Writing SQL Queries - I

CO226 : Database Systems

Lab - 05

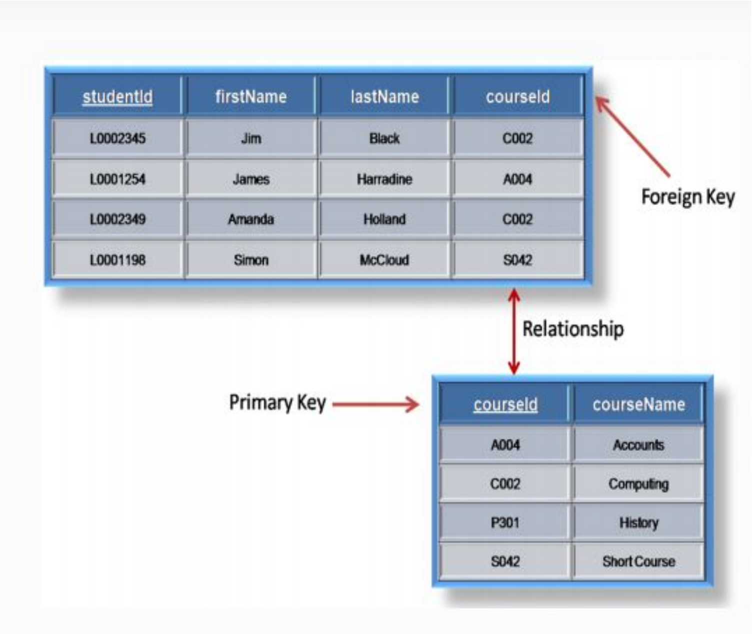
Anjalee Wanigarathne
anj.wanigarathne@eng.pdn.ac.lk

Outline

- Creating Relationships
- Specifying Primary Keys
- Specifying Foreign Keys
- Inserting Data
- Displaying Data
- Ordering Data
- Modifying Data
- Deleting Data
- Truncating a Table

Creating Relationships

- Relationships can be created between tables using primary keys and foreign keys.



studentId	firstName	lastName	courseId
L0002345	Jim	Black	C002
L0001254	James	Harradine	A004
L0002349	Amanda	Holland	C002
L0001198	Simon	McCloud	S042

courseId	courseName
A004	Accounts
C002	Computing
P301	History
S042	Short Course

Specifying Primary Keys

- **Syntax**

```
CREATE TABLE <TableName>

( <Field1><DataType>(<FieldSize>)
PRIMARY KEY,

.........................................................,

.........................................................);
```

Example

```
CREATE TABLE Student

(IndexNo VARCHAR(4) PRIMARY KEY,

StudentName VARCHAR(30),

Age INT,

CourseID VARCHAR(3));
```

Specifying Primary Keys (cont.)

- **Syntax (for composite primary key)**

```
CREATE TABLE <TableName>
( <Field1><DataType>(<FieldSize>),
.........................................................,
.........................................................
PRIMARY KEY (<KeyField1>,
<KeyField2>));
```

Example

```
CREATE TABLE Student_Grade
(IndexNo VARCHAR(4),
SubjectID VARCHAR(4),
Grade INT,
PRIMARY KEY (IndexNo, SubjectID));
```

Specifying Foreign Keys

- **Syntax**

```
CREATE TABLE <TableName>
        (<Field> <DataType> (<FieldSize>),
        .......................................................,
        .......................................................,
        FOREIGN KEY (<Field1>) REFERENCES <TableName>(<Field>));

                                    OR

CREATE TABLE <TableName>
         (<Field1> <DataType> (<FieldSize>),
        <Field2> <DataType> (<FieldSize>) REFERENCES <TableName>(<Field>),
        .......................................................,
        .......................................................);
```

Specifying Foreign Keys (cont.)

- **Example**

```
CREATE TABLE Student
        ( IndexNo VARCHAR(4) PRIMARY KEY,
        StudentName VARCHAR(30),
        Age INT,
        CourseID VARCHAR(3),
        FOREIGN KEY ( CourseID ) REFERENCES Course( CourseID ));
```

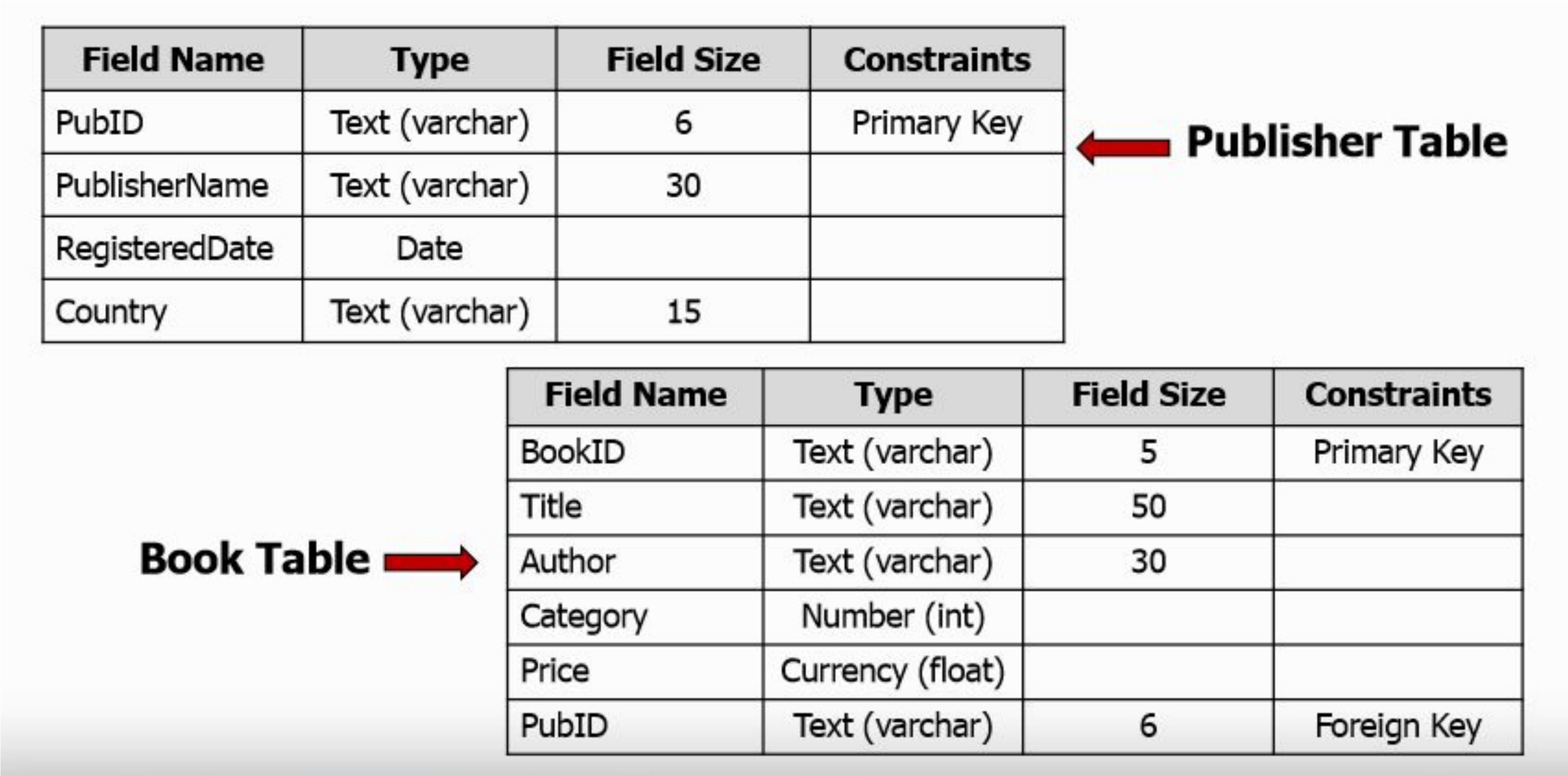
Submission

- You need to create a pdf file named “**E17_XXX_InclassLab5.pdf**” . Here XXX is your registration number. Your **.pdf** file will includes,
    - Answers for **Activity 1** to **Activity 5** with respective SQL queries and the screenshots of each output.

