Upgrading a database to RediSearch 2.x

RediSearch 2.x includes some significant changes to the architecture of RediSearch that provide improved functionality. The main architectural change is that the indexes are stored outside of the Redis database that contains the data. This allows for improvements in command efficiency and improvements in replication between clusters because the index changes are managed by the participating clusters rather than being synchronized with the data.

This change allows databases with RediSearch to support:

- **Active-Active databases**
- Database cluster re-sharding
- Replica Of to a sharded destination database
- **EXPIRE** of documents reflected in the index

In addition, RediSearch 2.x indexes data that already existed in the database at the time that the index was created.

To upgrade a Redis Enterprise Software (RS) database with RediSearch 1.x to RediSearch 2.x, you have to set up a new database with RediSearch 2.x and use the `RediSearch_Syncer.py` script to replicate the data from the old database into the new database.

**Prerequisites**

Make sure that you have Python 3 (`sudo apt install python3`) installed on the host where you will run the synchronization script.

**Limitations**

1. Suggestions (FT.SUG APIs) and spell check dictionaries are not replicated from the source database. You need to add them manually.

2. If there are multiple indexes on the source and the documents do not have prefixes that identify them with an index, RediSearch 2.x can’t index the documents in their respective indexes.

3. The NOSAVE option is no longer supported. Indexes created with the NOSAVE option can’t be upgraded.

4. Databases that contain temporary indexes can’t be upgraded.

5. Any attempt to add, delete, or modify an index during the upgrade will cause the replication to fail. During the upgrade, the source database can only receive `ft.add` and `ft.del` commands.

**Replicating data from RediSearch 1.x to RediSearch 2.x:**

To replicate a RediSearch 1.x database to a RediSearch 2.x database:

1. Log in to the admin console of the RS cluster that you want to host the new database with RediSearch 2.x.

2. Add the RediSearch 2.x module to the cluster:
1. Go to the Redis Download Center and download the RediSearch 2.x module package.

2. In the Redis Enterprise admin console, go to the: settings

3. In redis modules, click Add Module.

4. Browse to the module package and upload it to the cluster.

3. Create a new database with RediSearch 2.x:
   1. Create a database and configure the database settings.
   2. In the Redis Modules section, click 🛠️, select RediSearch 2, and click 📐.
   3. Click Activate to create the database.

4. Migrate the data to the RediSearch 2.x database:
   1. Download the RediSearch_Syncer package for your operating system.
      - RedHat Enterprise Linux 7
      - Ubuntu 18.04
      - Ubuntu 16.04
      - Ubuntu 14.04
   2. Extract the package with: `unzip <package_name>`
   3. In the extracted directory, run the RediSearch_Syncer.py script:
      ```
      python3 RediSearch_Syncer.py -d <destination_url> -s <source_url> [-l] [--add-prefix <prefix>]
      ```
   
   Where:
   - destination url - The replication URL of the RediSearch 2.x database that you see when you click on Get Replica of source URL in the database configuration in the admin console.
   - source url - The replication URL of the RediSearch 1.x database that you see when you click on Get Replica of source URL in the database configuration in the admin console.
- `--add-prefix <prefix>` (optional) - Adds a prefix to all of the hashes that are replicated to the new database.

| Note: The add-prefix option is only recommended when you want to index all of the hashes in the same index in the source database. |

The script shows a table with the progress of the replication. Press **F5** to see the updated status of the replication process.

For example:

```python
python RediSearch_Syncer.py -d redis://admin:IBrS0xl6rShfB1wKTtUkcQG1g3UWA1Td53AotPdTcQvGIVP@redis-19472.cluster1.local:19472 -s redis://admin:1GjFuUbBqTSPDbrfaxEPLSoXpFmRdFxm8KMD0BuxwMJ2DEaO@redis-19636.cluster1.local:19636
```

4. Stop the processes that are sending requests to the source database so all of the data gets synchronized to the destination database.

5. Verify that the replication is complete by comparing the number of indexed documents on FT.INFO on both source and destination databases until the number of indexed documents is the same in the source and destination databases.

6. When the status field is `st_in_sync` then you can press **Ctrl-C** to cancel the synchronization process.

7. Press **Q** to quit the `RediSearch_Syncer.py`.

You can now redirect your database connections to the database with RediSearch 2.x.

**Updated:** August 10, 2021