IS664 Database Programming
Fall 2022

Fundamentals

Database Programming

LECTURE 7: STORED PROCEDURE PROGRAMMING

Professor HG Locklear
hlocklear@pace.edu

General

- The programming power of MySQL **is limited** when compared to other languages.
- MySQL language constructs are **designed specifically to work with MySQL databases** rather than as a general-purpose programming language.
- MySQL provided **extensions to SQL** known as stored programs. **Stored programs** can include procedural code that controls the flow of execution of a database operation.
- **There are four types of stored programs:**
  - **Stored Procedure**
    - Can be called from an application that has access to the database.
  - **Stored Function**
    - Can be called from a SQL statement.
  - **Trigger**
    - Is executed in response to an INSERT, UPDATE, or DELETE statement on a specific table.
  - **Event**
    - Is executed at a scheduled time.

General

- **MySQL supports three types of programming structures.**

Types of Stored Programs		
Type		**Description**
Stored Routines	**Stored Procedure**	Can be called from a SQL statement Can be called from an application that has access to the database.
	Stored Function	Can be called from a SQL statement. Can be considered a user-defined function.
Trigger		Is executed in response to an INSERT, UPDATE, or DELETE statement on a specific table.
Event		Is executed at a scheduled time.

Programming Methodology

- Creating stored programs in MySQL is about solving problems using the tools you know.
- At this point you know:
  - **Database Schema Construction**
  - **Single Table Queries**
  - **Use of Native Functions in Queries**
  - **Fundamental Language Programming Constructs**
  - **Use of User-Defined Functions in Queries**
  - **Multiple Table Queries**
- The more tools you know, the more complex problems you can solve and the faster you can solve it.
- Also…more tools means more elegant solutions.
  - **Elegance means less code and easier debugging**
- **DO NOT BE IN A HURRY**
- **CONSTRUCT TEST CASES/SCRIPTS BEFORE YOU CODE**
- **DEVELOP A STYLE…STICK TO IT**
- **USE STEPWISE REFINEMENT**

Stored Routines

- MySQL supports stored routines (procedures and functions).
- **https://dev.mysql.com/doc/refman/8.0/en/stored-routines.html**

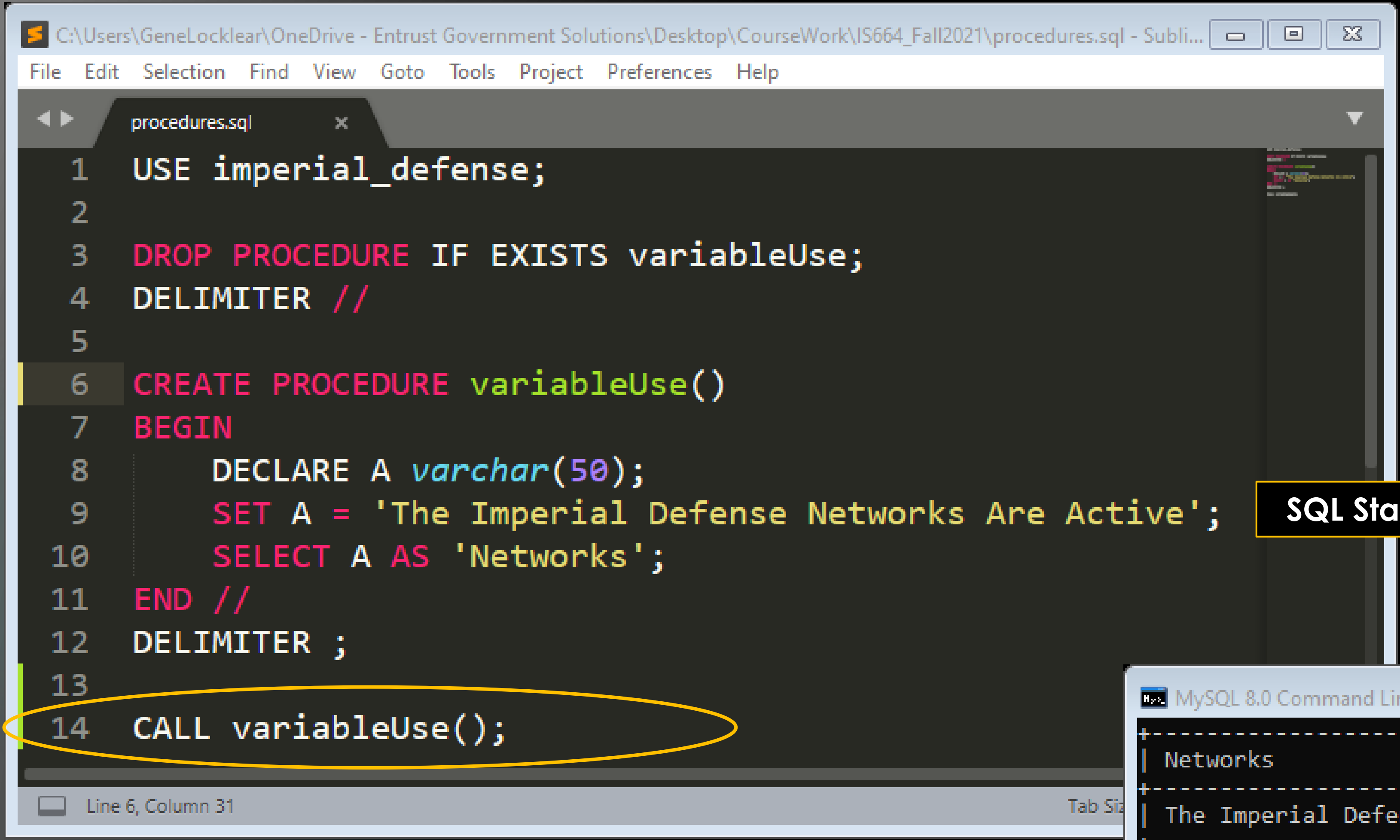
- A **Stored Routine** is a set of SQL statements that can be stored in the server. Once this has been done, clients don't need to keep reissuing the individual statements but can refer to the stored routine instead.
- **Stored routines can be particularly useful in certain situations:**
  - When multiple client applications are written in different languages or work on different platforms but need to perform the same database operations.
  - When security is paramount. (***Access to data is only through use of stored program)***
  - Stored routines can provide improved performance because less information needs to be sent between the server and the client.
- Stored routines also enable you to have libraries of functions in the database server.

Stored Procedure

- Stored procedures are created with the **CREATE PROCEDURE** statement.
- A stored procedure is invoked using a **CALL** statement and can only pass back values using output variables.
- Stored procedures can be dropped with **DROP PROCEDUE** and altered with the **ALTER PROCEDURE** statements .

- **A stored procedure is associated with a particular database. This implies:**
  - **USE** statements within stored procedures are **not permitted**.
  - You can qualify procedure names with the database name. This can be used to **refer to a procedure that is not in the current database**.
  - When a database is dropped, **all stored routines associated** with it are dropped as well.

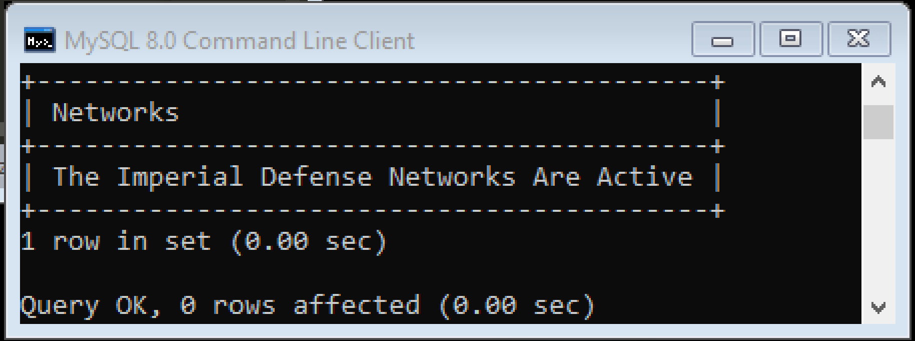
Simple Stored Procedure



C:\Users\GeneLocklear\OneDrive - Entrust Government Solutions\Desktop\CourseWork\IS664_Fall2021\procedures.sql - Subli...
File Edit Selection Find View Goto Tools Project Preferences Help
procedures.sql
Line 6, Column 31
Tab Si

