Opdracht 5: Materialized Views

Inleiding

Deze opdracht kan worden uitgevoerd door gebruik te maken van een eigen Oracle Enterprise Edition OF door een VirtualBox (VB) te gebruiken:

<http://www.oracle.com/technetwork/database/enterprise-edition/databaseappdev-vm-161299.html>

Wellicht heb je voor het downloaden van de tool of de appliance een OTN-account nodig, maar die kun je gratis aanmaken. Voor het ‘bouwen’ van de VBox heb je, naast de download (±4,8 GB) ook nog 12GB vrije ruimte op de HD nodig en de geïnstalleerde versie van Virtual Box (< 150Mb).

Volg voor het installeren de stappen zoals op de pagina beschreven. Zie ook het kopje “Requirements” op de website voor een tip over performance als je gebruik maakt van Windows.

Environment setup

1. Log in op de virtual machine (username en password zijn: oracle).
2. Start de database (deze mogelijkheid staat in het menu van de virtuele machine).
3. Voer onderstaand script uit op de database vanuit Oracle SQL Developer (staat op het bureaublad in de virtual machine):

create user fontys identified by fontys default tablespace users temporary tablespace temp

/

grant connect, resource, create materialized view to fontys

/

create or replace synonym fontys.emp for SCOTT.emp

/

grant all on SCOTT.emp to fontys

/

create or replace synonym fontys.dept for SCOTT.dept

/

grant all on SCOTT.dept to fontys

/

Opdracht 1:

1. Maak een materialized view gebaseerd op de volgende query:  
   SELECT dept.deptno

,dept.dname

,COUNT(\*) number\_of\_employees

,SUM(sal) total\_salary\_of\_employees

FROM emp

,dept+

WHERE emp.deptno = dept.deptno

GROUP BY dept.deptno

,dept.dname

CREATE MATERIALIZED VIEW emp\_dept

BUILD IMMEDIATE

REFRESH ON DEMAND

ENABLE QUERY REWRITE

AS

SELECT dept.DEPARTMENT\_ID

,dept.DEPARTMENT\_NAME

,COUNT(\*) number\_of\_employees

,SUM(salary) total\_salary\_of\_employees

FROM emp

,dept

WHERE emp.DEPARTMENT\_ID = dept.DEPARTMENT\_ID

GROUP BY dept.DEPARTMENT\_ID

,dept.DEPARTMENT\_NAME

1. Laat zien dat de materialized view sneller is dan de query.  
   (☞Tip: Use the explain plan)

Query uitvoeren met en zonder materialized view

Met materialized view -> 0.003 sec

Zonder materialized view -> 0.57 sec

1. Hoe moet je de materialized view veranderen zodat hij gebruikt wordt ook al roep je hem in de query niet expliciet aan?

Door het gebruiken van Query Rewrite in een materialized view wordt deze automatisch gebruikt wanneer deze query wordt uitgevoerd.

1. Hoe kun je ervoor zorgen dat de materialized view altijd up to date is?

Bij het aanmaken van een materialized view kan een REFRESH keyword worden meegegeven, bijvoorbeeld REFRESH ON COMMIT. Hierdoor is de query altijd up to date als de materialized view wordt gebruikt.

Opdracht 2:

Je gaat proberen de volgende business rule op technische verschillende manieren te implementeren (we garanderen niet dat dit op elke van de vier manieren kan):

**☞De som van de salarissen in een afdeling mag niet meer zijn dan 20000☜**

1. Probeer “Check Constraints” te gebruiken. Verklaar waarom dat wel of niet werkt.

Dit is niet mogelijk met check constraints, er kan namelijk geen constraint worden gezet op de som van een kolom.

1. Probeer unieke indexen te gebruiken. Verklaar waarom dat wel of niet werkt.

Dit kan ook niet worden gedaan met unieke indexen. Indexen kunnen niet worden gebruikt om een constraint toe te passen.

1. Probeer triggers te gebruiken. Verklaar waarom dat wel of niet werkt.

create or replace trigger check\_salary

before insert or update

on emp

for each row

declare

total\_salary number;

cursor salary\_cursor is

select sum(e.salary)

from emp e;

begin

open salary\_cursor;

fetch salary\_cursor into total\_salary;

if (total\_salary > 20000) then

raise\_application\_error(-19999, 'Salaris mag niet hoger dan 20000 zijn');

end if;

end;

dit werkt wel. Als door een update of insert het salaris te hoog wordt, dan wordt de statement niet uitgevoerd. Bij sommige departments was het salaris al hoger dan 20000. Hier wordt nu ook niks mee gedaan.

1. Probeer “Materialized Views” te gebruiken. Verklaar waarom dat wel of niet werkt.

create materialized view log on emp

with sequence, primary key, rowid

(department\_id, salary) including new values

create materialized view emp\_deptsalsum\_mv

refresh fast on commit

as

select department\_id,

sum(salary) sumsal,

count(\*),

count(salary) head\_count

from emp

group by department\_id

alter table emp\_deptsalsum\_mv

add constraint check\_dept\_salsum

check ( sumsal < 20000)

deferrable

In combinatie met een materialized view kan een check constraint worden toegepast, omdat de som van de salarissen nu als een kolom wordt toegevoegd.

Opleveren

Antwoorden op bovenstaande vragen en de SQL-scripts die je hebt gemaakt om tot de antwoorden te komen. De antwoorden op de vragen moeten kunnen worden toegelicht.