**Holubec training**

1. **Row\_number Vyjed mi neco z mediadata co pouziva row number dance rank rank**

[**C:\Users\phlavenka\OneDrive\Nielsen prace\Moje poznamky Nielsen\SQL + LINQ\YouTubeTutorial\SQL TRAINING.docx**](file:///C:\Users\phlavenka\OneDrive\Nielsen%20%20prace\Moje%20poznamky%20Nielsen\SQL%20+%20LINQ\YouTubeTutorial\SQL%20TRAINING.docx)

1. **SingleThreadedAppartment STA**

Proc se musi aplikace vykreslovat v hlavnim vlakne? Uvaha: Pokud by mohly vykreslovat vsechny vlakna tak by se navzajem prekreslovaly a hlavne by se prekreslovaly podle toho jak by dobihaly napr po nacteni dat z databaze. Uzivatel ale muze mezitim klidne prekliknout jinam.

<https://d.docs.live.net/b22fb0fb09218bf0/Nielsen%20%20prace/Moje%20poznamky%20Nielsen/Dispatcher%20DispatchMessage%20SingleThreadedApartment%20STA.docx>

1. **Execution plan**

Exekucni plany:

<https://d.docs.live.net/b22fb0fb09218bf0/Nielsen%20%20prace/Moje%20poznamky%20Nielsen/SQL%20Execution%20plans.docx>

Indexy:

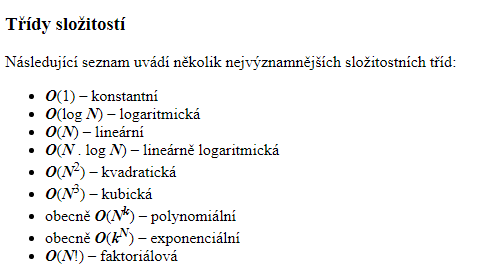
<https://d.docs.live.net/b22fb0fb09218bf0/Nielsen%20%20prace/Moje%20poznamky%20Nielsen/Indexy.docx>

1. **Zachytavadlo – analyza query #44628 -> Zachytavadlo.docx - Zadani: Analyza query Holubec**
2. **statistiky sql zevrubne, asymptoticka slozitost**

[**https://www.algoritmy.net/article/102/Asymptoticka-slozitost**](https://www.algoritmy.net/article/102/Asymptoticka-slozitost)

**Asymptoticka slozitost** – slozitost algoritmu, **udava jak je algoritmus rychly vzhledem k mnozine vstupnich dat**.

Asymptoticka slozitost je rozdeleni algoritmu do trid slozitosti, u kterych plati, ze od urcite velikosti dat je algoritmus dane tridy vzdy pomalejsi nez algoritmus tridy predchozi, bez ohledu na to, jestli je nektery z pocitacu c – nasobne vykonnejsi .



* O(1) konstantní  (výměna hodnot dvou proměnných)
* O(log n) logaritmická (vyhledání prvku v seřazeném poli metodou půlení intervalu )
* O(n) lineární (vyhledání prvku v neseřazeném poli sekvenčním vyhledáváním)
* O(n log n) lineárně logaritmická (seřazení pole reálných čísel podle velikosti algoritmem merge sort )
* O(n2) kvadratická (seřazení pole reálných čísel podle velikosti algoritmem bubble sort )
* O(n3) kubická (naivní násobení dvou matic)
* O(nc) pro c > 0 polynomiální
* O(cn) proc c > 1 exponenciální (problém obchodního cestujícího)

Jedna operace zde trvá jednu nanosekundu. Na procesoru o frekvecni 2GH trva 1/ 2 000 000 000 coz je asi 0.5 ns



Amortizovana casova slozitost – oznacuje casovou slozitost algoritmu v sekvenci nejhorsich moznych vstupnich dat. Nevyuziva pravdepodobnosti a je proto zarucena. Je to **průměrný čas potřebný pro vykonání určité operace v sekvenci operací v nejhorším případě**. Na rozdíl od časové složitosti v průměrném případě nevyužívá pravděpodobnosti a předpokladů o rozložení dat. U amortizované složitosti je **průměrný čas na operaci** skutečně zaručený.

Některé datové struktury mají totiž takovou vnitřní organizaci, že na ní závisí složitost, a organizovanost dat se může během posloupnosti operací měnit. Základní myšlenka amortizované analýzy tkví v tom, že operace s velkou složitostí změní stav struktury tak, že tento nejhorší případ nemůže nastat po dlouhý čas, tudíž amortizuje svou cenu.

Amortizovaná složitost se používá v případě, kdy některá konkrétní operace (typicky na datové struktuře) má v nejhorším případě velkou složitost, ale na vykonání této operace si dokážeme našetřit z předchozích operací v (libovolné) posloupnosti.

Jako jednoduchý příklad můžeme uvést specifickou implementaci dynamického pole, která zdvojnásobuje velikost pole pokaždé, když dojde k jeho naplnění. V tomto případě je tedy nutná realokace, v nejhorším případě tato operace potřebuje čas až O(N). Samotné vkládání prvků (bez nutnosti realokace) vyžaduje čas O(1), pro N prvků tedy také O(N). Pro vložení N prvků (včetně realokace) je tedy potřeba O(N) + O(N) = O(N), amortizovaný čas na jedno vložení prvku je pak O(N)/N = O(1).

Optimalizace query

Connection pooling hospodareni se spojenim se serverem

exec sp\_executesql N'

SELECT mm.Id, mm.NormCreativeId, mm.CreativeId, mm.AdvertisedFrom, mm.AdvertisedTo, mm.CreatedBy, mm.ModifiedBy, tmm.Footage, mm.Modified, mm.MediumId, mdv.Name AS Medium,

mm.PlacementId, mm.Note,

mm.AdvertisementTypeId

FROM Media.MediaMessage mm

INNER JOIN Media.TvMediaMessage tmm ON mm.Id = tmm.Id

INNER JOIN Media.Medium md ON mm.MediumId = md.Id

INNER JOIN Media.MediumVersion mdv ON md.Id = mdv.MediumId AND mdv.ActiveFrom <= mm.AdvertisedFrom AND mdv.ActiveTo > mm.AdvertisedFrom

INNER JOIN Media.CodingPlausibility cp ON mm.CodingPlausibilityId = cp.Id

WHERE mm.AdvertisedFrom >= @fromHelper AND mm.AdvertisedFrom < @to AND mm.AdvertisedTo > @from AND mm.AdvertisedTo < @to

AND mm.MediumId IN (497)

--;WITH A AS -- #42544 Nahraj pro kazdou MM TapeCode od normy. Vezme se TapeCode od nejstarsi zpravy normy

--(

-- SELECT temp.\*, ti2.TapeCode AS NormsTapeCode,

-- ROW\_NUMBER() OVER (PARTITION BY mm2.NormCreativeId, temp.Id ORDER BY mm2.Created) AS RN

-- FROM @TempCatchingMessages temp

-- JOIN Media.MediaMessage mm2 ON mm2.NormCreativeId = temp.NormCreativeId

-- JOIN Media.TvMediaMessage tmm2 ON mm2.Id = tmm2.Id

-- LEFT JOIN Media.TapeInfo ti2 ON tmm2.TapeInfoId = ti2.Id

--)

--SELECT \* FROM A WHERE a.RN = 1

',N'@from datetime,@to datetime,@fromHelper datetime',@from='2018-08-18 07:59:55.600',@to='2018-08-18 09:15:25.600',@fromHelper='2018-08-18 06:59:55.600'

**Vyzkoumej, co by v Kodovadle zjednodusilo kod, v com nejvic plaves, co by novackum pomohlo v orientaci v kodovadle. Nejakou technologii.**

hledani spravnych VM podle View.

roztahovani oken, jednou jsem ho otevrel na notebooku a nektere veci mi zmizely, s tim, ze okno neslo ani posunout

Auditable

Request – response

Await - async

Thready, cancellation token, wait one()

**Thread**

new Thread(threadstart delegat)

Thread.Start(), Sleep(), Join()

**Tasks –** continueWith, TaskScheduler.FromCurrentSynchronisationContext, Continuation option, CancellationToken

**Await-async** – wrapper nad Tasky. Await pozastavi metodu, spusteni v jinem vlakne Task.Run()

<https://d.docs.live.net/b22fb0fb09218bf0/Nielsen%20%20prace/Moje%20poznamky%20Nielsen/Tasks.docx>

**Synchronizacni primitiva**

Wait.One() – vlakno ktere tuto metodu executne se pauzne, dokud jine vlakno nezavola Handle.Set()

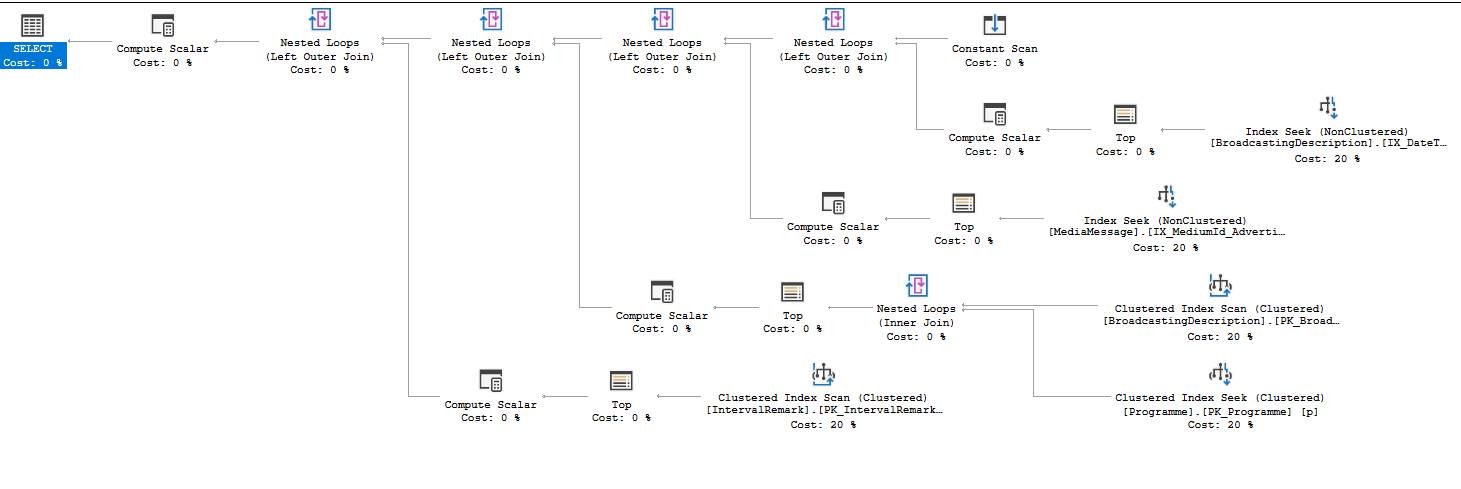
**Dispatcher**

[CurrentDispatcher STA UIThread.docx](CurrentDispatcher%20STA%20%20UIThread.docx)

[Dispatcher DispatchMessage SingleThreadedApartment STA.docx](Dispatcher%20DispatchMessage%20SingleThreadedApartment%20STA.docx)

Invoke, BeginInvoke, MessageLoop,

# Indexy – proc jsou tak slozene, proc je dotaz neefektivni:



(1 row affected)

Table 'IntervalRemark'. Scan count 1, logical reads 377, physical reads 0, read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob read-ahead reads 0.

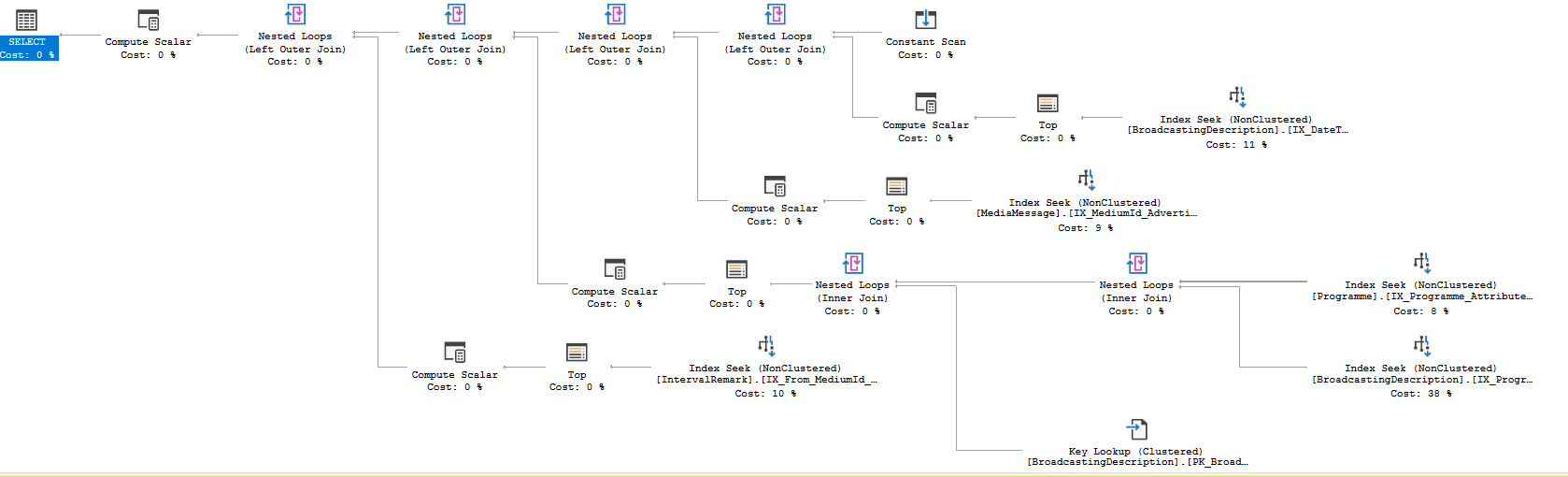
Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob read-ahead reads 0.

Table 'BroadcastingDescription'. Scan count 19, logical reads 8231, physical reads 0, read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob read-ahead reads 0.

Table 'Programme'. Scan count 2, logical reads 6, physical reads 0, read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob read-ahead reads 0.

Table 'MediaMessage'. Scan count 1, logical reads 4, physical reads 0, read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob read-ahead reads 0.

Po pridani indexu (create script nize)



(1 row affected)

Table 'IntervalRemark'. Scan count 1, logical reads 152, physical reads 0, read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob read-ahead reads 0.

Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob read-ahead reads 0.

Table 'BroadcastingDescription'. Scan count 19, logical reads 8231, physical reads 0, read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob read-ahead reads 0.

Table 'Programme'. Scan count 2, logical reads 6, physical reads 0, read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob read-ahead reads 0.

Table 'MediaMessage'. Scan count 1, logical reads 4, physical reads 0, read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob read-ahead reads 0.

Po uprave dotazu (upraveny dotaz nize)

(1 row affected)

Table 'IntervalRemark'. Scan count 1, logical reads 2, physical reads 0, read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob read-ahead reads 0.

Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob read-ahead reads 0.

Table 'BroadcastingDescription'. Scan count 19, logical reads 8231, physical reads 0, read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob read-ahead reads 0.

Table 'Programme'. Scan count 2, logical reads 6, physical reads 0, read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob read-ahead reads 0.

Table 'MediaMessage'. Scan count 1, logical reads 4, physical reads 0, read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob read-ahead reads 0.

**upraveny dotaz**

declare @mediumId tinyint, @dateTimeFrom datetime, @dateTimeTo datetime;

set @mediumId = '68'; --lze preskocit

--set @mediumId = '1'; --nelze preskocit

set @dateTimeFrom = '2019-04-12 01:00:00.000';

set @dateTimeTo = dateadd(hour, 1, @dateTimeFrom);

select CAST(

        ISNULL(

        (SELECT TOP 1 1

        FROM SimLog.BroadcastingDescription bd

        WHERE bd.MediumId IN (@mediumId) AND bd.DateTimeFrom > DATEADD(HOUR, -12, @dateTimeFrom) AND bd.DateTimeTo >= @dateTimeTo)

        , 0)

        \*

        ISNULL(

        (SELECT TOP 1 0

        FROM Media.MediaMessage mm

        where mm.MediumId IN (@mediumId) AND mm.AdvertisedFrom > DATEADD(HOUR, -1, @dateTimeFrom) AND mm.AdvertisedFrom < @dateTimeTo AND mm.AdvertisedTo > @dateTimeFrom)

        , 1)

             \*

             ISNULL(

        (SELECT TOP 1 0

        FROM SimLog.BroadcastingDescription bd

            inner join [SimLog].[Programme] [p] on [p].[Id] = [bd].[ProgrammeId]

        where bd.MediumId IN (@mediumId) AND bd.DateTimeFrom > DATEADD(HOUR, -1, @dateTimeFrom) AND bd.DateTimeFrom < @dateTimeTo AND bd.DateTimeTo > @dateTimeFrom

                           and [p].[AttributeId] in (1, 3))

        , 1)

