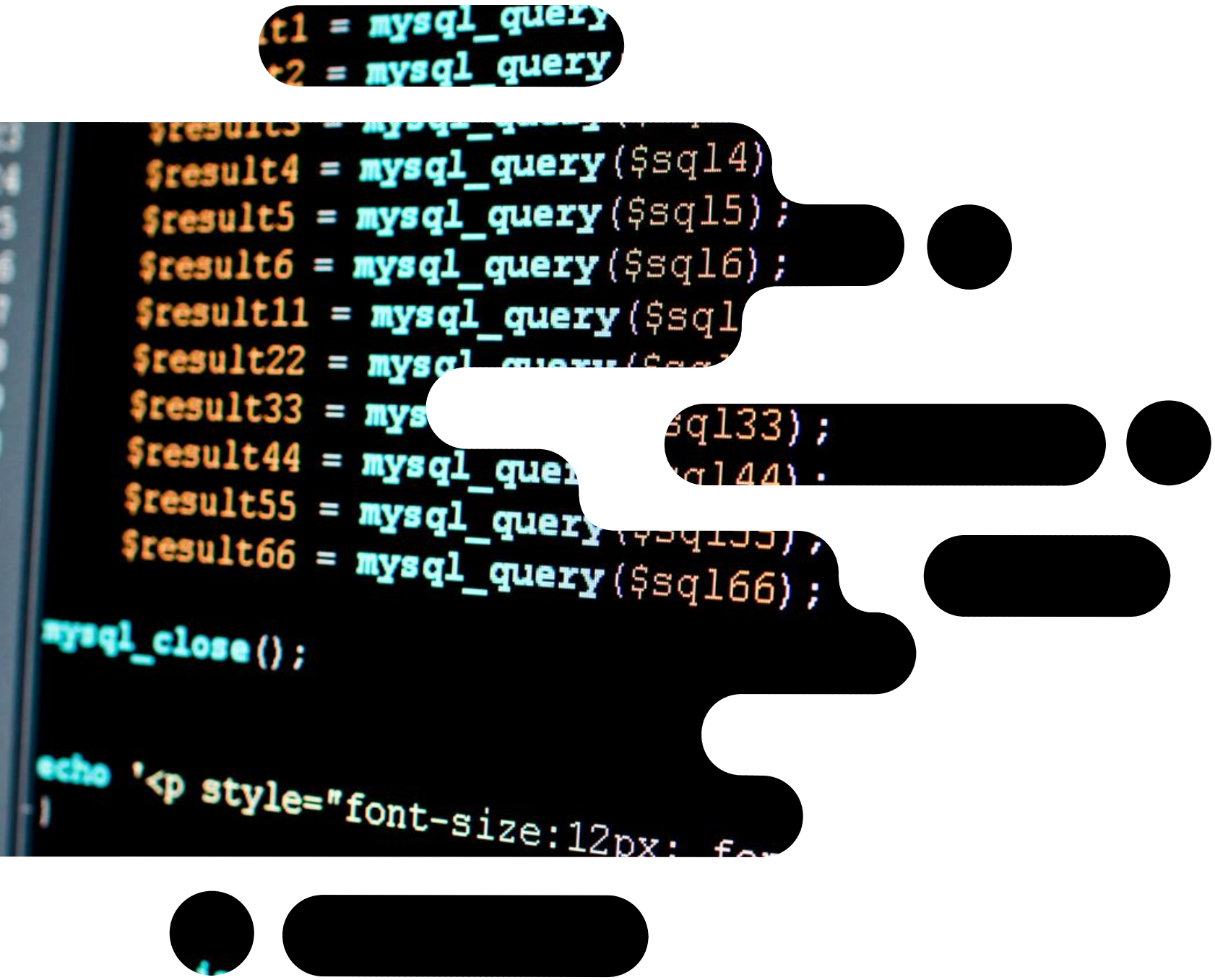
MySQL- DDL

Course Objective

- To create ,drop and alter the tables in MySQL Database.
- To implement constraints in table while creating or altering the table.

Session Objective

- DDL – create, alter, drop & truncate.
- Constraints and its types.



