Ausführungsplanoptimierung in PostgreSQL

Belegarbeit

eingereicht am Fachbereich

Informatik

der Hochschule Zittau/Görlitz (HAW)

als Prüfungsleistung im Fach

Fortgeschrittene Datenbank-Konzepte 2

vorgelegt von:

Christof Ochmann (35989)

Ingo Körner (40586)

Görlitz, 9. Juli 2012

Betreuer: Prof. ten Hagen

Abstract

Inhaltsverzeichnis

Abbildungsverzeichnis

Listings

Abkürzungsverzeichnis

API	Application Programming Interface
ACID	Atomicity, Consistency, Isolation, Durability
BLOB	Binary Large Object
DBMS	Database management system
ERD	Entity-Relationship Diagram
IDE	Integrated Development Environment
JDK	Java Development Kit
MyISAM	My Indexed Sequential Access Method
OLAP	Online Analytical Processing
OLTP	Online Transaction Processing
PHP	Hypertext Preprocessor
SQL	Structured Query Language
WLAN	Wireless Local Area Network

Literaturverzeichnis

[1] Martin, Robert C. (2008): Clean Code: A Handbook of Agile Software Craftsmanship. Prentice Hall International

[2] Freeman, Eric (2007): Entwurfsmuster von Kopf bis Fuß. O'REILLY

[3] `http://www.easymock.org/EasyMock3_0_Documentation.html Abruf: 21.12.2011`

[4] `http://dev.mysql.com Abruf (11.01.2012)`

[5] Däßler, Rolf (2011): MySQL. bhv

[6] Schwartz, Baron. (2008): High Performance MySQL. O'REILLY

[7] `http://code.google.com/p/google-guice Abruf: 11.01.2012`

1 Theorie

1.1 Einleitung

1.2 Aufgabenstellung

2 Datengenerator

A Codebeispiele

Listing A.1: Testabfrage 1

```
1 Select adbc.Produkt.Name, Count(*)
2 From adbc.Produkt, adbc.Warenkorb_has_Produkt
3 where adbc.Produkt.PRODUKT_ID = adbc.Warenkorb_has_Produkt.
      Produkt_PRODUKT_ID
4 Group by adbc.Produkt.Name;
```

Listing A.2: Testabfrage 2

```
1 SELECT adbc.kunde.Name, SUM(adbc.produkt.Preis)
2 FROM adbc.kunde,adbc.Produkt, adbc.Warenkorb, adbc.
      Warenkorb_has_Produkt
3 WHERE produkt.PRODUKT_ID = warenkorb_has_produkt.
      Produkt_PRODUKT_ID
4       AND warenkorb_has_produkt.Warenkorb_WARENKORB_ID =
            warenkorb.WARENKORB_ID
5       AND warenkorb.Kunde_KUNDE_ID = kunde.KUNDE_ID
6       AND (warenkorb_has_produkt.Datum BETWEEN '2011-01-01'
            AND '2011-03-01')
7 Group by adbc.kunde.Name;
```

Listing A.3: Testabfrage 3

```
1 SELECT Count(DISTINCT(adbc.kunde.KUNDE_ID))
2 FROM adbc.Kunde, adbc.Warenkorb, adbc.warenkorb_has_produkt
3 WHERE warenkorb.Kunde_KUNDE_ID = kunde.KUNDE_ID
4 AND warenkorb.WARENKORB_ID = warenkorb_has_produkt.
      Warenkorb_WARENKORB_ID
5       AND warenkorb_has_produkt.Datum = '2011-01-01';
```

