Willkommen zum „IBM Informix Newsletter“

Inhaltsverzeichnis

Aktuelles

Liebe Leserinnen und Leser,

im Neuen Jahr starten wir gleich mit der Erfüllung eines oft genannten Wunsches: „**Restore einer einzelnen Datenbank aus einem binären Backup**“. Mit Hilfe der Technik der Tenants kann die Rücksicherung nun mittels „onbar“ auf einen frei gewählten Zeitpunkt erfolgen, ohne dass die anderen Tenants (Datenbanken) in ihrer Arbeit beeinträchtigt werden. Hiermit ist einer der ersten Plätze der Wunschliste für neue Features abgearbeitet. Lesen Sie den Artikel zum Thema und testen Sie selbst !

Wie immer haben wir für Sie eine Reihe an Tipps und Tricks zusammengestellt.
Viel Spaß mit den Tipps der aktuellen Ausgabe.

Ihr TechTeam

TechTipp: Neu in der ONCONFIG der Version 12.10.xC6

Seit einigen Wochen ist die Version 12.10.xC6 verfügbar. Trotz Weihnachtsurlaub und Wintersport hat sich die Redaktion die Änderungen angesehen und getestet.
Neben einer Reihe von Verbesserungen und Bereinigungen, enthält die Version auch wieder ein paar neue Features.
Der Vergleich der Parameter in der Konfigurationsdatei ONCONFIG zwischen der Version C5 und C6 zeigt neu die Parameter

BAR_MAX_RESTORE,
SHARD_MEM, SHARD_ID,
TENANT_LIMIT_SPACE, TENANT_LIMIT_MEMORY und
TENANT_LIMIT_CONNECTIONS

In den folgenden Artikeln wollen wir auf die meisten der neuen Parameter näher eingehen. Die Parameter SHARD_MEM und SHARD_ID werde in einer ausführlichen Einführung zu Sharded Tables in der Ausgabe Februar des Informix Newsletters behandelt.

TechTipp: ONCONFIG - BAR_MAX_RESTORE

Analog der Konfiguration beim Backup mit ONBAR kann nunmehr auch die Anzahl der Restore-Prozesse konfiguriert werden. Da bei einem Restore der Datenbankserver meist nicht unter Last steht und die Datenbank nicht mit tausenden Transaktionen beschäftigt ist, könnten dadurch für den Restore mehr Ressourcen zur Verfügung gestellt werden als für den Backup. Wird der Wert nicht gesetzt, so wird als Default der Wert von BAR_MAX_BACKUP genutzt.

Mögliche Werte für BAR_MAX_RESTORE sind:

- 0 Unlimited, nur durch die Systemgrenzen beschränkt
- 1 Serieller Restore mit einem Prozess
- n n parallele Prozesse für den Restore, falls von der Architektur unterstützt

TechTipp: ONCONFIG - TENANT_LIMIT_SPACE

Mit dem Parameter TENANT_LIMIT_SPACE lässt sich der für einen Tenant nutzbare Platz in DBSpaces (ausgenommen TempSpaces und der Bereich für physical und logical Logs) beschränken. Dieser Parameter hat besonders dann Bedeutung, wenn die DBSpaces auf „autoextend“ gesetzt sind, bzw. ein „autoexpand“ vorgesehen ist.

Der Default für den Parameter ist 0 (ohne Limit). Gültige Werte können im Bereich von 1048576 bis 1717986918400 eingegeben werden, wobei der Wert die verfügbaren kB angibt.

TechTipp: ONCONFIG - TENANT_LIMIT_MEMORY

Mit dem Parameter TENANT_LIMIT_MEMORY lässt sich der für einen Tenant nutzbare Speicherplatz im Memory beschränken. Dieser Parameter hat besonders dann Bedeutung, wenn konkurrierende Tenants auf dem System abgegrenzt werden sollen und der Bufferpool als „dynamisch“ konfiguriert wurde.

Der Default für den Parameter ist 0 (ohne Limit). Gültige Werte können im Bereich von 102400 bis 2147483648 eingegeben werden, wobei der Wert die verfügbaren kB angibt.

