

Custom Learnings

Day 4

Creation of Elastic pool: It is used when we know one of the databases will occupy less space.

The screenshot displays the Azure portal interface for an SQL elastic pool. The left sidebar contains navigation options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Settings, Monitoring, and Alerts. The main content area shows the 'Essentials' section for the elastic pool 'idapool_03' (under resource group 'idashellserver03'). Key details include: Resource group (RG_ShellIDA_03), Status (Ready), Location (East US), Subscription (npunext-1680261698581), and Subscription ID (7179a1e4-9824-4360-9eda-2930d7e2dc08). On the right, it lists Server name (idashellserver03.database.windows.net), Pricing tier (Basic: 50 eDTUs), Elastic databases (0 databases), and Elastic database settings (0-5 eDTUs). Below this, there's a 'Show data for last' selector set to '1 hour' and an 'Aggregation type' dropdown set to 'Max'. A 'Resource utilization (idapool_03)' chart is shown at the bottom, with a y-axis ranging from 70% to 100%.

Joins:

1. Inner Join: Returns only matched records from the joining tables.

The screenshot shows the Azure portal's Query editor for an SQL database 'idashelldb' (under resource group 'idashellserver03'). The left sidebar includes navigation options like Overview, Activity log, Tags, Diagnose and solve problems, Query editor (preview), Settings, Data management, and Integrations. The main content area displays the 'Query 1' editor with a SQL query that performs an inner join between 'SalesLT.Product' and 'SalesLT.ProductCategory'. The query is as follows:

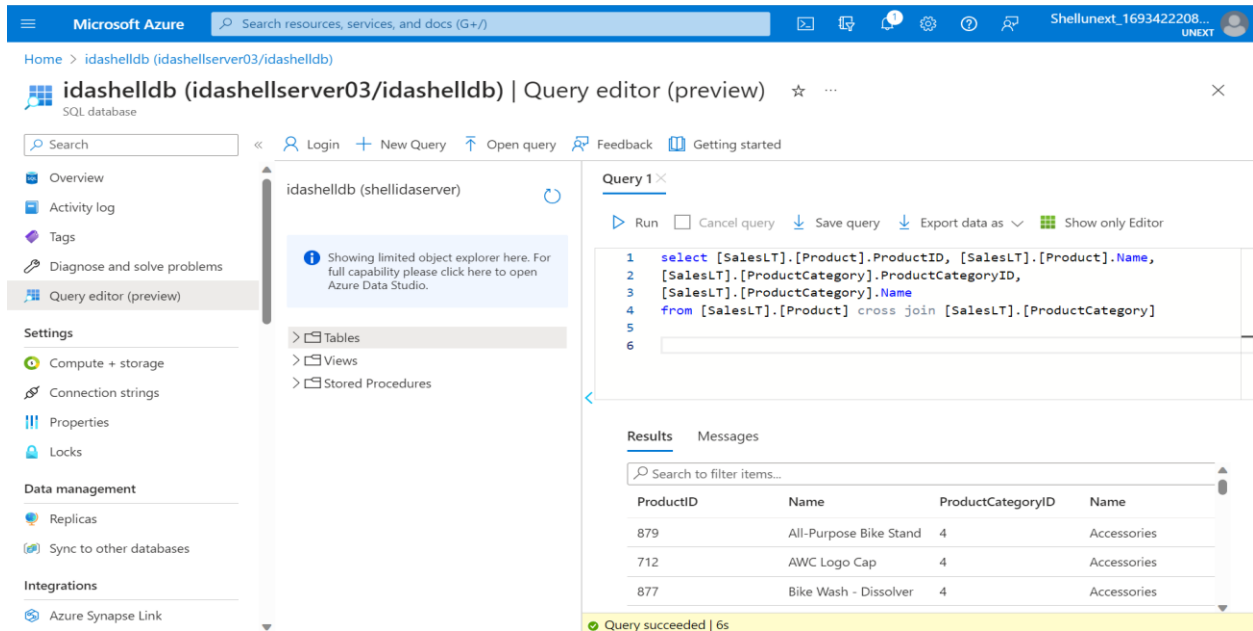
```
1 select [SalesLT].[Product].ProductID, [SalesLT].[Product].Name,  
2 [SalesLT].[ProductCategory].ProductCategoryID, [SalesLT].[ProductCategory].  
3 from [SalesLT].[Product] INNER JOIN [SalesLT].[ProductCategory] on  
4 [SalesLT].[Product].ProductCategoryID = [SalesLT].[ProductCategory].Produc  
5  
6
```

