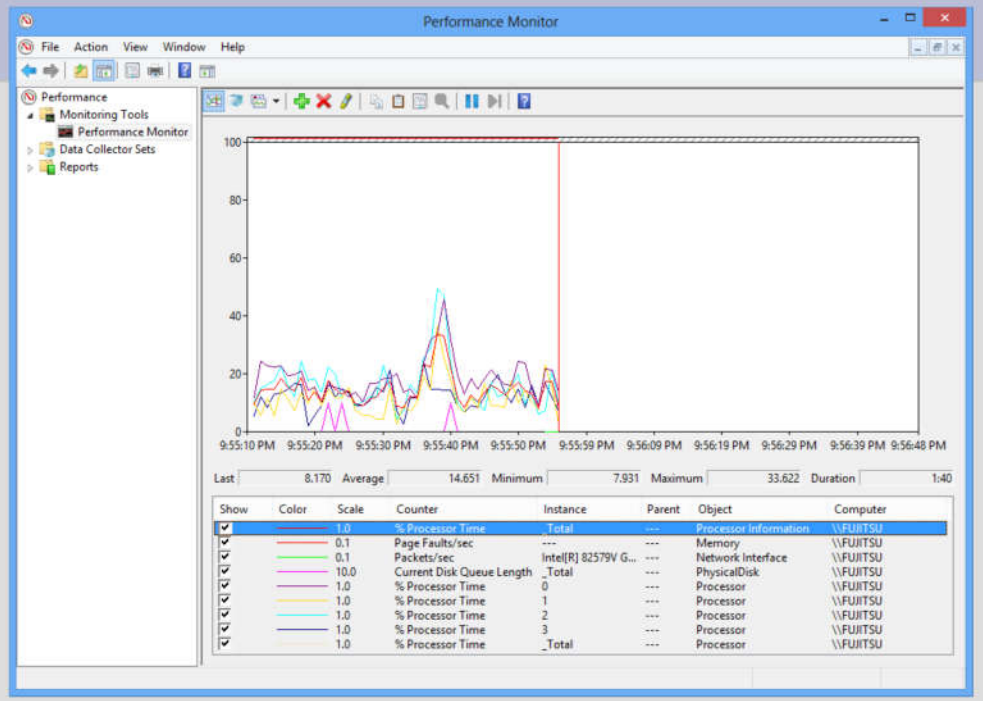
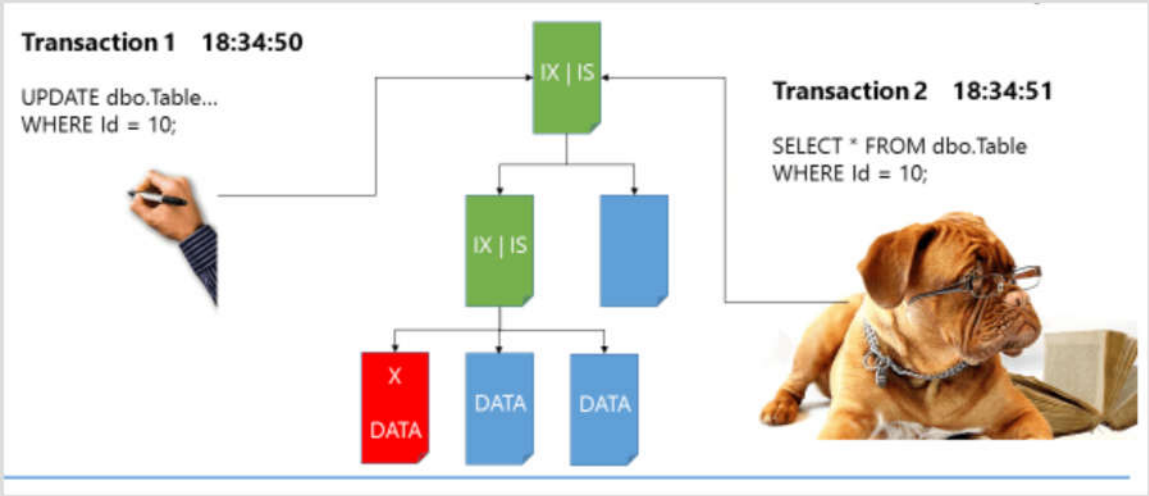
SQL SERVER TUNNING

Dead Lock

Isolation Level





Comprender **que son los bloqueos** y para que sirven

Introducir mecanismos de **monitorización de bloqueos**

Introducir técnicas para **combatir a los bloqueos**

- ¿Qué son los bloqueos?
- Niveles de aislamiento
- Tipos de bloqueos
- Combatiendo los bloqueos
- Conclusiones
- Preguntas

¿Qué son los Bloqueos?

