Indexarea în SQL Server

S5

Indecşi

- Un index este o structură associată unei tabele sau *view* care optimizează timpul de acces la înregistrările tabelei sau *view*-ului.

- Indexare corectă → viteză & flexibilitate
- Indexare necorespunzătoare → încetineşte întregul SGBD

```
CREATE [ UNIQUE ] [ CLUSTERED | NONCLUSTERED ]
       INDEX index_name

ON <object> ( column [ ASC | DESC ] [ ,...n ] )

[ INCLUDE ( column_name [ ,...n ] ) ]

[ WHERE <filter_predicate> ]

[ WITH ( <relational_index_option> [ ,...n ] ) ]
```

Caracteristicile Indecşilor

- *Clustered* vs. *non-clustered*
- Unic vs. non-unic
- Simplu vs. compus (multicoloană)
- Ordonarea crescătoare sau descrescătoare
- Conținut dens vs filtrat (variantă de index *rar*)

Index *Clustered* vs *Non-Clustered*

- Index *clustered* : memorează şi sortează înregistrările din tabelă după valorile cheii de căutare

```
CREATE CLUSTERED INDEX Index_Name
                    ON Schema.TableName(Column);
```

- Index *non-clustered*: memorează valorile cheii de căutare şi un pointer la datele din tabelă

```
CREATE INDEX Index_Name
                    ON Schema.TableName(Column);
```

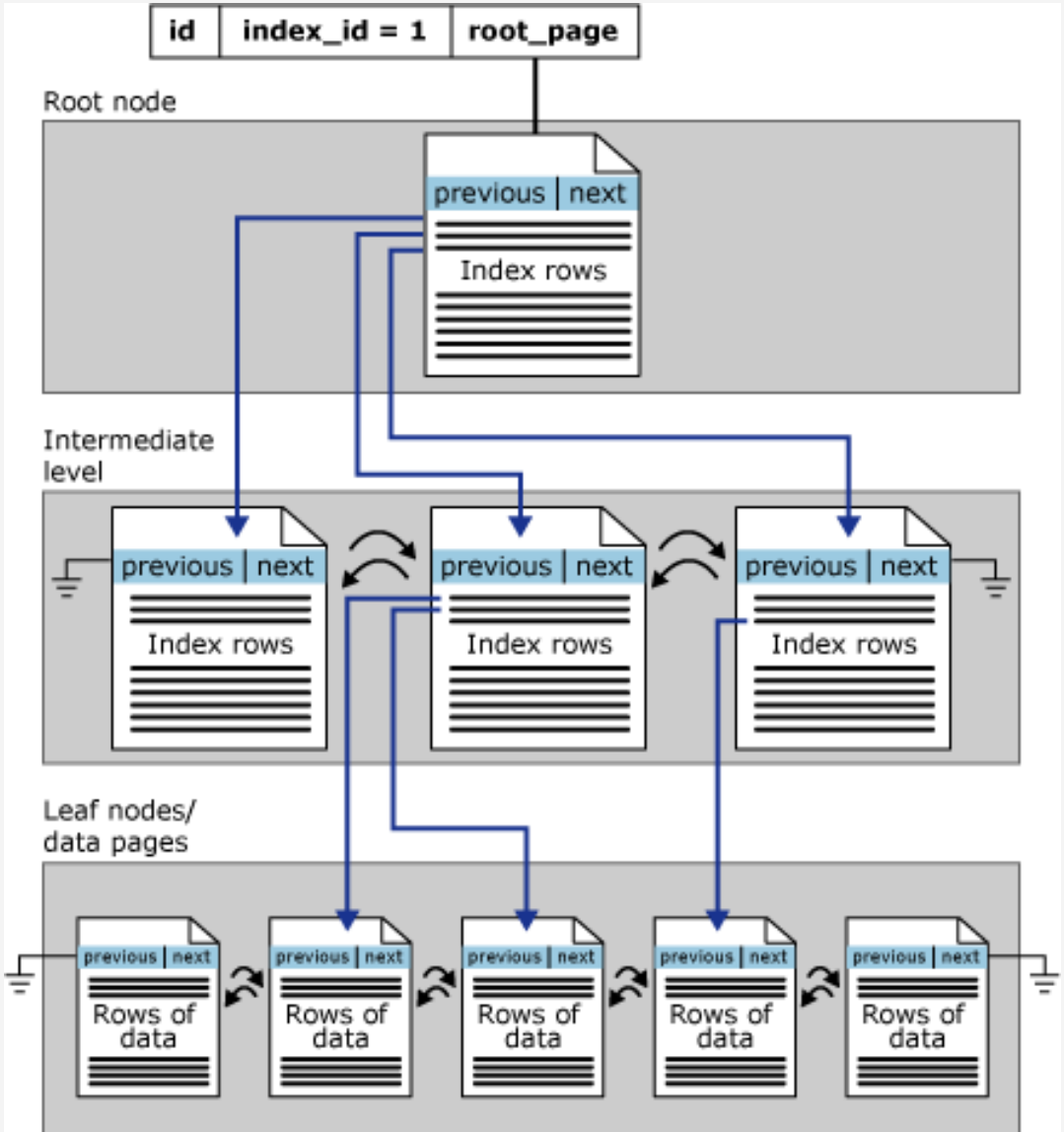
Index *Clustered* vs *Non-Clustered*

- Paginile de date ale unui index *clustered* va include *toate coloanele* tabelului

