Willkommen zum „IBM DB2 Newsletter“

Liebe Leserinnen und Leser,

nach dem die vom Osterhasen versteckten Eier im Gras gefunden und vetilgt wurden, der Frühling eingekehrt ist, sprinten wir zum nächsten großen Ereignis der IOD in Berlin, Anfang Juni, zu dem wir einen Sondernewsletter "IOD 2009" herausgebracht haben.
DB2 Cobra Version wurde offiziell angekündigt, das IBM Database Magazin bekommt einen neuen Namen. Diese und weitere interessante Themen haben wir für Sie in diesem Newsletter zusammengestellt.

Beim Lesen und Ausprobieren wünschen wir Ihnen viel Spaß.

Für Fragen und Anregungen unsere Kontaktadresse: db2news@de.ibm.com.

Ihr TechTeam

Inhaltsverzeichnis

ANMERKUNGEN ZUR FEBRUAR AUSGABE....1

IBM INFORMATION ON DEMAND CONFERENCE IN BERLIN....2

DB2 9.7 ANGEKÜNDIGT....2

DB2 MONITORING MIT WLM....3

- DB2 WLM lizenzfreie Funktionen....3
- WLM Monitoring Setup....3
- System Überblick durch Histogramme....4
- Ein-/Ausschalten des WLM Monitoring....6

NUTZUNG VON SAMPLING IN DB2....7

VERMEIDEN UNPERFORMANTER VIEWS....9

- Situationsabhängige SQL-Anpassungen....9

CHATS MIT DEM LABOR....11

ANKÜNDIGUNG IBM DATABASE MAGAZIN....11

NEWSLETTER ARCHIV....11

ANMELDUNG/ABMELDUNG....11

DIE AUTOREN DIESER AUSGABE:....11

Anmerkungen zur Februar Ausgabe

In der Februar Ausgabe haben wir auf das Ende des DB2 V8 Supports hingewiesen. Dies führte bei Newsletter Lesern, basierend auf z/OS Systemen zur Verwirrung. Daher wollen wir dies hier nun richtig stellen: Der Artikel zum DB2 Support Ende der DB2 V8 Version bezog sich auf DB2 V8 LUW. Vielen Dank für den Hinweis.

IBM Information On Demand Conference in Berlin

Der Termin für die IOD in Berlin rückt näher. Anmeldung nicht vergessen.

Eine Vielzahl an interessanten Vorträgen und Workshops wartet auf Sie. Die Highlights aus DB2 Sicht werden wir für Sie in einer separaten email zusammenstellen.

Die IBM Information On Demand (IOD) Konferenz ist die wichtigste Fachkonferenz ihrer Art in der Region Europa, Mittlerer Osten und Afrika (EMEA). Hier treffen rund 2.000 Vertreter führender Unternehmen aufeinander, um sich in einem offenen Forum über ihre Herausforderungen, Erfolge, Strategien und Ziele rund um das Information Management auszutauschen.
Auf der 2. IBM IOD EMEA Konferenz in Berlin erwartet den Teilnehmern ein hochkarätiges Programm zu den verschiedensten Aspekten des Information Managements. Angeboten werden unter anderem spannende Vorträge von Branchenkennern und IBM Experten, über 300 Breakout-Sessions zu den aktuellsten IT- und Strategiethemen, Kundenszenarios, dazu technische Schulungen, Workshops und Produktdemos.
Darüber hinaus werden IBM Business Partner in der Information Management Expo Ausstellung einige der innovativsten Lösungen live vorstellen. Speziell für Business Partner gibt es ein dediziertes Business Partner Programm.
Weitere Informationen finden Sie unter: www.ibm.com/software/europe/data/conf

