# Table of Contents

1. Introduction to NosDB ....................................................................................................................... 8

## Getting Started with NosDB ........................................................................................................... 9

2. Open Firewall TCP Port for Clients ................................................................................................. 10
   2.1. Configuration Server Port ........................................................................................................... 10
   2.2. Management Port ....................................................................................................................... 10
   2.3. Distributor Port ......................................................................................................................... 11

3. Accessing Database Cluster across Domains .................................................................................. 12
   3.1. Using Runas Command .......................................................................................................... 12

## Configuring Database Clusters and Shards ..................................................................................... 13

4. Database Clusters ............................................................................................................................. 14
   4.1. Create Database Cluster .......................................................................................................... 14
   4.2. Connect to Existing Database Cluster ..................................................................................... 20
   4.3. Disconnect from Cluster ........................................................................................................... 21
   4.4. Remove Database Cluster ........................................................................................................ 21

5. Shards ................................................................................................................................................ 22
   5.1. Add Shard ............................................................................................................................... 22
   5.2. Remove Shard .......................................................................................................................... 24
   5.3. Start Shard ............................................................................................................................... 26
   5.4. Start All Shards ....................................................................................................................... 27
   5.5. Stop Shard ............................................................................................................................... 27
   5.6. Shard Nodes ............................................................................................................................. 29
       5.6.1. Add Node to Shard .......................................................................................................... 29
       5.6.2. Remove Node from Shard .............................................................................................. 30
       5.6.3. Start Node on Shard ...................................................................................................... 31
       5.6.4. Stop Node on Shard ...................................................................................................... 32

6. Configuring Shard Properties ......................................................................................................... 33
   6.1. Priority of Shard Node ............................................................................................................. 33
   6.2. Management Port ..................................................................................................................... 34
   6.3. Shard Heartbeat ....................................................................................................................... 35
   6.4. Node Heartbeat ....................................................................................................................... 37
   6.5. Replication ............................................................................................................................... 39

7. Configure MapReduce Properties ................................................................................................. 41
Configuring Databases ........................................................................................................... 43

6. Databases .......................................................................................................................... 44
   6.1. Create New Database .................................................................................................. 44
   6.2. Configure Database Properties ................................................................................ 45
      6.2.1. Journal Settings ............................................................................................... 45
      6.2.2. Storage Settings .............................................................................................. 47
      6.2.3. Cache Settings ................................................................................................. 48
      6.2.4. Attachments ..................................................................................................... 48
   6.3. Delete Database ........................................................................................................ 50

7. Collections ....................................................................................................................... 51
   7.1. Create Collection ..................................................................................................... 51
      7.1.1. Composite Key ............................................................................................... 52
      7.1.2. Hash Based Distribution ............................................................................... 53
      7.1.3. Range Based Distribution ............................................................................. 54
      7.1.4. No Distribution Strategy ............................................................................... 56
   7.2. Delete Collection ...................................................................................................... 57
   7.3. Manage Ranges ........................................................................................................ 58
      7.3.1. Add Ranges ..................................................................................................... 58
      7.3.2. Redefine Ranges ............................................................................................ 58

8. Indexes ............................................................................................................................. 60
   8.1. Conceptual Overview ............................................................................................... 60
      8.1.1. Index Creation ................................................................................................. 60
      8.1.2. Index Types .................................................................................................... 61
      8.1.3. Index Behaviors .............................................................................................. 63
   8.2. Create Index ............................................................................................................. 64
   8.3. Drop Index ............................................................................................................... 66

9. CLR Triggers ..................................................................................................................... 67
   9.1. Create Trigger ......................................................................................................... 67
   9.2. Delete Trigger ......................................................................................................... 70

10. CLR Functions ............................................................................................................... 70
    10.1. Create Function .................................................................................................... 70
    10.2. Drop Function ..................................................................................................... 73

11. Data Expiration ............................................................................................................. 74
    11.1. Expiration Properties ......................................................................................... 75
17.1. Configure SSL Certificate in NosDB ................................................................. 102
17.2. Configure SSL Certificate on REST Server .................................................... 105
17.3. Configure SSL Certificate on REST Client ....................................................... 108

Database Administration ...................................................................................... 110
18. Backup Database ............................................................................................. 111
   18.1. Conceptual Overview ...................................................................................... 111
       18.1.1. Full backup .............................................................................................. 111
       18.1.2. Differential backup ................................................................................. 112
   18.2. Create Database Backup ............................................................................... 113
       18.2.1. Distributed Backup Storage ................................................................... 113
       18.2.2. Consolidated Storage .............................................................................. 114
19. Restore Database ............................................................................................. 116
   19.1. Conceptual Overview ...................................................................................... 116
       19.1.1. Full Restore .............................................................................................. 116
       19.1.2. Differential Restore ................................................................................. 117
   19.2. Restore Backed up Database ........................................................................ 117
       19.2.1. Distributed Storage ................................................................................. 118
       19.2.2. Consolidated Storage .............................................................................. 119
20. View Task Status ............................................................................................. 121
21. View Task History ........................................................................................... 122
22. Disable Differential Logs .................................................................................. 123
23. Configure Multiple NICs .................................................................................. 125
24. Deploy Providers ............................................................................................... 126
25. Launch NosDB PowerShell ................................................................................. 129
26. Configure Offline Tasks .................................................................................... 129
27. Using SQL Queries .......................................................................................... 131
   27.1. JSON String Output ....................................................................................... 134
   27.2. Tabular Output ............................................................................................. 135
   27.3. Copying Output ............................................................................................. 136
   27.4. Select Top 1000 Rows ................................................................................. 137
28. Convert Standalone to Clustered Database ...................................................... 139

Database Monitoring .......................................................................................... 141
29. NosDB Monitor ............................................................................................... 142
   29.1. Starting NosDB Monitor ................................................................................. 142
29.1.1. From NosDB Management Studio ................................................................. 142
29.1.2. From Program Group .................................................................................. 143
29.2. Using Server Dashboard ................................................................................. 145
29.3. Using Report-View Dashboard ....................................................................... 146
29.4. Custom Dashboard .......................................................................................... 147
30. NosDB Counters ............................................................................................... 150
31. NosDB Management Studio Statistics ............................................................... 152
32. Windows Performance Monitor ....................................................................... 153
33. Windows Event Logging ................................................................................... 158
34. NosDB Logs ...................................................................................................... 159
   34.1. NosDB Log Viewer ....................................................................................... 160
      34.1.1. Open Logs ............................................................................................ 160
      34.1.2. Export Log Entries .............................................................................. 161
      34.1.3. Search Specific Logs ......................................................................... 162
      34.1.4. Split Logs based on Field .................................................................... 163
      34.1.5. Reload Log Files ................................................................................. 164
      34.1.6. Clear All Logs ..................................................................................... 164
      34.1.7. Toggle Bookmark on Log Entry ............................................................ 164
      34.1.8. Bookmark Navigation ........................................................................ 165
      34.1.9. Settings ............................................................................................... 165
   34.2. NosDB Monitor Logging .............................................................................. 167

Configuring NosDB REST API .............................................................................. 169
35. Configuring Internet Information Services (IIS) for REST API ....................... 170
36. Installing REST API Package on IIS ................................................................. 172
   36.1. Adding New Application Pool .................................................................... 172
   36.2. Installing Package to Existing Pool ............................................................ 174
   36.3. Adding Application to Pool ....................................................................... 175

Using Microsoft Power BI with NosDB ................................................................. 177
37. Using Microsoft Power BI with NosDB ............................................................ 178
   37.1. Selecting NosDB as Data Source ............................................................... 179
   37.2. Loading NosDB Data into Power BI .......................................................... 181
   37.3. Loading Multiple Collections ................................................................... 183
      37.3.1. Loading NosDB data ......................................................................... 183
      37.3.2. Loading data from recently added sources ....................................... 184
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>37.3.3.</td>
<td>Creating table within Power BI</td>
<td>185</td>
</tr>
<tr>
<td>37.2.</td>
<td>Visualize Data</td>
<td>186</td>
</tr>
<tr>
<td>37.2.1.</td>
<td>Selecting Visualization Type</td>
<td>186</td>
</tr>
<tr>
<td>37.2.2.</td>
<td>Selecting Legend or Axis</td>
<td>186</td>
</tr>
<tr>
<td>37.2.3.</td>
<td>Selecting Values</td>
<td>187</td>
</tr>
<tr>
<td>37.3.</td>
<td>Managing Relationships</td>
<td>188</td>
</tr>
<tr>
<td>37.4.</td>
<td>Querying NosDB Data</td>
<td>191</td>
</tr>
<tr>
<td>37.4.1.</td>
<td>Filter based Querying</td>
<td>191</td>
</tr>
<tr>
<td>37.4.2.</td>
<td>Querying through Query Editor</td>
<td>192</td>
</tr>
</tbody>
</table>
1. Introduction to NosDB

Welcome to NosDB! NosDB is a schema-less and scalable NoSQL database solution to handle ad-hoc querying on huge amounts of real-time, unstructured data. As NosDB scales out to accommodate the rapidly increasing volume of your data, it applies robust data distribution strategies to ensure availability and fault tolerance at all times. Keeping in mind the suitability of NosDB for Big Data applications, MapReduce and Aggregation support has also been introduced to dramatically enhance performance due to parallel processing.

NosDB features and tools are designed to be tuned flexibly into applications of any size – from small to enterprise-wide global installations.

Support

NosDB provides various sources of technical support. Please refer to Alachisoft’s Support page to select a support resource you find suitable for your issue.

To request additional features in the future, or if you notice any discrepancy regarding this document, please drop an email at support@alachisoft.com.

Document Conventions

The following conventions in text have been used throughout this document:

<table>
<thead>
<tr>
<th>Convention</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>bold</strong></td>
<td>Specifies terms of importance for the reader.</td>
</tr>
<tr>
<td><em>italics</em></td>
<td>Specifies any application names.</td>
</tr>
<tr>
<td>📝</td>
<td>Specifies additional and vital information for the user.</td>
</tr>
</tbody>
</table>
Getting Started with NosDB

NosDB Administrators’ Guide
2. Open Firewall TCP Port for Clients

2.1. Configuration Server Port

If a firewall is enabled between the clients and the configuration server, then you must open a TCP port (default 9950) in the firewall for the client to communicate with the configuration server through the connection string.

"Data Source=127.0.0.1; Port=9950; Database=northwind; Local Instance=false;"

If you want to manually configure the Config Server Port, changes must be made on all servers running the Configuration Service. The port can be modified in the "NosDB.ConfigurationService.exe.config" file located in "[InstallDir]/bin/service":

```xml
<appSettings>
  ...
  <add key="ConfigServerPort" value="9950"/>
</appSettings>
```

2.2. Management Port

The configuration server further communicates with the Database Service using the Management Port. The default Management Port for the Database Service is 9960.

If you want to manually configure the Management Server Port, changes must be made on all servers running the Configuration Service. The port can be modified in the "NosDB.ConfigurationService.exe.config" file located in "[InstallDir]/bin/service":

```xml
<appSettings>
  ...
  <add key="ManagementServerPort" value="9960"/>
</appSettings>
```

However, this port is being used by the database service so it also has to be modified accordingly in the "NosDB.DatabaseService.exe.config" file located in "[InstallDir]/bin/service":

```xml
<appSettings>
  ...
  <add key="ManagementServerPort" value="9960"/>
</appSettings>
```
Every time the configuration files (of Configuration Service and Database Service) are modified, the corresponding services must be restarted.

2.3. Distributor Port

Any client for NosDB which is not .NET (Java/Node.JS), needs to connect to the Distributor Service (NosDistributorSvc). The default port for the Distributor Service is 9970.

If you want to manually configure the Distributor Service Port, changes must be made on all servers running the Distributor Service. The port can be modified in the "NosDB.DistributorService.exe.config" file located in [InstallDir]/bin/service:

```xml
<appSettings>
  ...
  <add key="Port" value="9970"/>
  <add key="IP" value="200.0.0.1"/>
  ...
</appSettings>
```
3. Accessing Database Cluster across Domains

To access a cluster across a domain, different permissions are required respective to the domain. This can also result in restricted access because of Windows Authentication, along with a similar limitation to view the PerfMon counters across a different domain.

3.1. Using Runas Command

Keeping in mind the aforementioned limitations, NosDB has incorporated the Runas command-line tool by Microsoft into its management modules so to access the cluster across domains, the management module specified with a "(Runas)" can be launched and used with ease:

- Launch `psmgmt.runas.cmd` from `[InstallDir]\bin\tools`.
- Enter the **User ID** of the login you want to access in the format `user@domain` or `domain\user`.
- Enter the **password** for the ID.
- You can now run the commands specific to the domain with permissions granted to the user you have logged on as.

Monitoring PerfMon Counters across Domains

- Follow the steps to access a database cluster as mentioned above.

- Run the following command through Command Prompt to launch PerfMon:

  ```
  runas /env /netonly /user:user@domain perfmon.exe
  ```

- You can now monitor PerfMon Counters for a cluster in the domain you are logged in to.
Configuring Database Clusters and Shards

NosDB Administrators’ Guide
4. Database Clusters

This feature is only available in NosDB Enterprise Edition.

A database cluster is a set of two interconnected server nodes forming a cluster of servers which behave as a single unit from the outside. Database clusters are usually preferred in scenarios where performance and scalability are needed.

4.1. Create Database Cluster

Run NosDB Management Studio from the configuration server and create a new database cluster through the following steps:

- Right click on Database Cluster in Object Explorer and select New.