             \*

             ISNULL(

             (SELECT TOP 1 0

             FROM SimLog.IntervalRemark ir

             WHERE ir.MediumId IN (@mediumId) AND ir.[From] < @dateTimeTo AND ir.[To] > @dateTimeFrom AND ir.[From] > DATEADD(HOUR, -1, @dateTimeFrom))

             , 1) AS bit)

**create script na index**

CREATE NONCLUSTERED INDEX [IX\_From\_MediumId\_i\_IntervalRemarkTypeId\_To] ON [SimLog].[IntervalRemark]

(

       [From] ASC,

       [MediumId] ASC

)

INCLUDE (    [IntervalRemarkTypeId],

       [To]) WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, SORT\_IN\_TEMPDB = OFF, DROP\_EXISTING = OFF, ONLINE = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

Prespristi utery se zeptam na tyto otazky (obou samostatne):

* proc je index poskladany tak, jak je
* proc je bez upravy dotazu stale neefektivni

Kdyz mi pri tom odpovite i na nejake doplnujici otazky, mozna se dohodnu s Hobbem na nejakem malem skokanovi :)

Petr

**From:** Karel Honzl   
**Sent:** Thursday, April 18, 2019 1:45 PM  
**To:** Petr Holubec <[Petr.Holubec@admosphere.cz](mailto:Petr.Holubec@admosphere.cz)>  
**Subject:** rozsireni podminky pro preskoceni streamu v Z o IntervalRemark

Ahoj,

Posilam spustitelnou query:

use MediaData3BGAuto

declare @mediumId tinyint, @dateTimeFrom datetime, @dateTimeTo datetime;

set @mediumId = '68'; --lze preskocit

--set @mediumId = '1'; --nelze preskocit

set @dateTimeFrom = '2019-04-12 01:00:00.000';

set @dateTimeTo = dateadd(hour, 1, @dateTimeFrom);

select CAST(

        ISNULL(

        (SELECT TOP 1 1

        FROM SimLog.BroadcastingDescription bd

        WHERE bd.MediumId IN (@mediumId) AND bd.DateTimeFrom > DATEADD(HOUR, -12, @dateTimeFrom) AND bd.DateTimeTo >= @dateTimeTo)

        , 0)

        \*

        ISNULL(

        (SELECT TOP 1 0

        FROM Media.MediaMessage mm

        where mm.MediumId IN (@mediumId) AND mm.AdvertisedFrom > DATEADD(HOUR, -1, @dateTimeFrom) AND mm.AdvertisedFrom < @dateTimeTo AND mm.AdvertisedTo > @dateTimeFrom)

        , 1)

             \*

             ISNULL(

        (SELECT TOP 1 0

        FROM SimLog.BroadcastingDescription bd

            inner join [SimLog].[Programme] [p] on [p].[Id] = [bd].[ProgrammeId]

        where bd.MediumId IN (@mediumId) AND bd.DateTimeFrom > DATEADD(HOUR, -1, @dateTimeFrom) AND bd.DateTimeFrom < @dateTimeTo AND bd.DateTimeTo > @dateTimeFrom

                           and [p].[AttributeId] in (1, 3))

        , 1)

             \*

             ISNULL(

             (SELECT TOP 1 0

             FROM SimLog.IntervalRemark ir

             WHERE ir.MediumId IN (@mediumId) AND ir.[From] < @dateTimeTo AND ir.[To] > @dateTimeFrom)

             , 1) AS bit)

K.

-------------------------------------------------------

* proc je index poskladany tak, jak je

Pridaval se check na Interval remark. Bez indexu je tam skaredy index scan( index tam je nad Id, protoze je tam primary key). Protoze nema zadny jiny index, musi projit vsechny radky, aby mohl najit pozadovane datumy. Logickym krokem, je pridat index nad sloupce From a To.

**Include klauzule** se pouziva v pripadech, kde je dobre znat hodnotu sloupce uz v ramci indexu bez toho abychom pri kazde nalezene hodnote museli udelat lookup do dat.

Je vhodne to pouzit v tabulkach kde se casto neupdatuje.

Proc je v include IntervalRemarkTypeId nevim, v query se nepouziva.

* proc je bez upravy dotazu stale neefektivni

jde o ready – tj o pocet tabulek, ktere musi sql server projit, aby se dostal k vysledku. Mnozstvi readu snizime tak, ze polozime dotaz ktery je vice selektivni.