DB2 9.7 angekündigt

Am 22.4. hat die IBM die nächste Version von DB2 angekündigt (Pressemeldung). Die Cobra genannte Version 9.7 bringt viele Verbesserungen gegenüber der aktuellen Version. So lassen sich nun nicht nur relationale Daten sondern auch Indexe und XML Daten komprimieren. Die pureXML Technologie zur hierarchischen Speicherung von XML Daten ist auch bei partitionierten verfügbar und Indexe können lokal zu Tabellenpartitionen sein. Insbesondere enthält DB2 9.7 Funktionen, die den Aufwand für eine Migration von anderen Datenbanksystemen deutlich reduziert (siehe Break-Free). Verfügbar wird DB2 9.7 voraussichtlich Ende Juni, aber die vielen neuen Funktionen können Sie heute schon ausprobieren, indem Sie am Early Access Programm teilnehmen. Auf der IOD Konferenz in Berlin vom 2.-5. Juni haben Sie außerdem die Gelegenheit, direkt mit den Architekten und Experten über Cobra zu diskutieren. In einem speziellen Chat with the Lab am 6. Mai wird zudem detailliert über die Cobra Version informiert.

DB2 Monitoring mit WLM

DB2 Workload Management (WLM) wurde mit der DB2 Version 9.5 eingeführt und ist ein Teil der DB2 Engine, eine zusätzliche Installation ist daher unnötig. Weitere Informationen sind zu finden unter - Best Practices: DB2 Workload Management

Um den Großteil der WLM Funktionen anwenden zu können, ist der Lizenzschlüssel für das DB2 Performance Optimization Feature (POF) notwendig, siehe unter: http://www-01.ibm.com/software/data/db2/9/editions_features_perf_ent.html

Es gibt jedoch WLM Funktionen, die lizenzkostenfrei genutzt werden können. Diese werden im folgenden beschrieben.

DB2 WLM lizenzfreie Funktionen

In DB2 V9.5 wird WLM standardmäßig aktiviert, und in der Standard-Konfiguration laufen alle Anwendungen in der Service-Klasse - `SYSDEFAULTUSERCLASS`. DB2 WLM bietet einige freie Funktionen an, für die keine Lizenz benötigt wird.

Folgende Funktionen sind lizenzfrei möglich:

1. Sammeln statistischer Daten /Histogramm für alle Anwendungen

2. Verwendung Aktivitäten-Ereignis Monitor für alle Anwendungen

3. Verwendung der WLM Tabellen Funktionen

4. Verwendung der WLM_CANCEL_ACTIVITY Prozedur, um eine Abfrage zu stoppen.

WLM Monitoring Setup

Um DB2 WLM die Möglichkeit zu geben, Ereignis-Daten auf allen Partitionen (in DPF) zu sammeln, ist ein Ereignis Monitor (event monitor) anzulegen, der seine Daten in einen Tablespace schreibt, der über alle Partitionen liegt (z.b. der Standard USERSPACE1)
Es ist aber auch möglich einen eigenen Tablespace, nur für den Ereignis Monitor anzulegen, z.B. für eine AIX Balanced Warehouse Umgebung ergibt sich folgende DDL:

```
CREATE TABLESPACE monitor
   IN DATABASE PARTITION GROUP IBMDEFAULTGROUP
  PAGESIZE 16384 MANAGED BY DATABASE
        USING (file '/db2fs1p0/bcuaix/BCUDB/monitor' 4G) on DBPARTITIONNUMS(0)
     USING (file '/db2fs1p $N /bcuaix/BCUDB/monitor' 1G,
            file '/db2fs2p $N /bcuaix/BCUDB/monitor' 1G,
            file '/db2fs3p $N /bcuaix/BCUDB/monitor' 1G,
            file '/db2fs4p $N /bcuaix/BCUDB/monitor' 1G
          ) on DBPARTITIONNUMS (1 to 16)
  EXTENTSIZE 16
  AUTORESIZE YES MAXSIZE 40G;
```

Als nächstes ist der Event-Monitor anzulegen, der seine Daten in den neu definierten Tablespace ablegt. Es gibt dabei 3 Arten des Monitors:

- einer für Aktivitäten (activities - queries)
- einer für die Workload Statistik
- und einer für Schwellwert Verletzungen (threshold violations) - Ohne Lizenz gibt es keine Thresholds, deswegen ist es nicht sinnvoll den Monitor zu erzeugen.

```
CREATE EVENT MONITOR WLM_EVENT FOR ACTIVITIES WRITE TO TABLE
        ACTIVITY (TABLE WLM_EVENT IN MONITOR),
        ACTIVITYSTMT (TABLE WLM_EVENT_STMT IN MONITOR),
        ACTIVITYVALS (TABLE WLM_EVENT_VALS IN MONITOR),
        CONTROL (TABLE WLM_EVENT_CONTROL IN MONITOR)
        AUTOSTART;

CREATE EVENT MONITOR WLM_STATS FOR STATISTICS WRITE TO TABLE
        SCSTATS (TABLE WLM_STATS_SC IN MONITOR),
        WCSTATS (TABLE WLM_STATS_WC IN MONITOR),
        WLSTATS (TABLE WLM_STATS_WL IN MONITOR),
        QSTATS  (TABLE WLM_STATS_Q  IN MONITOR),
        HISTOGRAMBIN (TABLE WLM_STATS_HISTOGRAM IN MONITOR),
        CONTROL (TABLE WLM_STATS_CONTROL IN MONITOR)
        AUTOSTART;
```

Die oben angezeigte DDL ist auch zu finden unter ~/sqllib/misc/wlmevmon.ddl, mit geringfügig angepasster Konfiguration.

WLM Event Monitore sind nach der Anlage deaktiviert und werden durch folgende Statements aktiviert[1]:

```
db2 "SET EVENT MONITOR WLM_EVENT STATE 1"
db2 "SET EVENT MONITOR WLM_STATS STATE 1"
```

Da die Event Monitore mit der "AUTOSTART" Option erzeugt wurden, werden die Monitore automatisch aktiviert, wann immer die dazugehörige Datenpartition aktiviert wird. Alternativ dazu kann die "AUTOSTART" Option auch weggelassen werden und die Monitore manuell gestartet werden.

In der DB2 WLM Standard Konfiguration sammeln die WLM Event Monitore keine Ereignisdaten. Um für einzelne Abfragen Aktivitätsinformationen zu sammeln, werden die Schlüsselwörter `COLLECT ACTIVITY DATA` und für das Sammeln der statistischen Informationen die Schlüsselwörter `COLLECT AGGREGATE ACTIVITY DATA` des `ALTER SERVICE CLASS` Statements zur Verfügung gestellt.

Um sowohl Informationen zu den Abfrageaktivitäten, als auch statistische Werte zu erhalten, sieht das ALTER Statement wie folgt aus:

```
ALTER SERVICE CLASS SYSDEFAULTSUBCLASS UNDER SYSDEFAULTUSERCLASS
        COLLECT ACTIVITY DATA ON ALL DATABASE PARTITIONS
        WITH DETAILS
        COLLECT AGGREGATE ACTIVITY DATA EXTENDED
```

Anmerkung:

1. Für das SQL-Statement Schlüsselklausel `WITH DETAILS` verwenden
2. Die Schlüsselklausel **`ON ALL DATABASE PARTITIONS`** werden benötigt um Daten aller Datenparitionen in einer DPF Umgebung zu sammeln. Per Default werden Aktivitätsdaten nur auf dem Coordinator gesammelt.
3. Die Schlüsselklausel `COLLECT AGGREGATE ACTIVITY DATA` ist nur in Zusammenhang mit einer Service Subklasse zulässig.

System Überblick durch Histogramme

Einen Überblick über die Auslastung des Daten Servers erhält man mit der Histogramm Darstellung, die die Verteilung der Aktivitäten über die Ausführungszeit anzeigt. Histogramm Statistiken werden unter Verwendung des `WLM_STATS` Event Monitors gesammelt, der im Abschnitt WLM Monitoring Setup angelegt wurde.

[1] in einer DPF Umgebung kann die Warnung SQLSTATE=01651 erscheinen nachdem der Event Monitor aktiviert wurde (state 1), wenn der Tablespace in den ist

Um Histogramm Statistiken für die Standard-Serviceklasse (SYSDEFAULTSUBCLASS unter SYSDEFAULTUSERCLASS) zu sammeln, muss die Serviceklasse zum Sammeln der statistischen Informationen freigegeben werden.

Gesammelte Statistiken müssen nicht notwendigerweise ständig in Event Tabellen materialisiert werden. Um den Zeitpunkt des Wegschreibens der Inhalte des statistischen Datenpuffer in die entsprechende Event Tabelle zu definieren, wird der Parameter `WLM_COLLECT_INT` verwendet. Nach dem Wegschreiben, werden die Statistiken zurückgesetzt. Dieser Parameter ist gültig für alle aktiven Statistik Event Monitore.

```
## (to collect statistics every 10 minutes)
db2 update db cfg for bcudb using WLM_COLLECT_INT 10
```

Um manuell die statistischen Daten weg zuschreiben, kann folgende Stored Procedure verwendet werden:

```
CALL WLM_COLLECT_STATS()
```

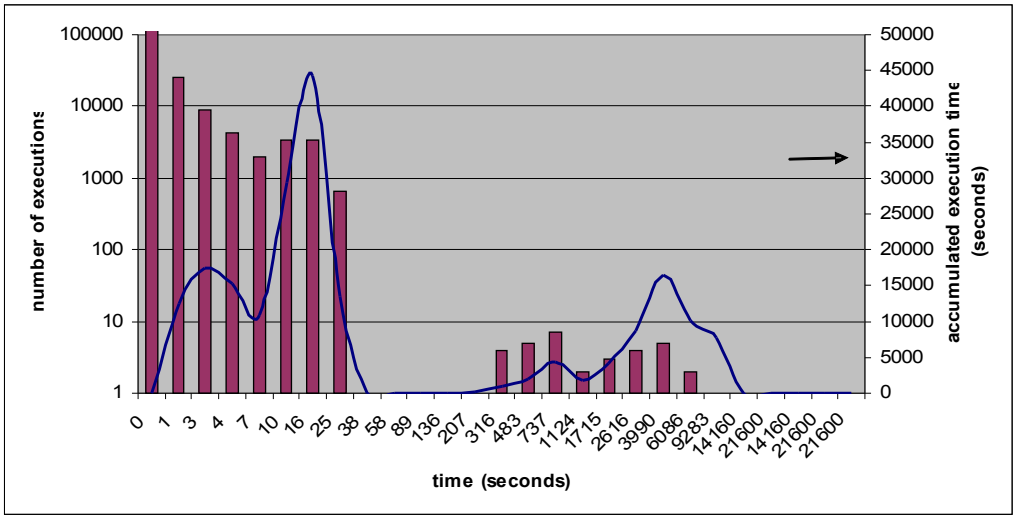
Um die Verteilung der Laufzeiten anzuzeigen, kann die Tabelle `WLM_STATS_HISTOGRAM` abgefragt werden(diese ist verbunden mit dem statistischen Event Monitor). Ein '`CoordActExecTime`' Histogram zeigt die Ausführungszeiten von nicht-verschachtelten Aktivitäten des Koordinators an. Primär wird das Ergebnis verwendet, um z.B. Aktivitäten zu ermitteln, die länger als eine Sekunde laufen.

```
SELECT TOP/1000 AS TIME_seconds,
SUM(NUMBER_IN_BIN) AS #EXECUTIONS
FROM WLM_STATS_HISTOGRAM
WHERE HISTOGRAM_TYPE = 'CoordActExecTime'
GROUP BY TOP/1000 order by TOP/1000
```

Die Ausgabe eines Tests (5 Anwendungen mit komplexen und 200 Anwendungen mit einfachen Abfragen in parallel) würde ungefähr so aussehen:

```
TIME_SECONDS         #EXECUTIONS
-------------------- --------------------
                   0               286038
                   1                24843
                   3                 8715
                   4                 4371
                   7                 1979
                  10                 3452
                  16                 3410
                  25                  662
                  38                    0
                  58                    0
                  89                    1
                 136                    0
                 207                    1
                 316                    4
                 483                    5
                 737                    7
                1124                    2
                1715                    3
                2616                    4
                3990                    5
                6086                    2
                9283                    1
               14160                    0
               21600                    0

24 record(s) selected.
```



Wenn die Werte als Graph dargestellt werden, sieht man, das die meisten Abfragen in weniger als einer Minute beendet werden. Aber es gibt einige Abfragen, die wesentlich länger als eine Minute benötigen. Die aufsummierte Ausführungszeit ("number of execution" multipliziert mit "execution time") wird durch die blaue Linie dargestellt.