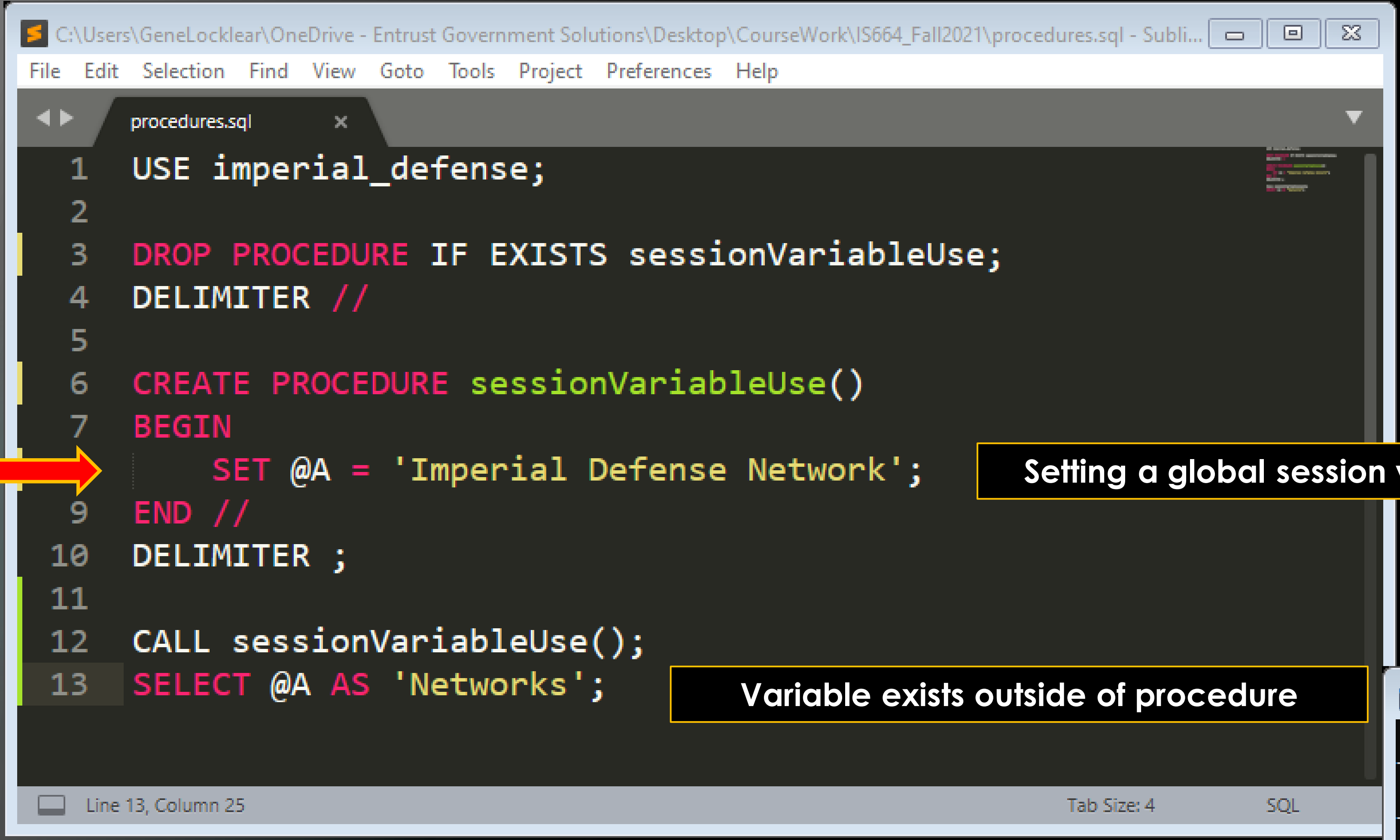
```sql
USE imperial_defense;

DROP PROCEDURE IF EXISTS variableUse;
DELIMITER //

CREATE PROCEDURE variableUse()
BEGIN
    DECLARE A varchar(50);
    SET A = 'The Imperial Defense Networks Are Active';
    SELECT A AS 'Networks';
END //
DELIMITER ;

CALL variableUse();
```

SQL Statement



```
+------------------------------------------+
| Networks                                 |
+------------------------------------------+
| The Imperial Defense Networks Are Active |
+------------------------------------------+
1 row in set (0.00 sec)

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

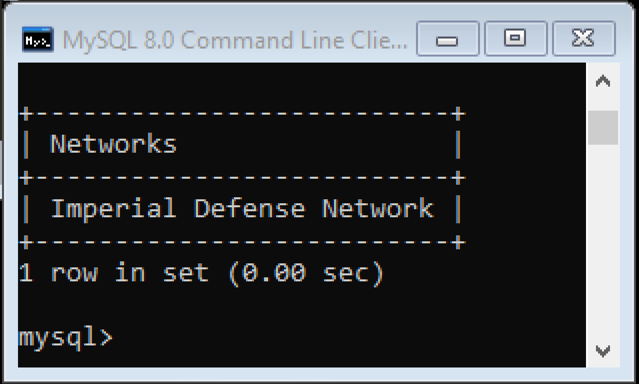
Simple Stored Procedure


C:\Users\GeneLocklear\OneDrive - Entrust Government Solutions\Desktop\CourseWork\IS664_Fall2021\procedures.sql - Subli...
File Edit Selection Find View Goto Tools Project Preferences Help
procedures.sql
Line 13, Column 25
Tab Size: 4
SQL

```sql
USE imperial_defense;

DROP PROCEDURE IF EXISTS sessionVariableUse;
DELIMITER //

CREATE PROCEDURE sessionVariableUse()
BEGIN
    SET @A = 'Imperial Defense Network';
END //
DELIMITER ;

CALL sessionVariableUse();
SELECT @A AS 'Networks';
```

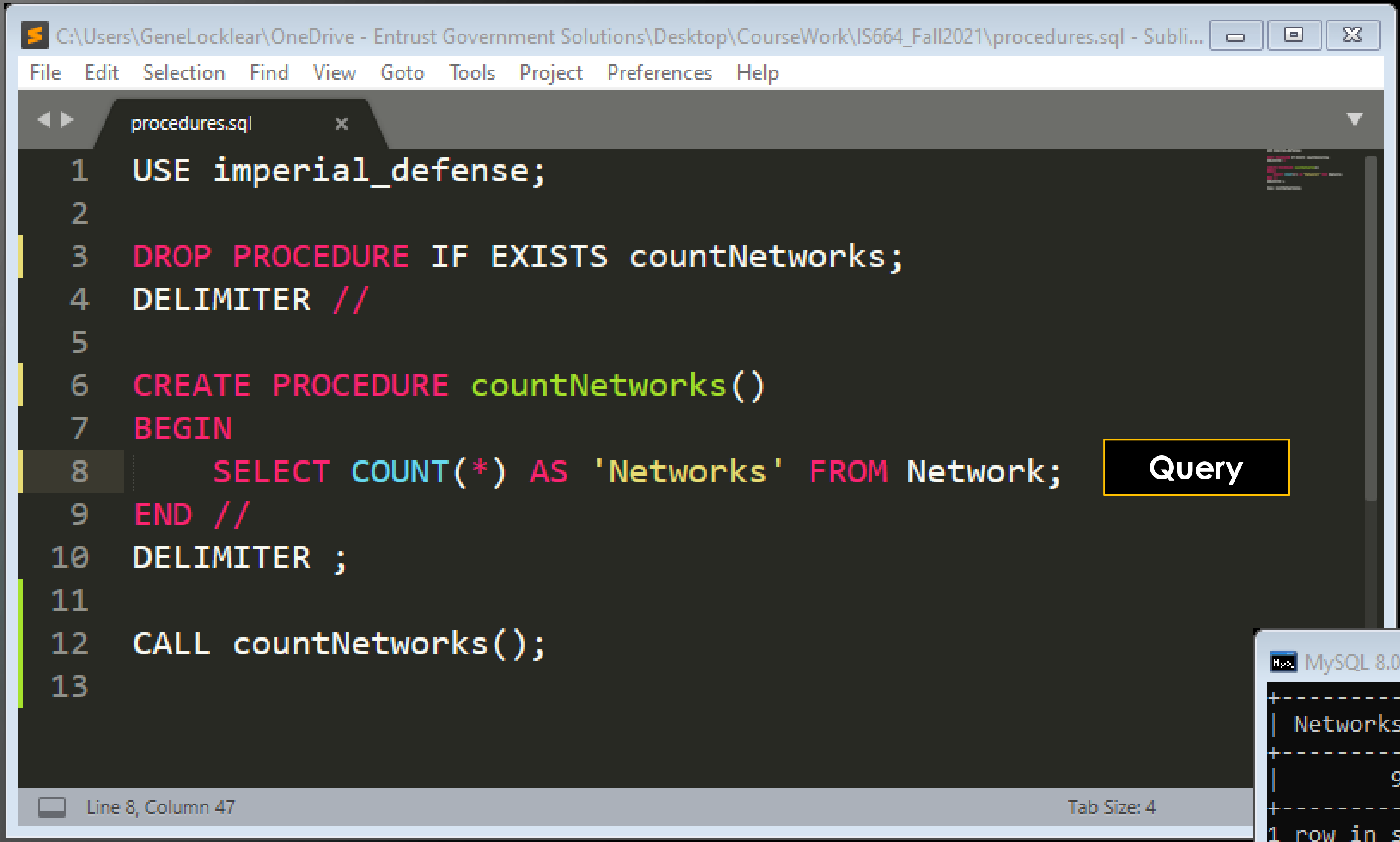
Setting a global session variable

Variable exists outside of procedure



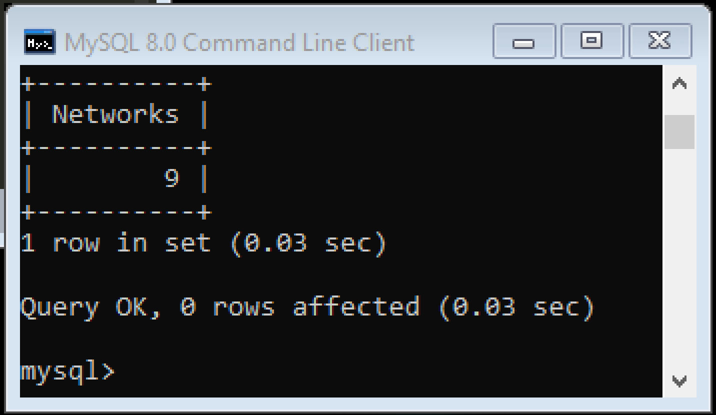
```
+--------------------------+
| Networks                 |
+--------------------------+
| Imperial Defense Network |
+--------------------------+
1 row in set (0.00 sec)