HEXAWA

Database

Commercial Data Bases

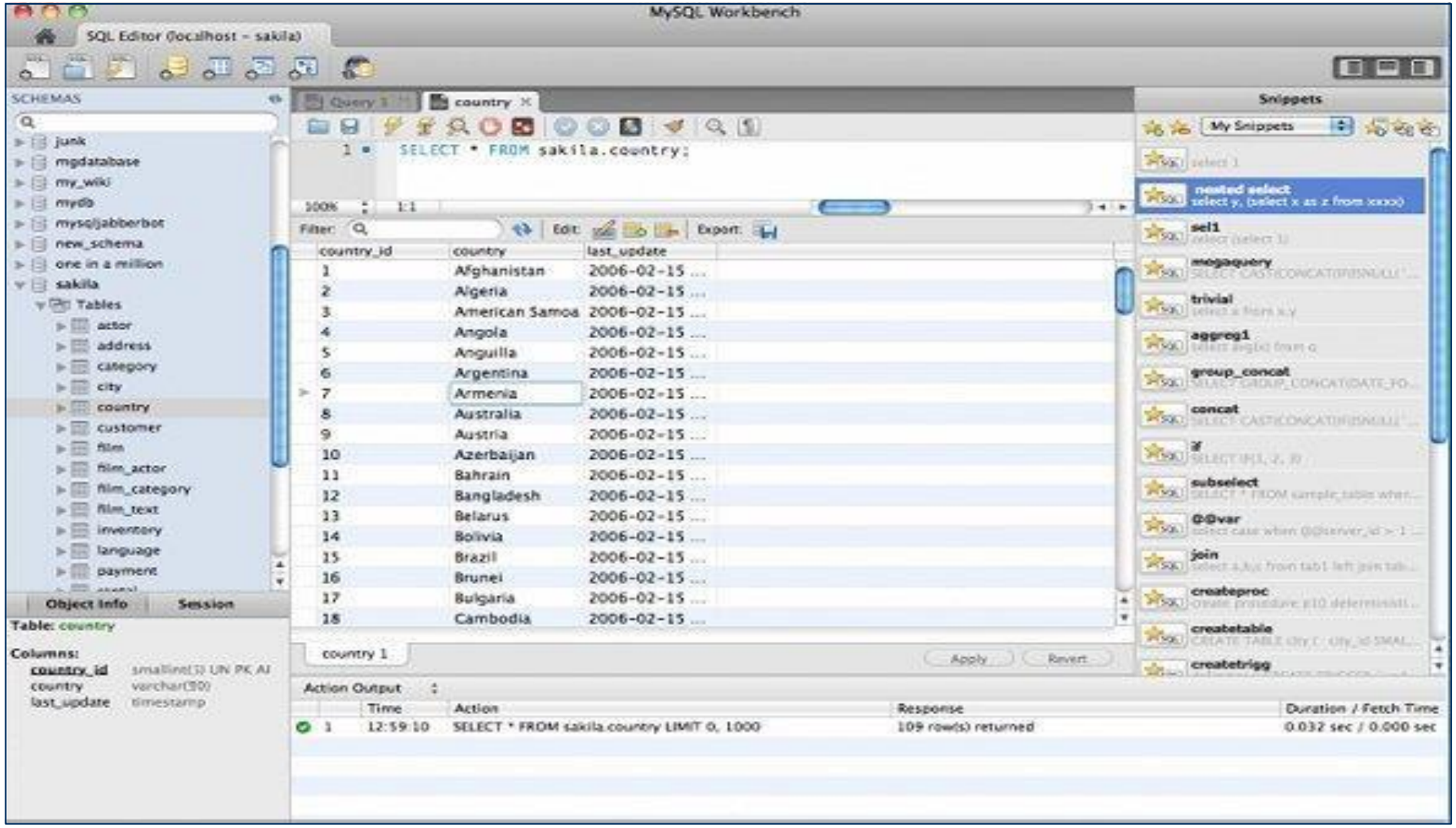
MySQL Introduction

- MySQL is a database management system used for many small and big businesses.
- MySQL is developed, marketed and supported by MySQL AB - a Swedish company.
- MySQL is a open source database.
- MySQL supports large databases, up to 50 million rows or more in a table. The default file size limit for a table is 4GB, but you can increase
- to a theoretical limit of 8 million terabytes (TB).

Database Client GUI

Database Client GUI
Workbench
Sequel Pro
HeidiSQL
SQLyog
SQLWave
DBTools Manager
MyDB Studio
Navicat for MySQL

Database Client GUI - Workbench



Show Database

```
mysql> SHOW DATABASES;


+--------------+
 | Database |
 +--------------+
 | mysql      |
 | test       |
 +____________+

2 rows in set (0.13 sec)
```

Show databases command

Display all database instances in MySQL database

Create Database

You can create and drop a MySQL database instance by using My SQL Workbench by using the command

Create Database:

