Indeksi (Microsoft SQL Server)

SOFTVERSKI ALATI BAZA PODATAKA

Indeksi

Glavna svrha indeksa je povećanje performansi. Postojanje indeksa omogućava optimizaciju upita nad bazom.

Može se vršiti poređenje sa telefonskim imenikom, gde su osobe sortirane po prezimenu, zatim po imenu kod onih osoba koje imaju isto prezime.

Indeksi

Vrste indeksa:

1. Clustered
2. Nonclustered
3. Unique
4. Filtered
5. Index with included columns
6. Index on computed columns
7. XML
8. Full Text
9. Spatial
10. Columnstore

Primer baze podataka

Data je baza podataka sa sledećim relacijama:

Status (status_code, status_desc)

Category (category_no, category_desc, category_code)

Region (region_no, region_name, street, city, state_prov, country, mail_code, phone_no, region_code)

Corporation (corp_no, corp_name, street, city, state_prov, country, mail_code, phone_no, expr_dt, region_no, corp_code)

Provider (provider_no, provider_name, street, city, state_prov, mail_code, country, phone_no, issue_dt, expr_dt, *region_no*, provider_code)

Member (member_no, lastname, firstname, middleinitial, street, city, state_prov, country, mail_code, phone_no, photograph, issue_dt, expr_dt, *region_no, corp_no,* prev_balance, curr_balance, member_code)

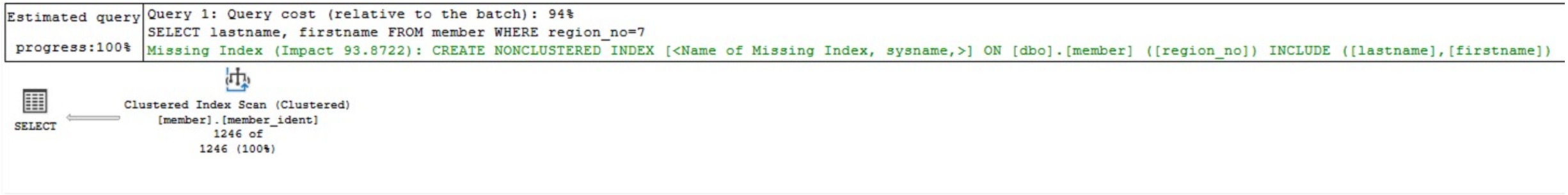
Statement (statement_no, *member_no*, statement_dt, due_dt, statement_amt, statement_code)

Charge (charge_no, *member_no, provider_no, category_no,* charge_dt, charge_amt, *statement_no*, charge_code)

Payment (payment_no, *member_no*, payment_dt, payment_amt, statement_no, payment_code)

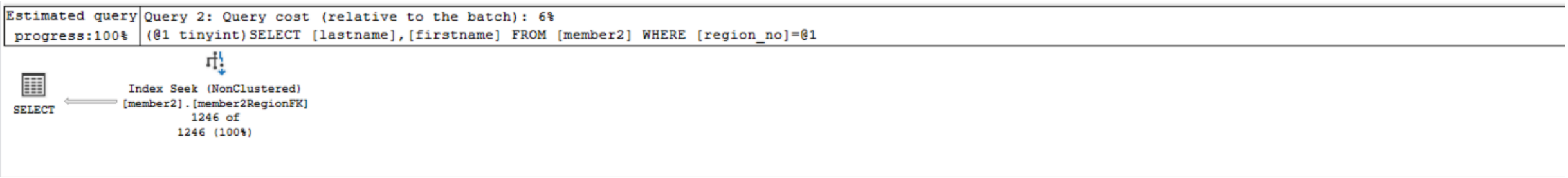
Izvršavanje upita bez odgovarajuceg indeksa

```
SELECT lastname, firstname
FROM member
WHERE region_no=7
```



