SQL – Tipps und Tricks – Part II

PHP-User-Group Stuttgart

08.12.2010

- SQL JOIN Techniken richtig einsetzen
- Einfluß von Datentypen auf die Abfragegeschwindigkeit
- Performanceanalyse mit MySQL EXPLAIN (ein Einstieg)
- SQL Tipps und Tricks

SQL – Tipps und Tricks – Part II

Wer bin ich ?

Thomas Wiedmann

- > 21 Jahre Problemlösungen in der Softwareentwicklung
- Seit acht Jahren Projekte mit PHP und Oracle PL/SQL bzw. DB2/NT
- Zend Certified PHP Engineer (ZCE)
- IBM Certified Solution Expert - DB2 UDB v7.1 Database Administration
- Autor diverser Fachartikel in der „Toolbox“ und im PHP-Magazin
- Autor des Buches „DB2 – SQL, Programmierung, Tuning“ © 2001
- SQL-Tipps, MySQL-EXPLAIN und Performance in der **SQL-Backstube**

SQL – Tipps und Tricks – Part II

Überblick

- Theta (old style) JOIN und ANSI JOIN vermischen
- Tabellendesign Tuning (CHARSET und VARCHAR)
- Performancesteigerung mit tinyint(1) ?
- Einstieg in MySQL EXPLAIN anhand eines Beispiels
- Nächste AUTO_INCREMENT ID vorab ermitteln
- Spaß mit SQL (..hm, gibt es sowas überhaupt? ..)

ANSI JOIN und Theta JOIN mischen

Tabellenverknüpfungen oder JOIN sind absolut wesentlich für SQL-Abfragen in relationalen Datenbanken.

a) Theta oder „old style join“

```
...
 FROM tabelle1 t1, tabelle2 t2, tabelle3 t3
WHERE t1.id = t2.id
  AND t2.id = t3.id
  AND t3.name LIKE '%Bond%';
```

b) ANSI JOIN SQL-92

```
...
 FROM tabelle1 t1
 JOIN tabelle2 t2
   ON t2.id = t1.id
 JOIN tabelle3 t3
   ON t3.id = t2.id
WHERE t3.name LIKE '%Bond%';
```

JOIN - Bedingungen

Filter- / Suchbedingungen

ANSI JOIN und Theta JOIN mischen

Bei Projekt-Refakturing werden dann schon mal beide JOIN Typen gemischt. Das kann funktionieren, oder auch nicht..

I) So geht's nicht..

```
...
 FROM tabelle1 t1,
      tabelle2 t2,
      tabelle3 t3
 JOIN tabelle4 t4
   ON t4.id = t1.id
WHERE t1.id = t2.id
  AND t2.id = t3.id
  AND t3.name LIKE '%Bond%'
  AND t4.code = '007';
```

..weil bei den meisten SQL-Parsern der „JOIN" zuerst ausgewertet wird, ist der ALIAS t1.id beim „JOIN" noch gar nicht definiert.

```
MySQL: ERROR 1054 (42S22): Unknown column 't1.id' in 'on clause'
```

ANSI JOIN und Theta JOIN mischen

Bei Projekt-Refakturing werden dann schon mal beide JOIN Typen gemischt. Das kann funktionieren, oder auch nicht..

II) So geht's ..

```
...
 FROM tabelle1 t1
 JOIN tabelle4 t4
   ON t4.id = t1.id
    , tabelle2 t2,
      tabelle3 t3
WHERE t1.id = t2.id
  AND t2.id = t3.id
  AND t3.name LIKE '%Bond%'
  AND t4.code = '007';
```

..weil bei den meisten SQL-Parsern der „JOIN" zuerst ausgewertet und der ALIAS **t1.id** nun rechtzeitig existiert.

=> Funktioniert, ist aber meiner Meinung nach nicht sinnvoll (verwirrend)

SQL – Tipps und Tricks – Part II

Zusammenfassung JOIN

Theta-JOIN versus ANSI JOIN

- Beide JOIN Techniken sind erlaubt
- JOIN ist ein Mythos bei Tuning Diskussionen, denn die Performance der beiden JOIN Techniken ist identisch(!)
- ANSI JOIN ist seit SQL-92 Standard
- Mischen beider JOIN Techniken geht, ist aber fehleranfällig
- ANSI JOIN bietet OUTER JOIN

Tabellendesign Tuning

```
CREATE TABLE kontakte_utf8 (
 id INT unsigned NOT NULL,
 vorname VARCHAR(255) NOT NULL,
 name VARCHAR(255) NOT NULL,
 strasse VARCHAR(255),
 hausnummer smallint,
 plz INT,
 ort VARCHAR(255),
 birthday VARCHAR(10),
 phone_home VARCHAR(255) ,
 phone_office VARCHAR(255),
 phone_mobil VARCHAR(255) ,
 email_home VARCHAR(255),
 email_office VARCHAR(255),
 web VARCHAR(255),
 info TEXT,
 PRIMARY KEY (ID)
) ENGINE=MyISAM DEFAULT CHARSET=UTF8;
```