Durch die Verwendung des Histogramm-Types **`'CoordActEstCost'(HISTOGRAM_TYPE = 'CoordActEstCost')`** wird die Verteilung entsprechend der vom Optimizer ermittelten Abfragekosten (timerons) dargestellt.

Weitere Informationen zur Verwendung der Histogramme und Verwendung und Änderung der Histogramm-Templates sind dem Infocenter zu entnehmen.

Ein-/Ausschalten des WLM Monitoring

Um die Beeinflussung eines Systems zu minimieren, können die Monitore online ein- und ausgeschaltet werden. Die obig definierten Monitore lassen sich über folgende Befehle deaktivieren:

```
db2 "SET EVENT MONITOR WLM_EVENT STATE 0"
db2 "SET EVENT MONITOR WLM_STATS STATE 0";
db2 "SET EVENT MONITOR WLM_THRESHOLDS STATE 0"
```

Falls die Event Monitore mit „AUTOSTART" angelegt wurden, müssen die Befehle nach jedem Neustart der Datenbank wiederholt werden.

Das Sammeln der Informationen in der Serviceklasse kann ebenfalls gestoppt werden. Dies geschieht durch folgenden Befehl:

```
$db2 "ALTER SERVICE CLASS SYSDEFAULTSUBCLASS UNDER SYSDEFAULTUSERCLASS
            COLLECT ACTIVITY DATA NONE
            COLLECT AGGREGATE ACTIVITY DATA NONE"
```

Der Status und die Definition von WLM kann über folgenden Befehl abgefragt werden:

```
$ db2look -d dbname -wlm
```

Mehr zu dem Thema WLM - wie Beispiele zu Activity Event Monitore - gibt es in der nächsten Ausgabe des DB2 Newsletter.

Nutzung von Sampling in DB2

Datenbank Tabellen werden oft sehr groß. Manchmal es ist unpraktisch und gar nicht nötig, auf alle für eine Abfrage relevanten Daten zuzugreifen. In einigen Fällen sind allgemeine Trends oder Muster interessant, und eine Schätzung innerhalb gewisser Fehlertoleranzen ist vollkommen ausreichend.

Die folgende Abfrage kann sehr viel Zeit/Ressourcen benötigen.

```
select count_big(*) from tablename
```

Es ist oft ausreichend, die Kardinalität aus den Statistiken zu lesen:

```
select card from syscat.tables where tabname='TABLENAME'
```

Eine andere Möglichkeit, diese und ähnliche Abfragen zu beschleunigen besteht darin, statt allen Daten eine zufälligen Stichprobe der zu lesen. Mögliche Anwendung von Stichprobendaten sind Abfragen mit Aggregaten wie AVG, SUM und COUNT, bei denen angenäherte Ergebniswerte aus einer Stichprobe der Daten gewonnen werden können.

DB2 stellt zwei Methoden zur Stichprobenerstellung bereit: Stichproben auf Zeilenebene (BERNOULLI) oder Stichproben auf Blockebene (SYSTEM). Bernoulli ist statistisch gesehen genauer, System jedoch effizienter. Die Anzahl der gelesenen Sätze ist bei beiden Verfahren gleich, jedoch sorgt System dafür, dass deutlich weniger Seiten von Platte gelesen werden müssen. System reduziert also die IO Last zusätzlich.

Stichproben können auch benutzt werden, um Runstats zu beschleunigen (siehe Handbuch).

