# Diseño de Índices

**Contenido**

# Notas para el instructor

Este módulo proporciona a los alumnos una introducción al diseño de índices. Explica cómo los índices pueden mejorar el rendimiento de la base de datos. Describe cómo Microsoft® SQL Server™ 2000 almacena índices agrupados y no agrupados y el modo en que SQL Server recupera filas mediante índices. También examina el modo en que SQL Server mantiene los índices. El módulo concluye con directrices para decidir las columnas que se van a indizar.

En la práctica, los alumnos explorarán dos métodos para determinar los índices de una tabla.

Después de completar este módulo, los alumnos serán capaces de:

- Describir por qué y cuándo debe utilizarse un índice.
- Describir cómo SQL Server utiliza índices agrupados y no agrupados.
- Describir cómo la arquitectura de índices de SQL Server facilita la recuperación de datos.
- Describir cómo SQL Server mantiene índices y montones.
- Describir la importancia de la selectividad, densidad y distribución de datos al decidir qué columnas indizar.

# Introducción

**Objetivo del tema**
Proporcionar una introducción a los temas y objetivos del módulo.

**Explicación previa**
En este módulo, aprenderá cuándo y por qué se deben crear índices y los diferentes tipos de índices. Aprenderá cómo SQL Server utiliza y mantiene índices y el modo de diseñar el índice adecuado para sus necesidades.

- Introducción a los índices
- Arquitectura de los índices
- Cómo SQL Server recupera los datos almacenados
- Cómo SQL Server mantiene las estructuras de los índices y los montones
- Decisión de las columnas que se van a indizar

Este módulo proporciona una introducción al diseño de índices. Explica cómo los índices pueden mejorar el rendimiento de la base de datos. Describe cómo Microsoft® SQL Server™ 2000 almacena índices agrupados y no agrupados y el modo en que SQL Server recupera filas mediante índices. También examina el modo en que SQL Server mantiene los índices. El módulo concluye con directrices para decidir las columnas que se van a indizar.

Después de realizar esta práctica, el alumno será capaz de:

- Describir por qué y cuándo debe utilizarse un índice.
- Describir cómo SQL Server utiliza índices agrupados y no agrupados.
- Describir cómo la arquitectura de índices de SQL Server facilita la recuperación de datos.
- Describir cómo SQL Server mantiene índices y montones.
- Describir la importancia de la selectividad, densidad y distribución de datos al decidir qué columnas indizar.

# Introducción a los índices

**Objetivo del tema**
Presentar los índices.

**Explicación previa**
Esta sección describe por qué y cuándo debe utilizarse un índice.

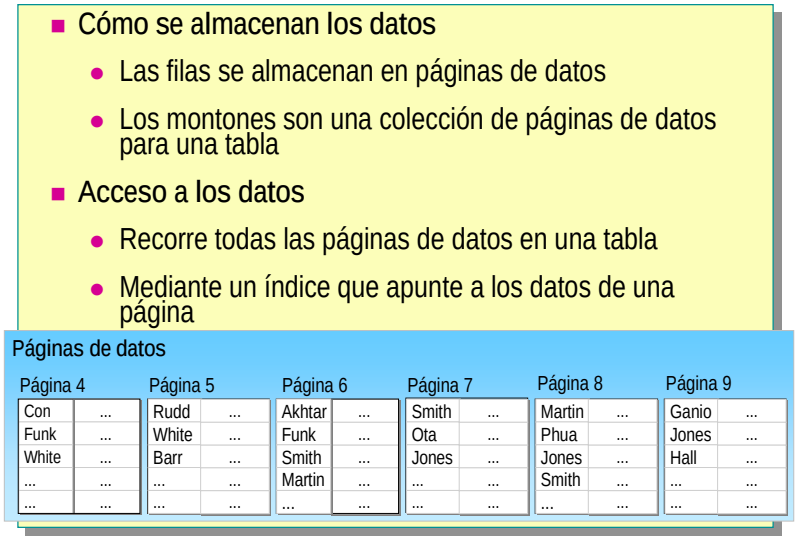
- Cómo SQL Server almacena y tiene acceso a los datos
- Ventajas e inconvenientes de crear índices

El uso de índices puede mejorar extraordinariamente el rendimiento de la base de datos. Esta sección presenta los conceptos básicos acerca de los índices y explica cuándo y por qué utilizarlos.

# Cómo SQL Server almacena y tiene acceso a los datos

**Objetivo del tema**
Explicar cómo SQL Server almacena los datos y permite el acceso a ellos.

**Explicación previa**
Comprender cómo se almacenan los datos es el fundamento para entender cómo SQL Server tiene acceso a ellos.




Comprender cómo se almacenan los datos es el fundamento para entender cómo SQL Server tiene acceso a ellos.

**Sugerencia**
Señale que en la diapositiva sólo se muestran los apellidos en las páginas de datos, aunque éstas almacenan filas completas.

**Nota** En la ilustración sólo se muestran los apellidos en las páginas de datos, aunque éstas almacenan filas completas.

## Cómo se almacenan los datos

Un montón es una colección de páginas de datos que contienen las filas de una tabla:

- Cada página de datos contiene 8 kilobytes (KB) de información. Un grupo de ocho páginas adyacentes se denomina *extensión*.
- Las filas de datos no se almacenan en un orden determinado y tampoco existe un orden específico en la secuencia de páginas de datos.
- Las páginas de datos no están vinculadas en una lista vinculada.
- Cuando se insertan filas en una página llena, ésta se divide.

## Acceso a los datos

SQL Server tiene acceso a los datos de dos maneras:

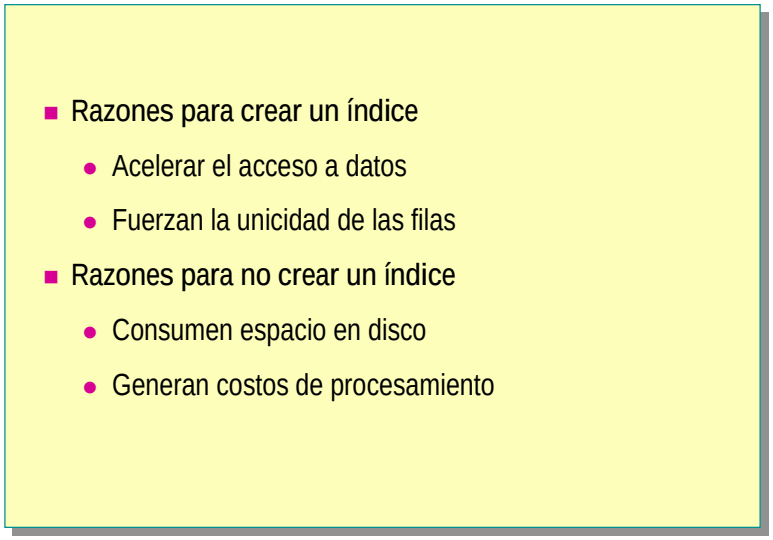
- Recorre todas las páginas de datos de las tablas, en lo que se denomina un *recorrido de tabla*. Cuando SQL Server realiza un recorrido de tabla, sigue estos pasos:
  - Comienza por el principio de la tabla.
  - Recorre página a página todas las filas de la tabla.
  - Extrae las filas que cumplen los criterios de la consulta.
- Utiliza índices. Cuando SQL Server utiliza un índice, hace lo siguiente:
  - Recorre la estructura de árbol del índice para buscar las filas que solicita la consulta.
  - Extrae únicamente las filas necesarias que cumplen los criterios de la consulta.

En primer lugar, SQL Server determina si existe un índice. A continuación, el optimizador de consultas, que es el componente responsable de generar el plan de ejecución óptimo de las consultas, determina si para el acceso a los datos resulta más eficiente recorrer una tabla o utilizar un índice.

# Ventajas e inconvenientes de crear índices

**Objetivo del tema**
Explicar cuándo conviene crear índices.

**Explicación previa**
La creación de un índice no es obligatoria. Veamos por qué puede ser conveniente crear un índice.




Al considerar si debe crear un índice, debe evaluar dos factores para asegurar que el índice va a ser más eficiente que recorrer la tabla: la naturaleza de los datos y la naturaleza de las consultas basadas en la tabla.

## Razones para crear un índice

Los índices aceleran la recuperación de los datos. Por ejemplo, sin un índice, tendría que recorrer todo un libro página a página para encontrar información acerca de un tema específico.