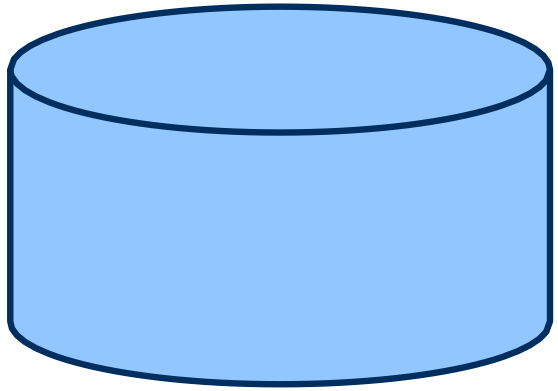
➔ Create database <<Database Name >>

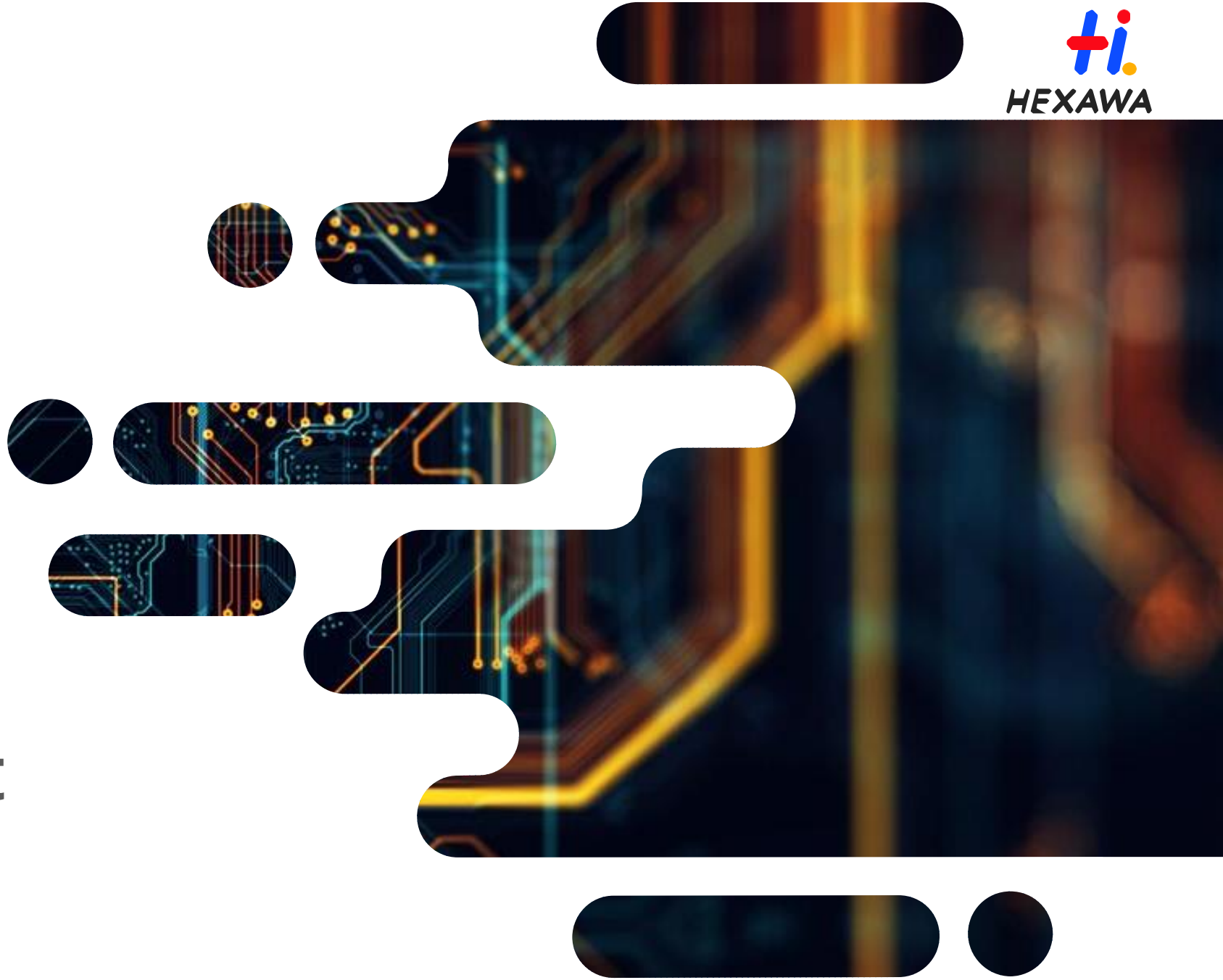
```
Create database Training
```

Drop Database :

➔ Drop Database <<Database Name >>

```
Drop database Training
```





DDL Statement

DDL

- DDL is short name of Data Definition Language.
- DDL deals with database schemas like table.

DDL Commands

- **CREATE –** create the structure of a data base object (ex: table).
- **ALTER –** alters the structure of the existing database.
- **DROP** – delete objects from the database.
- **TRUNCATE –** remove all records from a table, including all spaces allocated for the records are removed.