Tabellendesign Tuning

```
CREATE TABLE kontakte_utf8 (
 id INT unsigned NOT NULL,
 vorname VARCHAR(255) NOT NULL,
 name VARCHAR(255) NOT NULL,
 strasse VARCHAR(255),
 hausnummer smallint,
 plz INT,
 ort VARCHAR(255),
 birthday VARCHAR(10),
 phone_home VARCHAR(255) ,
 phone_office VARCHAR(255),
 phone_mobil VARCHAR(255) ,
 email_home VARCHAR(255),
 email_office VARCHAR(255),
 web VARCHAR(255),
 info TEXT,
 PRIMARY KEY (ID)
) ENGINE=MyISAM DEFAULT CHARSET=UTF8;
```

per Default
- VARCHAR(255) ?
- UTF-8 ?

Birthday VARCHAR(10) ?

Tabellendesign Tuning

```
mysql> LOAD DATA LOCAL INFILE 'kontakte.csv'
    ->   REPLACE INTO TABLE kontakte_utf8
    ->           FIELDS TERMINATED BY '|'
    ->           LINES TERMINATED BY '\r\n';
Query OK, 499999 rows affected (6.70 sec)
Records: 499999  Deleted: 0  Skipped: 0  Warnings: 0

mysql>

mysql> SELECT COUNT(*) FROM kontakte_utf8;
+----------+
| COUNT(*) |
+----------+
|   499999 |
+----------+
1 row in set (0.01 sec)
```

Tabellendesign Tuning

```
mysql> select * from kontakte_utf8
         where vorname = 'Stefan';
Empty set (0.38 sec)

mysql> select vorname, count(*) from kontakte_utf8
         group by vorname;
+---------+----------+
| vorname | count(*) |
+---------+----------+
| Bob     |    50033 |
| Emil    |    49971 |
| Franz   |    49968 |
| Fredel  |    50007 |
| Frida   |    50007 |
| Fritz   |    49997 |
| Georg   |    50013 |
| Otto    |    49968 |
| Uli     |    50024 |
| Xaver   |    50011 |
+---------+----------+
10 rows in set (5.05 sec)
```

Tabellendesign Tuning

```
CREATE TABLE kontakte_latin1 (
 id INT unsigned NOT NULL,
 vorname VARCHAR(255) NOT NULL,
 name VARCHAR(255) NOT NULL,
 strasse VARCHAR(255),
 hausnummer smallint,
 plz INT,
 ort VARCHAR(255),
 birthday VARCHAR(10),
 phone_home VARCHAR(255) ,
 phone_office VARCHAR(255),
 phone_mobil VARCHAR(255) ,
 email_home VARCHAR(255),
 email_office VARCHAR(255),
 web VARCHAR(255),
 info TEXT,
 PRIMARY KEY (ID)
) ENGINE=MyISAM DEFAULT CHARSET=LATIN1;
```

Tabellendesign Tuning

```
CREATE TABLE kontakte_latin1 (
 id INT unsigned NOT NULL,
 vorname VARCHAR(255) NOT NULL,
 name VARCHAR(255) NOT NULL,
 strasse VARCHAR(255),
 hausnummer smallint,
 plz INT,
 ort VARCHAR(255),
 birthday VARCHAR(10),
 phone_home VARCHAR(255) ,
 phone_office VARCHAR(255),
 phone_mobil VARCHAR(255) ,
 email_home VARCHAR(255),
 email_office VARCHAR(255),
 web VARCHAR(255),
 info TEXT,
 PRIMARY KEY (ID)
) ENGINE=MyISAM DEFAULT CHARSET=LATIN1;
```

Tabellendesign Tuning

```
SELECT
 MAX(LENGTH(vorname)) AS max_vorname,
 MAX(LENGTH(name)) AS max_name,
 MAX(LENGTH(strasse)) AS max_strasse,
 MAX(LENGTH(ort)) AS max_ort,
 MAX(LENGTH(phone_home)) AS max_phone_home,
 MAX(LENGTH(phone_office)) AS max_phone_office,
 MAX(LENGTH(phone_mobil)) AS max_phone_mobil,
 MAX(LENGTH(email_home)) AS max_email_home,
 MAX(LENGTH(email_office)) AS max_email_office,
 MAX(LENGTH(web)) AS max_web
FROM kontakte_latin1;

+-------------+----------+-------------+-...
| max_vorname | max_name | max_strasse | ...
+-------------+----------+-------------+-...
|           6 |       40 |          32 | ...
+-------------+----------+-------------+-...
1 row in set (5.48 sec)
```