- You need to submit it right after the lab. This will use to mark your attendance. If you fail to submit, I assume you as an absent person.

Activity 1

1. Recreate the following tables with relationships using MySQL syntaxes.



Publisher Table

Field Name	Type	Field Size	Constraints
PubID	Text (varchar)	6	Primary Key
PublisherName	Text (varchar)	30	
RegisteredDate	Date		
Country	Text (varchar)	15	

Book Table

Field Name	Type	Field Size	Constraints
BookID	Text (varchar)	5	Primary Key
Title	Text (varchar)	50	
Author	Text (varchar)	30	
Category	Number (int)		
Price	Currency (float)		
PubID	Text (varchar)	6	Foreign Key

Inserting Data

Inserting Data to a Table

- **Syntax:**

```
INSERT INTO <Table_Name> (<Field1>, <Field2>, <Field3>)
VALUES (<Value1>, <Value2>, <Value3>);
```

OR

```
INSERT INTO <Table_Name>
VALUES (<Value1>, <Value2>, <Value3>);
```

OR

Inserting multiple rows:

```
INSERT INTO <Table_Name>
VALUES (<Value1>, <Value2>, <Value3>),
(<Value1a>, <Value2a>, <Value3a>);
```

Inserting Data to a Table (cont.)

- **Example:**

```
INSERT INTO Student ( IndexNo , StudentName , Age, CourseID )
VALUES (‘S1’, ’P. Perera ’, 20, ‘CO226’);
```

OR

```
INSERT INTO Student
VALUES (‘S1’, ’P. Perera ’, 20, ‘CO226’);
```

Inserting Multiple Data to a Table

- **Example:**

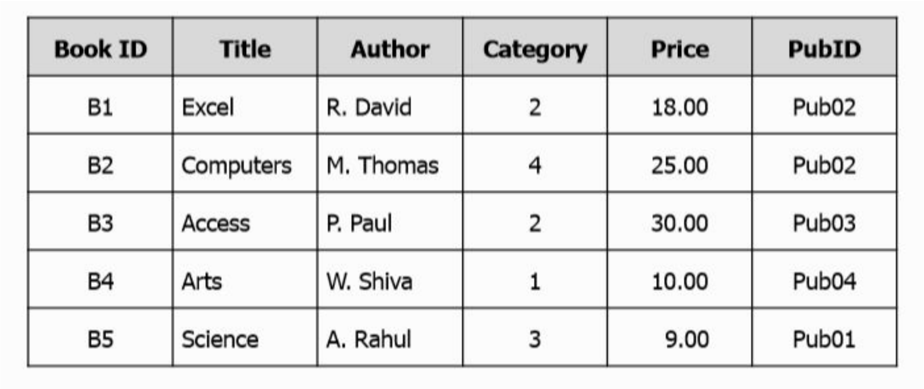
```
INSERT INTO Student ( IndexNo , StudentName , Age, CourseID )
VALUES (‘S1’, ‘Perera’, 20, ‘CO226’),
(‘S2’, ‘Alex’, 22, ‘CO326’),
(‘S3’, ‘Tim’, 21, ‘CO222’),
(‘S4’, ‘John’, 20, ‘CO226’);
```

OR

```
INSERT INTO Student
VALUES (‘S1’, ‘Perera’, 20, ‘CO226’),
(‘S2’, ‘Alex’, 22, ‘CO326’),
(‘S3’, ‘Tim’, 21, ‘CO222’),
(‘S4’, ‘John’, 20, ‘CO226’);
```

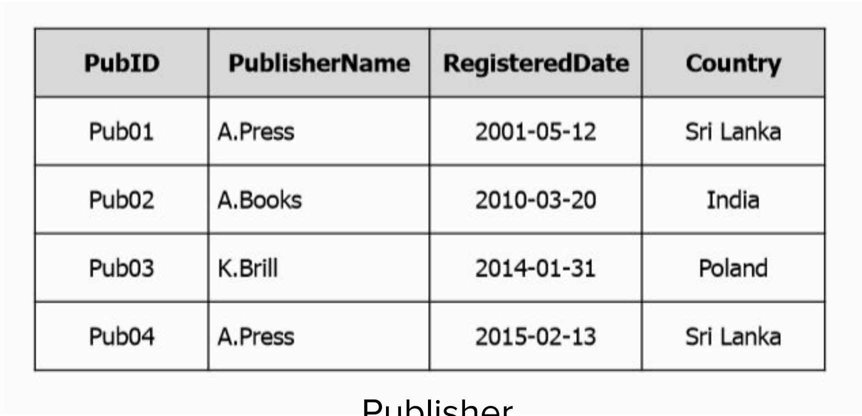
Activity 2

- Insert following records to Publisher table and Book table.



Book ID	Title	Author	Category	Price	PubID
B1	Excel	R. David	2	18.00	Pub02
B2	Computers	M. Thomas	4	25.00	Pub02
B3	Access	P. Paul	2	30.00	Pub03
B4	Arts	W. Shiva	1	10.00	Pub04
B5	Science	A. Rahul	3	9.00	Pub01

Book



PubID	PublisherName	RegisteredDate	Country
Pub01	A.Press	2001-05-12	Sri Lanka
Pub02	A.Books	2010-03-20	India
Pub03	K.Brill	2014-01-31	Poland
Pub04	A.Press	2015-02-13	Sri Lanka

Publisher

Displaying Data

Displaying Data

Syntax:

```
SELECT <Field_Name(s)>
FROM <Table_Name(s)>
WHERE <.......>
GROUP BY <.....>
HAVING <......>
ORDER BY <.....>;
```

Displaying Data (cont.)

- Displaying all the data (all rows and columns) in a table:

- **Syntax:**

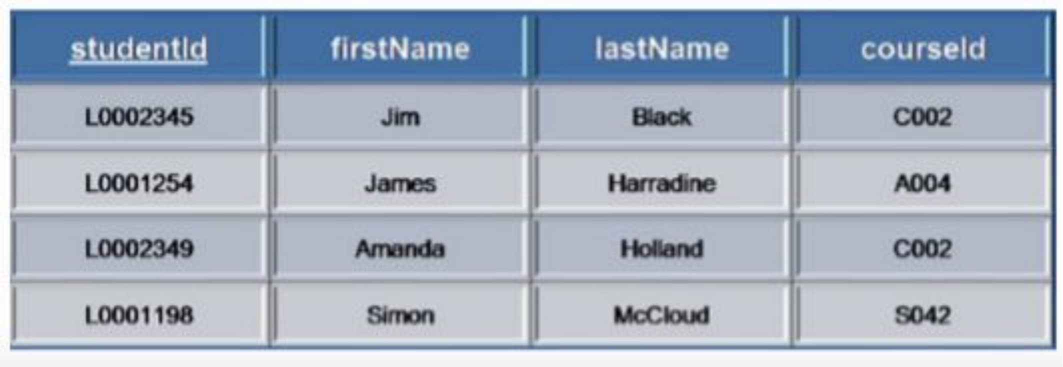
  SELECT *

  FROM <Table_Name>;

- **Example:**

  SELECT *

  FROM Student;



studentId	firstName	lastName	courseId
L0002345	Jim	Black	C002
L0001254	James	Harradine	A004
L0002349	Amanda	Holland	C002
L0001198	Simon	McCloud	S042

Displaying Data (cont.)