- Există un singur index *clustered* per tabelă.
- SQL Server suportă până la 999 indecşi *non-clustered* per tabelă.
- Cheia unui index – *clustered* sau *non-clustered* – poate avea maximum 16 câmpuri şi 900 bytes.

Index *clustered*

- indecşii sunt organizați sub formă de arbori-B.



Index *clustered* vs. *non-clustered*

- Atunci când se defineşte cheia primară a unei tabele
  - + dacă nu este definit un index *clustered*
  - + dacă nu e speecificat un index *non-clustered*

  → este creat un index *clustered* unic pe câmpurile cheii primare

- Dacă toate câmpurile returnate într-o interogare se află în index: *covering index*

Coloane cheie vs non-cheie în index

- Coloane *cheie*: câmpuri specificate la crearea indexului.
- Coloane *non-cheie*: câmpurile adăugate la clauza INCLUDE

  a unui index *non-clustered*

```
CREATE INDEX Index_Name
ON Schema.TableName(Column)
INCLUDE (ColumnA, ColumnB);
```

Coloane cheie vs non-cheie în index

- Avantajele utilizării câmpurilor non-cheie
  - Câmpurile pot fi accesate prin scanare index-ului
  - Tipurile de date care nu sunt permise în coloanele cheie sunt permise în coloanele non-cheie (ex. text, ntext, image).
  - Coloanele non-cheie nu sunt luate în considerare la calculul limitei de 900 bytes.

Proiectarea indecşilor

- Determinarea tipului de bază de date:
  - OLTP - On-line Transaction Processing
  - OLAP - On-line Analytical Processing
- Înțelegerea caracteristicilor celor mai fecvent utilizate interogări
- Înțelegerea caracteristicilor coloanelor utilizate în interogări
- Determinare modului de stocare optim al indexului

Reguli generale de indexare

- La nivelul bazei de date
  - prea mulți indecşi definiți pentru o tabelă afectează performanța comenzilor INSERT, UPDATE, DELETE, MERGE
  - Indexarea tabelelor mici este deseori inutilă
  - Indexare *view*-irilor are sens când acestea conțin agregări sau *join*-uri între tabele

Reguli generale de indexare

- Interogări
  - Creați indecşi *non-clustered* indexes pentru coloanele frecvent utilizate în clauzele WHERE şi JOIN
  - Un *covering index* poate îmbunătății semnificativ performanța unei interogări
  - Inserați sau modificați cât mai multe înregistrări posibile într-o singură comandă

Reguli generale de indexare

- Coloane
  - Păstrați lungimea cheii de căutare cât mai redusă pentru indecşii *clustered*
  - Indecşii *clustered* funcționează f bine pe coloane unice şi nenule
  - Coloanele de tip **ntext**, **text**, **image**, **varchar(max)**, **nvarchar(max)**, **varbinary(max)** nu pot fi utilizate în cheile de căutare
  - Examinați unicitatea coloanelor
  - Examinați distribuția datelor în coloane (nu indexați coloane cu puține valori unice) – utilizați indexare filtrată
  - Examinați ordinea coloanelor într-un index multiplu. Coloanele ce apar în egalități (=) sau inegalități (>, <, BETWEEN) ar trebui plasate primele.
  - Coloanele adiționale trebuie ordonate de la cele mai distincte la cele mai puțin distincte

Indecşi unici

- Un index unic garantează faptul că cheia de căutare nu conține valori duplicate
- Specificare unui index unic are sens doar dacă coloanele ce fac parte din cheie sunt unice :)
- Unicitatea reprezintă o informație utilă pentru optimizatorul de interogări

Indecşi filtraţi

Index filtrat: un index *non-clustered* optimizat, potrivit pentru acoperirea interogărilor ce selectează date dintr-un subset bine definit