Beispiel 1.

```
==> time db2 "select count_big(*) count from tpcd.lineitem"

1
--------------------------------
                      5999989709.
 1 record(s) selected.

real 2m59.693s

==> time db2 "select count_big(*) count from tpcd.lineitem TABLESAMPLE SYSTEM(0.1)"

COUNT
--------------------------------
                         6079273.

 1 record(s) selected.

real 0m2.615s
```

Das Ergebnis muss noch korrigiert werden, weil nur 0.1% der Datenseiten gelesen wurden. Es muss also mit 1000 multipliziert werden.

```
==> time db2 "select 1000*count_big(*) from tpcd.lineitem TABLESAMPLE SYSTEM(0.1)"

COUNT
--------------------------------
                      6035381000.


 1 record(s) selected.

real 0m2.523s
```

Der Unterschied zwischen der exakten Zählung der Sätze und der Schätzung über eine 0.1%-ige Stichprobe ist kleiner als 1% und die Laufzeit um den Faktor 100

schneller.

Beispiel 2.

```
==> time db2 "select count(*) count, year(L_SHIPDATE) year
          from tpcd.lineitem group by year(L_SHIPDATE)"

COUNT       year

----------- -----------

  760594472       1992
  910220849       1993
  910244122       1994
  910208622       1995
  912662642       1996
  910247097       1997
  685811905       1998

 7 record(s) selected.

real 9m12.572s
```

```
==> time db2 "select 1000*count(*) count, year(L_SHIPDATE) year
          from tpcd.lineitem TABLESAMPLE SYSTEM(0.1)group by year(L_SHIPDATE)"

COUNT       year

----------- -----------
 735073000      1992
 903343000      1993
 913321000      1994
 927272000      1995
 926918000      1996
 898293000      1997
 693819000      1998

 7 record(s) selected.

 real 0m3.789s
```

Beispiel 3.

In einer DPF Umgebung ist gleichmäßige Datenverteilung sehr wichtig. Wir prüfen Anzahl der Sätze pro Datenbankpartition mit:

```
==> time db2 "SELECT COUNT(*) count, DBPARTITIONNUM(L_ORDERKEY) partition FROM TPCD.LINEITEM
          GROUP BY DBPARTITIONNUM(L_ORDERKEY) with UR"

COUNT         PARTITION

-----------   -----------

  749974798             1
  750020018             2
  749974720             3
  749993644             4
  750014153             5
  749986641             7
  750018895             6
  750006840             8

 8 record(s) selected.

 real 13m24.834s
```

```
==> time db2 "SELECT 1000*COUNT(*) count, DBPARTITIONNUM(L_ORDERKEY) partition FROM TPCD.LINEITEM
            TABLESAMPLE SYSTEM (0.1) GROUP BY DBPARTITIONNUM(L_ORDERKEY) with UR"

 COUNT          PARTITION

 -----------  -----------

   750902000             8
   745020000             1
   740171000             2
   716710000             3
   775445000             4
   742215000             6
   758425000             5
   727258000             7

  8 record(s) selected.
```

```
real 0m4.932s
```

Es ist auch zu erwähnen, dass Select-Abfragen oft eine viel bessere Performance mit Isolationslevel UR (dirty read) haben. Wenn ein dirty read sinnvoll ist, lohnt es sich daher meist, „…with UR" am Ende einer Abfrage zuzufügen.

Vermeiden unperformanter Views

Im Data Warehouse Umfeld ist die Verwendung von Views sehr beliebt. So können Entwickler Sichten auf die Daten erzeugen, die andere Entwickler in ihren SQLs oder Views verwenden. Auf diese Art und Weise kommen oft mehrschichtige Konstrukte zustande (also Views auf Views auf Views ... u.s.w.). Komplexe Zusammenhänge können so gekapselt und auf höherer Ebene verborgen werden.
Ein ständiges Thema ist aber auch die Abfrageperformance. Die Gründe für schlechte Laufzeiten können aber vielschichtig sein. Beispiele dafür sind:

- Fehlende/veraltete Statistikdaten,
- erweiterte Statistiken werden nicht genutzt,
- fehlende Indexe,
- schlechte Partitionierungsschlüssel,
- schlechtes Clustering,
- …u.s.w.

Manchmal liegt es aber auch an ungünstig definierten Views.

In diesem Artikel werden einige Varianten aufgeführt, die in speziellen Fällen zu einer Performance-Steigerung führten. Diese Liste ist nicht komplett und auch sicherlich nicht generell gültig, soll aber helfen, auch über andere Alternativen nachzudenken. Natürlich muss man im Einzelfall anhand von Zugriffsplänen entscheiden.

Situationsabhängige SQL-Anpassungen

1.) Vermeide UNION

Im ersten Beispiel wollte ein Entwickler eine Spalte einfügen, die abhängig von anderen Spalten, einen gewissen Spaltenwert haben sollte. Folgender Ansatz wurde gewählt: UNION über mehrere Selektionen derselben Tabelle

```
Create View MY_VIEW as
Select A.* , cast('X' as char(1)) as C1
From Tabelle A
Where KEY in (…1.Liste mit Werten …)
UNION
Select A.* , cast('Y' as char(1)) as C1
From Tabelle A
Where KEY in (…2.Liste mit Werten …)
UNION …
```

Die Laufzeiten von Queries, die diese View benutzten, war sehr schlecht. Ursache dafür ist, dass die zugrunde liegende Tabelle mehrfach gelesen werden muss. Da die Werte in den IN-Listen jedoch unterschiedlich sind, konnte die View unter Verwendung von Case umdefiniert werden:

```
Create View MY_VIEW as
Select A.* ,
case when KEY in (…1.Liste mit Werten …) then 'X'
when KEY in (…2.Liste mit Werten …) then 'Y'
else '-' end as C1
from Tabelle A
where KEY in (…Liste mit Werten aus 1 und 2 …)
```

Die Tabelle muss somit nur noch einmal gelesen werden und die Laufzeit der Queries, die diese View benutzten wurde wesentlich besser.

2.) Überprüfe Joins über „virtuelle" Spalten.

Im zweiten Beispiel wollte ein Entwickler das Anhängen zusätzlicher Spalten vermeiden. Er stellte Views mit folgendem Aufbau zur Verfügung:

```
Create View V1 as select ....viele Spalten, …,
Date('31.12.2008') as Buchungsende from …
```

Die Spalte „Buchungsende" ist nicht Bestandteil einer Tabelle sondern eine durch die View hinzugefügte Konstante. Sie wurde aber in Views und Abfragen als Joinkriterium zur Konstruktion komplexerer Abfragen verwendet.

```
Select ...
From …
Tabelle T1
Inner join
V1
On  V1.Buchungsende=T1.Datum
... where ....
```

Letztendlich wurde die Spalte der Tabelle hinzugefügt und ein Index darüber gelegt. Der Unterschied war viele Stunden zu wenigen Minuten Laufzeit.
Nicht immer bringt dieses Verfahren jedoch eine Verbesserung der Laufzeit. Hier ist jeder Fall einzeln zu bewerten!

3.) Vermeide Joins über gecastete Spalten.

In diesem Beispiel wurde in einer View über zwei unterschiedliche Feldtypen gejoint. Dazu hat der Entwickler die Castfunktion verwendet.

```
Create View V1 as Select ...
From ...
Tabelle T1
Left outer Join
Tabelle T2
On T1.COLUMNn=DEC(T2.COLUMNn)
…
```

In den übergeordneten Abfragen führte das zu NLJOINs, welche sich ungünstig auf die Laufzeit auswirkten (viele Stunden Laufzeit). Wir konnten durch die Verwendung von nested Tabellen den Zugriffsplan positiv beeinflussen.

```
Select …
From
Tabelle T1
Left outer Join
(select DEC(T2.COLUMNn) as CN from Tabelle T2) X
on T1.COLUMNn=X.CN
…
```

Die Laufzeit war danach nur ca. 20 Minuten.

4.) Vermeide *-Views bei vielen Spalten.