SQL Server utiliza índices para señalar la ubicación de una fila en una página de datos, en lugar de tener que mirar en todas las páginas de datos de la tabla. Al utilizar índices, tenga en cuenta los siguientes hechos y directrices:

- En general, los índices aceleran las consultas que combinan tablas y que realizan operaciones de ordenación o agrupamiento.
- Los índices fuerzan la unicidad de las filas si ésta se define al crear el índice.
- Los índices se crean y mantienen en orden ascendente o descendente.
- Los índices más adecuados son los creados con columnas que tienen un alto grado de selectividad; es decir, columnas o combinaciones de columnas en las que la mayoría de los datos son únicos.

Sugerencia
Pregunta: ¿Son imprescindibles los índices?

Respuesta: No. Es posible consultar y manipular los datos sin tener ningún índice. Sin embargo, el acceso a los datos es considerablemente más lento.

## Razones para no crear un índice

Los índices son útiles, pero consumen espacio en disco y generan costos de procesamiento y mantenimiento adicionales. Al utilizar índices, tenga en cuenta los siguientes hechos y directrices:

- Al modificar los datos de una columna indizada, SQL Server actualiza los índices asociados.
- El mantenimiento de los índices requiere tiempo y recursos. Por lo tanto, no deben crearse índices que no se vayan a usar con frecuencia.
- Los índices basados en columnas que contengan gran cantidad de datos duplicados pueden no suponer apenas ninguna ventaja.

# Arquitectura de los índices

**Objetivo del tema**
Presentar la arquitectura agrupada y no agrupada de los índices.

**Explicación previa**
Esta sección describe cómo SQL Server utiliza índices agrupados y no agrupados.

- Arquitectura de índices de SQL Server
- Uso de montones
- Uso de los índices agrupados
- Uso de los índices no agrupados

La arquitectura de los índices agrupados y no agrupados es distinta. Entender las diferencias de la arquitectura le ayudará a crear índices del tipo más efectivo en cada caso.

# Arquitectura de índices de SQL Server

## Índices agrupados

En un índice agrupado, el nivel de hoja es la página de datos. Los datos están almacenados físicamente en páginas de datos en orden ascendente. El orden de los valores en las páginas de índice también es ascendente.

## Índices no agrupados creados sobre un montón

Cuando se crea un índice no agrupado sobre un montón, SQL Server utiliza en las páginas de índice identificadores de fila que apuntan a filas de las páginas de datos. Los identificadores de fila almacenan información acerca de la ubicación de los datos.

## Índices no agrupados creados sobre un índice agrupado

Cuando un índice no agrupado se crea sobre una tabla con un índice agrupado, SQL Server utiliza una clave de agrupación en las páginas de índice para apuntar al índice agrupado. La clave de agrupación almacena información acerca de la ubicación de los datos.

# Uso de montones

**Objetivo del tema**
Explicar cómo SQL Server utiliza montones.

**Explicación previa**
SQL Server mantiene las páginas de datos en un montón, a menos que se haya definido un índice agrupado en la tabla.

SQL Server:

- Utiliza las páginas de Mapa de asignación de índices que:
  - Contienen información acerca del lugar donde están almacenadas las extensiones de un montón
  - Se utilizan para recorrer el montón y encontrar espacio disponible para insertar nuevas filas
  - Conectan páginas de datos
- Recupera espacio para las nuevas filas del montón cuando se elimina una fila

SQL Server mantiene las páginas de datos en un montón, a menos que se haya definido un índice agrupado en la tabla. SQL Server:

- Utiliza las páginas de Mapa de asignación de índices (IAM, *Index Allocation Map*) para mantener los montones. Las páginas IAM:
  - Contienen información acerca del lugar donde están almacenadas las extensiones de un montón.

    La tabla de sistema **sysindexes** almacena un puntero a la primera página IAM asociada a un montón.
  - Se utilizan para recorrer el montón y encontrar espacio disponible para insertar nuevas filas.
  - Conectan las páginas de datos.

    Las páginas de datos y las filas que hay en ellas no tienen un orden específico y no están vinculadas entre sí. La única conexión lógica entre las páginas de datos es la información registrada en las páginas IAM.
- Recupera espacio para las nuevas filas del montón cuando se elimina una fila.

# Uso de los índices agrupados

**Objetivo del tema**
Explicar algunos hechos acerca de los índices agrupados.

**Explicación previa**
Los índices agrupados son útiles para las columnas en las que se buscan frecuentemente intervalos de valores de clave o a las que se tiene acceso siguiendo un orden.

- Cada tabla sólo puede tener un índice agrupado
- El orden físico de las filas de la tabla y el orden de las filas en el índice son el mismo
- La unicidad de los valores de clave se mantiene explícitamente o implícitamente

**Sugerencia**
Después de explicar los puntos de la diapositiva, pregunte a los alumnos qué le ocurre a un índice agrupado cuando se agregan filas a la tabla.

**Pregunta:** ¿Por qué no es posible tener dos índices agrupados en una tabla?

**Respuesta:** SQL Server sólo almacena una ordenación física de las filas de cada tabla.

Los índices agrupados son útiles para las columnas en las que se buscan frecuentemente intervalos de valores de clave o a las que se tiene acceso siguiendo un orden. Al crear un índice agrupado, tenga en cuenta los siguientes hechos y directrices:

- Cada tabla sólo puede tener un índice agrupado.
- El orden físico de las filas de la tabla y el orden de las filas en el índice son el mismo. Debe crear el índice agrupado antes de crear cualquier índice no agrupado, ya que el primero cambia el orden físico de las filas de la tabla. Las filas se ordenan secuencialmente y se mantienen con esa ordenación.
- La unicidad de los valores de clave se mantiene explícitamente, con la palabra clave UNIQUE, o implícitamente, con un identificador único interno. Estos identificadores únicos son internos de SQL Server y el usuario no tiene acceso a ellos.
- El tamaño medio de un índice agrupado es aproximadamente el cinco por ciento del tamaño de la tabla. Sin embargo, varía en función del tamaño de la columna indizada.
- Cuando se elimina una fila, el espacio se recupera y queda disponible para una fila nueva.
- Durante la creación de un índice, SQL Server utiliza temporalmente espacio de disco de la base de datos actual. Un índice agrupado requiere aproximadamente 1,2 veces el tamaño de la tabla como espacio de trabajo mientras se crea. El espacio de disco que se utiliza durante la creación del índice se recupera automáticamente cuando éste se ha creado.

**Sugerencia**
**Pregunta:** ¿De dónde procede el valor 1,2?

**Respuesta:** 1 = datos y 0,2 = índice. Estos valores son estimaciones conservadoras.

**Nota** Asegúrese de que hay espacio en disco suficiente en la base de datos al crear índices agrupados.

# Uso de los índices no agrupados

**Objetivo del tema**
Explicar algunos hechos acerca de los índices no agrupados.

**Explicación previa**
Los índices no agrupados son útiles cuando los usuarios requieren varios modos de buscar datos.

- Los índices no agrupados son los predeterminados de SQL Server
- Los índices no agrupados existentes se vuelven a generar automáticamente
  - Se quita un índice agrupado existente
  - Se crea un índice agrupado
  - Se utiliza la opción DROP_EXISTING para cambiar las columnas que definen el índice agrupado

Los índices no agrupados son útiles cuando los usuarios requieren varios modos de buscar datos. Por ejemplo, supongamos que un lector busca frecuentemente en un libro de jardinería los nombres corrientes y científicos de plantas. Podría crear un índice no agrupado para obtener los nombres científicos y otro para obtener los nombres corrientes. Al crear un índice no agrupado, tenga en cuenta los siguientes hechos y directrices:

- Si no se especifica un tipo de índice, de forma predeterminada será un índice no agrupado.
- SQL Server vuelve a generar automáticamente los índices no agrupados existentes cuando se produce alguna de las situaciones siguientes.
  - Se quita un índice agrupado existente.
  - Se crea un índice agrupado.
  - Se utiliza la opción DROP_EXISTING para cambiar las columnas que definen el índice agrupado.
- El orden de las páginas del nivel de hoja de un índice no agrupado es distinto del orden físico de la tabla. El nivel de hoja está ordenado de forma ascendente.
- La unicidad se mantiene en el nivel de hoja, ya sea con claves de agrupación o con identificadores de fila.
- Cada tabla puede tener hasta 249 índices no agrupados.
- Los índices no agrupados se deben crear preferiblemente con columnas en las que los datos sean altamente selectivos o únicos.
- Cree los índices agrupados antes que los no agrupados.
- Los identificadores de fila especifican la ordenación lógica de las filas y constan de un Id. de archivo, un número de página y un Id. de fila.

