Daffodil Replicator

Console Guide

August ‘2005

SCO 42,3rd Floor
Old Judicial Complex, Civil lines
Gurgaon - 122001
Haryana, India.
www.daffodildb.com

Table of Contents

Preface

Purpose

Daffodil Replicator Console Guide explains the setting up of Daffodil Replicator using Graphical User Interface or Console. All the steps are illustrated with screen shots and examples. It describes the following:

1. Installation and getting started with Daffodil Replicator.
2. Working with Publication Server and Subscription Server.
3. Replication technologies like Snapshot, Synchronization, Pull and Push Replication.
4. Daffodil Replicator Menus and Panels.

This guide does not provide detailed description about the working of Daffodil Replicator or other concepts and terminology. Daffodil Software recommends reading the *Daffodil Replicator Developer's Guide* for more detailed information about concepts and terminology.

Audience

Daffodil Replicator Console Guide is intended to act as a ready reference tool for database administrators, developers, application programmers and any user who wish to replicate their databases.

It assumes that you have basic knowledge of the following:

- Concepts and terminology of Daffodil Replicator.
- Server and Client concepts.
- Structured Query Language (SQL).
- Java programming language.
- Java Database Connectivity (JDBC) API
- RDBMS concepts.

Installation Guidelines

This section describes how to install Daffodil Replicator and start working with it.

Pre-requisites

Before starting off with the installation of Daffodil Replicator, make sure that JRE (Java Runtime Environment) 1.4 or above is installed on your system. For customizing Replicator to suit your needs, you need to install the full J2SDK 1.4 or above. If you don't have JRE and J2SDK, you can download these from http://java.sun.com/j2se/1.4/download.html.You will also need log4j.jar which can be downloaded from http://jakarta.apache.org/log4j/docs/download.html.

Installation Steps

1. Go to http://www.daffodildb.com/dbreplicator.html and click on the Download button.
2. You will be prompted to enter your email address and other details.
3. In the download page, you will have various options to download. Choose the latest one or the one suited for your specific replication needs.
4. After clicking download button, you will be redirected to SourceForge.net
5. Download the appropriate zip file (For example, DaffodilReplicator1_8.zip, if you want to install Daffodil Replicator1_8)
6. Extract the corresponding zip file to an appropriate folder.
7. Make the following changes by editing setpath.bat (windows) or **PubServer.sh and SubServer.sh** (Linux)

    - Remove Rem from SET JAVA_HOME= and SET

- JDBC_CLASSPATH ="Replicator.jar"
- Set JAVA_HOME. JAVA_HOME is the directory path where you have installed JRE/J2SDK.

  Example: SET JAVA_HOME=C:\j2sdk1.4.2
- Set JDBC_CLASSPATH for the corresponding data source, log4j.jar and Replicator.jar.

  Example: If you are using Daffodil DB, then add the path for the following jar files.
  SET JDBC_CLASSPATH= C:\DaffodilDB3_4\lib\DaffodilDB_Client.jar;C:\DaffodilDB3_4\lib\DaffodilDB_Common.jar;Replicator.jar; log4j.jar

Note: *Directory path should not contain any blank spaces.*
For other database JDBC Driver jars, please refer *APPENDIX 1(A)*

8. Run PubServer.bat/PubServer.sh to start publication server
9. Run SubServer.bat/SubServer.sh to start subscription server

CONFIG.INI

Config.ini file contains configuration information for **REPLICATION HOME**, **TRANSACTION LOG FILE**, **SHADOW TABLE**, **TRIGGER**, **SEQUENCE** AND **INDEX PREFIX**. Detail of each is given below.

➢ *Replication Home*

Replication home is a path where transaction logs, error log and XML files are created. XML files come in use during replication process when snapshot, synchronization, push and pull operations are performed. Transaction log file have information related to DML (Insert/Update/Delete) operations performed in data source during replication process. You can find the entire Stack trace related to exceptions which have occurred during replication process, in error log file.

There is a variable named '**REPLICATIONHOME'** in config.ini to represent the replication home path. User can set the replication home at specified path according to the requirement. But If the REPLICATIONHOME is set to Default as given below then all above mentioned file will be created in user's home.

REPLICATIONHOME=Default

➢ *Transaction Log File*

Transaction log file maintains the record of the transactions performed by replicator during replication processes. For each replication process, a new transaction log will be created with name specified by current date and time. In a transaction log file, it is also specified that which replication process is being performed.

There are two types of transaction detail ***full*** and ***partial.*** If users want to get the transaction detail according to his requirement, there is a variable named '**TRANSACTIONDETAIL'** in Config.ini which will allow the user to do so by setting the properties of transaction log file.

TRANSACTIONDETAIL=TRUE

If the property '**TRANSACTIONDETAIL**' is set to **TRUE** then it will contain the '**Detail**' (Full) information of the transactions that are performed during replication process as given below :-

Operation Performed on Date: 2005-04-23 13:01:03.746

Synchronization Type [PULL]

Inserts [5]
Updates [2]
Deletes [1]

[SYNCHRONIZE COMPLETED SUCCESSFULLY]

[PULL REPLICATION][DBO.STUDENT][I][3, Again, NULL, NULL, NULL, NULL]

[PULL REPLICATION][DBO.STUDENT][I][4, Again, NULL, NULL, NULL, NULL]

[PULL REPLICATION][DBO.STUDENT][I][5, Again, NULLNULL, NULLNULL]

[PULL REPLICATION][DBO.STUDENT][U][PRIMARY KEY VALUES rollNo = 10][CHANGED COLUMNS rollNo = 1]

[PULL REPLICATION][DBO.STUDENT][U][PRIMARY KEY VALUES rollNo = 20][CHANGED COLUMNS rollNo = 2]

[PULL REPLICATION][DBO.STUDENT][D][PRIMARY KEY VALUES rollNo = 0]

TRANSACTIONDETAIL=FALSE

But if **'TRANSACTIONDETAIL'** property is set to **FALSE** then it will contain '**Partial**' information of the transactions that are performed during replication process. It will give information about number of records only. For example transaction log file will have

following information if five new records are inserted, two records updated and one record is deleted during pull operation.

Operation Performed on Date: 2004-03-23 13:01:03.746

Synchronization Type [SYNCHRONIZE]

Inserts [5]
Updates [2]
Deletes [1]

[SYNCHRONIZE COMPLETED SUCCESSFULLY]

***NOTE**: User must delete transaction log files in case of scheduling time to time.*

- ***Provision for Shadow Tables, Triggers, Sequences and Indexes prefix***

User can change the pre-fixes of *Shadow Tables, Triggers, Sequences and Indexes* using *config.ini* file. The prefix values can be changed accordingly for each of the respective schema objects as given below.

ShadowTablePrefix=R_S_

Here *'R_S_'* is a prefix for shadow table name .This can be modified according to the requirement.

log4j.properties(Debugging)

In your current path, there is a file name: **log4j.properties**.This file will be created as soon as replication server is started or its instance is created.

Replicator is providing the facility of ***debugging*** using '**Log4j**'. In this file, user can set the logging option as per the requirement. One can set **debug level** to any one of the following to control the amount of logging:

- **Warn**: The WARN level designates potentially harmful situations
- **Error**: The ERROR level designates error events that might still allow the application to continue running.
- **Fatal**: The FATAL level designates very severe error events that will presumably lead the application to abort.
- **Debug**: The DEBUG Level designates fine-grained informational events that are most useful to debug an application
- **Info**: The INFO level designates informational messages that highlight the progress of the application at coarse-grained level
- **Off**: The OFF Level has the highest possible rank and is intended to turn off logging.

- **All**: The ALL Level has the lowest possible rank and is intended to turn on all logging.

Format of the File is:

log4j.rootCategory=off
 # log4j.rootCategory=DEBUG, A1
log4j.appender.A1=org.apache.log4j.ConsoleAppender
log4j.appender.A1.layout=org.apache.log4j.PatternLayout
log4j.appender.A1.layout.ConversionPattern=%d {ABSOLUTE} %p [%c {1}] [%M %L] %m%n

By default, all loggings are **OFF** and if user wants to have loggings on, he/she can set *log4j.rootCategory* to any of above mentioned debug level.

A1-It is used to set the properties of **log4j.appender** which are used to specify where logging prints must be appended and what must be its layout. By default, **log4j.appender** is set to '**Console**', but one can set to other properties also.

Users can choose from several destinations for log statements, such as console, file, database, SMTP servers, GUI components etc.; with option of assigning different destinations to different categories of log statements. These log destinations can be changed anytime by simply changing log4j configuration files.

Encodeconfig.ini

Special Characters Handling

Special characters with ASCII value less than 32 are not parsable and create problem in replication operations. This can be fixed by using **EncodeConfig.ini** file. It is necessary to enter the column name (those having special characters) in **EncodeConfig.ini** file against the respective table name. If number of columns is more than one, they should be separated by commas as listed below:

CMSADM2.CMS_USERS=PWD1, PWD2

Here '*CMSADM2.CMS_USERS*' is table name and **'***PWD1*','*PWD"*' are the column names that have special characters.

Backward Compatibility:

If a user working with old version of Daffodil Replicator and now would like to use the latest version, need to update the previous version using **UpdateVersion.bat**. It updates the structure of system tables and creates new tables that are introduced in latest version. Before running the above mentioned file update the config.ini file.

Replication Concept

Daffodil Replicator is a data replication tool that synchronizes data residing on corporate back-end databases and desktop databases operating in the network. Daffodil Replicator performs Snapshot and Merge replication based on Publish and Subscribe model (in client and server architecture). The publications and subscriptions are defined using GUI or APIs on existing database servers.

Daffodil Replicator has been developed for easy replication between homogeneous and heterogeneous data sources. It performs **bi-directional synchronization** between **Oracle**, **SQL Server**, **DB2**, **Daffodil DB**, **PostgreSQL** and **Derby**, enabling high-performance distributed connectivity with corporate databases residing on **Windows**, **Linux**, **UNIX** or **Solaris**. Replicator's publish-and-subscribe model allows a client application to have subscriptions to a publisher defined on the server side, which can consist of single or multiple data tables. Replicator *filters* enable application developers to specify conditions for individual tables, through which the client can subscribe to a subset of rows.

The replication technology provided by Daffodil Replicator enhance the application performance by offering greater autonomy to users who can work with a local copy of the database and then transfer the changes to remote users across the network. The data can be transferred either on-demand or periodically over existing LAN / WAN set ups.

Replication Technologies

Daffodil Replicator supports the following Replication technologies:

1. **Snapshot Replication** – ***Snapshot replication*** simply takes a "**snapshot**" of the database on one server and moves that data to another database. This one-time process is a pre-requisite for Merge, Pull and Push replication.

2. **Merge Replication** – ***Merge replication*** compares data from Publisher and Subscriber and update the changes to both the databases. Before attempting Merge replication (for the first time), it is important to take the snapshot (a one-time process) of data from the Publisher. Once the snapshot is taken, you can perform merge replication any number of times. Merge Replication is also called as Change-based Replication or Synchronization.

3. **Pull Replication** – ***Pull replication*** compares Publishers and Subscribers data and the changes made to the data by the Publisher are updated in the Subscriber's database. But the changes made by the Subscriber are not updated to the Publisher's database. So in Pull replication, records (that are changed due to insert, update and delete queries at Publisher's database) are pulled from Publisher to Subscriber by the Subscriber. Before attempting Pull replication (for the first time), it is important to take the snapshot (a one-time process) of data from the Publisher. Once the snapshot is taken, you can perform Pull replication any number of times.

4. **Push Replication** – ***Push replication*** compares Publishers and Subscribers data and the changes made to the data by the Subscriber are updated in the Publisher's database. But the changes made by the Publisher are not updated to the Subscriber's database. So in Push replication (for the first time), records (that are changed due to insert, update and delete queries at Subscriber's database) are pushed to Publisher by the Subscriber. Before attempting Push replication (for the first time), it is important to take the snapshot (a one-time process) of data from the Publisher. Once the snapshot is taken, you can perform Push replication any number of times.

Note

- *Taking the snapshot of Publisher data (a one-time process) by the Subscriber is a pre-requisite for Merge, Pull and Push replication. Merge, Pull and Push replication never takes the snapshot of the data; they only compares the data on both Publisher and Subscriber and update changes, if any, corresponding to the respective replication methods.*

- *All the Replication methods described above can **only** be initiated by the Subscriber.*

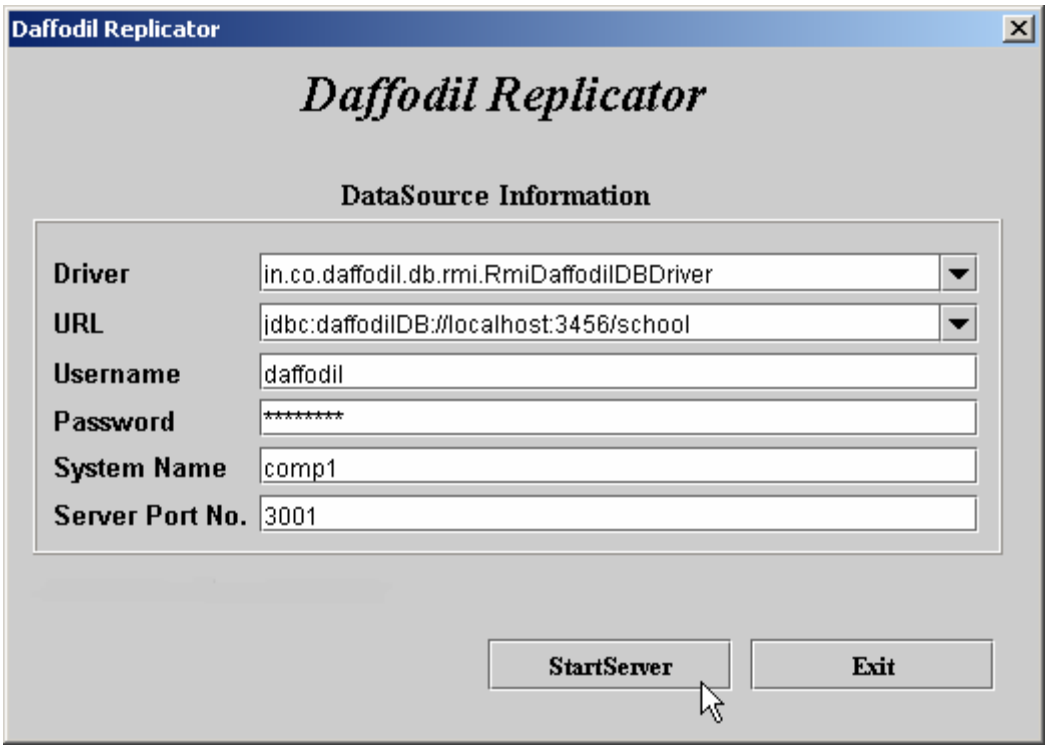
For more details and concepts of Replication methods, see *Daffodil Replicator Developer's Guide*

Starting the Publication Server

- To start Publication Server on your system, run the file PubServer.bat or PubServer.sh (for Linux) available under the installation directory of Daffodil Replicator and a dialog box appears.

- Fill up the following DataSource Information in the dialog box as shown in the following screen shot.

  Note: *In the examples below, the default values are for Daffodil DB database (Network Edition). Users can change it according to their database.*



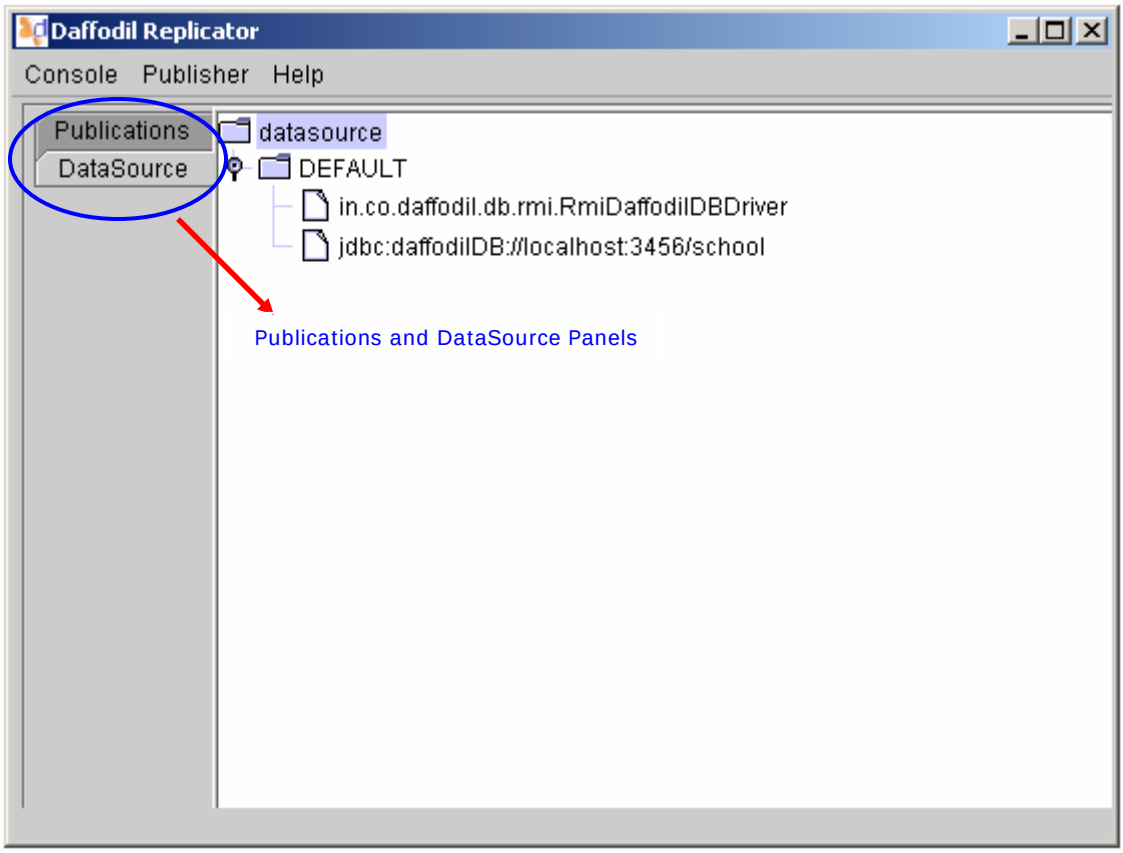
  - **Driver** – This is the name of the driver with which you wish to connect to the database.
    Example: in.co.daffodil.db.rmi.RmiDaffodilDBDriver (For Daffodil DB Network mode)
    in.co.daffodil.db.jdbc.DaffodilDBDriver (For Daffodil DB Embedded Edition)
    NOTE*: For Drivers of other databases, please refer* *APPENDIX 1(B)*

  - **URL** – Specify the URL of the database.
    Example: jdbc: daffodilDB: //localhost:3456/<database name> (For Network mode)
    Jdbc: DaffodilDB_embedded :< database name> (For Embedded mode)

    NOTE*: For URLs of other databases, please refer* *APPENDIX 1(B)*

- **Username** – Enter a valid username for the specified database. Default username of Daffodil DB is daffodil
- **Password** – Enter the password of the above specified user. Default password of Daffodil DB is daffodil
- **System Name** – Specify the name or network IP of the local machine.
- **Server Port No.** – Specify the port no. on which you wish to run Publication Server. By default it is 3001 but you are free to change it to any available port.

- After filling up the DataSource Information, **click** on the **StartServer** button and the Publication dialog box appears as shown below:
- **Click** on the **DataSource** panel to view the details of the database to which you have established a connection.



Using the Publication Wizard

Creating a new Publication

To create a new publication,

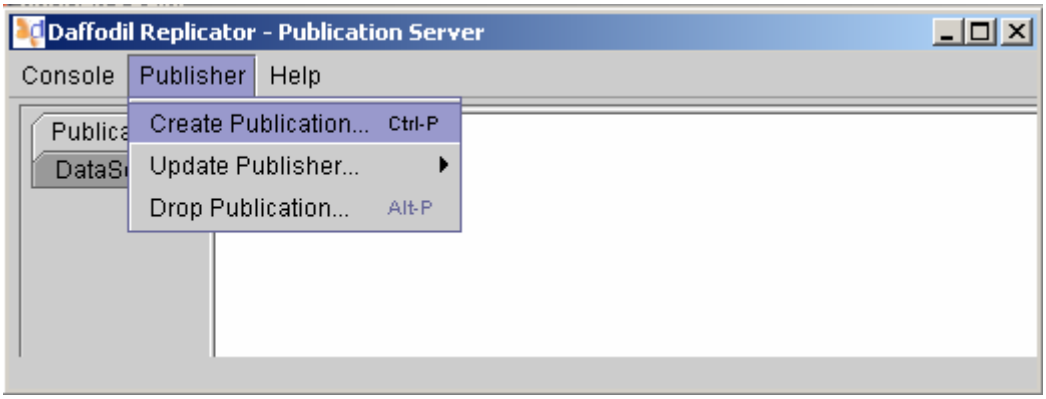
- **Click** on the **Publisher** menu and select **Create Publication** (or press CTRL+P). A **Create Publication** dialog box appears.

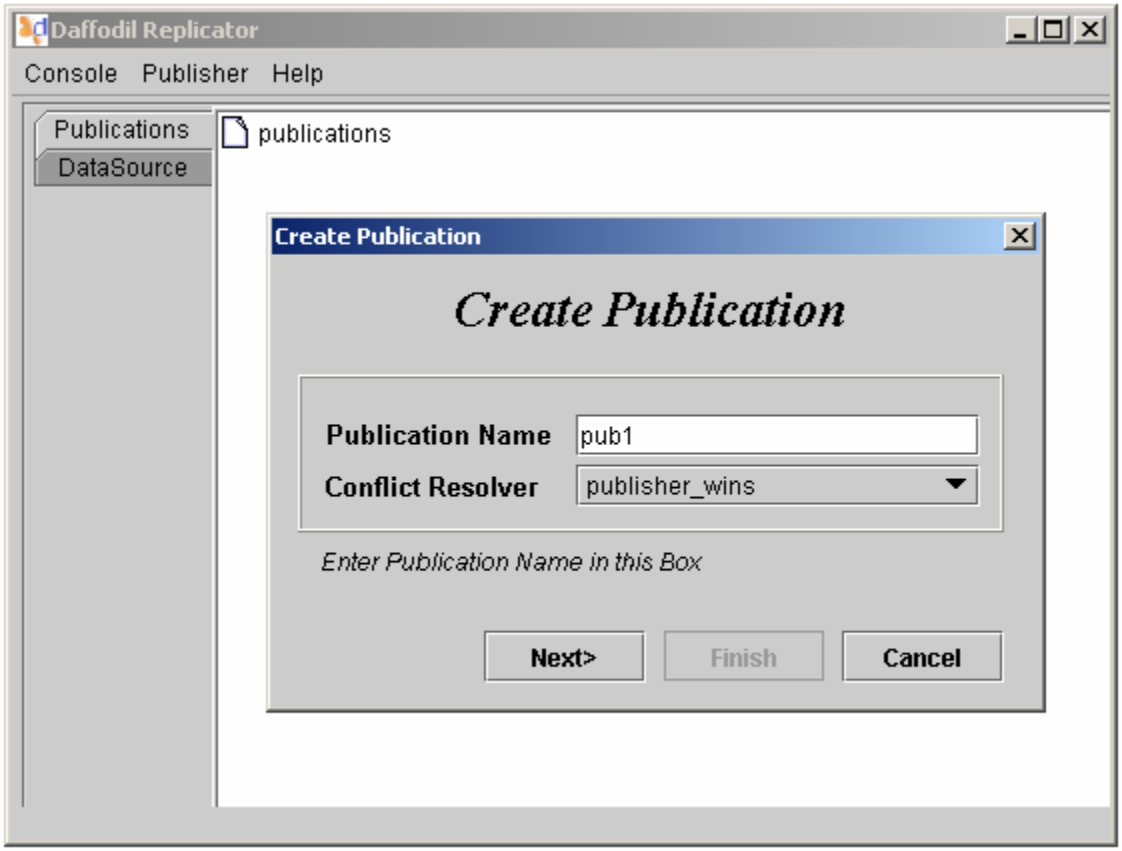
  Fill up the following fields in the dialog box

  - **Publication Name:** Enter a name for the Publication you are creating.
  - **Conflict Resolver:** Conflict Resolver helps to resolve the conflicts that may occur between Publisher and Subscriber. They are of two types: Publisher_wins and Subscriber_wins. Select your choice from the combo box.

