Database User Roles

Every database includes the following client roles:

read

Provides the ability to read data on all *non*-system collections and on the following system collections: [system.indexes](https://docs.mongodb.com/manual/reference/system-collections/#<database>.system.indexes), [system.js](https://docs.mongodb.com/manual/reference/system-collections/#<database>.system.js), and [system.namespaces](https://docs.mongodb.com/manual/reference/system-collections/#<database>.system.namespaces) collections. The role provides read access by granting the following [actions](https://docs.mongodb.com/manual/reference/privilege-actions/#security-user-actions):

* [collStats](https://docs.mongodb.com/manual/reference/privilege-actions/#collStats)
* [dbHash](https://docs.mongodb.com/manual/reference/privilege-actions/#dbHash)
* [dbStats](https://docs.mongodb.com/manual/reference/privilege-actions/#dbStats)
* [find](https://docs.mongodb.com/manual/reference/privilege-actions/#find)
* [killCursors](https://docs.mongodb.com/manual/reference/privilege-actions/#killCursors)
* [listIndexes](https://docs.mongodb.com/manual/reference/privilege-actions/#listIndexes)
* [listCollections](https://docs.mongodb.com/manual/reference/privilege-actions/#listCollections)

readWrite

Provides all the privileges of the read role plus ability to modify data on all *non*-system collections and the [system.js](https://docs.mongodb.com/manual/reference/system-collections/#<database>.system.js) collection. The role provides the following actions on those collections:

* [collStats](https://docs.mongodb.com/manual/reference/privilege-actions/#collStats)
* [convertToCapped](https://docs.mongodb.com/manual/reference/privilege-actions/#convertToCapped)
* [createCollection](https://docs.mongodb.com/manual/reference/privilege-actions/#createCollection)
* [dbHash](https://docs.mongodb.com/manual/reference/privilege-actions/#dbHash)
* [dbStats](https://docs.mongodb.com/manual/reference/privilege-actions/#dbStats)
* [dropCollection](https://docs.mongodb.com/manual/reference/privilege-actions/#dropCollection)
* [createIndex](https://docs.mongodb.com/manual/reference/privilege-actions/#createIndex)
* [dropIndex](https://docs.mongodb.com/manual/reference/privilege-actions/#dropIndex)
* [find](https://docs.mongodb.com/manual/reference/privilege-actions/#find)
* [insert](https://docs.mongodb.com/manual/reference/privilege-actions/#insert)
* [killCursors](https://docs.mongodb.com/manual/reference/privilege-actions/#killCursors)
* [listIndexes](https://docs.mongodb.com/manual/reference/privilege-actions/#listIndexes)
* [listCollections](https://docs.mongodb.com/manual/reference/privilege-actions/#listCollections)
* [remove](https://docs.mongodb.com/manual/reference/privilege-actions/#remove)
* [renameCollectionSameDB](https://docs.mongodb.com/manual/reference/privilege-actions/#renameCollectionSameDB)
* [update](https://docs.mongodb.com/manual/reference/privilege-actions/#update)

Database Administration Roles

Every database includes the following database administration roles:

dbAdmin

Provides the following [actions](https://docs.mongodb.com/manual/reference/privilege-actions/#security-user-actions) on the database’s [system.indexes](https://docs.mongodb.com/manual/reference/system-collections/#<database>.system.indexes), [system.namespaces](https://docs.mongodb.com/manual/reference/system-collections/#<database>.system.namespaces), and[system.profile](https://docs.mongodb.com/manual/reference/system-collections/#<database>.system.profile) collections:

* [collStats](https://docs.mongodb.com/manual/reference/privilege-actions/#collStats)
* [dbHash](https://docs.mongodb.com/manual/reference/privilege-actions/#dbHash)
* [dbStats](https://docs.mongodb.com/manual/reference/privilege-actions/#dbStats)
* [find](https://docs.mongodb.com/manual/reference/privilege-actions/#find)
* [killCursors](https://docs.mongodb.com/manual/reference/privilege-actions/#killCursors)
* [listIndexes](https://docs.mongodb.com/manual/reference/privilege-actions/#listIndexes)
* [listCollections](https://docs.mongodb.com/manual/reference/privilege-actions/#listCollections)
* [dropCollection](https://docs.mongodb.com/manual/reference/privilege-actions/#dropCollection) and [createCollection](https://docs.mongodb.com/manual/reference/privilege-actions/#createCollection) on [system.profile](https://docs.mongodb.com/manual/reference/system-collections/#<database>.system.profile) *only*

*Changed in version 2.6.4:*dbAdmin added the [createCollection](https://docs.mongodb.com/manual/reference/privilege-actions/#createCollection) for the [system.profile](https://docs.mongodb.com/manual/reference/system-collections/#<database>.system.profile)collection. Previous versions only had the [dropCollection](https://docs.mongodb.com/manual/reference/privilege-actions/#dropCollection) on the [system.profile](https://docs.mongodb.com/manual/reference/system-collections/#<database>.system.profile) collection.

Provides the following actions on all *non*-system collections. This role *does not* include full read access on non-system collections:

* [bypassDocumentValidation](https://docs.mongodb.com/manual/reference/privilege-actions/#bypassDocumentValidation)
* [collMod](https://docs.mongodb.com/manual/reference/privilege-actions/#collMod)
* [collStats](https://docs.mongodb.com/manual/reference/privilege-actions/#collStats)
* [compact](https://docs.mongodb.com/manual/reference/privilege-actions/#compact)
* [convertToCapped](https://docs.mongodb.com/manual/reference/privilege-actions/#convertToCapped)
* [createCollection](https://docs.mongodb.com/manual/reference/privilege-actions/#createCollection)
* [createIndex](https://docs.mongodb.com/manual/reference/privilege-actions/#createIndex)
* [dbStats](https://docs.mongodb.com/manual/reference/privilege-actions/#dbStats)
* [dropCollection](https://docs.mongodb.com/manual/reference/privilege-actions/#dropCollection)
* [dropDatabase](https://docs.mongodb.com/manual/reference/privilege-actions/#dropDatabase)
* [dropIndex](https://docs.mongodb.com/manual/reference/privilege-actions/#dropIndex)
* [enableProfiler](https://docs.mongodb.com/manual/reference/privilege-actions/#enableProfiler)
* [reIndex](https://docs.mongodb.com/manual/reference/privilege-actions/#reIndex)
* [renameCollectionSameDB](https://docs.mongodb.com/manual/reference/privilege-actions/#renameCollectionSameDB)
* [repairDatabase](https://docs.mongodb.com/manual/reference/privilege-actions/#repairDatabase)
* [storageDetails](https://docs.mongodb.com/manual/reference/privilege-actions/#storageDetails)
* [validate](https://docs.mongodb.com/manual/reference/privilege-actions/#validate)

dbOwner

The database owner can perform any administrative action on the database. This role combines the privileges granted by the readWrite, dbAdmin and userAdmin roles.

userAdmin

Provides the ability to create and modify roles and users for a database. A user with this role on a database can assign any role or privilege to any user for that database, including themselves.