- Create Database Cluster window will appear.

- Specify the Name of the cluster.

- Select a Mode of authentication for security:
  - If Windows Authentication is selected, the domain user is automatically authenticated.
For **NosDB Authentication**, specify the **User name** and **Password** for the cluster.
Click Next.

Configure Shard window appears.

Specify the Shard Name and Port.

Click Next.
• **Add Node to Shard** window appears.

• Select the node(s) to become a part of the shard by clicking the > button.

• For **Windows Authentication**, the node is simply added with Windows credentials.

• For **NosDB Authentication**, the following window appears where the credentials for the node can be specified. Note that the credentials provided for the shard are Windows account credentials, belonging to the same domain (for remote shard) or local machine (for single shard).
Specify the Priority for each node.

Click Next.

Cluster Summary window appears. The shard and its corresponding nodes are displayed. You can add another shard or add it later as required.
- Click **Finish**.
4.2. Connect to Existing Database Cluster

- Open NosDB Management Studio.
- Connect to Server window appears.
- Enter the server name which will be used to connect to the database cluster.
- Specify the type of Authentication for the connection:
  - If Windows Authentication is selected, it will automatically log in with the Windows credentials.
  - If NosDB Authentication is selected, enter the username and password.
• Click Connect.

You can now perform all operations on NosDB, through the database cluster or standalone database.

4.3. Disconnect from Cluster

• Right click on the cluster under Database Clusters and select Disconnect.

4.4. Remove Database Cluster

• Right click on the cluster under Database Clusters and select Remove.
5. Shards

5.1. Add Shard

- Right click on Shards in Object Explorer and select Add Shard.

- Configure Shard window appears.
- Specify the shard name and a unique port for the shard.
- Click Next.
- **Add Node** wizard will appear.
- Specify the node name/IP address to add to the shard from the list of the network.
- Click > to add the selected node.

In case of NosDB Authentication, all nodes being added with the user name **admin must** have the same password.

- The **Selected Nodes** window on the right displays the added nodes and their respective priorities which are configurable from the drop-down menu.
5.1. Add Shard

To add a shard to the cluster:

- Click Finish.
- The shard is added with the respective node.

5.2. Remove Shard

To remove a shard from the cluster:

- Right click on the shard name in Object Explorer and select Remove.
• **Remove Shard** window appears, offering the option for *forceful removal*. In case the option is not selected, graceful removal will be initiated as default.

• Click **OK**.
5.3. Start Shard

In case you do not want all shards to be started, you can start nodes of a single shard as well:

- Right click on the **shard name** in **Object Explorer** and select **Start**.

  ![Object Explorer](image1)

- The **Shard Status** window appears, which shows the progressing status of the shard being started.

  ![Shard Status](image2)

- Once started, the icon of the shard node(s) will change from black to sky blue, indicating a started shard. In case of primary node, a yellow star appears over the icon.

  ![Shard Icon](image3)
5.4. Start All Shards

For multiple shards, all the shards can be started simultaneously.

- Right click on Shards in Object Explorer and select Start Shards.

- All shards are started accordingly.

5.5. Stop Shard

In case you do not want all shards to be stopped, you can stop the nodes a single shard as well:

- Right click on the shard name in Object Explorer and select Stop.
The **Shard Status** window appears, showing the progressing status of the shard.

The icon of the shard node will change from sky blue to black, indicating a stopped shard.
5.6. Shard Nodes

5.6.1. Add Node to Shard

To add more nodes to the shard,

- Right click on the **shard name** in **Object Explorer** and select **Add Node**.

![Add Node window](image)

- **Add Node** window appears.
- Specify the **node name/IP address** of the node to be added and its priority.

In case of NosDB Authentication, all nodes being added with the user name **admin** **must** have the same password.

- Click **OK**.
5.6.2. Remove Node from Shard

To remove node from a shard,

- Right click on the node to be removed in Object Explorer and select Remove.
  - NosDB Management Studio will ask for confirmation. Click Yes.
  - The nodes’ list will be refreshed and the node will be removed from the cluster.
5.6.3. Start Node on Shard

- Right click on the node to be started in Object Explorer and select Start.

- The Node Status window appears, showing the shard name, node IP and progressing status.
5.6.4. Stop Node on Shard

- Right click on the node to be stopped in Object Explorer and select Stop.
• The icon of the shard node will change from sky blue to black, indicating a stopped shard.

5.7. Configuring Shard Properties

5.7.1. Priority of Shard Node

The priority of a node is used in election mechanism while electing the primary node.

• Right click on the node in Object Explorer and select Properties.

• Node Properties window appears, where the priority of a node can be changed according to preference.

• Click OK.
5.7.2. Management Port

If you have manually changed the management port in the configuration file, you need to specify the port through manager as well.

- Go to **Tools** -> **Options** -> **Management** tab.
- Specify the **Management Port**.
- Click **OK**.
5.7.3. Shard Heartbeat

The shard heartbeat interval specifies the interval after which a heartbeat is sent to the configuration server.

- Right click on **Shards** in **Object Explorer** and select **Properties**.
- **Shards Properties** window appears, where the heartbeat interval of a shard can be changed according to preference.
- Click **OK**.
5.7.4. Node Heartbeat

The node heartbeat interval specifies the interval after which a heartbeat is sent to its neighboring nodes.

- Right click on the shard in **Object Explorer** and select **Properties**.

- **Shard Properties** window appears, where the heartbeat interval of a node can be changed according to preference.

- Click **OK**.
Specifies the interval for a node to send a heartbeat to its neighboring nodes.

**Node Heartbeat Interval:** 5 sec

Name of shard. Shard is a partition of NosDB which further consists of multiple nodes bound to each other in a replicated topology.

**Shard Name:** shard1

Each shard uses a specific port to communicate with other shards. This port must be unique for every shard and must not be blocked by firewall.

**Shard Port:** 8800

List of shard servers along with their priority.

<table>
<thead>
<tr>
<th>Server IP/Name</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>20.200.20.44</td>
<td>1</td>
</tr>
<tr>
<td>20.200.20.24</td>
<td>2</td>
</tr>
</tbody>
</table>
5.7.5. Replication

During replication, the replica nodes wait for a specific interval and pull data from the primary node in bulks. The time interval for the wait and the bulk size.

- Right click on Shards in Object Explorer and select Properties.

- Shards Properties window appears. Click on the Replication option on the left panel.
- The Time Interval and Replication Bulk Size can be configured according to preference.
- Click OK.
Specify the interval for replication. Secondary/Replica nodes will wait for the specified interval, when all data will be replicated. After the interval they will again ask primary node for next data chunk.

**Time Interval:** 15 sec

The size of data chunk, which will be replicated from primary to secondary.

**Replication Bulk Size:** 5 MB
5.8. Configure MapReduce Properties

Before proceeding to configure MapReduce, please refer to the in-depth explanation of MapReduce in Conceptual Guide.

- Right click on the shard in Object Explorer and select Properties.
- Shard Properties window appears.
- Click on the Map Reduce icon on the left pane.
MapReduce is a programming model and an associated implementation for processing and generating large data sets.

Specify the maximum number of tasks that can be run at a time.

*Maximum Running Tasks: 10

Specify the number of emitted elements a chunk must have before it is transmitted to reducers. Chunk size zero means elements are transmitted immediately.

*Chunk Size: 100

Specify maximum of tasks that can wait in queue before they are executed.

*Maximum Pending Tasks: 10

Maximum number of exception during a task execution that can be bearable.

*Maximum Exception: 10

*For these changes to take effect, ‘NosDB database service’ restart is required.
6. Databases

6.1. Create New Database

- Right click on Databases in Object Explorer and select New Database...

- Create Database window appears, where the configurations for the database can be provided according to preference.
- Specify a unique Database Name.
- Specify the Database Owner. The Database owner has all permissions to perform operations on the database.

- Refer to Configure Database Properties if you wish to specify custom configurations. If none is selected, the database is created with default configurations.
- Click OK.
6.2. Configure Database Properties

While creating the database, you can specify any configurations for journal, storage, caching and attachments. For more detail on these components, please refer to Components and Communication in Conceptual Guide.

- Right click on the database you want to configure and select Properties.

6.2.1. Journal Settings

- Click on the Journal option on the left panel.
- Specify the Journal File Size Limit in MB.
- Enable/Disable Check Sum to detect data corruption during storage or transfer.
- Specify the Clean Up Interval.
If you are satisfied with the configurations, click OK and restart NosDB Database Service. Else, proceed to Storage configuration.
6.2.2. Storage Settings

- Click on the option on the left panel.
- Specify whether the data should be stored in a **Multi File Store** or not.
- Specify the **Max File Size**. Once the max file size is reached, a new file will be created in case of multi-file store.

Specify the following configurations for the LMDB Storage Provider:

- If you are satisfied with the configurations, click **OK** and restart NosDB Database Service. Else, proceed to **Caching** configuration.
6.2.3. Cache Settings

You can specify a limit on the cache size.

- Click on the Caching option on the left panel.
- Specify the Cache Size in MB.

- If you are satisfied with the configurations, click OK and restart NosDB Database Service. Else, proceed to Attachment configuration.

6.2.4. Attachments

Attachments enable BLOB storage that can be images, videos or any other format. One database contains just one collection for attachments which is created as a System Collection. The user can enable attachments at run-time, that is, enabling attachments is a hot-applicable configuration.

Note that once the attachments are disabled, either through GUI or through the DELETE query, the data and user metadata of the attachment is removed. Hence, such an operation should be performed with caution.

- Click on the Attachment option on the left panel.
- Check the Enable Attachment checkbox.
- Specify a Shared Path which is accessible by all nodes in a shard. The attachment collection will be created in this directory.
Click **OK**.
6.3. Delete Database

- Right click on a database name in Object Explorer and select Delete.

- NosDB Management Studio will ask for confirmation. Click Yes.

- The Databases list will be refreshed and the database will be removed from the cluster.
7. Collections

7.1. Create Collection

- Right click on Collections under the database name and select New Collection...

- Create Collection window appears, where the configurations for the collection can be provided according to preference.

- Provide the Collection Name.

The collection name should be alpha-numeric, to allow REST API to access the collection.

- Specify the Primary Field. If not specified, a default field ("_key") is assigned to the document to be used as the primary field.
7.1.1. Composite Key

To specify a composite key, provide the fields in the required order.
7.1.2. Hash Based Distribution

- Click on the **Distribution Strategy** option on the left panel.
- Select **Hash Based** Distribution Strategy from the drop down menu.
- Click **OK**.
7.1.3. Range Based Distribution

- Click on the [Distribution Strategy] option on the left panel.
- Select [Range Based] from the drop-down menu.
- For [Range Based Strategy], the ranges need to be defined.
- Click on [Add].
- The [Add Range] window appears. Select the desired [Shard Name], and specify the [start] and [end ranges].
The range is added with Remove option enabled. You can remove it by clicking Remove.
7.1.4. No Distribution Strategy

- If the distribution strategy is **None**, specify the **Shard Name**.
- **Limit** the size and number of documents in the collection.

- Click **OK**.
- The **Collections** list will be refreshed and the collection will be added to the database.
7.2. Delete Collection

- Right click on the collection name under Collections and select Delete.

- NosDB Management Studio will ask for confirmation. Click Yes.

- The Collections list is refreshed and the collection is removed from the database.
7.3. Manage Ranges

7.3.1. Add Ranges

If a collection exists that has been created with the range based strategy, the ranges can be added on run-time as well.

- Right-click on the collection with range based strategy.
- Select Ranges -> Add Ranges.
- Add Range window appears.
- Specify the Start and End range.
- Click OK.

![Add Range Window](image)

7.3.2. Redefine Ranges

If a collection exists that has been created with the range based strategy, the shards for the ranges can be redefined.

- Right-click on the collection with range based strategy.
- Select Ranges -> Redefine Ranges.
- **Modify Ranges** window appears.
- Select the shard to modify for the range and click **Modify**.
8. Indexes

8.1. Conceptual Overview

Indexing is a universal method to enhance search and retrieval of any item within a collection of items. This can be illustrated by the example where a database contains terabytes of items, and the client requests to update the department of all employees of the Engineering department to Technology. If the data has not been indexed, the system steps into the dangerous terrain of traversing the whole dataset, which is a huge blow to the performance. Moreover, since most systems are distributed nowadays, there is an additional overhead of traveling over the network to search each distributed data collection for the respective Employee records. In the worst case, there may be just one record of the Engineering department, that too lying at the last location of the traversal – an unnecessary wastage of resources.

However, if the data has been indexed on Department, it takes just one query on the respective field to extract the values from their particular locations in one hop. There is no need to traverse the whole dataset, and scalability boosts the purpose of indexing as it further partitions the data.

8.1.1. Index Creation

A NosDB index is created in the form of a B tree, which contains the document keys of the indexed documents against the index key in an ordered fashion. This makes resolving queries with comparisons like "Age > 20" very simple, as all records stored after Age = 20 are understandably greater than 20. A single index can cater all of the following data structures together in itself:

- String
- Boolean
- Null
- Number
- DateTime
- Array
- Object of a custom class

Because of the various data structures being catered, indices in NosDB are very fine grained. All formats are supported in comparisons. However, the custom objects only support equality comparison.

NosDB indexes multiple data types on a single index by following a type comparison rule. What this implies is that a type is considered greater or lesser than the other. So if the index is queried as follows:

```
int value <attribute> <= string value
```

The operation will be valid and performed.

