

**Usman Institute of Technology**

**Department of Computer Science**

**Course Code: CS311**

**Course Title: Introduction to Database Systems**

**SPRING 2023**

**Lab 01**

**Objective:**

Overview of the features of SQL.

.

**Student Information**

|  |  |
| --- | --- |
| Student Name | MAIRA USMAN |
| Student ID | 21B-011-SE |
| Date | 13 MARCH,2023 |
| **Assessment** | |
| Marks Obtained |  |
| Remarks |  |
| Signature |  |

**Usman Institute of Technology**

**Department of Computer Science**

**CS311 – Introduction to Database Systems**

**Lab 01**

**Instructions**

* Come to the lab in time. Students who are late more than 20 minutes, will not be allowed to attend the lab.
* Students have to perform the examples and exercises by themselves.
* Lab work must be submitted on the same day it is performed.

1. **Objective**

Overview of the features of SQL.

1. **Labs Descriptions**

A ***database (db)*** is an organized collection of data, typically stored in electronic format. It allows you to input, organize, and retrieve data quickly. Traditional databases are organized by fields, records, and files. To better understand what a database is, consider the telephone book as a simple example.

If you had the telephone book stored on disk, the book would be the file. Within the telephone book, you would have a list of records each of which contains a name, address, and telephone number. These single pieces of information (name, address, phone number) would each constitute a separate field.

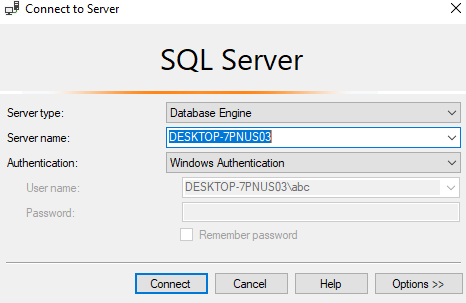
Because a database can store thousands of records, it would be a chore if you had to open a table and go through each record one at a time until you found the record you needed. Of course, the process would be even more difficult if you had to retrieve multiple records.

Thankfully, you don’t have to go through database records in this way. Rather, to retrieve data within a database, you run a database ***query***, which is an inquiry into the database that returns information back from the database. In other words, a query is used to ask for information from a database.

Databases are often found on ***database servers*** so that they can be accessed by multiple users and provide a high level of performance. One popular database server is Microsoft SQL Server. Database servers like SQL Server do not actually house graphical programs, word-processing applications, or any other type of applications. Instead, these servers are entirely optimized to serve only the purposes of the database itself, usually using advanced hardware that can handle the high processing needs of the database. It is also important to note that these servers do not act as workstations; they generally are mounted on racks located in a central data center and can be accessed only through an administrator’s desktop system.

### Creating Database by SQL SERVER EXPRESS

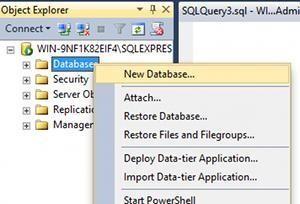
Select Start-->All Programs-->Microsoft SQL Server Express -->SQL SERVER ManagementStudio.



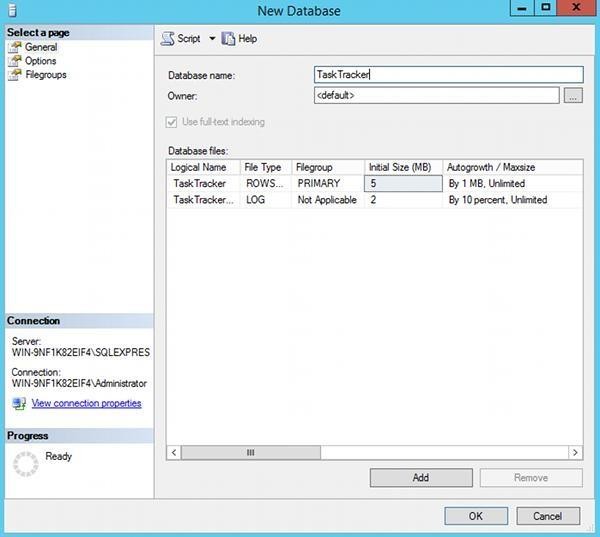
##### CREATE A DATABASE USING SSMS

The following steps demonstrate how to create a database in SQL Server 2019 using SQL Server Management Studio.