TechTipp: ONCONFIG - TENANT_LIMIT_CONNECTIONS

Mit dem Parameter TENANT_LIMIT_CONNECTIONS lässt sich der für einen Tenant die Anzahl gleichzeitiger Verbindungen zum Datenbankserver begrenzen.

Der Default für den Parameter ist 0 (ungeschänkt). Gültige Werte können im Bereich von 1 bis 65536.

Wird das gesetzte Limit überschritten, kommt eine klare Fehlermeldung:

```
9664: Session limit exceeded; transaction aborted or session aborted.
```

TechTipp: Untermieter in der Instanz (Task: tenant create)“

Die Option „Tenants“ als abtrennte Bereiche innerhalb einer Informix Instanz zu erstellen, wurde bereits mit der Version 12 eingeführt. Ab der Version 12.10.xC6 kamen weitere Optionen bei der Erstellung von Tenants hinzu, die es erlauben die Ressourcen flexibel einzugrenzen.

Mit dem Task „tenant create“ können nun neben der Ressource der DBSpaces auch die Sessions-Limits und Tenant-Limits angegeben werden (Die ausführliche Syntax finden Sie in unserer Dokumentation). Hier im Newsletter wollen wir an einem Beispiel zwei Tenants erstellen, von denen einer im folgenden Artikel separat, unabhängig von allen anderen DBSpaces und Tenants auf einen ausgewählten Zeitpunkt restored werden soll.

Beispiel:

```
database sysadmin;
EXECUTE FUNCTION task('tenant create', 'ifx_ten1',
   '{dbspace:"ten1data",
     sbspace:"ten1sb",
     vpclass:"t1cpu,num=2",
     dbspacetemp:"ten1tmp",
     session_limit_locks:"1000",
     session_limit_memory:"100MB",
     session_limit_tempspace:"25MB",
     session_limit_logspace:"30MB",
     session_limit_txn_time:"120",
     tenant_limit_space:"2GB",
     tenant_limit_memory:"200MB",
     tenant_limit_connections:"2",
     logmode:"UNBUFFERED",
     locale:"en_us.utf8",
     case:"insensitive"}'
);
```

Ergebnis:

```
      Tenant ifx_ten1 successfully added.
```

Den zweiten Tenant erstellen wir analog dem ersten Tenant mit dem Namen ifx_ten2. Zu beachten ist, dass keine DBSpaces oder virtuelle Prozessoren zugeordnet werden, die bereits in der Definition eines anderen Tenant aufgeführt sind. Die Prozesse für diesen Tenant, die die Aufgaben der CPU-VPs übernehmen, erhalten einen individuellen Namen und sind als „oninit-Prozesse“ im Betriebssystem zu finden. Im „onstat -g glo sind diese mit ihrer Prozess-ID zu finden. Diese CPU-VPs der Tenants werden jedoch erst gestartet, sobald sich eine Session mit dem Tenant verbindet.
Der folgende „onstat“ zeigt die Prozesse, die zu unseren Tenants gehören als t1cpu und t2cpu.

Tenants können VPs gemeinsam nutzen (selber Namen beim der zugeordneten VPs), um jedoch für die Abrechnung die Ressourcen eindeutig zuordnen zu können, wird empfohlen je Tenant separate VPs zu definieren.

onstat -g glo:

```
Virtual processor summary:
 class        vps        usercpu    syscpu     total
 cpu          1          1.09       0.26       1.35
 aio          3          0.02       0.16       0.18
 lio          1          0.00       0.05       0.05
 pio          1          0.01       0.04       0.05
 adm          1          0.09       0.12       0.21
 soc          1          0.25       0.31       0.56
 msc          1          0.00       0.00       0.00
 fifo         1          0.02       0.03       0.05
 kalu_vp      2          0.06       0.06       0.12
 t1cpu        2          0.00       0.00       0.00
 t2cpu        2          0.00       0.00       0.00
 total        16         1.54       1.03       2.57

Individual virtual processors:
 vp     pid        class        usercpu    syscpu     total      Thread     Eff
 1      2288       cpu          1.09       0.26       1.35       5.21       25%
 2      2299       adm          0.09       0.12       0.21       0.00        0%
 3      2300       lio          0.00       0.05       0.05       0.05      100%
 4      2302       pio          0.01       0.04       0.05       0.05      100%
 5      2303       aio          0.00       0.07       0.07       0.59       11%
 6      2308       msc          0.00       0.00       0.00       0.00        0%
 7      2318       fifo         0.02       0.03       0.05       0.05      100%
 8      2319       kalu_vp      0.04       0.02       0.06       0.00        0%
 9      2320       kalu_vp      0.02       0.04       0.06       0.00        0%
 10     2322       soc          0.25       0.31       0.56       NA          NA
 11     2323       aio          0.01       0.05       0.06       0.06      100%
 12     2324       aio          0.01       0.04       0.05       0.05      100%
 13     12565      t1cpu        0.00       0.00       0.00       0.17        0%
 14     12566      t1cpu        0.00       0.00       0.00       0.00        0%
 15     12578      t2cpu        0.00       0.00       0.00       0.17        0%
 16     12580      t2cpu        0.00       0.00       0.00       0.00        0%
                   tot          1.54       1.03       2.57
```