Create Table

- CREATE TABLE Table_Name (column_specifications)

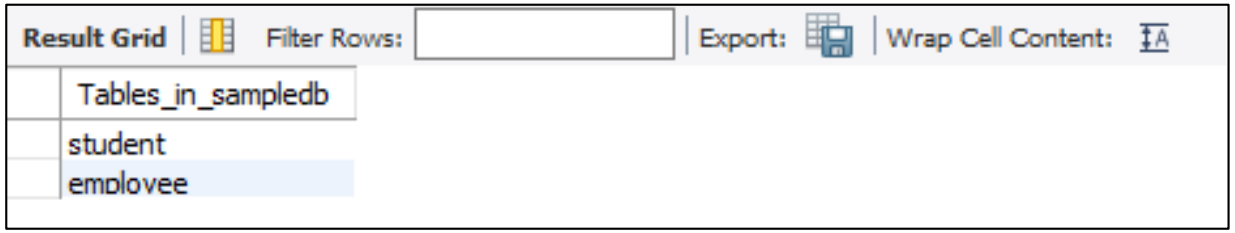
- **Example**

```
CREATE TABLE student  (
        student_ID INT UNSIGNED   NOT NULL,
        name        VARCHAR(20)    NOT NULL,
        major       VARCHAR(50),
        grade       VARCHAR(5)
      );
```

Display Table Structure

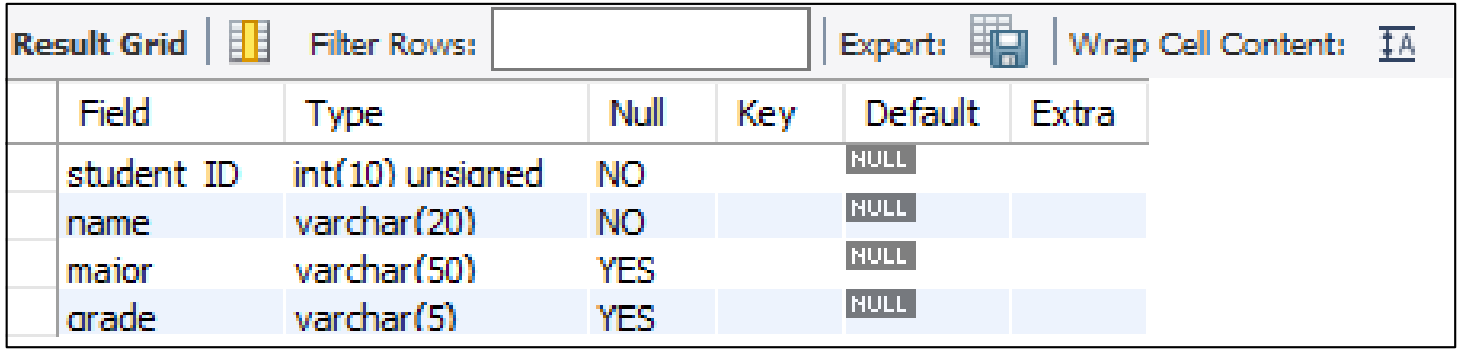
- **show tables :** command display the tables from current database

  SHOW tables;



- **describe** : command display the structure of the table

  DESCRIBE student; / DESC student;



