Kwaliteit van het testproces wordt verhoogd

Meet performance van de rapportageomgeving

Rudolf van der Heide

Topanalisten noemen de performance van de rapportageomgeving niet als één van de top-succesfactoren voor geslaagde Business Intelligence. Grote leveranciers zien performance nauwelijks als een probleem, veel ontwikkelaars ook niet. Er komen wel steeds meer nieuwe (database) producten die een veel betere performance beloven.

Performance is voor de gewone gebruiker in de praktijk één van de belangrijkste aspecten van de bruikbaarheid van een rapportageomgeving en er bestaat een directe relatie tussen performance en kwaliteit en bruikbaarheid van de rapportageomgeving. Recent is dit door twee bronnen bevestigd. Uit de BI-Survey 2007 van Nigel Pendse komt naar voren dat de performance van de BI-omgeving problematisch is. Meer dan twintig procent van de gebruikers geeft aan problemen te hebben met de performance van rapportages. Consultants en leveranciers vinden het echter een veel minder groot probleem. Uit het Nationaal Data Warehouse Onderzoek 2008 komt naar voren dat 55 procent van de organisaties het gebruik niet meet en nog eens dertig procent geeft aan dat het gebruik wel wordt gemeten, maar niets met de resultaten wordt gedaan.
Er is dus nog een hoop te doen. In dit artikel worden eenvoudig toe te passen methodes besproken voor het meten van het gebruik en de performance van de rapportageomgeving en voor het meten van de doorlooptijd van laadprocessen.

Meting

Een goede meting moet aan een aantal aspecten voldoen. Ten eerste moeten de metingen voor iedereen, zowel eindgebruikers als beheerders, begrijpelijk en herkenbaar zijn. De klassieke ratio's die door DBA's graag worden gebruikt voldoen hier bijvoorbeeld niet aan. Ook gemiddeldes zijn niet erg verhelderend. Als een rapport zogenaamd goed draait omdat het gemiddeld in twintig seconden terugkomt, is de gebruiker niet blij altijd drie minuten te moeten wachten. De gebruiker wil graag direct uit de rapportage kunnen zien dat het rapport ook wel eens langer duurt. De tweede eis volgt direct uit de eerste: de meting moet simpel zijn. Verder is het belangrijk dat de resultaten bruikbaar zijn voor een structurele verbetering van de omgeving en ook dat een verbetering zichtbaar is in de meting.

> Een ontwikkelteam kan vaak vooraf veel meer doen dan een DBA achteraf

De meeste repository's van rapportagetools geven genoeg informatie om de volgende gegevens te verzamelen. Per opvraging in elk geval: de rapportnaam; de versie; de gebruiker en afdeling; de responstijd in seconden; de datum. Extra nuttige data zijn soms ook eenvoudig te krijgen: tabblad of subquery; de geselecteerde tabellen, folder, feitentabel of gebruikte kubus; de gebruikte omgeving (voor een vergelijking tussen test en productie).

Ster: Polisfeit	Aantallen per responstijd					Percentage < gegeven responstijd		
	< 20 Sec	20 sec - 1 minuut	1-5 minuten	> 5 minuten	Totaal	<20 sec	<1 minuut	< 5 minuten
Oude server	138	146	164	86	534	26%	53%	84%
Nieuwe server	43	20	10		73	59%	86%	100%

Afbeelding 1: Effect van migratie naar een nieuwe server.

	Aantal rapporten per draaitijd				
	0-10 sec.	10-20 sec.	20-60 sec	>1 minuut	Totaal
200701	261	99	81	9	450
200702	208	112	68	12	400
200703	201	103	61	15	380
200704	222	96	71	8	397
200705	210	118	62	21	411
200706	285	145	92	16	538
200707	190	86	48	21	345
200708	269	42	10	0	321
200709	409	23	18	0	450
200710	448	40	10	5	503
200711	403	37	23	0	463
200712	318	30	26	0	374

Afbeelding 2: Performance van alle standaardrapporten over de maanden.