Listing A.4: Testabfrage 4

```
1  SELECT adbc.kunde.Name, SUM(adbc.produkt.Preis)
2  FROM adbc.kunde,adbc.Produkt, adbc.Warenkorb, adbc.
       Warenkorb_has_Produkt
3  WHERE produkt.PRODUKT_ID = warenkorb_has_produkt.
       Produkt_PRODUKT_ID
4        AND warenkorb_has_produkt.Warenkorb_WARENKORB_ID =
             warenkorb.WARENKORB_ID
5        AND warenkorb.Kunde_KUNDE_ID = kunde.KUNDE_ID
6        AND ((warenkorb_has_produkt.Datum = '2011-01-01')
7        OR    (warenkorb_has_produkt.Datum = '2011-01-05')
8        OR    (warenkorb_has_produkt.Datum = '2011-01-09')
9        OR    (warenkorb_has_produkt.Datum = '2011-02-02')
10       OR    (warenkorb_has_produkt.Datum = '2011-02-06')
11       OR    (warenkorb_has_produkt.Datum = '2011-02-10')
12       OR    (warenkorb_has_produkt.Datum = '2011-03-10')
13       OR    (warenkorb_has_produkt.Datum = '2011-03-14')
14       OR    (warenkorb_has_produkt.Datum = '2011-03-18'))
15 Group by adbc.kunde.Name;
```

Listing A.5: Tabellenerzeugung mit Hash-Partitioning

```
1  SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
2  SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS,
       FOREIGN_KEY_CHECKS=0;
3  SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='TRADITIONAL';
4
5  CREATE SCHEMA IF NOT EXISTS `ADBC` DEFAULT CHARACTER SET
       utf8 COLLATE utf8_general_ci ;
6
7  USE `ADBC`;
8
9  CREATE  TABLE IF NOT EXISTS `ADBC`.`Kunde` (
10   `KUNDE_ID` INT(11) NULL DEFAULT NULL ,
11   `Name` VARCHAR(45) NULL DEFAULT NULL ,
12   `Kundennummer` VARCHAR(45) NULL DEFAULT NULL )
13 ENGINE = InnoDB
```

```
14 DEFAULT CHARACTER SET = utf8
15 COLLATE = utf8_general_ci;
16
17 CREATE  TABLE IF NOT EXISTS `ADBC`.`Warenkorb` (
18   `WARENKORB_ID` INT(11) NULL DEFAULT NULL ,
19   `Kunde_KUNDE_ID` INT(11) NULL DEFAULT NULL )
20 ENGINE = InnoDB
21 DEFAULT CHARACTER SET = utf8
22 COLLATE = utf8_general_ci;
23
24 CREATE  TABLE IF NOT EXISTS `ADBC`.`Produkt` (
25   `PRODUKT_ID` INT(11) NULL DEFAULT NULL ,
26   `Name` VARCHAR(45) NULL DEFAULT NULL ,
27   `Preis` INT(11) NULL DEFAULT NULL )
28 ENGINE = InnoDB
29 DEFAULT CHARACTER SET = utf8
30 COLLATE = utf8_general_ci;
31
32 CREATE  TABLE IF NOT EXISTS `ADBC`.`Warenkorb_has_Produkt`
     (
33   `Warenkorb_WARENKORB_ID` INT(11) NULL DEFAULT NULL ,
34   `Produkt_PRODUKT_ID` INT(11) NULL DEFAULT NULL ,
35   `WARENKORB_HAS_PRODUKT_ID` INT(11) NULL DEFAULT NULL ,
36   `Datum` DATE NULL DEFAULT NULL )
37 ENGINE = InnoDB
38 DEFAULT CHARACTER SET = utf8
39 COLLATE = utf8_general_ci
40
41     PARTITION BY HASH( MONTH(Datum) )
42     PARTITIONS 12;
43
44 ;
45
46
47 SET SQL_MODE=@OLD_SQL_MODE;
48 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
```

```
49 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

Listing A.6: Tabellenerzeugung mit List-Partitioning