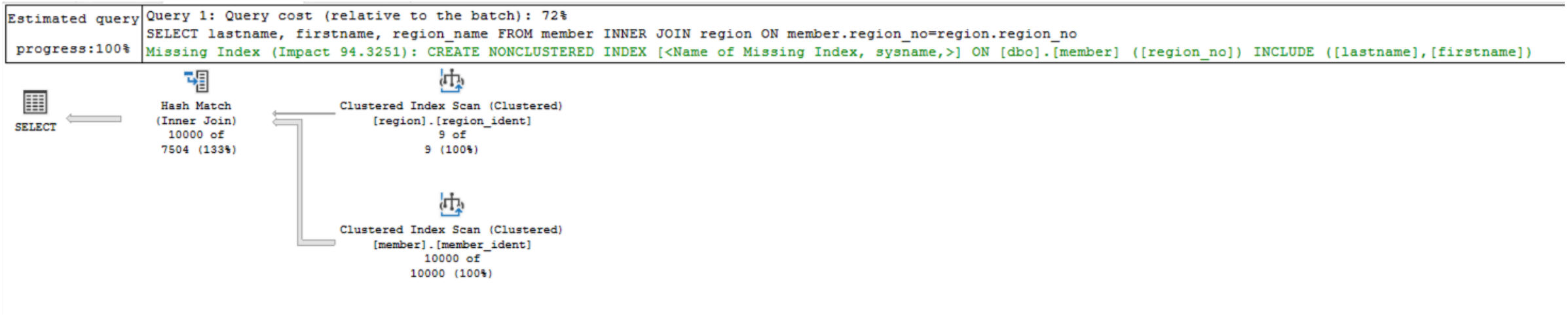
Izvršavanje upita kad indeks postoji

```
SELECT lastname, firstname
FROM member2
WHERE region_no=7
```



Izvršavanje upita bez odgovarajuceg indeksa

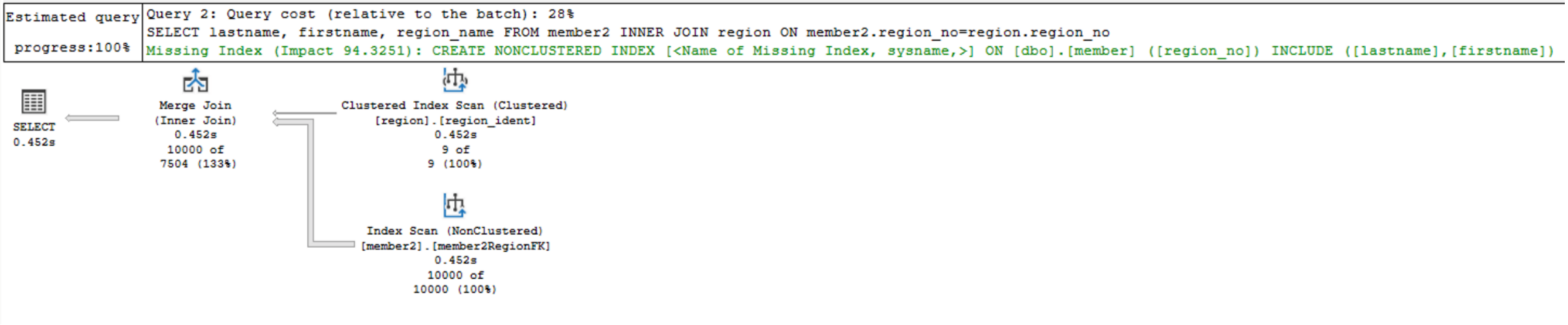
```
SELECT lastname, firstname, region_name
FROM member  INNER JOIN region ON member.region_no=region.region_no
```