The userAdmin role explicitly provides the following actions:

* [changeCustomData](https://docs.mongodb.com/manual/reference/privilege-actions/#changeCustomData)
* [changePassword](https://docs.mongodb.com/manual/reference/privilege-actions/#changePassword)
* [createRole](https://docs.mongodb.com/manual/reference/privilege-actions/#createRole)
* [createUser](https://docs.mongodb.com/manual/reference/privilege-actions/#createUser)
* [dropRole](https://docs.mongodb.com/manual/reference/privilege-actions/#dropRole)
* [dropUser](https://docs.mongodb.com/manual/reference/privilege-actions/#dropUser)
* [grantRole](https://docs.mongodb.com/manual/reference/privilege-actions/#grantRole)
* [revokeRole](https://docs.mongodb.com/manual/reference/privilege-actions/#revokeRole)
* [setAuthenticationRestriction](https://docs.mongodb.com/manual/reference/privilege-actions/#setAuthenticationRestriction)
* [viewRole](https://docs.mongodb.com/manual/reference/privilege-actions/#viewRole)
* [viewUser](https://docs.mongodb.com/manual/reference/privilege-actions/#viewUser)

**WARNING**

It is important to understand the security implications of granting the userAdmin role: a user with this role for a database can assign themselves any privilege on that database. Granting theuserAdmin role on the admin database has further security implications as this indirectly provides[superuser](https://docs.mongodb.com/manual/reference/built-in-roles/#superuser) access to a cluster. With admin scope a user with the userAdmin role can grant cluster-wide roles or privileges including userAdminAnyDatabase.

Cluster Administration Roles

The admin database includes the following roles for administering the whole system rather than just a single database. These roles include but are not limited to [replica set](https://docs.mongodb.com/manual/reference/glossary/#term-replica-set) and [sharded cluster](https://docs.mongodb.com/manual/reference/glossary/#term-sharded-cluster) administrative functions.

clusterAdmin

Provides the greatest cluster-management access. This role combines the privileges granted by the clusterManager, clusterMonitor, and hostManager roles. Additionally, the role provides the [dropDatabase](https://docs.mongodb.com/manual/reference/privilege-actions/#dropDatabase) action.

clusterManager

*Changed in version 3.4.*

Provides management and monitoring actions on the cluster. A user with this role can access the configand local databases, which are used in sharding and replication, respectively.

Provides the following actions on the cluster as a whole:

* [addShard](https://docs.mongodb.com/manual/reference/privilege-actions/#addShard)
* [appendOplogNote](https://docs.mongodb.com/manual/reference/privilege-actions/#appendOplogNote)
* [applicationMessage](https://docs.mongodb.com/manual/reference/privilege-actions/#applicationMessage)
* [cleanupOrphaned](https://docs.mongodb.com/manual/reference/privilege-actions/#cleanupOrphaned)
* [flushRouterConfig](https://docs.mongodb.com/manual/reference/privilege-actions/#flushRouterConfig)
* [listSessions](https://docs.mongodb.com/manual/reference/privilege-actions/#listSessions) (New in version 3.6)
* [listShards](https://docs.mongodb.com/manual/reference/privilege-actions/#listShards)
* [removeShard](https://docs.mongodb.com/manual/reference/privilege-actions/#removeShard)
* [replSetConfigure](https://docs.mongodb.com/manual/reference/privilege-actions/#replSetConfigure)
* [replSetGetConfig](https://docs.mongodb.com/manual/reference/privilege-actions/#replSetGetConfig)
* [replSetGetStatus](https://docs.mongodb.com/manual/reference/privilege-actions/#replSetGetStatus)
* [replSetStateChange](https://docs.mongodb.com/manual/reference/privilege-actions/#replSetStateChange)
* [resync](https://docs.mongodb.com/manual/reference/privilege-actions/#resync)

Provides the following actions on *all* databases in the cluster:

* [enableSharding](https://docs.mongodb.com/manual/reference/privilege-actions/#enableSharding)
* [moveChunk](https://docs.mongodb.com/manual/reference/privilege-actions/#moveChunk)
* [splitChunk](https://docs.mongodb.com/manual/reference/privilege-actions/#splitChunk)
* [splitVector](https://docs.mongodb.com/manual/reference/privilege-actions/#splitVector)

On the config database, provides the following privileges:

| **Resource** | **Actions** |
| --- | --- |
| All collections in the config database | [collStats](https://docs.mongodb.com/manual/reference/privilege-actions/#collStats)  [dbHash](https://docs.mongodb.com/manual/reference/privilege-actions/#dbHash)  [dbStats](https://docs.mongodb.com/manual/reference/privilege-actions/#dbStats)  [enableSharding](https://docs.mongodb.com/manual/reference/privilege-actions/#enableSharding)  [find](https://docs.mongodb.com/manual/reference/privilege-actions/#find)  [insert](https://docs.mongodb.com/manual/reference/privilege-actions/#insert)  [killCursors](https://docs.mongodb.com/manual/reference/privilege-actions/#killCursors)  [listCollections](https://docs.mongodb.com/manual/reference/privilege-actions/#listCollections)  [listIndexes](https://docs.mongodb.com/manual/reference/privilege-actions/#listIndexes)  [moveChunk](https://docs.mongodb.com/manual/reference/privilege-actions/#moveChunk)  [planCacheRead](https://docs.mongodb.com/manual/reference/privilege-actions/#planCacheRead)  [remove](https://docs.mongodb.com/manual/reference/privilege-actions/#remove)  [splitChunk](https://docs.mongodb.com/manual/reference/privilege-actions/#splitChunk)  [splitVector](https://docs.mongodb.com/manual/reference/privilege-actions/#splitVector)  [update](https://docs.mongodb.com/manual/reference/privilege-actions/#update) |
| [system.indexes](https://docs.mongodb.com/manual/reference/system-collections/#<database>.system.indexes),  [system.js](https://docs.mongodb.com/manual/reference/system-collections/#<database>.system.js),  [system.namespaces](https://docs.mongodb.com/manual/reference/system-collections/#<database>.system.namespaces) collections | [collStats](https://docs.mongodb.com/manual/reference/privilege-actions/#collStats)  [dbHash](https://docs.mongodb.com/manual/reference/privilege-actions/#dbHash)  [dbStats](https://docs.mongodb.com/manual/reference/privilege-actions/#dbStats)  [find](https://docs.mongodb.com/manual/reference/privilege-actions/#find)  [killCursors](https://docs.mongodb.com/manual/reference/privilege-actions/#killCursors)  [listCollections](https://docs.mongodb.com/manual/reference/privilege-actions/#listCollections)  [listIndexes](https://docs.mongodb.com/manual/reference/privilege-actions/#listIndexes)  [planCacheRead](https://docs.mongodb.com/manual/reference/privilege-actions/#planCacheRead) |