# Cómo SQL Server recupera los datos almacenados

**Objetivo del tema**
Presentar la recuperación de los datos.

**Explicación previa**
Esta sección describe cómo la arquitectura de índices de SQL Server facilita la recuperación de datos.

- Cómo SQL Server utiliza la tabla sysindexes
- Búsqueda de filas sin índices
- Búsqueda de filas en un montón con un índice no agrupado
- Búsqueda de filas en un índice agrupado
- Búsqueda de filas en un índice agrupado con un índice no agrupado

---

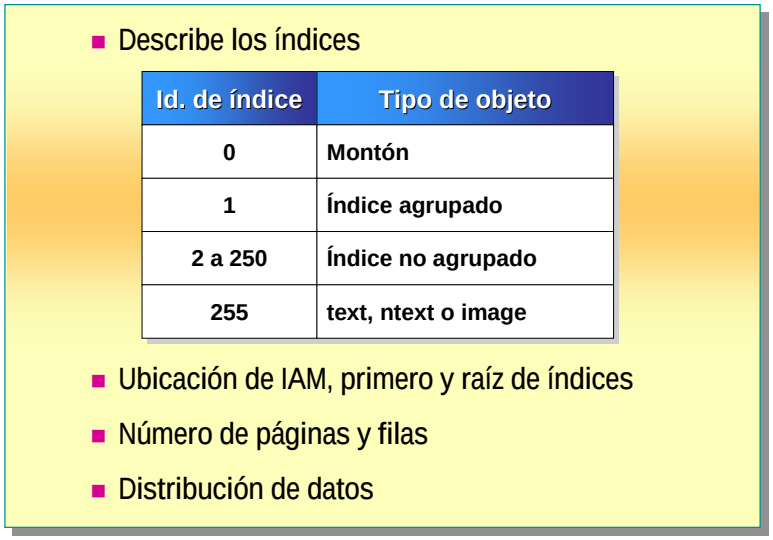
En ella se repite información contenida en los cursos de requisitos previos. Repásela rápidamente si los alumnos tienen una buena compresión de una instrucción SELECT.

Para diseñar bases de datos eficientes, es importante comprender cómo SQL Server almacena los datos. Esta sección describe cómo la arquitectura de índices de SQL Server facilita la recuperación de datos.

# Cómo SQL Server utiliza la tabla sysindexes

**Objetivo del tema**
Describir la función de sysindexes en una búsqueda de datos.

**Explicación previa**
La tabla del sistema sysindexes proporciona el primer paso en una búsqueda de datos.




La tabla del sistema **sysindexes** es la ubicación central donde reside la información vital acerca de tablas e índices. Contiene información estadística, como el número de filas y las páginas de datos de cada tabla y describe cómo encontrar la información almacenada en una tabla de datos.

Los punteros de página de la tabla **sysindexes** delimitan todas las colecciones de páginas de tablas e índices. Cada tabla tiene una colección de páginas de datos, junto con colecciones de páginas adicionales para implementar los índices definidos para la tabla.

Una fila en **sysindexes** de cada tabla e índice se identifica de forma exclusiva mediante la combinación de la columna de identificador de objeto (**id**) y la columna de identificador de índice (**indid**).

## La columna indid

A continuación se indica cómo las columnas de la tabla **sysindexes** ayudan a encontrar páginas de datos para diferentes tipos de objetos:

- Un montón tiene una fila en **sysindexes** con la columna **indid** establecida en cero. La columna **FirstIAM** de **sysindexes** apunta a la cadena de páginas IAM para la colección de páginas de datos de la tabla. SQL Server debe utilizar las páginas IAM para buscar las páginas en la colección de páginas de datos ya que dichas páginas no están vinculadas juntas.
- Un índice agrupado creado para una tabla tiene una fila en **sysindexes** con la columna **indid** establecida en 1. La columna **root** de **sysindexes** apunta a la parte superior del *árbol equilibrado* (árbol B) del índice agrupado.

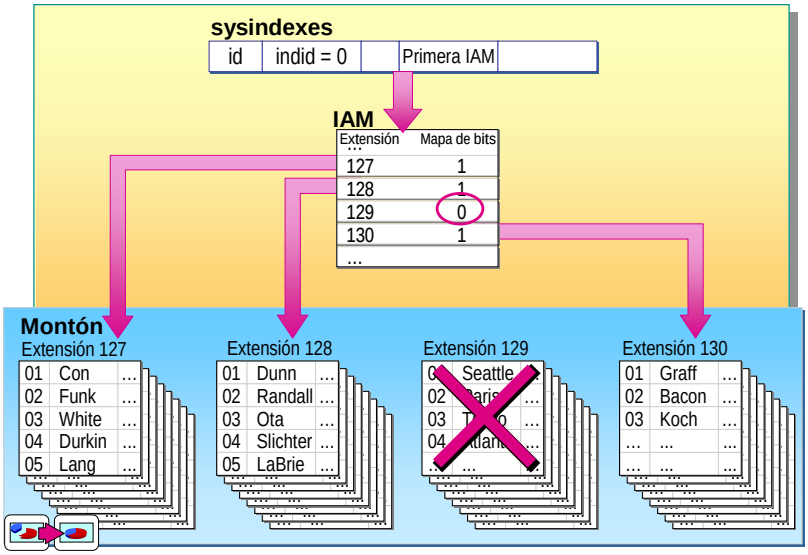
- Cada índice no agrupado creado para una tabla tiene una fila en **sysindexes** con un valor en la columna **indid.** El valor de la columna **indid** de un índice no agrupado oscila entre 2 y 250. La columna **root** de **sysindexes** apunta a la parte superior del árbol B del índice no agrupado.
- Cada tabla que tiene al menos una columna **text**, **ntext** o **image** tiene también una fila en **sysindexes** con la columna **indid** establecida en 255. La columna **FirstIAM** de **sysindexes** apunta a la cadena de páginas IAM que administran las páginas **text**, **ntext** e **image**.

Sugerencia
La información generada por el comando UPDATE STATISTICS también se encuentra en sysindexes.

# Búsqueda de filas sin índices

**Objetivo del tema**
Describir la búsqueda de datos en un montón.

**Explicación previa**
Cuando una tabla no tiene un índice agrupado o índices no agrupados útiles, SQL Server utiliza la página IAM para iniciar un recorrido de tabla.




**Sugerencia**
Ésta es una diapositiva animada. Consulte las Notas para el instructor si necesita ayuda para desplazarse por esta diapositiva.

Indique que la página IAM con frecuencia se encuentra en memoria y contiene información eficiente densamente empaquetada.

Señale que, en ausencia de un índice, sólo un recorrido de tabla puede recuperar filas.

Cuando no hay índices en una tabla, SQL Server debe utilizar un recorrido de tabla para recuperar filas. SQL Server utiliza la tabla **sysindexes** para buscar la página IAM. Gracias a que la página IAM contiene una lista de todas las páginas relacionadas con esa tabla, en forma de un mapa de bits de extensiones de ocho páginas, SQL Server puede leer todas las páginas de datos.

