AWS RDS MySQL to Azure MySQL Online Migration

We will be migrating a MySQL RDS instance hosted in AWS to Azure MySQL using online migration. Online migration continues to send database updates from the source to the target server up until the point when cut over is initiated, reducing the downtime to a minimum.

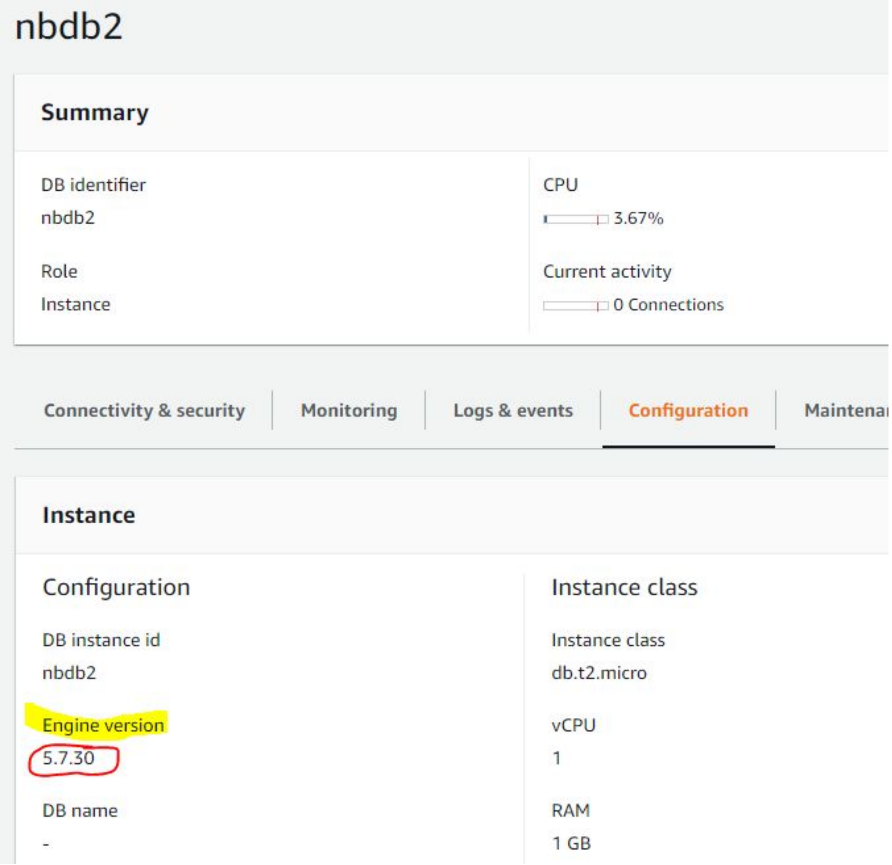
The Microsoft documentation for migration a RDS MySQL database to Azure MySQL is at:
https://docs.microsoft.com/en-us/azure/dms/tutorial-rds-mysql-server-azure-db-for-mysql-online

- Setup AWS RDS MySQL for migration

First we need to verify that the MySQL database we're trying to migrate is either version 5.6 or 5.7.

Also verify that you are using InnoDB engine, as that is the only one supported in Azure MySQL Database.

Open the AWS Management Console and navigate to your RDS instance. Click on the Configuration tab and verify the Engine.



nbdb2

Summary

DB identifier
nbdb2

Role
Instance

CPU
3.67%

Current activity
0 Connections

Instance

Configuration

DB instance id
nbdb2

Engine version
5.7.30

DB name
-

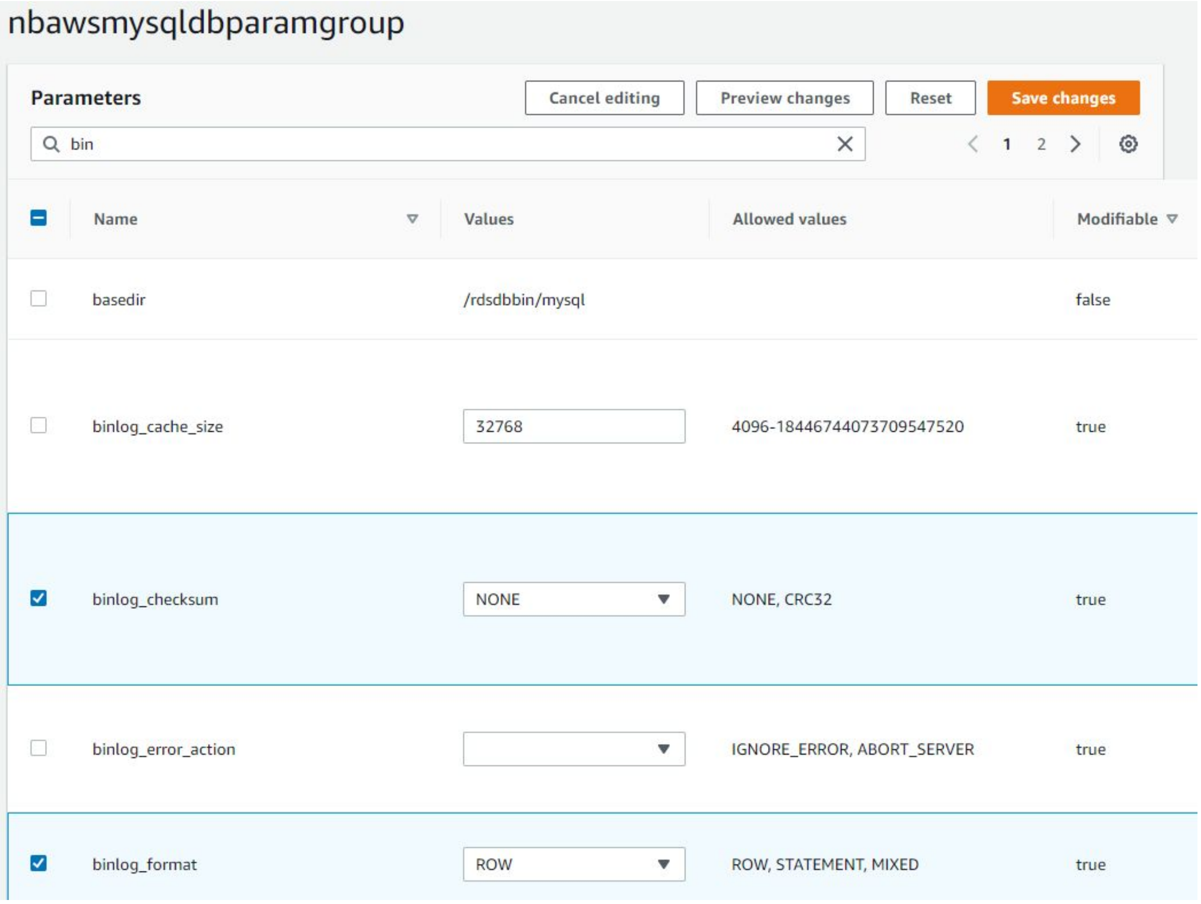
Instance class

Instance class
db.t2.micro

vCPU
1

RAM
1 GB

- Create a new parameter group for the MySQL source database and set binlog_format = row and binlog_checksum = NONE
- Associate the parameter group with the MySQL RDS instance. A reboot is required so you may need to choose 'Apply during the maintenance window' for a production instance.



nbawsmysqldbparamgroup

Parameters

bin

▬	Name	Values	Allowed values	Modifiable
□	basedir	/rdsdbbin/mysql		false
□	binlog_cache_size	32768	4096-18446744073709547520	true
☑	binlog_checksum	NONE	NONE, CRC32	true
□	binlog_error_action		IGNORE_ERROR, ABORT_SERVER	true
☑	binlog_format	ROW	ROW, STATEMENT, MIXED	true

- Open the Heidi SQL client and export the schema from the source database to a file on your local computer:

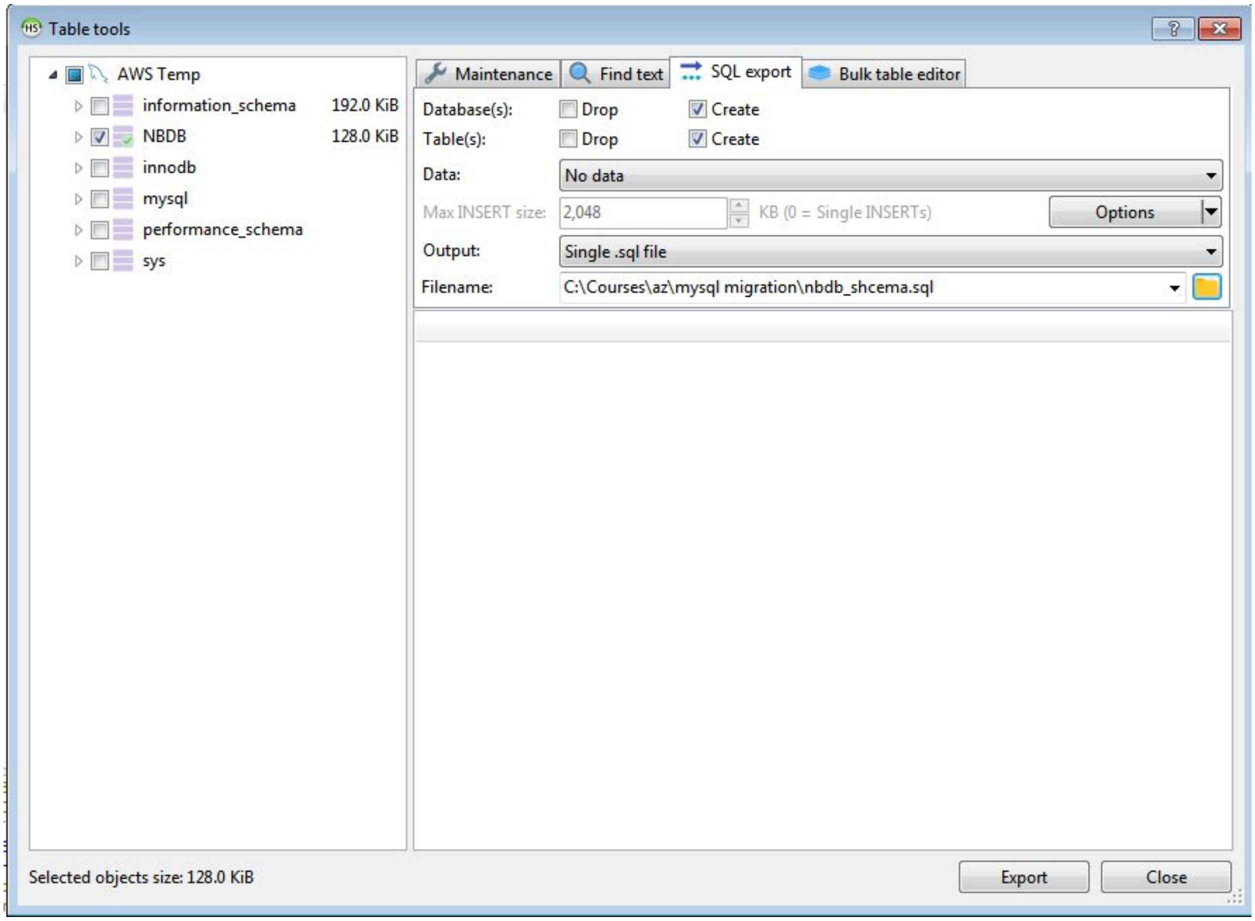


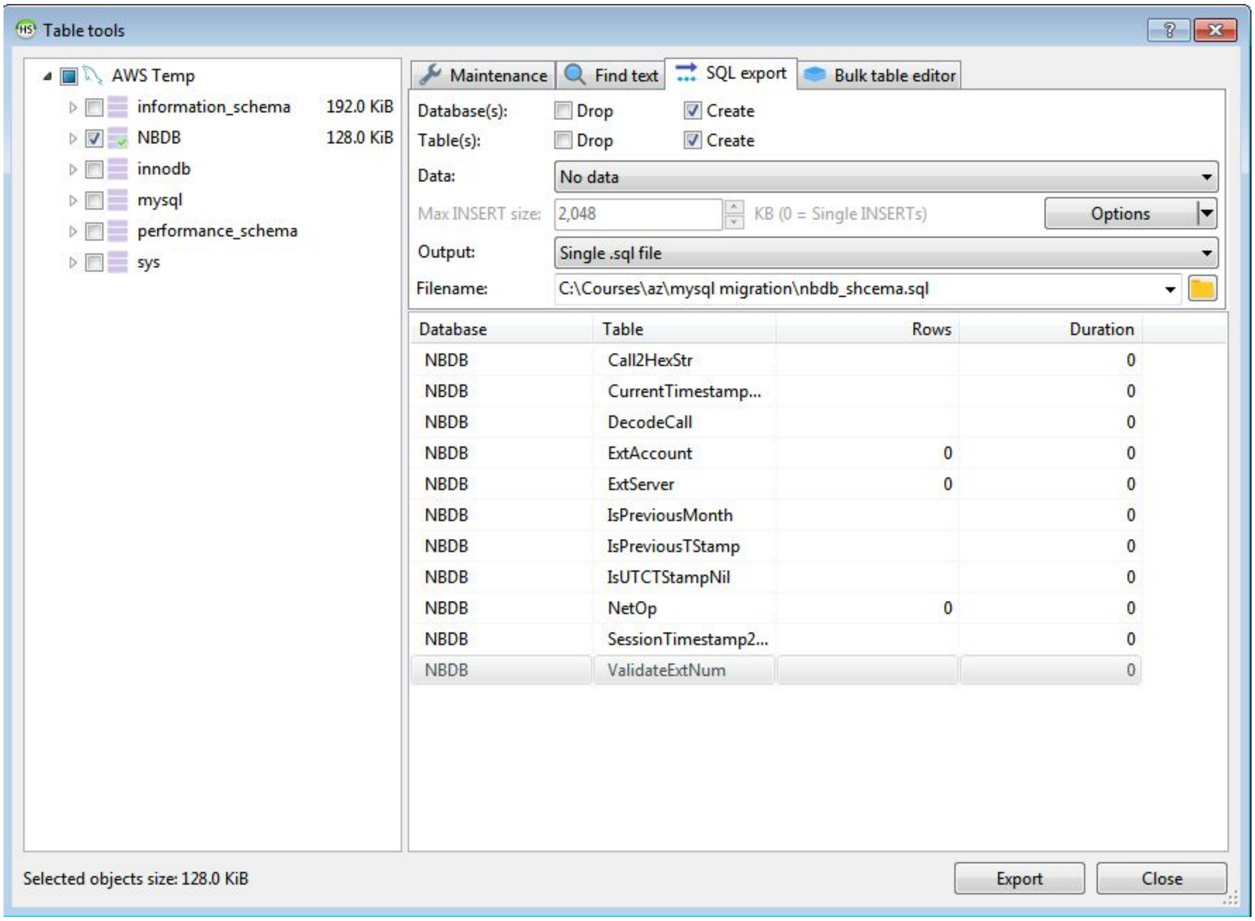
Table tools

- ■ AWS Temp
  - □ information_schema 192.0 KiB
  - ☑ NBDB 128.0 KiB
  - □ innodb
  - □ mysql
  - □ performance_schema
  - □ sys

Database(s): □ Drop ☑ Create
Table(s): □ Drop ☑ Create
Data: No data
Max INSERT size: 2,048 KB (0 = Single INSERTs)
Output: Single .sql file
Filename: C:\Courses\az\mysql migration\nbdb_shcema.sql

Selected objects size: 128.0 KiB

Click Export



- AWS Temp
  - □ information_schema 192.0 KiB
  - ☑ NBDB 128.0 KiB
  - □ innodb
  - □ mysql
  - □ performance_schema
  - □ sys

Database(s): □ Drop ☑ Create
Table(s): □ Drop ☑ Create
Data: No data
Max INSERT size: 2,048 KB (0 = Single INSERTs)
Output: Single .sql file
Filename: C:\Courses\az\mysql migration\nbdb_shcema.sql

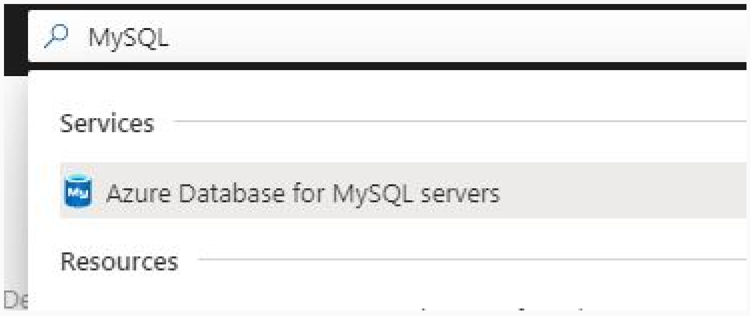
Database	Table	Rows	Duration
NBDB	Call2HexStr		0
NBDB	CurrentTimestamp...		0
NBDB	DecodeCall		0
NBDB	ExtAccount	0	0
NBDB	ExtServer	0	0
NBDB	IsPreviousMonth		0
NBDB	IsPreviousTStamp		0
NBDB	IsUTCTStampNil		0
NBDB	NetOp	0	0
NBDB	SessionTimestamp2...		0
NBDB	ValidateExtNum		0

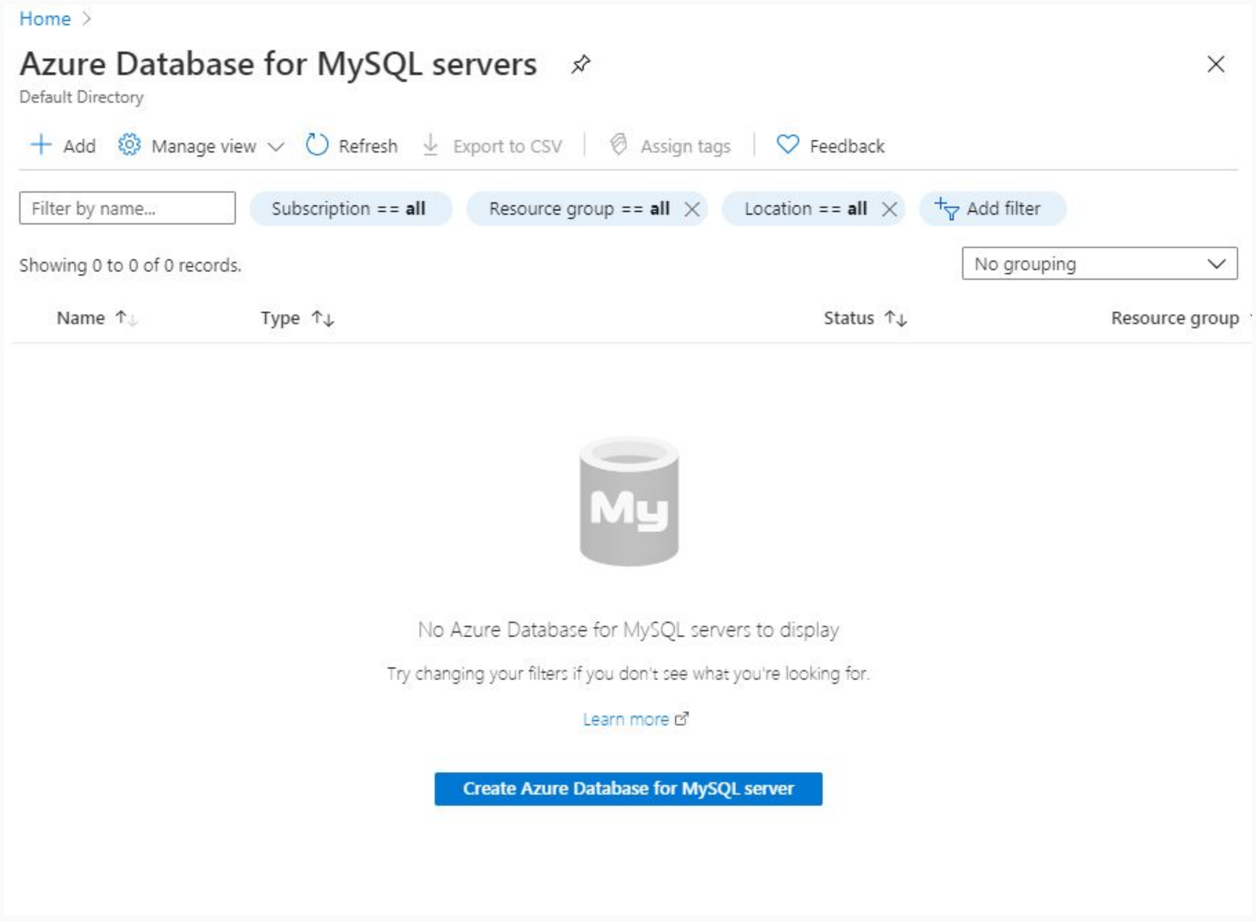
Selected objects size: 128.0 KiB

Verify the contents of the schema file that gets created in the filename path you specified.