- Modelos de Aislamiento
  - **Bloqueos (pesimista)**
  - Versionado de filas (optimista)
- Aseguran el **Aislamiento**
  - Leyendo datos consistentes
  - Si no son consistentes **esperas**

A	• Atomicidad
C	• Consistencia
I	• Aislamiento
D	• Durabilidad

Niveles de Aislamiento – Bloqueos

Tipos de Bloqueos básicos y compatibilidades

Shared Lock (S)

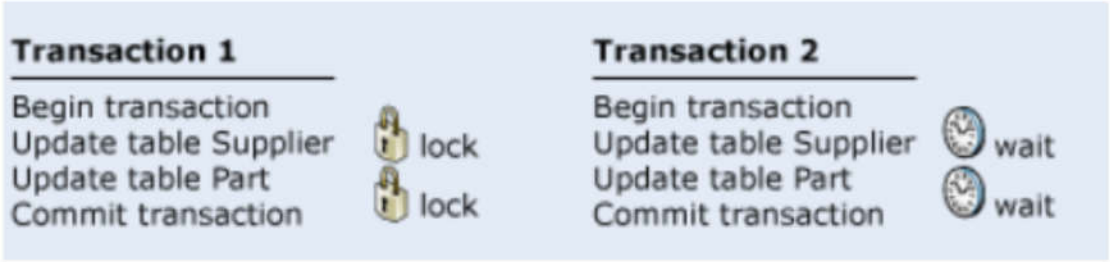
- Lecturas de datos

Exclusive Lock (X)

- Modificaciones de datos
- INSERT / DELETE / UPDATE

Compatibilidades:

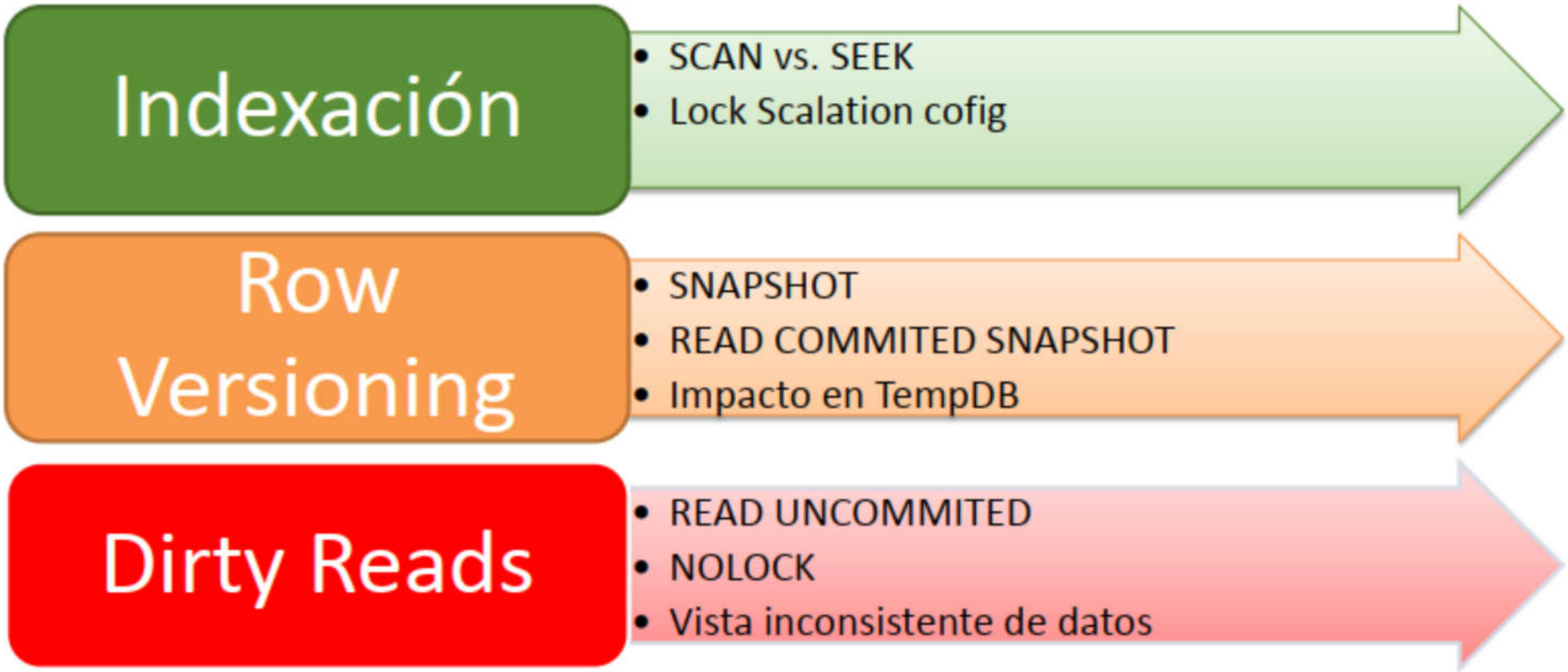
	Shared (S)	Exclusive (X)
Shared (S)	N	C
Exclusive (X)	C	C