Below the query editor, the 'Results' tab is active, showing a table with four columns: ProductID, Name, ProductCategoryID, and Name. The table contains three rows of data:

| ProductID | Name | ProductCategoryID | Name |
|-----------|--------------------------|-------------------|----------------|
| 771 | Mountain-100 Silver, ... | 5 | Mountain Bikes |
| 772 | Mountain-100 Silver, ... | 5 | Mountain Bikes |
| 773 | Mountain-100 Silver, ... | 5 | Mountain Bikes |

The status bar at the bottom indicates 'Query succeeded | 0s'.

2. Cross Join: Returns the cartesian product of the records from both the tables.



The screenshot shows the Microsoft Azure portal interface for the 'idashelldb' database. The query editor displays a SQL query that performs a cross join between the 'SalesLT.Product' and 'SalesLT.ProductCategory' tables. The query is as follows:

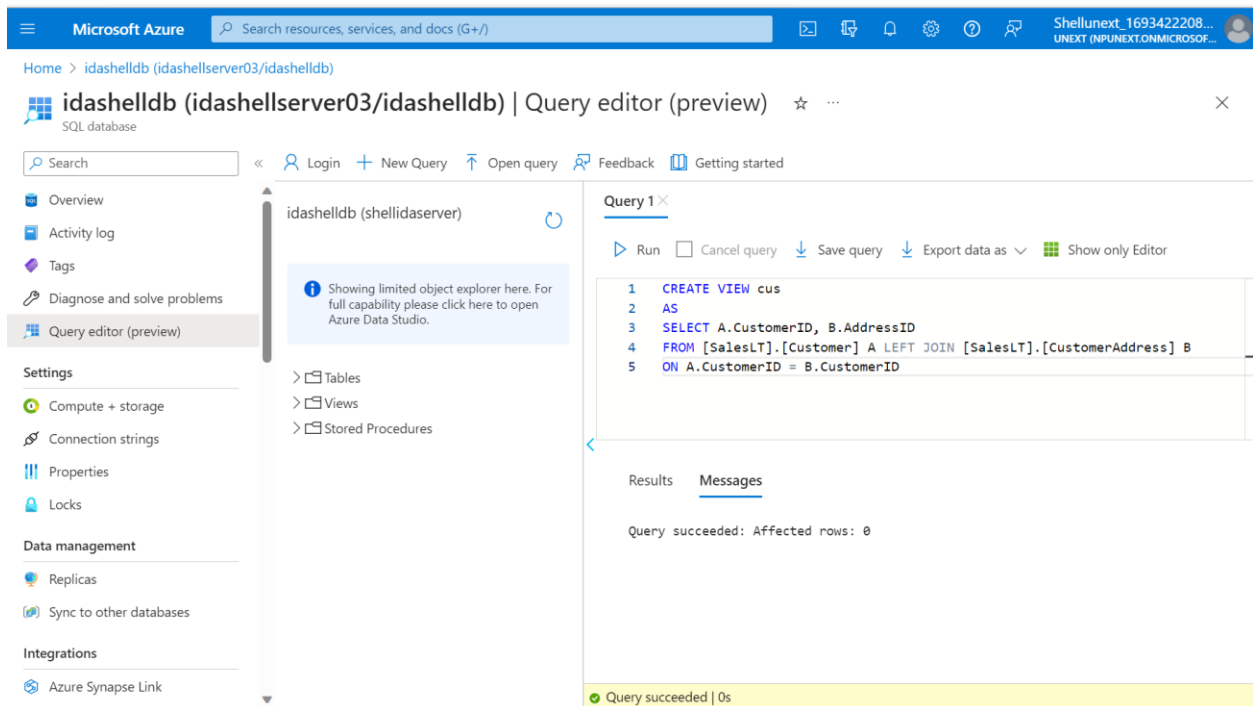
```
1 select [SalesLT].[Product].ProductID, [SalesLT].[Product].Name,  
2 [SalesLT].[ProductCategory].ProductCategoryID,  
3 [SalesLT].[ProductCategory].Name  
4 from [SalesLT].[Product] cross join [SalesLT].[ProductCategory]  
5  
6
```