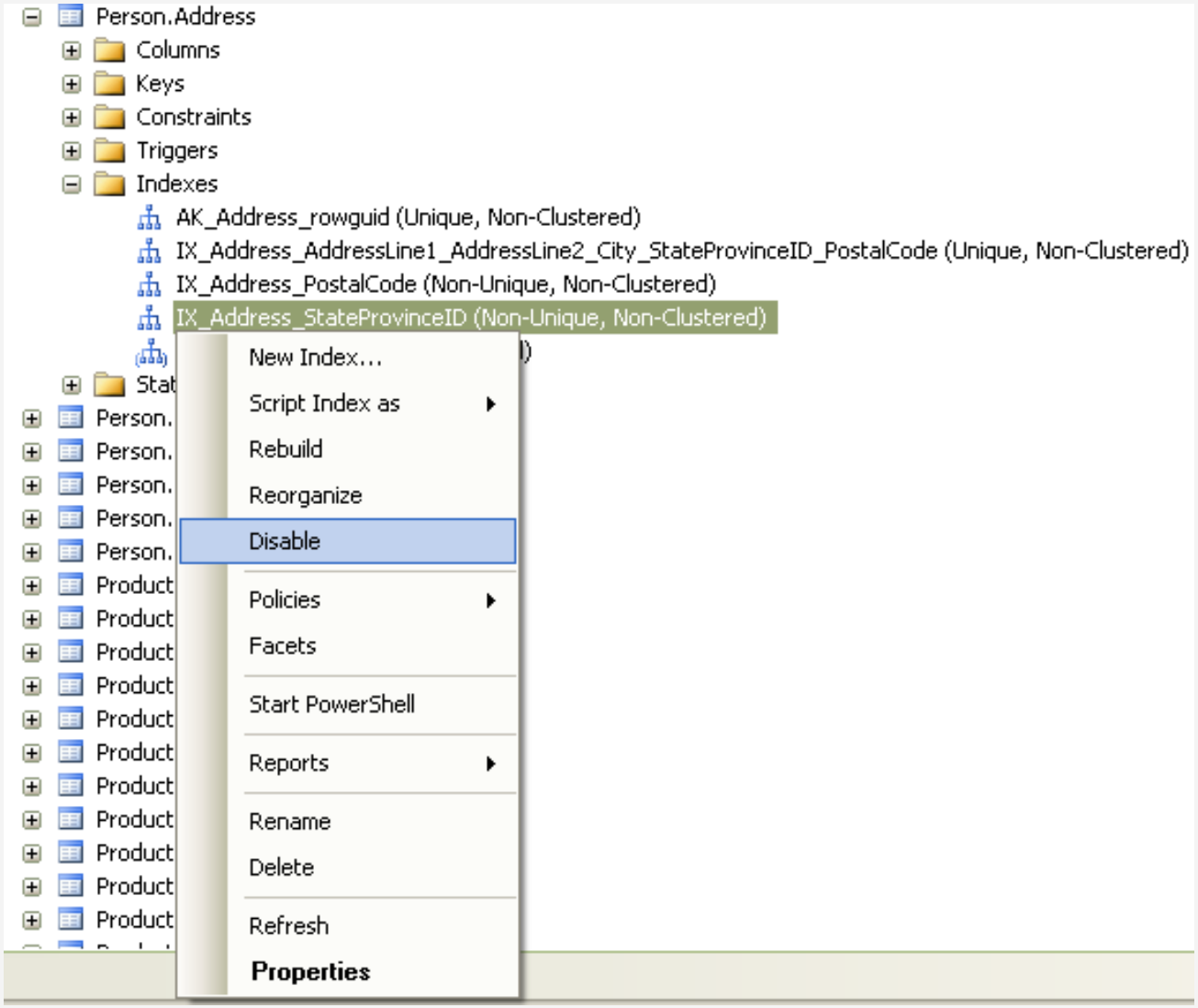
```
CREATE NONCLUSTERED INDEX FI_EndDate ON
Products (ProductID,  EndDate)
 WHERE EndDate IS NOT NULL ;
GO
```

- Creste performanta interogarii
- Reduce costurile de intretinere a indexului
- Reduce costurile de stocare a indexului

