Red Hat Enterprise Linux for SAP Solutions 9

障害復旧のための SAP HANA スケールアップマルチターゲットシステムレプリケーションの設定

Last Updated: 2024-02-26

Red Hat Enterprise Linux for SAP Solutions 9 障害復旧のための SAP HANA スケールアップマルチターゲットシステムレプリケーションの設定

法律上の通知

概要

このガイドでは、特に障害復旧シナリオ向けに調整された、スケールアップ設定で HANA System Replication を設定するプロセスの概要を説明します。このガイドでは、複数のサイトにわたる HANA System Replication の実装について、3 つ以上のサイトにまたがる環境に焦点を当てて説明します。

目次

多様性を受け入れるオープンソースの強化

Red Hat では、コード、ドキュメントにおける配慮に欠ける用語の置き換えに取り組んでいます。まずは、マスター (master)、スレーブ (slave)、ブラックリスト (blacklist)、ホワイトリスト (whitelist) の 4 つの用語の置き換えから始めます。この取り組みは膨大な作業を要するため、今後の複数のリリースで段階的に用語の置き換えを実施して参ります。多様性を受け入れる用語に変更する取り組みの詳細は、Red Hat CTO である Chris Wright のメッセージ を参照してください。

RED HAT ドキュメントへのフィードバック (英語のみ)

Red Hat ドキュメントに関するご意見やご感想をお寄せください。また、改善点があればお知らせください。

Jira からのフィードバック送信 (アカウントが必要)

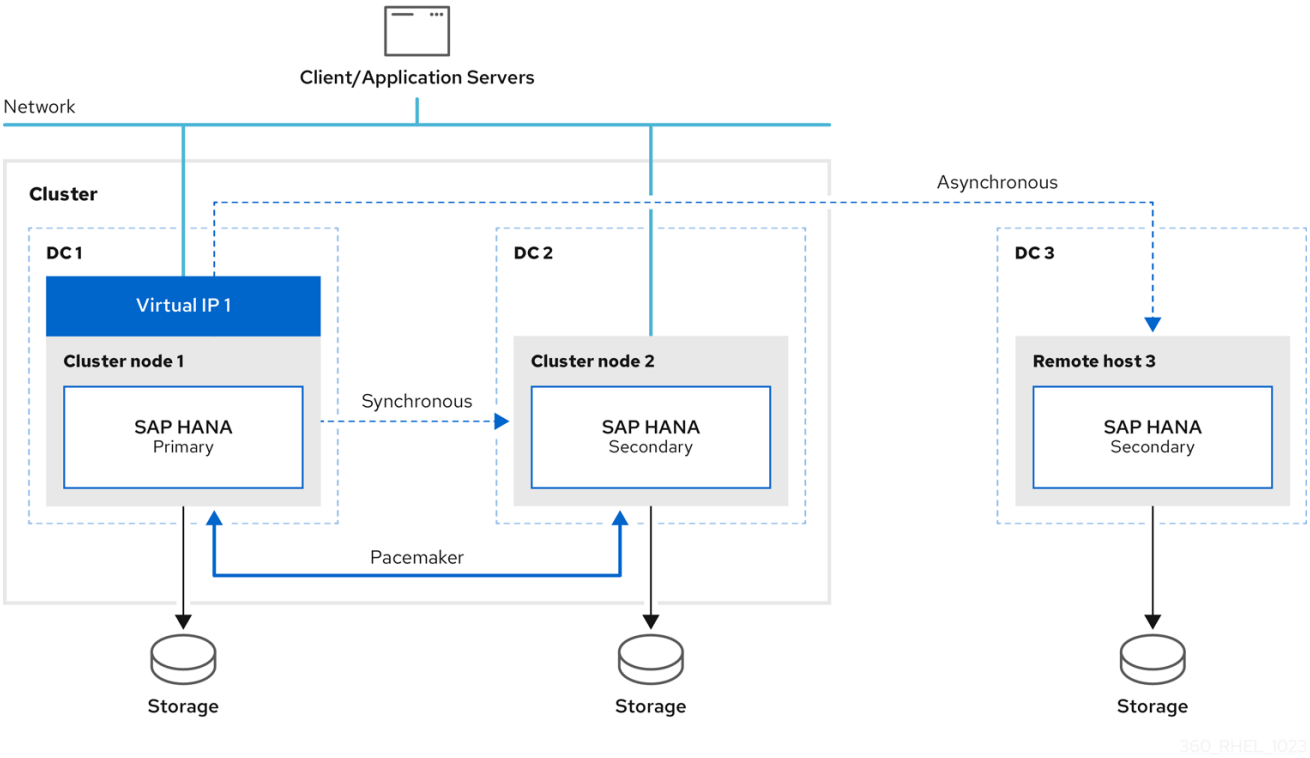
1. Jira の Web サイトにログインしていることを確認してください。
2. こちらのリンク をクリックして、フィードバックをお寄せください。
3. **Summary** フィールドにわかりやすいタイトルを入力します。
4. **Description** フィールドに、ドキュメントの改善に関するご意見を記入してください。ドキュメントの該当部分へのリンクも追加してください。
5. 今後の更新に関する通知を受け取りたい場合は、**Reporter** としてご自身が割り当てられていることを確認してください。
6. ダイアログの下部にある **Create** をクリックします。