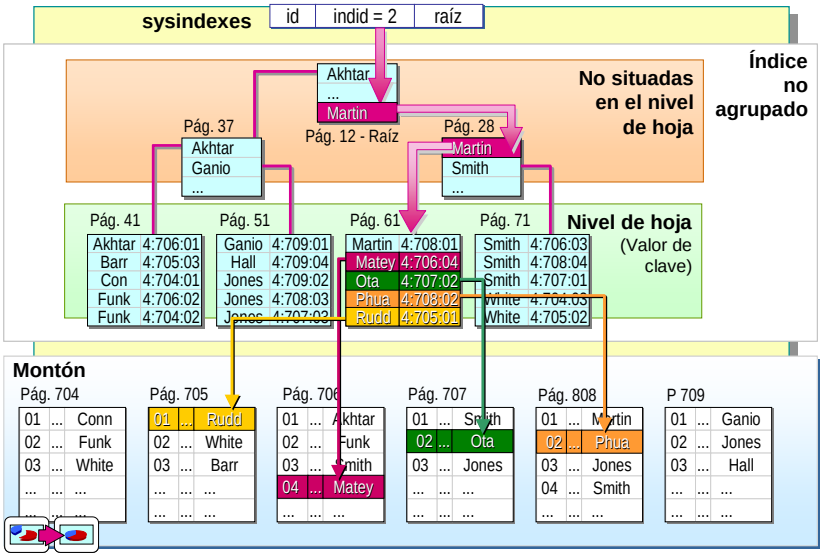
Iniciar una búsqueda de datos en un montón mediante una página IAM puede ser un buen método para un recorrido de tabla, pero no es conveniente para buscar un número pequeño de filas en una tabla grande.

Las filas se devolverán sin orden. Aunque inicialmente pueden devolverse en orden de inserción, este orden no se mantiene. Una vez producidas las eliminaciones, las nuevas inserciones llenarán los huecos, lo que hará que el orden sea impredecible.

# Búsqueda de filas en un montón con un índice no agrupado

**Objetivo del tema**
Describir la búsqueda de datos con un índice no agrupado.

**Explicación previa**
Los punteros son muy importantes para la búsqueda de índices no agrupados.




**Sugerencia**
Ésta es una diapositiva animada. Consulte las Notas para el instructor si necesita ayuda para desplazarse por esta diapositiva.

Los alumnos deben estar ya familiarizados con la estructura del árbol B. Subraye el uso y la estructura de los punteros.

Un índice no agrupado es como el índice de un libro de texto. Los datos se almacenan en un lugar y el índice en otro. Los punteros indican el lugar de almacenamiento de los elementos indizados en la tabla subyacente.

Los índices de SQL Server se organizan como árboles B. Cada página de un índice contiene un encabezado de página seguido de las filas de índice. Cada fila de índice contiene un valor de clave y un puntero a otra página o una fila de datos.

Cada página de un índice se llama nodo de índice. El nodo superior del árbol B se llama nodo raíz o nivel raíz. El nodo inferior se llama nodo de hoja o nivel de hoja. Cualquier nivel de índice entre los nodos raíz y de hoja son niveles intermedios. Cada página de los niveles intermedio o inferior tiene un puntero a las páginas precedentes y posteriores en una lista doblemente vinculada.

En una tabla que sólo contiene un índice no agrupado, las nodos de hoja contienen localizadores de filas con punteros a las filas de datos que contienen los valores de clave. Cada puntero (Id. de fila o RID) se crea a partir del Id. de archivo, el número de página y el número de la fila en la página.

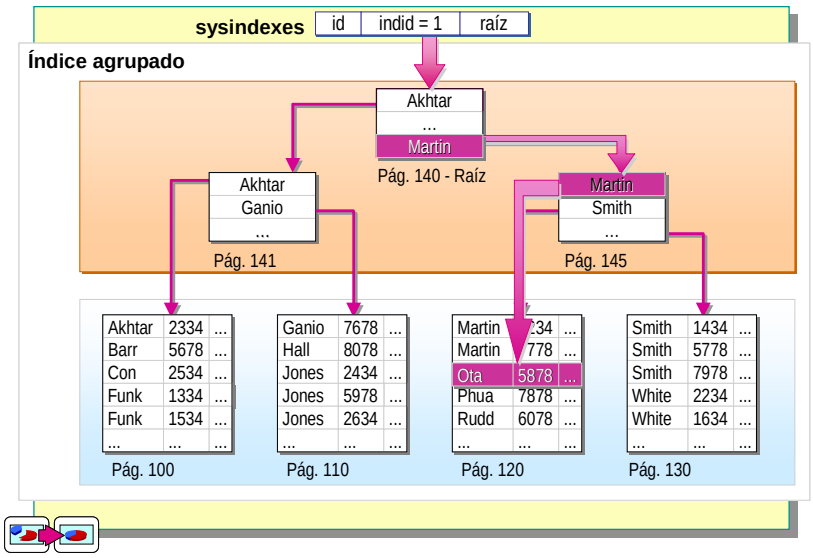
Ejemplo

```
SELECT lastname, firstname
FROM member
WHERE lastname
BETWEEN 'Masters' AND 'Rudd'
```

# Búsqueda de filas en un índice agrupado

**Objetivo del tema**
Describir la búsqueda de datos con un índice agrupado.

**Explicación previa**
Los índices agrupados están ordenados de forma secuencial en función de su clave agrupada.




**Sugerencia**
Ésta es una diapositiva animada. Consulte las Notas para el instructor si necesita ayuda para desplazarse por esta diapositiva.

Los índices agrupados y no agrupados comparten una estructura de árbol B similar. Las diferencias son las siguientes:

- Las páginas de datos de un índice agrupado son los nodos de hoja de la estructura de árbol B.
- Las filas de datos de un índice agrupado están ordenadas y almacenadas de forma secuencial en función de su clave agrupada.

Un índice agrupado es como un directorio de teléfonos en el que todas las filas de clientes con el mismo apellido se agrupan juntas en la misma parte de la libreta. Al igual que la organización de una guía telefónica facilita la búsqueda a una persona, SQL Server busca rápidamente una tabla con un índice agrupado. Debido a que un índice agrupado determina la secuencia de almacenamiento de las filas en una tabla, sólo puede haber un índice agrupado por cada tabla a la vez.

Mantener la clave agrupada en un valor pequeño aumenta el número de filas de índice que se pueden colocar en una página de índice y reduce el número de niveles que se tienen que recorrer. Esto reduce las operaciones de E/S.

**Nota** Si en un índice agrupado existen valores duplicados, SQL Server debe distinguir entre filas que contienen valores idénticos en la columna o columnas de la clave. Esto lo hace mediante el uso de un entero de 4 bytes (valor de *exclusividad*) en una columna de exclusividad adicional sólo del sistema.

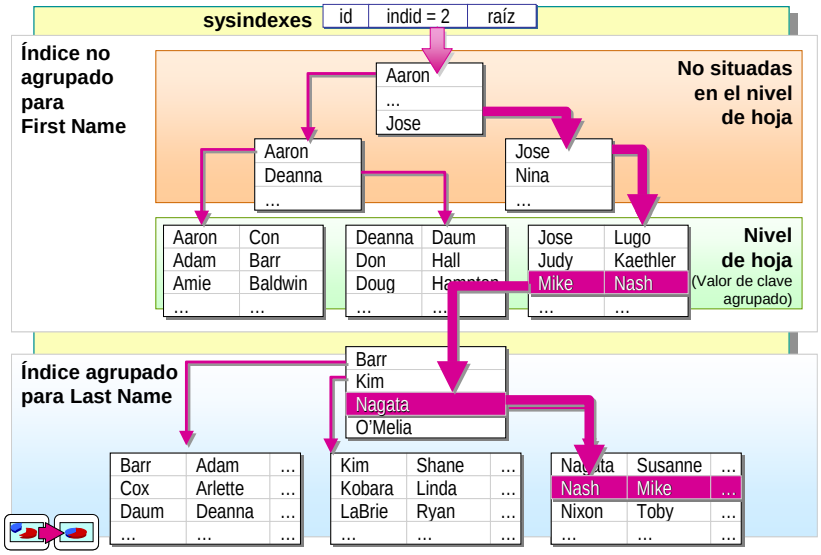
**Ejemplo**

```
SELECT lastname, firstname
FROM member
WHERE lastname = 'Ota'
```

# Búsqueda de filas en un índice agrupado con un índice no agrupado

**Objetivo del tema**
Describir la búsqueda de datos cuando ambos tipos de índices están presentes.

**Explicación previa**
El índice secundario se utiliza para acelerar las búsquedas en columnas adicionales.




---

**Sugerencia**
Ésta es una diapositiva animada. Consulte las Notas para el instructor si necesita ayuda para desplazarse por esta diapositiva.

Cuando un índice no agrupado se agrega a una tabla que ya tiene un índice agrupado, el localizador de filas de cada índice no agrupado contiene el valor de índice de la clave agrupada de la fila.

Al utilizar índices agrupados y no agrupados en la misma tabla, la estructura de árbol B de ambos índices tiene que recorrerse para encontrar los datos. Esto genera operaciones de E/S adicionales.

