

IS664 Database Programming

Fall 2022

Fundamentals



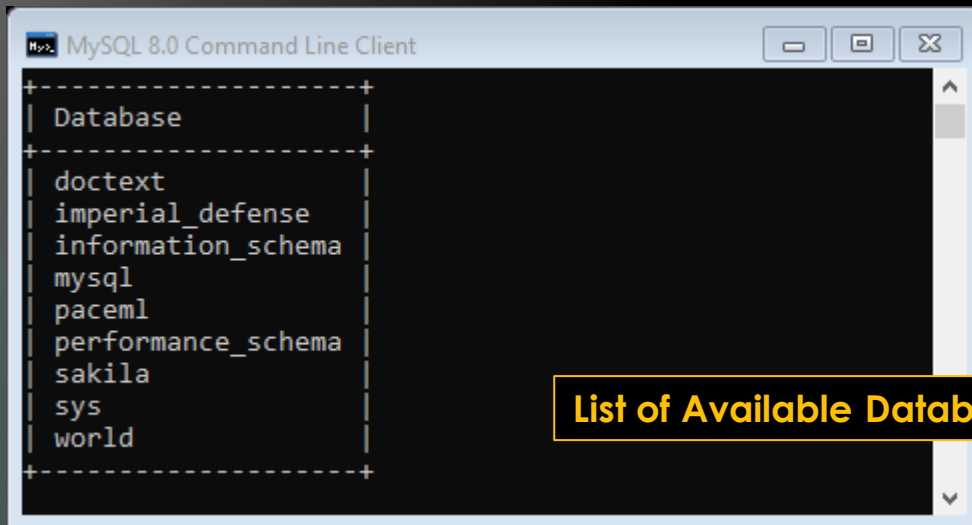
Database Programming

LECTURE 2: SIMPLE DATABASE CREATION

Seeing Databases

2

- ▶ We can see what databases (namespaces) are available to us on our MySQL Server using the `show databases` command.



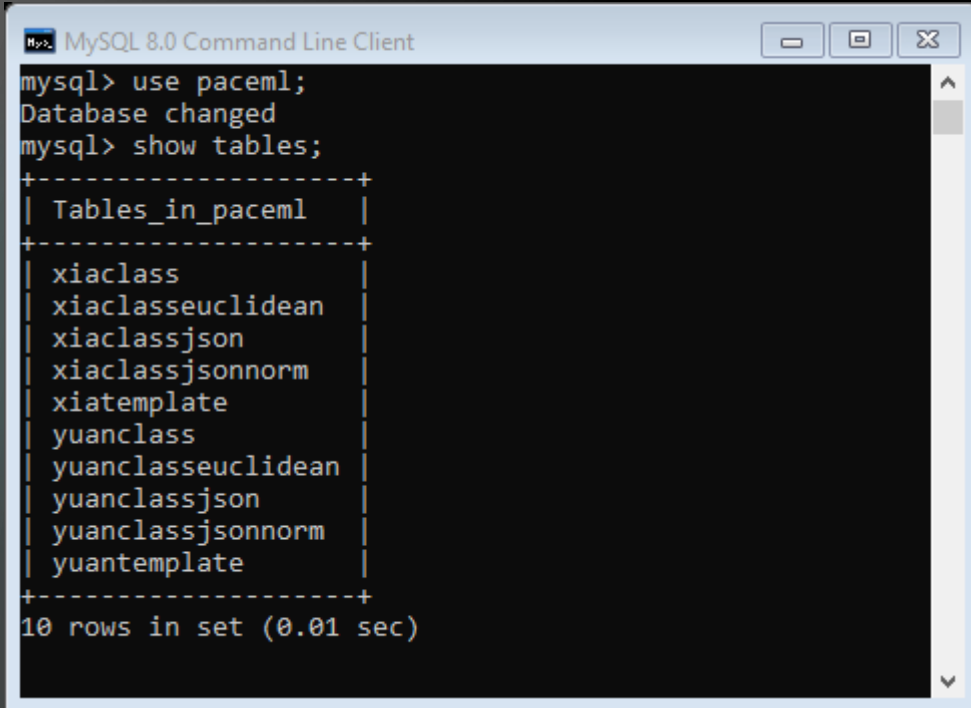
```
MySQL 8.0 Command Line Client
+-----+
| Database |
+-----+
| doctest  |
| imperial_defense |
| information_schema |
| mysql    |
| paceml   |
| performance_schema |
| sakila   |
| sys      |
| world    |
+-----+
```

List of Available Databases that you have access to

Seeing Tables in a Database

3

- ▶ We can see what relations (tables) are available to us in a specific database by selecting the databases using the **USE** command and then using the **show tables** command to see a list of the tables.

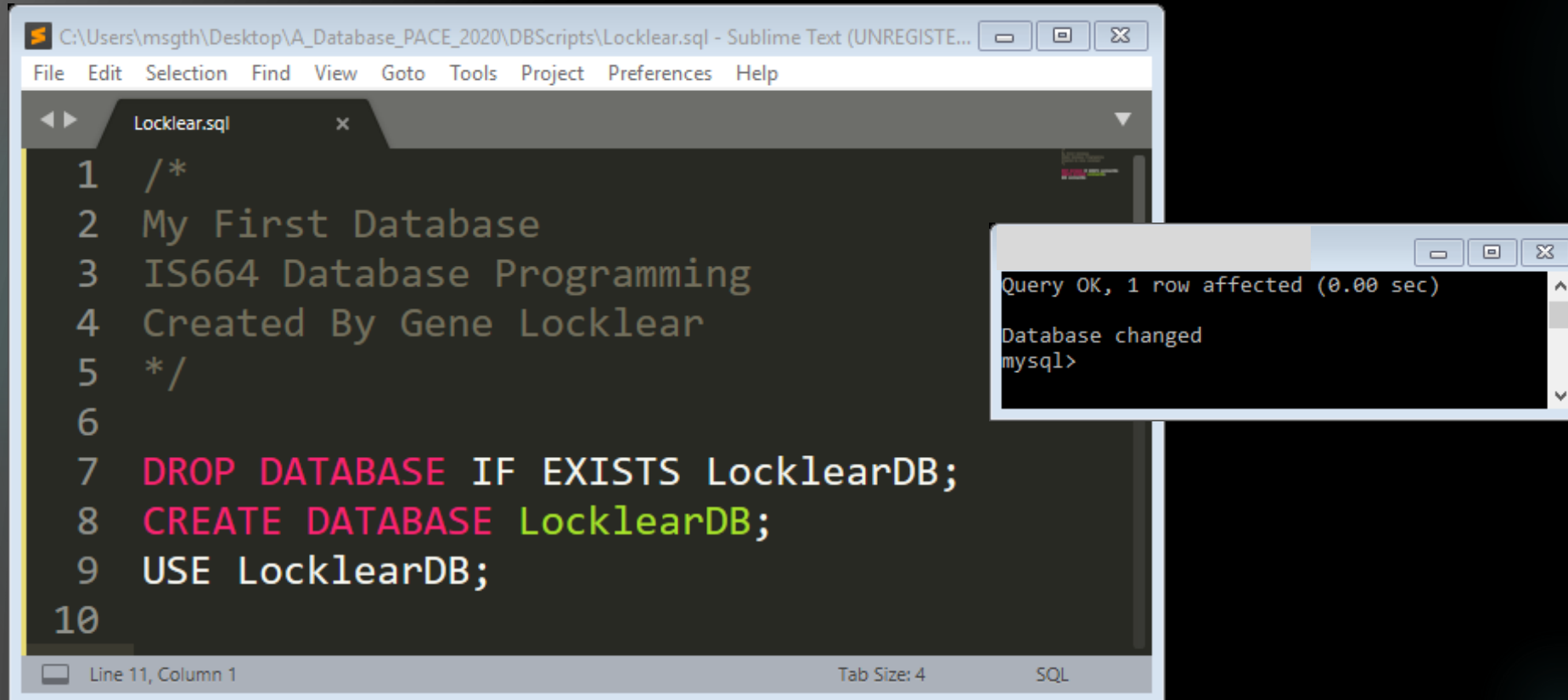


```
mysql> use paceml;
Database changed
mysql> show tables;
+-----+
| Tables_in_paceml |
+-----+
| xiaclass           |
| xiaclasseuclidean |
| xiaclassjson       |
| xiaclassjsonnorm   |
| xiatemplate        |
| yuanclass          |
| yuanclasseuclidean |
| yuanclassjson      |
| yuanclassjsonnorm  |
| yuantemplate       |
+-----+
10 rows in set (0.01 sec)
```

Create and Select a Database

4

- ▶ In order to create a database, we use the **CREATE DATABASE** command.
- ▶ Once we have created the database, we can select it using the **USE** [database name] command.



