A Tutorial on Data Virtualization within IBM Cloud Private for Data

Sanjit Chakraborty

Table of Contents

Introduction

In this tutorial, you will learn how to use Data Virtualization (DV) add-on component within the IBM® Cloud Private for Data (ICP for Data) to integrates data sources across multiple types and locations and turns it into one logical data view. Creating connections to your data sources enables you to quickly view across your organization's data. This virtual data platform enables real-time analytics without moving data, ETLs, and additional storage requirements, so processing times are greatly accelerated. This produces real-time intutive results quickly and dependably.

Tutorial will use two different data sources from Db2 and Informix. On Db2, you have MORTGAGE_CUSTOMER table that stores individual customer information, those who applied for mortgage loan. Other hand at Informix side, you have MORTGAGE_DEFAULT table that tracks default status of individual mortgage. Using these two tables you need to find out which RESIDENCE type mostly default on mortgage.

Table MORTGAGE_CUSTOMER on Db2 | Table MORTGAGE_DEFAULT on Informix

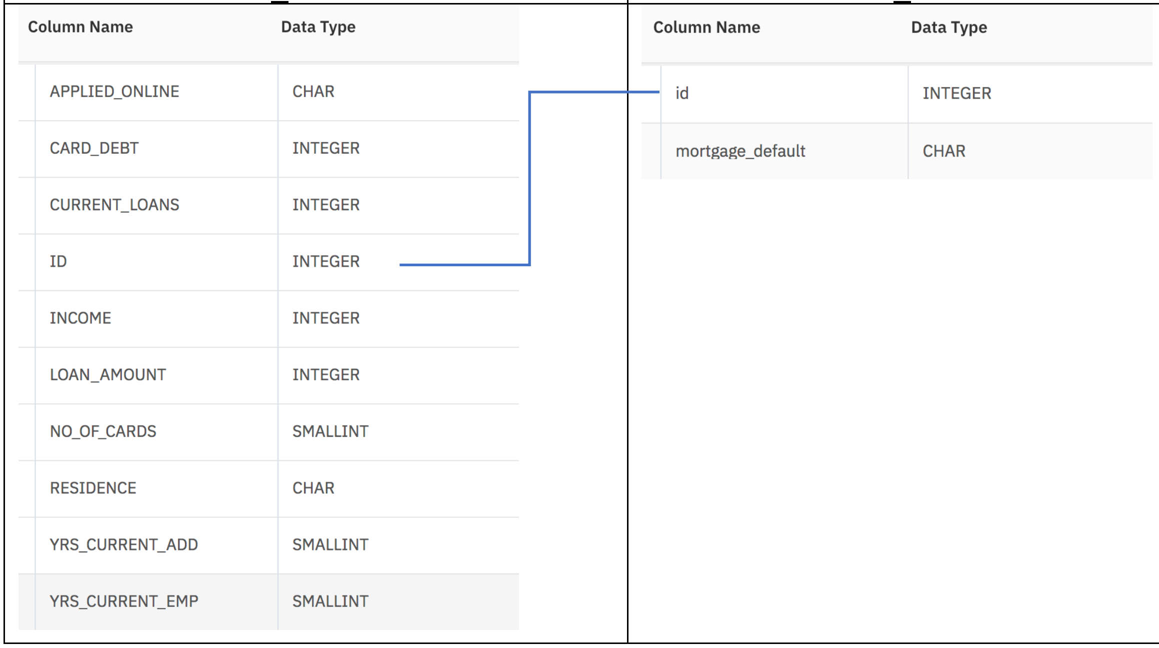


Table MORTGAGE_CUSTOMER on Db2

Column Name	Data Type
APPLIED_ONLINE	CHAR
CARD_DEBT	INTEGER
CURRENT_LOANS	INTEGER
ID	INTEGER
INCOME	INTEGER
LOAN_AMOUNT	INTEGER
NO_OF_CARDS	SMALLINT
RESIDENCE	CHAR
YRS_CURRENT_ADD	SMALLINT
YRS_CURRENT_EMP	SMALLINT

Table MORTGAGE_DEFAULT on Informix

Column Name	Data Type
id	INTEGER
mortgage_default	CHAR

Using DV you will create a virtual table from above two data sources. Next use a R script to access the virtual table and find out the solution. There is absolutely no need of moving data, ETLs, and additional storage.

Before you continue with this tutorial make sure DV and RStudio are provisioned on your environment. Both are add-on components in ICP for Data.

This tutorial has been developed based on ICP for Data v.1.2.1.

1. Access Credentials

To work through the demo, you need access to Db2 and Informix databases. You will use following access credential while define data sources.