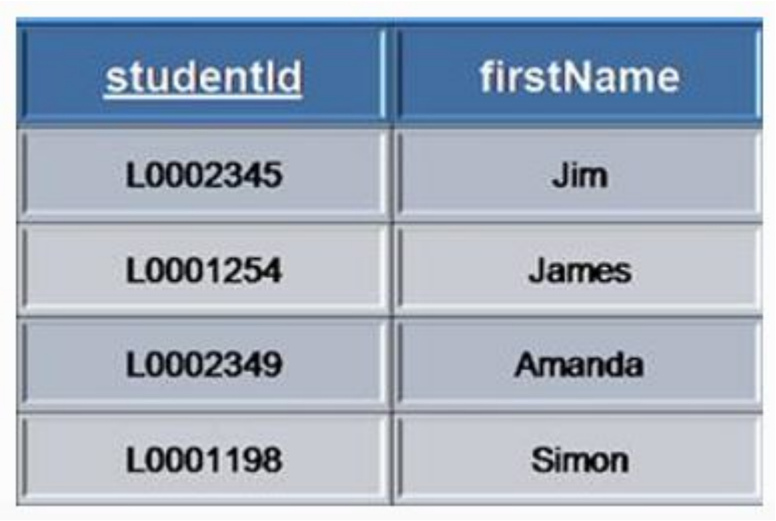
- Displaying specific columns in a table:

- **Syntax:**

  SELECT <Column1>, <Column2>
  FROM <Table_Name>;

- **Example:**

  SELECT studentId , firstName
  FROM Student;



studentId	firstName
L0002345	Jim
L0001254	James
L0002349	Amanda
L0001198	Simon

Displaying Data (cont.)

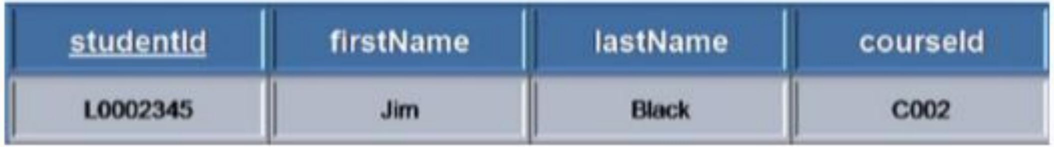
- Displaying specific rows in a table:

- **Syntax:**

```
SELECT * FROM <Table_Name>
WHERE <Condition>;
```

- **Example:**

```
SELECT *
FROM Student
WHERE studentId = ‘L0002345’;
```



studentId	firstName	lastName	courseId
L0002345	Jim	Black	C002

Displaying Data (cont.)

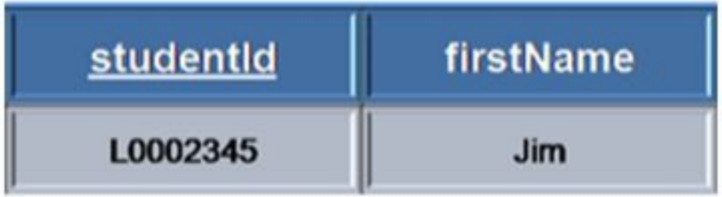
- Displaying specific columns and rows in a table:

- **Syntax:**

```
SELECT <Column1>, <Column2>
FROM <Table_Name>
WHERE <Condition>;
```

- **Example:**

```
SELECT studentId , firstName
FROM Student
WHERE studentId = ‘L0002345’;
```

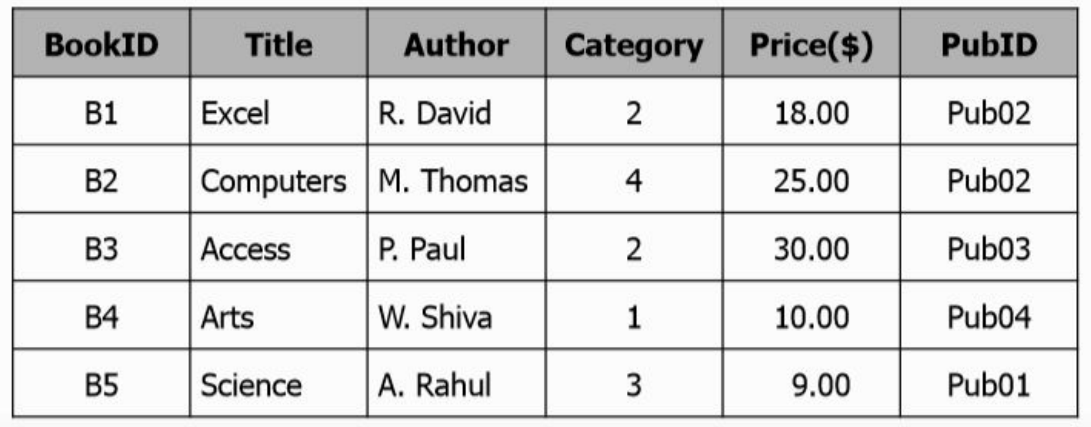


studentId	firstName
L0002345	Jim

Activity 3

Write SQL queries/statements considering the following table. **Table Name: Book**

1. Display all the details in the Book table.
2. Display all the titles of the books.
3. Display all the titles of the books with prices.
4. Display the Book ID, the Title and the Author.
5. Display the details of the book with the Book ID B2.
6. Display the titles of the books with the prices less than $20.00.
7. Display the details of the books in category 2.



BookID	Title	Author	Category	Price($)	PubID
B1	Excel	R. David	2	18.00	Pub02
B2	Computers	M. Thomas	4	25.00	Pub02
B3	Access	P. Paul	2	30.00	Pub03
B4	Arts	W. Shiva	1	10.00	Pub04
B5	Science	A. Rahul	3	9.00	Pub01

Ordering Data

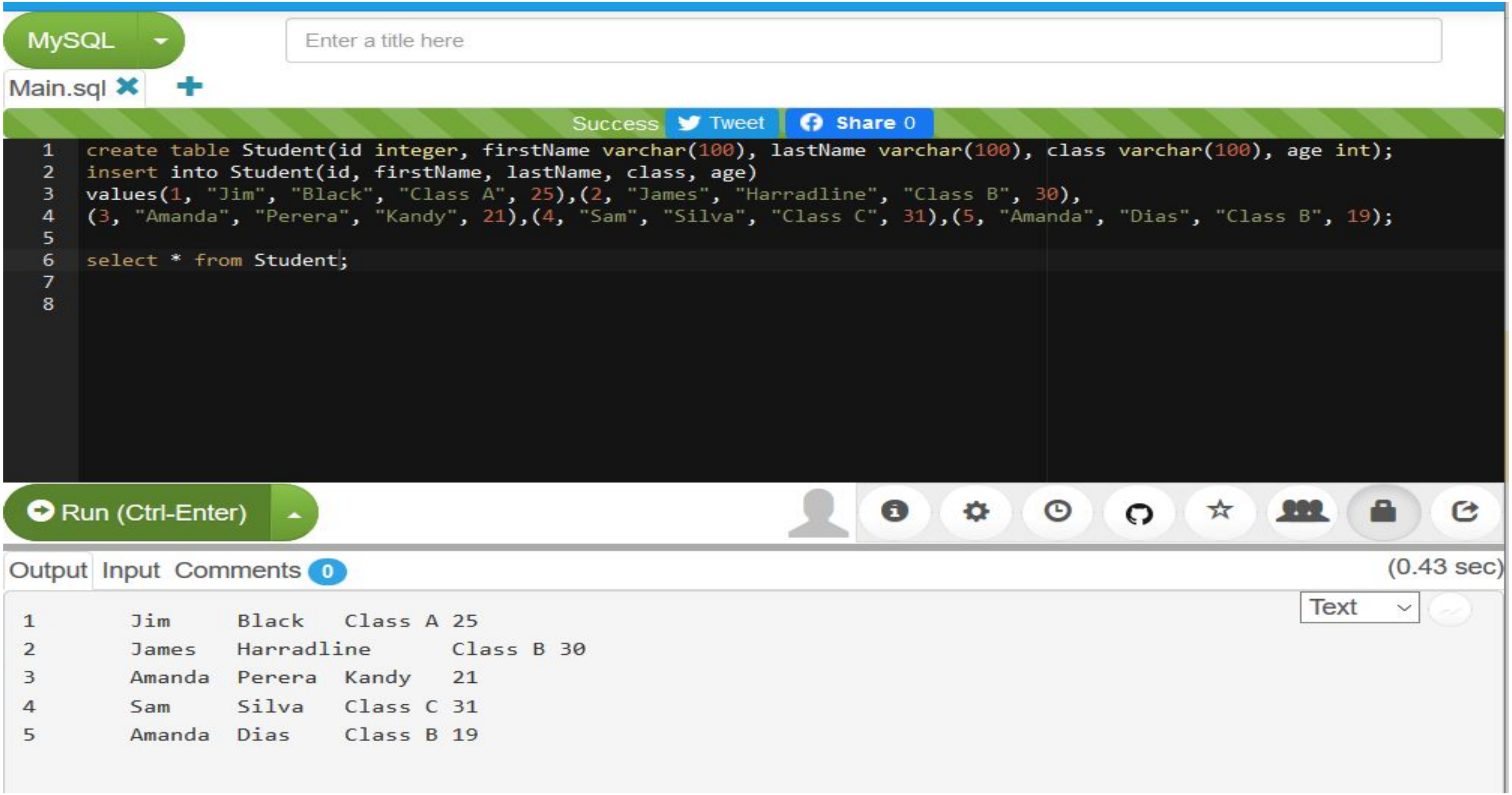
Ordering Data

- Ordering rows in a sequence (Ascending Order):
- **Syntax:**

```
SELECT <FieldName>

FROM <TableName>

ORDER BY <FieldName(s)>;
```

