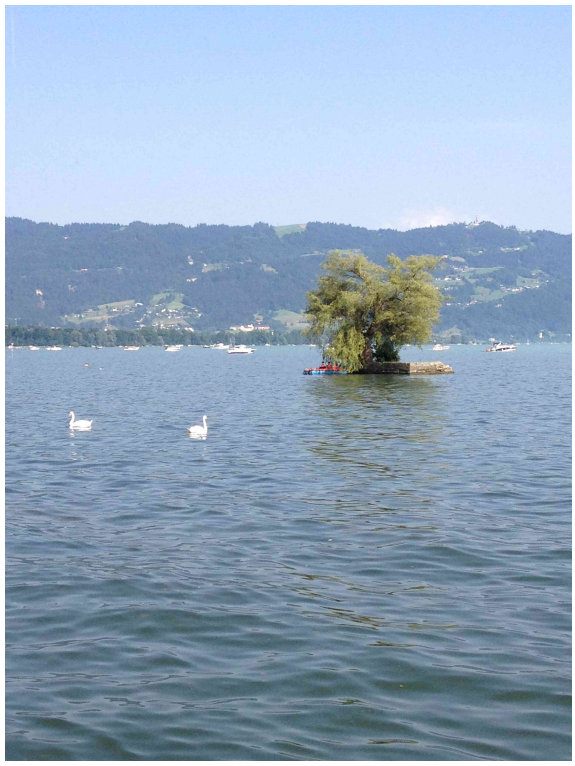
Willkommen zum „IBM Informix Newsletter“

Inhaltsverzeichnis

Aktuelles

Liebe Leserinnen und Leser,



Europa wächst zusammen, und so haben wir unsere Zusammenarbeit mit dem „Informix Newsletter Frankreich“ verstärkt. Nachdem in den letzten Jahren immer wieder sporadisch Artikel unserer Ausgabe von den Kollegen in Frankreich übernommen wurden, und wir im Gegenzug Ideen aus Frankreich umgesetzt haben, wurde nun eine intensivere Zusammenarbeit vereinbart, von der alle Seiten profitieren.
Wahrscheinlich sitzen Sie aber derzeit gerade entspannt auf einer Insel, lassen es sich gut gehen und geniessen ein kühles Getränk ...
Da kommt dieser Newsletter sicher gerade recht als interessante Lektüre.
Wer dazu bereits Informix in der Cloud über Softlayer nutzt, der kann vom Urlaubsort auch über eine sichere Verbindung unsere Tipps direkt umsetzen bzw. ausprobieren.

Viel Spaß mit den Tipps der aktuellen Ausgabe.
Ihr TechTeam

TechTipp: TimeSeries – Daten löschen mit DelRange / DelClip / DelTrim

Zeitreihen (TimeSeries) sind eine sehr effiziente Art der Datenspeicherung. Das Prinzip beruht auf die Speicherung des Schlüssels, wie z.B. der Kennung einer Messstelle, dem Startzeitpunkt der Messung, und den eigentlichen Messdaten. Bei regulären Zeitreihen werde die einzelnen Messwerte mit Offset, ohne Speicherung der Zeitmarke, abgelegt. Bei irregulären Zeitreihen muss der Zeitstempel mit gespeichert werden.
Das Löschen von Werten aus einer Zeitreihe unterschiedet sich, je nachdem ob es sich um die Entfernung der gesamten Zeitreihe (z.B. Messpunkt) mit allen Werten handelt, oder ob nur gewisse Messwerte aus der Zeitreihe gelöscht werden sollen.
Die gesamte Zeitreihe löschen geht im SQL mit einem einfachen Aufruf der Art:

```
delete from … where …
```

Hierdurch werden alle Werte, die zu der angegebenen Zeitreihe gehören mit gelöscht. Dies können z.B. bei minutengenauen Werten zur Stromerzeugung einige hunderttausend Werte sein.

