Azure SQL Database 2 - SSMS를 사용하여 연결 및 쿼리

주신영 bit1010@live.com

이번 실습은 MS의 실습 자료를 참고하였습니다.

빠른 시작: SSMS를 사용하여 Azure SQL Database 또는 Azure SQL Managed Instance 쿼리

지역은 계정에 제한이 없다면 Korea Central을 선택합니다.

SSMS(SQL Server Management Studio) 설치

아래 링크를 통해 SSMS를 설치합니다.

- SSMS(SQL Server Management Studio)

설치 시간이 오래 걸리면 VSCode로 먼저 실습 합니다.

Azure SQL Database 3 - VSCode를 사용하여 연결 및 쿼리

서버 연결 정보 가져오기

데이터베이스에 연결하는 데 필요한 연결 정보를 가져옵니다.

1. Azure Portal에 로그인합니다.
2. 쿼리하려는 **데이터베이스** 또는 **관리되는 인스턴스**로 이동합니다.
3. **개요** 페이지에서 SQL Database의 데이터베이스에 대한 **서버 이름** 옆에 도메인으로 구성된 주소를 마우스로 해당 이름 위를 가리키고 **복사** 아이콘을 선택합니다.

데이터베이스 연결

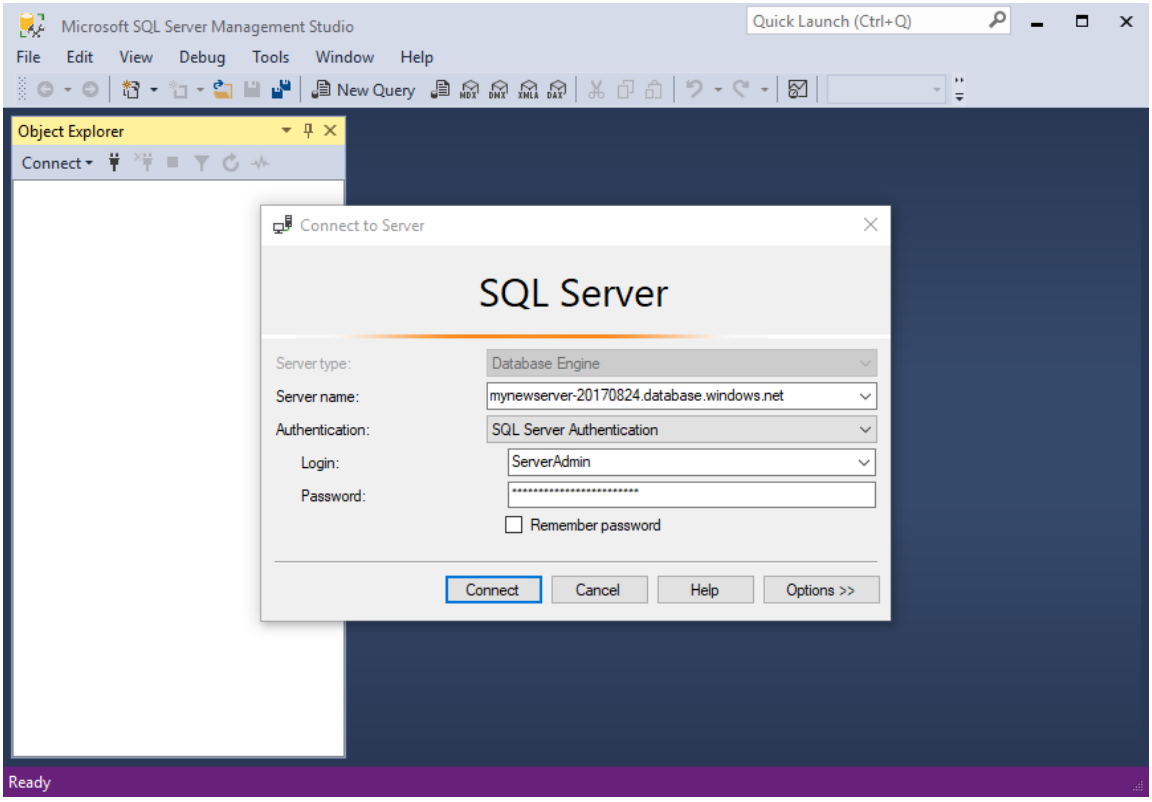
SSMS에서 서버에 연결합니다.

