Simple Usage `mongoimport` restores a database from a backup taken with `mongoexport`. Most of the arguments to `mongoexport` also exist for `mongoimport`.

In the following example, `mongoimport` imports the data in the *JSON* data from the `contacts.json` file into the collection `contacts` in the `users` database.

```
mongoimport --db users --collection contacts --file contacts.json
```

Import JSON to Remote Host Running with Authentication In the following example, `mongoimport` imports data from the file `/opt/backups/mdb1-examplenet.json` into the `contacts` collection within the database `marketing` on a remote MongoDB database with authentication enabled.

`mongoimport` connects to the `mongod` instance running on the host `mongodb1.example.net` over port `37017`. It authenticates with the username `user` and the password `pass`.

```
mongoimport --host mongodb1.example.net --port 37017 --username user --password pass --collection co
```

CSV Import In the following example, `mongoimport` imports the *csv* formatted data in the `/opt/backups/contacts.csv` file into the collection `contacts` in the `users` database on the MongoDB instance running on the localhost port numbered `27017`.

Specifying `--headerline` instructs `mongoimport` to determine the name of the fields using the first line in the CSV file.

```
mongoimport --db users --collection contacts --type csv --headerline --file /opt/backups/contacts.csv
```

`mongoimport` uses the input file name, without the extension, as the collection name if `-c` or `--collection` is unspecified. The following example is therefore equivalent:

```
mongoimport --db users --type csv --headerline --file /opt/backups/contacts.csv
```

Use the "`--ignoreBlanks`" option to ignore blank fields. For *CSV* and *TSV* imports, this option provides the desired functionality in most cases because it avoids inserting fields with null values into your collection.

Additional Resources

- Backup and its Role in Disaster Recovery White Paper[101]
- Cloud Backup through MongoDB Cloud Manager[102]
- Blog Post: Backup vs. Replication, Why you Need Both[103]
- Backup Service with Ops Manager, an on-premise solution available in MongoDB Enterprise Advanced[104]

Backup and Restore Sharded Clusters

The following tutorials describe backup and restoration for sharded clusters:

Backup a Small Sharded Cluster with mongodump **(page 255)** If your *sharded cluster* holds a small data set, you can use `mongodump` to capture the entire backup in a reasonable amount of time.

[101] https://www.mongodb.com/lp/white-paper/backup-disaster-recovery?jmp=docs
[102] https://cloud.mongodb.com/?jmp=docs
[103] http://www.mongodb.com/blog/post/backup-vs-replication-why-do-you-need-both?jmp=docs
[104] https://www.mongodb.com/products/mongodb-enterprise-advanced?jmp=docs