Das Löschen von Werten über eine vituelle Tabelle, die für die Tabelle der Zeitreihen erstellt wurde, ist nicht möglich und wird mit einer Fehlermeldung beantwortet:

```
240: Could not delete a row.
12801: An attempt was made to invoke an access_method
          routine that does not exist.
```

Die Löschung einzelner Werte aus einer Zeitreihe muss daher statt auf der virtuellen Tabelle auf der Zeitreihentabelle selbst mit TimeSeries-Funktionen erfolgen.
Um Werte aus einem Bereiche zu löschen, kann eine der Funktionen „DelTrim“, „DelClip“ oder „DelRange“ verwendet werden, die wir in den folgenden Beispielen vorstellen wollen.

Für den Test erstellen wir eine Beispieltabelle mit einer Zeitreihe:

```
create row type sensor_t
(
        timestamp       datetime year to fraction(5),
        value   decimal(15,2)
);

create table sensor_ts (
        sensor_id char(40),
        sensor_type char(20),
        sensor_unit char(6),
        sensor_values timeseries(sensor_t),
        primary key (sensor_id)
) lock mode row;
```

Darauf definieren wir eine virtuelle Tabelle:

```
execute procedure TSCreateVirtualTab (
        'sensor_data',
        'sensor_ts',
        'origin(2015-07-20
08:00:00.00000),calendar(ts_1min),container(sensor_cont),threshold
(0),regular',0,'sensor_values');
```

und füllen ein paar Werte ein:

```
insert into sensor_data values ("Sensor01", "Temp", "C", "2015-07-
20 08:06"::datetime year to minute, 28.5);
insert into sensor_data values ("Sensor01", "Temp", "C", "2015-07-
20 08:07"::datetime year to minute, 29.6);
insert into sensor_data values ("Sensor01", "Temp", "C", "2015-07-
20 08:08"::datetime year to minute, 31.9);
insert into sensor_data values ("Sensor01", "Temp", "C", "2015-07-
20 08:22"::datetime year to minute, 32.1);
insert into sensor_data values ("Sensor01", "Temp", "C", "2015-07-
20 08:23"::datetime year to minute, 30.2);
insert into sensor_data values ("Sensor01", "Temp", "C", "2015-07-
20 08:33"::datetime year to minute, 29.5);
```

Zum Test formulieren wir die Abfrage auf die TimeSeries-Tabelle direkt:

```
select * from sensor_ts;

sensor_id       Sensor01
sensor_type     Temp
sensor_unit     C
sensor_values   origin(2015-07-20 08:00:00.00000),
calendar(ts_1min), container(
                sensor_cont), threshold(0), regular, [NULL, NULL,
NULL, NULL, NULL, NULL, (28.50    ), (29.60    ),(31.90      ),
NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL,
NULL, NULL, (32.10            ), (30.20            ), NULL, NULL,
NULL, NULL, NULL, NULL, NULL, NULL, NULL, (29.50    )]
```

und betrachten die Werte mit der Transpose Funktion:

```
time                        value
2015-07-20 08:06            28.50
2015-07-20 08:07            29.60
2015-07-20 08:08            31.90
2015-07-20 08:22            32.10
2015-07-20 08:23            30.20
2015-07-20 08:33            29.50
```

Nun löschen wir die Werte vom 20.Juli zwischen 08:07 und 08:23 mittels „DelRange“:

```
update sensor_ts
set sensor_values = DelRange(sensor_values,
          “2015-07-20 08:07:00.00000“,
          “2015-07-20 08:23:00.00000“)
where sensor_id = “Sensor01 “;

 1 row(s) updated.
```

Der Abfrage zeigt nun nur noch die verbliebenen Werte:

```
sensor_id       Sensor01
sensor_type     Temp
sensor_unit     C
sensor_values   origin(2015-07-20 08:00:00.00000),
calendar(ts_1min), container(
                sensor_cont), threshold(0), regular, [NULL, NULL,
NULL, NULL, NULL, NULL, (28.50    ), NULL, NULL, NULL, NULL, NULL,
NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL,
NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, (29.50
)]

time                            value
2015-07-20 08:06                28.50
2015-07-20 08:33                29.50
```