중요

서버는 포트 1433에서 수신 대기합니다. 회사 방화벽 뒤에서 서버에 연결하려면 방화벽에서 이 포트가 열려 있어야 합니다.

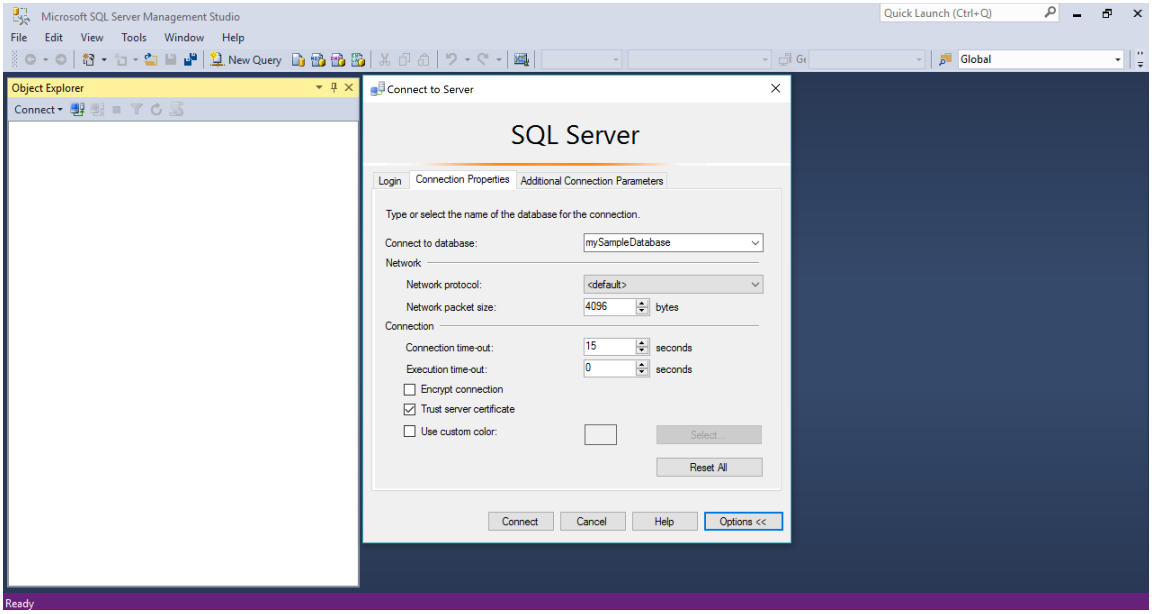
1. SSMS를 엽니다.
2. **서버에 연결** 대화 상자가 표시됩니다. 다음 정보를 입력합니다.

설정	제안 값	설명
서버 유형	데이터베이스 엔진	필수 값.
서버 이름	정규화된 서버 이름	예: **servername.database.windows.net**
인증	SQL Server 인증	**이 자습서에서는 SQL 인증을 사용합니다.**
로그인	서버 관리자 계정 사용자 ID	서버를 만드는 데 사용되는 서버 관리자 계정의 사용자 ID입니다.
암호	서버 관리자 계정 암호	서버를 만드는 데 사용되는 서버 관리자 계정의 암호입니다.



3. **서버에 연결** 대화 상자에서 **옵션**을 선택합니다. **데이터베이스에 연결** 드롭다운 메뉴에서 **mySampleDatabase**를 선택(찾아보기)합니다. 필수 구성 요소 섹션에서 빠른 시작