1.1. Access credential for Db2 database

JDBC connection credential for Db2:

JDBC Host name	<Same IP address as your web console>
Port number	50000
Database name	MORTGAGE
User ID	db2inst1
Password	password
Db2	Version 11.1
JDBC connection string	jdbc:db2://<same IP as Web Console>:50000/MORTGAGE

1.2. Access credential for Informix database

JDBC connection credential for Informix:

JDBC Host name	<Same IP address as your web console>
Port number	9088
Database name	MORTGAGEDB
User ID	informix
Password	in4mix
Server name	informix
Informix	Version 12.10.FC12W1DE
JDBC connection string	jdbc:informix-sqli://<same IP as Web console>:9088/mortgagedb: INFORMIXSERVER=informix;user=informixt;password=in4mix

1.3. Setting up the databases and sample tables

This tutorial host Db2 and Informix server instances in Docker containers and load sample data to respective databases. You need to run this setup step from command prompt as user root.

a) Log in to the cluster where ICP for Data is deployed.

b) From your home directory, clone the tutorial sample files:

```
git clone https://github.com/sanjitc/icp4d-tutorials.git
```

or

```
git clone https://github.com/IBM-ICP4D/icp4d-tutorials.git
```

c) Change to the tutorials directory:

```
cd icp4d-tutorials/tutorials
```

d) Run the following command to load the sample data into a Db2 database:

```
./load_samples.sh -t mortgage-001
```

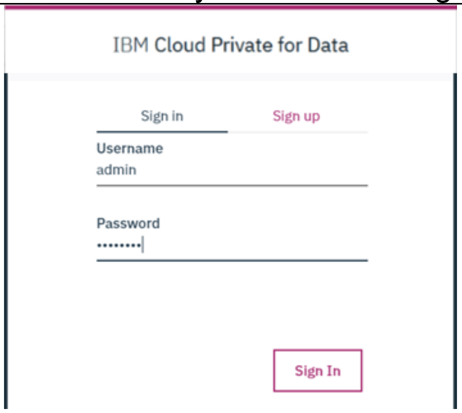
e) Run the following command to load the sample data into an Informix database:

```
./load_samples.sh -t data-virtualization-001
```

f) After data loading completes, Db2 and Informix are hosted on your cluster as a Docker container.

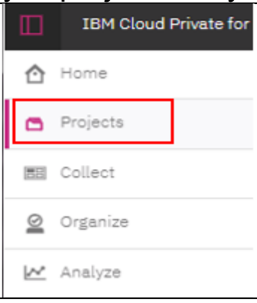
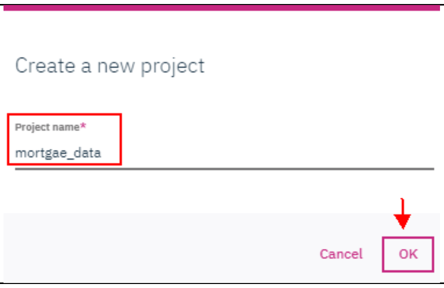
1.4. Sign in to ICP for Data web console as Administrator

You should use latest version of Firefox or Google Chrome browser to access the ICP for Data web console. Starting from here all instruction needs to execute on ICP for Data web console only. You need to login as admin who has administrator privileges.

 Username admin Password ••••••••	Sigh in to the ICPD web console as user ‘admin’ and password is ‘password.

2. Create analytic project

Crate an analytic project that you going to use for virtualizing data.

 IBM Cloud Private for Home Projects Collect Organize Analyze	Select ‘Projects’ from right pane to create a new analytical project Click on the ⊕ New project icon
 Project name* mortgae_data	Provide a project name as **mortgage_data** and click **OK** On the next ‘Create project’ window, click on **Create**

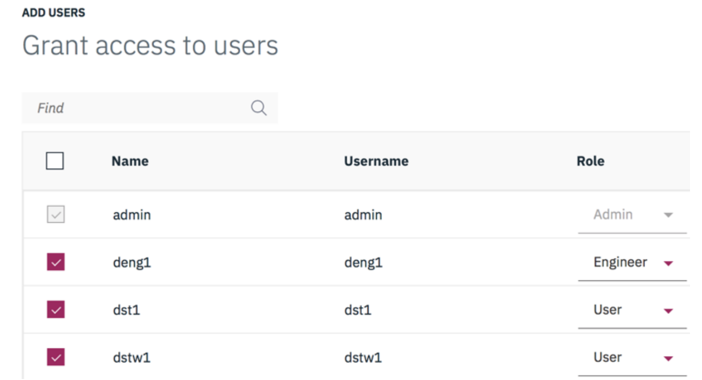
3. Data Virtualization

Data virtualization (DV) integrates data sources across multiple types and locations and turns it into one logical data view. Creating connections to your data sources enables you to quickly view across your organization's data.

3.1. Giving users access to data virtualization (optional)

In order for a user to have access to the data virtualization service, you must assign them to appropriate data virtualization roles.

This is for information only. In this demo you will use user ‘admin’, which has all necessary virtualization roles.



ADD USERS

Grant access to users

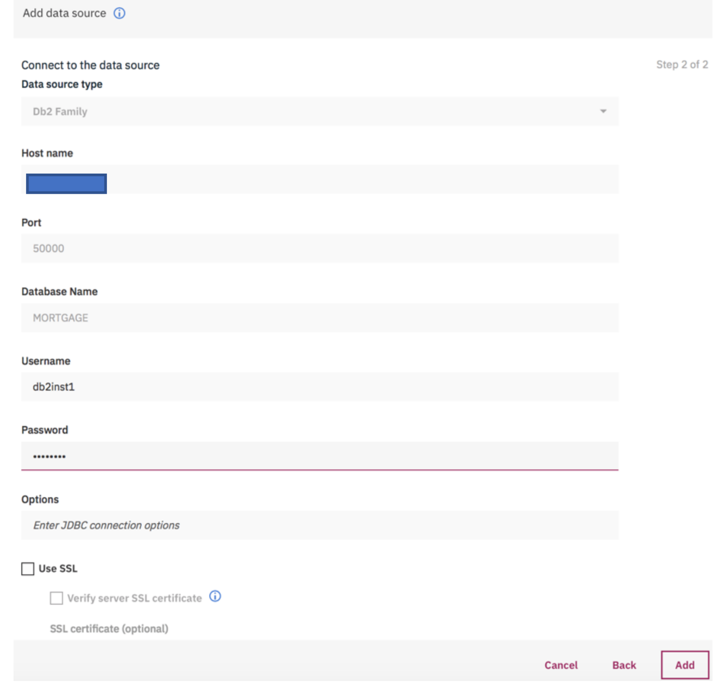
□	Name	Username	Role
☑	admin	admin	Admin
☑	deng1	deng1	Engineer
☑	dst1	dst1	User
☑	dstw1	dstw1	User

1. Select **Collect** > **Virtualize data** from left pane
2. Select **Menu** > **Manage users** > **Add users** from top
3. Check all users that you created earlier and keep their default role.
4. Click **Add**

3.2. Adding a new data source for Db2

DV supports many relational and non-relational data sources, as well as files that reside on a local disk or network file system, that you can add to your data source ecosystem. After a data source has been added, any user that has virtualize permission has the ability to create virtual tables. DV agents connect to relational data sources using JDBC protocol. In this tutorial you will add two data sources, one for Db2 and other one for Informix.

Define a data connection to Db2. Use following connection information with ‘Add connection’. If you already have an existing database connection, select that for Db2 data source.



Add data source ⓘ

Connect to the data source — Step 2 of 2

Data source type: Db2 Family

Host name: [redacted]

Port: 50000

Database Name: MORTGAGE

Username: db2inst1

Password: ••••••••

Options: *Enter JDBC connection options*

□ Use SSL

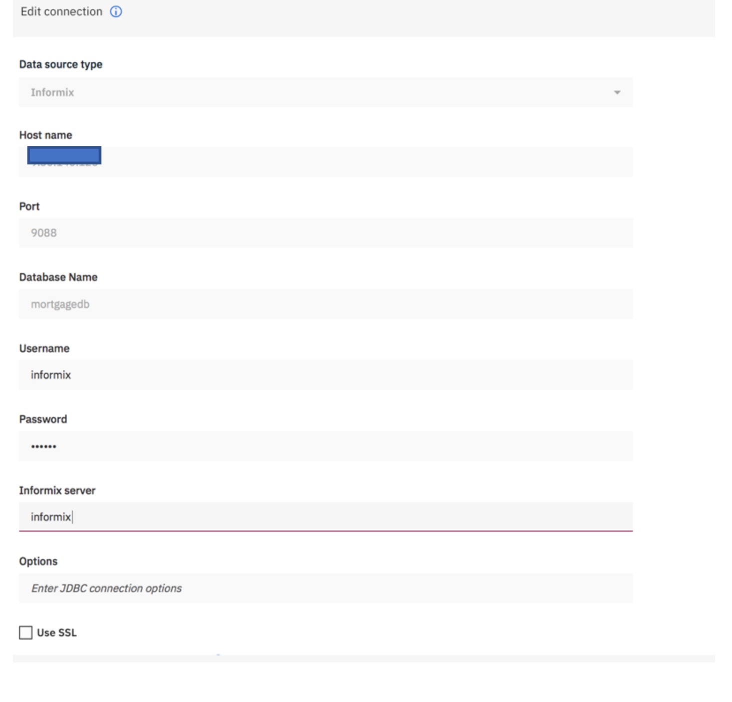
□ Verify server SSL certificate ⓘ

SSL certificate (optional)

1. Go to **Collect** > **Virtualized data** > **Menu** > **Data sources**
2. Click **Add data source** > **New data source**
3. Update data source with following information:
   Data source type = Db2
   Host name = <IP of node 1>
   Port = 50000
   Database Name = MORTGAGE
   Username = db2inst1
   Password = password
4. Click **Add**

3.3. Adding a new data source for Informix

Let's add a new data source for the Informix. First define a data connection to Informix. Use following connection information with 'Add connection'. If you already have an existing database connection, select that for Informix data source.



Edit connection ⓘ

Data source type: Informix

Host name: [redacted]

Port: 9088

Database Name: mortgagedb

Username: informix

Password: ••••••

Informix server: informix

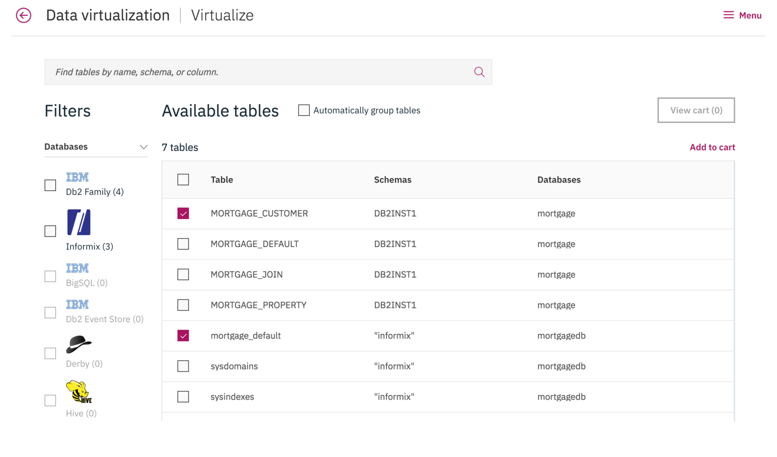
Options: *Enter JDBC connection options*

□ Use SSL

1. Go to **Collect** > **Virtualized data** > **Menu** > **Data sources**
2. Click **Add data source** > **New data source**
3. Update data source with following information:
   Data source type = Informix
   Host name = <IP of node 1>
   Port = 9088
   Database Name = mortgagedb
   Username = informix
   Password = in4mix
   Informix server = informix
4. Click **Add**

3.4. Select tables for virtualization

The most common mechanism for virtualizing data is to create a "view" or virtual table. Virtual table can be full or segment of data from one or more tables. You can then run queries against the resulting virtual table.