Izvršavanje upita kad indeks postoji

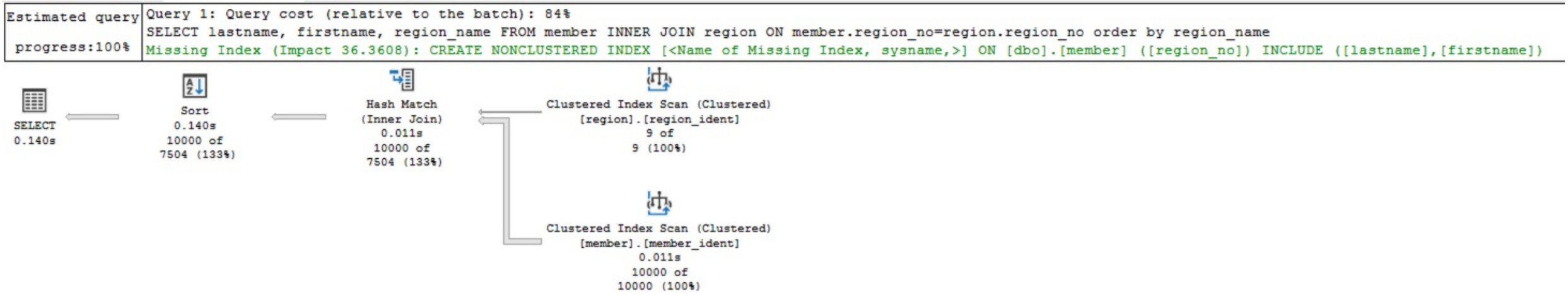
```
SELECT lastname, firstname, region_name

FROM member2  INNER JOIN region ON member2.region_no=region.region_no
```



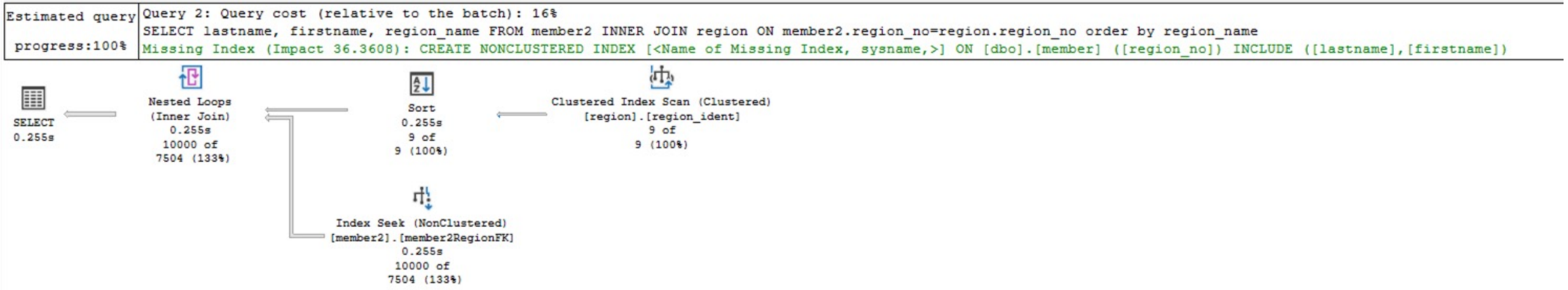
Izvršavanje upita bez odgovarajuceg indeksa

```
SELECT lastname, firstname, region_name
FROM member  INNER JOIN region ON member.region_no=region.region_no
order by region_name
```



Izvršavanje upita kad indeks postoji

```
SELECT lastname, firstname, region_name
FROM member2  INNER JOIN region ON member2.region_no=region.region_no
order by region_name
```



