Logotipo

Descripción generada automáticamente con confianza media

**Universidad Nacional del Nordeste**

Imagen que contiene edificio, ventana, competencia de atletismo, dibujo

Descripción generada automáticamente

**Facultad de Ciencias Exactas y Naturales y Agrimensura**

Cátedra: Base de Datos 1

Año: 2023

*Grupo 10: Réplicas de bases de datos*

ALUMNOS

|  |  |
| --- | --- |
| Garay, Ruben Ernesto | 38274 |
| Goytia, Jeremias Jesus | 53850 |
| Peralta, Santiago Martín | 54916 |
| Rebollo, Facundo Gabriel | 54786 |
| Zarate, Enzo Javier | 53434 |

PROFESOR: *Lic. Darío Oscar Villegas*

Contenido

[**CAPÍTULO 1: INTRODUCCIÓN** 4](#_Toc151641480)

[**1.1.** **Objetivos Específicos** 4](#_Toc151641481)

[**CAPÍTULO 2: Réplicas de bases de datos** 5](#_Toc151641482)

[**2.1. ¿Qué es la replicación de datos?** 5](#_Toc151641483)

[**2.2. Ejemplo de Replicación** 6](#_Toc151641484)

[**2.2.1. Base de Datos Principal** 6](#_Toc151641485)

[**2.2.2. Réplicas** 6](#_Toc151641486)

[**2.2.3. Proceso de Replicación** 7](#_Toc151641487)

[**2.2.4. Tolerancia a Fallos y Rendimiento** 7](#_Toc151641488)

[**2.3. Beneficios de la Replicación de Datos** 7](#_Toc151641489)

[**2.3.1. Escalabilidad horizontal** 7](#_Toc151641490)

[**2.3.2. Seguridad de Datos** 7](#_Toc151641491)

[**2.3.3. Análisis de Datos** 8](#_Toc151641492)

[**2.3.4. Distribución de Datos a Larga Distancia** 8](#_Toc151641493)

[**2.3.5. Recuperación de Datos** 8](#_Toc151641494)

[**2.3.6. Balanceo de Carga** 8](#_Toc151641495)

[**2.4. Desventajas de la Replicación de Datos:** 9](#_Toc151641496)

[**2.4.1. Costo adicional** 9](#_Toc151641497)

[**2.4.2. Complejidad** 9](#_Toc151641498)

[**2.4.3. Consistencia de datos** 9](#_Toc151641499)

[**2.4.4. Complejidad de la recuperación de desastres** 9](#_Toc151641500)

[**2.4.5. Mayor complejidad de respaldo** 9](#_Toc151641501)

[**2.4.6. Riesgo de errores humanos** 9](#_Toc151641502)

[**CAPÍTULO 3: Metodología de Réplicas** 10](#_Toc151641503)

[**3.1. Metodología Maestro-Esclavo** 10](#_Toc151641504)

[**3.1.1. Maestro (Master)** 10](#_Toc151641505)

[**3.1.2. Esclavo (Slave)** 10](#_Toc151641506)

[**3.2. Replicación Tipo Snapshot (Instantánea)** 12](#_Toc151641507)

[**3.2.1. Características de la Replicación tipo Snapshot** 12](#_Toc151641508)

[**3.2.2. Ventajas de la Replicación tipo Snapshot** 12](#_Toc151641509)

[**3.2.3. Desventajas de la Replicación de tipo Snapshot** 13](#_Toc151641510)

[**3.2.4. Casos de Uso Replicación Tipo Snapshot** 13](#_Toc151641511)

[**3.3. Replicación de Tipo Transaccional** 15](#_Toc151641512)

[**3.3.1. Características** 15](#_Toc151641513)

[**3.3.2. Ventajas de la Replicación Transaccional** 17](#_Toc151641514)

[**3.3.3. Desventajas de la Replicación Transaccional** 17](#_Toc151641515)

[**3.3.4. Casos de Uso Replicación Transaccional** 18](#_Toc151641516)

[**3.4. Replicación de Tipo P2P (Peer to Peer)** 19](#_Toc151641517)

[**3.4.1. Características de la Replicación P2P** 19](#_Toc151641518)

[**3.4.2. Ventajas de la Replicación P2P** 20](#_Toc151641519)

[**3.4.3. Desventajas de la Replicación P2P** 20](#_Toc151641520)

[**3.5. Replicación de Tipo Mezcla (O Merge)** 21](#_Toc151641521)

[**3.5.1. Características de la Replicación de Tipo Mezcla** 21](#_Toc151641522)

[**3.5.2. Ventajas de la Replicación Tipo Mezcla** 22](#_Toc151641523)

[**3.5.3. Desventajas de la Replicación de Tipo Mezcla** 22](#_Toc151641524)

[**3.5.4. Casos de Uso Replicación Tipo Mezcla** 22](#_Toc151641525)

[**CAPÍTULO 4: Paso a Paso del Desarrollo de la Réplica en Metodología Maestro-Esclavo** 23](#_Toc151641526)

[**4.1. PASO 1 – Instalación VM con Windows Server 2019** 23](#_Toc151641527)

[**4.2. PASO 2 – Creación Carpeta para Replicación** 26](#_Toc151641528)

[**4.3. PASO 3 – Configuración Permisos Carpeta Compartida** 27](#_Toc151641529)

[**4.4 PASO 4 – Configuración Funciones SQL Server Replication** 28](#_Toc151641530)

[**4.5. PASO 5 – Creación de una nueva publicación** 31](#_Toc151641531)

[**4.6. PASO 6 – Selección tipo de Publicación** 32](#_Toc151641532)

[**4.7. PASO 7 – Selección Datos a Mezclar** 33](#_Toc151641533)

[**4.8. PASO 8 – Conexión Cuenta de Replicación** 36](#_Toc151641534)

[**4.9. PASO 9 – Creación de la Suscripción** 41](#_Toc151641535)

[**4.10. PASO 10 – Nombramiento Base de Datos** 45](#_Toc151641536)

[**4.11. PASO 11 – Configuración Agent Schedule** 46](#_Toc151641537)

[**4.12. PASO 12 – Selección Tipo de Suscriptor** 47](#_Toc151641538)

[**CAPITULO 5: Conclusión** 49](#_Toc151641539)

[**CAPÍTULO 6: Información Adicional** 51](#_Toc151641540)

[**6.1. Procesos y Enfoques de Trabajo** 51](#_Toc151641541)

[**6.2. Herramientas y Software Utilizados** 51](#_Toc151641542)

# **CAPÍTULO 1: INTRODUCCIÓN**

El presente Trabajo Práctico aborda el tema de la "Replicación de Bases de Datos". Este tema implica la investigación y comprensión de cómo las organizaciones gestionan y mantienen copias de sus bases de datos, con el propósito de mejorar la disponibilidad, redundancia y el rendimiento de los sistemas de información.

Este Trabajo Práctico se centra en el problema de la replicación de bases de datos y cómo se ha convertido en una práctica esencial en la gestión de datos en entornos empresariales. La replicación plantea desafíos y preguntas importantes relacionadas con su implementación, configuración y administración efectiva.

El objetivo general de este trabajo es analizar en profundidad el concepto de replicación de bases de datos y cómo se utiliza para mejorar la disponibilidad, redundancia y el rendimiento de sistemas de información críticos. Al abordar este objetivo, se pretende brindar una visión general del panorama de la replicación de bases de datos.

## **Objetivos Específicos**

* Explorar los distintos tipos de replicación de bases de datos y sus aplicaciones específicas.
* Identificar los beneficios clave de la replicación de datos en entornos empresariales.
* Analizar casos de uso reales de replicación de bases de datos y sus impactos en la continuidad del negocio y el rendimiento de sistemas críticos.
* Examinar desafíos y consideraciones prácticas en la implementación y administración efectiva de la replicación de bases de datos.

Este trabajo tiene como finalidad proporcionar información valiosa sobre un tema fundamental en la gestión de bases de datos y su relevancia en la actualidad. A lo largo de los siguientes capítulos, se abordarán los objetivos especificados para lograr una comprensión completa de la replicación de bases de datos y sus aplicaciones prácticas.

# **CAPÍTULO 2: Réplicas de bases de datos**

La replicación de bases de datos es un proceso en el que se crea y mantiene una copia exacta de una base de datos en uno o varios servidores adicionales, con el propósito de mejorar la disponibilidad, escalabilidad y redundancia de los datos. Esto permite que los cambios realizados en la base de datos original se reflejen automáticamente en las copias, lo que proporciona tolerancia a fallos y la capacidad de distribuir la carga de trabajo entre múltiples servidores, mejorando así el rendimiento y la confiabilidad del sistema. La replicación de bases de datos se utiliza comúnmente en entornos empresariales para respaldar aplicaciones críticas y garantizar la continuidad del negocio.

## **2.1. ¿Qué es la replicación de datos?**

La replicación de datos es cuando los mismos datos se almacenan intencionalmente en más de un sitio o servidor. Hay varias razones por las que las empresas replican datos. Permite que los datos estén disponibles sin problemas en el caso de un tiempo de inactividad del servidor o mucho tráfico hacia el servidor. Los datos se vuelven accesibles para los usuarios de manera constante sin interferir ni ralentizar el acceso de otros usuarios. Para las aplicaciones en la nube, la replicación de datos le permite acceder a una copia de los datos en una base de datos local con un rendimiento mucho mayor que acceder a los datos a través de la API de la aplicación en la nube, que es especialmente útil para análisis y la ciencia de datos. La replicación de datos también puede permitirle evitar los límites de transacción de la API y la aceleración que tienen algunas aplicaciones en la nube.

Diagrama

Descripción generada automáticamente

## **2.2. Ejemplo de Replicación**

Supongamos una empresa de comercio electrónico que opera un sitio web de ventas en línea. En su base de datos principal se encuentra el inventario de productos, cuya actualización es esencial para el funcionamiento del sitio. Para garantizar la disponibilidad ininterrumpida de este recurso crítico y mejorar su rendimiento, se ha implementado la replicación de bases de datos de la siguiente manera:

### **2.2.1. Base de Datos Principal**

La empresa mantiene una base de datos principal en un servidor central. En esta base de datos, se registran todas las transacciones relacionadas con el inventario, como la adición de nuevos productos, actualizaciones de cantidades o cambios de precios.

### **2.2.2. Réplicas**

Para mejorar la disponibilidad, la empresa ha configurado dos servidores adicionales, cada uno con una copia idéntica de la base de datos principal. Estas copias son conocidas como "réplicas."

### **2.2.3. Proceso de Replicación**

Cada vez que se produce un cambio en la base de datos principal, ya sea una actualización del stock de un producto o cualquier otra modificación, se registra y se envía automáticamente a las réplicas. Las réplicas aplican estos cambios de manera automática y sincronizada, asegurando así la consistencia de los datos en todas las instancias.