第1章 概要

可用性に対する要求が高まっているため、データのコピーは 1 つだけでは十分ではありません。ビジネスの継続性を確保するには、信頼性と可用性の高いアーキテクチャーを使用し、複数のシステム間でデータをレプリケートする必要があります。マルチターゲットシステムレプリケーションを使用すると、プライマリーシステムのデータ変更を複数のセカンダリーシステムにレプリケートできます。詳細は、SAP HANA Multitarget System Replication を参照してください。

このドキュメントでは、RHEL HA Add-On を使用した SAP HANA Scale-Up System Replication の自動化 の説明に従って、インストールされた 2 ノードクラスター上の SAP HANA Multitarget System Replication を使用して、障害復旧用にレプリケーションサイトを設定する方法について説明します。

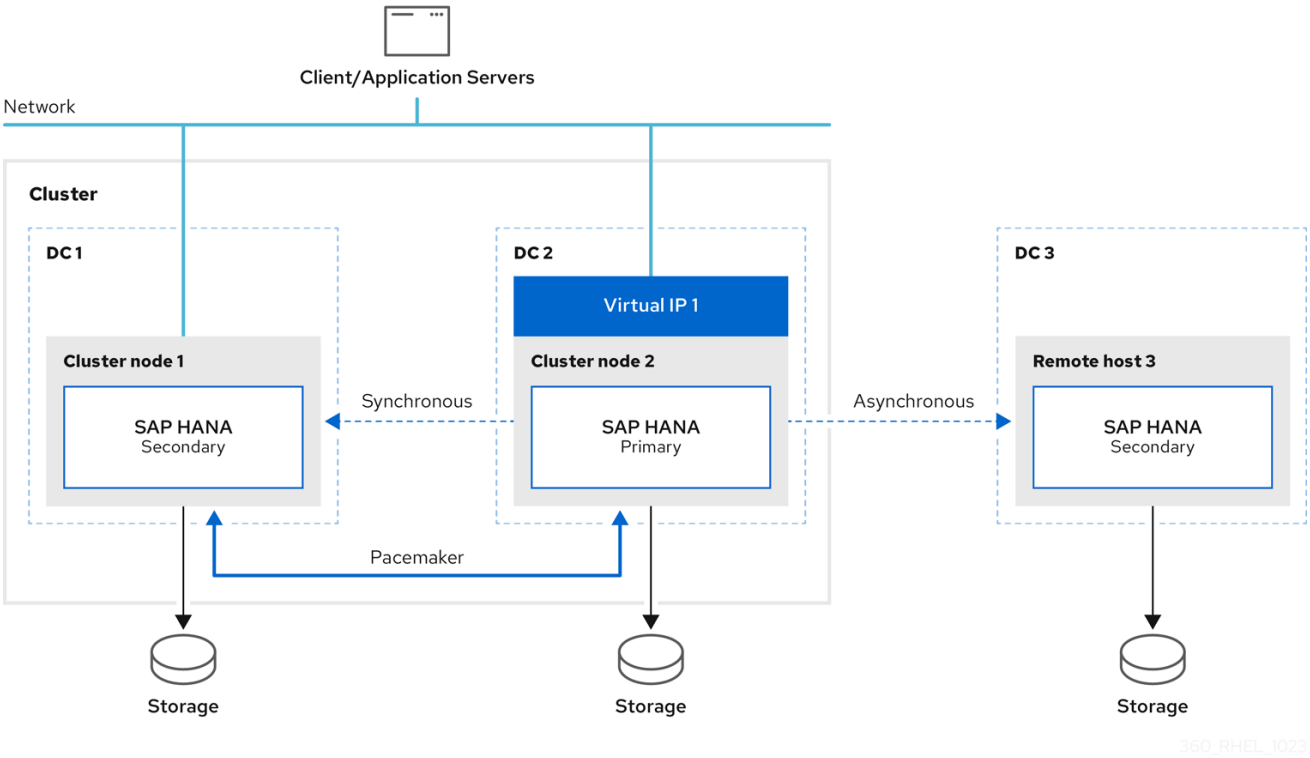
設定の例を以下に示します。



初期設定は次のとおりです。

- プライマリーサイト 1 (DC1) をセカンダリーサイト 2 (DC2) にレプリケート
- プライマリーサイト 1 (DC1) をセカンダリーサイト 3 (DC3) にレプリケート

プライマリーに障害が発生した場合、プライマリーはセカンダリーサイト 2 (DC2) に切り替わり、以前のプライマリーサイト 1 (DC1) がセカンダリーサイトになります。