On the local database, provides the following privileges:

| **Resource** | **Actions** |
| --- | --- |
| All collections in the local database | * [enableSharding](https://docs.mongodb.com/manual/reference/privilege-actions/#enableSharding) * [insert](https://docs.mongodb.com/manual/reference/privilege-actions/#insert) * [moveChunk](https://docs.mongodb.com/manual/reference/privilege-actions/#moveChunk) * [remove](https://docs.mongodb.com/manual/reference/privilege-actions/#remove) * [splitChunk](https://docs.mongodb.com/manual/reference/privilege-actions/#splitChunk) * [splitVector](https://docs.mongodb.com/manual/reference/privilege-actions/#splitVector) * [update](https://docs.mongodb.com/manual/reference/privilege-actions/#update) |
| [system.replset](https://docs.mongodb.com/manual/reference/local-database/#local.system.replset) collection | * [collStats](https://docs.mongodb.com/manual/reference/privilege-actions/#collStats) * [dbHash](https://docs.mongodb.com/manual/reference/privilege-actions/#dbHash) * [dbStats](https://docs.mongodb.com/manual/reference/privilege-actions/#dbStats) * [find](https://docs.mongodb.com/manual/reference/privilege-actions/#find) * [killCursors](https://docs.mongodb.com/manual/reference/privilege-actions/#killCursors) * [listCollections](https://docs.mongodb.com/manual/reference/privilege-actions/#listCollections) * [listIndexes](https://docs.mongodb.com/manual/reference/privilege-actions/#listIndexes) * [planCacheRead](https://docs.mongodb.com/manual/reference/privilege-actions/#planCacheRead) |

clusterMonitor

*Changed in version 3.4.*

Provides read-only access to monitoring tools, such as the [MongoDB Cloud Manager](https://www.mongodb.com/cloud/cloud-manager/?jmp=docs) and [Ops Manager](https://docs.opsmanager.mongodb.com/current/) monitoring agent.

Provides the following actions on the cluster as a whole:

* [connPoolStats](https://docs.mongodb.com/manual/reference/privilege-actions/#connPoolStats)
* [getCmdLineOpts](https://docs.mongodb.com/manual/reference/privilege-actions/#getCmdLineOpts)
* [getLog](https://docs.mongodb.com/manual/reference/privilege-actions/#getLog)
* [getParameter](https://docs.mongodb.com/manual/reference/privilege-actions/#getParameter)
* [getShardMap](https://docs.mongodb.com/manual/reference/privilege-actions/#getShardMap)
* [hostInfo](https://docs.mongodb.com/manual/reference/privilege-actions/#hostInfo)
* [inprog](https://docs.mongodb.com/manual/reference/privilege-actions/#inprog)
* [listDatabases](https://docs.mongodb.com/manual/reference/privilege-actions/#listDatabases)
* [listSessions](https://docs.mongodb.com/manual/reference/privilege-actions/#listSessions) (New in version 3.6)
* [listShards](https://docs.mongodb.com/manual/reference/privilege-actions/#listShards)
* [netstat](https://docs.mongodb.com/manual/reference/privilege-actions/#netstat)
* [replSetGetConfig](https://docs.mongodb.com/manual/reference/privilege-actions/#replSetGetConfig)
* [replSetGetStatus](https://docs.mongodb.com/manual/reference/privilege-actions/#replSetGetStatus)
* [serverStatus](https://docs.mongodb.com/manual/reference/privilege-actions/#serverStatus)
* [shardingState](https://docs.mongodb.com/manual/reference/privilege-actions/#shardingState)
* [top](https://docs.mongodb.com/manual/reference/privilege-actions/#top)

Provides the following actions on *all* databases in the cluster:

* [collStats](https://docs.mongodb.com/manual/reference/privilege-actions/#collStats)
* [dbStats](https://docs.mongodb.com/manual/reference/privilege-actions/#dbStats)
* [getShardVersion](https://docs.mongodb.com/manual/reference/privilege-actions/#getShardVersion)
* [indexStats](https://docs.mongodb.com/manual/reference/privilege-actions/#indexStats)
* [useUUID](https://docs.mongodb.com/manual/reference/privilege-actions/#useUUID) (New in version 3.6)

Provides the [find](https://docs.mongodb.com/manual/reference/privilege-actions/#find) action on all [system.profile](https://docs.mongodb.com/manual/reference/system-collections/#<database>.system.profile) collections in the cluster.

On the config database, provides the following privileges:

| **Resource** | **Actions** |
| --- | --- |
| All collections in the config database | [collStats](https://docs.mongodb.com/manual/reference/privilege-actions/#collStats)  [dbHash](https://docs.mongodb.com/manual/reference/privilege-actions/#dbHash)  [dbStats](https://docs.mongodb.com/manual/reference/privilege-actions/#dbStats)  [find](https://docs.mongodb.com/manual/reference/privilege-actions/#find)  [getShardVersion](https://docs.mongodb.com/manual/reference/privilege-actions/#getShardVersion)  [indexStats](https://docs.mongodb.com/manual/reference/privilege-actions/#indexStats)  [killCursors](https://docs.mongodb.com/manual/reference/privilege-actions/#killCursors)  [listCollections](https://docs.mongodb.com/manual/reference/privilege-actions/#listCollections)  [listIndexes](https://docs.mongodb.com/manual/reference/privilege-actions/#listIndexes)  [planCacheRead](https://docs.mongodb.com/manual/reference/privilege-actions/#planCacheRead) |
| [system.indexes](https://docs.mongodb.com/manual/reference/system-collections/#<database>.system.indexes),  [system.js](https://docs.mongodb.com/manual/reference/system-collections/#<database>.system.js),  [system.namespaces](https://docs.mongodb.com/manual/reference/system-collections/#<database>.system.namespaces) collections | [collStats](https://docs.mongodb.com/manual/reference/privilege-actions/#collStats)  [dbHash](https://docs.mongodb.com/manual/reference/privilege-actions/#dbHash)  [dbStats](https://docs.mongodb.com/manual/reference/privilege-actions/#dbStats)  [find](https://docs.mongodb.com/manual/reference/privilege-actions/#find)  [killCursors](https://docs.mongodb.com/manual/reference/privilege-actions/#killCursors)  [listCollections](https://docs.mongodb.com/manual/reference/privilege-actions/#listCollections)  [listIndexes](https://docs.mongodb.com/manual/reference/privilege-actions/#listIndexes)  [planCacheRead](https://docs.mongodb.com/manual/reference/privilege-actions/#planCacheRead) |