C: CONFLICT
N: NO CONFLICT

DEMO

Combatiendo los bloqueos



Combatiendo los bloqueos - Indexación

Lo mas común: Key/RID Locks

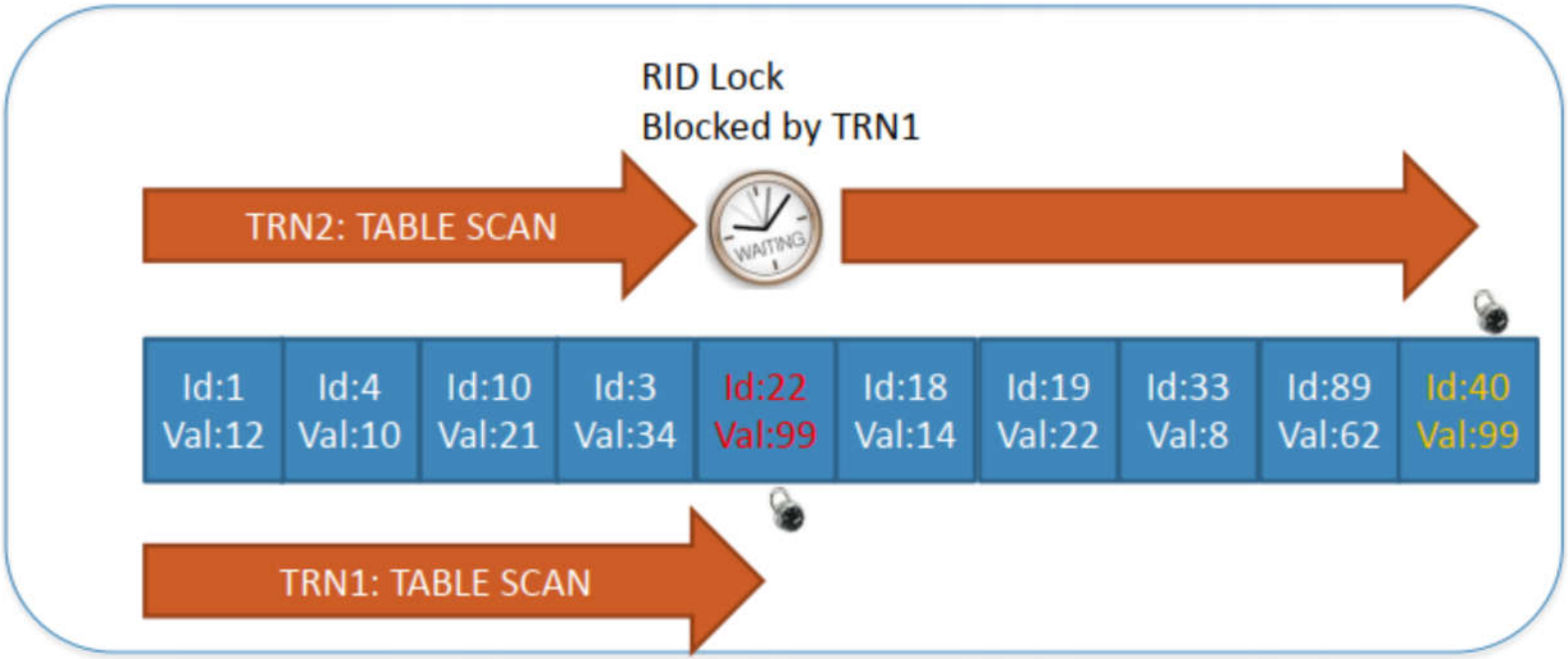
Tipo Tabla:HEAP

TRN 1:

```
update dbo.tab
set val=99
where id=22;
```

TRN 2:

```
update dbo.tab
set val=99
where id=40;
```



Combatiendo los bloqueos - Indexación

Lo mas común: Key/RID Locks

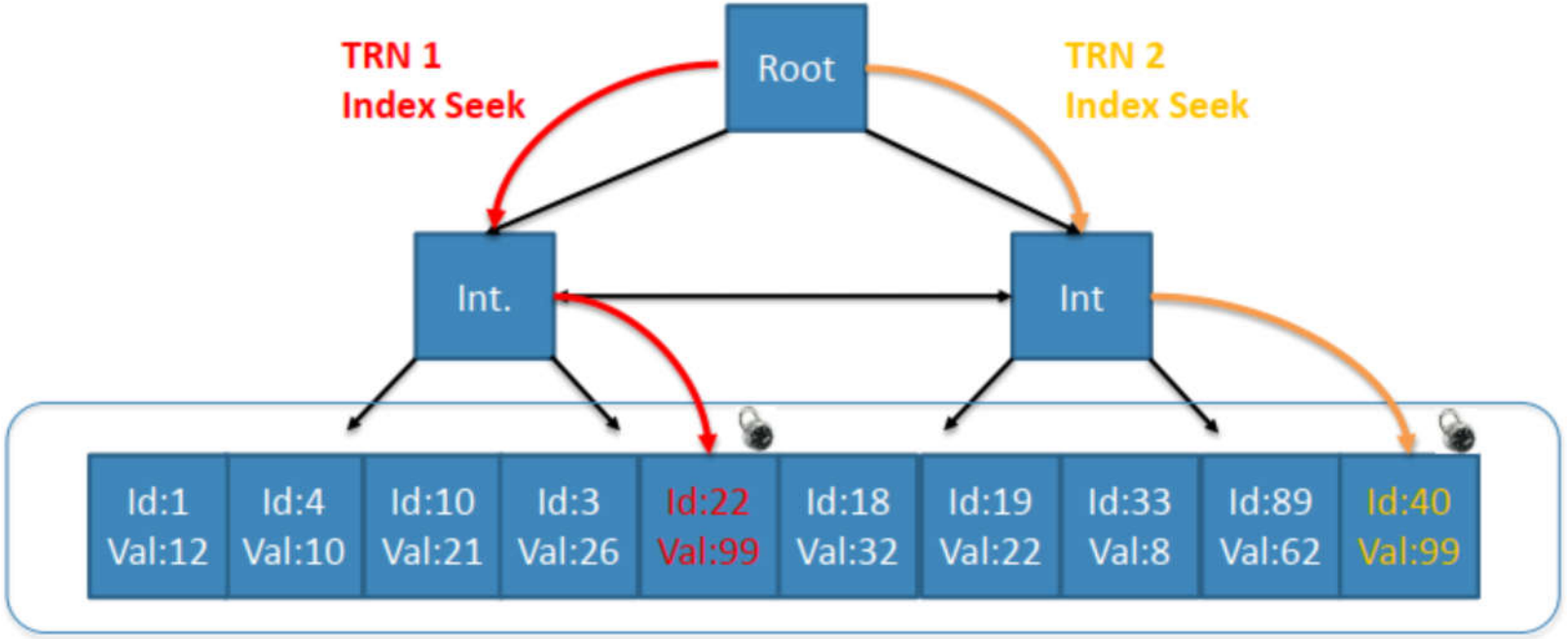
Tipo Tabla: CLUSTERED INDEX

TRN 1:

```
update dbo.tab
set val=99
where id=22;
```

TRN 2:

```
update dbo.tab
set val=99
where id=40;
```



Sin Transacciones bloqueadas!

Table Scan

Table Scan
[Ciudadano]
Cost: 100 %

Table Scan
[Ciudadano]
Cost: 97 %

- Si buscamos un tema en un libro sin usar el índice, sin un orden apropiado, tendríamos que ir hoja por hoja del libro hasta encontrar el tema necesitado, de la misma manera un Table Scan indica que el motor necesita leer completamente la tabla sin utilizar un índice porque no existe no le es funcional, lo cual la mayor parte del tiempo es deficiente.