Available tables □ Automatically group tables

7 tables

□	Table	Schemas	Databases
☑	MORTGAGE_CUSTOMER	DB2INST1	mortgage
□	MORTGAGE_DEFAULT	DB2INST1	mortgage
□	MORTGAGE_JOIN	DB2INST1	mortgage
□	MORTGAGE_PROPERTY	DB2INST1	mortgage
☑	mortgage_default	"informix"	mortgagedb
□	sysdomains	"informix"	mortgagedb
□	sysindexes	"informix"	mortgagedb

1. Click **Collect** > **Virtualized data** > **Menu** > **Virtualize**

2. Select three tables **MORTGAGE_CUSTOMER** from MORTGAGE database and **mortgage_default** from mortgagedb, then click **Add to cart**

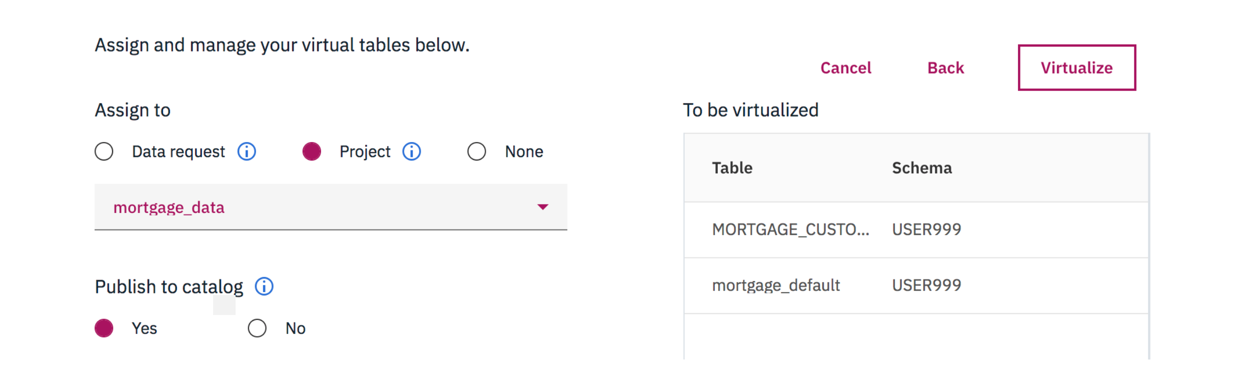
3. Click **View cart**

4. Click **Next**

1. Select **project** to assign virtualized table to your analytics project. Then, choose the **mortgage_data** project.

2. Choose **publish to catalog** for include virtualized table to the data catalog. This operation will create a publishing request, a data steward must approve the request before the asset is added to the enterprise data catalog.

3. Click **Virtualize** to complete the process



Assign and manage your virtual tables below.

