A. Presentación

La replicación es una poderosa funcionalidad de SQL Server que permite distribuir datos y ejecutar los procedimientos almacenados sobre varios servidores de la empresa. La tecnología de replicación ha evolucionado considerablemente y ahora permite copiar, trasladar los datos a diferentes lugares y sincronizarlos automáticamente. La replicación puede establecerse entre las bases de datos residentes sobre el mismo servidor o sobre servidores diferentes. Los servidores pueden estar sobre una red local (LAN) o global (WAN) o sobre Internet.

SQL Server distingue dos grandes categorias de replicación :

- la replicación de servidor a servidor,
- la replicación de servidor a clientes.

En el caso de replicación de servidor a servidor, la replicación permite una mayor integración o aproximación de los datos entre varios servidores de base de datos. El objetivo de este tipo de replicación es efectuar un intercambio de información entre servidores de base de datos. Los usuarios que trabajan sobre las bases de datos que participan en la replicación pueden, de esta manera, consultar datos de mayor calidad.

La replicación de servidor a cliente afecta principalmente a los usuarios desconectados de la red de la empresa que desean trabajar con todo o parte de los datos de la empresa. Los usuarios trabajan con una aplicación específica y utilizan SQL Server como servidor de base de datos local. Cuando el usuario se conecta a la red, la sincronización de los datos entre su puesto y la instancia SQL Server central la realiza el mecanismo de replicación de SQL Server.

La gestión de la replicación se ha simplicado para permitir un establecimiento y un mantenimiento más fáciles. Para las soluciones más complejas y que necesiten una integración completa a un progama, SQL Server ofrece API RMO (*Replication Management Object*). Esta API permite manipular mediante programación todos los elementos de la replicación. API RMO está disponible para los lenguajes que se basan en el framework.Net.

El concepto de esquema que permite un agrupamiento lógico de los objetos en la base de datos es tomado en cuenta por la replicación con la posibilidad de realizar cambios de esquemas.

B. Las necesidades para la replicación

La replicación es una tecnología compleja y no puede existir una solución única para cubrir todas las necesidades. SQL Server ofrece diferentes tecnologías de replicación que se pueden adaptar y combinar para responder lo más fielmente posible a las necesidades de las aplicaciones. Cada tecnología tiene ventajas e inconvenientes. Los tres criterios principales para seleccionar una tecnología de replicación son:

- la coherencia de los datos replicados,
- la autonomía de los sitios,

- el particionamiento de datos para evitar conflictos.

No es posible optimizar los tres criterios al mismo tiempo, de manera que una solución que favorezca la coherencia de los datos deberá poca autonomía a los sitios para averiguar en todo momento el conjunto de modificaciones que tienen lugar sobre los datos.

1. Coherencia de los datos replicados

Existen dos tipos principales de coherencia:

- la homogeneidad de las transacciones,
- la convergencia de los datos.

La coherencia de las operaciones distribuidas, como la replicación, es mucho más complicada de mantener en comparación con la coherencia de las transacciones locales, por lo que es suficiente con respetar el test ACAD (*Atomicidad, Coherencia, Aislamiento y Durabilidad*).

La homogeneidad de las transacciones en la replicación obliga a que los datos sean idénticos en todos los sites que participan en la replicación, como si la transacción fuera a ejecutarse sobre todos los sites.

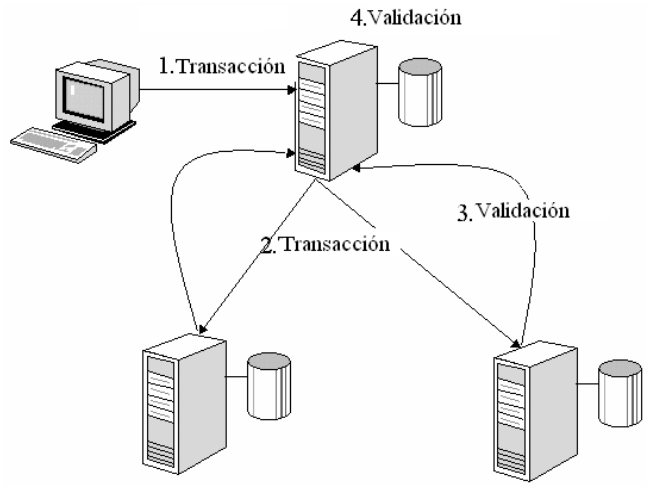
La convergencia de los datos significa que todos los sites que participan en las replicaciones deben tener el mismo juego de datos, que no es necesariamente el que se obtendría si todas las replicaciones se hubieran desarrollado sobre el mismo servidor.

a. Coherencia de las transacciones

En todos los casos, los sites contienen un juego de valores idéntico a aquel sobre el que han hecho o se podían haber hecho todas las operaciones de modificación.

Coherencia transaccional inmediata

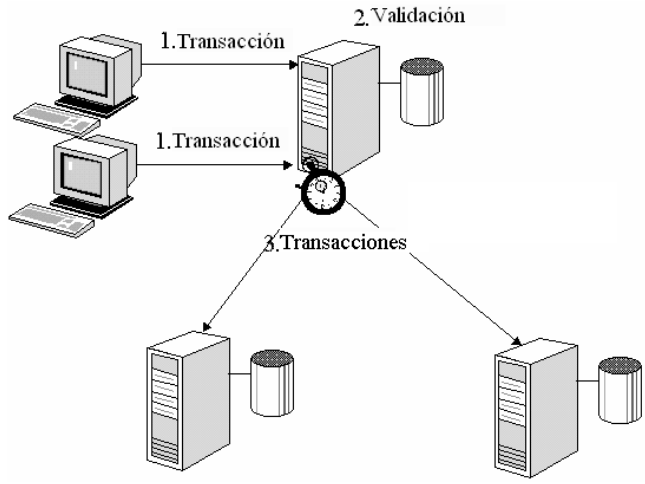
Con esta coherencia, todos los sitios que participan en la replicación tienen la garantía de ver siempre los mismos valores en el mismo momento. Para asegurar la coherencia transaccional, SQL Server dispone de un protocolo de validación en dos fases con todos los sitios participantes. Las modificaciones se efectúan en todos los sitios o en ninguno. Esta solución está muy limitada en la realidad, ya que los problemas de red prohíben toda validación de una transacción mientras el servidor no se conecte de nuevo a la red.



En el ejemplo anterior, el cliente efectúa una transacción sobre el servidor al que está conectado. Esta transacción sólo se validará si se ha ejecutado con éxito en todos los servidores que participan en la replicación.

Coherencia transaccional latente

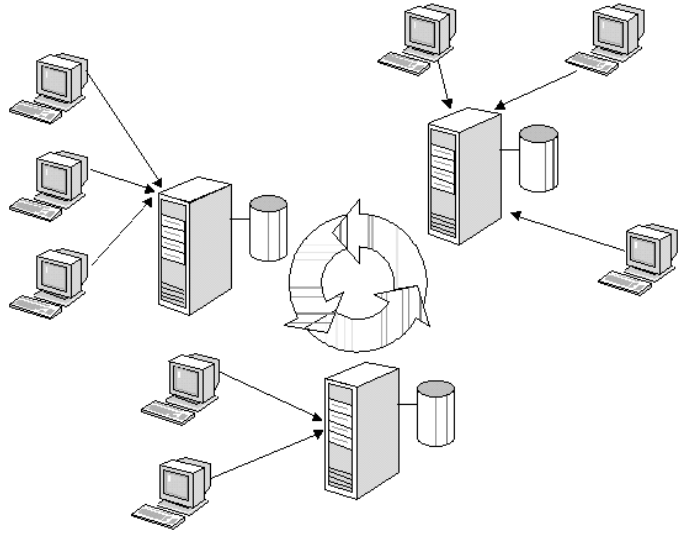
La coherencia latente de las transacciones garantiza que todos los participantes obtendrán los mismos valores que los contenidos en el site de publicación en un momento dado. Puede transcurrir un período de tiempo entre el instante en que se efectúa la transacción en el servidor de publicación y el instante en que las modificaciones se reflejan en los otros sitios.



En este ejemplo, el cliente envía una transacción al servidor y esta se ejecuta de manera local inmediatamente. Después, de forma periódica, los servidores que participan en la replicación repiten localmente el conjunto de transacciones efectuadas sobre el servidor principal.

b. Convergencia de los datos

Con este proceso, todos los sitios acaban obteniendo el mismo juego de datos, lo que no habría sido posible si todas las modificaciones se hubieran realizado sobre un único servidor. Todos los sites evolucionan libre e independientemente, unos de otros. La convergencia de los datos se establece con ayuda de la replicación de fusión que tiende a llevar a todos los sitios que participan a gestionar el mismo juego de datos.



Todos los servidores son accessibles en lectura/escritura y las transacciones se ejecutan localmente. Entonces el proceso de replicación realiza las transacciones sobre los otros servidores teniendo en cuenta las modificaciones que han podido intervenir localmente.

2. Autonomía de los sitios

La autonomía de los sites se mide por el impacto que tiene una operación efectuada sobre un site en el resto de sites. La autonomía es perfecta cuando un site puede evolucionar totalmente de manera libre e independiente. El site autónomo no se preocupa por las operaciones que pueden intervenir en los otros sites. Por ejemplo, cuando la replicación garantiza la convergencia de los datos, la autonomía de los sites está en su máximo, ya que cada servidor SQL puede evolucionar libremente en relación a los otros sites. A la inversa, la coherencia transaccional inmediata impone una autonomía casi nula de los sites que participan en ella, ya que una transacción debe ser aprobada por todo el mundo antes de ser validada. Si un solo servidor no puede, entonces la transacción no se valida en ninguno.