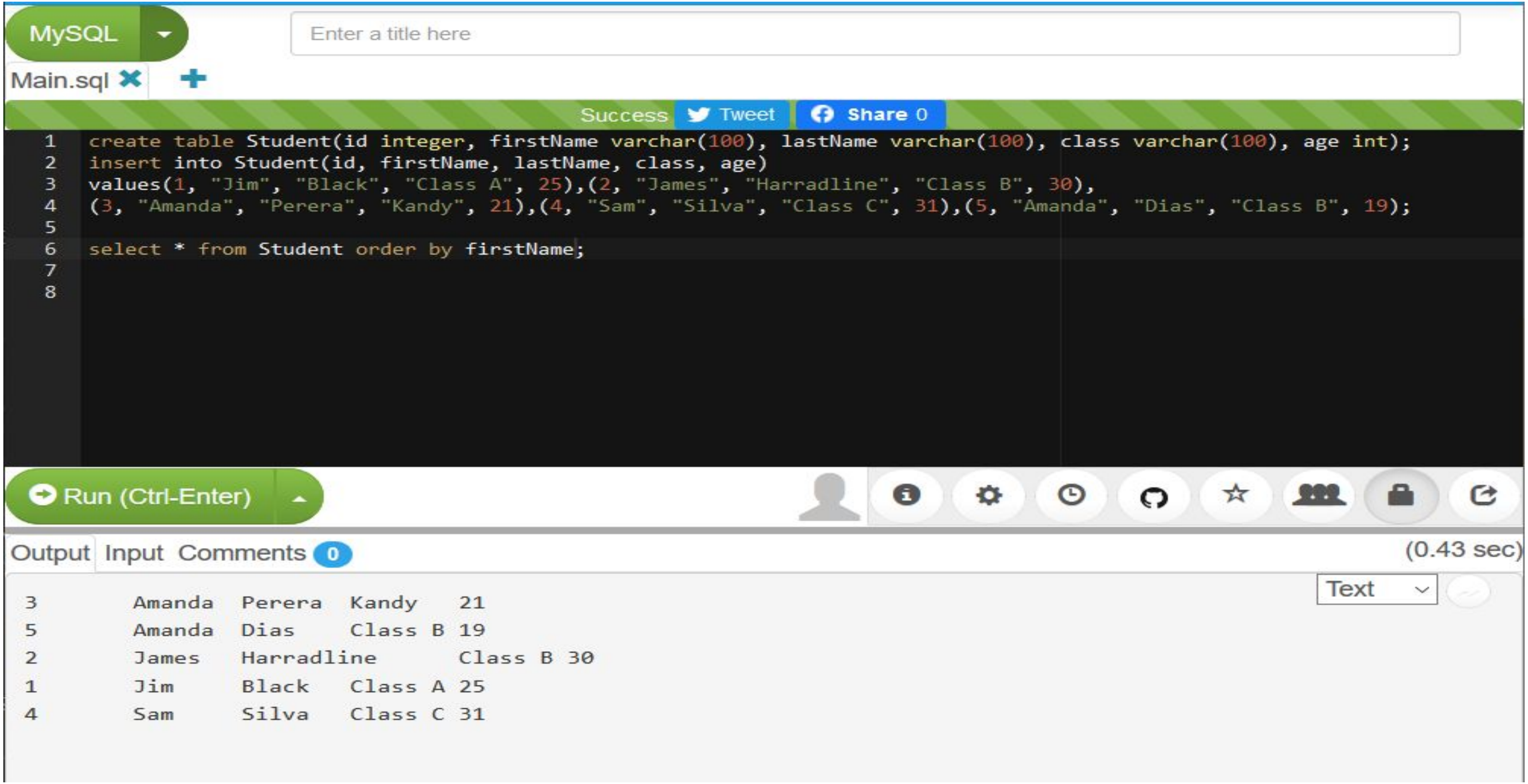
Example:

```
SELECT * FROM Student ORDER BY Name;

SELECT * FROM Student ORDER BY Name, Age;
```



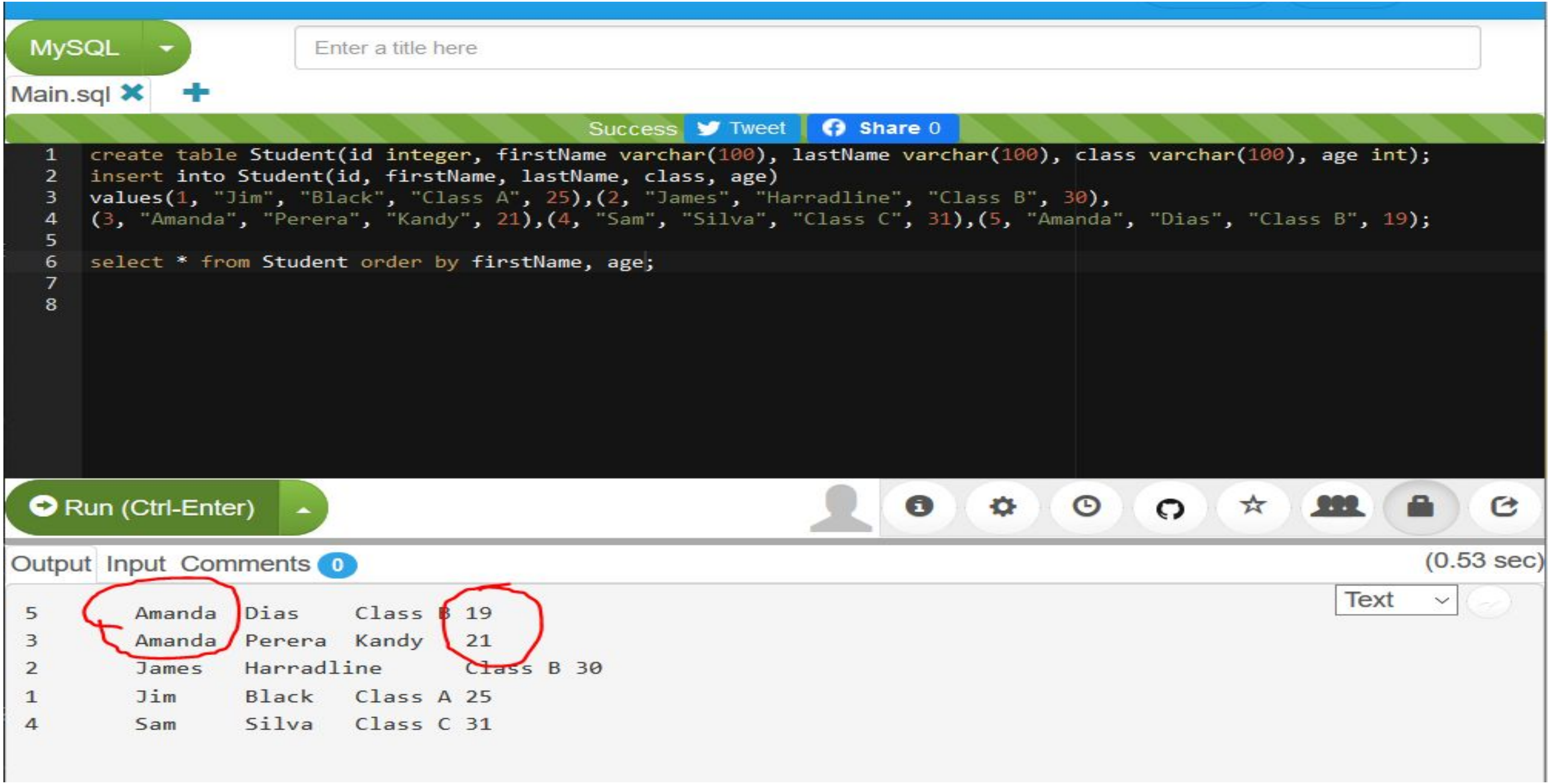
```sql
create table Student(id integer, firstName varchar(100), lastName varchar(100), class varchar(100), age int);
insert into Student(id, firstName, lastName, class, age)
values(1, "Jim", "Black", "Class A", 25),(2, "James", "Harradline", "Class B", 30),
(3, "Amanda", "Perera", "Kandy", 21),(4, "Sam", "Silva", "Class C", 31),(5, "Amanda", "Dias", "Class B", 19);

select * from Student;
```

```
1       Jim     Black   Class A 25
2       James   Harradline      Class B 30
3       Amanda  Perera  Kandy   21
4       Sam     Silva   Class C 31
5       Amanda  Dias    Class B 19
```



```sql
create table Student(id integer, firstName varchar(100), lastName varchar(100), class varchar(100), age int);
insert into Student(id, firstName, lastName, class, age)
values(1, "Jim", "Black", "Class A", 25),(2, "James", "Harradline", "Class B", 30),
(3, "Amanda", "Perera", "Kandy", 21),(4, "Sam", "Silva", "Class C", 31),(5, "Amanda", "Dias", "Class B", 19);

select * from Student order by firstName;
```

```
3       Amanda  Perera  Kandy   21
5       Amanda  Dias    Class B 19
2       James   Harradline      Class B 30
1       Jim     Black   Class A 25
4       Sam     Silva   Class C 31
```



```sql
create table Student(id integer, firstName varchar(100), lastName varchar(100), class varchar(100), age int);
insert into Student(id, firstName, lastName, class, age)
values(1, "Jim", "Black", "Class A", 25),(2, "James", "Harradline", "Class B", 30),
(3, "Amanda", "Perera", "Kandy", 21),(4, "Sam", "Silva", "Class C", 31),(5, "Amanda", "Dias", "Class B", 19);

select * from Student order by firstName, age;
```

```
5       Amanda  Dias      Class B 19
3       Amanda  Perera    Kandy   21
2       James   Harradline        Class B 30
1       Jim     Black     Class A 25
4       Sam     Silva     Class C 31
```