```
 1 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
 2 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS,
     FOREIGN_KEY_CHECKS=0;
 3 SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='TRADITIONAL';
 4
 5 CREATE SCHEMA IF NOT EXISTS `ADBC` DEFAULT CHARACTER SET
     utf8 COLLATE utf8_general_ci ;
 6
 7 USE `ADBC`;
 8
 9 CREATE TABLE IF NOT EXISTS `ADBC`.`Kunde` (
10   `KUNDE_ID` INT(11) NULL DEFAULT NULL ,
11   `Name` VARCHAR(45) NULL DEFAULT NULL ,
12   `Kundennummer` VARCHAR(45) NULL DEFAULT NULL )
13 ENGINE = InnoDB
14 DEFAULT CHARACTER SET = utf8
15 COLLATE = utf8_general_ci;
16
17 CREATE TABLE IF NOT EXISTS `ADBC`.`Warenkorb` (
18   `WARENKORB_ID` INT(11) NULL DEFAULT NULL ,
19   `Kunde_KUNDE_ID` INT(11) NULL DEFAULT NULL )
20 ENGINE = InnoDB
21 DEFAULT CHARACTER SET = utf8
22 COLLATE = utf8_general_ci;
23
24 CREATE TABLE IF NOT EXISTS `ADBC`.`Produkt` (
25   `PRODUKT_ID` INT(11) NULL DEFAULT NULL ,
26   `Name` VARCHAR(45) NULL DEFAULT NULL ,
27   `Preis` INT(11) NULL DEFAULT NULL )
28 ENGINE = InnoDB
29 DEFAULT CHARACTER SET = utf8
30 COLLATE = utf8_general_ci;
31
```

```
32 CREATE  TABLE IF NOT EXISTS `ADBC`.`Warenkorb_has_Produkt`
       (
33   `Warenkorb_WARENKORB_ID` INT(11) NULL DEFAULT NULL ,
34   `Produkt_PRODUKT_ID` INT(11) NULL DEFAULT NULL ,
35   `WARENKORB_HAS_PRODUKT_ID` INT(11) NULL DEFAULT NULL ,
36   `Datum` DATE NULL DEFAULT NULL )
37
38 ENGINE = InnoDB
39 DEFAULT CHARACTER SET = utf8
40 COLLATE = utf8_general_ci
41
42 PARTITION BY LIST(MONTH(Datum)) (
43     PARTITION quartal1 VALUES IN (1,2,3),
44     PARTITION quartal2 VALUES IN (4,5,6),
45     PARTITION quartal3 VALUES IN (7,8,9),
46     PARTITION quartal4 VALUES IN (10,11,12)
47 );
48
49
50 SET SQL_MODE=@OLD_SQL_MODE;
51 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
52 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

Listing A.7: Tabellenerzeugung mit Range-Partitioning

```
1  SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
2  SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS,
       FOREIGN_KEY_CHECKS=0;
3  SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='TRADITIONAL';
4
5  CREATE SCHEMA IF NOT EXISTS `ADBC` DEFAULT CHARACTER SET
       utf8 COLLATE utf8_general_ci ;
6
7  USE `ADBC`;
8
9  CREATE  TABLE IF NOT EXISTS `ADBC`.`Kunde` (
10   `KUNDE_ID` INT(11) NULL DEFAULT NULL ,
```

```
11    `Name` VARCHAR(45) NULL DEFAULT NULL ,
12    `Kundennummer` VARCHAR(45) NULL DEFAULT NULL )
13  ENGINE = InnoDB
14  DEFAULT CHARACTER SET = utf8
15  COLLATE = utf8_general_ci;
16
17  CREATE  TABLE IF NOT EXISTS `ADBC`.`Warenkorb` (
18    `WARENKORB_ID` INT(11) NULL DEFAULT NULL ,
19    `Kunde_KUNDE_ID` INT(11) NULL DEFAULT NULL )
20  ENGINE = InnoDB
21  DEFAULT CHARACTER SET = utf8
22  COLLATE = utf8_general_ci;
23
24  CREATE  TABLE IF NOT EXISTS `ADBC`.`Produkt` (
25    `PRODUKT_ID` INT(11) NULL DEFAULT NULL ,
26    `Name` VARCHAR(45) NULL DEFAULT NULL ,
27    `Preis` INT(11) NULL DEFAULT NULL )
28  ENGINE = InnoDB
29  DEFAULT CHARACTER SET = utf8
30  COLLATE = utf8_general_ci;
31
32  CREATE  TABLE IF NOT EXISTS `ADBC`.`Warenkorb_has_Produkt`
      (
33    `Warenkorb_WARENKORB_ID` INT(11) NULL DEFAULT NULL ,
34    `Produkt_PRODUKT_ID` INT(11) NULL DEFAULT NULL ,
35    `WARENKORB_HAS_PRODUKT_ID` INT(11) NULL DEFAULT NULL ,
36    `Datum` DATE NULL DEFAULT NULL )
37  ENGINE = InnoDB
38  DEFAULT CHARACTER SET = utf8
39  COLLATE = utf8_general_ci
40
41  PARTITION BY RANGE COLUMNS(Datum) (
42      PARTITION quartal1 VALUES LESS THAN ('2011-04-01'),
43      PARTITION quartal2 VALUES LESS THAN ('2011-07-01'),
44      PARTITION quartal3 VALUES LESS THAN ('2011-10-01'),
45      PARTITION quartal4 VALUES LESS THAN MAXVALUE
```

```
46 );
47
48 SET SQL_MODE=@OLD_SQL_MODE;
49 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
50 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

Listing A.8: Tabellenerzeugung mit Sub-Partitioning