mysql>
```

Simple Stored Procedure


C:\Users\GeneLocklear\OneDrive - Entrust Government Solutions\Desktop\CourseWork\IS664_Fall2021\procedures.sql - Subli...
File Edit Selection Find View Goto Tools Project Preferences Help
procedures.sql
Line 8, Column 47
Tab Size: 4

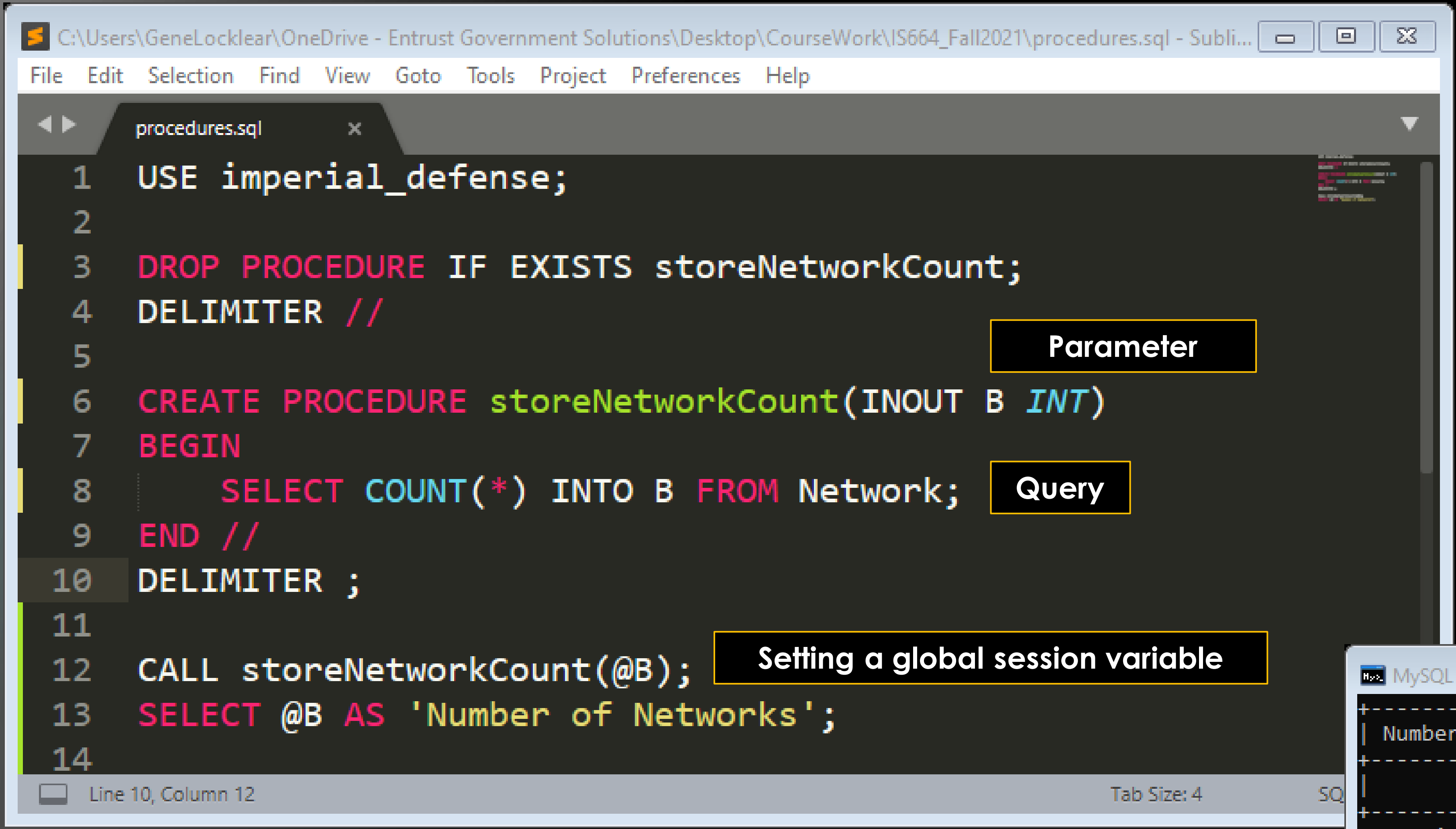
```sql
USE imperial_defense;

DROP PROCEDURE IF EXISTS countNetworks;
DELIMITER //

CREATE PROCEDURE countNetworks()
BEGIN
    SELECT COUNT(*) AS 'Networks' FROM Network;
END //
DELIMITER ;

CALL countNetworks();
```

Query



```
+----------+
| Networks |
+----------+
|        9 |
+----------+
1 row in set (0.03 sec)

Query OK, 0 rows affected (0.03 sec)

mysql>
```

Simple Stored Procedure

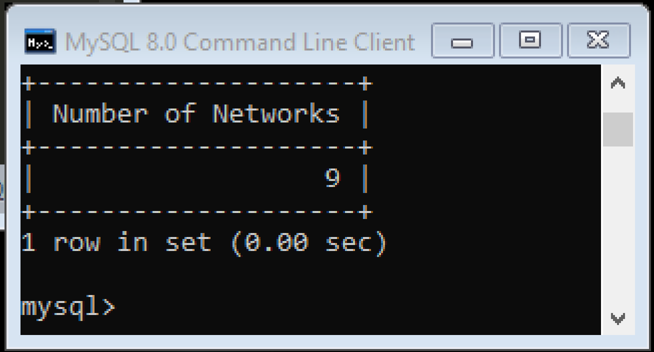

C:\Users\GeneLocklear\OneDrive - Entrust Government Solutions\Desktop\CourseWork\IS664_Fall2021\procedures.sql - Subli...
File Edit Selection Find View Goto Tools Project Preferences Help
procedures.sql
Line 10, Column 12
Tab Size: 4

```sql
USE imperial_defense;

DROP PROCEDURE IF EXISTS storeNetworkCount;
DELIMITER //

CREATE PROCEDURE storeNetworkCount(INOUT B INT)
BEGIN
    SELECT COUNT(*) INTO B FROM Network;
END //
DELIMITER ;