- Create an instance of Azure MySQL database.

  In the Azure Portal search for MySQL and select Azure Database for MySQL servers.



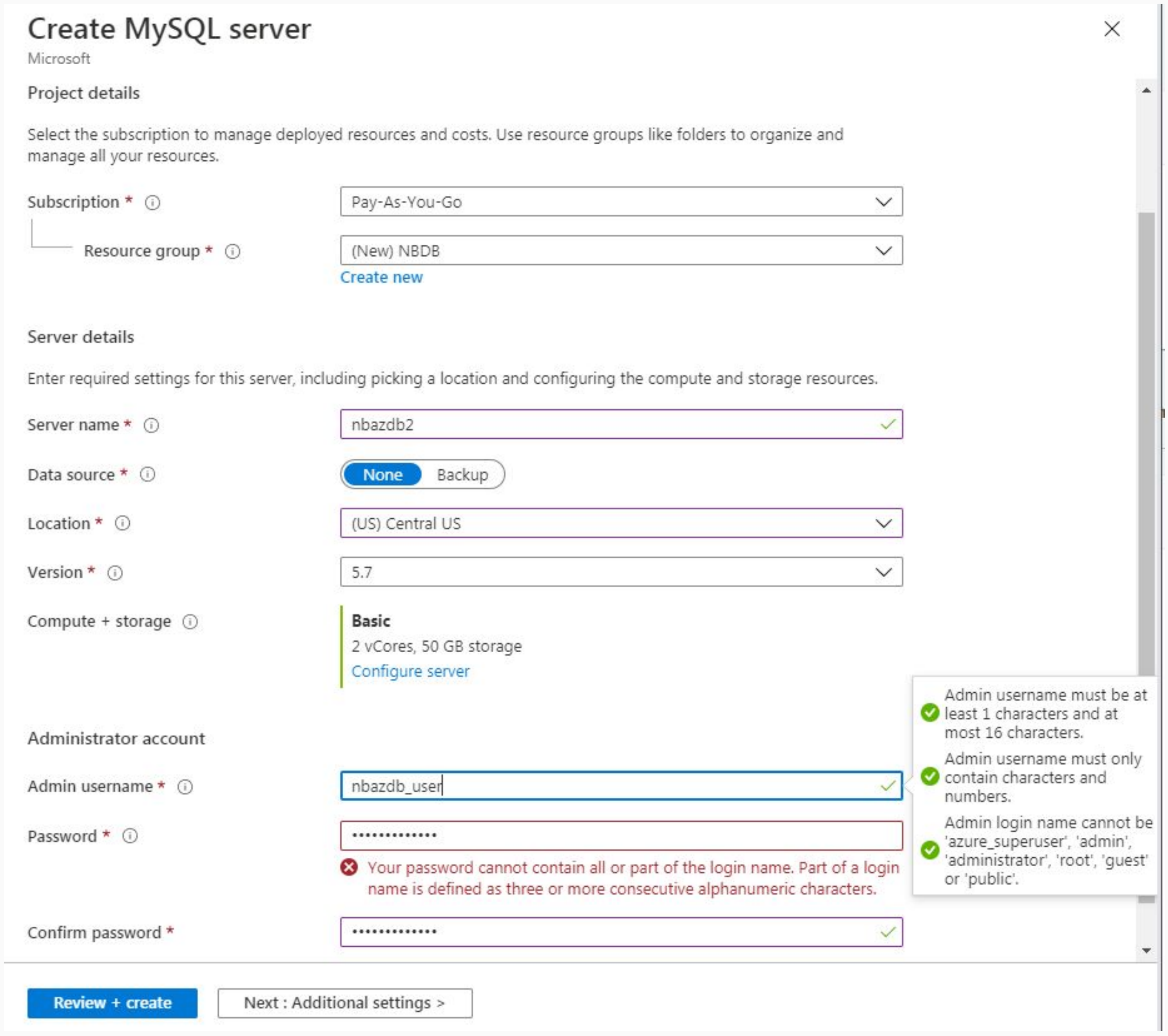


Showing 0 to 0 of 0 records.

No Azure Database for MySQL servers to display

Try changing your filters if you don't see what you're looking for.

We need to create the target database in the premium tier to be able to use the Azure Migration Service. Specify a resource group name, server name, location. Select ‘None’ for the source. Specify an admin username and password. Note the password for using later.



Create MySQL server

Microsoft

Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription * : Pay-As-You-Go

Resource group * : (New) NBDB

Create new

Server details

Enter required settings for this server, including picking a location and configuring the compute and storage resources.

Server name * : nbazdb2

Data source * : None | Backup

Location * : (US) Central US

Version * : 5.7

Compute + storage : **Basic**
2 vCores, 50 GB storage
Configure server

Administrator account

Admin username * : nbazdb_user

Password * : •••••••••••••

Your password cannot contain all or part of the login name. Part of a login name is defined as three or more consecutive alphanumeric characters.

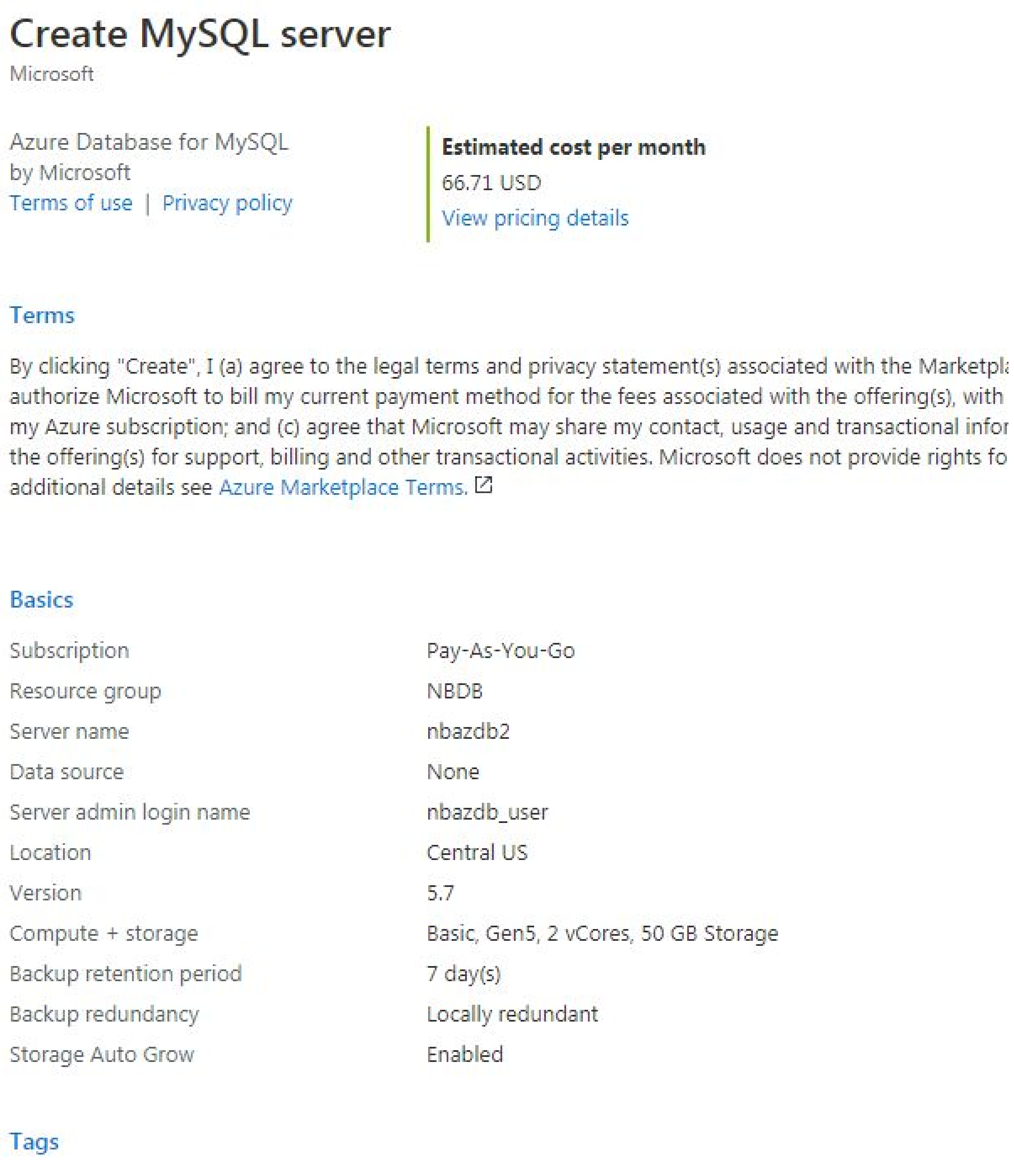
Confirm password * : •••••••••••••

Admin username must be at least 1 characters and at most 16 characters.

Admin username must only contain characters and numbers.

Admin login name cannot be 'azure_superuser', 'admin', 'administrator', 'root', 'guest' or 'public'.

Click on the Review + Create button, Verify the settings and click Create.



Create MySQL server

Microsoft

Azure Database for MySQL
by Microsoft
Terms of use | Privacy policy

Estimated cost per month
66.71 USD
View pricing details

Terms

By clicking "Create", I (a) agree to the legal terms and privacy statement(s) associated with the Marketpl authorize Microsoft to bill my current payment method for the fees associated with the offering(s), with my Azure subscription; and (c) agree that Microsoft may share my contact, usage and transactional infor the offering(s) for support, billing and other transactional activities. Microsoft does not provide rights fo additional details see Azure Marketplace Terms.

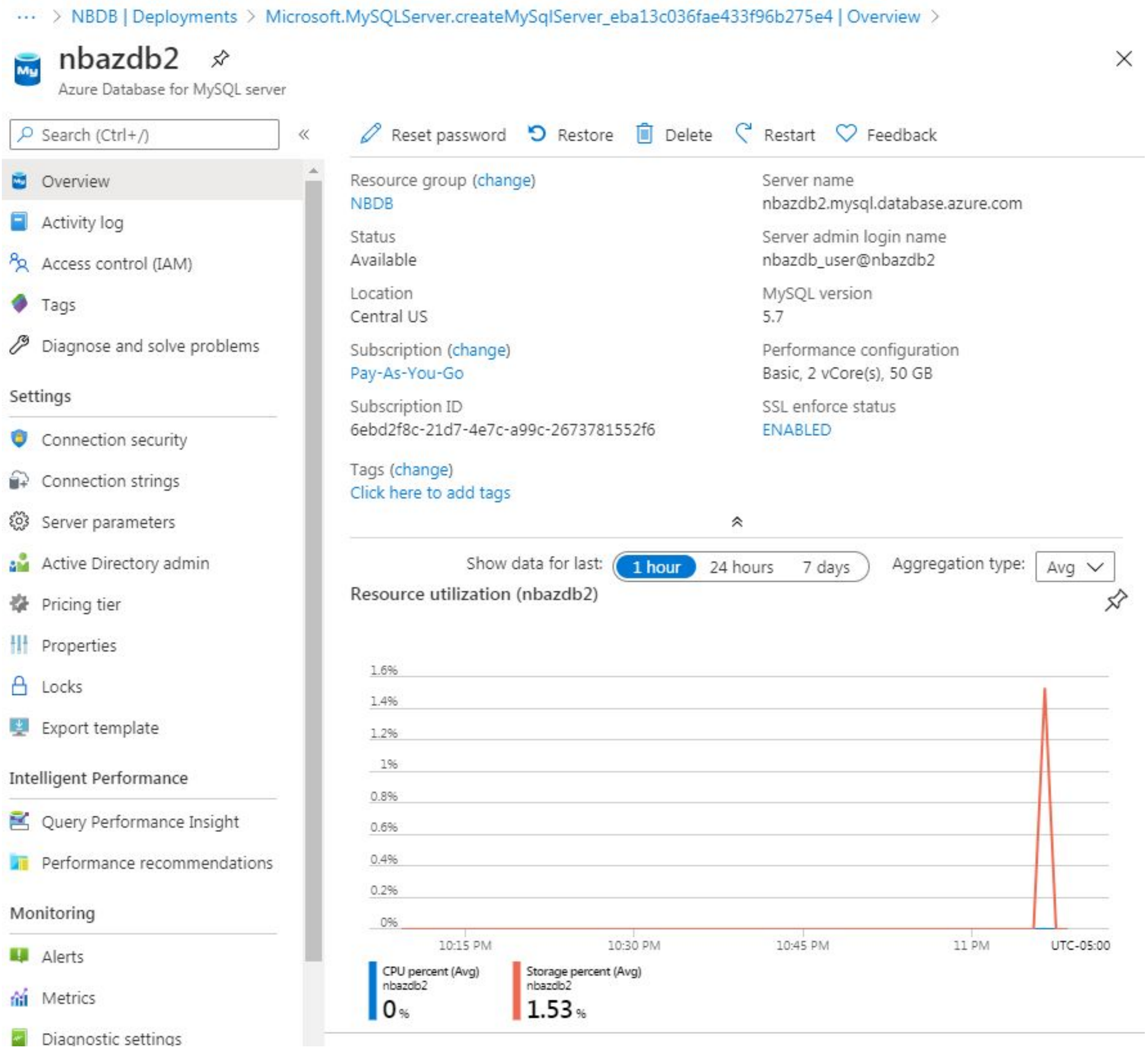
Basics

Subscription	Pay-As-You-Go
Resource group	NBDB
Server name	nbazdb2
Data source	None
Server admin login name	nbazdb_user
Location	Central US
Version	5.7
Compute + storage	Basic, Gen5, 2 vCores, 50 GB Storage
Backup retention period	7 day(s)
Backup redundancy	Locally redundant
Storage Auto Grow	Enabled

Tags

The Basic version of the database took less than 3 minutes to deploy.

The server name will be suffixed with “.mysql.database.azure.com”



··· > NBDB | Deployments > Microsoft.MySQLServer.createMySqlServer_eba13c036fae433f96b275e4 | Overview >

nbazdb2

Azure Database for MySQL server

Resource group (change)
NBDB

Status
Available

Location
Central US

Subscription (change)
Pay-As-You-Go

Subscription ID
6ebd2f8c-21d7-4e7c-a99c-2673781552f6

Tags (change)
Click here to add tags

Server name
nbazdb2.mysql.database.azure.com

Server admin login name
nbazdb_user@nbazdb2

MySQL version
5.7

Performance configuration
Basic, 2 vCore(s), 50 GB

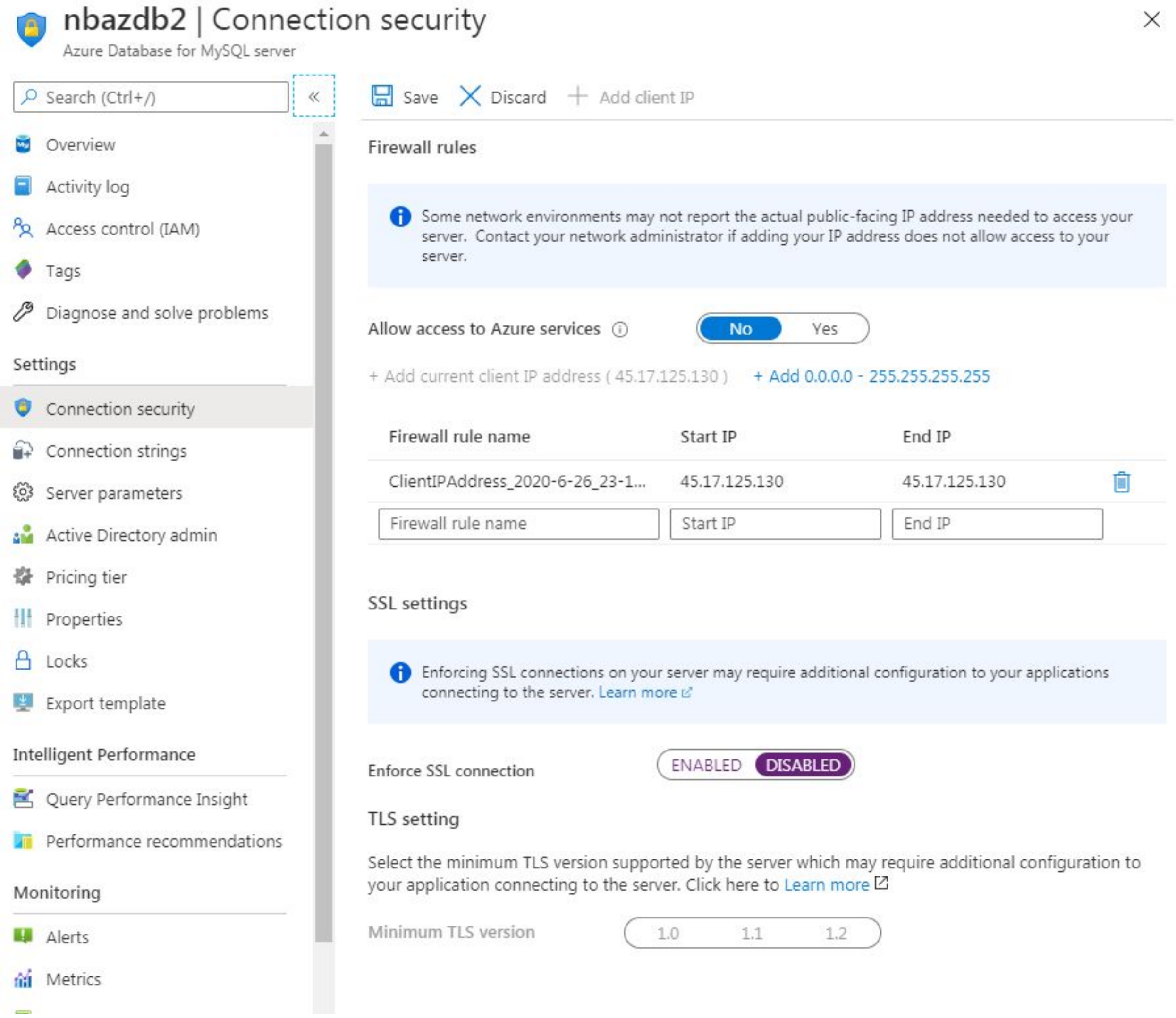
SSL enforce status
ENABLED

Show data for last: 1 hour 24 hours 7 days

Aggregation type: Avg

Resource utilization (nbazdb2)

Under Settings, go to Connection security and Choose “Add current client IP address” which will allow you to connect to the MySQL instance. To avoid configuring the client certificates for this exercise, select Disabled for Enforce SSL connection (not recommended for production)



nbazdb2 | Connection security

Azure Database for MySQL server

Firewall rules

Some network environments may not report the actual public-facing IP address needed to access your server. Contact your network administrator if adding your IP address does not allow access to your server.