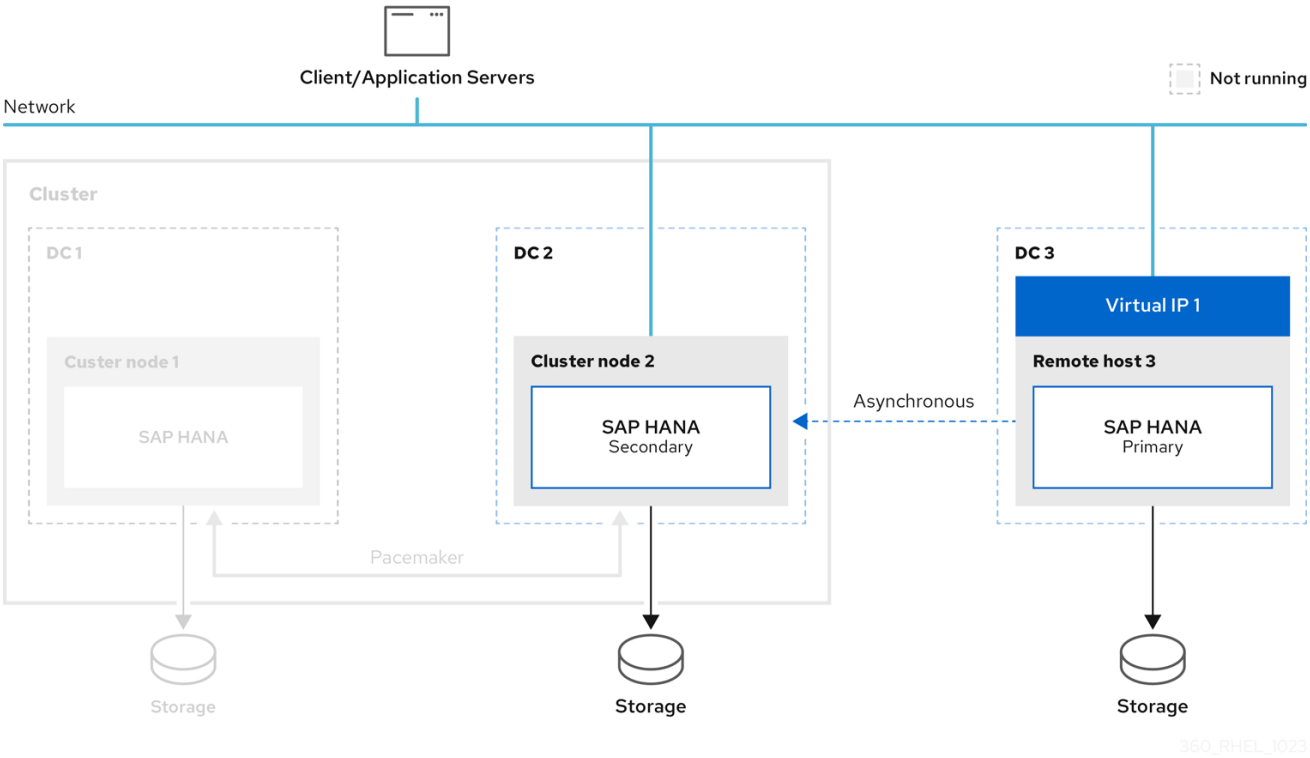


フェイルオーバーが発生すると、このソリューションにより、設定されたプライマリーサイトが 3 番目の DR サイトでも確実に切り替えられます。フェイルオーバー後の設定は次のとおりです。

- プライマリーが DC2 で稼働
- セカンダリーが DC1 で稼働 (DC2 から同期)
- セカンダリーが DC3 で稼働 (DC2 から同期)

このインスタンスがフェイルオーバー中に稼働している限り、remotehost3 上の SAP HANA インスタンスは、新しいプライマリーに自動的に再登録されます。

このドキュメントでは、プライマリーデータベースを 3 番目のサイトに切り替える例についても説明します。



クライアントをデータベースに接続するには、さらにネットワーク設定が必要であることに注意してください。ネットワーク設定はこのドキュメントの範囲外です。

詳細は、以下を参照してください。

- SAP HANA Administration Guide for SAP HANA Platform
- How to Setup SAP HANA Multi-Target System Replication

第2章 パラメーター

既存の 2 ノードクラスターの次のパラメーターが、3 番目のサイトのセットアップに使用されます。

パラメーター	例	説明
SID	RH2	HANA データベースのシステム ID
First SITE	DC1	1 番目のデータセンター/サイトの名前
Second SITE	DC2	2 番目のデータセンター/サイトの名前
Third SITE	DC3	3 番目のデータセンター/サイトの名前
InstanceNr	02	HANA インスタンス番号
<sid>adm uid	1000	sidadm ユーザーのユーザー ID
sapsys gid	980	sapsys のグループ ID

3 つの HANA インスタンスすべてで、以下に対して同じ値を使用する必要があります。

- SID
- InstanceNr
- <sid>adm uid
- sapsys gid

第3章 前提条件

ソリューションが機能するには、次の要件を満たす必要があります。

次のものがすべてのノードで同じである必要があります。