CALL storeNetworkCount(@B);
SELECT @B AS 'Number of Networks';

```

Parameter

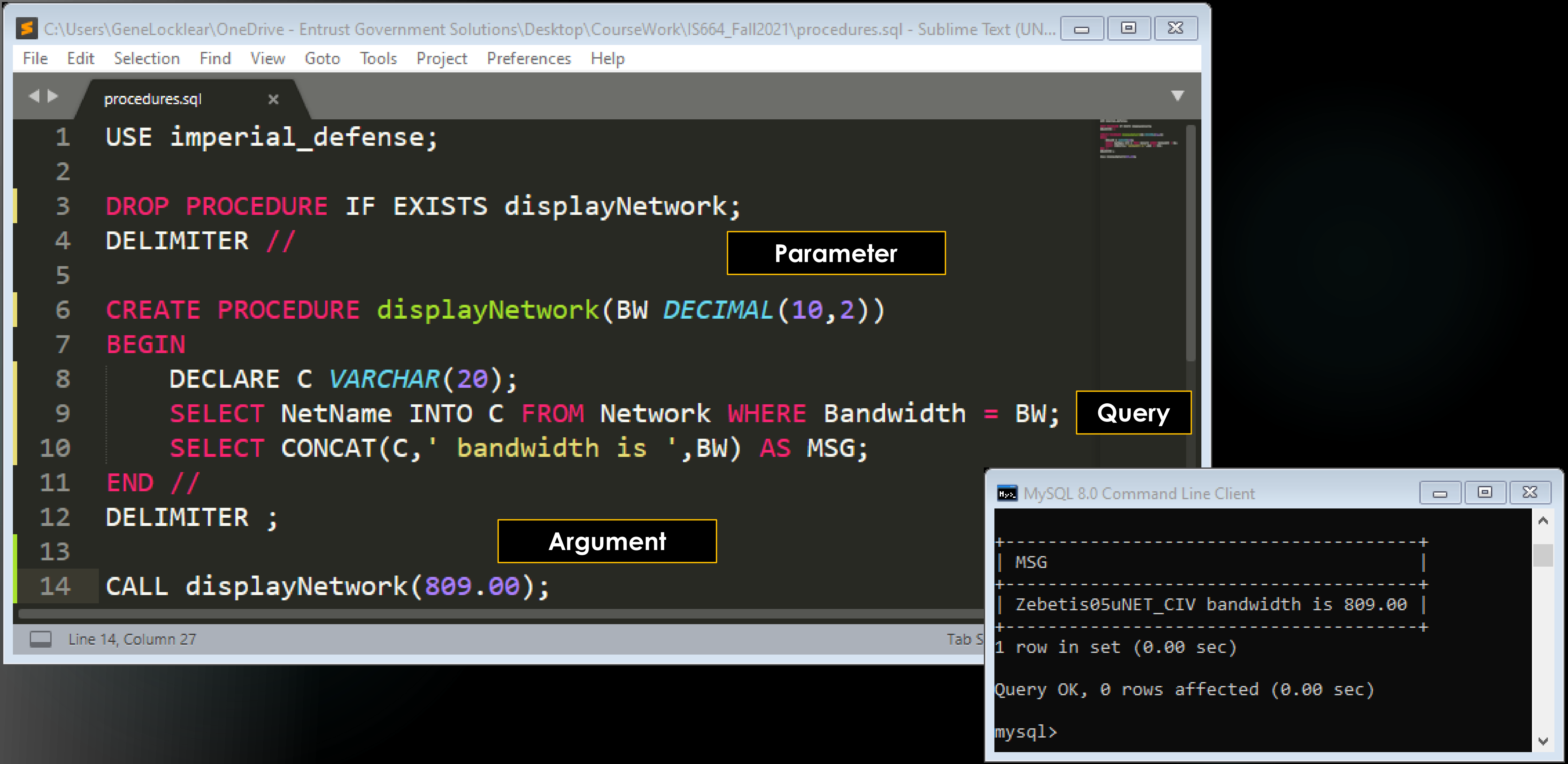
Query

Setting a global session variable



```
+--------------------+
| Number of Networks |
+--------------------+
|                  9 |
+--------------------+
1 row in set (0.00 sec)

mysql>
```

Simple Stored Procedure


C:\Users\GeneLocklear\OneDrive - Entrust Government Solutions\Desktop\CourseWork\IS664_Fall2021\procedures.sql - Sublime Text (UN...
File Edit Selection Find View Goto Tools Project Preferences Help
procedures.sql
Line 14, Column 27
MySQL 8.0 Command Line Client
Parameter
Query
Argument

```sql
USE imperial_defense;

DROP PROCEDURE IF EXISTS displayNetwork;
DELIMITER //

CREATE PROCEDURE displayNetwork(BW DECIMAL(10,2))
BEGIN
    DECLARE C VARCHAR(20);
    SELECT NetName INTO C FROM Network WHERE Bandwidth = BW;
    SELECT CONCAT(C,' bandwidth is ',BW) AS MSG;
END //
DELIMITER ;

CALL displayNetwork(809.00);
```

```
+--------------------------------------+
| MSG                                  |
+--------------------------------------+
| Zebetis05uNET_CIV bandwidth is 809.00 |
+--------------------------------------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

mysql>
```

Block Structure

- A **Block** consist of various types of declarations (variables, cursors, handlers) and program code (assignments, conditionals statements, loops…)
- **The order in which these occur matters:**
  - Variables and Condition declarations
  - Cursor Declarations
  - Exception Handler Declarations
  - Program Code