Zoals eerder gesteld, zijn gemiddeldes niet zo nuttig. Het aantal rapporten binnen een bepaald aantal seconden is wel heel nuttig. Enkele voorbeelden van rapporten zijn in afbeeldingen 1, 2 en 3 te zien. Deze rapporten zijn gegenereerd met behulp van Excel-draaitabellen. Het is handig om een simpele kubus te maken om de percentages te vergelijken.
Voorbeeld 1. Het effect van de migratie in afbeelding 1 is in één oogopslag zichtbaar. Een verbetering van het percentage rapporten binnen twintig seconden van 26 naar 59 procent.
Voorbeeld 2. Performance van alle standaardrapporten over de maanden. Er waren materialized views geïmplementeerd, zodat de standaardrapporten in principe binnen tien seconden zouden draaien. Uit de tabel in afbeelding 2 blijkt duidelijk een verbetering vanaf augustus 2007. Ook zie je de performance al weer langzaam achteruit gaan vanaf oktober 2007.
Voorbeeld 3. Aantallen rapportages per business unit, zie afbeelding 3. Heel geschikt voor publicatie op het intranet. Met een beetje geluk gaan managers erover praten en krijgen ze te horen wat de reden is dat de ene business unit zo veel succesvoller is.

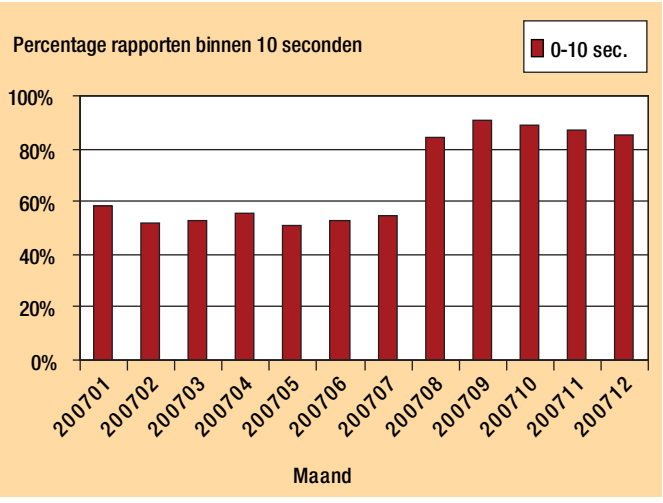
Actie

Uit de rapporten volgen automatisch acties. Enkele voorbeelden hiervan volgen. Als men ziet dat rapporten op de ene ster veel slechter draaien dan op een andere, is het mogelijk specifiek hiernaar te kijken. Misschien is het ontwerp niet goed, misschien is er technisch wat te verbeteren, betere indexering of een materialized view, of misschien wordt de informatie niet goed gebruikt, waardoor de gebruikers 'onhandige' rapporten maken.

Als de performance langzaam achteruit gaat of opeens een dip vertoont, kan men gericht naar oorzaken zoeken. Een praktijkvoorbeeld betreft een index, die te lang werd, waardoor de query de index niet meer gebruikte en de rapportage opeens veel langer duurde. Dit had overigens al eerder opgemerkt kunnen worden, door de gestage achteruitgang in performance.
In een ander voorbeeld produceerde een gebruiker regelmatig rapporten die een uur draaiden. Een kleine beetje uitleg of opleiding kan hier uitkomst bieden.

Enkele waarschuwingen mogen niet ontbreken. Maak het niet te ingewikkeld en exact. Rapportageomgevingen vertonen enorme schommelingen. Een rapport kan voor de ene afdeling veel meer data bevatten dan voor de andere en daarom is ook de responstijd van de rapportages niet altijd vergelijkbaar. Ook zijn er piek- en daluren; schommelingen hiervan zijn duidelijk zichtbaar. Gebruik de rapportages dan ook niet om bijvoorbeeld SLA's te meten. Iedereen zal creatieve manieren vinden om de afgesproken targets te halen in plaats van de problemen op te lossen.

Bij een rapportageomgeving zijn er veel verschillende partijen betrokken: business users, front-end ontwikkelaars, back-end (ETL) ontwikkelaars, DBA's en operationeel beheerders. Alle disciplines hebben hun eigen aandachtsgebieden en opvattingen. Werk samen en zoek oplossingen waar ze thuis horen. Als bijvoorbeeld de business vraag niet realistisch is, moet men het probleem daar oplossen en niet proberen de DBA's een onmogelijke query te laten tunen. Gebruik de rapportages om knelpunten zichtbaar te maken en dus ook projecten op te kunnen starten om ze op te lossen.
Een anekdote met Oracle Discoverer volgt ter illustratie van het laatste punt. De performance van een set standaardrapportages was verbeterd met behulp van materialized views in Oracle en bij alle testen kwamen de rapporten in seconden terug. Toch bleek in de performance-rapportage 'maar' 90 procent binnen tien seconden terug te komen. Wat bleek: na opening van een



Afbeelding 3: Aantallen rapportages per business unit.

