NosDB vs DocumentDB

Comparison

For .NET and Java Applications

NosDB 1.3 vs. DocumentDB v8.6

This document compares NosDB and DocumentDB. Read this comparison to:

- Understand NosDB and DocumentDB major feature differences
- See how NosDB and DocumentDB compare on qualitative aspects such as performance, scalability, high availability, data reliability, administration, and more.

Table of Content

Disclaimer

The comparison provided in this document is for the purpose of helping you get a better understanding of NosDB versus DocumentDB. Information obtained about DocumentDB is from the freely available downloads, documents, and forums.

We did not conduct any scientific benchmarks for performance and scalability of DocumentDB so our assessment about it may be different from yours. NosDB benchmarks are already published on our website (http://www.alachisoft.com) for you to see.

Additionally, we have made a conscious effort to be objective, honest, and accurate in our assessments in this document. But, any information about DocumentDB could be unintentionally incorrect or missing, and we do not take any responsibility for it.

Instead, we strongly recommend that you do your own comparison of NosDB with DocumentDB and arrive at your own conclusions. We also encourage you to do performance benchmarks of NosDB and DocumentDB both in your environment for the same purpose.

1 Executive Summary

This document compares NosDB with DocumentDB, and contrasts their significant differences. This comparison focuses on all the major areas that a good NoSQL database should provide.

Feature	DocumentDB	NosDB
Platform & Technology		
DB Server	C++ & Favors Linux (No Server-side .NET)	100% Native .NET
.NET API	Supported	Full support
Java API	Supported	Full support
REST API	Supported	Full support
Data Model		
JSON Documents	Supported	Full support
Normalized Data Model	Supported	Full support
Embedded Data Model	Supported	Full support
Single-Shard Collections	Supported	Full support
Capped Single-Shard Collections	Not Supported	Full support
Multi-Shard Collections	Supported	Full support
Attachments	Not Supported (Preview)	Full support
CLR Triggers	Partial support	Full support
CLR User Defined Functions (UDF)	Partial support	Full support
Stored Procedures	Supported	No Support
Performance and Scalability		
Database Performance	*Please verify yourself*	Excellent
Scalability	*Please verify yourself*	Excellent
Multiple Shards & Replicas	Supported	Full support
Hash-Based Distribution	Supported	Full support
Range-Based Distribution	Supported	Full support
Custom Distribution	Supported	No Support
Read-load Shared by Replicas	Not Supported	Full support
Async Operations	Supported	Full support
Fire & Forget Operations	Not Supported	Full support

Feature	DocumentDB	NosDB
Bulk Operations	Not Supported	Full support
Database as Service	Supported	No Support
On Premise	Not Supported	Full support
High Availability		
Dynamic Clustering	Not Supported	Full support
Connection Failover	N/A	Full support
Dynamic Configuration	Supported	Full support
Shard Replicas (dedicated & shared replicas)	Not Supported	Full support
Split Brain Avoidance	N/A	Full support
Auto State Transfer within Shard	N/A	Full support
Auto State Transfer across Shards	Not Supported	Full support
Journaling (for data consistency)	N/A	Full support
Multi-Data Center Support	Supported	Full support
Upgrade without Downtime	Supported	Full support
In-Memory Caching		
DB Server In-Memory Caching	N/A	Full support
Client-side In-Memory Caching	Not Supported	Full support
Pluggable Distributed Cache (NCache)	Not Supported	Full support
Cache Documents, Collections, Query Results	N/A	Full support
Client Cache Supported	Not Supported	Full support
Synchronize Cache with NosDB	Not Supported	Full support
SQL Support		
SQL (standard + with JSON)	Supported	Full support
Advanced SQL (with JSON)	Supported	Full support
LINQ	Supported	Full support
Database Management (DDL)	Not Supported	Full support
Indexing	Supported	Full support
Geo Indexing	Supported	No Support
SQL Dependency	Not Supported	Full support