Tabellendesign Tuning

```
CREATE TABLE kontakte_latin1_short (
 id INT unsigned NOT NULL,
 vorname VARCHAR(50) NOT NULL,
 name VARCHAR(50) NOT NULL,
 strasse VARCHAR(50),
 hausnummer smallint,
 plz INT,
 ort VARCHAR(50),
 birthday VARCHAR(10),
 phone_home VARCHAR(50),
 phone_office VARCHAR(50),
 phone_mobil VARCHAR(50),
 email_home VARCHAR(50),
 email_office VARCHAR(50),
 web VARCHAR(100),
 info TEXT,
 PRIMARY KEY (ID)
) ENGINE=MyISAM DEFAULT CHARSET=LATIN1;
```

Tabellendesign Tuning

```
CREATE TABLE kontakte_mixed (
 id INT unsigned NOT NULL,
 vorname VARCHAR(50) CHARACTER SET utf8 NOT NULL,
 name VARCHAR(50) CHARACTER SET utf8 NOT NULL,
 strasse VARCHAR(50) CHARACTER SET utf8,
 hausnummer smallint,
 plz INT,
 ort VARCHAR(50) CHARACTER SET utf8,
 birthday VARCHAR(10) CHARACTER SET utf8,
 phone_home VARCHAR(50) CHARACTER SET utf8,
 phone_office VARCHAR(50) CHARACTER SET utf8,
 phone_mobil VARCHAR(50) CHARACTER SET utf8,
 email_home VARCHAR(50) CHARACTER SET utf8,
 email_office VARCHAR(50) CHARACTER SET utf8,
 web VARCHAR(100) CHARACTER SET utf8,
 info TEXT CHARACTER SET utf8,
 PRIMARY KEY (ID)
) ENGINE=MyISAM DEFAULT CHARSET=LATIN1;
```

Tabellendesign Tuning

	A	B	C	D	E
1		UTF8	LATIN1	LATIN1_SHORT	MIXED
2	a) SELECT	0,38	0,38	0,36	0,38
3	b) GROUP	5,05	0,64	0,55	0,89
4					
5		Abfragedauer in Sekunden			
6					

Vergleich der beiden Abfragen und die Auswirkungen der Datenlänge (VARCHAR), sowie CHARSET bei UTF8 im Vergleich mit LATIN1

```
a)  select * from kontakte_xxx where vorname = 'Stefan';

b)  select vorname, count(*) from kontakte_xxx
    group by vorname;
```

SQL – Tipps und Tricks – Part II

Zusammenfassung Tabellendesign

Tabellendesign Tuning

- Muss es immer und überall UTF-8 sein (?)
- unnötige Spaltenlänge VARCHAR(255) kostet Performance (allgemein wird das Gegenteil behauptet: Mythos!)
- Einzelne Spalten auf UTF-8 setzen
- Maximale Spaltenlängen ermitteln, „Geiz ist performance“
- Die richtigen Datentypen verwenden, nicht pauschal „VARCHAR(255)“

MySQL Tuning mit tinyint(1) ?

```
CREATE TABLE test_tinyint (
 ..
 t1 tinyint(1) NOT NULL,
 ..
)
```

Welchen Zahlenbereich kann die Spalte **t1** aufnehmen?

Wählen Sie eine mögliche Antwort

a)
in einem tinyint(1) kann nur der Wert 0 oder 1 gespeichert werden

b)
in einem tinyint(1) kann der Wert -128 bis 127 gespeichert werden

MySQL Tuning mit tinyint(1) ?

```
CREATE TABLE test_tinyint (
 t tinyint NOT NULL,
 t1 tinyint(1) NOT NULL,
 t4 tinyint(4) NOT NULL,
 tu tinyint unsigned NOT NULL,
 tu1 tinyint(1) unsigned NOT NULL,
 tu4 tinyint(4) unsigned NOT NULL ,
 tuz tinyint unsigned zerofill NOT NULL,
 tuz1 tinyint(1) unsigned zerofill NOT NULL,
 tuz4 tinyint(4) unsigned zerofill NOT NULL
);

INSERT INTO test_tinyint
(    t,      t1,   t4,    tu,   tu1,   tu4,   tuz,  tuz1,  tuz4)
VALUES
(    1,      1,     1,     1,     1,     1,     1,     1,     1),
(   -1,     -1,    -1,    -1,    -1,    -1,    -1,    -1,    -1),
(   10,     10,    10,    10,    10,    10,    10,    10,    10),
(  -10,    -10,   -10,   -10,   -10,   -10,   -10,   -10,   -10),
(  100,    100,   100,   100,   100,   100,   100,   100,   100),
( -100,   -100,  -100,  -100,  -100,  -100,  -100,  -100,  -100),
( 1000,   1000,  1000,  1000,  1000,  1000,  1000,  1000,  1000),
(-1000, -1000, -1000, -1000, -1000, -1000, -1000, -1000, -1000);
Query OK, 8 rows affected, 36 warnings (0.00 sec)
Records: 8  Duplicates: 0  Warnings: 36
```