1. From the Object Explorer, right click on the Databases folder/icon and select New database...:

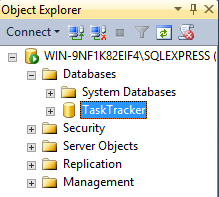


Name your database (I called mine TaskTracker) and click OK:



##### Your New Database

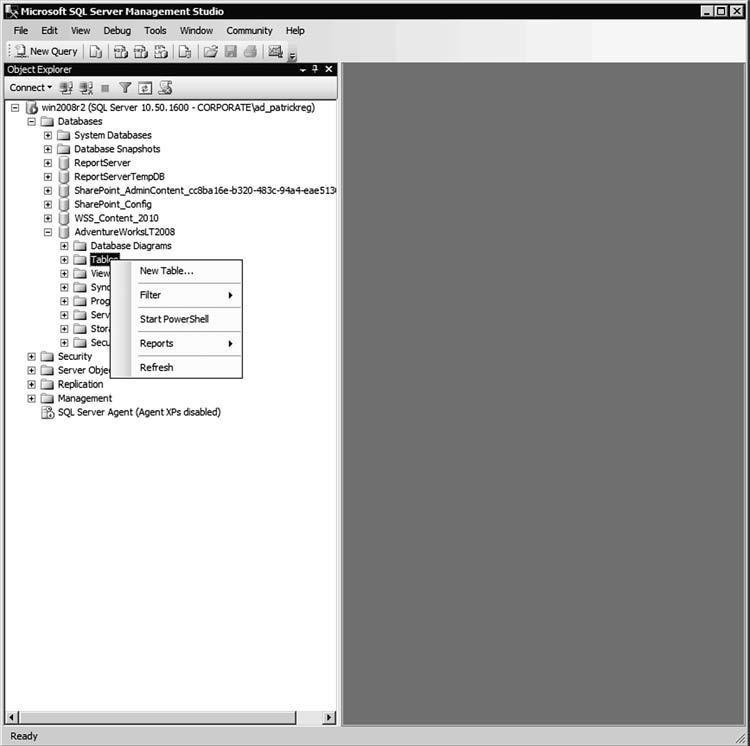
Your new database will appear under the Databases section of the Object Explorer (just under the System Databases folder). Here's mine:



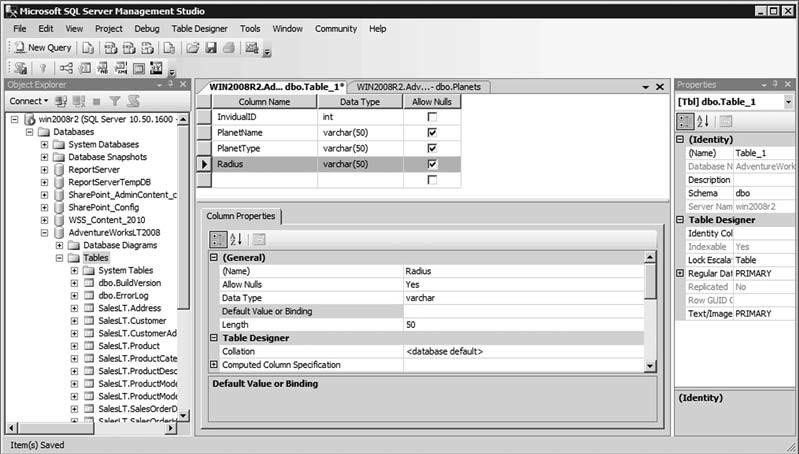
**CREATE A TABLE USING SSMS**

Before you begin, be sure to launch SQL Server Management Studio. Make sure you’ve expanded the particular database in which you wish to create the new table, then follow these steps:

**Step1 .** Right-click the **Table** folder and select **New Table**, as shown in Figure 1-1:

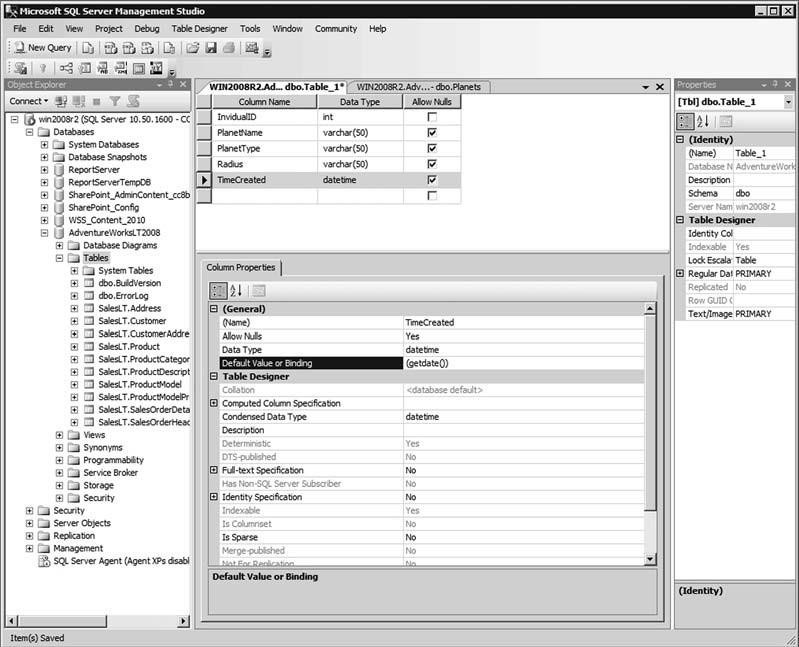


**Figure 1-1** Creating a new table

**Step 2.** Use the information shown in Figure 1-2 to complete the details for Column Name, Data Type, and Length, as specified in the parentheses and Allow Nulls columns.

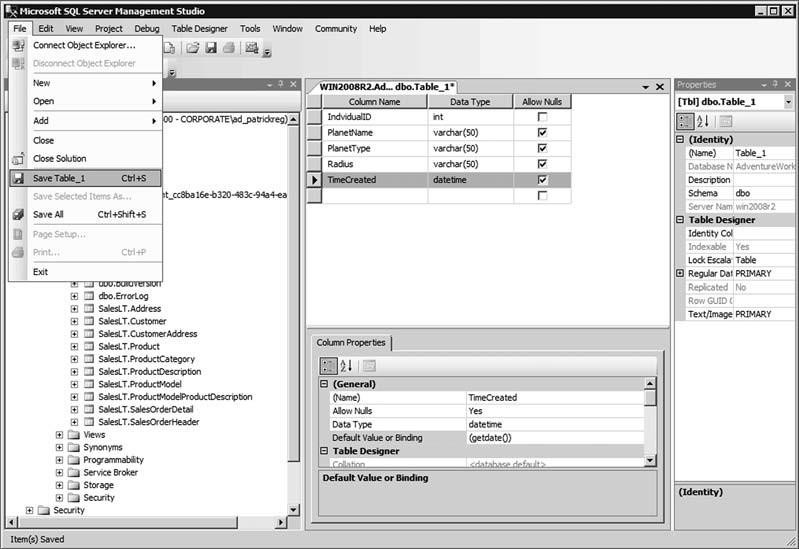
**Figure 1-2** Column names and identifying informatio

**Step 3.** Set the Default Value of the DateCreated column to *(getdate())*; this will insert the current date within each new record for that specific field. See Figure 1-3.



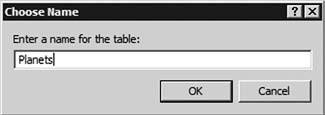
**Figure 1-3** Setting the Table Designer properties

**Step 4.** Save your new table by selecting File > Save Table\_1, as shown in Figure 1-4.



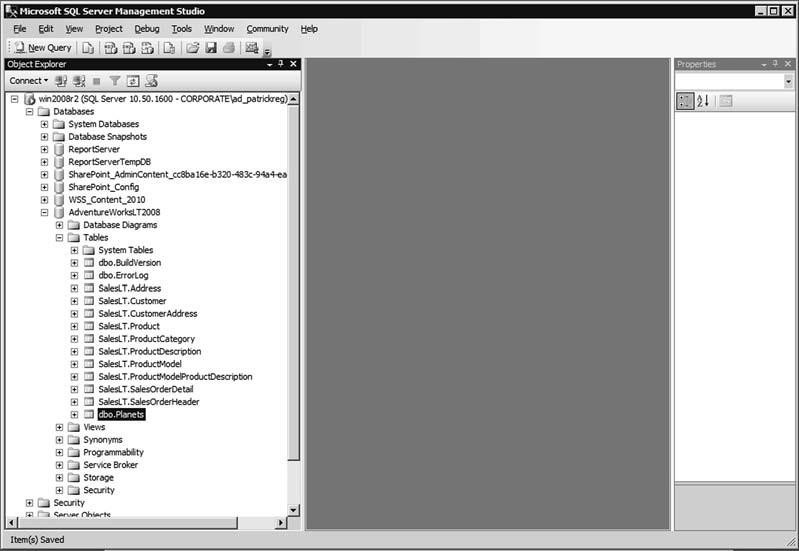
**Figure 1-4** Saving the new table

**Step 5.** Type the new name of the table you are saving, as shown in Figure 1-5.



**Figure 1-5** Naming the table

Your new table will appear under the **Tables** section, as depicted in Figure 1-6.



**Figure 1-6** The newly created table

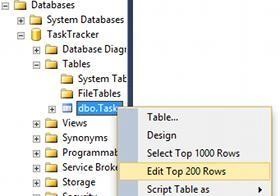
### ADDING DATA

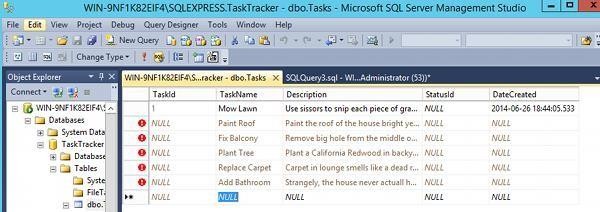
We can use the Edit Top 200 Rows option to manually type data directly into the table rows.

Manually entering data is OK if you only have a little bit of data to enter. But it's a bit clunky and can impractical if you have a lot of data. Plus it doesn't really suit most business needs, where non-technical users need to be able to update the database.

In any case, here's how to manually enter data directly into the table:

1. In the Object Explorer, right click on the table you wish to open, and select :

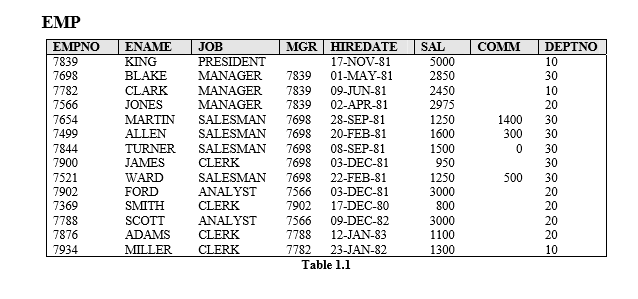


You can now start entering the

data directly into your table

The table 1.1 below shows the contents of the EMP table or relation that stores data about employees presently working in an organization.

* The table has eight columns namely EMPNO, ENAME. JOB, MGR, HIREDATE, SAL, COMM, DEPTNO storing the different attributes of an employee.
* The table has fourteen rows each representing all data that is required for a particular employee. Each row in a table should be identified by a ***primary key***, which allows no duplicate rows. In table 1.1, EMPNO is the primary key where as in table 1.2, DEPTNO is the primary key.
* A ***foreign key*** isa column or a set of columns that refers to a primary key or a unique key in the same table or another table. In EMP table, DEPTNO is the foreign key.
* A field can be found at the intersection of a row and a column. There can be only one value in it.
* A field may have no value in it. This is called *null value*. In the EMP table, only employees who have a role of *salesman* have a value in the COMM (Commission) field.



The table DEPT below contains data about different departments in an organization and SALGRADE gives details of salaries for various grades. The table JOB\_HISTORY stores data about the job history of employees.

**DEPT**

|  |  |  |
| --- | --- | --- |
| **DEPTNO** | **DNAME** | **LOC** |
| 10 | ACCOUNTING | NEW YORK |
| 20 | RESEARCH | DALLAS |
| 30 | SALES | CHICAGO |
| 40 | OPERATIONS | BOSTON |

**Table 1.2**

**SALGRADE**

|  |  |  |
| --- | --- | --- |
| **GRADE** | **LOSAL** | **HISAL** |
| 1 | 700 | 1200 |
| 2 | 1201 | 1400 |
| 3 | 1401 | 2000 |
| 4 | 2001 | 3000 |
| 5 | 3001 | 9999 |

**Table 1.3**

**JOB\_HISTORY**

|  |  |  |  |
| --- | --- | --- | --- |
| **EMPNO** | **JOB** | **START\_DATE** | **END\_DATE** |
| 7698 | ASSISTANT | 04-MAR-80 | 30-APR-81 |
| 7654 | RECEPTIONIST | 13-JAN-80 | 09-SEP-80 |
| 7654 | SALESMAN | 10-SEP-80 | 20-SEP-81 |
| 7788 | PROGRAMMER | 13-FEB-80 | 03-DEC-82 |
| 7876 | TYPIST | 12-APR-80 | 13-NOV-81 |
| 7876 | OPERATOR | 15-NOV-81 | 11-JAN-83 |
| 7839 | ANALYST | 13-JUN-78 | 10-OCT-81 |