```
BEGIN
-- This is a Block
-- Declarations and Code
END
```

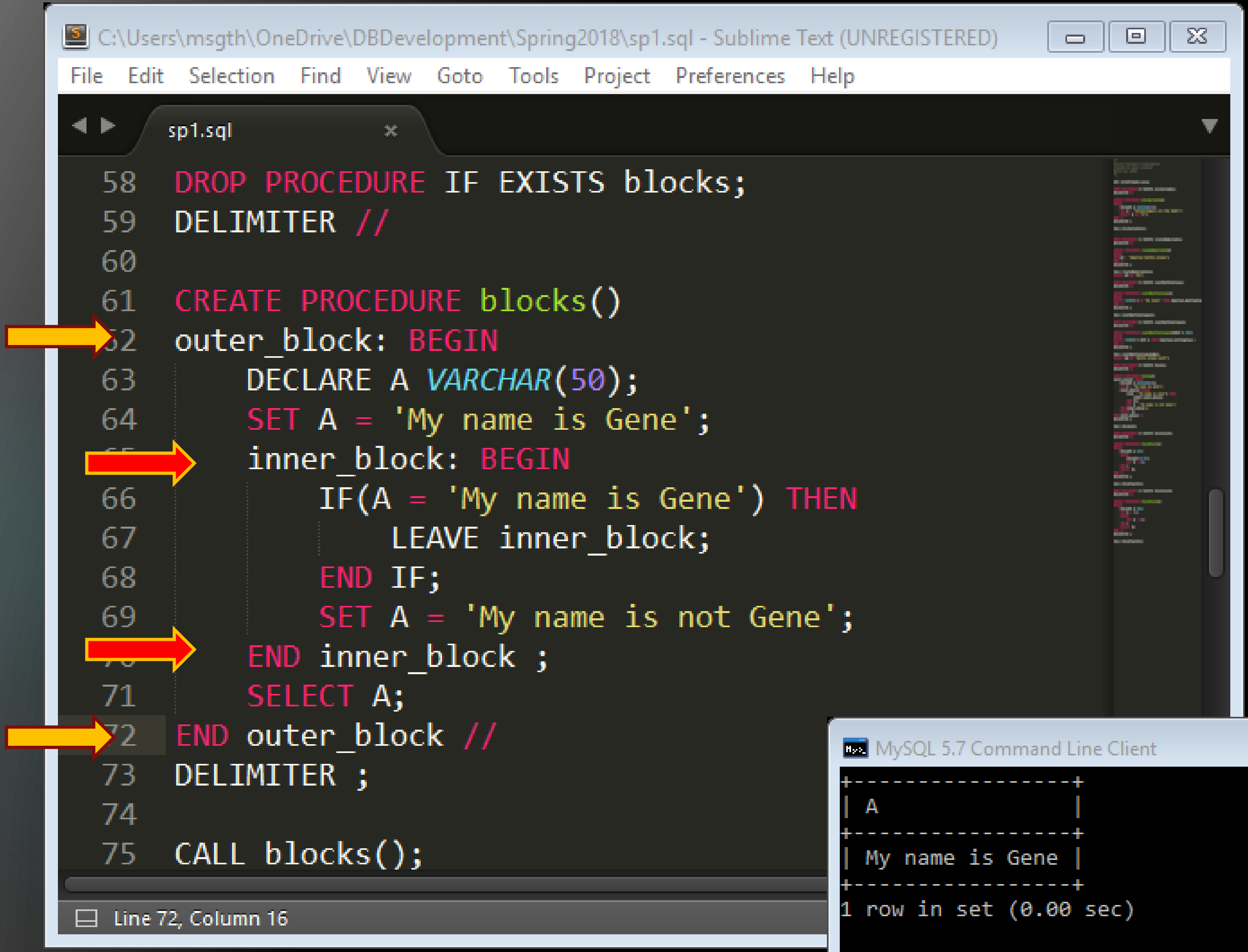
- MySQL will generate an error **if the order is not adhered** to in the stored procedure.
  - ***The error message will not indicate that this is the problem.***
- **Blocks have two purposes:**
  - Logically group related code segments.
  - Control the scope of variables and other objects.
    - Define a variable that is not visible outside the block.
    - Define a variable that overrides the definition with the same name outside the block.

Block Structure

- A Block can be labelled.
- The label can occur both before the **BEGIN** statement and after the **END** statement.
- **Labelling a Block can:**
  - Improve readability
  - Allow block execution to be terminated with a **LEAVE** statement.

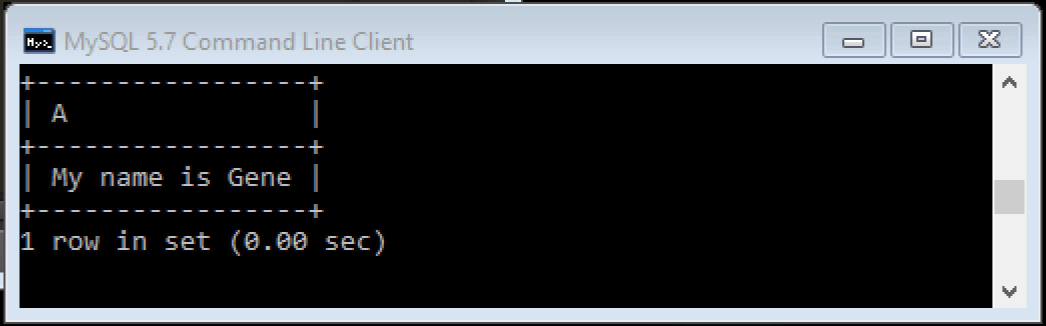
```
[label:] BEGIN
-- This is a Block
-- Declarations and Code
END [label] ;
```

Labelled Blocks (Scope)



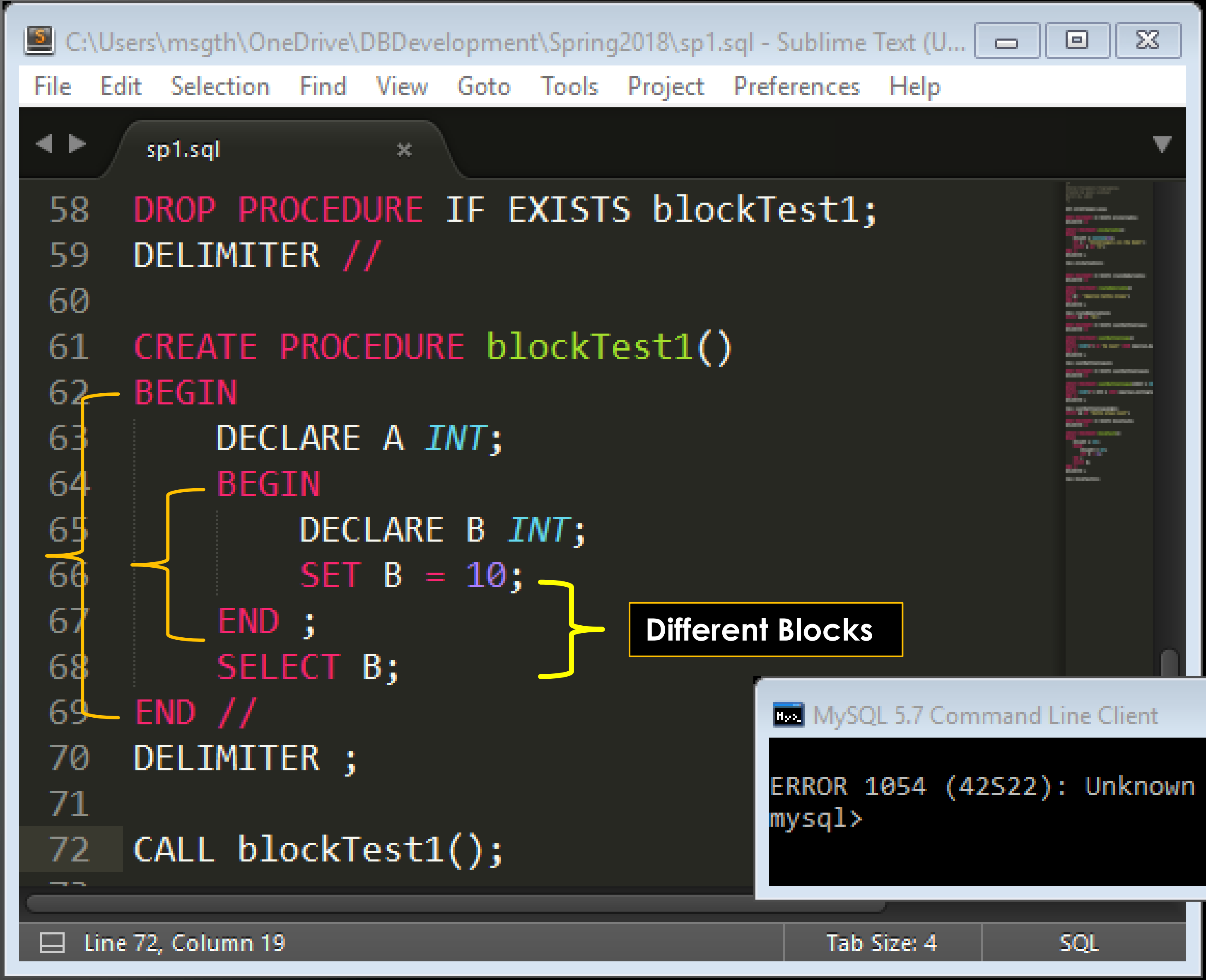
C:\Users\msgth\OneDrive\DBDevelopment\Spring2018\sp1.sql - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
sp1.sql

```sql
DROP PROCEDURE IF EXISTS blocks;
DELIMITER //

CREATE PROCEDURE blocks()
outer_block: BEGIN
    DECLARE A VARCHAR(50);
    SET A = 'My name is Gene';
    inner_block: BEGIN
        IF(A = 'My name is Gene') THEN
            LEAVE inner_block;
        END IF;
        SET A = 'My name is not Gene';
    END inner_block ;
    SELECT A;
END outer_block //
DELIMITER ;