- **Click** on **Next**

These steps are illustrated in the following screen shot:





Selecting tables to be included in Publication

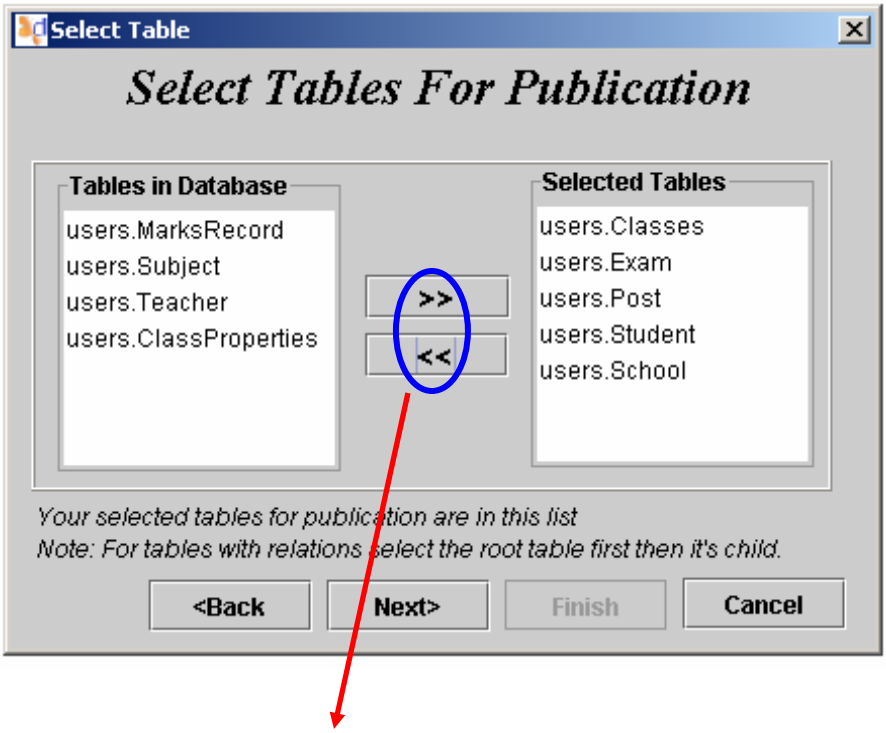
Upon clicking **Next** in the previous step, a **'Select Tables for Publication'** dialog box appears. The existing tables in the selected database are shown on the left pane.

- Select the tables which are to be included in the publication with the help of the forward arrow button. (You can deselect the selected tables with the help of backward arrow button)

Note: *For tables with relations, select the root table first and then its child. Further, Daffodil Replicator doesn't allow users to publish individual tables that are related. All the related fields should be published together.*

- **Click** on **Next**

These steps are illustrated in the following screen shot.

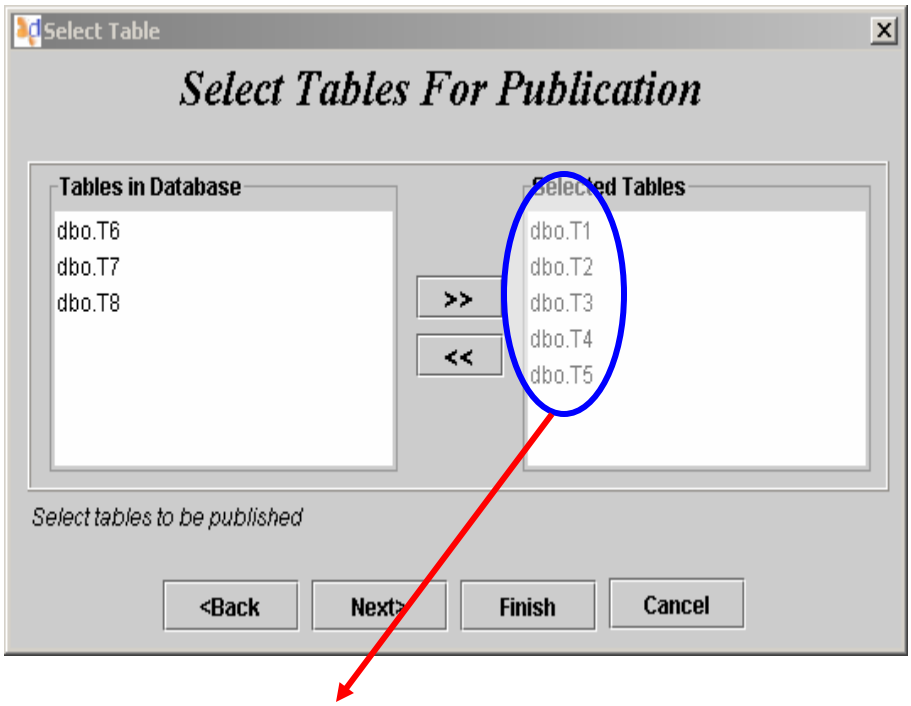


By using these two buttons, we can add and remove the item(s) from the two list boxes "Tables in Database" and "Selected Tables "

Selecting Cyclic tables to be included in Publication

Upon clicking **Next** in the previous step, a **'Select Tables for Publication'** dialog box appears. The existing tables in the selected database are shown on the left pane.

- Select the cyclic tables which are to be included in the publication with the help of the forward arrow button. (You can deselect the selected tables with the help of backward arrow button)



These are the selected cyclic tables for publication.

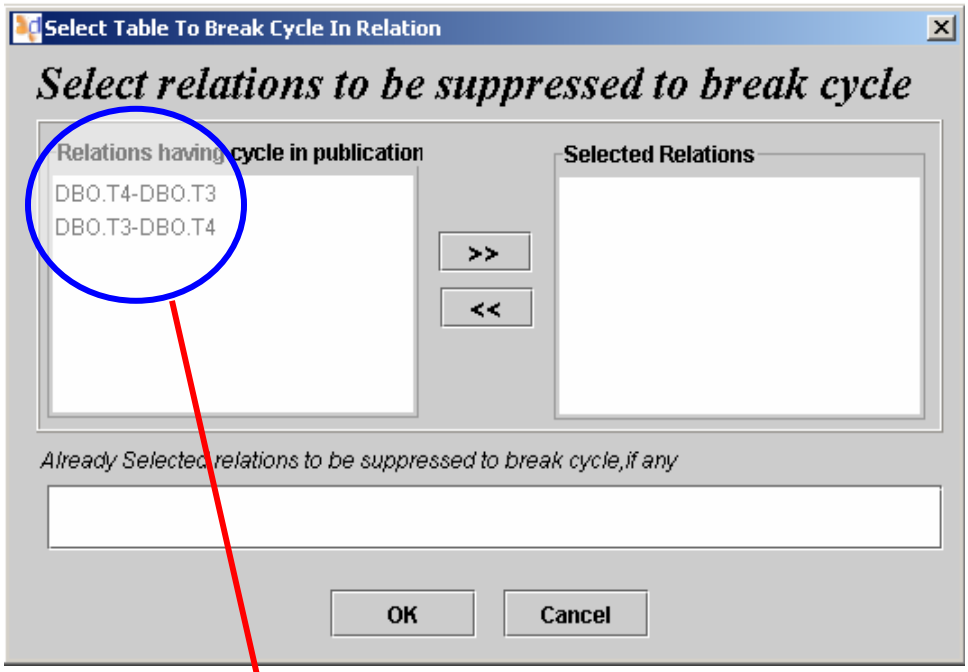
Here tables T1, T2, T3, T4 and T5 are cyclic tables. So when you click on the **Next** button for making publications, you get the following screen shot.

Upon clicking on **Next** Button an error message box appears. You should give the relation of tables which is to be published. If you want continue with cyclic tables click on **Yes** button otherwise on **No** button.

Select relations to be suppressed to break cycle

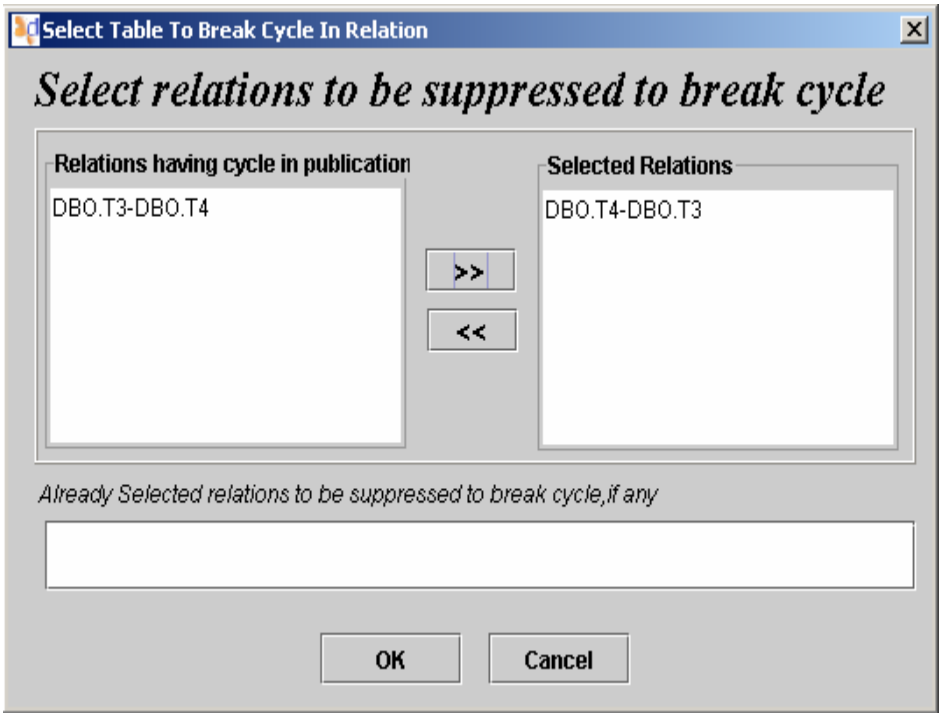
Upon clicking **yes** Button in the previous step, a **'Select relations to be suppressed to break cycle'** dialog box appears. The existing relations in the selected tables are shown on the left pane.

- Select the relation between tables which are to be included in the publication with the help of the forward arrow button. (You can deselect the selected tables with the help of backward arrow button)

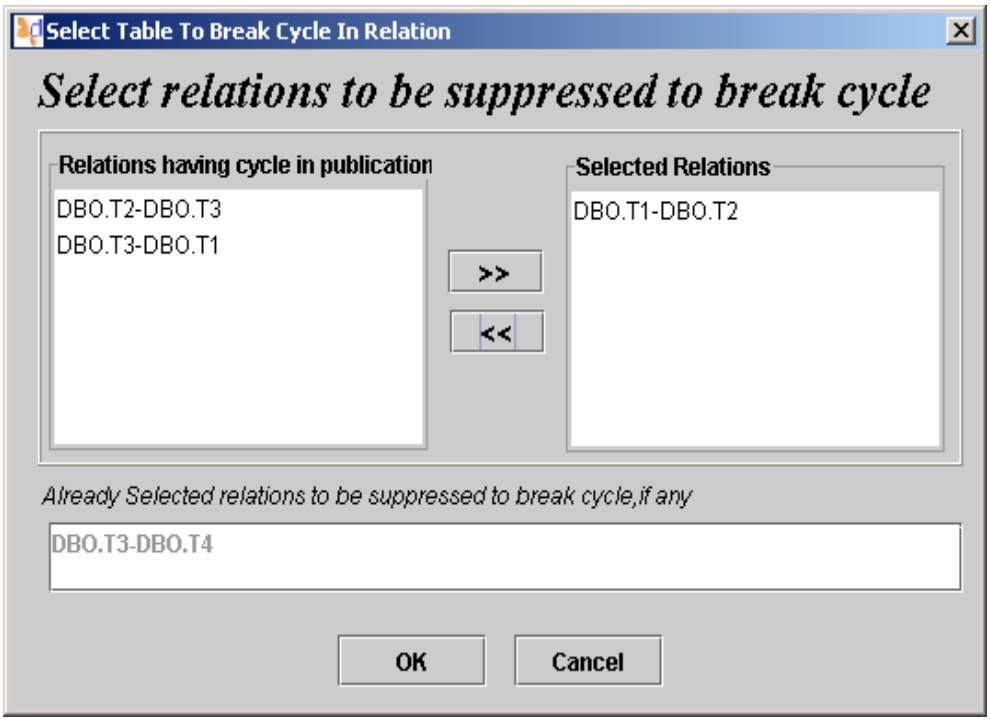


These are the relations which are to be suppressed.

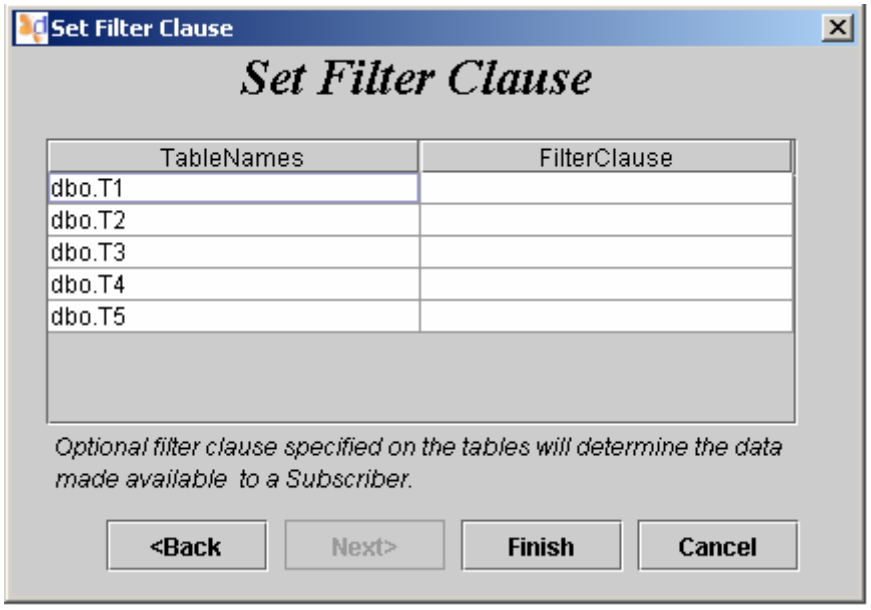
Upon selecting the relation from left pane from previous step you get the following screen shot:



Upon clicking **Ok** button in the previous step, again **'Select relations to be suppressed to break cycle'** dialog box appears if more than one relation is to be suppressed. This process will repeat until all the relation is to be suppressed.



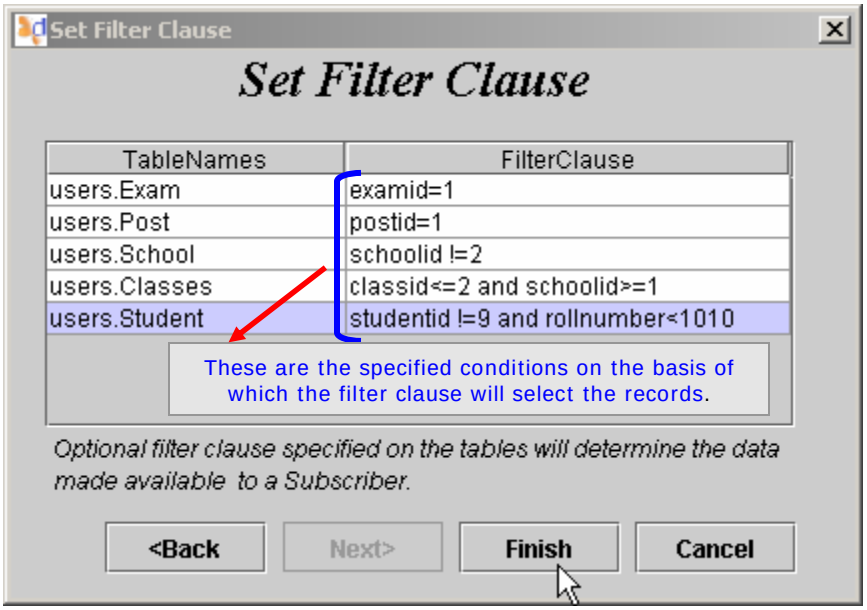
Upon clicking **Ok** Button in the previous step, a ' Set Filter Clause' dialogue box appears as



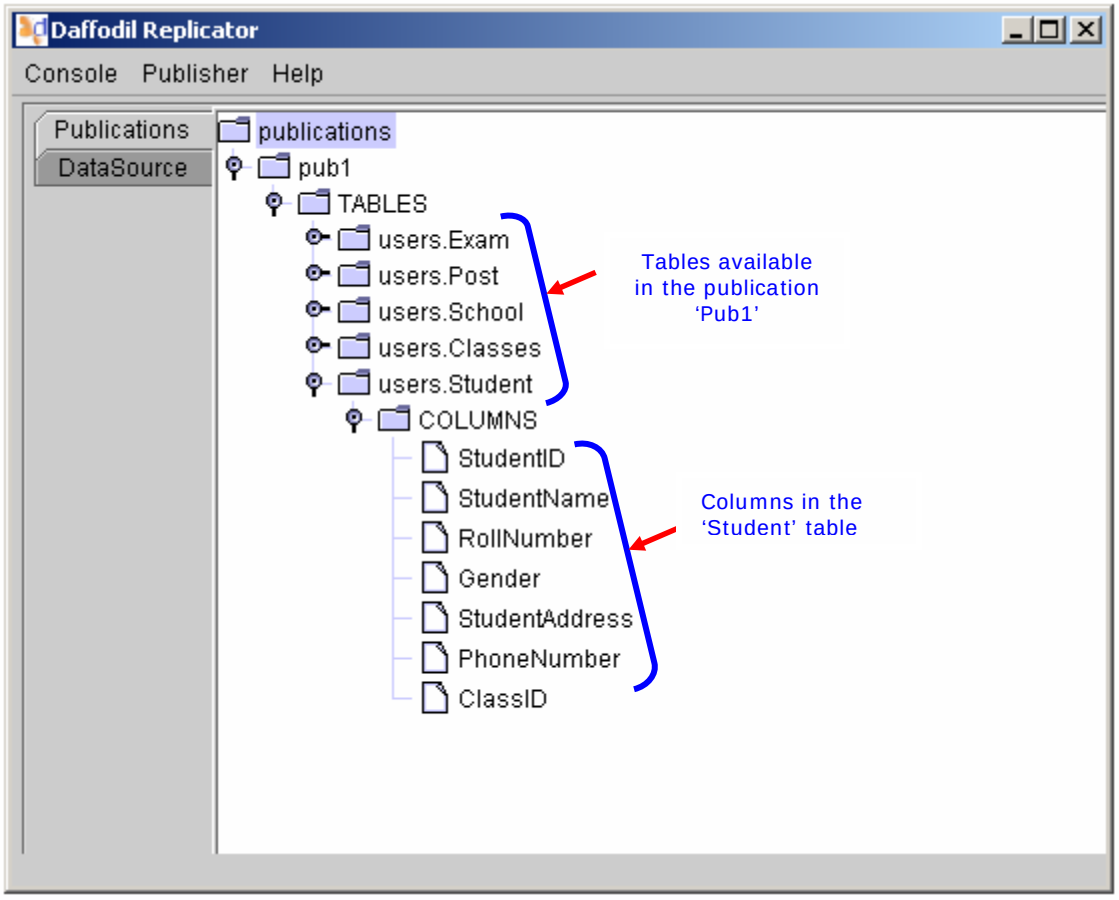
Applying Filters on Publication

Upon clicking **Next** in the previous step, a 'Set Filter Clause' dialog box appears. Set Filter Clause allows filtering in selected tables by defining conditions (e.g. classId > 200). This facility helps you to transmit only the specific data that is required at the Subscriber's end.

- Users should make sure that filters should refer to the names of tables and columns in the Publication. When Subscribers replicate data from a Publication, all the filters set on the Publication are combined and applied to the tables, comprehensively.



- Set filters for tables as shown in the above screen shot.
- After setting the filters, **click** on **Finish.**
- On successful publishing, you get a message "**Published Successfully**". Now you can subscribe this publication on your client database.
- On clicking the **Publication** panel, a user can view the Publication which he/she has created with all the tables and columns. In the following screen shot, user can view the details of the **Publication "pub1"** that we have created just now.

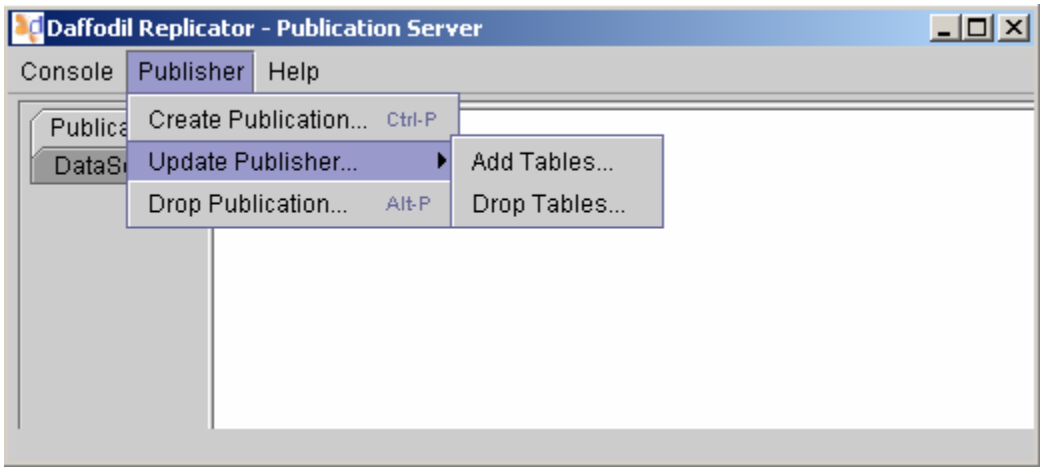

Daffodil Replicator
Console
Publisher
Help
Publications
DataSource
publications
pub1
TABLES
users.Exam
users.Post
users.School
users.Classes
users.Student
COLUMNS
StudentID
StudentName
RollNumber
Gender
StudentAddress
PhoneNumber
ClassID
Tables available in the publication 'Pub1'
Columns in the 'Student' table

Updating an existing Publication

Sometimes a user needs to add /remove some table(s) in the existing publication. **"Update Publisher"** option does just that.

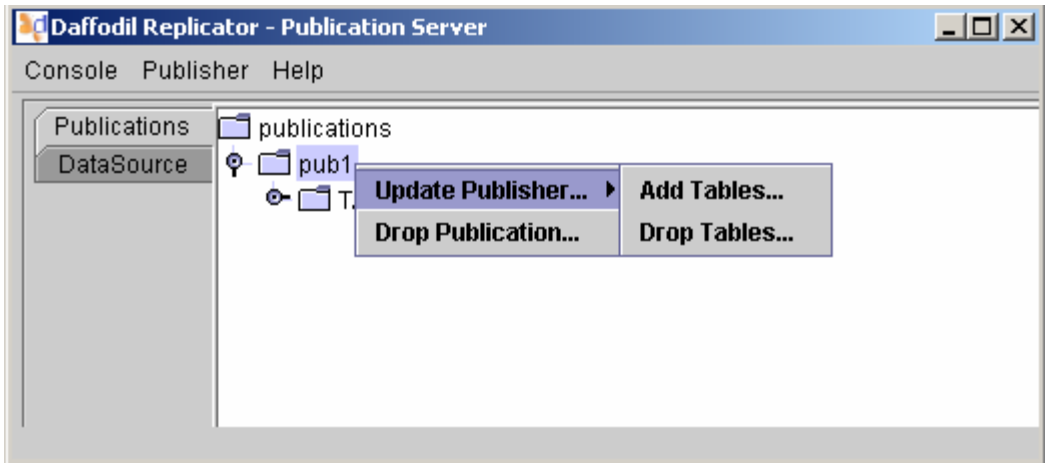
Follow the below steps to Update a publication.