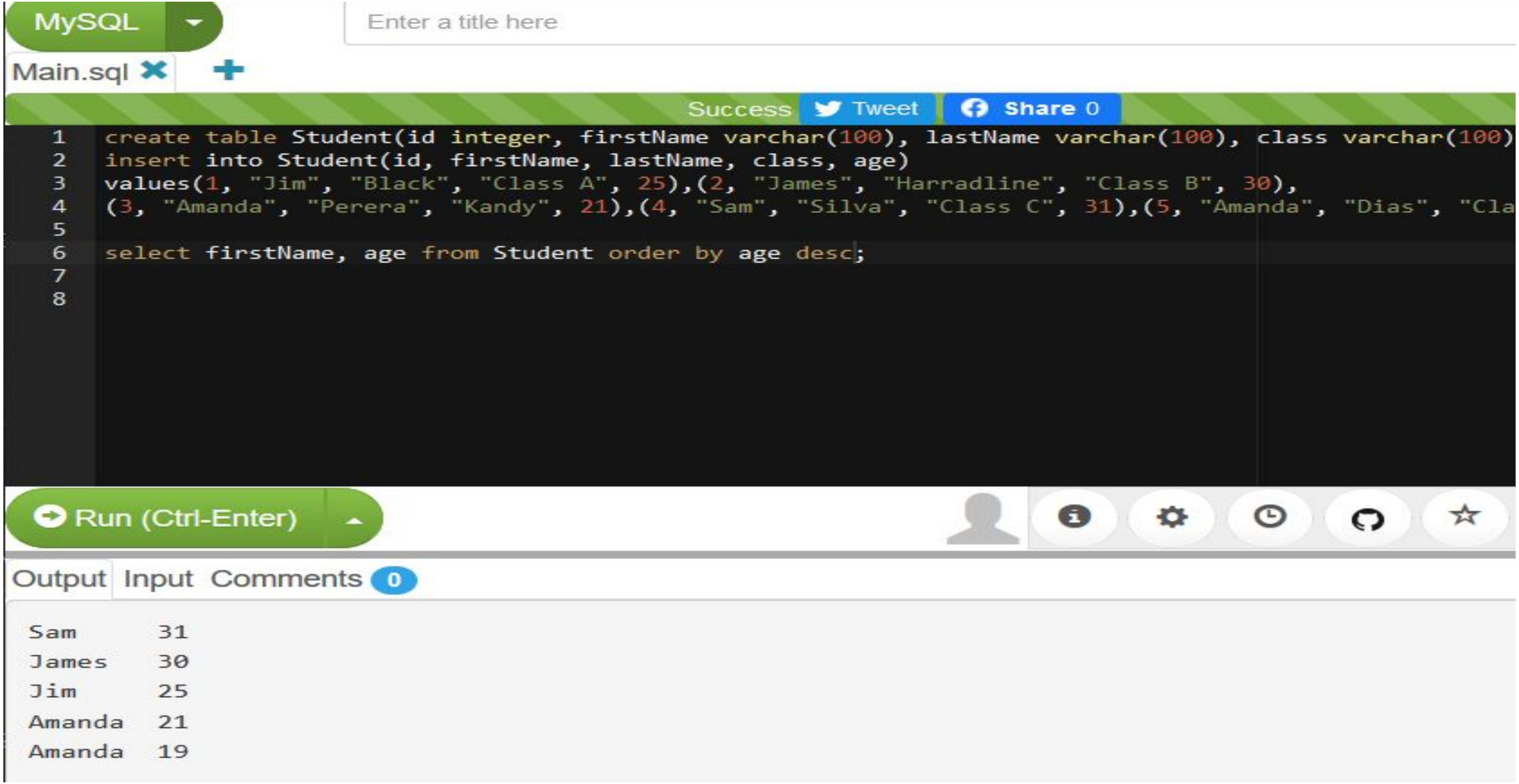
Ordering Data cont.

- Ordering rows in a sequence (Descending Order):
- **Syntax:**

```
SELECT <FieldName>

FROM <TableName>

ORDER BY <FieldName(s)> DESC;
```