Modify Table Structure

- alter the existing structure of the table

```
ALTER TABLE student ADD PRIMARY KEY (student_ID);
```

```
15  14:35:59  alter table student add primary key (student...  0 row(s) affected Records: 0  Duplicates: 0  Warnings: 0
```

```
DESCRIBE student;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
student_ID	int(10) unsigned	NO	PRI	NULL	
name	varchar(20)	NO		NULL	
maior	varchar(50)	YES		NULL	
grade	varchar(5)	YES		NULL	

Drop

Syntax:

DROP TABLE table_name;

Example

DROP TABLE student;

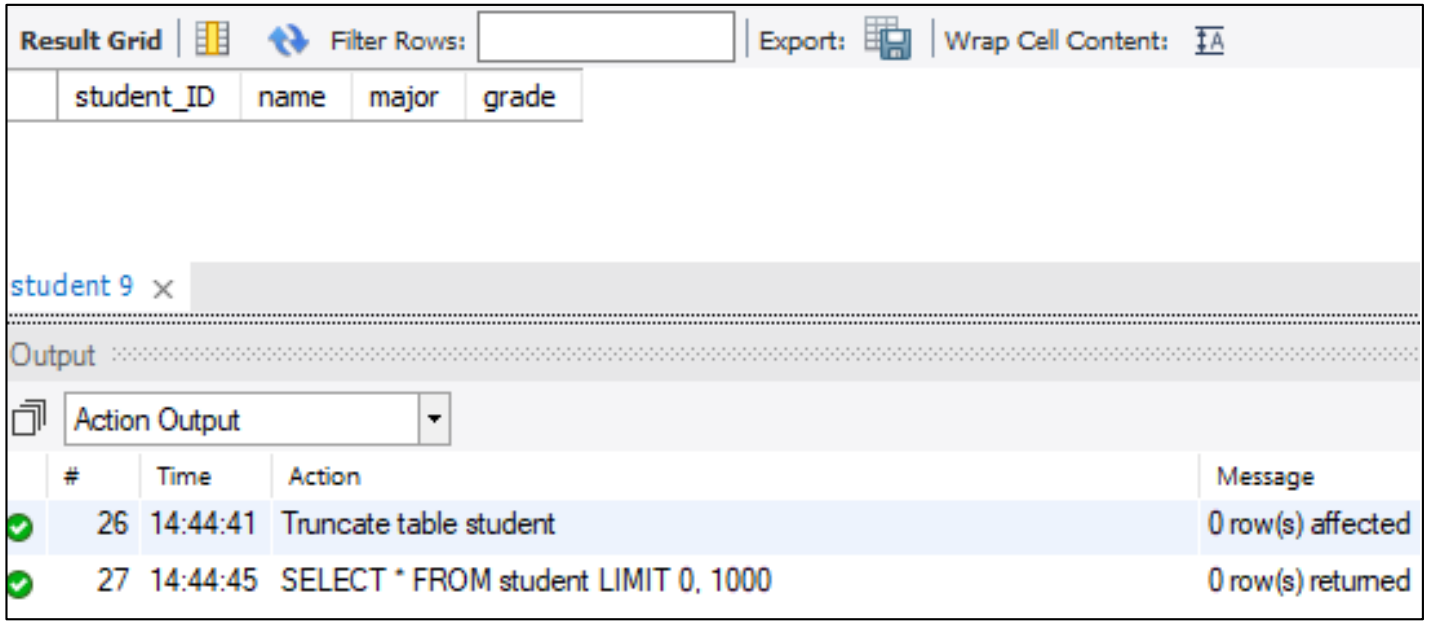
✔	23	14:42:19	drop table student	0 row(s) affected
✖	24	14:42:22	SELECT * FROM student LIMIT 0, 1000	Error Code: 1146. Table 'sampledb.student' doesn't exist

Truncate

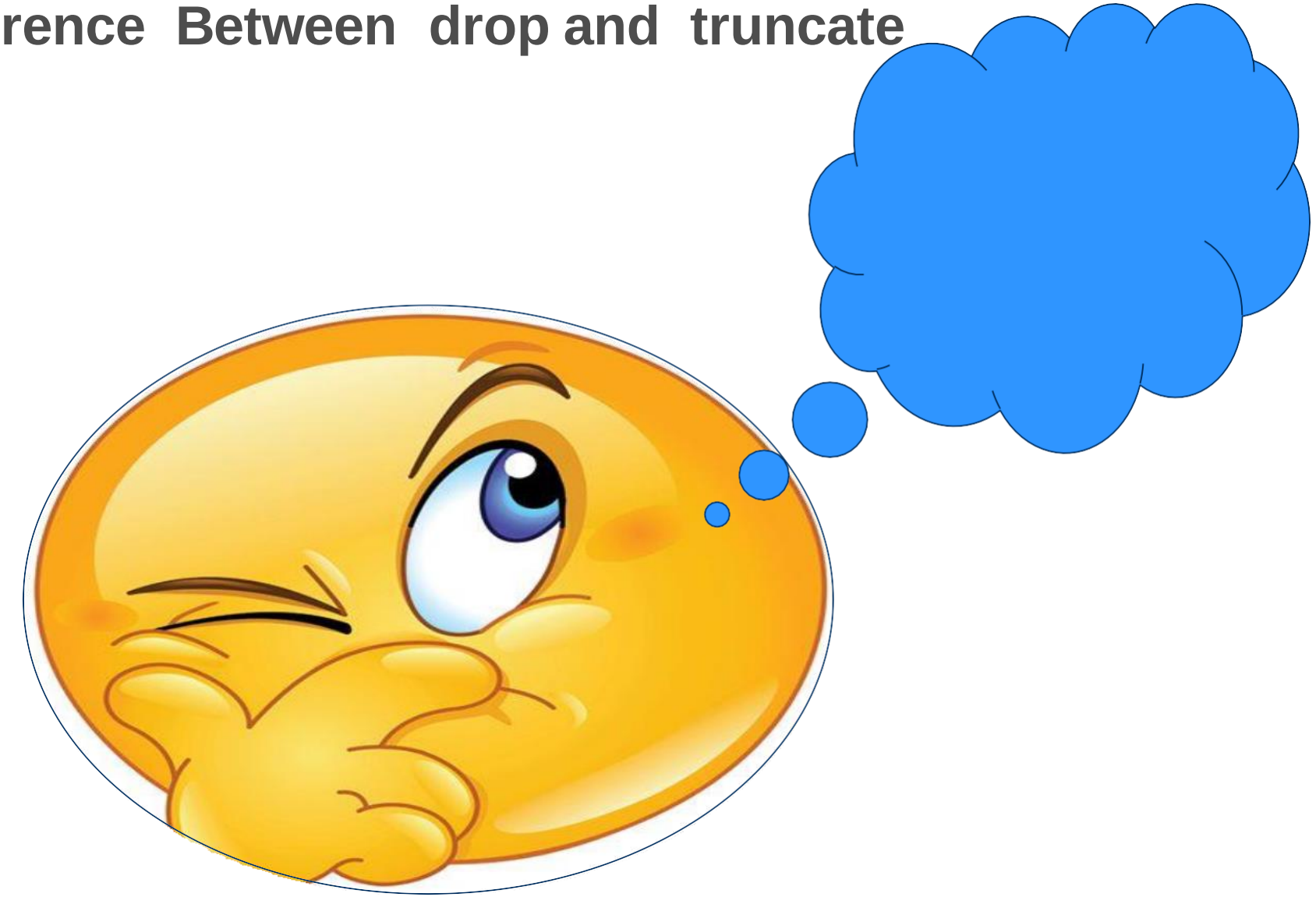
Syntax : TRUNCATE TABLE table_name;

Example:

TRUNCATE TABLE student



Difference Between drop and truncate



Constraints

What are Constraints?

- Constraints enforce rules at the table level.
- Constraints prevent the deletion of a table if there are dependencies.
- The following constraint types are valid:
  - NOT NULL
  - UNIQUE
  - PRIMARY KEY
  - FOREIGN KEY
  - DEFAULT

Defining Constraints

- **Syntax:**