The screenshot shows a Sublime Text editor window with a file named 'Locklear.sql'. The script contains the following SQL commands:

```
1  /*
2  My First Database
3  IS664 Database Programming
4  Created By Gene Locklear
5  */
6
7  DROP DATABASE IF EXISTS LocklearDB;
8  CREATE DATABASE LocklearDB;
9  USE LocklearDB;
10
```

Below the editor, a terminal window displays the output of the script execution:

```
Query OK, 1 row affected (0.00 sec)

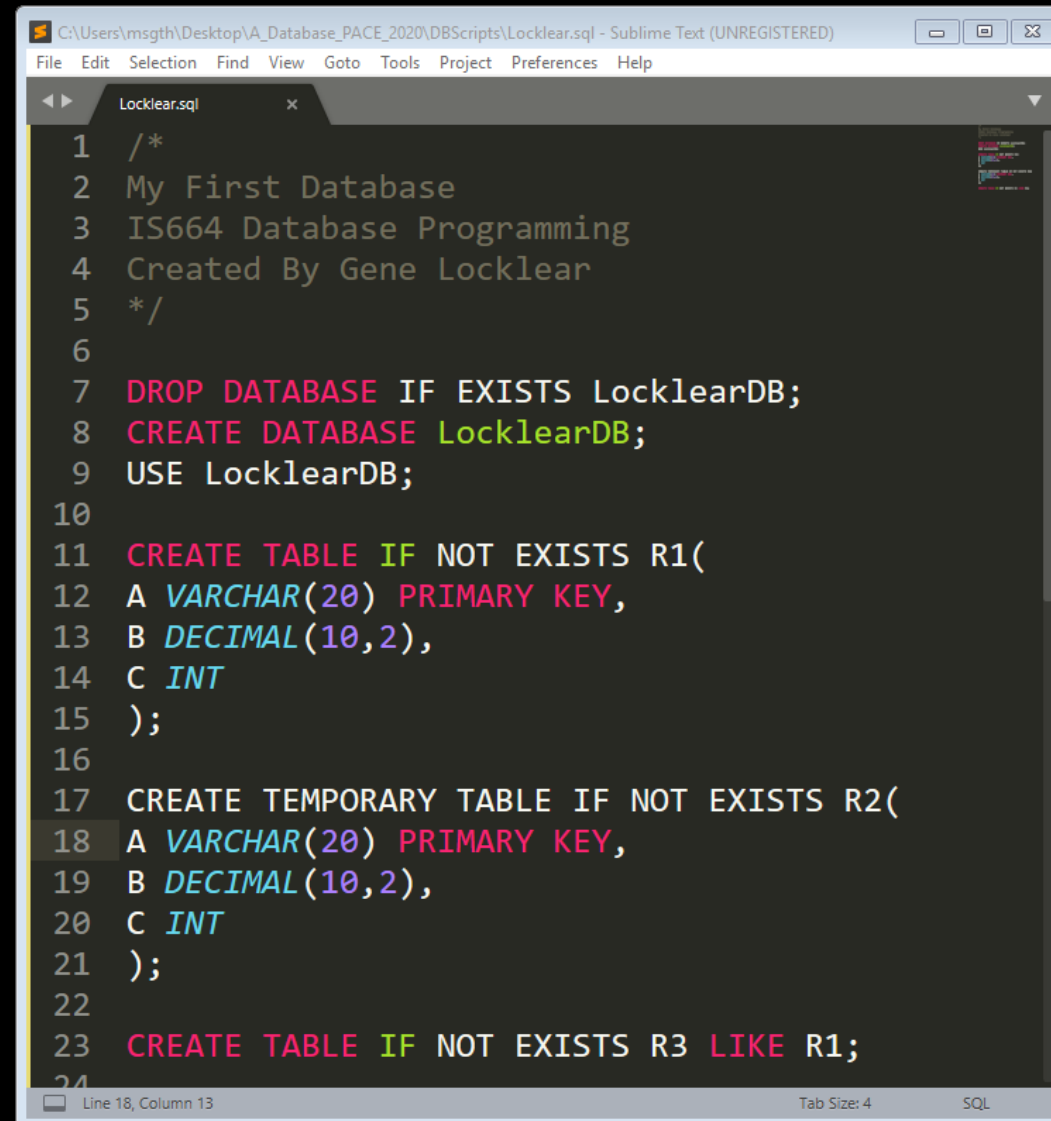
Database changed
mysql>
```

The status bar at the bottom of the editor indicates 'Line 11, Column 1', 'Tab Size: 4', and 'SQL'.

Relation in MySQL

5

- ▶ A maximum of **4096** attributes can be added to a relation.
- ▶ Individual storage engines might impose additional restrictions.
- ▶ Every relation has a maximum tuple size **65,535** bytes.
- ▶ **IF NOT EXISTS** prevents an error when the relation already exists.
- ▶ **TEMPORARY** relations are only visible to the current session and dropped automatically when the session is ended.
- ▶ **LIKE** is used to create a table based on the definition of another table.

A screenshot of a Sublime Text editor window titled 'C:\Users\msgth\Desktop\A_Database_PACE_2020\DBScripts\Locklear.sql - Sublime Text (UNREGISTERED)'. The editor shows a SQL script with the following content:

```
1  /*
2  My First Database
3  IS664 Database Programming
4  Created By Gene Locklear
5  */
6
7  DROP DATABASE IF EXISTS LocklearDB;
8  CREATE DATABASE LocklearDB;
9  USE LocklearDB;
10
11  CREATE TABLE IF NOT EXISTS R1(
12  A VARCHAR(20) PRIMARY KEY,
13  B DECIMAL(10,2),
14  C INT
15  );
16
17  CREATE TEMPORARY TABLE IF NOT EXISTS R2(
18  A VARCHAR(20) PRIMARY KEY,
19  B DECIMAL(10,2),
20  C INT
21  );
22
23  CREATE TABLE IF NOT EXISTS R3 LIKE R1;
24
```

The status bar at the bottom indicates 'Line 18, Column 13', 'Tab Size: 4', and 'SQL'.

Structure of a Relation

6

- ▶ The **DESCRIBE** command shows the structure of a relation.

The image shows a SQL IDE window titled 'C:\Users\msgth\Desktop\AA_DBProgrammingFall2020\Lecture2DB.sql - ...'. The script contains the following SQL commands:

```
7 DROP DATABASE IF EXISTS LocklearDB;
8 CREATE DATABASE LocklearDB;
9 USE LocklearDB;
10
11 CREATE TABLE R1(
12 A VARCHAR(20) PRIMARY KEY,
13 B DECIMAL(10,2),
14 C INT,
15 );
16
17 DESCRIBE R1;
```

An orange arrow points from the 'DESCRIBE R1;' command on line 17 to a 'Client' window. The 'Client' window displays the output of the DESCRIBE command as a table:

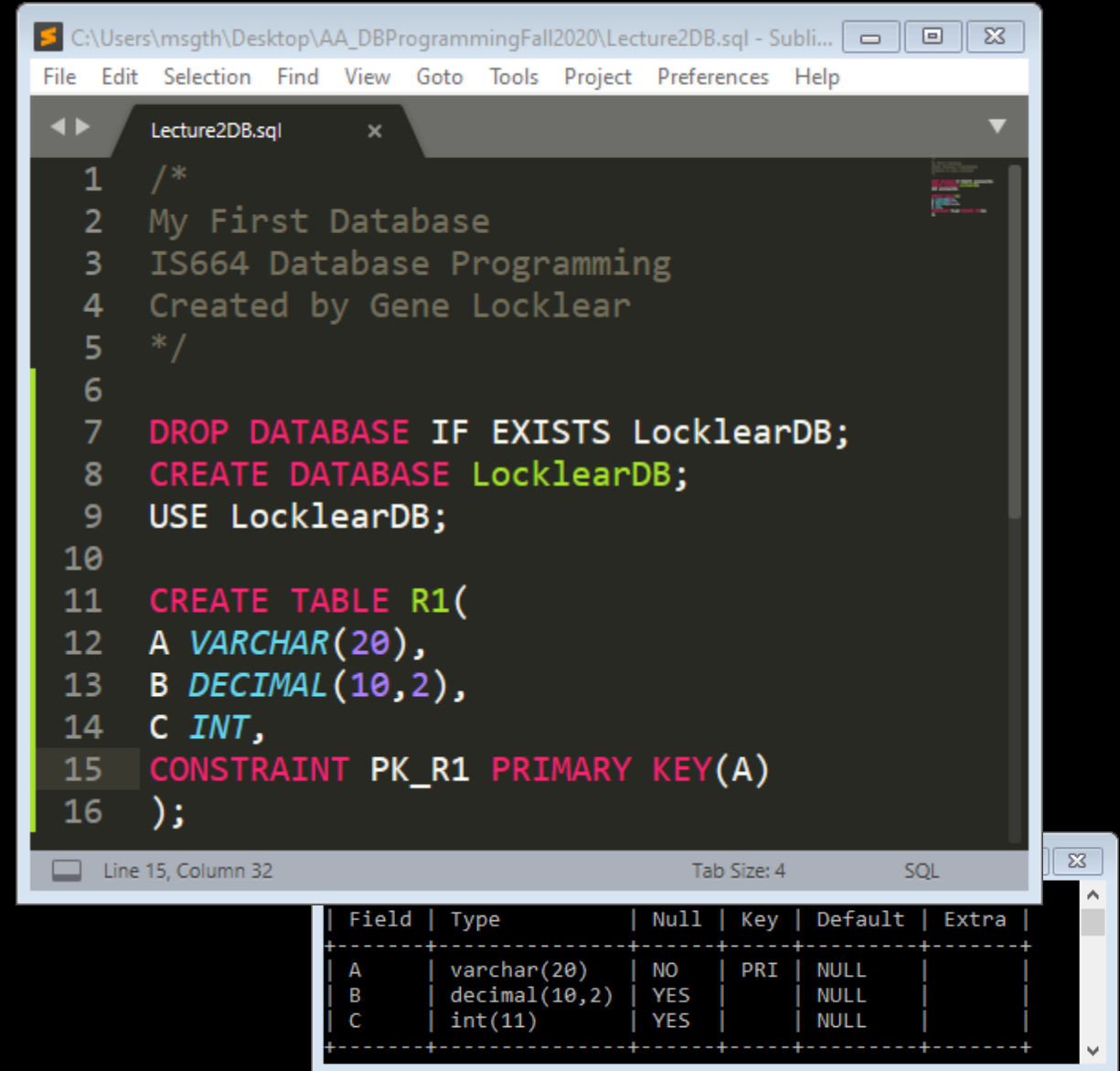
Field	Type	Null	Key	Default	Extra
A	varchar(20)	NO	PRI	NULL	
B	decimal(10,2)	YES		NULL	
C	int(11)	YES		NULL	

Below the table, it says '3 rows in set (0.00 sec)'.

Primary Keys

7

- ▶ A **primary key** uniquely identifies a tuple (row) in a relation.
- ▶ A primary key is defined as a **Constraint** on the relation.
- ▶ **Primary Keys**
 - ▶ By default, are Unique.
 - ▶ Cannot be null.
 - ▶ Can be only one.
 - ▶ Defined on one or more columns.
 - ▶ **Composite Key**
- ▶ A relation **should always** have a primary key.
- ▶ It is preferable to name constraints so that they can be manipulated programmatically.
 - ▶ **In the case of the PRIMARY KEY however, we don't need to name it because there is only ever one Primary Key in a Relation.**



```
1  /*
2  My First Database
3  IS664 Database Programming
4  Created by Gene Locklear
5  */
6
7  DROP DATABASE IF EXISTS LocklearDB;
8  CREATE DATABASE LocklearDB;
9  USE LocklearDB;
10
11  CREATE TABLE R1(
12  A VARCHAR(20),
13  B DECIMAL(10,2),
14  C INT,
15  CONSTRAINT PK_R1 PRIMARY KEY(A)
16  );
```

Field	Type	Null	Key	Default	Extra
A	varchar(20)	NO	PRI	NULL	
B	decimal(10,2)	YES		NULL	
C	int(11)	YES		NULL	

Composite Primary Keys

8

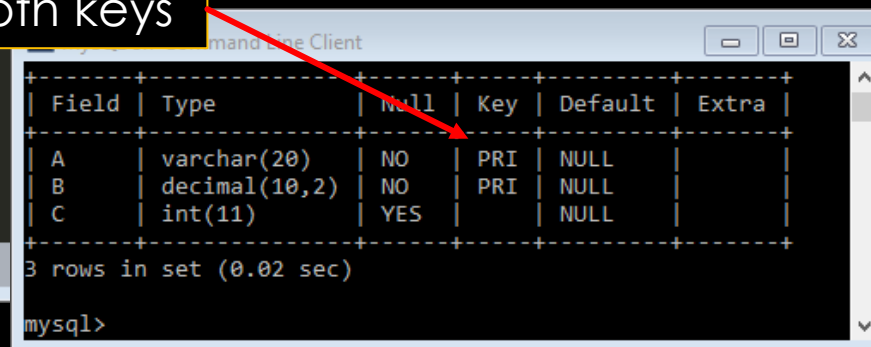
- ▶ A **primary key** that is defined on multiple columns is known as a **composite** key.

```
C:\Users\msgth\Desktop\AA_DBProgrammingFall2020\Lecture2DB.sql - Sublim...
File Edit Selection Find View Goto Tools Project Preferences Help

Lecture2DB.sql x
1  /*
2  My First Database
3  IS664 Database Programming
4  Created by Gene Locklear
5  */
6
7  DROP DATABASE IF EXISTS LocklearDB;
8  CREATE DATABASE LocklearDB;
9  USE LocklearDB;
10
11  CREATE TABLE R1(
12  A VARCHAR(20),
13  B DECIMAL(10,2),
14  C INT,
15  CONSTRAINT PK_R1 PRIMARY KEY(A,B)
16  );

Line 12, Column 15      Tab Size: 4      SQL
```

Both keys



Field	Type	Null	Key	Default	Extra
A	varchar(20)	NO	PRI	NULL	
B	decimal(10,2)	NO	PRI	NULL	
C	int(11)	YES		NULL	

3 rows in set (0.02 sec)

mysql>

Altering Primary Keys

9

- ▶ We can alter the Primary Key programmatically using the **ALTER** command.

The screenshot shows a SQL editor window titled 'C:\Users\msgth\Desktop\AA_DBProgrammingFall2020\Lecture2DB.sql - Sublime Text (UNREGISTERED)' with the following SQL code:

```
7 DROP DATABASE IF EXISTS LocklearDB;
8 CREATE DATABASE LocklearDB;
9 USE LocklearDB;
10
11 CREATE TABLE R1(
12 A VARCHAR(20),
13 B DECIMAL(10,2),
14 C INT,
15 CONSTRAINT PK_R1 PRIMARY KEY(A)
16 );
17
18 DESCRIBE R1;
19
20 ALTER TABLE R1 DROP PRIMARY KEY;
21 ALTER TABLE R1 ADD CONSTRAINT NPK_R1 PRIMARY KEY(B);
22
23 DESCRIBE R1;
```

Below the editor, a 'Command Line Client' window displays the output of the SQL commands. It shows the 'Before' state where table R1 has a primary key on column A, and the 'After' state where the primary key has been moved to column B.