- **Click** on the Menu option **Publisher-> Update Publisher,** as shown in the dialog box below.



Or

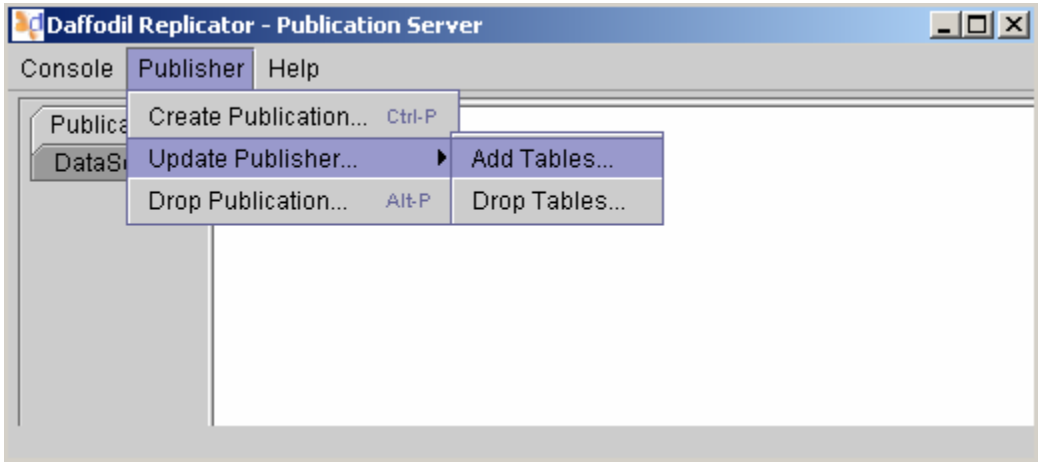
- **Right Click** on any existing publisher and choose **'Update Publisher'**.



Add Tables

Now for adding a new table in the existing publication,

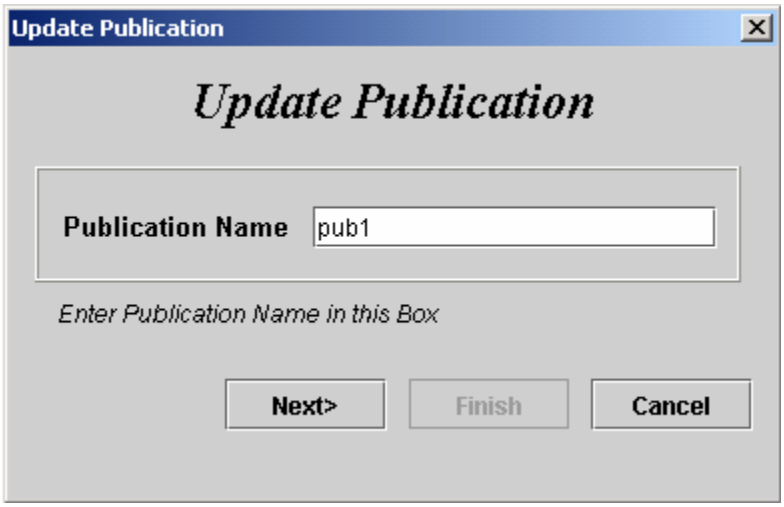
- **Click** on **'Add Tables'** as shown in the below dialog box.



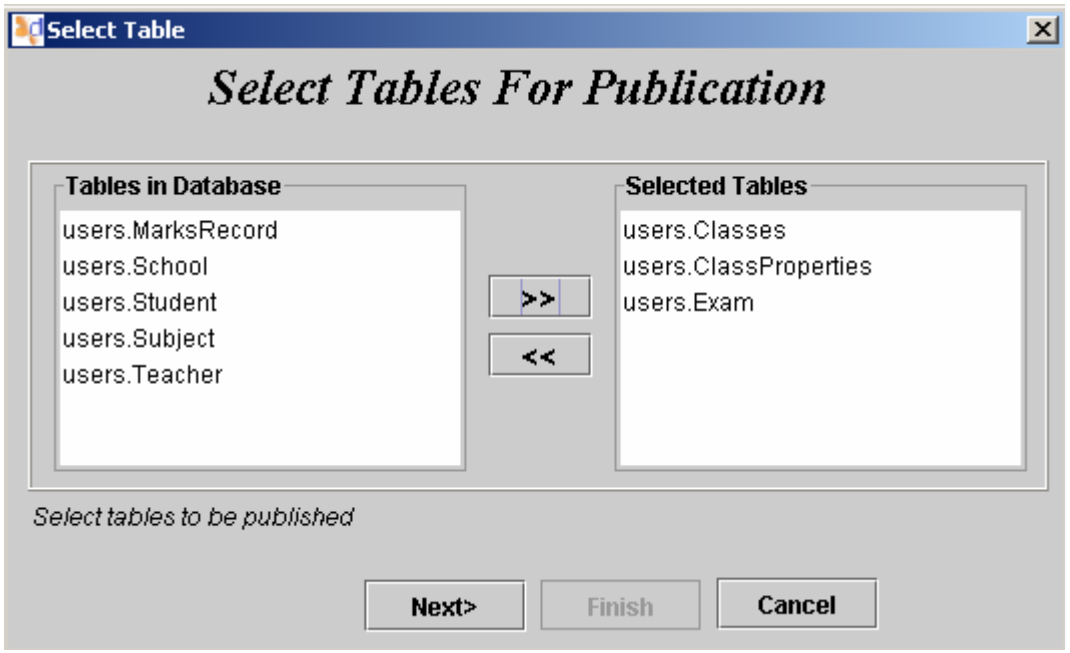
Another dialog box named **'Update Publication'** appears.

Here the user will have to specify the publication name to which he wants to add the new table.

Now **Click** on the button '**Next'**

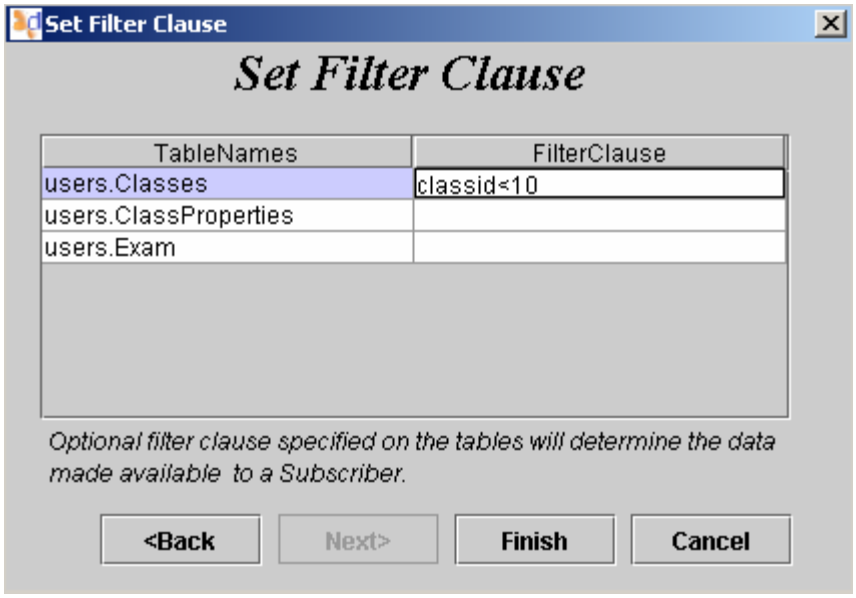


Another dialog box **'*Selecting for Publication*'** appears as shown below.



Select tables to be added to the publisher. Then **click** on '**Next**'.

A dialog box **'*Applying Filters on Publication*'** appears.



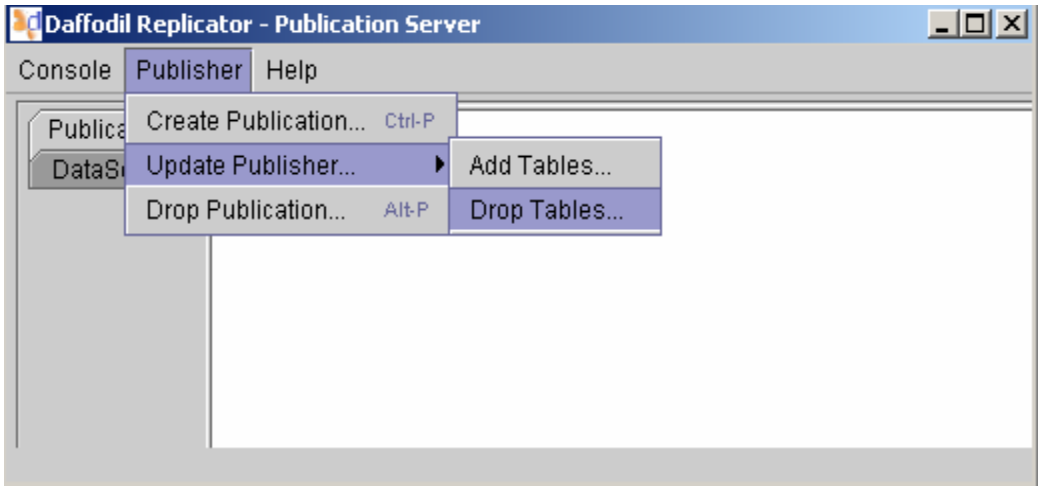
After applying filters, click on **Finish**.

In case of success, a dialog box **'Publication updated successfully'** will appear.

Drop Tables

We can also drop a table from the existing publication. Follow the steps given below.

- **Click** on **Publisher-> Update Publisher-> Drop Table**



Another dialog box named **'Update Publication'** appears. Here the user will have to specify the publication name to which he wants to add the new table.

Then click on **Next.**

Another dialog box **'Selecting tables to be included in Publication'** appears. Select the tables to be dropped from the publisher. Then click on Finish.

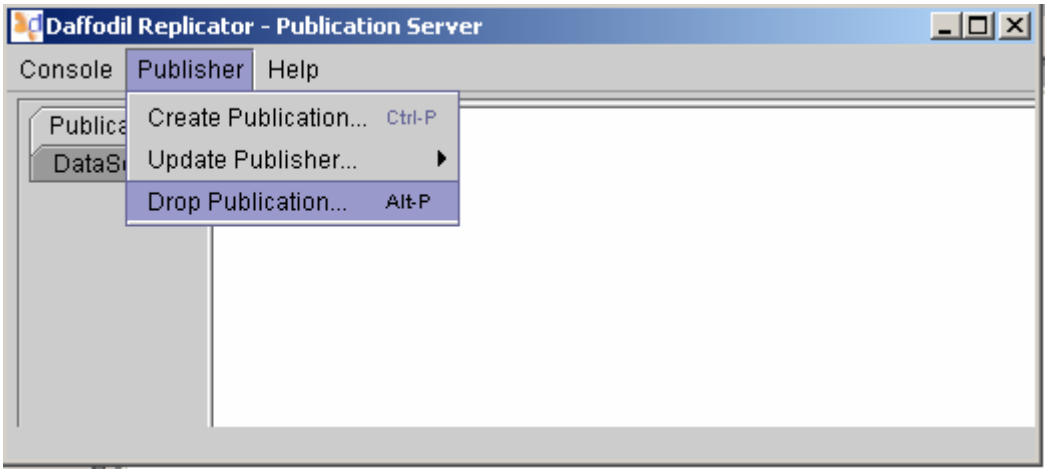
Dropping a Publication

A Publication can be dropped **only** if all the Subscriptions subscribed to it are dropped/unsubscribed, otherwise you will get the following error:

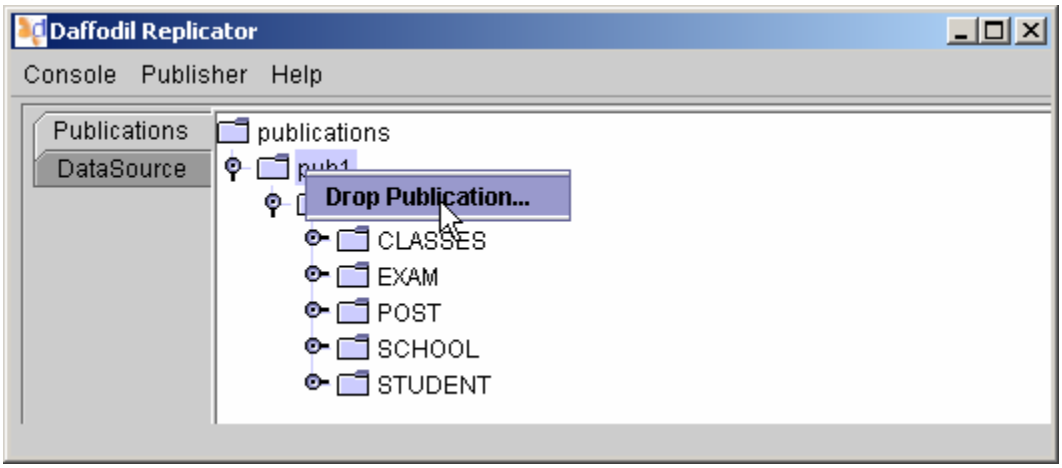
Can not unpublish the publication, one or more subscription(s) has subscribed to publication 'pub1'

Once all the Subscriptions have been dropped, follow the listed steps:

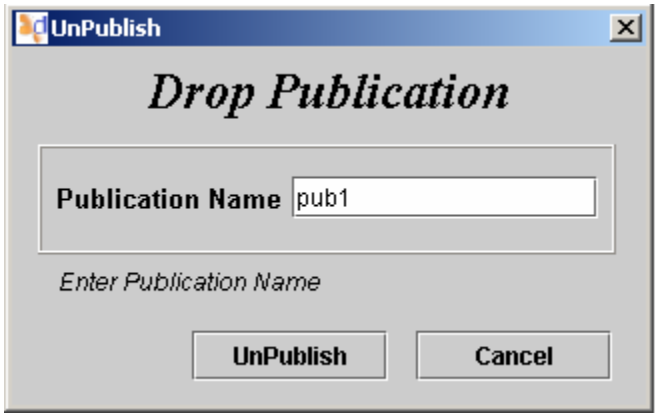
- From the **Publisher** menu, select **Drop Publication** (or Press ALT+P) OR **right click** on subscription name and **click** on **Drop Publication**



OR



- **'Drop Publication'** dialog box will appear as shown below:



- Enter **Publication Name** to be dropped and **click** on **UnPublish.**
- On success, you get a message "**Unpublished Successfully**"

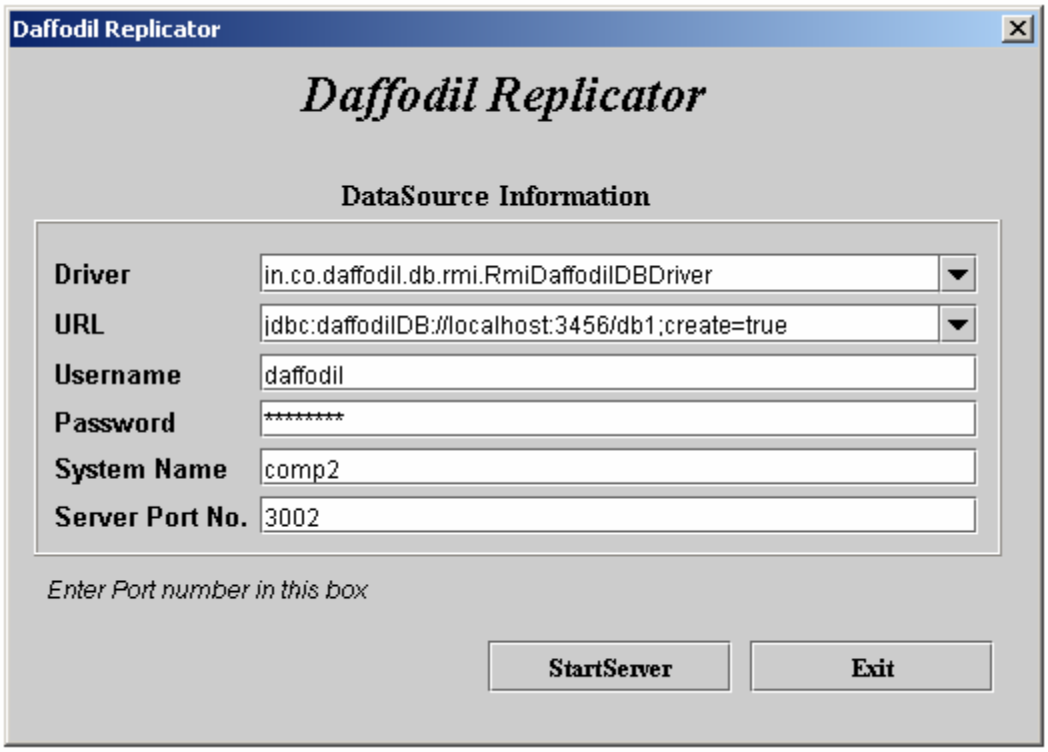
Note: *Even after dropping a Publication, the tables created by the Publication in the database will remain. You have to manually delete them.*

Starting the Subscription Server

Before starting Subscription Server, ensure that Publication Server is running.

- To start Subscription Server on your system, run the file SubServer.bat available under the installation directory of Daffodil Replicator and a dialog box appears.
- Fill up the following DataSource Information in the dialog box as shown in the following screen shot.

Note: *In the examples below, the default values are for Daffodil DB database. Users can change it according to their database.*



- **Driver** – This is the name of the driver with which you wish to connect to the database.
  Example: in.co.daffodil.db.rmi.RmiDaffodilDBDriver (For Daffodil DB Network mode)
  in.co.daffodil.db.jdbc.DaffodilDBDriver (For Daffodil DB Embedded Edition)
- **URL** – Specify the URL of the database. Optional "create=true" command will create the database, if it does not exist.
  Example: jdbc: daffodilDB: //localhost:3456/<database name>; create=true (For Network mode)

Jdbc: DaffodilDBembedded :< database name>; create=true (For Embedded mode)

- **Username** – Enter a valid username for the specified database. Default username of Daffodil DB is daffodil

- **Password** – Enter the password of the above specified user. Default password of Daffodil DB is daffodil

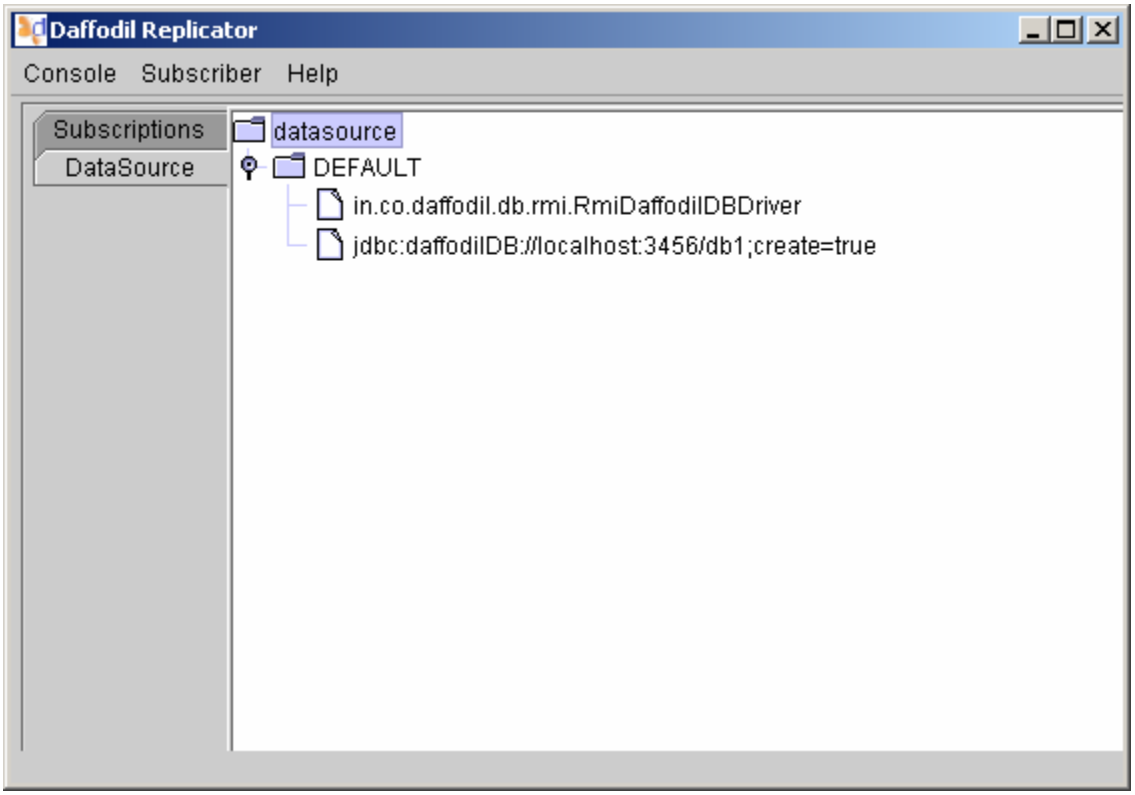
- **System Name** – Specify the name or network IP of the local machine.

- **Server Port No.** – Specify the port no. on which you wish to run Subscription Server. By default it is 3001 but you are free to change it to any available port.

  Note: *If you are running both Publication and Subscription servers on the same system, then the port numbers should be* ***different****.*

1. After filling up the DataSource Information, click on the **StartServer** button and the Subscription dialog box appears as shown below.

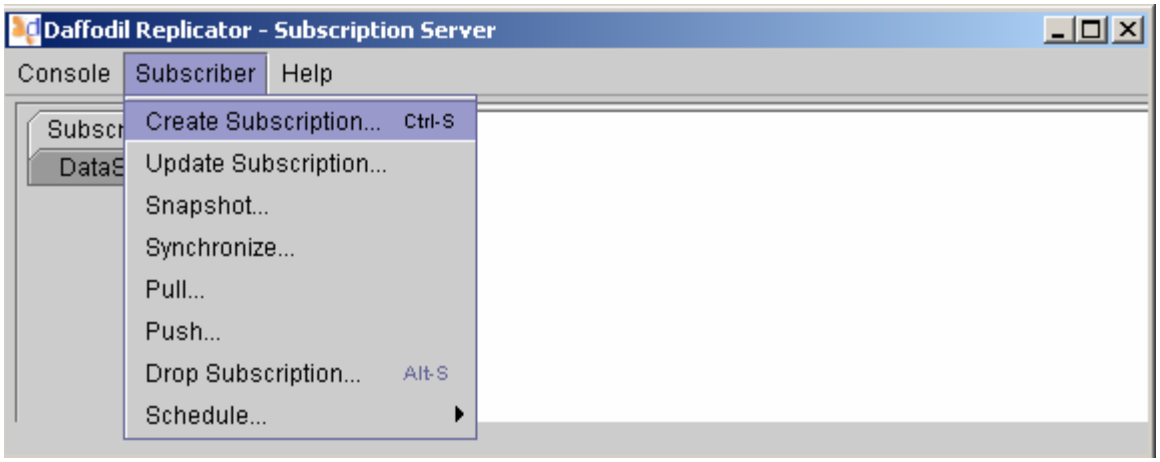
2. **Click** on the **DataSource** panel to view the details of the database to which you have established a connection.



Using the Subscription Wizard

Creating a new Subscription

To create a new Subscription, **click** the **Subscription** menu and select **Create Subscription** (or press CTRL+S).



A **Create Subscription** dialog box appears.

