SQL Execution plans

**sp\_helpindex** je systemova storovana procedura, ktera vypise indexy na tabulce

**Scan vs Seek**

Scan rika jdi a precti vse v tabulce. To je u velkych tabulek, kde je hodne zaznamu problem.

Seek znamena, ze nad sloucem ktery hledame, existuje nonClusteredIndex a ze se server podiva do tohoto indexu, aby nasel pozadovane data.

**Paralelism**

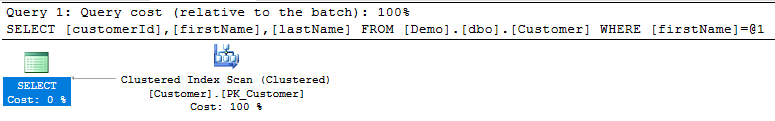
I kdyz se provadi scan, server se to snazi udelat nejlepe jak umi a pouzije k prohledavani vice vlaken.

**Actual Execution Plan** pracuje nad spustenou query, zobrazime ho kdyz je stlacene tlacitko actual execution plan pri spusteni query.

**Estimated Execution Plan** Estimated = odhadovany. Nespousti query.

**Missing index -** exekucni plan nam nabidne vylepseni, ktere by mohlo aktualni query zrychlit

Kdyz vytvorim nad sloupcem index zmenim scan na seek a zbavim se paralelniho cteni.



~~Každý krok exekučného plánu je operátor.~~

~~Každý operátor má Properties: coast, object a output list~~

~~Šípky medzi operátori ukazuji estimated and or actual rows returned~~

~~Plán se cte z prava do leva~~

~~Estimated znamená odhadovaný~~

Join znamená nested Loop tj pro kazdy radek tabulky projde joinovanou tabulku a hleda spojovany sloupec.

~~Pokud spustíme více query najednou ukáže se mi cost to thé batch, narocnost jednotlivych dotazu v rámci jedne davky.~~

~~To je výhodné například v procedurách kde se spouští více dotazů~~

Compute scalar operator se pouzije pro výpočet hodnoty pro každý řádek, například pokud je v tabulce computed column

(počet řádků je v šipce) .

**Index scan** prohledáva celou tabulku na základě clustered indexu.

**Index Seek** který projde tabulku na základě nonClustered indexu

Když mám nonClustered v tabulce na sloupci LastName a query zni :

Select FirstName from tbl where LastName = ‘Hlavenka’

Server se podiva do indexu, vezme radky kde je hledane LastName a pro tyto radky udela nested Loop i kdyz v query neni zadny join. Najoinuje vlastne k indexu puvodni tabulku na radky kde je toto LastName a v nich hleda FirstName.

**Set Statistics IO on** ukáže logical and physical reads

**Sort options** bývají často drahé operace řazení se zbavíme díky správnému indexu

**Merge** join

Je výhodný Pokud sloupce podle kterých spojujeme jsou seřazené stejně

**Hash match join**

Pokud máme velmi velký data set případně spojujeme dvě velké tabulky je výhodný hash match join vezme data set menší udělá z něj hash tabulku a ve vetsi hleda stejny hash.

**Nested Loop join**

Je výhodny jen pokud spojujeme malou tabulku s velkou. Spatny nested loops

se dá poznat tak, že počet předpokládaných řádků neodpovídá počtu aktuálne vracenych řádků (najetím na šipku).

SQL Server plány cache - uje a znovu pouziva.

**With recompile**

Při tvorbě proceduru můžeme použít klíčová slova with recompile která zaručí že server bude vytvářet plán pro proceduru znovu a znovu. To může pomoct vytvoření lepšího plánu na základě odlisnych vstupních parametrů procedury.

**Optimize**

Klicovým slovem Optimize můžu serveru říct aby exekucni plan procedury upravil tak, jako by byl vstupní parametr to, co určím jako Optimize.