### **2.2.4. Tolerancia a Fallos y Rendimiento**

La replicación no solo garantiza la disponibilidad continua de datos, sino que también proporciona tolerancia a fallos. Si el servidor principal experimenta una falla, una de las réplicas puede asumir el control, evitando interrupciones en el funcionamiento del sitio web. Además, cuando el tráfico del sitio aumenta, la carga de trabajo se distribuye entre la base de datos principal y las réplicas, lo que mejora el rendimiento global del sistema.

## **2.3. Beneficios de la Replicación de Datos**

### **2.3.1. Escalabilidad horizontal**

La escalabilidad horizontal en la replicación de bases de datos se refiere a la capacidad de distribuir la carga de trabajo de una base de datos replicada al agregar más servidores o nodos al sistema. En lugar de aumentar la potencia de un único servidor o nodo, se añaden servidores adicionales para manejar la carga, lo que permite una mayor capacidad de procesamiento y almacenamiento. Este enfoque se utiliza comúnmente para mejorar el rendimiento y la disponibilidad de sistemas de bases de datos.

### **2.3.2. Seguridad de Datos**

La replicación proporciona una capa adicional de seguridad de datos. Dado que los datos se replican en los esclavos y la replicación se puede pausar, es posible realizar copias de seguridad en los esclavos sin riesgo de corromper los datos en el servidor maestro correspondiente.

### **2.3.3. Análisis de Datos**

Los datos pueden generarse en tiempo real en los servidores maestros, mientras que el análisis de la información puede llevarse a cabo en los esclavos sin afectar el rendimiento de los servidores maestros. Esto facilita la realización de análisis en un entorno de producción sin impactar negativamente en las operaciones en curso.

### **2.3.4. Distribución de Datos a Larga Distancia**

La replicación se utiliza para crear copias locales de datos en sitios remotos, lo que permite el acceso a los datos sin necesidad de una conexión permanente con el servidor maestro. Esto es especialmente útil para aplicaciones que necesitan datos en ubicaciones geográficamente dispersas.

### **2.3.5. Recuperación de Datos**

Las réplicas de bases de datos también pueden utilizarse para recuperar datos históricos o para realizar análisis y generación de informes sin afectar al servidor principal.

### **2.3.6. Balanceo de Carga**

Las réplicas de bases de datos se pueden utilizar para distribuir de manera eficiente el tráfico y equilibrar la carga entre servidores, lo que mejora la capacidad de respuesta del sistema.

## **2.4. Desventajas de la Replicación de Datos:**

### **2.4.1. Costo adicional**

Mantener réplicas de bases de datos implica costos adicionales en términos de hardware, software, ancho de banda y administración. Cuantas más réplicas tengas, mayores serán los gastos.

### **2.4.2. Complejidad**

La configuración y administración de réplicas puede ser compleja, especialmente en entornos de alta disponibilidad o escalabilidad. Requiere experiencia y conocimientos técnicos sólidos.

### **2.4.3. Consistencia de datos**

La replicación en tiempo real puede llevar a problemas de consistencia si no se gestionan adecuadamente las transacciones entre servidores. Asegurarse de que los datos sean coherentes en todas las réplicas puede ser un desafío.

### **2.4.4. Complejidad de la recuperación de desastres**

Aunque las réplicas son valiosas para la recuperación de desastres, su configuración y gestión en escenarios de recuperación pueden ser complicadas.

### **2.4.5. Mayor complejidad de respaldo**

La existencia de réplicas puede requerir estrategias de copia de seguridad más complejas para garantizar que todos los datos estén respaldados adecuadamente.

### **2.4.6. Riesgo de errores humanos**

Los errores en la configuración o administración de réplicas pueden tener un impacto significativo en la integridad de los datos. Se requiere un cuidadoso monitoreo y medidas de seguridad para evitar errores.

# **CAPÍTULO 3: Metodología de Réplicas**

## **3.1. Metodología Maestro-Esclavo**

La metodología maestro-esclavo, también conocida como replicación maestro-esclavo, es un enfoque común en la replicación de bases de datos. En este método, se establece una relación de replicación entre dos servidores de bases de datos: el "maestro" (o "principal") y el "esclavo" (o "secundario"). Aquí se explica cómo funciona:

## **3.1.1. Maestro (Master)**

El servidor maestro es la instancia principal de la base de datos donde se originan las transacciones o cambios en los datos. Este servidor es la fuente autorizada de los datos y registra todas las transacciones.

## **3.1.2. Esclavo (Slave)**

El servidor esclavo es una copia del servidor maestro que recibe y aplica las transacciones del maestro. Los esclavos reflejan los datos y el estado del servidor maestro en tiempo real o con un ligero retraso.

Imagen que contiene Interfaz de usuario gráfica

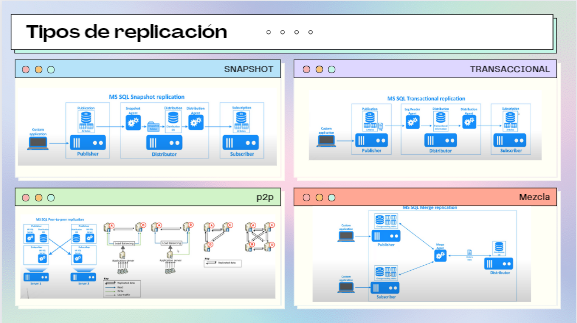
Descripción generada automáticamente

La replicación maestro-esclavo suele ser un proceso asincrónico, lo que significa que los cambios se replican en los servidores esclavos con un cierto retraso desde el servidor maestro. Sin embargo, se esfuerza por mantener la consistencia de datos y garantizar que los cambios se repliquen de manera confiable.

Es importante destacar que esta metodología es un paso esencial en la implementación de soluciones de alta disponibilidad y recuperación de desastres en bases de datos, pero requiere una configuración y administración cuidadosa para garantizar su efectividad y seguridad.

Además de la Metodología Maestro-Esclavo (Considerada la más común para realizar las Réplicas en Bases de Datos), existen 4 Tipos o Metodologías de Replicación más importantes a considerar: La Replicación tipo Snapshot (O también Instantánea), la Replicación tipo Transaccional, la Replicación tipo P2P (Más bien conocida Peer to Peer) y la Replicación de tipo Mezcla (También conocida como tipo Merge).

En este caso, explicaremos las 4 metodologías junto con sus escenarios o casos de uso de los cuáles las replicaciones pueden sacar su mayor provecho para evitar escenarios en los cuales las Bases de Datos creadas no se encuentren en peligro.



## **3.2. Replicación Tipo Snapshot (Instantánea)**

¿Qué es la Replicación Tipo Snapshot?

La Replicación de tipo Snapshot es un enfoque en el que se toma una instantánea (snapshot) de la base de datos principal en un momento específico y se replica esa imagen a las bases de datos secundarias o réplicas. A diferencia de la replicación transaccional o síncrona, la replicación de tipo Snapshot no replica las transacciones individuales en tiempo real, sino que copia toda la base de datos en un solo paso.

### **3.2.1. Características de la Replicación tipo Snapshot**

*- Instantánea:* La replicación de tipo Snapshot captura un estado específico de la base de datos principal en un momento determinado.

*- Período de Latencia:* La replicación se realiza en intervalos programados, lo que significa que puede haber un período de latencia entre la modificación de datos en la base de datos principal y la actualización de las réplicas.

*- Operación en Lotes:* Las actualizaciones se envían a las réplicas en lotes completos en lugar de transacciones individuales, lo que puede simplificar el proceso de replicación.

*- Menor Impacto en el Rendimiento:* En comparación con la replicación síncrona, la replicación de tipo Snapshot tiende a tener un menor impacto en el rendimiento del sistema principal, ya que no hay una sincronización constante en tiempo real.

### **3.2.2. Ventajas de la Replicación tipo Snapshot**

*-Menor Impacto en el Rendimiento:* Dado que la replicación se realiza en intervalos programados, hay menos carga continua en el sistema principal, lo que puede ser beneficioso en términos de rendimiento.

*-Simplicidad:* La replicación de tipo Snapshot puede ser más sencilla de implementar y mantener en comparación con enfoques más complejos como la replicación síncrona.

*-Menor Dependencia de la Conectividad:* No se requiere una conectividad constante entre la base de datos principal y las réplicas, lo que puede ser ventajoso en entornos con limitaciones de red.

### **3.2.3. Desventajas de la Replicación de tipo Snapshot**

*-Período de Latencia:* Debido al enfoque en lotes y la programación de intervalos, puede haber un período de latencia entre la modificación de datos en la base de datos principal y su reflejo en las réplicas.

*-Posible Pérdida de Datos:* Si ocurre un fallo antes de que se realice una nueva instantánea, puede haber una pérdida de datos entre las actualizaciones programadas.

*-Consistencia Menos Inmediata:* La replicación de tipo Snapshot no proporciona consistencia en tiempo real entre las réplicas y la base de datos principal.

La Replicación de tipo Snapshot es adecuada en escenarios donde la consistencia inmediata no es crítica y donde se pueden aceptar ciertos intervalos de latencia en la replicación de datos.

### **3.2.4. Casos de Uso Replicación Tipo Snapshot**

Ahora bien, ¿En qué casos de uso se puede utilizar esta Replicación?

La Replicación de tipo Snapshot puede ser apropiada en varios casos de uso donde la consistencia inmediata no es crítica y donde los intervalos de latencia en la replicación de datos son aceptables.

*-Informes y Análisis:* En entornos donde se generan informes y análisis periódicos, la replicación de tipo Snapshot puede ser suficiente, ya que los informes no requieren datos en tiempo real.

*-Almacenes de Datos (Data Warehouses):* En sistemas de almacenamiento de datos donde se consolidan y almacenan grandes cantidades de datos para análisis posterior, la replicación de tipo Snapshot puede ser eficiente y menos intrusiva.

*-Copias de Seguridad y Restauración:* Para realizar copias de seguridad periódicas de la base de datos principal, la replicación de tipo Snapshot puede ser utilizada para crear réplicas que sirvan como puntos de restauración.

*-Desarrollo y Pruebas:* En entornos de desarrollo y pruebas donde se necesitan conjuntos de datos consistentes para realizar pruebas, la replicación de tipo Snapshot puede ser adecuada.

*-Entornos con Requerimientos de Baja Complejidad:* En situaciones donde se busca una solución más simple y los requisitos de consistencia en tiempo real no son críticos.

*-Aplicaciones con Bajas Tasas de Cambio:* En sistemas donde la tasa de cambio de datos es baja y la necesidad de replicación inmediata no es esencial.

*-Escenarios con Limitaciones de Ancho de Banda:* En entornos donde hay limitaciones significativas en el ancho de banda de la red y la replicación síncrona sería costosa o poco práctica.

Es importante considerar cuidadosamente los requisitos específicos de cada aplicación antes de optar por la Replicación de tipo Snapshot. Si la consistencia inmediata es esencial para la aplicación, se podría considerar otras metodologías de replicación más orientadas a la sincronización en tiempo real, como la replicación síncrona o la replicación transaccional.

## **3.3. Replicación de Tipo Transaccional**

Es un método en bases de datos donde las transacciones individuales realizadas en la base de datos principal se replican de manera precisa y en el mismo orden en una o más bases de datos secundarias, asegurando consistencia e integridad transaccional.