CALL blocks();
```



```
+-----------------+
| A               |
+-----------------+
| My name is Gene |
+-----------------+
1 row in set (0.00 sec)
```

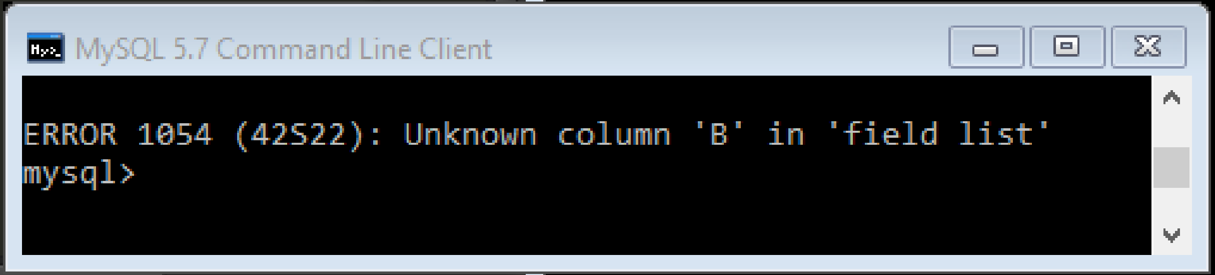
Nested Block Structure



C:\Users\msgth\OneDrive\DBDevelopment\Spring2018\sp1.sql - Sublime Text (U...
File Edit Selection Find View Goto Tools Project Preferences Help
sp1.sql
Line 72, Column 19
Tab Size: 4
SQL

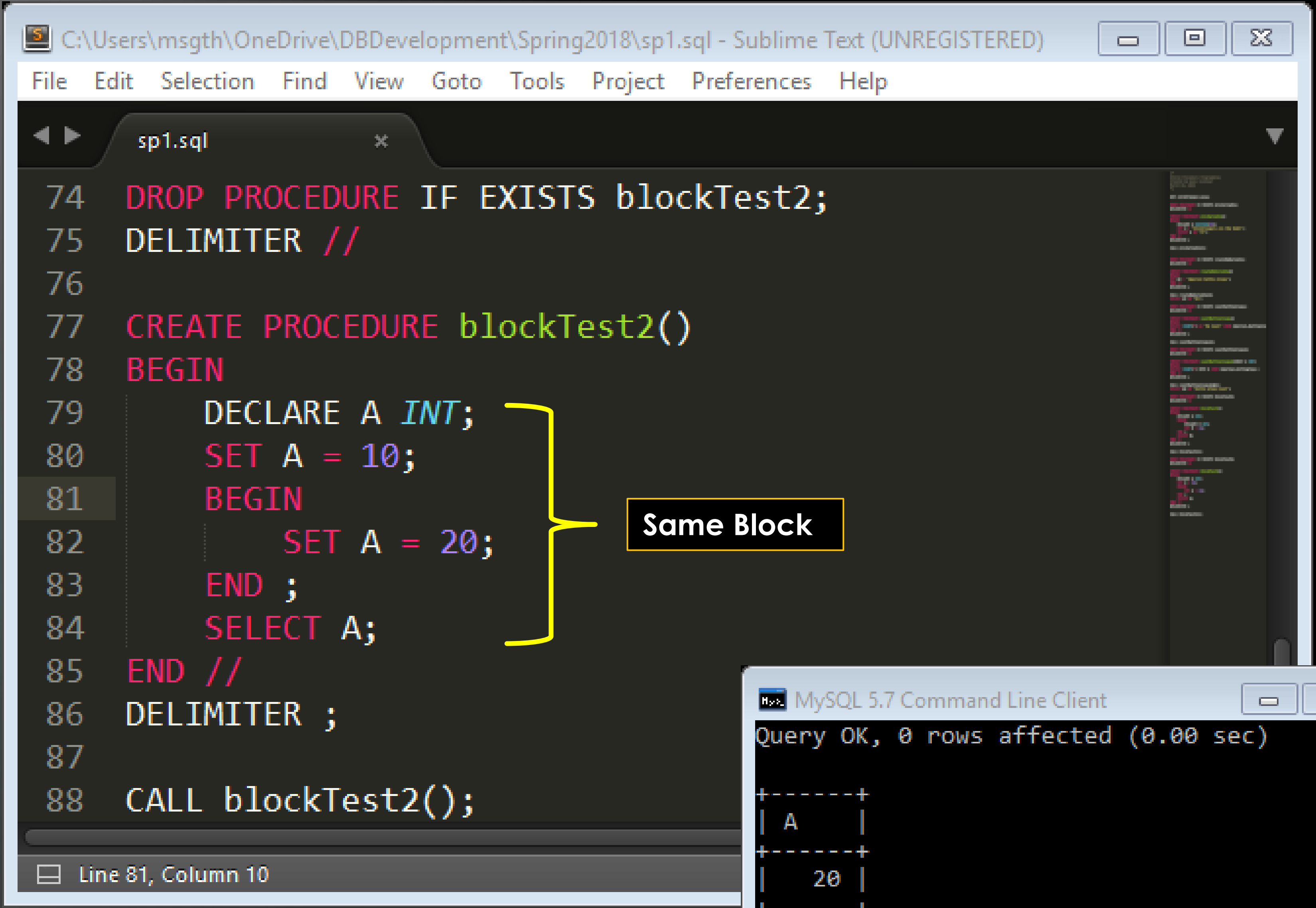
```sql
DROP PROCEDURE IF EXISTS blockTest1;
DELIMITER //

CREATE PROCEDURE blockTest1()
BEGIN
    DECLARE A INT;
    BEGIN
        DECLARE B INT;
        SET B = 10;
    END ;
    SELECT B;
END //
DELIMITER ;

CALL blockTest1();
```

Different Blocks



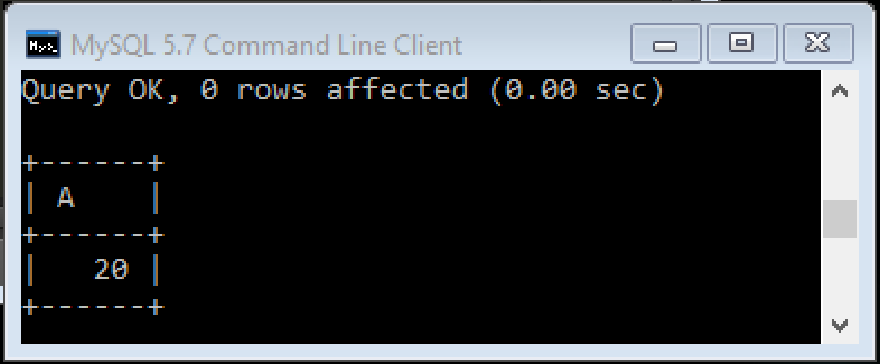
```
ERROR 1054 (42S22): Unknown column 'B' in 'field list'
mysql>
```

Nested Block Structure



C:\Users\msgth\OneDrive\DBDevelopment\Spring2018\sp1.sql - Sublime Text (UNREGISTERED)

```sql
DROP PROCEDURE IF EXISTS blockTest2;
DELIMITER //

CREATE PROCEDURE blockTest2()
BEGIN
    DECLARE A INT;
    SET A = 10;
    BEGIN
        SET A = 20;
    END ;
    SELECT A;
END //
DELIMITER ;

CALL blockTest2();
```



MySQL 5.7 Command Line Client

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

+------+
| A    |
+------+
|   20 |
+------+
```