Clustered Index Scan

Clustered Index Scan (Clustered)
[PK_INDIVIDUO]
Cost: 100 %

- Similar al Table Scan pero en este caso la tabla cuenta con un índice clustereado que pre-ordena los datos, entonces esta vez recorre los datos ordenados, de cualquier forma es lento que recorra todo el índice

Clustered Index Seek

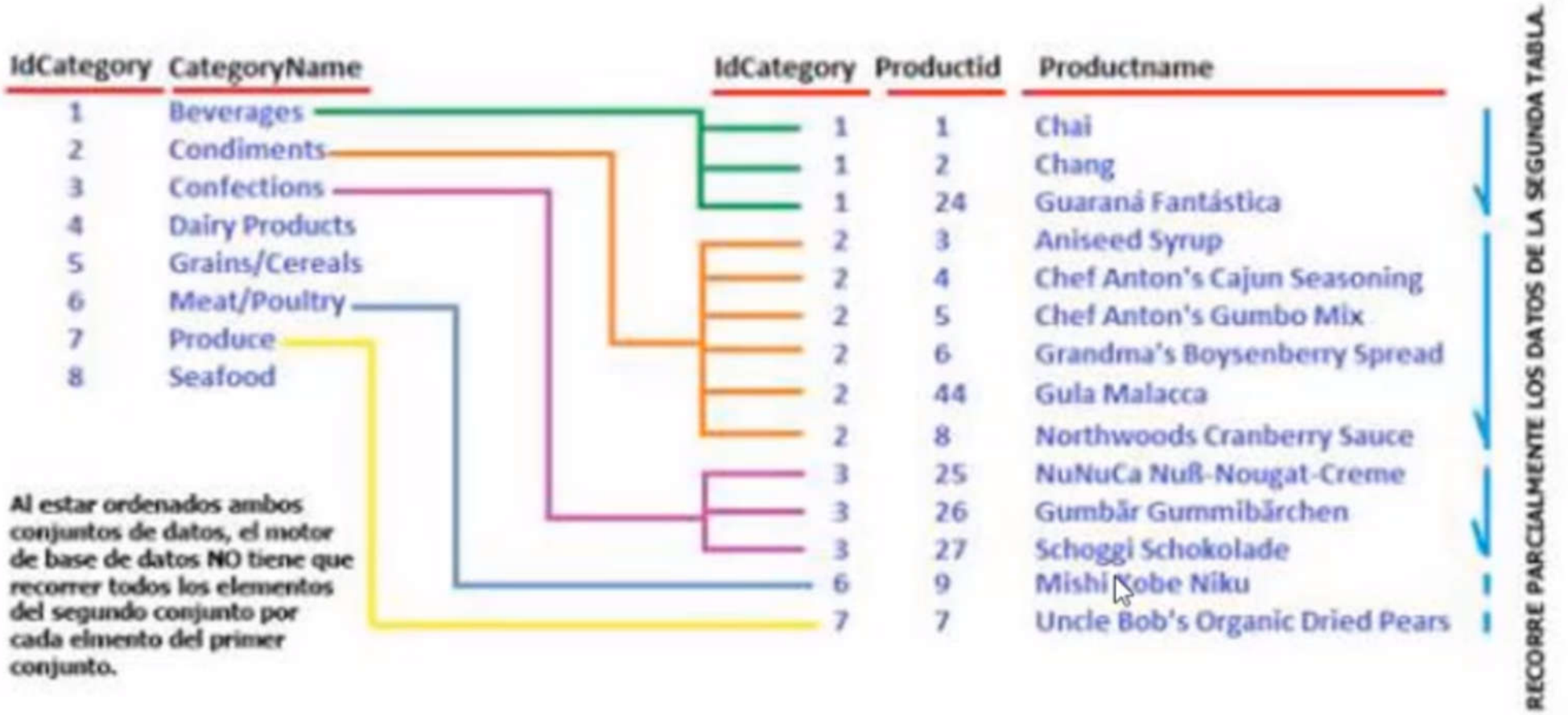
- Esto si es eficiente, este tipo de acción es como buscar en el diccionario que sabemos esta ordenado alfabéticamente si buscamos a la palabra “Zapato” por ejemplo, no vamos a empezar a buscar desde la primera página que empieza por la letra “A” , vamos a buscar casi el final a partir de donde se encuentra los de la letra “Z”

Clustered Index Seek (Clustered) [PK_Customers] Cost: 100 %

Nested Loop Join

Como los elementos de la segunta tabla no estan ordenados, es necesario recorrer cada elemento de la segunda tabla para comprobar que no se quede fuera ningun elemento correspodiente.