Allow access to Azure services: [✓ No] Yes

+ Add current client IP address (45.17.125.130)   + Add 0.0.0.0 - 255.255.255.255

Firewall rule name	Start IP	End IP
ClientIPAddress_2020-6-26_23-1...	45.17.125.130	45.17.125.130
Firewall rule name	Start IP	End IP

SSL settings

Enforcing SSL connections on your server may require additional configuration to your applications connecting to the server. Learn more

Enforce SSL connection: ENABLED [✓ DISABLED]

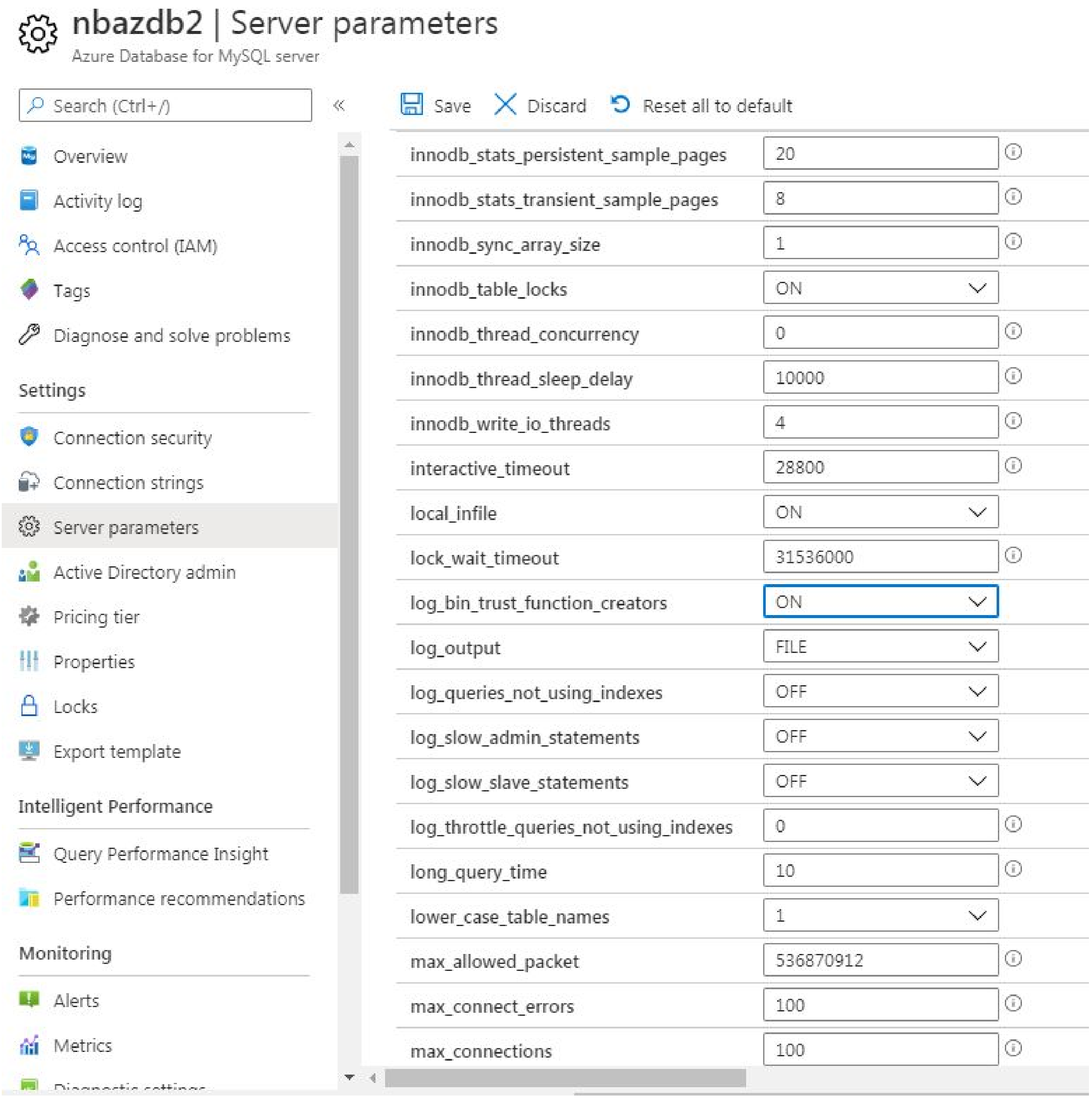
TLS setting

Select the minimum TLS version supported by the server which may require additional configuration to your application connecting to the server. Click here to Learn more

Minimum TLS version: 1.0 1.1 1.2

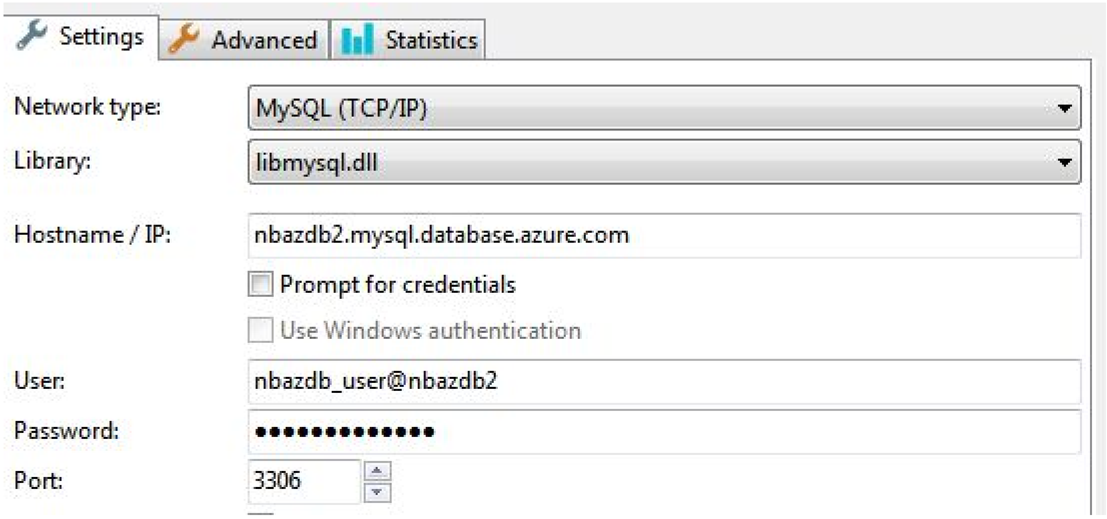
Click Save to update the Connection security settings.

If your source database has functions, you'll need to enable the log_bin_trust_function_creators parameter. Navigate to Server parameters, look for log_bin_trust_function_creators and set it to ON. Save the parameters.



innodb_stats_persistent_sample_pages	20
innodb_stats_transient_sample_pages	8
innodb_sync_array_size	1
innodb_table_locks	ON
innodb_thread_concurrency	0
innodb_thread_sleep_delay	10000
innodb_write_io_threads	4
interactive_timeout	28800
local_infile	ON
lock_wait_timeout	31536000
log_bin_trust_function_creators	ON
log_output	FILE
log_queries_not_using_indexes	OFF
log_slow_admin_statements	OFF
log_slow_slave_statements	OFF
log_throttle_queries_not_using_indexes	0
long_query_time	10
lower_case_table_names	1
max_allowed_packet	536870912
max_connect_errors	100
max_connections	100

Using any MySQL client interface, connect to the database. Use the Server name, server admin login from the database overview page and the password you specified at creating time to configure the connection. We've used Heidi SQL



Network type: MySQL (TCP/IP)

Library: libmysql.dll

Hostname / IP: nbazdb2.mysql.database.azure.com

□ Prompt for credentials

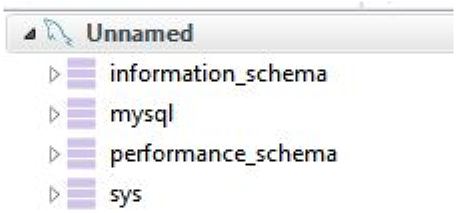
□ Use Windows authentication

User: nbazdb_user@nbazdb2

Password: ●●●●●●●●●●●●●

Port: 3306

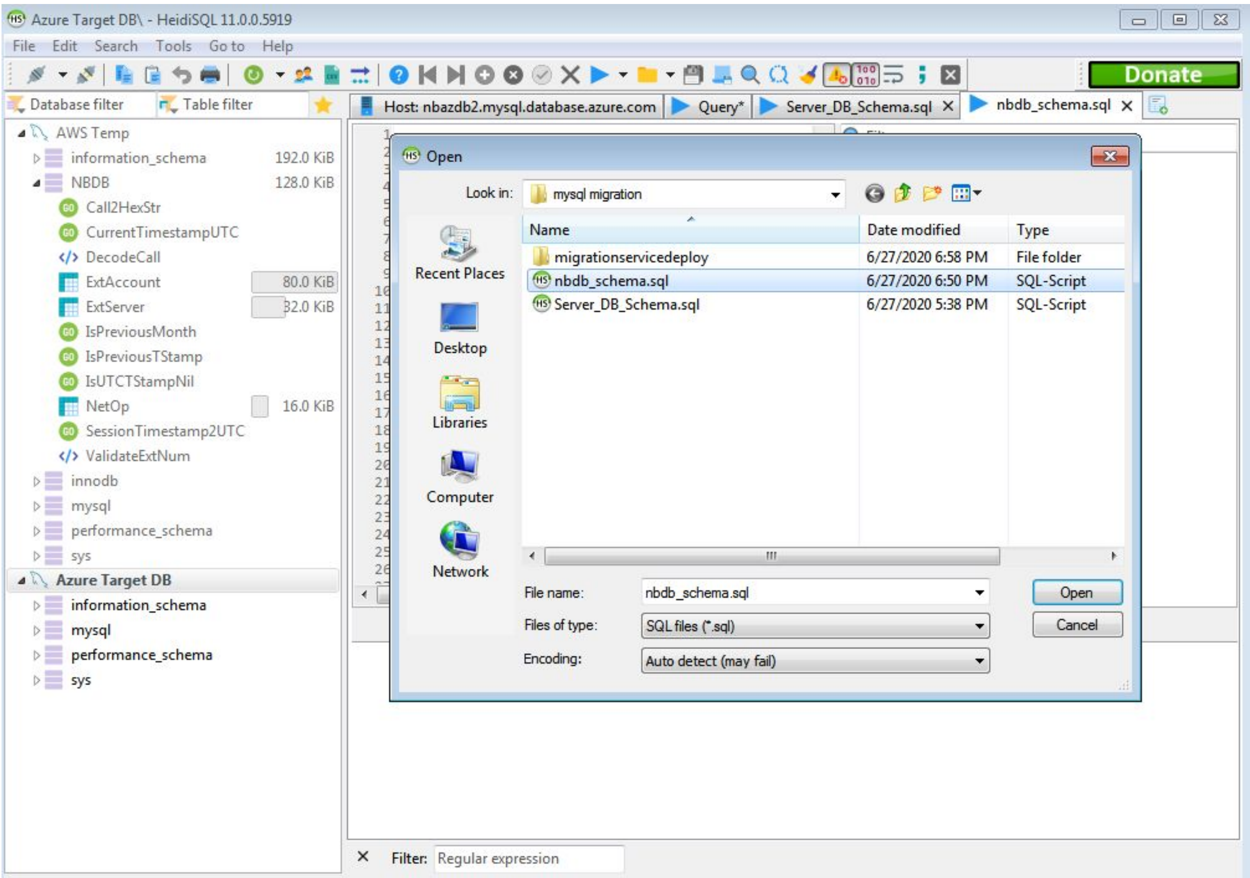
Once connected, you see the following default databases exist: information_schema, mysql, performance_schema, and sys.



Unnamed
- information_schema
- mysql
- performance_schema
- sys

- Import the source schema that you saved perviously to the target Azure Database for MySQL

In HeidiSQL Navigate to File - Run SQL file - Select the schema file:



Look in: mysql migration

Name	Date modified	Type
migrationservicedeploy	6/27/2020 6:58 PM	File folder
nbdb_schema.sql	6/27/2020 6:50 PM	SQL-Script
Server_DB_Schema.sql	6/27/2020 5:38 PM	SQL-Script

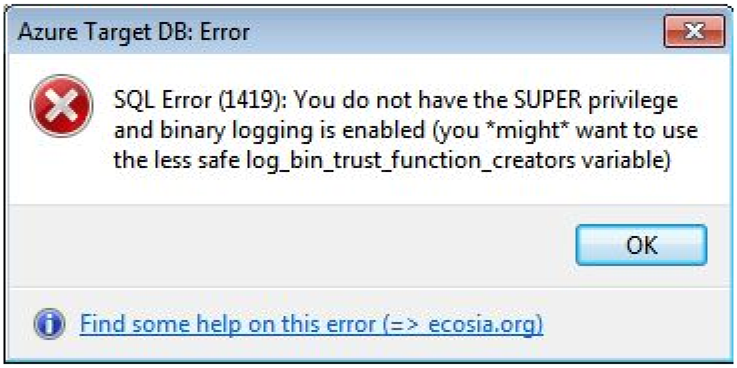
File name: nbdb_schema.sql

Files of type: SQL files (*.sql)

Encoding: Auto detect (may fail)

- Verify that the database and tables have been created.

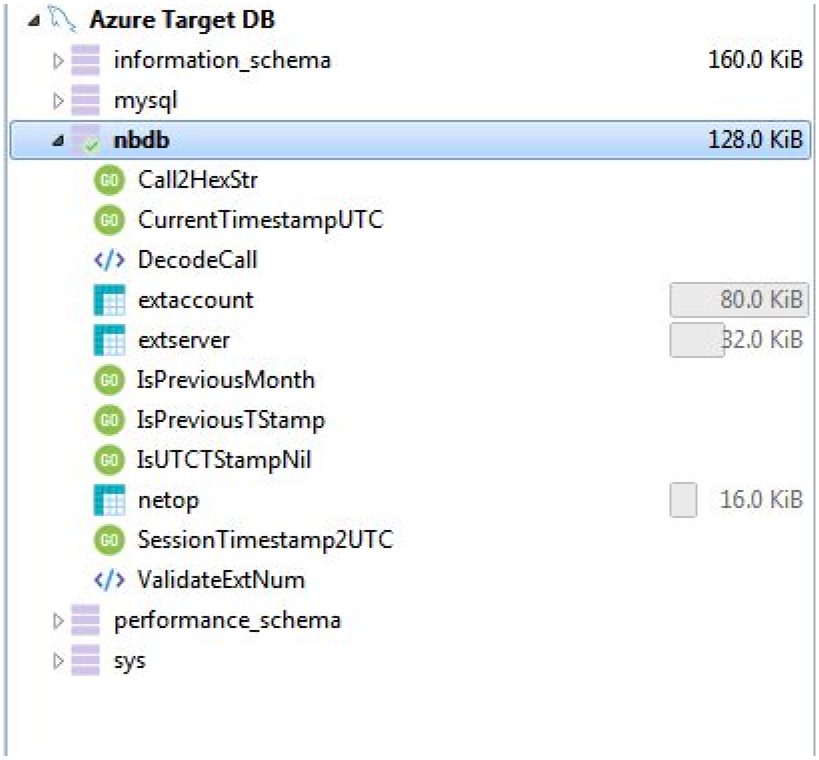
If you see this error, then you'll need to enable the log_bin_trust_function_creators server parameter in the target Azure MySQL database.



Azure Target DB: Error

SQL Error (1419): You do not have the SUPER privilege and binary logging is enabled (you *might* want to use the less safe log_bin_trust_function_creators variable)

Find some help on this error (=> ecosia.org)



- Azure Target DB
  - information_schema 160.0 KiB
  - mysql
  - nbdb 128.0 KiB
    - Call2HexStr
    - CurrentTimestampUTC
    - DecodeCall
    - extaccount 80.0 KiB
    - extserver 32.0 KiB
    - IsPreviousMonth
    - IsPreviousTStamp
    - IsUTCTStampNil
    - netop 16.0 KiB
    - SessionTimestamp2UTC
    - ValidateExtNum
  - performance_schema
  - sys

The nbdb database, tables and functions have been created in the Azure Database.

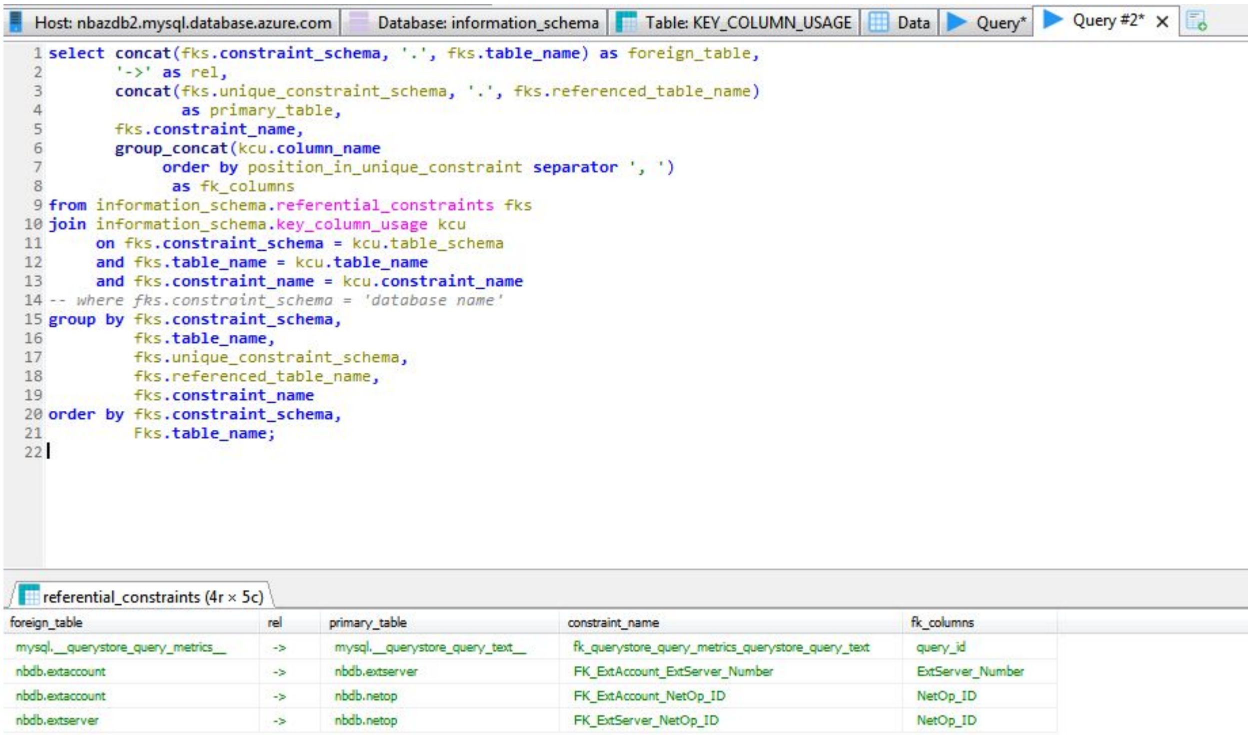
- Since we have foreign keys in the schema, the initial load and continuous sync of the migration would fail unless we drop the foreign keys. Extract the drop foreign key script and add foreign key script at the destination (Azure Database for MySQL), run the following script: (Note: The script provided in the Microsoft documentation didn’t produce any results)

```
select concat(fks.constraint_schema, '.', fks.table_name) as foreign_table,

       '->' as rel,

       concat(fks.unique_constraint_schema, '.', fks.referenced_table_name)

              as primary_table,
       fks.constraint_name,
       group_concat(kcu.column_name
            order by position_in_unique_constraint separator ', ')
             as fk_columns
from information_schema.referential_constraints fks
join information_schema.key_column_usage kcu
     on fks.constraint_schema = kcu.table_schema
     and fks.table_name = kcu.table_name
     and fks.constraint_name = kcu.constraint_name
-- where fks.constraint_schema = 'database name'
group by fks.constraint_schema,
         fks.table_name,
         fks.unique_constraint_schema,
         fks.referenced_table_name,
         fks.constraint_name
order by fks.constraint_schema,
         Fks.table_name;
```


Host: nbazdb2.mysql.database.azure.com
Database: information_schema
Table: KEY_COLUMN_USAGE
Data
Query*
Query #2*

```sql
select concat(fks.constraint_schema, '.', fks.table_name) as foreign_table,
       '->' as rel,
       concat(fks.unique_constraint_schema, '.', fks.referenced_table_name)
              as primary_table,
       fks.constraint_name,
       group_concat(kcu.column_name
            order by position_in_unique_constraint separator ', ')
             as fk_columns
from information_schema.referential_constraints fks
join information_schema.key_column_usage kcu
     on fks.constraint_schema = kcu.table_schema
     and fks.table_name = kcu.table_name
     and fks.constraint_name = kcu.constraint_name
-- where fks.constraint_schema = 'database name'
group by fks.constraint_schema,
         fks.table_name,
         fks.unique_constraint_schema,
         fks.referenced_table_name,
         fks.constraint_name
order by fks.constraint_schema,
         Fks.table_name;
```

referential_constraints (4r × 5c)