Example:

```
SELECT firstName FROM Student ORDER BY Name DESC;
```

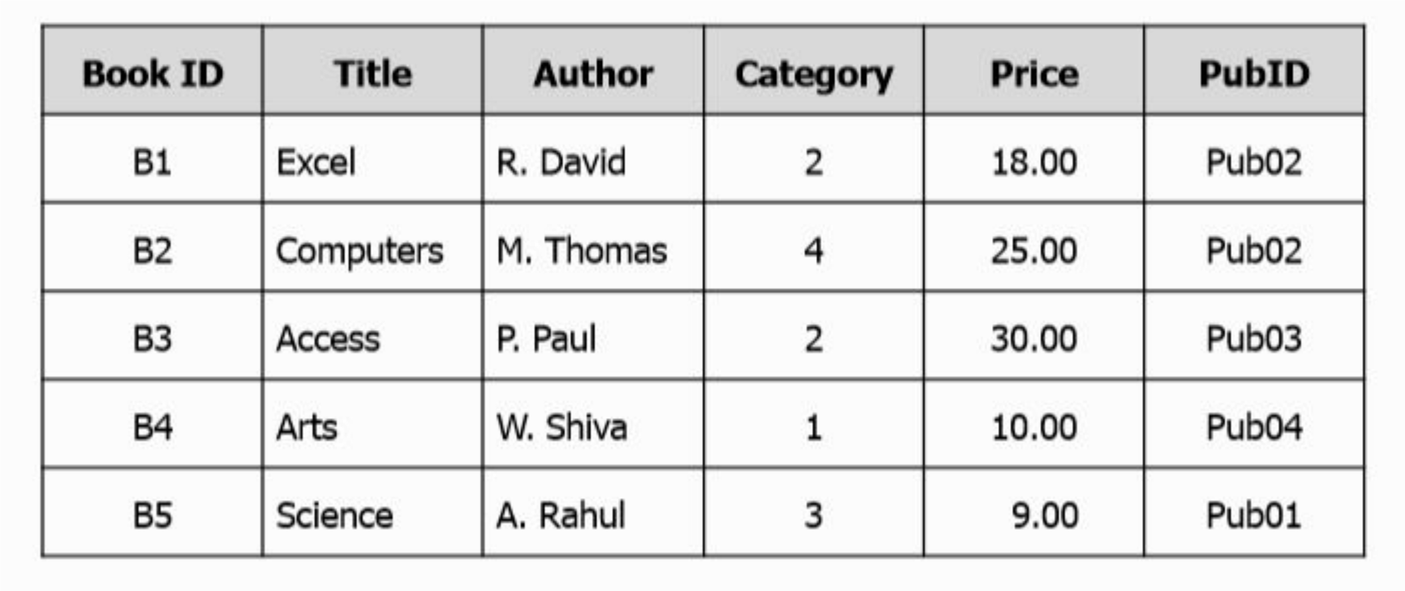


```sql
create table Student(id integer, firstName varchar(100), lastName varchar(100), class varchar(100)
insert into Student(id, firstName, lastName, class, age)
values(1, "Jim", "Black", "Class A", 25),(2, "James", "Harradline", "Class B", 30),
(3, "Amanda", "Perera", "Kandy", 21),(4, "Sam", "Silva", "Class C", 31),(5, "Amanda", "Dias", "Cla

select firstName, age from Student order by age desc;
```

```
Sam      31
James    30
Jim      25
Amanda   21
Amanda   19
```

Activity 4

1. Display all the Titles in the Book table in ascending order.
2. Display all the details in the Book table in descending order by Price.
3. Display all the details in the Book table in ascending order by Price and Title.



Book ID	Title	Author	Category	Price	PubID
B1	Excel	R. David	2	18.00	Pub02
B2	Computers	M. Thomas	4	25.00	Pub02
B3	Access	P. Paul	2	30.00	Pub03
B4	Arts	W. Shiva	1	10.00	Pub04
B5	Science	A. Rahul	3	9.00	Pub01

Modifying Data

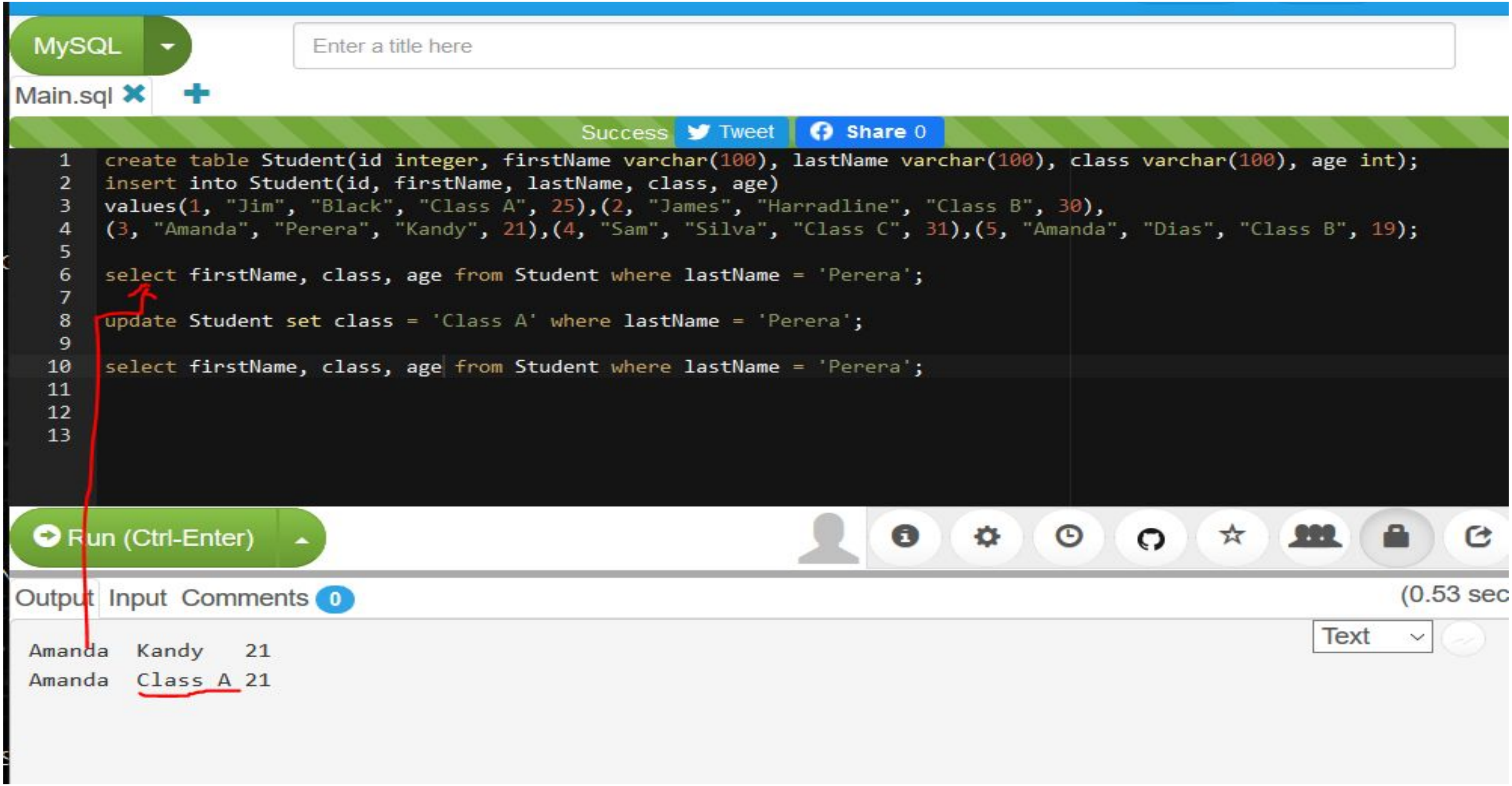
Modifying Data

- Changing/modifying existing data in a table:
- **Syntax:**

```
UPDATE <TableName>

SET <FieldName> = <NewValue>

WHERE <Condition>;
```

Example:

```
UPDATE Student SET Class = ‘Class A’ WHERE lastName = ‘Perera’;
```



```sql
create table Student(id integer, firstName varchar(100), lastName varchar(100), class varchar(100), age int);
insert into Student(id, firstName, lastName, class, age)
values(1, "Jim", "Black", "Class A", 25),(2, "James", "Harradline", "Class B", 30),
(3, "Amanda", "Perera", "Kandy", 21),(4, "Sam", "Silva", "Class C", 31),(5, "Amanda", "Dias", "Class B", 19);

select firstName, class, age from Student where lastName = 'Perera';

update Student set class = 'Class A' where lastName = 'Perera';

select firstName, class, age from Student where lastName = 'Perera';
```

```
Amanda   Kandy   21
Amanda   Class A 21
```

Modifying Data cont.