MySQL Tuning mit tinyint(1) ?

```
INSERT INTO test_tinyint
(    t,      t1,   t4,    tu,   tu1,   tu4,   tuz,  tuz1,  tuz4)
VALUES
(    1,      1,     1,     1,     1,     1,     1,     1,     1),
(   -1,     -1,    -1,    -1,    -1,    -1,    -1,    -1,    -1),
(   10,     10,    10,    10,    10,    10,    10,    10,    10),
(  -10,    -10,   -10,   -10,   -10,   -10,   -10,   -10,   -10),
(  100,    100,   100,   100,   100,   100,   100,   100,   100),
( -100,   -100,  -100,  -100,  -100,  -100,  -100,  -100,  -100),
( 1000,   1000,  1000,  1000,  1000,  1000,  1000,  1000,  1000),
(-1000, -1000, -1000, -1000, -1000, -1000, -1000, -1000, -1000);

SELECT * FROM test_tinyint ORDER BY t;
+------+------+------+-----+-----+-----+-----+------+------+
| t    | t1   | t4   | tu  | tu1 | tu4 | tuz | tuz1 | tuz4 |
+------+------+------+-----+-----+-----+-----+------+------+
| -128 | -128 | -128 |   0 |   0 |   0 | 000 |    0 | 0000 |
| -100 | -100 | -100 |   0 |   0 |   0 | 000 |    0 | 0000 |
|  -10 |  -10 |  -10 |   0 |   0 |   0 | 000 |    0 | 0000 |
|   -1 |   -1 |   -1 |   0 |   0 |   0 | 000 |    0 | 0000 |
|    1 |    1 |    1 |   1 |   1 |   1 | 001 |    1 | 0001 |
|   10 |   10 |   10 |  10 |  10 |  10 | 010 |   10 | 0010 |
|  100 |  100 |  100 | 100 | 100 | 100 | 100 |  100 | 0100 |
|  127 |  127 |  127 | 255 | 255 | 255 | 255 |  255 | 0255 |
+------+------+------+-----+-----+-----+-----+------+------+
8 rows in set (0.00 sec)
```

MySQL Tuning mit tinyint(1) ?

```
+------+------+------+-----+-----+-----+-----+------+------+
| t    | t1   | t4   | tu  | tu1 | tu4 | tuz | tuz1 | tuz4 |
+------+------+------+-----+-----+-----+-----+------+------+
| -128 | -128 | -128 |   0 |   0 |   0 | 000 |    0 | 0000 |
| -100 | -100 | -100 |   0 |   0 |   0 | 000 |    0 | 0000 |
|  -10 |  -10 |  -10 |   0 |   0 |   0 | 000 |    0 | 0000 |
|   -1 |   -1 |   -1 |   0 |   0 |   0 | 000 |    0 | 0000 |
|    1 |    1 |    1 |   1 |   1 |   1 | 001 |    1 | 0001 |
|   10 |   10 |   10 |  10 |  10 |  10 | 010 |   10 | 0010 |
|  100 |  100 |  100 | 100 | 100 | 100 | 100 |  100 | 0100 |
|  127 |  127 |  127 | 255 | 255 | 255 | 255 |  255 | 0255 |
+------+------+------+-----+-----+-----+-----+------+------+
8 rows in set (0.00 sec)

SELECT * FROM test_tinyint
 WHERE t4 = tuz4
ORDER BY t;

+-----+-----+-----+-----+-----+-----+-----+------+------+
| t   | t1  | t4  | tu  | tu1 | tu4 | tuz | tuz1 | tuz4 |
+-----+-----+-----+-----+-----+-----+-----+------+------+
|   1 |   1 |   1 |   1 |   1 |   1 | 001 |    1 | 0001 |
|  10 |  10 |  10 |  10 |  10 |  10 | 010 |   10 | 0010 |
| 100 | 100 | 100 | 100 | 100 | 100 | 100 |  100 | 0100 |
+-----+-----+-----+-----+-----+-----+-----+------+------+
```

SQL – Tipps und Tricks – Part II

Zusammenfassung Tuning mit tinyint(1)?

