Willkommen zum Informix Newsletter der IUG Deutschland

Inhaltsverzeichnis

Aktuelles....1
TechTipp: Environment - CLIENT_LABEL....2
TechTipp: Enterprise Replication - Einrichtung mit „cdr migrate server“....4
TechTipp: Fragmentation / Partitioning – Auswirkung auf Indexe....7
TechTipp: DBMS_LOB_COMPARE....8
TechTipp: DBMS_LOB_APPEND....9
TechTipp: DBMS_LOB_INSTR....9
TechTipp: INFORMIX-HQ – Monitoring Nachfolger des OAT....10
TechTipp: onrestorept – Restore a failed upgrade....14
TechTipp: ONCONFIG - CONVERSION_GUARD....14
TechTipp: ONCONFIG - RESTORE_POINT_DIR....15
TechTipp: Sicherheitslücke im Spatial Datablade 8.22.xx....15
TechTipp: Ändern der Informix Edition mittels JAR File....16
Nutzung des INFORMIX Newsletters....17
Die Autoren dieser Ausgabe....17

Aktuelles

Liebe Leserinnen und Leser,

nachdem das Jahr 2020 für uns Alle ein Ausnahmejahr mit unerwarteten Wendungen und Hindernissen war, hoffen wir, dass Sie den Jahresausklang in Ruhe mit Ihren Lieben verbringen können.
Ein Spaziergang im Schnee, ein Glühwein bei Kerzenlicht, fröhliche Unterhaltung, … all das ist eine wohltuende Abwechslung zum hektischen Alltag, und eine sehr gute Alternative zu den vollen Weihnachtsmärkten, die in diesem Jahr ohnehin kaum oder nur sehr eingeschränkt stattfinden können.
In diesem Sinne wünscht Ihnen das Team des Informix Newsletters eine besinnliche Zeit um Kraft zu schöpfen für die Herausforderungen, die uns in 2021 erwarten.

Bleiben Sie gesund !

Ihr TechTeam

TechTipp: Environments - CLIENT_LABEL

In vielen Applikationen werden die individuellen Benutzer durch technische Benutzer ersetzt, so dass in der Datenbank immer nur Namen der technischen Benutzer erscheinen, was eine Rückverfolgung zum einzelnen, tatsächlichen Benutzer erschwert.

Über die Umgebungsvariable „CLIENT_LABEL“ ist es nun möglich, einem Client einen Herkunfts-Stempel zu verpassen, ohne in die Applikation einzugreifen und ohne den Benutzernamen explizit verwenden zu müssen.

Hierzu wird vor dem Aufruf der Applikation die Umgebungsvariable CLIENT_LABEL gesetzt. Dies kann z.B. in der Shell mittels

```
export CLIENT_LABEL="Kalu_Test"
```

erfolgen.

Gibt es die Möglichkeit, einen SQL-Befehl abzusetzen, so kann der Wert auch mittels

```
set environment CLIENT_LABEL "Kalu_Test";
```

gesetzt werden.

Das CLIENT_LABEL ist am Datenbankserver als eine der Umgebungsvariablen des Clients zu sehen:

```
$ onstat -g env 42

IBM Informix Dynamic Server Version 14.10.FC4W1DE -- On-Line -- Up
00:15:17 -- 181252 Kbytes

Environment for session 42:

Variable            Value [values-list]
CLIENT_LABEL        Kalu_Test
CLIENT_LOCALE       en_us.utf8
CLNT_PAM_CAPABLE    1
DBDATE              DMY4.
DBDELIMITER         |
DBLANG              en_us.utf8
DBMONEY             ,
DBPATH              .
                    [.]
                    [//kalu]
DBPRINT             lp -s
DBTEMP              /tmp
DB_LOCALE           en_us.utf8
```

Diese Umgebungsvariable ist auch in der Tabelle sysenvses zu sehen:

```
sysmaster:sysenvses
envses_sid    42
envses_id     13
envses_name   CLIENT_LABEL
envses_value  Kalu_Test
```

Mit Hilfe des Labels können nun weitere Informationen zur Datenbankverbindung abgefragt werden, wie z.B. das Session Profile:

```
select *
from sysmaster:syssesprof
where sid = (
        select envses_sid
        from sysenvses
        where envses_name = "CLIENT_LABEL"
        and envses_value = "Kalu_Test"
);
```

oder die Sperren, die gerade von diesem Benmutzer gehalten werden:

```
select *
from sysmaster:syslocks
where owner = (
        select envses_sid
        from sysenvses
        where envses_name = "CLIENT_LABEL"
        and envses_value = "Kalu_Test"
)
;
```

Ab Version C-SDK 4.10.xC10 und JDBC 4.10.JC10 kann die Umgebungsvariable CLIENT_LABEL auf den entsprechenden Clients gesetzt werden, um am Server sichtbar zu sein.

TechTipp: Enterprise Replication - Einrichtung mit „cdr migrate server“

Zur Einrichtung der Enterprise Replikation ist üblicherweise eine Vielzahl an Befehlen in der richtigen Reihenfolge notwendig.
Dazu zählen die Definition der CDR Server, die Erstellung der Replikate, das Hinzufügen der CDR Keys wo kein Primary Keys vorhanden ist, sowie die Übertragung der Daten von der Quelle auf den Zielrechner.
All diese Befehle können nun mittels „cdr migrate server“ automatisch erstellt werden.
Die anschliessende Ausführung der Befehle in der richtigen Reihenfolge ist ebenfalls vom Quellserver aus möglich.