Das Ergebnis des Löschvorgangs ist unabhängig davon, welche der Funktionen „DelClip“, „DelRange“ oder „DelTrim“ genutzt wurde. Der Unterschied liegt in der internen Verwaltung des freigegebenen Platzes in der Zeitreihe. DelTrim gibt beim Löschen am Ende der Zeitreihe die Pages, in dnen keine Daten mehr gespeichert sind, wieder an den Container im DBSpace zurück. DelClip behält den Platz bei und füllt die gelöschten Einträge mit NULL. DelRange gibt an beliebiger Stelle der Zeitreihe die Pages frei, auf denen alle Daten gelöscht wurden.

TechTipp: TimeSeries – ältestes Element finden (GetFirstElementStamp)

Das Beispiel im vorhergehenden Artikel zeigt die einfache Möglichkeit alle Werte ab einem gewissen Zeitpunkt bis jetzt zu löschen.
z.B.:

```
update sensor_ts
set sensor_values = DelRange(sensor_values,
          "2015-07-20 08:42:00.00000",
          current)
where sensor_id = "Sensor01";
```

Meist stellt sich jedoch die Aufgabe, Werte zu löschen, die älter sind als eine vorgegebene Lebenszeit, z.B. 42 Stunden. Da die Funktionen zum Löschen immer einen Startzeitpunkt benötigen, stellt sich die Problematik, entweder den initialen Zeitpunkt der Zeitreihe zu ermitteln (Werte vor diesem Zeitpunkt werden als Fehler abgewiesen), oder die Zeitmarke des ältesten Elements der Zeitreihe zu ermitteln.
Das älteste Element aus einer Zeitreihe kann mittels GetFirstElementStamp(<ts>) ermittelt werden.
Im Beispiel könnten so z.B. alle Werte, die älter sind als 42 Stunden sind, gelöscht werden:

```
update sensor_ts
set sensor_values = DelRange(sensor_values,
          GetFirstElementStamp (sensor_values),
          current – 42 units hour)
where sensor_id = "Sensor01";
```

Den Ursprung einer Zeitreihe ermitteln wir im folgenden Artikel.

TechTipp: TimeSeries – Ursprung der TimeSeries ermittlen (GetOrigin)

Viele Funktionen auf Zeitreihen benötigen ein gültiges Datum zur Ausführung. Werte, die vor dem Ursprung der Zeitreihe liegen, oder Funktionen auf Ranges, deren Startzeitpunkt vor dem Ursprung der Zeitreihe wäre, werden mit einer Fehlermeldung abgewiesen. Mit der Funktion GetOrigin() kann je Zeitreihe der Ursprung ausgelesen werden.

Beipiel:

```
select GetOrigin(sensor_values)
from sensor_ts

(expression)
2015-07-20 08:00:00.00000
```

TechTipp: ONCONFIG – SMX_NUMPIPES (ab 12.10.xC5)

Die Idee zu diesem Artikel stammt von Yoram Benchetrit (Informix Newsletter Frankreich).