### **3.3.1. Características**

Las características con las que cuenta son las siguientes:

*-Replicación Precisa de Transacciones:* Cada transacción individual realizada en la base de datos principal se replica de manera precisa en las bases de datos secundarias. Esto garantiza que todas las réplicas tengan una copia exacta de las transacciones realizadas.

*-Consistencia en el Orden de las Transacciones:* La replicación transaccional mantiene el orden exacto en el que se realizaron las transacciones en la base de datos principal. Este orden se conserva en todas las réplicas para garantizar la consistencia.

*-Integridad Transaccional:* Preserva la integridad transaccional, lo que significa que una transacción se considera confirmada en una réplica solo después de que se ha replicado de manera exitosa en todas las réplicas.

*-Registro Detallado de Transacciones:* Utiliza un registro detallado de transacciones para capturar y registrar cada cambio realizado en la base de datos principal. Este registro actúa como un historial completo de las operaciones realizadas.

*-Latencia Controlada:* Aunque puede haber cierta latencia en la replicación, se controla para minimizar cualquier discrepancia temporal entre la base de datos principal y las réplicas.

*-Eficiencia en la Replicación:* En comparación con la replicación síncrona, la replicación transaccional puede ser más eficiente en términos de tráfico de red, ya que se replican transacciones en lugar de enviar actualizaciones en tiempo real.

*-Configuración de Filtros:* Permite la configuración de filtros para seleccionar qué datos y transacciones se deben replicar, lo que brinda flexibilidad para adaptarse a requisitos específicos.

*-Escalabilidad:* Puede escalar para manejar grandes volúmenes de transacciones y múltiples réplicas, lo que la hace adecuada para entornos empresariales que experimentan crecimiento.

*-Recuperación ante Fallos:* Ofrece mecanismos para la recuperación ante fallos, asegurando que las réplicas se puedan volver a sincronizar correctamente después de eventos inesperados.

*-Implementación en Topologías Distribuidas:* Puede ser implementada en topologías distribuidas, lo que significa que las réplicas pueden ubicarse en diferentes lugares geográficos para mejorar la disponibilidad y la resistencia a fallos.

Estas características hacen que la Replicación Transaccional sea especialmente adecuada para entornos donde la consistencia, la integridad transaccional y la replicación precisa de transacciones son de alta importancia.

### **3.3.2. Ventajas de la Replicación Transaccional**

*-Consistencia:* Garantiza la consistencia entre la base de datos principal y las réplicas, ya que las transacciones se replican en el mismo orden.

*-Integridad Transaccional:* Preserva la integridad transaccional, ya que todas las transacciones se replican de manera precisa.

*-Eficiencia en la Replicación:* Al replicar solo transacciones individuales, se minimiza el tráfico de red y se optimiza la eficiencia del proceso.

### **3.3.3. Desventajas de la Replicación Transaccional**

*-Impacto en el Rendimiento:* Puede haber un impacto en el rendimiento del sistema principal debido a la captura y registro detallado de cada transacción.

*-Complejidad:* La implementación y gestión de la replicación transaccional pueden ser más complejas en comparación con otros métodos de replicación.

*-Latencia:* La latencia puede ser un problema si las réplicas están geográficamente separadas, ya que las transacciones deben viajar hasta ellas.

La Replicación Transaccional es adecuada para entornos donde la consistencia y la integridad de las transacciones son críticas. Se utiliza comúnmente en aplicaciones financie s, sistemas de gestión de inventarios y otras situaciones donde se requiere una replicación precisa de cada transacción.

### **3.3.4. Casos de Uso Replicación Transaccional**

¿En qué Casos de Uso se puede aplicar la Replicación Transaccional?

-*Reportes y análisis:* Al replicar los datos en casi tiempo real a bases de datos secundarias, estas pueden utilizarse para ejecutar reportes y consultas analíticas sin afectar el rendimiento de la base de datos primaria.

*-Recuperación ante desastres y alta disponibilidad:* Se pueden configurar bases de datos secundarias en ubicaciones remotas que tomen el rol principal en caso de que falle la primaria.

*-Integración de datos entre sistemas:* Los cambios replicados pueden consumirse desde otros sistemas que requieran mantener sincronía con la fuente de datos primaria.

*-Migraciones de bases de datos:* Se puede usar replicación para mantener sincronía temporal entre la base de datos antigua y nueva durante etapas de migración.

*-Pruebas y desarrollo:* Los ambientes de pruebas pueden alimentarse de datos de producción replicados para las pruebas sean más realistas.

## **3.4. Replicación de Tipo P2P (Peer to Peer)**

La replicación punto a punto implica replicar datos directamente de un nodo de publicación a un nodo de suscripción sin pasos intermedios.

En términos más extensos, la replicación punto a punto es un método para copiar y distribuir datos entre múltiples bases de datos SQL de forma directa entre un nodo de "publicación" y uno o varios nodos de "suscripción", sin pasar por intermediarios centralizados.

A diferencia de otros modelos de replicación como el transaccional o de mezcla, en P2P no hay un distribuidor o agentes centrales.

### **3.4.1. Características de la Replicación P2P**

*- Topología flexible:* Permite crear una red de nodos de publicación y suscripción interconectados según las necesidades, incluso en topologías en malla complejas con múltiples caminos de sincronización.

*- Baja latencia:* Como la comunicación es directa entre los nodos, se reducen los retrasos en la replicación comparado con utilizar intermediarios centralizados.

*- Sincronización bidireccional:* Los cambios pueden originarse tanto en el nodo de publicación como en el de suscripción y luego se propagan en ambas direcciones para mantener consistencia.

*- Propagación asíncrona:* Un nodo puede aplicar cambios de datos provenientes de otro nodo incluso si el emisor no está activo en ese momento. Facilita la desconexión intermitente.

*- Resolución de Conflictos Personalizada:* Al no haber un mecanismo de resolución de conflictos predeterminado, permite codificar reglas propias para fusionar cambios divergentes entre nodos.

*- Seguridad y encriptación configurables:* Se puede proteger la comunicación entre nodos con certificados, firmas digitales, VPNs, según se requiera.

### **3.4.2. Ventajas de la Replicación P2P**

*-Simplicidad:* No requiere un distribuidor centralizado o agentes complejos. La comunicación es directa entre los nodos.

*-Menos sobrecarga:* Al no tener componentes intermedios, se reduce la sobrecarga del sistema.

*-Flexibilidad:* Permite crear topologías en malla entre múltiples nodos según sea necesario.

### **3.4.3. Desventajas de la Replicación P2P**

*-Administración compleja en escala*: Puede volverse más desafiante de gestionar con muchos nodos.

*-Sin Resolución de Conflictos automática:* Requiere codificar la lógica de resolución de conflictos personalizada.

*-Requiere más desarrollo:* Hay una mayor carga de desarrollo inicial para configurar la topología.

En resumen, la replicación P2P intercambia simplicidad y menos sobrecarga por más responsabilidad en el desarrollador para gestionar la topología, escalabilidad y resolución de conflictos. Es útil para casos de uso simples de replicación bidireccional.

## **3.5. Replicación de Tipo Mezcla (O Merge)**

La replicación de tipo mezcla es un método el cual permite que los datos se puedan modificar en el servidor principal de publicación o en uno o más suscriptores, y luego se intercambian y sincronizan los cambios entre los nodos bidireccionalmente.

Esto significa que puede haber múltiples fuentes de cambios de datos, en contraste con otros tipos de replicación donde solo el publicador origina cambios. Los nodos actúan tanto como publicadores y suscriptores.

### **3.5.1. Características de la Replicación de Tipo Mezcla**

*-Actualizaciones en múltiples nodos:* A diferencia de otros tipos como la transaccional, cualquiera de los nodos participantes puede modificar datos, en lugar de solo uno central.

*-Acumulación local de cambios:* Los cambios no se envían de inmediato. Se guardan temporalmente en tablas locales en cada nodo mientras hay desconexión.

*-Intercambio programado de cambios:* Basado en una agenda o bajo demanda, los nodos establecen una sesión RPC para intercambiar bidireccionalmente los cambios acumulados.

*-Detección automática de conflictos:* Mediante algoritmos y uso de timestamp, se identifican cambios en conflicto sobre los mismos registros en diferentes nodos.

*-Resolución configurable de conflictos:* A través de reglas y prioridades predefinidas, se resuelven los conflictos automáticamente, o se dejan para resolución manual.

*-Aplicación de cambios y sincronización:* Finalmente se aplican en cada nodo los cambios faltantes, quedando los datos sincronizados integralmente.

### **3.5.2. Ventajas de la Replicación Tipo Mezcla**

*-Modo de Trabajo Offline:* Soporta desconexión y cambios locales temporales mientras se está desconectado.

*-Mejoría de Latencia:* Baja latencia de cambios dentro de cada nodo. No depende de conectividad global.

*-Mayor Flexibilidad:* Permite cambios en cualquiera de los nodos según el nivel de necesidad.

### **3.5.3. Desventajas de la Replicación de Tipo Mezcla**

*-Mayor Consumo de Recursos:* Requiere más recursos de procesamiento y almacenamiento en cada nodo.

*-Alto Nivel de Complejidad:* Hay una mayor complejidad en la resolución de conflictos entre nodos.

*-Capacitación Previa del Usuario:* Se necesita capacitación más extensa de usuarios para evitar conflictos.

### **3.5.4. Casos de Uso Replicación Tipo Mezcla**

¿En qué Casos de Uso se podría aplicar esta Replicación?

-Bases de datos distribuidas con equipos que trabajan desconectados frecuentemente (aplicaciones móviles, trabajo de campo).

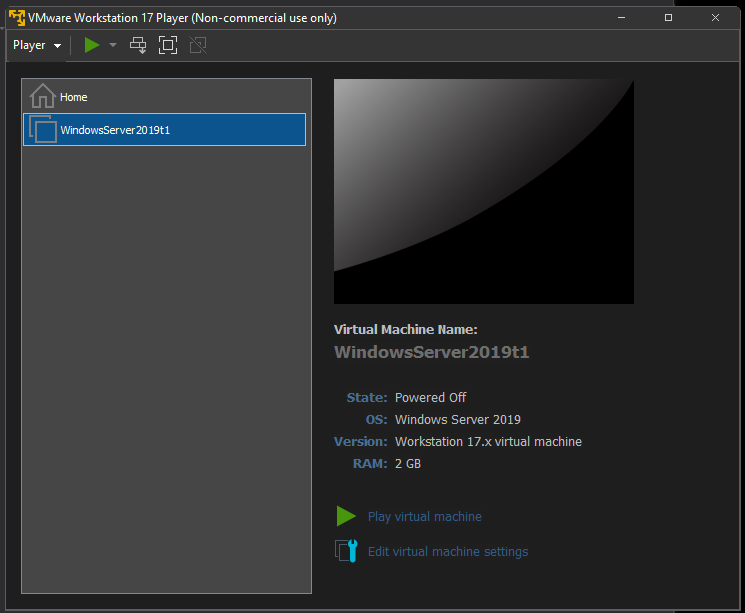
-Integración de sistemas con bases de datos locales independientes que deben sincronizarse.

-Colaboración distribuida en documentos compartidos donde se editan en paralelo.

# **CAPÍTULO 4: Paso a Paso del Desarrollo de la Réplica en Metodología Maestro-Esclavo**

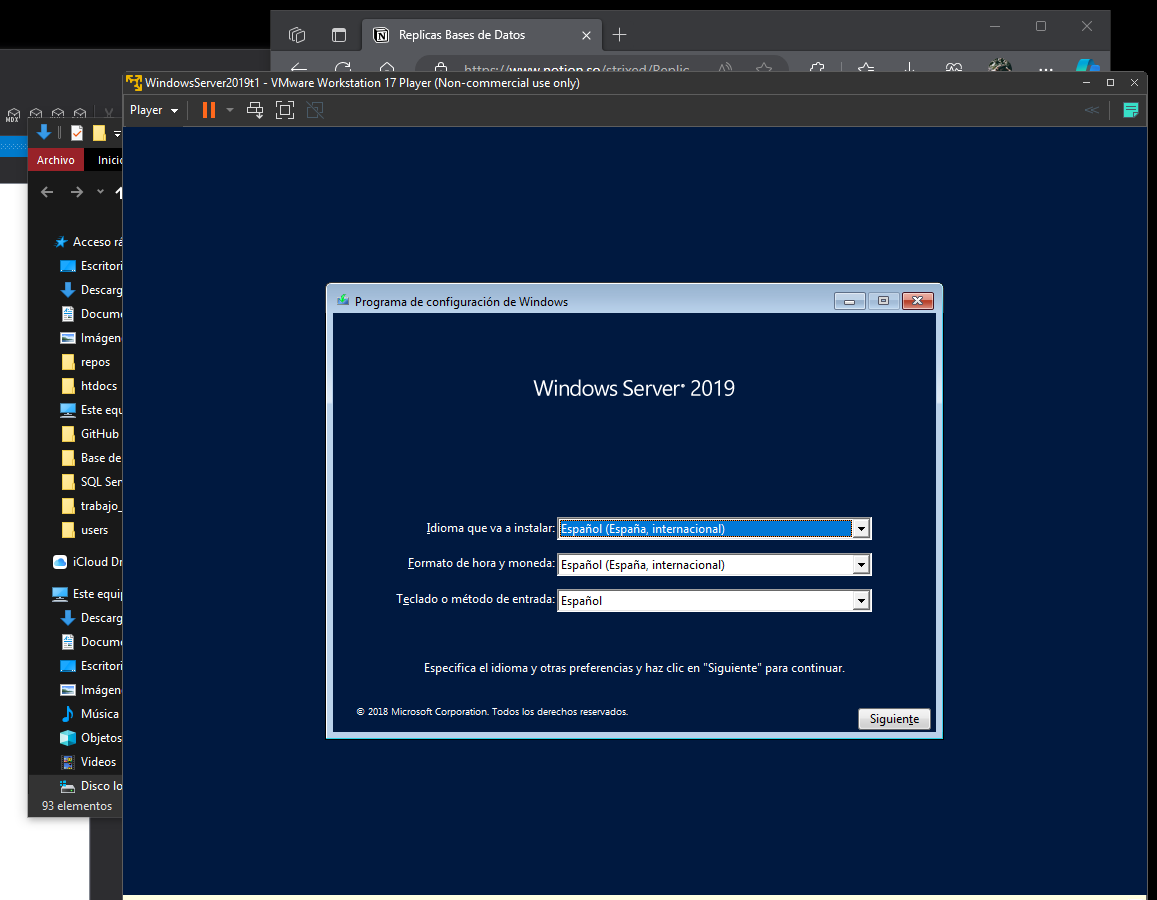
## **4.1. PASO 1 – Instalación VM con Windows Server 2019**

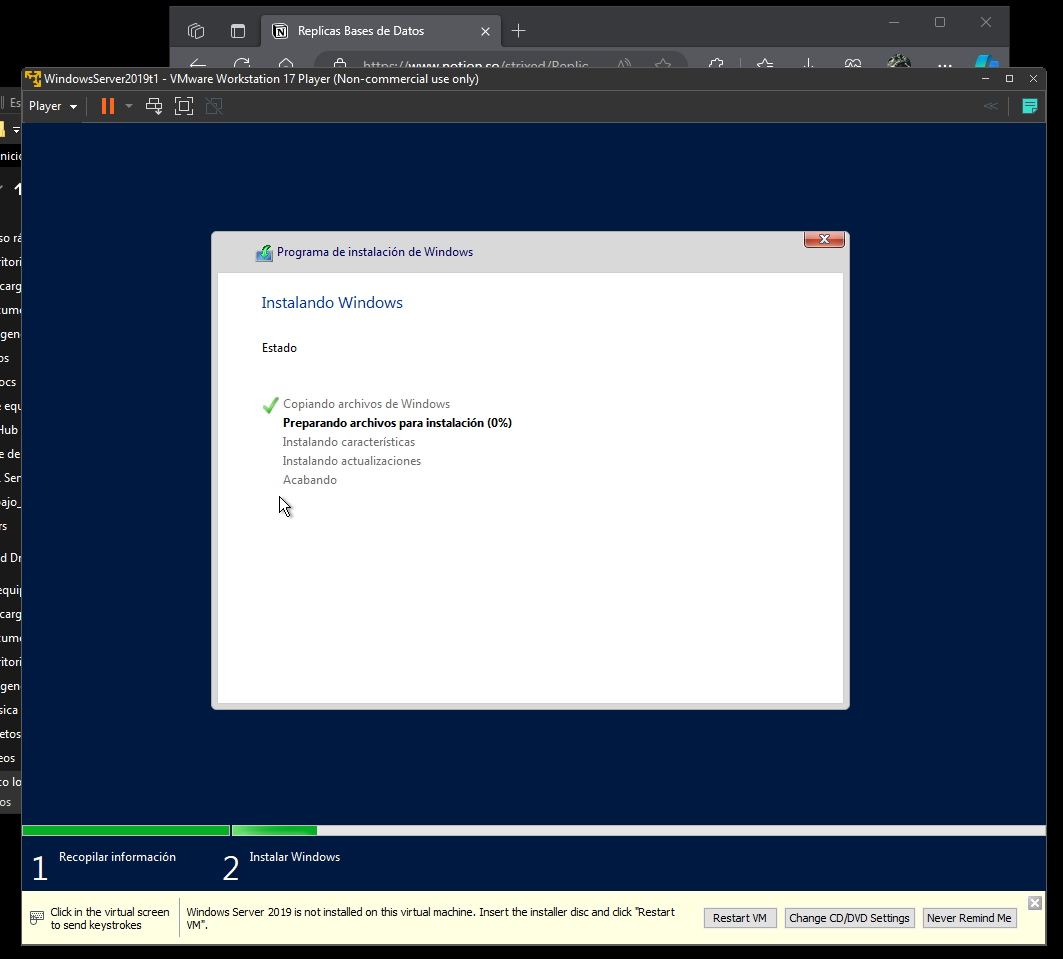
Lo primero a realizar es instalar una máquina virtual en Windows (En este caso usaremos **VMware Workstation** en su versión 17 que cuenta con una versión gratuita), para luego iniciar el sistema operativo **Windows Server 2019** en la misma.