Der Befehl, der dies ermöglicht, lautet „cdr migrate“:

Die Syntax:

```
cdr migrate server -s source -t target -p phase [-d database] [--exec]
-s --source Source server name
-t --target Target server name
-p --phase migration phase
-e --exec Execute commands. Default: print only.
-d --database Database to replicate
-T --receiveonly Setup oneway replication from source to
      target ('create_replicates' phase)
-r --checkrepair Synchronize data using 'cdr check' instead
      of 'cdr sync'('sync_data' phase)
-g --grid Y/N Enable/Disable grid. Default:Y
      ('create_replicates' phase)
-A --atsdir ATS files directory path('define_er' phase)
-R --risdir ATS files directory path('define_er' phase)
-P --parallelsync Number of table sync jobs to run
      in parallel('sync_data' phase)
-f --outfile <file path> Output file for commands
-E --exclude db:owner.tab specify table to exclude
```

Beispiele:
Um die Befehle auszugeben, die für das Aufsetzen notwendig sind, reicht der Befehl:

cdr migrate server -s ifx_source -t ifx_target --phase all

Diese Befehle können zudem ausgeführt werden:

cdr migrate server -s ifx_source -t ifx_target --phase all --exec

Es ist auch möglich die Befehle einzelner Phasen auszugeben bzw. auszuführen.
Bei der Ausführung ist auf die Reihenfolge zu achten.

cdr migrate server -s ifx_source -t ifx_target --phase define_er --atsdir=/work --risdir=/work

Die möglichen Phasen sind:

1. **define_er**: Definition der Enterprise Replikation
2. **add_erkey**: Die Spalte ERKEY im Source Schema hinzufügen
3. **create_spaces**: CDR DBSpaces erstellen
4. **create_schema_loaddata**: Schemata erstellen und Daten laden
    - Dies erfolgt in den Schritten:
        - Schritt 1:
            - Migration der Tabellen Schemata ohne Indexe, Constraints und Primary Keys
            - Erstellen aller Tabellen als “RAW” Tabellen
        - Schritt 2:
            - Starten von vielen parallelen Jobs für data load und index builds
            - Jeder Job führt aus:
                Sicherstellen, dass die Tabelle im Modus RAW ist.
                Laden der Daten mit “insert into … select * from …” als verteilte Abfrage über die Server hinweg.
                Erstellen der Indexe, primary key und unique key constraints
        - Schritt 3:
            - Erstellen der Referential Constraints
5. **create_replicates**: Erstellen der Replikate, Replikat-Sets und Grid-Definitionen
6. **sync_data**: Synchronisieren der Daten
7. **all**: Alle Phasen ausführen
8. **static**: Ausführen der Phasen 3 und 4
9. **dynamic**: Alle Phasen ausführen

Voraussetzung ist, dass der Quell- und der Zielserver sich gegenseitig erreichen können, und beide Informix Instanzen gegenseitig in der Datei $INFORMIXDIR/etc/sqlhosts mit Gruppen eingetragen sind.

Beispiel (sqlhosts):

```
r_1          group       -          -       i=41
ifx_source   onsoctcp    127.0.1.1  9061    g=r_1
r_2          group       -          -       i=42
ifx_target   onsoctcp    127.0.1.1  9062    g=r_2
```

Der eigentliche Aufruf:

```
cdr migrate server -s ifx_source -t ifx_target -p all -d stores
```

Die Ausgaben enthalten dann alle Befehle, die zum Aufsetzen der Enterprise Replikation für die Datenbank „stores“ zwischen den Servern ifx_source und ifx_target notwendig sind:

```
The version of the server you are using has full ERKEY support.
WARNING:  CDR_SERIAL value on rep1 can cause collisions.
 #--
 #--  Configuration changes
 #--
cdr change config -c r_1 "CDR_QUEUEMEM 262144"
cdr change config -c r_2 "CDR_QUEUEMEM 262144"
cdr change config -c r_1 "CDR_SUPPRESS_ATSRISWARN 1,2,3"
cdr change config -c r_2 "CDR_SUPPRESS_ATSRISWARN 1,2,3"
```

```
 #--
 #--  Define ER for the first time.
 #--
cdr define serv -c r_1  -I r_1
cdr define serv -c r_2  -I r_2 -S r_1

 #--
 #--  Creating Replication Key
 #--
dbaccess - - <<EOF
database stores@ifx_source;
alter table 'informix'.classes add ERKEY;
alter table 'informix'.employee add ERKEY;
alter table 'informix'.tab add ERKEY;
...
```