On the local database, provides the following privileges:

| **Resource** | **Actions** |
| --- | --- |
| All collections in the local database | [collStats](https://docs.mongodb.com/manual/reference/privilege-actions/#collStats)  [dbHash](https://docs.mongodb.com/manual/reference/privilege-actions/#dbHash)  [dbStats](https://docs.mongodb.com/manual/reference/privilege-actions/#dbStats)  [find](https://docs.mongodb.com/manual/reference/privilege-actions/#find)  [getShardVersion](https://docs.mongodb.com/manual/reference/privilege-actions/#getShardVersion)  [indexStats](https://docs.mongodb.com/manual/reference/privilege-actions/#indexStats)  [killCursors](https://docs.mongodb.com/manual/reference/privilege-actions/#killCursors)  [listCollections](https://docs.mongodb.com/manual/reference/privilege-actions/#listCollections)  [listIndexes](https://docs.mongodb.com/manual/reference/privilege-actions/#listIndexes)  [planCacheRead](https://docs.mongodb.com/manual/reference/privilege-actions/#planCacheRead) |
| [system.indexes](https://docs.mongodb.com/manual/reference/system-collections/#<database>.system.indexes),  [system.js](https://docs.mongodb.com/manual/reference/system-collections/#<database>.system.js),  [system.namespaces](https://docs.mongodb.com/manual/reference/system-collections/#<database>.system.namespaces) collections | [collStats](https://docs.mongodb.com/manual/reference/privilege-actions/#collStats)  [dbHash](https://docs.mongodb.com/manual/reference/privilege-actions/#dbHash)  [dbStats](https://docs.mongodb.com/manual/reference/privilege-actions/#dbStats)  [find](https://docs.mongodb.com/manual/reference/privilege-actions/#find)  [killCursors](https://docs.mongodb.com/manual/reference/privilege-actions/#killCursors)  [listCollections](https://docs.mongodb.com/manual/reference/privilege-actions/#listCollections)  [listIndexes](https://docs.mongodb.com/manual/reference/privilege-actions/#listIndexes)  [planCacheRead](https://docs.mongodb.com/manual/reference/privilege-actions/#planCacheRead) |
| system.replset,  [system.profile](https://docs.mongodb.com/manual/reference/system-collections/#<database>.system.profile), | [find](https://docs.mongodb.com/manual/reference/privilege-actions/#find) |

hostManager

Provides the ability to monitor and manage servers.

Provides the following actions on the cluster as a whole:

* [applicationMessage](https://docs.mongodb.com/manual/reference/privilege-actions/#applicationMessage)
* [closeAllDatabases](https://docs.mongodb.com/manual/reference/privilege-actions/#closeAllDatabases)
* [connPoolSync](https://docs.mongodb.com/manual/reference/privilege-actions/#connPoolSync)
* [cpuProfiler](https://docs.mongodb.com/manual/reference/privilege-actions/#cpuProfiler)
* diagLogging
* [flushRouterConfig](https://docs.mongodb.com/manual/reference/privilege-actions/#flushRouterConfig)
* [fsync](https://docs.mongodb.com/manual/reference/privilege-actions/#fsync)
* [invalidateUserCache](https://docs.mongodb.com/manual/reference/privilege-actions/#invalidateUserCache)
* [killAnySession](https://docs.mongodb.com/manual/reference/privilege-actions/#killAnySession) (New in version 3.6)
* [killop](https://docs.mongodb.com/manual/reference/privilege-actions/#killop)
* [logRotate](https://docs.mongodb.com/manual/reference/privilege-actions/#logRotate)
* [resync](https://docs.mongodb.com/manual/reference/privilege-actions/#resync)
* [setParameter](https://docs.mongodb.com/manual/reference/privilege-actions/#setParameter)
* [shutdown](https://docs.mongodb.com/manual/reference/privilege-actions/#shutdown)
* [touch](https://docs.mongodb.com/manual/reference/privilege-actions/#touch)
* [unlock](https://docs.mongodb.com/manual/reference/privilege-actions/#unlock)

Provides the following actions on *all* databases in the cluster:

* [killCursors](https://docs.mongodb.com/manual/reference/privilege-actions/#killCursors)
* [repairDatabase](https://docs.mongodb.com/manual/reference/privilege-actions/#repairDatabase)

Backup and Restoration Roles

The admin database includes the following roles for backing up and restoring data:

backup

*Changed in version 3.4.*

Provides minimal privileges needed for backing up data. This role provides sufficient privileges to use the [MongoDB Cloud Manager](https://www.mongodb.com/cloud/cloud-manager/?jmp=docs) backup agent, [Ops Manager](https://docs.opsmanager.mongodb.com/current/) backup agent, or to use [mongodump](https://docs.mongodb.com/manual/reference/program/mongodump/#bin.mongodump) to back up an entire [mongod](https://docs.mongodb.com/manual/reference/program/mongod/#bin.mongod) instance.

Provides the [insert](https://docs.mongodb.com/manual/reference/privilege-actions/#insert) and [update](https://docs.mongodb.com/manual/reference/privilege-actions/#update) actions on the mms.backup collection in the admin database and on the [settings](https://docs.mongodb.com/manual/reference/config-database/#config.settings) collection in the config database.

On [anyResource](https://docs.mongodb.com/manual/reference/resource-document/#anyresource), provides the

* [listDatabases](https://docs.mongodb.com/manual/reference/privilege-actions/#listDatabases) action
* [listCollections](https://docs.mongodb.com/manual/reference/privilege-actions/#listCollections) action
* [listIndexes](https://docs.mongodb.com/manual/reference/privilege-actions/#listIndexes) action

On the [cluster](https://docs.mongodb.com/manual/reference/resource-document/#resource-cluster) as a whole, provides the

* [appendOplogNote](https://docs.mongodb.com/manual/reference/privilege-actions/#appendOplogNote)
* [getParameter](https://docs.mongodb.com/manual/reference/privilege-actions/#getParameter)
* [listDatabases](https://docs.mongodb.com/manual/reference/privilege-actions/#listDatabases)

Provides the [find](https://docs.mongodb.com/manual/reference/privilege-actions/#find) action on the following:

* all *non*-system collections in the cluster, including those in the config and local databases
* The following system collections in the cluster: [system.indexes](https://docs.mongodb.com/manual/reference/system-collections/#<database>.system.indexes), [system.namespaces](https://docs.mongodb.com/manual/reference/system-collections/#<database>.system.namespaces),[system.js](https://docs.mongodb.com/manual/reference/system-collections/#<database>.system.js), and [system.profile](https://docs.mongodb.com/manual/reference/system-collections/#<database>.system.profile)
* the [admin.system.users](https://docs.mongodb.com/manual/reference/system-collections/#admin.system.users) and [admin.system.roles](https://docs.mongodb.com/manual/reference/system-collections/#admin.system.roles) collections
* the [config.settings](https://docs.mongodb.com/manual/reference/config-database/#config.settings) collection
* legacy system.users collections from versions of MongoDB prior to 2.6

Provides [insert](https://docs.mongodb.com/manual/reference/privilege-actions/#insert) and [update](https://docs.mongodb.com/manual/reference/privilege-actions/#update) action on the [config.settings](https://docs.mongodb.com/manual/reference/config-database/#config.settings) collection.

*Changed in version 3.2.1:*The backup role provides additional privileges to back up the [system.profile](https://docs.mongodb.com/manual/reference/system-collections/#<database>.system.profile) collections that exist when running with [database profiling](https://docs.mongodb.com/manual/administration/analyzing-mongodb-performance/#database-profiling). Previously, users required an additional read access on this collection.

restore

*Changed in version 3.6:*Provides [convertToCapped](https://docs.mongodb.com/manual/reference/privilege-actions/#convertToCapped) on non-system collections.