- CPU と RAM の数
- ソフトウェア設定
- RHEL リリース
- ファイアウォールの設定
- SAP HANA リリース (SAP HANA 2.0 SPS04 以降)

Pacemaker パッケージはクラスターノードにのみインストールされ、同じバージョンの resource-agents-sap-hana (0.162.1 以降) を使用する必要があります。

SAP HANA Multitarget System Replication をサポートできるようにするには、SAP HANA Multitarget System Replication の自動登録サポートの追加 を参照してください。また、以下を設定してください。

- **register_secondaries_on_takeover=true** の使用
- **log_mode=normal** の使用

初期設定は、インストールガイド RHEL HA Add-On を使用した SAP HANA Scale-Up System Replication の自動化 に基づいています。

すべての SAP HANA インスタンスのシステムレプリケーション設定は、SAP 要件に基づいています。詳細は、SAP HANA Administration Guide に基づく SAP のガイドラインを参照してください。

第4章 インストール

この章では、追加の SAP HANA インスタンスのインストールについて説明します。

4.1. フェイルオーバーテストで 2 ノードの基本インストールを確認する

RHEL HA Add-On を使用した SAP HANA Scale-Up System Replication の自動化 に基づいてインストールを行ったことを確認します。

SAP HANA Multitarget System Replication を使用できるようにするには、resource-agents-sap-hana のバージョンが 0.162.1 以降である必要があります。これは、以下のコマンドで確認できます。

```
# rpm -q resource-agents-sap-hana
resource-agents-sap-hana-0.162.1-0.el8_6.1.noarch
```

