Task Engine

Release 4

Handleiding

1 Introductie 3

1.1 Doel van Task Engine 3

1.2 Begrippen 3

1.3 Achtergrond 3

1.4 Task Engine in relatie tot SQL Agent 4

1.5 Task Engine versus SSIS 4

1.6 Splitsing code en data 4

2 Toepassing 5

2.1 Universeel inzetbaar 5

2.2 ETL 5

3 Belangrijkste features 6

3.1 Dynamic Fast Lane 6
3.1.1 Waar komt de term Fast Lane vandaan? 6
3.1.2 “Dynamic”, hoezo? 6
3.1.3 Implementatie in Task Engine 6
3.2 Doorlooptijd optimalisatie 7
4 Vereisten 8
4.1 SQL Server versies 8
4.2 Automatische configuratie 8
5 Installatie en configuratie 9
6 Gebruik 10
6.1 Autobatch 11
6.2 Zelf taken aanmaken 11

1 Introductie

N.B.: deze documentatie is nog in bewerking

1.1 Doel van Task Engine

Doel is zo optimaal mogelijk uitvoeren van Batches T-SQL taken, daarbij wordt er een slimme mix gemaakt van parallelle en sequentiële taken.

Task Engine houdt rekening met:

- CPU belasting;
- Afhankelijkheden tussen taken;
- Extra prio voor een subset taken (Fast Lane functie).

Belangrijkste kenmerken:

- CPU wordt optimaal benut;
- Taken worden zoveel mogelijk parallel uitgevoerd;
- Fast Lane functie minimaliseert wachttijden;
- Zelflerend: looptijdoptimalisatie o.b.v. de gemiddelde looptijden.

1.2 Begrippen

Bij Task Engine zijn de belangrijkste begrippen:

- Instance: SQL Server installatie;
- Task: een specifieke Stored Procedure;
- Batch: het totaal van Stored Procedures voor een gezamenlijk doel;
- Prio, Rank en Lane; groeperende / controlerende attributen t.b.v. de algemene flow.

Eén en ander wordt hierna verder uitgelegd.

Normaliter iets dergelijks alleen mogelijk door gebruik te maken van SSIS, Task Engine is hiervoor een flexibel alternatief en is zeer geschikt om in ETL-achtige omgevingen ingezet te worden.

1.3 Achtergrond

Task Engine's eerste code dateert van januari 2014 en het heeft zich sindsdien ruimschoots bewezen in grote datamigratie projecten in de Nederlandse Zorgsector waarbij batches van meer dan 2.500 taken en doorlooptijden van vele uren werden aangeboden en verwerkt.

In eerste instantie had het geen 'gezicht' en werd het puur vanuit de back-end geconfigureerd en aangestuurd.

Sindsdien is het uitgegroeid tot een volwaardig product met een Console voor configuratie en monitoring.

1.4 Task Engine in relatie tot SQL Agent

SQL Agent prima in staat om meerdere taken (Jobs) tegelijk uit te voeren, daarom wordt SQL Agent door Task Engine gebruikt. Daarbij fungeert Task Engine als intermediair tussen de betreffende applicatie en SQL Agent; Task Engine maakt de Jobs gecontroleerd aan[1] en SQL Agent voert ze uit.

1.5 Task Engine versus SSIS

SSIS is het enige standaard alternatief naast SQL Agent om taken parallel uit te voeren maar hoewel SSIS veelzijdiger is dan Task Engine wat het soort taken betreft dat het aankan, is Task Engine veel dynamischer dan SSIS:

- Bouwen van een flow is niet nodig, het aanbieden of registreren van de taken is voldoende;
- De voor SSIS vereiste aanvullende software en kennis daarvan is niet nodig:
  - SSIS Service;
  - Visual Studio;
  - Sql Server Data Tools.

Het belangrijkste probleem met SSIS is dat het erg gevoelig is voor wijzigingen in DB-objecten die hele packages ongeldig kunnen maken (en er zijn *altijd* wijzigingen). Omdat Task Engine alleen de flow van een vooraf gedefinieerde set opgeslagen procedures regelt en zich niet bezig houdt met onderliggende metadata, heeft het dat probleem niet.

1.6 Splitsing code en data

Task Engine heeft twee eigen databases, één waar de functionaliteit is ondergebracht, en de andere voor opslag van data. Zodoende kan de functionaliteit worden vervangen door een vernieuwde versie van die database zonder consequenties voor de bestaande data.

[1] Hierin schuilt de kracht van Task Engine, later meer hierover.

2 Toepassing

2.1 Universeel inzetbaar

Task Engine heeft geen kennis van de processen waarvoor het wordt ingezet, dat hoeft ook niet want opdrachten worden met bepaalde parameters aangeboden op basis waarvan Task Engine ze oppakt en als Task uitvoert. Deze aanpak maakt Task Engine universeel inzetbaar; uitgaande van 3-part names[2] kan Task Engine elke Stored Procedure uitvoeren.

2.2 ETL

Als voorbeeld wordt een standaard ETL proces gebruikt. Biedt bijvoorbeeld als volgt alle taken aan Task Engine aan:

- Extractietaken met Prio 1;
- Transformaties met Prio 2;
- Load taken met Prio 3.

Zolang de CPU het toelaat (instelbare drempel) zullen er zoveel mogelijk Taken met dezelfde Prio tegelijk als Job worden uitgevoerd.

Wanneer de laatste Prio 1 is voltooid worden de Prio 2 taken als Job uitgevoerd en uiteindelijk de Prio 3 taken.

Naast Prio specificeert u ook Rank; taken worden binnen een Prio op volgorde van Rank uitgevoerd, dus eerst alle Prio's 1 met Rank 1 en dan alle Prio's 1 met Rank 2 enz.

Stel dat er bij de Extractie een rapportage moet worden gedraaid die bijv. aantallen verzamelt en rapporteert. Deze moet uiteraard pas *na* de laatste Extractietaak worden uitgevoerd.
U bereikt dit door bij de reguliere Extractietaken Rank 1 e.d. te specificeren en bij de rapportage een Rank van bijv. 99.

Maar… het kan best zijn dat die rapportage lang duurt terwijl de volgende fase (Transformatie) daar niet van afhankelijk is dus u wil eigenlijk dat dat doorloopt. Een mogelijkheid is om alle rapportages m.b.v. een hoge Prio als laatste te laten lopen maar dat is niet optimaal, we komen daarop terug in het volgende hoofdstuk bij ***3.1.3***.

[2] Ook 4-part names zijn uiteraard mogelijk, of dat qua performance verstandig is, is per geval verschillend.

3 Belangrijkste features

3.1 Dynamic Fast Lane

3.1.1 Waar komt de term Fast Lane vandaan?

Deze term impliceert dat er ook een Slow Lane bestaat, beide termen zijn zelfstandige naamwoorden overgenomen uit het Engels en worden gebruikt in het verkeer:

- Fast lane: “the traffic lane for vehicles that are moving rapidly”
- Slow lane: “the traffic lane for vehicles that are moving slowly”.

Misschien heeft u wel eens op een snelweg door de bergen gereden; bij een steile helling is er vaak een tijdelijke extra baan, bij rechts rijdend verkeer uiteraard aan de rechterkant. Deze is bestemd voor voertuigen die moeite hebben om de helling op te komen zoals vrachtauto’s en auto’s met caravan, zodat het verkeer dat makkelijk omhoog rijdt daar geen hinder van heeft.

Dit is een Slow Lane, de andere banen zijn dan Fast Lanes.

De Fast Lane functie van Task Engine is ontworpen t.b.v. processen die onafhankelijk zijn van afrondende zaken van voorgaande processen en alvast kunnen worden gestart om simultaan te lopen daarmee.

3.1.2 “Dynamic”, hoezo?

Het Fast Lane principe wordt gebruikt wanneer er nog actieve taken zijn binnen dezelfde Batch, met dezelfde Prio maar verdeeld over meerdere Lanes; de Lane met het laagste[3] nummer wordt als Fast Lane behandeld, met als doel om wachttijden daar te minimaliseren.

Wanneer een Fast Lane niet meer actief is maar er zijn nog wel meerdere andere Lanes binnen dezelfde Batch met dezelfde Prio, dan wordt de eerst hogere[3] Lane de nieuwe Fast Lane.

3.1.3 Implementatie in Task Engine

Het werkt als volgt, voortbordurend op het ETL voorbeeld van hiervoor:

De Extractie heeft Prio 1, volgende fase is Transformatie met Prio 2.

Logisch zou zijn om het Extractie rapport Prio 1, Rank 99 te geven zodat dit als laatste van de Extractie draait. Nadeel hiervan is dat volgende fase (Transformatie) pas gaat draaien wanneer het

[3] Dit is het Default gedrag, zie ***Samenvattend*** aan het einde van deze alinea.