Der Datentyp tinyint(1) ist.. (ein „Tuningmythos“)

- TINYINT belegt 1 Byte (mit Vorzeichen -128 bis 127)
  (ohne Vorzeichen (0 bis 255)
- tinyint(1) formatiert einzig die Anzeigebreite und schränkt nicht den Wertebereich ein
- tinyint(4) formatiert die Ausgabe auf 4 Zeichen mit führenden Leerzeichen
- tinyint(4) zerofill formatiert die Ausgabe auf 4 Zeichen mit führenden Nullen

MySQL EXPLAIN

Neulich im MySQL-Forum

Frager:

Hey, meine SQL-Abfrage funzt echt unflott. Hab schon stundenlang gegooglet, aber nichts gefunden...

```
SELECT kunde_id FROM verkauf
WHERE artikel_id IN ( 10, 30);
```

Antworter:

Mach mal'n EXPLAIN...

Frager:

EXPLAIN ? :-(nie gehört...

Antworter:

Man...google doch bei Yahoo...
und streng dich mal an..

Link zu den Forenregeln

Thread closed

MySQL EXPLAIN

Was können wir dem „coolen“ Frager empfehlen?

1. Tabellenstruktur und Anzahl der Datensätze ermitteln

```
mysql> DESCRIBE verkauf;
+------------+---------------+------+-----+---------+-------+
| Field      | Type          | Null | Key | Default | Extra |
+------------+---------------+------+-----+---------+-------+
| id         | int(11)       | NO   | PRI | NULL    |       |
| kunde_id   | int(11)       | NO   | MUL | NULL    |       |
| artikel_id | int(11)       | NO   |     | NULL    |       |
| vk_preis   | decimal(10,2) | NO   |     | NULL    |       |
| vk_datum   | date          | NO   |     | NULL    |       |
+------------+---------------+------+-----+---------+-------+
5 rows in set (0.03 sec)

mysql> SELECT COUNT(*) FROM verkauf;
+----------+
| COUNT(*) |
+----------+
|       10 |
+----------+
1 row in set (0.00 sec)
```

MySQL EXPLAIN

Was können wir dem „coolen“ Frager empfehlen?

2. Tabellenindices ermitteln

```
mysql> SHOW INDEX FROM verkauf;
+---------+..+--------------------+------+-------------+----------+-------------+-
| Table   |..| Key_name           | Seq. | Column_name | Collation | Cardinality |
+---------+..+--------------------+------+-------------+----------+-------------+-
| verkauf |..| PRIMARY            |    1 | id          | A         |          10 |
| verkauf |..| idx_kunde_artikel  |    1 | kunde_id    | A         |           3 |
| verkauf |..| idx_kunde_artikel  |    2 | artikel_id  | A         |          10 |
+---------+..+--------------------+------+-------------+----------+-------------+-
3 rows in set (0.00 sec)

mysql>
```

Kombinierter „combined“ Index
Besteht aus „kunde_id“ und „artikel_id“

MySQL EXPLAIN

Was können wir dem „coolen“ Frager empfehlen?

3. EXPLAIN ausführen

```
mysql> EXPLAIN
    -> SELECT kunde_id FROM verkauf
    ->  WHERE artikel_id IN ( 10, 30);
.+---------+------+---------------+--------------------+-..-+------+-------------
.| table   | type | possible_keys | key                | .. | rows | Extra
.+---------+------+---------------+--------------------+-..-+------+-------------
.| verkauf | index | NULL          | idx_kunde_artikel | .. |   10 | Using where;
.|         |       |               |                    |    |      | Using index
.+---------+------+---------------+--------------------+-..-+------+-------------
1 row in set (0.00 sec)

mysql>
```

KEY:
Kombinierter „combined“ Index wird auch von MySQL genutzt

MySQL EXPLAIN

Analyse „type“

4a. EXPLAIN auswerten

```
mysql> EXPLAIN
    -> SELECT kunde_id FROM verkauf
    ->  WHERE artikel_id IN ( 10, 30);
.+---------+-------+---------------+-------------------+-..-+------+-------------
.| table   | type  | possible_keys | key               | .. | rows | Extra
.+---------+-------+---------------+-------------------+-..-+------+-------------
.| verkauf | index | NULL          | idx_kunde_artikel | .. |   10 | Using where;
.|         |       |               |                   |    |      | Using index
.+---------+-------+---------------+-------------------+-..-+------+-------------
1 row in set (0.00 sec)

mysql>
```

Problem: type = index

*..Dieser Join-Typ ist mit **ALL** bis auf die Tatsache identisch, dass nur der Indexbaum gescannt wird.*
*Insofern ist er in der Regel schneller als **ALL**, weil die **Indexdatei** gewöhnlich kleiner ist als die Datendatei...*

MySQL EXPLAIN

Analyse „possible_keys“ und „key“

4b. EXPLAIN auswerten

```
mysql> EXPLAIN
    -> SELECT kunde_id FROM verkauf
    ->  WHERE artikel_id IN ( 10, 30);
.+---------+-------+---------------+--------------------+-..-+------+-------------
.| table   | type  | possible_keys | key                | .. | rows | Extra
.+---------+-------+---------------+--------------------+-..-+------+-------------
.| verkauf | index | NULL          | idx_kunde_artikel  | .. |   10 | Using where;
.|         |       |               |                    |    |      | Using index
.+---------+-------+---------------+--------------------+-..-+------+-------------
1 row in set (0.00 sec)

mysql>
```

Problem: possible_keys = NULL ?