Feature	DocumentDB	NosDB
Big Data & Business Intelligence		
Map Reduce	Not Supported	Full support
Aggregator	Not Supported	Full support
Security		
Windows Authentication	Supported	Full support
NosDB Authentication	Supported	Full support
Authorization	Supported	Full support
SSL Support	Only through HTTPS	Full support
Management Tools		
NosDB Management Studio	Supported	Full support
NosDB Monitor	Supported	Full support
PowerShell Management	Not Supported	Full support
PerfMon Counters	Not Supported	Full support
Windows Event Log	Not Supported	Full support
Server Management API (.NET, REST)	Not Supported	Full support
Full Backup/Restore	Supported	Full support
Differential Backup/Restore	Not Supported	Full support
Import/Export	Supported	Full support
Move Single-Shard Collection Tool	Not Supported	Full support
Stand-alone to Sharded DB Migration	Not Supported	Full support
Third-Party Integrations		
Visual Studio 2013/2015	Supported	Full support
ADO.NET Provider	Not Supported	Full support
Microsoft Power BI	Supported	Full support

2 Qualitative Differences Explained

2.1 Platform & Technology

This section goes over the technology used to develop the NoSQL database product and what platforms & technologies does it support and whether it provides native .NET support or not.

For example, just because a NoSQL product provides a .NET client API does not mean it is fully consistent with the .NET development environment. This section covers this aspect.

Feature Area	DocumentDB	NosDB
DB Server	C++ (Favors Linux) DocumentDB is developed in C++ mainly for Linux platform. And, although it is ported to Windows, Linux seems to be its main focus.	Native .NET (Windows) 100% native .NET product. Both server and client portions are developed in .NET. This means that database servers run natively on Windows and are fully compatible with .NET application stacks. NosDB fully supports Windows 2012 & 2016 including PerfMon counters and Windows Event Logs.
.NET API	Supported	Full support NosDB provides a rich native .NET Client API to give your .NET applications access to all the features of NosDB.
Java API	Supported	Full support NosDB provides a native Java API that is identical to the .NET API in features. This Java client API is available on both Windows and Linux platforms
REST API	Supported	Full support NosDB provides REST API with most features of the regular .NET API. You can use this REST API from any programming language including Node.js, Python, PHP, and others.

2.2 Data Model

NosDB is a NoSQL document database and supports JSON format. JSON documents provide a very flexible schema-less data model and allows you to speed up your application development and also easily handle frequently changing requirements. Below is a list of features related to the data model.

Feature Area	DocumentDB	NosDB
JSON Documents	Supported	Full support NosDB is a NoSQL document database which supports JSON format. JSON is a very flexible schema that allows you to rapidly develop your applications and then address your quickly changing requirements afterwards.
Normalized Data Model	Supported	Full support NosDB allows you to use a normalized JSON schema as compared to an embedded one. Normalized schema means each sub-document is stored separately and a relationship between them is maintained through a related key. You can then run SQL queries to find related documents of a given document.
Embedded Data Model	Supported	Full support Unlike a Normalized Data Model, an Embedded Data Model lets you embed sub-documents inside the main document. Then, you can treat the sub-document as an attribute of the main document but with further attributes of its own. You can specify JSON sub-document attributes in SQL queries to make your search more powerful.
Single-Shard Collections	Supported	Full support NosDB provides multiple types of collections. The first one is Single-Shard Collection where the entire collection is kept within one shard even if you have multiple shards. This is often very useful and allows you to specify location affinity

Feature Area	DocumentDB	NosDB
		to data. The limitation is that your collection cannot grow beyond one shard.
Capped Single-Shard Collections	Not Supported	Full support Capped Collections are also Single-Shard Collections but with circular behavior. You specify size of a Capped Collection and then documents get overwritten in a FIFO manner once the Capped Collection reaches it "Cap". Capped collections are very useful in situations where you're saving data in a fixed-sized log.
Multi-Shard Collections	Supported	Full support A Multi-Shard Collection is extremely scalable because it spans over all the shards in the database. And, this means you have a lot of storage space available for this collection because you can always add more shards to the database. A multi-shard collection uses either Hash-based distribution or Range-based distribution across multiple shards.
Attachments	Not Supported (In Preview)	Full support Attachments are BLOBs that are stored in the database as attributes of JSON documents. But, since they're very large, they cannot be read or written in one call and multiple calls are needed to read and write chunks of them.
CLR Triggers	Partial support DocumentDB does not support Triggers and only allow you to register validators. Unlike Triggers these validators execute only on data insert and updates. These scripts are written in JavaScript therefore they run out-of-process of the DocumentDB instance.	Full support CLR Triggers are .NET functions registered against database operations on a collection. CLR Triggers are executed inside the NosDB database server as soon as the associated operation. There are Pre- and Post-Triggers for Insert, Update, and Delete operations on a collection.

Feature Area	DocumentDB	NosDB
CLR User Defined Functions (UDF)	Partial support User Defined Functions are also written in JavaScript therefore they run out-of-process of the DocumentDB instance.	Full support User Defined Functions (UDFs) are collection-level CLR functions, written in .NET, which can be used in SELECT queries just like built-in SQL functions. They are particularly useful in cases where a certain calculation is frequently required to be performed. CLR functions also run inside the NosDB database server.
Stored Procedures	Supported	No Support

2.3 Performance and Scalability

Performance is defined as how fast database operations are performed at a normal transaction load. Scalability is defined as how fast the same database operations are performed under higher and higher transaction loads. NosDB is extremely fast and scalable.

See NosDB benchmarks at Performance and Scalability Benchmarks.

Feature Area	DocumentDB	NosDB
Database Performance	*Please verify it yourself*	Extremely good NosDB is extremely fast and provides parallel reads and writes in the database server for maximum throughput. It also uses in-memory caching within the database server for further performance boost. Finally, NosDB allows you to use client-side in-memory caching to cache documents, collections, and query results while keeping them synchronized with the database. This reduces database trips and significantly improves your application performance.
Scalability	*Please verify it yourself*	Extremely good NosDB provides linear scalability and allows you to grow the database cluster

Feature Area	DocumentDB	NosDB
		to 100's of servers. As a result, your database tier never becomes a scalability bottleneck.
Multiple Shards & Replicas	Supported	Full support NosDB database cluster consists of multiple shards (partitions). You can also create shard-replicas. All of this allows you to linearly scale your transaction and data load while at the same time achieve high availability through replication.
Hash-Based Distribution	Supported	Full support NosDB distributes data to multiple shards based either on Hash-based or Range-based distribution. Hash-based distribution uses hashing algorithm to evenly distribute data across all shards. And, database clients are notified about all the shards and the distribution information so they can directly go where the data is. This boosts performance.
Range-Based Distribution	Supported	Full support Range-Based distribution allows the user to specify multiple sets of ranges against a given key. And, each range is then mapped to a shard and all documents belonging to this range are stored in this shard.
Custom Distribution	Supported	No Support
Read-load Shared by Replicas	Supported	Full support NosDB allows you to create both shared replicas and dedicated replicas. Shared replicas are created on active servers of other shards whereas dedicated replicas are created on separate servers within the same shard. NosDB allows you to send SELECT queries or other READ operations to dedicated replicas so their resources are

Feature Area	DocumentDB	NosDB
		also used to handle transaction load. This improves overall performance and scalability.
Async Operations	Supported	Full support Asynchronous add, insert, and remove operations are supported. Asynchronous operations return control to the application and perform database operation in the background and a "task tracker" is returned to the application. This "task tracker" then returns success or failure when the operation is completed. This greatly improves application response time.
Fire & Forget Operations	Supported	Full support These are asynchronous operations that don't return any data or callback to the client. The operation is assumed to have been done successfully. This operation decreases bandwidth usage and improves application response time greatly.
Bulk Operations	Supported	Full support Bulk Get, Add, Insert, and Remove operations are supported. You can fetch multiple documents in one database trip. And, this greatly improves performance.
Database as Service	Supported	No Support
On Premise	Not Supported	Full support

2.4 High Availability

A NoSQL database runs in production environment and your applications depend on it. Therefore, a good NoSQL database must be highly available. And, since a NoSQL database is usually sharded (distributed), high availability becomes more complicated.

To assess high availability, try to answer the following questions. Are you able to perform the following operations at runtime without stopping the database cluster or your application?

1. Add or remove database servers at runtime without stopping the database cluster
2. Make database cluster config changes without stopping the database cluster
3. Add or remove web/application servers without stopping the database cluster
4. Have failover support in case any server goes down (meaning are database clients are able to continue working seamlessly).

NosDB provides a self-healing dynamic cache clustering that makes NosDB highly elastic.