The indexes are initialized upon an index creation request from the user but remain non-functional until the index is completely populated. The index population occurs in background.
8.1.2. Index Types

8.1.2.1. Single Attribute Indices

NosDB allows the user to index any field in a document. This is useful if you have certain frequent queries and fields to apply the operations on.

For example, if a telecommunication company requires the total incoming call duration of each user at the end of each day for analysis, indexing on the field *IncomingCallDuration* will ease access to the respective data because unnecessary traversal is not required.

```json
{
    "_key": "12345",
    "FirstName": "Mary",
    "LastName": "Evans",
    "Subscription": "Post-Paid",
    "IncomingCallDuration": "20:34:04",
    ...
}
```

8.1.2.2. Embedded Indices

If the data is in the form of JSON documents, indices can be defined within the embedded documents as well. The fields of the embedded documents will be internally accessed through dot notation.

For example, if *Customer* documents are stored with an embedded document of *Address* for each employee, and operations are to be performed on the records of customers of a specific state, the index will be created on *Address.State*.

```json
{
    "_key": "12345",
    "FirstName": "Mary",
    "LastName": "Evans",
    "Age": 26,
    "MaritalStatus": "Single",
    "Transport": "None",
    "Address": {
        "Street": "999 St",
        "Town": "Houston",
        "State": "TX"
    }
}
```
8.1.2.3. Compound Indices

In order to further make the indices more specific, NosDB provides the feature of compound indexing. This means that you can create composite indices of multiple fields of the document.

For example, in the following sample document, if a firm wants to analyze the records of married customers above the age of 40 who travel by car, an index can be defined on the attributes MaritalStatus, Transport and Age. Additionally, the sequence of the attributes in the index can be switched according to the user’s preference.

```json
{
  "_key": "cust_123",
  "FirstName": "Joe",
  "LastName": "Drew",
  "MaritalStatus": "Married",
  "Transport": "Car",
  "Age": 45,
  "Address": {...}
}
```

Note that only a single array field is allowed in Compound index.

The index also provides the ability to query on prefixes. For example, the MaritalStatus, Transport and Age index created above can also be utilized for querying on following combinations of attributes:

- MaritalStatus
- MaritalStatus and Transport
- MaritalStatus, Transport and Age

However, the prefixes follow the ‘previous must’ pattern, i.e. the index cannot be used to query on attributes if their preceding attributes (prefixes) are not included. So querying on the following combinations will be delegated to full store scan:

- Transport
- Age
- Transport and Age
- MaritalStatus and Age

![Figure 8.1: Permissible and non-permissible prefixes in indexes.](image-url)
8.1.2.4. Embedded Indices

Compound indices can also be defined with embedded fields. For example, if a firm wants to analyze the records of customers of Texas older than 25 years, an index can be defined on the attributes `Age` and `Address.State`.

```json
{
  "_key": "cust_345",
  "FirstName": "Mary",
  "LastName": "Evans",
  "MaritalStatus": "Married",
  "Transport": "None",
  "Age": 26,
  "Address": {
    "Street": "999 St",
    "Town": "Houston",
    "State": "TX",
    "ZipCode": "77002"
  }
}
```

8.1.3. Index Behaviors

8.1.3.1. Sparse Indices

Since NosDB is schema-less, each document contains varying fields and formats. Thus, all of the aforementioned index types are sparse by default - they will only store those documents which contain the indexed field and will overlook the rest of the documents. This ensures the index is compact and precise; there is no point in directing a key towards an empty field if it does not exist.

For example, if one document contains the indexed field `Age` and the next document does not, the next document will not be indexed.

A null value is considered a value and will be indexed.

8.1.3.2. Caching

The index contains a cache to store the frequently used indices instead of fetching them from the file storage. This makes searching of the relevant items even faster as the index is in-memory. NosDB provides three configurable modes for caching the indices:

1. **All**: All of the index data stored on disk will be cached in memory. This impacts performance if majority of the indices are not even being used.

2. **Recent**: Stores all of the recently used data from the index. On the execution of the query, the index first checks the index cache for the required data. If it does not exist there, it will be loaded in-memory from the
file storage and reside in the cache unless internal eviction takes place according to LRU policy within the cache.

3. **None**: None of the data in the index will be cached, and every time a query is executed the data in the index will have to be accessed through the file storage. It is useful if there is memory limitation or such queries are rare which require index resolving.

### 8.1.3.3. Journaling

Each index logs its operations in a journal which are persisted in the commit cycle. If journal option is enabled, the log will be persisted on disk, else it will be stored in system memory. The journal logs the index activity like insertion, updation and removal.

### 8.2. Create Index

- Right click on **Indexes** under the collection name and select **New Index**.

  ![Object Explorer](image)

  - **Create Index Wizard** appears.
  - Specify the **Index Name**.
  - Add attributes by clicking on **Add**.
- **Add Attribute** window appears.
- Specify the **Attribute Name** and **Sort Order**.
- Click **Next**.

- **Configure Index Caching** window appears.
- Select the **Cache Policy** to specify how much of an index should be loaded in-memory.
- Specify whether the Index **Journal** should be file based or not.
• Click Finish.

![Create Index Wizard](image)

• The Indexes list is refreshed and the index is created for the collection.

8.3. **Drop Index**

• Right click on the index name under Indexes and select Drop.

![Object Explorer](image)

• The Indexes list is refreshed and the index is removed.
9. CLR Triggers

Before proceeding with creating triggers, please refer to the in-depth explanation and requirements of CLR Triggers in Conceptual Guide.

9.1. Create Trigger

- Right click on CLR Triggers under the collection name and select New Trigger.

- Configure Triggers window appears.
- Provide the Deployment through the drop down menu. If a trigger is being created for the first time, there will be no option in the drop down menu.
- Specify the Assembly Name which contains implementation of the IDatabaseTrigger interface.
- Browse to the location of the assemblies by clicking on ‘…’.
- Any classes implementing the interface IDatabaseTrigger will be automatically detected and displayed against Class Name.
- Check the checkbox if any dependent assemblies reside in the same folder the implementation of IDatabaseTrigger lies in.
  - If the dependent assemblies reside in another location, specify the location.
- Select the Class Name.
- Click Next.
- **Select Trigger Action** window appears. Select the action(s) for which you want the trigger to be fired.
- Click Finish.
The Triggers list is refreshed and trigger for the actions is added.

The deployed assemblies are created at the location: [InstallDir]\database\deployment.

In case the assembly is reloaded, restart NosDB Database Service to load the fresh assembly.
9.2. Delete Trigger

- Right click on the trigger action name under CLR Triggers and select Delete.

- The Triggers list is refreshed and the specific action is deleted.

10. CLR Functions

Before proceeding with creating user defined functions, please refer to the in-depth explanation and requirements of CLR Functions in Conceptual Guide.

10.1. Create Function

- Right click on CLR Functions under the database name and select New Function.

- Create Function window appears.

- Provide the Deployment through the drop down menu. If a CLR Function is being created for the first time, there will be no option in the drop down menu.
- Specify the **Assembly Name** which contains implementation of the `IUserDefinedFunctionProvider` interface.
- **Browse** to the location of the assemblies by clicking on ‘...’.
- Any classes implementing the interface `IUserDefinedFunctionProvider` will be automatically detected and displayed against the textbox **Classes in the selected library**.
- Check the checkbox if any dependent assemblies reside in the same folder the implementation of `IUserDefinedFunctionProvider` lies in.
  - If the dependent assemblies reside in another location, specify the location.
- Select the class name.
- Click **Next**.

![Create Function dialog box](image)

- **Select Function** window appears, where the functions against the selected class are displayed.
- Check the required functions.
• Click **Finish**.

![Create Function dialog box](image)

- The **Functions** list is refreshed and the deployed functions are displayed.
- The deployed assemblies are created at the location: `[InstallDir]\database\deployment`.
- In case the assembly is reloaded, **restart** NosDB Database Service to load the fresh assembly.
10.2. Drop Function

- Right click on the function name under **CLR Functions** and select **Delete**.

- The **Functions** list is refreshed and the specified function is removed.
11. Data Expiration

Enormous datasets of rapidly increasing volume often contain data that can become void or stale after a period of time. Such nature of data requires that it is expired on a regular basis to enhance transactions of the database with relevant data. For example, expiration aids in maintaining sessions. A session can be expired by setting expiration on a DateTime-specific field like sessionCreated. If the difference between current time and sessionCreated is greater than the session duration, the session can be expired.

Hence, incorporating data expiration in a large scale NoSQL database like NosDB results in optimized performance on account of the following factors:

- Network traffic is lowered as transaction load is specific to a certain time duration. For example, let’s suppose new logs are generated with the start of any activity. There is no need to fetch the logs from the previous activity, thus they can be expired.
- Memory consumption is regulated as the stale data is removed periodically. For example, expiration can be enabled for a system set to record video for a 5 hours duration. NosDB will then automatically remove the data which was recorded more than 5 hours ago, freeing memory for the next batch.
- Eliminates need to write code for carrying out expiration, as NosDB offers simple GUI and DDL querying based expiration management.

Expiration Interval vs. Cleanup Interval

Note that there is difference in expiration of data and actual deletion (“clean up”) of the data from the database. NosDB provides expiration interval at collection level and cleanup interval at database level. An index is maintained internally on the specified field to keep track of the documents that need to be expired after every expiration interval. Once expired, a background thread executes after a configurable interval (default is 60 seconds) that deletes expired documents. This is the cleanup interval. There may be a delay between when a document is expired and when it is actually removed from the database.

Cleanup Interval > Expiration Interval

While a document might be expired, there is possibility that it has not been removed from the database (cleanup interval is greater than expiration interval). For example, expiration interval is set to 30 seconds, while the cleanup interval is 60 seconds. Hence, 30 seconds after insertion of the document, it gets expired from the collection but still hasn’t been cleaned up from the database. Therefore, it can still entertain retrieval operations till cleanup interval is completed. Note that this also means that the lifespan of the document is actually [Expiration Interval + Remaining Cleanup Interval] in the database.

Expiration Interval > Cleanup Interval

Similarly, if expiration interval is 60 seconds and cleanup interval 30 seconds, the deletion of data from the database will only take place after the cleanup interval which succeeds the expiration interval. This can also result in an overlap, as shown in Figure 1.
11.1. Expiration Properties
NosDB expiration has the following properties:

11.1.1. Expiration Types
- Interval Based: A time duration in minutes/seconds is specified while enabling expiration. The document will automatically expire once the specified time has elapsed since the indexed field value.
- Absolute Time: Upon specifying expiration interval as 0, the expiration strategy compares field value with the current time and expires the document accordingly.

11.1.2. Document Specific
- If a document does not contain the specified field, the document will never expire.
- If a specified field is an array of date field, the item will be expired based on the lowest value in array.

11.1.3. Execution Preference
- Expiration of documents will only take place on the primary node of a shard.
- Expiration is performed by a background thread.

11.1.4. Monitoring
Number of documents expired can be monitored from a PerfMon counter "Expiration Count".

11.1.5. Limitations
- Expiration strategy cannot be configured on capped collections.
- If the indexed field is not a date field, the document will never expire.
- Expiration strategy can only be created on date fields (value of these fields must match a certain DateTime format). These are single field indexes.
### Supported DateTime Formats

<table>
<thead>
<tr>
<th>Format 1</th>
<th>Format 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>M/d/yyyy h:mm:ss tt</td>
<td>M/dd/yyyy hh:mm</td>
</tr>
<tr>
<td>M/d/yyyy h:mm tt</td>
<td>dd-MM-yy</td>
</tr>
<tr>
<td>MM/dd/yyyy hh:mm:ss</td>
<td>dd/mm/yyyy</td>
</tr>
<tr>
<td>M/d/yyyy h:mm:ss</td>
<td>dd/MM/yyyy HH:mm:ss</td>
</tr>
<tr>
<td>M/d/yyyy h:mm tt</td>
<td>yyyy-MM-dd</td>
</tr>
<tr>
<td>M/d/yyyy hh tt</td>
<td>yyyy-MM-dd HH:mm:ss</td>
</tr>
<tr>
<td>M/d/yyyy h:mm</td>
<td>yyyy-MM-ddTHH:mm:ssK</td>
</tr>
<tr>
<td>MM/dd/yyyy hh:mm</td>
<td>yyyy-MM-ddTHH:mm:sszzz</td>
</tr>
<tr>
<td>dd-MM-yyyy</td>
<td></td>
</tr>
</tbody>
</table>

11.2. **Enabling Expiration**

- Right click on **Collections** under the **database name** and select **New Collection**...

  ![Object Explorer](image)

- **Create Collection** window appears, where the configurations for the collection can be provided according to preference.

- Select **Expiration** on the left panel.

- Check the **Enable Expiration** checkbox.

- Specify the **Field Name** and **Expire After** value for the specified attribute. The expiration value can be specified in string format in either minutes (e.g. 10min) or seconds (e.g. 30sec).

- Click **OK**.
11.2.1. Configure Collection Properties

Expiration can be enabled after collection creation as well.

- Right click on **Properties** under the collection name.
  
  - Collection Properties Window appears.
  - Select **Expiration** on the left panel.
  - Repeat the process as explained in Enabling Expiration.
Alachisoft

Configuring Security

NosDB Administrators’ Guide
12. Security Concepts

12.1. Logins

Logins facilitate authentication in a system. Authentication includes the process of validating if a user has access to the system on the basis of an authentication mechanism. This is the first step in ensuring security in NosDB.