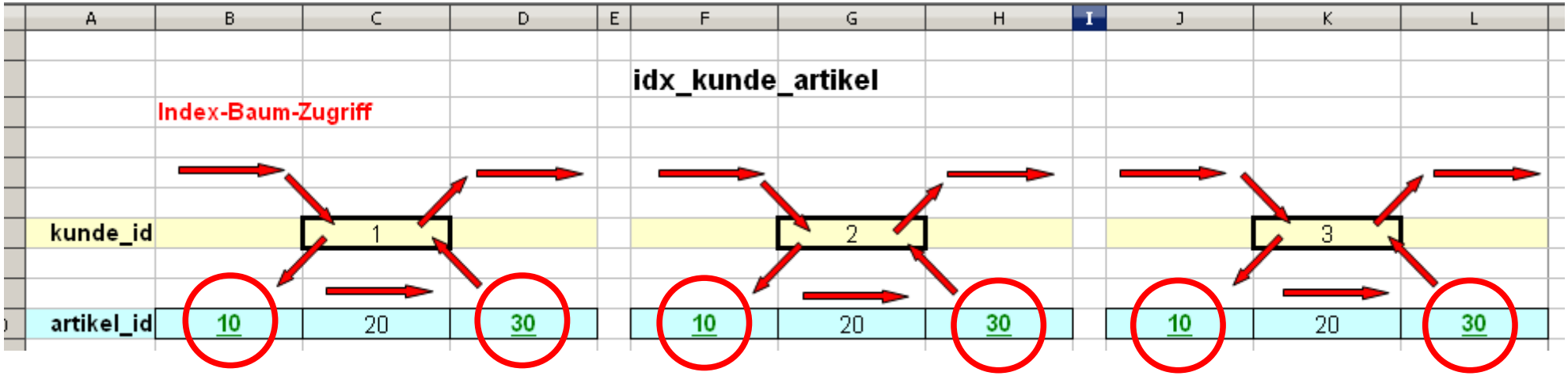
Eigentlich passt der Index nicht richtig, aber es ist wenigsten ein „Index-Only“ Zugriff über den Key **idx_kunde_artikel** möglich. Dabei werden aber alle **10** Rows durchsucht. (Achtung: Die Spalte „rows“ zeigt nicht unbedingt die tatsächlich verarbeitete Anzahl rows an, sondern eine Annäherung (aus den META-Daten).

MySQL EXPLAIN

Analyse „possible_keys“ und „key“

Schematischer Durchlauf des Index „idx_kunde_artikel“ auf der Suche nach Kunden, die die Artikel **10** oder **30** gekauft haben.

```
mysql> EXPLAIN
    -> SELECT kunde_id FROM verkauf
    ->  WHERE artikel_id IN ( 10, 30);
```



Der Index ist nicht optimal für diese Abfrage!
Wir brauchen einen anderen Index!

MySQL EXPLAIN

Zusätzlichen Index erzeugen

```
mysql> CREATE INDEX idx_artikel_kunde ON verkauf (artikel_id, kunde_id);
Query OK, 10 rows affected (0.08 sec)
Records: 10  Duplicates: 0  Warnings: 0

mysql> ANALYZE TABLE verkauf;
+----------------+---------+----------+----------+
| Table          | Op      | Msg_type | Msg_text |
+----------------+---------+----------+----------+
| sample.verkauf | analyze | status   | OK       |
+----------------+---------+----------+----------+
1 row in set (0.02 sec)

mysql> SHOW INDEX FROM verkauf;
+---------+-..-+--------------------+------+-----------+----------+-------------+
| Table   | .. | Key_name           | Seq. | Column_name| Collation| Cardinality|
+---------+-..-+--------------------+------+-----------+----------+-------------+
| verkauf | .. | PRIMARY            |    1 | id         | A        |          10|
| verkauf | .. | idx_kunde_artikel  |    1 | kunde_id   | A        |           3|
| verkauf | .. | idx_kunde_artikel  |    2 | artikel_id | A        |          10|
| verkauf | .. | idx_artikel_kunde  |    1 | artikel_id | A        |           3|
| verkauf | .. | idx_artikel_kunde  |    2 | kunde_id   | A        |          10|
+---------+-..-+--------------------+------+-----------+----------+-------------+
5 rows in set (0.05 sec)
```

MySQL EXPLAIN

5. EXPLAIN ausführen

```
mysql> EXPLAIN
    -> SELECT kunde_id FROM verkauf
    -> WHERE artikel_id IN ( 10, 30);
.+--------+-------+------------------+------------------+..+------+-------------
.| table  | type  | possible_keys    | key              |..| rows | Extra
.+--------+-------+------------------+------------------+..+------+-------------
.| verkauf| range | idx_artikel_kunde| idx_artikel_kunde|..|    6 | Using where;
.|        |       |                  |                  |..|      | Using index
.+--------+-------+------------------+------------------+..+------+-------------
1 row in set (0.03 sec)

mysql>
```

Besser: type = range

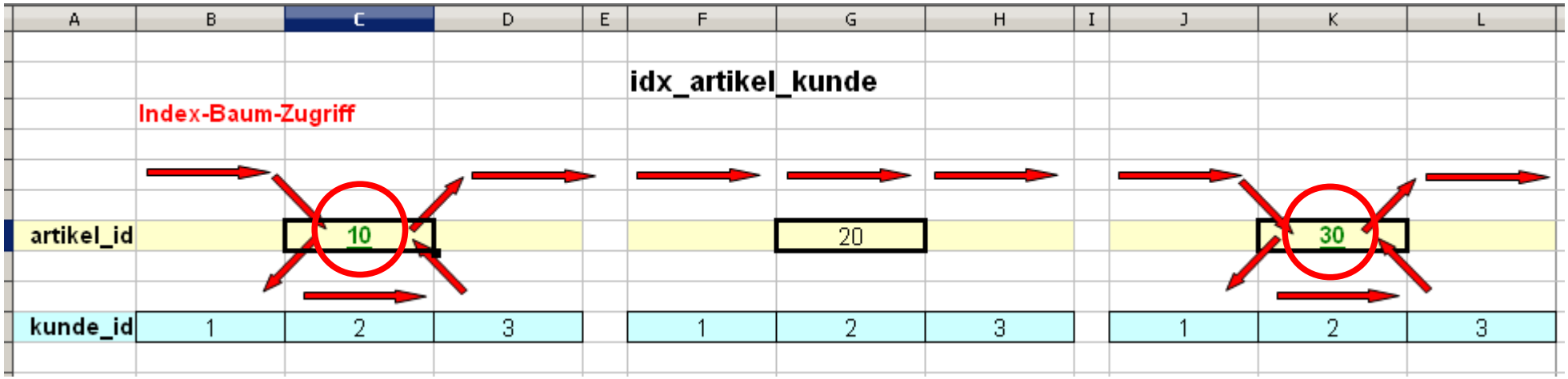
..Es werden nur Datensätze abgerufen, die in einem gegebenen Bereich liegen. Sie werden anhand eines Indexes ausgewählt. Die Spalte key im Ausgabedatensatz zeigt an, welcher Index verwendet wird. key_len enthält den längsten verwendeten Schlüsselteil. Die Spalte ref ist für diesen Typ NULL..

MySQL EXPLAIN

Analyse „possible_keys“ und „key“

Schematischer Durchlauf des Index „idx_artikel_kunde“ auf der Suche nach Kunden, die die Artikel **10** oder **30** gekauft haben.

```
mysql> EXPLAIN
    -> SELECT kunde_id FROM verkauf
    ->  WHERE artikel_id IN ( 10, 30);
```



Dieser Index ist besser für diese Abfrage!
Nur die relevanten Knoten des Index-Baumes werden durchsucht!

MySQL EXPLAIN

Kardinalität und Selektivität

Die Kardinalität bzw. die Selektivität einer Tabellenspalte entscheidet über die Wirksamkeit eines Indices. (Ermittelt sich vereinfacht so).

```
SELECT COUNT(*) AS rows,
       COUNT(DISTINCT kunde_id) AS kardi_kun,
       COUNT(DISTINCT artikel_id) AS kardi_art,
       1 / COUNT(DISTINCT kunde_id) AS sel1_kun,
       1 / COUNT(DISTINCT artikel_id) AS sel1_art,
       COUNT(*) / COUNT(DISTINCT kunde_id) AS sel2_kun,
       COUNT(*) / COUNT(DISTINCT artikel_id) AS sel2_art
  FROM verkauf;

+------+----------+----------+---------+---------+---------+---------+
| rows | kardi_kun | kardi_art | sel1_kun | sel1_art | sel2_kun | sel2_art |
+------+----------+----------+---------+---------+---------+---------+
|   10 |         3 |         3 |   0.3333 |   0.3333 |   3.3333 |   3.3333 |
+------+----------+----------+---------+---------+---------+---------+
1 row in set (0.00 sec)

mysql>
```

http://de.wikipedia.org/wiki/Kardinalit%C3%A4t_(Datenbanken)
http://de.wikipedia.org/wiki/Selektivit%C3%A4t_(Informatik)

SQL – Tipps und Tricks – Part II

Zusammenfassung MySQL EXPLAIN

- Performance-Tuning ist eine komplexe Sache mit vielen Fasetten
- Performance-Tuning ist eine permanente Aufgabe
- Dies war ein **Einstieg** in MySQL EXPLAIN
- EXPLAIN Ausgabe analysieren und verstehen (können!)
- Erkennen, welche Indices sinnvoll sind, welche nicht.
- Mehr zu EXPLAIN auf http://dev.mysql.com/doc/refman/5.1/de/explain.html

AUTO_INCREMENT ID vorab ermitteln

```
CREATE TABLE test_auto_inc (
 id INT NOT NULL AUTO_INCREMENT,
 wert VARCHAR(10) NOT NULL,
 PRIMARY KEY(id)
);

INSERT INTO test_auto_inc ( wert ) VALUES
( 'eins'),
( 'zwei'),
( 'drei');

mysql> SHOW CREATE TABLE test_auto_inc;

CREATE TABLE `test_auto_inc` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `wert` varchar(10) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=MyISAM AUTO_INCREMENT=4 DEFAULT CHARSET=latin1
```

AUTO_INCREMENT ID vorab ermitteln

Mit Hilfe der MySQL Metadaten(-bank)

INFORMATION_SCHEMA

läßt sich die nächste AUTO_INCREMENT ID vorab ermitteln

```
SELECT auto_increment
  FROM information_schema.tables
 WHERE table_schema = 'meine_datenbank'
   AND table_name = 'test_auto_inc';

+----------------+
| auto_increment |
+----------------+
|              4 |
+----------------+
```

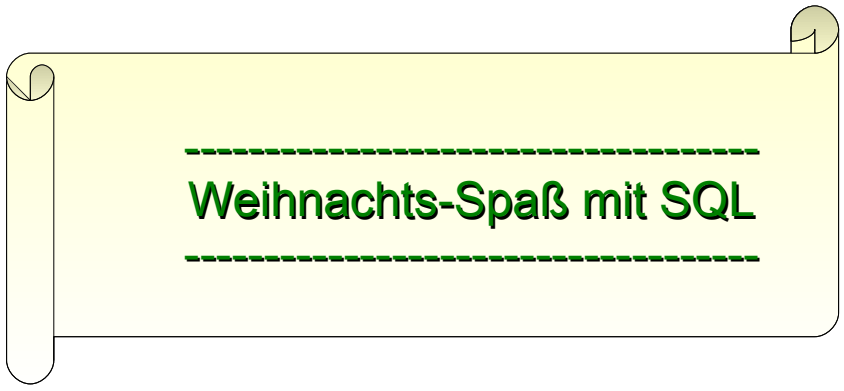
Spaß mit SQL

```
CREATE TABLE tree (
 pos INT NOT NULL,
 width INT NOT NULL
);

INSERT INTO tree VALUES
( 1, 1), ( 2, 3), (3, 5),
( 4, 7), ( 5, 9), (6, 11),
( 7, 1), ( 8, 1), (9, 1);

SELECT CONCAT(
         REPEAT(' ',
           (SELECT ROUND(MAX(width)/2,0) FROM tree) -
             ROUND(width/2,0) + 1),
         REPEAT('*',width),
         REPEAT(' ',
          (SELECT ROUND(MAX(width)/2,0) FROM tree) -
            ROUND(width/2,0) + 1)
        ) AS  ``
  FROM tree
ORDER BY pos;
```

```
SELECT CONCAT(
         REPEAT(' ',
           (SELECT ROUND(MAX(width)/2,0) FROM tree) -
             ROUND(width/2,0) + 1),
         REPEAT('*',width),
         REPEAT(' ',
         (SELECT ROUND(MAX(width)/2,0) FROM tree) -
           ROUND(width/2,0) + 1)
       ) AS  ``
  FROM tree
ORDER BY pos;
+---------------+
|               |
+---------------+
|       *       |
|      ***      |
|     *****     |
|    *******    |
|   *********   |
|  ***********  |
|       *       |
|       *       |
|       *       |
+---------------+
9 rows in set (0.06 sec)
mysql>
```



SQL – Tipps und Tricks – Part II

Überblick

- Theta (old style) JOIN und ANSI JOIN vermischen
- Tabellendesign: Tuning (CHARSET und VARCHAR)
- Performancesteigerung mit tinyint(1) ?
- Einstieg in MySQL EXPLAIN anhand eines Beispiels
- Nächste AUTO_INCREMENT-ID vorab ermitteln
- Spaß mit SQL (..hm, gibt es sowas überhaupt? ..)

The End!

Fragen ?