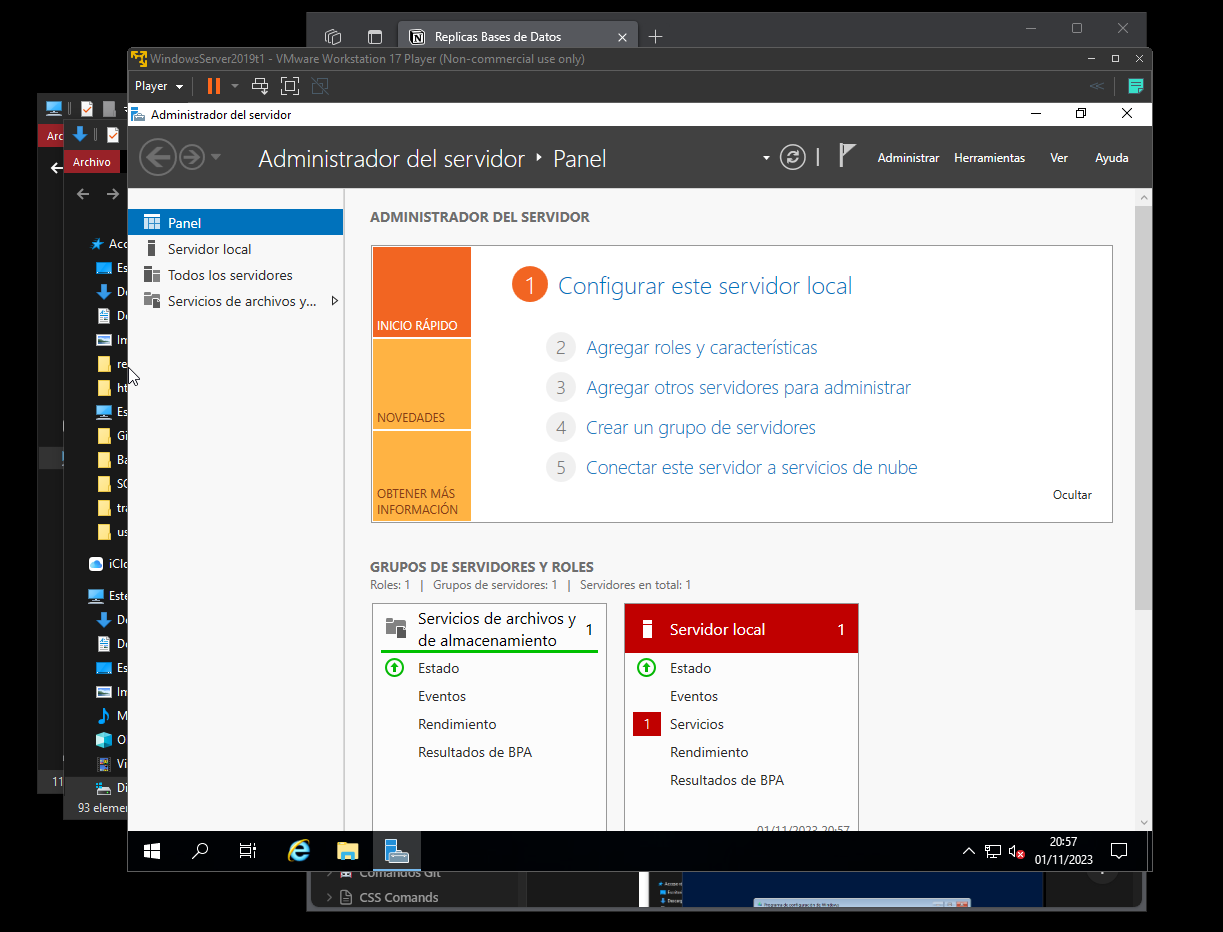


Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

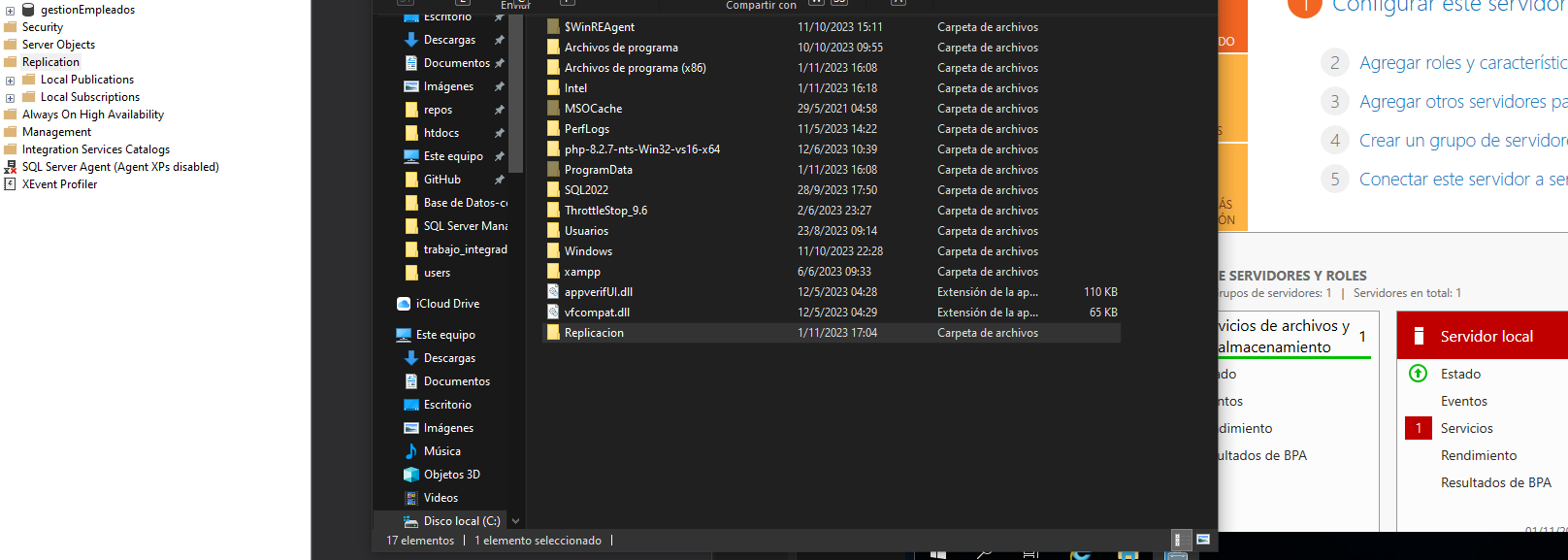






## **4.2. PASO 2 – Creación Carpeta para Replicación**

Una vez hecha la instalación del sistema, crearemos una carpeta en el Disco Local C para poder realizar la replicación (Puede ser cualquier nombre, pero en este caso para darle coherencia usaremos el nombre Replicación).



**4.3. PASO 3 – Configuración Permisos Carpeta Compartida**

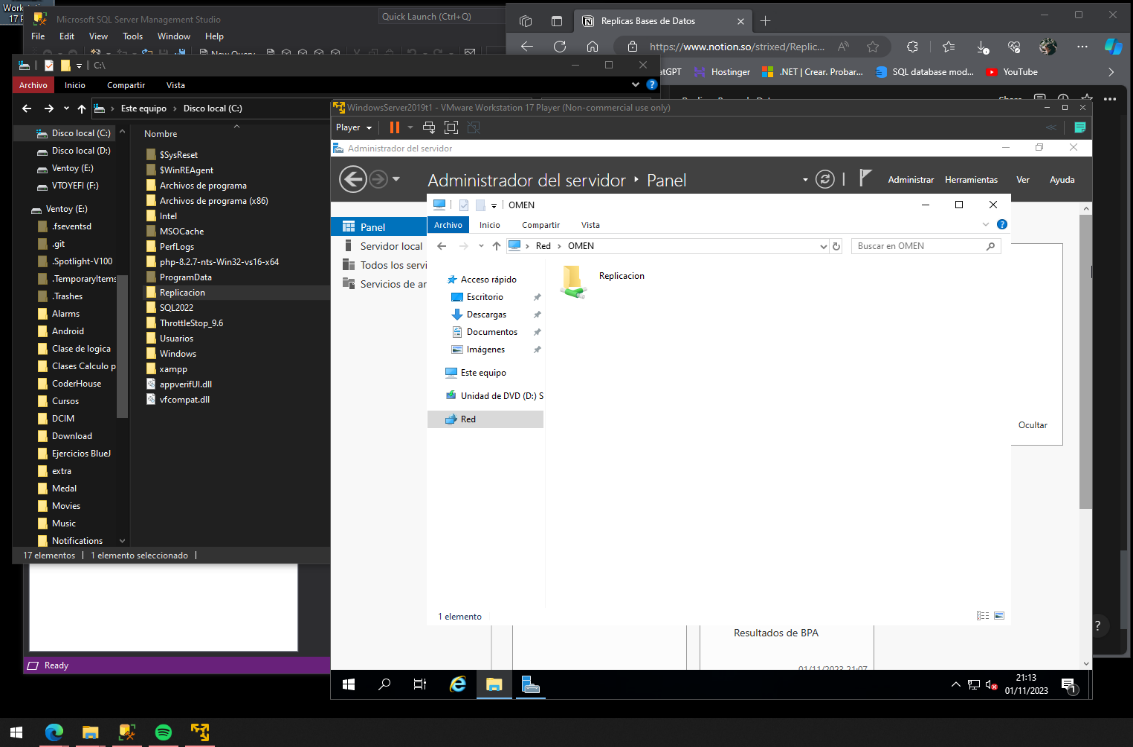
Ya creada la carpeta, accederemos al Uso Compartido Avanzado desde las Propiedades para poder darle los permisos necesarios y así lograr que esta sea un nexo de circulación de archivos entre Maestro y Esclavo.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamenteInterfaz de usuario gráfica

Descripción generada automáticamente

Hechos estos pasos, en la carpeta Red del Servidor Esclavo debería ser posible observar la carpeta con los permisos modificados



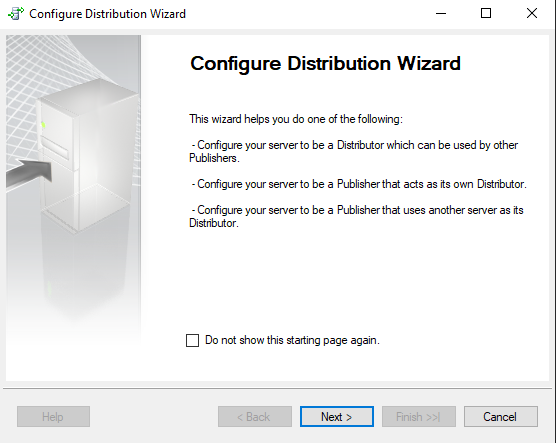
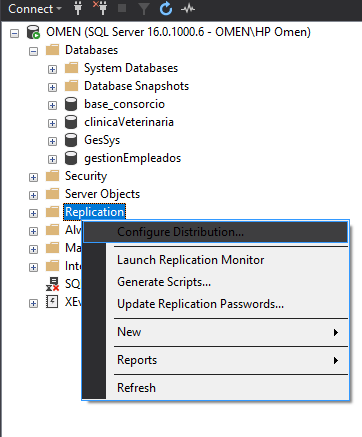
## **4.4 PASO 4 – Configuración Funciones SQL Server Replication**

Algo a tener en cuenta es que, cuando se instale el SQL Server que se utilizará en esta Máquina Virtual, lo que hay que marcar en la parte de las Funciones es el SQL Server Replication, de modo en que nos pueda permitir realizar la replicación sin errores.

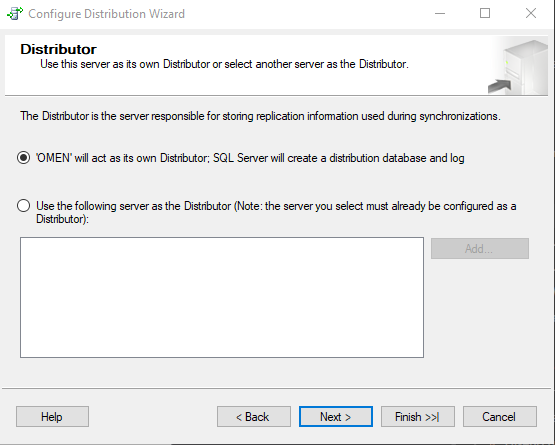
Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

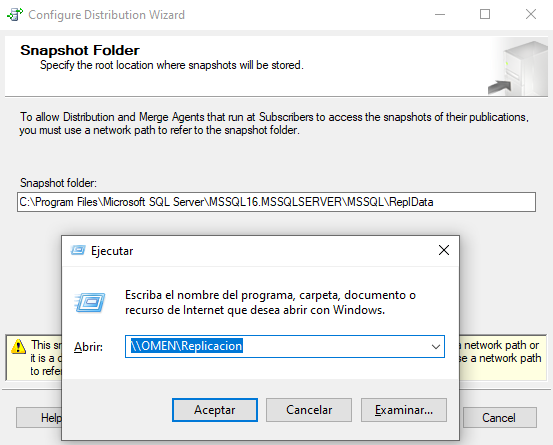
Hecha su instalación, lo siguiente que se deberá de realizar es la configuración del mismo desde la carpeta que se desea replicar.



En este apartado, la mejor opción sería dejar la opción marcada por defecto (es decir, la OMEN).

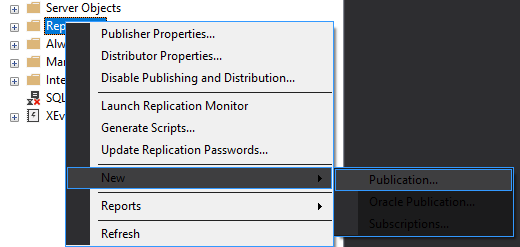


Colocamos la ruta de la carpeta a replicar junto con el directorio OMEN, para luego darle al botón Siguiente hasta Finalizar la configuración.

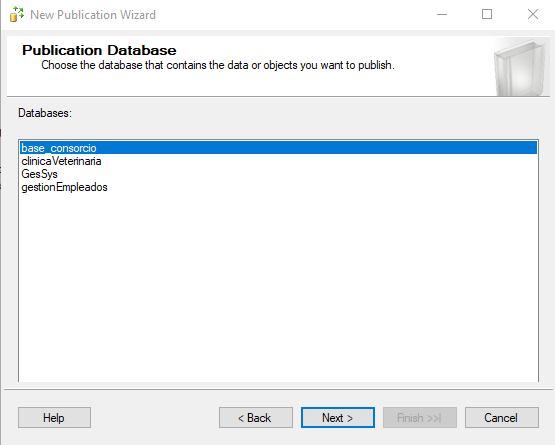


## **4.5. PASO 5 – Creación de una nueva publicación**

Hecha ya la configuración del paso anterior, ahora vamos a crear una nueva publicación en esta carpeta para poder iniciar la replicación.