```
CREATE TABLE [schema.]table  (column datatype [DEFAULT expr]
[column_constraint],
...
[table_constraint][,...]);

);
```

Example:

```
CREATE TABLE employees(
emp_id VARCHAR(8) NOT NULL ,  emp_name VARCHAR(50) NOT NULL,
CONSTRAINT PRIMARY KEY (emp_id)
);
```

```
57  15:20:24  CREATE TABLE employees( emp_id varchar(8) NOT NULL ,  ...  0 row(s) affected
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
emp_id	varchar(8)	NO	PRI	NULL	
emp_name	varchar(50)	NO		NULL	

The NOT NULL Constraint

- The NOT NULL Constraint Ensures that null values are not permitted for the column

- The NOT NULL constraint can be specified only at the column level, not at the table level.

Example:

```
CREATE TABLE employee (     id INT,
                            last_name VARCHAR(255) NOT NULL,
                            salary DOUBLE(5,2),
                            hire_date DATE NOT NULL
                        );
```

```
60  15:22:37  CREATE TABLE employee (  id INT,  last_name VARCHAR(255) NO...  0 row(s) affected
```

Result Grid | Filter Rows: | Export:

Field	Type	Null	Key	Default	Extra
id	int(11)	YES		NULL	
last_name	varchar(255)	NO		NULL	
salary	double(5,2)	YES		NULL	
hire_date	date	NO		NULL	

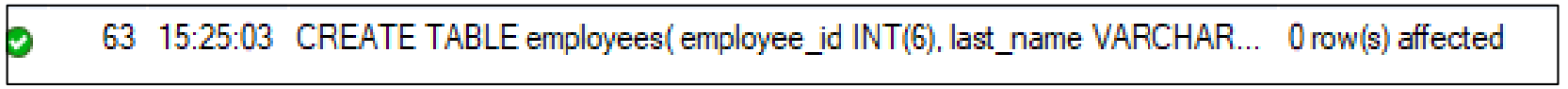
The UNIQUE Constraint

- A UNIQUE key integrity constraint requires that every value in a column or set of columns (key) be unique

- Defined at either the table level or the column level

Example:

```
CREATE TABLE employees(
                    employee_id INT(6),
                    last_name VARCHAR(25) NOT NULL,
                    email VARCHAR(25),
                    salary DOUBLE(8,2),
                    commission_pct DOUBLE(2,2),
                    hire_date DATE NOT NULL,
                    CONSTRAINT emp_email_uk UNIQUE(email)
                );
```



Contd..