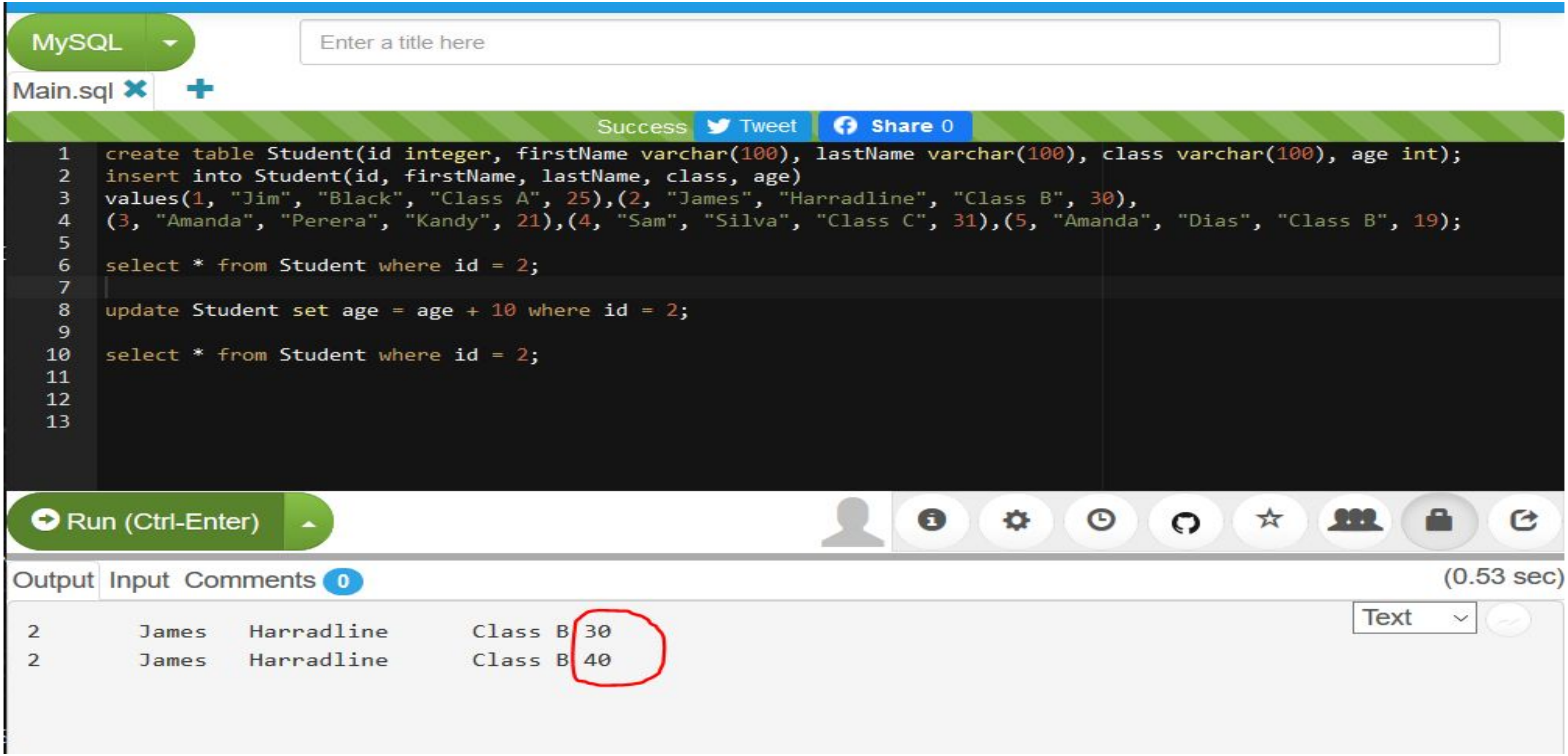
- Changing/modifying multiple columns in a table:
- **Syntax:**

```
UPDATE <TableName>

SET <FieldName> = <NewValue> , <FieldName> = <NewValue>

WHERE <Condition>;
```

Example:

UPDATE Student SET Class = ‘Class A’, lastName = ‘Perera’ WHERE Marks >75;

Example → Change the Age of the student with the StudentID ‘2’ by adding 10 to the current age.



```
create table Student(id integer, firstName varchar(100), lastName varchar(100), class varchar(100), age int);
insert into Student(id, firstName, lastName, class, age)
values(1, "Jim", "Black", "Class A", 25),(2, "James", "Harradline", "Class B", 30),
(3, "Amanda", "Perera", "Kandy", 21),(4, "Sam", "Silva", "Class C", 31),(5, "Amanda", "Dias", "Class B", 19);

select * from Student where id = 2;


update Student set age = age + 10 where id = 2;

select * from Student where id = 2;
```

```
2       James   Harradline      Class B 30
2       James   Harradline      Class B 40
```

Deleting Data

Deleting Data

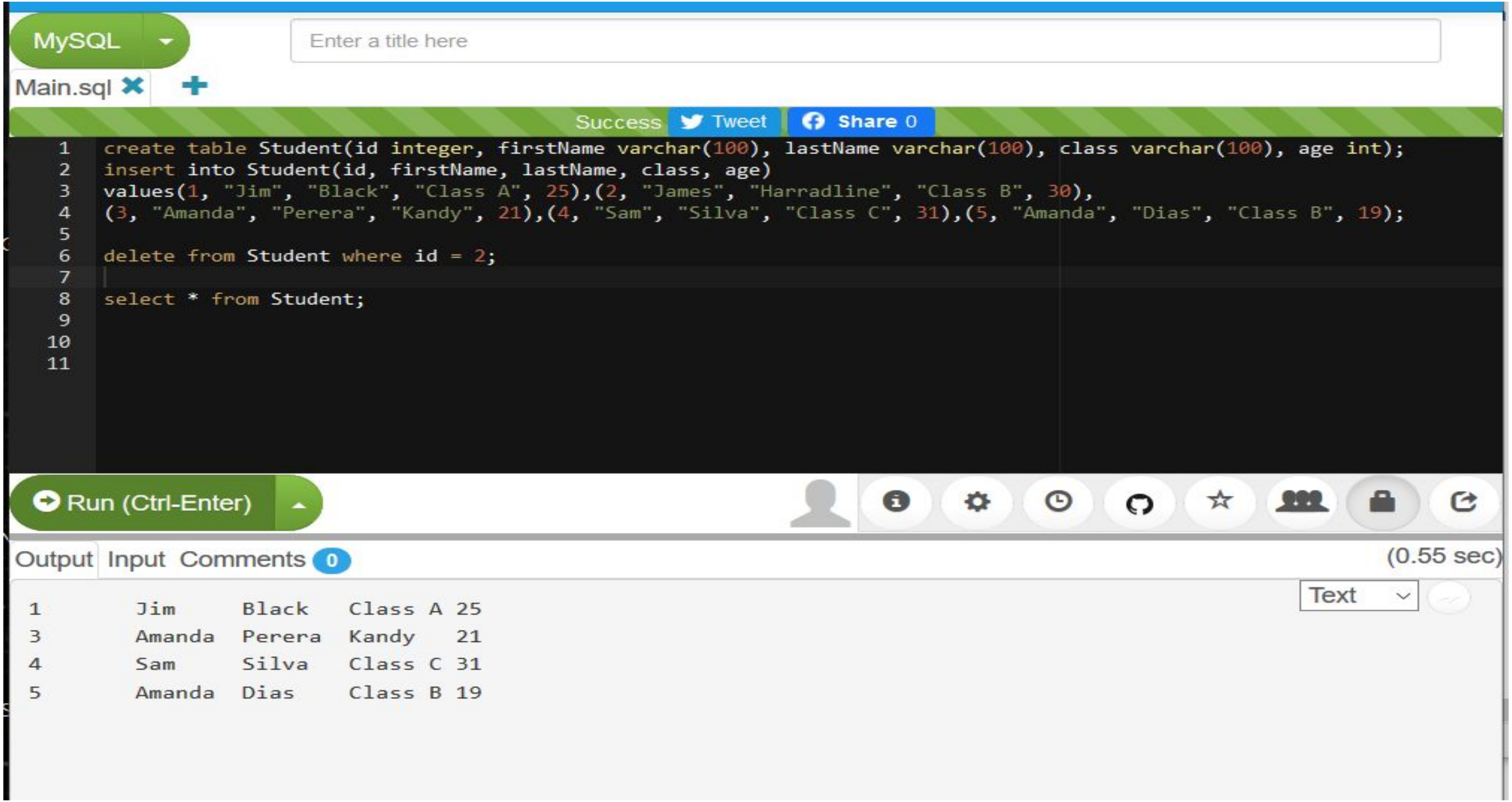
- To delete a particular row(s) data in a table:
- **Syntax:**

```
DELETE FROM <Table_Name>

WHERE <Field_Name> = <Value>;
```

