Apica
https://www.apica.io

Apica Data Repository

Last Revised: Dec 07, 2021

Table of Contents

1 Introduction

Apica Data Repository (ADR) a.k.a Apica Table Server (ATS) will help to accomplish the data sharing across the scripts during the execution, so eliminating the need to attach data to the script prior to the execution to manage data. The ADR tool provide shared data tables that can be accessed by multiple scripts during the execution to manage data (CRUD operations). Matches the functionality up and above for the “HP Virtual Table server “.

Apica Data Repository can be used in any use cases where the script execution requires dynamic data, like feeding script with credentials, feeding script with security tokens, sharing data across different micro services scripts in a same application and so on. So basically, by using ADR tool, customer can save time, so it basically apt for cost saving and flexibility.

Currently the delivery model is installing the ADR in the private infrastructure of the customer or a private installation by Apica to customers.

1.1 Release Notes

1.4 version – Release date Dec 02, 2021

- Releasing as a standalone package (before it was delivered along with ZebraTester)
- Tables are saved as encrypted files, means not as plain CSV files
- Supports Search option by a keyword on the created tables

1.3 version – Release date April 22, 2021

- Supports client certificate authentication when communicating to ATS API end points. (Refer section 1.2 for more details)
- Supports new API end point for query table based on column names and values like below example

GET url - http://127.0.0.1:8992?cmd=ats_retrieve_row_by_column
body JSON- {
"TableName”: "table_name",
"MatchColumnNames": ["column_name1",”column_name2”],
"MatchColumnValues" : ["column_value1","column_value2"]
}
This API should also support retrieval based on multiple column names and values. This API also supports an optional input **OuputColumnNames**, the column names in a row to be retrieved based on the matching column names and values.

- Supports the user case of not removing the retrieved column value from the table when querying based on the row index.

1.2 version – Release date January 17, 2021

- Authentication using the username and password, Authorization by the help of Users- Groups - Tables relations.
- Create New Table option, so that user can create new tables that will be automatically added to the groups, the user belongs to.
- Fixed bugs related to intermittent issues - duplicate retrieval and insertion of data.

1.1 version – Release date September 30, 2020

- Support for Column actions like insert value at the start, append value to the end, insert a column and delete a column.
- Support for Row actions like edit a row, means modify the values of the columns inside a row and delete a row.
- Support for server-side pagination for the ATS tables, so that table with huge amount of data will load in milliseconds.
- Enable /Disable API Access, if disabled the APIs supported by ATS won’t be reachable.
- Added a new API call for getting the size of a table on ATS.

1.0 version – Release date July 20, 2020

- Import the CSV files, Contents of the CSV files will be stored as a table and can viewed in the ATS GUI.
- ATS support more than one table and there is a drop down available for the selection of a particular table.
- Export feature is available for the selected table.
- Deletion of the selected table is also possible.
- ATS also have REST endpoints available for the retrieval and feeding of data.

1.2 Installation of ADR a.k.a ATS

ADR or ATS can be deployed by the following the procedure mentioned here https://apica-kb.atlassian.net/wiki/spaces/DAZT/pages/953876543/Installing+the+Apica+Data+Repository+a.k.a+Apica+Table+Server+ATS

In Windows after the installation of the ADR kit, if you want to install ADR as service, there are 2 bat files available for the installation and uninstallation of ADR. **InstallADRServiceWin64.bat** and **UninstallADRServiceWin64.bat**

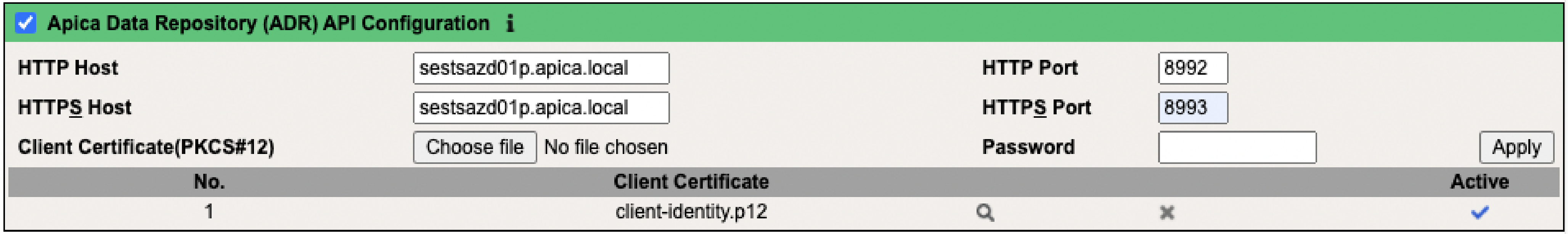
For manual installation on Linux, follow the below procedure

Note that a Adopt Open JDK 11 must already be pre-installed.