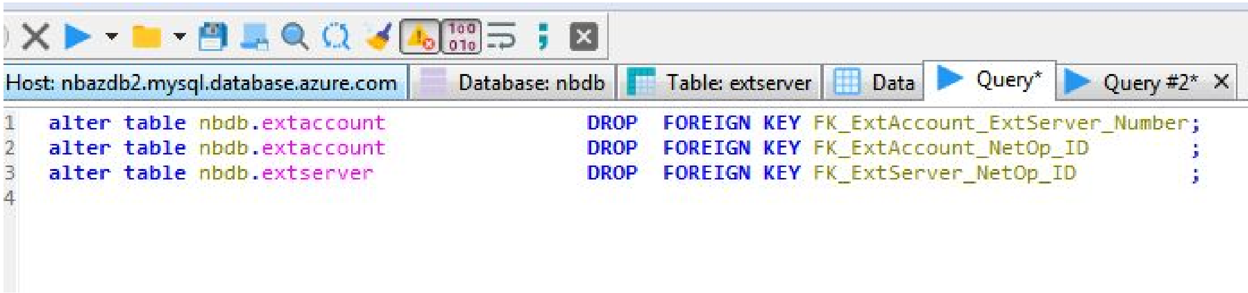
foreign_table	rel	primary_table	constraint_name	fk_columns
mysql.__querystore_query_metrics__	->	mysql.__querystore_query_text__	fk_querystore_query_metrics_querystore_query_text	query_id
nbdb.extaccount	->	nbdb.extserver	FK_ExtAccount_ExtServer_Number	ExtServer_Number
nbdb.extaccount	->	nbdb.netop	FK_ExtAccount_NetOp_ID	NetOp_ID
nbdb.extserver	->	nbdb.netop	FK_ExtServer_NetOp_ID	NetOp_ID

Copy the foreign key rows produced and build the drop table queries:

```
 alter table nbdb.extaccount                DROP  FOREIGN KEY
FK_ExtAccount_ExtServer_Number;

  alter table nbdb.extaccount                DROP  FOREIGN KEY
FK_ExtAccount_NetOp_ID       ;

  alter table nbdb.extserver                 DROP  FOREIGN KEY
FK_ExtServer_NetOp_ID        ;
```



```
alter table nbdb.extaccount                DROP  FOREIGN KEY FK_ExtAccount_ExtServer_Number;
alter table nbdb.extaccount                DROP  FOREIGN KEY FK_ExtAccount_NetOp_ID        ;
alter table nbdb.extserver                 DROP  FOREIGN KEY FK_ExtServer_NetOp_ID         ;
```

This database did not have any tiggers, but if you do, you'll have to disable the triggers with this query so that they don't go off during the migration:

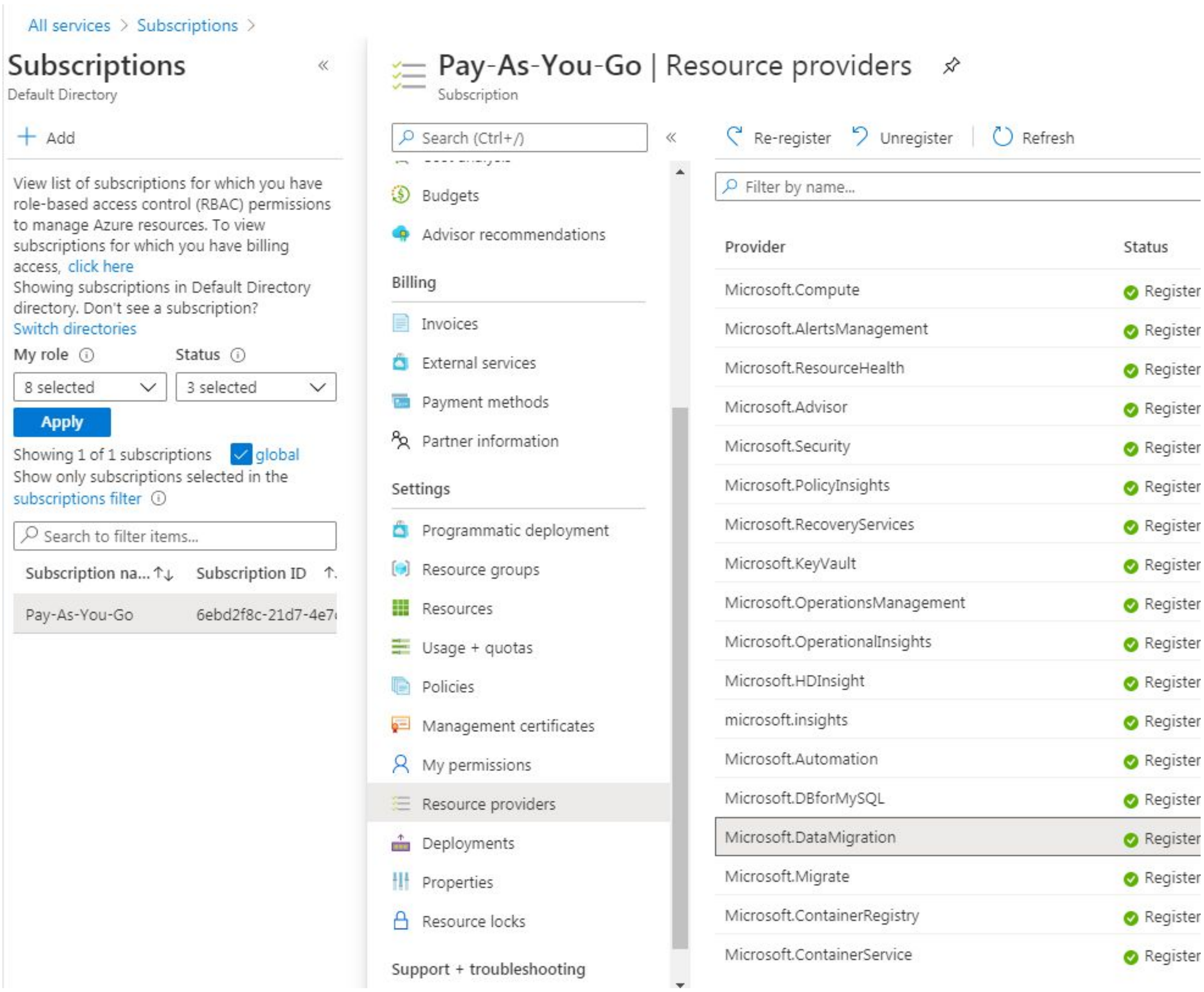
```
select concat ('alter table ', event_object_table, ' disable
trigger ', trigger_name)from information_schema.triggers;
```

Also if there are instances of the ENUM data type in any tables, temporarily update them to the ‘character varying’ datatype in the target table. When data replication is complete, then revert the data type to ENUM.

- Register the Microsoft.DataMigration resource provider

  Azure Portal - All Services - Subscriptions - Select your subscription - Settings - Resource Providers

  Find Microsoft.DataMigration - Select Register



All services > Subscriptions >

Subscriptions

Default Directory

+ Add

View list of subscriptions for which you have role-based access control (RBAC) permissions to manage Azure resources. To view subscriptions for which you have billing access, click here

Showing subscriptions in Default Directory directory. Don't see a subscription? Switch directories

My role ⓘ: 8 selected

Status ⓘ: 3 selected

Apply

Showing 1 of 1 subscriptions ☑ global

Show only subscriptions selected in the subscriptions filter ⓘ

Search to filter items...

Subscription na... ↑↓	Subscription ID ↑.
Pay-As-You-Go	6ebd2f8c-21d7-4e7

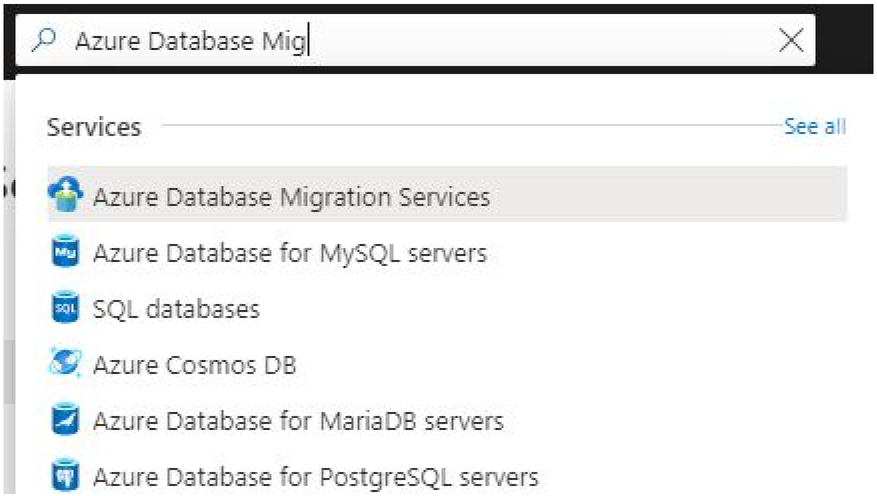
Pay-As-You-Go | Resource providers

Subscription

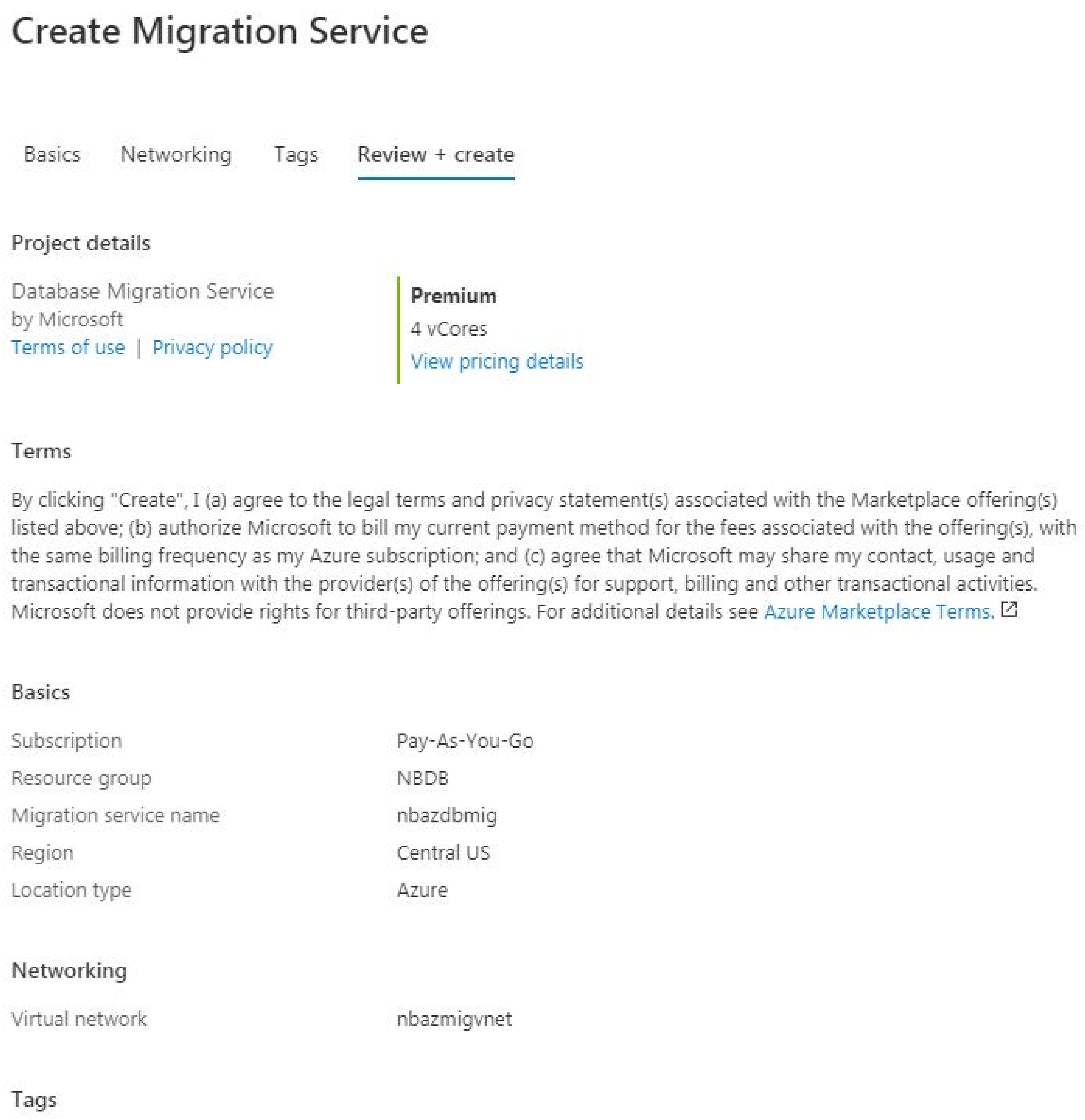
Provider	Status
Microsoft.Compute	✓ Register
Microsoft.AlertsManagement	✓ Register
Microsoft.ResourceHealth	✓ Register
Microsoft.Advisor	✓ Register
Microsoft.Security	✓ Register
Microsoft.PolicyInsights	✓ Register
Microsoft.RecoveryServices	✓ Register
Microsoft.KeyVault	✓ Register
Microsoft.OperationsManagement	✓ Register
Microsoft.OperationalInsights	✓ Register
Microsoft.HDInsight	✓ Register
microsoft.insights	✓ Register
Microsoft.Automation	✓ Register
Microsoft.DBforMySQL	✓ Register
Microsoft.DataMigration	✓ Register
Microsoft.Migrate	✓ Register
Microsoft.ContainerRegistry	✓ Register
Microsoft.ContainerService	✓ Register

- Create an instance of Azure Database Migration Service in the Premium Tier.

Premium Tier is required to perform online migrations. The service is free to try for 6 months.


Azure Database Mig
Services
See all
Azure Database Migration Services
Azure Database for MySQL servers
SQL databases
Azure Cosmos DB
Azure Database for MariaDB servers
Azure Database for PostgreSQL servers

Specify Migration Service name, and a VNet where the Azure Database Migration Service will reside. Click Review + Create then Create



Create Migration Service

Project details

Database Migration Service
by Microsoft
Terms of use | Privacy policy

Premium
4 vCores
View pricing details

Terms

By clicking "Create", I (a) agree to the legal terms and privacy statement(s) associated with the Marketplace offering(s) listed above; (b) authorize Microsoft to bill my current payment method for the fees associated with the offering(s), with the same billing frequency as my Azure subscription; and (c) agree that Microsoft may share my contact, usage and transactional information with the provider(s) of the offering(s) for support, billing and other transactional activities. Microsoft does not provide rights for third-party offerings. For additional details see Azure Marketplace Terms.

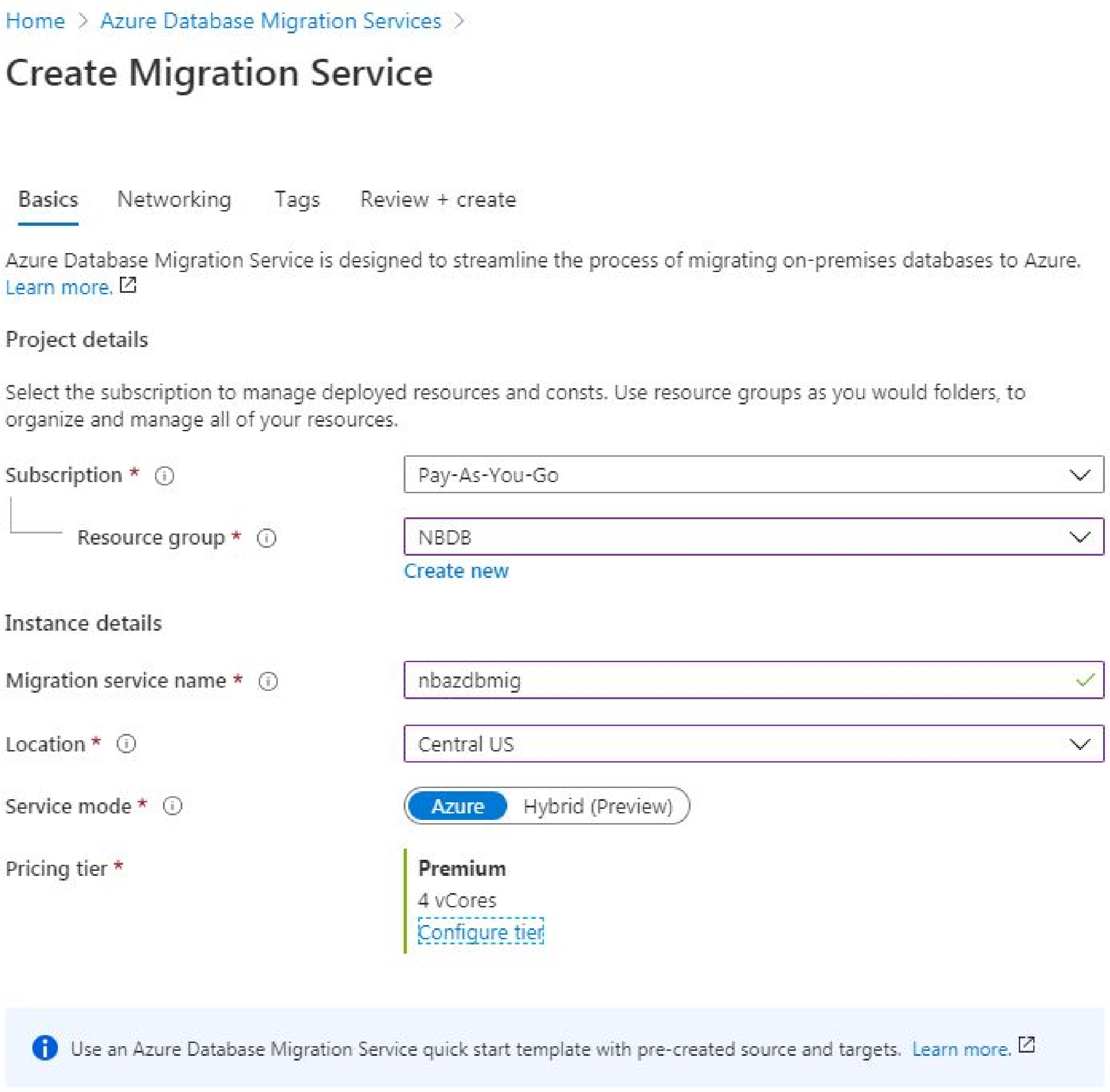
Basics

Subscription	Pay-As-You-Go
Resource group	NBDB
Migration service name	nbazdbmig
Region	Central US
Location type	Azure

Networking

Virtual network	nbazmigvnet

Tags



Home > Azure Database Migration Services >

Create Migration Service

Azure Database Migration Service is designed to streamline the process of migrating on-premises databases to Azure. Learn more.

Project details

Select the subscription to manage deployed resources and consts. Use resource groups as you would folders, to organize and manage all of your resources.