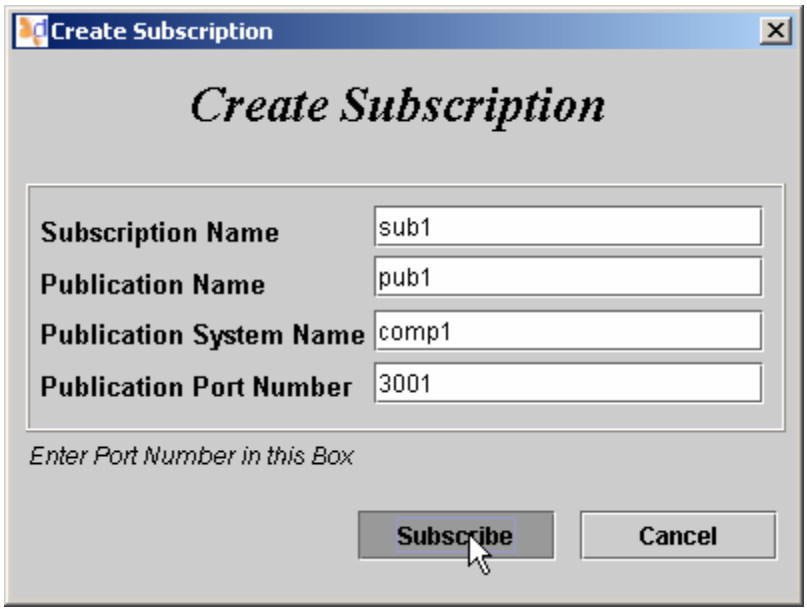
- Fill up the following fields in the dialog box as illustrated in the screen shot:

  - **Subscription Name** – Enter a name for the Subscription you are creating.

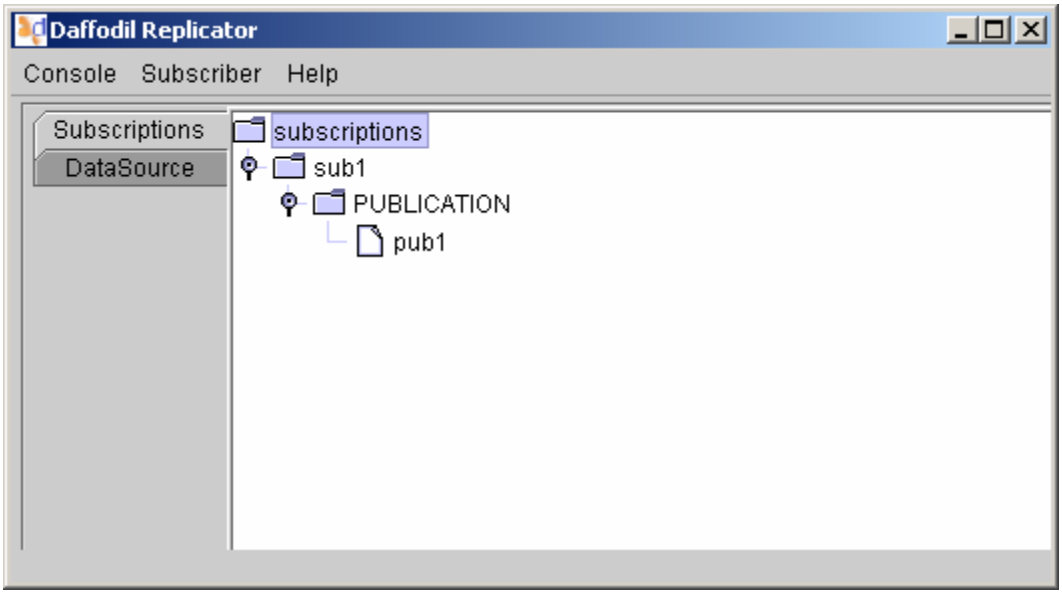
  - **Publication Name** – Enter a valid Publication Name that exists in the Publication Server. (Publication has to be created before you attempt Subscription.)

  - **Publication System Name** – Specify the name or IP of the system where Publication Server is running.

  - **Publication Port Number** – Specify the port where Publication Server is running.



- **Click** on **Subscribe**
- On successful Subscription, you get a message "**Subscription Created Successfully**"
- The created Subscription can be viewed by clicking on the Subscription panel as shown below:



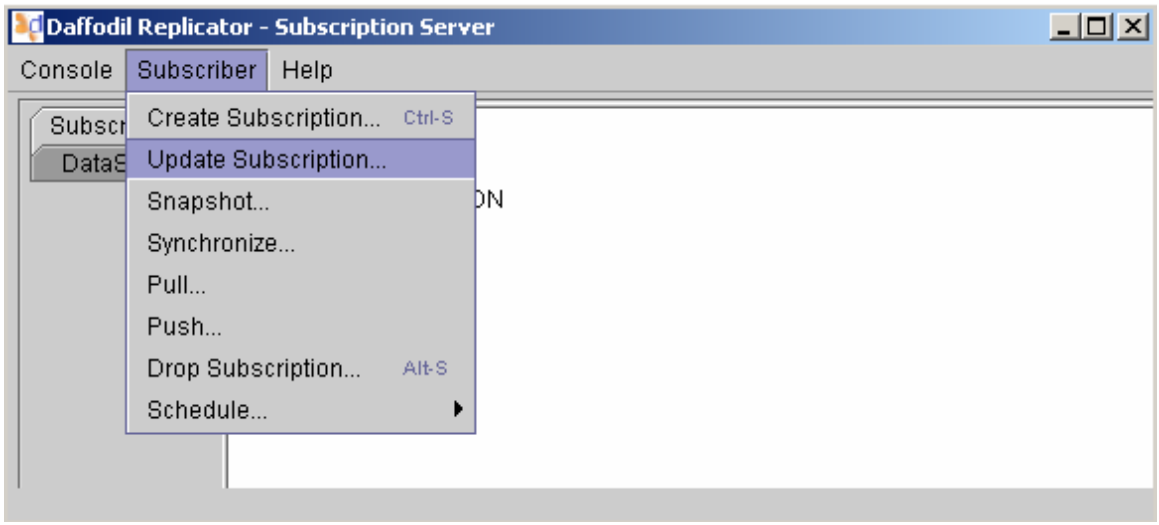
Note: *At this moment, empty tables will be created in Subscription corresponding to those present in Publication. Data will be inserted only after taking the snapshot.*

Updating a Subscription

Whenever publisher modifies his publication by adding or dropping tables from publication, subscriber must update its subscription .

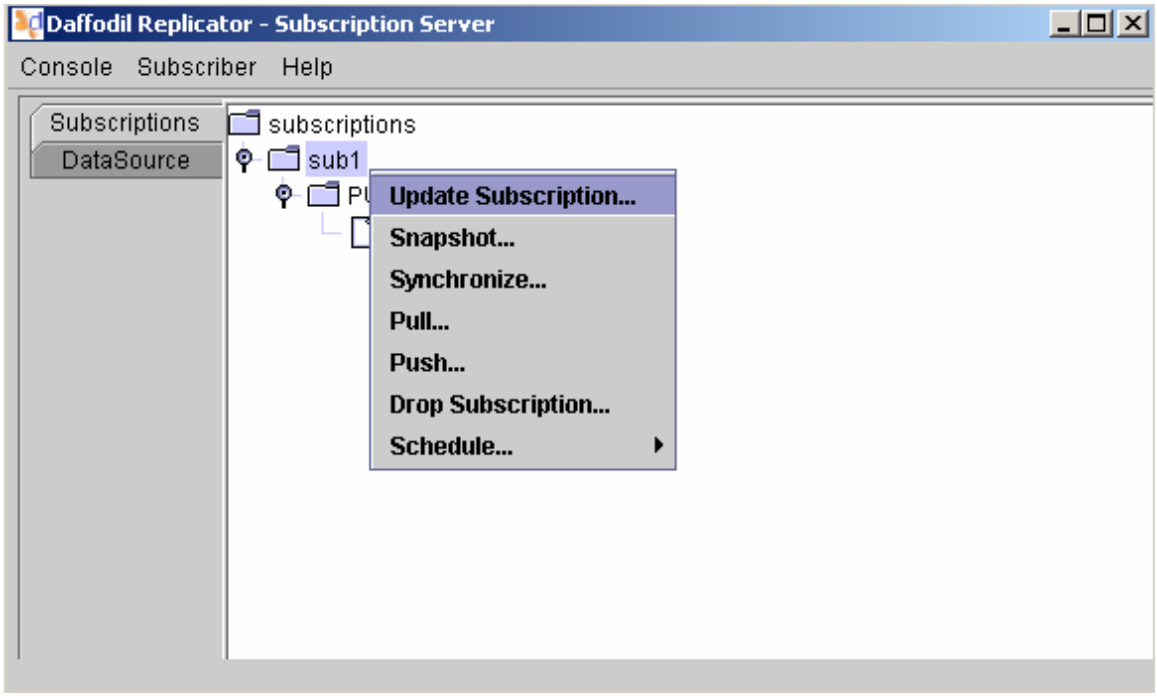
Follow the steps given below to do so.

- **Click** on the menu **Subscriber->Update Subscription**.

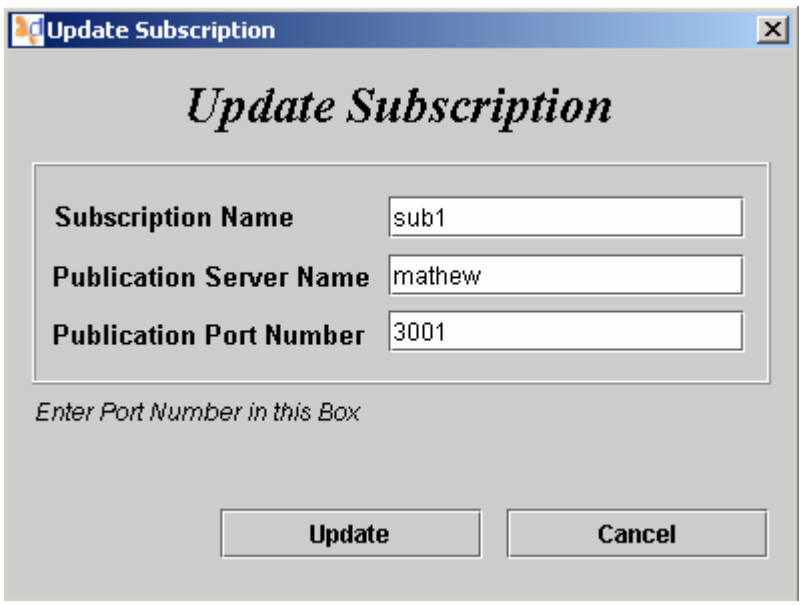


Or

- **Right Click** on the Subscription Name and **Click** on **Update Subscription.**



Following dialog bow will appear.



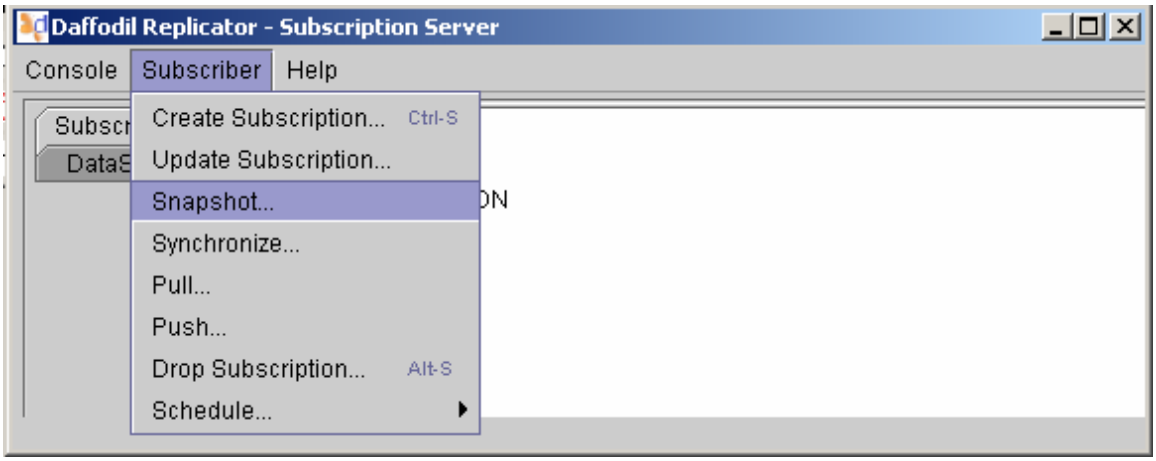
Here enter the name of the Subscription which you wish to update, name of the Publication server and its port no.

- Now **click** on the button **Update**.

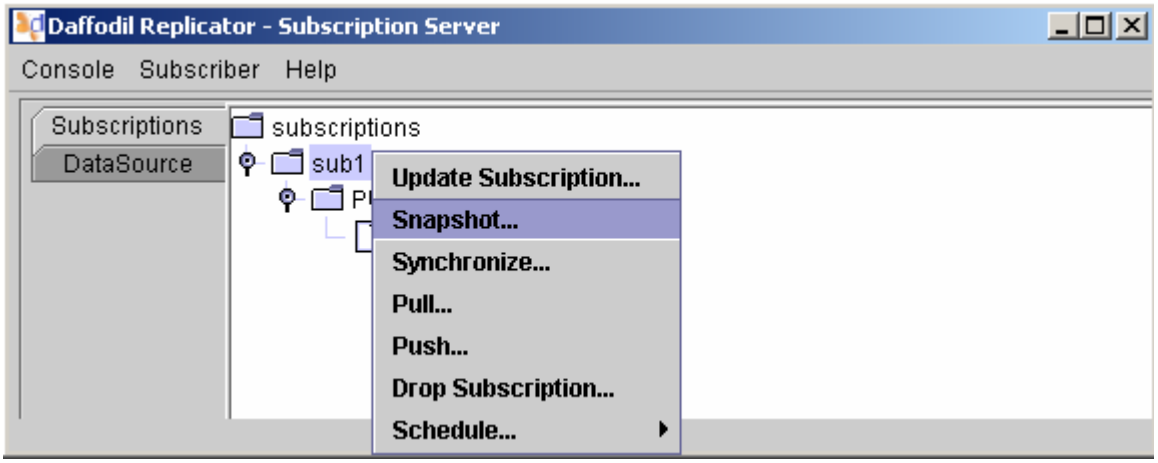
In case of success, user will get the message **'Subscription updated successfully'**.

Snapshot Replication

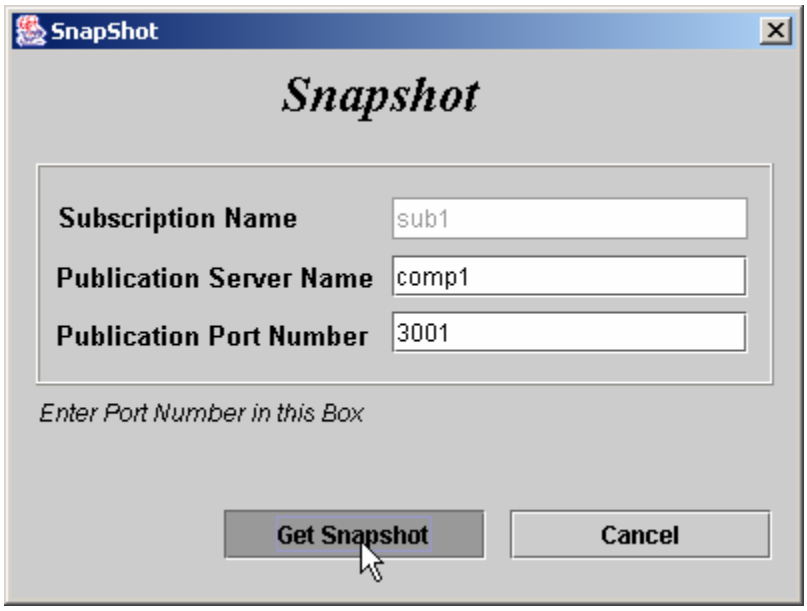
- To take the Snapshot of a Publication, **click** on **Subscriber** menu and select **Snapshot** OR Right click on the name of subscriber and **click** on **Snapshot.**



Or



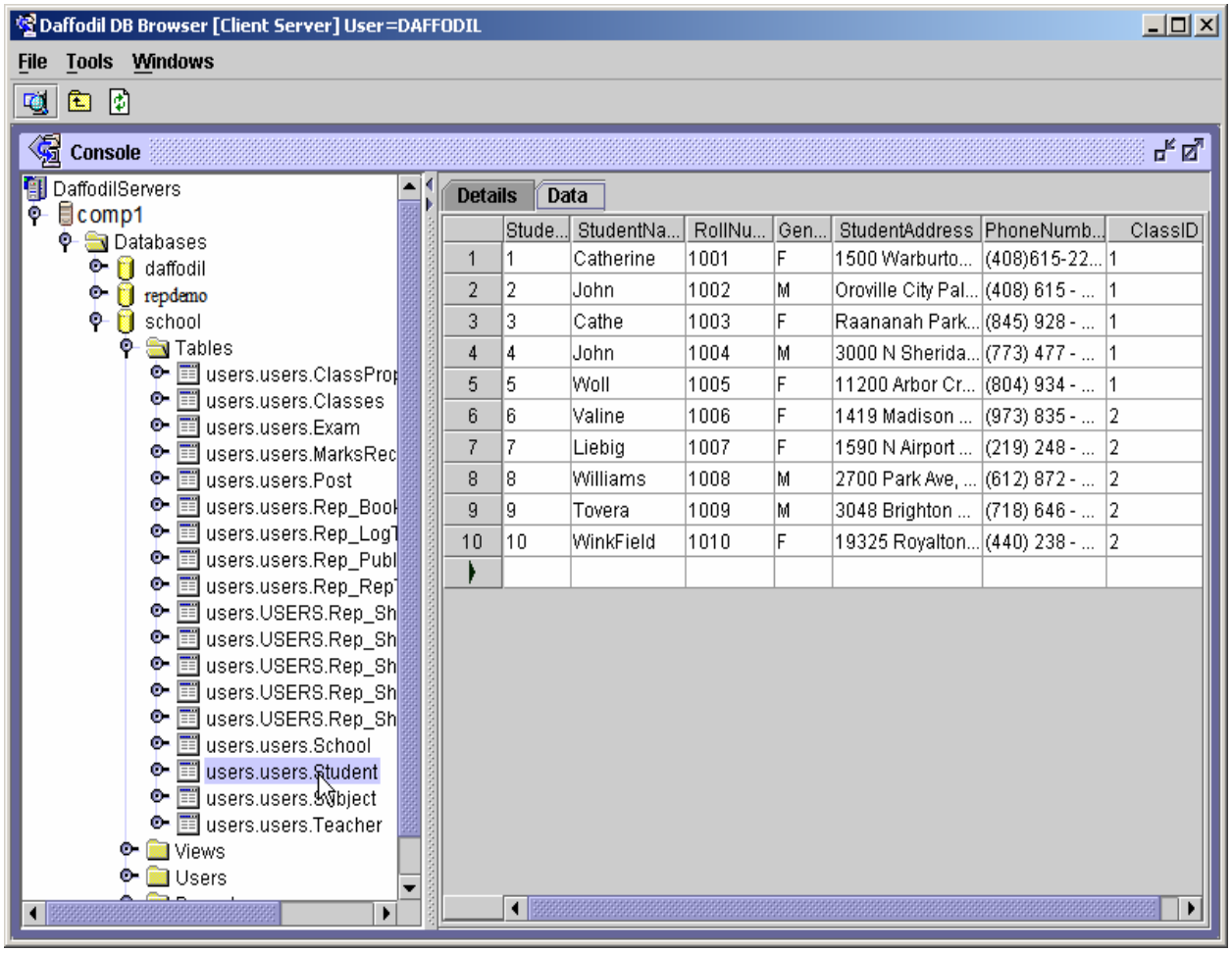
- A **Snapshot** dialog box appears.
- Fill up the following fields in the dialog box as shown below:
  - Enter Subscription name
  - Enter Publication Server Name
  - Enter Publication Port No



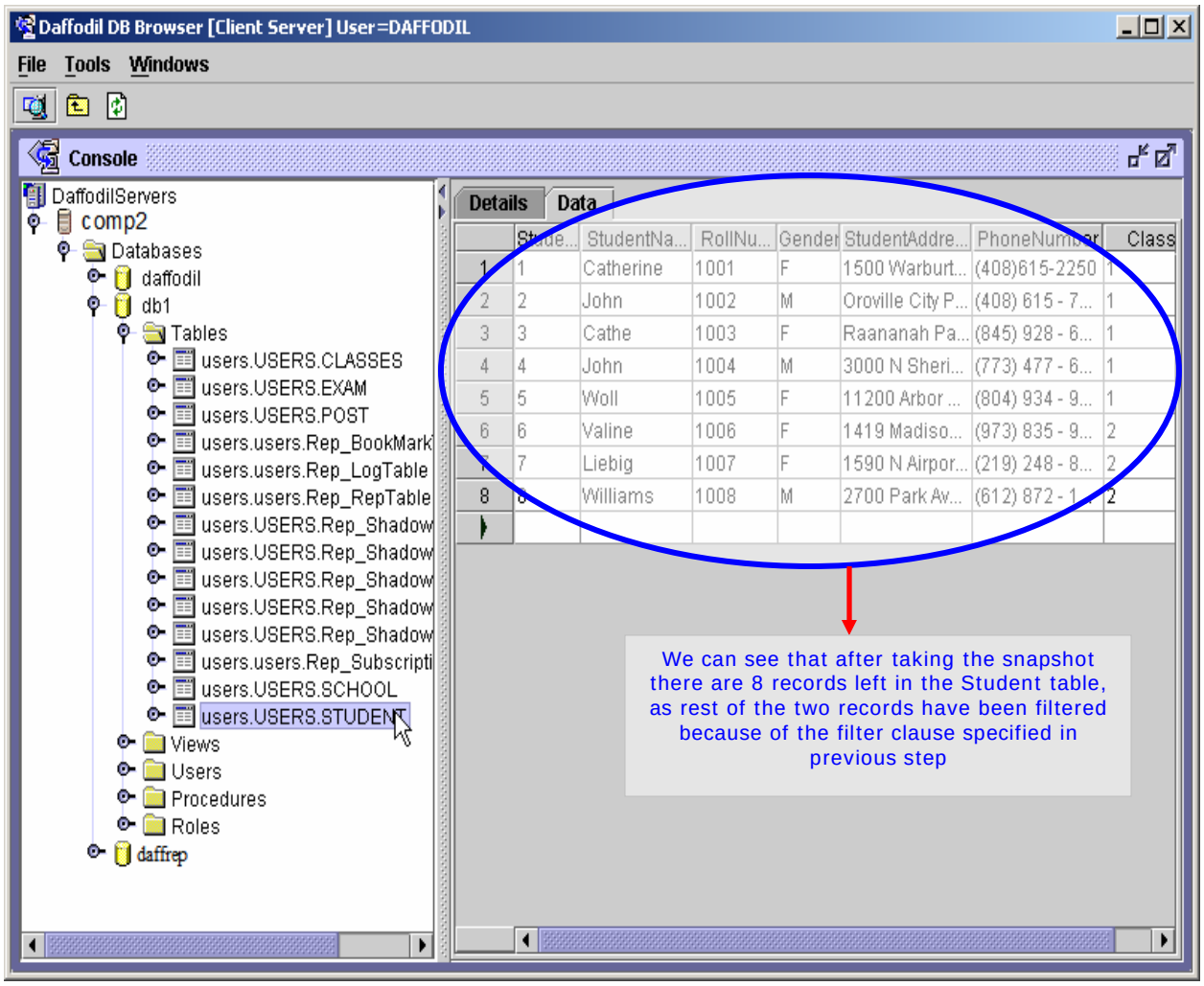
- **Click** on **Get Snapshot**
- On success, you will get a message "**Snapshot Taken Successfully**"

Comparing the data in Publisher database and Subscriber database after taking Snapshot

The following screen shot shows the data in Student table of Publication database (school).



After taking snapshot, the data in Student table of Subscription database (db1) is shown in the following screen shot:



Note that student table in client side contains only 8 records (10 in server) as the other 2 records has been filtered by the filter clause set during Publishing.