Merge Join

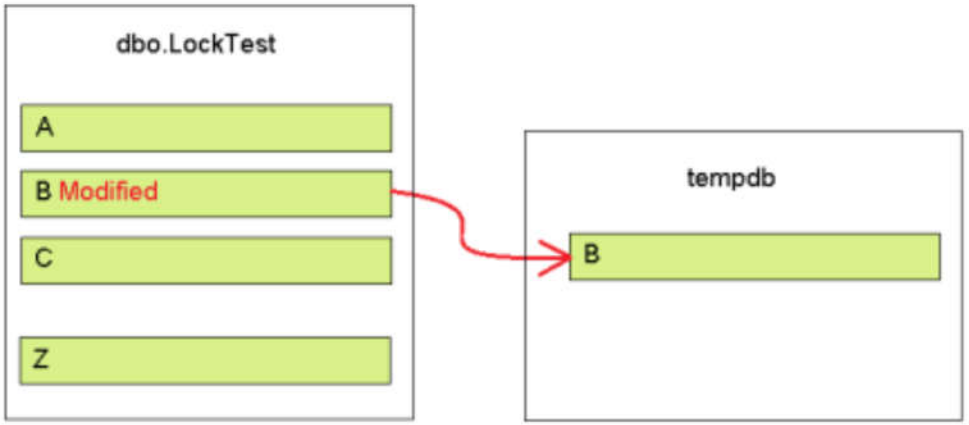

IdCategory
CategoryName
1 Beverages
2 Condiments
3 Confections
4 Dairy Products
5 Grains/Cereals
6 Meat/Poultry
7 Produce
8 Seafood
IdCategory
Productid
Productname
1 1 Chai
1 2 Chang
1 24 Guaraná Fantástica
2 3 Aniseed Syrup
2 4 Chef Anton's Cajun Seasoning
2 5 Chef Anton's Gumbo Mix
2 6 Grandma's Boysenberry Spread
2 44 Gula Malacca
2 8 Northwoods Cranberry Sauce
3 25 NuNuCa Nuß-Nougat-Creme
3 26 Gumbär Gummibärchen
3 27 Schoggi Schokolade
6 9 Mishi Kobe Niku
7 7 Uncle Bob's Organic Dried Pears
Al estar ordenados ambos conjuntos de datos, el motor de base de datos NO tiene que recorrer todos los elementos del segundo conjunto por cada elmento del primer conjunto.
RECORRE PARCIALMENTE LOS DATOS DE LA SEGUNDA TABLA

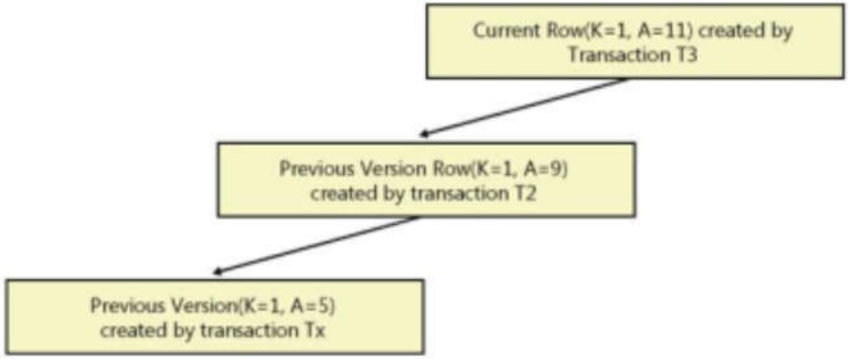
Combatiendo los bloqueos – Versionado de filas

SNAPSHOT / READ COMMITED NAPSHOT

IMPLICACIONES

- Impacto en TempDB (row versions)
- Configuraciones adicionales sobre las BBDDs
- Cambio estructura de las páginas





Conclusiones

- Bloqueos pueden causar deterioro en el rendimiento de nuestras Apps
  - Necesarios para mantener la coherencia de los datos (modelo pesimista)
- No son problemas físicos, más bien lógicos
  - Más hardware no lo solucionará
- Para solucionarlo:
  - Revisa estrategias de indexación
  - Considera utilizar SNAPSHOT | READ COMMITED SNAPSHOT
    - Dimensiona acorde (TempDB)
- Evita realizar lecturas sucias (Dirty Reads)
  - NOLOCK | READ UNCOMMITED
  - Pueden causar problemas y mostrar datos erróneos

DEMO

A continuación …

Optimización de motores SQL Server desde el código hasta la administración