Nach der Erstellung von Tenants können die Informationen über diese in der Datenbank „sysadmin“ abgefragt werden. Die Definition des Tenant ist als BSON-Struktur abgelegt, die durch die Umwandlung in JSON lesbar wird:

```
database sysadmin;
select tenant_dbsname,
        tenant_create_time,
        tenant_last_updated,
        tenant_resources::json as info
from tenant
order by 1
```

Das Ergebnis im Beispiel:

```
tenant_dbsname      ifx_ten1
tenant_create_time  2016-01-15 18:54:48
tenant_last_updat+  2016-01-15 18:54:48
info
{"dbspace":"ten1data","sbspace":"ten1sb","vpclass":"t1cpu,num=2",
             "dbspacetemp":"ten1tmp","session_limit_locks":"1000",
             "session_limit_memory":"100mb",
             "session_limit_tempspace":"25mb",
             "session_limit_logspace":"30mb",
             "session_limit_txn_time":"120",
             "tenant_limit_space":"2gb","tenant_limit_memory":"200mb",
             "tenant_limit_connections":"2","logmode":"unbuffered",
             "locale":"en_us.utf8","case":"insensitive"}

tenant_dbsname      ifx_ten2
tenant_create_time  2016-01-15 18:54:59
tenant_last_updat+  2016-01-15 19:33:52
info
{"dbspace":"ten2data","sbspace":"ten2sb","vpclass":"t2cpu,num=2",
             "dbspacetemp":"ten2tmp","session_limit_locks":"1000",
             " session_limit_memory":"100mb",
             "session_limit_tempspace":"25mb",
             "session_limit_logspace":"30mb",
             "session_limit_txn_time":"120",
             "tenant_limit_space":2097152,"tenant_limit_memory":"200mb",
             "tenant_limit_connections":"2","logmode":"unbuffered",
             "locale":"en_us.utf8","case":"sensitive",
             "tenant_state": "ok"}
```

Anmerkung: Der „tenant_state“ wird erst gesetzt, wenn es Änderungen am Tenant gegeben hat.

Für die Hochverfügbarkeit und Replikation gelten bei Tenants folgende Regeln:

- Tenants können sowohl mit der Enterprise Replikation (CDR), als auch mit SDS, HDR oder RSS genutzt werden.
- Tenants können bei der Enterprise Replikation im Grid erstellt und verändert werden.
- Die Erstellung und Änderung von Tenants von einem „updatable secondary“ ist nicht möglich.

TechTipp: Tenant Level Restore

NEU in 12.10.xC6 ist das Feature, dass ein Tenant aus einem binären Backup separat restored werden kann, ohne dass die restliche Instanz, bzw. andere Tenants davon etwas mitbekommen. In unserem Testaufbau (aus dem vorhergehenden Artikel) haben wir zwei Tenants erstellt. Nun erstellen wir in jedem dieser Tenants jeweils eine Tabelle, die neben Werten auch die Timestamp speichert, wann die Werte eingefügt wurden.