1. Manually create the installation directory **/usr/local/ADR (**you may also choose any other installation directory)
2. Copy the files **ADR.jar**, **iaik_jce_full.jar**, **iaik_ssl.jar**, **iaik_eccelerate.jar**, **iaik_eccelerate_ssl.jar** to this directory (you can copy these files from a Windows or a Mac OS X installation of ADR. These files can be used for all operating systems).
3. Set the Java CLASSPATH environment variable to include the installation directory, the default directory ("."), and the path to all *.jar files.

   Example: export CLASSPATH==$CLASSPATH:**ADR.jar: iaik_jce_full.jar: iaik_ssl.jar:iaik_eccelerate.jar:iaik_eccelerate_ssl.jar**

4. Start ADR with the following command: **java –Xmx4096m io.apica.tableserver.TableServerController**.
5. Start a web browser and enter http://127.0.0.1:8990/**.**
6. After starting the ADR, before doing the test from ZT via supported inline functions, you need to configure the ADR API server details in ZT personal settings like below



☑ **Apica Data Repository (ADR) API Configuration** ℹ

HTTP Host	sestsazd01p.apica.local	**HTTP Port**	8992	
HTTPS Host	sestsazd01p.apica.local	**HTTPS Port**	8993	
Client Certificate(PKCS#12)	No file chosen	**Password**		

No.	Client Certificate			Active
1	client-identity.p12	🔍	✖	✓

From 1.3 version ADR supports client certificate authentication when communicate to is API end points. To achieve this, you need to follow the below instructions.

- Pass the parameters -*serverCertspath <path where servers cert, private key, intermediate cert, and root cert exists>* when starting ADR process
- Upload the client certificates in ZT Apica Data Repository API configuration (refer the above screenshot)

ADR allows plain TCP connections, but From Apica we always recommend communication over TLS using mTLS authentication between ZT agent and ADR when executing script. Also, if https host and port are configured, then it will have the precedence.

2 REST End points

1. Retrieve column values from a table

```
GET url - http://127.0.0.1:8992?cmd=ats_retrieve_row
body JSON- {
    "TableName”: "table_name",
    "ColumnNames”: ["column_name1","column_name2"]
}
```

2. Set values of multiple columns in a table

```
POST url - http://127.0.0.1:8992?cmd=ats_send_row
body JSON - {
    "TableName”: "table_name",
    "ColumnNames”: ["column_name1","column_name2"],
    "ColumnValues”: ["column_value1","column_value2"],
    "Type”: "ROW"                                          Type have three values ROW, STACKED and UNIQUE
}
```

3. Retrieve a column value from a table

```
GET url - http://127.0.0.1:8992?cmd=ats_retrieve_column
body JSON- {
    "TableName”: "table_name",
    "ColumnName”: "column_name"
}
```

*Also, there is an optional parameter available for the **ats_retrieve_column,** that is **RowIndex,** this is for retrieving a column value by row index.*

4. Set a column value in a table

```
POST url - http://127.0.0.1:8992?cmd=ats_send_column
body JSON- {
    "TableName”: "table_name",
    "ColumnName”: "column_name",
    "ColumnValue”: "column_value",
    "Unique”: true
}
```

5. Update a column value by row index in a table

```
POST url - http://127.0.0.1:8992?cmd=ats_update_column
    body JSON - {
    "TableName”: "table_name",
    "ColumnName”: "column_name",
    "ColumnValue”: "column_value”,
  “Unique”: false,
    " RowIndex “: 1
}
```

6. Get the size of a table or column in a table

```
     POST url - http://127.0.0.1:8992?cmd= ats_retrieve_table_size
    body JSON - {
    "TableName”: "table_name",
    "ColumnName”: "column_name"                         ColumnName is optional
    }
```

7. Retrieval of rows based on multiple column names and values. This API also supports an optional input OuputColumnNames, the column names in rows to be retrieved based on the matching column names and values

```
     GET url - http://127.0.0.1:8992?cmd=ats_retrieve_row_by_column
     body JSON- {
     "TableName”: "table_name",
     "MatchColumnNames": ["column_name1",”column_name2”],
     "MatchColumnValues" : ["column_value1","column_value2"]
     }
```

3 ADR a.k.a ATS Authentication and authorization

Proceed as follows:

1. Run CreateSuperAdmin.bat or CreateSuperAdmin.sh file from the ADR installation directory:
2. Create a super user by calling the menu option 7 (add account)