**Table 1.4**

**Relating Multiple Tables**

Each table contains data that describes exactly one entity. For example, the EMP table contains information about employees. Since data about different entities is stored in different tables, it may be needed to combine two or more tables to answer a particular question. For example, someone may want to know the location of the department where an employee works. In this scenario, information is needed from both the EMP and DEPT table. An RDBMS enables to relate the data in one table to the data in another table by using the foreign keys.

**Guidelines for Primary and Foreign Keys**

* No duplicate values are allowed in a primary key.
* Primary keys generally cannot be changed.
* Foreign keys are based on data values and are purely logical, not physical pointers.
* A foreign key value must match an existing primary key value or unique key value, or else be null.

**SQL (Structured Query Language)**

Most commercial database management systems support a query language, SQL, which is the most influential commercially marketed product. SQL is a nonprocedural language: you specify *what* information you require, rather than *how* to get it. In other words, SQL does not require you to specify the access methods to the data. As a result, it doesn’t provide traditional programming structures. Instead, it is a language for specifying the operations at an unusually high level. The details of the implementation are left to the DBMS.

In Oracle, SQL is a language for communication with the Oracle Server from any tool or application. Oracle SQL has many extensions. Although we refer to the SQL language as a *query language*, it contains many other capabilities besides querying a database. It includes features for defining the structure of the data, for inserting and modifying data in the database, and for specifying security constraints.

It has following advantages:-

* Efficient
* Easy to learn and use
* Functionally complete (SQL allows to define, retrieve, and manipulate data in the tables.)

**Background of SQL**

The SQL language, originally called SEQUEL, was developed by IBM as part of the extensive work on the relational model in 1970s at their San Jose Research Laboratory, California. In order to test the viability of implementing the relational model in a DBMS, workers at the San Jose Research Laboratory undertook the project of the development of a prototype RDBMS named *System R*. This project took place from 1974 to 1979. The System R project led to the development of SEQUEL (Structured English Query Language) which was later renamed as SQL. Because System R was well received at the user sites where it was installed, other vendors began developing relational products that used SQL. In the late 1970s, the database management system *Oracle* was produced by what is now called the Oracle Corporation and was probably the first commercial implementation of a relational DBMS based on SQL. ORACLE is now available in mainframe, client-server and PC-based platforms for many operating systems including DOS, OS/2, and various UNIX operating systems, Windows, VAX/VMS and MVS. Numerous RDBMS products now support the SQL language.

In 1986, the American National Standards Institute (ANSI) and the International Standards Organization (ISO) published an SQL standard, called SQL-86. IBM published its own corporate SQL standard, the *Systems Application Architecture Database Interface* (SAASQL) in 1987. An extended standard for SQL, SQL-89, was published in 1989. The ISO and

ANSI committees created SQL-92 which was a more extensive expansion of SQL-86. SQL92 which was in fact a first major revision of the SQL standard is sometimes referred to as SQL2. It was not until 1999 that the next release of the standard was formalized, commonly referred to as SQL3. This release contains additional features that support object-oriented data management.

**Scope**

SQL is used for all types of database activities by all types of users including:

* System administrators
* Database administrators
* Security administrators
* Application programmers
* Decision support system personnel
* Many other types of end users

**Language Components**

The SQL language has several parts: -

**Data Retrieval**: The SQL includes a query language based on both the relational algebra and the tuple relational calculus.

**Data-definition language (DDL)**: The SQL DDL provides commands for defining relation schemas, deleting relations, creating indices, and modifying relation schemas.

**Interactive data-manipulation language (DML)**: It includes commands to insert tuples into, delete tuples from, and to modify tuples in the database.

**Embedded DML**: The embedded form of SQL is designed for use within general purpose programming languages, such as COBOL, Pascal and C.

**View definition**: The SQL DDL includes commands for defining views.

**Authorization**: The SQL DDL includes commands for specifying access rights for relations and views.

**Integrity**: The SQL DDL includes commands for specifying integrity constraints that the data stored in the database must satisfy. Updates that violate integrity constraints are disallowed.

**Transaction control**: SQL includes commands for specifying the beginning and ending of transaction. Several implementations also allow explicit locking of data for concurrency control.

**Basic Data Retrieval**

The basic structure of an SQL query consists of three clauses: SELECT, FROM and WHERE. SELECT \* | {[DISTINCT] column | expression [alias], …}

FROM table

[WHERE condition];

**Note**: The keywords and clauses enclosed in square brackets are *optional*.

**Examples**

1. Selecting all columns from a table

SELECT \*

FROM DEPT;

1. To select names of all jobs in a department, use

SELECT DISTINCT JOB

FROM EMP;

**Note**: The DISTINCT clause before a column name suppresses duplicate values iii. To select all employees whose salary is greater than 2200.

SELECT \*

FROM EMP

WHERE SAL > 2200; iv. To display the name and department number of employees who were hired before 12th May, 1981.

SELECT ENAME, DEPTNO

FROM EMP

WHERE HIREDATE < ’12-MAY-1981’;

1. To display the name and job of employees using literal character strings and concatenation operators. SELECT ENAME || ‘ is a ‘ || JOB

AS “Employee Details”

FROM EMP;

1. Retrieving data from multiple tables: To select employee name, job and department name,

SELECT E.ENAME, E.JOB, D.DNAME

FROM EMP E, DEPT D

WHERE E.DEPTNO = D.DEPTNO;

**EXERCISES**

1. What are relational databases? Describe, with the help of an example, how an RDBMS enables to relate data in one table to the data in another table.

\_Relational databases are a type of database management system that organizes data into tables or relations with each row representing a unique record and each column representing a field. An RDBMS enables relating data in one table to data in another table through common fields, known as keys. This allows for efficient storage and retrieval of information in a variety of applications.

For example, a company might use a relational database to manage its customer orders and inventory. The customer orders table might include information such as order number, customer name, and shipping address, while the inventory table might include information such as product name, SKU, and quantity on hand. By linking these tables using common fields such as order number and SKU, the company can easily retrieve information about a specific customer order and the products included in that order. Overall, relational databases provide a flexible and efficient means of organizing and managing data, making them a widely used tool in a variety of industries and applications.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

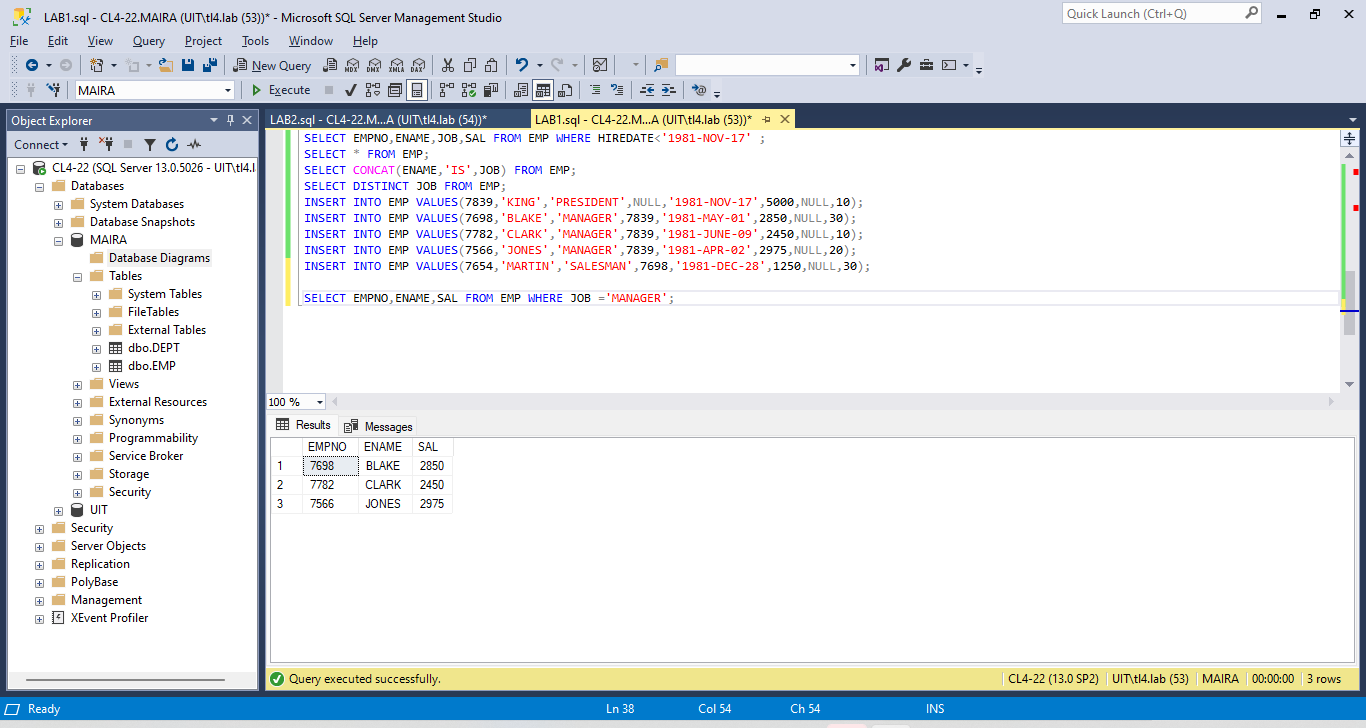
1. Why SQL is called a nonprocedural language?

.SQL, or Structured Query Language, is considered a nonprocedural language because it focuses on what data to retrieve rather than how to retrieve it. Unlike procedural languages, where the user must specify a series of steps for the computer to execute, SQL allows the user to declare the desired result set and let the database management system determine the most efficient way to execute the query. This declarative approach provides flexibility and simplicity in writing queries and allows for optimization of query execution. SQL's nonprocedural nature makes it a popular choice for data analysts and database administrators seeking to easily and efficiently query large sets of data.

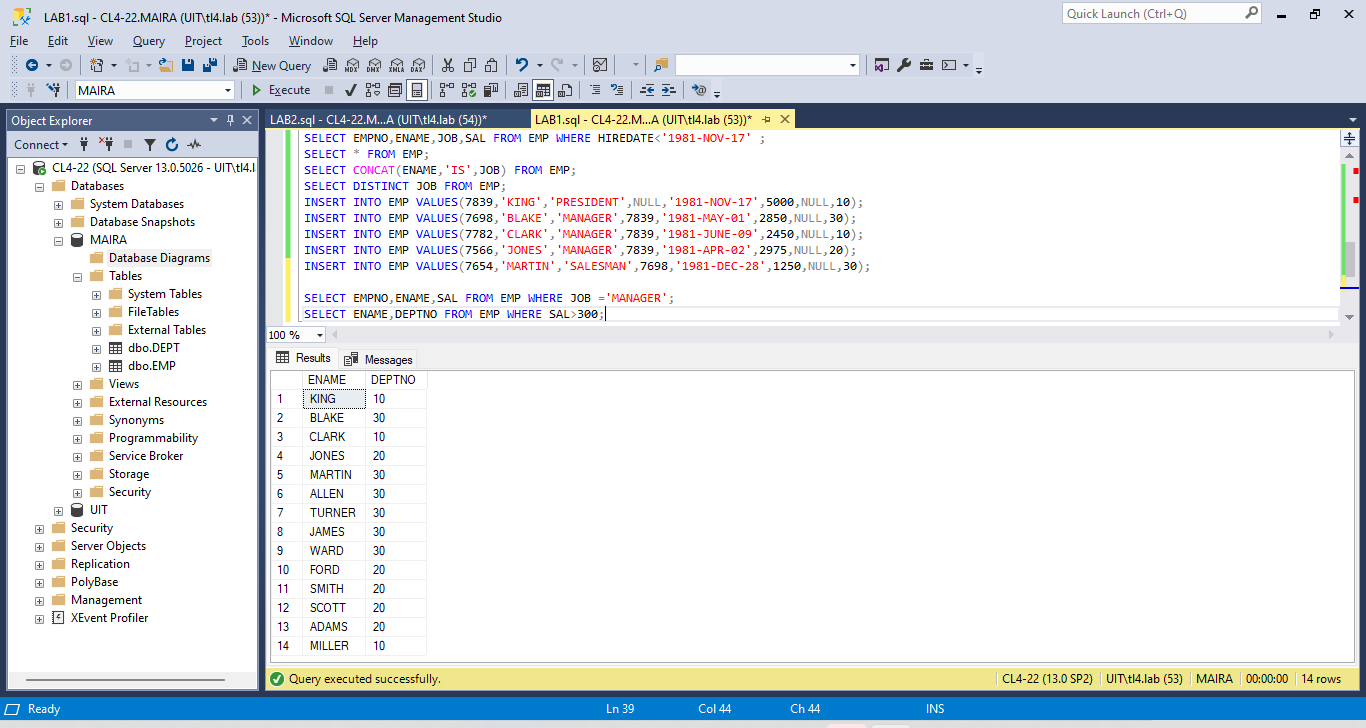
1. Write down a brief history of SQL and explain why its standardization was necessary.