フェイルオーバーテストを実行すると、環境が機能していることを確認できます。SAPHana リソースを移動することもできます。これについては、move を使用した SAPHana リソースのフェイルオーバー でも説明されています。

4.2. 3 番目のサイトに SAP HANA をインストールする

3 番目のサイトでは、次に示すように、2 ノード Pacemaker クラスター上の SAP HANA インスタンスと同じバージョンとパラメーターを使用して、SAP HANA をインストールする必要もあります。

パラメーター	値
SID	RH2
InstanceNumber	02
<sid>adm user ID	rh2adm 999
sapsys group ID	sapsys 999

SAP HANA のインストールは **hdblcm** を使用して行います。詳細は、hdbclm を使用した SAP HANA のインストール を参照してください。必要に応じて、Ansible を使用してインストールを実行することもできます。

この章の例では、以下を使用しています。

- ホスト: DC1 サイトの clusternode1、DC2 サイトの clusternode2、および DC3 サイトの remotehost3
- SID RH2
- adminuser rh2adm

4.3. 3 番目のノードで SAP HANA SYSTEM REPLICATION をセットアップする

既存のインストールでは、2 ノードクラスター内のプライマリー SAP HANA インスタンスとセカンダ

リー SAP HANA インスタンスの間にすでに SAP HANA System Replication が設定されています。SAP HANA System Replication は、稼働中のプライマリー SAP HANA データベースインスタンスで有効になります。

この章では、3 番目の SAP HANA インスタンスを、DC3 サイトのノード remotehost3 上にある追加のセカンダリー HANA System Replication サイトとして登録する方法について説明します。この手順は、ノード clusternode2 上の元のセカンダリー HANA インスタンス (DC2) の登録に似ています。詳細は、以降の章で説明します。さらに詳しい情報が必要な場合は、General Prerequisites for Configuring SAP HANA System Replication も参照してください。

4.3.1. プライマリーデータベースの確認

他のデータベースが実行中であり、システムのレプリケーションが適切に動作していることを確認する必要があります。以下を参照してください。

- データベースの確認
- SAP HANA System Replication のステータスの確認
- プライマリーおよびセカンダリー SAP HANA データベースの検出

次の方法でプライマリー HANA インスタンスを検出できます。

```
clusternode1:rh2adm> hdbnsutil -sr_state | egrep -e "primary masters|^mode"
mode: primary
```

4.3.2. データベースキーのコピー

新しいセカンダリー HANA インスタンスを登録する前に、プライマリー HANA インスタンスのデータベースキーを新しい追加の HANA レプリケーションサイトにコピーする必要があります。この例では、3 番目のサイトのホスト名は、remotehost3 です。

たとえば、プライマリーノード clusternode1 で次を実行します。

```
clusternode1:rh2adm> scp -rp
/usr/sap/${SAPSYSTEMNAME}/SYS/global/security/rsecssfs/data/SSFS_${SAPSYSTEMNAME}.DAT
remotehost3:/usr/sap/${SAPSYSTEMNAME}/SYS/global/security/rsecssfs/data/SSFS_${SAPSYSTEMN
AME}.DAT
clusternode1:rh2adm> scp -rp
/usr/sap/${SAPSYSTEMNAME}/SYS/global/security/rsecssfs/key/SSFS_${SAPSYSTEMNAME}.KEY
remotehost3:/usr/sap/${SAPSYSTEMNAME}/SYS/global/security/rsecssfs/key/SSFS_${SAPSYSTEMN
AME}.KEY
```

4.3.3. 3 番目のサイトをセカンダリーとして登録する

プライマリーデータベース を実行しているノードの名前を確認する必要があります。
登録を監視するには、プライマリーノードの別のターミナルで次のコマンドを実行します。

```
clusternode1:rh2adm> watch python
/usr/sap/${SAPSYSTEMNAME}/HDB${TINSTANCE}/python_support/systemReplicationStatus.py
```