Result Grid | Filter Rows: | Export:

Field	Type	Null	Key	Default
emplovee_id	int(6)	YES		NULL
last_name	varchar(25)	NO		NULL
email	varchar(25)	YES	UNI	NULL
salary	double(8,2)	YES		NULL
commission_pct	double(2,2)	YES		NULL
hire_date	date	NO		NULL

The PRIMARY KEY Constraint

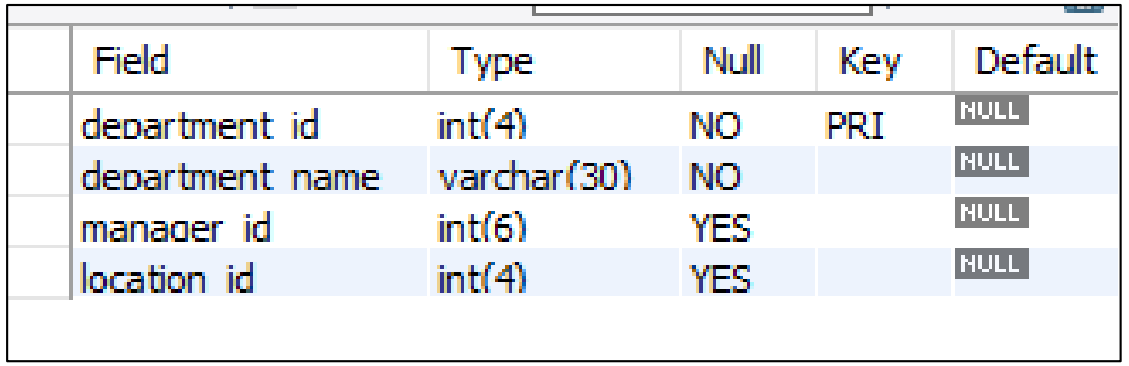
- A PRIMARY KEY constraint creates a primary key for the table
- Defined at either the table level or the column level

Example:

```
CREATE TABLE departments(
                    department_id INT(4),
                    department_name VARCHAR(30) NOT NULL,
                    manager_id INT(6),
                    location_id INT(4),
                    CONSTRAINT dept_id_pk PRIMARY KEY(department_id)
                );
```

✔	68	15:29:15	CREATE TABLE departments(department_id INT(4), department_name ...	0 row(s) affected

Contd..



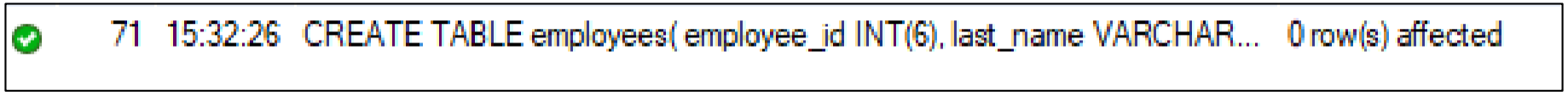
Field	Type	Null	Key	Default
department_id	int(4)	NO	PRI	NULL
department_name	varchar(30)	NO		NULL
manager_id	int(6)	YES		NULL
location_id	int(4)	YES		NULL

The FOREIGN KEY Constraint

- The FOREIGN KEY, or referential integrity constraint, designates a column or combination of columns as a foreign key and establishes a relationship between a primary key or a unique key in the same table or a different table.

Example:

```
CREATE TABLE employees(
                    employee_id INT(6),
                    last_name VARCHAR(25) NOT NULL,
                    email VARCHAR(25),
                    salary DOUBLE(8,2),
                    commission_pct DOUBLE(2,2),
                    hire_date DATE NOT NULL,
                    department_id INT(4),
CONSTRAINT emp_dept_fk FOREIGN KEY (department_id)
REFERENCES departments(department_id),
CONSTRAINT emp_email_uk UNIQUE(email)
                    );
```



Contd..

Field	Type	Null	Key	Default
emplovee_id	int(6)	YES		NULL
last_name	varchar(25)	NO		NULL
email	varchar(25)	YES	UNI	NULL
salary	double(8,2)	YES		NULL
commission_pct	double(2,2)	YES		NULL
hire_date	date	NO		NULL
department_id	int(4)	YES	MUL	NULL

FOREIGN KEY Constraint Keywords

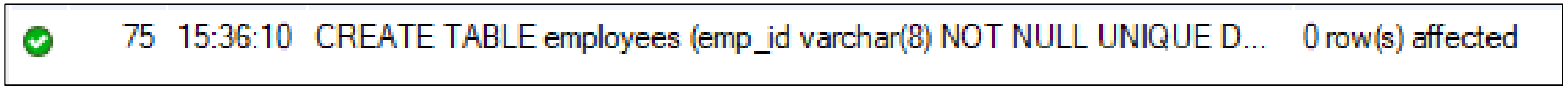
- **FOREIGN KEY:** Defines the column in the child table at the table constraint level
- **REFERENCES:** Identifies the table and column in the parent table
- **ON DELETE CASCADE:** Deletes the dependent rows
- in the child table when a row in the parent table is deleted.
- **ON DELETE SET NULL:** Converts dependent foreign key values to null

Default constraint

- DEFAULT is used to set a default value for a column .
- Can be implemented using DEFAULT default_value