Subscription * : Pay-As-You-Go

Resource group * : NBDB

Create new

Instance details

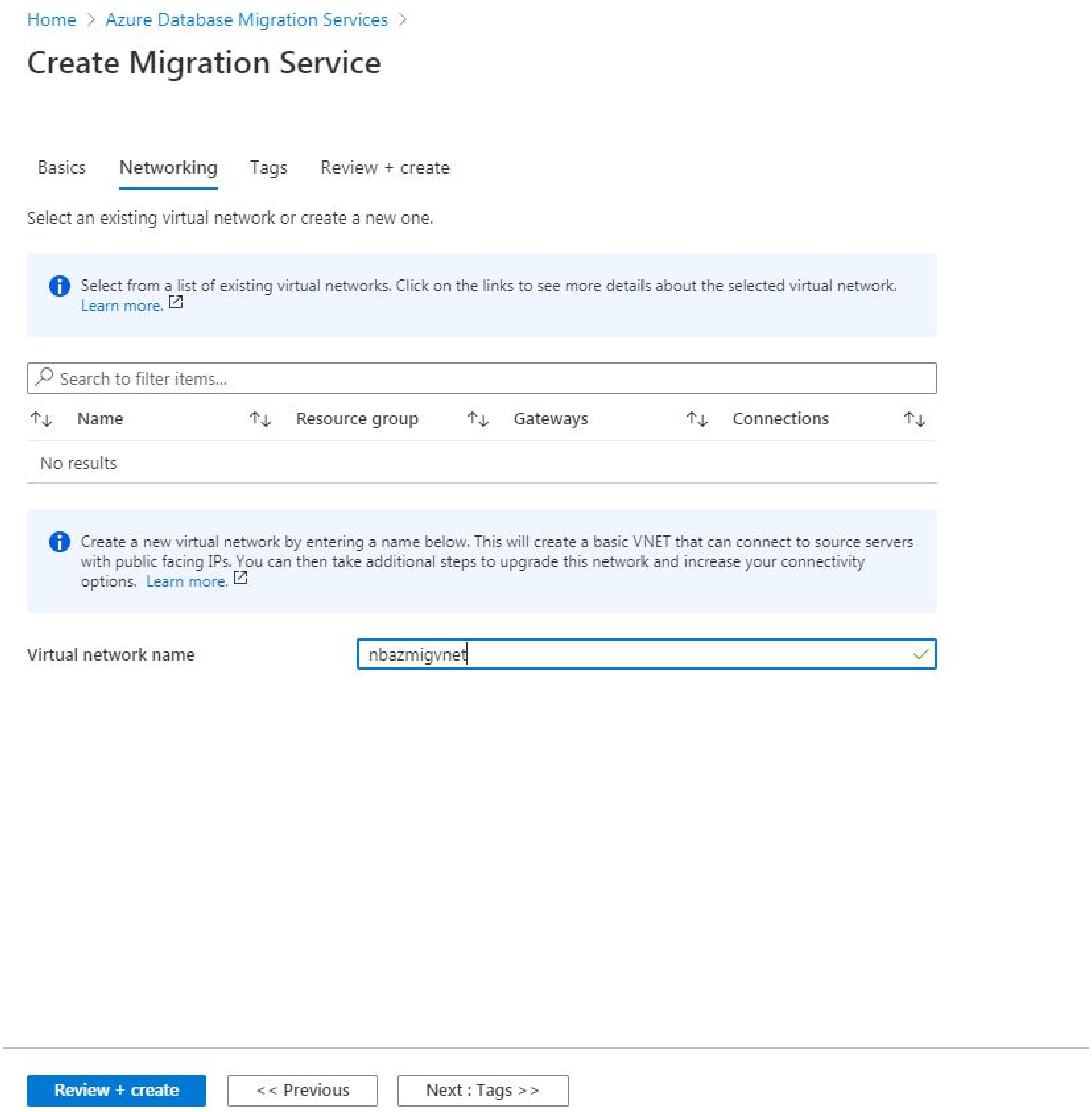
Migration service name * : nbazdbmig

Location * : Central US

Service mode * : [✓ Azure] Hybrid (Preview)

Pricing tier * : **Premium**
4 vCores
Configure tier

Use an Azure Database Migration Service quick start template with pre-created source and targets. Learn more.



Create Migration Service

Select an existing virtual network or create a new one.

Select from a list of existing virtual networks. Click on the links to see more details about the selected virtual network. Learn more.

Search to filter items...

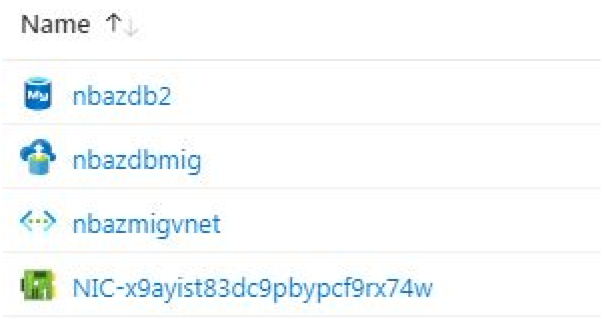
Name	Resource group	Gateways	Connections
No results			

Create a new virtual network by entering a name below. This will create a basic VNET that can connect to source servers with public facing IPs. You can then take additional steps to upgrade this network and increase your connectivity options. Learn more.

Virtual network name: nbazmigvnet

The Database Migration Service also deploys a NIC

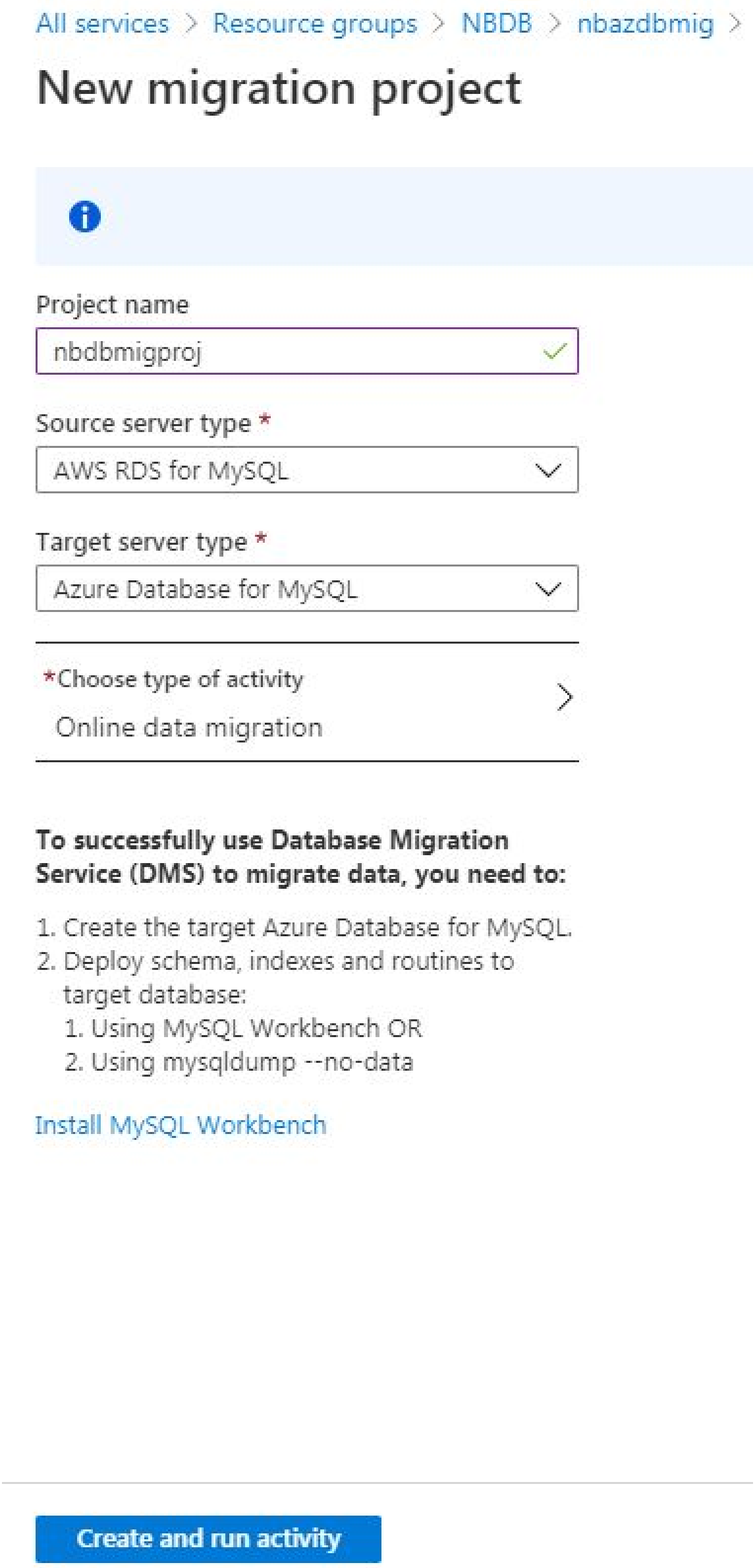
You'll end up with the following resources in the resource group:



Name ↑↓
nbazdb2
nbazdbmig
nbazmigvnet
NIC-x9ayist83dc9pbypcf9rx74w

- Find the IP address for Azure Database Migration Service and allow traffic from this IP address to access your AWS MySQL RDS.

- Create a migration project by using Azure Database Migration Service.



New migration project

Project name

nbdbmigproj ✓

Source server type *

AWS RDS for MySQL

Target server type *

Azure Database for MySQL

*Choose type of activity

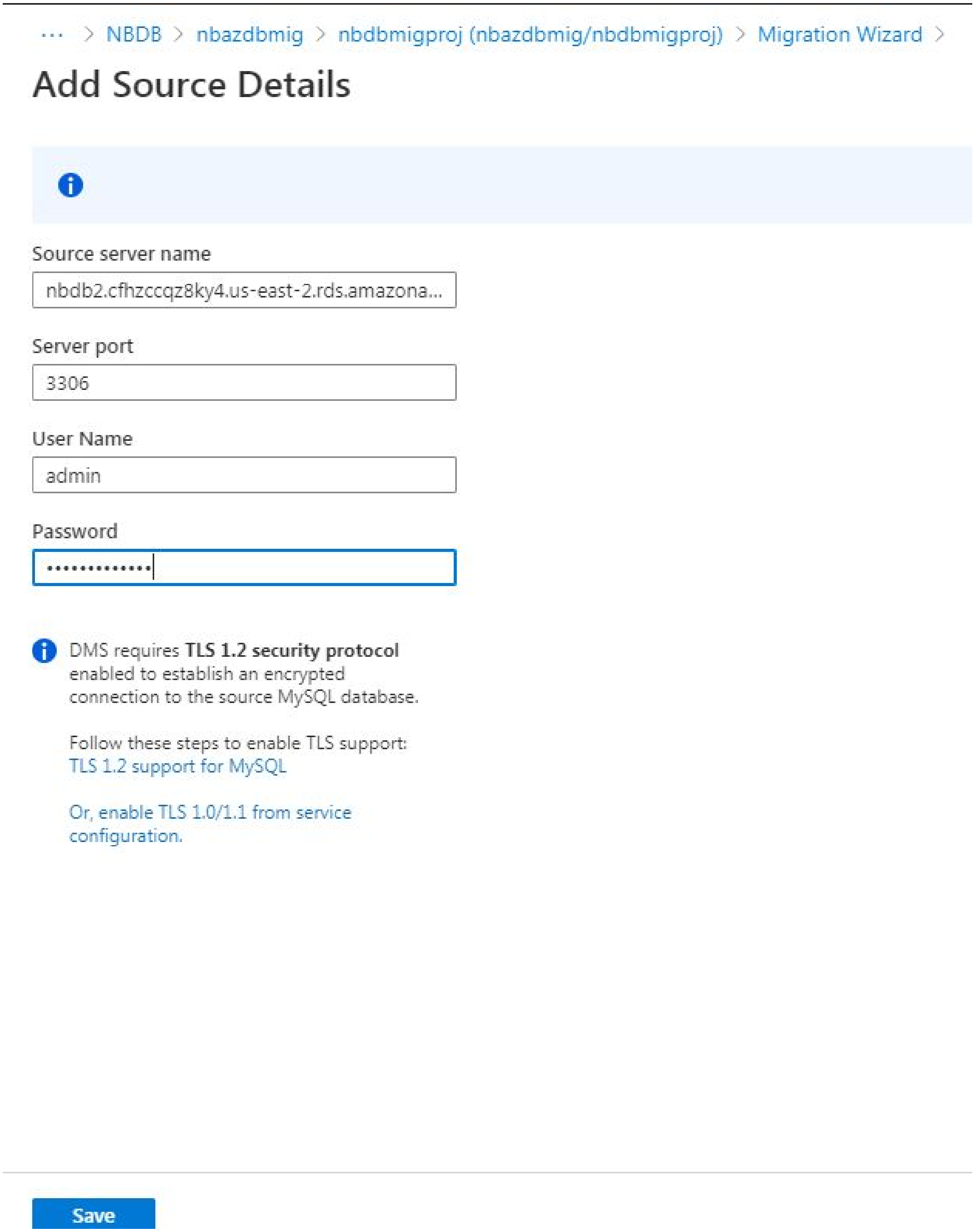
Online data migration

To successfully use Database Migration Service (DMS) to migrate data, you need to:

1. Create the target Azure Database for MySQL.
2. Deploy schema, indexes and routines to target database:
   1. Using MySQL Workbench OR
   2. Using mysqldump --no-data

Install MySQL Workbench

Add source Details


··· > NBDB > nbazdbmig > nbdbmigproj (nbazdbmig/nbdbmigproj) > Migration Wizard >

Add Source Details

Source server name

nbdb2.cfhzccqz8ky4.us-east-2.rds.amazona...

Server port

3306

User Name

admin

Password

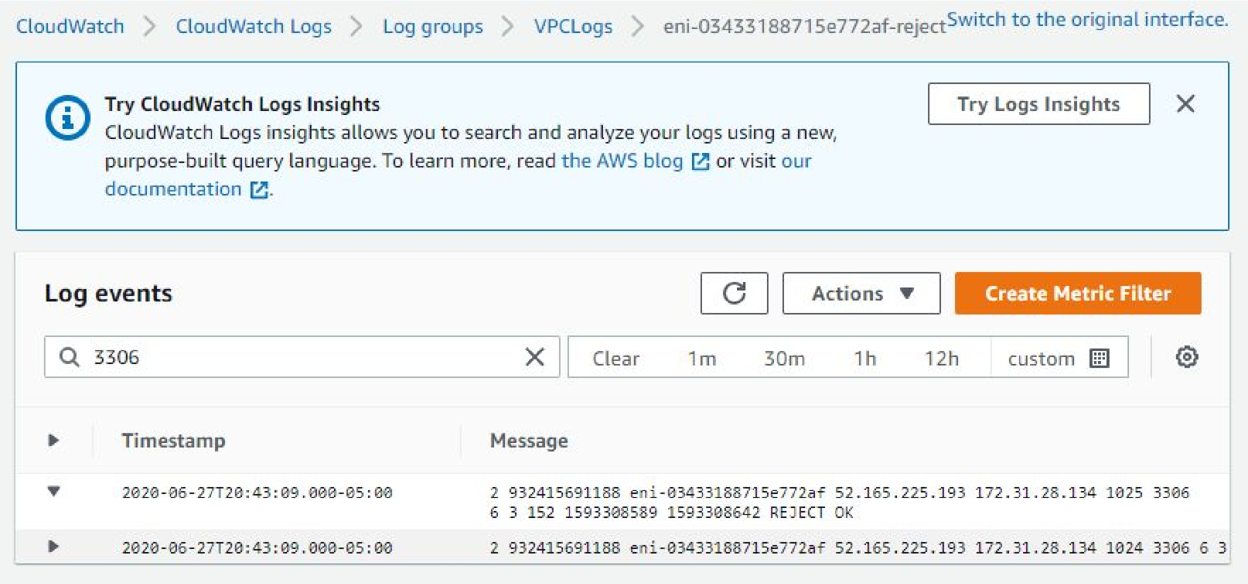
•••••••••••••

DMS requires **TLS 1.2 security protocol** enabled to establish an encrypted connection to the source MySQL database.

Follow these steps to enable TLS support: TLS 1.2 support for MySQL

Or, enable TLS 1.0/1.1 from service configuration.

If the connection fails, you may have to inspect the AWS VPC Flow Logs and look for a connection to port 3306 for the MySQL database. Use the IP address that was rejected and add it to the Security group of the MySQL RDS VPC.

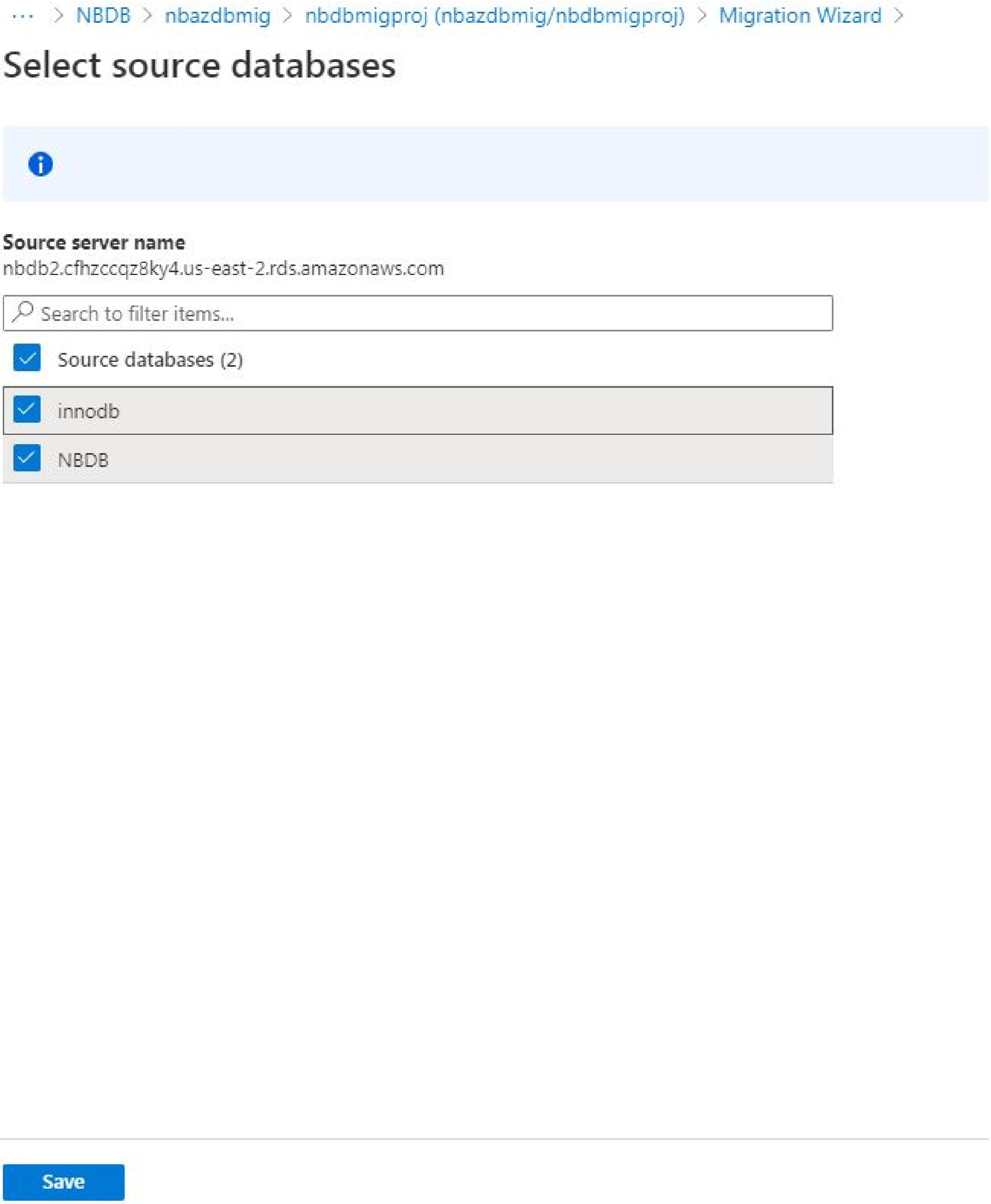

CloudWatch > CloudWatch Logs > Log groups > VPCLogs > eni-03433188715e772af-reject
Switch to the original interface.
Try Logs Insights
Log events
Actions ▼
Create Metric Filter
3306
Clear
1m
30m
1h
12h
custom
Timestamp
Message

Try CloudWatch Logs Insights
CloudWatch Logs insights allows you to search and analyze your logs using a new, purpose-built query language. To learn more, read the AWS blog or visit our documentation.

```
2020-06-27T20:43:09.000-05:00    2 932415691188 eni-03433188715e772af 52.165.225.193 172.31.28.134 1025 3306
                                 6 3 152 1593308589 1593308642 REJECT OK
2020-06-27T20:43:09.000-05:00    2 932415691188 eni-03433188715e772af 52.165.225.193 172.31.28.134 1024 3306 6 3
```

Once the connection to the AWS RDS MySQL source DB succeeds, the Migration Service will display the databases on that instance.