In den Scripts sind einige Aufrufe von „SLEEP“ enthalten, damit z.B. die Queues leer werden, bevor der nächste Schritt vorgenommen wird.
Ohne weitere Aufrufe wurde die Replikation eingerichtet und gestartet:

```
cdr list server

SERVER                 ID STATE    STATUS     QUEUE  CONNECTION CHANGED
-----------------------------------------------------------------------
r_1                    61 Active   Local          0
r_2                    62 Active   Connected      0 Sep 16 12:19:34

cdr list repl brief

REPLICATE            TABLE                                        SELECT
-----------------------------------------------------------------------------
--
ifx_migrate_grid_copy syscdr@r_2:informix.grid_copy            select *
from grid_copy where gcpy_gridid = 3997698
ifx_migrate_grid_copy syscdr@r_1:informix.grid_copy            select *
from grid_copy where gcpy_gridid = 3997698
ifx_migrate_1_1600251323_call_type stores@r_1:informix.call_type
select t.call_code, t.code_descr from 'informix'.call_type t
ifx_migrate_1_1600251323_call_type stores@r_2:informix.call_type
select t.call_code, t.code_descr from 'informix'.call_type t
ifx_migrate_2_1600251323_catalog stores@r_1:informix.catalog
select t.catalog_num, t.stock_num, t.manu_code, t.cat_descr,
t.cat_picture, t.cat_advert from 'informix'.catalog t
...
```

In der Syntax ist zu sehen, dass es zudem die Optionen gibt, um einzelne Tabellen von der Replikation auszuschliessen und die Parallelität der Datensynchronisation zu steuern.

TechTipp: Fragmentation / Partitioning – Auswirkung auf Indexe

Wird beim „create index“ keine explizite Fragmentierungsstrategie angegeben, so folgt die Fragmentierung des Index der Fragmentierung der Tabelle.
Dies ist in der Tabelle sysfragments beim Eintrag für den Index im Feld „strategy“ als Flag „T“ zu erkennen.

Das Feld „strategy“ kann folgende Werte beinhalten:

- R = Round-robin distribution strategy
- E = Expression-based distribution strategy
- I = IN DBSPACE clause specifies a storage location as part of distribution strategy
- N = raNge-iNterval (or rolliNg wiNdow) distribution strategy
- L = List distribution strategy
- T = Table-based distribution strategy
- H = table is a subtable within a table Hierarchy

Sollen Fragmente mittels ATTACH oder DETACH zur Tabelle hinzugefügt oder entfernt werden, so entscheidet die Index Strategie massgeblich über die Performance.
Sind die Indexe nicht analog der Tabelle fragmentiert, so müssen alle Indexe der Tabelle neu erstellt werden, sobald sich etwas an der Fragmentierung ändert.
Optimale Performance beim Attach oder Detach bedingt somit, dass die Indexe identisch zur Fragmentierung der Tabelle fragmentiert wurden.

Beispiel:

```
create table if not exists xx (
f1 int,
f2 char(8)
) fragment by expression
f1 <= 5 in rootdbs,
f1 > 5 in datadbs
;

create index if not exists xx_i1 on xx(f1);
create index if not exists xx_i2 on xx(f1,f2) fragment by
expression
f1 <= 5 in rootdbs,
f1 > 5 in datadbs
;
```

Die beiden Indexe scheinen (falls der erste Index xx_i1 der Fragmentierungsstrategie der Tabelle folgt) gleich fragmentiert zu sein, da die identischen Regeln beim Index wie bei der Tabelle angewandt wurden. Betrachtet man wie die Fragmentierung in der Datenbank gespeichert wurde, so zeigen sich wichtige Unterschiede.

Obwohl die Abfrage der Fragmentierungsstrategie für beide Indexe identische Ausdrücke anzeigt, wird nur die Strategie beim Index xx_i1 mit „T“ (follow Table) angegeben, beim Index xx_i2 jedoch mit „E“ für Expression.
Beim ATTACH/DETACH von Fragmenten auf der Tabelle wird somit der zweite Index immer neu aufgebaut, nur der erste Index kann bezüglich der unveränderten Fragmente beibehalten werden.

```
select *
from sysfragments
where tabid = (select tabid from systables where tabname = 'xx')
```

Ergebnis:

```
fragtype indexname  strategy exprtext

I       xx_i1       T        (f1 <= 5
I       xx_i1       T        (f1 > 5
I       xx_i2       E        (f1 <= 5
I       xx_i2       E        (f1 > 5
T                   E        (f1 <= 5
T                   E        (f1 > 5
```

TechTipp: DBMS_LOB_COMPARE

Das DBMS_LOB Package beinhaltet einige Funktionen, um mit BLOB und CLOB Daten besser umgehen zu können. Die Funktionen DBMS_LOB_GETLENGTH() und DBMS_LOB_SUBSTR_() wurden bereits im INFORMIX Newsletter Q3/2017 vorgestellt.

Bevor die Funktionen zur Verfügung stehen, muss das Datablade in der Datenbank registriert werden. Dies kann entweder über die Funktion:

```
EXECUTE FUNCTION sysbldprepare('excompat.*', 'create');
```

erfolgen oder über den BladeManager und die Registrierung der Extension excompat.1.0.

Die Funktion DBMS_LOB_COMPARE ist ein nützliches Hilfsmittel, um Objekte vom Typ BLOB oder CLOB zu vergleichen. Zum Beispiel kann so recht einfach ein neues Dokument in die Datenbank geladen werden, falls dies noch nicht gespeichert ist. Ansonsten wird es als Duplikat erkannt, unabhängig davon wie der Name der Quelldatei vor dem Laden war.

```
select dbms_lob_compare (f4.txt,f3.txt)
   from  features f4, features f3
   where f4.version = '14.10.xC4'
   and f3.version = '14.10.xC3'
;
```

Stimmen die Dateien überein, so wird der Wert 0 zurückgegeben. Gibt es Abweichungen ist der Rückgabewert -1.

