**Distributed Database Applications: Replication**

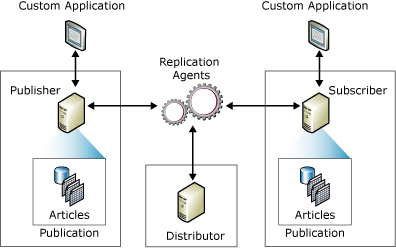
Replication is a set of technologies for copying and distributing data and database objects from one database to another and then synchronising between databases to maintain consistency. Using replication, data can be distributed to different locations and to remote or mobile users over local and wide area networks, dial-up connections, wireless connections, and the Internet.

Replication can be divided into two broad categories: replicating data in a server to server environment and replicating data between a server and clients. Replicating data between servers typically supports improving scalability and availability, data warehousing and reporting, and integrating data from multiple sites. Replicating data between servers and clients typically supports exchanging data with mobile users, consumer point of sale (POS) applications, and integrating data from multiple sites. Replicating data between servers and clients is the form of replication that we will focus on in the labs.

Replication uses a publishing industry metaphor to represent the components in a replication topology, which include *Publisher, Distributor, Subscribers, publications, articles, and subscriptions*.

* A magazine **publisher** produces one or more **publications**
* A **publication** contains **articles**
* The **publisher** either distributes the magazine directly or uses a **distributor**
* **Subscribers** receive **publications** to which they have **subscribed**

A *replication topology* defines the relationship between servers and copies of data and clarifies the logic that determines how data flows between servers. There are several replication processes (referred to as *agents*) that are responsible for copying and moving data between the Publisher and Subscribers. The following illustration is an overview of the components and processes involved in replication.



Publisher

The Publisher is a database instance that makes data available to other locations through replication. The Publisher can have one or more publications, each defining a logically related set of objects and data to replicate.

Distributor

The Distributor is a database instance that acts as a store for replication specific data associated with one or more Publishers. Each Publisher is associated with a single database (known as a distribution database) at the Distributor. The distribution database stores replication status data, metadata about the publication, and, in some cases, acts as a queue for data moving from the Publisher to the Subscribers. In many cases, a single database server instance acts as both the Publisher and the Distributor (this will be the configuration that we will use in the labs). This is known as a *local Distributor*. When the Publisher and the Distributor are configured on separate database server instances, the Distributor is known as a *remote Distributor*.

Subscribers

A Subscriber is a database instance that receives replicated data. A Subscriber can receive data from multiple Publishers and publications. Depending on the type of replication chosen, the Subscriber can also pass data changes back to the Publisher or republish the data to other Subscribers.

Article

An article identifies a database object that is included in a publication. A publication can contain different types of articles, including tables, views, stored procedures, and other objects. When tables are published as articles, filters can be used to restrict the columns and rows of the data sent to Subscribers.

Publication

A publication is a collection of one or more articles from one database. The grouping of multiple articles into a publication makes it easier to specify a logically related set of database objects and data that are replicated as a unit.

Subscription

A subscription is a request for a copy of a publication to be delivered to a Subscriber. The subscription defines what publication will be received, where, and when. There are two types of subscriptions: push and pull.

*Push Subscriptions*

With a push subscription, the Publisher propagates changes to a Subscriber without a request from the Subscriber. Changes can be pushed to Subscribers on demand, continuously, or on a scheduled basis. The Distribution Agent or Merge Agent runs at the Distributor. This approach is used when:

* Data will typically be synchronised continuously or on a frequently recurring schedule.
* Publications require near real-time movement of data.
* The higher processor overhead at the Distributor does not affect performance.
* Most often used with snapshot and transactional replication.

*Pull Subscriptions*

With a pull subscription, the Subscriber requests changes made at the Publisher. Pull subscriptions allow the user at the Subscriber to determine when the data changes are synchronised. The Distribution Agent or the Merge Agent runs at the Subscriber. This approach is used when:

* Data will typically be synchronised on demand or on a schedule rather than continuously.
* The publication has a large number of Subscribers, and/or it would be too resource-intensive to run all the agents at the Distributor.
* Subscribers are autonomous, disconnected, and/or mobile. Subscribers will determine when they will connect and synchronise changes.
* Most often used with merge replication.

In our labs, we will use Pull type subscriptions to cater for mobile users who determine when they will connect and synchronise changes.

**Types of Replication**

SQL Server provides the following types of replication for use in distributed applications:

**Snapshot replication**

Snapshot replication distributes data exactly as it appears at a specific moment in time and does not monitor for updates to the data. When synchronisation occurs, the entire snapshot is generated and sent to Subscribers.

**Transactional replication**

Transactional replication typically starts with a snapshot of the publication database objects and data. As soon as the initial snapshot is taken, subsequent data changes and schema modifications made at the Publisher are usually delivered to the Subscriber as they occur (in near real time). The data changes are applied to the Subscriber in the same order and within the same transaction boundaries as they occurred at the Publisher; therefore, within a publication, transactional consistency is guaranteed.

**Merge replication**

Merge replication typically starts with a snapshot of the publication database objects and data. Subsequent data changes and schema modifications made at the Publisher and Subscribers are tracked with triggers. The Subscriber synchronises with the Publisher when connected to the network and exchanges all rows that have changed between the Publisher and Subscriber since the last time synchronisation occurred.

Merge replication is typically used in server-to-client environments. Merge replication is appropriate in any of the following situations:

* Multiple Subscribers might update the same data at various times and propagate those changes to the Publisher and to other Subscribers.
* Subscribers need to receive data, make changes offline, and later synchronise changes with the Publisher and other Subscribers.
* Each Subscriber requires a different partition of data.
* Conflicts might occur and, when they do, you need the ability to detect and resolve them.

Merge replication allows various sites to work autonomously and later merge updates into a single, uniform result. Because updates are made at more than one node, the same data may have been updated by the Publisher and by more than one Subscriber. Therefore, conflicts can occur when updates are merged and merge replication provides a number of ways to handle conflicts. Merge replication is the form of replication that we will use in our labs.

**Replication Agents**

Replication uses a number of standalone programs, called agents, to carry out the tasks associated with tracking changes and distributing data. By default, replication agents run as jobs scheduled under SQL Server Agent, and SQL Server Agent must be running for the jobs to run.

**SQL Server Agent**

SQL Server Agent hosts and schedules the agents used in replication and provides an easy way to run replication agents. SQL Server Agent also controls and monitors operations outside of replication.

**Snapshot Agent**

The Snapshot Agent is typically used with all types of replication. It prepares schema and initial data files of published tables and other objects, stores the snapshot files, and records information about synchronisation in the distribution database. The Snapshot Agent runs at the Distributor.

**Distribution Agent**

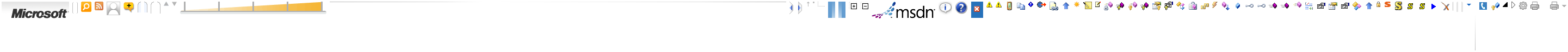
The Distribution Agent is used with snapshot replication and transactional replication. It applies the initial snapshot to the Subscriber and moves transactions held in the distribution database to Subscribers. The Distribution Agent runs at either the Distributor for push subscriptions or at the Subscriber for pull subscriptions.

**Merge Agent**

The Merge Agent is used with merge replication (described in the next section). It applies the initial snapshot to the Subscriber and moves and reconciles incremental data changes that occur. Each merge subscription has its own Merge Agent that connects to both the Publisher and the Subscriber and updates both. The Merge Agent runs at either the Distributor for push subscriptions or the Subscriber for pull subscriptions. By default, the Merge Agent uploads changes from the Subscriber to the Publisher and then downloads changes from the Publisher to the Subscriber.

**Web Synchronisation for Merge Replication**

[Other Versions](javascript:;)



* [SQL Server 2008](http://msdn.microsoft.com/en-us/library/ms151763(d=printer,v=SQL.100).aspx)
* [SQL Server 2005](http://msdn.microsoft.com/en-us/library/ms151763(d=printer,v=SQL.90).aspx)

Web synchronisation for merge replication lets you replicate data by using the HTTPS protocol, and is useful for the following scenarios:

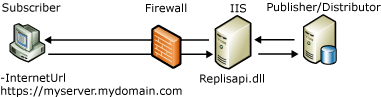
* Synchronising data from mobile users over the Internet.
* Synchronising data between Microsoft SQL Server databases across a corporate firewall.

For example, a traveling sales representative can use Web synchronisation. A company may have sales representatives that travel to various stores and suppliers throughout their regions. On longer trips the representatives may stay in hotels and need a convenient way to upload sales data and download any product updates from head office at the end of each day. If each representative’s laptop computer is installed with SQL Server and has merge replication enabled, Web synchronisation could be used to provide this functionality. The Merge Agent on each laptop has an Internet URL that points to the replication components installed on the head office computer running Microsoft Internet Information Services (IIS). These components synchronise the Subscriber with the Publisher. Each representative can now connect through any available Internet connection, and can upload and download the appropriate data. The Internet connection uses Secure Sockets Layer (SSL); therefore, a virtual private network (VPN) is not required.

[**Overview of How Web Synchronisation Works**](javascript:void(0))

When Web synchronisation is used, updates at the Subscriber are packaged and sent as an XML message to the computer that is running IIS by using the HTTPS protocol. The computer that is running IIS then sends the commands to the Publisher in a binary format, typically by using TCP/IP. Updates at the Publisher are sent to the computer that is running IIS and then packaged as an XML message for delivery to the Subscriber.

The following illustration shows some of the components that are involved in Web synchronisation for merge replication.



The SQL Server Replication Listener (Replisapi.dll) is configured on the computer that is running IIS and is responsible for handling messages that are sent to the server from the Publisher and Subscribers. Each node in the topology handles the XML data stream by using the Merge Replication Reconciler (Replrec.dll).

**Synchronisation Process**

The following steps occur during synchronisation:

1. The Merge Agent is started at the Subscriber. The agent does the following:
   1. Makes an SQL connection to the subscription database.
   2. Extracts any changes from the database.
   3. Makes an HTTPS request to the computer that is running IIS.
   4. Uploads data changes as an XML message.
2. The SQL Server Replication Listener and Merge Replication Reconciler that are hosted on the computer that is running IIS do the following:
   1. Respond to the HTTPS request.
   2. Make an SQL connection to the publication database.
   3. Apply the upload changes to the publication database.
   4. Extract the download changes for the Subscriber.
   5. Send an HTTPS response back to the Merge Agent.
3. The Merge Agent at the Subscriber then accepts the HTTPS response and applies the download changes to the subscription database.

We will use web synchronisation to implement replication in labs that will allow mobile users to receive and upload data from a remote SQL Server database.

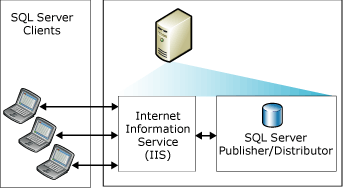
**Topologies for Web Synchronisation**

To common ways to configure Web synchronisation include:

* Single server
* Two servers

**Single Server**

In the simplest topology, IIS, the SQL Server Publisher, and the SQL Server Distributor all reside on a single server. Subscribers synchronise by connecting to IIS on the Publisher. The Publisher can be located behind a firewall. This is the topology that we will implement in the lab.



**Two Servers**

You can place IIS on one server and configure the SQL Server Publisher and Distributor on another server. The server running IIS can be isolated from the Internet by a firewall. Subscribers synchronise by connecting to IIS.