Before

Field	Type	Null	Key	Default	Extra
A	varchar(20)	NO	PRI	NULL	
B	decimal(10,2)	YES		NULL	
C	int(11)	YES		NULL	

3 rows in set (0.00 sec)

Query OK, 0 rows affected (1.38 sec)
Records: 0 Duplicates: 0 Warnings: 0

After

Field	Type	Null	Key	Default	Extra
A	varchar(20)	NO		NULL	
B	decimal(10,2)	NO	PRI	NULL	
C	int(11)	YES		NULL	

Query OK, 0 rows affected (0.87 sec)
Records: 0 Duplicates: 0 Warnings: 0

Line 20, Column 32 Tab Size: 4 SQL

Primary Key: Auto_Increment

10

- ▶ We can automatically generate a unique identity for each row in a table using the **AUTO_INCREMENT** attribute.

```
C:\Users\msgth\Desktop\AA_DBProgrammingFall2020\Lecture2DB.sql - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

Lecture2DB.sql x
7 DROP DATABASE IF EXISTS LocklearDB;
8 CREATE DATABASE LocklearDB;
9 USE LocklearDB;
10
11 CREATE TABLE R1(
12 IDNumber INT AUTO_INCREMENT,
13 A VARCHAR(20),
14 B DECIMAL(10,2),
15 C INT,
16 CONSTRAINT PK_R1 PRIMARY KEY(IDNumber)
17 );
18
19 DESCRIBE R1;
```

Field	Type	Null	Key	Default	Extra
IDNumber	int(11)	NO	PRI	NULL	auto_increment
A	varchar(20)	YES		NULL	
B	decimal(10,2)	YES		NULL	
C	int(11)	YES		NULL	

4 rows in set (0.01 sec)

mysql>

Copy a Relation

11

- ▶ The **AS SELECT * FROM** command allows the data and structure of a table to be copied into another table.

```
C:\Users\msgth\Desktop\A_Database_PACE_2020\DBScripts\Locklear.sql - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
Locklear.sql x
25 DESCRIBE R1;
26 CREATE TABLE R4 AS SELECT * FROM R1;
27 DESCRIBE R4;
28
Line 26, Column 1 Tab Size: 4 SQL
```

```
Query OK, 0 rows affected, 1 warning (0.00 sec)

Query OK, 0 rows affected (0.38 sec)

+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| A     | varchar(20)   | NO   | PRI | NULL    |       |
| B     | decimal(10,2) | YES  |     | NULL    |       |
| C     | int(11)       | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)

Query OK, 0 rows affected (0.34 sec)
Records: 0 Duplicates: 0 Warnings: 0

+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| A     | varchar(20)   | NO   |     | NULL    |       |
| B     | decimal(10,2) | YES  |     | NULL    |       |
| C     | int(11)       | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)

mysql>
```

Populate a Relation

12

- ▶ **INSERT INTO** [relation] **VALUES**(v1,v2,...) allows the relation to be populated with data.

```
C:\Users\msgth\Desktop\AA_DBProgrammingFall2020\Lecture2DB.sql - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

Lecture2DB.sql x DATA_DB.sql x
7 DROP DATABASE IF EXISTS LocklearDB;
8 CREATE DATABASE LocklearDB;
9 USE LocklearDB;
10
11 CREATE TABLE R1(
12 A VARCHAR(20),
13 B DECIMAL(10,2),
14 C INT,
15 CONSTRAINT PK_R1 PRIMARY KEY(A)
16 );
17
18 INSERT INTO R1 VALUES('GENE',100.00,54);
19 INSERT INTO R1 VALUES('GEORGE',200.00,45);
20
21 SELECT * FROM R1;
```

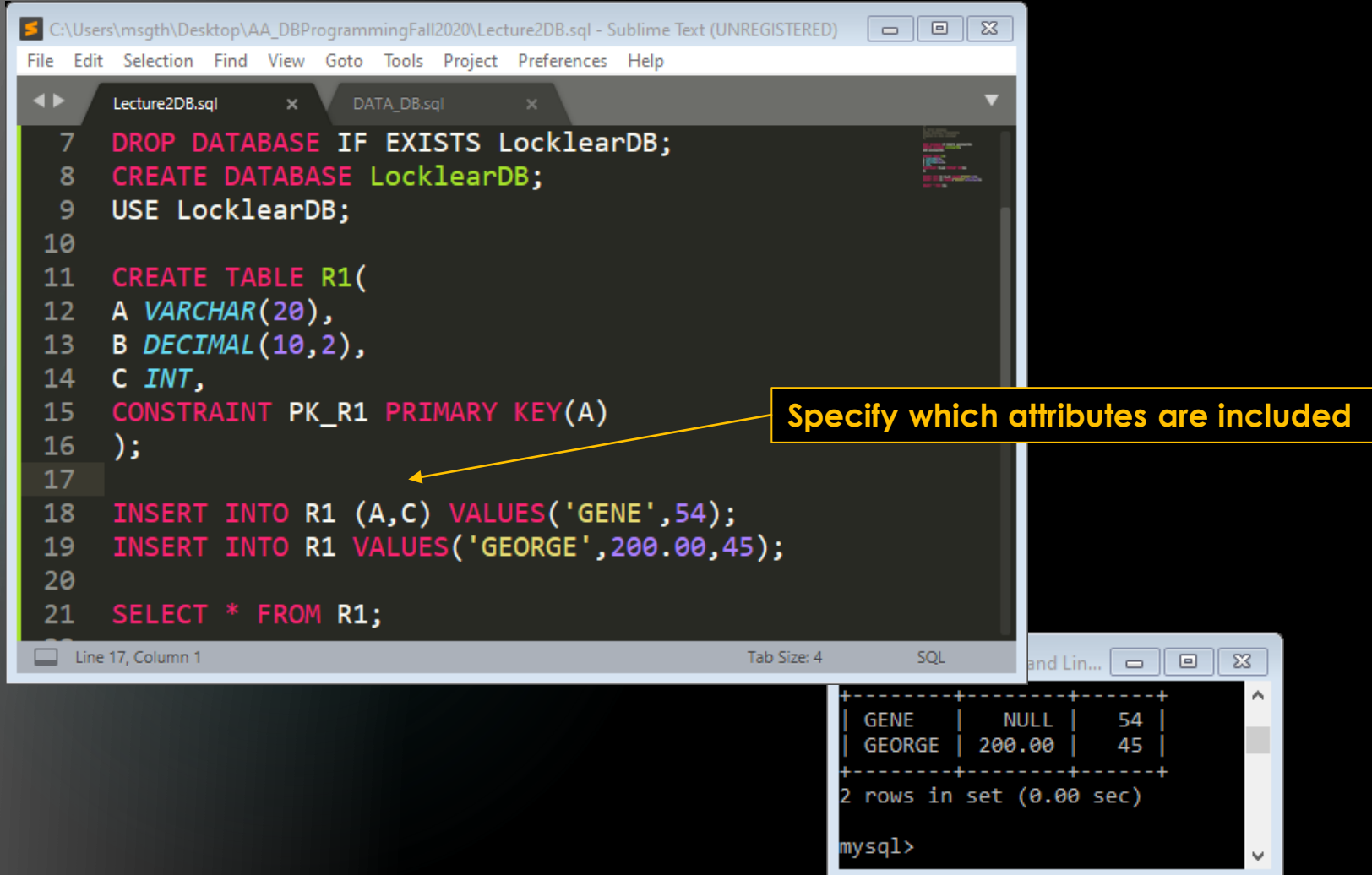
```
+-----+-----+-----+
| GENE  | 100.00 | 54   |
| GEORGE| 200.00 | 45   |
+-----+-----+-----+
2 rows in set (0.00 sec)

mysql>
```

Populate a Relation

13

- ▶ If an attribute can be null and we do not have the value for that attribute, **we must specify**, in the insert statement, that we are **only including certain attribute** values.



```
C:\Users\msgth\Desktop\AA_DBProgrammingFall2020\Lecture2DB.sql - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

Lecture2DB.sql x DATA_DB.sql x
7 DROP DATABASE IF EXISTS LocklearDB;
8 CREATE DATABASE LocklearDB;
9 USE LocklearDB;
10
11 CREATE TABLE R1(
12 A VARCHAR(20),
13 B DECIMAL(10,2),
14 C INT,
15 CONSTRAINT PK_R1 PRIMARY KEY(A)
16 );
17
18 INSERT INTO R1 (A,C) VALUES('GENE',54);
19 INSERT INTO R1 VALUES('GEORGE',200.00,45);
20
21 SELECT * FROM R1;
```

Specify which attributes are included

```
mysql>
+-----+-----+-----+
| GENE  | NULL  | 54   |
| GEORGE| 200.00| 45   |
+-----+-----+-----+
2 rows in set (0.00 sec)

mysql>
```

Populate a Relation

14

- ▶ **LOAD DATA LOCAL INFILE [filepath] INTO TABLE [relation]** populates a relation by reading a text file where the attribute values of each tuple are separated by tabs.