Feature Area	DocumentDB	NosDB
Dynamic Clustering	Supported	Full support NosDB creates a highly dynamic database cluster. This lets you add or remove database servers and shards at runtime without any interruption to the database cluster or your application. Data is automatically rebalanced whenever a new shard is added or removed. This process is called state transfer.
Connection Failover	Supported	Full support NosDB provides connection failover support between your application and database servers and also within the database cluster. In case of a database server failure, NosDB clients continue working with other servers and replicas in the cluster without any interruption.
Dynamic Configuration	Supported	Full support NosDB provides highly dynamic and available configuration management. There is a Configuration Manager that uses either a single node or builds a two-node configuration cluster. All servers in the database cluster talk to this Configuration Manager. NosDB clients also learn about all the database servers and a variety of other configuration at runtime from the

Feature Area	DocumentDB	NosDB
		Configuration Manager.
Shard Replicas (dedicated & shared replicas)	Supported	Full support A NosDB cluster is divided into multiple shards (partitions). Each shard consists of one primary and multiple secondary servers (replicas). The replicas can either be created on separate servers (called dedicated replicas) or on a primary server of another shard (shared replicas). If a primary server fails or crashes, the secondary servers go through a vote to promote one of them to become the new primary. This provides uninterrupted operations for the clients. NosDB also sends READ operations to replicas so as to use their resources and improve scalability.
Split Brain Avoidance	Supported	Full support NosDB provides split-brain detection and avoidance. A split brain occurs when a primary server within a shard loses contact with the Configuration Manager or some or all of secondary servers. Split brain avoidance includes various rules where a new primary is elected or if the existing primary has majority of secondary nodes with it, then it continues. Either way, split-brain situations are automatically handled by NosDB.
Auto State Transfer within Shard	Not Supported	Full support When a server (primary or secondary) rejoins the shard after a failure or network disconnection, the active running servers start a state transfer within the shard and bring the rejoining server up to date. State transfer within a shard is also executed when a new secondary is added in the shard. Hence you can say a single shard is self-

Feature Area	DocumentDB	NosDB
		healing in case of any failures or it auto adjusts to changes in the shard cluster.
Auto State Transfer across Shards	Supported	Full support State transfer across shards is executed whenever a shard is added or removed in the cluster. NosDB automatically rebalances the data across shards and clients are automatically notified of a new data distribution map.
Journaling (for data consistency)	Supported	Full support Before NosDB updates database on the disk, it writes an entry into a journal (on the disk). The journal write is very fast because it is local and does not involve a lot of work that a full database update does. After this journal update, the database server updates the database on the disk. By using a Journal, the chances of losing database update drops significantly because this update already exists in the Journal. So, a Journal provides extra safety in ensuring data consistency.
Multi-Data Center Support	Supported	Full support NosDB allows you to configure your database to span multiple data centers. First is the active-passive configuration where only replicas exist in the passive datacenter. And, NosDB sends READ operations to these passive datacenter replicas based on the “nearest replica” algorithm for applications running in that data center. Second is to have active-active database with partitioning so some of the shards exist in data center 1 while others exist in datacenter 2. And, applications have access to all the shards.
Upgrade without	Supported	Full support

Feature Area	DocumentDB	NosDB
Downtime		NosDB allows you to upgrade to a newer version without having to stop the database server. You can upgrade one server at a time and keep the upgraded server in the compatibility mode to the older version until all servers are upgraded and then you can turn off the compatibility mode so new version is active.

2.5 In-Memory Caching

In-Memory caching speeds up database because memory is much faster than disk. This section focuses on the In-Memory Caching features that a good NoSQL database should have to achieve good performance.

Feature Area	DocumentDB	NosDB
DB Server In-Memory Caching	Supported	Full support NosDB database server has an internal in-memory cache. All updates are first written to this cache before being written either to the Journal or the database storage itself. This speeds up the database server performance greatly. You can specify if NosDB should return control to the application immediately after writing to the cache or should it also update the database first. Returning after the cache update is really fast.
Client-side In-Memory Caching	Not Supported	Full support NosDB also provides a built-in client-side In-Memory Cache. NosDB internally uses NCache for this feature. Client-side cache allows NosDB to cache JSON documents, collections, and query results and keep it within the NosDB client application memory (InProc). This makes accessing them super-fast. Also, the client-side cache is kept synchronized with the NosDB database

Feature Area	DocumentDB	NosDB
		through NosDB's SqlDependency feature so you don't have to worry about cached data becoming stale.
Pluggable Distributed Cache (NCache)	Not Supported	Full support If you have a need to cache a lot of data then a Client-Side In-Memory Cache will not be sufficient (because all cached data is kept within your client application process), then you should plug-in NCache as a caching tier without any code changes. NCache is industry's .NET distributed caching leader for 10 years and is really popular among .NET developers.
Cache Documents, Collections, Query Results	Not Supported	Full support You can cache JSON documents, collections, and query results. You can do all of this through passing parameters options in NosDB method calls. Therefore, no extra programming is required for you to do this. You can do all of this regardless of whether you're using Client-Side In-Memory Cache or NCache Integration.
Client Cache (Near Cache)	Not Supported	Full support If you choose to use NCache as a caching tier, you still have the option of using Client Cache feature of NCache. Client Cache is a local cache (InProc or OutProc) that sits on your application server but is connected to the caching tier and keeps the Client Cache synchronized with the caching tier. This allows you to cache frequently used data within application process and greatly speed up your application. NosDB uses this feature if enabled to cache your JSON documents, collections, and query results within application process.

2.6 SQL Support

A good NoSQL database allows you to search data through sophisticated SQL-like queries. This section describes all the important features in this area.

Feature Area	DocumentDB	NosDB
Basic SQL (standard)	Not Supported	Full support You can easily use the standard SQL for SELECT, INSERT, UPDATE, and DELETE statements. Other than joins and nested queries, NosDB SQL supports most of standard SQL syntax. For joins and nested queries, you can implement a solution within your application layer or use MapReduce in .NET. This allows you to easily migrate many relational database applications to NosDB.
Advanced SQL (w JSON)	Not Supported	Full support You can use SQL with JSON attributes to search embedded documents thereby avoiding any need for joins or nested queries. NosDB support for SQL is a very powerful way for you to access data in the database.
LINQ	Supported	Full support NosDB fully supports LINQ to search the database. LINQ is a very popular object searching mechanism for .NET developers.
Database Management (DDL)	Not Supported	Full support You can use the standard Data Definition Language (DDL) syntax of SQL to create and manage databases, collections, indexes, and more.
Indexing	Supported	Full support NosDB allows you to define indexes on JSON attributes as well as keys.

Feature Area	DocumentDB	NosDB
		Indexes can be configured to either be in-memory or persisted to disk. Keeping it in-memory improves indexing performance. In-Memory indexes are recreated if a database server restarts. NosDB also supports Compound Indexing. A compound index consists of one or more attributes combined together to form a composite index. Compound Index helps to perform AND, OR and other operation in queries faster thus significantly improves query performance.
Geo Indexing	Supported	No Support
SQL Dependency	Not Supported	Full support NosDB provides SqlDependency feature similar to SQL Server. SqlDependency allows your application to specify a SQL search criteria based dataset to monitor. Then, if any data is added, updated, or removed from this dataset, NosDB notifies your application through a database notification. This allows your application to then take the appropriate action. NosDB client-side In-Memory Cache and NCache both use this feature to keep their caches synchronized with NosDB database.

2.7 Big Data & Business Intelligence

A NoSQL database is able to store a lot of data due to its distributed storage. This means that Big Data processing can be done on a NoSQL database. This section covers all the Big Data processing features that a good NoSQL database should have.

Feature Area	DocumentDB	NosDB
Map Reduce	Partial support MapReduce code is written in JavaScript and runs out-of-process of the	Full support NosDB provides a .NET based MapReduce

Feature Area	DocumentDB	NosDB
	DocumentDB instance.	Framework where you can run your MapReduce code on all the shards in parallel. This distributed processing is done close to the data and allows you to handle Big Data processing for business analytics purposes.
Aggregator	Partial support MapReduce code is written in JavaScript and runs out-of-process of the DocumentDB instance.	Full support As part of the MapReduce Framework, NosDB also provides an Aggregator interface that you can implement. And, then you can aggregate results of a MapReduce execution into statistical data.

2.8 Security

Like any database, a NoSQL database must provide adequate security mechanism. This section covers features in this area.

Feature Area	DocumentDB	NosDB
Windows Authentication	Supported	Full support NosDB provides a Windows Authentication option where you can authenticate users against Active Directory or LDAP.
NosDB Authentication	Supported	Full support NosDB also provides its own NosDB Authentication. You create users in NosDB and then all those users must authenticate against NosDB before they can access their database.

Feature Area	DocumentDB	NosDB
Authorization	Supported	Full support NosDB provides role-based authorization to determine what each user is allowed to do.
SSL Support	Supported	Full support NosDB applies Transport Layer Security (TLS) which ensures that data will be encrypted before being transferred over the network between clients and server.

2.9 Management Tools

Cache administration is a very important aspect of any NoSQL database. A good cache should provide the following:

1. GUI based and command line tools for database administration including database creation and editing/updates.
2. GUI based tools for monitor the database activities at runtime.
3. Database statistics based on PerfMon (since for Windows PerfMon is the standard)

NosDB provides powerful support in all these areas.