Example:

```
DELETE FROM Student WHERE studentId = 1 ;
```

- Note: If you do not use the where clause, all the data in the table will get deleted.



```
create table Student(id integer, firstName varchar(100), lastName varchar(100), class varchar(100), age int);
insert into Student(id, firstName, lastName, class, age)
values(1, "Jim", "Black", "Class A", 25),(2, "James", "Harradline", "Class B", 30),
(3, "Amanda", "Perera", "Kandy", 21),(4, "Sam", "Silva", "Class C", 31),(5, "Amanda", "Dias", "Class B", 19);

delete from Student where id = 2;

select * from Student;
```

```
1       Jim     Black   Class A 25
3       Amanda  Perera  Kandy   21
4       Sam     Silva   Class C 31
5       Amanda  Dias    Class B 19
```

Truncating a Table

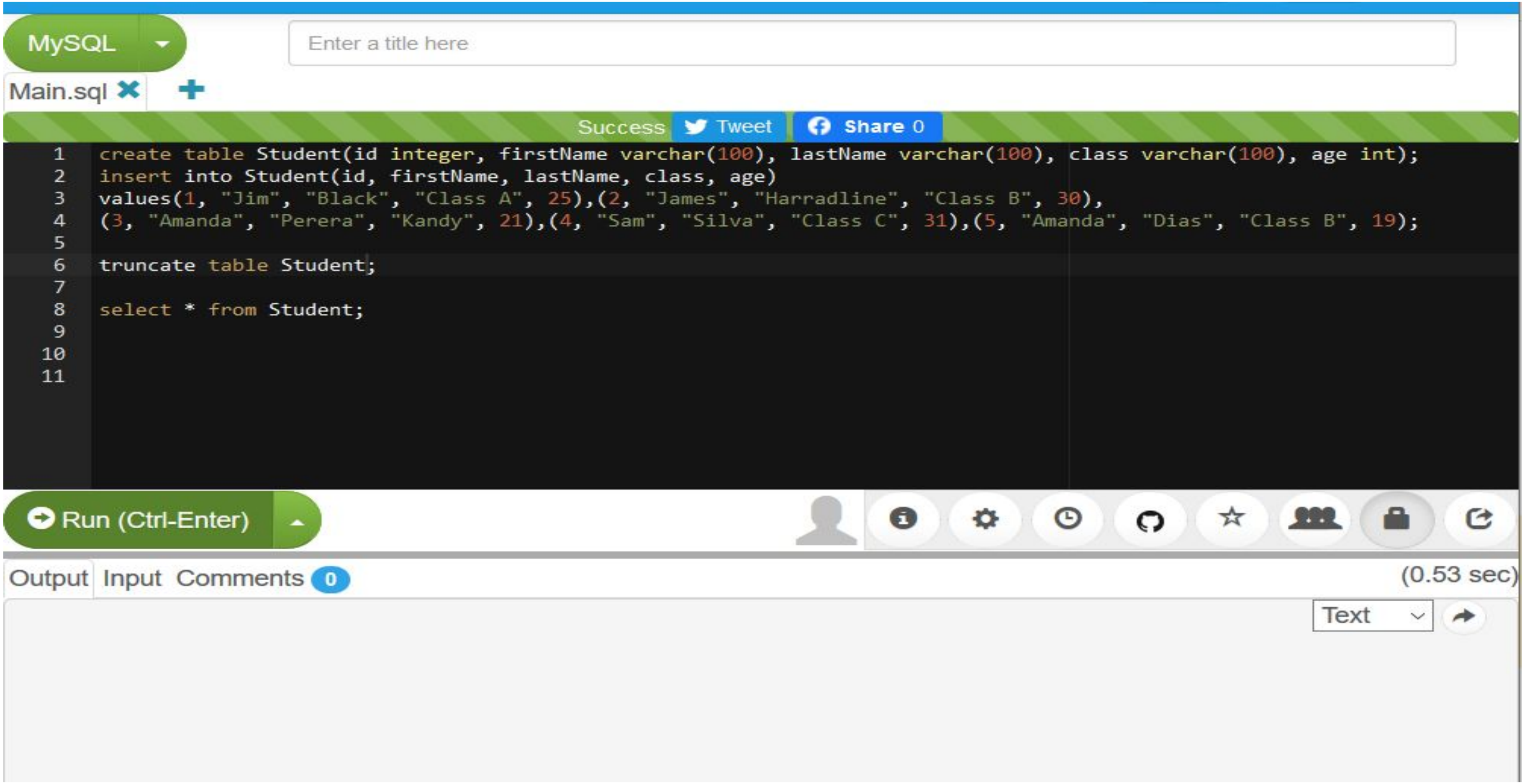
Truncating Table

- To delete all the data inside a table without removing the table:
- **Syntax:**

```
TRUNCATE TABLE <Table_Name>;
```

Example:

```
TRUNCATE TABLE Student;
```

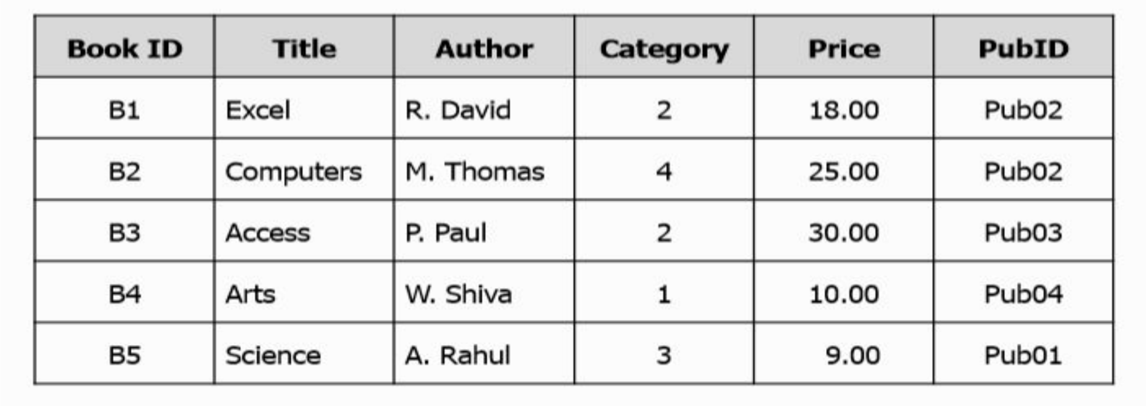


```sql
create table Student(id integer, firstName varchar(100), lastName varchar(100), class varchar(100), age int);
insert into Student(id, firstName, lastName, class, age)
values(1, "Jim", "Black", "Class A", 25),(2, "James", "Harradline", "Class B", 30),
(3, "Amanda", "Perera", "Kandy", 21),(4, "Sam", "Silva", "Class C", 31),(5, "Amanda", "Dias", "Class B", 19);

truncate table Student;

select * from Student;
```

Activity 5

Write SQL queries/statements considering the ‘Book’ table.

1. Change the category as 3 of the book with the BookID ‘B3’.
2. Change the price of the book with the BookID ‘B3’ by adding 10.00 to the current price.
3. Delete the details of the book with the BookID ‘B3’.
4. Delete all the books in category 2.
5. Remove all the data in the Book table.



Book ID	Title	Author	Category	Price	PubID
B1	Excel	R. David	2	18.00	Pub02
B2	Computers	M. Thomas	4	25.00	Pub02
B3	Access	P. Paul	2	30.00	Pub03
B4	Arts	W. Shiva	1	10.00	Pub04
B5	Science	A. Rahul	3	9.00	Pub01

Summary

- Creating Relationships
- Specifying Primary Keys
- Specifying Foreign Keys
- Inserting Data
- Displaying Data
- Ordering Data
- Modifying Data
- Deleting Data
- Truncating a Table