Die Verwendung des *-Operators in der Viewdefinition vermeidet, dass man alle Spaltennamen hinschreiben muss. Da der Mensch eher „schreibfaul" ist, wird der Operator auch gerne dann verwendet, wenn man gar nicht alle Spalten braucht. In dem Beispiel stellte ein Entwickler eine View zur Verfügung, die etwa folgenden Aufbau hatte:

```
create view V1 as Select * from ….komplexes SQL mit gaaanz vielen und langen Spalten…
```

Im übergeordneten, sehr komplexen SQL, konnte der Optimizer eine Einschränkung

(Reduktion auf wenige Spalten) leider nicht durch Queryrewrite weitergeben und erzeugte riesengroße Zwischendateien – er wählte die Joinmethode NLJOIN. Wir ersetzten die View durch:

```
create view V1 as Select ... nur die Spalten die ich wirklich brauche ….from ….
```

Damit änderte sich der Join in HSJOIN, welches eine dramatisch bessere (Faktor über 1000) Laufzeit zur Folge hatte.

Chats mit dem Labor

Der nächste Chat mit dem Lab zum Thema "DB2 9.7 Overview" findet am 6. Mai statt.

Eine Liste der bereits durchgeführten Chats ist hier zu finden.

Die Präsentationen der Chats, können angeschaut und heruntergeladen werden.

Ankündigung IBM Database Magazin

Das "IBM Database Magazin" kommt in der nächsten Ausgabe unter dem neuen Namen "IBM Data Management Magazin" heraus.
Der Internet-Auftritt des IBM Data Management Magazins wird zu finden sein unter der IBM developerWorks Webseite.

Newsletter Archiv

Alte Ausgaben vom DB2-NL sind nun zum Nachlesen in den Archiven zu finden von:

- BYTEC : https://www.bytec.de/de/software/ibm_software/newsletter/db2newsletter/
- DRAP Solutions: http://www.drap.de/link/db2
- Lis.Tec: http://www.listec.de/DB2_Newsletter/View_category.html
- Cursor Software AG : http://www.cursor-distribution.de/index.php/aktuelles/db2-newsletter

Anmeldung/Abmeldung

Sie erhalten diesen Newsletter bis zur 3ten Ausgabe ohne Anmeldung. Wenn Sie weiterhin diesen Newsletter empfangen wollen, schicken Sie Ihre Anmeldung mit dem Subjekt „ANMELDUNG“ an db2news@de.ibm.com.

Die Autoren dieser Ausgabe:

Sollten Sie Anfragen zu den Artikeln haben, können Sie sich entweder direkt an den jeweiligen Autor wenden oder stellen Ihre Frage über den DB2 NL, denn vielleicht interessiert ja die Antwort auch die anderen DB2 NL Leser.

Doreen Stein	IT-Spezialist für DB2 LUW, IBM SWG; djs@de.ibm.com
Nela Krawez	IBM SWG, InfoSphere Balanced Warehouse Development Artikel: Nutzung von Sampling in DB2
Wilfried Hoge	IBM SWG, Technical Sales IM Artikel: DB2 9.7 angekündigt
Frank Berghofer	IT-Spezialist für DB2 LUW, IBM SWG Artikel: Vermeiden unperformanter Views

Reviewer und Ideenlieferanten:

Nela Krawez	IBM SWG, InfoSphere Balanced Warehouse

	Development
Wilfried Hoge	IBM SWG, Technical Sales IM

A Smarter Planet: IBM SWG IM Services in enger Zusammenarbeit mit Business Partnern und ISV's.

SWG Information Management Services bietet seit Anfang des Jahres IBM Business Partnern, Distributoren und ISV's eine enge Zusammenarbeit in allen Bereichen rund um die IBM Information Management Produkte an.

Im Rahmen der "A Smarter Planet" Initiative soll diese Zusammenarbeit unseren Partnern die Möglichkeit geben neue Themen zu entdecken und Lösungen zu implementieren, welche über den bisherigen Standard hinausgehen: um unsere Welt ein bisschen intelligenter und transparenter zu machen.

Durch unser perfekt aufgestelltes Team von über 120 Personen, in den Bereichen Technik, Architektur und Projektleitung, können wir unseren Partnern genau die Skills und Erfahrungen zur Seite stellen, die es ihnen ermöglichen die neuen Wege zu beschreiten.

Möchten auch Sie mithelfen unsere Welt "smarter" zu machen?
Dann schreiben Sie eine kurze Email an volker.fraenkle@de.ibm.com.