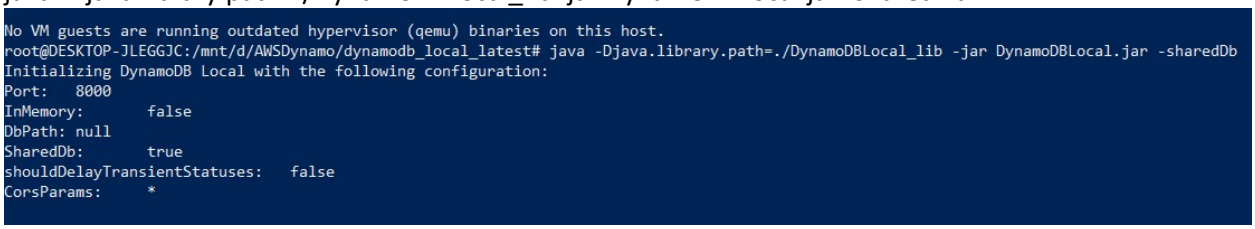
Pre-requisite

1. For Windows, WSL Ubuntu Installation
2. Java JRE install (sudo apt install default-jre)

Getting Started

Note: **--endpoint-url http://localhost:8000** is very important to include in CLI to execute locally but not important if CLI will be executed with Access Key ID and Secret Access Key from AWS Subscription
Note: The commands below are ubuntu bash based shell.

1. In one window, execute the below line to initiate the Dynamo DB Locally
   java -Djava.library.path=./DynamoDBLocal_lib -jar DynamoDBLocal.jar -sharedDb



2. Create a table
   Sample Script:

```
aws dynamodb create-table \
  --table-name Music \
  --attribute-definitions \
     AttributeName=Artist,AttributeType=S \
     AttributeName=SongTitle,AttributeType=S \
  --key-schema AttributeName=Artist,KeyType=HASH
AttributeName=SongTitle,KeyType=RANGE \
  --provisioned-throughput ReadCapacityUnits=1,WriteCapacityUnits=1 \
  --table-class STANDARD \
        --endpoint-url http://localhost:8000
```

```
root@DESKTOP-JLEGGJC:~# aws dynamodb create-table \
>     --table-name Music \
>     --attribute-definitions \
>         AttributeName=Artist,AttributeType=S \
>         AttributeName=SongTitle,AttributeType=S \
>     --key-schema AttributeName=Artist,KeyType=HASH AttributeName=SongTitle,KeyType=RANGE \
rovision>     --provisioned-throughput ReadCapacityUnits=1,WriteCapacityUnits=1 \
>     --table-class STANDARD \
ndpoint-> --endpoint-url http://localhost:8000
{
    "TableDescription": {
        "AttributeDefinitions": [
            {
                "AttributeName": "Artist",
                "AttributeType": "S"
            },
            {
                "AttributeName": "SongTitle",
                "AttributeType": "S"
            }
        ],
        "TableName": "Music",
        "KeySchema": [
            {
                "AttributeName": "Artist",
                "KeyType": "HASH"
            },
            {
                "AttributeName": "SongTitle",
                "KeyType": "RANGE"
            }
        ],
        "TableStatus": "ACTIVE",
        "CreationDateTime": 1660066172.404,
        "ProvisionedThroughput": {
            "LastIncreaseDateTime": 0.0,
            "LastDecreaseDateTime": 0.0,
            "NumberOfDecreasesToday": 0,
            "ReadCapacityUnits": 1,
            "WriteCapacityUnits": 1
        },
        "TableSizeBytes": 0,
        "ItemCount": 0,
        "TableArn": "arn:aws:dynamodb:ddblocal:000000000000:table/Music"
    }
}
root@DESKTOP-JLEGGJC:~#
```

3. Get the table status
   Script: **aws dynamodb describe-table --table-name Music --endpoint-url http://localhost:8000 | grep TableStatus**

```
root@DESKTOP-JLEGGJC:~# aws dynamodb describe-table --table-name Music --endpoint-url http://localhost:8000 | grep TableStatus
        "TableStatus": "ACTIVE",
root@DESKTOP-JLEGGJC:~#
```

4. Get list of DynamoDB tables
   Script: **aws dynamodb list-tables --endpoint-url http://localhost:8000**

```
root@DESKTOP-JLEGGJC:~# aws dynamodb list-tables --endpoint-url http://localhost:8000
{
    "TableNames": [
        "Music"
    ]
}
root@DESKTOP-JLEGGJC:~#
```

   (base) C:\Users\User>
5. Insert records
   Sample:

**aws dynamodb put-item **
**--table-name Music **
**--item **
**'{"Artist": {"S": "No One You Know"}, "SongTitle": {"S": "Call Me Today"}, "AlbumTitle": {"S": "Somewhat Famous"}}' **
**--return-consumed-capacity TOTAL **
--endpoint-url http://localhost:8000

```
root@DESKTOP-JLEGGJC:~# aws dynamodb put-item \
    --ta>     --table-name Music \
>     --item \
>         '{"Artist": {"S": "No One You Know"}, "SongTitle": {"S": "Call Me Today"}, "AlbumTitle": {"S": "Somewhat Famous"}}' \
>     --return-consumed-capacity TOTAL  \
> --endpoint-url http://localhost:8000
{
    "ConsumedCapacity": {
        "TableName": "Music",
        "CapacityUnits": 1.0
    }
}
root@DESKTOP-JLEGGJC:~#
```

Another sample insert:

**aws dynamodb put-item **
**--table-name Music **
**--item '{"Artist": {"S": "Mighty Band"}, "SongTitle": {"S": "Stronger"}, "AlbumTitle": {"S": "Life Lessons"}}' **
**--return-consumed-capacity TOTAL **
--endpoint-url http://localhost:8000

```
root@DESKTOP-JLEGGJC:~# aws dynamodb put-item \
>     --table-name Music \
>     --item '{"Artist": {"S": "Mighty Band"}, "SongTitle": {"S": "Stronger"}, "AlbumTitle": {"S": "Life Lessons"}}' \
>     --return-consumed-capacity TOTAL \
> --endpoint-url http://localhost:8000
{
    "ConsumedCapacity": {
        "TableName": "Music",
        "CapacityUnits": 1.0
    }
}
```

Sample PartiQL:

**aws dynamodb execute-statement --statement "INSERT INTO Music **
**VALUE **

**{'Artist':'Michael Learns to Sing','SongTitle':'Sing with me', 'AlbumTitle':'I love to Sing', 'Awards':'12'}" **

--endpoint-url http://localhost:8000

```
root@DESKTOP-JLEGGJC:~# aws dynamodb execute-statement --statement "INSERT INTO Music  \
>               VALUE  \
>               {'Artist':'Michael Learns to Sing','SongTitle':'Sing with me', 'AlbumTitle':'I love to Sing', 'Awards':'12'}" \
>                       --endpoint-url http://localhost:8000
{
    "Items": []
}
root@DESKTOP-JLEGGJC:~#
```

6. Retrieve Records

Sample by CLI:

**aws dynamodb get-item --consistent-read **

**--table-name Music **

**--key '{ "Artist": {"S": "Mighty Band"}, "SongTitle": {"S": "Stronger"}}' **

--endpoint-url http://localhost:8000

```
root@DESKTOP-JLEGGJC:~# aws dynamodb get-item --consistent-read \
>       --table-name Music \
> --key '{ "Artist": {"S": "Mighty Band"}, "SongTitle": {"S": "Stronger"}}' \
> --endpoint-url http://localhost:8000
{
    "Item": {
        "Artist": {
            "S": "Mighty Band"
        },
        "SongTitle": {
            "S": "Stronger"
        },
        "AlbumTitle": {
            "S": "Life Lessons"
        }
    }
}
root@DESKTOP-JLEGGJC:~#
```

Sample by PartiQL:

**aws dynamodb execute-statement --statement "SELECT * FROM Music **

**WHERE Artist='Mighty Band' AND SongTitle='Stronger'" **

--endpoint-url http://localhost:8000

```
root@DESKTOP-JLEGGJC:~# aws dynamodb execute-statement --statement "SELECT * FROM Music   \
>                                         WHERE Artist='Mighty Band' AND SongTitle='Stronger'" \
>                               --endpoint-url http://localhost:8000
{
    "Items": [
        {
            "Artist": {
                "S": "Mighty Band"
            },
            "SongTitle": {
                "S": "Stronger"
            },
            "AlbumTitle": {
                "S": "Life Lessons"
            }
        }
    ]
}
```

7. Update Records

Sample:

```
aws dynamodb update-item \
   --table-name Music \
   --key '{ "Artist": {"S": "Mighty Band"}, "SongTitle": {"S": "Stronger"}}' \
   --update-expression "SET AlbumTitle = :newval" \
   --expression-attribute-values '{":newval":{"S":"A Beautiful Life"}}' \
   --return-values ALL_NEW \
         --endpoint-url http://localhost:8000
```

```
root@DESKTOP-JLEGGJC:~# aws dynamodb update-item \
-table-name Mus>     --table-name Music \
>     --key '{ "Artist": {"S": "Mighty Band"}, "SongTitle": {"S": "Stronger"}}' \
>     --update-expression "SET AlbumTitle = :newval" \
>     --expression-attribute-values '{":newval":{"S":"A Beautiful Life"}}' \
>     --return-values ALL_NEW \
> --endpoint-url http://localhost:8000
{
    "Attributes": {
        "Artist": {
            "S": "Mighty Band"
        },
        "AlbumTitle": {
            "S": "A Beautiful Life"
        },
        "SongTitle": {
            "S": "Stronger"
        }
    }
}
root@DESKTOP-JLEGGJC:~#
```

Sample PartiQL:

**aws dynamodb execute-statement --statement "UPDATE Music **
**SET AlbumTitle='It''s My Life' **

**WHERE Artist='Mighty Band' AND SongTitle='Stronger' **
**RETURNING ALL NEW *" **
--endpoint-url http://localhost:8000

```
root@DESKTOP-JLEGGJC:~# aws dynamodb execute-statement --statement "UPDATE Music  \
>                                         SET AlbumTitle='It''s My Life'  \
>                                         WHERE Artist='Mighty Band' AND SongTitle='Stronger' \
>                                         RETURNING ALL NEW *" \
>                                 --endpoint-url http://localhost:8000
{
    "Items": [
        {
            "Artist": {
                "S": "Mighty Band"
            },
            "AlbumTitle": {
                "S": "It's My Life"
            },
            "SongTitle": {
                "S": "Stronger"
            }
        }
    ]
}
root@DESKTOP-JLEGGJC:~#
```

8. Query Records

Sample:

**aws dynamodb query **
**--table-name Music **
**--key-condition-expression "Artist = :name" **
**--expression-attribute-values '{":name":{"S":"Mighty Band"}}' **
--endpoint-url http://localhost:8000

```
root@DESKTOP-JLEGGJC:~# aws dynamodb query \
>     --table-name Music \
>     --key-condition-expression "Artist = :name" \
>     --expression-attribute-values  '{":name":{"S":"Mighty Band"}}' \
>     --endpoint-url http://localhost:8000
{
    "Items": [
        {
            "Artist": {
                "S": "Mighty Band"
            },
            "SongTitle": {
                "S": "Stronger"
            },
            "AlbumTitle": {
                "S": "It's My Life"
            }
        }
    ],
    "Count": 1,
    "ScannedCount": 1,
    "ConsumedCapacity": null
}
root@DESKTOP-JLEGGJC:~#
```

Sample PartiQL:

**aws dynamodb execute-statement --statement "SELECT * FROM Music **

**WHERE Artist='Mighty Band'" **
--endpoint-url http://localhost:8000

```
root@DESKTOP-JLEGGJC:~# aws dynamodb execute-statement --statement "SELECT * FROM Music   \
>                                       WHERE Artist='Mighty Band'" \
>                                 --endpoint-url http://localhost:8000
{
    "Items": [
        {
            "Artist": {
                "S": "Mighty Band"
            },
            "SongTitle": {
                "S": "Stronger"
            },
            "AlbumTitle": {
                "S": "It's My Life"
            }
        }
    ]
}
root@DESKTOP-JLEGGJC:~#
```

Import Table Data

Sample import script:
aws dynamodb batch-write-item --request-items
file:///mnt/d/AWSDynamo/sampledata/Customized/Music.json --endpoint-url http://localhost:8000

```
root@DESKTOP-JLEGGJC:/mnt/d/AWSDynamo# aws dynamodb batch-write-item --request-items file:///mnt/d/AWSDynamo/sampledata/Customized/Music.json --endpoint-url http://localhost:8000
{
    "UnprocessedItems": {}
}
root@DESKTOP-JLEGGJC:/mnt/d/AWSDynamo#
```

Sample query to check if there are records inserted inserted:
**aws dynamodb query **
**--table-name Music **
**--key-condition-expression "Artist = :name" **
**--expression-attribute-values '{":name":{"S":"James Band"}}' **
--endpoint-url http://localhost:8000

CLI Output:

```
root@DESKTOP-JLEGGJC:/mnt/d/AWSDynamo# aws dynamodb query \
 --table>    --table-name Music \
>    --key-condition-expression "Artist = :name" \
>    --expression-attribute-values  '{":name":{"S":"James Band"}}' \
>    --endpoint-url http://localhost:8000
{
   "Items": [
      {
         "Artist": {
            "S": "James Band"
         },
```

```
    "Awards": {
      "S": "5"
    },
    "AlbumTitle": {
      "S": "Spy Life"
    },
    "SongTitle": {
      "S": "Dangerous"
    },
    "SongNo": {
      "S": "5"
    }
  },
  {
    "Artist": {
      "S": "James Band"
    },
    "AlbumTitle": {
      "S": "Spy Life"
    },
    "SongTitle": {
      "S": "I'll be Stronger"
    },
    "SongNo": {
      "S": "1"
    }
  },
  {
    "Artist": {
      "S": "James Band"
    },
    "AlbumTitle": {
      "S": "Spy Life"
    },
    "SongTitle": {
      "S": "Killer"
    },
    "SongNo": {
      "S": "6"
    }
  },
  {
    "Artist": {
      "S": "James Band"
    },
    "Awards": {
      "S": "1"
    },
```

```
        "AlbumTitle": {
          "S": "Spy Life"
        },
        "SongTitle": {
          "S": "Let's Play"
        },
        "SongNo": {
          "S": "2"
        }
      },
      {
        "Artist": {
          "S": "James Band"
        },
        "Awards": {
          "S": "5"
        },
        "AlbumTitle": {
          "S": "Spy Life"
        },
        "SongTitle": {
          "S": "Money Money"
        },
        "SongNo": {
          "S": "3"
        }
      },
      {
        "Artist": {
          "S": "James Band"
        },
        "AlbumTitle": {
          "S": "Spy Life"
        },
        "SongTitle": {
          "S": "Spy Girls"
        },
        "SongNo": {
          "S": "4"
        }
      }
    ],
    "Count": 6,
    "ScannedCount": 6,
    "ConsumedCapacity": null
}
```