En



··· > NBDB > nbazdbmig > nbdbmigproj (nbazdbmig/nbdbmigproj) > Migration Wizard >

Select source databases

Source server name

nbdb2.cfhzccqz8ky4.us-east-2.rds.amazonaws.com

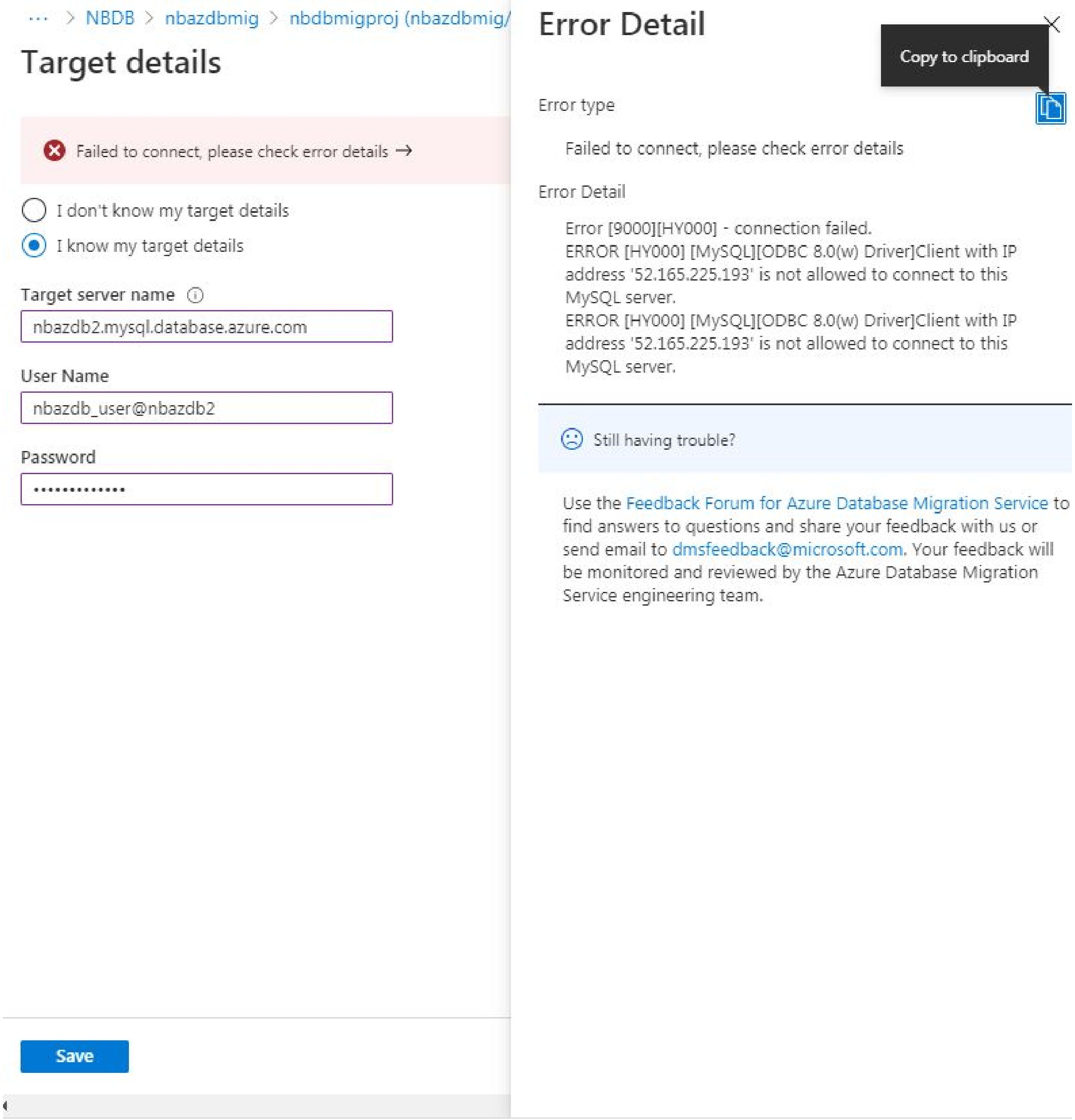
Search to filter items...

☑ Source databases (2)

☑ innodb

☑ NBDB

Enter the Azure MySQL instance information in the Target details.



Target details

Failed to connect, please check error details →

○ I don't know my target details

● I know my target details

Target server name: nbazdb2.mysql.database.azure.com

User Name: nbazdb_user@nbazdb2

Password: •••••••••••••

Error Detail

Error type

Failed to connect, please check error details

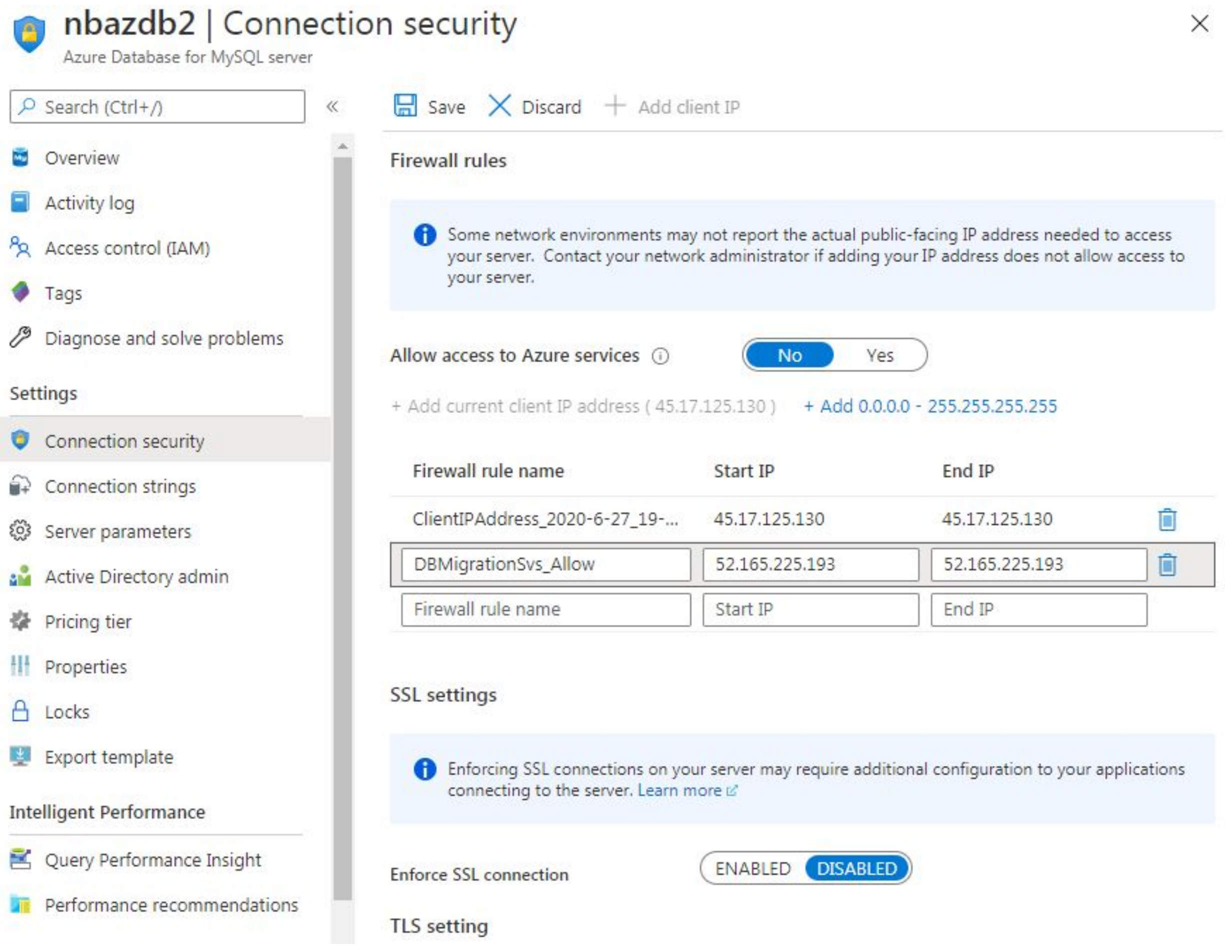
Error Detail

```
Error [9000][HY000] - connection failed.
ERROR [HY000] [MySQL][ODBC 8.0(w) Driver]Client with IP address '52.165.225.193' is not allowed to connect to this MySQL server.
ERROR [HY000] [MySQL][ODBC 8.0(w) Driver]Client with IP address '52.165.225.193' is not allowed to connect to this MySQL server.
```

Still having trouble?

Use the Feedback Forum for Azure Database Migration Service to find answers to questions and share your feedback with us or send email to dmsfeedback@microsoft.com. Your feedback will be monitored and reviewed by the Azure Database Migration Service engineering team.

There will be a connection error when the DB Migration Service tries to connect to it. In a separate window, pull up the Azure Database for MySQL Connection security settings and add the DB Migration Service IP address to the Firewall Rules. Save the changes.



Firewall rules

Some network environments may not report the actual public-facing IP address needed to access your server. Contact your network administrator if adding your IP address does not allow access to your server.

Allow access to Azure services: [✓ No] Yes

+ Add current client IP address (45.17.125.130)   + Add 0.0.0.0 - 255.255.255.255

Firewall rule name	Start IP	End IP
ClientIPAddress_2020-6-27_19-...	45.17.125.130	45.17.125.130
DBMigrationSvs_Allow	52.165.225.193	52.165.225.193
Firewall rule name	Start IP	End IP

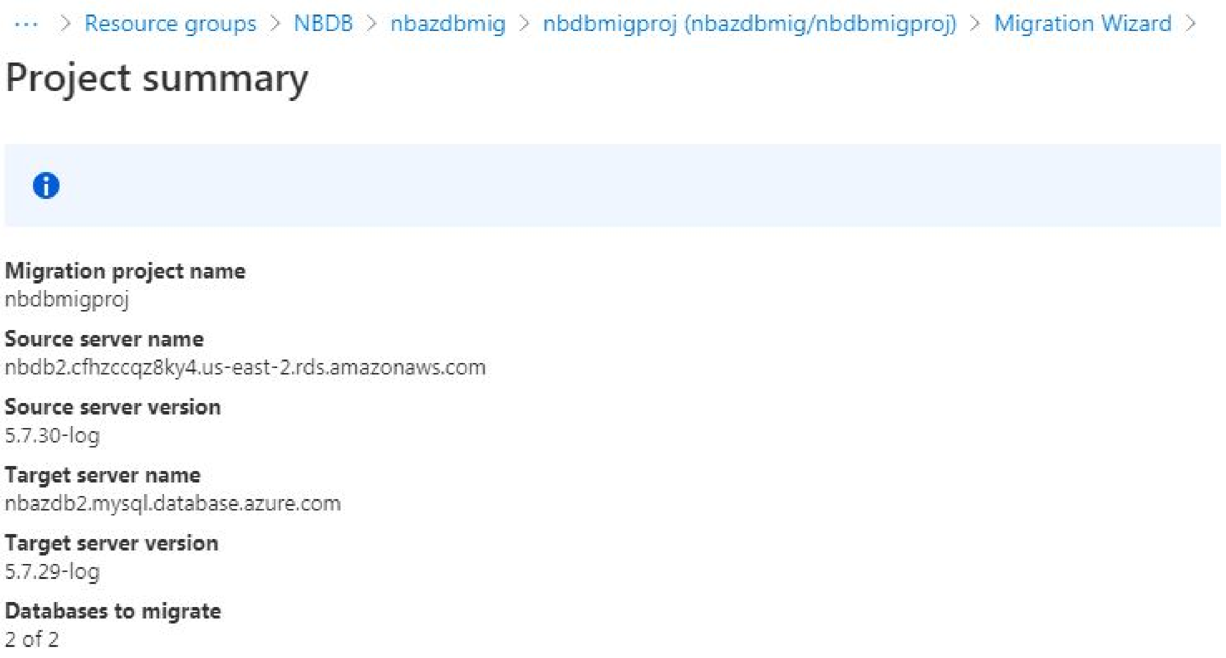
SSL settings

Enforcing SSL connections on your server may require additional configuration to your applications connecting to the server. Learn more

Enforce SSL connection: ENABLED [✓ DISABLED]

TLS setting

Go back to the DB Migration Wizard and attempt to save the Target details again. The connection should succeed and display the Project Summary screen. Click on Save.


··· > Resource groups > NBDB > nbazdbmig > nbdbmigproj (nbazdbmig/nbdbmigproj) > Migration Wizard >

Project summary

Migration project name
nbdbmigproj

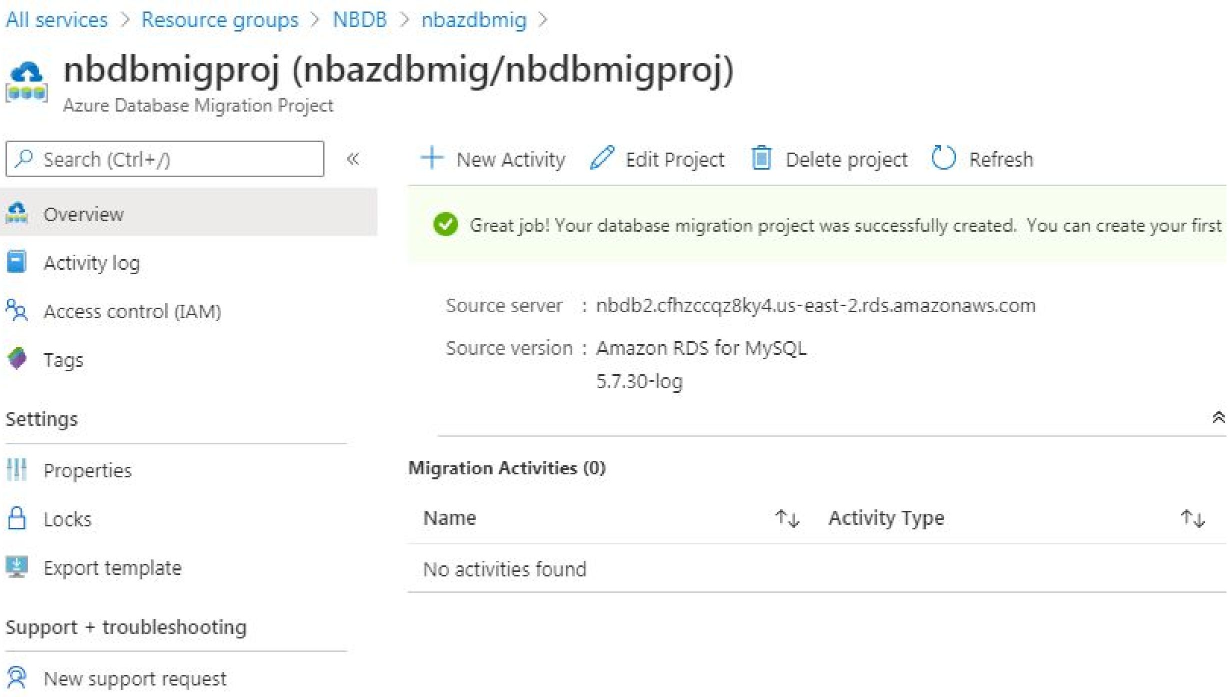
Source server name
nbdb2.cfhzccqz8ky4.us-east-2.rds.amazonaws.com

Source server version
5.7.30-log

Target server name
nbazdb2.mysql.database.azure.com

Target server version
5.7.29-log

Databases to migrate
2 of 2



nbdbmigproj (nbazdbmig/nbdbmigproj)

Azure Database Migration Project

Great job! Your database migration project was successfully created. You can create your first

Source server : nbdb2.cfhzccqz8ky4.us-east-2.rds.amazonaws.com

Source version : Amazon RDS for MySQL
5.7.30-log

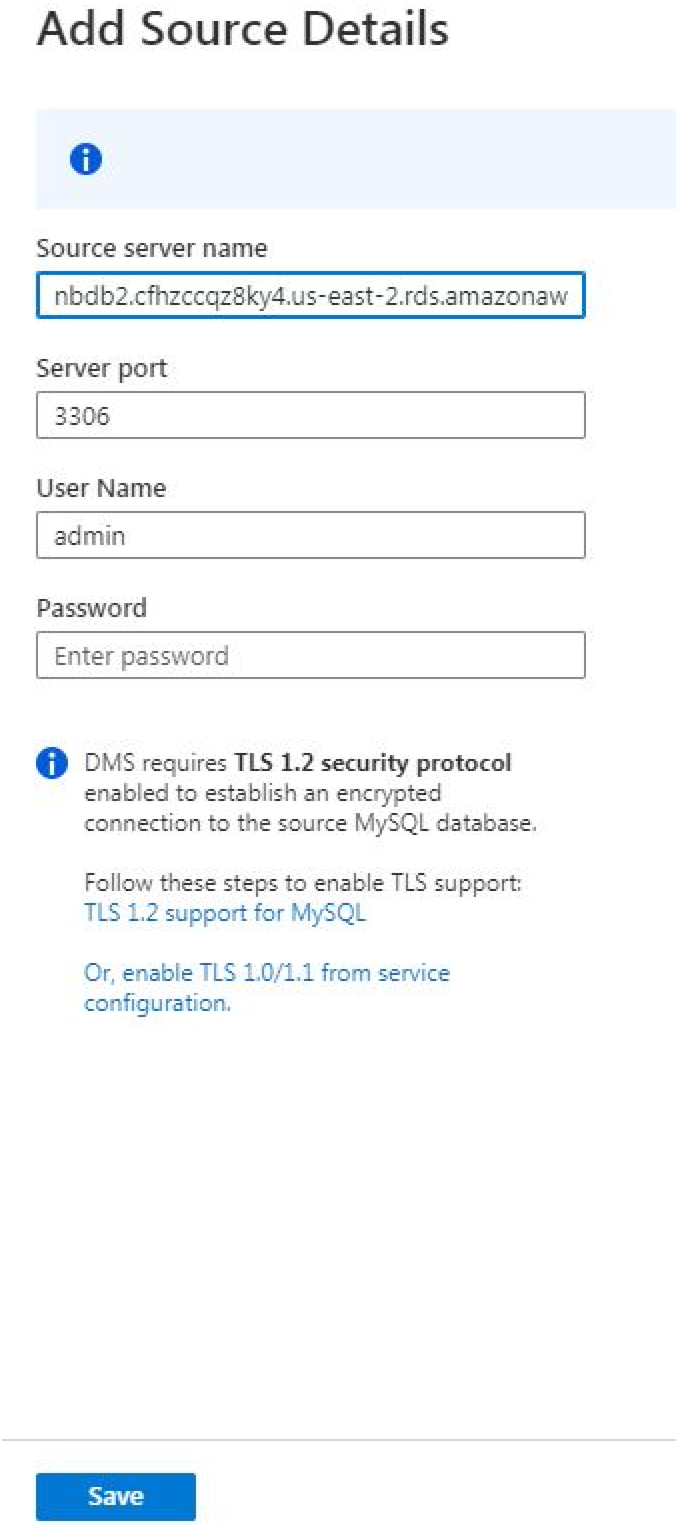
Migration Activities (0)

Name	Activity Type
No activities found	

- Run the migration.

  Select +New Activity, Online Data Migration

Specify the AWS RDS MySQL source password



Add Source Details

Source server name: nbdb2.cfhzccqz8ky4.us-east-2.rds.amazonaw

Server port: 3306

User Name: admin

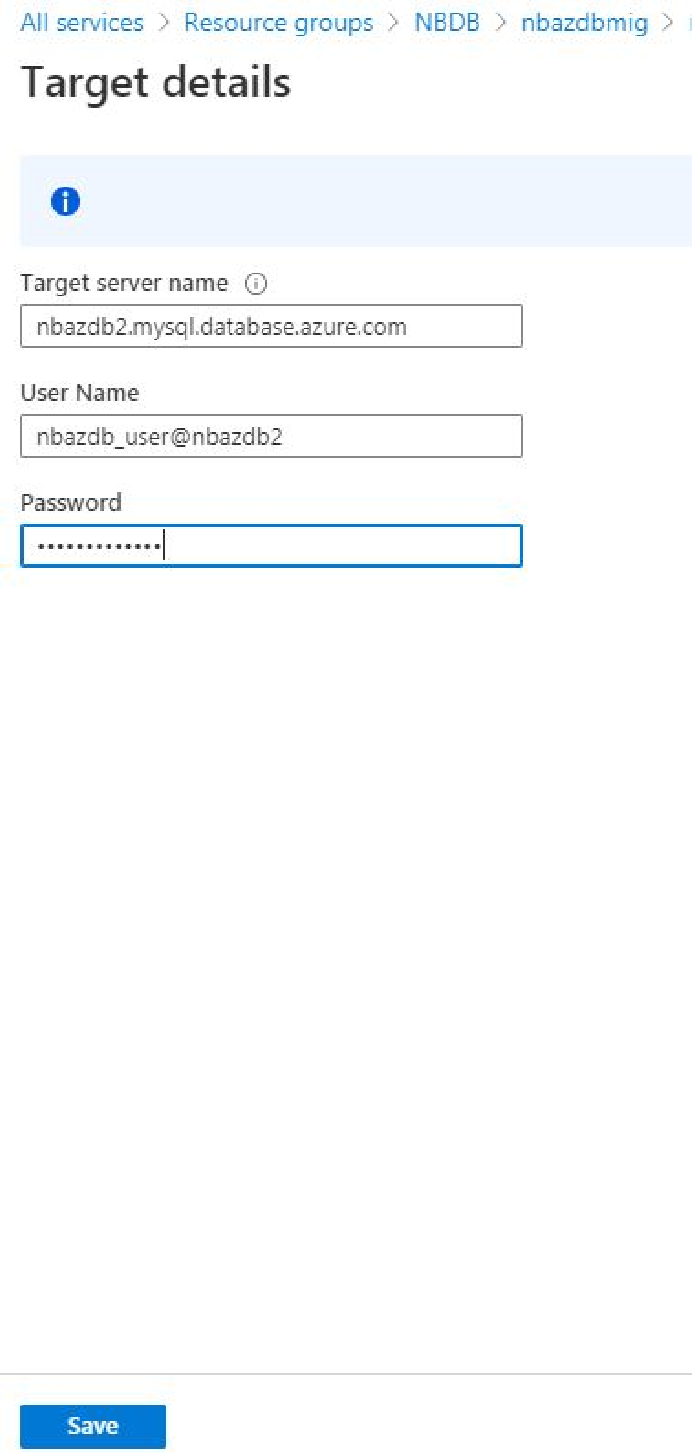
Password: Enter password

DMS requires **TLS 1.2 security protocol** enabled to establish an encrypted connection to the source MySQL database.

Follow these steps to enable TLS support: TLS 1.2 support for MySQL

Or, enable TLS 1.0/1.1 from service configuration.

Specify the Azure DB password



All services > Resource groups > NBDB > nbazdbmig >

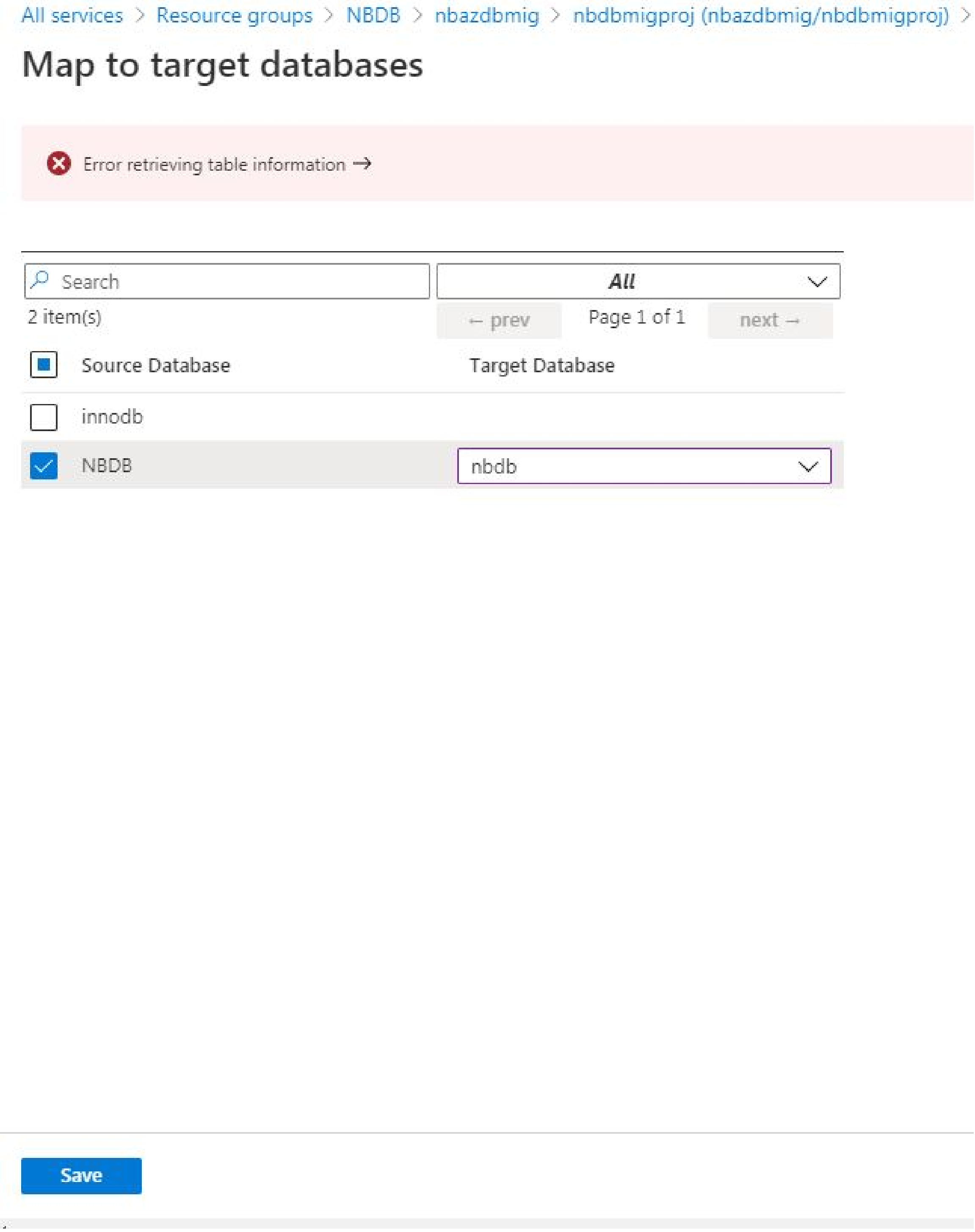
Target details

Target server name: nbazdb2.mysql.database.azure.com

User Name: nbazdb_user@nbazdb2

Password: •••••••••••••

Select the DB mapping. (Note: Do not select the INNODB db)



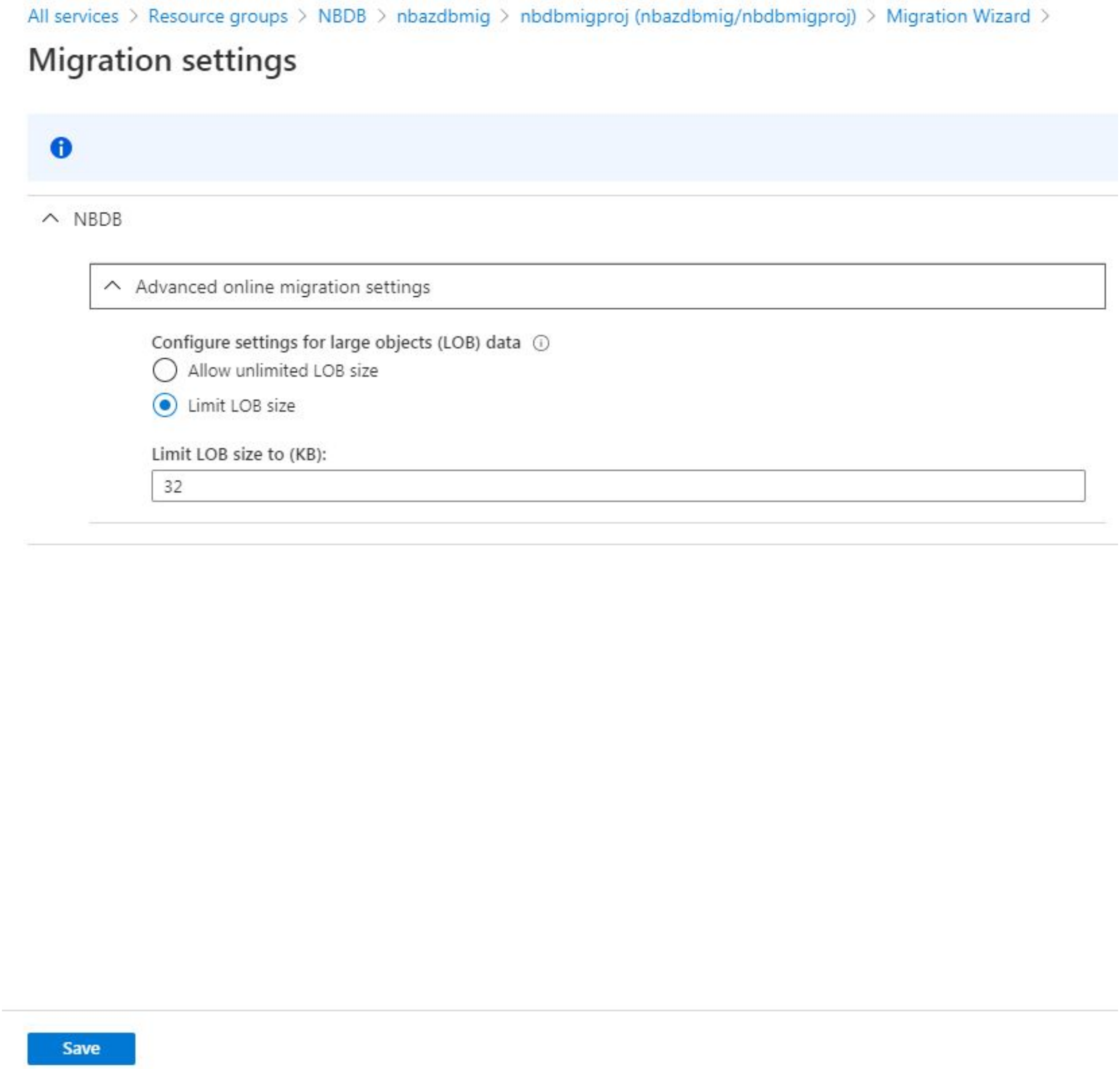
All services > Resource groups > NBDB > nbazdbmig > nbdbmigproj (nbazdbmig/nbdbmigproj) >

Map to target databases

Error retrieving table information →

	Source Database	Target Database
□	innodb	
☑	NBDB	nbdb

Modify the Migration setting for the LOB if necessary.



All services > Resource groups > NBDB > nbazdbmig > nbdbmigproj (nbazdbmig/nbdbmigproj) > Migration Wizard >

Migration settings

NBDB

Advanced online migration settings

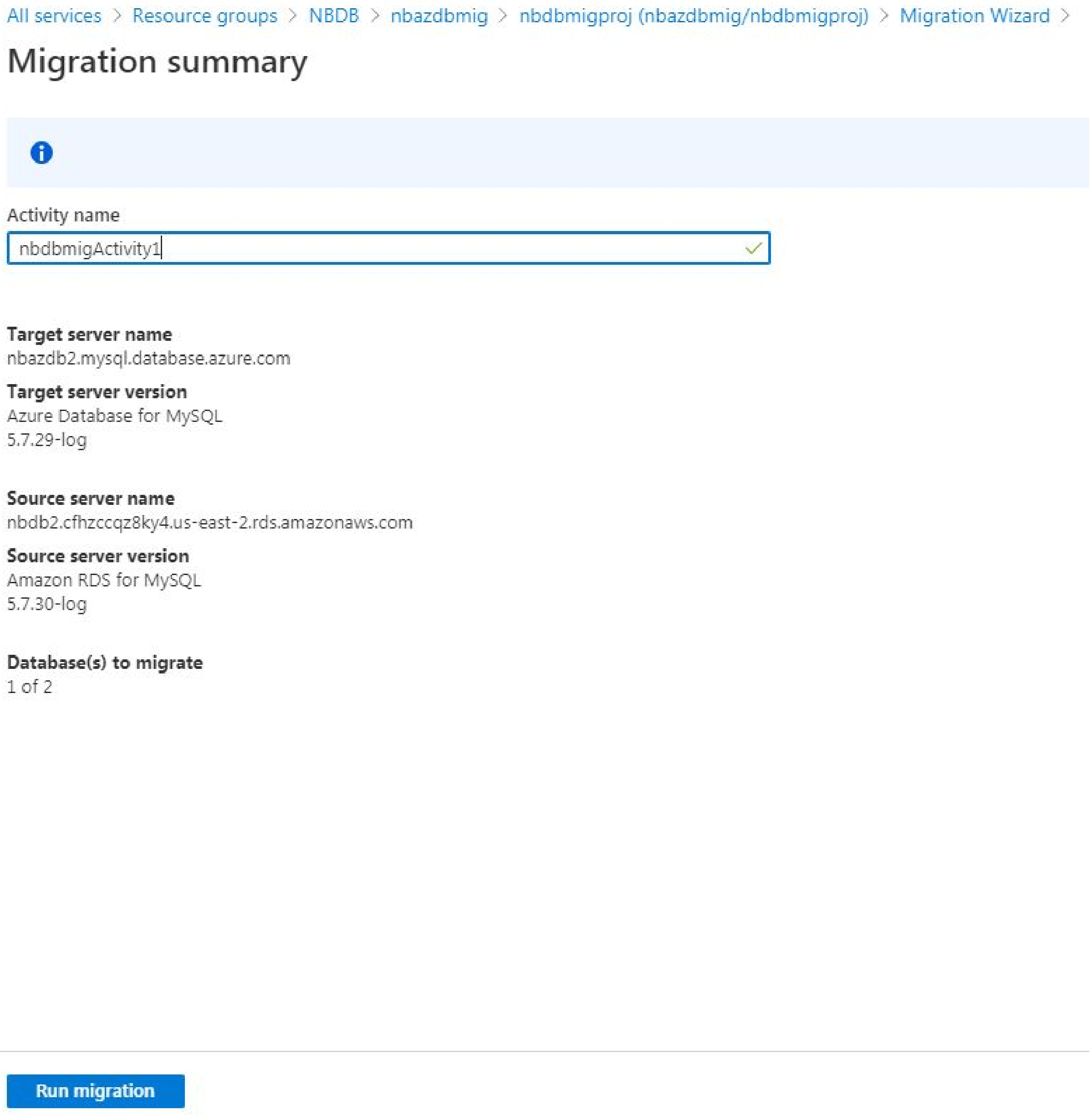
Configure settings for large objects (LOB) data

○ Allow unlimited LOB size

● Limit LOB size

Limit LOB size to (KB): 32

Verify the details and name the migration activity. Then click Run Migration.



All services > Resource groups > NBDB > nbazdbmig > nbdbmigproj (nbazdbmig/nbdbmigproj) > Migration Wizard >

Migration summary

Activity name

nbdbmigActivity1

Target server name
nbazdb2.mysql.database.azure.com

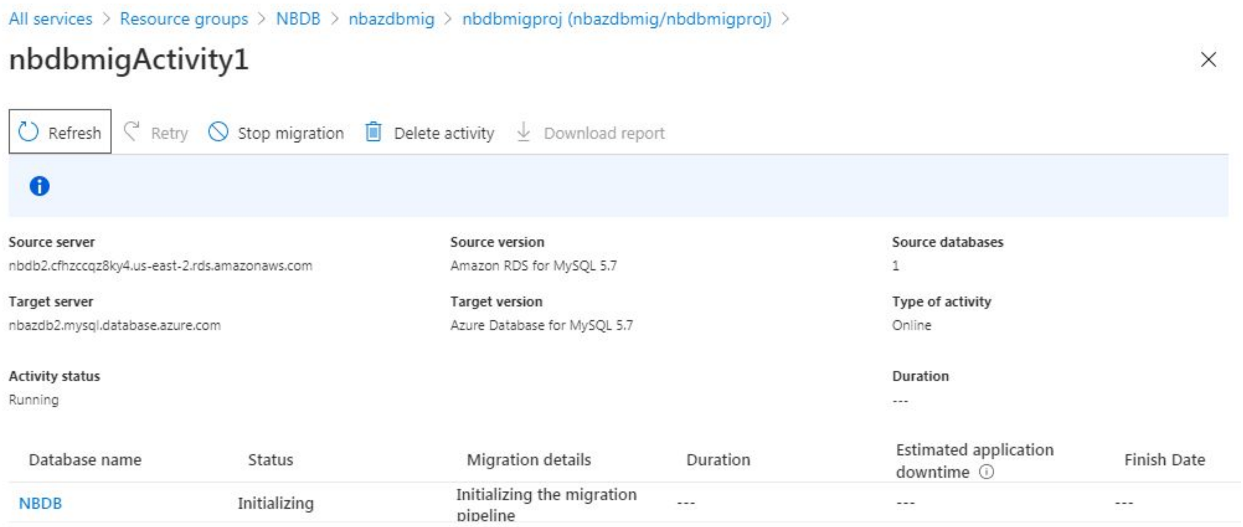
Target server version
Azure Database for MySQL
5.7.29-log

Source server name
nbdb2.cfhzccqz8ky4.us-east-2.rds.amazonaws.com

Source server version
Amazon RDS for MySQL
5.7.30-log

Database(s) to migrate
1 of 2

- Monitor the migration


All services > Resource groups > NBDB > nbazdbmig > nbdbmigproj (nbazdbmig/nbdbmigproj) >
Refresh
Retry
Stop migration
Delete activity
Download report

nbdbmigActivity1

Source server
nbdb2.cfhzccqz8ky4.us-east-2.rds.amazonaws.com

Source version
Amazon RDS for MySQL 5.7

Source databases
1

Target server
nbazdb2.mysql.database.azure.com

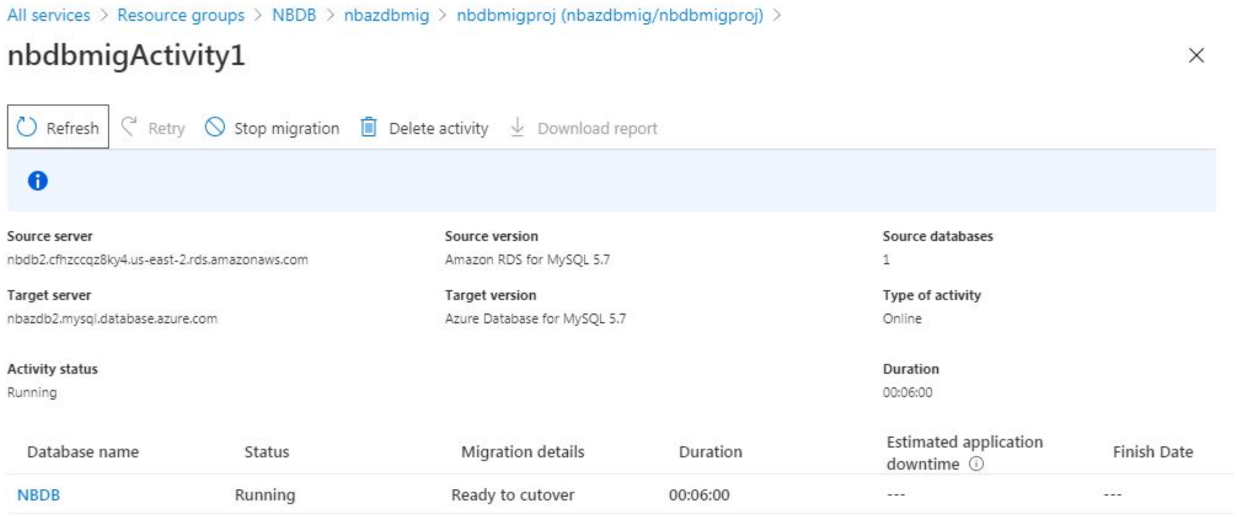
Target version
Azure Database for MySQL 5.7

Type of activity
Online

Activity status
Running

Duration

Database name	Status	Migration details	Duration	Estimated application downtime ⓘ	Finish Date
NBDB	Initializing	Initializing the migration pipeline	---	---	---