The results of the query are displayed in a table with 4 columns: ProductID, Name, ProductCategoryID, and Name. The results show 6 rows of data:

| ProductID | Name | ProductCategoryID | Name |
|-----------|------------------------|-------------------|-------------|
| 879 | All-Purpose Bike Stand | 4 | Accessories |
| 712 | AWC Logo Cap | 4 | Accessories |
| 877 | Bike Wash - Dissolver | 4 | Accessories |
| | | | |
| | | | |
| | | | |

The query succeeded, and the results are displayed in a table with 4 columns: ProductID, Name, ProductCategoryID, and Name. The results show 6 rows of data.

3. Left Join: Returns all the records from the left table.



The screenshot shows the Microsoft Azure portal interface for the 'idashelldb' database. The query editor displays a SQL query that performs a left join between the 'SalesLT.Customer' and 'SalesLT.CustomerAddress' tables. The query is as follows:

```
1 CREATE VIEW cus  
2 AS  
3 SELECT A.CustomerID, B.AddressID  
4 FROM [SalesLT].[Customer] A LEFT JOIN [SalesLT].[CustomerAddress] B  
5 ON A.CustomerID = B.CustomerID
```

The results of the query are displayed in a table with 2 columns: CustomerID and AddressID. The results show 0 rows of data.

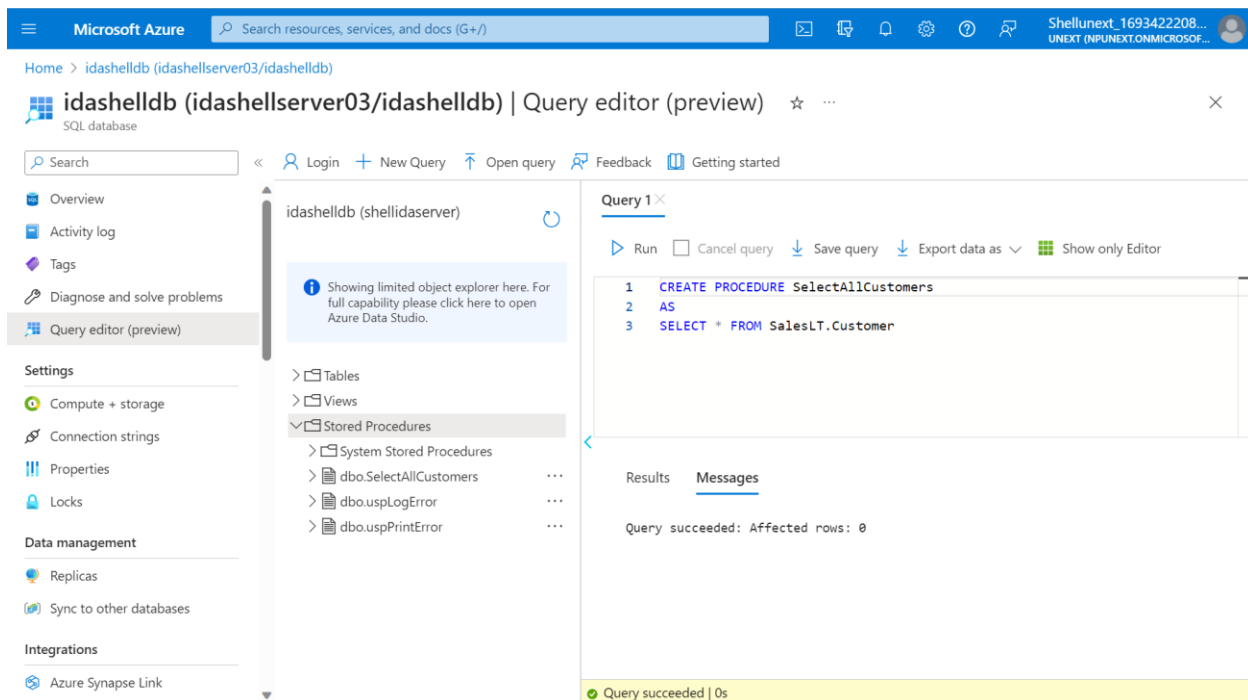
| CustomerID | AddressID |
|------------|-----------|
|------------|-----------|

The query succeeded, and the results are displayed in a table with 2 columns: CustomerID and AddressID. The results show 0 rows of data.