NosDB provides two types of authentication mechanisms:

Windows Authentication

This includes a challenge-response based model involving Microsoft Active Directory for the users of a local system, domain or workgroup. If a user belongs to the domain and a registered login of NosDB, he/she is authenticated to access NosDB. The following factors should be kept in mind:

- For a local machine account in domain environment, the client, configuration service and database service must be running on the same node.
- For a local machine account in workgroup environment, the account must be registered.

NosDB Authentication

This is similar to the SQL Server Authentication in SQL Server, which consists of a custom username-password authentication model. The credentials for NosDB authentication will be either passed through the connection string, or as parameters of the API and tools in an encrypted format. By default, the user admin is created, of which the password is specified during installation.

For any server-side deployments, make sure that NosDB authentication is used.

12.2. Roles and Users

A role is a set of operations that can be assigned to a user to define his/her responsibilities. In other words, a user is created once a role has been granted to a registered login in NosDB. An operation is the utilization of a resource to demonstrate certain behaviors by the user. NosDB provides a set of built-in roles with varying levels of granularity:

12.2.1. Database Roles

<table>
<thead>
<tr>
<th>Database Roles</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>db_datareader</td>
<td>A user with the db_datareader role can perform only read operations on the database. The db_datareader role is owned by db_user.</td>
</tr>
<tr>
<td>db_datawriter</td>
<td>A user with the db_datawriter role can perform only write (INSERT/UPDATE/DELETE) operations on the database. The db_datawriter role is owned by db_user.</td>
</tr>
<tr>
<td>db_user</td>
<td>A user with the db_user role can perform read and write operations on the database.</td>
</tr>
</tbody>
</table>
A user with the `db_admin` role can CREATE, DROP or ALTER a collection/index/stored procedure/CLR function/CLR trigger in NosDB. In addition, a `db_admin` can also GRANT and REVOKE roles to/from a user on the database.

A user with the `db_owner` role can perform all managerial operations on the database as performed by the `db_admin`, along with having the authorization to DROP the database.

### 12.2.2. Cluster Roles

<table>
<thead>
<tr>
<th>Role</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>dbcreator</code></td>
<td>A user with the <code>dbcreator</code> role can perform any data definition (DDL) operation on the databases, like CREATE, DROP or ALTER databases over the cluster.</td>
</tr>
<tr>
<td><code>clustermanager</code></td>
<td>A user with the <code>clustermanager</code> role can perform start/stop operations on the shards and nodes of the cluster.</td>
</tr>
<tr>
<td><code>clusteradmin</code></td>
<td>A user with the <code>clusteradmin</code> role can GRANT and REVOKE roles over the cluster and can perform managerial operations on the cluster like adding and removing the shards and nodes of the cluster.</td>
</tr>
</tbody>
</table>

### 12.2.3. Server Roles

<table>
<thead>
<tr>
<th>Role</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>securityadmin</code></td>
<td>A user with the <code>securityadmin</code> role can perform any data definition (DDL) operation on the users, like CREATE, DROP or ALTER users.</td>
</tr>
<tr>
<td><code>sysadmin</code></td>
<td>A user with the <code>sysadmin</code> role can perform any operation on the NosDB server, including cluster and database managerial operations.</td>
</tr>
<tr>
<td><code>distributor</code></td>
<td>The distributor is a special role, used only for the distributor service. A <code>distributor</code> is only authorized to distribute client operations.</td>
</tr>
</tbody>
</table>

### 12.2.4. Custom Roles

NosDB provides the flexibility of defining custom roles for users which grant a customized set of varying operation permissions to a user. Custom roles have database level scope, meaning the custom role can only be created on or can have permissions on databases only.

The process of creating and granting custom roles requires the following privileges:

1. The custom role is created by the `clusteradmin`.

The role name can contain any alpha-numeric combination and special characters except for `\`. 
2. Once created, permissions can now be granted to the role. The db_securityadmin, db_owner and db_admin have the privilege to grant permissions to the role. These permissions can be selected from the following permission set:

<table>
<thead>
<tr>
<th>Permissions</th>
<th>DropDatabase</th>
<th>AlterDatabase</th>
<th>CreateCollection</th>
</tr>
</thead>
<tbody>
<tr>
<td>DropCollection</td>
<td>CreateIndex</td>
<td>AlterCollection</td>
<td>CreateCollection</td>
</tr>
<tr>
<td>DropIndex</td>
<td>CreateIndex</td>
<td>AlterCollection</td>
<td>CreateIndex</td>
</tr>
<tr>
<td>CreateUserDefinedFunction</td>
<td></td>
<td>DropUserDefinedFunction</td>
<td></td>
</tr>
<tr>
<td>CreateTrigger</td>
<td>GrantPermission</td>
<td>DropTrigger</td>
<td>BackupDatabase</td>
</tr>
<tr>
<td>GrantDBRole</td>
<td>GrantPermission</td>
<td>RevokeDBRole</td>
<td>Read</td>
</tr>
<tr>
<td>RevokePermission</td>
<td>Write</td>
<td></td>
<td>Read</td>
</tr>
</tbody>
</table>

3. Once the role has been granted with the permissions, the db_securityadmin, db_owner and db_admin can grant the role to an existing user in NosDB. The user now has permissions to perform the operations on the specified database.

A custom role can contain an empty permission set and be granted to a user. Once any permissions are granted to the role, they will be granted to the specified user.

### 12.3. Encryption

Encryption makes data undecipherable using a key and password to render it useless without that key. Hence NosDB provides encryption to enhance the security of data; any data added to the database collection will be encrypted, and will be of no use unless the key is available to decrypt.

NosDB encryption implements the Data Encryption Key (DEK) and Master Key model. The DEK is determined through the encryption provider specified by the user. The encryption provider is an encryption standard, either 3DES or AES. This DEK is used to encrypt the data before it is written to the store.

Moreover, further security is established by encrypting the Data Encryption Keys through the master key. To encrypt/decrypt the DEKs, the user creates a master key for NosDB by specifying a password for the master key. Using that password, NosDB gets a master key from DPAPI (Data Protection Application Programming Interface) and stores it with the DEK in encrypted form in the configuration store to be later used to encrypt and decrypt the Data Encryption Keys. The master key is encrypted through DPAPI as well.

NosDB provides the flexibility to choose from any of the built-in encryption providers to configure encryption at database level:

- 3DES_128
- 3DES_192
- AES_128
- AES_192
- AES_256

Encryption can be configured and enabled at runtime. However, the encryption provider cannot be reconfigured once it is specified. Note that encryption is configured at the database level and enabled at a collection level. This means...
while the provider is configured on the database, encryption will need to be enabled individually on each desired collection.

Any existing data in the collection will not be encrypted unless it is updated after encryption is enabled. Any data added after encryption is enabled will be encrypted by default.

NosDB supports document level encryption - complete document is encrypted.

- **Encryption Process**

1. Using DDL query, the master key is created from a user specified password.
2. A Database Encryption Key is automatically generated when the encryption provider is configured using NosDB Management Studio or DDL query. This DEK will be used to encrypt the data, while the master key encrypts the key.
3. Once the encryption provider is configured, encryption can be enabled and disabled on the collection.

13. **Logins**

Before proceeding to configure security and assign roles and users, please refer to the in-depth explanation of Security in Conceptual Guide.

13.1. **Add Windows Login**

- Right click on Logins under the cluster and select New Login.

- Create Login window appears.

- Specify the Login Name by clicking on the ‘...’ button.
• **Select Users or Groups** window appears.

• Specify the **Object Types** and **Location**.

• Click **OK**.

• Specify **Windows Authentication**.

• Click **OK**.
13.2. Add NosDB Login

- Right click on Logins under the cluster and select New Login.
• **Create Login** window appears.
  
  • Specify the **Login Name** by clicking on the ‘…’ button.

![Create Login window](image1)

• **Select Users or Groups** window appears.
  
  • Specify the **Object Types** and **Location**.

![Select Users or Groups window](image2)

• Click **OK**.
  
  • Specify **NosDB Authentication**.
  
  • Click **OK**.
Create Login

Specify the name for login. Login is associated to users to access NosDB.

**Login name:** JOHN-SMITH\Administrator

- No Window Authentication
- NosDB Authentication

**Password:** ********

**Conform Password:** ********
13.3. Drop Login

- Right click on the login name under Logins and select Delete.

- The login is dropped from NosDB.
14. Users

14.1. Create Cluster User

- Right click on Users under Security of the cluster and select New User.

![Object Explorer](image)

- New User window appears, select the Login Name and cluster roles to be granted to the user.
- Click OK.

![New User](image)

- The Users list is refreshed and the login name is added to the cluster with granted roles.
14.2. Create Database User

- Right click on Users under Security of the database and select New User.

- New User window appears, select the Login Name and database roles to be granted to the user.
- Click OK.

- The Users list is refreshed and the login name is added.
14.3. Drop User

- Right click on the login name under **Users** and select **Drop**.

- NosDB asks for confirmation. Click **Yes**.

- The Users list is refreshed and the user is dropped.
15. Roles

You can assign the following roles to the login:

15.1. Server Roles

- Click on the Server Roles option on the left panel.
- Check the roles to grant to the user.

![Server Roles](image)

15.2. Cluster Roles

- Click on the Cluster Roles option on the left panel.
- Check the roles to grant to the user.
15.3. Database Roles

- Click on the option on the left panel.
- Check the roles to grant to the user.
Click OK. The login is created with the granted roles.

If you wish to modify the roles, right-click on the login name and select **Properties** to modify the properties.

### 15.4. Custom Roles

- Right click on **Roles** under **Security** and select **New Role**.
• **Create Role** window appears. Specify a **Name** for the custom role and select the **Resource(s)** over which this role is to be created. The resources are all databases existing in the cluster.

• Select the custom **Permissions** to be granted to this role.

• Click **OK**.
- The new custom role will be added under the list of **Roles**. The roles are also visible in **Roles** under the specific database.
You can now assign this role to an existing or new user. Right click on a user name and click **Properties**.
• **User Properties** window pops up. Check the checkbox against the custom role that has been created.
• Click **OK**.
User Properties

Name of user. Username is associated to users to access NosDB.

User name: admin

Following are the list of database roles that can be granted to a user.

Roles:
- db_owner
- db_admin
- db_backupoperator
- db_securityadmin
- db_user
- db_datawriter
- db_datareader
- CustomDropRole

OK  Cancel
16. Encryption

16.1. Create Master Key

Create Master Key from the user specified password.

Invoke-SQL -Query 'CREATE MASTER KEY BY PASSWORD = ''12345678'''

16.2. Configuring Encryption

- Right-click on a database name and select Properties.
- Select the Storage tab.
- Select the desired Encryption Provider from the provided five encryption standards in the drop down menu.
- Click OK.
16.3. Enabling Encryption

Encryption can now be enabled on the documents.

- Right-click on the collection you want to enable encryption on. Select **Properties**.
- Select the **Encryption** tab.
- Check the **Enabled** checkbox to enable encryption.
- Click **OK**.
Note that if encryption has not been configured, this window will display the following reminder:

![Collection Properties](image)

**Note:** To enable encryption, you must configure the encryption provider on the database.
17. SSL Security

17.1. Configure SSL Certificate in NosDB

This feature is only available in NosDB Enterprise Edition.

NosDB provides the facility to enable TLS/SSL encryption over the network to ensure secure data exchange takes place between the server and the authorized client.

Users can hence enable any issued or self-signed SSL certificate to enable connections with of the client with the configurations server, or enable component-to-component connections in NosDB. This guarantees encrypted data transmission by default.

Before creating a SSL certificate in NosDB, the following prerequisites must be considered:

- The certificate must have a private key.
- All NosDB services must be running.
- The private key must be exportable, to be exported to all nodes including the client machine.

Before enabling SSL security, make sure that:

- All NosDB services have been stopped.
- NosDB Management Studio and PowerShell have been closed down.
- The certificate is installed in the Trusted Root Certificate Authority store in Microsoft Management Console. If it is in the Personal folder, drag-and-drop the certificate to the Certificates folder in Trusted Root Certificate Authority.
- The certificate is installed in the Root Store, on LOCAL MACHINE.

Enabling SSL Certificate in NosDB

- The certificate can now be enabled for NosDB through the Registry Editor in the following key location:

  HKEY_LOCAL_MACHINE\SOFTWARE\Alachisoft\NosDB\Security

- This key contains the following properties of the certificate, which have to be provided by the user:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CertName</td>
<td>Name of the certificate to be enabled for NosDB SSL security</td>
</tr>
<tr>
<td>SSLEnabled</td>
<td>Boolean value to enable or disable SSL certificate</td>
</tr>
<tr>
<td>Thumbprint</td>
<td>Unique identifier for each certificate. <strong>Must</strong> be provided without spaces.</td>
</tr>
</tbody>
</table>
The details for the certificate can be obtained through the following steps:

- Go to Microsoft Management Console.
- Open the Trusted Root Certificate Authority folder in the standard way.
- Double-click on the certificate name. This name is to be provided as the value for CertName property,
- Click on the Details tab.
- Click on the Thumbprint option.
- Copy the thumbprint to edit it and remove spaces from the value. This value is to be provided to the Thumbprint property,
- Provide the CertName and Thumbprint and enable SSL by setting the value of the property SSLEnabled to 1.

- Export the certificate (through the standard way of exporting certificates) to all nodes in the cluster along with the client machine, and perform the same steps to enable SSL security on all nodes.

Once the certificates have been enabled, make sure that all client applications run on 64-bit machines.
17.2. Configure SSL Certificate on REST Server

- Open **Internet Information Services Manager**.
- Select **Server Certificates** under the option IIS.

![Internet Information Services Manager](image)

- Server Certificates window displays any previously installed and default certificates.
- Select **Import...** from the **Actions** pane on the right.
- Browse to select the **Certificate File** you want to use (must be on local computer and trusted root).
- Enter the **Password** for the certificate.
- Set the **Certificate Store** option to **Personal**.
- Allow this certificate to be exported through the check box.
- Click **OK**.
• The certificate is now added to the server certificates. If you have enabled https protocol previously, the following steps are to be ignored.

• To enable server side encryption through the https protocol:
  o Select Default Web Site on the left panel
  o Select Bindings... on the Action pane on the right.
  o Site Bindings window displays the type of Site Bindings enabled and their respected details.
  o Click Add...
  o Set the Type to https and select the SSL Certificate which has been imported to the server.
  o Click OK.
- The https binding is set to the site.
- Click Close.
- Go to nosdbdata under default web Site.
- Select Browse *.443 (https).
- Localhost will be launched with HTTPS successfully. You can now perform REST based Querying through the URL securely.

```json
{
   "@odata.context": "https://localhost/nosdbdata/clustered/northwind/$metadata", "value": [
   {
      "name": "products", "kind": "EntitySet", "url": "products"
   }
   ]
}
```
17.3. Configure SSL Certificate on REST Client

In case you wish to authenticate the client before the result is fetched through REST querying, you can do so by providing the client certificates which have been stored in the Personal account of the current user.

- Go to nosdbdata under Default Web Site in IIS Manager.
- Check the box Require SSL.
- Select Require or Accept, based on your requirements.
- Click Apply on the Actions pane on the right.

Once the URL is entered, the browser shows a prompt before proceeding to fetch the results.

The prompt asks to select a certificate and displays all certificates deployed in the Personal Account.
• Select the desired certificate to authenticate the client.
• Click OK.
• The result be displayed once the client is authenticated. Else, a **Forbidden 403** status code is displayed.
Database Administration

NosDB Administrators’ Guide
18. Backup Database

18.1. Conceptual Overview

In a distributed environment as that of NosDB, it is essential to provide backup and restore operations to ensure fault-tolerance and protection of data in case of unexpected system failure.

NosDB provides database-level backup and restore operations on running nodes, i.e., moment-in-time snapshots of the data will be taken for any operations being performed during the backup process. This will ensure data consistency while restoring the database. An atomic backup or restore operation is performed against a clustered/stand-alone database.

Backup ensures that a copy of the data is preserved so that it can be restored in the event of a system failure. Moreover, backups can be made to store a copy of the source database to a new location on either configuration:

- **Distributed**: Creates backup of each shard to a specified location on the primary node of the respective shard. For example, `D:\database\NosDBbackups`.

  The location specified for backup must be valid for all primary nodes of each shard.

- **Consolidated**: Creates a backup of all the shards sequentially on a shared location. For example, `//server1/backups`.

  Make sure that NosDB Database service (`NosDBSvc`) and NosDB Configuration service (`NosConfSvc`) have write access on the location.

NosDB offers two levels of backup jobs:

18.1.1. Full backup

A full backup, as the name implies, will store the whole data and all components associated with it to either a local or shared location. A full backup copies the following:

- Configuration meta-info about the clustered or stand-alone database like collections, indexes, storage provider from the configuration server. The meta-info aids in creating a database at any location with same or a different name.

- The Oplog for the operations being performed during the backup process. Since NosDB backup can be performed without going offline, the Oplog backup enables a moment-in-time data snapshot of the operations to prevent any inconsistency in restoration.

  If an operation which triggers state transfer (e.g., adding/removing shard) is performed during backup, the job would be cancelled. Thus make sure that no state transfer occurs while the backup is in progress.
- Actual data from the source databases against the shards. However, for server side deployments and attachments, only the configuration information will be backed up – not the actual data.

### 18.1.2. Differential backup

Since full backup is an expensive process for huge amounts of data, NosDB offers differential backup which only backs up the modifications in the database that have occurred since the last full backup. This is particularly useful if regular backups are made, as the data, time and network cost involved is much lesser compared to a full backup. The backup is performed in a cumulative manner - the changes are backed up as they keep on occurring at the database.

A differential backup copies the following:

- The Oplog of any operations being performed during the backup process. The operations are tracked in a dedicated log (DIFLog) which stores the Oplog operations being performed against each shard since the last full backup.

> If a new shard is added to a configuration against which a DIFLog is being created, the tracking would be cancelled.

- Actual data from the shards that has been changed because of add, update and delete operations.

Figure 6.1. Backup

Referring to Figure 6.1, the differential backup made on $T_{D1}$ will copy all modifications made since the full backup at $T_{F1}$. The backup made on $T_{D2}$ will copy all changes made since $T_{F1}$, $T_{D3}$ will store all changes made since $T_{F1}$ and so on.

Once the next full backup is made at $T_{F2}$, the whole data will be copied along with the configuration details and Oplog, which will include the changes from the differential backup.
18.2. Create Database Backup

To create a backup of a database,

- Right-click on Databases in Object Explorer, select Task -> Backup.

![Object Explorer with Backup option](image)

- Backup window appears.
- Select the Database Name for which you want to create the backup.
- Select the Backup Type, which can be Full or Differential.

**Differential Backup** is only available in NosDB Enterprise Edition.

- Select whether the backup storage type is Distributed (on local path) or Consolidated (on a shared path).

18.2.1. Distributed Backup Storage

- For Distributed Backup storage, specify the path where the backup files should be created.
18.2.2. **Consolidated Storage**

- For **Consolidated backup storage**, specify the **UNC path**, **User Name** for the domain and **Password** for authentication.
- You can even test the connection to the UNC path before starting backup by clicking the **Test Connection** button.
In case you wish to enable differential tracking, check the **Enable Differential** checkbox.

Click **OK**.

**Task Status Dialog** appears, with details of the backup job like database name, job type, status and percentage of job completion.

The backup is created in the specified location in a folder named with the format *databasename-timestamp-status*. For example, for a complete backup made on the database *northwind*, the folder is created with the name "northwind-20160621161706536-completed".

In case the backup job fails due to any unexpected failure, there might be some shards showing the status "completed", while others might still be named with the status "in-progress". This helps in distinguishing whether the backup was fully made or not.
19. Restore Database

19.1. Conceptual Overview

Restoration is a multi-phase process that recreates a database from the provided backup. Just like backup, NosDB offers two levels of restoration jobs: Full and Differential. This implies that the restore type should correspond to the type of backup you have performed.

For example, a Differential Restore will only be performed on a database backed up through the Differential option. This is because the DIFLog is used during the Differential Restore, and a Full Backup does not create the DIFLog.

19.1.1. Full Restore

In a full restore, the entire database is reconstructed from the provided backup.

During a full restore, NosDB:

i. Configures the destination database according to the configuration of the provided backup in the config server meta-info. The provided configuration must be consistent with the one existing during the backup phase. These configurations include:
   - Number of shards
   - Shard names
   However, number of nodes in those shards and participating nodes (IPs) can be changed.

   Consistency in configuration is mandatory because the Oplog operations are applied to recreate a moment-in-time snapshot of the source data. Hence, in case of any inconsistent configuration, NosDB will wipe out the existing data and restore the database with the new configuration.

ii. Recreates data and indexes through the backed up database against all participating shards in the database.
iii. Performs any operations in the Oplog that were performed during the backup window, ensuring consistency.

19.1.2. Differential Restore

As full restoration can prove a heavily expensive operation if performed regularly, NosDB offers differential restoration which only restores the modifications that have occurred since the last full restoration.

A differential restore must be provided with a database that has been backed up through the differential backup option.

Restoration is a comparatively costly process because of storage limitations, and the reconstruction of an entire database and index creation. Thus, in case a database exists with restored data, NosDB will apply differential restoration of the same database to the existing one.

A database will not be restored with an existing name.

In case the database does not exist, NosDB performs a full backup then performs differential restore against the database.

19.2. Restore Backed up Database

To restore a backed up database,

- Right-click on Databases in Object Explorer and select Task -> Restore.

  - Restore window appears.
  - Select the Restore Type, which can be Full or Differential.
Differential Restore is only available in NosDB Enterprise Edition.

A restoration type must be provided with a database that has been backed up through the corresponding backup type. For example, Full Backup will allow Full Restore.

- Select whether the restoration is being made from a Distributed backup (on local path) or Consolidated backup (on a shared path).

19.2.1. Distributed Storage
- For Distributed Backup, specify the path of the backup folder.
- Specify the Source Database name which was backed up.
- Specify the Destination Database name, if you want to restore it with a new name.
19.2.2. Consolidated Storage

- For **Consolidated backup**, specify the **UNC path** of the backup folder, **User Name** for the domain and **Password** for authentication.

- You can even test the connection to the UNC path before starting backup by clicking the **Test Connection** button.

- Specify the **Source Database** name which was backed up.

- Specify the **Destination Database** name, if you want to restore it with a new name.
Click **Start** to initiate restoration.

The **Task Status** window appears, showing the progress of the task. Click **OK**.

Once completed, refresh the **Databases** option in **Database Cluster**. The restored database is now visible.
20. **View Task Status**

To view the status of backup and restore operations including percentage completion,

- Right-click on **Databases** in **Object Explorer** and select **Task -> Task Status**.

- **Task Status** window appears, displaying detailed information about the task.
- Click **OK**.
21. View Task History

To view the history of operations including percentage completion,

- Right-click on Databases in Object Explorer and select Task -> Task History.
- **Task History** window appears, showing the starting date, database name, job type, status and completion percentage.

![Task History Dialog]

### 22. Disable Differential Logs

Differential backup can be tracked once it is enabled during backup. However, you can disable differential tracking any time.

- Right-click on **Databases** in **Object Explorer** and select **Task -> Differential Logs**.
• **Differential Logs** window appears, displaying the databases for which differential backup has been enabled. Check on the database for which you want differential logging to be stopped.

• Click **Done**.
23. **Configure Multiple NICs**

- Right click on a node IP address under a shard in Object Explorer and select the **Select NIC** option.

- **Available Network Interface Cards** dialog opens. It contains all network cards available on that node with drop down lists in front of them.

- The **Bound With** column displays 3 options for communication:
  - **Cluster** will enable inter-cluster communication between the servers using the IP.
  - **Client-Server** will bind the new IP to allow communication between the client and server.
  - **Both (Cluster & Client/Server)** uses the same IP for both communications.

- Select the type of binding you want against each NIC.

- **Click Apply.**
NosDB will ask for confirmation to apply configuration which results in restart of NosDB configuration and database services. Click Yes.

At this stage, NosDB might ask for credentials if the node is remote or does not have the rights.

- If the node belongs to the same domain, provide the **Username** and **Password**.
- If the node does not belong in the same domain, add the user to the Administrator group in Windows and then provide the username as `domainname\username` and the password.

This restarts the NosDB service to use the selected IP addresses for the specified communication (inter-cluster).

### 24. Deploy Providers

Deploying providers allow copying the required provider assemblies to the server so that operations are performed over the server, cutting down the traveling costs. This includes providers for MapReduce, Aggregator, Triggers and UDFs.

For any server-side deployments, make sure that NosDB authentication is used or the services (`NosDBSvc` and `NosConfSvc`) are being run from the account which is logged on.

- Right click on a database name in **Object Explorer** and select **Deploy Provider**.
• **Deploy Assembly** window appears.

• Specify the **Deployment Name** to deploy the assemblies.

• Specify the location where the assembly exists or browse by clicking on the “...” button.

• Any assemblies within the specified location will be detected and displayed in the **Assemblies** textbox.

• Check the required assemblies.

• Click **OK**.
Deploy Assembly

Select the deployment name to deploy assembly/assemblies with this database, you can create new deployment by selecting create new.

Deployment Name: < create new >
- Triggers

Browse or specify the physical location of disk where the assembly/assemblies reside.

Location: F:\Projects\myTrigger\myTrigger\bin\Debug

List of assemblies reside within the specified location.

Assemblies:
- Aachisoft.NosDB.Client.dll
- Aachisoft.NosDB.Common.dll
- Aachisoft.NosDB.Core.dll
- Aachisoft.NosDB.Distributor.dll
- Aachisoft.NosDB.Serialization.dll
- myTrigger.dll

[OK] [Cancel]
25. Launch NosDB PowerShell

- Right-click on a **database name** in **Object Explorer** and select **PowerShell**.

- NosDB PowerShell Module launches, where you can perform operations through the command line tools provided by NosDB. Refer to **Administrators' Guide – PowerShell** to perform operations through PowerShell.

26. Configure Offline Tasks

By default, the operations are performed over databases in online mode – including backup and restore. This means that the database is active at the time of operation. However, if you wish to perform any operation over the database in offline mode, you can do so through the Management Studio.

- Right click on a **database name** in **Object Explorer** and select **Task**.
- It shows two statuses, where by default, the task is **Online**.
- Select **Offline**. No client operations will now be performed over the database until it is online again.
• The database icon turns from red to grey, indicating an offline database.

• The database will not have any option enabled except for the tools, **Deploy Provider** and **Task**.

• Right click on the database name, go to **Task->Online** to go back online.
27. Using SQL Queries

- Right click on a database name in Object Explorer and select New Query.
- Right click on a **collection name** in **Object Explorer** and select **New Query**.