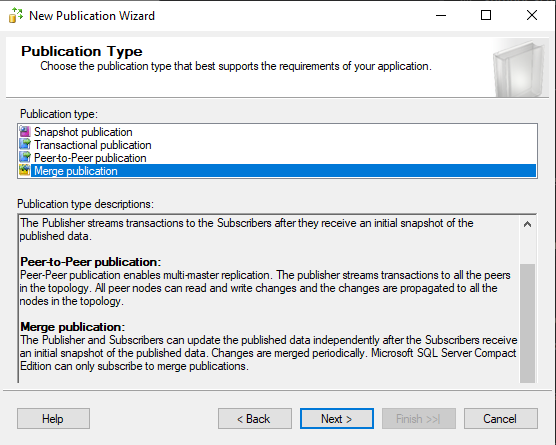


Seleccionamos la base de datos que deseamos replicar, en este caso usaremos la de este proyecto que sería base\_consorcio.

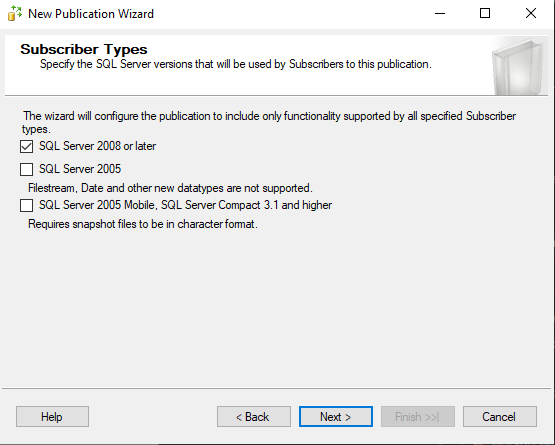


## **4.6. PASO 6 – Selección tipo de Publicación**

En este cuadro de diálogo se dispondrán los distintos tipos de publicación que ofrece la aplicación en sí, en este caso usaremos la Publicación de Mezcla (Merge Publication).

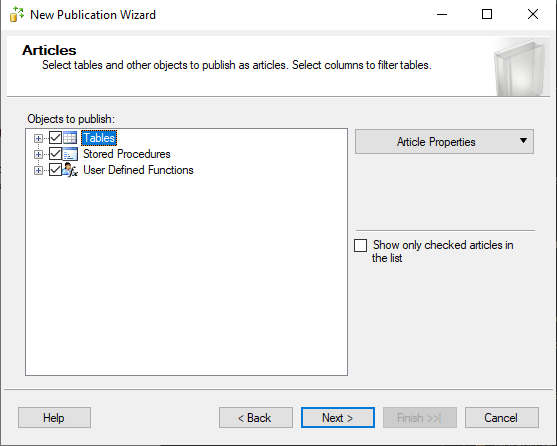


Hecho esto, nos desplazará a otro cuadro donde seleccionaremos las versiones del suscriptor que admitirán compatibilidad con esta publicación, como mejor opción es dejarlo en SQL 2008 o posteriores.

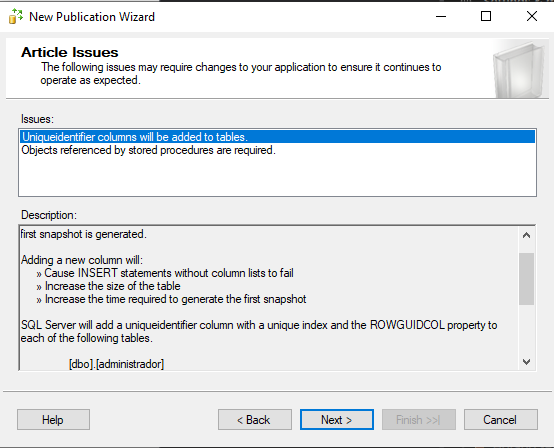


## **4.7. PASO 7 – Selección Datos a Mezclar**

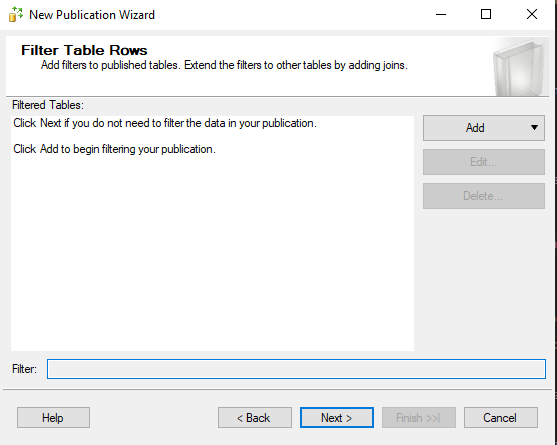
Ahora viene la parte donde veremos qué partes de la base de datos queremos mezclar, por lo que, para conservar todos sus datos intactos lo mejor sería seleccionar todo.

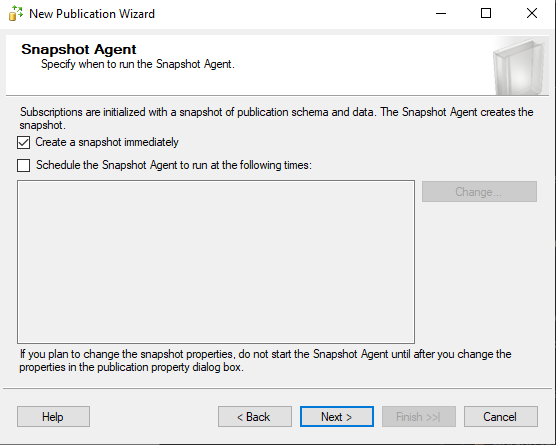


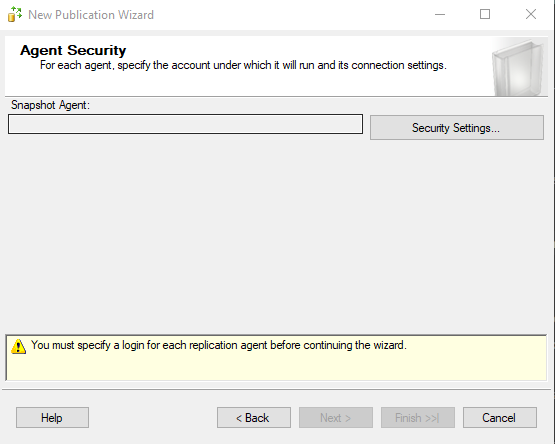
En este cuadro de diálogo es mejor dejar marcada la primera opción que aparece.



Y de aquí solamente es darle Siguiente a todo hasta llegar al cuadro de diálogo donde nos pedirá una ruta de destino.

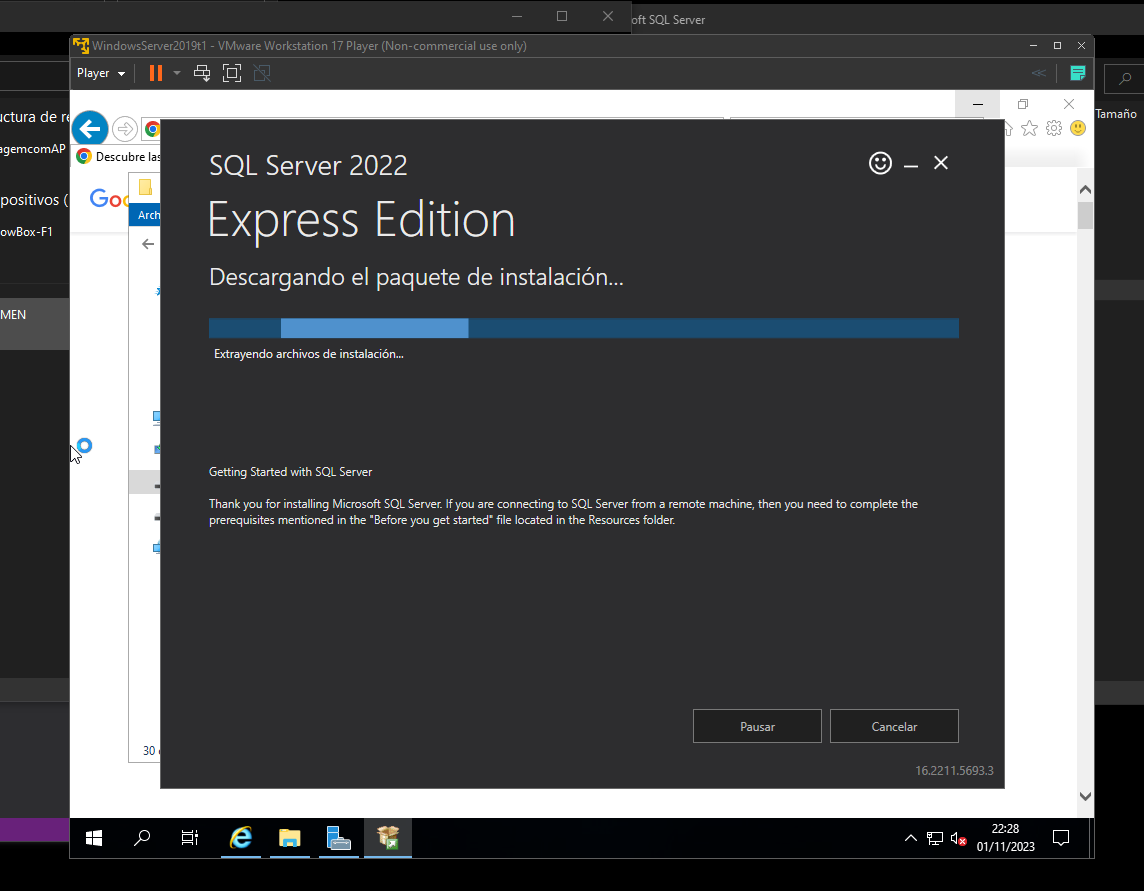


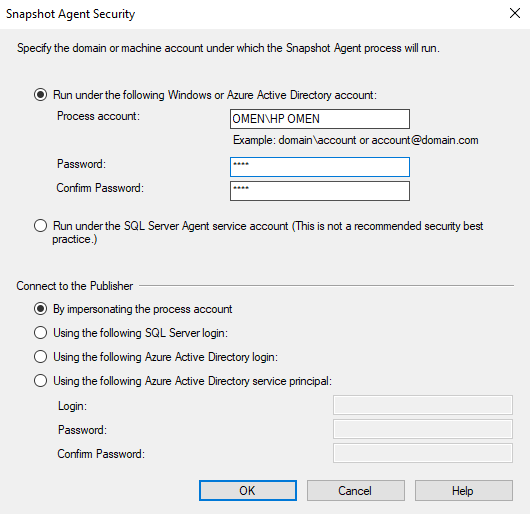




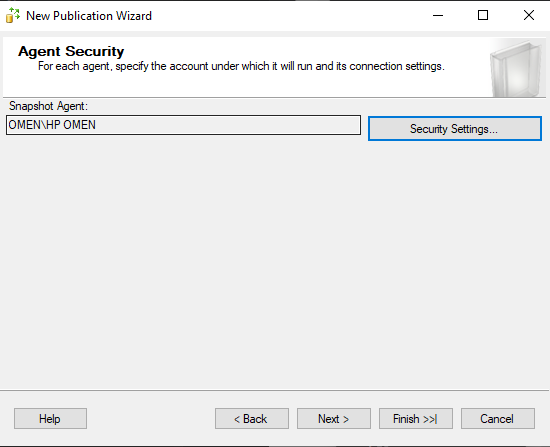
## **4.8. PASO 8 – Conexión Cuenta de Replicación**

Llegado a este punto, el siguiente paso a realizar es conectarse con la cuenta en la cual se realizará la replicación. Lo que se debe hacer es instalar en la Máquina Virtual el SQL Server que servirá como nexo de comunicación (En este caso se utilizará el Express Edition 2022).