TechTipp: DBMS_LOB_APPEND

Die Funktion DBMS_LOB_APPEND ermöglicht die Verknüpfung von zwei Objekten. Beide Objekte müssen vom selben Typ (CLOB oder BLOB) sein. Da das zweite Argument an das erste Argument angehängt wird, ist dieses sowohl Input- als auch Output Parameter und es muss eine Hilfsprozedur verwendet werden.

Das folgende Beispiel zeigt, wie das Textfeld (CLOB) der Features der Version 14.10. FC4 um den Text der Features der Version 14.10.FC3 erweitert wird. Die Rückgabe ist im Beispiel das CLOB, das beide Informationen beinhaltet.

```
create procedure kalu_append(x1 clob, x2 clob)
returning clob

define x3 clob;;
let x3 = x1;;
execute procedure dbms_lob_append(x3,x2);;
return x3;;
end procedure;
Routine created.

;

execute procedure kalu_append (
   (select txt from features where version = '14.10.xC4'),
   (select txt from features where version = '14.10.xC3')
)
;
```

TechTipp: DBMS_LOB_INSTR

Die Funktion DBMS_LOB_INSTR() ermittelt, ob sich ein gesuchter Text in einem BLOB oder CLOB finden lässt. Der Suchstring ist ein BLOB, CLOB oder LVARCHAR. Die Suche erfolgt mittels Pattern Vergleich, daher werden auch Übereinstimmungen von Bildsequenzen oder z.B. ähnliche Passagen in Musikstücken gefunden.
Optional kann ein OFFSET angegeben werden, ab dem die Suche starten soll (Default ist der Beginn des Objekts), sowie eine Zahl, die angibt, welches von mehreren Vorkommen eines Pattern als Position ausgegeben werden soll (Default ist 1 für das erste Vorkommen).
Der Rückgabewert ist beim CLOB die Anzahl der Zeichen bis zum Vorkommen, bei BLOB die Anzahl der Bytes bis zum Treffer.

Beispiel:
```
select dbms_lob_instr (txt,"ODBC")
from  features
where version = '14.10.xC4‘;

(expression)
    153
```

TechTipp: INFORMIX-HQ – Monitoring Nachfolger des OAT

Das Monitoring Tool INFORMIX-HQ wird seit Version 12.10.xC6 mit dem Server mitgeliefert und (falls es nicht abgewählt wurde) mit installiert. Als Verzeichnis wird $INFORMIXDIR/hq verwendet.

Für das Setup muss in einer Properties-Datei lediglich das initiale Passwort gesetzt werden, bevor der Dienst gestartet werden kann.

HINWEIS: Das Passwort muss mindestens 8 Zeichen lang sein, sowohl Grossbuchstaben, Kleinbuchstaben als auch mindestens eine Zahl enthalten.

Der Default Port ist 8080, ist jedoch frei konfigurierbar.

Der Start erfolgt mittels:

java Dfile.encoding=utf8 -jar informixhq-server.jar informixhq-server.properties

Danach kann INFORMIX-HQ im Browser unter der Adresse http://<hostname>:<port> aufgerufen werden. In der Konfigurationsdatei gibt es die Option, den Dienst über SSL (https://<hostname>:<port>) anzubieten.

Für die Erfassung der Kennwerte sind „Agents“ notwendig. Diese werden ebenfalls im Bereich „$INFORMIXDIR/hq“ mittels einer Properties Datei (informixhq-agent.properties) konfiguriert und über die Angabe einer eindeutigen ID, sowie des Hosts und Ports des HQ-Servers mit diesem verbunden. Im Gegensatz zum bisherigen OAT ist es damit möglich, Agents auf allen Informix Servern mit einem HQ Server zu überwachen.

Der Start des Agents erfolgt mittels:

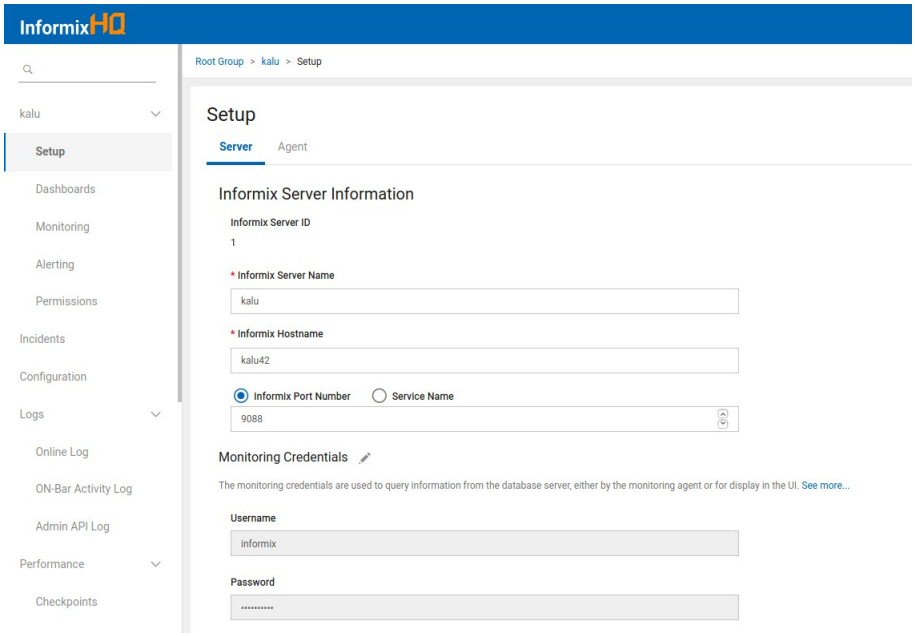
java Dfile.encoding=utf8 -jar informixhq-agent.jar informixhq-agent.properties

Der nächste Schritt erfolgt im Browser. Zuerst müssen die Informix Server unter
„Root Group“ → „Servers“
mit „Add Server“ eingetragen werden.