```
1  SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
2  SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS,
       FOREIGN_KEY_CHECKS=0;
3  SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='TRADITIONAL';
4
5  CREATE SCHEMA IF NOT EXISTS `ADBC` DEFAULT CHARACTER SET
       utf8 COLLATE utf8_general_ci ;
6
7  USE `ADBC`;
8
9  CREATE TABLE IF NOT EXISTS `ADBC`.`Kunde` (
10   `KUNDE_ID` INT(11) NULL DEFAULT NULL ,
11   `Name` VARCHAR(45) NULL DEFAULT NULL ,
12   `Kundennummer` VARCHAR(45) NULL DEFAULT NULL )
13 ENGINE = InnoDB
14 DEFAULT CHARACTER SET = utf8
15 COLLATE = utf8_general_ci;
16
17 CREATE TABLE IF NOT EXISTS `ADBC`.`Warenkorb` (
18   `WARENKORB_ID` INT(11) NULL DEFAULT NULL ,
19   `Kunde_KUNDE_ID` INT(11) NULL DEFAULT NULL )
20 ENGINE = InnoDB
21 DEFAULT CHARACTER SET = utf8
22 COLLATE = utf8_general_ci;
23
24 CREATE TABLE IF NOT EXISTS `ADBC`.`Produkt` (
25   `PRODUKT_ID` INT(11) NULL DEFAULT NULL ,
26   `Name` VARCHAR(45) NULL DEFAULT NULL ,
27   `Preis` INT(11) NULL DEFAULT NULL )
```

```
28 ENGINE = InnoDB
29 DEFAULT CHARACTER SET = utf8
30 COLLATE = utf8_general_ci;
31
32 CREATE  TABLE IF NOT EXISTS `ADBC`.`Warenkorb_has_Produkt`
       (
33   `Warenkorb_WARENKORB_ID` INT(11) NULL DEFAULT NULL ,
34   `Produkt_PRODUKT_ID` INT(11) NULL DEFAULT NULL ,
35   `WARENKORB_HAS_PRODUKT_ID` INT(11) NULL DEFAULT NULL ,
36   `Datum` DATE NULL DEFAULT NULL )
37 ENGINE = InnoDB
38 DEFAULT CHARACTER SET = utf8
39 COLLATE = utf8_general_ci
40
41     PARTITION BY RANGE COLUMNS(Datum)
42     SUBPARTITION BY HASH( TO_DAYS(Datum) ) (
43         PARTITION quartal1 VALUES LESS THAN ('2011-04-01')
               (
44              SUBPARTITION s0,
45              SUBPARTITION s1,
46              SUBPARTITION s2,
47              SUBPARTITION s3
48         ),
49         PARTITION quartal2 VALUES LESS THAN ('2011-07-01')
               (
50              SUBPARTITION s4,
51              SUBPARTITION s5,
52              SUBPARTITION s6,
53              SUBPARTITION s7
54         ),
55         PARTITION quartal3 VALUES LESS THAN ('2011-10-01')
               (
56              SUBPARTITION s8,
57              SUBPARTITION s9,
58              SUBPARTITION s10,
59              SUBPARTITION s11
```

```
60          ),
61          PARTITION quartal4 VALUES LESS THAN MAXVALUE (
62              SUBPARTITION s12,
63              SUBPARTITION s13,
64              SUBPARTITION s14,
65              SUBPARTITION s15
66          )
67      );
68
69
70  SET SQL_MODE=@OLD_SQL_MODE;
71  SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
72  SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

Listing A.9: Indexerstellung als Hash

```
1  CREATE INDEX kunde_idx
2      USING HASH
3      ON adbc.kunde (KUNDE_ID);
4
5  CREATE INDEX produkt_idx
6      USING HASH
7      ON adbc.produkt (PRODUKT_ID);
8
9  CREATE INDEX warenkorb_idx
10     USING HASH
11     ON adbc.warenkorb (WARENKORB_ID);
12
13 CREATE INDEX warenkorb_kunde_idx
14     USING HASH
15     ON adbc.warenkorb (Kunde_KUNDE_ID);
16
17 CREATE INDEX warenkorb_has_produkt_idx
18     USING HASH
19     ON adbc.warenkorb_has_produkt (WARENKORB_HAS_PRODUKT_ID
           );
20
```

```
21 CREATE INDEX
       warenkorb_has_produkt_Warenkorb_WARENKORB_ID_idx
22       USING HASH
23       ON adbc.warenkorb_has_produkt (Warenkorb_WARENKORB_ID);
24
25 CREATE INDEX warenkorb_has_produkt_Produkt_PRODUKT_ID_idx
26       USING HASH
27       ON adbc.warenkorb_has_produkt (Produkt_PRODUKT_ID);
```

Listing A.10: Indexerstellung als B-Tree

```
 1 CREATE INDEX kunde_idx
 2       USING BTREE
 3       ON adbc.kunde (KUNDE_ID);
 4
 5 CREATE INDEX produkt_idx
 6       USING BTREE
 7       ON adbc.produkt (PRODUKT_ID);
 8
 9 CREATE INDEX warenkorb_idx
10       USING BTREE
11       ON adbc.warenkorb (WARENKORB_ID);
12
13 CREATE INDEX warenkorb_kunde_idx
14       USING BTREE
15       ON adbc.warenkorb (Kunde_KUNDE_ID);
16
17 CREATE INDEX warenkorb_has_produkt_idx
18       USING BTREE
19       ON adbc.warenkorb_has_produkt (WARENKORB_HAS_PRODUKT_ID
             );
20
21 CREATE INDEX
       warenkorb_has_produkt_Warenkorb_WARENKORB_ID_idx
22       USING BTREE
23       ON adbc.warenkorb_has_produkt (Warenkorb_WARENKORB_ID);
24
```

```
25 CREATE INDEX warenkorb_has_produkt_Produkt_PRODUKT_ID_idx
26       USING BTREE
27       ON adbc.warenkorb_has_produkt (Produkt_PRODUKT_ID);
```

B Arbeitsaufteilung

Arbeit	C. Ochmann	I. Körner
Abstract		0

Tabelle B.1: Aufteilung vom Abstract

Arbeit	C. Ochmann	I. Körner
Einleitung		1.1
Aufgabenstellung		
Forschungsgegenstand		
akt. Wissensstand		
Eingesetzte Datenbank		
Projektplanung		
Anwendungsfälle		
EasyMock		
Dependency Injection		

Tabelle B.2: Aufteilung von Kapitel 2

Arbeit	C. Ochmann	I. Körner
Datengenerator		

Tabelle B.3: Aufteilung von Kapitel 3

Arbeit	**C. Ochmann**	**I. Körner**
Ausblick		

Tabelle B.4: Aufteilung von Kapitel 6

C Eigenständigkeitserklärung

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe. Mir ist bekannt, dass jede Form des Plagiats mit der Note 5 (Betrugsversuch) bewertet wird.

Ochmann, Christof Unterschrift:

Körner, Ingo Unterschrift: