# Practicumopdracht Query-optimalisatie

In deze opdracht onderzoek je een aantal SQL-query’s welke veelgemaakte fouten bevatten. Het is in veel gevallen de bedoeling dat je de performanceproblemen verklaart en/of oplossingen zoekt om de performance te verbeteren.

Gebruik het [meegeleverde create script](https://portal.fhict.nl/IS/S6/Lesmateriaal/DBO-SEI%20Opdracht%206%20-%20Create%20script.txt) om onderstaande opdrachten te kunnen uitvoeren.

## Deel 1 – Parsing time

a) Voer onderstaande query 5 maal uit. Noteer de benodigde tijd in de kolom onder de query.

SELECT \* FROM orders WHERE order\_id\_char = 1200;

|  |  |
| --- | --- |
| Meting | Tijd (sec) |
| 1 | 0,0315 |
| 2 | 0,0004 |
| 3 | 0,0005 |
| 4 | 0,0005 |
| 5 | 0,0005 |

b) Doe hetzelfde voor de volgende query:

SELECT \* FROM orders WHERE order\_id\_char = 10;

|  |  |
| --- | --- |
| Meting | Tijd (sec) |
| 1 | 0.0563 |
| 2 | 0,0004 |
| 3 | 0,0005 |
| 4 | 0,0005 |
| 5 | 0,0005 |

c) Wat is je conclusie? De eerste keer duurt het langst, daarna is de query snel.

Primary key is standaard een BTREE

## Deel 2 – Query-optimalisatie

1. Maak de tabellen “products” en “costs” aan in de database met in ieder geval de kolommen die in onderstaande query genoemd worden. Vul de tabellen met willekeurige testdata (bijvoorbeeld met behulp van je testdatagenerator).
2. Voer onderstaande query uit en noteer de benodigde tijd om de query uit te voeren in de tabel hieronder.
3. Verbeter de performance van de query en noteer ook van deze query de benodigde tijd om de query uit te voeren.

|  |  |  |
| --- | --- | --- |
| Query | Tijd (sec) | Cost |
| SELECT COUNT(\*)  FROM products as p  WHERE prod\_list\_price < 1.15 \* (SELECT avg(unit\_cost)  FROM costs as c  WHERE c.prod\_id = p.prod\_id); | 0.0673 | 17 pesos |
| Geoptimaliseerde query: \*testen (join?)  SELECT COUNT(\*)  FROM products as p  WHERE prod\_list\_price < (SELECT 1.15 \* avg(unit\_cost)  FROM costs  WHERE prod\_id = p.prod\_id); | 0.0070 |  |

1. Verklaar aan de hand van het execution plan van de twee query’s de verschillen in benodigde tijd. 1x duurt het lang, hierna staat het in het egeheugen waardoor het sneller gaat

Werkelijk waar geen idee… een typo doet wonderen.

## Deel 3 – Query optimalisatie

1. Voer onderstaande query uit en noteer de benodigde tijd om de query uit te voeren.
2. Verbeter de performance van de query en noteer ook van deze query de benodigde tijd om hem uit te voeren.

|  |  |  |
| --- | --- | --- |
| Query | Tijd (sec) | Cost |
| SELECT count(\*)  FROM job\_history jh, employees e  WHERE  substr(to\_char(e.employee\_id),1)=substr(to\_char(jh.employee\_id),1);  mysql:  SELECT count(\*)  FROM job\_history as jh, employees as e  WHERE  substring(e.employee\_id,1)=substring(jh.employee\_id,1); | 5 minuten + timeout | To fucking much |
| Geoptimaliseerde query:  SELECT count(\*)  FROM job\_history as jh, employees as e  WHERE  e.employee\_id=jh..employee\_id; | 5 minuten + timout |  |

c) Verklaar aan de hand van het execution plan van de twee queries de verschillen in benodigde tijd.

## Deel 4 – Query optimalisatie

1. Voer onderstaande query’s uit en noteer de benodigde tijd.

|  |  |  |
| --- | --- | --- |
| Query | Tijd (sec) | Cost |
| select count(\*)  from (select o.name from old as o union select n.name from new as n) as abc; | 1.1653 |  |
| select count(\*)  from (select name from old union all select name from new); | 0.6193 |  |

1. Verklaar aan de hand van het execution plan van de twee queries de verschillen in tijd.

De lataste voert ook nog een sort uit en de andere is distinct

## Deel 5 – Query’s combineren

1. Voer onderstaande query’s uit en noteer per query de benodigde tijd om hem uit te voeren. Tel de tijden op en noteer het resultaat.

|  |  |  |
| --- | --- | --- |
| Query | Tijd (sec) | Cost |
| SELECT COUNT(\*)  FROM myemp  WHERE salary < 2000; | 2.34850450 |  |
| SELECT COUNT (\*)  FROM myemp  WHERE salary BETWEEN 2000 AND 4000; | 0.43521950 |  |
| SELECT COUNT (\*)  FROM myemp  WHERE salary>4000; | 0.37878850 |  |
| Totaal: | Ongeveer 4 seconden |  |

b) Maak een query die de drie gegeven query’s combineert en dus minder vaak door dezelfde rijen hoeft te lopen. Noteer ook van deze query de benodigde tijd om hem uit te voeren. Tip: maak gebruik van het CASE WHEN statement.

|  |  |  |
| --- | --- | --- |
| Geoptimaliseerde query | Tijd (sec) | Cost |
| CREATE PROCEDURE `myempsaltary`()  BEGIN  DECLARE salt INT;  DECLARE lowest INT(10) DEFAULT 0;  DECLARE mid INT(10) DEFAULT 0;  DECLARE highest INT(10) DEFAULT 0;  DECLARE fin INT default 0;  DECLARE curs CURSOR FOR SELECT saltary FROM myemp;  DECLARE CONTINUE HANDLER FOR NOT FOUND SET fin =1;  OPEN curs;  get\_salt: LOOP  FETCH curs INTO salt;  if fin =1 THEN leave get\_salt; END IF;  CASE  WHEN salt<2000 THEN SET lowest = lowest +1;  WHEN salt BETWEEN 2000 AND 4000 THEN SET mid = mid +1;  WHEN salt >4000 THEN SET highest = highest +1;  END CASE;  END LOOP get\_salt;  CLOSE curs;  select lowest, mid, highest;  END | 0.3352 |  |

1. Vergelijk de tijd voor het uitvoeren van de losse query’s opgeteld met de tijd voor de zelfgemaakte query en verklaar het verschil aan de hand van de execution plans.

Alles wordt in een loop gehandeld, dit is sneller dan 3x dezelfde routine te runnen

## Deel 6 – Conversies reduceren

1. Voer onderstaande query uit en noteer de benodigde tijd om hem uit te voeren.
2. Bekijk in de database het datatype van de kolom order\_id\_char en vergelijk dit met de meegegeven waarde in de query.
3. Verbeter de performance van de query en noteer ook van deze query de benodigde tijd om hem uit te voeren.

|  |  |  |
| --- | --- | --- |
| Query | Tijd (sec) | Cost |
| SELECT \* FROM orders WHERE order\_id\_char = 50; | 0.4348 |  |
| Geoptimaliseerde query:  SELECT \* FROM orders WHERE order\_id\_char = ‘50’;  Extra variant: SELECT \* FROM orders limit 49, 1 | 0.0345 |  |

1. Verklaar aan de hand van het execution plan van de twee query’s de verschillen in tijd.

Veschil tussen string en int

1. Heeft de INDEX op de tabel invloed op dit verschil? Zo ja, waarom?

Ja, de index zorgt voor een vaste volgoorde… dit maakt het mogelijk om resultaten over te slaan als je op id zoekt. Btree (binairy search)

## Deel 7 – Optionele extra opdracht om hoger te kunnen scoren voor dit onderdeel

Maak drie wezenlijk (ook van bovenstaande query’s) verschillende query’s die niet optimaal functioneren. Licht toe wat verbeterd kan worden en geef de verbeterde versies.

|  |  |  |
| --- | --- | --- |
|  | Tijd (sec) | Cost |
| Query 1: |  |  |
| Query 1 (verbeterd):  Toelichting: |  |  |
| Query 2: |  |  |
| Query 2 (verbeterd):  Toelichting: |  |  |
| Query 3: |  |  |
| Query 3 (verbeterd):  Toelichting: |  |  |

## Opleveren

Je kunt jullie ingevulde versie van dit document tonen en toelichten.

|  |  |  |  |
| --- | --- | --- | --- |
|  | 4 | 7 | 10 |
| Opdracht 4: Query-optimalisatie-opdracht (tweetal) | Alleen metingen uitgevoerd met matige onderbouwing. | Metingen uitgevoerd en onderbouwde antwoorden op de normale vragen. | Metingen uitgevoerd en onderbouwde antwoorden op de normale en optionele vragen. |