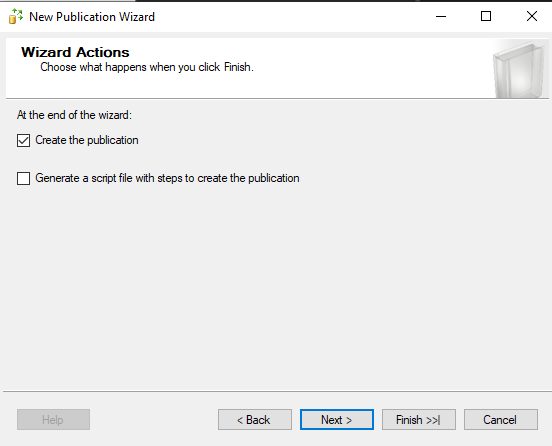


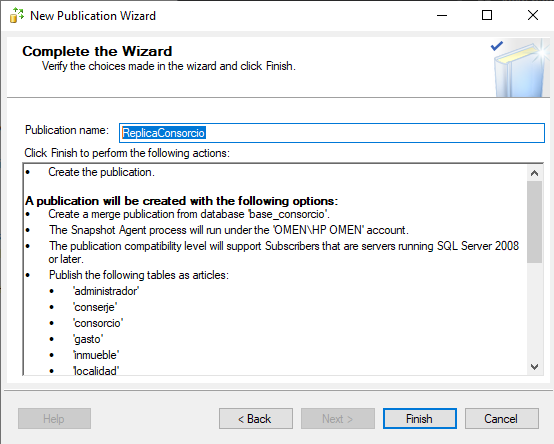


En la PC donde se realiza la replicación colocaremos el Dominio y Contraseña donde tenemos cargada la Base de Datos.

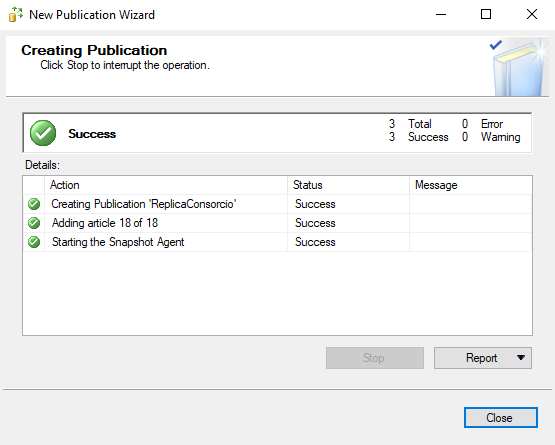


Ya llegado a este punto, el cuadro de diálogo del Agent Security nos debería de reconocer la ruta creada en pasos anteriores para poder ir a los pasos siguientes.



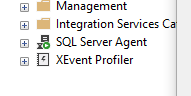


Creamos la publicación sin generar un script, y luego le daremos un nombre representativo a la publicación para no perder la cohesión. Antes de finalizar, el cuadro de diálogo nos mostrará qué tablas se verán publicadas de la base de datos original, el destino del cual va a arrancar, y la compatibilidad en la que ésta será admitida poder correr.

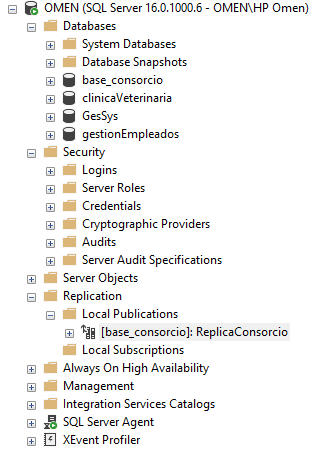


Y aquí veríamos que con éxito se creó la publicación, la cantidad de tablas que añadió y el estado en que se encuentra el Snapshot.

Algo muy importante a tener en cuenta es que el SQL Server Agent debe de estar iniciado.

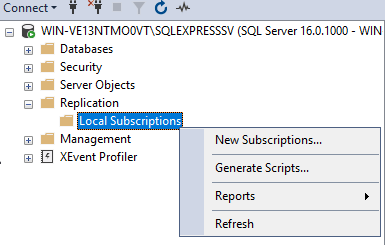


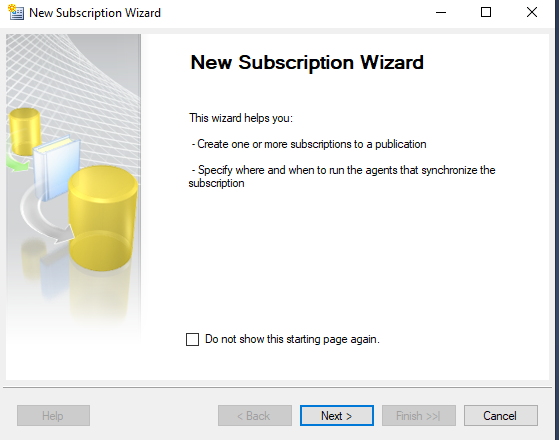
Ya creada la publicación, en nuestro Explorador de Objetos del SQL que utilicemos, nos debería de aparecer esto en el Servidor Físico:



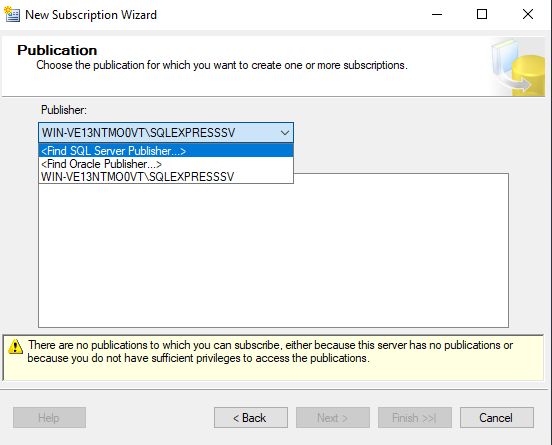
## **4.9. PASO 9 – Creación de la Suscripción**

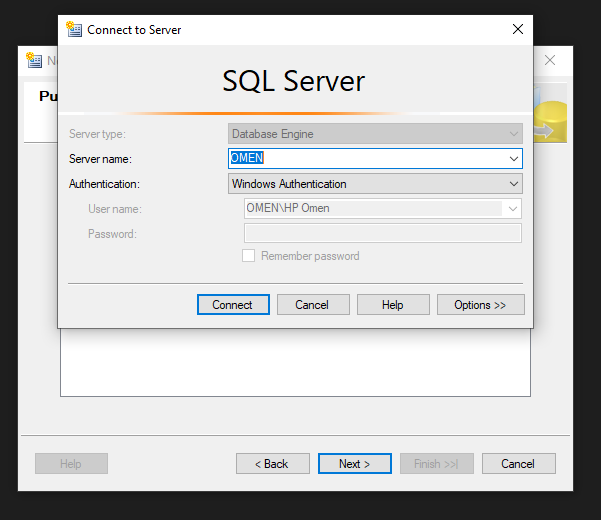
Una vez creada ya la publicación, lo que haremos desde el Servidor Virtual es crear una nueva Suscripción en la carpeta Local Subscriptions de la Carpeta Replication para poder ver si desde el Servidor Físico se crea una réplica.





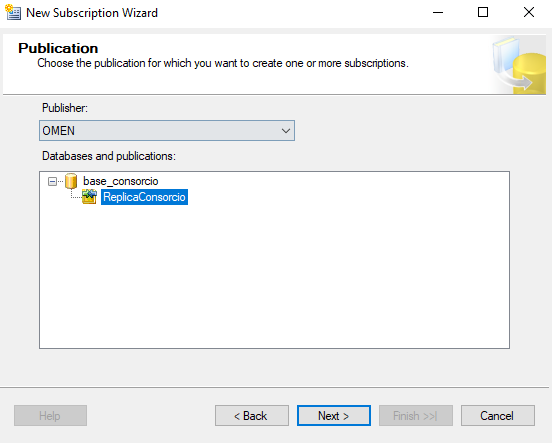
Buscaremos desde el Servidor Virtual nuestro Servidor Publicador (El cual sería nuestro Servidor Físico).



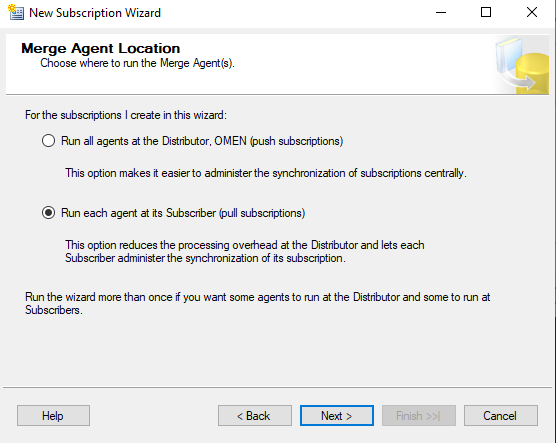


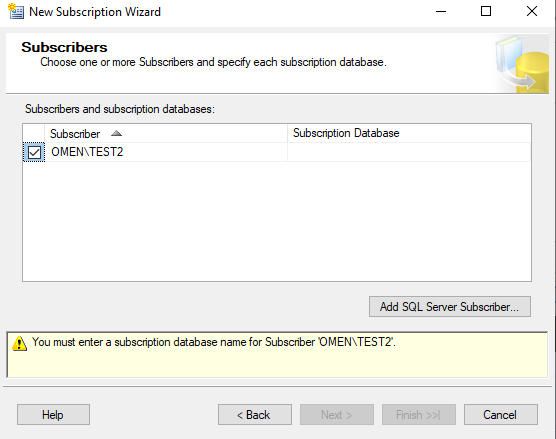
Y en este cuadro de diálogo nos debería permitir seleccionar nuestro Servidor Publicador.

Si los pasos se siguieron correctamente, nos debería permitir observar la base de datos de la cual se desea suscribir.