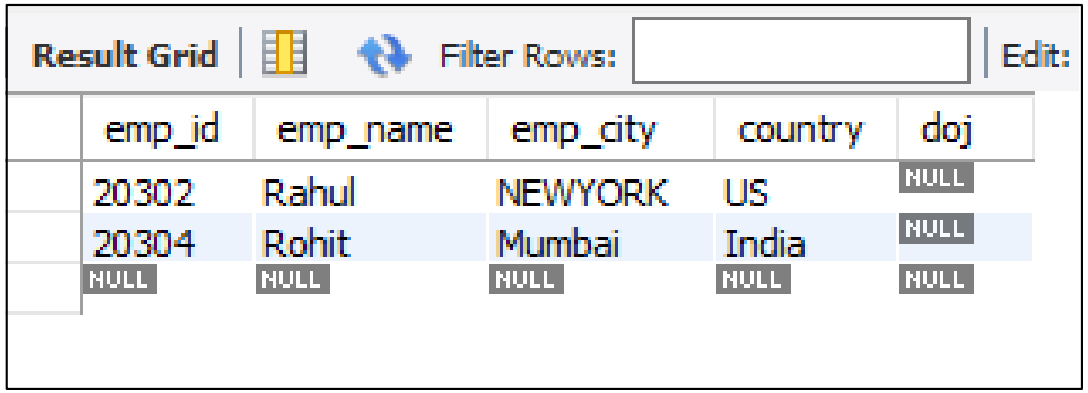
  where default_value is the default value set to the column.

```
CREATE TABLE employees(
                    emp_id varchar(8) NOT NULL UNIQUE DEFAULT ' ' ,
                    emp_name varchar(50) NOT NULL,
                    emp_city varchar(25) NOT NULL  ,
                    country varchar(25) NOT NULL DEFAULT 'India',
                    PRIMARY KEY (emp_id));
```



```
INSERT INTO  employees(emp_id,emp_name,emp_city,country) VALUES('20302','Rahul','NEWYORK','US');
INSERT INTO  employees(emp_id,emp_name,emp_city) VALUES('20304','Rohit','Mumbai');

SELECT * FROM employees;
```



Result Grid | Filter Rows: | Edit:

emp_id	emp_name	emp_city	country	doj
20302	Rahul	NEWYORK	US	NULL
20304	Rohit	Mumbai	India	NULL
NULL	NULL	NULL	NULL	NULL

List constraints

```
SELECT column_name,constraint_name,referenced_column_name,referenced_table_name
FROM information_schema.KEY_COLUMN_USAGE
where TABLE_NAME='employees'
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

column_name	constraint_name	referenced_column_name	referenced_table_name
emp_id	PRIMARY	NULL	NULL
emp_id	emp_id	NULL	NULL
email	emp_email_uk	NULL	NULL
department_id	emp_dept_fk	department_id	departments
employee_id	PRIMARY	NULL	NULL
email	emp_email_uk	NULL	NULL
department_id	emp_dept_fk	department_id	departments

Adding a Constraint Syntax

- Use the ALTER TABLE statement to:
  - Add or drop a constraint, but not modify its structure
  - Enable or disable constraints
  - Add a NOT NULL constraint by using the MODIFY Clause

- **Syntax**

```
ALTER TABLE table
ADD [CONSTRAINT constraint] type (column);
```


Adding a Constraint

- Add a FOREIGN KEY constraint to the Orders table indicating that a person must be a valid user in the Persons table.

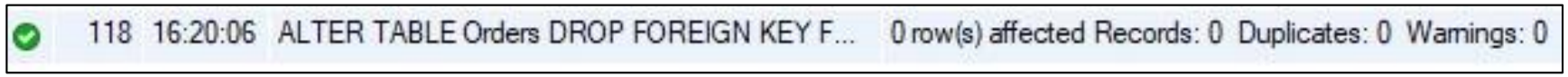
- **Example:**

```
ALTER TABLE Orders
ADD CONSTRAINT FK_PersonOrder FOREIGN KEY (PersonID)
REFERENCES Persons(PersonID);
```

Dropping a Constraint

- Remove the fk_PersonOder constraint from the Orders table.
- **Example:**

```
ALTER TABLE Orders
DROP FOREIGN KEY FK_PersonOrder;
```



- Remove the PRIMARY KEY constraint on the DEPARTMENTS
- **Example:**

```
ALTER TABLE departments
DROP PRIMARY KEY;
```


www.hexaware.com | © Hexaware Technologies. All rights reserved.


GAMIFICATION

Gamification

Objective:

To make the participants familiarize with tables, fields and keys through activity.

Assignment

1. DDL

2. Constraints

All material is available at the Sharepoint.

thank you

www.hexaware.com