Lewis University  
CPSC 50900: Database Systems   
Spring 2023 Term Project

TITLE **"Inventory control management system"**

Siva Naga Raju Parimisetty, sivanparimisetty@lewisu.edu

Mohammed Asif Nawaz, mohammedasifnawaz@lewisu.edu

Balaramakrishna Yadav Keerthi, balaramakrishnayad@lewisu.edu

Work products stored in the Github repository

Table of Contents

[Initial Proposal 2](#_gjdgxs)

[Data Sources 2](#_30j0zll)

[Data Storage Alternatives 2](#_2et92p0)

[Relational Database Design Process 3](#_3dy6vkm)

[Relational Database Design 3](#_4d34og8)

[Data Definition Language (DDL) Scripts 3](#_2s8eyo1)

[Data Manipulation Language Scripts 4](#_17dp8vu)

[Indexes 5](#_3rdcrjn)

[Views 5](#_26in1rg)

[Triggers 5](#_lnxbz9)

[Transactions 5](#_35nkun2)

[Database Security 6](#_1ksv4uv)

[Locking and Concurrent Access 6](#_44sinio)

[Backing Up Your Database 6](#_2jxsxqh)

[Python Programming 7](#_z337ya)

[PHP Programming 7](#_4i7ojhp)

[Suggested Future Work 8](#_3j2qqm3)

[Activity Log 8](#_1y810tw)

# Initial Proposal

**Initial proposal**

We are going to develop an inventory control management system for our group project. We plan to build an effective inventory control management system web application for this project. It will help us better manage our inventory levels and streamline ordering and fulfillment processes. We can build a system that will lead to a faster inventory tracking system and purchase order management. The system of sales order management and customer relationship management also can be improved with this inventory control management system in an effective way. This system will provide detailed reporting and analytics capabilities, including inventory levels, order history, and sales trends. We utilize proper data sources for our project. The alternative ways to store the data also will be explored in this project. The timeline of this project is one month with a budget of $100,000. We believe that implementing an inventory control management system will help us reduce inventory costs, improve order fulfillment, and provide better customer service.

# Data Sources

**Data sources**

Data sources for an inventory control management system are the various types of data that the system will use to manage inventory levels, ordering, and fulfillment processes. These data sources include inventory levels, purchase orders, sales orders, and customer information. Inventory levels data is used to track and manage stock levels and replenish stock when necessary. Purchase orders and sales orders data is used to manage the ordering process, including tracking deliveries and reconciling invoices. Customer information data is used to manage customer relationships, track customer orders and preferences, and generate customer reports. Accurate and timely data from these sources is critical for effective inventory management, cost control, and customer service. The inventory control management system should be designed to integrate and manage these data sources effectively.

# Data Storage Alternatives

**Alternative ways to store the data.**

There are several alternative ways to store data for an inventory control management system. One option is to use cloud-based storage, which allows for easy access to the system from anywhere, and provides scalability and flexibility. Another option is to use a distributed database system, which can offer better performance and reliability. This involves storing data across multiple physical locations, reducing the risk of data loss or downtime. We may also use a hybrid approach, combining both cloud-based and distributed database storage to take advantage of the benefits of both approaches. Careful consideration of these alternatives can result in an efficient and effective inventory control management system.

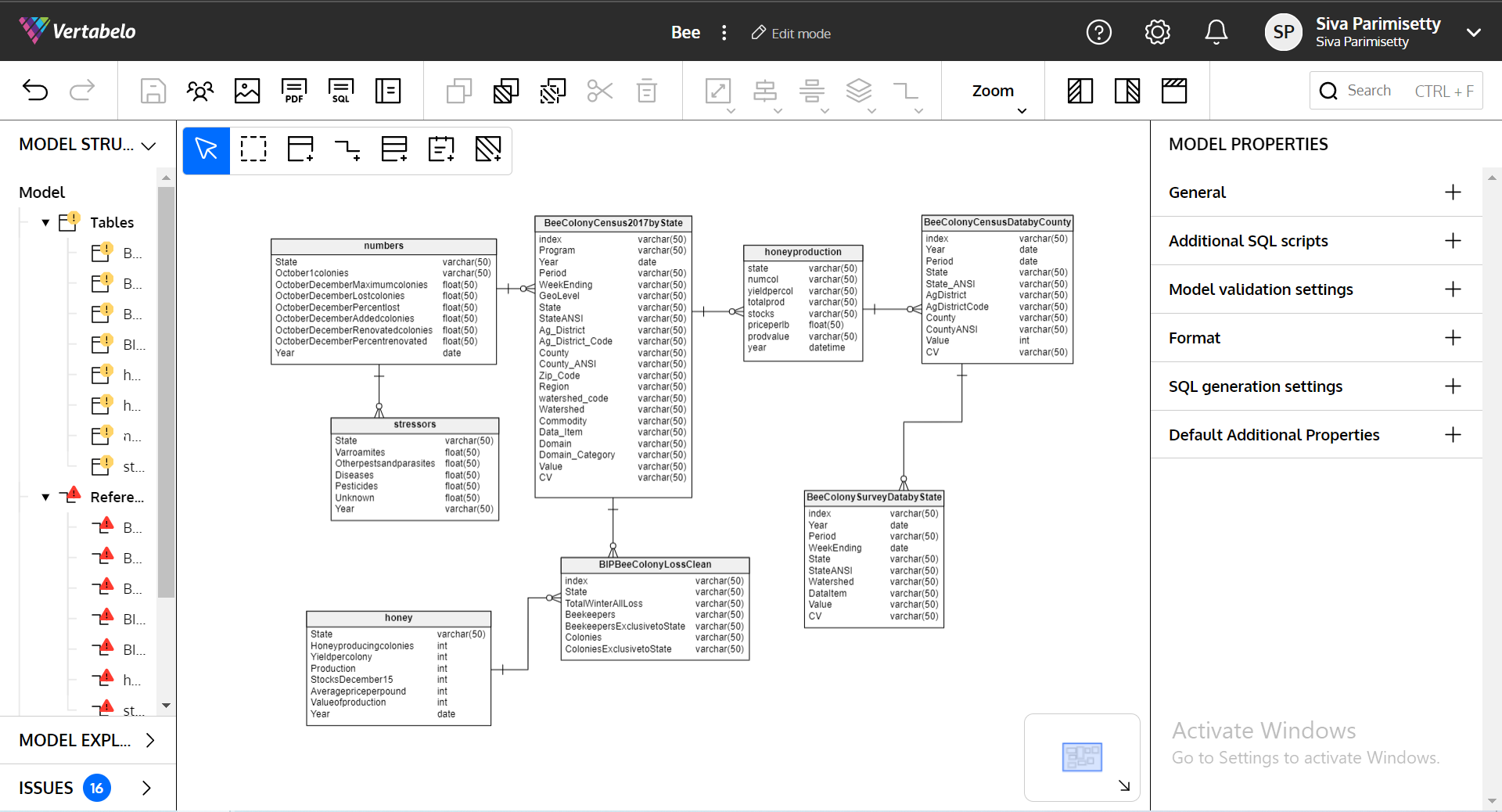
# Relational Database Design Process

It can be seen from the list of fields that I have identified in part c that the details of Beekeepers are given. A functional dependency exists among these data values of beekeepers. The index number is the key identity of all the rows in the table. Similarly, the State, which shows the location, can be linked with the number of beekeepers and how much loss has happened throughout the winter. Without “Year”, it would not be easy for me to justify any of the statistics related to the Beekeeper.

It can be clearly seen that a total of 8 tables are making relationships with each other. Table “Bee Colony Census 2017 by State” establishes a one-to-many relationship with table “Bee Colony Census Data by Country” and many-to-one relationship with table “numbers”. Further, “number” table makes a one-to-many relationship with the table “stressor”. Likewise, the one-to-many relationships between the tables have been drawn in a repeated manner where the table on the side of one can be considered as the parent table and child table lies on the side of many.

The Bee colonies also consist of spares and tools where inventory management is required to deal with various functionalities.

# Relational Database Design

****

# Data Definition Language (DDL) Scripts

Database structures are defined and managed using scripts written in the Data Definition Language (DDL). When working with DDL scripts, bear the following factors in mind:

Database objects like tables, views, indexes, and constraints can all be created, modified, and deleted using DDL scripts.

The standard language for handling relational databases, SQL (Structured Query Language), is used to create DDL scripts.

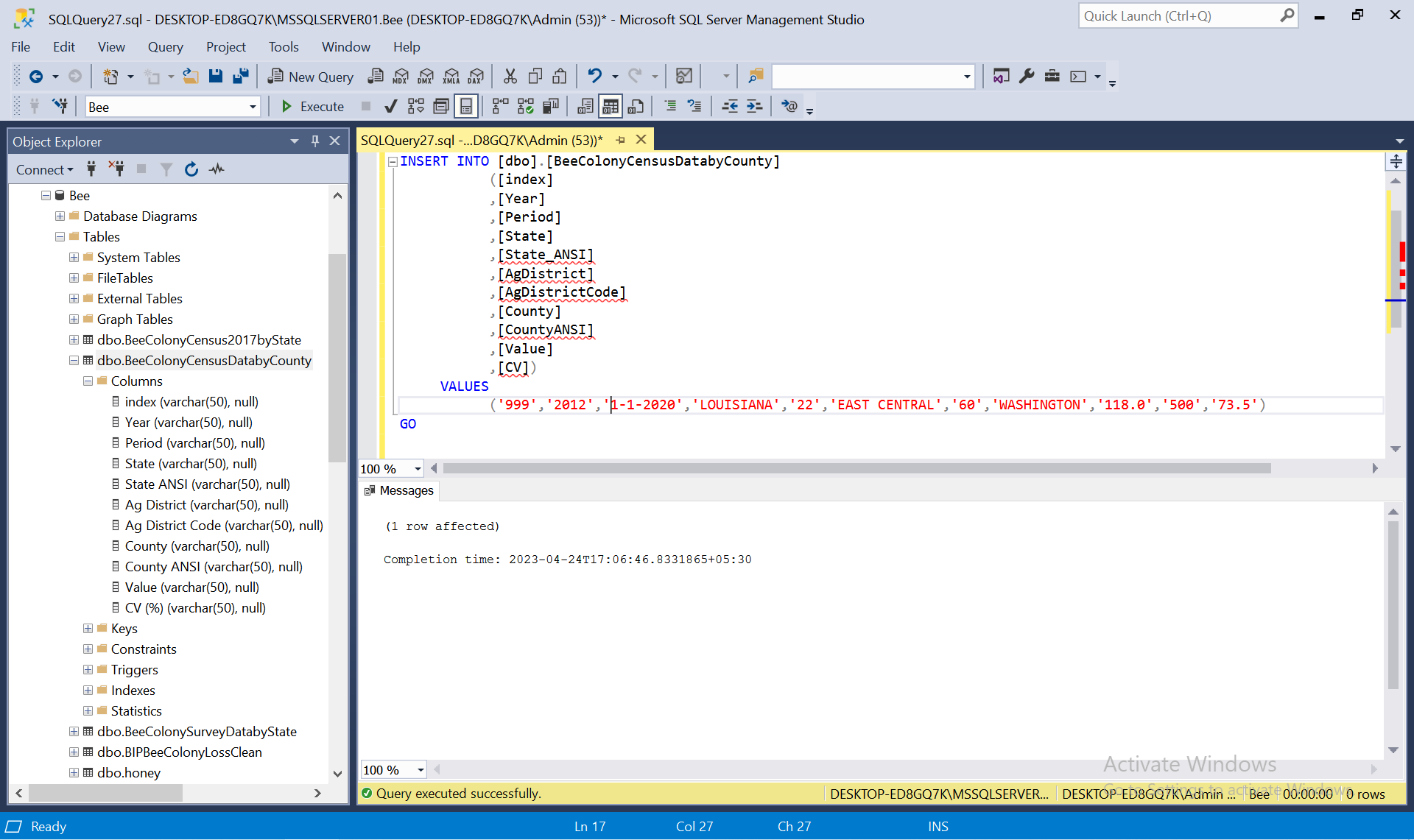
DDL scripts can be run through a command-line interface or a database administration tool.

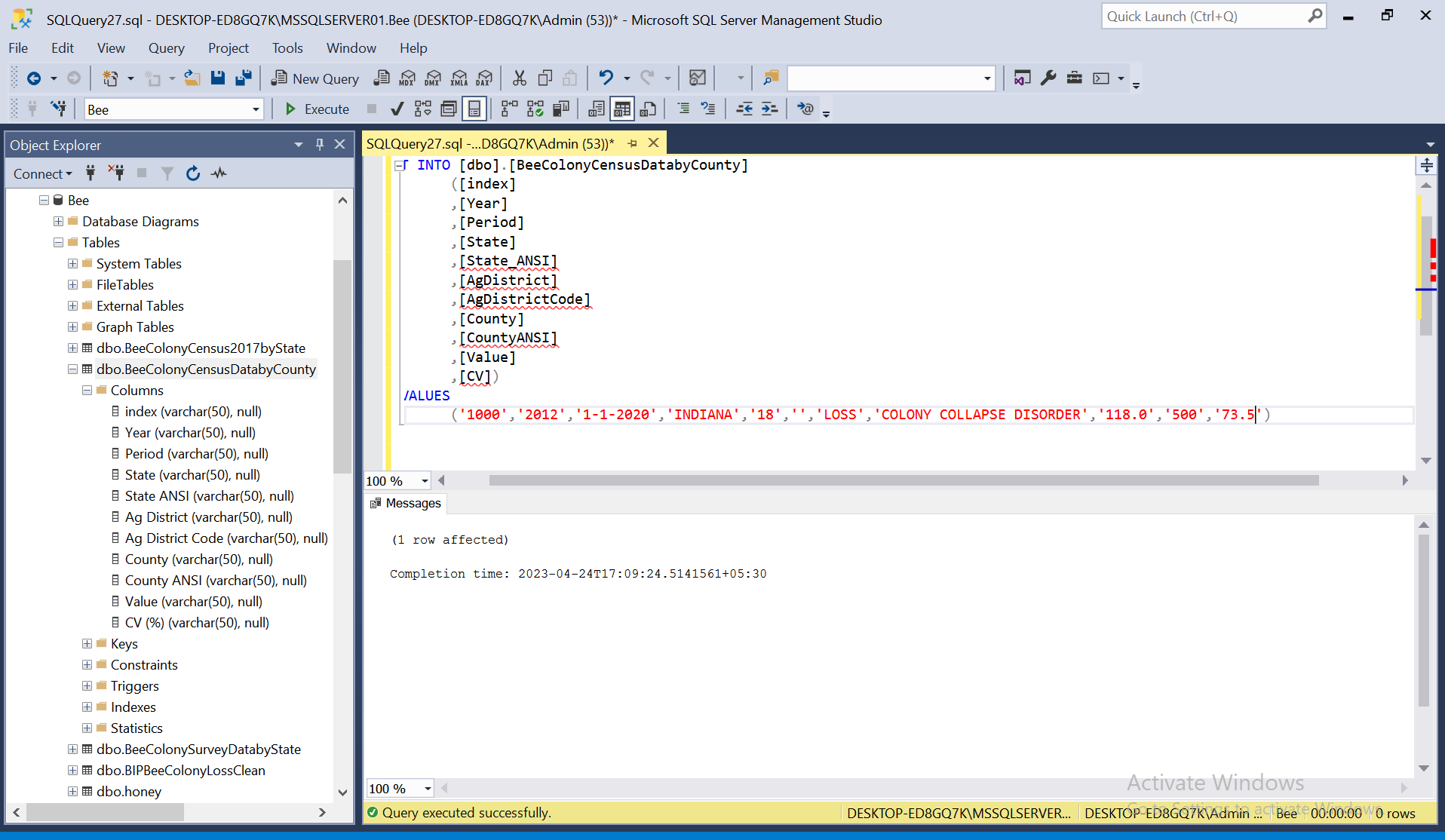
Before running, DDL scripts should be carefully written and tested because they might significantly affect the database's structure and data integrity.