  ![Object Explorer]

- The **Query Executor** opens up, where the query can be executed on any existing collection of the database.
- Type in the query according to DQL syntax.
- To verify if the query syntax is valid, click on **Parse Query** in the toolbar or **Ctrl + F5**.
  - **Invalid Syntax**

  ![Query Executor]

  ```sql
  SELECT CustomerID, Country, Phone FROM Customers
  ```
  
  ![JSON String]

<table>
<thead>
<tr>
<th>RowID</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Invalid query syntax specified. Syntax Error near 'Country' on line number: 1</td>
</tr>
</tbody>
</table>
Valid Syntax

- To execute the query, click on **Execute Query** in the toolbar or **Shift+F5**.
- The results are displayed in JSON and tabular forms if the collection exists and has data.
27.1. JSON String Output

```sql
SELECT CustomerID, Country, Phone FROM Customers
```

<table>
<thead>
<tr>
<th>RowID</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>{&quot;CustomerID&quot;: &quot;ALFKI&quot;, &quot;Country&quot;: &quot;Germany&quot;, &quot;Phone&quot;: &quot;030-0074321&quot;}</td>
</tr>
<tr>
<td>1</td>
<td>{&quot;CustomerID&quot;: &quot;ANATR&quot;, &quot;Country&quot;: &quot;Mexico&quot;, &quot;Phone&quot;: &quot;(5) 555-4729&quot;}</td>
</tr>
<tr>
<td>2</td>
<td>{&quot;CustomerID&quot;: &quot;ANON&quot;, &quot;Country&quot;: &quot;Mexico&quot;, &quot;Phone&quot;: &quot;(5) 555-3932&quot;}</td>
</tr>
<tr>
<td>3</td>
<td>{&quot;CustomerID&quot;: &quot;AROUT&quot;, &quot;Country&quot;: &quot;UK&quot;, &quot;Phone&quot;: &quot;(171) 555-7788&quot;}</td>
</tr>
<tr>
<td>4</td>
<td>{&quot;CustomerID&quot;: &quot;BERGS&quot;, &quot;Country&quot;: &quot;Sweden&quot;, &quot;Phone&quot;: &quot;0921-12 34 65&quot;}</td>
</tr>
<tr>
<td>5</td>
<td>{&quot;CustomerID&quot;: &quot;BLAUS&quot;, &quot;Country&quot;: &quot;Germany&quot;, &quot;Phone&quot;: &quot;0621-08460&quot;}</td>
</tr>
<tr>
<td>6</td>
<td>{&quot;CustomerID&quot;: &quot;BLOMP&quot;, &quot;Country&quot;: &quot;France&quot;, &quot;Phone&quot;: &quot;88.60.15.31&quot;}</td>
</tr>
<tr>
<td>7</td>
<td>{&quot;CustomerID&quot;: &quot;BOLID&quot;, &quot;Country&quot;: &quot;Spain&quot;, &quot;Phone&quot;: &quot;(91) 555 22 82&quot;}</td>
</tr>
<tr>
<td>8</td>
<td>{&quot;CustomerID&quot;: &quot;BONAP&quot;, &quot;Country&quot;: &quot;France&quot;, &quot;Phone&quot;: &quot;91.24.45.40&quot;}</td>
</tr>
<tr>
<td>9</td>
<td>{&quot;CustomerID&quot;: &quot;BOTTM&quot;, &quot;Country&quot;: &quot;Canada&quot;, &quot;Phone&quot;: &quot;(604) 555-4729&quot;}</td>
</tr>
<tr>
<td>10</td>
<td>{&quot;CustomerID&quot;: &quot;BSBV&quot;, &quot;Country&quot;: &quot;UK&quot;, &quot;Phone&quot;: &quot;(171) 555-1212&quot;}</td>
</tr>
<tr>
<td>11</td>
<td>{&quot;CustomerID&quot;: &quot;CACTU&quot;, &quot;Country&quot;: &quot;Argentina&quot;, &quot;Phone&quot;: &quot;(1) 135-5555&quot;}</td>
</tr>
<tr>
<td>12</td>
<td>{&quot;CustomerID&quot;: &quot;CENTC&quot;, &quot;Country&quot;: &quot;Mexico&quot;, &quot;Phone&quot;: &quot;(5) 555-3392&quot;}</td>
</tr>
<tr>
<td>13</td>
<td>{&quot;CustomerID&quot;: &quot;CHOPS&quot;, &quot;Country&quot;: &quot;Switzerland&quot;, &quot;Phone&quot;: &quot;0452-076545&quot;}</td>
</tr>
<tr>
<td>14</td>
<td>{&quot;CustomerID&quot;: &quot;COMM&quot;, &quot;Country&quot;: &quot;Brazil&quot;, &quot;Phone&quot;: &quot;(11) 555-7647&quot;}</td>
</tr>
<tr>
<td>15</td>
<td>{&quot;CustomerID&quot;: &quot;CONSH&quot;, &quot;Country&quot;: &quot;UK&quot;, &quot;Phone&quot;: &quot;(171) 555-2282&quot;}</td>
</tr>
<tr>
<td>16</td>
<td>{&quot;CustomerID&quot;: &quot;DUMON&quot;, &quot;Country&quot;: &quot;France&quot;, &quot;Phone&quot;: &quot;40.67.88.88&quot;}</td>
</tr>
<tr>
<td>17</td>
<td>{&quot;CustomerID&quot;: &quot;EASTC&quot;, &quot;Country&quot;: &quot;UK&quot;, &quot;Phone&quot;: &quot;(171) 555-0297&quot;}</td>
</tr>
</tbody>
</table>
### 27.2. Tabular Output

```
SELECT CustomerID, Country, Phone FROM Customers
```

<table>
<thead>
<tr>
<th>CustomerID</th>
<th>Country</th>
<th>Phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALFKI</td>
<td>Germany</td>
<td>030-0074321</td>
</tr>
<tr>
<td>ANATR</td>
<td>Mexico</td>
<td>(5) 555-4729</td>
</tr>
<tr>
<td>ANTON</td>
<td>Mexico</td>
<td>(5) 555-3932</td>
</tr>
<tr>
<td>AROUT</td>
<td>UK</td>
<td>(171) 555-7788</td>
</tr>
<tr>
<td>BERGS</td>
<td>Sweden</td>
<td>0921-123456</td>
</tr>
<tr>
<td>BLAUS</td>
<td>Germany</td>
<td>0621-08460</td>
</tr>
<tr>
<td>BLONP</td>
<td>France</td>
<td>88.60.15.31</td>
</tr>
<tr>
<td>BOLID</td>
<td>Spain</td>
<td>(91) 5552282</td>
</tr>
<tr>
<td>BONAP</td>
<td>France</td>
<td>91.24.45.40</td>
</tr>
<tr>
<td>BOTTM</td>
<td>Canada</td>
<td>(604) 555-4729</td>
</tr>
<tr>
<td>BSBEV</td>
<td>UK</td>
<td>(171) 555-1212</td>
</tr>
<tr>
<td>CACTU</td>
<td>Argentina</td>
<td>(1) 135-5555</td>
</tr>
<tr>
<td>CENTC</td>
<td>Mexico</td>
<td>(5) 555-3392</td>
</tr>
<tr>
<td>CHOPS</td>
<td>Switzerland</td>
<td>0452-076545</td>
</tr>
<tr>
<td>COMMI</td>
<td>Brazil</td>
<td>(11) 555-7647</td>
</tr>
<tr>
<td>CONSH</td>
<td>UK</td>
<td>(171) 555-2282</td>
</tr>
<tr>
<td>DUMON</td>
<td>France</td>
<td>40.67.86.88</td>
</tr>
<tr>
<td>EASTC</td>
<td>UK</td>
<td>(171) 555-0297</td>
</tr>
</tbody>
</table>
27.3. Copying Output

If you wish to copy the query result:

- **JSON Output**
  
  Right-click on a record and select **Copy**.

- **Tabular Output**
  
  Right-click on a record and select either **Copy with Headers** or **Copy** according to your preference.
27.4. Select Top 1000 Rows

NosDB allows fetching the top 1000 rows of a collection through the Management Studio, instead of typing in a query specifically for it.

- Right click on a database name in Object Explorer and select ‘Select Top 1000 Rows’.

- The Query Executor opens up, where the query “SELECT TOP 1000 * FROM <collection>” is executed on the selected collection of the database, displaying the results.
```sql
SELECT TOP 1000 * FROM "orders"
```

<table>
<thead>
<tr>
<th>Key</th>
<th>OrderID</th>
<th>CustomerID</th>
<th>EmployeeID</th>
<th>OrderDate</th>
<th>RequiredDate</th>
<th>ShippedDate</th>
<th>ShippedVia</th>
</tr>
</thead>
<tbody>
<tr>
<td>819ad3b1-53be-...</td>
<td>10248</td>
<td>VNET</td>
<td>5</td>
<td>1996-07-04 10:00:00</td>
<td>1996-08-01 00:00:00</td>
<td>1996-07-16 00:00:00</td>
<td>3</td>
</tr>
<tr>
<td>4003decf-e325-...</td>
<td>10249</td>
<td>TOMSP</td>
<td>6</td>
<td>1996-07-05 10:00:00</td>
<td>1996-06-16 00:00:00</td>
<td>1996-07-10 00:00:00</td>
<td>1</td>
</tr>
<tr>
<td>a2943057-de84-...</td>
<td>10250</td>
<td>HANAR</td>
<td>4</td>
<td>1996-07-08 10:00:00</td>
<td>1996-08-05 00:00:00</td>
<td>1996-07-12 00:00:00</td>
<td>2</td>
</tr>
<tr>
<td>ca874897-413d-...</td>
<td>10251</td>
<td>VICTE</td>
<td>3</td>
<td>1996-07-08 10:00:00</td>
<td>1996-08-05 00:00:00</td>
<td>1996-07-15 00:00:00</td>
<td>1</td>
</tr>
<tr>
<td>7dc6bb54-c2b5-...</td>
<td>10252</td>
<td>SUFRD</td>
<td>4</td>
<td>1996-07-09 10:00:00</td>
<td>1996-06-06 00:00:00</td>
<td>1996-07-11 00:00:00</td>
<td>2</td>
</tr>
<tr>
<td>e95f9e1e-859a-...</td>
<td>10253</td>
<td>HANAR</td>
<td>3</td>
<td>1996-07-10 10:00:00</td>
<td>1996-07-24 00:00:00</td>
<td>1996-07-15 00:00:00</td>
<td>2</td>
</tr>
<tr>
<td>fd9781e1-72eb-...</td>
<td>10254</td>
<td>CHOPS</td>
<td>5</td>
<td>1996-07-11 10:00:00</td>
<td>1996-08-08 00:00:00</td>
<td>1996-07-23 00:00:00</td>
<td>2</td>
</tr>
<tr>
<td>6c492e45-6f19-...</td>
<td>10255</td>
<td>RICSU</td>
<td>9</td>
<td>1996-07-12 10:00:00</td>
<td>1996-08-09 00:00:00</td>
<td>1996-07-15 00:00:00</td>
<td>3</td>
</tr>
<tr>
<td>0f757e56-e6b4-...</td>
<td>10256</td>
<td>WELLI</td>
<td>3</td>
<td>1996-07-15 10:00:00</td>
<td>1996-06-12 00:00:00</td>
<td>1996-07-17 00:00:00</td>
<td>2</td>
</tr>
<tr>
<td>aabc6403-2a74-...</td>
<td>10257</td>
<td>HEA</td>
<td>4</td>
<td>1996-07-16 10:00:00</td>
<td>1996-08-13 00:00:00</td>
<td>1996-07-22 00:00:00</td>
<td>3</td>
</tr>
<tr>
<td>8b07755a-e1d4-...</td>
<td>10258</td>
<td>ERNSH</td>
<td>1</td>
<td>1996-07-17 10:00:00</td>
<td>1996-08-14 00:00:00</td>
<td>1996-07-23 00:00:00</td>
<td>1</td>
</tr>
<tr>
<td>7a8f529c-11b7-...</td>
<td>10259</td>
<td>CENTC</td>
<td>4</td>
<td>1996-07-18 10:00:00</td>
<td>1996-06-15 00:00:00</td>
<td>1996-07-25 00:00:00</td>
<td>3</td>
</tr>
<tr>
<td>7adb1b70-a356-...</td>
<td>10260</td>
<td>OTTIX</td>
<td>4</td>
<td>1996-07-19 10:00:00</td>
<td>1996-08-06 00:00:00</td>
<td>1996-07-29 00:00:00</td>
<td>1</td>
</tr>
</tbody>
</table>
28. Convert Standalone to Clustered Database

This feature is only available in NosDB Enterprise Edition.

NosDB offers the flexibility of migrating a stand-alone database into a clustered database through the Hash Based Distribution Strategy.

Pre-conditions:

- The node of the standalone database should be primary of the configuration cluster.
- Complete cluster (all nodes) should be running.

State transfer is initiated as soon as it converted to a cluster. During that, the database will not be active as shown by the status Under Migration, and will not entertain any client operations.

- Right click on the standalone database to migrate under Standalone Databases.
- Select Make Cluster Database.

- Make Cluster Database window appears.
- Specify if you want to provide a **New Database Name**, else the database is migrated with the same name provided that a database with the same name does not exist in the cluster.
- The collections from the database are displayed with corresponding distribution strategies.
- Click **OK**.

The database is removed from the **Standalone Databases** tree and is displayed in the **Database Cluster**. During migration, the icon of the database is grey, with the status **Under Migration**.
- Once the database icon turns red, it indicates it is online and has been fully migrated.
Database Monitoring

NosDB Administrators’ Guide
29. NosDB Monitor

Databases can be monitored to view performance counters in live visual graphs using NosDB Monitor shipped with NosDB Enterprise edition. In NosDB Monitor, we have two types of dashboards:

- **Graphical View**
  
  In graphical dashboards, counters are shown in the form of graphs with respect to time. Counter value will be plotted along y-axis and time on x-axis. Graphical dashboards are helpful in the scenarios where you want to monitor cache for some specific time.