		Doorlooptijd in minuten									
Proces	Component	Maand -3	Maand -2	Maand -1	Week -3	Week -2	Week -1	3 runs geleden	2 runs geleden	1 run geleden	laatste run
pro010	Totaal	10	12	13	15	16	18	20	25	30	43
pro020	Totaal	15	15	16	15	16	15	15	15	16	15
pro030	Totaal	4	5	5	5	4	6	5	5	4	6

Afbeelding 4: Doorlooptijd per proces.

standaardrapport dat vervolgens aangepast werd, stond in de repository nog steeds de naam van het standaardrapport. De gebruikers hadden geleerd om even alle producten toe te voegen om de details te controleren: einde gebruik van materialized view en dus een veel langere responstijd. Dat zou een DBA toch echt nooit zelf ontdekt hebben!
Aanbevelingen zijn: stel de rapportage van performance en gebruik aan iedereen beschikbaar, verrassend bijeffect hiervan is vaak dat de klachten verminderen; kijk naar de trends, merk de incidenten vroegtijdig op en onderneem actie; werk samen en kijk naar de rapportageomgeving als geheel, men is samen verantwoordelijk voor de omgeving.

Laadomgeving

Het meten van de performance van de laadomgeving is eigenlijk heel simpel. Meet de doorlooptijd van de processen en de componenten en, als het beschikbaar is, aantallen geladen en geselecteerde records. De data komen uit de repository of kunnen vaak heel gemakkelijk zelf worden gecreëerd. Veel organisaties gebruiken al een *wrapper procedure* om ETL-componenten aan te roepen of een aparte start en stop procedure. Hier is eenvoudig een logging in te bouwen.
Vragen die een beheerafdeling simpel wil beantwoorden zijn:

- Welke laadprocessen duren lang? Tien minuten is in een middelgrote omgeving al erg lang. Met redelijke hardware zijn in tien minuten al miljoenen records te laden. Lange processen kunnen na verloop van tijd opeens niet meer blijken te werken;
- Welke laadprocessen groeien gedurende de tijd? Dit is een indicatie voor mogelijke performance-problemen, maar kunnen ook op inhoudelijke problemen wijzen. Vooral plotselinge sprongen zijn heel verdacht;
- Welke component is de hoofdoorzaak van een traag proces? Dit maakt de analyse veel gerichter en dus het oplossen een stuk efficiënter;
- Als het proces 's nachts stuk loopt en de volgende dag pas weer wordt opgestart, wanneer is het laadproces dan afgerond? Vertel dit ook aan de business, dan kan men samen besluiten of dit acceptabel is.

Ik bouw hiervoor graag een kubus en rapporteer dan doorlooptijd per proces of job en per processtap, zoals te zien in afbeelding 4. Bij proces pro010 is duidelijk een probleem te zien.
De andere twee processen vertonen normale schommelingen.
Een *drill down* genereert de gegevens te zien in afbeelding 5. Ogenblikkelijk is te zien waar het probleem ligt: mapping 1 en create index en je kunt gericht gaan zoeken. Een alternatief rapport is: Proces, component, gemiddelde laatste 3 maanden, gemiddelde laatste 3 weken, gemiddeld laatste 10 keer, 3,2,1 runs geleden, laatste run.

		Doorlooptijd in minuten									
Proces	Component	Maand -3	Maand -2	Maand -1	Week -3	Week -2	Week -1	3 runs geleden	2 runs geleden	1 run geleden	laatste run
pro010	Totaal	10	12	13	15	16	18	20	25	30	43
pro010	drop index	0	0	0	0	0	0	0	0	0	0
pro010	mapping 1	3	4	5	5	6	7	8	10	12	20
pro010	mapping 2	4	4	4	4	4	4	4	5	5	5
pro010	create index	2	3	3	4	4	5	6	8	10	15
pro010	analyze	1	1	1	2	2	2	2	2	3	3

Afbeelding 5: Drill down.

Aanbevelingen hierbij zijn: kijk naar processen als geheel en niet alleen naar de afzonderlijke componenten; wees er op tijd bij. Groeiprocessen leiden altijd tot incidenten als men niets doet en de incidenten komen altijd op een pijnlijk moment.

Nog een voorbeeld bij de eerste aanbeveling. Het gaat om het proces als geheel. Het komt vaak voor dat een aantal stappen in een laadproces 'netjes' wordt ontwikkeld en geoptimaliseerd, maar dat daardoor volgende stappen opeens veel langer duren. Ontwerp dus laadprocessen als een geheel en de verschillende stappen als subprocessen.
Bijvoorbeeld, je krijgt twintig miljoen records binnen. Vijf miljoen krijgen een update vanwege een bepaalde business rule, vijf miljoen vanwege een andere business rule, tien miljoen blijven ongewijzigd. Vaak is dan te zien:
- Stap 1: laad 20 miljoen records;
- Stap 2: update eerste 5 miljoen;
- Stap 3: update volgende 5 miljoen.