The screenshot displays a SQL development environment with two main windows. The primary window, titled 'C:\Users\msgth\Desktop\AA_DBProgrammingFall2020\Lecture2DB.sql - Sublime Text (UNRE...', contains the following SQL code:

```
4 Created by Gene Locklear
5 */
6
7 DROP DATABASE IF EXISTS LocklearDB;
8 CREATE DATABASE LocklearDB;
9 USE LocklearDB;
10
11 CREATE TABLE R1(
12 A VARCHAR(20),
13 B DECIMAL(10,2),
14 C INT,
15 CONSTRAINT PK_R1 PRIMARY KEY(A)
16 );
17
18 LOAD DATA LOCAL INFILE 'C:\\Users\\msgth\\Desktop\\AA_DBProgrammingFall12020\\DATA_DB.sql' INTO TABLE R1;
19
20 SELECT * FROM R1;
```

The status bar at the bottom indicates 'Line 13, Column 17'.

A secondary window, titled 'C:\Users\msgth\Desktop\AA_DBProgrammingFall2020\DATA_DB.sql - Sub...', is overlaid on the main window. It shows the contents of the data file, which is a tab-separated text file:

```
1 GENE 100.00 54
2 GREG 200.00 45
3 GEORGE 300.00 99
```

The status bar for this window indicates 'Line 3, Column 7'.

Below the data file window, a 'SQL Client' window displays the result of the SQL query. It shows a table with three columns: A, B, and C. The data is as follows:

A	B	C
GENE	100.00	54
GEORGE	300.00	99
GREG	200.00	45

Below the table, it states '3 rows in set (0.00 sec)'. A yellow arrow points from the 'LOAD DATA LOCAL INFILE' statement in the main window to the data file window, and another yellow arrow points from the 'SELECT * FROM R1;' statement to the SQL Client window.

Populate a Relation

15

- ▶ We can also specify some other separation when using the **LOAD DATA LOCAL INFILE** command by using **LOAD DATA LOCAL INFILE [filepath] INTO TABLE [relation] FIELD TERMINATED BY [symbol]**

The screenshot displays a Sublime Text editor window with two open files: `Lecture2DB.sql` and `DATA_DB.sql`. The `Lecture2DB.sql` file contains the following SQL code:

```
4 Created by Gene Locklear
5 */
6
7 DROP DATABASE IF EXISTS LocklearDB;
8 CREATE DATABASE LocklearDB;
9 USE LocklearDB;
10
11 CREATE TABLE R1(
12 A VARCHAR(20),
13 B DECIMAL(10,2),
14 C INT,
15 CONSTRAINT PK_R1 PRIMARY KEY(A)
16 );
17
18 LOAD DATA LOCAL INFILE 'C:\\Users\\msgth\\Desktop\\AA_DBProgrammingFall2020\\DATA_DB.sql'
19 INTO TABLE R1 FIELDS TERMINATED BY ',';
20
21 SELECT * FROM R1;
```

The `DATA_DB.sql` file contains the following data:

```
1 ANDY,120.00,51
2 AL,210.00,41
3 ALISON,340.00,91
```

A small window in the foreground shows the resulting table structure and data:

A	B	C
AL	210.00	41
ALISON	340.00	91
ANDY	120.00	51

Generated Columns

16

- Generated Columns are a way to **compute data for an attribute** based on the values of other attributes.

```
C:\Users\msgth\Desktop\AA_DBProgrammingFall2020\Lecture2DB.sql - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
Lecture2DB.sql x DATA_DB.sql x
10
11 CREATE TABLE R1(
12 A VARCHAR(20),
13 B DECIMAL(10,2),
14 C INT,
15 D DECIMAL AS (B * C),
16 CONSTRAINT PK_R1 PRIMARY KEY(A)
17 );
18
19 INSERT INTO R1 (A,B,C) VALUES ('GENE',100.00,54);
20 INSERT INTO R1 (A,B,C) VALUES ('GEORGE',200.00,45);
21
22 SELECT * FROM R1;
23
```

Line 20, Column 24

Tab Size: 4 SQL

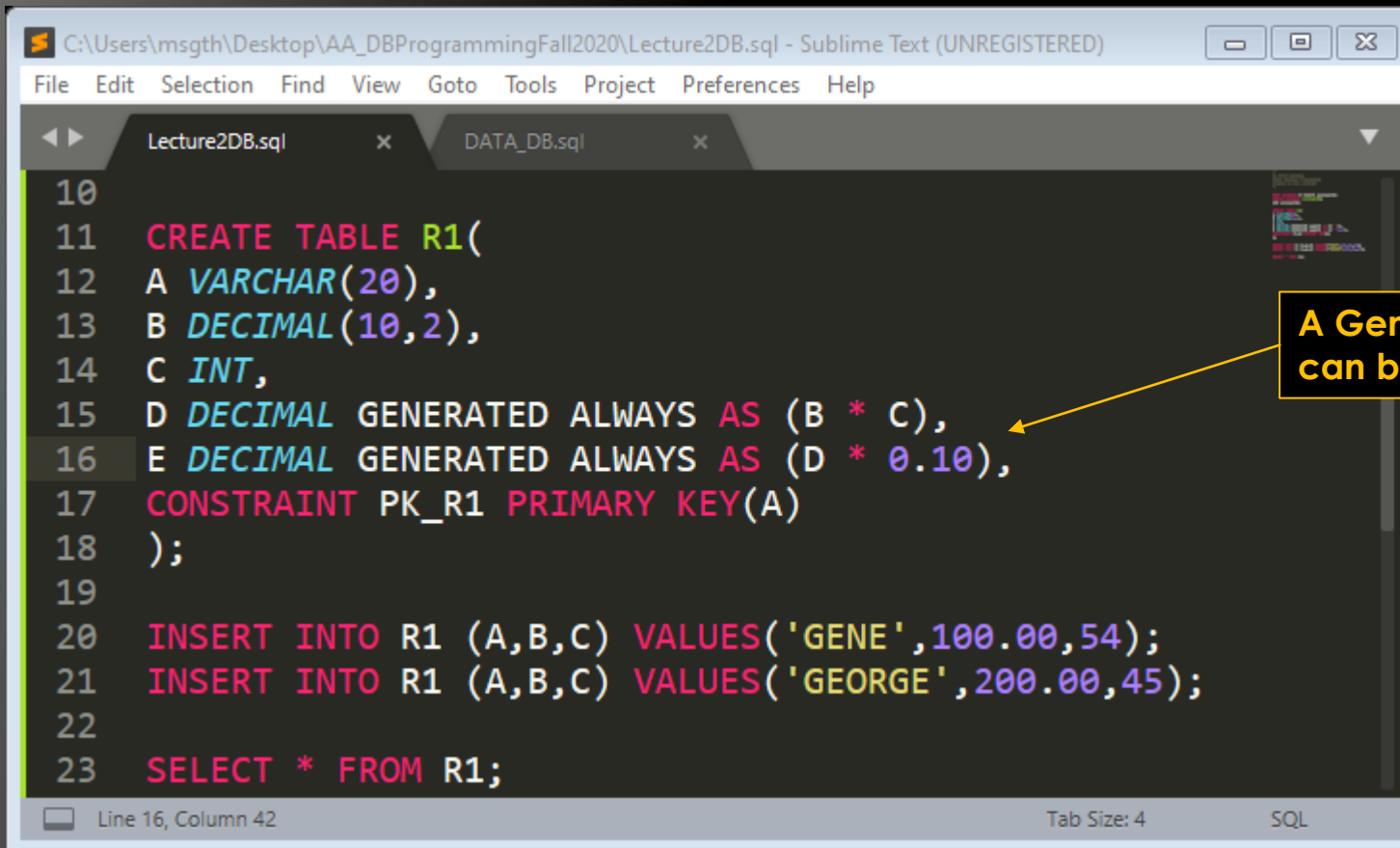
Because a generated column's value is unknown and cannot be included in the INSERT statement, we must always specify the attributes that will be included.

GENE	100.00	54	5400
GEORGE	200.00	45	9000

Generated Columns

17

- ▶ Generated Columns can be used in other Generated Columns.
- ▶ The use of the **Generated Always** keyword is optional but preferred for clarity.