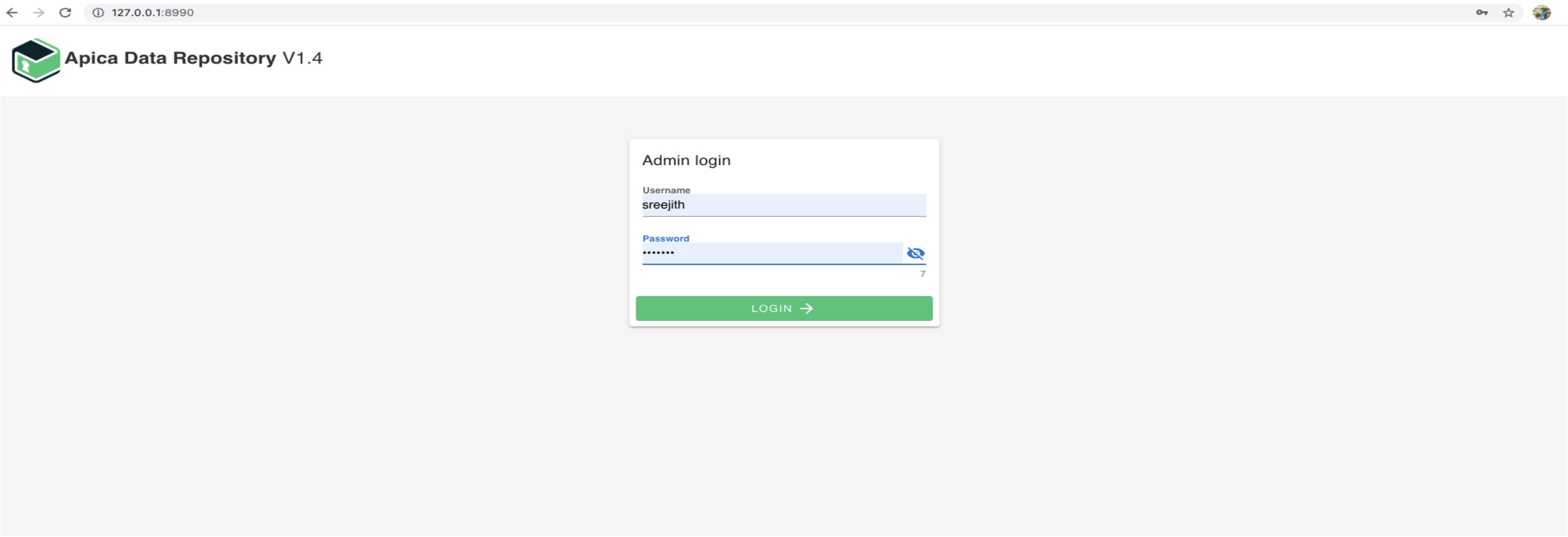
```
1 = list all accounts
2 = reset password
3 = enable account
4 = disable account
5 = set superuser
6 = remove superuser
7 = add account
8 = delete account

0 = exit

select =
```

Enter now 7 and create a new super user account. Then exit this utility.

After created a super user account you can login into ADR like below



127.0.0.1:8990

Apica Data Repository V1.4

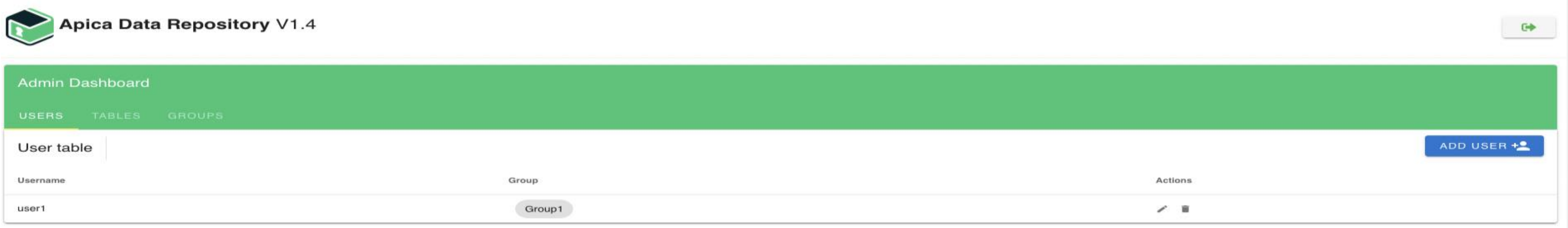
Admin login

Username
sreejith

Password
•••••••

7

After the successful login, you can manage users, groups, users - groups relations and groups - tables relation using following GUI



Apica Data Repository V1.4

Admin Dashboard

User table

Username	Group	Actions
user1	Group1	

After a creating a user, the created user can login to ADR by providing the username and the default password “password”. User can change the password by the help of the change password screen.