To ensure proper database management and maintenance over time, DDL scripts should be version-controlled and well-documented.

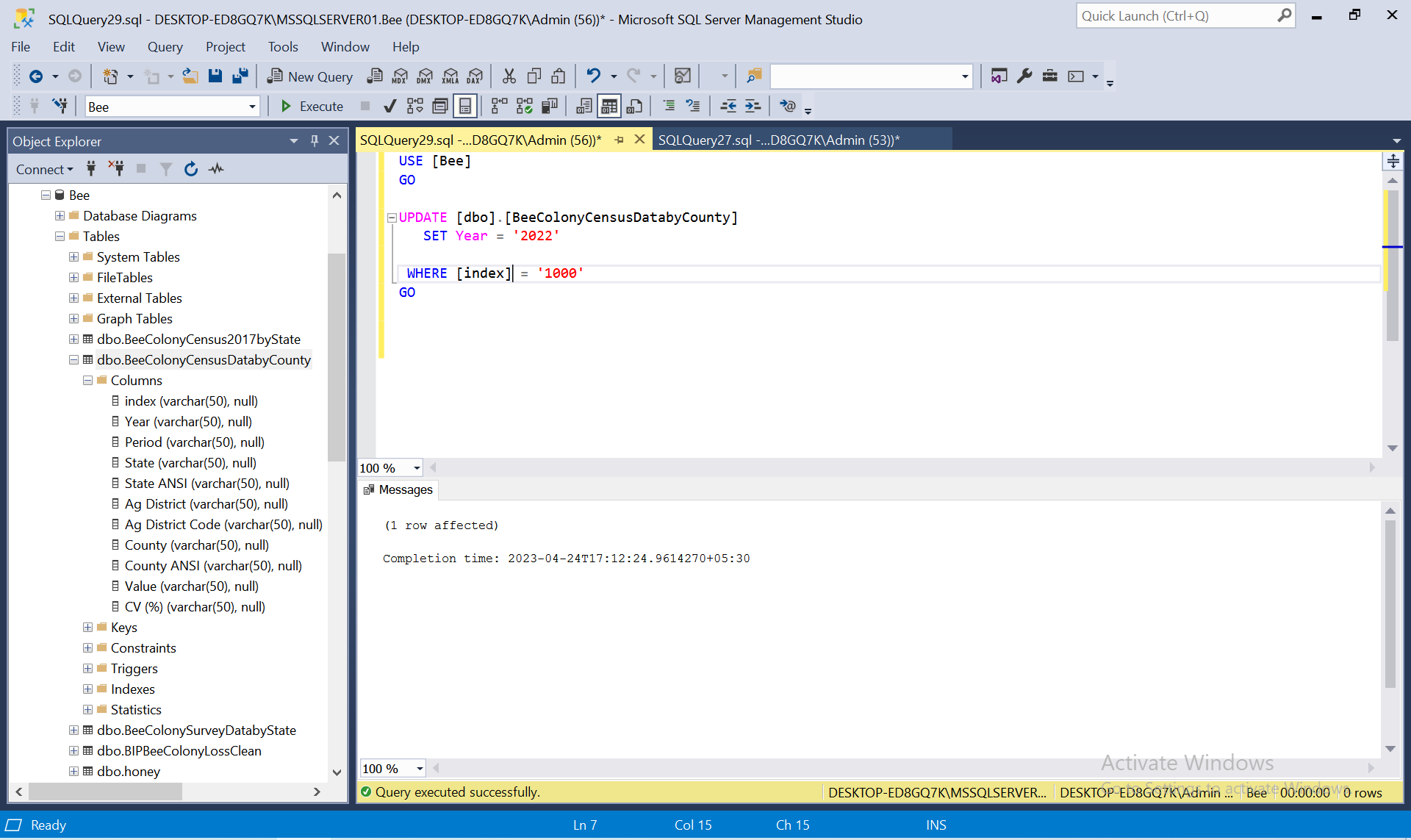
# Data Manipulation Language Scripts

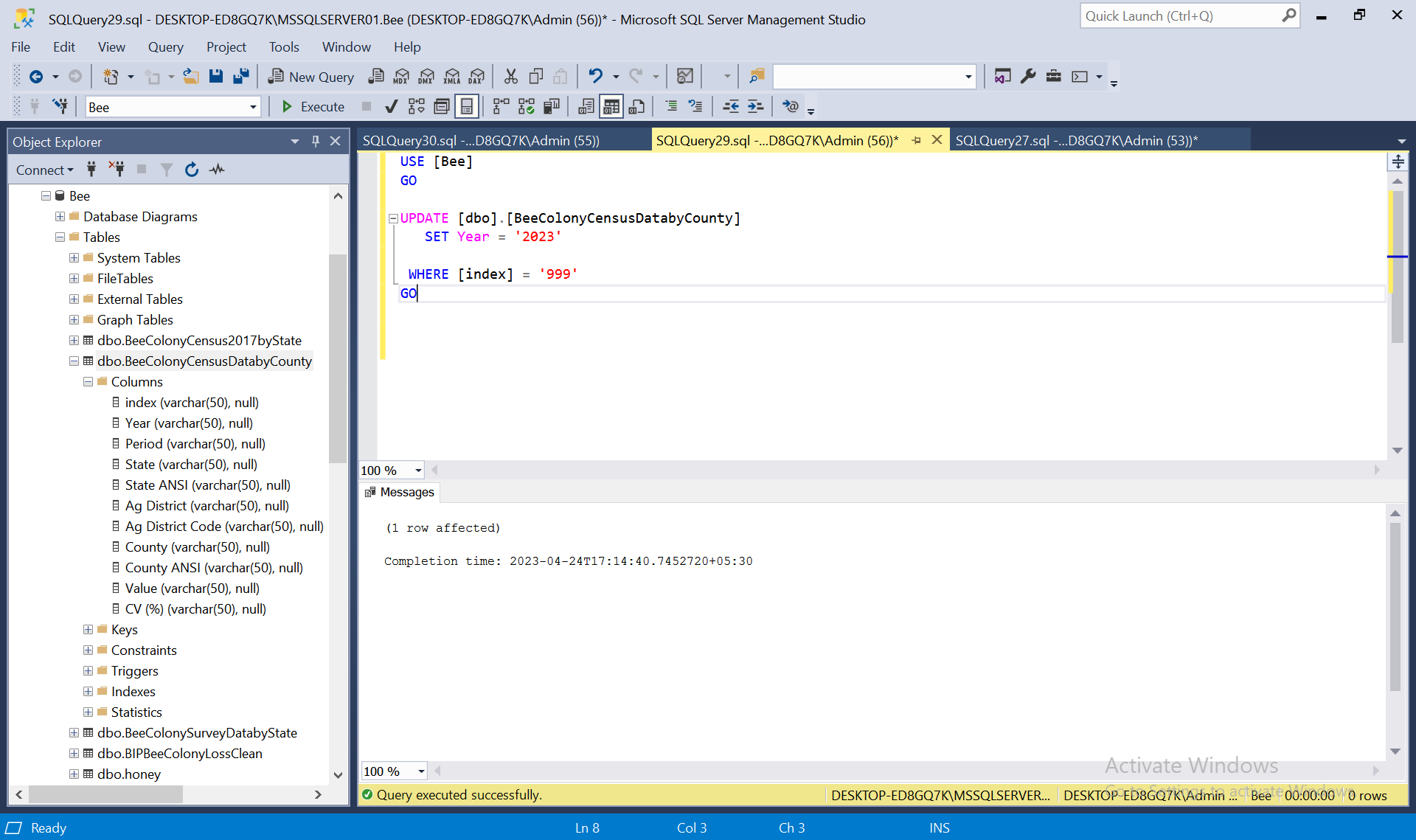
**Insert statements**

****

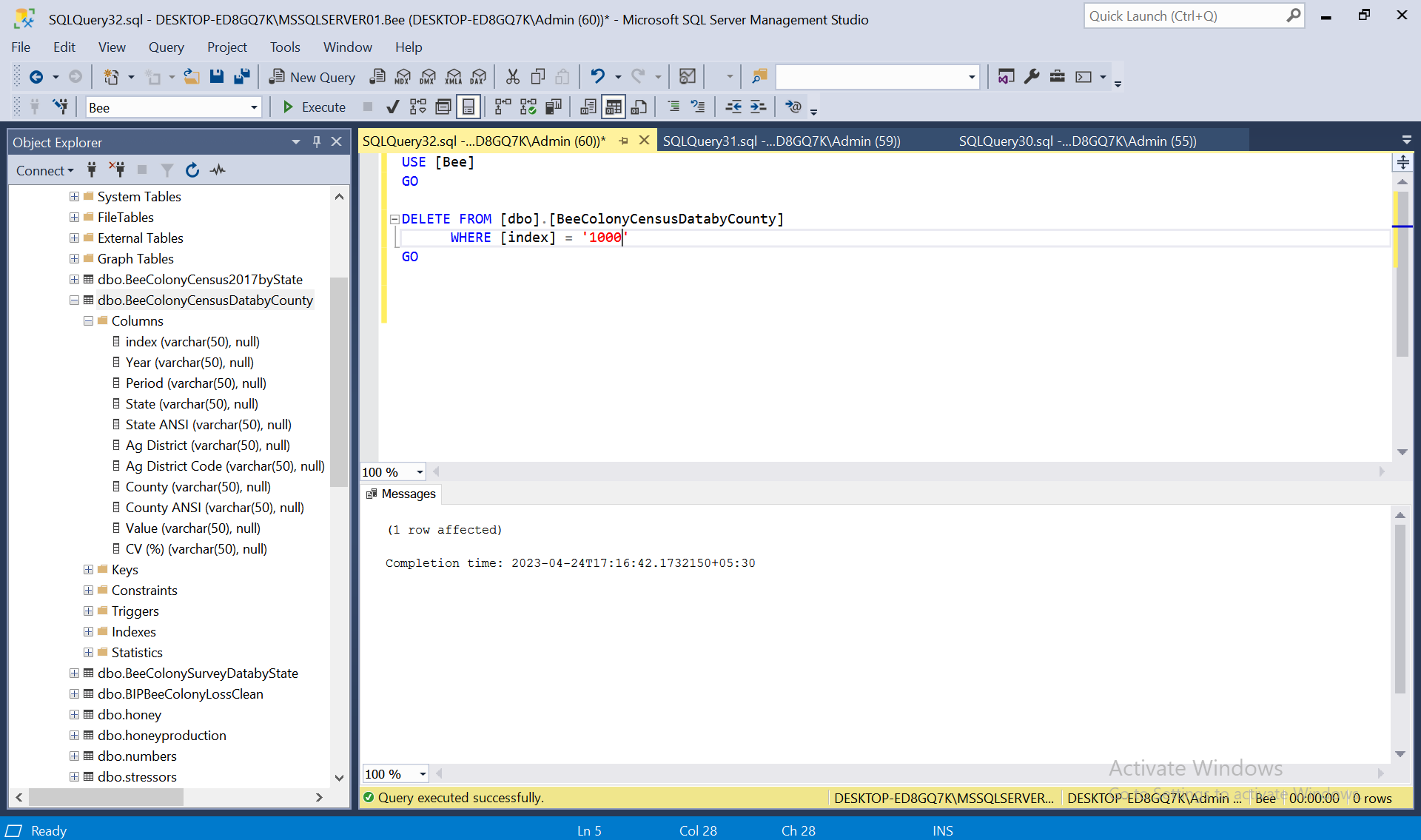
****

**update statements**

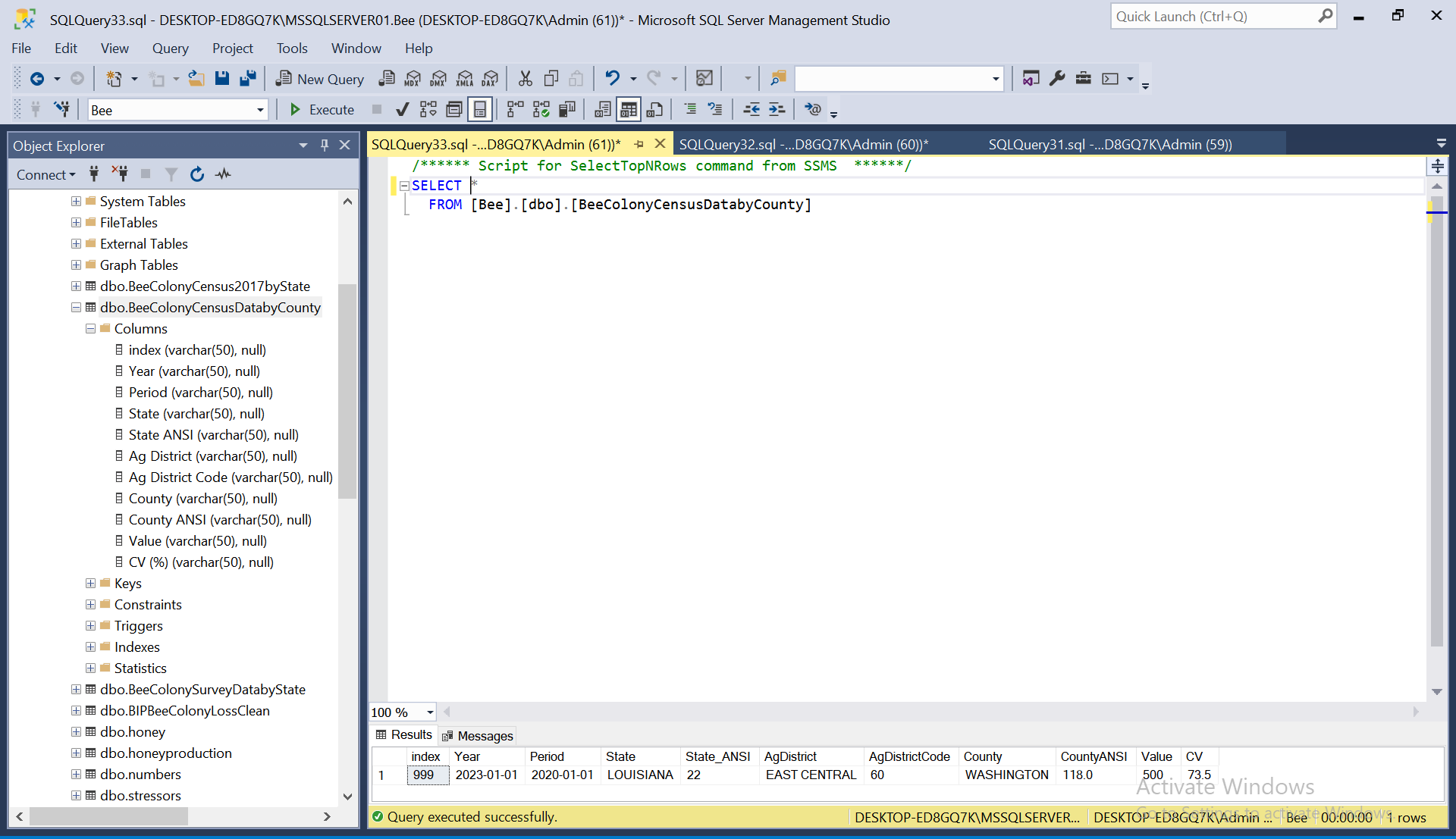