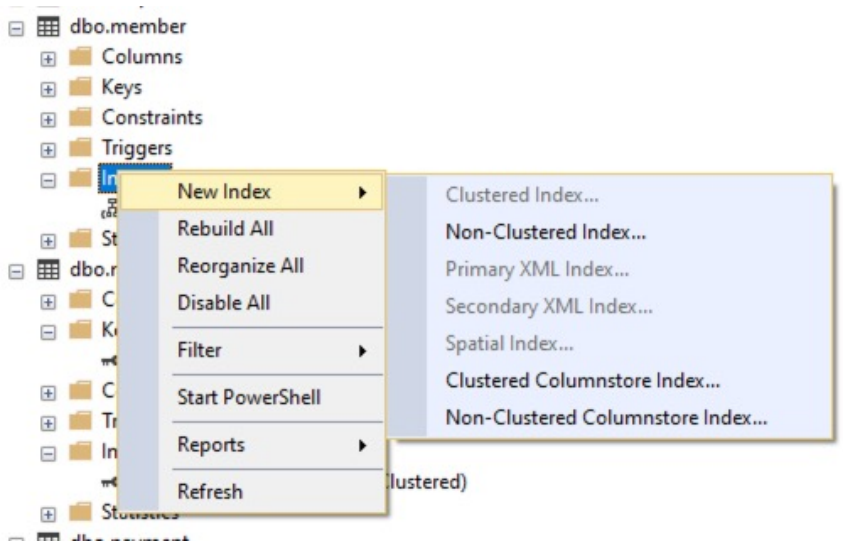
Kreiranje indeksa

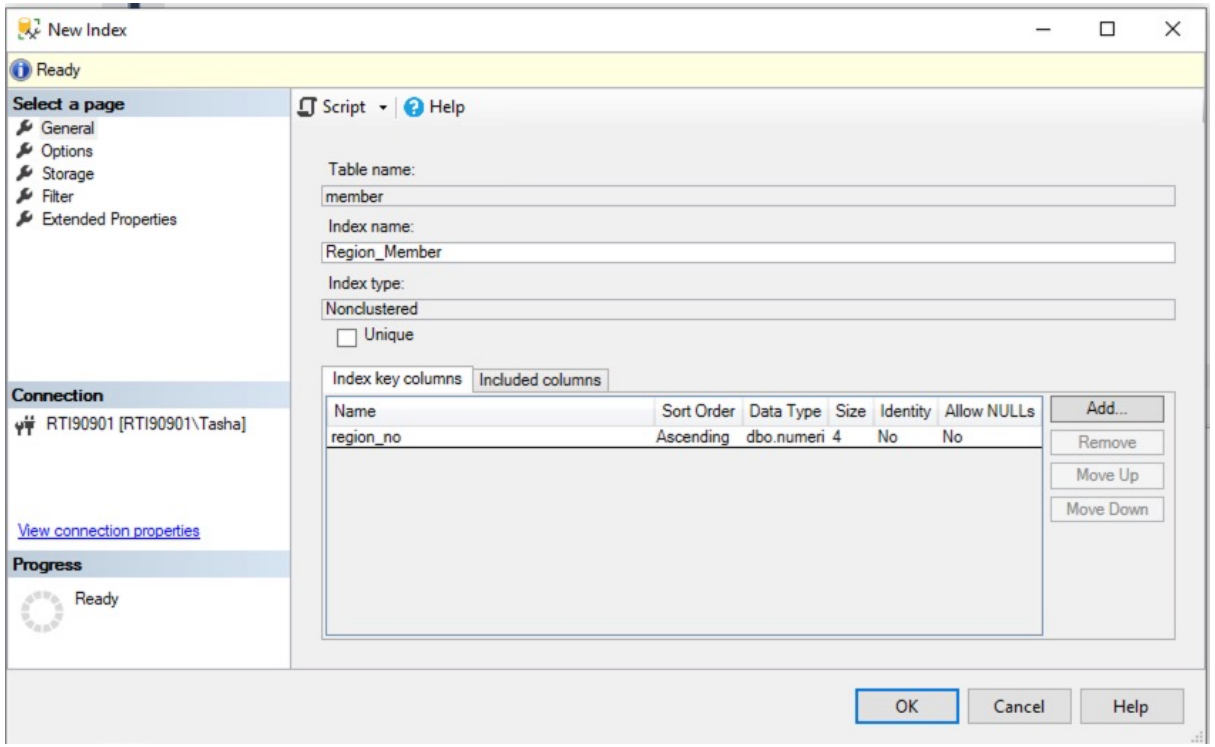
```
CREATE UNIQUE CLUSTERED INDEX ID_Member
ON Member (member_no)

CREATE INDEX Region_Member
ON Member (region_no)
```

Kreiranje indeksa

```
CREATE INDEX Region_Member
ON Member (region_no)
```





Prikaz informacija i brisanje indeksa

Prikaz svih indeksa nad tabelom:

```
EXECUTE sp_helpindex Member
```

	index_name	index_description	index_keys
1	ID_Member	clustered, unique located on PRIMARY	member_no
2	Region_Member	nonclustered located on PRIMARY	region_no

Brisanje indeksa:

```
DROP INDEX Member.ID_Member
```

Cluster index

Clustered index je analogan telefonskom imeniku. Svi podaci u samoj tabeli su sortirani po kolonama koje taj index obuhvata. U imeniku bi kolone bile prezime i ime osobe.