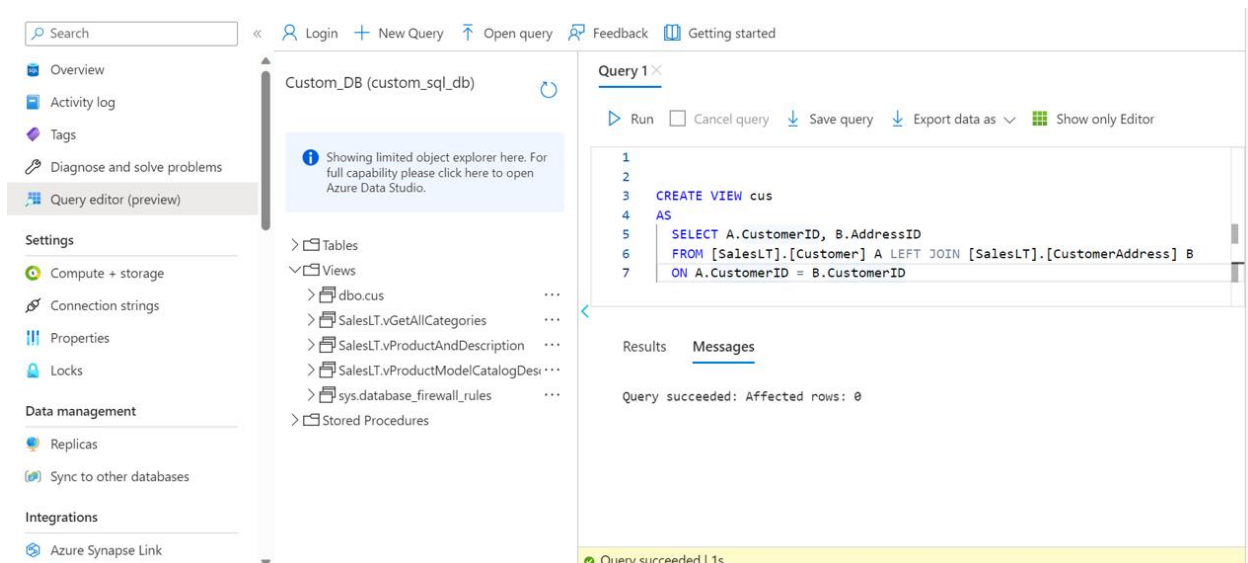
4. Right Join: Returns all the records from the right table.

5. Self Join: Joins with the same table only. Ex: Employee and Manager
6. Full Outer Join: Returns all the records from both the tables.

Stored Procedure: To store a lengthy and complex query we can use stored procedure. We can write parameterized stored procedure. We can put transaction to maintain the atomicity of the queries performed. We also put temporary table in the stored procedure.



Views: View of the table used in cases when limited columns should be available to an user.



Temporary Table: Table which is created during a session and gets automatically created at the end of the session.