****

****

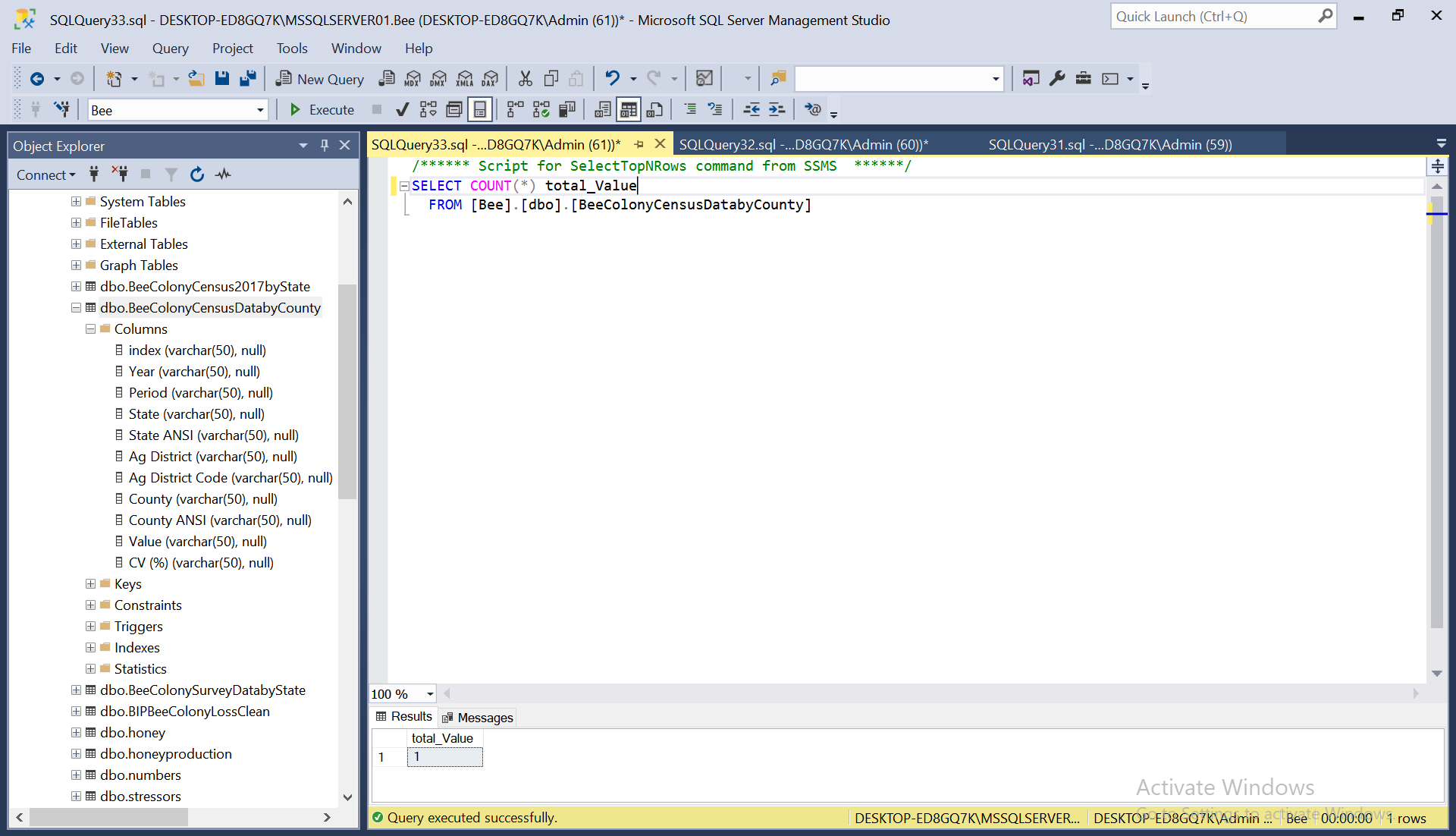
**Delete statement**

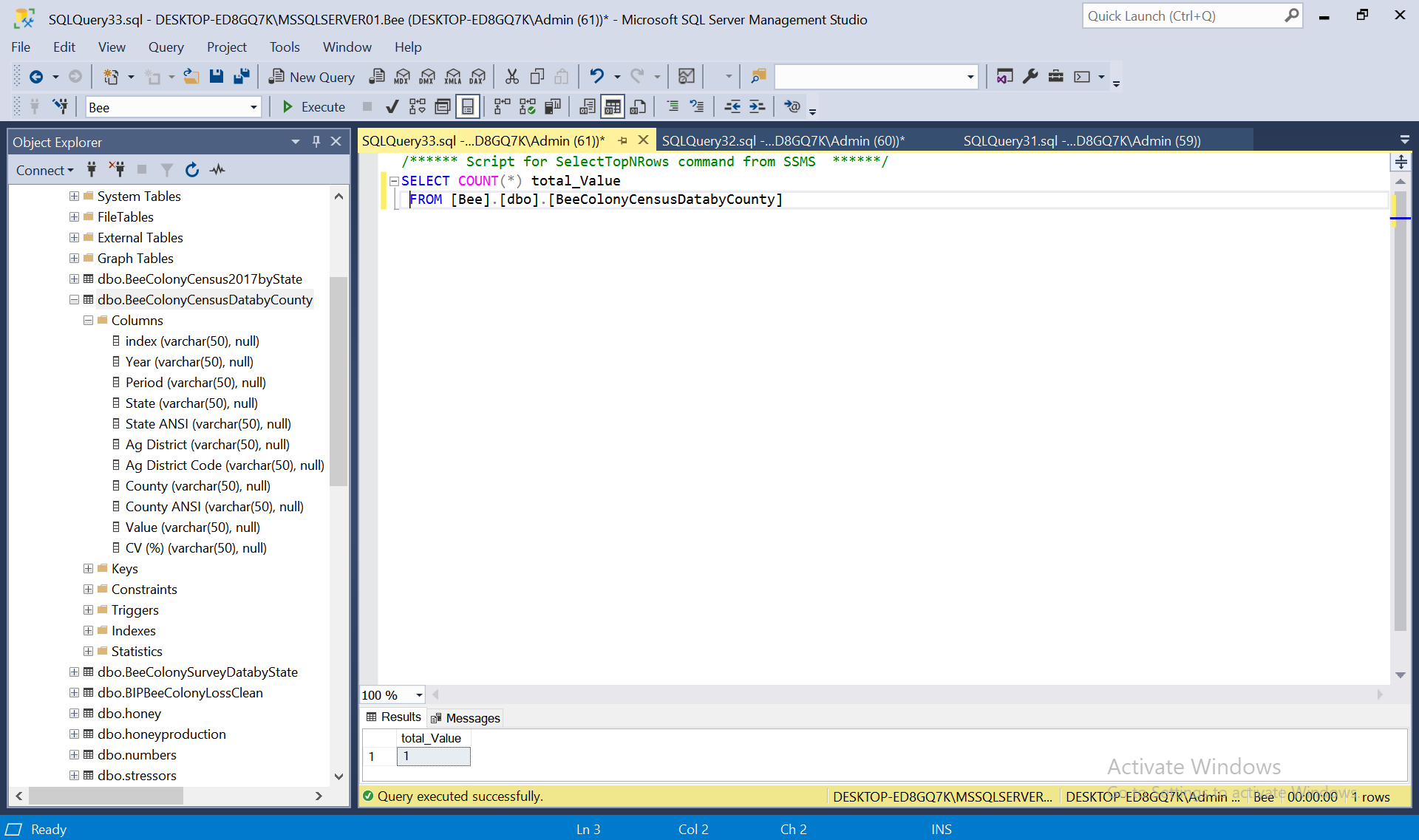
****

**simple select statement**

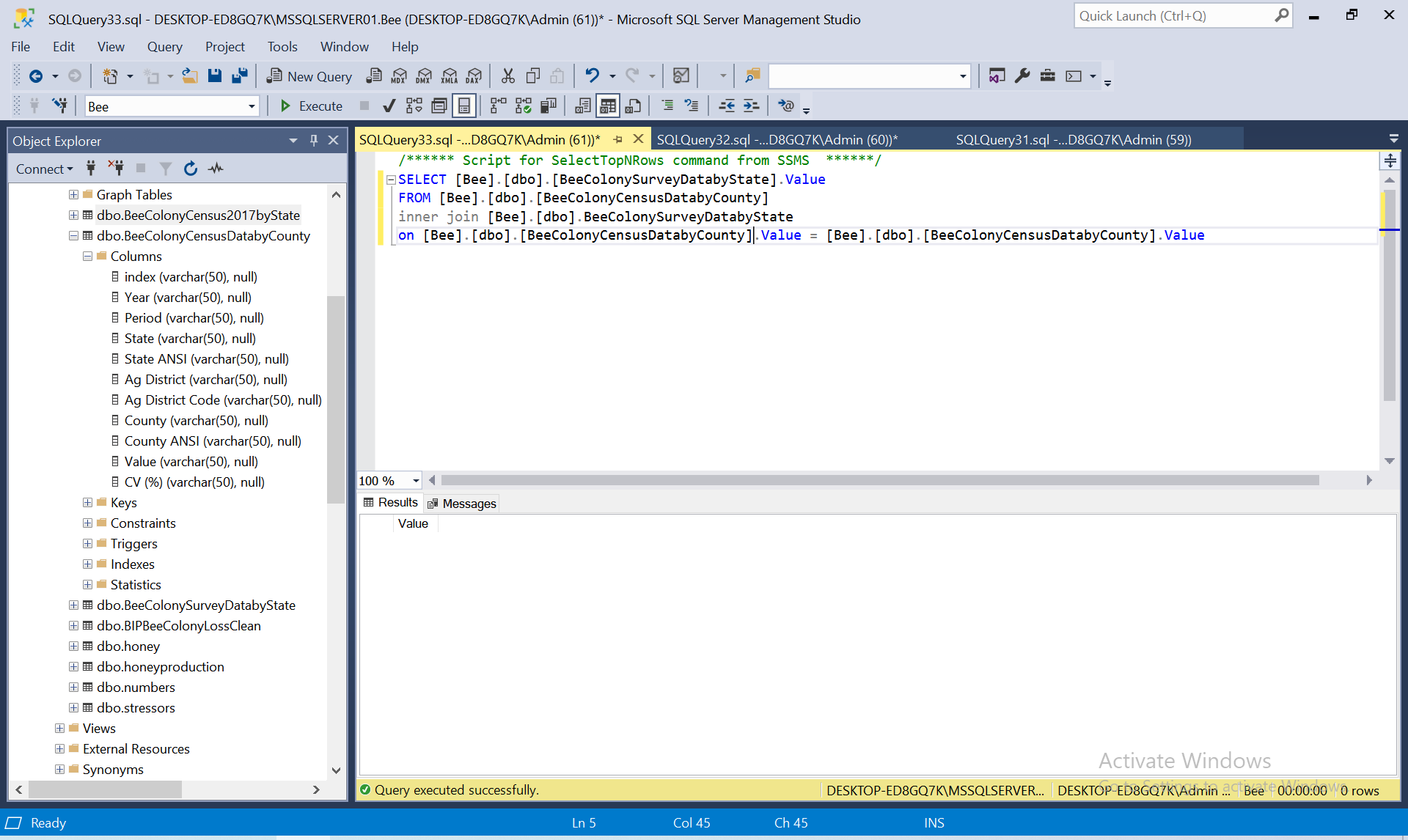
****

**summary statements:**

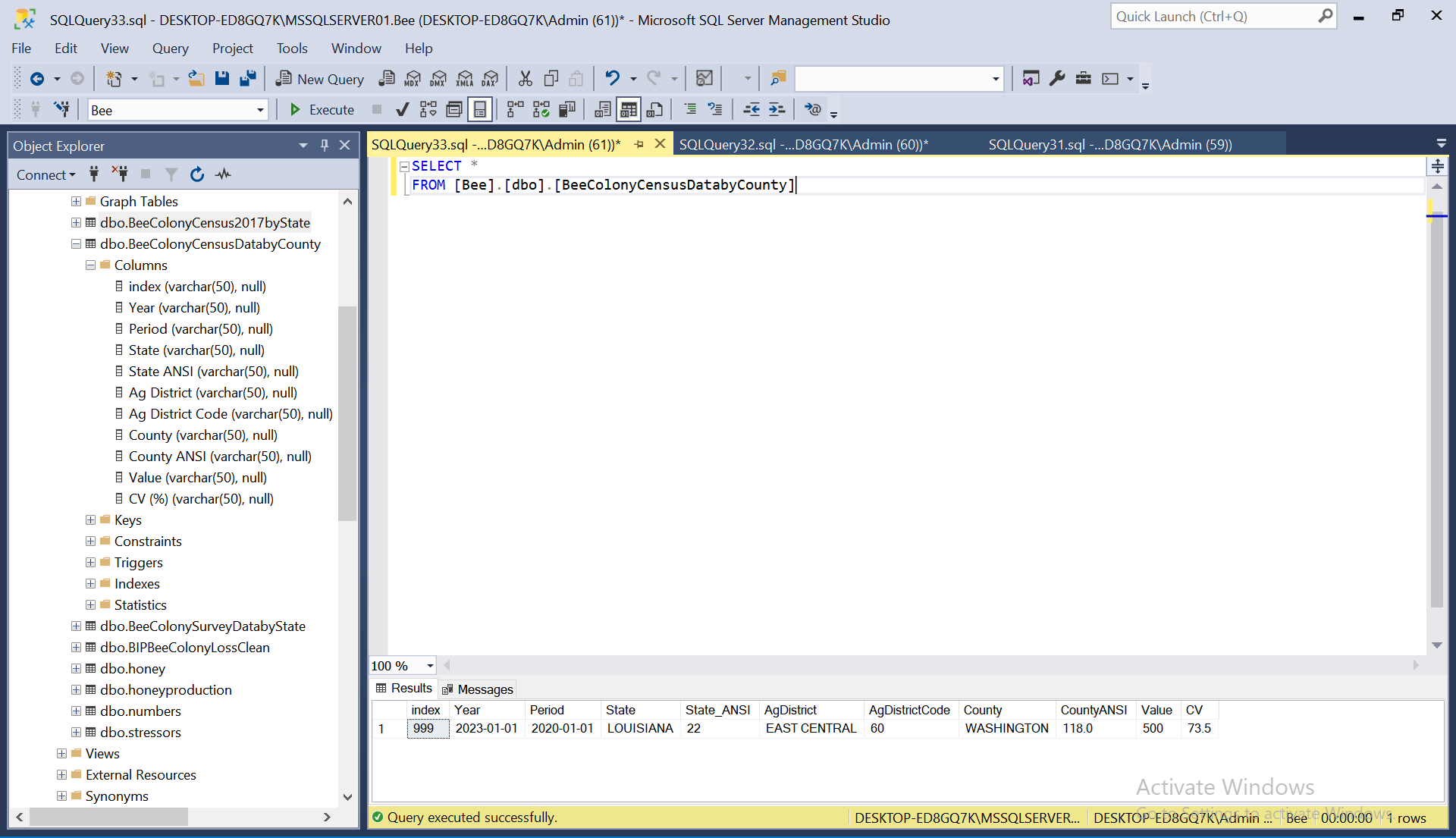
****

****

**multi-table query**

****

**Choice query:**

****

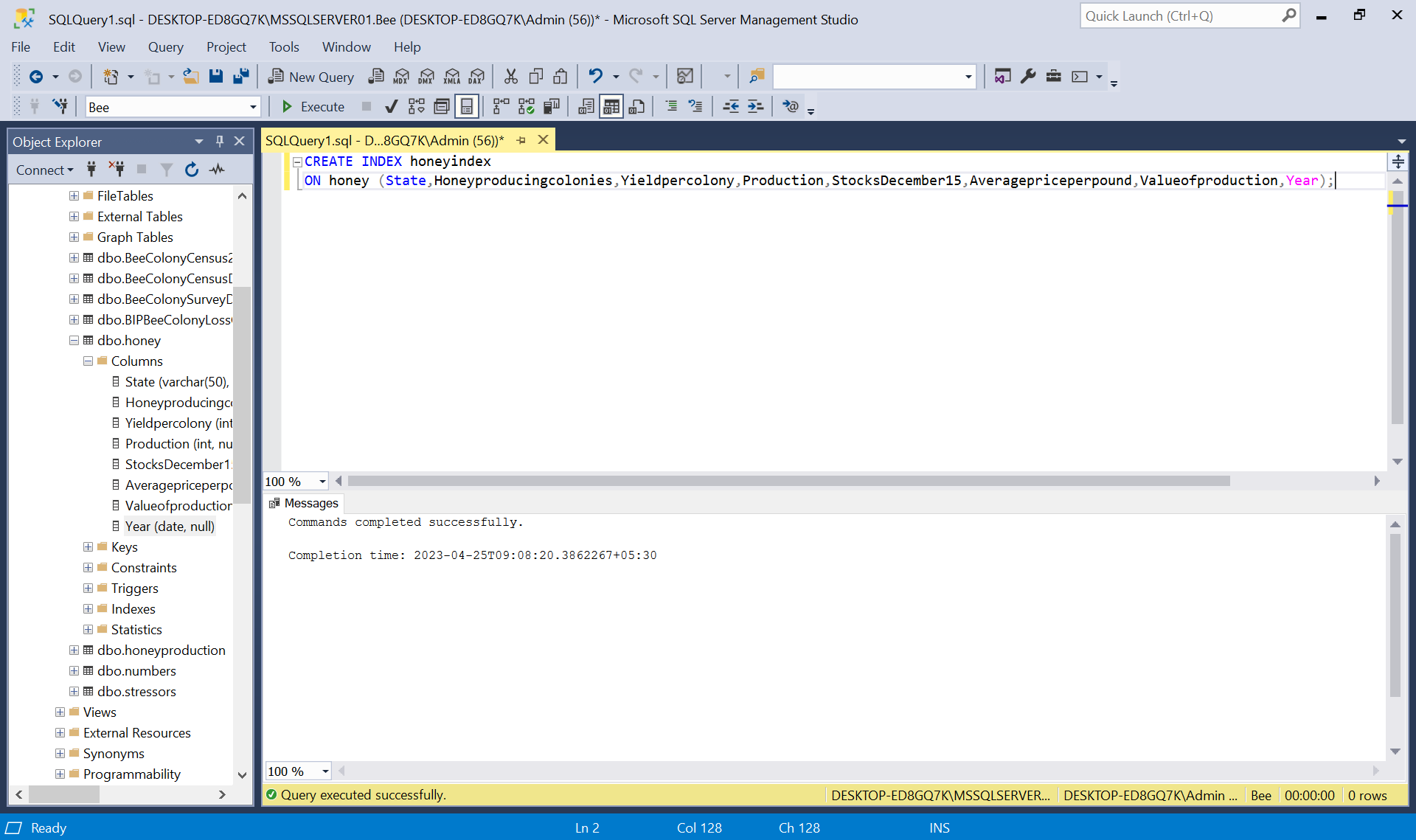
# Indexes

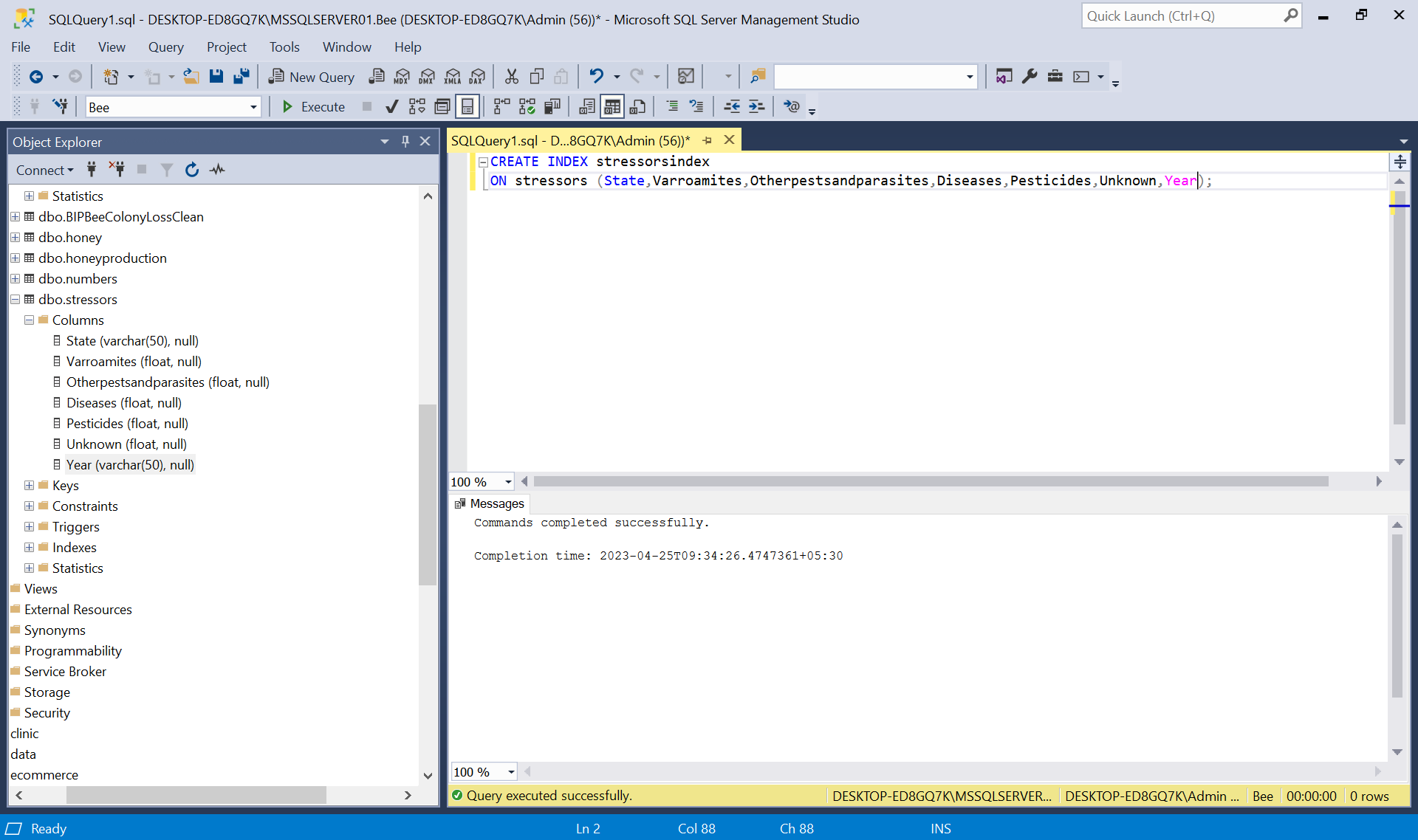
**I) Indexes:**

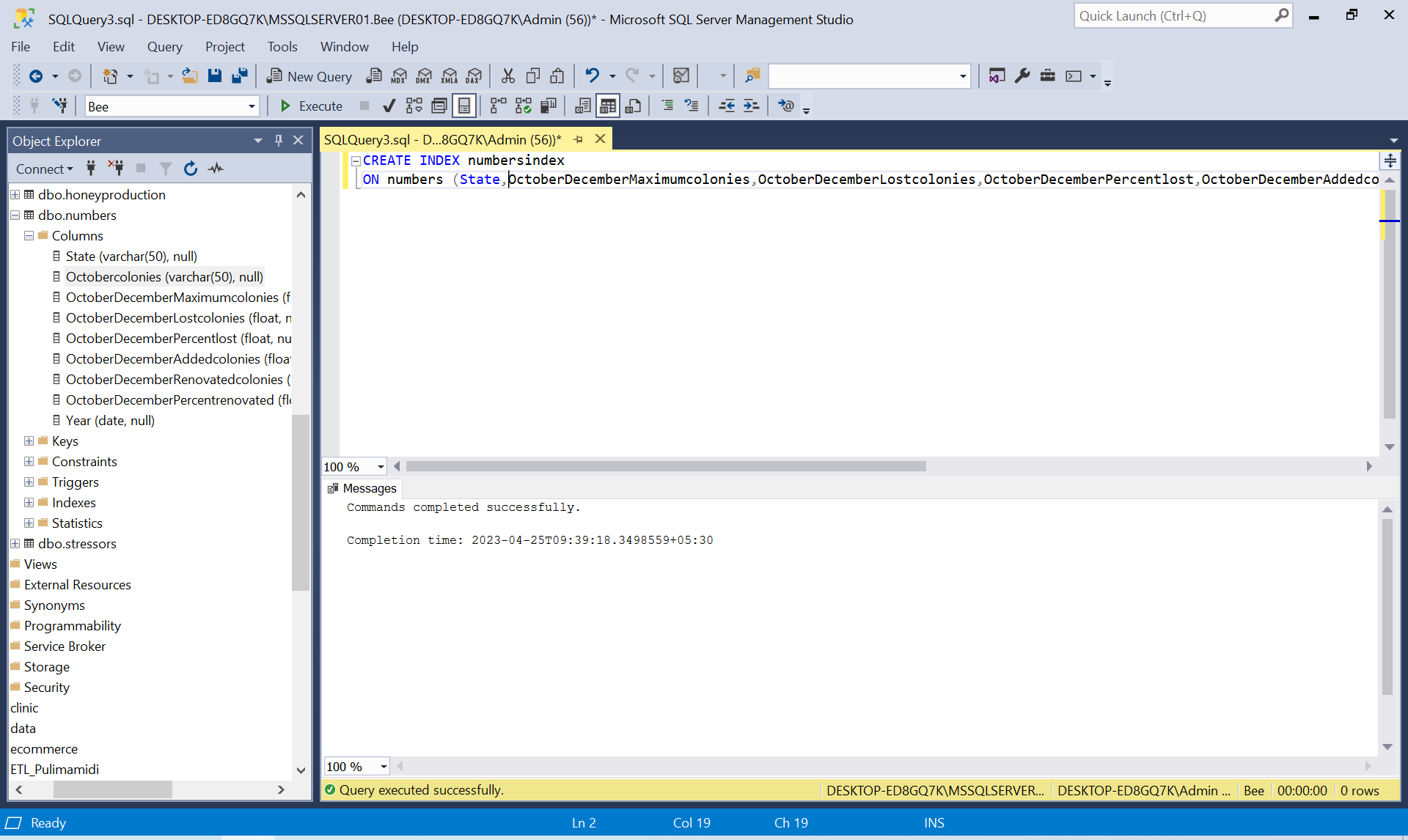
I have chosen these indexes because it would help me to fetch data values from tables. By choosing these indexes, the performance of the queries have improved because it was locating the data rapidly. Likewise, it had made the criteria of the query to be matched quickly.

3 indexes are i) honeyindex ii) stressorsindex iii) numbersindex

**Generate the indexes:**







# Views

**Two views are: i)BeeColonyCensusDatabyCountyview ii) honeyview**

**CREATE VIEW BeeColonyCensusDatabyCountyview AS**

**SELECT [index]**

**,[Year]**

**,[Period]**

**,[State]**

**,[State\_ANSI]**

**,[AgDistrict]**

**,[AgDistrictCode]**

**,[County]**

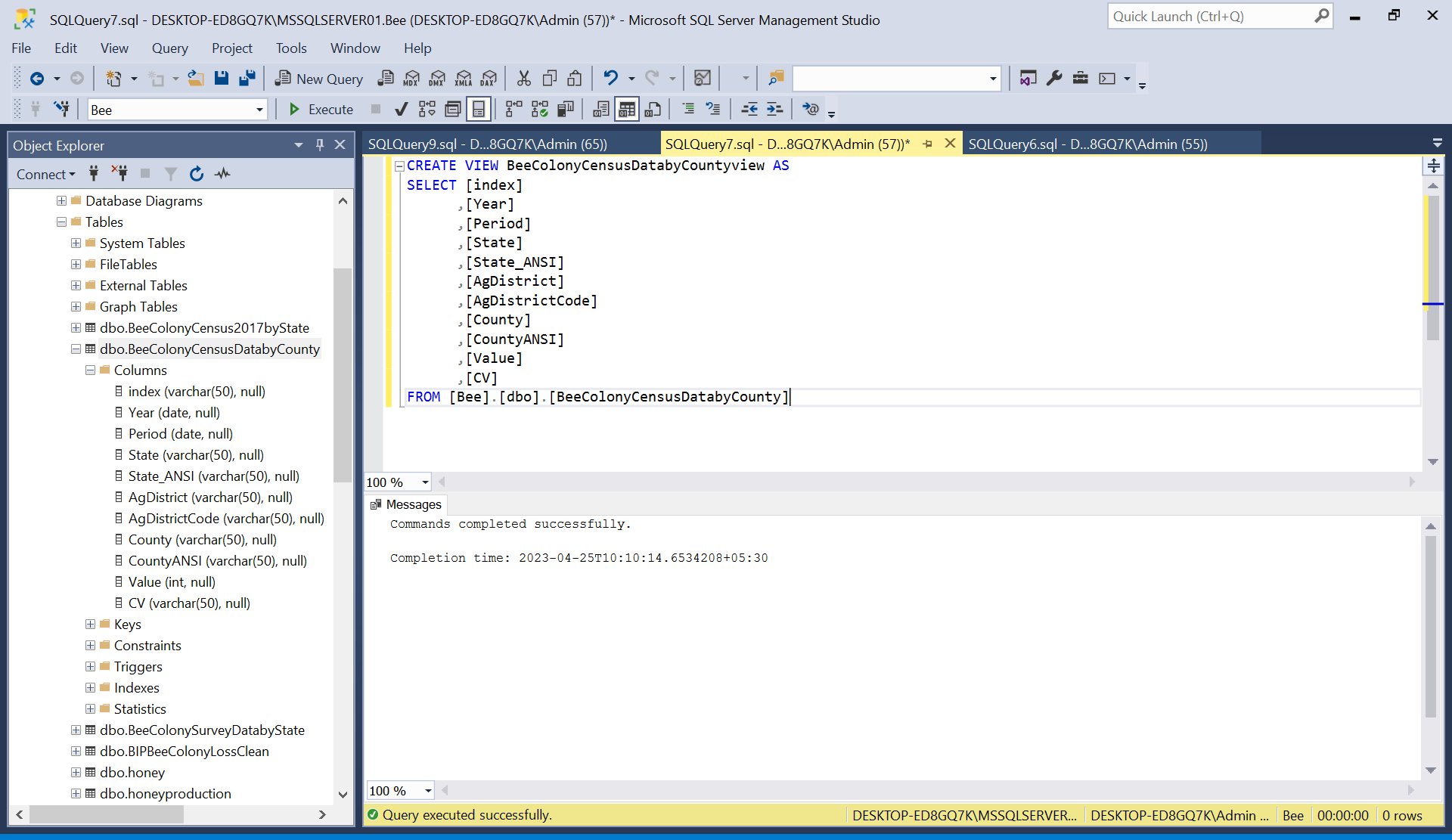
**,[CountyANSI]**

**,[Value]**

**,[CV]**

**FROM [Bee].[dbo].[BeeColonyCensusDatabyCounty]**

**Answer:** Addition of each view is valuable to this database system because it reduces the complexities of queries into a simplified form. Views predefined the joins and other database related calculations. In addition, the view enhances the security factors of the database system.

****

**Sql:**

**CREATE VIEW honeyview AS**

**SELECT [State]**

**,[Honeyproducingcolonies]**

**,[Yieldpercolony]**

**,[Production]**

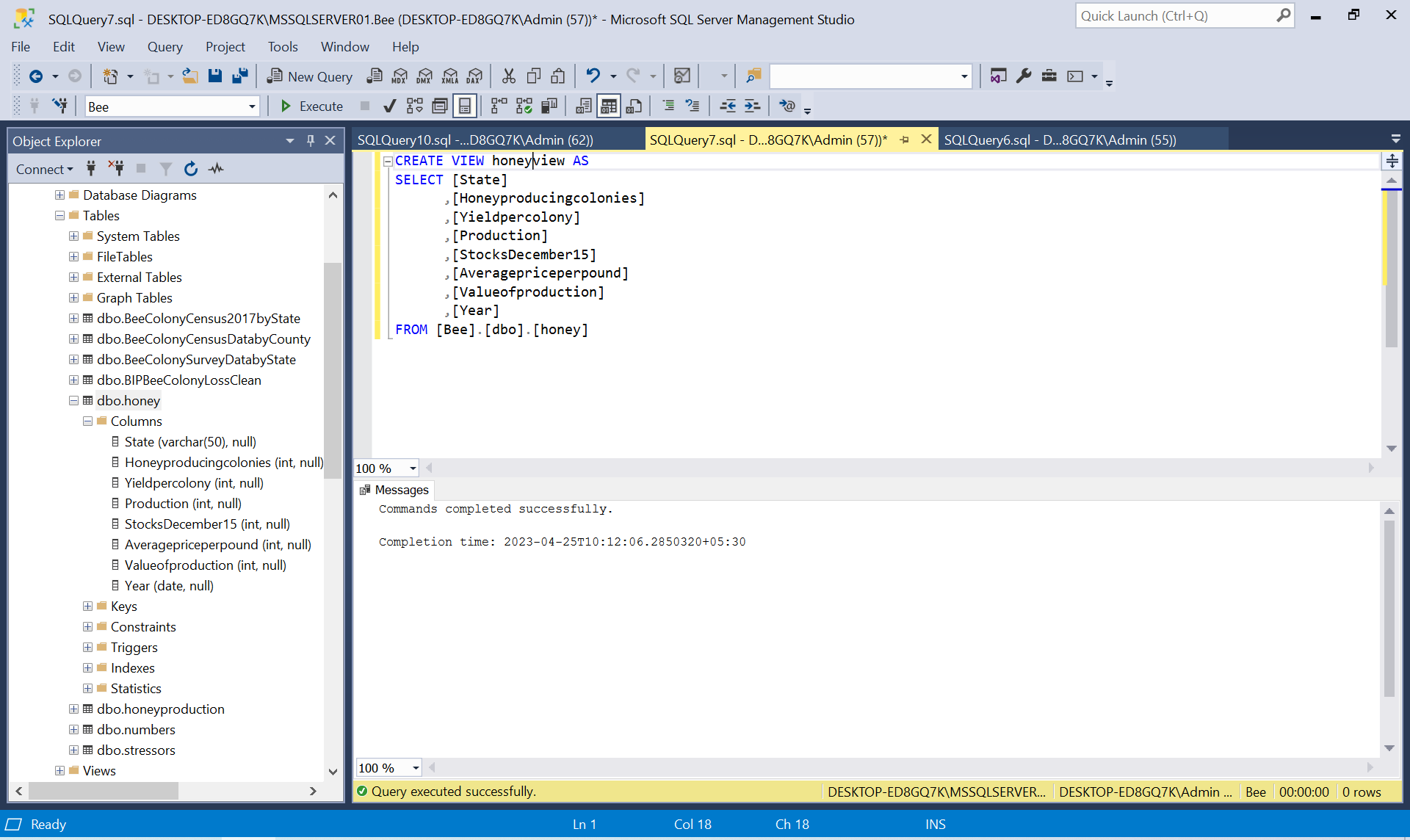
**,[StocksDecember15]**

**,[Averagepriceperpound]**

**,[Valueofproduction]**

**,[Year]**

**FROM [Bee].[dbo].[honey]**

****

# Triggers

*A trigger in SQL is a particular kind of stored procedure that is automatically carried out in response to specific occurrences or actions in a database table. When a given event or action occurs, such as when a record is added, changed, or removed in a table, certain activities or procedures that must be carried out at that time are automated by triggers in SQL.*

*In SQL, triggers are frequently used for a number of tasks, including upholding business rules that ensure data integrity, reviewing database changes, and automatically updating relevant data in other tables. When a row is added to or altered in another table, for instance, a trigger may be constructed to automatically update a linked table.*

*In SQL, triggers may be defined to run either before or after a certain action, such as an insert, update, or delete transaction. They may also be programmed to run once for each row that the event affects or once for each statement that the event triggers.*

**sqlcode:**

CREATE TRIGGER honey\_insert\_trigger

ON honey

AFTER INSERT

AS

BEGIN

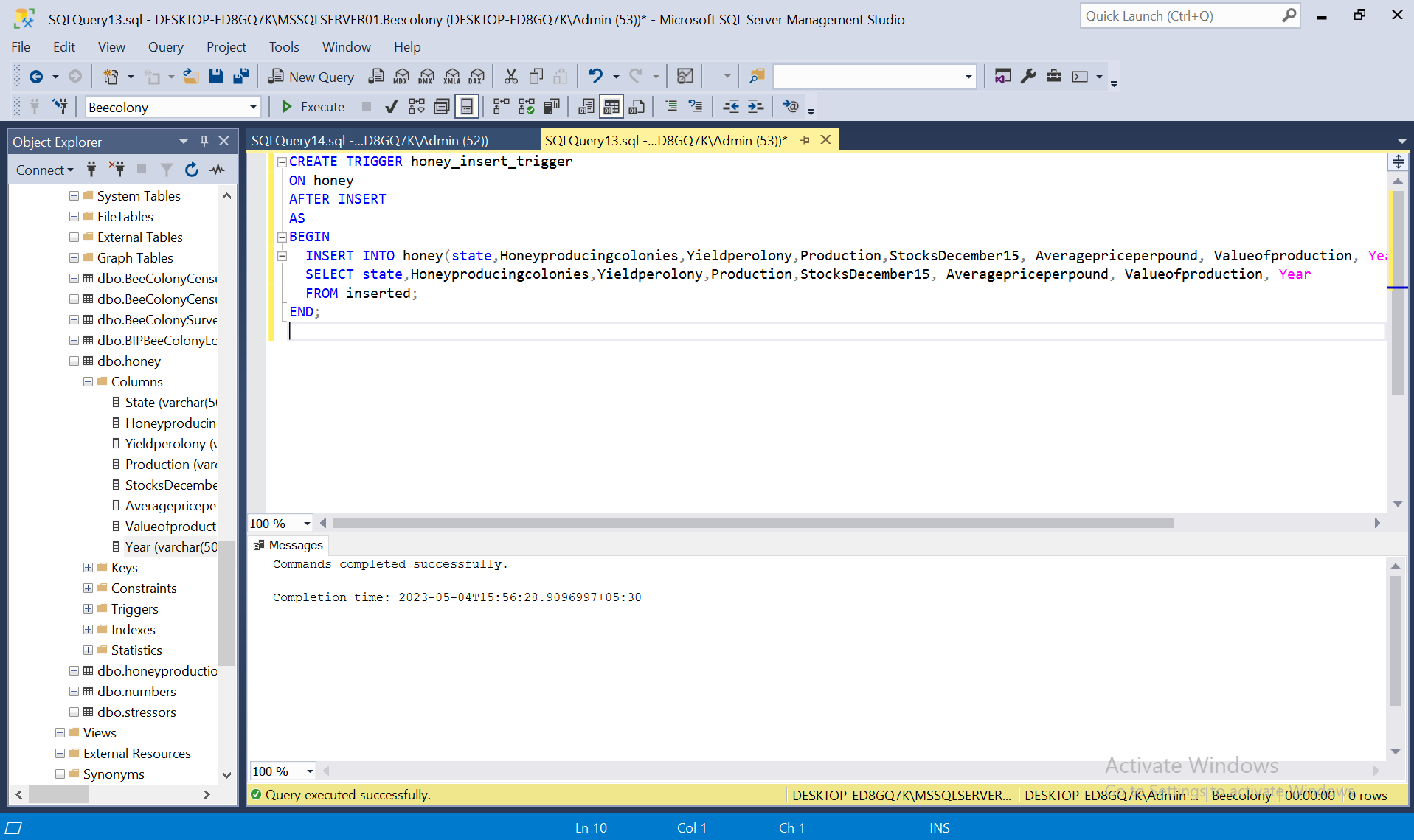
INSERT INTO honey(state,Honeyproducingcolonies,Yieldperolony,Production,StocksDecember15, Averagepriceperpound, Valueofproduction, Year)

SELECT state,Honeyproducingcolonies,Yieldperolony,Production,StocksDecember15, Averagepriceperpound, Valueofproduction, Year

FROM inserted;

END;

Screenshot of output:



# Transactions

**i. Transaction**

When one or more transactions of a database occur at a sequence, then, it is called a MySQL transaction.It can be said that the consistency and integrity of the dataset existed in the database system is maintained effectively.

**SQL query:**

**BEGIN TRANSACTION;**

**UPDATE [Bee].[dbo].[BeeColonyCensusDatabyCounty]**

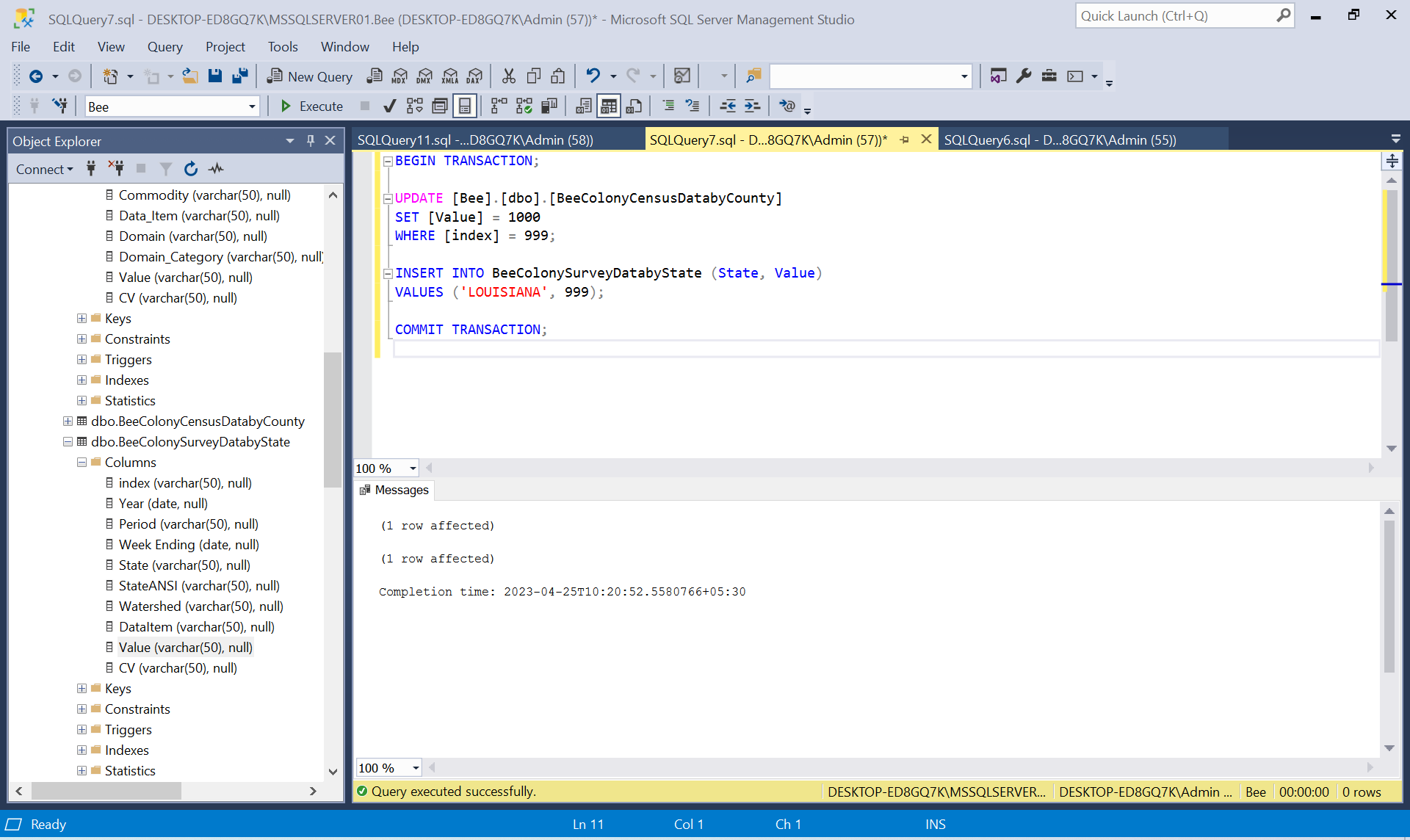
**SET [Value] = 1000**

**WHERE [index] = 999;**

**INSERT INTO BeeColonySurveyDatabyState (State, Value)**

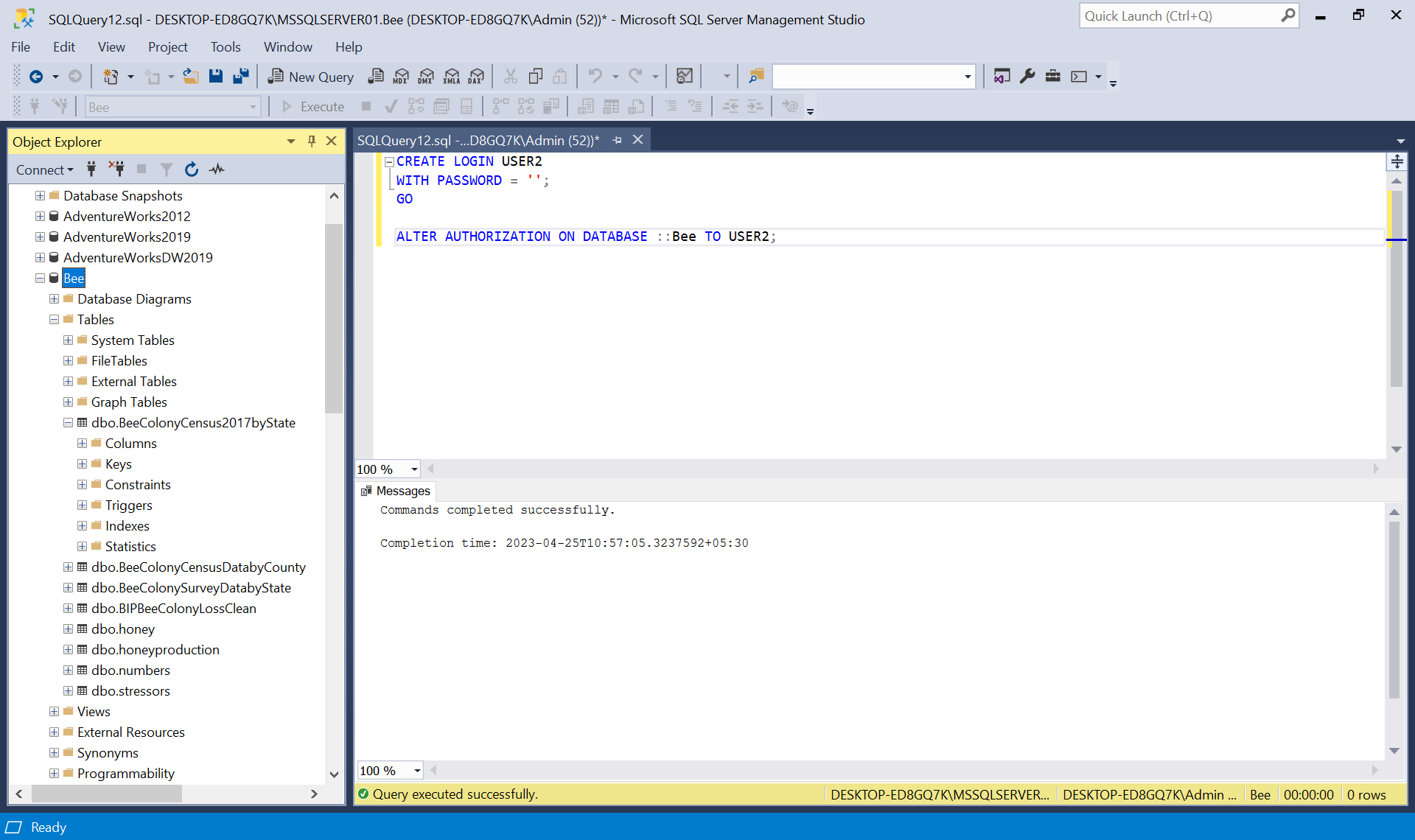
**VALUES ('LOUISIANA', 999);**

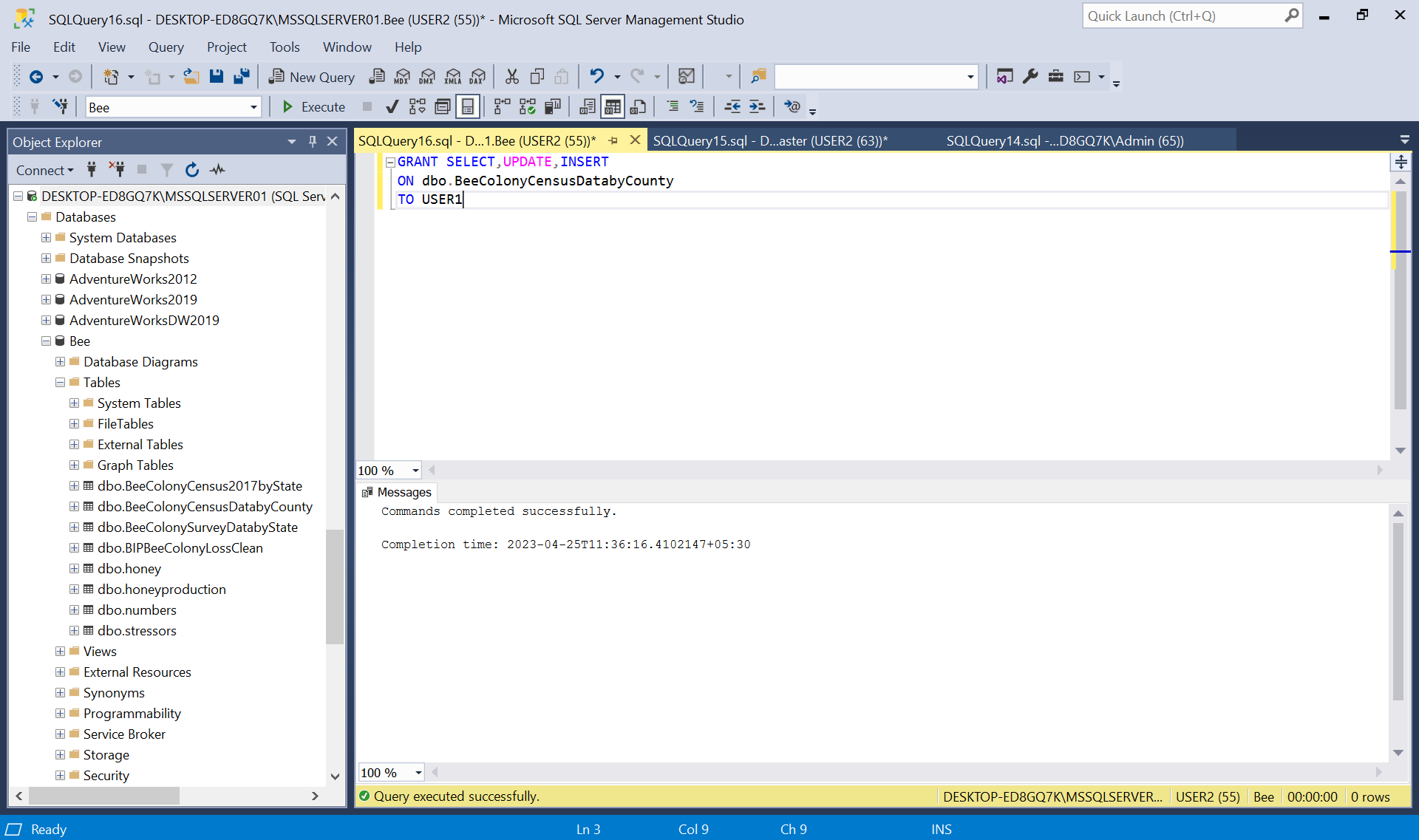
**COMMIT TRANSACTION;**

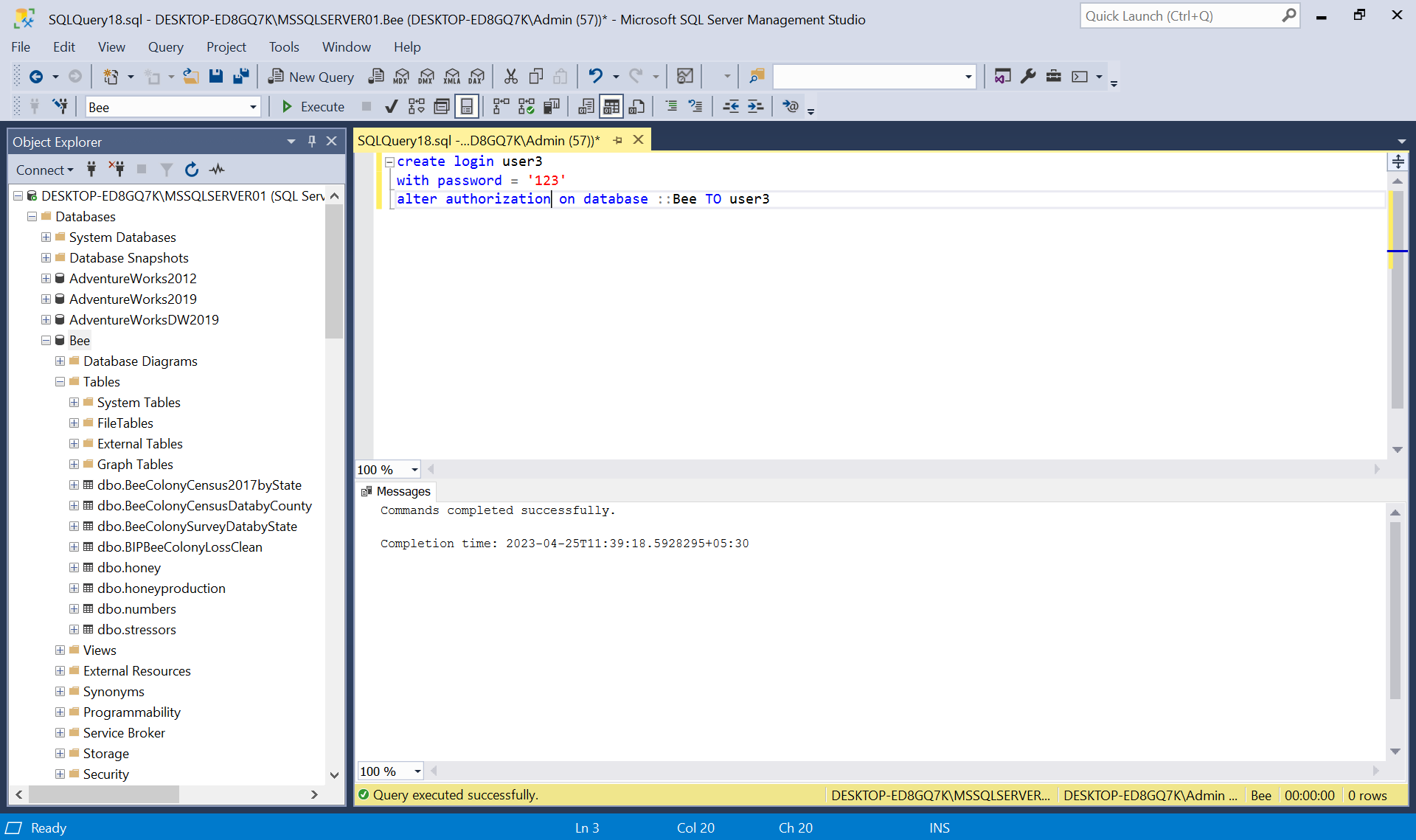
****

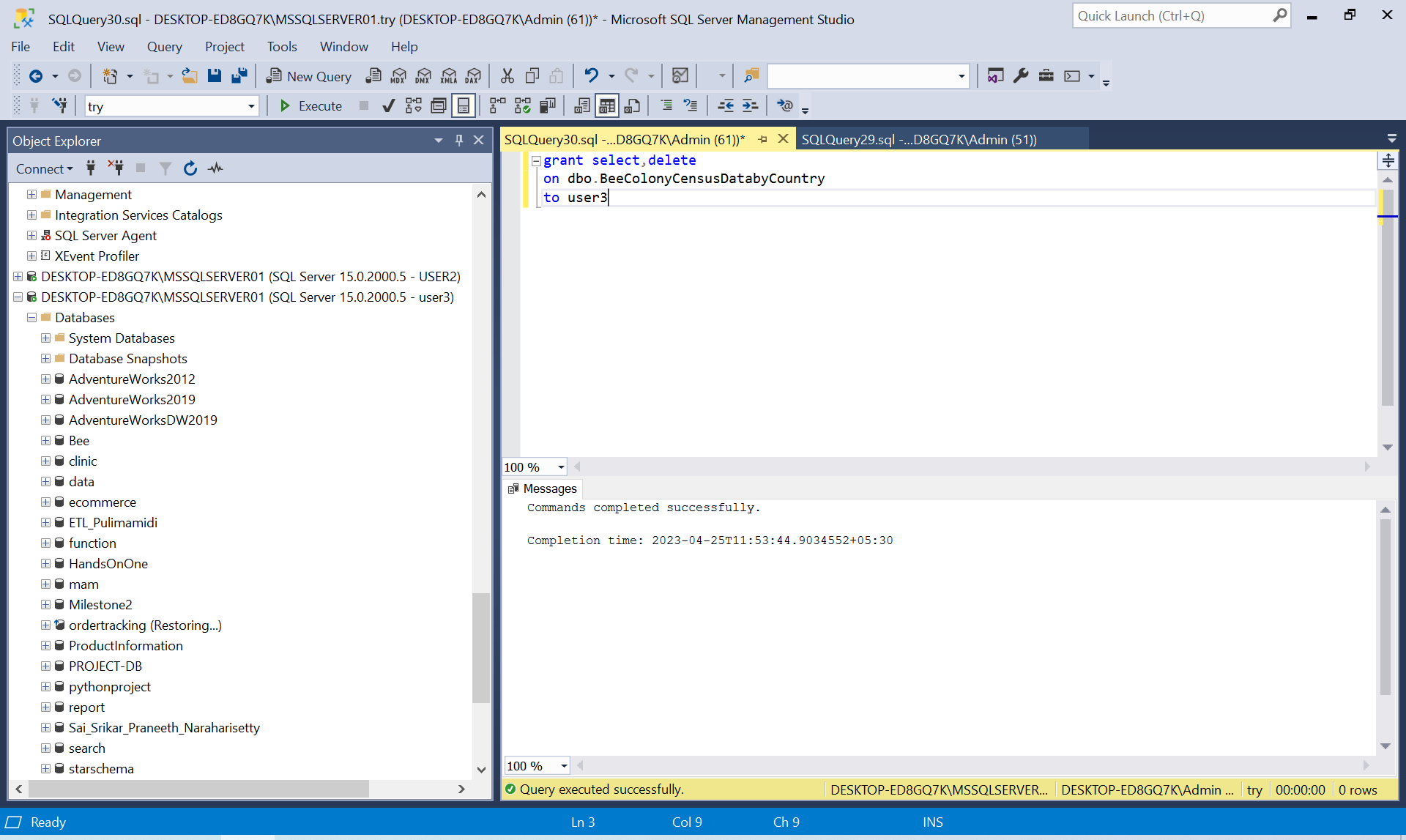
# Database Security

Yes, the Grant statement used in this database system is distinguishing the user in the database system. It also specifies the function performed by the user in the database system.

****

****

****

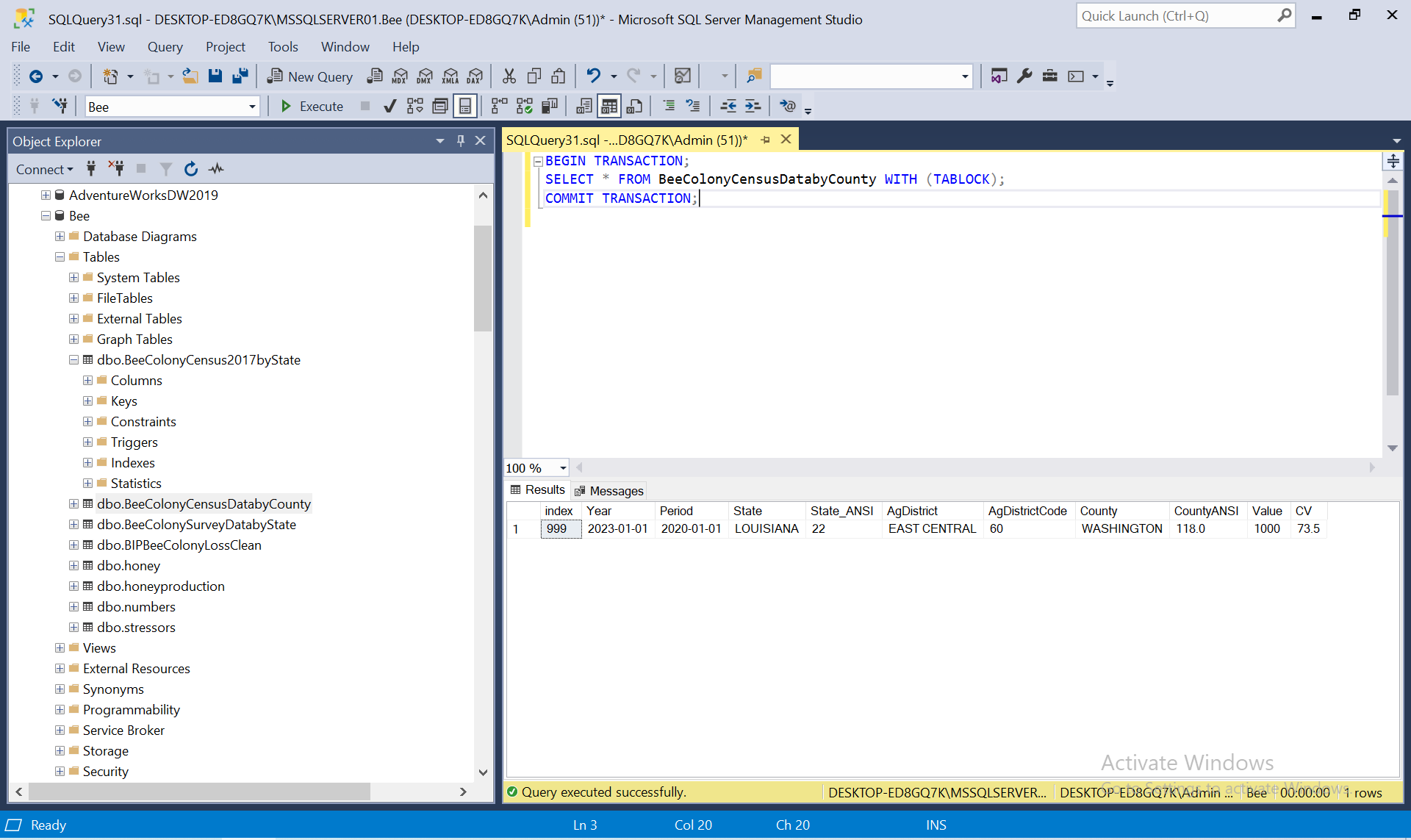
****

# Locking and Concurrent Access

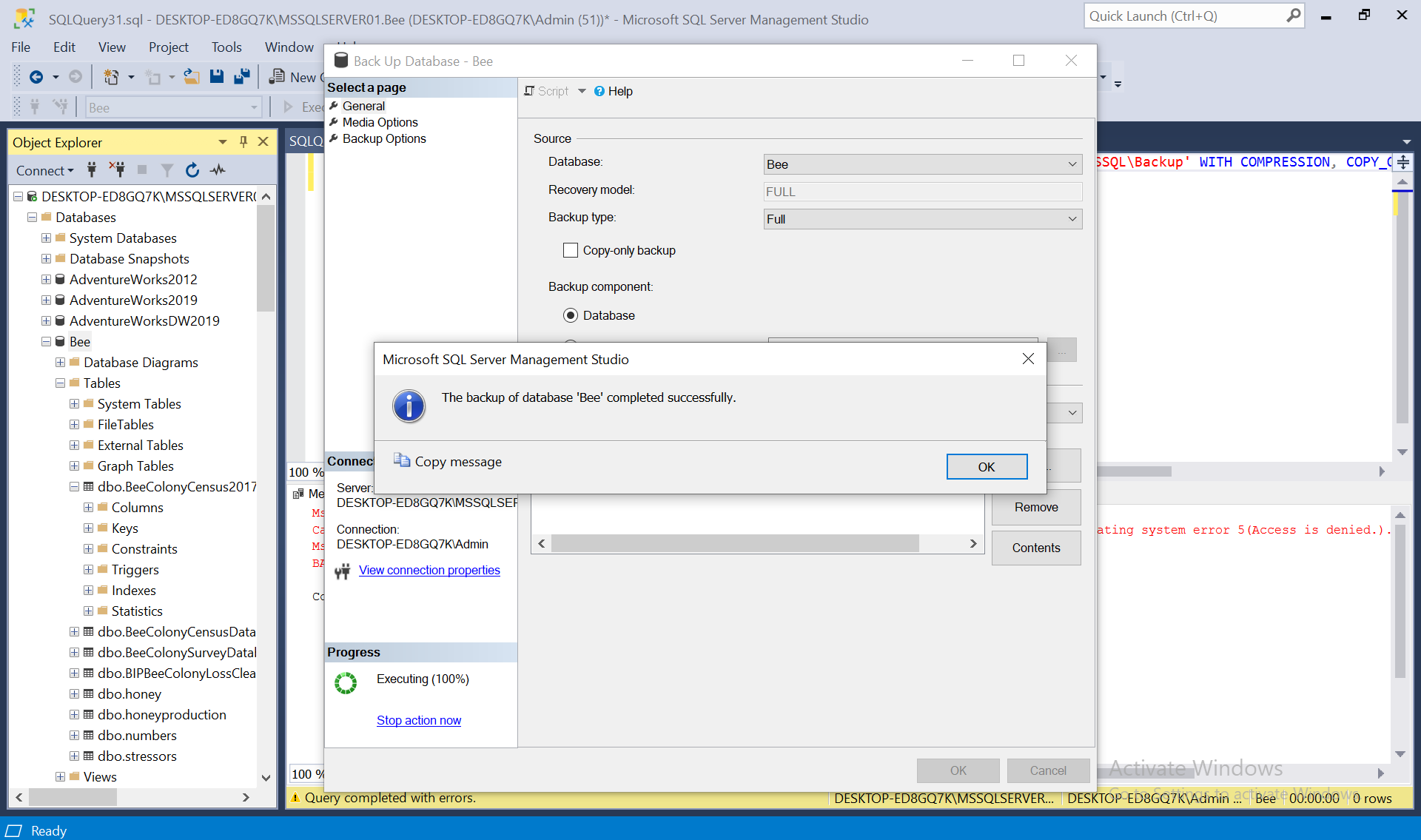
BEGIN TRANSACTION;

SELECT \* FROM BeeColonyCensusDatabyCounty WITH (TABLOCK);

COMMIT TRANSACTION;

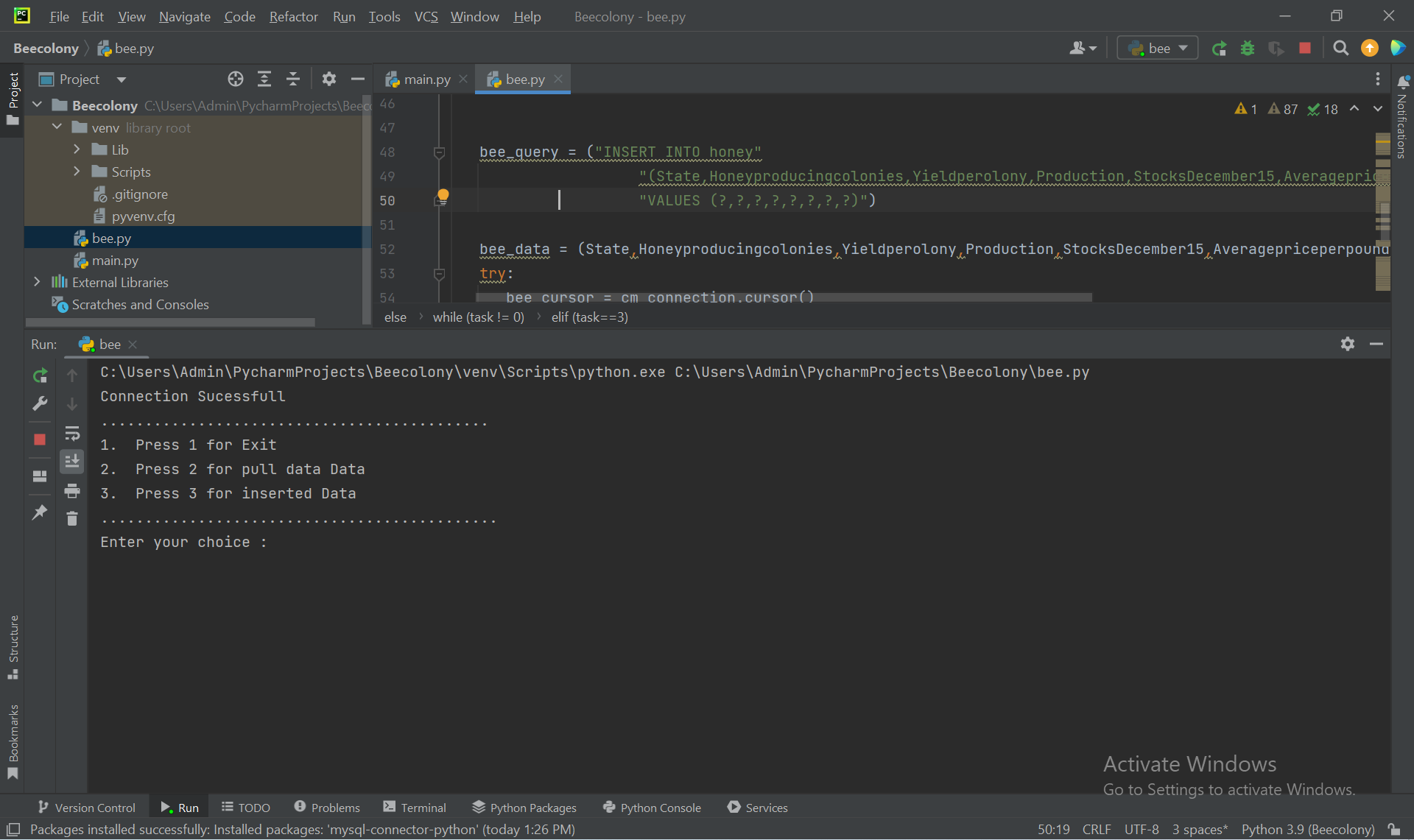
****

# Backing Up Your Database

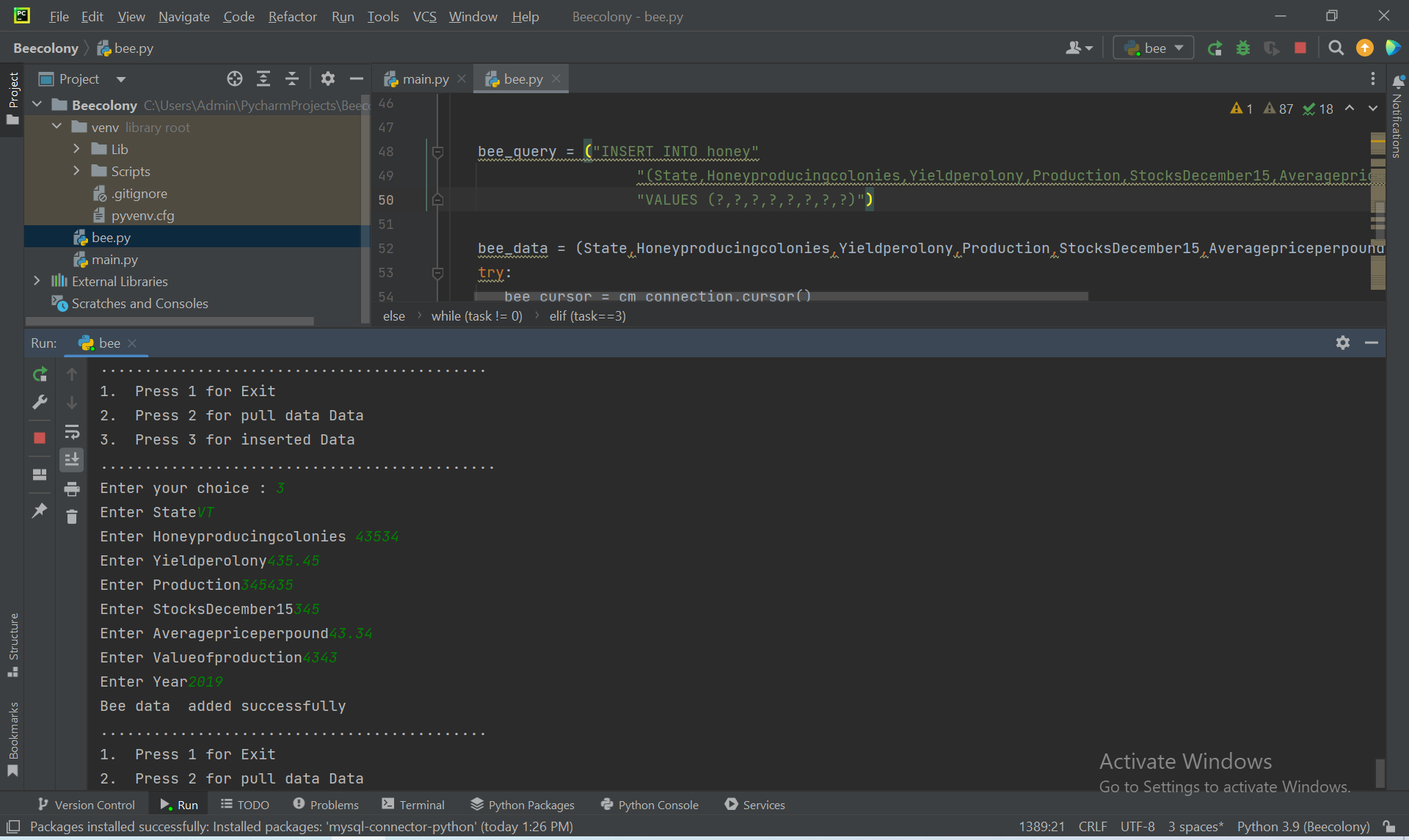
****

# Programming

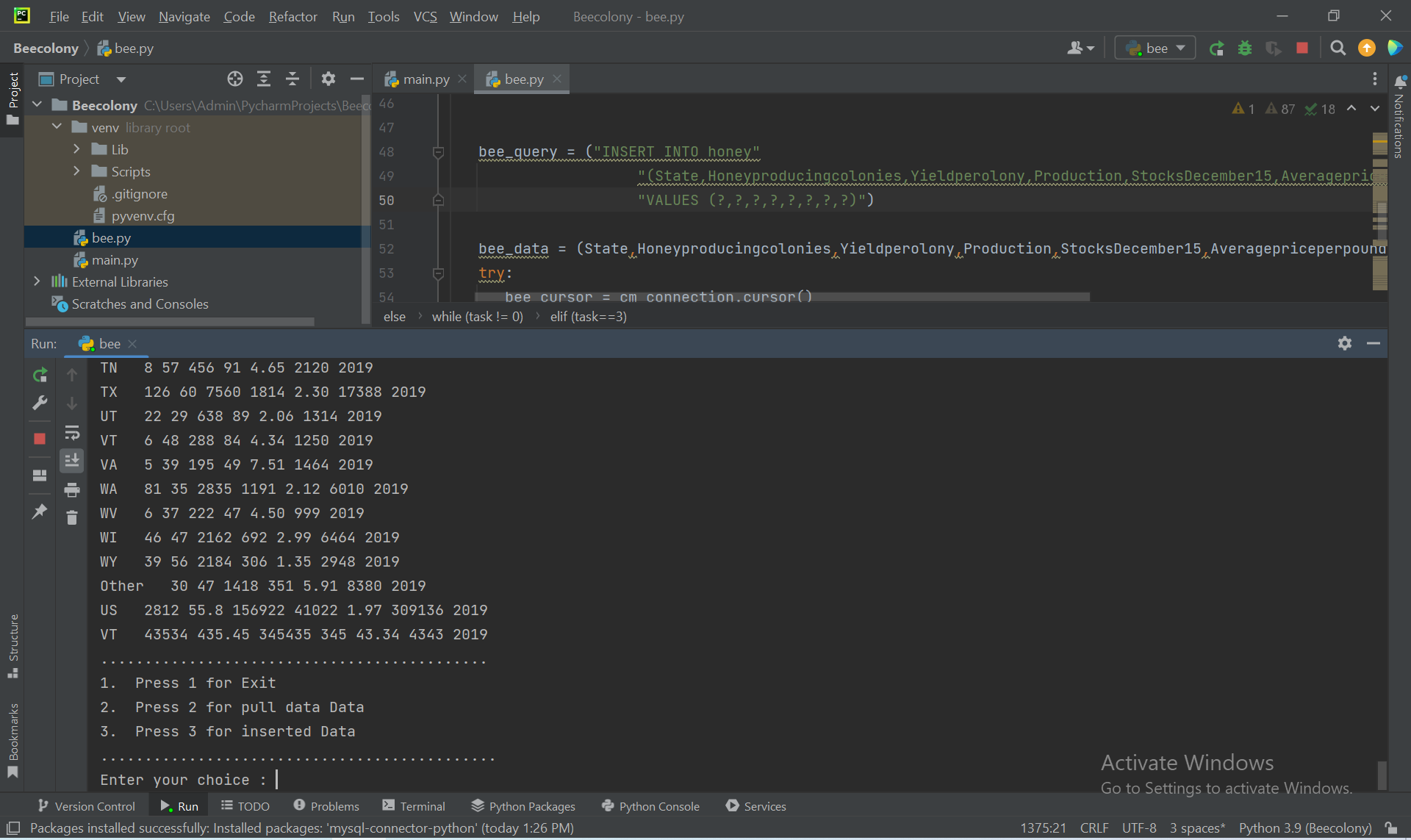
**Connection to the database in python programming:**

****

**Add bee data into beecolony database**

****

**pull data:**

****

python code:

import mysql.connector ##here i import the mysqlconnector

import pyodbc ##here import the pyodbc driver

from mysql.connector import errorcode ##here include the connector

try:

cm\_connection = pyodbc.connect( "Driver={SQL Server};"

"Server=DESKTOP-ED8GQ7K\MSSQLSERVER01;" ##here in connection to the database

"Database=Beecolony;"

"Trusted\_Connection=yes;")

except pyodbc.connect.Error as e: ## here include the authentication of the database

if e.errno == errorcode.ER\_ACCESS\_DENIED\_ERROR:

print("credentials is Invalid")

elif e.errno == errorcode.ER\_BAD\_DB\_ERROR:

print("no database") ##if no database are available the this message will be printed

else:

print("can't: connect", e) ## here is show the not connected error message

else:

print("Connection Sucessfull")

my\_cursor = cm\_connection.cursor()

task = 1

while (task != 0): ##here is the option for choose which option you want to perform

print ("............................................")

print("1. Press 1 for Exit")

print ("2. Press 2 for pull data Data")

print("3. Press 3 for inserted Data")

# print("4. Press 4 for delete Data")

print(".............................................")

task = int(input("Enter your choice : "))

if (task==2) :

select\_query = ("SELECT \* FROM honey") ##here is the code for pull data from the database

my\_cursor.execute(select\_query)

for row in my\_cursor.fetchall():

print("{} {} {} {} {} {} {} {}".format(row[0], row[1], row[2], row[3],row[4],row[5],row[6],row[7]))

elif (task==3) : ##here is the data added code

State = input("Enter State")

Honeyproducingcolonies = input("Enter Honeyproducingcolonies ")

Yieldperolony = input("Enter Yieldperolony")

Production = input("Enter Production")

StocksDecember15 = input("Enter StocksDecember15")

Averagepriceperpound = input("Enter Averagepriceperpound")

Valueofproduction = input("Enter Valueofproduction")

year=input("Enter Year")

bee\_query = ("INSERT INTO honey"

"(State,Honeyproducingcolonies,Yieldperolony,Production,StocksDecember15,Averagepriceperpound,Valueofproduction,year)"

"VALUES (?,?,?,?,?,?,?,?)")

bee\_data = (State,Honeyproducingcolonies,Yieldperolony,Production,StocksDecember15,Averagepriceperpound,Valueofproduction,year)

try:

bee\_cursor = cm\_connection.cursor()

bee\_cursor.execute(bee\_query, bee\_data)

cm\_connection.commit()

print("Bee data added successfully") ##here is successfully message

bee\_cursor.close()

except mysql.connector.Error as e:

print("Bee data not added")

print("Error: {}".format(e))

cm\_connection.close()

# Suggested Future Work

Utilizing cloud services will improve a database's performance and availability.It will managed database services, like Amazon RDS, Azure SQL Database, or Google Cloud SQL, are offered by cloud providers in near future. These services will take care of infrastructure maintenance, patching, backup, and replication, as well as automatic scalability and failover. Utilizing a cloud-based database service would enable quicker deployment and scaling while lowering management costs and improving availability.

Both benefits and drawbacks might be associated with data storage in a NoSQL format. NoSQL databases are very scalable and performant. They would be built to manage massive amounts of unstructured or semi-structured data in future.

# Activity Log

**Activity Log – first two weeks**

**March 15th**

* Conducted a meeting with the team.
* Discuss the feature and functionality of the system and set goals.
* Assigned roles and responsibilities to team members.

**March 18th**

* Completed data mapping.
* Review the data sources.
* Developed a plan for data migration and integration.

**March 21st**

* Completed the installation and configuration of the system.
* Tested the system to ensure that all the features were working correctly.
* Identified and resolved any issues related to the testing process.

**March 23th**

* Conducted a training session for the team on how to use the system.
* Provided hands-on training on the various features of the system.
* Answered questions and provided support to team members during the training session.

**March 24th**

* Began the data migration process from the old system to the new system.
* Monitored the data migration process to ensure that all data was transferred correctly.
* Identified and resolved any issues that arose during the data migration process.

**March 26th**

* Conducted a review of the new system with the team.
* Discussed any issues that arose during the implementation process and identified opportunities for improvement.

Set goals for ongoing maintenance and support of the system.