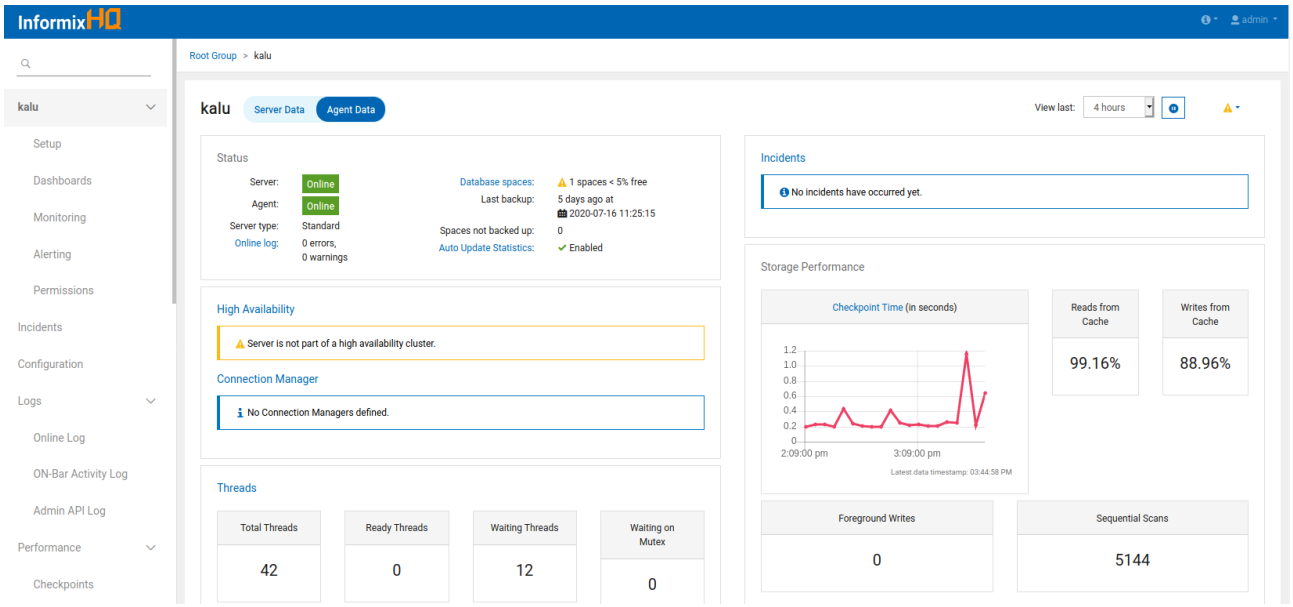
Diese Server stehen dann für die Administration zur Verfügung.
Die weitere Angaben zu den Benutzern für „Monitoring“ und „Administration“ werden im SETUP eingetragen.
Zur Speicherung der Daten, welche die Agents sammeln, muss eine Datenbank ausgewählt werden. Hier empfiehlt es sich eine neue, separate Datenbank zu erstellen (z.B. mit dem Namen „informix_hq“), um die Daten nicht mit den Daten einer produktiven Datenbank zu mischen. Optional kann hierfür eine eigene Instanz auf einem anderen Rechner (in einer anderen Umgebung als der Produktion) erstellt werden.

Im Bereich „Monitoring“ können verschiedene, vorgefertigte Sensoren aktiviert werden. Diese werden dann von den Agents abgerufen und die Daten in der angegebenen Datenbank gespeichert.

Nachfolgend einige Snapshots der Informix HQ Oberflächen:

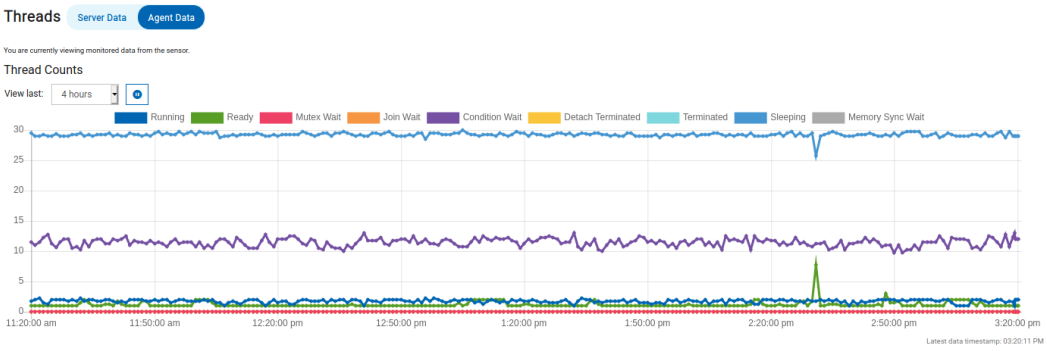


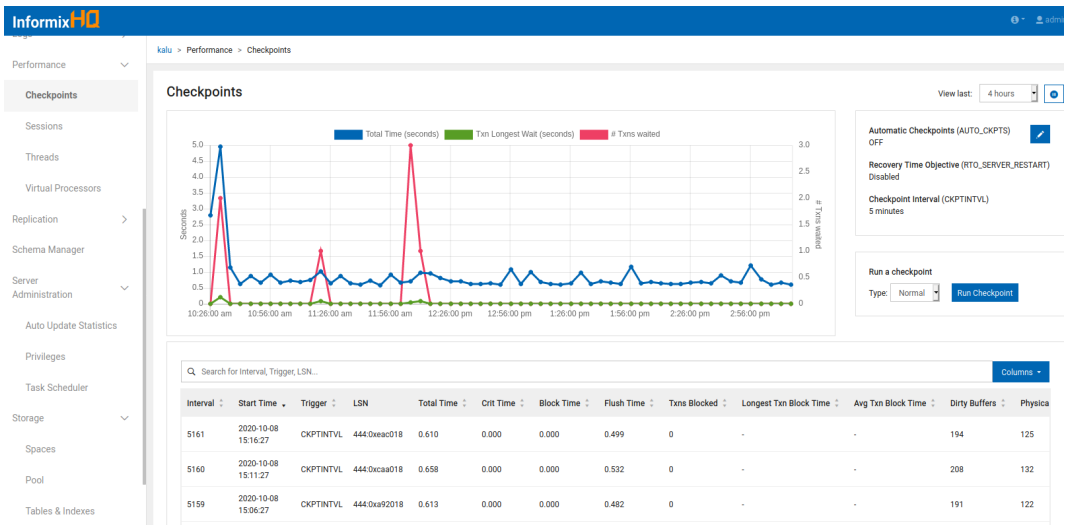
Auf der Übersichtsseite ist dann der aktuelle Informix Server zu sehen:

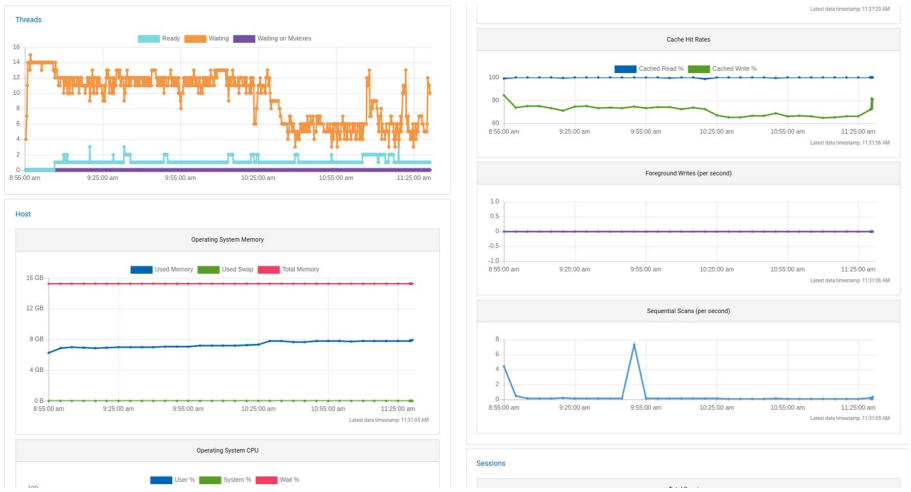


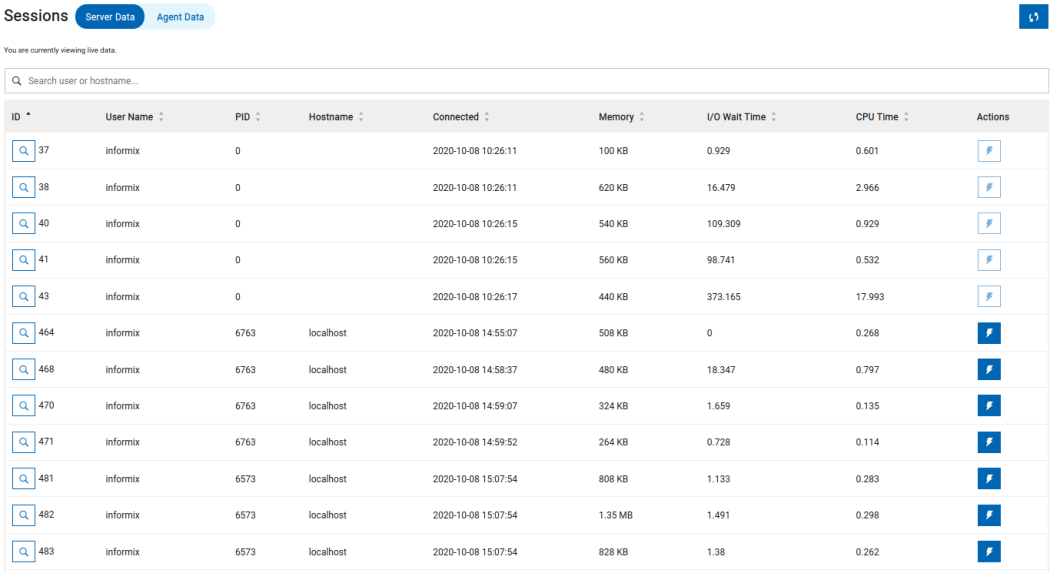
Inzwischen ist INFORMIX HQ so umfangreich, dass der Umstieg vom gewohnten Open AdminTool nicht mehr schwer fällt.
Einziger Nachteil ist, dass die eigenen Erweiterungen zum OAT hier nicht mehr eingebunden werden können und ggf. neu geschrieben werden müssen.
Eine Erweiterung um eigene Funktionen ist derzeit nicht so einfach wie mit „plugin_install“ beim OAT möglich.

Hier noch einige Snapshots, die ggf. den Umstieg schmackhaft machen:

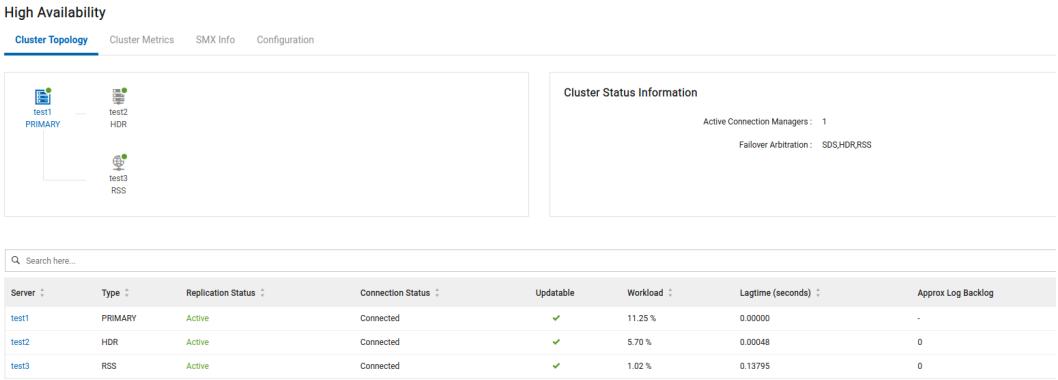








Auch der Bereich des Connection Managers und der Hochverfügbarkeit ist übersichtlich dargestellt:



Tipp: Wer (noch) nicht vom alten OAT los kommt, der kann derzeit noch in einem separaten Verzeichnis das C-SDK 4.10 installieren und den OAT daraus nutzen. Dieser ist mit den Features des Servers in Version 14.10 kompatibel.

TechTipp: onrestorept – Restore a failed upgrade

Sogenannte „InPlace“ Migrationen sind bei INFORMIX üblich. Wer die wichtigsten Regeln befolgt und dafür sorgt, dass alle logischen Logs gesichert sind, alle Checks fehlerfrei durch laufen und die alte Instanz geordnet herunter fährt, der wird in den meisten Fällen eine schnelle und problemlose „InPlace“ Migration erleben (Ggf. muss die Datenbank sysadmin separat behandelt werden).

Sollte es doch einmal zu einem Problem bei der Migration kommen, so ist es wichtig, schnell wieder zu einem laufenden System zu kommen.

Eine geprüfte Sicherung vor dem Start der Migration zu erstellen, ist dabei nie falsch (und entspricht ohnehin der notwendigen Vorsicht auf Produktionssystemen).
Schneller als ein Restore, um wieder auf die alte Version zu gelangen ist die Option „onrestorept“, die darauf basiert, dass alle Änderungen während der Migration in einem Verzeichnis abgelegt werden, um diese als Rollback zurück zu fahren.
Wichtige Konfigurationsparmeter hierfür sind

CONVERSION_GUARD und
RESTORE_POINT_DIR

Diese werden in den folgenden beiden Artikeln näher beschrieben.

Der Befehl „restorept“ wird üblicherweise nach einer fehlgeschlagenen Migration auf eine höhere Version ohne Parameter aufgerufen. Nachdem der Befehl beendet ist, kann die bisherige Version wieder gestartet werden, um ggf. Probleme zu beheben (wie z.B. Pending Alter Table).

Vor einer erneuten Migration müssen alle Restore Dateien, die im RESTORE_POINT_DIR durch die Migration angelegt wurden, gelöscht werden.

TechTipp: ONCONFIG - CONVERSION_GUARD

Der Parameter CONVERSION_GUARD steht als Default auf dem Wert 2. Damit ist die Option im Falle eines Fehlers bei der Migration mittels onrestorept zurück zu fahren gegeben.
Allerdings ist der Parameter 2 nicht die sicherste Variante, denn dieser setzt die Migration auch dann fort, wenn z.B. im Verzeichnis, in dem die Restore Points gespeichert werden sollten, nicht mehr genügend Platz ist und damit der Rückweg nicht mehr funktionieren würde. Will man ganz sicher gehen, dass die Migration in jedem Fall bei einem Fehler zurück gesetzt werden kann, so empfiehlt es sich, den Parameter auf 1 zu ändern.

Der Parameter ist nicht dynamisch änderbar, sondern erfordert einen Neustart der Instanz. Da dieser jedoch ohnehin nur für eine Migration auf eine höhere Version benötigt wird, bei der die Instanz neu gestartet wird, stellt dies keine Einschränkung dar.

TechTipp: ONCONFIG - RESTORE_POINT_DIR

Der Parameter RESTORE_POINT_DIR bestimmt das Verzeichnis, in dem die Informationen über einen möglichen Restore einer „InPlace“ Migration gespeichert werden.