Die Kommunikation zwischen dem Primary Server und den RSS-Servern erfolgt mittels Server Multiplexer Group Connection (SMX). Bisher war die Anzahl der Pipes, die für diese Verbindung genutzt werden konnten, fest auf 1 (+UPDATABLE_SECONDARY) definiert. Ab dem Release 12.10.xC5 ist die Anzahl der SMX-Pipes nun grösser konfigurierbar. Entscheidend hierfür ist der Primary Server, dessen Konfiguration die Anzahl der Pipes auch auf den HDR und RSS Servern bestimmt, unabhängig davon, wie dort jeweils der Wert in der ONCONFIG Datei gesetzt ist. Die SMX Verbindung kann verschlüsselt werden, oder wie im Newsletter Juni beschrieben, über eine SSL Verbindung erfolgen.
Konfiguriert wird mit diesem Parameter die Anzahl paralleler TCP Connections (innerhalb derer dann jeweils SMX multiplexed/full-duplex Kommunikation stattfindet) zwischen dem Primary und einem Secondary Server, und damit also der mögliche Parallelitäts-Grad der darüber erfolgenden Kommunikation.
Der Nutzen hängt vorallem von der Netzwerk-Latenz ab. Das zu lösende Problem ist ggf. eine große WAN Latenz (bei nicht ausgenutzter Bandbreite).
In einem schnellen Netzwerk mit geringer Latenz ist somit nicht allzu viel Vorteil zu erwarten ausser wenn jedes einzelne Netzwerk Paket viel Zeit benötigt, um überhaupt übertragen zu werden.
Der Datenverkehr, dem dies hauptsächlich zu Gute kommt, betrifft die Loginformationen.

In der ONCONFIG ist ab der Version 12.10.xC5 folgender Eintrag zu finden:

```
# SMX_NUMPIPES      - Defines the number of network pipes to be used by
#                     RSS. This value is used to set the level of
#                     network threading for SMX, which can increase the
#                     transfer rates on slower network connections.
#                     The valid values are 1 and above.
```

Es ist dann sinnvoll den Wert zu erhöhen, wenn ein Secondary gegenüber dem Primary wegen langsamer Übertragung zu weit zurückfällt (zu sehen u.a. an der „Lag Time“ im OAT → Cluster), das Netzwerk aber von der Bandbreite her noch Kapazitäten hat.
SMX_NUMPIPES ist also ein probates Mittel, möglichst viele Replikations-Threads im Server auf modernen Netzwerken parallel zu beschäftigen.
In einem folgenden Artikel in diesem Newsletter wird beschrieben, wie die Auswirkung und Verteilung mittels „onstat -g smx“ überwacht werden kann.
Die HDR nutzt die SMX Threads nur in wenigen Bereichen, daher hat der Parameter bei Nutzung von HDR keine messbaren Auswirkungen auf die Kommunikation. Die Anzahl an SMX Threads wird jedoch mit gleicher Anzahl für jede HDR- und RSS-Verbindung erstellt.
Der Parameter in der ONCONFIG Datei lässt sich nicht dynamisch ändern. Mittels „onmode -x“ können dynamisch zusätzliche SMX-Pipes erstellt werden, wie in einem folgenden Artikel beschrieben.

TechTipp: ONCONFIG – SMX_COMPRESS

Der Parameter SMX_COMPRESS steuert, ob die Datenübertragung zu einem RSS Secondary komprimiert erfolgen soll und bestimmt den Komprimierungsgrad.

Werden Daten vor der Übertragung komprimiert, so spart dies Bandbreite bei der Übertragung, kostet aber für die Komprimierung und Dekomprimierung zusätzliche CPU-Resourcen.

Für den Parameter SMX_COMPRESS gibt es folgende Möglichkeiten:

- -1 = Der Server komprimiert nie die Daten für die Übertragung.
- 0 = Der Server komprimiert die Daten nur, wenn es der Zielserver erwartet.
- 1 = Die Daten werden mit geringer Komprimierung geschickt.
- 9 = Die Daten werden mit höchst möglicher Komprimierung geschickt.

Der Defaultwert ist 0.

Systeme mit ausreichend CPU-Resourcen, aber einem Engpass bei der Bandbreite der Übertragung zum RSS-Secondary profitieren besonders von der Komprimierung.