All services > Resource groups > NBDB > nbazdbmig > nbdbmigproj (nbazdbmig/nbdbmigproj) >
Refresh
Retry
Stop migration
Delete activity
Download report

nbdbmigActivity1

Source server
nbdb2.cfhzccqz8ky4.us-east-2.rds.amazonaws.com

Source version
Amazon RDS for MySQL 5.7

Source databases
1

Target server
nbazdb2.mysql.database.azure.com

Target version
Azure Database for MySQL 5.7

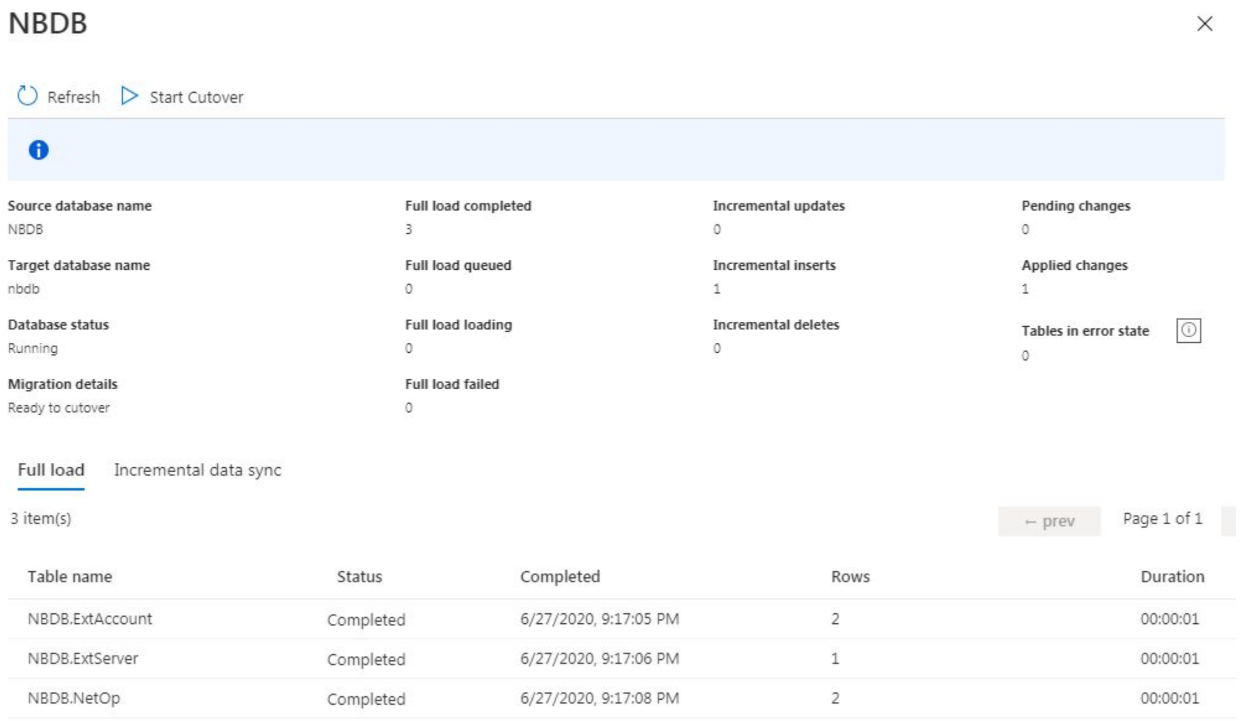
Type of activity
Online

Activity status
Running

Duration
00:06:00

Database name	Status	Migration details	Duration	Estimated application downtime ⓘ	Finish Date
NBDB	Running	Ready to cutover	00:06:00	---	---

Click on the database name to see details.



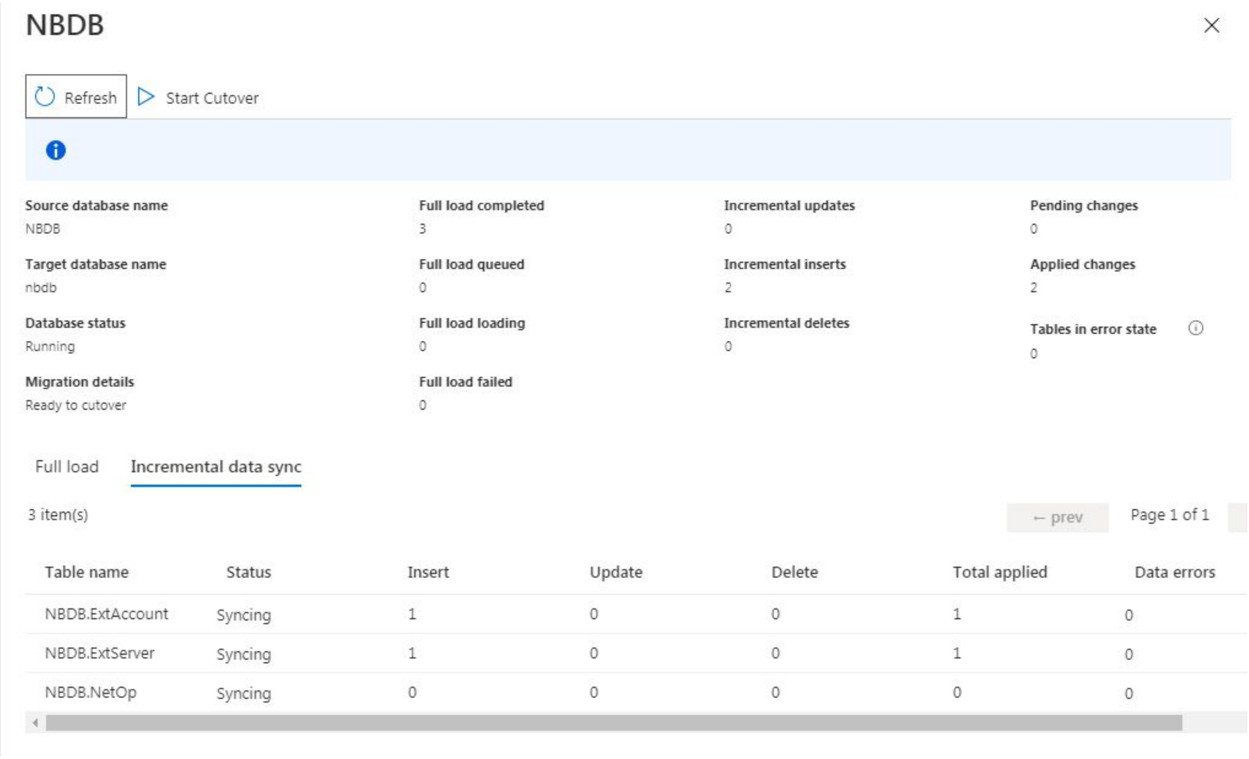
Source database name	**Full load completed**	**Incremental updates**	**Pending changes**
NBDB	3	0	0
Target database name	**Full load queued**	**Incremental inserts**	**Applied changes**
nbdb	0	1	1
Database status	**Full load loading**	**Incremental deletes**	**Tables in error state**
Running	0	0	0
Migration details	**Full load failed**		
Ready to cutover	0		

Table name	Status	Completed	Rows	Duration
NBDB.ExtAccount	Completed	6/27/2020, 9:17:05 PM	2	00:00:01
NBDB.ExtServer	Completed	6/27/2020, 9:17:06 PM	1	00:00:01
NBDB.NetOp	Completed	6/27/2020, 9:17:08 PM	2	00:00:01

Use HeidiSQL to verify that the Azure MySQL database contains the rows from the source table.

Add some more rows to the AWS RDS source table and they should replicate to the Azure target table within a few minutes.

Click on the Incremental Data Sync tab to view the details of the changed rows that are being synchronized.



Source database name
NBDB

Target database name
nbdb

Database status
Running

Migration details
Ready to cutover

Full load completed
3

Full load queued
0

Full load loading
0

Full load failed
0

Incremental updates
0

Incremental inserts
2

Incremental deletes
0

Pending changes
0

Applied changes
2

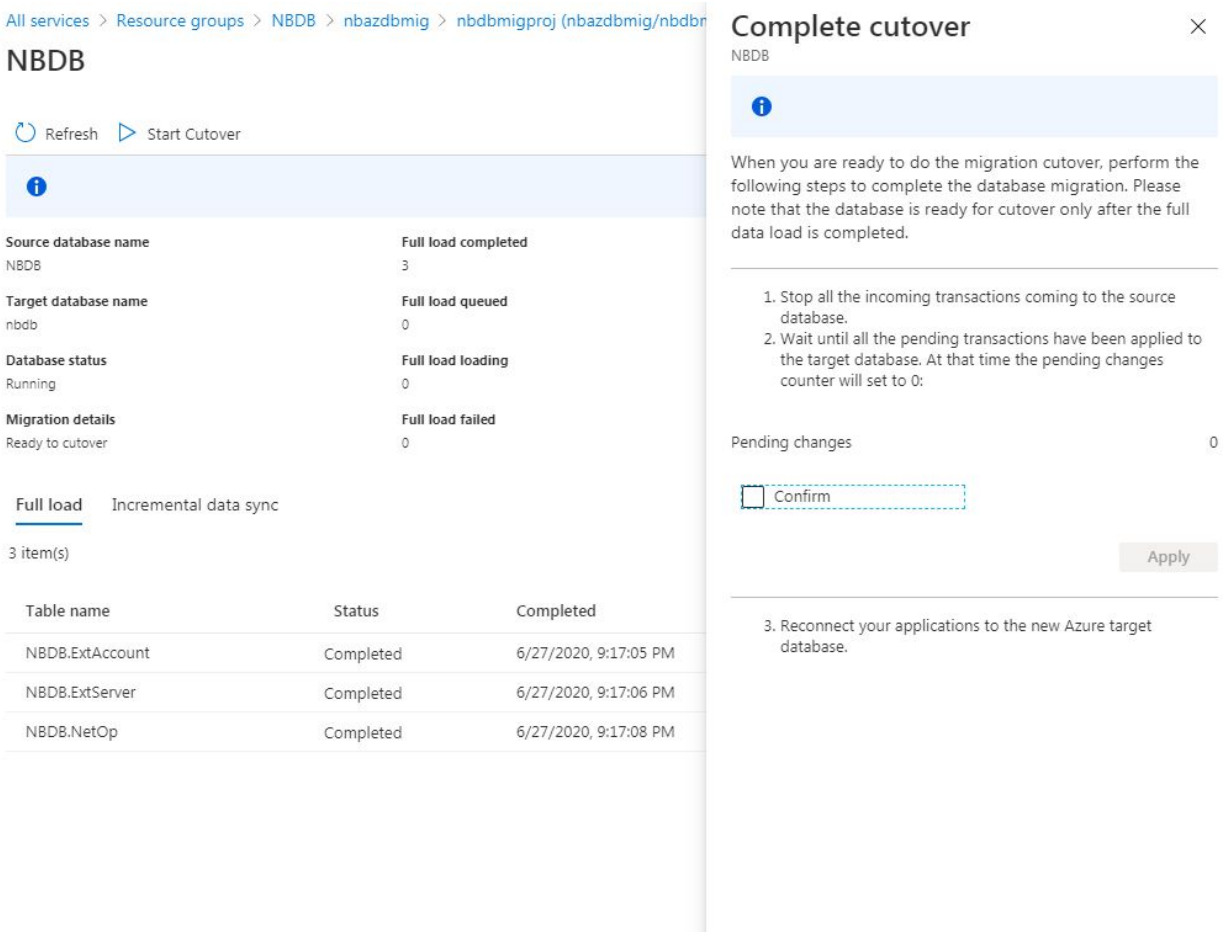
Tables in error state
0

3 item(s)

Table name	Status	Insert	Update	Delete	Total applied	Data errors
NBDB.ExtAccount	Syncing	1	0	0	1	0
NBDB.ExtServer	Syncing	1	0	0	1	0
NBDB.NetOp	Syncing	0	0	0	0	0

- Perform Migration Cutover

  The databases will not be available during the cutover so the cutover must be started in a maintenance window and applications must stop sending traffic to the source AWS RDS database.
  Select Start Cutover.



All services > Resource groups > NBDB > nbazdbmig > nbdbmigproj (nbazdbmig/nbdbm

NBDB

Source database name NBDB	**Full load completed** 3
Target database name nbdb	**Full load queued** 0
Database status Running	**Full load loading** 0
Migration details Ready to cutover	**Full load failed** 0

3 item(s)

Table name	Status	Completed
NBDB.ExtAccount	Completed	6/27/2020, 9:17:05 PM
NBDB.ExtServer	Completed	6/27/2020, 9:17:06 PM
NBDB.NetOp	Completed	6/27/2020, 9:17:08 PM

Complete cutover

NBDB

When you are ready to do the migration cutover, perform the following steps to complete the database migration. Please note that the database is ready for cutover only after the full data load is completed.

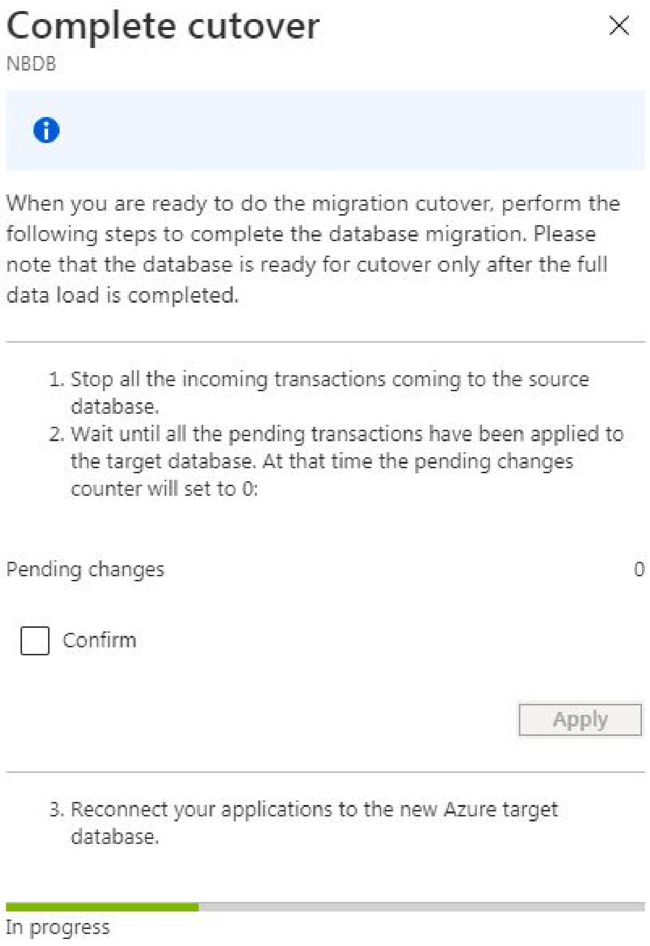
1. Stop all the incoming transactions coming to the source database.
2. Wait until all the pending transactions have been applied to the target database. At that time the pending changes counter will set to 0:

Pending changes 0

□ Confirm

3. Reconnect your applications to the new Azure target database.

Once you confirm, the cutover will start and a progress bar will be displayed.



Complete cutover

NBDB

When you are ready to do the migration cutover, perform the following steps to complete the database migration. Please note that the database is ready for cutover only after the full data load is completed.

1. Stop all the incoming transactions coming to the source database.
2. Wait until all the pending transactions have been applied to the target database. At that time the pending changes counter will set to 0:

Pending changes 0

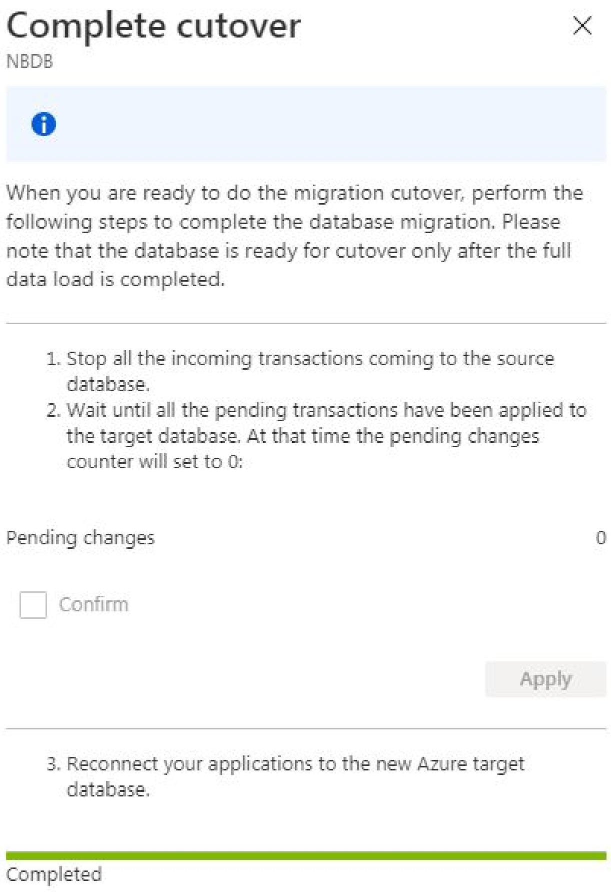
☐ Confirm

Apply

3. Reconnect your applications to the new Azure target database.

In progress

Once completed, close the cutover panel.



Complete cutover

NBDB

When you are ready to do the migration cutover, perform the following steps to complete the database migration. Please note that the database is ready for cutover only after the full data load is completed.

1. Stop all the incoming transactions coming to the source database.
2. Wait until all the pending transactions have been applied to the target database. At that time the pending changes counter will set to 0:

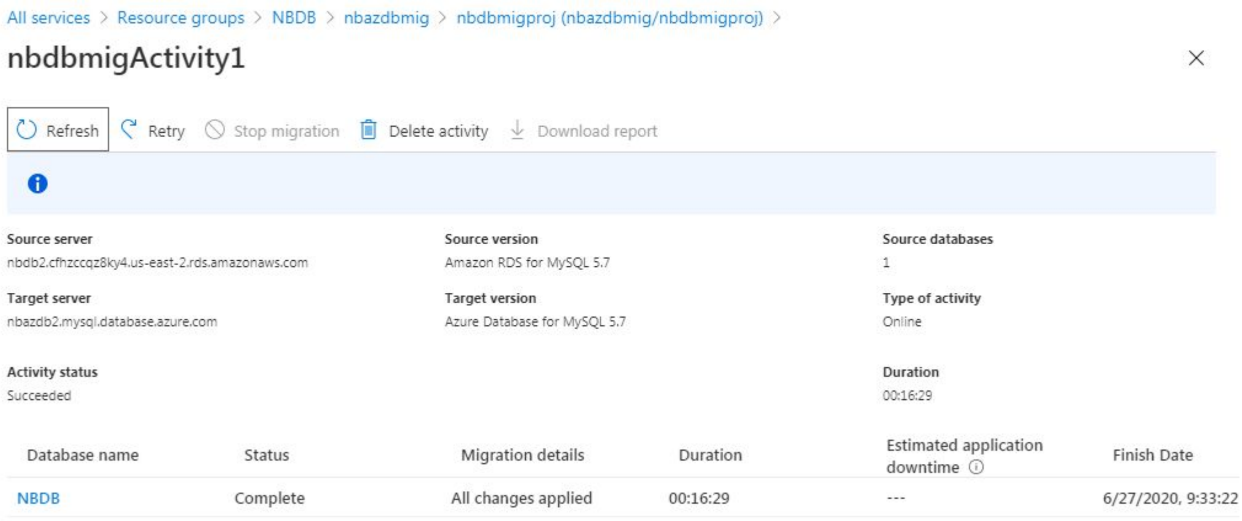
Pending changes 0

□ Confirm

3. Reconnect your applications to the new Azure target database.

Completed

When you go back to the activity after the Cutover completes, it will show a Status of Complete and All changes applied.



All services > Resource groups > NBDB > nbazdbmig > nbdbmigproj (nbazdbmig/nbdbmigproj) >

nbdbmigActivity1

Refresh
Retry
Stop migration
Delete activity
Download report

Source server: nbdb2.cfhzccqz8ky4.us-east-2.rds.amazonaws.com

Source version: Amazon RDS for MySQL 5.7

Source databases: 1

Target server: nbazdb2.mysql.database.azure.com

Target version: Azure Database for MySQL 5.7

Type of activity: Online

Activity status: Succeeded

Duration: 00:16:29

Database name	Status	Migration details	Duration	Estimated application downtime ⓘ	Finish Date
NBDB	Complete	All changes applied	00:16:29	---	6/27/2020, 9:33:22

Further changes to the source AWS RDS will no longer be synchronized. The Azure MySQL database is ready and application connection strings must be changed to target the new database.