Die Dateien werden in einem Unterverzeichnis angelegt falls während der Migration der Parameter CONVERSION_GUARD auf einem Wert ungleich 0 steht.
Als Default wird das Verzeichnis „$INFORMIXDIR/tmp“ verwendet.

Der Parameter kann nur mittels Neustart der Instanz geändert werden, was bei einer Migration ohnehin der Fall ist.

TechTipp: Sicherheitslücke im Spatial Datablade 8.22.xx

Das Spatial Datablade der Version 8.22.xC2, das mit den Versionen 12.10.xC7 und höher ausgeliefert wird, sowie das Spatial Datablade 8.22.xC4, das mit Version 14.10-xCx installiert wird, beinhaltet eine Sicherheitslücke.
Wird der Fehler „parameter out of range“ durch einen lokal am Datenbankserver angemeldeten Benutzer provoziert, so kann dieser mehr Rechte auf dem System erhalten, als ihm zugeordnet wurden. Diese Lücke kann nur ausgenutzt werden, falls dem Benutzer ohnehin Rechte zur Verbindung auf die Datenbank zugeordnet wurden (connect).

Falls das Spatial Datablade nicht genutzt wird, so reicht als Fix ein simples Umbenennen des Spatial Datablades in $INFORMIXDIR/extend von „spatial.8.22.FCx“ auf z.B. „gesperrt.spatial“, um die Ausnutzung dieser Lücke zu vermeiden.

Weitere Informationen sowie ein Interims-Fix stehen bereits auf den Support Pages von IBM zur Verfügung:

https://www.ibm.com/support/pages/node/6343587

TechTipp: Ändern der Informix Edition mittels JAR File

Die Installation von INFORMIX hat sich mit Version 14.10 grundlegend geändert.
Früher wurden unterschiedliche Produkte für die verschiedenen Editionen ausgeliefert.
Dies hatte zur Folge, dass z.B. ein Wechsel von einer Workgroup Edition zu einer Enterprise Edition eine komplette Neuinstallation der Software erforderlich machte. Um kein Risiko einzugehen bezüglich der Änderungen in den verschiedenen Softwareständen, musste dann mühsam der zur eingesetzten Version passende Download gesucht und geladen werden.

Inzwischen wird als Basis die kostenfreie Developer Edition installiert mittels

./ids_install

Wird im Verzeichnis, in dem der ./ids_install aufgerufen wurde, eine Edition-Datei gefunden, so ruft der Installer dieses am Ende der Installation auf.

Alternativ kann nach erfolgreicher Installation die Edition über das Einspielen einer Lizenz in Form einer JAR-Datei festgelegt werden.
Beispiel für eine EnterpriseEdition ee_edition.jar:

```
> $INFORMIXDIR/jvm/jre/bin/java -jar ee_edition.jar -i <console|gui>
```

Dies ist auch ohne Interaktion möglich:

```
> $INFORMIXDIR/jvm/jre/bin/java -jar ee_edition.jar     \
      -DUSER_INSTALL_DIR=$INFORMIXDIR                   \
      -DLICENSE_ACCEPTED=TRUE -i silent
```

Soll nun z.B. ein Wechsel von der Workgroup Edition auf die Advanced Enterprise Edition erfolgen, so ist lediglich das JAR-File aufzurufen, um die Limits in der Edition neu festzulegen.
Derzeit sind folgende Editions möglich:

Advanced Enterprise Edition (aee)
Advanced Developer Edition (ade)
Advanced Enterprise Time Limited Edition (aetl)
Enterprise Edition (ee)
Time Limited Edition (tl)
Workgroup Edition (we)
Express Edition (e)
Innovator-C Edition (ie)
Developer Edition (de)

Nutzung des INFORMIX Newsletters

Die hier veröffentlichten Tipps&Tricks erheben keinen Anspruch auf Vollständigkeit.
Die IUG hat sich dankenswerterweise dazu bereit erklärt, den INFORMIX Newsletter auf ihren Web Seiten zu veröffentlichen.
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.
Rückmeldungen hierzu sind herzlich Willkommen !
Die gefundenen Tippfehler dürfen zudem behalten und nach Belieben weiterverwendet werden.

Eine Weiterverbreitung in eigenem Namen oder eine Bereitstellung auf der eigenen HomePage ist ausdrücklich erlaubt. Alle hier veröffentlichen Scripts stehen uneingeschränkt zur weiteren Verwendung zur Verfügung.

Die Autoren dieser Ausgabe

Gerd Kaluzinski IT-Specialist Informix, DB2, InfoSphere CDC, DataStage
IBM Analytics
gerd.kaluzinski@de.ibm.com +49-175-228-1983

Besonderer Dank geht an Martin Fuerderer der HCL, der zuverlässig die Korrektur und Überprüfung der Inhalte und der Tippfehler übernimmt (ca. 30% der Kommata wurden von ihm gesponsert). Weitere 10% der Kommata stammen von Herrn Möller der IUG.

Nicht zu vergessen der Dank an die Informix User Group, ohne die es keinen Neuanfang des INFORMIX Newsletters gegeben hätte.

Foto Nachweis: Weihnachtsbaum Lindau Reutin 2020 (Gerd Kaluzinski)

Sowie unterstützende Teams im Hintergrund.