SQL, or Structured Query Language, was first developed in the 1970s at IBM as a way to interact with relational databases. It quickly became popular in the 1980s and 1990s as the use of relational databases grew. In 1986, SQL was standardized by the American National Standards Institute (ANSI), and later by the International Organization for Standardization (ISO) in 1987. This standardization was necessary to ensure that SQL was a universal language that could be used across different database management systems and platforms, and to promote compatibility and interoperability among different systems. Standardization has also allowed for the development of a wide range of tools and software that support SQL, making it a crucial tool for managing and querying large sets of data.

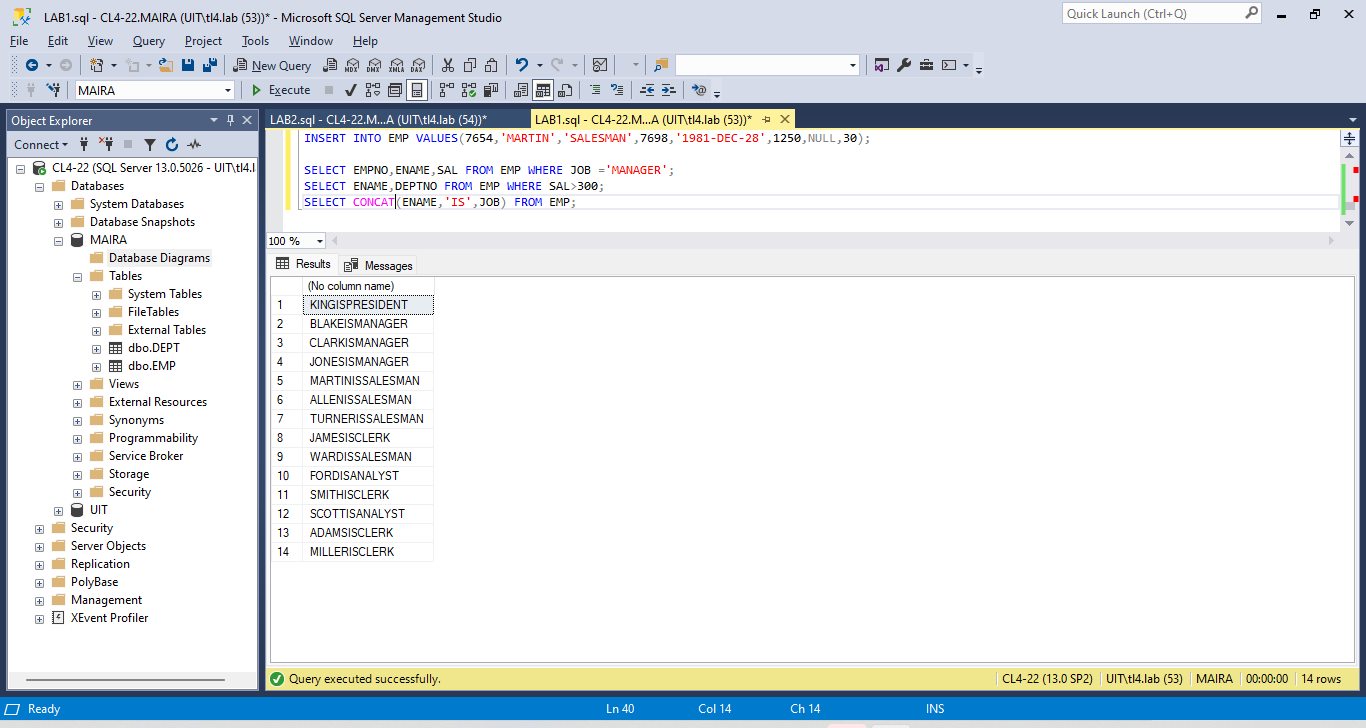
1. Write a query to display the employee number, name and salary of all managers.



1. Write a query to display the name and department number of all employees with existing salary greater than 300 also apply aliasing.



1. Write a query to display the name and job title of employee using concatenation.



\* \* \* \* \*