Assign to

○ Data request ⓘ ● Project ⓘ ○ None

mortgage_data

Publish to catalog ⓘ

● Yes ○ No

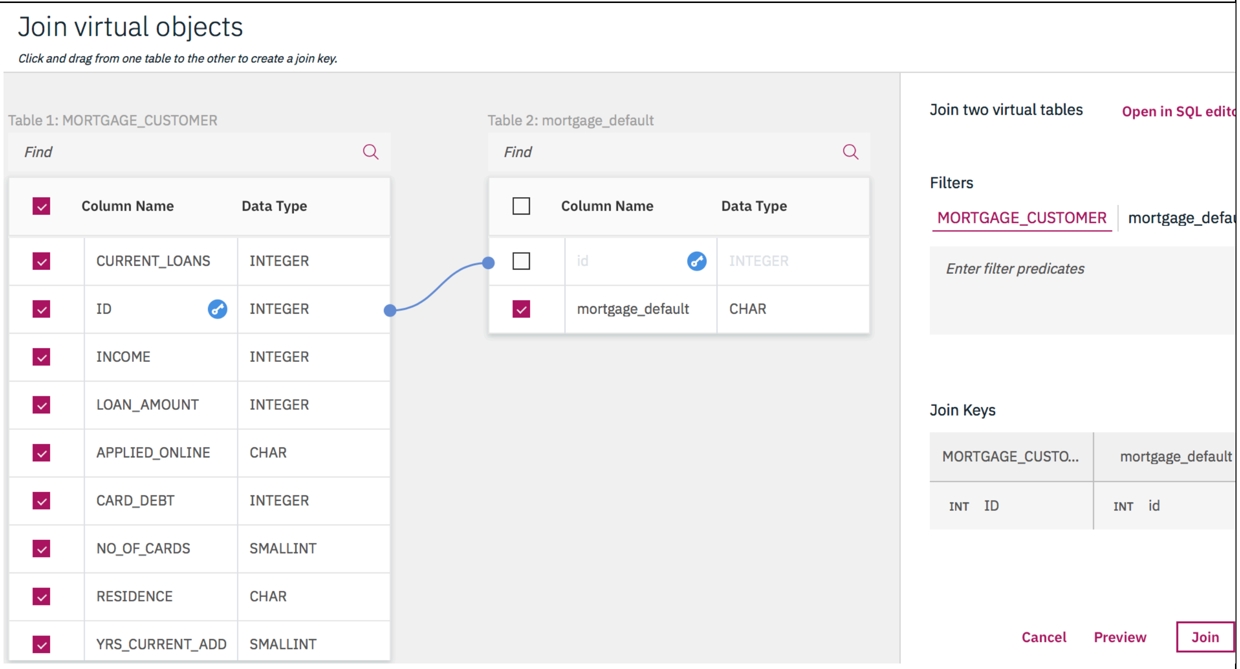
To be virtualized

Table	Schema
MORTGAGE_CUSTO...	USER999
mortgage_default	USER999

3.5. Creating virtual table

You can create a new virtual table based on existing tables. Join source tables and create the virtual table.

1. Click **Collect** > **Virtualized data** > **Menu** > **My data** to see your virtualized tables.
2. Check **MORTGAGE_CUSTOMER** and **mortgage_default** tables for join.
3. Click on **Join view**
4. Uncheck the ID column from mortgage_default table for reduction redundancy
5. Click and drag from one **ID** column to another to create a join key. Both join keys must be of the same data type.
6. Click **Join**



Join virtual objects

Click and drag from one table to the other to create a join key.

Table 1: MORTGAGE_CUSTOMER

Find

☑	Column Name	Data Type
☑	CURRENT_LOANS	INTEGER
☑	ID	INTEGER
☑	INCOME	INTEGER
☑	LOAN_AMOUNT	INTEGER
☑	APPLIED_ONLINE	CHAR
☑	CARD_DEBT	INTEGER
☑	NO_OF_CARDS	SMALLINT
☑	RESIDENCE	CHAR
☑	YRS_CURRENT_ADD	SMALLINT

Table 2: mortgage_default

Find

□	Column Name	Data Type
□	id	INTEGER
☑	mortgage_default	CHAR

Join two virtual tables

Open in SQL edit

Filters

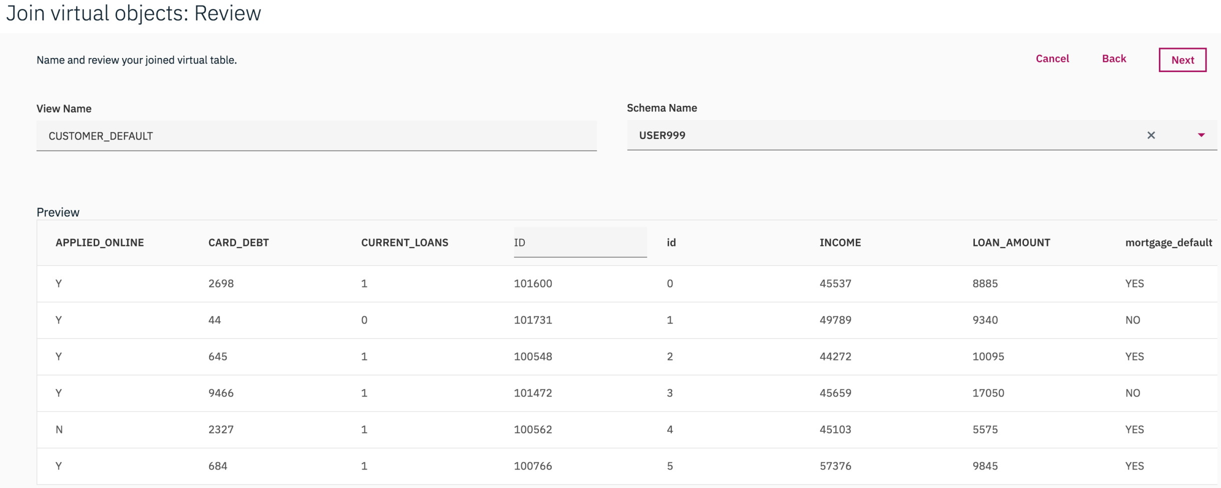
MORTGAGE_CUSTOMER | mortgage_defa

Enter filter predicates

Join Keys

MORTGAGE_CUSTO...	mortgage_default
INT ID	INT id

Name the view as **CUSTOMER_DEFAULT,** then click **Next.** Make sure use uppercase for view name.



Join virtual objects: Review

Name and review your joined virtual table.

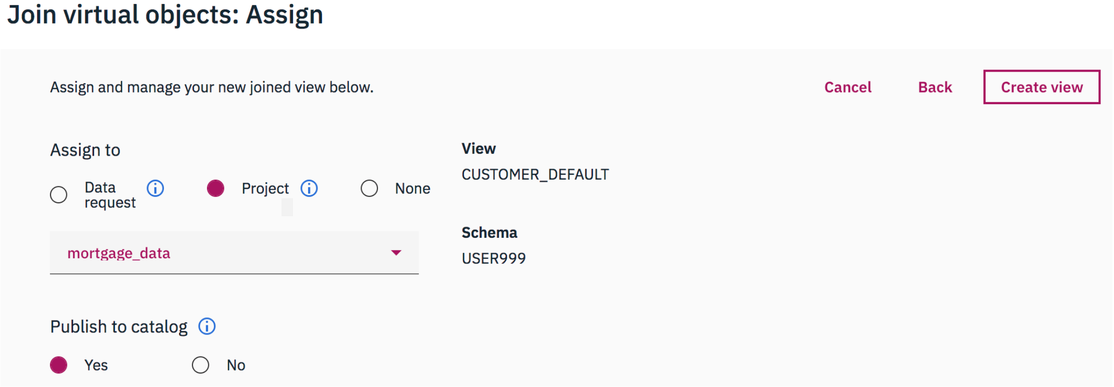
View Name: CUSTOMER_DEFAULT

Schema Name: USER999

Preview

APPLIED_ONLINE	CARD_DEBT	CURRENT_LOANS	ID	id	INCOME	LOAN_AMOUNT	mortgage_default
Y	2698	1	101600	0	45537	8885	YES
Y	44	0	101731	1	49789	9340	NO
Y	645	1	100548	2	44272	10095	YES
Y	9466	1	101472	3	45659	17050	NO
N	2327	1	100562	4	45103	5575	YES
Y	684	1	100766	5	57376	9845	YES

1. Select **project** to assign virtualized table to your analytics project. Then, choose the **mortgage_data** project.
2. Click **Create view**



Join virtual objects: Assign

Assign and manage your new joined view below.

Assign to

○ Data request ● Project ○ None

mortgage_data

View
CUSTOMER_DEFAULT

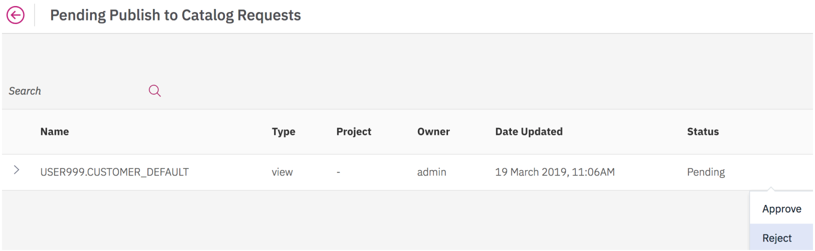
Schema
USER999

Publish to catalog

● Yes ○ No

3.6. Publish virtualized table

A data steward needs approve the published request before the asset is added to the enterprise data catalog. You signed in as user 'admin', it should allow to publish the virtual table.


Pending Publish to Catalog Requests

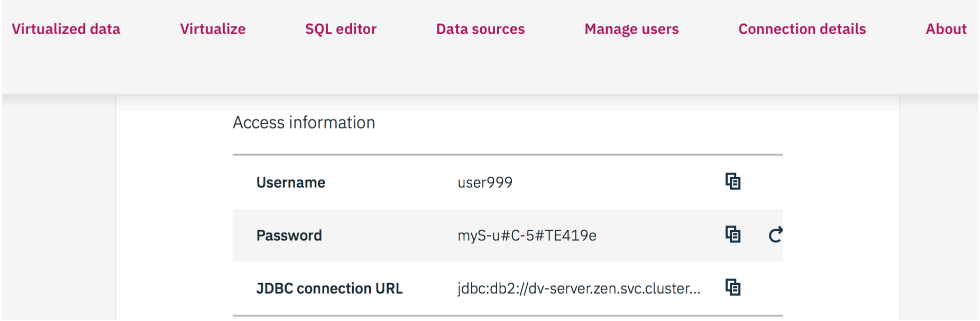
Search

	Name	Type	Project	Owner	Date Updated	Status
>	USER999.CUSTOMER_DEFAULT	view	-	admin	19 March 2019, 11:06AM	Pending

1. Click on ⌂ access the **Home** page
2. Click on **Pending Publish to Catalog Requests**
3. Click on ⋮ icon on left for virtual table **CUSTOMER_DEFAULT** that you created
4. Click on **Approve**

3.7. Access information for virtual table

To access virtual table from external application, you need the JDBC connection information. Click on **Collect** > **Virtualized data** > **Menu** > **Service details** to find out access information.



Access information

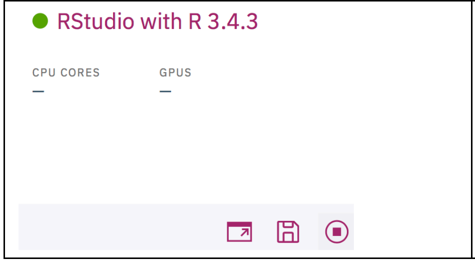
Username	user999
Password	myS-u#C-5#TE419e
JDBC connection URL	jdbc:db2://dv-server.zen.svc.cluster...

4. Virtual Table From IDE

You can access virtual table from inside or outside of ICP for Data. In this example a R script used for accessing the virtual table. ICP for Data comes with an add-on RStudio component that provides an Integrated Development Environment (IDE) for working with R.

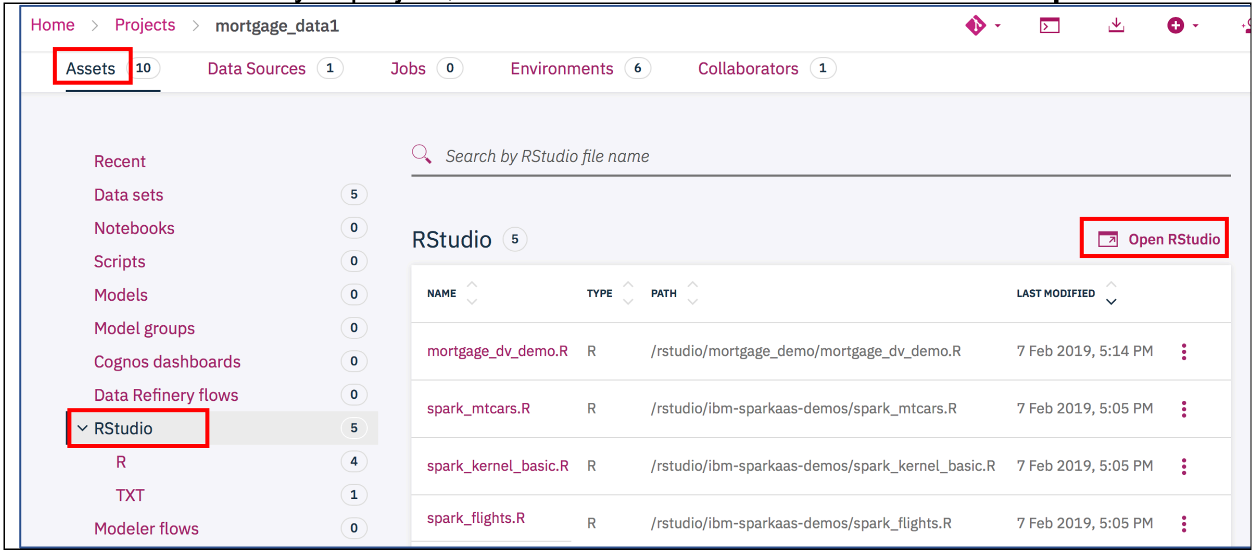
4.1. Start IDE

Make sure IDE has started before you can use it.

Select your project **mortgage_data** from project menu	
 ● RStudio with R 3.4.3 CPU CORES — GPUS —	Choose **Environments** tab. If RStudio not running, click on ▷ icon to start.

4.2. Launch RStudio

From inside the analytic project, choose **Assets** > **RStudio** and click on **Open RStudio**.



Home > Projects > mortgage_data1

Recent	
Data sets	5
Notebooks	0
Scripts	0
Models	0
Model groups	0
Cognos dashboards	0
Data Refinery flows	0
RStudio	5
R	4
TXT	1
Modeler flows	0

RStudio 5

NAME	TYPE	PATH	LAST MODIFIED
mortgage_dv_demo.R	R	/rstudio/mortgage_demo/mortgage_dv_demo.R	7 Feb 2019, 5:14 PM
spark_mtcars.R	R	/rstudio/ibm-sparkaas-demos/spark_mtcars.R	7 Feb 2019, 5:05 PM
spark_kernel_basic.R	R	/rstudio/ibm-sparkaas-demos/spark_kernel_basic.R	7 Feb 2019, 5:05 PM
spark_flights.R	R	/rstudio/ibm-sparkaas-demos/spark_flights.R	7 Feb 2019, 5:05 PM

4.3. Load R script

Within the RStudio, go to File > New File > R script to run script. Copy the following script and paste it in the source section. This simple R script will simply run a SELECT statement against the virtual table and retrieve data and create a treemap to visually displays mortgage default by residency type.

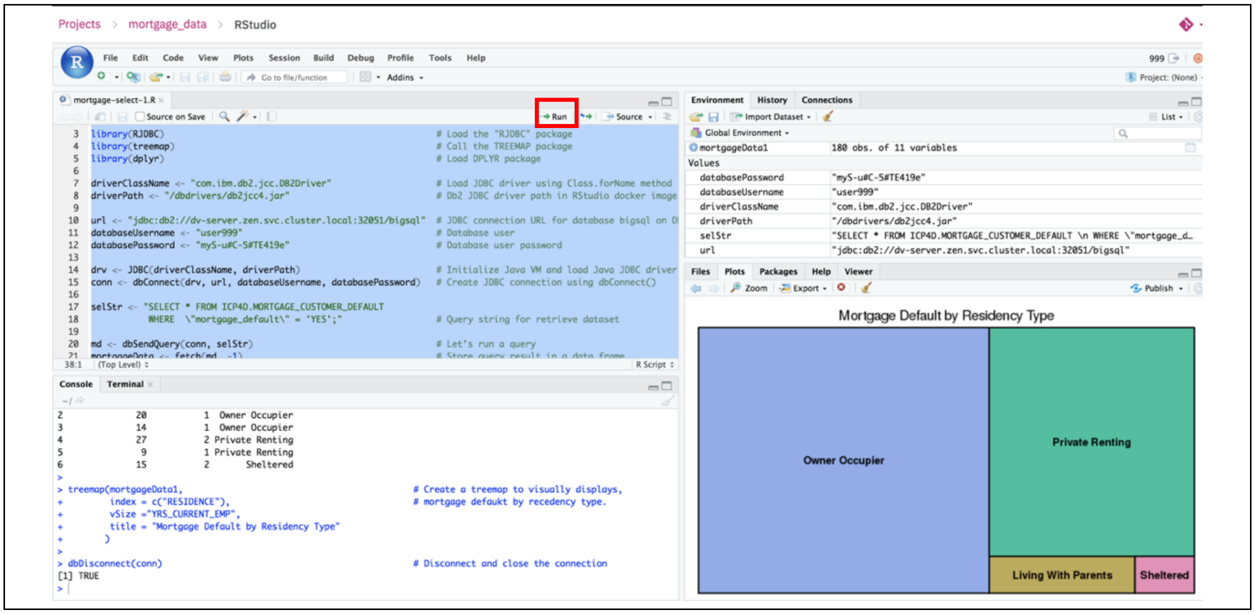
Before you run the script, please change the **url**, **databaseUsername** and **databasePassword** according to your system, which you found in step 3.7 (Access information for virtual table).

```
#####
### This is a sample R code that access the virtual table from Data Virtualization environment and create a treemap
### to visually displays mortgage default by residency type.
###
### Before executing the R script update URL, databaseUsernname and databasePassword variables according to your environment .
### Variable values can be found in 'Collect -> Virtualized data -> Menu -> Service details' section in Data Virtualization environment.
#####

library(RJDBC)                                                    # Load the "RJDBC" package
library(treemap)                                                  # Call the TREEMAP package
library(dplyr)                                                    # Load DPLYR package

driverClassName <- "com.ibm.db2.jcc.DB2Driver"                    # Load JDBC driver using Class.forName method
driverPath <- "/dbdrivers/db2jcc4.jar"                            # Db2 JDBC driver path in RStudio docker image

url <- "<JDBC Connection URL>"                                    # Update JDBC connection URL according to your
                                                                  # environment information in 'Collect -> Virtualized
                                                                  # data -> Menu -> About'

databaseUsername <- "<username>"                                  # Database user
databasePassword <- "password>"                                   # Database user password

drv <- JDBC(driverClassName, driverPath)                          # Initialize Java VM and load Java JDBC driver
conn <- dbConnect(drv, url, databaseUsername, databasePassword)   # Create JDBC connection using dbConnect()

selStr <- "SELECT * FROM USER999.MORTGAGE_CUSTOMER_DEFAULT
           WHERE  \"mortgage_default\" = 'YES';"                  # Query string for retrieve dataset

md <- dbSendQuery(conn, selStr)                                   # Let's run a query
mortgageData <- fetch(md, -1)                                     # Store query result in a data frame

mortgageData1 <- mortgageData %>%                                 # Expand residency type to meaningful name
  mutate(RESIDENCE = case_when(RESIDENCE == "O" ~ "Owner Occupier",
                               RESIDENCE == "P" ~ "Private Renting",
                               RESIDENCE == "L" ~ "Living With Parents",
                               RESIDENCE == "S" ~ "Sheltered"))

head(mortgageData1)                                               # Return first parts of data frame

treemap(mortgageData1,                                            # Create a treemap to visually displays,
        index = c("RESIDENCE"),                                   # mortgage default by residency type.
        vSize ="YRS_CURRENT_EMP",
        title = "Mortgage Default by Residency Type"
       )

dbDisconnect(conn)                                                # Disconnect and close the connection
```

If any R package not already installed, you need to install them first before executing the above script. Use **install.packages("*<package name>*")** command to install a R package.

4.4. Run the script

Highlight the R script in the RSudio and click on **Run** to retrieve data.



```
library(RJDBC)                                                      # Load the "RJDBC" package
library(treemap)                                                    # Call the TREEMAP package
library(dplyr)                                                      # Load DPLYR package

driverClassName <- "com.ibm.db2.jcc.DB2Driver"                      # Load JDBC driver using Class.forName method
driverPath <- "/dbdrivers/db2jcc4.jar"                              # Db2 JDBC driver path in RStudio docker image

url <- "jdbc:db2://dv-server.zen.svc.cluster.local:32051/bigsql"   # JDBC connection URL for database bigsql on D
databaseUsername <- "user999"                                       # Database user
databasePassword <- "myS-u#C-5#TE419e"                              # Database user password

drv <- JDBC(driverClassName, driverPath)                            # Initialize Java VM and load Java JDBC driver
conn <- dbConnect(drv, url, databaseUsername, databasePassword)     # Create JDBC connection using dbConnect()

selStr <- "SELECT * FROM ICP4D.MORTGAGE_CUSTOMER_DEFAULT
           WHERE  \"mortgage_default\" = 'YES';"                    # Query string for retrieve dataset

md <- dbSendQuery(conn, selStr)                                     # Let's run a query
mortgageData <- fetch(md, -1)                                       # Store query result in a data frame
```

```
2            20          1  Owner Occupier
3            14          1  Owner Occupier
4            27          2 Private Renting
5             9          1 Private Renting
6            15          2       Sheltered
>
> treemap(mortgageData1,                                    # Create a treemap to visually displays,
+         index = c("RESIDENCE"),                           # mortgage defaukt by recedency type.
+         vSize ="YRS_CURRENT_EMP",
+         title = "Mortgage Default by Residency Type"
+       )
>
> dbDisconnect(conn)                                        # Disconnect and close the connection
[1] TRUE
>
```

mortgageData1	180 obs. of 11 variables
Values	
databasePassword	"myS-u#C-5#TE419e"
databaseUsername	"user999"
driverClassName	"com.ibm.db2.jcc.DB2Driver"
driverPath	"/dbdrivers/db2jcc4.jar"
selStr	"SELECT * FROM ICP4D.MORTGAGE_CUSTOMER_DEFAULT \n WHERE \"mortgage_d…
url	"jdbc:db2://dv-server.zen.svc.cluster.local:32051/bigsql"

The above treemap figure shows RESIDENCE type “Owner Occupier” has highest rate of default on mortgage.