The screenshot shows a Sublime Text editor window with the title bar "C:\Users\msgth\Desktop\AA_DBProgrammingFall2020\Lecture2DB.sql - Sublime Text (UNREGISTERED)". The menu bar includes File, Edit, Selection, Find, View, Goto, Tools, Project, Preferences, and Help. There are two tabs open: "Lecture2DB.sql" and "DATA_DB.sql". The code in the "Lecture2DB.sql" tab is as follows:

```
10
11 CREATE TABLE R1(
12 A VARCHAR(20),
13 B DECIMAL(10,2),
14 C INT,
15 D DECIMAL GENERATED ALWAYS AS (B * C),
16 E DECIMAL GENERATED ALWAYS AS (D * 0.10),
17 CONSTRAINT PK_R1 PRIMARY KEY(A)
18 );
19
20 INSERT INTO R1 (A,B,C) VALUES ('GENE',100.00,54);
21 INSERT INTO R1 (A,B,C) VALUES ('GEORGE',200.00,45);
22
23 SELECT * FROM R1;
```

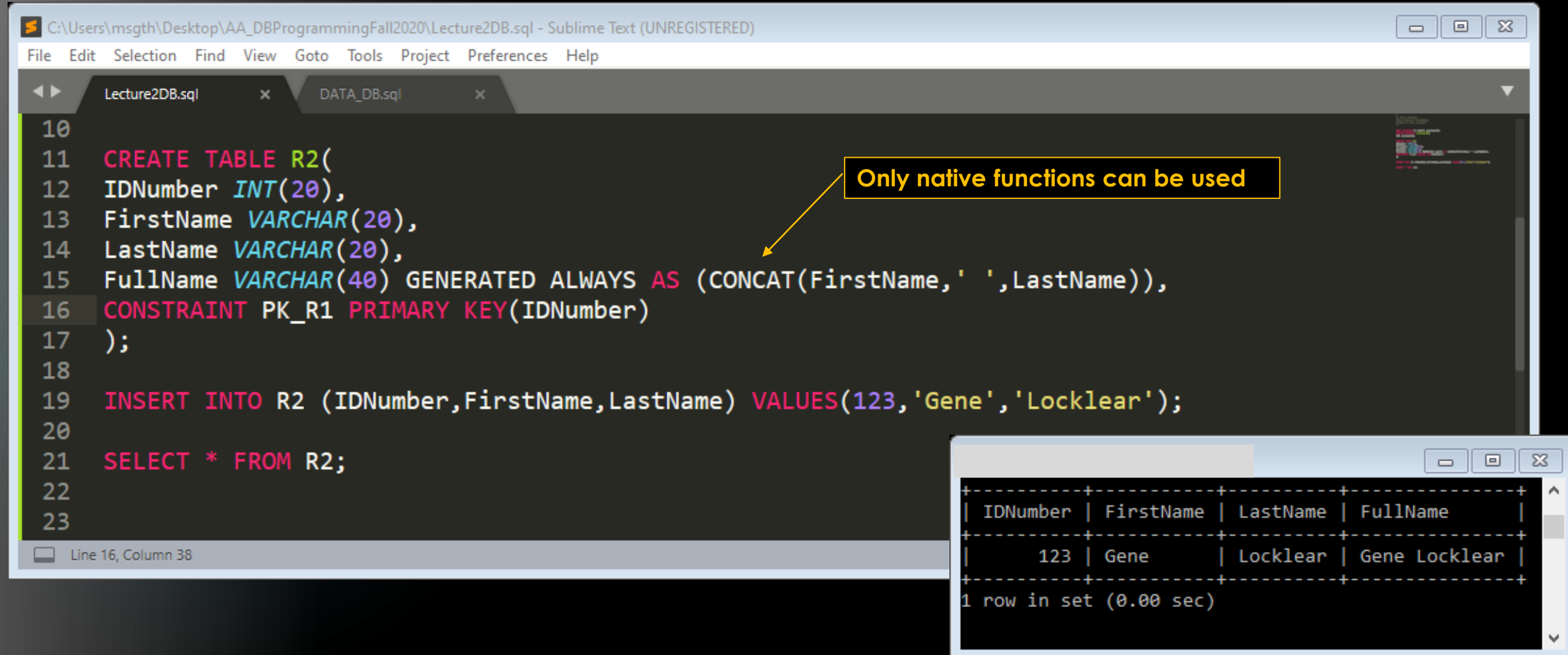
The status bar at the bottom indicates "Line 16, Column 42", "Tab Size: 4", and "SQL".

A Generated Column must be declared before it can be used in another Generated Column

Generated Columns

18

- ▶ Generated Columns can incorporate native MySQL functions.



```
C:\Users\msgth\Desktop\AA_DBProgrammingFall2020\Lecture2DB.sql - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
Lecture2DB.sql DATA_DB.sql
10
11 CREATE TABLE R2(
12 IDNumber INT(20),
13 FirstName VARCHAR(20),
14 LastName VARCHAR(20),
15 FullName VARCHAR(40) GENERATED ALWAYS AS (CONCAT(FirstName, ' ', LastName)),
16 CONSTRAINT PK_R1 PRIMARY KEY(IDNumber)
17 );
18
19 INSERT INTO R2 (IDNumber,FirstName,LastName) VALUES(123,'Gene','Locklear');
20
21 SELECT * FROM R2;
22
23
```

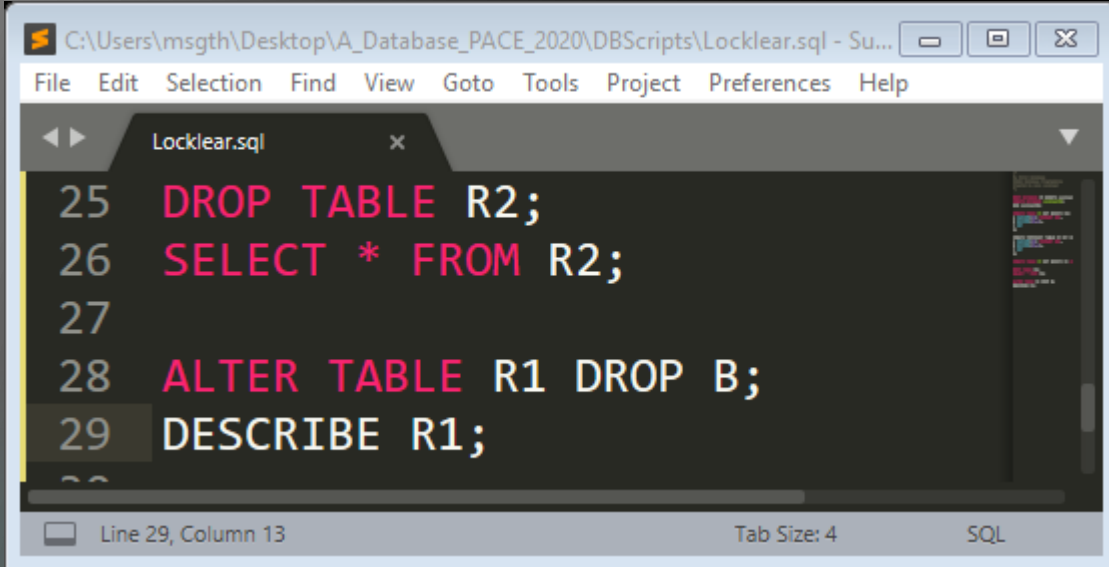
Only native functions can be used

IDNumber	FirstName	LastName	FullName
123	Gene	Locklear	Gene Locklear

1 row in set (0.00 sec)

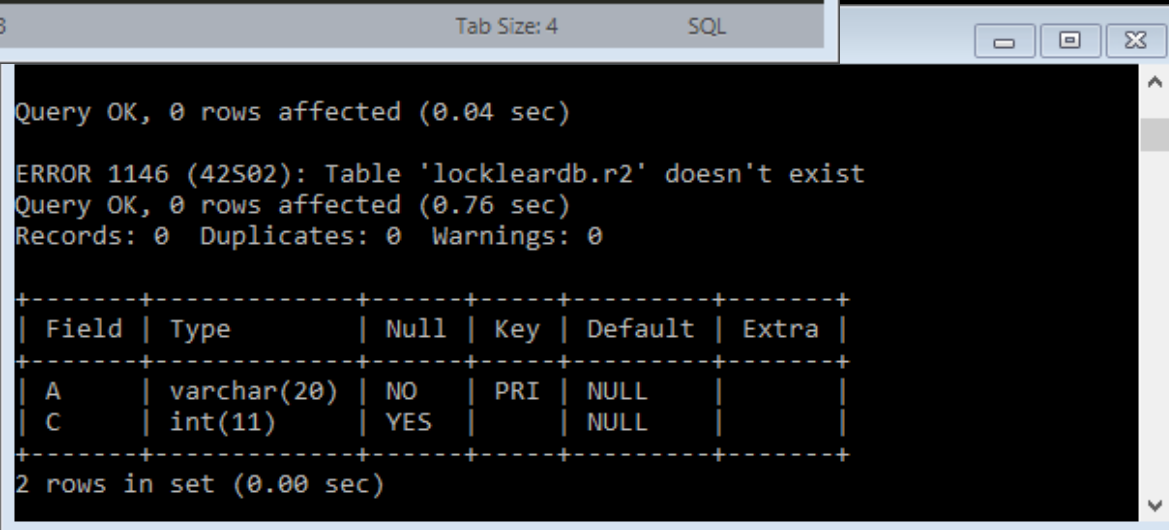
Remove a Relation or Attribute

19



```
C:\Users\msgth\Desktop\A_Database_PACE_2020\DBScripts\Locklear.sql - Su...
File Edit Selection Find View Goto Tools Project Preferences Help
Locklear.sql x
25 DROP TABLE R2;
26 SELECT * FROM R2;
27
28 ALTER TABLE R1 DROP B;
29 DESCRIBE R1;
...
```

- ▶ **DROP** is used to remove a relation from the database.
- ▶ **ALTER** in combination with **DROP** is used to remove an attribute from a relation.



```
Query OK, 0 rows affected (0.04 sec)

ERROR 1146 (42S02): Table 'lockleardb.r2' doesn't exist
Query OK, 0 rows affected (0.76 sec)
Records: 0 Duplicates: 0 Warnings: 0

+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| A     | varchar(20)   | NO   | PRI | NULL    |       |
| C     | int(11)       | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)
```

Modify a Relation

20

- ▶ For a single relation the **UPDATE** statement modifies the attributes in the relation with new values.
- ▶ Specific attributes can be modified using the **SET** clause.
- ▶ The **WHERE** clause can be used in conjunction with the **UPDATE** statement to specify the condition which identifies which tuple to modify.
- ▶ Without a **WHERE** clause all tuples are updated.
- ▶ The **LIMIT** clause specifies a limit on the number of rows that are updated.
- ▶ For multiple attributes, **UPDATE** modifies the tuples in each relation named that satisfy the conditions. **ORDER BY** and **LIMIT** cannot be used.

The screenshot shows a Sublime Text editor window titled 'Locklear.sql' with the following SQL code:

```
25 INSERT INTO R1 VALUES('Gene',100.00,54);
26 SELECT * FROM R1;
27
28 UPDATE R1 SET C = 30;
29 UPDATE R1 SET C = 30 WHERE A = 'Gene';
30 UPDATE R1 SET B = 130, C = 30 WHERE A = 'Gene';
31
32 SELECT * FROM R1;
```

Below the editor is a terminal window showing the execution results of the queries:

```
Query OK, 1 row affected (0.03 sec)

+-----+-----+-----+
| A     | B     | C     |
+-----+-----+-----+
| Gene  | 100.00 | 54    |
+-----+-----+-----+
1 row in set (0.00 sec)

Query OK, 1 row affected (0.04 sec)
Rows matched: 1  Changed: 1  Warnings: 0

Query OK, 0 rows affected (0.03 sec)
Rows matched: 1  Changed: 0  Warnings: 0

Query OK, 1 row affected (0.04 sec)
Rows matched: 1  Changed: 1  Warnings: 0

+-----+-----+-----+
| A     | B     | C     |
+-----+-----+-----+
| Gene  | 130.00 | 30    |
+-----+-----+-----+
1 row in set (0.00 sec)
```

Constraints on a Relation

21

- ▶ In practical terms a **constraint defines a rule to restrict** what values can be stored in the attributes of a relation.
- ▶ Constraints are **applied at the attribute level** and the relation (set of attributes) level.
- ▶ It is standard to apply constraints at the time of table creation.
- ▶ There are six type of constraints:
 - ▶ **NOT NULL** Specifies that an attribute cannot be NULL.
 - ▶ **UNIQUE** Specifies that no duplicate value can be assigned to the attribute.
 - ▶ **PRIMARY KEY** Specifies that the relation accepts unique values for this attribute and that it can be used to uniquely identify each tuple in the relation.
 - ▶ **FOREIGN KEY** Specifies a link between two tables by one specified attribute of both tables.
 - ▶ **CHECK** Specifies whether the value of an attribute is valid.
 - ▶ **DEFAULT** Specifies a value for an attribute when none is given.

Constraints on a Relation

22

```
C:\Users\msgth\Desktop\A_Database_PACE_2020\DBScripts\Locklear.sql - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

Locklear.sql x
31
32 CREATE TABLE R5(
33 A VARCHAR(20),
34 B VARCHAR(20),
35 C VARCHAR(20) NOT NULL DEFAULT 'PACE',
36 D VARCHAR(20) CHECK (C IN ('IS', 'CS', 'NOT SPECIFIED')),
37 CONSTRAINT pk_r5 PRIMARY KEY(A),
38 CONSTRAINT uk_r5 UNIQUE KEY(B),
39 CONSTRAINT fk_r5 FOREIGN KEY(A) REFERENCES R1(A)
40 );
41
42 DESCRIBE R5;
43
```

Line 39, Column 48 Tab Size: 4 SQL

Field	Type	Null	Key	Default	Extra
A	varchar(20)	NO	PRI	NULL	
B	varchar(20)	YES	UNI	NULL	
C	varchar(20)	NO		PACE	
D	varchar(20)	YES		NULL	

4 rows in set (0.01 sec)

mysql>

Constraints on a Relation

23

Parent Table

Faculty	
IDNumber	LastName
S01	Adams
S02	Brown
S03	Chambers

Child Table

Parking	
FID	ParkingSpot
S01	P1
S02	P2
S03	P3

Relationship

If there is an IDNumber in the Parking table then there must be a corresponding IDNumber in the Faculty table

```
C:\Users\msgth\Desktop\AA_DBProgrammingFall2020\Lecture2DB.sql - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

Lecture2DB.sql x DATA_DB.sql x

22
23 CREATE TABLE Faculty(
24   IDNumber VARCHAR(5),
25   LastName VARCHAR(10),
26   CONSTRAINT PK_Faculty PRIMARY KEY(IDNumber)
27 );
28
29 CREATE TABLE Parking(
30   FID VARCHAR(5),
31   ParkingSpot VARCHAR(10),
32   CONSTRAINT PK_Parking PRIMARY KEY(FID),
33   CONSTRAINT FK_Parking FOREIGN KEY(FID) REFERENCES Faculty(IDNumber)
34 );
35
36
37 INSERT INTO Faculty VALUES('S01','Adams'),('S02','Brown'),('S03','Chambers');
38 INSERT INTO Parking VALUES('S01','P1'),('S02','P2'),('S03','P3');
39
40
```

Foreign Key must be unique

Line 37, Column 66 Tab Size: 4 SQL

Constraints on a Relation

Faculty	
IDNumber	LastName
S01	Adams
S02	Brown
S03	Chambers

Parking	
IDNumber	ParkingSpot
S01	P1
S02	P2
S03	P3
S04	P4

```
C:\Users\msgth\Desktop\AA_DBProgrammingFall2020\Lecture2DB.sql - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

Lecture2DB.sql x DATA_DB.sql x

22
23 CREATE TABLE Faculty(
24 IDNumber VARCHAR(5),
25 LastName VARCHAR(10),
26 CONSTRAINT PK_Faculty PRIMARY KEY(IDNumber)
27 );
28
29 CREATE TABLE Parking(
30 FID VARCHAR(5),
31 ParkingSpot VARCHAR(10),
32 CONSTRAINT PK_Parking PRIMARY KEY(FID),
33 CONSTRAINT FK_Parking FOREIGN KEY(FID) REFERENCES Faculty(IDNumber)
34 );
35
36
37 INSERT INTO Faculty VALUES('S01','Adams'),('S02','Brown'),('S03','Chambers');
38 INSERT INTO Parking VALUES('S01','P1'),('S02','P2'),('S03','P3');
39 INSERT INTO Parking VALUES('S04','P4');
40
```

ERROR 1452 (23000): Cannot add or update a child row: a foreign key constraint fails (`lockleardb`.`parking`, CONSTRAINT `FK_Parking` FOREIGN KEY (`IDNumber`) REFERENCES `faculty` (`IDNumber`))