Beispiel:

```
database ifx_ten1;
drop table if exists tab1;
create table tab1 (
      f1    int,
      f2    char(42),
      f3    datetime year to second default current year to second
);

database ifx_ten2;
drop table if exists tab2;
create table tab2 (
      f1    int,
      f2    char(42),
      f3    datetime year to second default current year to second
);
```

Voraussetzung ist, dass der Tenant, der wiederhergestellt werden soll, mit Logging erstellt wurde. Ansonsten ist kein „Point-In-Time“ restore möglich.

Für einen Restore muss eine mit „onbar“ erstellte Sicherung zur Verfügung stehen. Dabei muss keine Rücksicht auf die darin definierten Tenants genommen werden:

```
      onbar -b -w -L 0
```

In unserem Beispiel nutzen wir als Storagemanager den PSM, der mit der Instanz mitgeliefert wird.

Die Logs müssen ebenfalls mit onbar gesichert und für den Restore zugreifbar sein.

Wir fügen nun von Zeit zu Zeit Werte in die Tabellen ein mit Befehlen wie den folgenden:

```
database ifx_ten1;
insert into tab1 (f1,f2) values (42,“Antwort“);

database ifx_ten2;
insert into tab2 (f1,f2) values (21,“Halbe Antwort“);
```

Wir nehmen nun an, dass nach einiger Zeit in Tenant ifx_ten2 Werte eingetragen wurden, die wir nicht dort haben wollten. Die Anforderung lautet somit: „**Bringe die Datenbank ifx_ten2 auf den Stand von 19:30, da danach ungewollte Werte eingefügt oder Veränderungen vorgenommen wurden.“**

Ein Restore der gesamten Instanz auf diesen Zeitpunkt kann nicht erfolgen, da die Anwender auf Tenant1 ihre erfassten Daten behalten wollen.

Die Anforderung kann mittels eines Tenent-Level-Restore erfüllt werden, bei dem nur die Komponenten dieses Tenants zurückgesichert werden, alle anderen Tenants und die restliche Instanz bleiben dabei auf dem aktuellen Stand und können weiter arbeiten.

Der Aufruf hierzu lautet:

onbar -r -T ifx_ten2 -t "2016-01-15 19:30:00" -O

Der neue Parameter „-T“ steht für Tenant. Die Option „-O“ muss genutzt werden, so lange die zurückzusichernden DBSpaces ONLINE sind, damit ein „warm restore“ ausgeführt werden kann.

Als Voraussetzung muss eine Sicherung vorhanden und lesbar sein, die Pfade zu den DBSpaces des Tenant müssen zur Verfügung stehen, es darf keine Sicherung und kein anderer Restore aktiv sein.

Als Einschränkung gilt, dass der Server kein SDS-Server sein darf und auch keine Secondary Server definiert sein dürfen, die „updatable“ sind.
Falls Read-Only Secondary Server definiert sind, dann müssen diese aktiv und verbunden sein. Der Restore kann nur auf dem Primary Server erfolgen.

Abschliessend können wir mittels SQL abfragen, ob der Restore auch wirklich den gewünschten Stand hergestellt hat:

```
database ifx_ten1;
select max(t1.f3)
from ifx_ten1:tab1 as t1;

database ifx_ten2;
select max(t2.f3)
from ifx_ten2:tab2 as t2;
```

TechTipp: Task Funktionen (modify chunk extendable [off])

Mittels dieses Aufrufs kann die Option zur automatischen Erweiterung von DBSpaces bzw. Chunks [de]aktiviert werden.
Voraussetzung ist, dass der Benutzer, der den Task aufruft, Rechte auf der Datenbank „sysadmin“ besitzt. Üblicherweise werden die Befehle als Benutzer „informix“ ausgeführt, da es sich um administrative Massnahmen der Instanz handelt.

Mit dem Befehl:

```
database sysadmin;
execute function task ("modify chunk extendable" ,8);
```

wird für den Chunk 8 (Chunknummer wird im „onstat -d“ angezeigt) die automatische Erweiterung aktiviert.
Als Ausgabe wird zurückgegeben:

```
(expression)  Chunk 8 is now extendable.
```

Diese Aktion kann rückgängig gemacht werden, indem man folgenden Task aufruft:

```
database sysadmin;
execute function task ("modify chunk extendable off" ,8);

(expression)  Chunk 8 is no longer extendable.
```

Der Defaultwert für die Erweiterung ist 10 MB, was für die meisten Systeme deutlich zu klein erscheint. Zudem ist im Default keine Grenze angegeben, bis zu welchem Limit sich die Chunks des DBSpace erweitern dürfen. Diese Werte können mit den Aufrufen im folgenden Artikel angepasst werden.