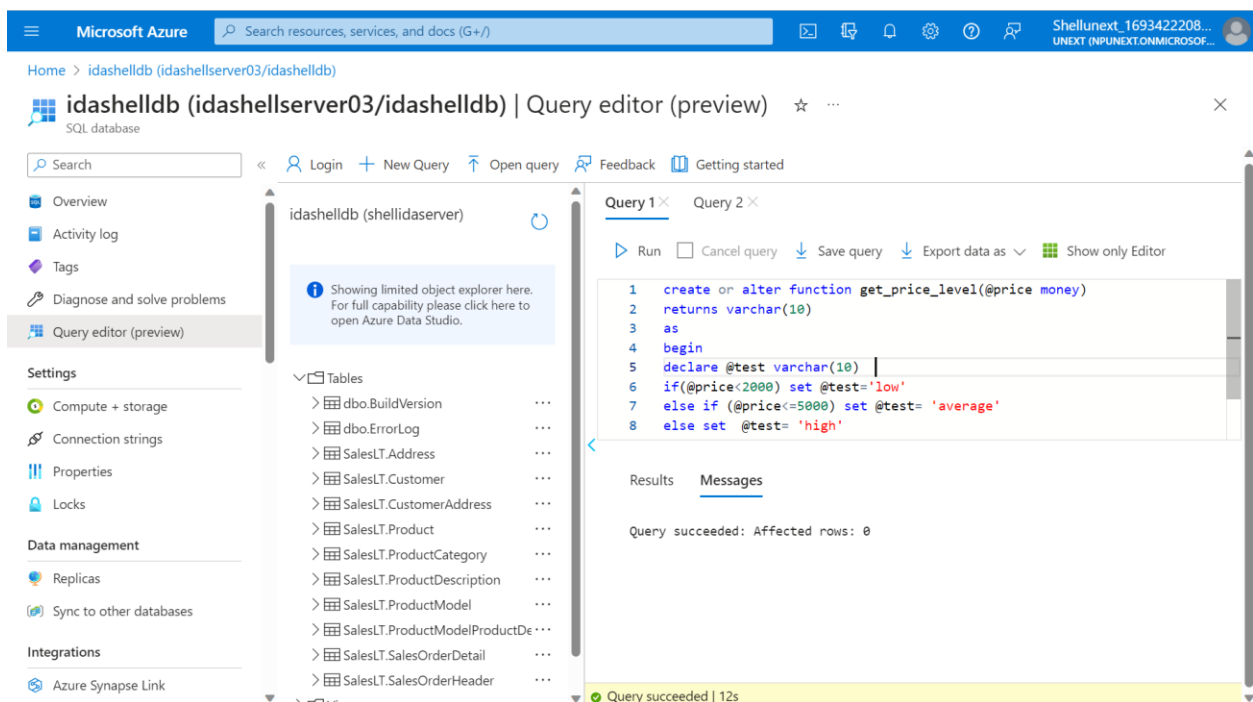
Select * into #testtmp from tablename;

Select * from #testtmp;

We run both the statements at once because it is considered as one session.

Functions: Piece of code which is going to decrease the code overhead and increase code reusability.

Sum, Avg etc. are system functions. We can create our own user-defined functions.



2 types:

Scalar : Sum, Avg which gives one single value.

Table Values: Gives output in the form of rows and columns.

#scalar function

```
CREATE FUNCTION dbo.GetSalaryCategory(@salary DECIMAL(10, 2)) RETURNS NVARCHAR(10)
```

```
AS
```

```
BEGIN
```

```
    DECLARE @category NVARCHAR(10)
```

```

IF @salary > 2000
    SET @category = 'High'
ELSE IF @salary < 1000
    SET @category = 'Low'
ELSE
    SET @category = 'NA'

RETURN @category
END;
GO

```

```

CREATE TABLE Employee (
    EmployeeID INT PRIMARY KEY,
    FirstName NVARCHAR(50),
    LastName NVARCHAR(50),
    Salary DECIMAL(10, 2),
    SalaryCategory NVARCHAR(10)
);

```

```

INSERT INTO Employee (EmployeeID, FirstName, LastName, Salary, SalaryCategory)
VALUES
    (1, 'John', 'Doe', 2500.00, dbo.GetSalaryCategory(2500.00)),
    (2, 'Jane', 'Smith', 800.00, dbo.GetSalaryCategory(800.00)),
    (3, 'Bob', 'Johnson', 1500.00, dbo.GetSalaryCategory(1500.00)),
    (4, 'Alice', 'Williams', 3000.00, dbo.GetSalaryCategory(3000.00));

```

Tabled Valued FUction

```

CREATE FUNCTION dbo.GetEmployeesBySalaryRange (@minSalary DECIMAL(10, 2), @maxSalary
DECIMAL(10, 2))
RETURNS TABLE
AS
RETURN (
    SELECT EmployeeID, FirstName, LastName, Salary
    FROM Employee
    WHERE Salary BETWEEN @minSalary AND @maxSalary
);

```

Sub-Query: Query inside a query. Sub-Query is executed first, then the main query.