3. Particionamiento de los datos

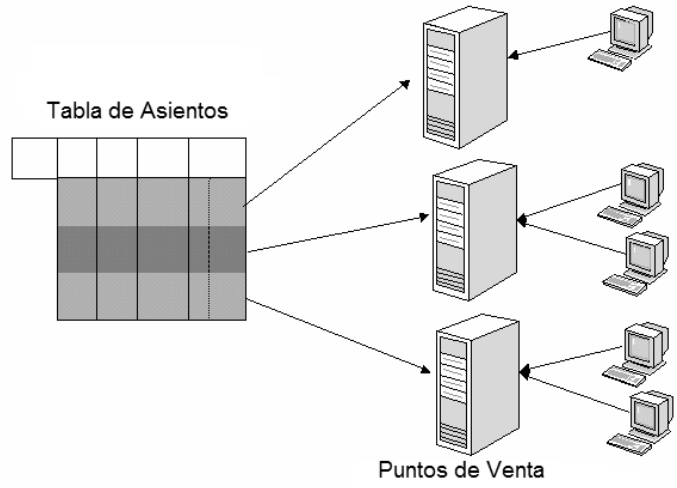
Es posible repartir los datos sobre varios sites con el objetivo de que cada site trabaje con su propio juego de datos, estrictamente distinto de los demás. De esta manera, las transacciones que intervienen sobre cada site sólo ponen en juego los datos del site y la coherencia global se conserva. Si, por ejemplo, cada agencia tiene un archivo de clientes sobre una zona geográfica bien determinada, un cliente sólo puede ser gestionado por una única agencia y toda fuente de conflicto queda excluida.

El particionamiento de los datos permite evitar todo conflicto de datos, lo que es preferible, ya que la resolución de los conflictos es un proceso pesado que demanda mucho tiempo de máquina. Cuanto más numerosos son los conflictos, más difícil es gestionar la situación.

El particionamiento de los datos permite funcionar con una coherencia de datos latente ya que cada site sólo modifica su propio juego de datos. La aplicación de esta coherencia es menos laboriosa que la coherencia transaccional inmediata que se basa en un proceso de validación en dos fases.

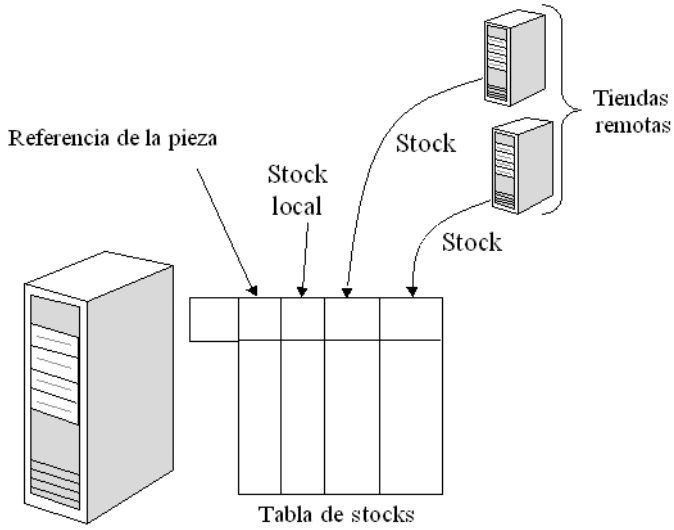
Este tipo de particionamiento no se debe confundir con el particionamiento de tablas. En el marco de la replicación, se define una partición lógica, mientras que en el marco de una tabla particionada, se define una partición física de la tabla.

Ejemplos:



Ejemplo de particionamiento horizontal

Las entradas a espectáculos se venden en varios puntos de venta. Para que el mismo asiento no se venda dos veces, la operación más sencilla de ejecutar consiste en asignar a cada punto de venta un número de asientos. Por lo tanto, la sala se particiona en función de los puntos de venta. Como consecuencia, cada punto de venta gestiona de manera autónoma los asientos que tiene asignados. Por el contrario, cada punto de venta puede saber qué asientos aún no han sido vendidos por los otros puntos de venta. Aquí la coherencia transaccional latente es suficiente.



Ejemplo de particionamiento vertical

Cada punto de venta de una cadena de tiendas de reparación de automóviles gestiona su propio stock de mercancías y por medio de la informática conoce el stock de los puntos de venta más próximos. El conocimiento de este stock se utilizará cuando falte una pieza y sea necesario satisfacer al cliente lo más rápido posible. Aquí la coherencia transaccional inmediata no es necesaria ya que lo que se necesita es un conocimiento general del stock de la tienda más cercana (accesible únicamente en modo lectura).

4. Tipos de replicación

Existen tres tipos de replicación proporcionados por SQL Server:

- Captura instantánea,
- Transaccional: captura instantánea y actualización inmediata de los suscriptores,
- Fusión.

Cada uno de los tipos de replicación responde a una necesidad bien concreta. Siguiendo el método seleccionado bien convergen los datos, o bien se garantiza la coherencia de las transacciones.