これにより、進行状況とエラー (発生した場合) が表示されます。

3 番目のサイト (DC3) の HANA インスタンスを追加のセカンダリー SAP HANA インスタンスとして登録するには、3 番目のサイトのホスト (remotehost3) で次のコマンドを実行します。

```
remotehost3:rh2adm> hdbnsutil -sr_register --name=DC3
--remoteHost=clusternode1 --remoteInstance=${TINSTANCE}
--replicationMode=async --operationMode=logreplay --online
```

この例では、DC3 は 3 番目のサイトの名前、clusternode1 はプライマリーノードの名前です。

データベースインスタンスがすでに実行されている場合は、停止する必要はありません。オプション **--online** を使用すると、オンライン中にインスタンスが登録されます。インスタンスの必要な再起動 (停止と起動) は、**hdbnsutil** 自体によって開始されます。

注記

--online オプションは、HANA インスタンスがオンラインでもオフラインでも、どのような場合でも機能します (このオプションは SAP HANA 2.0 SPS04 以降で使用できます)。

HANA インスタンスがオフラインの場合は、3 番目のノードが登録された後に起動する必要があります。詳細は、SAP HANA System Replication を参照してください。

4.3.4. SAP HANA Multitarget System Replication の自動登録サポートの追加

register_secondaries_on_takeover = true という SAP HANA System Replication オプションを使用しています。これにより、以前のプライマリーサイトと他のセカンダリーサイトの間でフェイルオーバーが発生した場合に、新しいプライマリーサイトが自動的に再登録されます。このオプションは、すべての潜在的なプライマリーサイトの **global.ini** ファイルに追加する必要があります。

すべての HANA インスタンスの **global.ini** に、このエントリーが含まれている必要があります。

```
[system_replication]
register_secondaries_on_takeover = true
```

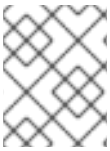
次の 2 つの章では、**global.ini** 設定について詳しく説明します。

注意

このパラメーターが設定されていても、フェイルオーバーの開始時に 3 番目のデータベースが **停止** してした場合、3 番目のインスタンスを手動で再登録する必要があります。

4.3.5. Pacemaker ノードでの global.ini の設定

Pacemaker クラスターによって管理されるサイト 1 およびサイト 2 の SAP HANA ノードの **global.ini** の **[system_replication]** セクションに、**register_secondaries_on_takeover = true** オプションを追加する必要があります。ファイル **global.ini** は、常にそれぞれのノードで編集し、別のノードからファイルをコピーしないでください。



注記

global.ini ファイルは、サイトの HANA インスタンスが処理を停止した場合にのみ編集してください。

rh2adm ユーザーとして **global.ini** を編集します。

```
clusternode1:rh2adm> vim
/usr/sap/${SAPSYSTEMNAME}/SYS/global/hdb/custom/config/global.ini
```

以下に例を示します。

```
# global.ini last modified 2023-07-14 16:31:14.120444 by hdbnsutil -sr_register --
remoteHost=hana07 --remoteInstance=02 --replicationMode=syncmem --operationMode=logreplay --
name=DC2
[multidb]
mode = multidb
database_isolation = low
singletenant = yes

[ha_dr_provider_SAPHanaSR]
provider = SAPHanaSR
path = /hana/shared/myHooks
execution_order = 1

[persistence]
basepath_datavolumes = /hana/data/RH2
basepath_logvolumes = /hana/log/RH2
log_mode = normal
enable_auto_log_backup = true

[system_replication]
register_secondaries_on_takeover = true
timetravel_logreplay_mode = auto
operation_mode = logreplay
mode = primary
actual_mode = syncmem
site_id = 1
site_name = DC2

[system_replication_site_masters]
2 = clusternode1:30201

[trace]
ha_dr_saphanasr = info
```