Provides privileges needed to restore data from backups that do not include [system.profile](https://docs.mongodb.com/manual/reference/system-collections/#<database>.system.profile)collection data. This role is sufficient when restoring data with [mongorestore](https://docs.mongodb.com/manual/reference/program/mongorestore/#bin.mongorestore) without the [--oplogReplay](https://docs.mongodb.com/manual/reference/program/mongorestore/#cmdoption-oplogreplay) option.

* If the backup data includes [system.profile](https://docs.mongodb.com/manual/reference/system-collections/#<database>.system.profile) collection data and the target database does not contain the [system.profile](https://docs.mongodb.com/manual/reference/system-collections/#<database>.system.profile) collection, [mongorestore](https://docs.mongodb.com/manual/reference/program/mongorestore/#bin.mongorestore) attempts to create the collection even though the program does not actually restore system.profile documents. As such, the user requires additional privileges to perform [createCollection](https://docs.mongodb.com/manual/reference/privilege-actions/#createCollection) and [convertToCapped](https://docs.mongodb.com/manual/reference/privilege-actions/#convertToCapped) actions on the [system.profile](https://docs.mongodb.com/manual/reference/system-collections/#<database>.system.profile) collection for a database.

The built-in roles dbAdmin and dbAdminAnyDatabase provide the additional privileges.

* If running [mongorestore](https://docs.mongodb.com/manual/reference/program/mongorestore/#bin.mongorestore) with [--oplogReplay](https://docs.mongodb.com/manual/reference/program/mongorestore/#cmdoption-oplogreplay), the restore role is insufficient to replay the oplog. To replay the oplog, create a [user-defined role](https://docs.mongodb.com/manual/tutorial/manage-users-and-roles/#create-user-defined-role) that has [anyAction](https://docs.mongodb.com/manual/reference/privilege-actions/#anyAction) on [anyResource](https://docs.mongodb.com/manual/reference/resource-document/#resource-anyresource) and grant only to users who must run [mongorestore](https://docs.mongodb.com/manual/reference/program/mongorestore/#bin.mongorestore) with [--oplogReplay](https://docs.mongodb.com/manual/reference/program/mongorestore/#cmdoption-oplogreplay).

Provides the following action on the cluster as a whole:

* [getParameter](https://docs.mongodb.com/manual/reference/privilege-actions/#getParameter)

Provides the following actions on all *non*-system collections:

* [bypassDocumentValidation](https://docs.mongodb.com/manual/reference/privilege-actions/#bypassDocumentValidation)
* [changeCustomData](https://docs.mongodb.com/manual/reference/privilege-actions/#changeCustomData)
* [changePassword](https://docs.mongodb.com/manual/reference/privilege-actions/#changePassword)
* [collMod](https://docs.mongodb.com/manual/reference/privilege-actions/#collMod)
* [convertToCapped](https://docs.mongodb.com/manual/reference/privilege-actions/#convertToCapped)
* [createCollection](https://docs.mongodb.com/manual/reference/privilege-actions/#createCollection)
* [createIndex](https://docs.mongodb.com/manual/reference/privilege-actions/#createIndex)
* [createRole](https://docs.mongodb.com/manual/reference/privilege-actions/#createRole)
* [createUser](https://docs.mongodb.com/manual/reference/privilege-actions/#createUser)
* [dropCollection](https://docs.mongodb.com/manual/reference/privilege-actions/#dropCollection)
* [dropRole](https://docs.mongodb.com/manual/reference/privilege-actions/#dropRole)
* [dropUser](https://docs.mongodb.com/manual/reference/privilege-actions/#dropUser)
* [grantRole](https://docs.mongodb.com/manual/reference/privilege-actions/#grantRole)
* [insert](https://docs.mongodb.com/manual/reference/privilege-actions/#insert)
* [revokeRole](https://docs.mongodb.com/manual/reference/privilege-actions/#revokeRole)
* [viewRole](https://docs.mongodb.com/manual/reference/privilege-actions/#viewRole)
* [viewUser](https://docs.mongodb.com/manual/reference/privilege-actions/#viewUser)

Provides the following actions on [system.js](https://docs.mongodb.com/manual/reference/system-collections/#<database>.system.js) collection:

* [bypassDocumentValidation](https://docs.mongodb.com/manual/reference/privilege-actions/#bypassDocumentValidation)
* [collMod](https://docs.mongodb.com/manual/reference/privilege-actions/#collMod)
* [createCollection](https://docs.mongodb.com/manual/reference/privilege-actions/#createCollection)
* [createIndex](https://docs.mongodb.com/manual/reference/privilege-actions/#createIndex)
* [dropCollection](https://docs.mongodb.com/manual/reference/privilege-actions/#dropCollection)
* [insert](https://docs.mongodb.com/manual/reference/privilege-actions/#insert)

Provides the following action on [anyResource](https://docs.mongodb.com/manual/reference/resource-document/#anyresource):

* [listCollections](https://docs.mongodb.com/manual/reference/privilege-actions/#listCollections)

Provides the [find](https://docs.mongodb.com/manual/reference/privilege-actions/#find) action on all the [system.namespaces](https://docs.mongodb.com/manual/reference/system-collections/#<database>.system.namespaces) collections in the cluster.

Provides the following actions on all non-system collections on the config and the local databases:

* [bypassDocumentValidation](https://docs.mongodb.com/manual/reference/privilege-actions/#bypassDocumentValidation)
* [collMod](https://docs.mongodb.com/manual/reference/privilege-actions/#collMod)
* [createCollection](https://docs.mongodb.com/manual/reference/privilege-actions/#createCollection)
* [createIndex](https://docs.mongodb.com/manual/reference/privilege-actions/#createIndex)
* [dropCollection](https://docs.mongodb.com/manual/reference/privilege-actions/#dropCollection)
* [insert](https://docs.mongodb.com/manual/reference/privilege-actions/#insert)

Provides the following actions on [admin.system.version](https://docs.mongodb.com/manual/reference/system-collections/#admin.system.version)

* [bypassDocumentValidation](https://docs.mongodb.com/manual/reference/privilege-actions/#bypassDocumentValidation)
* [collMod](https://docs.mongodb.com/manual/reference/privilege-actions/#collMod)
* [createCollection](https://docs.mongodb.com/manual/reference/privilege-actions/#createCollection)
* [createIndex](https://docs.mongodb.com/manual/reference/privilege-actions/#createIndex)
* [dropCollection](https://docs.mongodb.com/manual/reference/privilege-actions/#dropCollection)
* [find](https://docs.mongodb.com/manual/reference/privilege-actions/#find)
* [insert](https://docs.mongodb.com/manual/reference/privilege-actions/#insert)

Provides the following action on [admin.system.roles](https://docs.mongodb.com/manual/reference/system-collections/#admin.system.roles)

* [createIndex](https://docs.mongodb.com/manual/reference/privilege-actions/#createIndex)

Provides the following actions on [admin.system.users](https://docs.mongodb.com/manual/reference/system-collections/#admin.system.users) and legacy system.users collections:

* [bypassDocumentValidation](https://docs.mongodb.com/manual/reference/privilege-actions/#bypassDocumentValidation)
* [collMod](https://docs.mongodb.com/manual/reference/privilege-actions/#collMod)
* [createCollection](https://docs.mongodb.com/manual/reference/privilege-actions/#createCollection)
* [createIndex](https://docs.mongodb.com/manual/reference/privilege-actions/#createIndex)
* [dropCollection](https://docs.mongodb.com/manual/reference/privilege-actions/#dropCollection)
* [find](https://docs.mongodb.com/manual/reference/privilege-actions/#find)
* [insert](https://docs.mongodb.com/manual/reference/privilege-actions/#insert)
* [remove](https://docs.mongodb.com/manual/reference/privilege-actions/#remove)
* [update](https://docs.mongodb.com/manual/reference/privilege-actions/#update)

Although, restore includes the ability to modify the documents in the [admin.system.users](https://docs.mongodb.com/manual/reference/system-collections/#admin.system.users)collection using normal modification operations, *only* modify these data using the [user management methods](https://docs.mongodb.com/manual/reference/method/#user-management-methods).