IDNumber	LastName
S01	Adams
S02	Brown
S03	Chambers

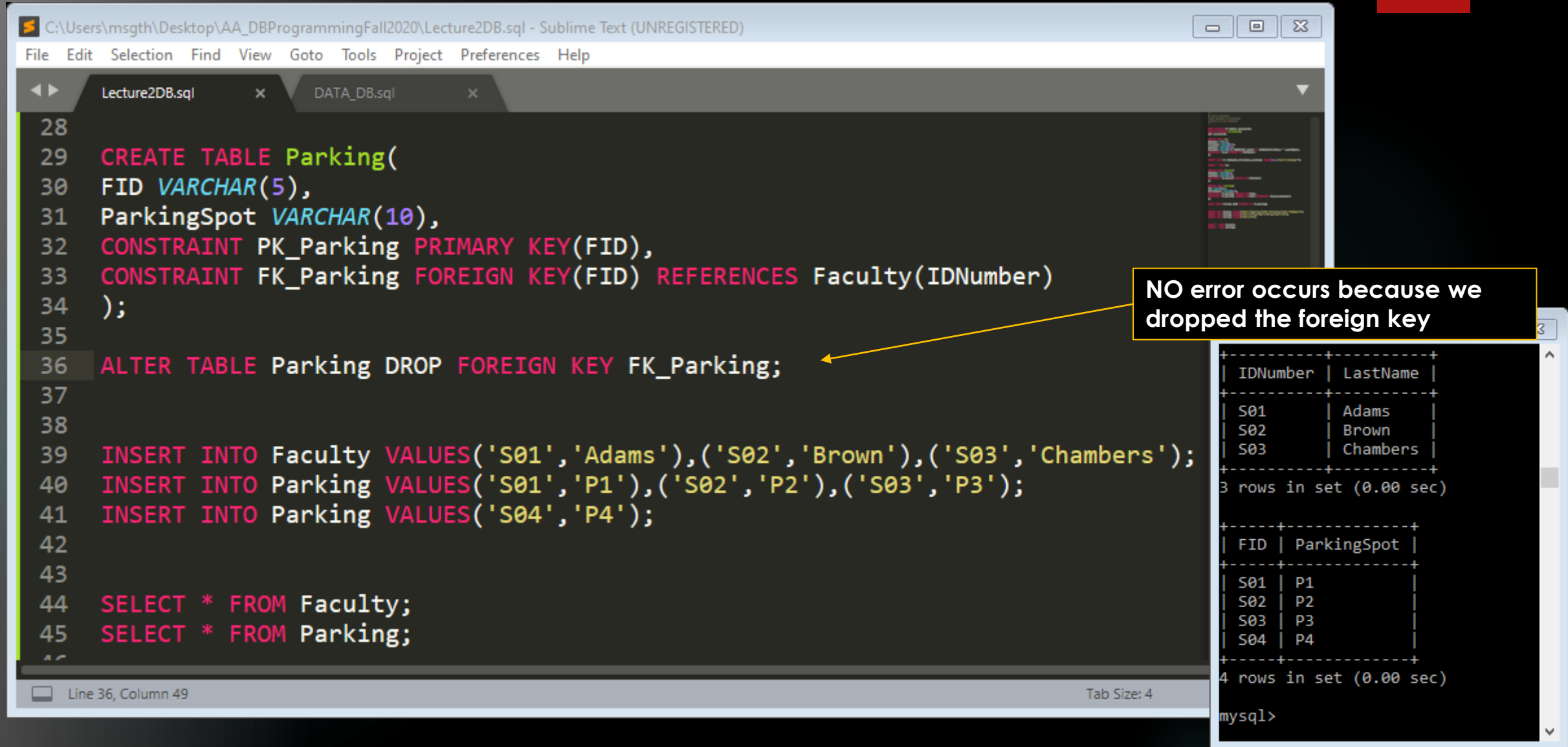
3 rows in set (0.00 sec)

IDNumber	ParkingSpot
S01	P1
S02	P2
S03	P3

ERROR 1452 occurs because Referential Integrity has been violated

Drop Constraints

25



The screenshot shows a Sublime Text editor window with two tabs: 'Lecture2DB.sql' and 'DATA_DB.sql'. The 'Lecture2DB.sql' tab is active and contains the following SQL code:

```
28
29 CREATE TABLE Parking(
30 FID VARCHAR(5),
31 ParkingSpot VARCHAR(10),
32 CONSTRAINT PK_Parking PRIMARY KEY(FID),
33 CONSTRAINT FK_Parking FOREIGN KEY(FID) REFERENCES Faculty(IDNumber)
34 );
35
36 ALTER TABLE Parking DROP FOREIGN KEY FK_Parking;
37
38
39 INSERT INTO Faculty VALUES('S01','Adams'),('S02','Brown'),('S03','Chambers');
40 INSERT INTO Parking VALUES('S01','P1'),('S02','P2'),('S03','P3');
41 INSERT INTO Parking VALUES('S04','P4');
42
43
44 SELECT * FROM Faculty;
45 SELECT * FROM Parking;
```

A yellow callout box with an arrow pointing to line 36 of the code contains the text: "NO error occurs because we dropped the foreign key".

Below the code, a MySQL terminal window shows the output of the queries:

```
mysql>
+-----+-----+
| IDNumber | LastName |
+-----+-----+
| S01      | Adams   |
| S02      | Brown   |
| S03      | Chambers|
+-----+-----+
3 rows in set (0.00 sec)

+-----+-----+
| FID | ParkingSpot |
+-----+-----+
| S01 | P1          |
| S02 | P2          |
| S03 | P3          |
| S04 | P4          |
+-----+-----+
4 rows in set (0.00 sec)

mysql>
```

Add Constraints

26

C:\Users\msgth\Desktop\AA_DBProgrammingFall2020\Lecture2DB.sql - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

Lecture2DB.sql

DATA_DB.sql

```
28
29 CREATE TABLE Parking(
30 FID VARCHAR(5),
31 ParkingSpot VARCHAR(10),
32 CONSTRAINT PK_Parking PRIMARY KEY(FID),
33 CONSTRAINT FK_Parking FOREIGN KEY(FID) REFERENCES Faculty(IDNumber)
34 );
35
36 ALTER TABLE Parking ADD CONSTRAINT UK_Parking UNIQUE(ParkingSpot);
37
38 DESCRIBE Parking;
```

Line 38, Column 18

Tab Size: 4

SQL

Field	Type	Null	Key	Default	Extra
FID	varchar(5)	NO	PRI	NULL	
ParkingSpot	varchar(10)	YES	UNI	NULL	

Normalization of Relations

27

- ▶ The Normalization of the Relations in a database is designed to **eliminate redundancy** and **prevent data anomalies**.
- ▶ Normalization eliminates redundancy by **removing duplicate data** and **reducing data dependency**.
- ▶ There are really only 3 forms of Normalization that we are concerned about in general.
- ▶ **STEP 1: First Normal Form**
 - ▶ All attribute values are atomic.
 - ▶ Every record (row) should be unique.
- ▶ **STEP 2: Second Normal Form**
 - ▶ Must be in First Normal Form
 - ▶ All attributes are dependent on the Primary Key (Primary Key may be a Composite Key)
- ▶ **STEP 3: Third Normal Form**
 - ▶ Must be in Second Normal Form
 - ▶ All transitive dependency has been eliminated (R3 is dependent on R2 is dependent on R1)

Normalization of Relations

28

Student			
IDNumber	LastName	StudentNumber	Level
S0186	Adams	Pace_001	Freshman
S0213	Brown	Pace_002	Sophomore
S0398	Chambers	Pace_003	Junior

Multiple candidate keys define a row

2NF

All attributes are dependent on a single key

Student	
IDNumber	LastName
S0186	Adams
S0213	Brown
S0398	Chambers

3NF

Student_Year	
StudentNumber	Level
Pace_001	Freshman
Pace_002	Sophomore
Pace_003	Junior

3NF

There is no dependency between the two tables

Practical Exercise

29

- ▶ Create the database **pacestudent**.
- ▶ Create and populate the relation **Student** in **pacestudent** that contains the following information:

StudentID	LastName	FirstName	Age	Gender	Major
U0001	Anderson	Donald	23	Male	CS
U0002	Baker	Erica	24	Female	IS

- Create and populate the relation **MealPlan** in **pacestudent** that contains the following information and maintains referential integrity with **Student**:

StudentMP	MealPlanCode
U0001	A678
U0002	B789