Read more about it at Administration and Monitoring Tools.

Feature Area	DocumentDB	NosDB
Admin Tool (GUI)	Supported	Full support NosDB Management Studio is a powerful GUI tool for managing clusters and shards, databases, collections, and more. It gives you an explorer style view and lets you quickly administer all the shards from a single place.
Monitoring Tool (GUI)	Supported	Full support NosDB Monitor is a powerful GUI tool that lets you monitor NosDB cluster wide activity from a single location. It also lets you monitor all of NosDB clients. You can incorporate even non-NosDB PerfMon counters in it for comparison

Feature Area	DocumentDB	NosDB
		with NosDB counters. This real-time comparison is often very important.
PowerShell Management	Supported	Full support NosDB integrates into PowerShell seamlessly. You can create clusters and add or remove shards, replica sets, collections and much more. Use these tools from your scripts and automate various admin operations.
PerfMon Counters	Not Supported	Full support NosDB provides a rich set of counters for PerfMon. This allows you to monitor NosDB from third-party Windows monitoring tools. It also allows you to combine your application counters with NosDB to get a comparative idea of how all of these things are working together.
Windows Event Log	Not Supported	Full support NosDB logs all important events in Windows Event Log. As a result, you can monitor for these events from third party tools and even notify people when these happen.
Server Management API (.NET, REST)	Supported	Full support In addition to providing NosDB Management Studio and PowerShell Integration, NosDB gives you server management API in .NET and REST. This way, you can write scripts and programs to directly manage various aspects of NosDB.
Full Backup/Restore	Supported	Full support NosDB provides a Full Backup/Restore tool. Use this tool to do periodic backups so in case your database crashes, you can restore it from the last backup.

Feature Area	DocumentDB	NosDB
Differential Backup/Restore	Supported	Full support NosDB also provides a Differential Backup/Restore tool that only backs up data and configuration that has changed since the last Full Backup. This reduces your backup size & time and allows you to restore until the last Differential Backup easily.
Import/Export	Supported	Full support NosDB allows you to import data into the database from known data formats like CSV or JSON documents. Similarly, you can export data from your database into the same formats.
Move Single-Shard Collection Tool	Not Supported	Full support When you create a Single-Shard Collection, the entire collection lives in one Shard. So, if you need to bring that Shard down, you would need to move this collection to another Shard. Or, if you feel that there is too much data in one Shard and you want to move some of the collections to other Shards, you can do that. This tool allows you to do all of that.
Stand-alone to Sharded DB Migration	Not Supported	Full support NosDB allows you to create Stand-alone databases. This allows you to migrate a Stand-alone database into a Sharded Cluster.

2.10 Third Party Integrations

A NoSQL database for .NET exists in the overall .NET development eco system. This sections covers all the different third-party integrations that are important for .NET development community.

Feature Area	DocumentDB	NosDB
Visual Studio 2013 / 2015	Not Supported	Full support

Feature Area	DocumentDB	NosDB
		You can use NosDB from within your favorite .NET IDE, namely Visual Studio. NosDB is fully integrated with Visual Studio 2013/2105. You can create and manage clusters, database, and collections. And, you can also run SQL queries. This speeds up your development time because you don't have to go back and forth between different environments. Also, being able to run SQL queries from within Visual Studio makes your coding much easier because you can test your SQL statements before putting them into your code.
ADO.NET Provider	Not Supported	Full support NosDB provides support for a subset of standard SQL (without joins and nested queries) so you develop applications in a familiar environment. Similarly, NosDB has developed an ASP.NET provider so you can do your database programming with an API you already know. Additionally, you can automatically plug-in third-party controls in your .NET applications with NosDB through ADO.NET. And, you can use third-party tools to access the NosDB database.
Microsoft Power BI	Not Supported	Full support NosDB is integrated with Microsoft Power BI, a suite of business analytics tools to analyze data and share insights. Microsoft Power BI allows you to quickly view all your business intelligence data in rich dashboards available on every device.

3 Conclusion

As you can see in a very detailed fashion, we have outlined all of NosDB features and all the corresponding DocumentDB features or a lack thereof. I hope this document helps you get a better understanding of NosDB versus DocumentDB.

Please read more about NosDB and also feel free to download a fully working 60-day trial of NosDB from:

- NosDB details.
- Download NosDB.