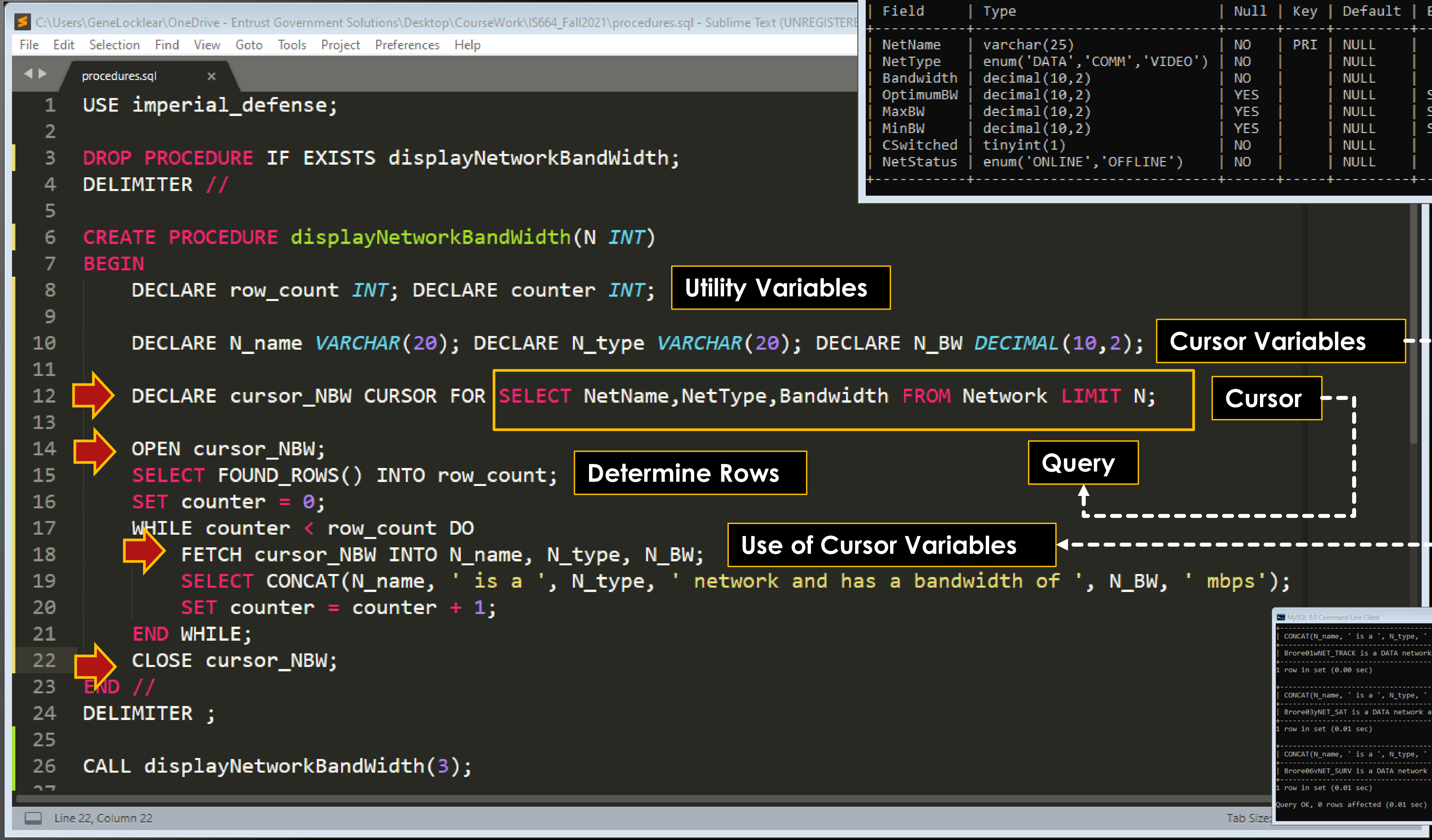
Use of Cursors

- To handle a SELECT statement that returns more than one row, we must create and manipulate a cursor.
- A **cursor** is an object that provide programmatic access to the result set returned by a SELECT statement.
- A cursor **is used to iterate through the rows in a result set** and take action for each row individually.
- MySQL supports cursors inside Stored Procedures.
- **https://dev.mysql.com/doc/refman/8.0/en/cursors.html**
- **Cursors have these properties:**
  - **Asensitive:** The server may or may not make a copy of its result set.
  - **Read Only:** Not updateable.
  - **Nonscrollable:** Can be traversed in only one direction and cannot skip rows.
- Cursor declaration **must appear** before handler declarations and after variable and condition declarations.

Use of Cursors

- The MySQL stored program language supports three statements for performing cursor operations:
- **OPEN**
    - Initialize the result set for the cursor.
    - **OPEN *[cursor name]***
- **FETCH**
    - Retrieves the next row from the cursor and moves the cursor to the following row in the result set.
    - **FETCH *[cursor name]* INTO *[variable list]***
    - The **variable list must contain one variable for each column** returned by the SELECT statement contained in the cursor declaration.
- **CLOSE**
    - Deactivates the cursor and releases memory associated with that cursor.
    - **CLOSE *[cursor name]***

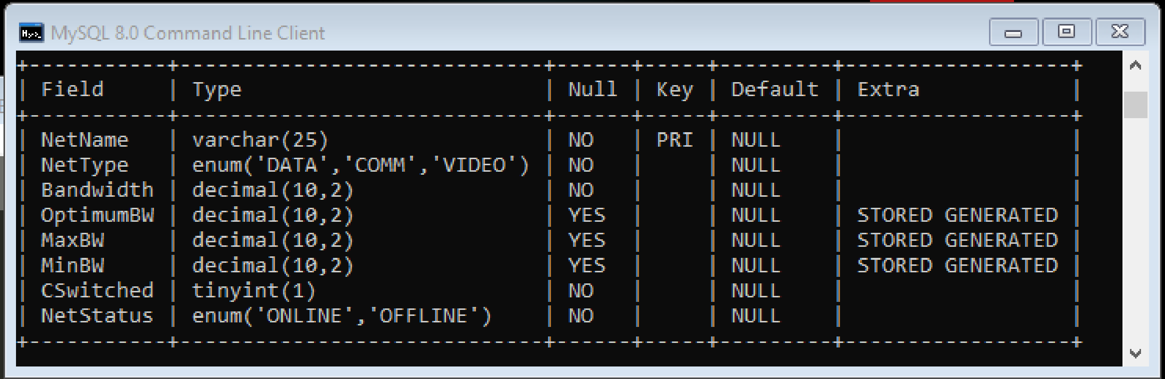
Use of Cursors



C:\Users\GeneLocklear\OneDrive - Entrust Government Solutions\Desktop\CourseWork\IS664_Fall2021\procedures.sql - Sublime Text (UNREGISTERE

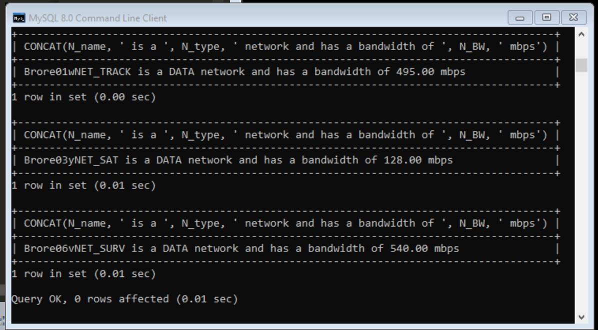
```sql
USE imperial_defense;

DROP PROCEDURE IF EXISTS displayNetworkBandWidth;
DELIMITER //

CREATE PROCEDURE displayNetworkBandWidth(N INT)
BEGIN
    DECLARE row_count INT; DECLARE counter INT;

    DECLARE N_name VARCHAR(20); DECLARE N_type VARCHAR(20); DECLARE N_BW DECIMAL(10,2);

    DECLARE cursor_NBW CURSOR FOR SELECT NetName,NetType,Bandwidth FROM Network LIMIT N;

    OPEN cursor_NBW;
    SELECT FOUND_ROWS() INTO row_count;
    SET counter = 0;
    WHILE counter < row_count DO
        FETCH cursor_NBW INTO N_name, N_type, N_BW;
        SELECT CONCAT(N_name, ' is a ', N_type, ' network and has a bandwidth of ', N_BW, ' mbps');
        SET counter = counter + 1;
    END WHILE;
    CLOSE cursor_NBW;
END //
DELIMITER ;

CALL displayNetworkBandWidth(3);
```

Line 22, Column 22



MySQL 8.0 Command Line Client

```
+-----------+-----------------------------------+------+-----+---------+------------------+
| Field     | Type                              | Null | Key | Default | Extra            |
+-----------+-----------------------------------+------+-----+---------+------------------+
| NetName   | varchar(25)                       | NO   | PRI | NULL    |                  |
| NetType   | enum('DATA','COMM','VIDEO')       | NO   |     | NULL    |                  |
| Bandwidth | decimal(10,2)                     | NO   |     | NULL    |                  |
| OptimumBW | decimal(10,2)                     | YES  |     | NULL    | STORED GENERATED |
| MaxBW     | decimal(10,2)                     | YES  |     | NULL    | STORED GENERATED |
| MinBW     | decimal(10,2)                     | YES  |     | NULL    | STORED GENERATED |
| CSwitched | tinyint(1)                        | NO   |     | NULL    |                  |
| NetStatus | enum('ONLINE','OFFLINE')          | NO   |     | NULL    |                  |
+-----------+-----------------------------------+------+-----+---------+------------------+
```



MySQL 8.0 Command Line Client

```
+------------------------------------------------------------------------------+
| CONCAT(N_name, ' is a ', N_type, ' network and has a bandwidth of ', N_BW, ' mbps') |
+------------------------------------------------------------------------------+
| Brore01wNET_TRACK is a DATA network and has a bandwidth of 495.00 mbps       |
+------------------------------------------------------------------------------+
1 row in set (0.00 sec)

+------------------------------------------------------------------------------+
| CONCAT(N_name, ' is a ', N_type, ' network and has a bandwidth of ', N_BW, ' mbps') |
+------------------------------------------------------------------------------+
| Brore03yNET_SAT is a DATA network and has a bandwidth of 128.00 mbps         |
+------------------------------------------------------------------------------+
1 row in set (0.01 sec)

+------------------------------------------------------------------------------+
| CONCAT(N_name, ' is a ', N_type, ' network and has a bandwidth of ', N_BW, ' mbps') |
+------------------------------------------------------------------------------+
| Brore06vNET_SURV is a DATA network and has a bandwidth of 540.00 mbps        |
+------------------------------------------------------------------------------+
1 row in set (0.01 sec)

Query OK, 0 rows affected (0.01 sec)
```

Use of Handlers

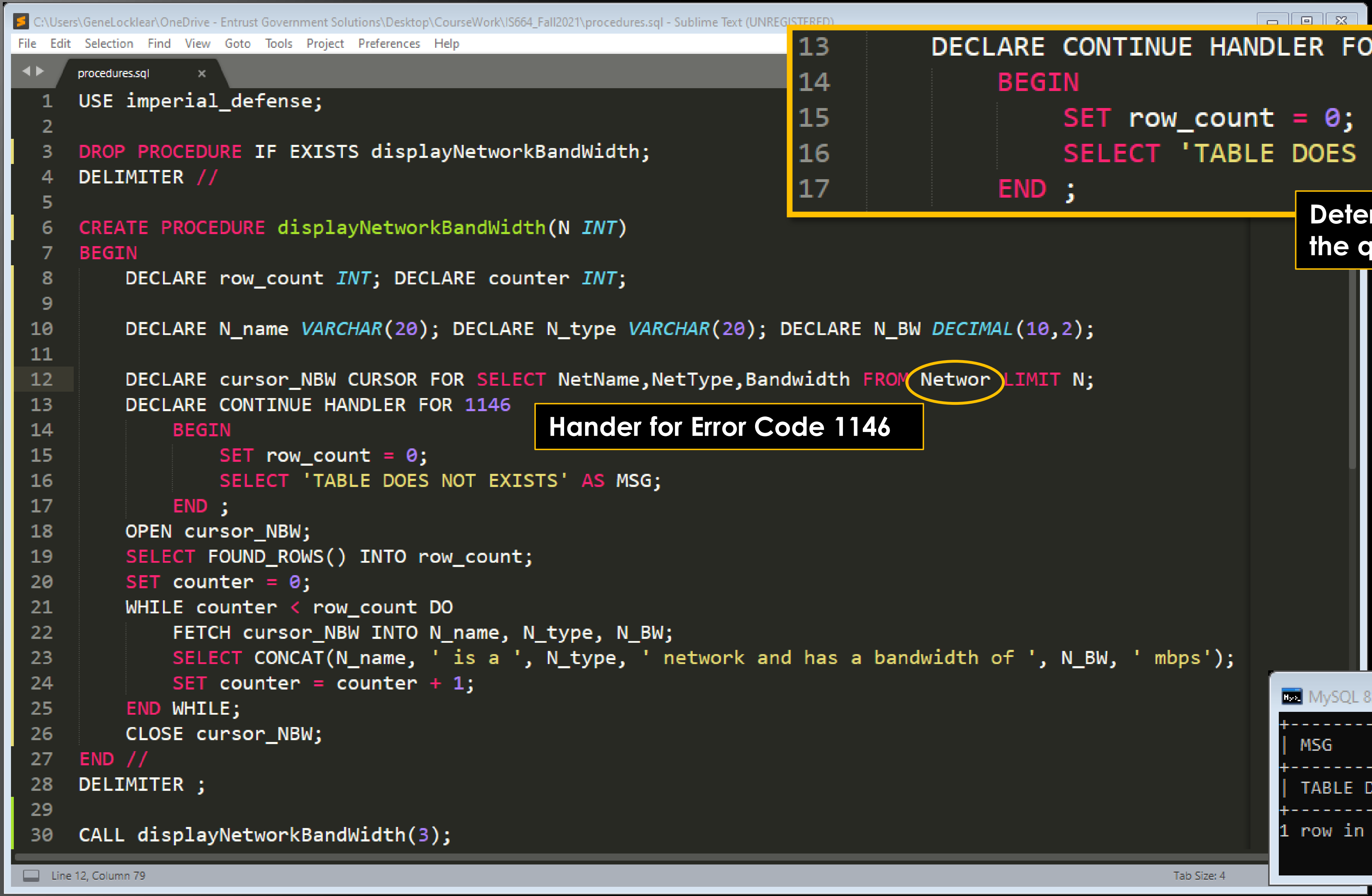
- A stored procedure may include **handlers** to be invoked **when certain conditions occur within the program**.
- Condition are such things as **SQLSTATE, SQLWARNING, NOT FOUND OR SQLEXCEPTION**
- A handler's action may be to continue or exit the procedure.
- **https://dev.mysql.com/doc/refman/8.0/en/handler-scope.html**

- The applicability of each handler depends on its location within the program definition and on the condition or conditions that it handles:

- A handler declared in a **BEGIN ... END** block is in scope only for the SQL statements following the handler declarations in the block.

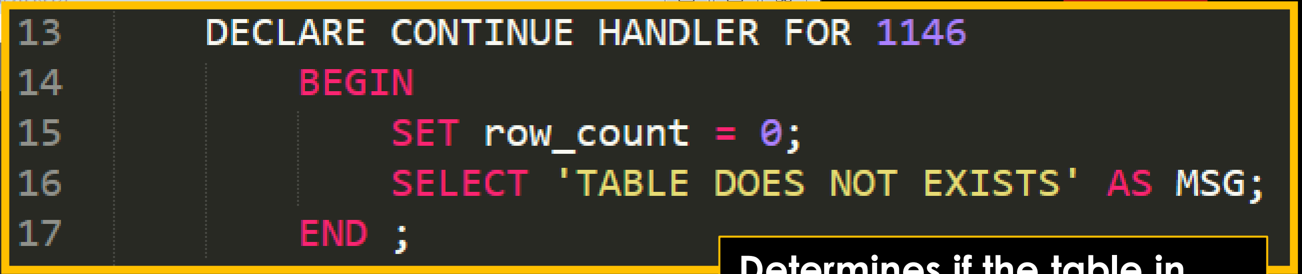
- If the handler itself raises a condition, **it cannot handle that condition**, nor can any other handlers that have been declared in the block.
- A handler is **in scope only for the block in which it is declared** and cannot be activated for conditions occurring outside that block.
- Multiple handlers **can be declared in different scopes** and with different specificities.

Use of Handler


C:\Users\GeneLocklear\OneDrive - Entrust Government Solutions\Desktop\CourseWork\IS664_Fall2021\procedures.sql - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
procedures.sql
Line 12, Column 79
Tab Size: 4

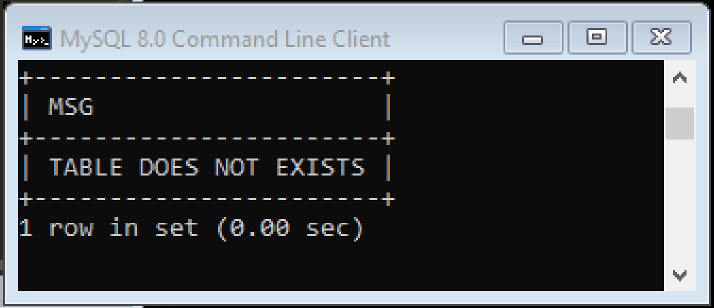
```
USE imperial_defense;

DROP PROCEDURE IF EXISTS displayNetworkBandWidth;
DELIMITER //

CREATE PROCEDURE displayNetworkBandWidth(N INT)
BEGIN
    DECLARE row_count INT; DECLARE counter INT;

    DECLARE N_name VARCHAR(20); DECLARE N_type VARCHAR(20); DECLARE N_BW DECIMAL(10,2);

    DECLARE cursor_NBW CURSOR FOR SELECT NetName,NetType,Bandwidth FROM Networ LIMIT N;
    DECLARE CONTINUE HANDLER FOR 1146
        BEGIN
            SET row_count = 0;
            SELECT 'TABLE DOES NOT EXISTS' AS MSG;
        END ;
    OPEN cursor_NBW;
    SELECT FOUND_ROWS() INTO row_count;
    SET counter = 0;
    WHILE counter < row_count DO
        FETCH cursor_NBW INTO N_name, N_type, N_BW;
        SELECT CONCAT(N_name, ' is a ', N_type, ' network and has a bandwidth of ', N_BW, ' mbps');
        SET counter = counter + 1;
    END WHILE;
    CLOSE cursor_NBW;
END //
DELIMITER ;

CALL displayNetworkBandWidth(3);
```

Hander for Error Code 1146



```
13        DECLARE CONTINUE HANDLER FOR 1146
14            BEGIN
15                SET row_count = 0;
16                SELECT 'TABLE DOES NOT EXISTS' AS MSG;
17            END ;
```

Determines if the table in the query exists.



```
+-----------------------+
| MSG                   |
+-----------------------+
| TABLE DOES NOT EXISTS |
+-----------------------+
1 row in set (0.00 sec)
```