TechTipp: ONCONFIG – SMX_PING_INTERVAL / SMX_PING_RETRY

Die Parameter SMX_PING_INTERVAL und SMX_PING_RETRY steuern auf der Client-Seite einer RSS Replikation, wie lange der RSS Server auf eine Meldung des Primary Servers wartet, bevor er auf Grund fehlender Kommunikation eine Fehlermeldung in das online.log schreibt und die Verbindung beendet.
Der Default für SMX_PING_INTERVAL ist 10. Der Wert wird in Sekunden angegeben und kann zwischen 0 (unendlich) bis 60 angegeben werden.
Der Default für SMX_PING_RETRY ist 6. Dieser Wert kann zwischen 1 und MAXINT gewählt werden.
Der RSS Secondary wartet (SMX_PING_INTERVAL x SMX_PING_RETRY) Sekunden, bevor eine Verbindung als fehlerhaft notiert und abgebaut wird. Im Default sind es somit 10 x6, also 60 Sekunden, die der Secondary auf eine Meldung des Primary wartet.
Beide Parameter können mittels „onmode -wf“ dynamisch geändert werden.
Auf dem Primary haben diese Werte keine Auswirkung, so lange kein Switch erfolgt und der Primary zum Secondary wird.

TechTipp: Optionen des ONMODE (onmode -x)

Der Parameter SMX_NUMPIPES ist nicht dynamisch mittels „onmode -wf“ änderbar.
Soll für eine RSS Verbindung die Anzahl der SMX-Pipes erhöht werden, so kann dies auf dem Primary Server mit dem Befehl „onmode -x <n> <instanz>“ erfolgen.

Im Gegensatz zum Parameter SMX_NUMPIPES in der ONCONFIG Datei, besteht hierbei die Möglichkeit die Anzahl je RSS-Server individuell je Instanz zu wählen.

Im folgenden Beispiel werden der Replikation zum Server „test3“ vier weitere SMX-Pipes hinzugefügt:

```
onmode -x 4 test3
```

Im online.log ist das Prodokoll zu sehen:

```
smx creates 4 transports to server test3
```

Im „onstat -g ath“ sind die zusätzlichen Treads ebenfalls zu finden:

```
Threads:

tid tcb      rstcb   prty status           vp-class       name
103 9db918b0 4662a9e8 3  cond wait  netnorm   1cpu   smxrcv test3
104 479a0ad0 4661a368 3  cond wait  smx pipe1 1cpu   smxsnd test3
105 9db2b568 4662b2a8 1  sleeping secs: 1     1cpu   smxRecvSnd
106 9db9a508 4662c428 3  cond wait  netnorm   1cpu   smxrcv test3
107 9db81028 4662cce8 3  cond wait  smx pipe1  1cpu   smxsnd test3
108 9db81758 4662d5a8 1  sleeping secs: 1    1cpu   smxRecvSnd
109 9def1028 4662de68 3  cond wait  netnorm   1cpu   smxrcv test3
110 9df5c928 4662e728 3  cond wait  smx pipe1  1cpu   smxsnd test3
111 9dfc70d0 4662efe8 1  sleeping secs: 1     1cpu   smxRecvSnd
320 9d74a028 46623828 3  cond wait  smx pipe1  1cpu   smxsnd test3
321 479ae778 466191e8 3  cond wait  netnorm   1cpu   smxrcv test3
322 9ebeb808 46621528 1  sleeping secs: 1     1cpu   smxRecvSnd
323 9ebebb48 466312e8 3  cond wait  smx pipe1  1cpu   smxsnd test3
324 9d793028 46620c68 3  cond wait  netnorm   1cpu   smxrcv test3
325 9d793368 46631ba8 1  sleeping secs: 1    1cpu   smxRecvSnd
326 9d7936a8 46632468 3  cond wait  smx pipe1  1cpu   smxsnd test3
327 9d7939e8 46632d28 3  cond wait  netnorm  1cpu   smxrcv test3
328 9d793d28 466335e8 1  sleeping secs: 1    1cpu   smxRecvSnd
…
```

SMX-Pipes wieder im laufenden Betrieb zu entfernen ist nicht möglich. Der Befehl „onmode -x ...“ nimmt keine negativen Wert an, im Gegensatz z.B. zum „onmode -p“.

TechTipp: Optionen des ONSTAT (onstat -g smx)

Der ONSTAT liefert Informationen über die Datenübertragung mit SMX Pipes.