- **Report View**
  
  In report view dashboards, counter are shown in the form of numeric values. Report view dashboards are helpful in the scenarios when you want to monitor instantaneous performance of NosDB.

29.1. Starting NosDB Monitor

29.1.1. From NosDB Management Studio

- Right-click on a **database name** in **Object Explorer** and select **Monitor**.

  ![Object Explorer](image)

  OR

- Click on the **Monitor** button on the toolbar.

  ![Monitor button](image)

  **NosDB Monitor** launches, where you can monitor the performance of the database accordingly.
29.1.2. From Program Group

- Type "NosDB Monitor" into Start Program of Windows.
- Right click on NosDB Monitor to launch it.

Once NosDB Monitor is launched,

- Select Database Wizard appears.
- Select Node Name/IP Address for the node running the cluster.
- Click Next.

- Specify whether you want to monitor a clustered or standalone database.
- Click Next.
- Select the database for monitoring. You can select multiple databases as well.
- Click Finish.
NosDB Monitor opens up, with the option to monitor the database(s) in any dashboard view.

29.2. Using Server Dashboard

The Server Dashboard contains **Cluster Health, Event Logs** along with some mostly required cache counter graphs like, **NosDB CPU Graph, NosDB Network Graph, Cache Size, NosDB Request/sec Graph** and **NosDB Memory Graph**. Server Dashboard is generally a graphical view dashboard.
29.3. Using Report-View Dashboard

![Dashboard Image]

- Counters:
  - CPU Usage
  - Memory Usage
  - Network Usage
  - Requests/sec
  - Inserts/sec
  - Updates/sec
  - Deletes/sec
  - Fetches/sec
  - Average Insert/msec
  - Average Update/msec
  - Average Delete/msec
  - Average Fetch/msec
  - Average Query Execution/msec
  - Documents Count
  - Average Documents Size
  - Database Size
  - Journal Size
  - Persistence Queue Size
  - Persistence/sec
  - Pending Replicated Operations
  - Cache Count
  - Cache Size
  - Cache Ejection/sec
  - MapReduce Running Tasks
  - MapReduce Waiting Tasks
  - MapReduce Mapped/sec
  - MapReduce Reduced/sec
  - MapReduce Combined/sec
  - Data Balance/sec
  - State Transfer/sec

- Network Information:
  - Network Usage

- Server Details:
  - Server Dashboard / Report-View Dashboard

- Ready
29.4. Custom Dashboard

You can also add a custom dashboard to place the required counters on it for the monitoring using NosDB Monitor.

- Click on the File menu of the NosDB Monitor and select New Dashboard.
  OR
- Click on the button on the Toolbar.
- Create Dashboard Wizard appears. Provide Dashboard Name and select the Dashboard Layout (Graphical or Report View).
- Click Next.

Select number of rows and column for this dashboard and click on the Finish button.
• The new dashboard, **Dashboard1**, is created with specified number of rows and columns.

• The required counters can be dragged and dropped from **Database Server** category or from the **System** category in the **Toolbox** in the left pane.
30. NosDB Counters

NosDB is not only enriched with powerful monitoring facilities, but besides that it also provides different categories of counters for Windows Performance Monitor. This lets the clients monitor the overall performance of NosDB. These categories include cluster, client and server. They cover all aspects of NosDB monitoring including remote clients, shards, clustered operations and server counters.

Performance counters provide valuable information about system hardware, services, networks and applications that can be used to determine their performance. It helps in diagnosing problems, fine-tuning systems and applications, monitoring real-time application performance and resolving bottleneck cause of system components.

Performance counters can be viewed both from PerfMon and NosDB Monitor. The Statistics view in NosDB Management Studio is also used to view the performance statistics of a database or a shard.

Performance Counters

<table>
<thead>
<tr>
<th>Counter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fetches/sec</td>
<td>Number of successful Get operations per second.</td>
</tr>
<tr>
<td>Inserts/sec</td>
<td>Number of successful Insert operations per second.</td>
</tr>
<tr>
<td>Updates/sec</td>
<td>Number of successful Update operations per second.</td>
</tr>
<tr>
<td>Deletes/sec</td>
<td>Number of successful Delete operations per second.</td>
</tr>
<tr>
<td>Cache Hits/sec</td>
<td>Number of successful Get operations from cache per second.</td>
</tr>
<tr>
<td>Cache Misses/sec</td>
<td>Number of failed Get operations from cache per second.</td>
</tr>
<tr>
<td>Cache Evictions/sec</td>
<td>Number of documents evicted from cache per second.</td>
</tr>
<tr>
<td>Requests/sec</td>
<td>Number of requests being processed per second.</td>
</tr>
<tr>
<td>Journaled Operations</td>
<td>Number of the operations present in Journal.</td>
</tr>
<tr>
<td>Cache Count</td>
<td>Number of documents in the cache.</td>
</tr>
<tr>
<td>Cache Size</td>
<td>Size of the cache in bytes.</td>
</tr>
<tr>
<td>Average Update Time</td>
<td>Average time in microseconds taken to complete one update operation.</td>
</tr>
<tr>
<td>Average Insert Time</td>
<td>Average time in microseconds taken to complete one insert operation.</td>
</tr>
<tr>
<td>Average Delete Time</td>
<td>Average time in microseconds taken to complete one delete operation.</td>
</tr>
<tr>
<td>Average Fetch Time</td>
<td>Average time in microseconds taken to complete one fetch operation.</td>
</tr>
<tr>
<td>Average Query Execution Time</td>
<td>Average time in microseconds taken to complete execution of a query.</td>
</tr>
<tr>
<td>Average Document Size</td>
<td>Average size of documents in bytes.</td>
</tr>
<tr>
<td>Database Size</td>
<td>Size of the current database in bytes.</td>
</tr>
<tr>
<td>Documents Count</td>
<td>Number of documents in the database.</td>
</tr>
<tr>
<td>Documents Persisted/second</td>
<td>Number of documents persisted to store per second.</td>
</tr>
<tr>
<td>Pending Replicated Documents</td>
<td>Number of pending documents in primary node which are yet to be replicated.</td>
</tr>
<tr>
<td>Pending Persistent Documents</td>
<td>Number of documents which are yet to be persisted to store.</td>
</tr>
<tr>
<td>MapReduce Running Tasks</td>
<td>Number of tasks in execution state at a time.</td>
</tr>
</tbody>
</table>
### MapReduce Waiting Tasks
Number of tasks in waiting state when the maximum running tasks limit for execution has been reached.

### MapReduce Mapped/sec
Number of keys mapped per second in running tasks.

### MapReduce Reduced/sec
Number of keys reduced per second in running tasks.

### MapReduce Combined/sec
Number of keys combined per second in running tasks.

### Expiration Count
Number of documents expired.

#### Debug Counters

<table>
<thead>
<tr>
<th>Counter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>State transfer/sec</td>
<td>Number of documents the current node is either reading from other nodes or sending to other nodes during a state transfer mode.</td>
</tr>
<tr>
<td>Data Balance/sec</td>
<td>Number of items the current node is either reading from other nodes or sending to other nodes during a Data Load Balancing mode.</td>
</tr>
</tbody>
</table>

To enable Performance counters for both Standalone and Clustered Databases,

- Go to **Tools** in the toolbar and select **Options**.
- OR
  - Click on the ![icon](image) icon on the toolbar to open the **Options** window.
31. NosDB Management Studio Statistics

Databases can be monitored from within NosDB Management Studio as well, where replica nodes will also be displayed.

- Right click on a database name in Object Explorer and select Statistics.

- Database Statistics window pops up, which is an integration of the NosDB Monitor. Refer to Monitoring Databases Using Monitor for more details on using the monitor.
32. Windows Performance Monitor

NosDB publishes cache server counters in PerfMon under category NosDB. This category has all the counters related to the cache server. Follow the steps given below to monitor the NosDB counters through PerfMon tool:

- Click on the Windows Start menu, type PerfMon and press ENTER.
- PerfMon tool opens up. Click on the Performance Monitor under Monitoring Tools.
- Click on the cross (X) button to remove the default counter which is already added to it.
- Click on the plus (+) button to open the Add Counters dialog box.
- Using the vertical slider of available counters list box, scroll upward to find NosDB category.
Click on the down arrow head icon to expand the **NosDB** category. All of its counters are listed under it. Select the required counters from this list.

All running shards and databases appear inside of **Instances of selected objects** list box.

Select the required instance or simply click on the **<All instances>** and click on the **Add >>** button. All of the selected counters for all the selected instances of databases appear in **Added counters** list box on the right side.
Click **OK**. All of the selected counters will appear in PerfMon tool and can be monitored:
- You can also change the view of PerfMon from **Report** view to **Line** view:

- Click on the **Line** option of the drop down menu. This changes the PerfMon view from Report to Line.
- In **Line** view, counters values are displayed in the form of a continuous graph:
33. **Windows Event Logging**

NosDB provides support to view events in Windows Event Log. NosDB logs its important events in *Window Event Logs*. This allows monitoring all such events through Windows Event Viewer or with any other tool that supports Window Event LOGGER. Some of the events that are logged include:

- All NosDB installation events. These events are combination of successful or aborted installation.
- All errors encountered by NosDB Configuration, Database and Distributor services.
- Service start and stop events.
- Shard start and stop.
- Node joining or leaving a shard.

<table>
<thead>
<tr>
<th>Application</th>
<th>Number of events: 3,552 (!) New events available</th>
</tr>
</thead>
<tbody>
<tr>
<td>Level</td>
<td>Date and Time</td>
</tr>
<tr>
<td>Information</td>
<td>4/11/2016 3:20:00 PM</td>
</tr>
<tr>
<td>Information</td>
<td>4/11/2016 3:19:43 PM</td>
</tr>
<tr>
<td>Information</td>
<td>4/11/2016 3:19:31 PM</td>
</tr>
<tr>
<td>Information</td>
<td>4/11/2016 3:19:12 PM</td>
</tr>
<tr>
<td>Information</td>
<td>4/11/2016 3:19:11 PM</td>
</tr>
<tr>
<td>Error</td>
<td>4/11/2016 3:18:48 PM</td>
</tr>
</tbody>
</table>
34. NosDB Logs

NosDB provides a rich set of modules for logging and viewing run-time statistics of different components available in NosDB. It also provides different level of logging that includes log-files, report views, statistics and events.

NosDB logs the states of the components into dedicated log files. These log files are created whenever the component is started, and are locked until the component or its respective service is stopped. Log files encapsulate different level of information based on logging status.

NosDB creates each log file for the particular component with specific time stamps. This time initializes as the component or tool starts and continues until its status changes from “running” to “stopped”.

- **Database** ([InstallDir]\logs\dblog_timestamp.log)
- **Configuration Server** ([InstallDir]\logs\cslog_timestamp.log)
- **Manager** ([InstallDir]\logs\manager\mgrlog_timestamp.log)
- **Monitor** ([InstallDir]\logs\monitor\monitorlog_timestamp.log)
- **REST Package** ([InstallDir]\logs\REST\restlog_timestamp.log)

NosDB writes the log file with the standard default format that provides the user with familiarity while reading the file. It also logs the thread name whenever an operation is performed by any running thread. NosDB presents different stages of logging. These logging levels are configurable and are described as follows:

- **INFO**: Info level describes any useful information about any operation performed on database, such as starting and stopping state of state transfer operations. Info level is disabled by default.

- **ERROR**: This log flag provides the cause of errors that are raised during operation execution. This status log is useful in most of troubleshooting scenarios such as during start of shard nodes, or while establishing connection with the server.

- **WARNING**: This flag logs the storage status of any action or operation on the database, and will be logged once storage is about to exceed. This log information is useful to optimize database performance.

- **DEBUG**: This flag is disabled by default but the user can configure it. This log option prints detailed information about each operation like key and buckets during state transfer. This log information allows analyzing any database issues in detail whenever normal debugging is not possible and lets the user to drill down into any hidden issue. However, note that enabling debug logging is an expensive operation, performance wise.
34.1. NosDB Log Viewer

While log files are one of the essentials in any processing, their readability and organization is always overlooked, making the task tedious of finding the problem. NosDB Log Viewer is an interactive GUI tool dedicated completely to logging, as it categorizes your log entries into separate fields and lets you customize and search entries in a much more convenient manner. You can also view multiple log files for comparison.

The features provided by NosDB Log Viewer to enhance your readability are as follows:

34.1.1. Open Logs

- Click on the button or type CTRL+O to open the files you want to import into NosDB Log Viewer.
- After selecting the log file, click Open.

![Open Logs](image)

- The logs are imported into NosDB Log Viewer, showing the following information:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node</td>
<td>The node on which the operations have taken place.</td>
</tr>
<tr>
<td>Timestamp</td>
<td>The timestamp of each log, in the format yyyy-mm-dd h:m:s, ms.</td>
</tr>
<tr>
<td>AppDomain</td>
<td>The node/server/database name</td>
</tr>
<tr>
<td>LoggerName</td>
<td>Name of the component of which the information is logged.</td>
</tr>
<tr>
<td>ThreadName</td>
<td>Identification of thread</td>
</tr>
<tr>
<td>Level</td>
<td>Level shows the type of log, either it is INFO or ERROR or WARN</td>
</tr>
</tbody>
</table>
In order to view the Message of the log without scrolling, click on any log.

The message is displayed in the Message division at the bottom of the Log Viewer.

34.1.2. **Export Log Entries**

Click on the button or type **CTRL+S** to export/save the file to any location.
• After selecting the location, click **Save**.

### 34.1.3. Search Specific Logs

• To search for a certain term or option in the viewer, select the option from the **Any Field** drop down menu on the toolbar.

• If you want to find the term in the whole log file and not specify the parameters, select **Any Field**.

• In the text bar next to the **Any Field** drop down menu, enter the term you want to search. It displays suggestions containing that term in a drop down menu, easing your search process.
After entering the option, click on the **button.

If you have selected an option from the drop down menu, press **Enter** or click on the **button.

**34.1.4. Split Logs based on Field**

You can also view the logs by splitting the entries based on one or more unique field entries using the **button next to each field name. For example, in the **ThreadName** field, there are 3 unique threads in the sample log:

Selecting one of the options will display the logs for that **ThreadName** only:
In order to clear the split log view, click on the \( \text{button. It will now be displaying the option of Clear.} \)

**34.1.5. **Reload Log Files

If changes have been made to your databases or service, it is advisable to reload the log file to view the updated changes.

To reload the updated log file, click \( \text{button.} \)

**34.1.6. **Clear All Logs

In order to clear the logs from the viewer, click on the \( \text{button. All logs will be cleared.} \)

**34.1.7. **Toggle Bookmark on Log Entry

- In case you want to bookmark a log entry for future reference, click on the entry.

- The \( \text{button becomes active.} \)

- Click on the active \( \text{button to bookmark the entry.} \)

- It marks a bookmark icon next to the entry.
34.1.8. Bookmark Navigation

Once an entry has been bookmarked, the and buttons become active.

You can navigate from one bookmark to another using the Previous Bookmark and Next Bookmark button.

34.1.9. Settings

You can customize the Viewer according to your preference using the Settings button available on the toolbar.

1. Minimize to Tray

Next time you minimize the Log Viewer, it will be minimized in the tray. You can click on the tray icon to maximize the Log Viewer.
2. **Show multiple node files in colors**
If you add multiple log files of different nodes, you can check this option to differentiate between the nodes and increase readability.

<table>
<thead>
<tr>
<th>Node</th>
<th>Timestamp</th>
<th>AppC</th>
</tr>
</thead>
<tbody>
<tr>
<td>20.200.20.102</td>
<td>2015-12-02 15:00:09,367</td>
<td>Alachisoft.N</td>
</tr>
<tr>
<td>20.200.20.154</td>
<td>2015-12-02 02:57:36,406</td>
<td>Alachisoft.N</td>
</tr>
<tr>
<td>20.200.20.102</td>
<td>2015-12-02 14:59:55,074</td>
<td>Alachisoft.N</td>
</tr>
<tr>
<td>20.200.20.102</td>
<td>2015-12-02 14:59:36,346</td>
<td>Alachisoft.N</td>
</tr>
<tr>
<td>20.200.20.154</td>
<td>2015-12-02 02:56:52,742</td>
<td>Alachisoft.N</td>
</tr>
<tr>
<td>20.200.20.154</td>
<td>2015-12-02 02:56:37,828</td>
<td>Alachisoft.N</td>
</tr>
<tr>
<td>20.200.20.102</td>
<td>2015-12-02 14:59:30,613</td>
<td>Alachisoft.N</td>
</tr>
</tbody>
</table>

3. **Max log entries in live mode**
You can customize the maximum number of entries to be shown in the viewer while it is running.

4. **Columns to View**
You can customize the columns you want to view or remove from the viewer by checking/unchecking the options.

**Note:** Some settings will take effect after NosDB Log Viewer is restarted.
34.2. NosDB Monitor Logging

Using Report-View in NosDB Monitor, you can instantly monitor performance and also log these counters in a .CSV file. All the counter values which are currently in dashboard will be logged into a .CSV file after each second. You can start the performance counters logging instantly or you can schedule the logging on a specific time.

**Manual Logging**

Follow the below given steps to log the counters using NosDB Monitor:

- Go to the **Report-View Dashboard** (built-in Report-View dashboard or your created dashboard).
- Click on the **Start Logging** button icon on the report view control. This starts the logging.
- The log files can be found under `[InstallDir]/logs/monitor/server`.
- You can stop the performance counter logging by clicking the **Stop Logging** icon in the **Report-View Dashboard**.

**Schedule Logging**

Rather than manually starting and stopping logging, you can also schedule logging. This way you can obtain logging data for a particular period of time.

- Click on the **Schedule Logging** icon in **Report-View Dashboard**.
- **Schedule Logging** dialog box appears. Check the **Log counters at the following schedule** check box and configure the **Start Time** and **Stop Time**.
- Click **OK**. Logging will start at the specified time automatically.
Configuring NosDB REST API

NosDB Administrators’ Guide
35. Configuring Internet Information Services (IIS) for REST API

The following must exist in order to enable use of REST API with NosDB:

- **Microsoft Web Deploy** 3.5 or higher
- **Internet Information Services** (IIS) 7.0 or higher, with the following configurations:
  - Go to **Turn Windows Features on or off** option in **Control Panel**.
  - Check the following settings under the **Internet Information Services** structure:

![Windows Features](image_url)
Once IIS is enabled and if Web Deploy has been installed, NosDB will create the default Application Pool \textit{NosDBRESTAPIPool} in IIS Manager, with the applications ‘\textit{nosdbdata}’ and ‘\textit{nosdbmgmt}’. This enables you to query using REST API with the URL ‘\texttt{localhost/nosdbdata}’. However, the following changes must be made to the pool configuration:

- If the database has been created with \textbf{Windows Authentication}, change the identity of \textit{NosDBRESTAPIPool} to \textit{LocalSystem}.
- If it has been created with \textbf{NosDB Authentication}, change the identity to \textit{Custom} and provide the credentials for NosDB Authentication.
- Recycle the \textit{NosDBRESTAPIPool} process.

You can change the identity through the following steps:

- Right-click on \textit{NosDBRESTAPIPool} in Application Pools and select \textbf{Advanced Settings}.
- \textbf{Advanced Settings} window appears, go to \textbf{Process Model} -> \textbf{Identity}.
- Select the required identity through the drop down menu.
- Click \textbf{OK}.

If you wish to install the REST API Package manually, refer to \texttt{Manually Installing REST API Package in IIS}. 

36. Installing REST API Package on IIS

Pre-requisites

- Internet Information Services 7.0 or higher
- Microsoft Web Deploy 3.5 or higher

By default, if Web Deploy has been installed, NosDB creates an Application Pool in IIS named NosDBRESTAPIPool with the applications ‘nosdbdata’ and ‘nosdbmgmt’. This enables you to query using REST API with the URL ‘localhost/nosdbdata’.

However, if you want to install the package to any other pool with your own application names, you can manually install NosDB’s REST API package to either DefaultAppPool (created in IIS by default), any pre-existing application pool, or to a new one.

33.1. Adding New Application Pool

- Open Internet Information Services (IIS) Manager.
- Right-click on Application Pools under the left pane Connections and select Add Application Pool.

- Add Application Pool window appears.
- Specify a name for the pool and click OK.
• The pool is added to the list of **Application Pools**.

• You need to change the identity of the pool to **LocalSystem**.

• Right-click on the pool name and select **Advanced Settings**.

  
  ![Advanced Settings](image)

• **Advanced Settings** window appears, go to **Process Model** -> **Identity**.

• Select **LocalSystem** through the drop down menu.

• Click **OK**.
• **Advanced Settings** window appears, go to **Process Model** -> **Identity**.

• Select **LocalSystem** through the drop down menu.

• Click **OK**.

### 36.2. Installing Package to Existing Pool

If you do not wish to create a new pool for the applications, you can add the applications to any existing pool or the default created DefaultAppPool.

However, the following configurations must be made to the existing pools:

- **.NET CLR version** is **v4.0.030319**
- **Managed pipeline mode** is **Integrated**
- **Identity** of the pool is **LocalSystem**
36.3. Adding Application to Pool

- Right-click on Default Web Site and select Add Application...

- Add Application window appears.

- Specify the Alias for the REST API package. This Alias will be used in the URL instead of the default package names provided by NosDB.

- Select the Application Pool you created by clicking on Select...

- Provide the Physical Path to the package provided by NosDB.
  - The REST data package for NosDB is found at \InstallDir\REST\nosdbdata
  - The REST management package for NosDB is found at \InstallDir\REST\nosdbmgmt

- Click OK.
- The package is installed as an application in under **Default Web Site**.
- The URL for using REST API in NosDB will now start as `'localhost/data/'`

- **Recycle** the configured Application Pool.
- The **Applications** count against your installed Application Pool is incremented.
Using Microsoft Power BI with NosDB

NosDB Administrators’ Guide
37. Using Microsoft Power BI with NosDB

This feature is only available in NosDB Enterprise Edition.

NosDB's RESTful APIs have been developed using Open Data Protocol (OData), which means that any application built on OData can fully utilize NosDB's functionality without any code changes. One interactive tool particularly used for big data analysis is Power BI from Microsoft, which uses OData to connect to the underlying databases.

Thus, you can analyze your distributed NosDB data in Power BI to predict and depict trends and data behavior.

Prerequisites

The following must exist in order to enable use of REST API with NosDB:

- Internet Information Services (IIS) 7.0 or higher, configured according to the chapter Configuring IIS for REST API.
- Microsoft Web Deploy 3.5 or higher
- Database (NosDB)

Making requests in REST is as simple as sending the request to the server via a dedicated URL. This implies that each collection has a dedicated URL to differentiate it. The URL contains the connection string name and collection name:

`localhost/nosdbdata/clustered/northwind/products`
37.1. Selecting NosDB as Data Source

- Start **Power BI**.

- To load NosDB’s data for visualization, click on the **Get Data** option if you are using Power BI for the first time.

- The following **Get Data** window appears, offering a wide and diverse range of data sources to import the data from.

- Select **OData Feed** as NosDB’s REST API has been built by extending OData.
- The following **OData Feed** window appears, which requires entering the URL of the collection you want to analyze.

- Enter the respective URL.
37.2. **Loading NosDB Data into Power BI**

- The following screen appears showing the URL's data in tabular form. This has been made using the schema provided dynamically to OData by NosDB.

- To load the data into Power BI, click **Load**.

- The collection data is loaded into the model. Note that mapping data takes time according to the size of collection.
• By default, the name of the collection in Power BI is “Query1”. The name of the collection can be renamed once it is loaded.

![Load](image)

• Once the data is loaded, Power BI is ready for analyzing. The collection fields are shown in the **Fields** column.

![Fields](image)

Note that Power BI does not show any embedded documents.

• Check on any of these fields to start analysis. There should be at least one numerical field in order to visualize the data.
37.3. Loading Multiple Collections

You can also load multiple collections on the same page if required. There are multiple ways to load data into Power BI:

37.3.1. Loading NosDB data

- Click on the **Get Data** option in the **Home** toolbar.
- A drop down menu appears, showing the options to import data from.
- Select **OData Feed**.

---

This page is extracted from the NosDB Administrators' Guide by Alachisoft. Copyright © 2005 - 2017 Alachisoft | 183
37.3.2. **Loading data from recently added sources**

- Click on the Recent Sources option on the Home toolbar.
- Select the source you want to load.

Load the data from the source as explained in [Loading Data into Power BI](#).
• Load the data from the source as explained in Loading Data into Power BI.

37.3.3. Creating table within Power BI

You can locally create a table using Power BI. Note that this data will not be persisted to the NosDB storage nor will be used external to Power BI.

• Click on the Enter Data option on the Home toolbar.

• The Create Table window appears.

• Enter the relevant data and load the table.
37.2. **Visualize Data**

### 37.2.1. Selecting Visualization Type

You can change the type of visual according to your preferences through the **Visualization** column which shows numerous chart types.

![Visualization Selection](image)

### 37.2.2. Selecting Legend or Axis

Once the visual type is selected, select the field to be analyzed in the axis:

![Axis Selection](image)
37.2.3. Selecting Values

The analysis can be shown by selecting the values for comparison, for example, we want to analyze whether the *UnitPrice* of a product impacts the *UnitsOnOrder*.

The visual displays the corresponding analysis:
37.3. Managing Relationships

Power BI automatically detects any relationship between the tables based on the attribute name. However, you can also define relationships manually as well.

- Once multiple tables have been added to Power BI, you can create a visual with values from the respective tables to view how they affect each other.
In order to view the relationship and cardinalities between multiple tables, click on the **Relationships** button on the left columnar toolbar.
37.4. Querying NosDB Data

You can dynamically query on the local data to depict trends and easily gather useful information.

Note that querying takes place over the loaded data which is local to Power BI. In case any change has occurred in NosDB data, the data will have to be reloaded by refreshing the data.

37.4.1. Filter based Querying

- Select a chart to perform the filtration on.

- In the Filters column, click on the attribute name on which the filtering will take place. The attribute must be numerical.

- A drop down menu of available filters is displayed.

  ![Filter Menu Example](image)

- Select the required filter and enter its corresponding value.

- Click on **Apply Filter** after specifying the filter.
37.4.2. Querying through Query Editor

For advanced querying, Power BI offers a Query Editor to perform querying over the loaded data.

- Click on Edit Queries on the Home toolbar.
- The Query Editor opens up, showing all loaded tables.
You can now perform in-depth querying over the data as explained in [Query Overview in Power BI documentation](#).