All-Database Roles

*Changed in version 3.4.*

The following roles are available only to users on the admin database. These roles provide privileges which apply to all collections except system.\* collections on all databases except local and config:

readAnyDatabase

Provides the same read-only privileges as read on all databases except local and config.readAnyDatabase also provides the [listDatabases](https://docs.mongodb.com/manual/reference/privilege-actions/#listDatabases) privilege action on the cluster.

*Changed in version 3.4:*readAnyDatabase no longer applies to the local and config databases. To provide read privileges on local and config, create a user on the admin database with the readrole on the local and config databases.

See also the clusterManager and clusterMonitor roles for access to the config and localdatabases.

readWriteAnyDatabase

Provides the same read and write privileges as readWrite on all databases except local and config. readWriteAnyDatabase also provides the [listDatabases](https://docs.mongodb.com/manual/reference/privilege-actions/#listDatabases) privilege action on the cluster.

*Changed in version 3.4:*readWriteAnyDatabase no longer applies to the local and configdatabases. To provide read and write privileges on local and config, create a user on the admindatabase with the readWrite role on the local and config databases.

See also the clusterManager and clusterMonitor roles for access to the config and localdatabases.

userAdminAnyDatabase

Provides the same access to user administration operations as userAdmin on all databases except local and config. userAdminAnyDatabase also provides the following privilege actions on the cluster:

* [authSchemaUpgrade](https://docs.mongodb.com/manual/reference/privilege-actions/#authSchemaUpgrade)
* [invalidateUserCache](https://docs.mongodb.com/manual/reference/privilege-actions/#invalidateUserCache)
* [listDatabases](https://docs.mongodb.com/manual/reference/privilege-actions/#listDatabases)

The role also provides the following privilege actions on the system.users and system.rolescollections on the admin database, and on legacy system.users collections from versions of MongoDB prior to 2.6:

* [collStats](https://docs.mongodb.com/manual/reference/privilege-actions/#collStats)
* [dbHash](https://docs.mongodb.com/manual/reference/privilege-actions/#dbHash)
* [dbStats](https://docs.mongodb.com/manual/reference/privilege-actions/#dbStats)
* [find](https://docs.mongodb.com/manual/reference/privilege-actions/#find)
* [killCursors](https://docs.mongodb.com/manual/reference/privilege-actions/#killCursors)
* [planCacheRead](https://docs.mongodb.com/manual/reference/privilege-actions/#planCacheRead)

*Changed in version 2.6.4:*userAdminAnyDatabase added the following privilege actions on the [admin.system.users](https://docs.mongodb.com/manual/reference/system-collections/#admin.system.users) and [admin.system.roles](https://docs.mongodb.com/manual/reference/system-collections/#admin.system.roles) collections:

* [createIndex](https://docs.mongodb.com/manual/reference/privilege-actions/#createIndex)
* [dropIndex](https://docs.mongodb.com/manual/reference/privilege-actions/#dropIndex)

The userAdminAnyDatabase role does not restrict the privileges that a user can grant. As a result, userAdminAnyDatabase users can grant themselves privileges in excess of their current privileges and even can grant themselves *all privileges*, even though the role does not explicitly authorize privileges beyond user administration. This role is effectively a MongoDB system [superuser](https://docs.mongodb.com/manual/reference/built-in-roles/#superuser).

*Changed in version 3.4:*userAdminAnyDatabase no longer applies to the local and configdatabases.

See also the clusterManager and clusterMonitor roles for access to the config and localdatabases.

dbAdminAnyDatabase

Provides the same access to database administration operations as dbAdmin on all databases except local and config. dbAdminAnyDatabase also provides the [listDatabases](https://docs.mongodb.com/manual/reference/privilege-actions/#listDatabases) privilege action on the cluster.

*Changed in version 3.4:*dbAdminAnyDatabase no longer applies to the local and configdatabases. To provide dbAdmin privileges on local and config, create a user on the admin database with the dbAdmin role on the local and config databases.

See also the clusterManager and clusterMonitor roles for access to the config and localdatabases.

Superuser Roles

Several roles provide either indirect or direct system-wide superuser access.

The following roles provide the ability to assign any user any privilege on any database, which means that users with one of these roles can assign *themselves* any privilege on any database:

* dbOwner role, when scoped to the admin database
* userAdmin role, when scoped to the admin database
* userAdminAnyDatabase role

The following role provides full privileges on all resources:

root

Provides access to the operations and all the resources of the readWriteAnyDatabase, dbAdminAnyDatabase, userAdminAnyDatabase, clusterAdmin roles, restore, and backuproles *combined*.

*Changed in version 3.4:*The root role includes privileges from the backup role.

*Changed in version 3.0.7:*The root has [validate](https://docs.mongodb.com/manual/reference/privilege-actions/#validate) action on system. collections. Previously, rootdoes **not** include any access to collections that begin with the system. prefix other than system.indexes and system.namespaces.

The root role includes privileges from the restore role.

Internal Role

\_\_system

MongoDB assigns this role to user objects that represent cluster members, such as replica set members and [mongos](https://docs.mongodb.com/manual/reference/program/mongos/#bin.mongos) instances. The role entitles its holder to take any action against any object in the database.

**Do not** assign this role to user objects representing applications or human administrators, other than in exceptional circumstances.

If you need access to all actions on all resources, for example to run [applyOps](https://docs.mongodb.com/manual/reference/command/applyOps/#dbcmd.applyOps) commands, do not assign this role. Instead, [create a user-defined role](https://docs.mongodb.com/manual/tutorial/manage-users-and-roles/#create-user-defined-role) that grants [anyAction](https://docs.mongodb.com/manual/reference/privilege-actions/#anyAction) on [anyResource](https://docs.mongodb.com/manual/reference/resource-document/#resource-anyresource) and ensure that only the users who need access to these operations have this access.

## Create New User - createUser Command

The createUser command is used to create a new user on the database where you run the command. The createUser command returns an error message if the user already exists.

A database name must have to mention at the time to create a new user with createUser action.

A grantRole action must have to mention on a role’s database to grant the role to another user.

If you have the userAdmin or userAdminAnyDatabase role, or if you are authenticated using the localhost exception, you have those actions.