Ovi indexi se čuvaju sa ostalim podacima tako da ne zahtevaju dodatni prostor.

Za jednu tabelu može postojati najviše jedan clustered index!

Ukoliko pri kreiranju tabele postavimo neke od atributa kao primarne ključeve alat će u pozadini napraviti clustered index sa tim atributima.

Primer kreiranja indeksa koji sortira podatke po prezimenu opadajuće i imenu rastuće:

```
CREATE CLUSTERED INDEX Region_Member
ON Member (lastname DESC, firstname)
```

Nonclustered index

Nonclustered index za raliku od **clustered index-**a se čuva odvojeno od samih podataka, kao posebna struktura

Iz tog razloga je potreban dodatan memorijski prostor za smeštanje. Fizička baza se nije izmenila nakon kreiranja nonclustered index-a.

Jedna baza može imati više nonclustered index-a!

Primer kreiranja indeksa nad kolonom region_no koja je strani ključ:

```
CREATE NONCLUSTERED INDEX member_region
ON member ( region_no )
```

Clustered VS Nonclustered

1. Samo jedan Clustered i više Nonclustered index-a po jednoj tabeli
2. Clustered index je brži, jer Nonclustered index ima jedno referenciranje više po svakom redu zbog postojanja pokazivača.
3. Clustered zauzima manje prostora jer se podaci u fizičkoj tabeli sortitraju pri postojanju ovog indexa. Nonclustered se čuva odvojeno u posebnoj strukturi pa samim tim zahteva više prostora. Svaki nonclustered index se čuva u zasebnoj strukturi u odvojenom prostoru.

Unique index

Unique i Nonunique su osobine indeksa. Clustered index i Nonclustered index mogu imati jednu ili drugu osobinu.

Unique ukazuje da kombinacija kolona koje dati index obuhvata mora biti jedinstvena.

Primary key ogranicenje po default-u pravi **Clustered Unique Index**.

Primer:

```
CREATE UNIQUE CLUSTERED INDEX Id_Member
ON Member (member_no)
```

Unique ograničenje

Unique ogranicenje po default-u pravi **NonClustered Unique Index**.

Primer:

```
ALTER TABLE Member
ADD CONSTRAINT UQ_Member_Name
UNIQUE (firstname)
```

Unique ogranicenje može u pozadini kreirati i **Clustered Unique Index**.

Primer:

```
ALTER TABLE Member
ADD CONSTRAINT UQ_Member_Name
UNIQUE CLUSTERED (firstname)
```

Filtered index

Filtered index je noncluster indeks kod koga se u strukturi noncluster index-a ne čuvaju svi redovi iz postojeće tabele, već samo oni koji zadovoljavaju neki uslov.

Primer kreiranja indeksa nad kolonom corp_no samo u slučaju da je ona definisana:

```
CREATE NONCLUSTERED INDEX member_corp
ON member ( corp_no )
WHERE corp_no IS NOT NULL
```

Index with included columns

Index with included columns je nonclustered indeks U toj strukturi noncluster indexa moguće je čuvati podatke iz još nekih kolona. Time se gubi jedan dodatan korak redirekcije.

Primer kreiranja indeksa nad kolonom region_no koja je strani ključ u kojoj se čuvaju podaci i o imenu i prezimenu:

```
CREATE NONCLUSTERED INDEX member_region
ON member ( region_no ASC )
INCLUDE ( lastname, firstname)
```

Indeksi

Prednosti:

- SELECT iskazi se generalno brže izvršavaju

Mane:

- Dodatni prostor za NonClustered indexe
- INSERT i DELETE upiti se generalno sporije izvršavaju kod Clustered indexa jer je potrebno ispomerati redove radi unosa i brisanja novog reda. Takođe potrebno je izmeniti i stablo kod Clustered i Nonclustered indexa.

Covering query su upiti koji u SELECT naredbi zahteva samo kolone za koje je index definisan (nema referenciranja na fizičku tabelu)

Link:

https://learn.microsoft.com/en-us/sql/relational-databases/indexes/indexes?view=sql-server-ver16

Index view

Omogućavaju materijalizovanje pogleda.

Za indeks view treba zadovoljiti sledeće uslove:

- Pogled treba biti napravljen sa SchemaBinding opcijom (nije dozvoljeno brisanje tabela koje zavise od tog pogleda sve dok se sam pogled ne izbriše, i nije dozvoljeno izvršavanje ALTER TABLE iskaza nad tabelama koje učestvuju u pogledu ukoliko takav iskaz remeti definiciju samog pogleda)
- Ukoliko agregatna funkcija, koja se nalazi u SELECT listi pogleda, ima mogućnost vraćanja NULL vrednosti potrebno je onemogućiti tako nešto uvođenjem vrednosti koja će zameniti NULL
- Ukoliko postoji GROUP BY, SELECT lista mora sadržati COUNT_BIG() iskaz
- Tabele korišćene u CREATE VIEW iskazu moraju biti referisane imenom iz dva dela: ImeŠeme.ImeTabele

Ukoliko definicija pogleda sadrži GROUP BY klauzulu, tada Unique Clustered Index za taj pogled može da referiše samo kolonu specificiranu u GROUP BY klauzuli.

Index view - Primer

```
CREATE VIEW member_payment_stat WITH SCHEMABINDING
AS
SELECT  M.member_no, firstname, lastname,
        SUM(ISNULL(P.payment_amt,0)) as payment_sum, COUNT_BIG(*) AS payment_num
FROM dbo.member M INNER JOIN  dbo.payment P ON M.member_no=P.member_no
GROUP BY M.member_no, firstname, lastname
```

```
SELECT * FROM member_payment_stat
```

- Za sada pogled nije materijalizovan, pa će se iskaz iznad izvršiti u ovom SELECT iskazu

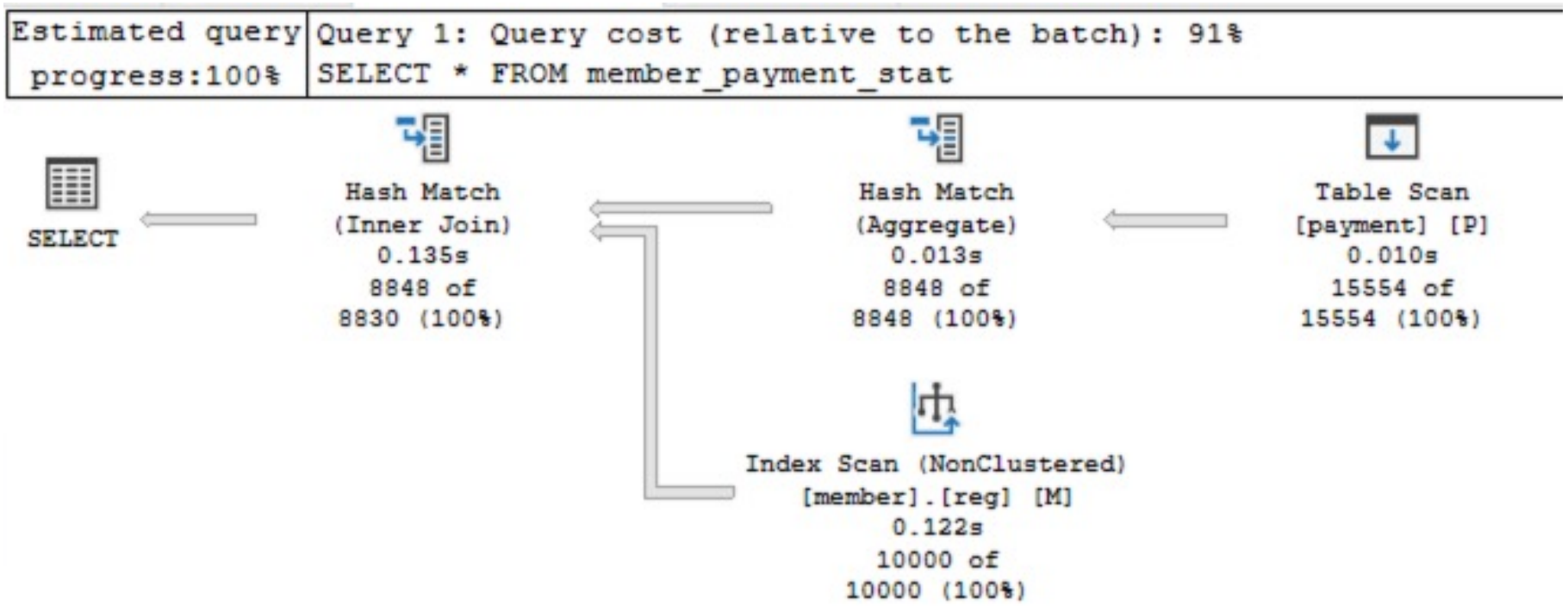
```
CREATE UNIQUE CLUSTERED INDEX UCX_member_payment_stat
ON member_payment_stat(member_no)
```

```
SELECT * FROM member_payment_stat WITH (NOEXPAND)
```

- Kada je Clustered index napravljen podaci se dobijaju iz kreirane index strukture pa se upit brže izvršava.

Izvšavanje upita bez pozivanja indexed view

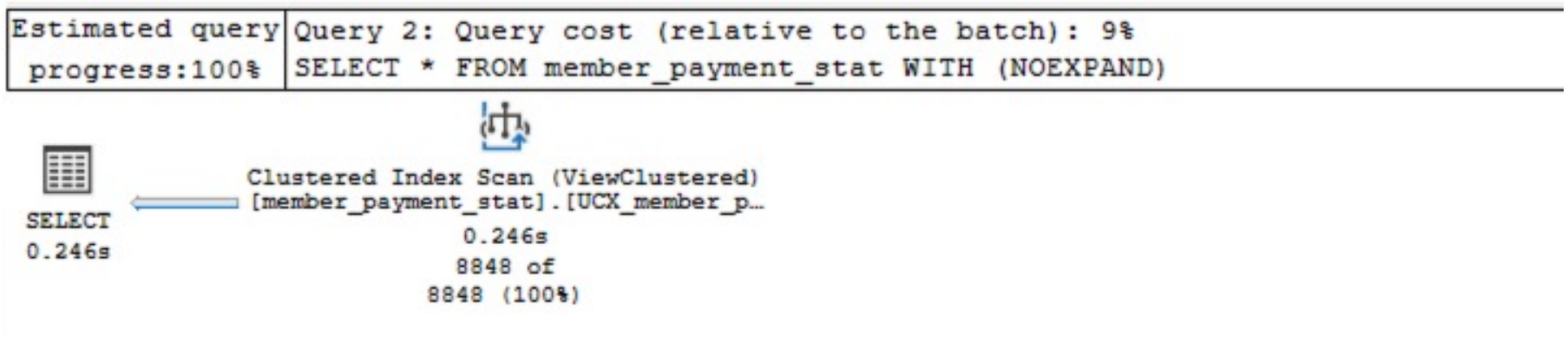
```
SELECT * FROM member_payment_stat
```



Izvšavanje upita sa pozivanjem indexed view

```
SELECT * FROM member_payment_stat WITH (NOEXPAND)
```

- **NoExpand** komada omogućava korišćenje Index view u SQL Server Standard Edition, dok se u SQL Server Enterprise Edition Indexed view poziva automatski.



Index view

Prednost:

- Ubrzava izvršavanje upita koji koriste materijalizovane poglede. Podaci koji se koriste se nalaze u indeksu, pa se iskaz koji definiše pogled ne izvršava iznova.

Mana:

- Može usporiti izvršavanje INSERT, UPDATE, DELETE iskaza nad tabelama koje su obuhvaćene indeksiranim pogledima. Ukoliko imamo dosta indeksiranih pogleda nad tabelom nad kojom vršimo neki od ova tri iskaza struktura svakog od indeksa za ove poglede će morati da se ažurira.

Link:
https://learn.microsoft.com/en-us/sql/relational-databases/views/create-indexed-views?view=sql-server-ver16