Microsoft Azure Search resources, services, and docs (G+/)

Home > idashelldb (idashellserv03/idashelldb)

idashelldb (idashellserv03/idashelldb) | Query editor (preview)

Search

Overview Activity log Tags Diagnose and solve problems Query editor (preview) Settings Compute + storage Connection strings Properties Locks Data management Replicas Sync to other databases Integrations Azure Synapse Link

SalesLT.Product

- ProductID (PK, int, not null)
- Name (Name, not null)
- ProductNumber (nvarchar, not null)
- Color (nvarchar, null)
- StandardCost (money, not null)
- ListPrice (money, not null)
- Size (nvarchar, null)
- Weight (decimal, null)
- ProductCategoryID (int, null)
- ProductModelID (int, null)
- SellStartDate (datetime, not null)
- SellEndDate (datetime, null)
- DiscontinuedDate (datetime, null)
- ThumbNailPhoto (varbinary, null)
- ThumbNailPhotoFileName (nvarchar, null)
- rowguid (uniqueidentifier, not null)

Query 1 X Query 2 X

Run Cancel query Save query Export data as Show only Editor

```

1 select max(StandardCost) from [SalesLT].[Product]
2 where StandardCost < (select max(StandardCost) from [SalesLT].[Product])
3

```

Results Messages

Search to filter items...

1912.1544

Query succeeded | 1s

Performing inner join on 3 tables.

Microsoft Azure Search resources, services, and docs (G+/)

Home > idashelldb (idashellserv03/idashelldb)

idashelldb (idashellserv03/idashelldb) | Query editor (preview)

Search

Overview Activity log Tags Diagnose and solve problems Query editor (preview) Settings Compute + storage Connection strings Properties Locks Data management Replicas Sync to other databases Integrations Azure Synapse Link

SalesLT.CustomerAddress

SalesLT.Product

- ProductID (PK, int, not null)
- Name (Name, not null)
- ProductNumber (nvarchar, not null)
- Color (nvarchar, null)
- StandardCost (money, not null)
- ListPrice (money, not null)
- Size (nvarchar, null)
- Weight (decimal, null)
- ProductCategoryID (int, null)
- ProductModelID (int, null)
- SellStartDate (datetime, not null)
- SellEndDate (datetime, null)
- DiscontinuedDate (datetime, null)
- ThumbNailPhoto (varbinary, null)
- ThumbNailPhotoFileName (nvarchar, null)

Query 1 X Query 2 X

Run Cancel query Save query Export data as Show only Editor

```

1 select * from [SalesLT].[Product] p
2 inner join
3 [SalesLT].[ProductCategory] pc on pc.ProductCategoryID = p.ProductCategoryID
4 inner join
5 [SalesLT].[SalesOrderDetail] od on od.ProductID = p.ProductID
6
7

```

Results Messages

Search to filter items...

| ProductID | Name | ProductNumber |
|-----------|-----------------------|---------------|
| 707 | Sport-100 Helmet, Red | HL-U509-R |
| 707 | Sport-100 Helmet, Red | HL-U509-R |
| 707 | Sport-100 Helmet, Red | HL-U509-R |

Query succeeded | 8s

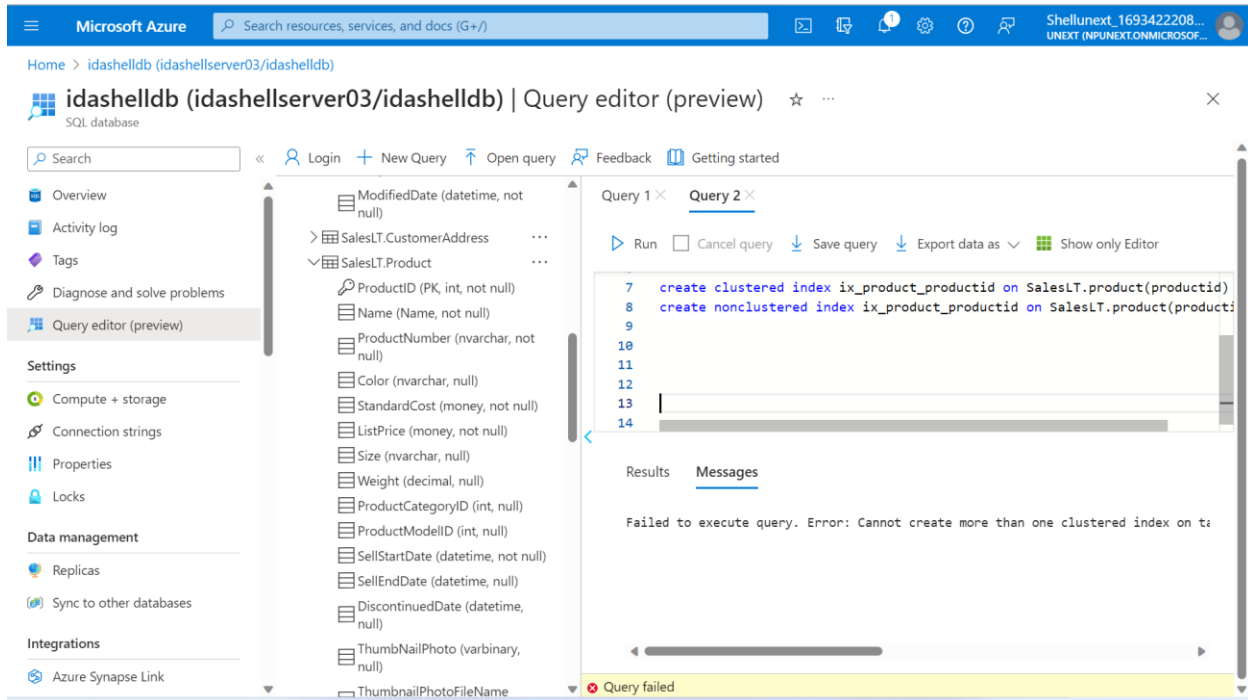
Union and Union All: Joins two table vertically. Should have same number of columns, same column names and same data type. Union all will accept duplicates and Union will remove the duplicates.

Intersect: Only common records are returned.

Index: 2 fetch data faster.

3 types:

1. Clustered index: On primary key only, it orders the rows and creates clusters with pointers.
2. Non-Clustered Index: It will not order rows but creates groups with pointers.



Microsoft Azure | Search resources, services, and docs (G+/)

Home > idashelldb (idashelldbserver03/idashelldb)

idashelldb (idashelldbserver03/idashelldb) | Query editor (preview)

Search | Login | New Query | Open query | Feedback | Getting started

Overview | Activity log | Tags | Diagnose and solve problems | Query editor (preview)

Settings

- Compute + storage
- Connection strings
- Properties
- Locks

Data management

- Replicas
- Sync to other databases

Integrations

- Azure Synapse Link

idashelldb (shellidserver)

Showing limited object explorer here. For full capability please click here to open Azure Data Studio.

- Tables
- Views
- Stored Procedures

Query 1 X | Query 2 X

Run | Cancel query | Save query | Export data as | Show only Editor

```

7 create clustered index ix_product_productid on SalesLT.product(productid)
8 create nonclustered index ix_product_productid on SalesLT.product(productid)
9
10
11
12
13
14

```

Results | Messages

Query succeeded: Affected rows: 0

Query succeeded | 1s

Microsoft Azure | Search resources, services, and docs (G+/)

Home > idashelldb (idashelldbserver03/idashelldb)

idashelldb (idashelldbserver03/idashelldb) | Query editor (preview)

Search | Login | New Query | Open query | Feedback | Getting started

Overview | Activity log | Tags | Diagnose and solve problems | Query editor (preview)

Settings

- Compute + storage
- Connection strings
- Properties
- Locks

Data management

- Replicas
- Sync to other databases

Integrations

- Azure Synapse Link

idashelldb (shellidserver)

Showing limited object explorer here. For full capability please click here to open Azure Data Studio.

- Tables
- Views
- Stored Procedures

Query 1 X

Run | Cancel query | Save query | Export data as | Show only Editor

```

1 create nonclustered index ix_product_productid on [SalesLT].[Product](productid)
2

```

Results | Messages

Query succeeded: Affected rows: 0

Query succeeded | 0s

3. Column Stored Index: We apply on a column to fetch data faster from a column.

Heap is default that means no index.