**Syntax:**

{ createUser: "<name>",

pwd: "<cleartext password>",

customData: { <any information> },

roles: [

{ role: "<role>", db: "<database>" } | "<role>",

...

],

writeConcern: { <write concern> }

}

**Parameters:**

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Description** |
| createUser | string | The name of the new user. |
| pwd | string | The user’s password. |
| customData | document | Optional. |
| roles | array | The roles granted to the user |
| writeConcern | document | Optional. The write concern briefs the guarantee that MongoDB provides at the time of reporting on the success of a write operation. w: 1 as the default write concern. |

Assume that our database name is userdetails and there are the following document:

db.userdetails.insert({"user\_id" : "user1","password" :"1a2b3c" ,

"date\_of\_join" : "16/10/2010" ,"education" :"M.C.A." ,

"profession" : "CONSULTANT","interest" : "MUSIC",

"community\_name" :["MODERN MUSIC", "CLASSICAL MUSIC","WESTERN MUSIC"],

"community\_moder\_id" : ["MR. Alex","MR. Dang","MR Haris"],

"community\_members" : [700,200,1500],

"friends\_id" : ["kumar","harry","anand"],

"ban\_friends\_id" :["Amir","Raja","mont"]});

## Example

The following createUser command creates a user myNewuser on the userdetails database. The command gives myNewuser the clusterAdmin and readAnyDatabase roles on the admin database and the readWrite role on the userdetails database:

db.getSiblingDB("userdetails").runCommand( { createUser: "myNewuser",

pwd: "thisPassword",

customData: { profession : "DOCTOR" },

roles: [

{ role: "clusterAdmin", db: "admin" },

{ role: "readAnyDatabase", db: "admin" },

"readWrite"

],

writeConcern: { interest : "SPORTS" }

} )

## Example: Create more user

The following createUser command creates a user myNewuser1 on the userdetails database. The command gives myNewuser1 the clusterAdmin and the readWrite role on the userdetails database:

db.getSiblingDB("userdetails").runCommand( { createUser: "myNewuser1",

pwd: "thisPassword",

customData: { profession : "DOCTOR" },

roles: [

{ role: "clusterAdmin", db: "admin" },

"readWrite"

],

writeConcern: { interest : "SPORTS" }

} )

If we type wrongly in the above statement myNewuser instead of myNewuser1 then the following error message as an output will appear beacuse the user already exists

{

"ok" : 0,

"errmsg" : "User \"myNewuser@userdetails\" already exists",

"code" : 11000

}

## To see the User Information - usersInfo Command

The command usersInfo returns information about one or more users.

**Syntax:**

{ usersInfo: { user: <name>, db: <db> },

showCredentials: <Boolean>,

showPrivileges: <Boolean>

}

**Parameters:**

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Description** |
| usersInfo | various | The user(s) about whom to return information. |
| showCredentials | Boolean | Optional. If the field set to true the user’s password hash will be displayed, it is false by default. |
| showPrivileges | Boolean | Optional. If the field set to true to show the user’s full set of privileges, including expanded information for the inherited roles, it is false by default. |

## Example to View Specific User

To see information but not the credentials, for the user "myNewuser" defined in "userdetails" database, run the following command:

db.runCommand(

{

usersInfo: { user: "myNewuser", db: "userdetails" },

showPrivileges: false

}

)

Output:

{

"users" : [

{

"\_id" : "userdetails.myNewuser",

"user" : "myNewuser",

"db" : "userdetails",

"customData" : {

"profession" : "DOCTOR"

},

"roles" : [

{

"role" : "clusterAdmin",

"db" : "admin"

},

{

"role" : "readAnyDatabase",

"db" : "admin"

},

{

"role" : "readWrite",

"db" : "userdetails"

}

]

}

],

"ok" : 1

}

## Example to View Multiple User

To view info for several users, use an array, with or without the optional fields showPrivileges and showCredentials. For example:

db.runCommand( { usersInfo: [ { user: "myNewuser", db: "userdetails" },

{ user: "myNewuser1", db: "userdetails" } ],

showPrivileges: false

} );

Output:

{

"users" : [

{

"\_id" : "userdetails.myNewuser",

"user" : "myNewuser",

"db" : "userdetails",

"customData" : {

"profession" : "DOCTOR"

},

"roles" : [

{

"role" : "clusterAdmin",

"db" : "admin"

},

{

"role" : "readAnyDatabase",

"db" : "admin"

},

{

"role" : "readWrite",

"db" : "userdetails"

}

]

},

{

"\_id" : "userdetails.myNewuser1",

"user" : "myNewuser1",

"db" : "userdetails",

"customData" : {

"profession" : "Engineer"

},

"roles" : [

{

"role" : "clusterAdmin",

"db" : "admin"

},

{

"role" : "readWrite",

"db" : "userdetails"

}

]

}

],

"ok" : 1

}

## Example to View All Users for a Database

To view all users on the database the following command can be used:

db.runCommand( { usersInfo: 1 , showCredentials : false} );

Output:

{

"users" : [

{

"\_id" : "userdetails.myNewuser",

"user" : "myNewuser",

"db" : "userdetails",

"customData" : {

"profession" : "DOCTOR"

},

"roles" : [

{

"role" : "clusterAdmin",

"db" : "admin"

},

{

"role" : "readAnyDatabase",

"db" : "admin"

},

{

"role" : "readWrite",

"db" : "userdetails"

}

]

},

{

"\_id" : "userdetails.myNewuser1",

"user" : "myNewuser1",

"db" : "userdetails",

"customData" : {

"profession" : "Engineer"

},

"roles" : [

{

"role" : "clusterAdmin",

"db" : "admin"

},

{

"role" : "readWrite",

"db" : "userdetails"

}

]

}

],

"ok" : 1

}

If showCredential is true the output will be the following:

{

"users" : [

{

"\_id" : "userdetails.myNewuser",

"user" : "myNewuser",

"db" : "userdetails",

"credentials" : {

"MONGODB-CR" : "28fcdbb5d879ccf086bd6404bd06b230"

},

"customData" : {

"profession" : "DOCTOR"

},

"roles" : [

{

"role" : "clusterAdmin",

"db" : "admin"

},

{

"role" : "readAnyDatabase",

"db" : "admin"

},

{

"role" : "readWrite",

"db" : "userdetails"

}

]

},

{

"\_id" : "userdetails.myNewuser1",

"user" : "myNewuser1",

"db" : "userdetails",

"credentials" : {

"MONGODB-CR" : "9b08b3a68eef34d462bdb2e5b2037438"

},

"customData" : {

"profession" : "Engineer"

},

"roles" : [

{

"role" : "clusterAdmin",

"db" : "admin"

},

{

"role" : "readWrite",

"db" : "userdetails"

}

]

}

],

"ok" : 1

}

## To Update the User Information - updateUser Command

The updateUser command updates the profile of the user on the specific database. This command completely replaces the data of the previous field’s values. It is required to specify the updateUser field and at least one other field, other than writeConcern to update a user.

**Syntax:**

{ updateUser: "<username>",

pwd: "<cleartext password&glt;",

customData: { <any information> },

roles: [

{ role: "<role>", db: "<database>" } | "<role>",

...