このオプションは、SAP HANA データベースインスタンスが開始されるとすぐにアクティブになります。

4.3.6. remotehost3 で **global.ini** を設定する

<sid>adm ユーザーとして **global.ini** を編集します。

```
% vim /usr/sap/${SAPSYSTEMNAME}/SYS/global/hdb/custom/config/global.ini
```

remotehost3 では、**ha_dr_provider_SAPhanaSR** セクションは使用されません。

remotehost3 上の **global.ini** の例:

```
# global.ini last modified 2023-06-22 17:22:54.154508 by hdbnameserver
[multidb]
mode = multidb
database_isolation = low
singletenant = yes

[persistence]
basepath_datavolumes = /hana/data/RH2
basepath_logvolumes = /hana/log/RH2
log_mode = normal
enable_auto_log_backup = true

[system_replication]
operation_mode = logreplay
register_secondaries_on_takeover = true
reconnect_time_interval = 5
timetravel_logreplay_mode = auto
site_id = 3
mode = syncmem
actual_mode = syncmem
site_name = DC3

[system_replication_site_masters]
2 = clusternode1:30201
```

4.3.7. インストールの検証

インストール後、すべての HANA インスタンスが稼働しているか、およびそれらの間で HANA System Replication が機能しているかを確認する必要があります。最も簡単な方法は、**systemReplicationStatus** を確認することです。詳細は、System Replication ステータスの確認 を参照してください。詳細は、データベースの確認 も参照してください。

HANA System Replication が正しく機能するには、“log_mode” パラメーターが “normal” に設定されていることを確認してください。詳細は、SAP HANA データベースの log_mode の確認 を参照してください。

セットアップが期待どおりに機能していることを確認するには、次の章で説明されているように、テストケース を実行してください。

第5章 テストケース

インストールの完了後、いくつかの基本テストを実行してインストールを確認し、SAP HANA Multitarget System Replication がどのように動作するか、および障害からどのように復旧するかを検証することを推奨します。実稼働を開始する前に、このようなテストケースを実行することを常に推奨します。可能であれば、実稼働環境に適用する前に、変更を検証するためのテスト環境を準備することもできます。

すべてのケースで、次の項目が説明されています。

- テストの内容
- テストの前提条件
- テストの手順
- テストのモニタリング
- テストの開始
- 期待される結果
- 初期状態に戻す方法

以前のプライマリー HANA レプリケーションサイトを、クラスターによって管理される HANA インスタンス上の新しいセカンダリー HANA レプリケーションサイトとして自動的に登録するには、SAPHana リソースでオプションの AUTOMATED_REGISTER=true を使用できます。詳細は、AUTOMATED_REGISTER を参照してください。

例で使用されている HA クラスターノードと HANA レプリケーションサイト (括弧内) の名前は次のとおりです。

- clusternode1 (DC1)
- clusternode2 (DC2)
- remotehost3 (DC3)

次のパラメーターは、HANA インスタンスとクラスターの設定に使用されます。

- SID=RH2
- INSTANCENUMBER=02
- CLUSTERNAME=cluster1

テスト環境の **/etc/hosts** で、clusternode1-2、remotehost3 をエイリアスとしても使用できます。

例や前提条件の追加チェックなど、テストについて詳しく説明します。最後には、さらなるテストに備えて、環境をクリーンアップする方法の例が示されています。

場合によっては、clusternode1-2 と remotehost3 の間の距離が長すぎる場合は、**–replicationMode=syncmem** の代わりに **–replcationMode=async** を使用する必要があります。適切なオプションを選択する前に、SAP HANA 管理者にも問い合わせてください。

5.1. テストの準備

テストを実行する前に、環境全体が正常で健全な状態である必要があります。次のコマンドで、クラスターとデータベースを確認する必要があります。

- **pcs status --full**
- **python systemReplicationStatus.py**
- **df -h**

pcs status --full の例は、pcs status を使用したクラスターのステータスの確認 にあります。"Migration Summary" に警告または以前の失敗がある場合は、テストを開始する前にクラスターをクリーンアップする必要があります。

```
[root@clusternode1]# pcs resource clear SAPHana_RH2_02-clone
```

クラスターのクリーンアップ で、クリーンアップを実行するその他の方法を説明しています。重要なのは、クラスターとすべてのリソースが起動していることです。

クラスターのほかに、データベースも稼働しており、同期している必要があります。データベースのステータスが適切であることを確認する最も簡単な方法は、システムのレプリケーションステータスを確認することです。レプリケーションステータス も参照してください。これはプライマリーデータベースで確認する必要があります。

プライマリーノードを検出するには、プライマリーデータベースの検出 をオンにするか、次のコマンドを使用します。

- **pcs status | grep -E "Promoted|Master"**
- **hdbnsutil -sr_stateConfiguration**