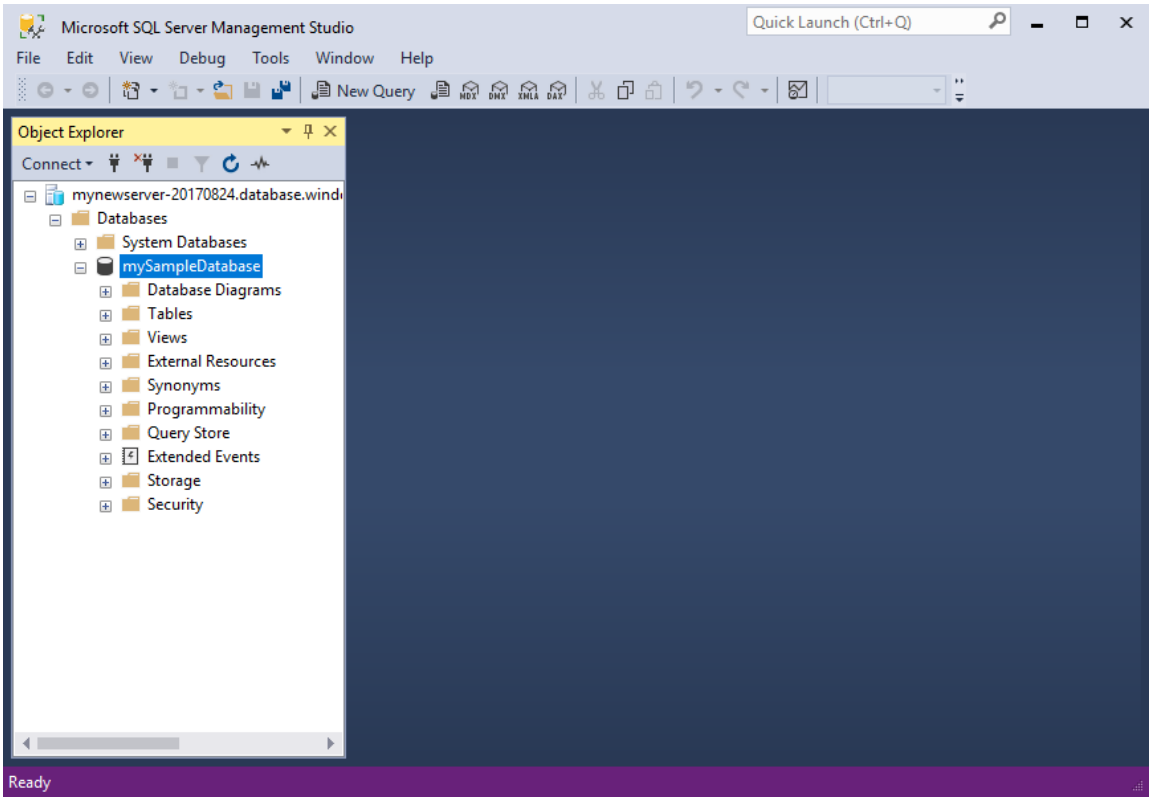
을 완료하면 mySampleDatabase라는 AdventureWorksLT 데이터베이스가 만들어집니다. AdventureWorks 데이터베이스의 작업 복사본 이름이 mySampleDatabase와 다른 경우 이를 대신 선택합니다.



4. **연결**을 선택합니다. 개체 탐색기 창이 열립니다.

> 접속이 안될 경우는 서버 방화벽 설정을 선택하여(SQL Server 리소스의 네트워킹 메뉴에서 방화벽 규칙) 현재 작업하는 네트워크의 IP(클라이언트 IP)가 추가 되었는지 확인합니다.

5. 데이터베이스의 개체를 보려면 **데이터베이스**를 확장한 다음, 데이터베이스 노드를 확장합니다.