],

writeConcern: { <write concern> }

}

**Parameters:**

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Description** |
| updateUser | string | The name of the user to update. |
| pwd | string | Optional. The user’s password. |
| customData | document | Optional. Any arbitrary information. |
| roles | array | Optional. The roles granted to the user. An update to the roles array replace the values of the previous array. |
| writeConcern | document | Optional. The write concern briefs the guarantee that MongoDB provides at the time of reporting on the success of a write operation. w: 1 as the default write concern. |

## Example

Here in below a user myNewuser in the userdetails database with the following user info:

{

"users" : [

{

"\_id" : "userdetails.myNewuser",

"user" : "myNewuser",

"db" : "userdetails",

"customData" : {

"profession" : "DOCTOR"

},

"roles" : [

{

"role" : "clusterAdmin",

"db" : "admin"

},

{

"role" : "readAnyDatabase",

"db" : "admin"

},

{

"role" : "readWrite",

"db" : "userdetails"

}

]

}

]

}

The following updateUser command completely replaces the user’s customData and roles data:

use userdetails

db.runCommand( { updateUser : "myNewuser",

customData : { profession : "Classical Music" },

roles : [

{ role : "read", db : "assets" }

]

} )

Here is the output after update the user:

> db.runCommand(

{

usersInfo: { user: "myNewuser", db: "userdetails" },

showPrivileges: false

}

);

{

"users" : [

{

"\_id" : "userdetails.myNewuser",

"user" : "myNewuser",

"db" : "userdetails",

"customData" : {

"profession" : "Classical Music"

},

"roles" : [

{

"role" : "read",

"db" : "assets"

}

]

}

],

"ok" : 1

}

## To Drop the User Information - dropUser Command

The dropUser command is used to remove the user from the concern database.

**Syntax:**

{

dropUser: "<user>",

writeConcern: { <write concern> }

}

**Parameters:**

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Description** |
| dropUser | string | The name of the user to delete. This command will work only the database where the user exists. |
| writeConcern | document | Optional. The write concern briefs the guarantee that MongoDB provides at the time of reporting on the success of a write operation. w: 1 as the default write concern. . |

## Example

If we want to remove the user myNewuser1 from the userdetails database the following sequence of operations in the mongo shell can be used.

> use userdetails

switched to db userdetails

> db.runCommand( { dropUser: "myNewuser1"} );

{ "ok" : 1 }

Now if we want to view all the user for database userdetails the following result will appear where the user myNewuser1 will be excluded.

> db.runCommand( { usersInfo: 1 , showCredentials : false} );

{

"users" : [

{

"\_id" : "userdetails.myNewuser",

"user" : "myNewuser",

"db" : "userdetails",

"customData" : {

"profession" : "Classical Music"

},

"roles" : [

{

"role" : "read",

"db" : "assets"

}

]

}

],

"ok" : 1

}

## To Drop All the Users Information - dropAllUsersFromDatabase Command

This command removes all users from the concern database on which you run the command.

**Syntax:**

{

dropAllUsersFromDatabase: 1,

writeConcern: { <write concern> }

}

**Parameters:**

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Description** |
| dropAllUsersFromDatabase | integer | Specify 1 to drop all the users from the current database. |
| writeConcern | document | Optional. The write concern briefs the guarantee that MongoDB provides at the time of reporting on the success of a write operation. w: 1 as the default write concern. |

## Example

If we want to remove the user myNewuser1 from the userdetails database the following sequence of operations in the mongo shell can be used.

> use userdetails;

switched to db userdetails

> db.runCommand( { dropAllUsersFromDatabase: 1, writeConcern: { w: "majority" }});

Here is the output

{ "n" : 1, "ok" : 1 }

It means that the n field in the results document shows the number of users removed, here 1 user have been removed.

Now if we want to view the information of the user for database userdetails the following result will appear.

> db.runCommand( { usersInfo: 1 , showCredentials : false} );

{ "users" : [ ], "ok" : 1 }

## To Grants additional roles to a user - grantRolesToUser Command

This command grants additional roles to a user.

**Syntax:**

{ grantRolesToUser: "<user>",

roles: [ <roles> ],

writeConcern: { <write concern> }

}

**Parameters:**

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Description** |
| grantRolesToUser | string | The name of the user to give additional roles. |
| roles | array | An array of additional roles to grant to the user. |
| writeConcern | document | Optional. The write concern briefs the guarantee that MongoDB provides at the time of reporting on the success of a write operation. w: 1 as the default write concern. |

## Example

Given a user myNewuser2 in the userdetails database with the following roles:

{

"\_id" : "userdetails.myNewuser2",

"user" : "myNewuser2",

"db" : "userdetails",

"customData" : {

"profession" : "painter"

},

"roles" : [

{

"role" : "read",

"db" : "usertransact"

},

{

"role" : "readWrite",

"db" : "userdetails"

}

]

}

The following grantRolesToUser operation gives myNewuser2 the read role on the usertransact database and the readWrite role on the userdetails database.

{

"\_id" : "userdetails.myNewuser2",

"user" : "myNewuser2",

"db" : "userdetails",

"customData" : {

"profession" : "painter"

},

"roles" : [

{

"role" : "read",

"db" : "usertransact"

},

{

"role" : "read",

"db" : "useraccount"

},

{

"role" : "readWrite",

"db" : "userdetails"

}

]

};

## To Revoke Roles From a User - revokeRolesFromUser Command

This command is used to remove a one or more roles from a user on the database where the roles exist.

**Syntax:**

{ revokeRolesFromUser: "<user>",

roles: [

{ role: "<role>", db: "<database>" } | "<role>",

...

],

writeConcern: { <write concern> }

}

**Parameters:**

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Description** |
| revokeRolesFromUser | string | The user to remove roles from. |
| roles | array | The roles to revoke from the user. |
| writeConcern | document | Optional. The write concern briefs the guarantee that MongoDB provides at the time of reporting on the success of a write operation. w: 1 as the default write concern. |

## Example

Given a user myNewuser2 in the userdetails database with the following roles:

{

"\_id" : "userdetails.myNewuser2",

"user" : "myNewuser2",

"db" : "userdetails",

"customData" : {

"profession" : "painter"

},

"roles" : [

{

"role" : "read",

"db" : "usertransact"

},

{

"role" : "read",

"db" : "useraccount"

},

{

"role" : "readWrite",

"db" : "userdetails"

}

]

};

To remove the two of the user’s roles such as the read role on the usertransact database and the readWrite role on the userdetails database the following sequence of commands can be used.

use userdetails;

db.runCommand( { revokeRolesFromUser: "myNewuser2",

roles: [

{ role: "read", db: "usertransact" },

"readWrite"

],

writeConcern: { w: "majority" }

} );

The user myNewuser2 in the userdetails database now has only one remaining role:

{

"\_id" : "userdetails.myNewuser2",

"user" : "myNewuser2",

"db" : "userdetails",

"customData" : {

"profession" : "painter"

},

"roles" : [

{

"role" : "read",

"db" : "useraccount"

}

]

}