Overview of the REST Music API

1. Creating a keyspace that is a collection of all the tables
    a. POST /keyspaces/{keyspacename}
    b. Input Json body with details about replication factor

2. Dropping a keyspace
    a. DELETE /keyspaces/<keyspacename>

3. Creating a table within a keyspace
    a. POST /keyspaces/<keyspacename>/tables/<tablename>
    b. Input Json body with details about column names and their types

4. Inserting a row into a table
    a. PUT /keyspaces/<keyspacename>/tables/<tablename>/rows
    b. Input Json body with row details

5. Update value in a table
    a. PUT /keyspaces/<keyspacename>/tables/<tablename>/rows?<rowspec>
    b. Input Json body with values to be updated

6. Delete values from a table
    a. DELETE /keyspaces/<keyspacename>/tables/<tablename>/rows?<rowspec>
    b. Input Json body with columns whose value needs to be deleted

7. Drop a table from a keyspace
    a. DELETE /keyspaces/<keyspacename>/tables/<tablename>

8. Select specific row from a table
    a. GET /keyspaces/<keyspacename>/tables/<tablename>/rows?<rowspec>
    b. Return value is a json body with column names and values of that row

9. Select all rows from a table
    a. GET /keyspaces/<keyspacename>/tables/<tablename>/rows
    b. Return value is a json body with column names and values of all rows

10. Creating a distributed lock
    a. POST /locks/create/<lockname>
    b. Return value is a unique lockid for the requester

11. Acquiring a distributed lock
    a. GET /locks/acquire/<lockid>
    b. Return value is true if the lockid is the current holder of the lock and false if not

12. Enquiring about a distributed lock
    a. GET /locks/enquire/<lockname>
    b. Return value is the lockid of the current lock holder

13. Unlocking a distributed lock
    a. DELETE /locks/release/<lockid>

14. Deleting a distributed lock
    a. DELETE /locks/delete/<lockName>

Some notes:

- All the data related queries that write to the Music data store have an option in the Json file where they need to specify consistency type as follows (as seen in the Json file):

```
consistencyInfo": {
        "type": "atomic",
        "lockId": "$TestKS.projects$x-94608776321630235-0000000000"
}
```

  If this type is specified as “atomic” then the lock id too has to be sent along so that Music can verify if the lock holder is performing the write. If the type is specified as “eventual” then no lock field is required.
- The select query returns the following Json with row 1, row 2, for each row etc. :

```
{
  "row 0": {
    "address": "{}",
    "emp_name": "joe",
    "emp_id": "abc66ccc-d857-4e90-b1e5-df98a3d40cd6",
    "emp_salary": "30"
  }
}
```