TechTipp: Task Funktionen (modify space sp_sizes)

Mittels „modify space sp_sizes“ kann sowohl die Extend Grösse, als auch die Maximalgrösse der Chunks eines DBSpace angepasst werden.
Dies erfolgt z.B. mit dem Aufruf:
database sysadmin;

```
execute function task
      ("modify space sp_sizes","datadbs1",0,142000,42000000);

(expression)  Succeeded: Create size changed to 0,
      Extend size changed to 142000, Max size changed to 42000000,
      for DBspace datadbs1.
```

Als Parameter werden die initiale Grösse eines neuen Chunks, die Extend Size bei automatischer Erweiterung, und die maximale Grösse (Angaben jeweils in kB) angegeben. Im Beispiel wurden 142 MB als Grösse zur Erweiterung und 42 GB als maximale Grösse angegeben.

Alternativ können als Werte auch Prozentzahlen angegeben werden. Hierbei werden Werte zwischen 0 und 100 als Prozent der Grösse des DBSpace angenommen.
Beispiel:

```
EXECUTE FUNCTION task("modify space sp_sizes","datadbs","12","1.2");
```

Legt die Grösse für neue, automatisch erstellte Chunks auf 12% des bestehenden DBSpaces fest. Die Extentsize für die automatische Erweiterung wird auf 1,2% des DBSpaces gesetzt.
Die Einstellungen für die Erweiterung gelten für alle Chunks eines DBSpaces und sind nicht je Chunk änderbar.

TechTipp: Task Funktionen (modify chunk extend)

Chunks können nicht nur automatisch, sondern auch manuell erweitert werden. Dies erfolgt mittels eines Task-Aufrufes. Voraussetzung ist, dass der Chunk erweiterbar ist (im „onstat -d“ ist dies zu sehen, wenn das Flag „E“ für „extendable“ gesetzt ist).

Ein SQL-Script um solch eine Erweiterung vorzunehmen könnte z.B. folgenden Ablauf haben:
Zuerst wird der Chunk (im Beispiel die Nummer 42) für Erweiterungen freigegeben, dann wird die Erweiterung durchführt und anschliessend wird die Option der Erweiterung wieder deaktiviert, falls keine automatische Erweiterung gewünscht ist:

```
EXECUTE FUNCTION task("modify chunk extendable on", 42);
EXECUTE FUNCTION task("modify chunk extend", 42, "2 GB");
EXECUTE FUNCTION task("modify chunk extendable off", 42);
```

TechTipp: SMI - sysdbstab (ExtendSize, MaxSize von Spaces)

In der Datenbank „sysmaster“ können die gesetzten Werte für die Erweiterung und die maximale Grösse der Chunks eines DBSpaces abgefragt werden. Diese stehen in der Tabelle „sysdbstab“.

Beispiel:

```
select t.dbsnum::char(3) as dnum, c.chknum::char(3) as cnum,
            c.is_extendable::char(1) as is_ext,
            n.name[1,20],
            t.extend_size::int as extend_kb,
            t.max_size as max_mb
 from sysmaster:sysdbstab t, sysmaster:sysdbspaces n, sysmaster:syschunks c
where t.dbsnum = n.dbsnum
and t.dbsnum = c.dbsnum
order by t.dbsnum,c.chknum
```

Ergebnis:

```
dnum cnum is_ext name                 extend_kb             max_mb

1    1    0      rootdbs                 10000                   0
2    2    0      logdbs                  10000                   0
3    3    0      datadbs                 10000                   0
4    4    0      sbdbs                       0                   0
5    5    0      tmpdbs                  10000                   0
6    6    0      tsdbs                   10000                   0
7    7    0      datadbs2                10000                   0
8    8    1      datadbs1               142000               41015
```

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

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

Sowie unterstützende Teams im Hintergrund.

Fotonachweis: Gerd Kaluzinski (Schnee in Lindau)