Merge Replication (Synchronization)

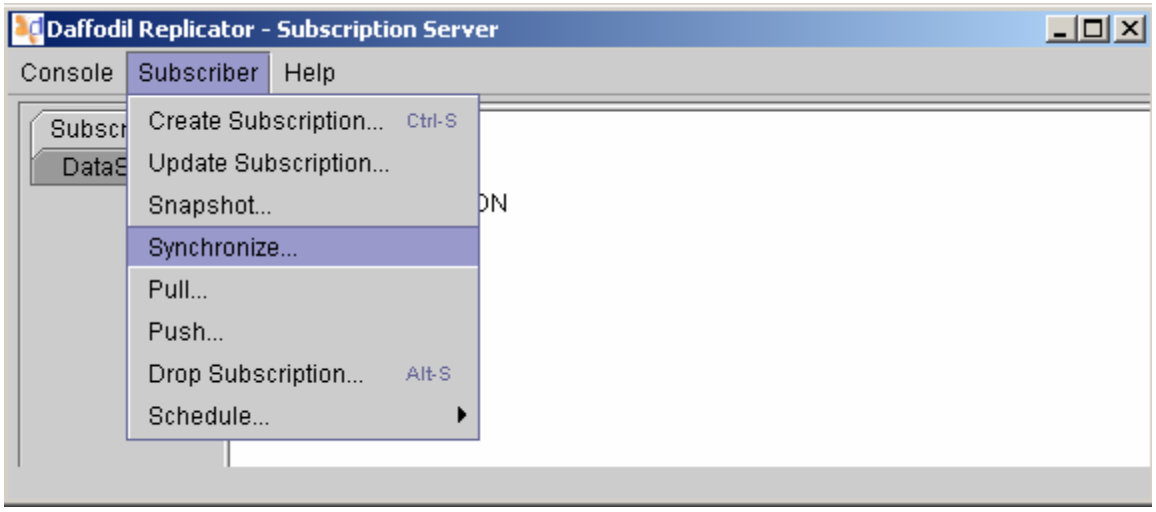
Synchronization compares Publishers and Subscribers data and the changes made to the data by both Publisher and the Subscriber are updated or merged.

Example: Updating the table Student in the Subscription database (db1)

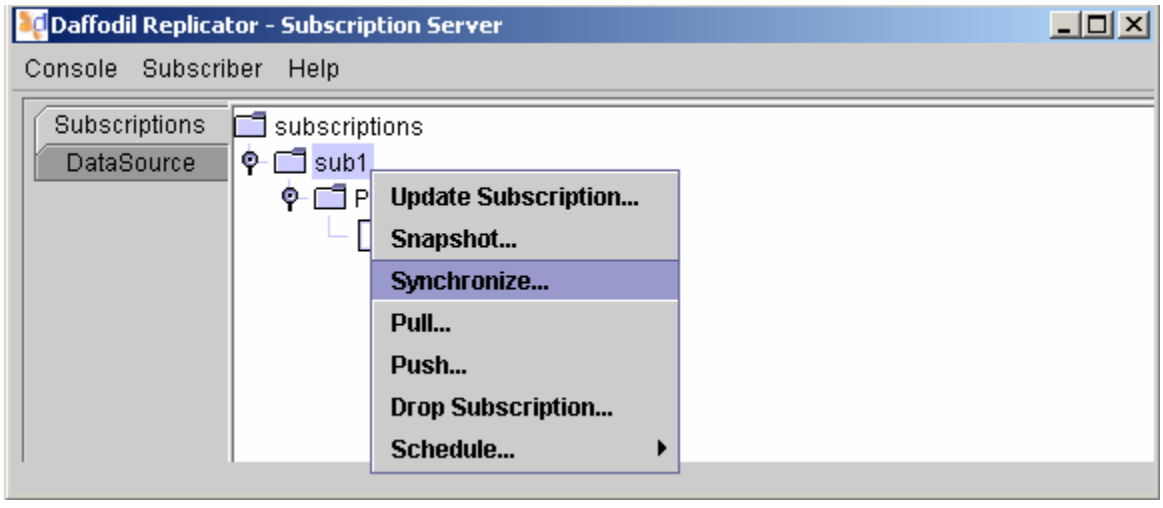
Query: Update Student set studentname='Adam' where StudentID=5

Now that the Subscriber database has updated, we want that data to be updated in Publisher database also. For this to happen, you need to synchronize the data between Publisher and Subscriber.

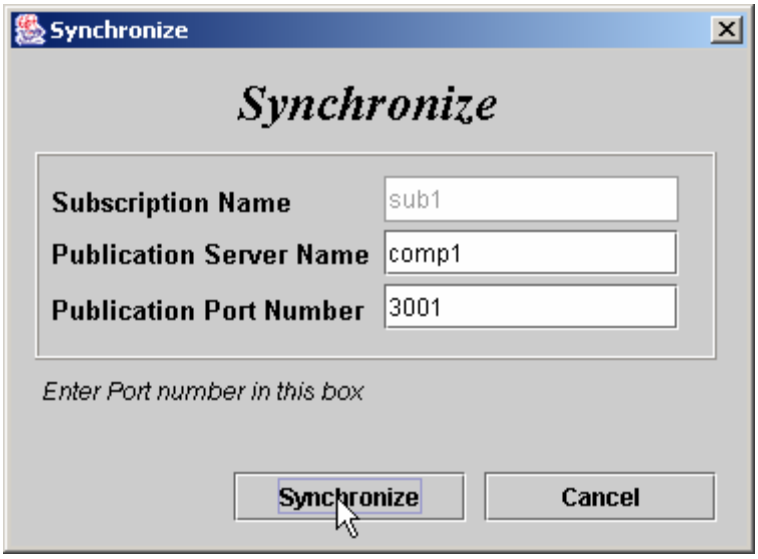
- To synchronize the data between Publication and Subscription, click on **Subscriber** menu and select **Synchronize** OR Right click on the name of subscriber and click on **Synchronize**.



Or



- A '**Synchronize'** dialog box appears.
- Fill up the following fields in the dialog box as shown below:
  - Enter **Subscription Name**
  - Enter **Publication Server Name**
  - Enter **Publication Port No**

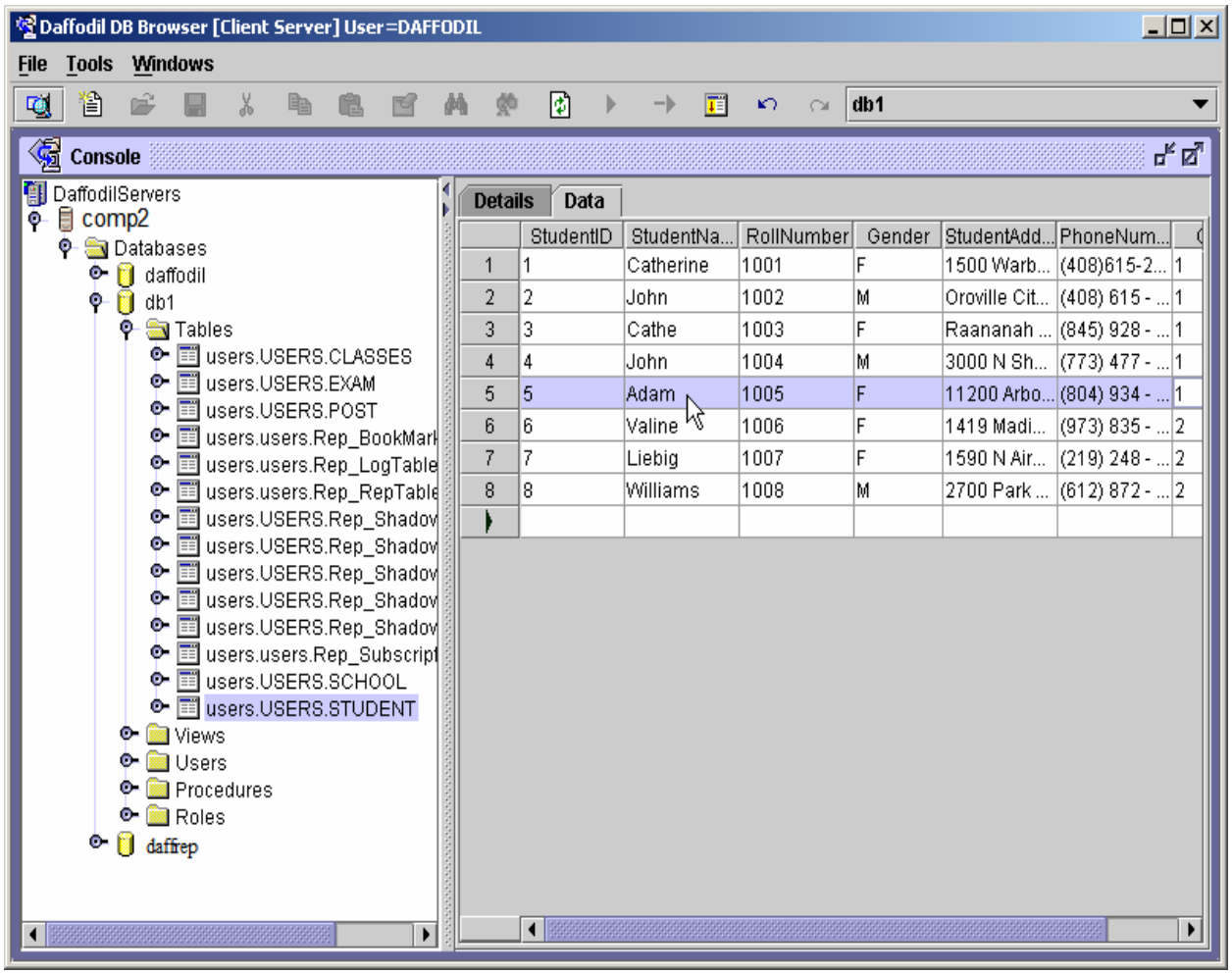


  - **Click** on **Synchronize**
  - On success, you get a message "**Synchronized Successfully**"

Note: *Publisher can't synchronize the data with Subscriber.*

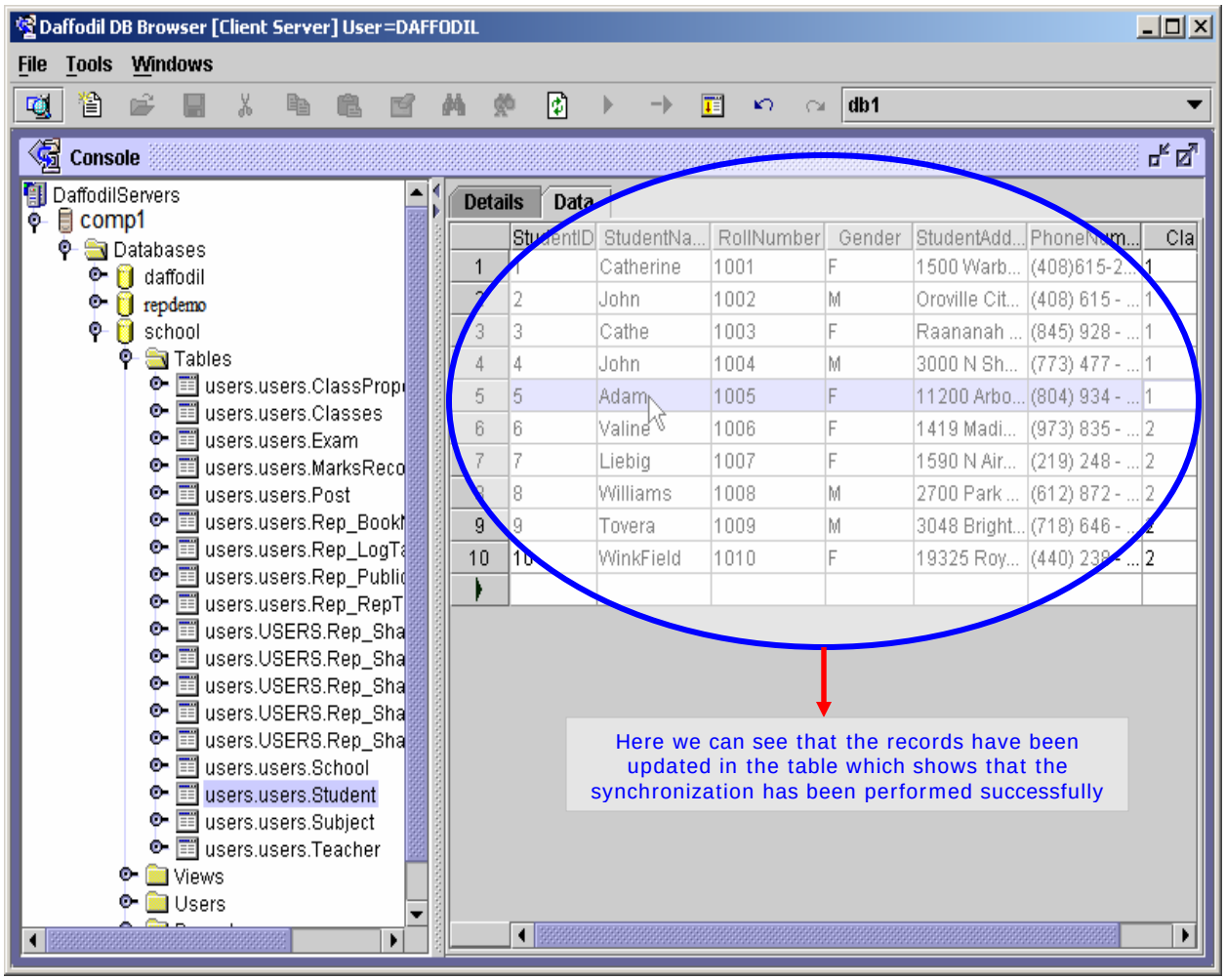
Comparing the data in Publisher database and Subscriber database

The data in Student table of Subscription database (db1) is shown in the following screen shot (after updating the student table):



Now perform synchronization. The updation done to Student table (in db1) by the Subscriber will be merged with the Student table of Publishers database (school).

The following screen shot shows the data in Student table of Publication database (school) after synchronization.



You can see that the record with StudentID=5 is updated in both the databases i.e. databases are synchronized successfully.

Pull Replication

Pull replication compares Publishers and Subscribers data and the changes made to the data by the Publisher are updated in the Subscriber's database. But the changes made by the Subscriber are not updated to the Publisher's database. So in Pull replication, records (that are changed due to insert, update and delete queries at the Publisher's database) are pulled from Publisher to Subscriber by the Subscriber.

Example: Insert a new record to the table *Student* in the Subscription database (db1)

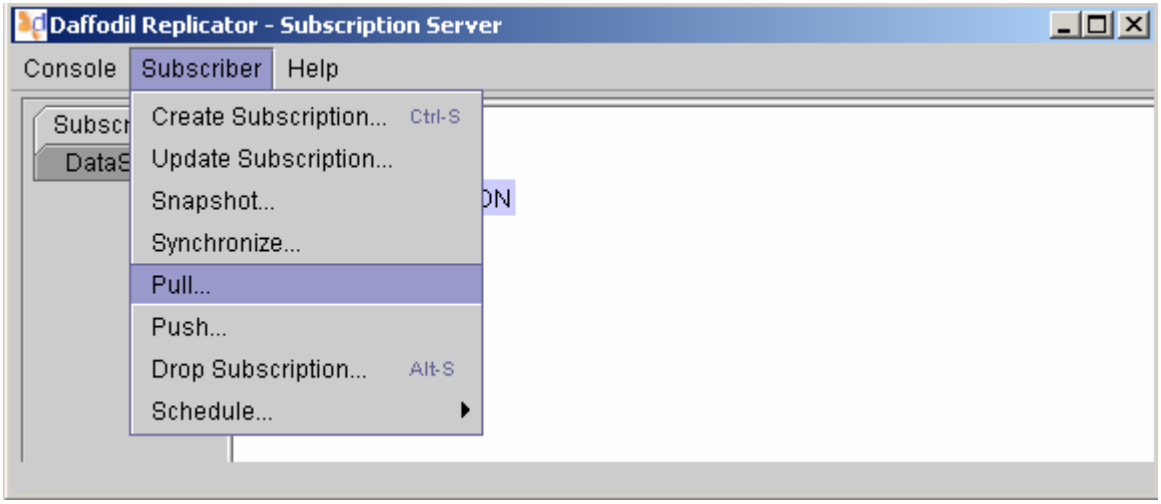
Query: Insert into Student values (11,'Mathew', 1000,'M','21-West Minister Square, Durham','(456)272-3454', 2)

Now update the table *Student* in the Publication database (school)

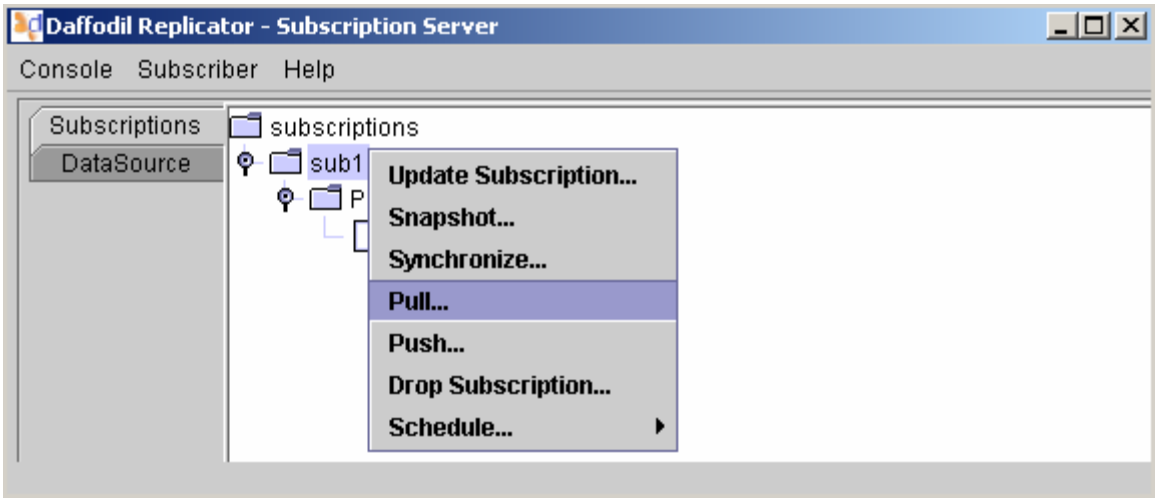
Query: Update Student set PhoneNumber='(971)679-2345' where StudentID=6

Now Subscriber wish to pull Publisher changes at Subscriber's end and at the same time does not want the changes at Subscriber's database to be reflected at Publisher's end. For this to happen, Subscriber initiates Pull replication.

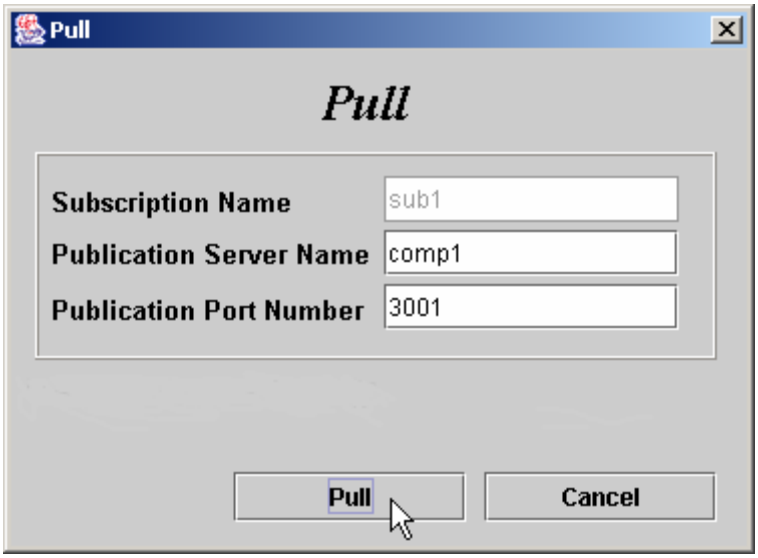
- To perform Pull replication, click on **Subscriber** menu and select **Pull** OR Right click on the name of subscriber and click on **Pull**.



Or



- A **Pull** dialog box appears.
- Fill up the following fields in the dialog box as shown below:
  - Enter **Subscription Name**
  - Enter **Publication Server Name**
  - Enter **Publication Port No**

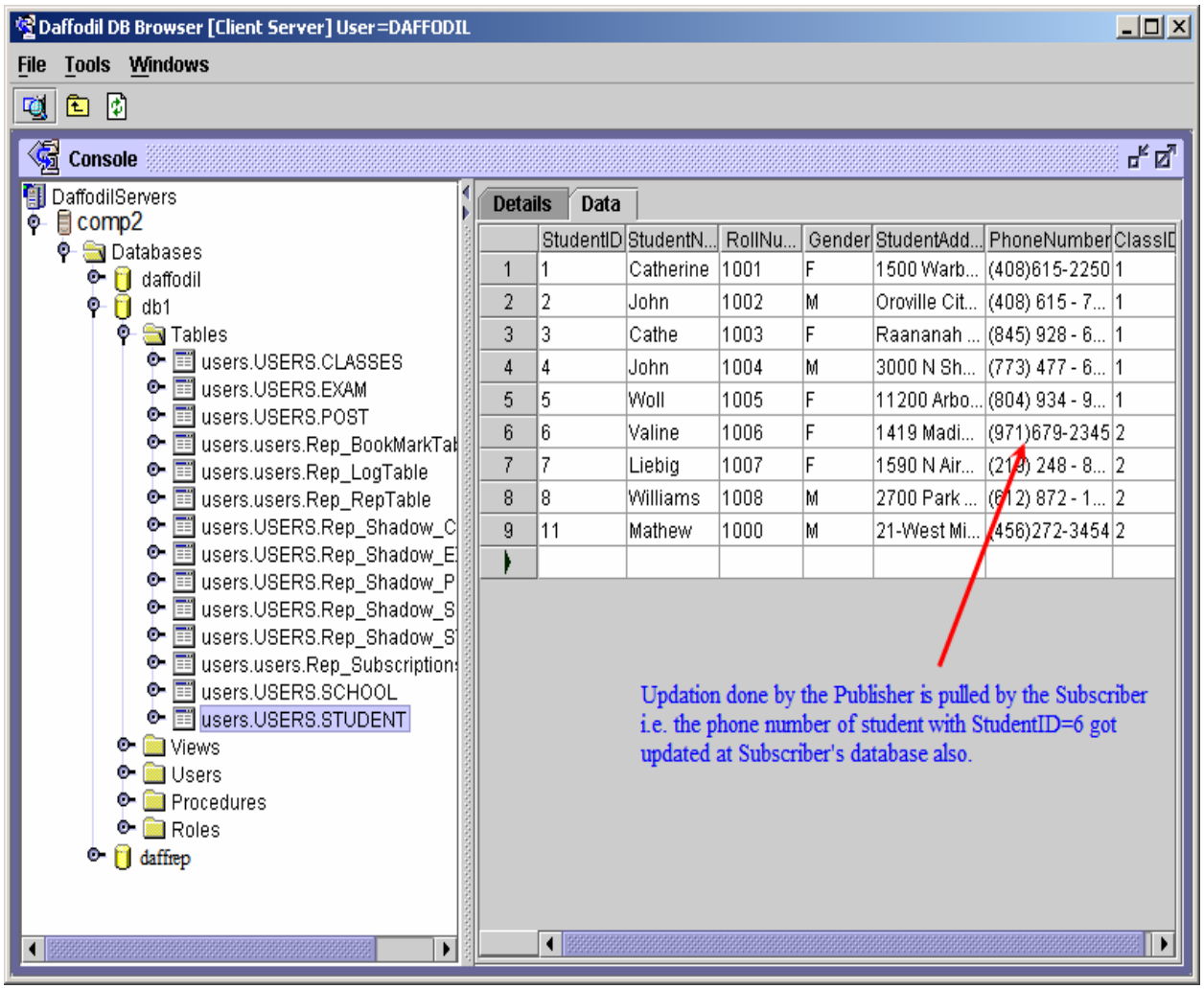


  - **Click** on **Pull**
  - On success, you get a message "**Pulled Successfully**"

Note: *Pull Replication can only be initiated by the Subscriber.*

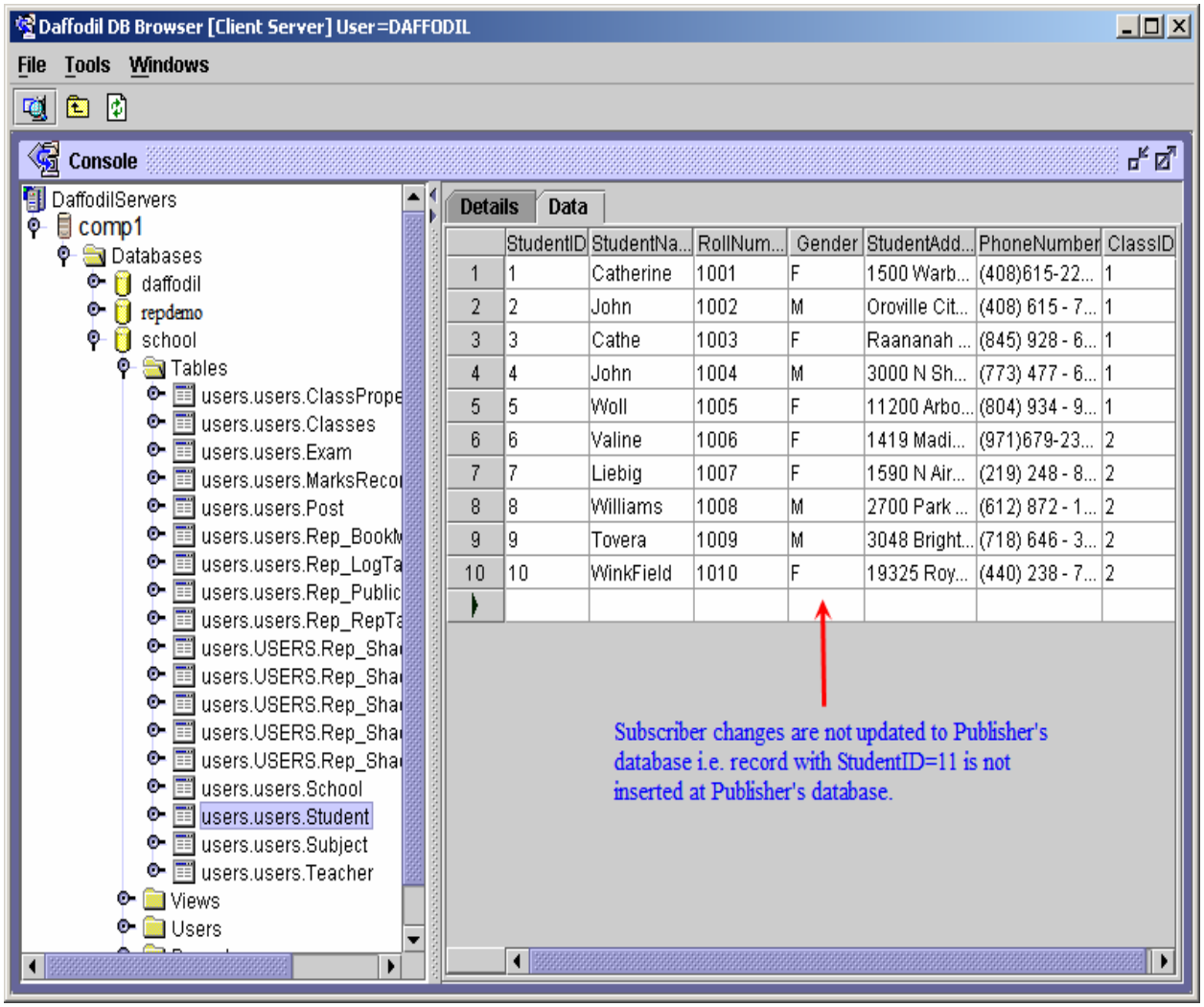
Comparing the data in Publisher database and Subscriber database

The data in *Student* table of Subscription database (db1) is shown in the following screen shot (after updating the student table and subsequently performing Pull replication):



The above screen shot shows that the updation done to *Student* table (in school database) by the Publisher is pulled by the Subscriber to its *Student* table (in db1 database).

The following screen shot shows the data in *Student* table of Publication database (school) after performing Pull replication:



You can see that the Subscriber changes are not reflected at Publisher's database.

Push Replication

Push replication compares Publisher's and Subscriber's data and the changes made to the data by the Subscriber are updated in the Publisher's database. But the changes made by the Publisher are not updated to the Subscriber's database. So in Push replication, records (that are changed due to insert, update and delete queries at Subscriber's end) are pushed to Publisher's database by the Subscriber.

Example: Insert a new record to the table *Student* in the Subscription database (db1)

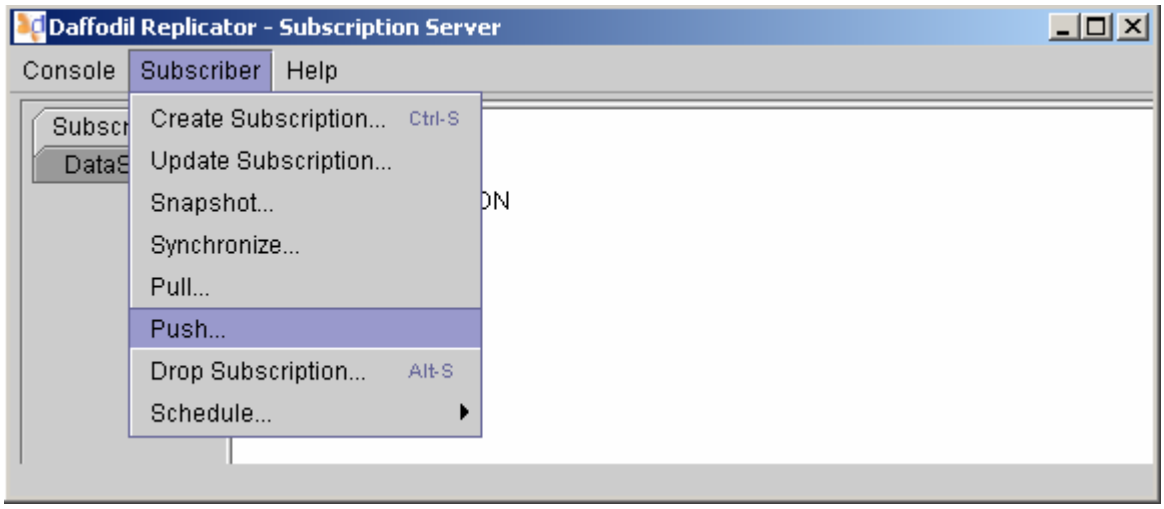
Query: Insert into Student values (11,'Mathew', 1000,'M','21-West Minister Square, Durham','(456)272-3454', 2)

Now update the table *Student* in the Publication database (school)

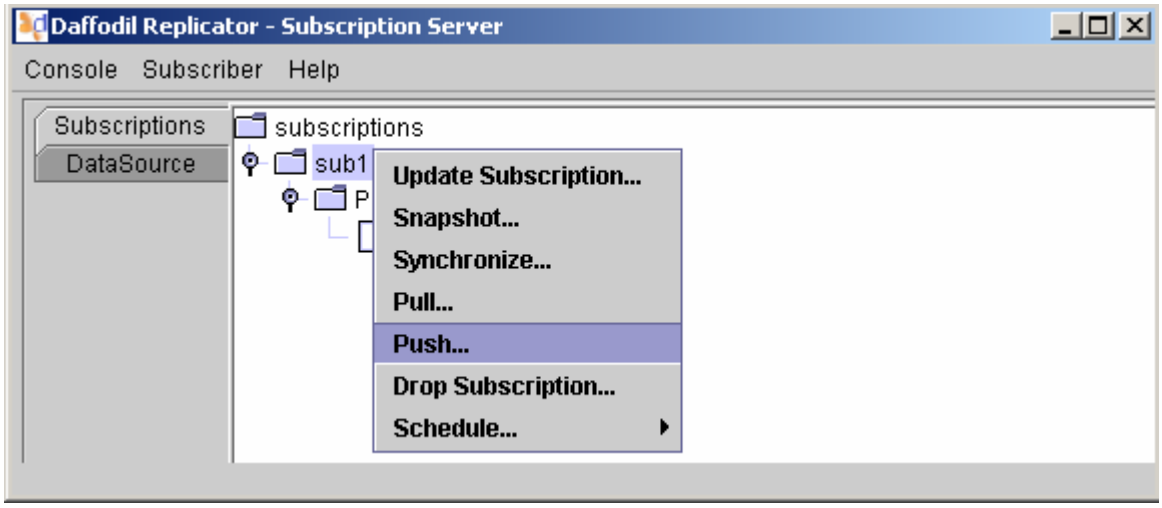
Query: Update Student set PhoneNumber='(971)679-2345' where StudentID=6

Now Subscriber wish to push the changes at Subscriber's end to Publisher and at the same time does not want the changes at Publisher's database to be reflected at Subscriber's end. For this to happen, Subscriber initiates Push replication.

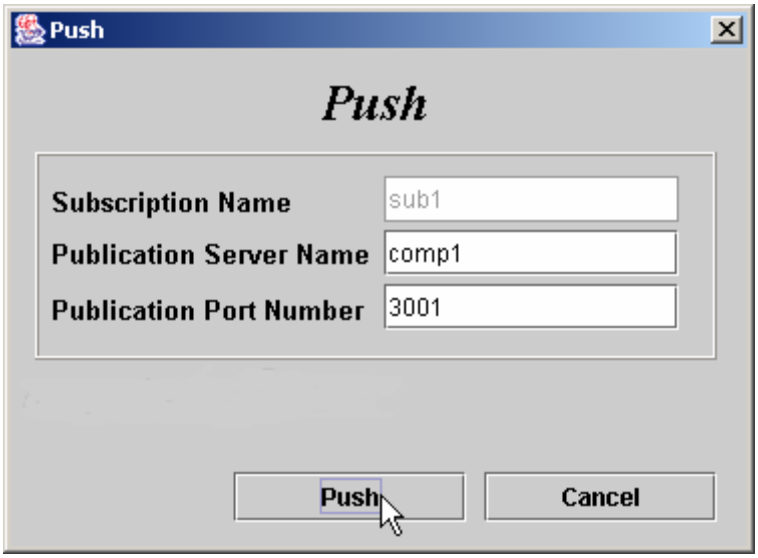
- To perform Push replication, **click** on **Subscriber** menu and select **Push** OR **Right click** on the name of subscriber and **click** on **Push**.



Or



- A **Push** dialog box appears.
- Fill up the following fields in the dialog box as shown below:
  - Enter Subscription Name
  - Enter Publication Server Name
  - Enter Publication Port No

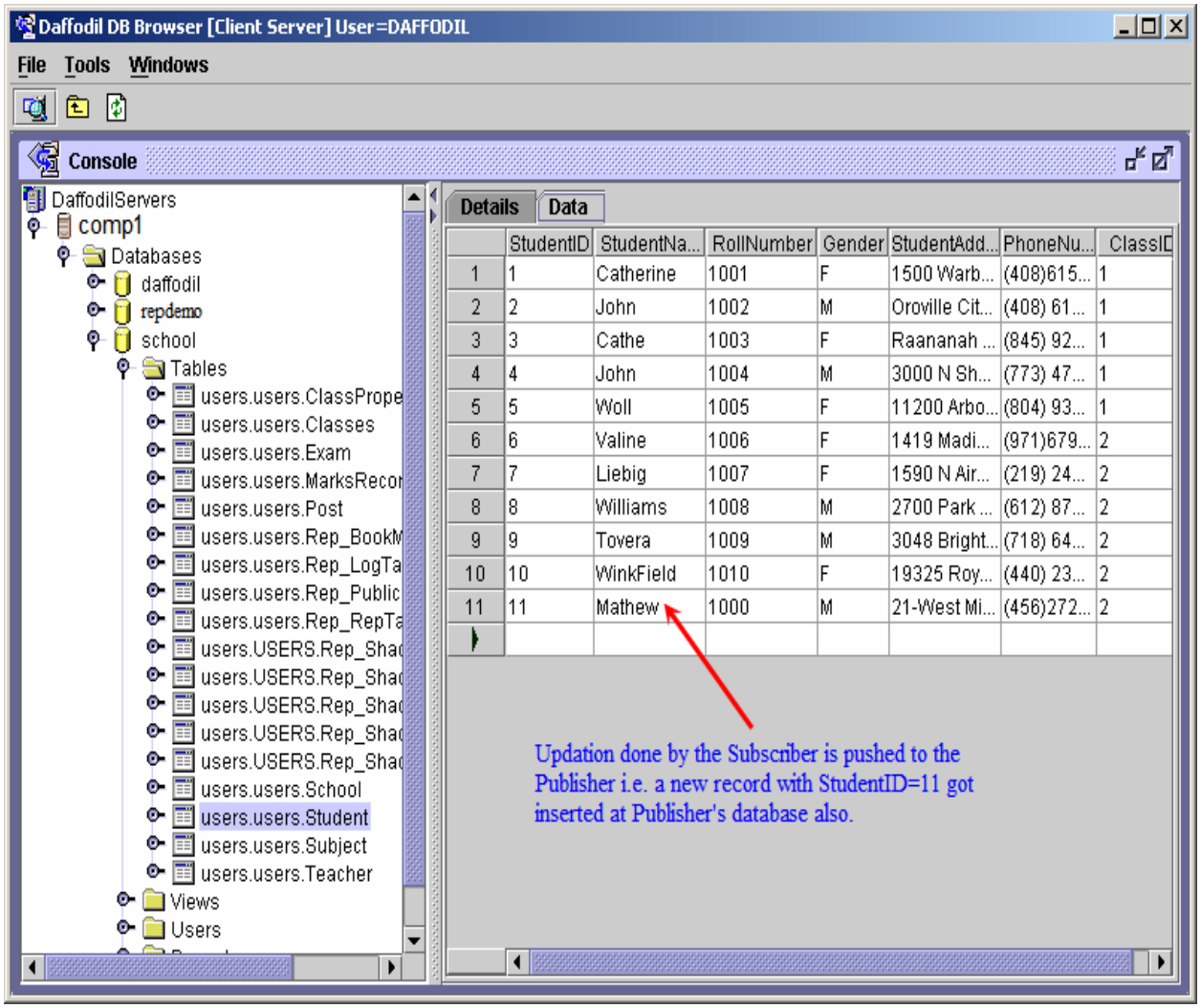


  - **Click** on **Push**
  - On success, you get a message "**Pushed Successfully**"

Note: *Push Replication can only be initiated by the Subscriber.*

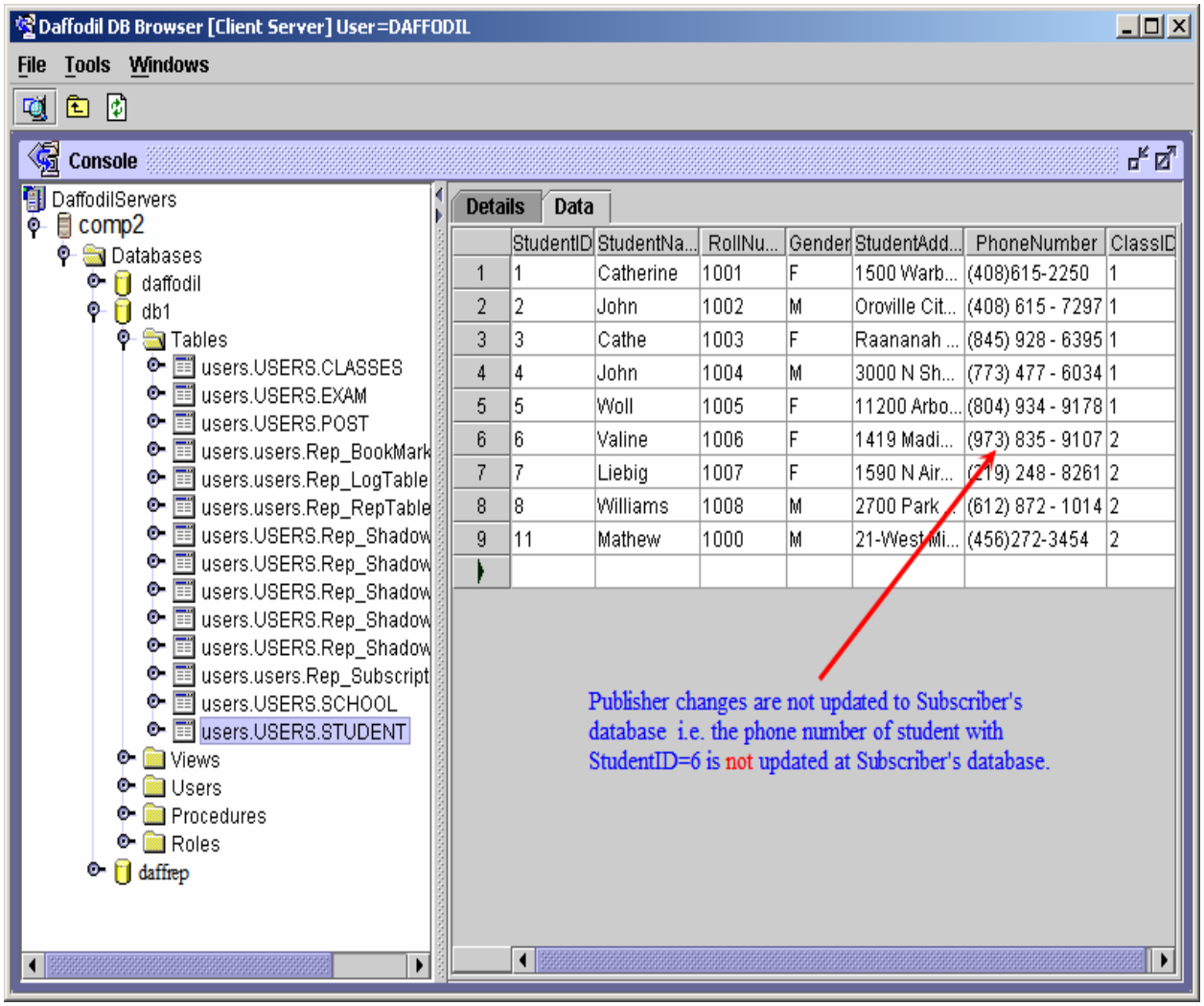
Comparing the data in Publisher database and Subscriber database

The data in *Student* table of Publication database (school) is shown in the following screen shot (after updating the *Student* table and subsequently performing Push replication):



The above screen shot shows that the updation done to *Student* table (in db1 database) by the Subscriber is pushed to the Publisher's *Student* table (in school database).

The following screen shot shows the data in *Student* table of Subscription database (db1) after performing Push replication:

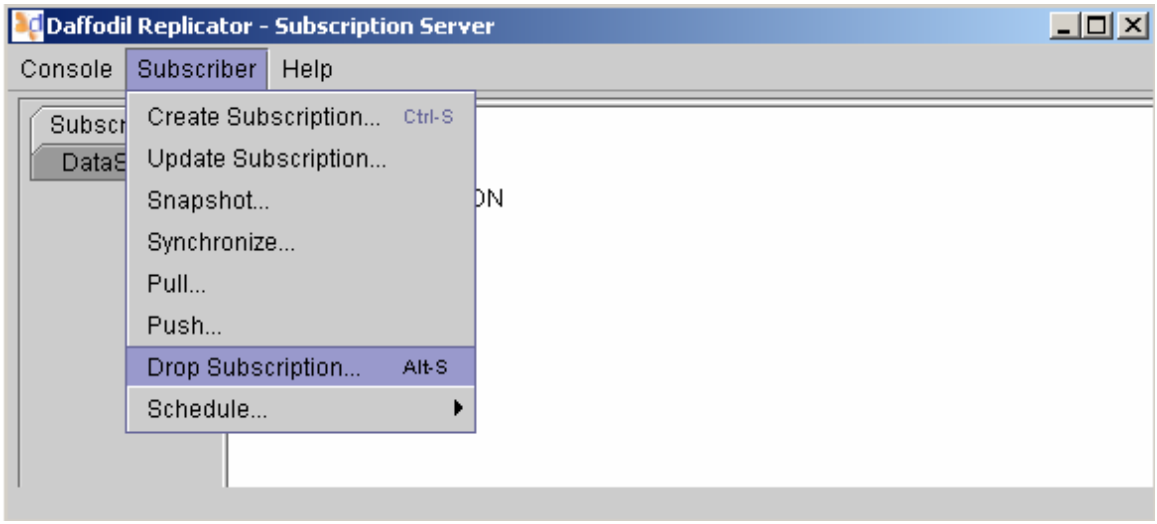


You can see that the Publisher changes are not reflected at Subscriber's database.

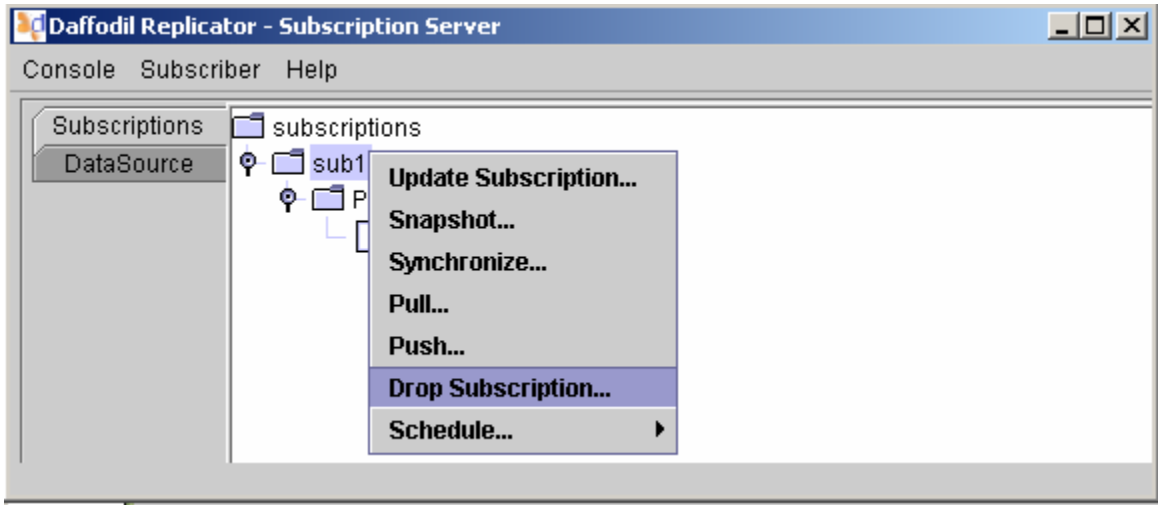
Dropping a Subscription

You can drop subscription by any of following ways:

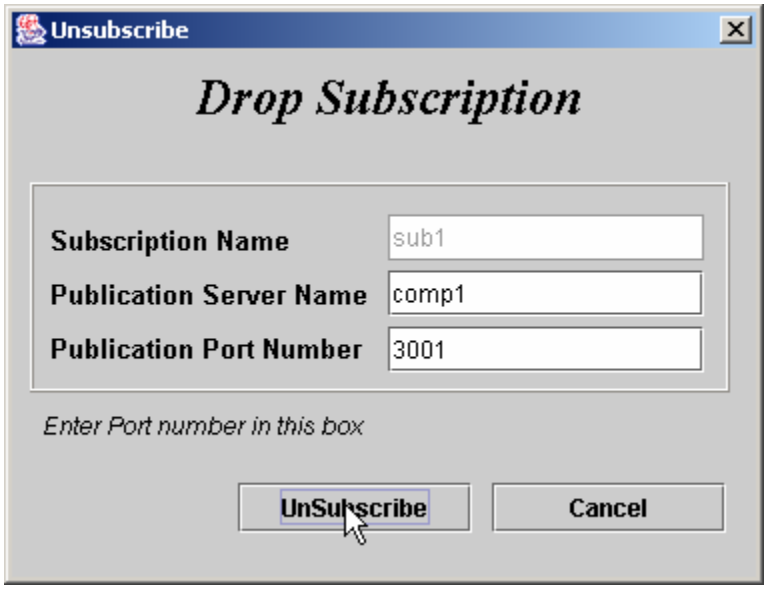
- From the **Subscriber** menu, select **Drop Subscription** (or Press ALT+S) OR right click on subscription name and **click** on **Drop Subscription.**



Or



- '**Drop Subscription'** dialog box appears.
- Fill up the following fields in the dialog box as shown below:
- Enter **Subscription Name** to be dropped
- Enter **Publication Server** Name
- Enter **Publication Port No**



- **Click** on **UnSubscribe**.
- On success, you get a message "**Unsubscribed Successfully**"

Adding, Editing and Removing Schedules

Scheduling is the process of assigning tasks to a set of resources. It is an important concept in many areas such as computing, production processes or airlines.

In the replication process, scheduling can be very handy as it automates the process of replication. Scheduling reduces the overhead of requesting the replication process time and again. A user can add a schedule once and remain free from sending a replication request to publisher again. Replicator will automatically replicate data on that scheduled time

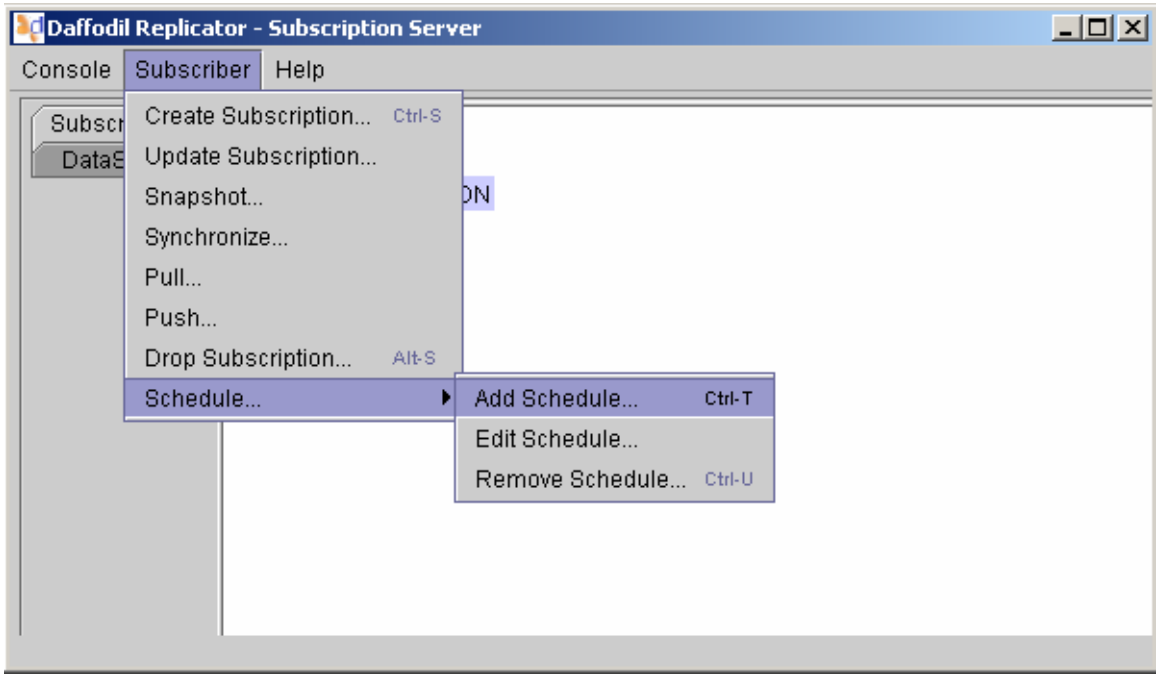
Replicator takes over your whole responsibility of sending replication request. Whenever user starts his Sub server, whatever schedule user has added, replicator will automatically start replication according to it.

Daffodil Replicator permits two modes of replication by using scheduling: Real time and Non Real Time. In Real Time mode of replication, replication process specified in schedule will be performed continuously without any time delay. In Non-Real Time mode of replication, whatever schedule is added by user, replication process specified in schedule will be performed at the scheduled time only.

Add Schedule

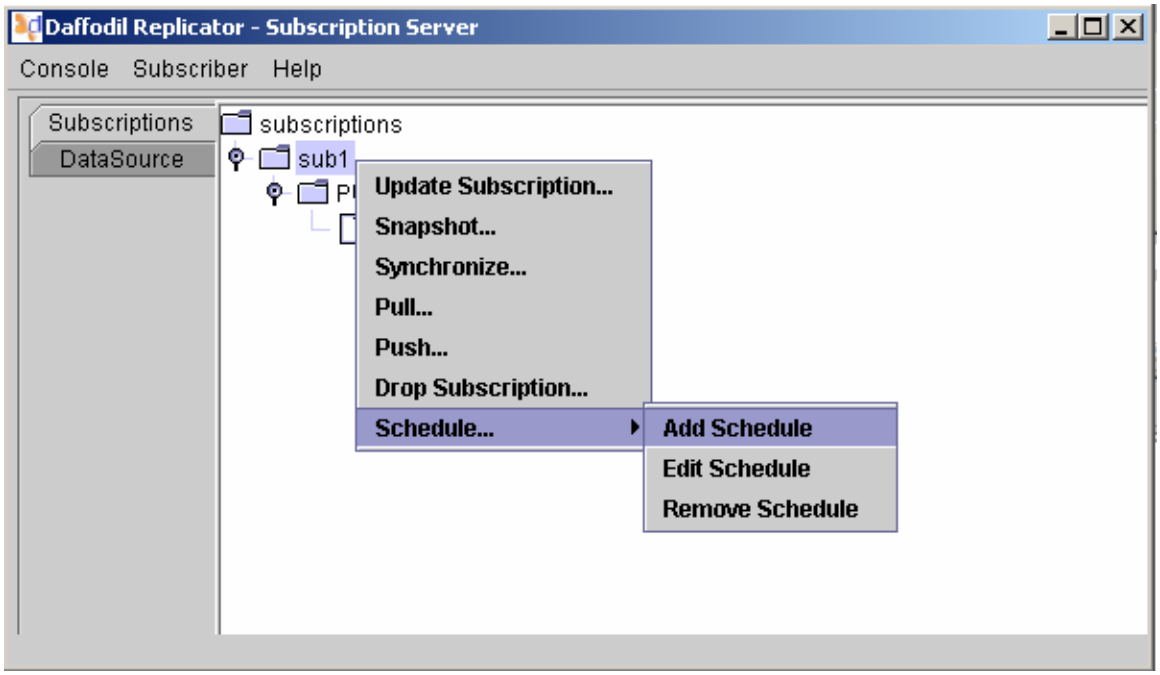
For adding a new schedule, follow the steps given below:

- **Click** on the menu **Subscriber -> Schedule -> Add Schedule**, as shown in the following snapshot.

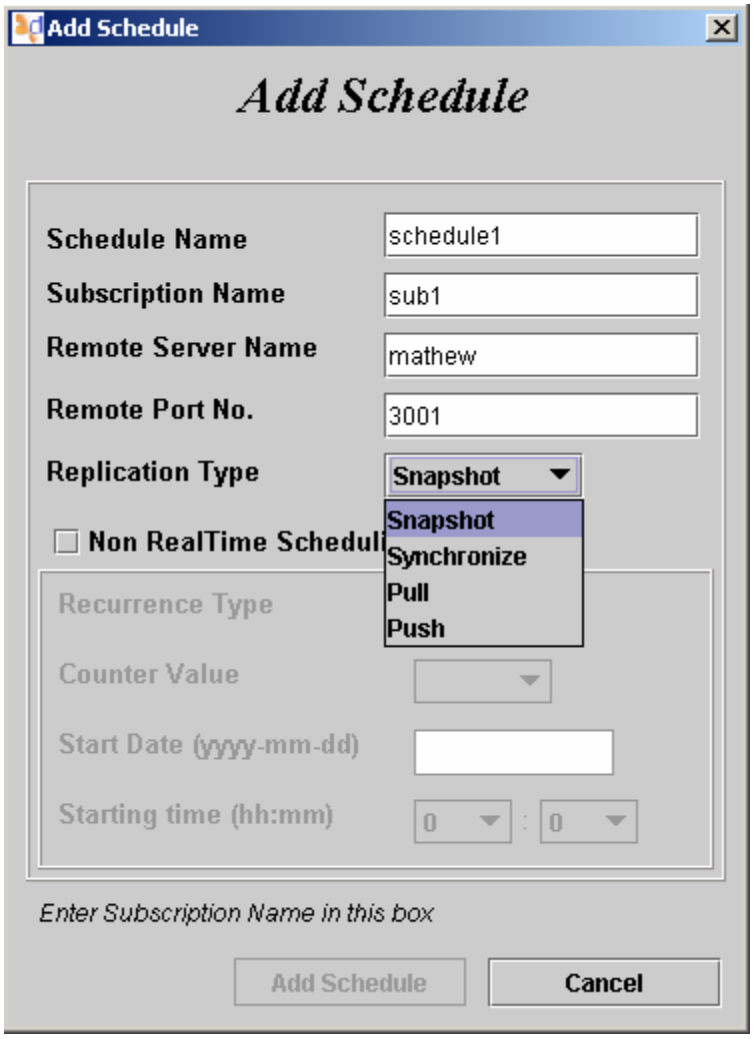


Or

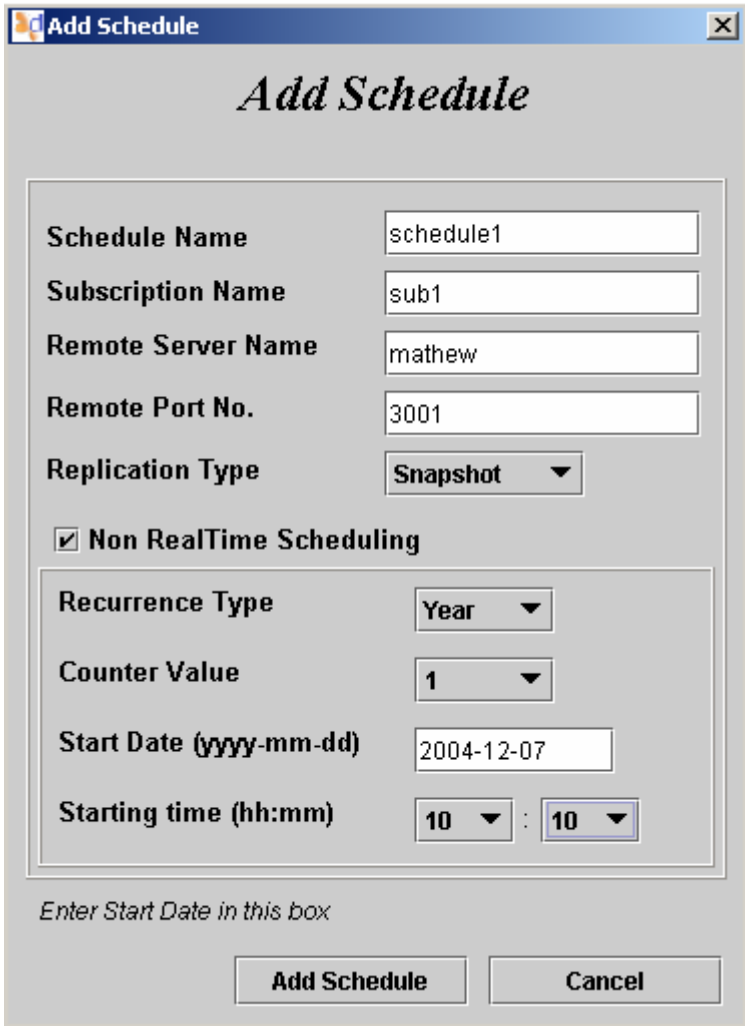
- **Right Click** on a subscription name and Go to **Schedule**-> **Add Schedule**



- A dialog box as shown in the following snapshot will appear. Fill up the following fields in the dialog box as shown below:
    - Enter Schedule Name
    - Enter Subscription Name
    - Enter Publication Server Name
    - Enter Publication Port No
    - Enter Replication Type



- If user wants add ***real time*** Schedule, click on **Add Schedule**.
- If user wants to add ***Non real time*** Schedule, check the Non Real Time checkbox, it will enable the box below the check box.
- Enter Recurrence Type
- Enter Counter Value
- Enter Start Date
- Enter Start Time
- click on **Add Schedule**

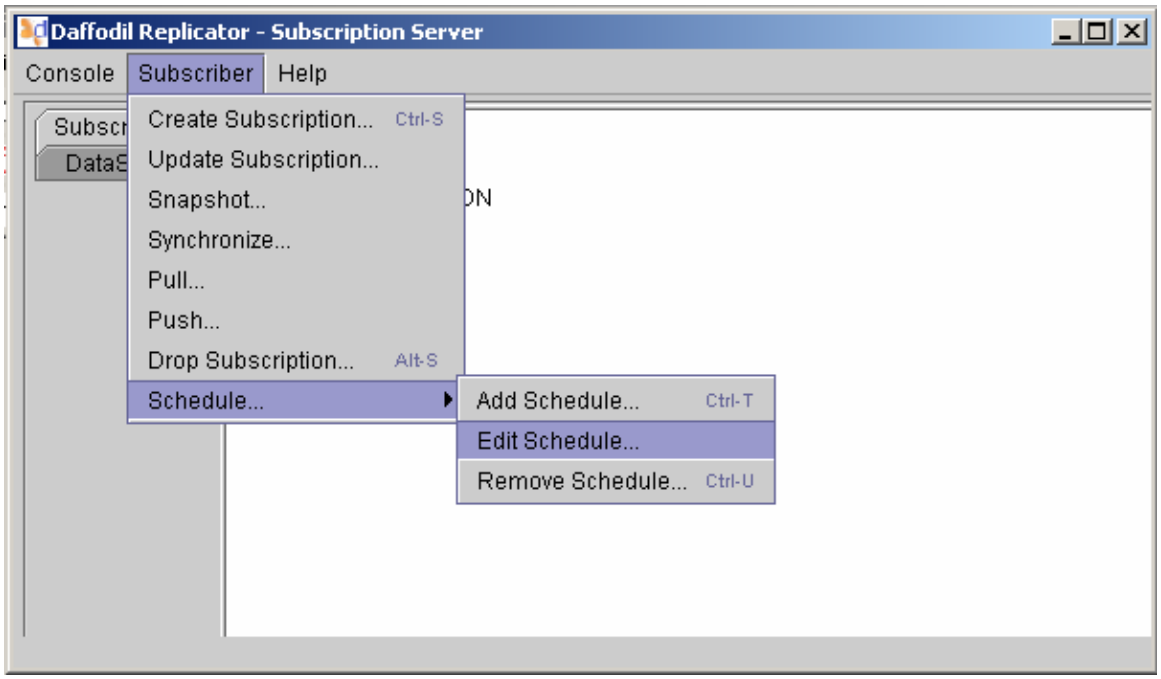


In the above snapshot, we have specified the **Replication type** as "**Snapshot**" and check box for Non real time scheduling has been checked.

Editing a Schedule

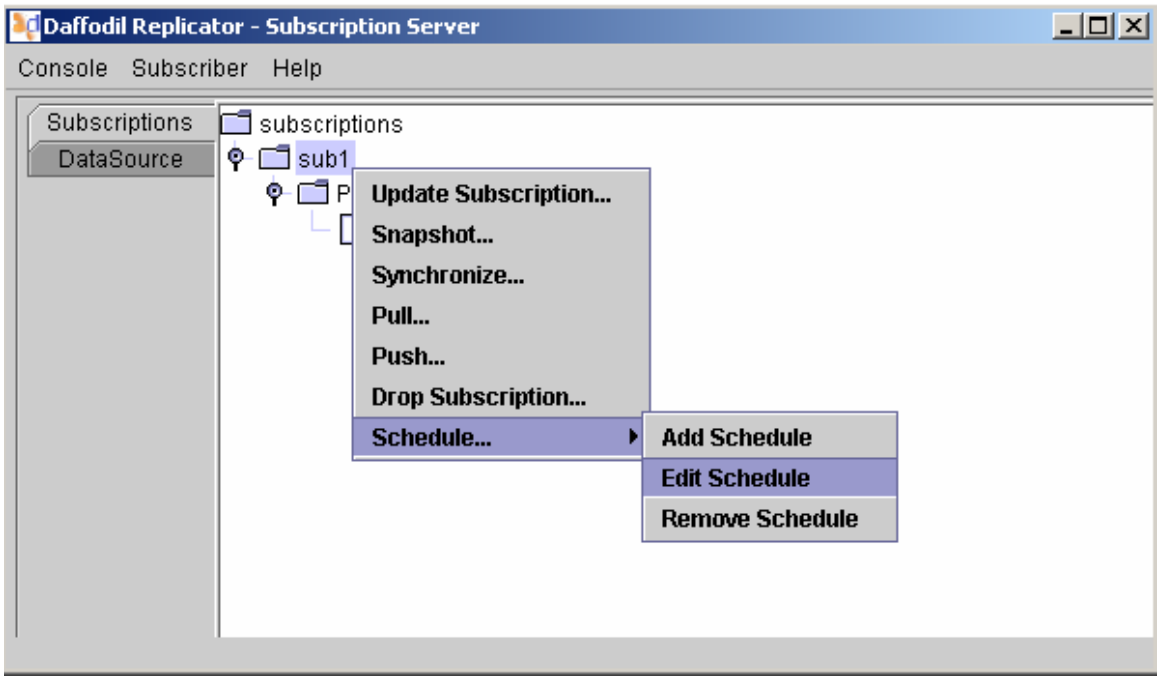
For modifying an existing schedule, follow the step given below.

- **Click** on the menu **Subscriber -> Schedule -> Edit Schedule**, as shown in the following snapshot.



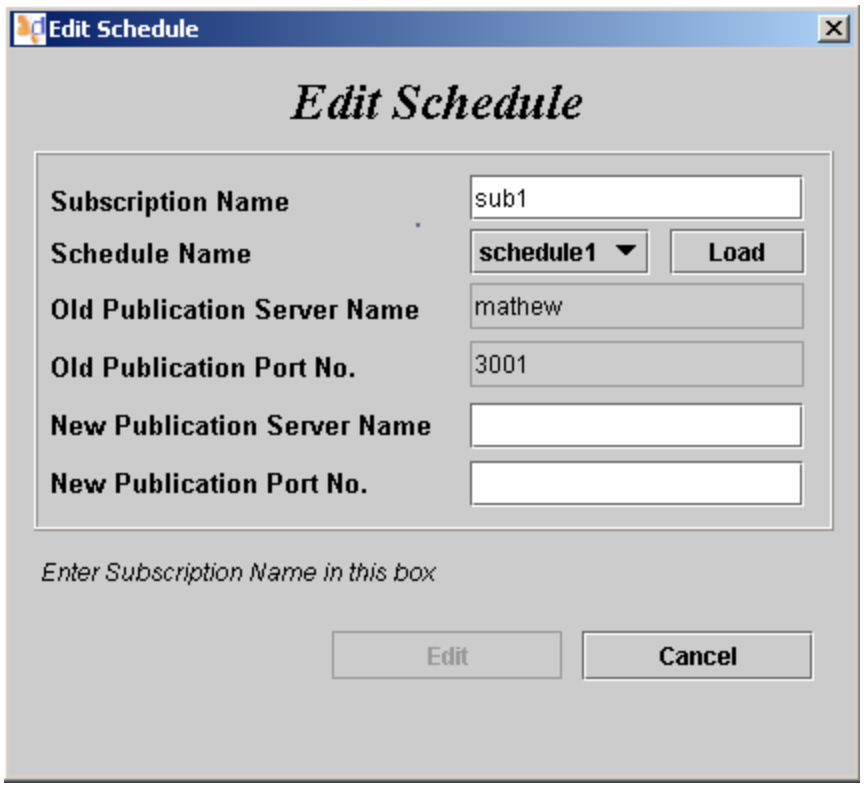
Or

- **Right Click** on a subscription name and Go to **Schedule**-> **Edit Schedule**



An **Edit Schedule** dialog box will appear as shown below.

Take snapshot of below image again after adding new name and port

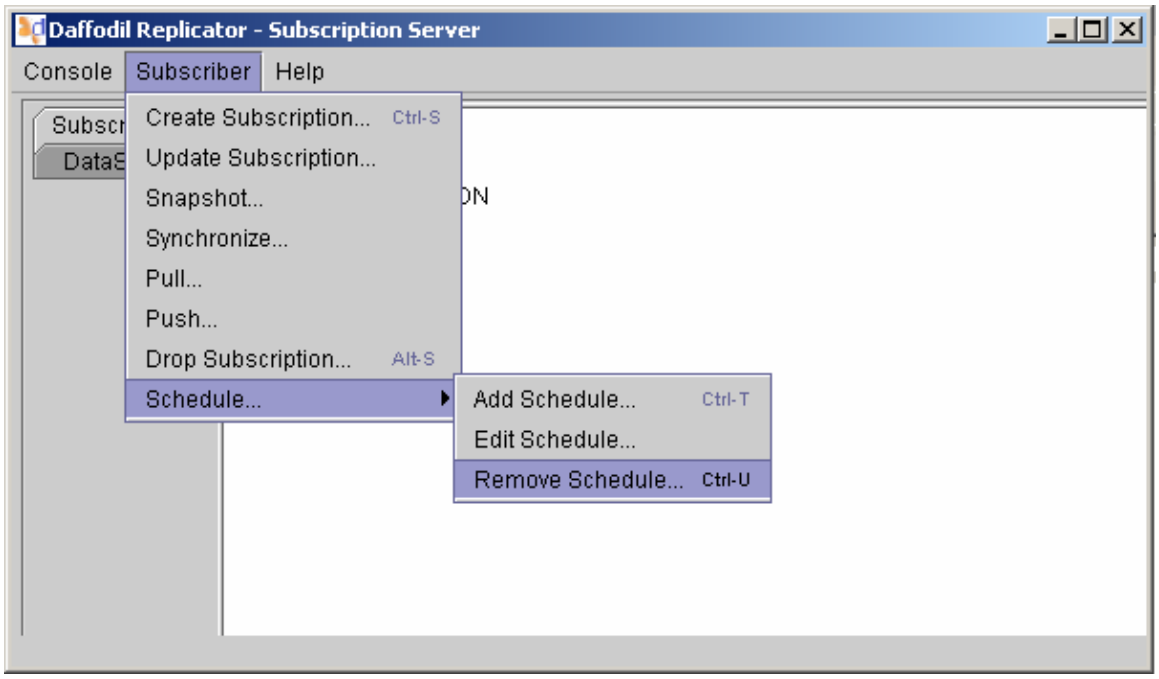


- Now follow the steps below.
- Enter the **Subscription Name.**
- **Click** on the button **'Load'.**
- When a user will click on the button '**Load**' the name of the old publication server and its corresponding port number will automatically be specified by the system.
- Now the user will have to specify the new publication server name and new publication port no. only.
- Now **click** on the button "**Edit**".
- You will see a message "**Schedule updated successfully**"

Removing a Schedule

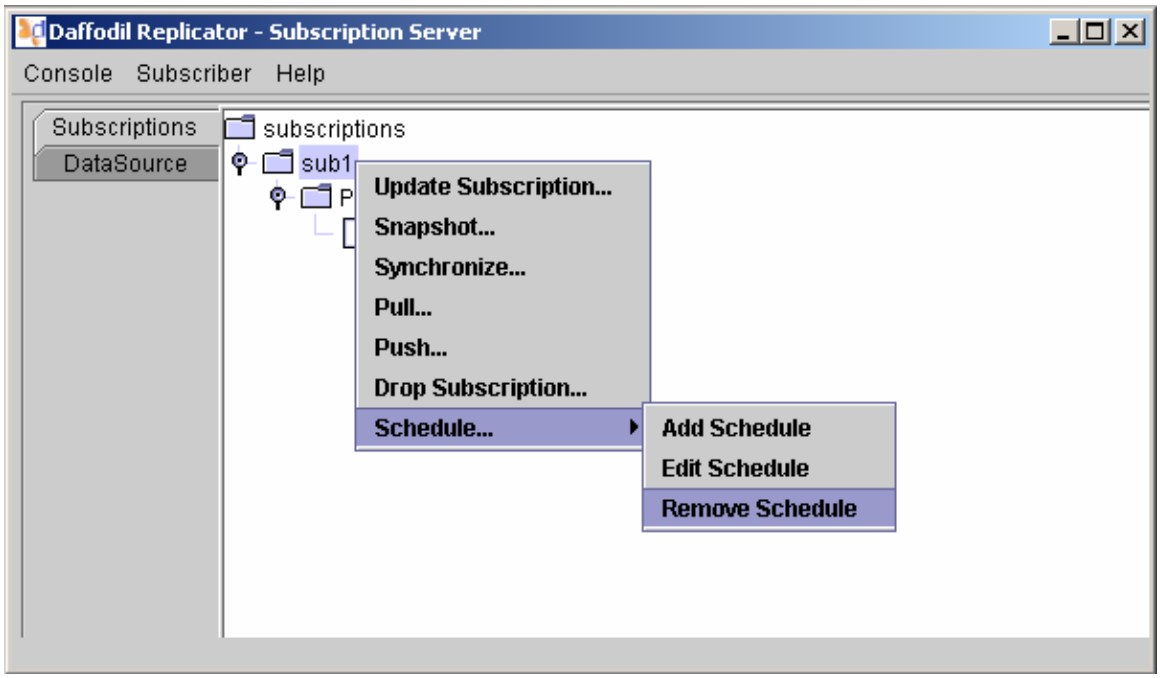
For removing a schedule, follow the step given below.

- **Click** on the menu **Subscriber -> Schedule -> Remove Schedule**, as shown in the following snapshot.

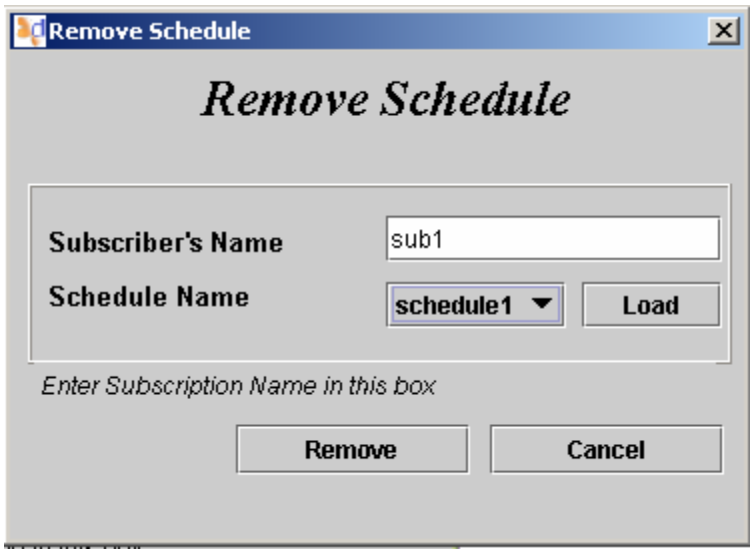


Or

- **Right Click** on a subscription name and Go to **Schedule**-> **Remove Schedule**



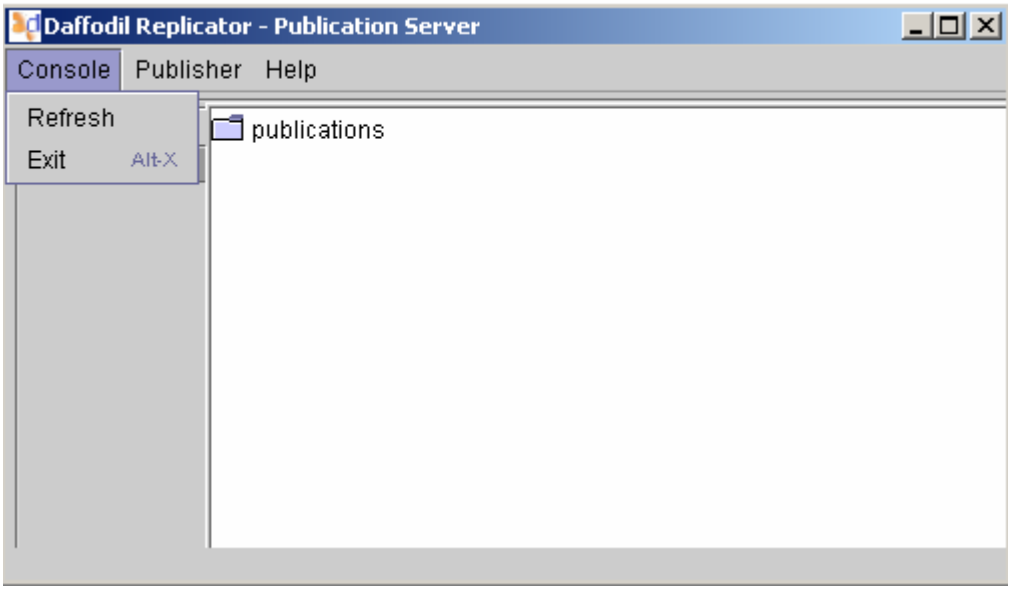
A **Remove Schedule** dialog box will appear, as shown below.



- Enter the **Subscriber's Name** (**'sub1'** in the above example)
- **Click** on the button **'Load'.**
- The Schedule name will automatically be displayed.
- Now **click** on the button "**Remove**".
- You will see a message "**Schedule removed successfully**".

The Console Menu

In **Console** Menu, There are two options .One is for **refresh** the screen and second is for **exiting** from the screen i.e. closing the dialog box.

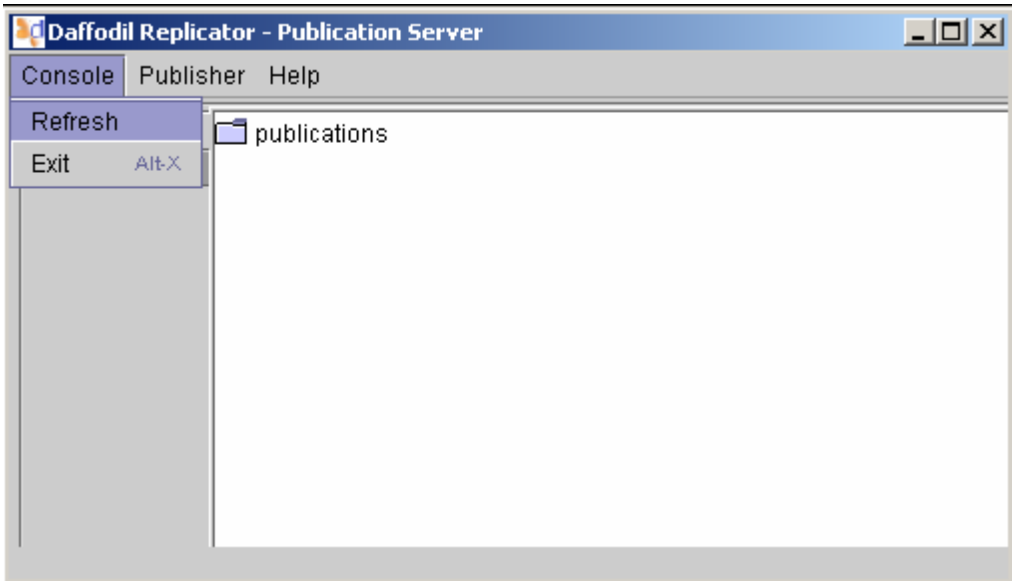


Refresh

Suppose a user has performed some modifications in the items of the publication/Subscription panel. Now in order to make these changes reflect, a user will have to refresh the screen.

- **Click** on the menu option **Console->Refresh**.

Now screen will be refreshed and the changed done by the user will come into effect.

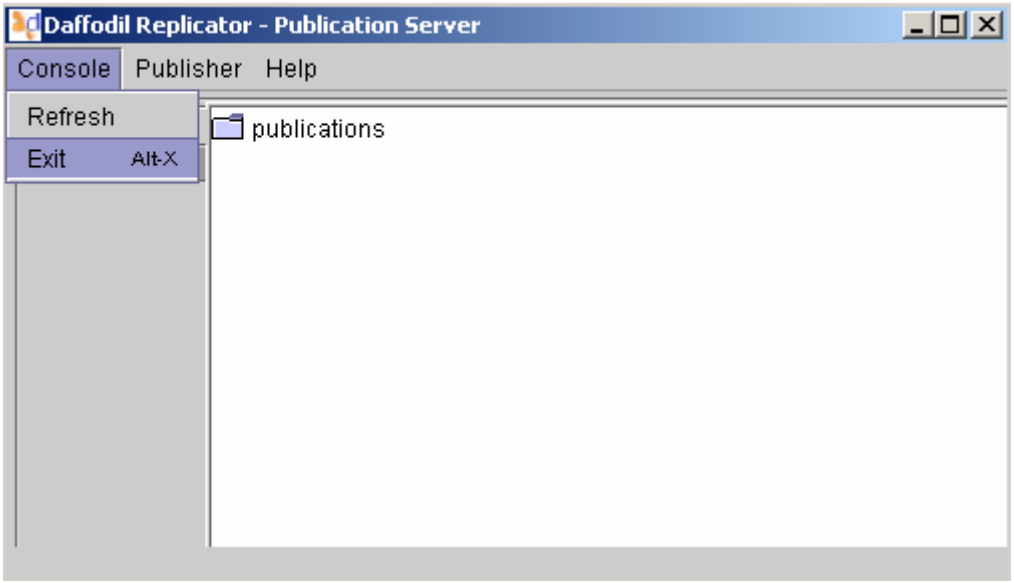


Exit

By clicking on the **exit** option of the **Console** menu, a user can close the dialog box.

- **Click** on the menu option **Console->Exit**.

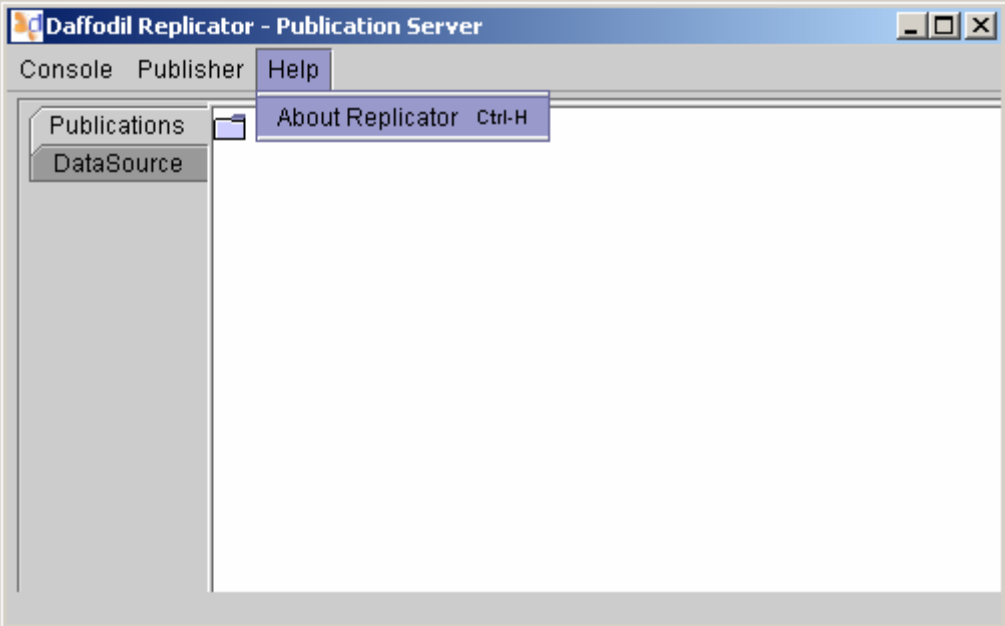
The dialog box will be closed.



The Help Menu

The **Help** menu has only one option named **'About Replicator'** which gives a user information about the product and version number.

Click on the menu option **Help->About Replicator,** as shown below.



A **'Daffodil Replicator Help**' dialog box, showing the version number of the product will appear

Known Issues and Limitations

Daffodil Replicator has the following issues and limitations:

1. **Issue with PostgreSQL**

   By editing the *config* file of PostgreSQL database, you need to specify the IP address of the system where Replication Server is running.

2. **Unsupported Microsoft SQL Server data types**

   uniqueidentifier and *timestamp* data types in SQL Server are not supported, as these data types need external functions for usage. Daffodil Replicator will not be able to perform operations on tables having such columns.

3. **Precision in heterogeneous databases**

   While dealing with heterogeneous databases, precision of data types shall be considered. If the precision of publishing RDBMS is more than that of the subscribing RDBMS, then the following problems may occur:
   (a) The users will not be able to subscribe such Publications.
   (b) Subscription will result in inconsistent data.

4. **Primary key constraint**

   While dealing with heterogeneous databases, if users are trying to subscribe a table having binary / varbinary data type column as the primary key from an external RDBMS (SQL Server) to Daffodil DB, this operation is not permitted in Daffodil Replicator as Daffodil DB does not allow creation of tables having binary / varbinary data type column as the primary key.

5. **Limitations regarding identity Columns in SQL Server**

   While using Daffodil Replicator with an SQL Server database as Publisher and if a table has an identity column as its primary key, in such cases users of Daffodil Replicator will not be able to synchronize such tables but snapshot operations can be performed.

6. **System Tables**

   Manual updation in system tables affect the functioning of the replicator. It may prevent synchronization of data, or show an error message or may give an undesired result.

7. **Auto Increment Column**

   As JDBC does not provide the required Meta data information to append the auto increment columns in table structure, the auto increment columns when subscribed from the publisher does not support this property.

Appendix 1 – Daffodil Replicator Library

(A): JDBC Driver Jars

DATABASE NAME	JDBC DRIVER JARS
Daffodil DB Network Edition	DaffodilDB_Client.jar
Daffodil DB Embedded Edition	DaffodilDB_Client.jar, DaffodilDB_Common.jar
Oracle	classes12.jar
SQL Server	JSQLConnect.jar
	msbase.jar, mssqlserver.jar, msutil.jar
Derby	db2jcc.jar
PostgreSQL	pg80b1.308.jdbc3.jar
DB2	db2jcc.jar

(B): Driver & URL

DATABASE NAME	DRIVER	URL
Daffodil DB Network Edition	in.co.daffodil.db.rmi.RmiDaffodilDBDriver	jdbc:daffodilDB://<DaffodilDB server name>:3456/<database name>; create=true
Daffodil DB Embedded Edition	in.co.daffodil.db.jdbc.DaffodilDBDriver	jdbc:daffodilDB_embedded:<database name>;create=true
Oracle	oracle.jdbc.driver.OracleDriver	jdbc:oracle:thin:@<Oracle server name>:1521:<database name>
SQL Server	com.jnetdirect.jsql.JSQLDriver	jdbc:JSQLConnect://<SQL Server server name>:1433/database=<database name>/lastUpdateCount=true
	com.microsoft.jdbc.sqlserver.SQLServerDriver	jdbc:microsoft:sqlserver://<SQL Server server name>: 1433;DatabaseName=<database name>
Derby	com.ibm.db2.jcc.DB2Driver	jdbc:db2j:net:// <server name>:1527/<database name>;retrieveMessagesFromServerOnGetMessage=true;deferPrepares=true;create=true
PostgreSQL	org.postgresql.Driver	jdbc:postgresql://<PostgreSQL server name>:5432/<database name>
DB2	com.ibm.db2.jcc.DB2Driver	jdbc:db2://<DB2 server name>::50000/<database name>

Appendix 2 – Daffodil Replicator Menus

Daffodil Replicator GUI tool or Console has the following Menus:

- Console
- Publisher (for Publication Server) or Subscriber (for Subscription Server)
- Help

(A): Publication Server Menus

PUBLICATION SERVER MENU ITEMS			DESCRIPTION
CONSOLE	Refresh		To refresh the Publication or subscription tree on screen
	Exit		To close the Publication Server.
PUBLISHER	Create Publication		Displays a dialog box for creating a new Publication.
	Update Publication	Add Tables	Displays a dialog box for adding a new table in Publication. This will allow a user add new tables in the publication
		Drop Tables	Displays a dialog box for dropping a table in Publication. This will allow a user drop any existing table from publication.
	Drop Publication		Displays a dialog box for dropping a Publication.
HELP	About Replicator		Contains version and copyright information of Daffodil Replicator.

(B): Subscription Server Menus

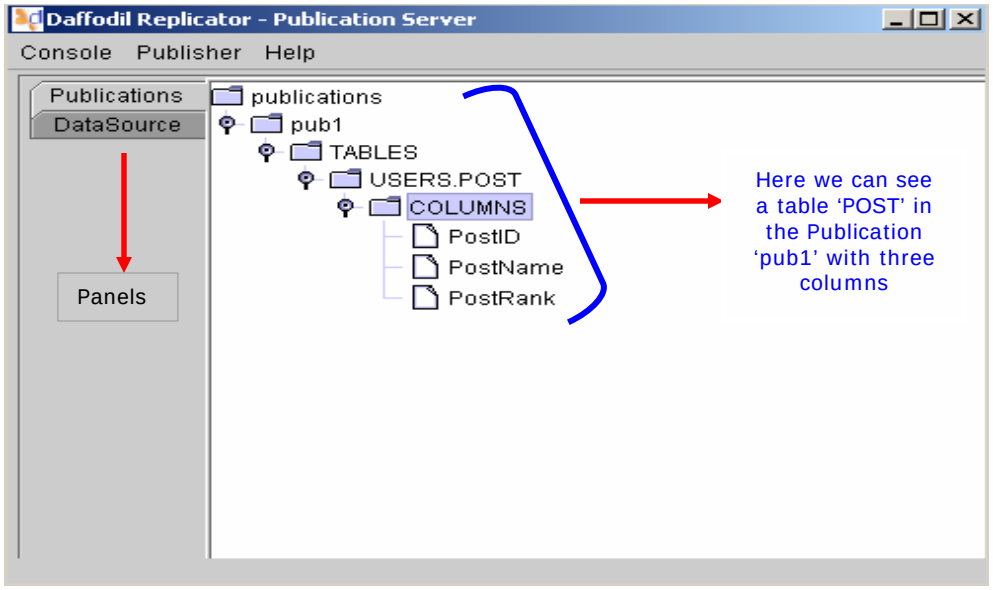
SUBSCRIPTION SERVER MENU ITEMS			DESCRIPTION
CONSOLE	Refresh		To refresh the Publication or subscription tree on screen
	Exit		To close the Publication Server.
SUBSCRIBER	Create Subscription		Displays a dialog box for creating a new Subscription.
	Update Subscription		Displays a dialog box for modifying an existing Subscription.
	Snapshot		Displays a dialog box taking a Snapshot.
	Synchronize		Displays a dialog box for Synchronization.
	Pull		Displays a dialog box for Pulling updated data from Publisher to Subscriber.
	Push		Displays a dialog box for Pushing updated data from Subscriber to Publisher.
	Scheduling	Add Schedule	Displays a dialog box for adding a schedule
		Edit Schedule	Displays a dialog box for editing a schedule
		Remove Schedule	Displays a dialog box for removing a schedule
	Drop Subscription		Displays a dialog box for dropping a Subscription.
HELP	About Replicator		Contains version and copyright information of Daffodil Replicator.

Appendix 3 – Daffodil Replicator Panels

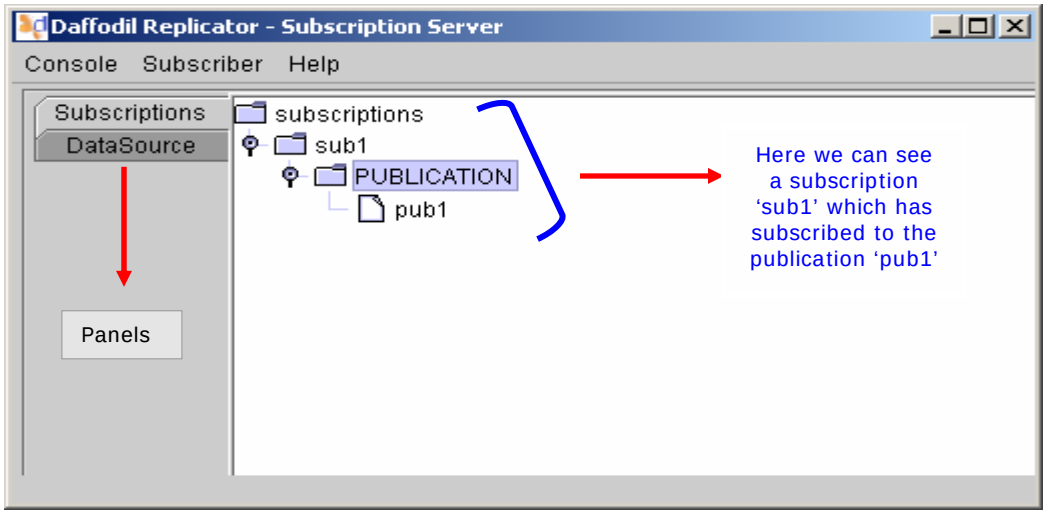
Daffodil Replicator GUI tool or Console has the following panels on the left side of the window:

1. **Publications or Subscriptions**
2. **Data Source**

Publication Server Panel

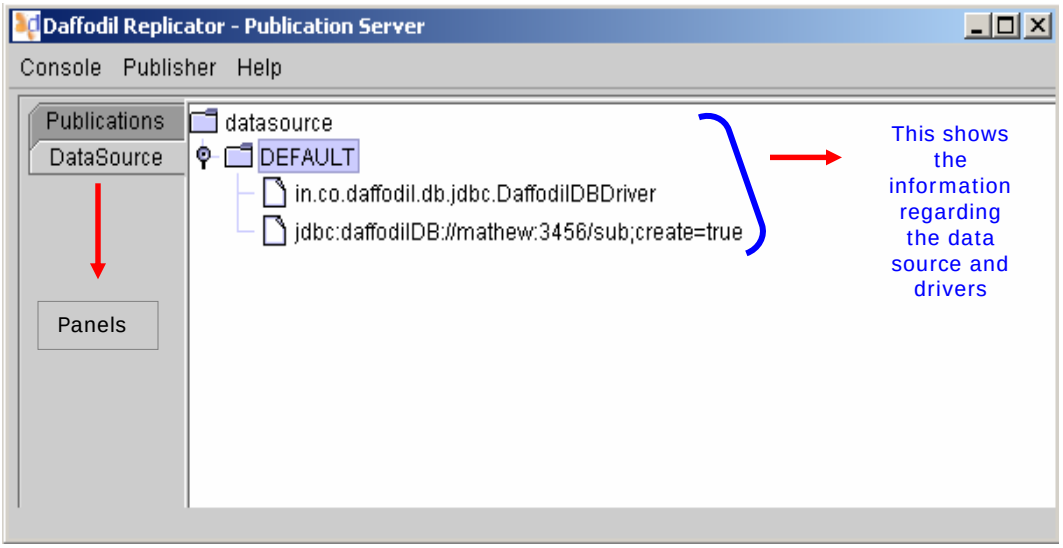


Subscription Server Panel



Data Source Panel

In Data Source panel we can see the information like the data source name, server name and server port number.



DAFFODIL REPLICATOR PANELS	DESCRIPTION
Publications (only for Publication Server)	It shows all the existing Publications and the details about the Tables in the Publication.
Subscriptions (only for Subscription Server)	It shows all the existing Subscriptions and the name of the Publication to which it is subscribed.
Data Source	It shows the information about the DRIVER and URL of the database to which you are connected.

End Note

Although this manual reflects the most current information possible, you should read the ***Daffodil Replicator Release Notes*** from time to time for latest information and updates on Daffodil Replicator.

Release Notes are available at: http://www.daffodildb.com/daffodil-release-notes.html

For more detailed information about Replicator concepts and terminologies, read *Daffodil Replicator Developer's Guide*.

Sign Up for Support

If you have successfully installed and started working with Daffodil Replicator, please remember to sign up for the benefits you are entitled to as a Daffodil Replicator customer.

For free support, be a part of our online developer community at Daffodil Developer Forum

For buying support packages, please visit: http://www.daffodildb.com/support-overview.html

For more information regarding support, write to us at: support@daffodildb.com

We Need Feedback!

If you spot a typographical error in the ***Daffodil Replicator Console Guide***, or if you have thought of a way to make this manual better, we would love to hear from you!

Please submit a report in Bugzilla http://www.daffodildb.com/bugzilla/index.cgi OR write to us at: feedback@daffodildb.com

License Notice