Aus dem „onstat -g cluster sehen wir, dass „test1“ der Primary Server ist, „test2“ ein HDR Secondary und „test3“ ein RSS-Secondary Server:

```
Primary Server:test1
test2  243,1292     243,1292     No   ASYNC(HDR),Connected,On
test3  243,1292     243,1292     Yes  ASYNC(RSS),Connected,Active
```

Der „onstat -g smx“ zeigt Details zu jeder SMX-Verbindung an. Zu sehen sind hier je Replikation (in der Ausgabe als Peer Server bezeichnet), Informationen zu Verschlüsselung und Komprimierung, sowie die Information über die Anzahl der gesendeten und empfangenen Byte, Buffer und Daten.

Beispiel:

```
  Peer server name: test3
   SMX connection address: 0x47665050
   Encryption status: Disabled
   Total bytes sent: 2292148
   Total bytes received: 107532
   Total buffers sent: 2068
   Total buffers received: 3984
   Total write calls: 1633
   Total read calls: 3984
  …
```

Zusätzliche lassen sich noch die Informationen zu den SMX Sessions abfragen:

```
onstat -g smx ses

SMX session statistics:
SMX control block: 0x47111d28

Peer      SMX session        client        reads        writes
name      address            type
test2     8f91d050 Primary RcvNotify          2            2
test3     9012e050         RSS Send        1021         3215
test3     9012e990    ProxyDispatch          3            3
```

Versionshinweis: Informix 12.10.xC5 für Mac OS X!

IBM® Informix® 12.10.xC5 steht nun auch für Mac OS X in Passport Advantage zum Download bereit. Informix ist damit der einzige Enterprise-Class Datenbank Server, der auch Mac OS X unterstützt.

Anmeldung / Abmeldung / Anmerkung

Der Newsletter wird ausschließlich an angemeldete Adressen verschickt. Die Anmeldung erfolgt, indem Sie eine Email mit dem Betreff „**ANMELDUNG**“ an **ifmxnews@de.ibm.com** senden.
Im Falle einer Abmeldung senden Sie „ABMELDUNG“ an diese Adresse.

Das Archiv der bisherigen Ausgaben finden Sie zum Beispiel unter:

http://www.iiug.org/intl/deu
http://www.iug.de/index.php?option=com_content&task=view&id=95&Itemid=149
http://www.informix-zone.com/informix-german-newsletter
http://www.drap.de/link/informix
http://www.nsi.de/informix/newsletter
http://www.cursor-distribution.de/index.php/aktuelles/informix-newsletter
http://www.listec.de/Newsletter/IBM-Informix-Newsletter/View-category.html
http://www.bereos.eu/software/informix/newsletter/

Die hier veröffentlichten Tipps&Tricks erheben keinen Anspruch auf Vollständigkeit. Da uns weder Tippfehler noch Irrtümer fremd sind, bitten wir hier um Nachsicht falls sich bei der Recherche einmal etwas eingeschlichen hat, was nicht wie beschrieben funktioniert.

Die Autoren dieser Ausgabe

Gerd Kaluzinski IT-Specialist Informix Dynamic Server und DB2 UDB
IBM Software Group, Information Management
gerd.kaluzinski@de.ibm.com +49-175-228-1983

Martin Fuerderer IBM Informix Entwicklung, München
IBM Software Group, Information Management
martinfu@de.ibm.com

Markus Holzbauer IBM Informix Advanced Support
IBM Software Group, Information Management Support
holzbauer@de.ibm.com

Yoram Benchetrit (Informix Newsletter France) - Anregung für SMX_NUMPIPES

Die Versionsinfo stammt aus dem Versions-Newsletter der CURSOR Software AG
http://www.cursor-distribution.de/download/informix-vinfo

Sowie unterstützende Teams im Hintergrund.

Fotonachweis: Gerd Kaluzinski (Insel Hoy)

Eine Woche nach dem Foto der ersten Seite hat ein Sturm den Inselbaum massiv geschädigt: