimal point.
DECIMAL(,)	A DOUBLE stored as a string , allowing for a fixed decimal point.

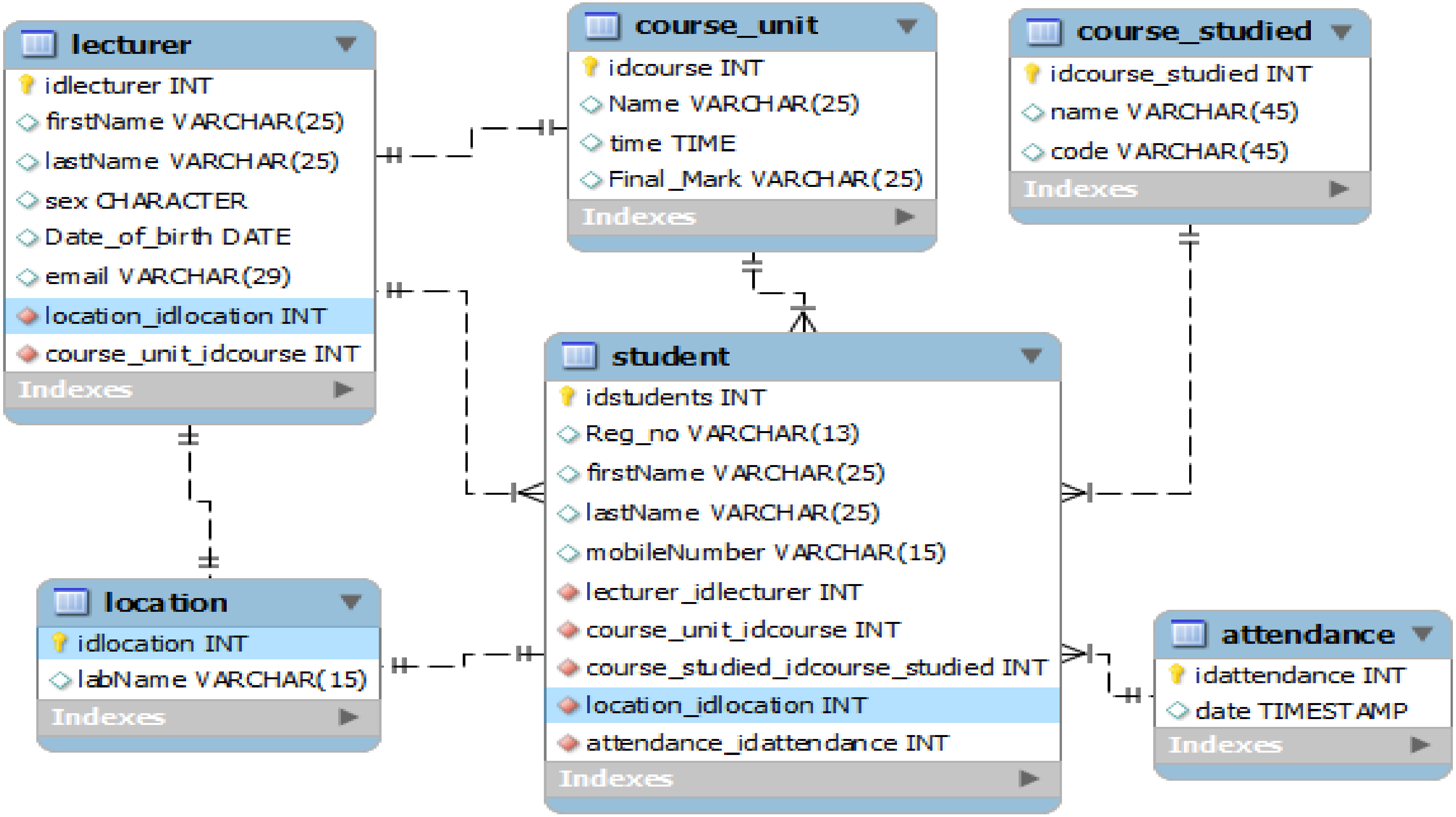


DATE	YYYY-MM-DD.
DATETIME	YYYY-MM-DD HH:MM:SS.
TIMESTAMP	YYYYMMDDHHMMSS.
TIME	HH:MM:SS.

MySQL Storage Engines

- Use the “show engines ” command: **MySQL 5.6 Supported storage Engines**
- InnoDB: A transaction-safe (ACID compliant) storage engine for MySQL that has commit, rollback, and crash-recovery capabilities to protect user data. To maintain data integrity, InnoDB also supports FOREIGN KEY referential-integrity constraints. InnoDB is the default storage engine in MySQL 5.6.
- MyISAM: it is often used in read-only or read-mostly workloads in Web and data warehousing configurations.
- Memory: Stores all data in RAM, for fast access in environments that require quick lookups of non-critical data.
- **CSV: Its tables are really text files with comma-separated values. CSV tables let you import or dump data in CSV format, to exchange data with scripts and applications that read and write that same format. Because CSV tables are not indexed, you typically keep the data in InnoDB tables during normal operation, and only use CSV tables during the import or export stage.**

Creating a table Example- student monitoring tool



lecturer
- idlecturer INT
- firstName VARCHAR(25)
- lastName VARCHAR(25)
- sex CHARACTER
- Date_of_birth DATE
- email VARCHAR(29)
- location_idlocation INT
- course_unit_idcourse INT
- Indexes

course_unit
- idcourse INT
- Name VARCHAR(25)
- time TIME
- Final_Mark VARCHAR(25)
- Indexes

course_studied
- idcourse_studied INT
- name VARCHAR(45)
- code VARCHAR(45)
- Indexes

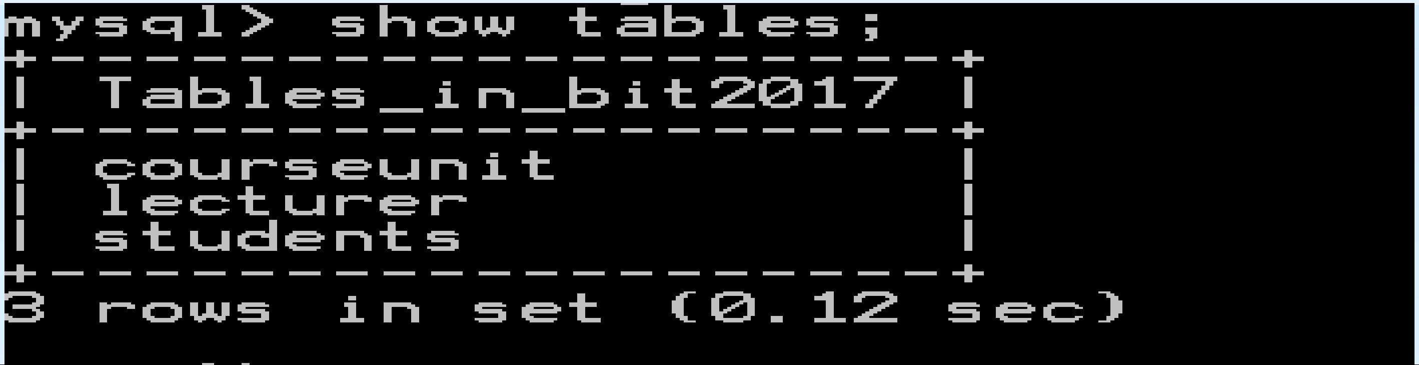
student
- idstudents INT
- Reg_no VARCHAR(13)
- firstName VARCHAR(25)
- lastName VARCHAR(25)
- mobileNumber VARCHAR(15)
- lecturer_idlecturer INT
- course_unit_idcourse INT
- course_studied_idcourse_studied INT
- location_idlocation INT
- attendance_idattendance INT
- Indexes

location
- idlocation INT
- labName VARCHAR(15)
- Indexes

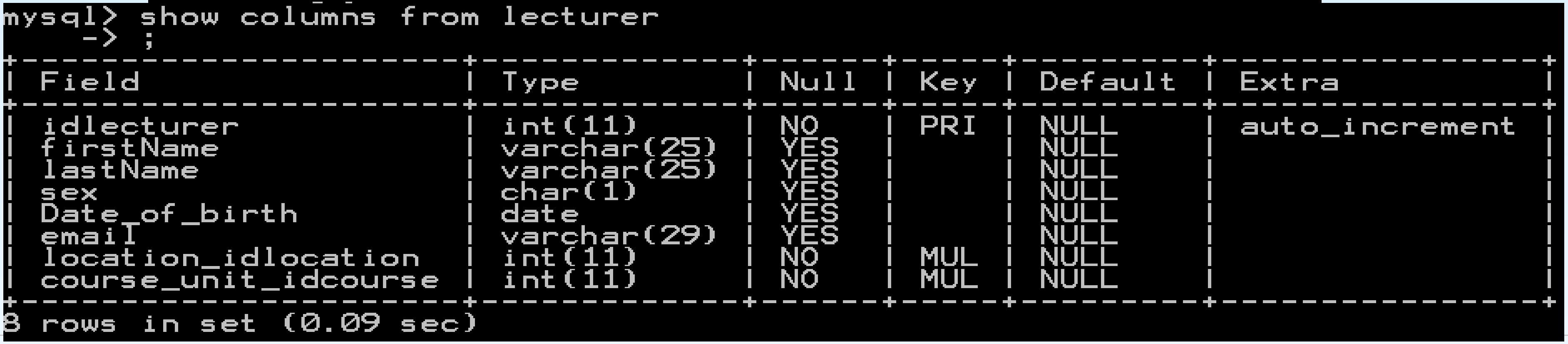
attendance
- idattendance INT
- date TIMESTAMP
- Indexes

Creating a Table

- Again once the SQL has been implemented the table can be viewed with the SHOW command. SHOW TABLES : displays all tables
  - SHOW COLUMNS FROM table_name : displays the columns for a particular table



```
mysql> show tables;
+-------------------+
| Tables_in_bit2017 |
+-------------------+
| courseunit        |
| lecturer          |
| students          |
+-------------------+
3 rows in set (0.12 sec)
```



```
mysql> show columns from lecturer
    -> ;
+----------------------+-------------+------+-----+---------+----------------+
| Field                | Type        | Null | Key | Default | Extra          |
+----------------------+-------------+------+-----+---------+----------------+
| idlecturer           | int(11)     | NO   | PRI | NULL    | auto_increment |
| firstName            | varchar(25) | YES  |     | NULL    |                |
| lastName             | varchar(25) | YES  |     | NULL    |                |
| sex                  | char(1)     | YES  |     | NULL    |                |
| Date_of_birth        | date        | YES  |     | NULL    |                |
| email                | varchar(29) | YES  |     | NULL    |                |
| location_idlocation  | int(11)     | NO   | MUL | NULL    |                |
| course_unit_idcourse | int(11)     | NO   | MUL | NULL    |                |
+----------------------+-------------+------+-----+---------+----------------+
8 rows in set (0.09 sec)
```

The ALTER statement

ALTER statements can be used to modify a number of attributes of a database including:

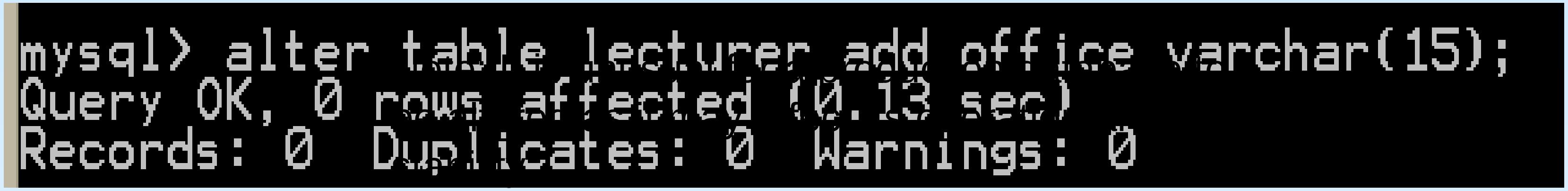
- The Database itself
- Tables
- Views

and so on

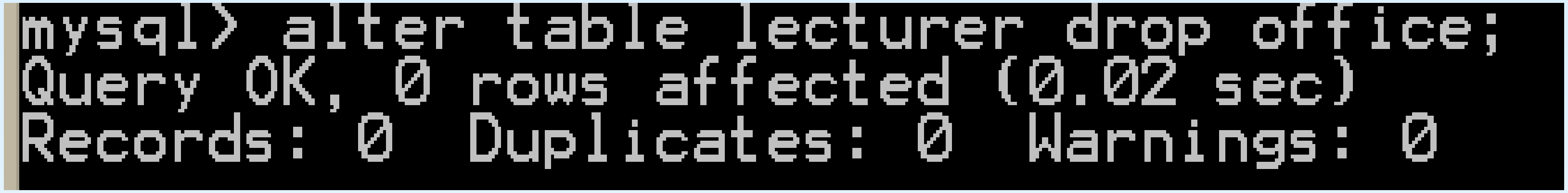
Data Definition Language - ALTER

- ALTER DATABASE

  This command allows you to modify a number of entities such as adding, removing, and editing column names.



```
mysql> alter table lecturer add office varchar(15);
Query OK, 0 rows affected (0.13 sec)
Records: 0  Duplicates: 0  Warnings: 0
```



```
mysql> alter table lecturer drop office;
Query OK, 0 rows affected (0.02 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

Note: It is unusual for a production database to be altered as it may effect its operation and stability

SUCCEED WE MUST

Data Definition Language - ALTER

- The RENAME statement

Sometimes it is applicable to RENAME content, the RENAME DATABASE & RENAME TABLE statements exists but are deemed as dangerous therefore we use the ALTER statement to carry out such renaming events.

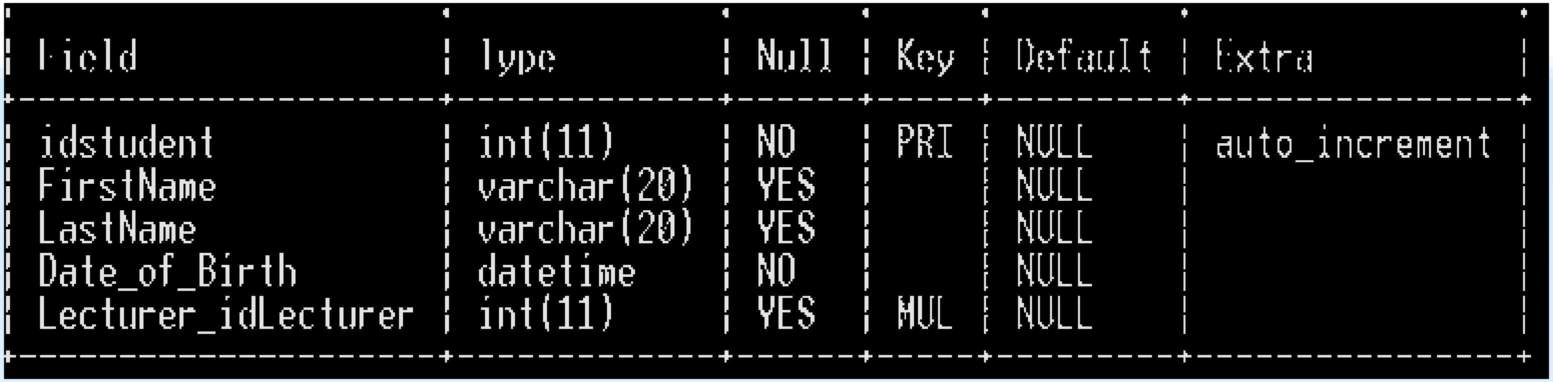
ALTER TABLE `bit3204_databases`.`new_table` RENAME TO `bit3204_databases`.`new_table_renamed`;

Renaming a database is not recommended

Home Work

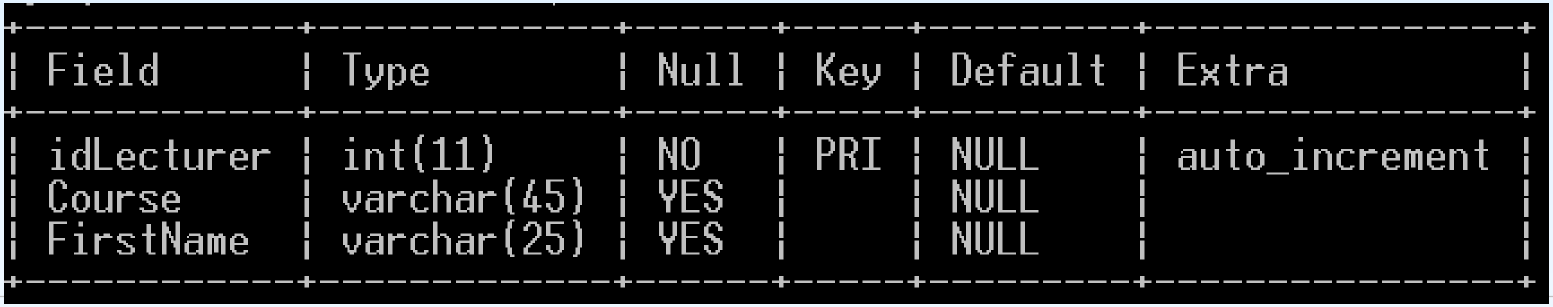
- Consider a database called **Bit2_Class** with two tables as below.

Student_reg



```
| Field               | Type        | Null | Key | Default | Extra          |
+---------------------+-------------+------+-----+---------+----------------+
| idstudent           | int(11)     | NO   | PRI | NULL    | auto_increment |
| FirstName           | varchar(20) | YES  |     | NULL    |                |
| LastName            | varchar(20) | YES  |     | NULL    |                |
| Date_of_Birth       | datetime    | NO   |     | NULL    |                |
| Lecturer_idLecturer | int(11)     | YES  | MUL | NULL    |                |
+---------------------+-------------+------+-----+---------+----------------+
```

Lecturer



```
+------------+-------------+------+-----+---------+----------------+
| Field      | Type        | Null | Key | Default | Extra          |
+------------+-------------+------+-----+---------+----------------+
| idLecturer | int(11)     | NO   | PRI | NULL    | auto_increment |
| Course     | varchar(45) | YES  |     | NULL    |                |
| FirstName  | varchar(25) | YES  |     | NULL    |                |
+------------+-------------+------+-----+---------+----------------+
```

SUCCEED WE MUST

HOME WORK

- Using the SQL, write statements that would implement the following;

1. Establishing the data base.
2. Establish the students’ table
3. Inserting a record in the students’ table
4. Adding an attribute called Responsibility in the Lecturers’ table
5. Changing the lecturers table name to Mentor
6. Showing the youngest and oldest students