Debido a que el valor de la clave de un índice agrupado es, por lo general, mayor que el RID de 8 bytes utilizado en los montones, los índices no agrupados pueden tener un tamaño bastante mayor en tablas indizadas agrupadas que cuando se crean en montones. Mantener pequeños los valores de la clave del índice agrupado ayuda a crear índices menores y más rápidos.

Ejemplo

```
SELECT lastname, firstname,
phone_no
FROM member
WHERE firstname = 'Mike'
```

# Cómo SQL Server mantiene las estructuras de los índices y los montones

**Objetivo del tema**
Presentar cómo SQL Server mantiene las estructuras de índices y montones.

**Explicación previa**
Esta sección describe cómo SQL Server mantiene índices y montones

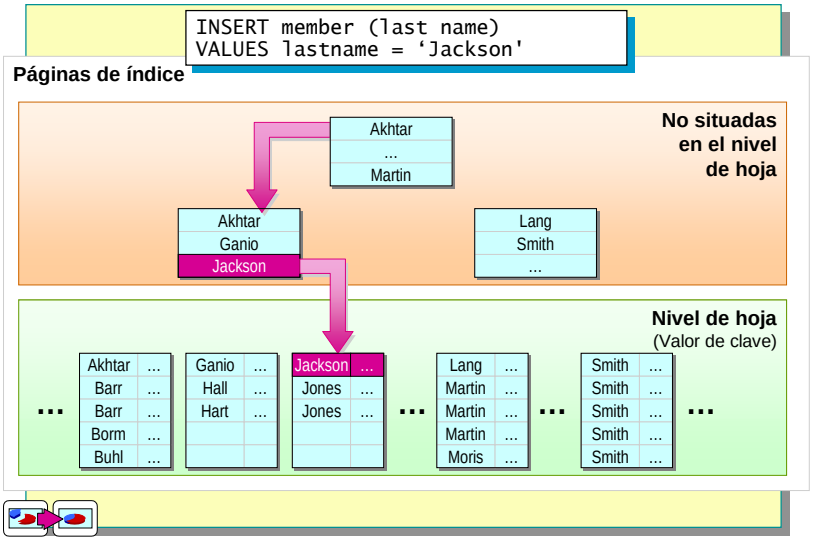
- Divisiones de páginas en un índice
- Puntero de reenvío en un montón
- Cómo SQL Server actualiza filas
- Cómo SQL Server elimina filas

Esta sección explica cómo SQL Server mantiene índices y montones al insertar, actualizar y eliminar filas.

# Divisiones de páginas en un índice

**Objetivo del tema**
Describir el concepto de una división de página.

**Explicación previa**
Insertar una fila en una página llena puede causar la división de la misma.




**Sugerencia**
Ésta es una diapositiva animada. Consulte las Notas para el instructor si necesita ayuda para desplazarse por esta diapositiva.

Un índice agrupado dirige una fila insertada o actualizada a una página específica, que viene determinada por el valor de la clave agrupada. Si la página de datos o la página de índice no dispone de espacio suficiente para acomodar los datos, se agrega una nueva página en un proceso conocido como *división de página*. La mitad de los datos aproximadamente permanecen en la página antigua y la otra mitad pasan a la nueva página.

Lógicamente, la nueva página sigue a la página original; físicamente, la nueva página puede asignarse a cualquier página disponible. Si un índice experimenta una gran cantidad de divisiones de página, al reconstruir el índice se mejorará el rendimiento.

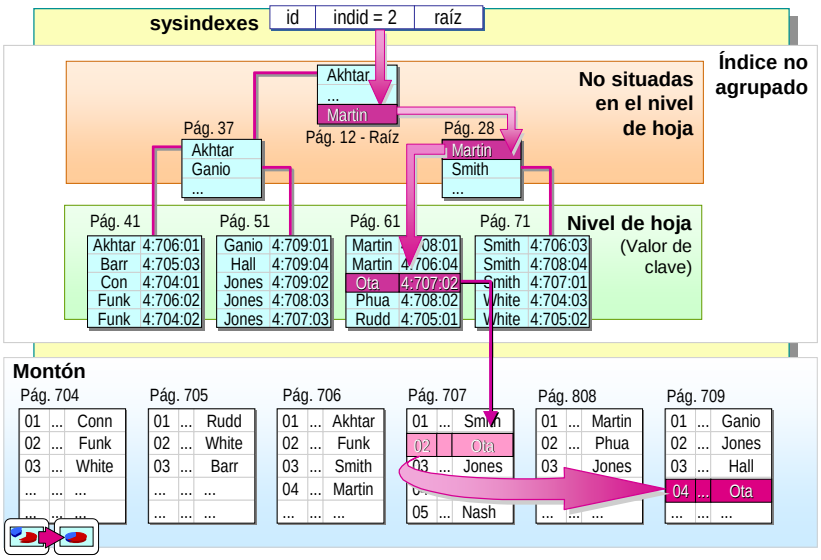
**Sugerencia**
Señale que el índice no agrupado debe modificarse para agregar Jackson, pero que no es necesario actualizarlo con la nueva ubicación de Jones.

**Nota** Si una página se divide en un índice agrupado, SQL Server no necesita mantener los índices no agrupados de todas las filas que se hayan movido a una nueva página. El localizador de filas continúa identificando el lugar correcto en la clave de agrupación.

# Puntero de reenvío en un montón

**Objetivo del tema**
Describir el concepto de un puntero de reenvío.

**Explicación previa**
Si una fila de un montón se hace demasiado grande para su ubicación original, se moverá a otra página.




**Sugerencia**
Ésta es una diapositiva animada. Consulte las Notas para el instructor si necesita ayuda para desplazarse por esta diapositiva.

Las divisiones de páginas no se producen en un montón. SQL Server dispone de un medio distinto de manejar actualizaciones e inserciones cuando las páginas de datos están llenas.

## Inserciones en un montón

La inserción de una fila en un montón no puede hacer que se divida una página, ya que es posible insertar una nueva fila siempre que haya espacio disponible.

## Punteros de reenvío

Si la actualización de una fila en un montón necesita más espacio del que hay actualmente disponible en esa página, la fila se moverá a una nueva página de datos. La nueva fila deja un *puntero de reenvío* en su ubicación original. Si la fila con el puntero de reenvío tiene que moverse otra vez, el puntero original se vuelve a dirigir a la nueva ubicación.

El puntero de reenvío asegura que no se tengan que cambiar los índices no agrupados. Si una actualización hace que la fila avanzada se reduzca lo suficiente como para que quepa en su ubicación original, el puntero se elimina y la actualización restaura el registro a su ubicación original

## Divisiones de páginas en índices no agrupados en un montón

Aunque una inserción o una actualización no puede producir la división de páginas en un montón, si éste contiene un índice sin agrupar, sí puede provocar la división de páginas en el índice sin agrupar.

# Cómo SQL Server actualiza filas

**Objetivo del tema**
Describir los efectos de actualizar datos.

**Explicación previa**
Las actualizaciones, por lo general, no afectan a la estructura de las filas de datos.

- Una actualización no suele hacer que una fila se mueva
- Una actualización puede ser una eliminación seguida de una inserción
- Las actualizaciones por lotes tocan cada índice una sola vez

Las actualizaciones, por lo general, no afectan a la estructura de las filas de datos.

## Una actualización no suele hacer que una fila se mueva

Las actualizaciones, por lo general, no precisan que las filas se muevan. Si la actualización no aumenta el registro o si cualquier aumento cabe en la misma página, no ocurre ningún movimiento. Las actualizaciones generan normalmente un único registro.

## Una actualización puede ser una eliminación seguida de una inserción

Una actualización que hace que una fila se mueva se registra como una eliminación seguida de una inserción en los siguientes casos:

- La actualización no cabe en una página de un montón.
- La tabla tiene un desencadenador de actualizaciones.
- La tabla está marcada para replicarse.
- El valor de la clave del índice agrupado precisa que la fila se coloque en un lugar distinto. Por ejemplo, un apellido que ha sido cambiado de Abercrombie a Yukish movería ese nombre en una guía telefónica.

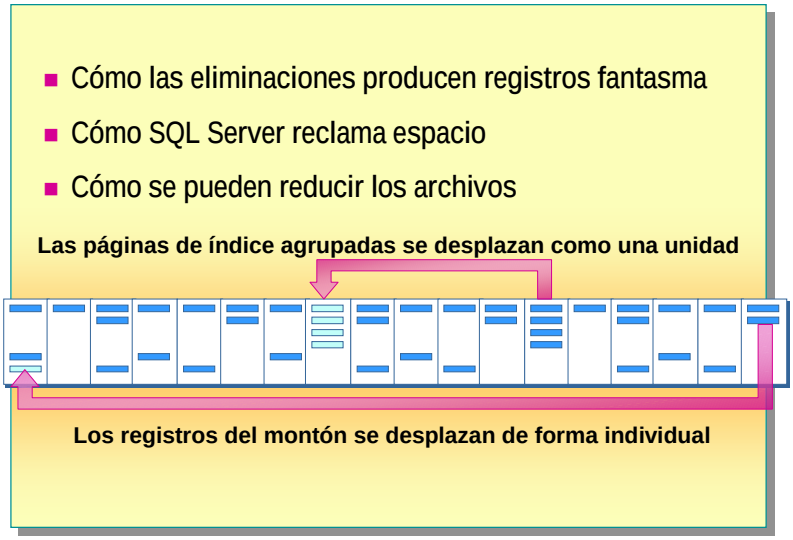
## Las actualizaciones por lotes tocan cada índice una sola vez

Si se inserta, actualiza o elimina un número significativo de filas de una tabla en una sola instrucción SQL, SQL Server ordena previamente los cambios de cada índice para que éstos se realicen en el orden del índice. Esta actualización por lotes da lugar a una mejora en el rendimiento notablemente superior que si se utiliza una serie de instrucciones Transact-SQL.

# Cómo SQL Server elimina filas

**Objetivo del tema**
Describir la eliminación de filas.

**Explicación previa**
Cuando una fila se elimina, tanto el índice como las páginas de datos cambian.




La eliminación de filas afecta tanto al índice como a las páginas de datos.

## Cómo las eliminaciones producen registros fantasma

Las filas eliminadas del nivel de hoja de un índice no desaparecen inmediatamente. Quedan marcadas como no válidas y se llaman *registros fantasma*. Este proceso puede evitar la necesidad de bloquear registros adyacentes. También puede prevenir los conflictos de bloqueo sobre intervalos de datos. SQL Server inicializa periódicamente un subproceso de almacenamiento especial que comprueba la existencia de registros fantasma en los índices y los elimina.

## Cómo SQL Server reclama espacio

Cuando se elimina la última fila de una página de datos, se anula la asignación de la página entera, salvo que sea la única página que queda en la tabla.

### Eliminación de filas en un índice

Inmediatamente después de que una fila es eliminada, las filas adyacentes pueden hacer uso del espacio disponible en un índice, si bien, generalmente, suelen quedar algunos huecos hasta que el índice se vuelve a crear.

### Eliminación de filas en un montón

Las filas eliminadas en un montón no se compactan hasta que se requiere espacio para una inserción.

## Cómo se pueden reducir los archivos

Sugerencia
Señale que las filas en un montón se mueven individualmente. Las filas no mantienen el orden de inserción.

Una vez eliminados los registros, los archivos se pueden reducir. SQL Server reduce los archivos al trasladar los datos a páginas disponibles al principio del archivo. Dentro de un índice, SQL Server mueve páginas enteras para que las filas se mantengan en el orden adecuado. Los punteros de página se ajustan para vincular la página movida dentro de la secuencia correcta en la tabla. Si no hay ningún índice agrupado, las filas individuales se pueden mover siempre que haya espacio en el archivo.

---

Nota La opción de base de datos **autoshrink** intenta reducir la base de datos sin intervención manual. Lo hace cada cinco minutos después del inicio y, posteriormente, cada treinta minutos. El archivo queda reducido a un tamaño donde el 25 por ciento del archivo es espacio sin utilizar o al tamaño de archivo que tenía cuando se creó, el que sea mayor.

---

# Decisión de las columnas que se van a indizar

**Objetivo del tema**
Presentar los temas de esta sección.

**Explicación previa**
El diseño de índices útiles es uno de los aspectos más importantes en la mejora del rendimiento de las consultas.

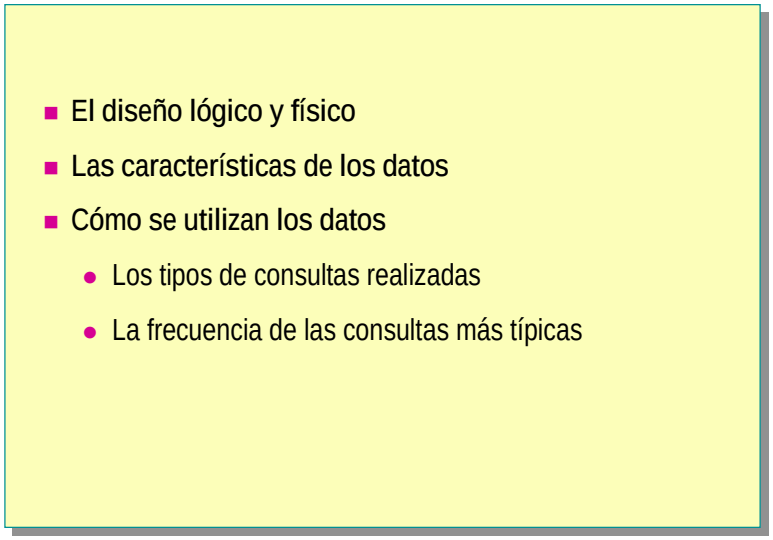
- Comprensión de los datos
- Directrices de indización
- Elección del índice agrupado adecuado
- Creación de índices que admiten consultas
- Determinación de la selectividad
- Determinación de la densidad
- Determinación de la distribución de datos

El diseño de índices útiles es uno de los aspectos más importantes en la mejora del rendimiento de las consultas. Requiere tanto una compresión de la estructura de los índices como del modo en que se utilizan los datos.

# Comprensión de los datos

**Objetivo del tema**
Señalar que el primer paso en la creación de índices es comprender los datos y la forma en que los usuarios tienen acceso a ellos.

**Explicación previa**
Antes de crear un índice, debe tener una profunda compresión de los datos.




Antes de crear un índice, debe tener una profunda compresión de los datos, incluido:

- El diseño lógico y físico.
- Las características de los datos.
- Cómo se utilizan los datos.

  Para diseñar índices útiles y efectivos, debe basarse en el análisis de las consultas que envían los usuarios. Respuestas lentas a consultas o incluso bloqueos innecesarios de tablas ponen de manifiesto un análisis deficiente de cómo los usuarios tienen acceso a los datos. Para conocer cómo los usuarios tienen acceso a los datos, debe observar lo siguiente:

  - Los tipos de consultas realizadas.
  - La frecuencia de las consultas más típicas.

Tener una profunda compresión de los requisitos de datos del usuario ayudará a determinar qué columnas indizar y qué tipos de índices crear. Puede que tenga que sacrificar algo de velocidad en una consulta para ganar mejor rendimiento en otra.

# Directrices de indización

Objetivo del tema
Considerar qué columnas se deben indizar.

Explicación previa
Al crear un índice, considere la naturaleza del entorno y cómo se distribuirán los datos.

- Columnas adecuadas para indizar
  - Claves principal y externa
  - En las que se buscan frecuentemente intervalos
  - A las que se tiene acceso de forma ordenada
  - Agrupadas juntas durante la agregación
- Columnas no adecuadas para indizar
  - Se incluyen con poca frecuencia en consultas
  - Contienen pocos valores únicos
  - Se definen con los tipos de datos text, ntext o image

El entorno de trabajo, las características de los datos y el uso que se haga de ellos determinarán las columnas que hay que especificar para crear un índice. La utilidad de un índice está directamente relacionada con el porcentaje de filas devueltas por una consulta. Los índices son más eficientes cuando el porcentaje de filas devueltas es bajo y la selección de filas es muy precisa.

Nota Al crear un índice basado en una columna, ésta se denomina columna de índice. El valor de una columna de índice se llama valor de clave.

Sugerencia
Pregunta: ¿Pueden indizarse todas las columnas?

Respuesta: Sí, pueden indizarse todas las columnas, pero hacerlo resultaría poco eficiente.

Pregunta: ¿Es posible indizar una única columna más de una vez?

Respuesta: Sí, pero en general hacerlo es poco eficiente.

## Columnas adecuadas para indizar

Cree índices basados en las columnas buscadas con frecuencia, por ejemplo:

- Claves principales.
- Claves externas o columnas utilizadas con frecuencia para combinar tablas.
- Columnas en las que se buscan intervalos o valores de clave.
- Columnas a las que se tiene acceso de forma ordenada.
- Columnas agrupadas juntas durante la agregación.

## Columnas no adecuadas para indizar

No cree índices basados en las columnas que:

- Se incluyen con poca frecuencia en una consulta.
- Contienen pocos valores únicos. Por ejemplo, un índice basado en una columna con dos valores, masculino y femenino, devuelve un alto porcentaje de filas.
- Se definen con los tipos de datos **text**, **ntext** e **image**. Las columnas con estos tipos de datos no se pueden indizar.

# Elección del índice agrupado adecuado

**Objetivo del tema**
Determinar el mejor tipo de índice para una base de datos.

**Explicación previa**
A la hora de seleccionar el índice agrupado, tenga en cuenta cómo se utiliza la tabla.

- Tablas continuamente actualizadas
  - Un índice agrupado con una columna de identidad mantiene las páginas actualizadas en memoria
- Ordenación
  - Un índice agrupado mantiene los datos preordenados
- Longitud de columna y tipo de datos
  - Limita el número de columnas
  - Reduce el número de caracteres
  - Utiliza los tipos de datos más pequeños posibles

**Sugerencia**
Indique a los alumnos que no deben colocar automáticamente el índice agrupado en la clave principal. Considere el uso de la tabla.

A la hora de seleccionar el índice agrupado para cada tabla, tenga en cuenta cómo se utiliza la tabla.

## Tablas continuamente actualizadas

Cuando optimice el rendimiento para la inserción de datos en una tabla muy utilizada, considere crear un índice agrupado en una columna de identidad de clave principal. Al forzar las inserciones en un grupo pequeño de páginas al final de la tabla, la velocidad aumenta. El acceso frecuente mantiene esas páginas en memoria.

## Ordenación

Las tablas que con frecuencia se ordenan para informes, se agrupan para agregaciones o en las que se buscan intervalos de datos pueden beneficiarse de un índice agrupado en la columna de ordenación. El uso de un índice agrupado resulta particularmente útil cuando se devuelven muchas columnas de la tabla y no es práctico utilizar un índice no agrupado. Por ejemplo, una tabla de lista de correo se beneficiaría de un índice agrupado en el código postal, ya que las etiquetas de correo deben imprimirse y aplicarse en un orden determinado.

## Longitud de columna y tipo de datos

SQL Server utiliza el valor del índice agrupado como identificador de filas dentro de cada índice no agrupado. El valor del índice agrupado puede repetirse muchas veces en la estructura de la tabla.

Para impedir que los índices agrupados de gran tamaño hagan mayores y más lentos sus índices no agrupados asociados:

- Limite el número de columnas en su índice agrupado.
- Reduzca el número medio de caracteres mediante el tipo de datos **varchar** en lugar de **char**.
- Utilice el tipo de datos menor posible; por ejemplo, **tinyint** en lugar de **int**.

# Creación de índices que admiten consultas

**Objetivo del tema**
Comprender por qué es importante limitar la búsqueda y cómo hacerlo.

**Explicación previa**
El rendimiento de las consultas dependerá de lo bien que haya diseñado sus índices y de lo selectivas que sean las consultas. Siempre debe escribir consultas que limiten una búsqueda.

- Uso de argumentos de búsqueda
- Escritura de buenos argumentos de búsqueda
  - Especificar una cláusula WHERE en la consulta
  - Comprobar que la cláusula WHERE limita el número de filas
  - Comprobar que existe una expresión para cada tabla a la que se hace referencia en la consulta
  - Evitar el uso de caracteres comodines iniciales

El rendimiento de las consultas depende de lo bien que haya diseñado sus índices. También es importante escribir las consultas con un argumento de búsqueda que pueda aprovechar una columna indizada.

## Uso de argumentos de búsqueda

Un argumento de búsqueda limita la búsqueda a una coincidencia exacta, un intervalo de valores o una combinación de dos o más elementos unidos por un operador AND. Un argumento de búsqueda contiene una expresión constante que actúa sobre una columna por medio de un operador. Al escribir consultas que contienen argumentos de búsqueda, el optimizador de consultas tiene mayores oportunidades de utilizar un índice.

## Escritura de buenos argumentos de búsqueda

Si una expresión no limita una búsqueda, se considera un argumento sin búsqueda. En muchos casos, deberá volver a escribir las consultas para convertir los argumentos sin búsqueda en argumentos de búsqueda.

Para limitar la búsqueda, debe hacer lo siguiente:

- Especificar una cláusula WHERE en la consulta.
- Comprobar que la cláusula WHERE limita el número de filas.
- Comprobar que existe una expresión para cada tabla a la que se hace referencia en la consulta.
- Evitar el uso de caracteres comodines iniciales.

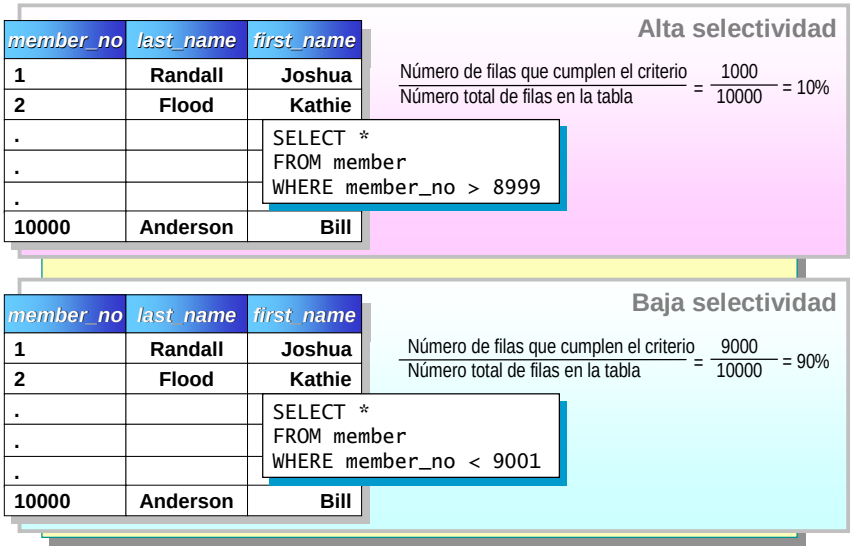
La siguiente tabla muestra algunos buenos argumentos de búsqueda:

| Buen argumento de búsqueda | Consulta |
|---|---|
| `WHERE cust_id = 47635` | Limita la búsqueda porque **cust_id** es único. |
| `WHERE date BETWEEN '07/23/2000' AND '07/30/2000'` | Limita la búsqueda a sólo un intervalo pequeño de datos. |
| `WHERE lastname LIKE 'Gre%'` | Limita la búsqueda únicamente a los apellidos que empiezan con las letras Gre. |

# Determinación de la selectividad

**Objetivo del tema**
Presentar el concepto de selectividad.

**Explicación previa**
La selectividad es un concepto y término frecuentemente utilizado para explicar los índices.




**Sugerencia**
Los ejemplos de la diapositiva no se pueden ejecutar en la base de datos credit.

La *selectividad* es un concepto y término frecuentemente utilizado para explicar los índices. A la hora de determinar las columnas que se deben indizar y seleccionar el tipo de índice que hay que crear, debe tener en cuenta el grado de selectividad que tengan los valores de datos:

## Definición de selectividad

La selectividad se deriva del porcentaje de filas de una tabla a las que tiene acceso o devuelve una consulta. El optimizador de consultas determina la selectividad de las instrucciones SELECT, UPDATE o DELETE. Cuando cree índices, debe crearlos en:

- Columnas a las que se haga referencia con frecuencia en operaciones de combinación o en la cláusula WHERE.
- Datos que sean muy selectivos.

### Alta selectividad y baja selectividad

En *alta selectividad*, los criterios de búsqueda limitan el número de filas devueltas a un bajo porcentaje del total posible. Una sola fila devuelta es la selectividad más alta que se puede conseguir.

En *baja selectividad*, los criterios de búsqueda devuelven un alto porcentaje de las filas de la tabla.

## Estimación de la selectividad

Puede determinar lo selectiva que es una consulta si estima el número de filas devueltas, con relación al número total de filas en una tabla para una consulta específica.

Ejemplo 1

En este ejemplo, suponiendo que hay 10.000 filas en la tabla **member** y que los números de los miembros están en el intervalo de 1 a 10.000 (todos valores únicos), la consulta devuelve una fila.

```
SELECT *
FROM member
WHERE member_no = 8999
```

Ejemplo 2

En este ejemplo, suponiendo que hay 10.000 filas en la tabla **member** y que los números de los miembros están en el intervalo de 1 a 10.000 (todos valores únicos), la consulta devuelve 999 filas.

```
SELECT *
FROM member
WHERE member_no > 9001
```

Ejemplo 3

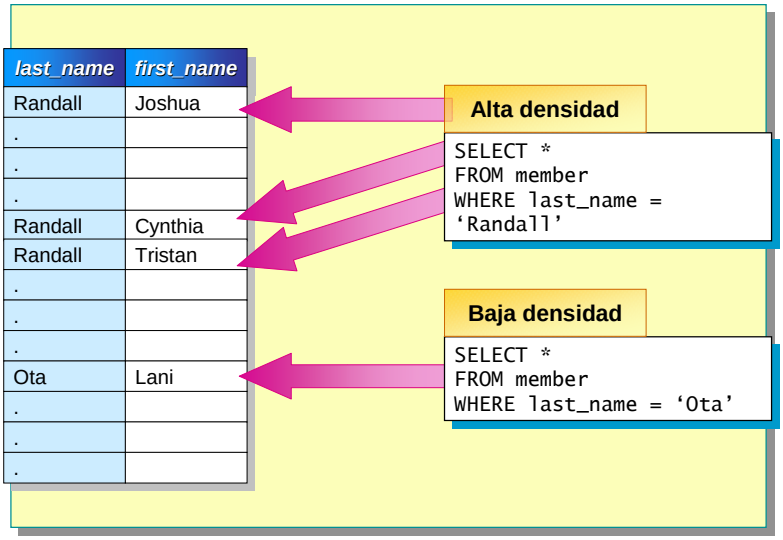
En este ejemplo, suponiendo que hay 10.000 filas en la tabla **member** y que los números de los miembros están en el intervalo de 1 a 10.000 (todos valores únicos), la consulta devuelve 9.000 filas.

```
SELECT *
FROM member
WHERE member_no < 9001
```

# Determinación de la densidad

**Objetivo del tema**
Presentar el concepto de densidad.

**Explicación previa**
La densidad es un concepto relacionado con la selectividad. Al determinar las columnas que se van a indizar, debe examinar la densidad de los datos.




**Sugerencia**
Los ejemplos de la diapositiva no se pueden ejecutar en la base de datos **credit**.

La *densidad* es un concepto relacionado con la selectividad. Al determinar las columnas que se van a indizar, debe examinar la densidad de los datos.

## Definición de densidad

La densidad es el porcentaje promedio de las filas duplicadas en un índice. Si los datos o las consultas no son muy selectivas (baja selectividad), se tiene un alto porcentaje de densidad.

- Un índice con un gran número de duplicados tiene alta densidad.

  Por ejemplo, un índice basado en la columna **lastname** puede ser muy denso.

- Un índice único tiene baja densidad.

  Un ejemplo de esto sería un índice basado en el número de la seguridad social, Id., número de permiso de conducir o apellido y nombre (compuesto).

## Relación de la densidad con los datos

Cuando determine la densidad de los datos, recuerde que se relaciona con los elementos de datos específicos. La densidad puede ser variable. Considere un índice basado en la columna **lastname**. Los elementos de datos de este índice son muy densos en los apellidos comunes como Randall, mientras que es probable que el apellido Ota no sea muy denso.

## Cómo la densidad afecta al plan de consulta

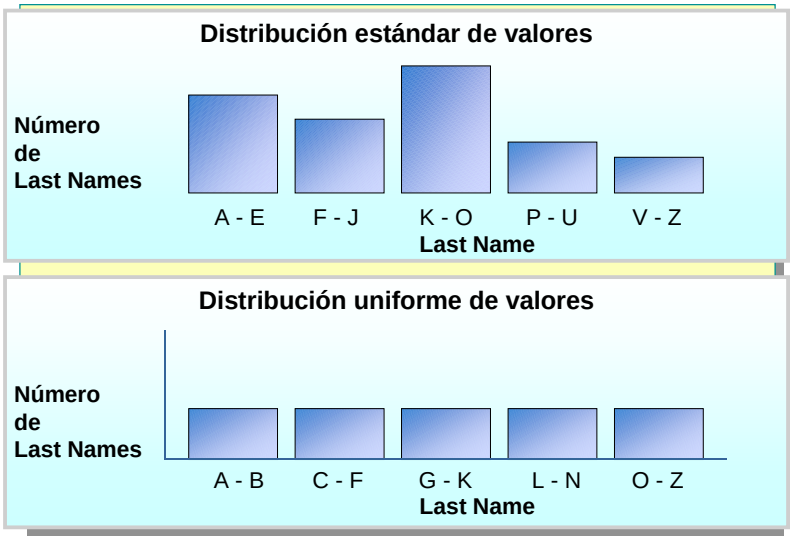
Debido a que los datos no se distribuyen uniformemente, el optimizador de consultas podría utilizar un índice o no utilizarlo. En la ilustración, el optimizador de consultas podría hacer lo siguiente:

- Realizar un recorrido de tabla para recuperar el apellido Randall.
- Utilizar un índice para tener acceso al apellido Ota.

# Determinación de la distribución de datos

**Objetivo del tema**
Explicar el concepto de distribución de datos y las diferentes formas en que se puede examinar la distribución de datos.

**Explicación previa**
La distribución de datos está relacionada con el concepto de densidad.
Al determinar la densidad de los datos, debe examinar también el modo en que se distribuyen.




La distribución de datos está relacionada con el concepto de densidad. Al determinar la densidad de los datos, debe examinar también el modo en que se distribuyen.

## Definición de distribución de datos

La *distribución de datos* indica la cantidad de datos que existen en un intervalo de valores de una tabla dada y el número de filas que se encuentran dentro de ese intervalo. Si una columna indizada tiene muy pocos valores únicos, la recuperación de datos puede ser lenta debido a su distribución. Por ejemplo, un directorio de teléfonos ordenado alfabéticamente por apellido puede mostrar que hay muchos casos de personas con el apellido Randall o Jones.

## Distribución estándar o uniforme

En una distribución estándar, los intervalos de valores de la clave permanecen constantes mientras que el número por intervalo cambia. Una distribución uniforme permite al optimizador de consultas determinar fácilmente la selectividad de una consulta al estimar el número de filas habilitadas como un porcentaje del total de filas de la tabla.

## Relación de la densidad con la distribución de datos

De forma similar a lo que ocurre con la densidad, los elementos de datos de un índice pueden variar en la forma en que se distribuyen los datos. Normalmente, los datos no se distribuyen uniformemente. Por ejemplo, si la tabla **member** contiene 10.000 filas y tiene un índice basado en la columna **lastname**, los apellidos no se suelen distribuir uniformemente.

## Estimación del porcentaje de filas devueltas

En muchos casos, puede estimar el porcentaje de datos que se devolverán en un conjunto de resultados. Por ejemplo, si el criterio es masculino/femenino, el conjunto de resultados para mujeres se puede estimar en un 50 por ciento. Al estimar el porcentaje de filas devueltas en valores como apellido, cuidad u otros datos demográficos, es esencial conocer los datos, puesto que la distribución de datos varía ampliamente en diferentes entornos.

Ejemplo

Esta consulta se utiliza para mostrar la distribución (cantidad de duplicados) de los valores de columna en una base de datos existente. En este ejemplo, la consulta devuelve cada valor una sola vez con un número (cuenta) que indica cuántas veces aparece en la tabla.

Sugerencia
Los ejemplos de la diapositiva no se pueden ejecutar en la base de datos credit.

```
SELECT column, count(*) AS 'Data Count'
FROM table
GROUP BY column
ORDER BY 'Data count' DESC
```