> Men zal manieren vinden om targets te halen in plaats van problemen op te lossen

Dit lijkt logisch en het volgt de beschrijving van de aanpak. Dat is waarom het zo vaak gaat. Het is echter absoluut de verkeerde aanpak. In een datawarehouse moet de update absoluut worden vermeden, tenzij er geen andere oplossing is. Die is er hier wel:
- Stap 1: laad 20 miljoen records in tijdelijke tabel;
- Stap 2: selecteer de 10 miljoen ongewijzigde records en laad ze naar de target;
- Stap 3: selecteer de eerste set van 5 miljoen records, pas de business rule toe en laad ze naar de target;
- Stap 4: selecteer de tweede set van 5 miljoen records, pas de business rule toe en laad ze naar de target.

Doordat het een stap extra is, hebben veel ontwikkelaars het gevoel dat het proces dus langzamer zal zijn. Insert's en select's zijn echter zo veel sneller dan een update, dat het totale proces veel sneller is.

De business case voor een goede performance

Als technisch architect wordt ik vaak wat meewarig aangekeken als ik performance-aspecten ter sprake breng. De constatering van Nigel Pendse dat consultants en leveranciers performance-problemen aanzienlijk minder belangrijk achten dan de gebruikers, is ook juist voor wat betreft ontwikkelaars en projectorganisaties. Zij zijn niet degenen die straks eindeloos op de rapporten hoeven te wachten en vastlopende processen moeten oplossen. Performance moet vanaf het begin een aandachtspunt zijn, het ontwikkelteam kan vaak vooraf veel meer doen dan een DBA achteraf. Een goede performance creëert indirect waarde voor de business. Enkele situaties kunnen dit illustreren. Stel, je maakt een rapportage aan en je vermoedt dat er iets niet helemaal in orde is. Ga je verder kijken als: het tien minuten duurde voor je rapport klaar was en een extra detail-rapport weer tien minuten gaat duren; je rapport in tien seconden klaar was en extra details hooguit twintig seconden kosten? Zou dit nog verschil maken voor hoe vaak je van een rapport gebruik maakt?
Hoeveel verschillende testgevallen voer je uit als ieder laadproces twee uur duurt en hoeveel als een laadproces tien minuten duurt? (Dit argument geldt natuurlijk niet als de organisatie een formeel testproces heeft dat ook echt werkt.)
Hoeveel tijd gaat er in het ontwikkelteam verloren doordat iemand een 'hopeloze' query draait op de ontwikkelomgeving, want "het is toch maar de ontwikkelomgeving"?

De baten zijn:
- 1. Betere datakwaliteit. Rapporten die snel op het scherm staan, worden meer gebruikt. Zeker in deze tijd nu we verwend zijn met Google willen we snelle resultaten. Snelle resultaten nodigen uit tot nieuwsgierigheid. Dit is heel belangrijk want de gebruiker van de rapporten is vaak de enige die ontdekt dat er iets mis is met datakwaliteit;
- 2. Meer gebruik van informatie. Larry English wees er al op dat informatie het enige productiemiddel is dat meer waard wordt, naarmate het meer wordt gebruikt. Hoe sneller de rapportage op te vragen is, hoe meer de informatie wordt gebruikt. Ga je tien minuten wachten op een rapport ter bevestiging van iets wat je eigenlijk al wel zeker weet? Met tien seconden is het een kleine moeite;
- 3. Lagere ontwikkelkosten. Een ontwikkelaar draait de laadprocessen als test erg vaak. Wat gaat hij doen in de tussentijd? Als het laadproces vijf minuten duurt is het een rustige onderbreking van de dag, als het een uur is, gaat hij wat anders in de tussentijd doen, met verlies van concentratie en kwaliteit tot gevolg. Ook zit men met query's met slechte performance collega's vaak in de weg, die misschien hun efficiënte proces gaan aanpassen omdat ze denken dat zij het probleem zijn;
- 4. Beter testen. Hoe sneller het proces, hoe gemakkelijker wordt het om meer testgevallen uit te voeren.

Conclusie

Met een betere performance zijn rapporten beter en onderzoek je mogelijke afwijkingen vaker. De praktijk leert dat op deze manier de meeste fouten en afwijkingen in informatie worden gevonden. Vanaf het begin rekening houden met een goede performance levert lagere ontwikkelkosten op en verhoogt de kwaliteit van het testproces.

Rudolf van der Heide (rudolf@xs4all.nl) is freelance specialist Datawarehousing & Business Intelligence.