쿼리 데이터

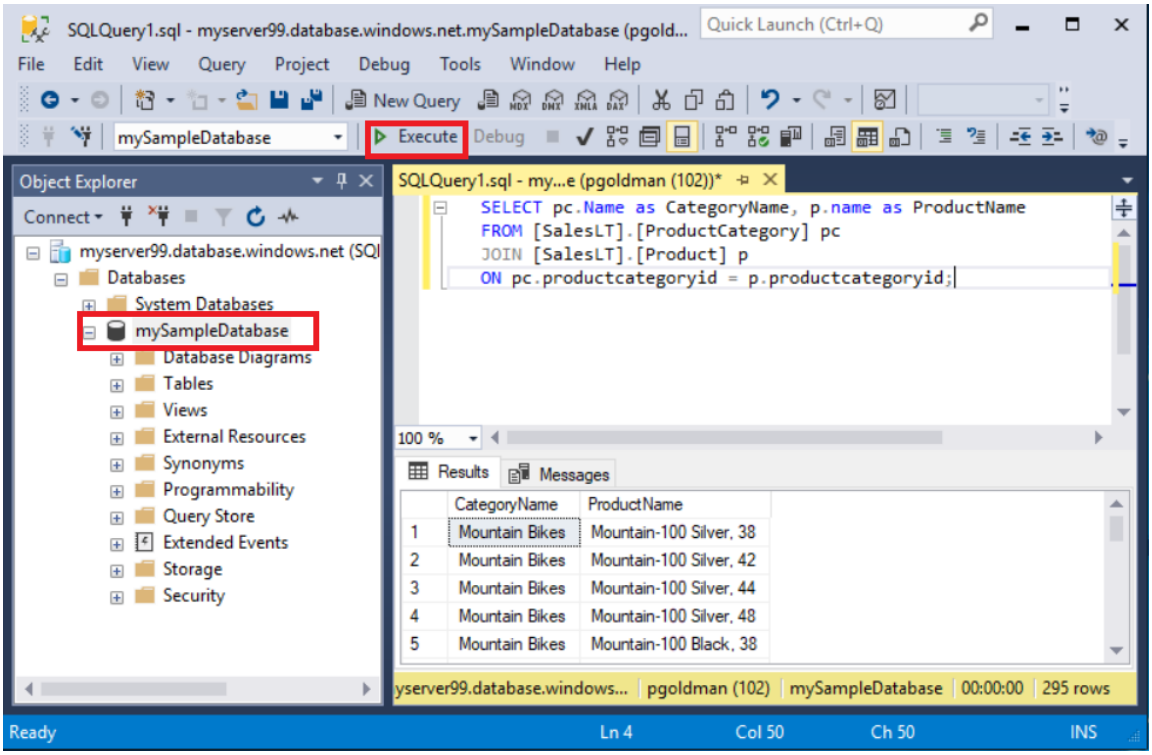
이 SELECT Transact-SQL 코드를 실행하여 범주별 상위 20개 제품을 쿼리합니다.

1. 개체 탐색기에서 **mySampleDatabase**를 마우스 오른쪽 단추로 클릭하고 **새 쿼리**를 선택합니다. 데이터베이스에 연결된 새 쿼리 창이 열립니다.
2. 쿼리 창에서 다음 SQL 쿼리를 붙여 넣습니다.

> 아래 쿼리로 SalesLT.Product, SalesLT.ProductCategory 테이블을 쿼리로 각각 확인해봅니다.
> SELECT * FROM SalesLT.ProductCategory
> SELECT * FROM SalesLT.Product

```
SELECT pc.Name as CategoryName, p.name as ProductName
FROM [SalesLT].[ProductCategory] pc
JOIN [SalesLT].[Product] p
ON pc.productcategoryid = p.productcategoryid;
```

3. 도구 모음에서 **실행**을 선택하여 쿼리를 실행하고 `Product` 및 `ProductCategory` 테이블에서 데이터를 검색합니다.



데이터 삽입

이 INSERT Transact-SQL 코드를 실행하여 `SalesLT.Product` 테이블에서 새 제품을 만듭니다.

1. 이전 쿼리를 삭제하고 다음 쿼리로 바꿉니다.

```
INSERT INTO [SalesLT].[Product]
        ( [Name]
        , [ProductNumber]
        , [Color]
        , [ProductCategoryID]
        , [StandardCost]
        , [ListPrice]
        , [SellStartDate] )
  VALUES
        ('myNewProduct'
        ,123456789
        ,'NewColor'
        ,1
        ,100
        ,100
        ,GETDATE() );
```

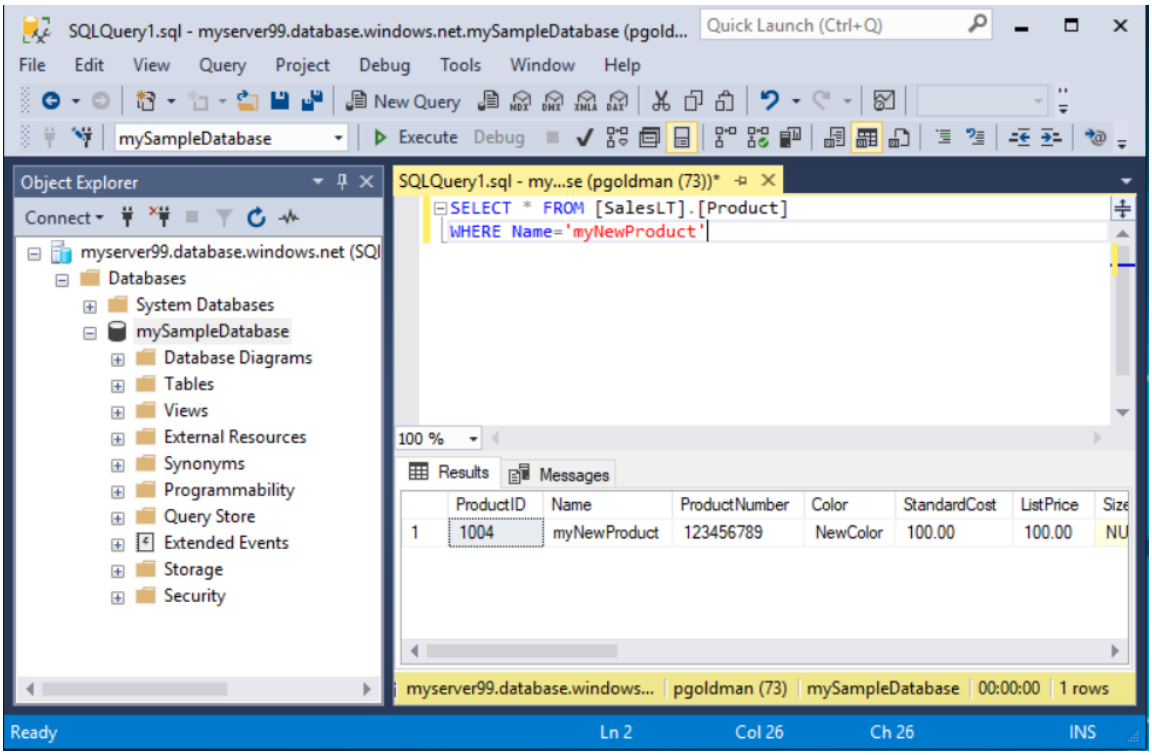
2. **실행**을 선택하여 `Product` 테이블에서 새 행을 삽입합니다. **메시지** 창에 **(영향을 받는 행 1개)** 가 표시됩니다.

결과 보기

1. 이전 쿼리의 실행 결과를 다음 쿼리로 확인합니다.

```
SELECT * FROM [SalesLT].[Product]
WHERE Name='myNewProduct'
```

2. **실행**을 선택합니다. 다음과 같은 결과가 나타납니다.



데이터 업데이트

이 UPDATE Transact-SQL 코드를 실행하여 새 제품을 수정합니다.

1. 이전 쿼리를 이전에 만든 새 레코드를 반환하는 쿼리로 바꿉니다.

```
UPDATE [SalesLT].[Product]
SET [ListPrice] = 125
WHERE Name = 'myNewProduct';
```

2. **실행**을 선택하여 `Product` 테이블에서 지정된 행을 업데이트합니다. **메시지** 창에 **(영향을 받는 행 1개)** 가 표시됩니다.
3. 위 **결과보기**의 쿼리로 확인합니다.

데이터 삭제

이 DELETE Transact-SQL 코드를 실행하여 새 제품을 제거합니다.

1. 이전 쿼리를 다음 쿼리로 바꿉니다.

```
DELETE FROM [SalesLT].[Product]
WHERE Name = 'myNewProduct';
```

2. **실행**을 선택하여 `Product` 테이블에서 지정된 행을 삭제합니다. **메시지** 창에 **(영향을 받는 행 1개)** 가 표시됩니다.
3. 위 **결과보기**의 쿼리로 확인합니다.