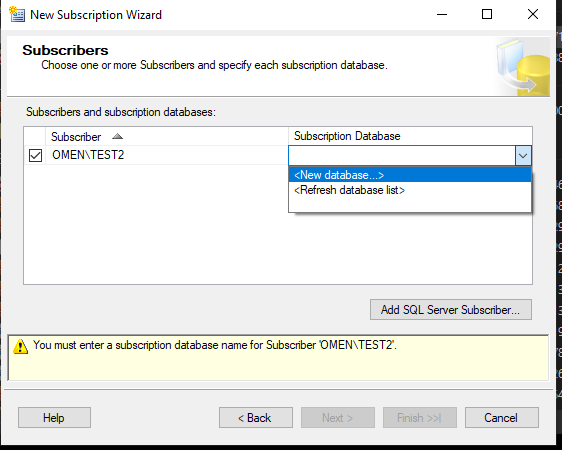


Dejamos la segunda opción marcada en este cuadro.



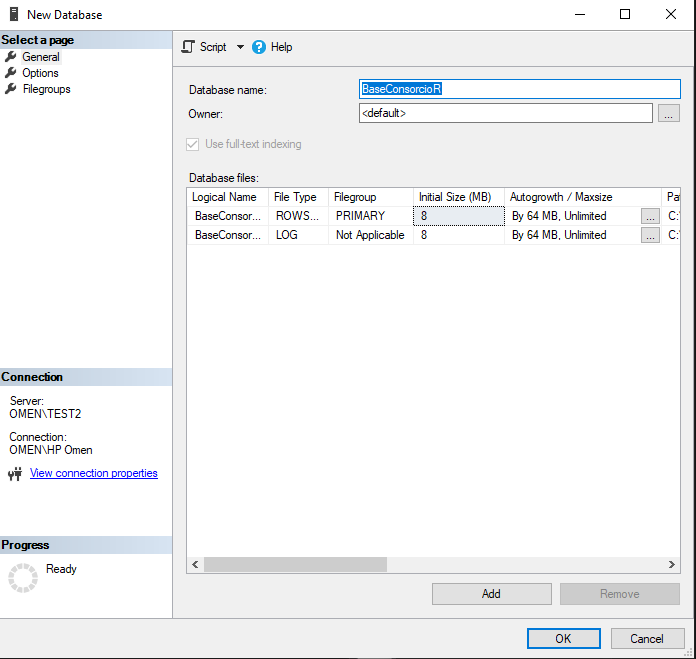


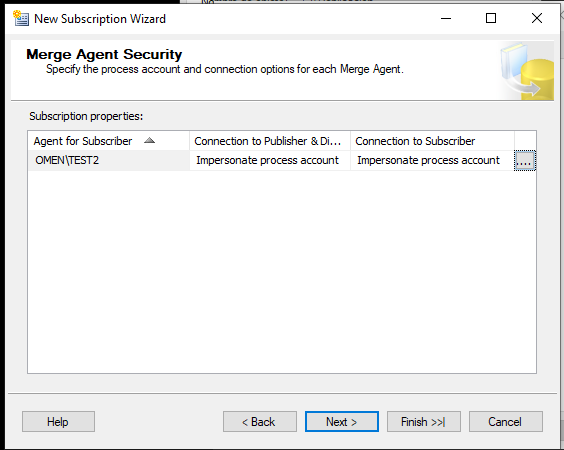
Y en este cuadro seleccionaremos la opción “New Database”.



## **4.10. PASO 10 – Nombramiento Base de Datos**

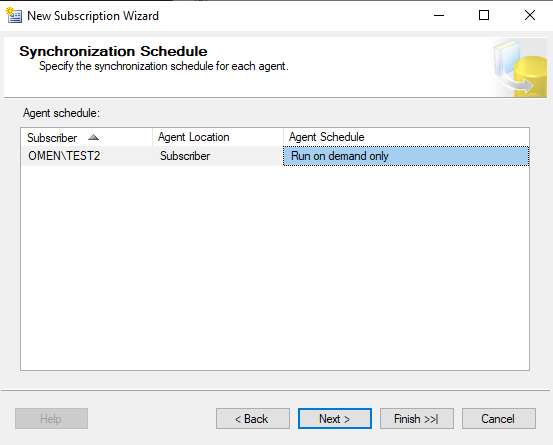
En este paso solamente le daremos el nombre correspondiente, el parámetro ***Owner*** lo dejaremos por “Default”.



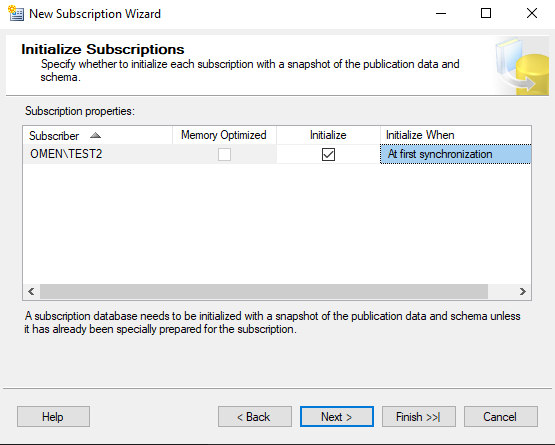


## **4.11. PASO 11 – Configuración Agent Schedule**

En este paso nos debería de aparecer el Agent Schedule en su opción “Run on Demand Only”.

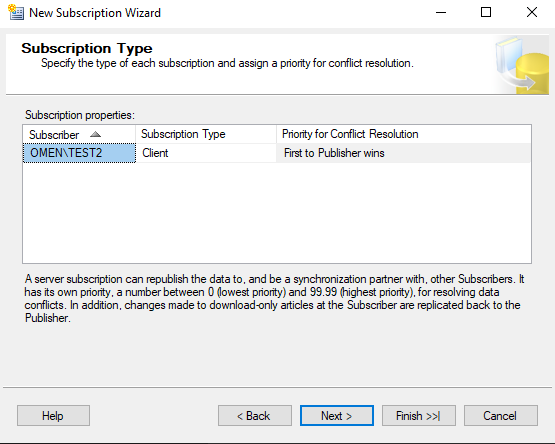


Marcamos la opción “Initialize” en caso de que esta no se encuentre marcada por defecto.

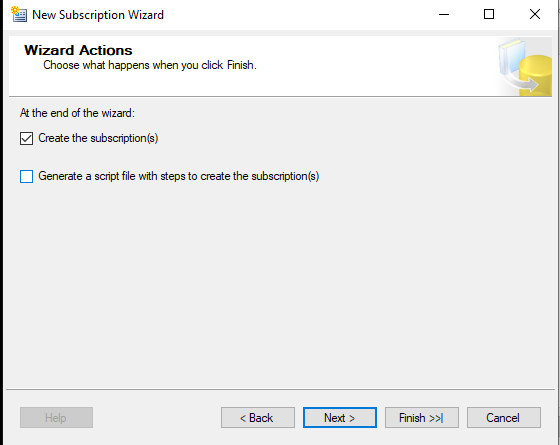


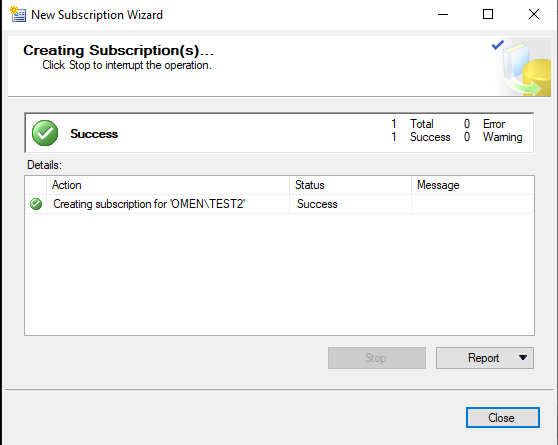
## **4.12. PASO 12 – Selección Tipo de Suscriptor**

Y en este cuadro de diálogo nos debería de aparecer qué tipo de Suscriptor es (En este caso Cliente).

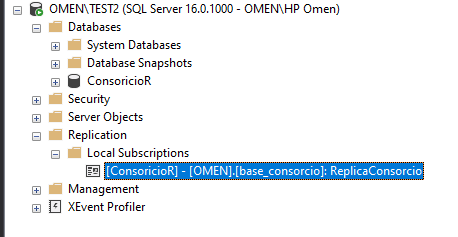


Le daremos a la opción de crear la suscripción sin generar el script.





Y ya hechos los pasos, nos debería de aparecer en nuestro Explorador de Objetos la Suscripción realizada, con la Base de Datos ConsorcioR(La Réplica) creada en el Servidor Suscriptor.



# **CAPITULO 5: Conclusión**

La replicación en bases de datos es una **estrategia poderosa para mantener datos actualizados y consistentes entre diferentes servidores**. A lo largo de este trabajo, hemos explorado cómo la replicación facilita la distribución eficiente de la información, permitiendo su acceso oportuno y mejorando la disponibilidad de los datos en entornos distribuidos.

Durante nuestra investigación, hemos comprendido que la replicación ofrece **flexibilidad** al permitir diferentes tipos de replicación, como la transaccional, de instantánea o de mezcla, cada una adaptándose a necesidades específicas de sincronización, volumen de datos y estructuras de red.

Además, hemos notado que la replicación no solo proporciona redundancia y disponibilidad de datos, sino que también puede mejorar el rendimiento al permitir la distribución de la carga entre servidores. Esta capacidad de escalabilidad y rendimiento es fundamental en entornos donde la eficiencia operativa y la respuesta rápida a las consultas son críticas.

Sin embargo, también hemos reconocido desafíos potenciales al implementar la replicación, como la **complejidad de la configuración**, la necesidad de mantener la coherencia entre los datos replicados y los costos asociados con el ancho de banda y el almacenamiento adicional.

En resumen, la replicación emerge como una **herramienta valiosa para la gestión de datos distribuidos**, ofreciendo una solución dinámica para garantizar la consistencia, disponibilidad y eficiencia en entornos donde la accesibilidad y la sincronización precisa de datos son fundamentales. Su implementación exitosa requiere un equilibrio entre las necesidades operativas específicas y la comprensión de sus implicaciones técnicas y desafíos inherentes.

# **CAPÍTULO 6: Información Adicional**

## **6.1. Procesos y Enfoques de Trabajo**

Se establecieron reuniones semanales para seguimiento y revisión del avance del proyecto. Además, se asignaron responsabilidades específicas a cada miembro del equipo, siguiendo un enfoque colaborativo que facilitó la comunicación y la resolución de problemas de manera ágil.

## **6.2. Herramientas y Software Utilizados**

Durante el desarrollo del proyecto, se emplearon diversas herramientas y software para la gestión, diseño y desarrollo. Algunas de las herramientas clave incluyeron:

* **Gestión de Proyectos:** Se utilizó Trello para la planificación y seguimiento de tareas, asignación de responsabilidades y gestión del flujo de trabajo.
* **Diseño de Base de Datos:** Para el diseño del modelo de datos se empleó SQL Server permitiendo la visualización, creación y modificación del esquema relacional.
* **Desarrollo y Codificación:** Se utilizaron Visual Studio Code para la codificación y desarrollo de scripts o aplicaciones necesarios para la implementación del proyecto.