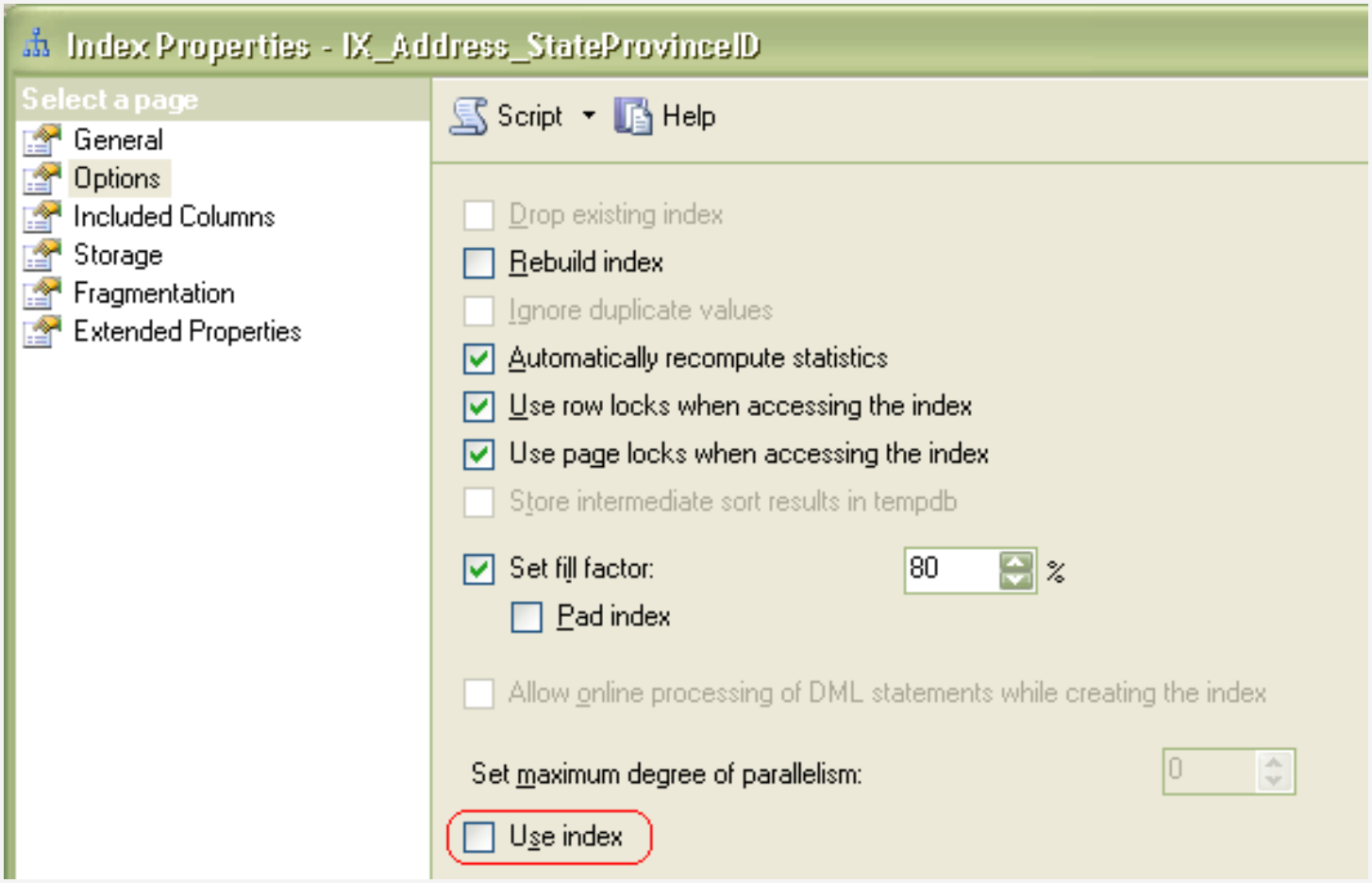
Dezactivarea Indecşilor

```
ALTER INDEX IX_Address_StateProvinceID
        ON Person.Address DISABLE
```

Dezactivarea Indecşilor



Dezactivarea Indecşilor



Reactivarea Indecşilor

```
ALTER INDEX IX_Address_StateProvinceID
        ON Person.Address REBUILD
```

Indecşi pentru ştergere

La DELETE:

- SQL Server verifică dependența înregistrărilor prin examinarea tuturor cheilor străine
- Toate tabelele identificate vor fi verificate dacă nu conțin referințe către înregistrarea de şters
  - Dacă există un index, SQL Server îl va folosi la verificare
  - Dacă nu există un index, SQL Server va trebui să scaneze întreaga tabelă
- Ştergerile pot fi foarte lente în absența indecşilor pentru chei străine

View-uri indexate

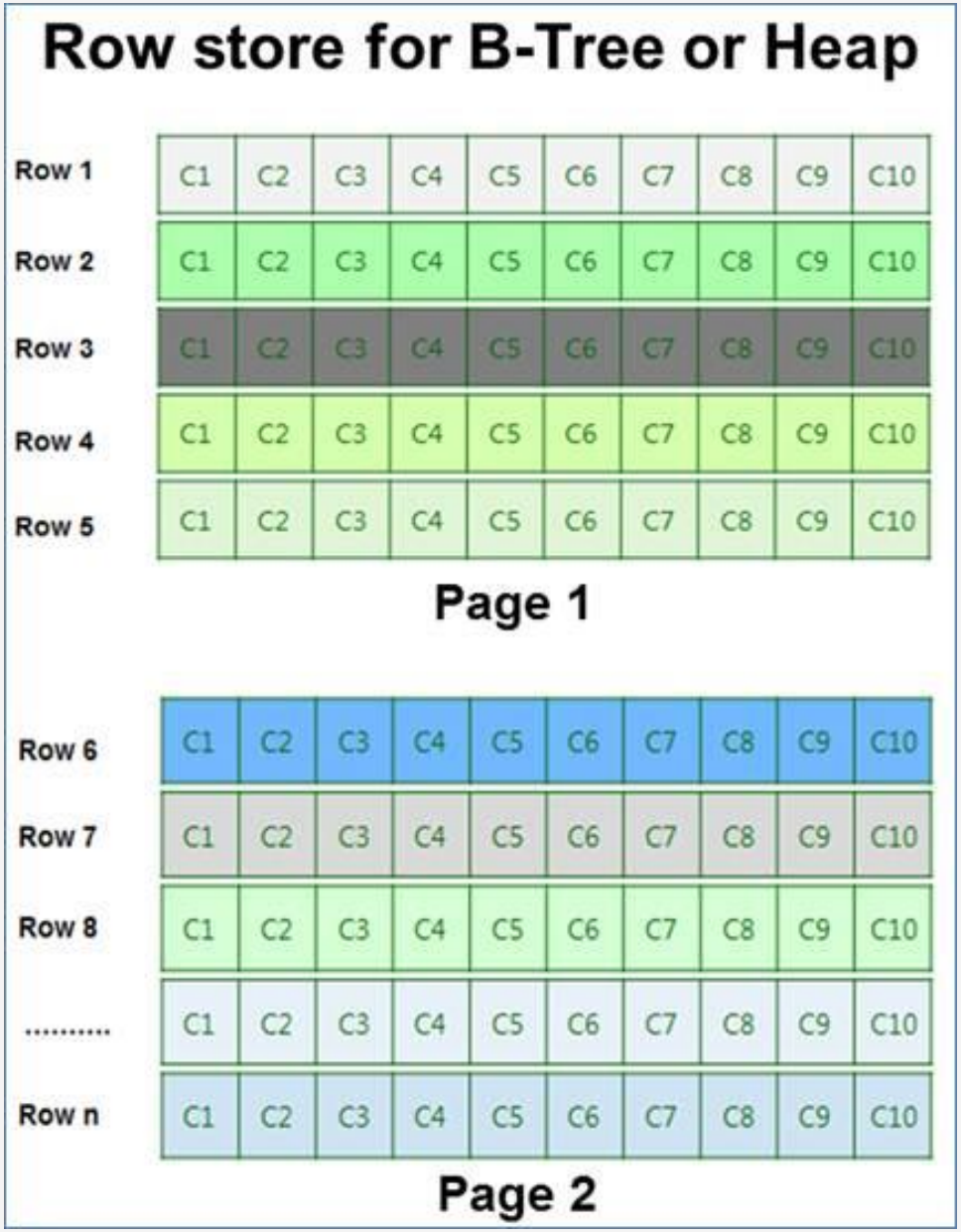
Opțiuni	Valoare necesară	Valoare implicită
ANSI_NULLS	ON	ON
ANSI_PADDING	ON	ON
ANSI_WARNINGS	ON	ON
ARITHABORT	ON	ON
CONCAT_NULL_YIELDS_NULL	ON	ON
NUMERIC_ROUNDABORT	OFF	OFF
QUOTED_IDENTIFIER	ON	ON

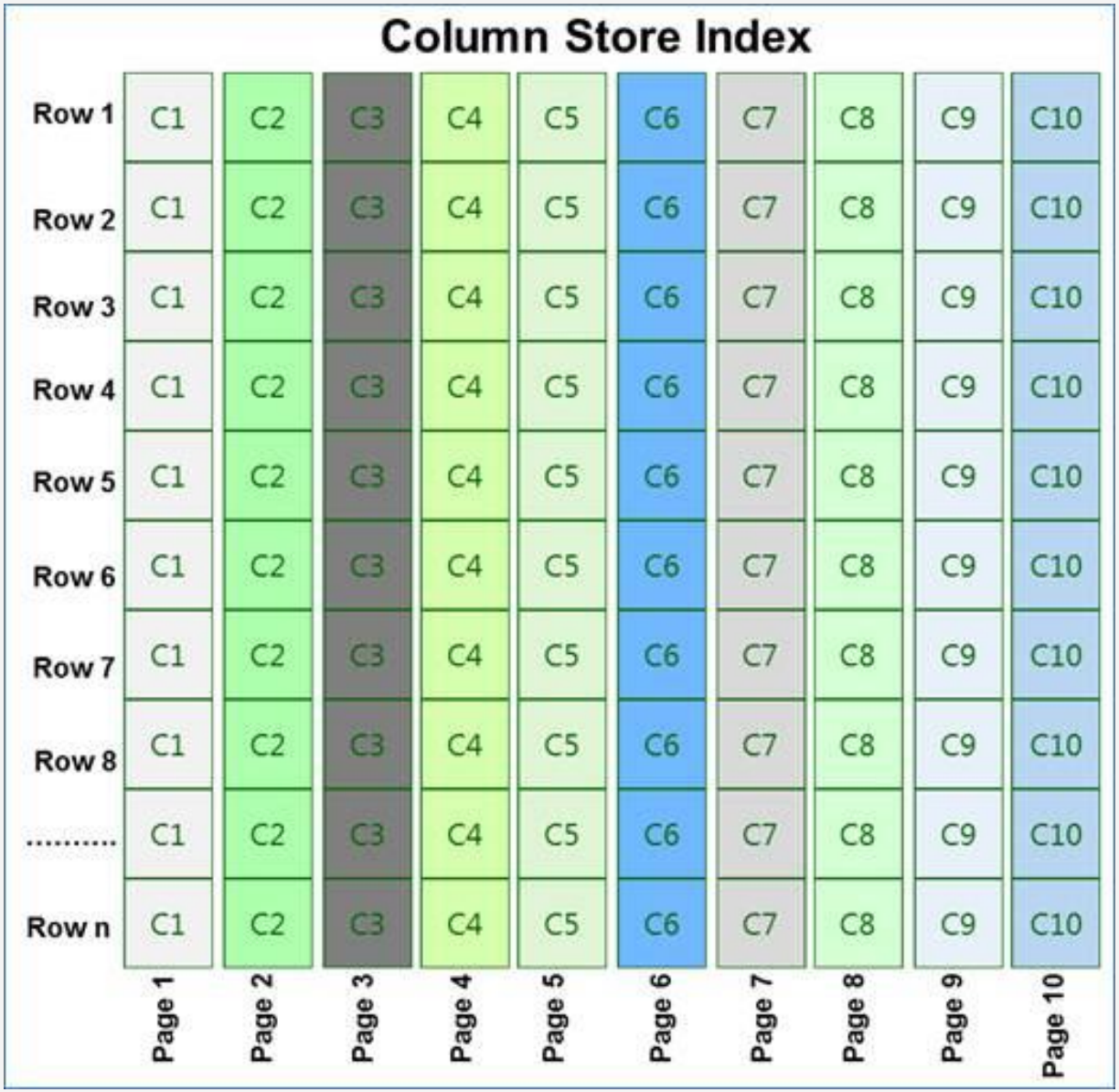
Restricții ale *view*-urilor indexate

- Interogările SELECT nu pot referi alte *view*-uri
- Interogările SELECT trebuie sa fie deterministe
- AVG, MIN, MAX, STDEV, STDEVP, VAR şi VARP nu sunt permise
- Indexul trebuie sa fie atât unic cât şi *clustered*
- Interogările SELECT nu pot conține sub-interogări, outer joins, EXCEPT, INTERSECT, TOP, UNION, ORDER BY, DISTINCT etc

Index *Column Store*

- Grupează şi stochează date pentru fiecare coloană şi apoi uneşte (join) toate coloanele pentru a completa întregul index
- Potrivit pentru bănci de date (*warehouses*) - tabele *read-only*
- Compresie a datelor de până la 10x
- Viteză de interogare de până la 10x mai mare
- Aceeasi tabela poate avea indecsi *row-store* si *column store*. Modulul de optimizare a interogarilor va decide ce index va utiliza


Row store for B-Tree or Heap
Row 1
Row 2
Row 3
Row 4
Row 5
C1 C2 C3 C4 C5 C6 C7 C8 C9 C10
Page 1
Row 6
Row 7
Row 8
.....
Row n
C1 C2 C3 C4 C5 C6 C7 C8 C9 C10
Page 2


Column Store Index
Row 1
Row 2
Row 3
Row 4
Row 5
Row 6
Row 7
Row 8
.....
Row n
C1
C2
C3
C4
C5
C6
C7
C8
C9
C10
Page 1
Page 2
Page 3
Page 4
Page 5
Page 6
Page 7
Page 8
Page 9
Page 10

Hard and fast rules for indexing

- Fiecare tabela trebuie să aibă un index *clustered* care este (in mode ideal) de dimensiuni reduse, selectiv, crescător şi static (o tabelă fără index *clustered* se numeşte *heap*.)
- Creați indecşi *non-clustered* pentru chei străine
- Creați indecşi *non-clustered* pentru câmpurile utilizate frecvent in clauza WHERE.
- Nu implementați indecşi simpli pentru fiecare câmp al unei tabele. Acest lucru va complica intreținerea tabelei.
- In indecşi compuşi, cel mai selectiv câmp (*cât mai* unic) va fi primul în cheie.
- Creați indecşi *covering* non-clustered pentru cele mai utilizate interogări

Fragmentare

- *Fragmentare internă*: înregistrările nu sunt stocate într-o zonă contiguă în interiorul paginii. Fragmentarea internă apare dacă este spațiu neutilizat între înregistrări într-o pagină.

Gradul de "umplere" a unei pagini poate varia in timp. Acest spatiu neutilizat duce la o utilizare ineficientă a cache-ului si la mai multe transferuri de pagini de memorie intre disc si memoria interna.

Fragmentare

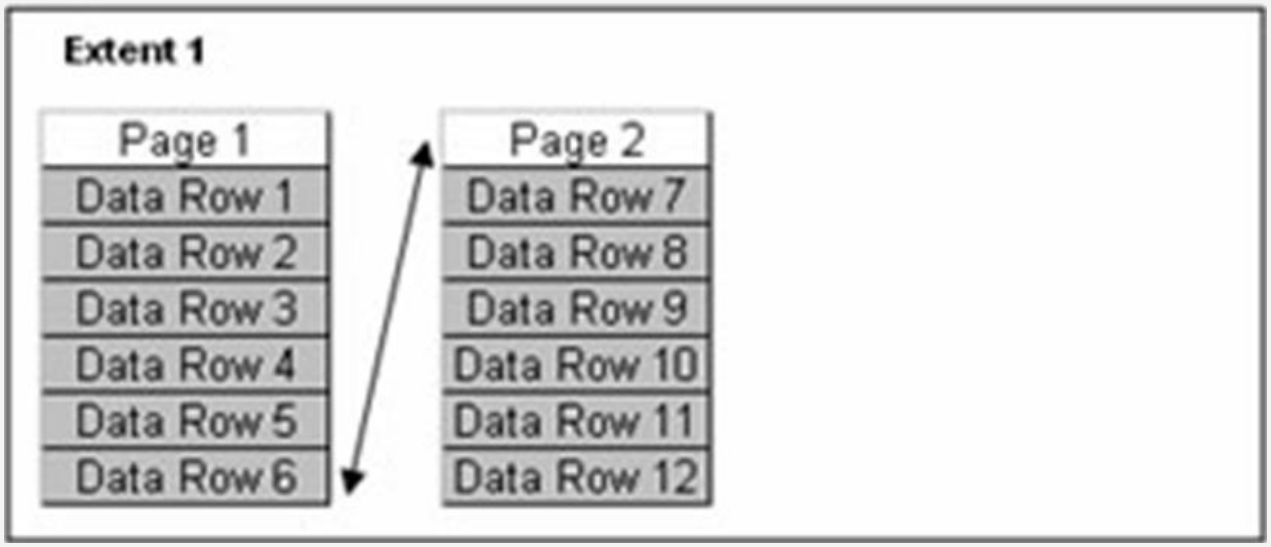
- *Fragmentare externă* : Pe disc, paginile şi extensiile (*extent* - grup de 8 pagini de memorie) nu sunt stocate într-o zonă contiguă. Trecerea de la o extensie la alta va cauza rotații mai mari ale discului.

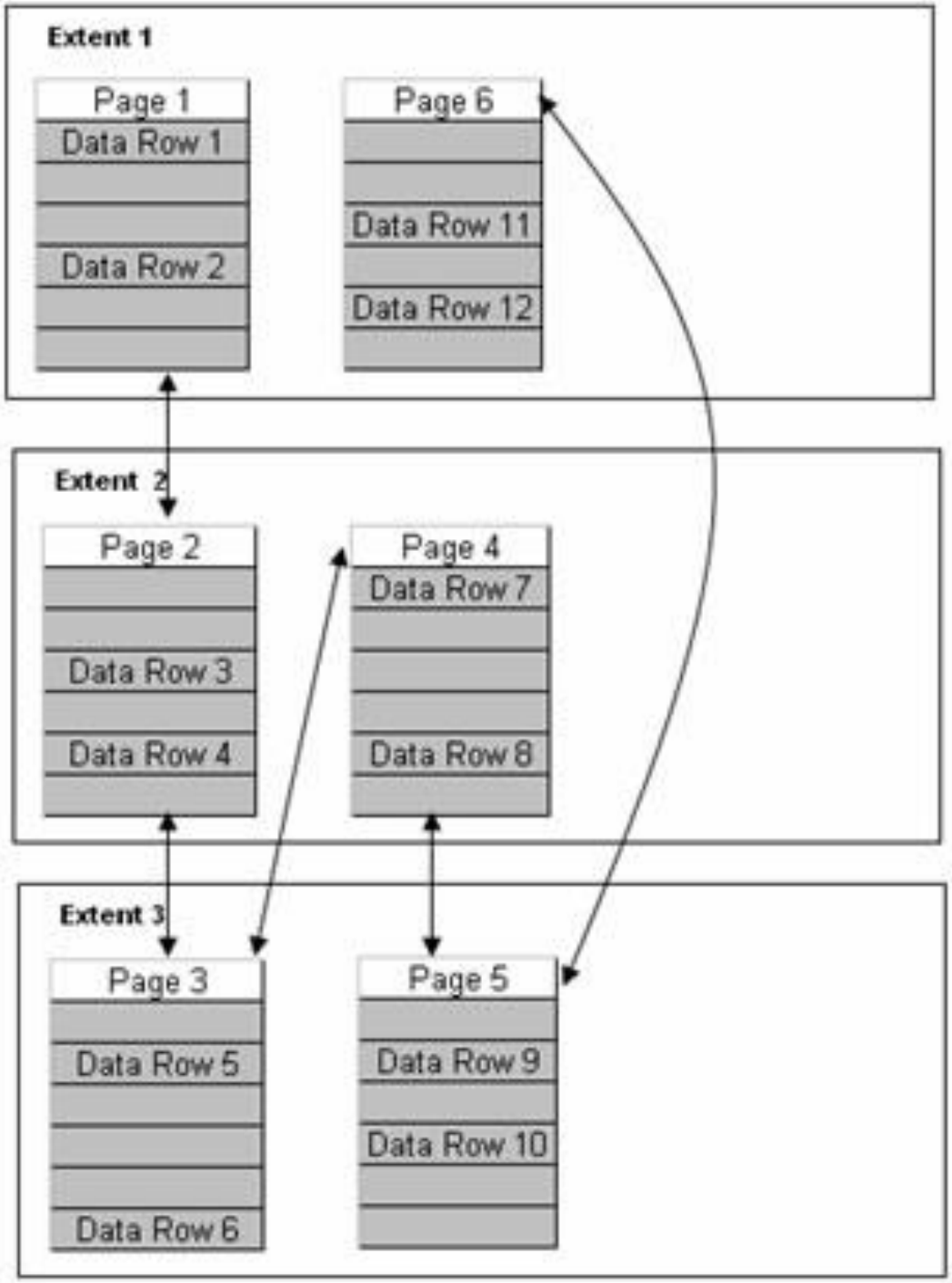
Fragmentare

- *Fragmentare logică*: Fiecare pagină a unui index este legată de precedenta şi următoarea pagină in ordinea logică a valorilor cheii. Din cauza umplerii unora dintre pagini si a redistribuirii valorilor, paginile devin *out-of-order*.

  O pagină *out-of-order* este o pagină pentru care următoarea pagină fizică din index nu este şi următoarea pagină logică.

Cereri de citire pagini: 2
Schimbări extensii: 0
Spatiu pe disc utilizat de tabela: 16 KB
avg_fragmentation_in_percent: 0
avg_page_space_used_in_percent: 100





Cereri de citire pagini : 6
Schimbări extensii: 5
Spatiu pe disc utilizat de tabela: 48 KB
avg_fragmentation_in_percent > 80
avg_page_space_used_in_percent: 33

Fragmentare

- *sys.dm_db_index_physical_stats*
  - **avg_fragmentation_in_percent**: valoare procentuală ce reprezintă fragmentarea externă
  - **avg_page_space_used_in_percent:** medie procentuală a utilizării paginilor corespunzătoare fragmentării interne.

- **Reducerea fragmentării in *heap*:**
  - Pentru reducerea fragmentării in *heap* se adaugă un index *clustered* tabelei
  - Creare inde *clustered*: se ordoneaza inregistrarile, si apoi se plaseaza in zone contigue pe disc.

```
SELECT OBJECT_NAME(object_id), index_id,index_type_desc,index_level,
avg_fragmentation_in_percent,avg_page_space_used_in_percent,page_count
FROM sys.dm_db_index_physical_stats
(DB_ID(N'AdventureWorksLT'), null, NULL, NULL , 'SAMPLED')
ORDER BY avg_fragmentation_in_percent DESC
```

Results | Messages

	(No column name)	index_id	index_type_desc	index_level	avg_fragmentation_in_perc
1	DF_BillOfMaterials_StartDate	1	CLUSTERED INDEX	0	99.009900990099
2	DF_Address_ModifiedDate	1	CLUSTERED INDEX	0	97.0588235294118
3	DF_Contact_rowguid	1	CLUSTERED INDEX	0	94.4444444444444
4	CK_Product_SafetyStockLevel	1	CLUSTERED INDEX	0	88.5714285714286
5	ContactCreditCard	1	CLUSTERED INDEX	0	85.7142857142857
6	ContactCreditCard	256000	PRIMARY XML INDEX	0	80
7	AWBuildVersion	1	CLUSTERED INDEX	0	75
8	DF_Address_ModifiedDate	2	NONCLUSTERED INDEX	0	66.6666666666667
9	DF_Contact_rowguid	2	NONCLUSTERED INDEX	0	66.6666666666667
10	CountryRegion	1	CLUSTERED INDEX	0	60
11	CountryRegion	2	NONCLUSTERED INDEX	0	50

Fragmentare

Reducerea fragmentarii intr-un Index:

- Daca *avg_fragmentation_in_percent* > 5% si < 30%, atunci folositi ALTER INDEX REORGANIZE:
  - reordoneaza frunzele indexului in dupa cheie.
- Daca *avg_fragmentation_in_percent* > 30%, atunci folositi ALTER INDEX REBUILD:
  - O varianta cu efect similar consta in stergerea si re-crearea indexului.
- Stergerea si re-crearea indexului *clustered*:
  - Re-crearea unui index clustered va determina redistribuirea datelor si are ca efect obtinerea unor pagini de date nefragmentate si "pline". Gradul de "plin" poate fi configurat prin optiunea FILLFACTOR in CREATE INDEX.