Extractie rapport klaar is terwijl dit lang kan duren i.v.m. evt. tellingen bijvoorbeeld, terwijl er geen afhankelijkheden tussen Extractie rapport en Transformatie zijn.

Door het Extractie rapport nu dezelfde Prio te geven als de volgende fase (Transformatie) maar een hogere[4] Lane, kan dit rapport tegelijk lopen met de Transformatie, Transformatie wacht dan niet en doorloopt in de Fast Lane alle Ranks terwijl het Extractie rapport rustig door pruttelt.

Het gedrag van de Fast Lane is configureerbaar; bij design worden nl. niet alle taken in de Fast Lane als eerste opgepakt maar hiervoor is een goede reden; in de praktijk is gebleken dat Task Engine het beste werkt wanneer de Next Task o.b.v. de gemiddelde taakduur (voor zover bekend) wordt gekozen. Dat betekent meestal dat juist de Slow Lanes voorrang krijgen bij het starten, maar dat is in de praktijk uiteindelijk gunstiger gebleken voor de totale doorlooptijd; de Fast Lane start dan wellicht iets later maar hoeft niet op de Slow Lanes te wachten. Het alternatief is dan om de Fast Lane wel altijd voorrang te geven, de praktijk zal per toepassing moeten leren wat het beste werkt.

*Samenvattend: Bij meerdere actieve Lanes is die met het **laagste** nummer de Fast Lane, de overige Lanes zijn dan impliciet Slow Lanes.*
*N.B.: Dit is echter het default gedrag, middels de Parameter **LowestLaneFastest** kan worden ingesteld dat de Lane met het **hoogste** nummer de Fast Lane is.*

3.2 Doorlooptijd optimalisatie

Taken binnen 1 Batch met gelijke Prio en Rank worden door Task Engine zoveel mogelijk via Jobs kort na elkaar opgestart en lopen dan simultaan. Doordat sommige taken langer lopen dan andere, zullen er gemiddeld aan het begin vaak meer taken tegelijk lopen dan aan het eind.

Wanneer de CPU of het aantal taken de geconfigureerde drempelwaarde bereikt, dan pauzeert het aanmaken van nieuwe Jobs totdat er meer CPU beschikbaar is.

In de praktijk kwam het soms voor dat er pas na verloop van tijd een taak werd aangemaakt met een lange doorlooptijd die dan uiteindelijk nog alleen liep omdat de rest al voltooid was.

Daarom wordt er nu gekeken naar de historie van elke specifieke taak (gebaseerd op de naam van de uitgevoerde Stored Procedure), en worden ze in *aflopende volgorde van gemiddelde doorlooptijd* afgehandeld, ofwel: de langstdurende worden eerst gestart.

Hierdoor is de gemiddeld totale doorlooptijd korter gebleken.

[4] Dit is het Default gedrag, zie ***Samenvattend*** aan het einde van deze alinea.

4 Vereisten

4.1 SQL Server versies

Task Engine draait zelf op minimaal SQL Server 2014, taken op hogere versies kunnen worden uitgevoerd wanneer deze voor die versie syntactisch juist zijn.

Met lagere versies kan Task Engine ook werken, als er een werkende linked server is dan moeten taken met 4-part-names kunnen worden uitgevoerd.

4.2 Automatische configuratie

Er wordt onderhuids gebruik gemaakt van CLR routines en Commandshell, dit wordt door het systeem geconfigureerd:

- CLR wordt enabled
- xp_cmdshell wordt enabled
- voor de Code database wordt trustworthy ON gezet

5 Installatie en configuratie

N.B.: Sysadmin privileges zijn vereist voor degene die de installatie uitvoert

Installeer de 2 databases vanuit de aangeleverde back-up files op de gewenste SQL Server.

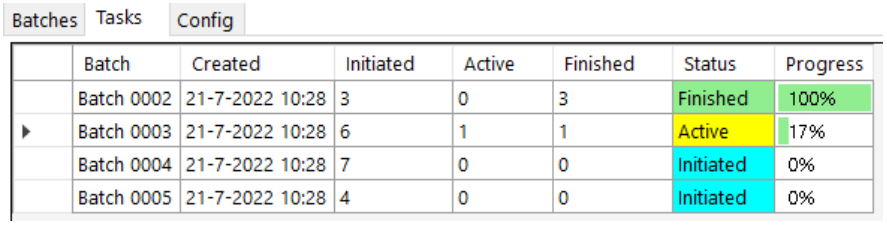
Plaats de files van de Front-end applicatie Task Engine Console op de gewenste plek en start de Console.

Maak connectie met de betreffende SQL Server, het programma detecteert wanneer de configuratie nog niet is uitgevoerd en oppert om dit nu te doen.

6 Gebruik

Task Engine voert batches uit vanuit de gedachte dat een Batch een verzameling Tasks is die gezamenlijk tot een bepaald resultaat moeten leiden.

De getoonde status van een Batch wordt ontleend aan de onderliggende Tasks:



	Batch	Created	Initiated	Active	Finished	Status	Progress
	Batch 0002	21-7-2022 10:28	3	0	3	Finished	100%
▶	Batch 0003	21-7-2022 10:28	6	1	1	Active	17%
	Batch 0004	21-7-2022 10:28	7	0	0	Initiated	0%
	Batch 0005	21-7-2022 10:28	4	0	0	Initiated	0%

- Initiated alle Tasks hebben status Created
- Active ten minste 1 Task is opgepakt voor uitvoering
- Finished alle Tasks zijn beëindigd, al dan niet met succes

De getoonde voortgang (Progress) is puur o.b.v. aantallen Finished Tasks t.o.v. Initiated Tasks en zegt niets over de doorlooptijd.

Bij de verschillende conversies in het verleden zijn groepen SP's gebouwd waarbij deze gegevens in de namen zijn opgenomen, o.b. waarvan een set opdrachten werd gegenereerd.

Voorbeeld; We hebben een schema Process met daarin de SP's die als taken moeten worden uitgevoerd:

Prio	Lane	Rank	SP-naam
1	1	1	Up_Conversie_p1_l1_r01_Extractie_personen
1	1	1	Up_Conversie_p1_l1_r01_Extractie_adressen
1	1	1	Up_Conversie_p1_l1_r01_Extractie_woonplaatsen
2	2	1	Up_Conversie_p2_l2_r99_Extractie_Rapportage
2	1	1	Up_Conversie_p2_l1_r01_Transformatie_woonplaatsen
3	2	1	Up_Conversie_p3_l2_r01_Transformatie_Rapportage
			…

De SP namen bevatten codes voor de sturing van het proces en zijn als volgt opgebouwd:

- Prefix_Batch_Lane_Prio_Rank_Beschrijving
- Tussen deze onderdelen moet 1 underscore staan
- Batch en Beschrijving mogen aanvullende underscores bevatten
- Prio en Lane en Rank bestaan uit 1 letter (respectievelijk P, L en R) en 1 of meer cijfers, voor het proces zijn voorloopnullen niet van belang zoals bij rank is gedaan, maar bedenk wel dat

dit ook bepalend is voor de sortering in Object Explorer, advies is om dit bij meerdere digits *wel* toe te passen.

Bij dit specifieke voorbeeld is het als volgt:

1. Prefix: Up_
2. Batch: Conversie
3. Prio, Lane en Rank
4. Een beschrijvende term

6.1 Autobatch

Task_Engine kan evt. zelf bepalen welke SP's binnen een Batch moeten worden uitgevoerd en op welke manier.

PLAATJE!

Vereiste is wel dat het format wordt gehanteerd zoals in het voorgaande hoofdstuk is beschreven. Een batch wordt dan gespecificeerd door de combinatie Batch, DatabaseNaam, SchemaNaam en Prefix op te geven.

Door een Batchnaam te specificeren kan Task_Engine in de betreffende database een discovery doen van de aanwezige SP's en worden de Prio, Lane en Rank bepaald o.b.v. de namen.

PLAATJE!

Door de SP up_Run_Autobatch (bij voorkeur vanuit een Job) aan te roepen met de Batchnaam als parameter wordt de batch samengesteld middels een discovery en uitgevoerd.

Alternatief is om de betreffende taken zelf aan te laten maken, zie volgende hoofdstuk.

6.2 Zelf taken aanmaken

Als men er om redenen de voorkeur aan geeft om Autobatch niet te gebruiken, is men uiteraard geheel vrij in de naamgeving van de uit te voeren SP's. Men moet er dan wel zelf voor zorgen dat de juiste taken met de juiste parameters worden aangemaakt.

Advies is om in dat geval toch een (zelfbedacht) format te gebruiken want anders is het niet altijd duidelijk wat de onderlinge relaties zijn en volgens welke regels één en ander uitgevoerd moet worden, vooral bij grotere hoeveelheden SP's.

Een taak kan worden aangemaakt door up_Task_Create aan te roepen met de juiste parameters:

PARAMETERS!