次のコマンドを実行して、ファイルシステムに十分なスペースがあるかどうかを確認します。

```
# df -h
```

続行する前に、システムチェック のガイドラインにも従ってください。環境がクリーンであれば、テストを実行する準備ができています。テスト中、モニタリングは進行状況を確認するのに役立ちます。

5.2. 環境のモニタリング

このセクションでは、テスト中の環境のモニタリングに焦点を当てます。このセクションでは、変更を確認するために必要なモニターのみを説明します。モニターは専用のターミナルから実行することを推奨します。テスト中に変化を検出できるように、テストを開始する前にモニタリングを開始することを推奨します。

便利なコマンド セクションで、さらに多くの例を紹介しています。

5.2.1. プライマリーノードの検出

プライマリーノードを検出してフェイルオーバーを監視するか、プライマリーノードで実行したときにレプリケーションステータスに関する情報のみを提供する特定のコマンドを実行する必要があります。

プライマリーノードを検出するには、**<sid>adm** ユーザーとして次のコマンドを実行します。

```
clusternode1:rh2adm> watch -n 5 'hdbnsutil -sr_stateConfiguration | egrep -e "primary
masters|^mode"'
```

出力例 (clusternode2 がプライマリーデータベースの場合):

```
mode: syncmem
primary masters: clusternode2
```

プライマリーノードを特定する 2 つ目の方法は、クラスターノード上で root として次のコマンドを実行することです。

```
# watch -n 5 'pcs status --full'
```

プライマリーデータベースを実行するノードの出力は次のとおりです。

```
mode: primary
```

5.2.2. レプリケーションステータスの確認

レプリケーションのステータスは、プライマリーデータベースノードとセカンダリーデータベースノード間の関係と、レプリケーションの現在のステータスを示します。

レプリケーションのステータスを確認するには、**<sid>adm** ユーザーとして次のコマンドを実行します。

```
clusternode1:rh2adm> hdbnsutil -sr_stateConfiguration
```

システムレプリケーションステータスの変更を永続的に監視したい場合は、次のコマンドを実行してください。

```
clusternode1:rh2adm> watch -n 5 'python
/usr/sap/${SAPSYSTEMNAME}/HDB${TINSTANCE}/exe/python_support/systemReplicationStatus.py
; echo Status $?'
```

この例では、レプリケーションのステータスを繰り返し取得します。また、現在の戻りコードを確認します。
戻りコード (ステータス) が 15 である限り、レプリケーションのステータスは正常です。他の戻りコードは次のとおりです。

- 10: NoHSR
- 11: Error
- 12: Unknown
- 13: Initializing
- 14: Syncing
- 15: Active

新しいセカンダリーを登録する場合は、プライマリーノード上の別のウィンドウでコマンドを実行し、レプリケーションの進行状況を確認できます。フェイルオーバーを監視する場合は、前のプライマリーデータベースサーバーと新しいプライマリーデータベースサーバーで並行してコマンドを実行できます。詳細は、SAP HANA System Replication のステータスの確認 を参照してください。

5.2.3. /var/log/messages のエントリーの確認

Pacemaker は、**/var/log/messages** ファイルに多くの情報を書き込みます。フェイルオーバー中に、大量のメッセージがこのメッセージファイルに書き込まれます。SAP HANA リソースエージェントに応じて重要なメッセージのみを追跡できるようにするには、Pacemaker SAP リソースの詳細なアクティビティーをフィルターすると便利です。これは、単一のクラスターノード上のメッセージファイルを確認する場合に十分です。

たとえば、次のエイリアスを使用できます。

```
# tmsl='tail -1000f /var/log/messages | egrep -s "Setting master-
rsc_SAPHana_${SAPSYSTEMNAME}_HDB${TINSTANCE}|sr_register|WAITING4LPA|PROMOTED|
DEMOTED|UNDEFINED|master_walk|SWAIT|WaitforStopped|FAILED|LPT"'
```

このエイリアスを別のウィンドウで実行して、テストの進行状況を監視します。フェイルオーバーと同期状態の監視 の例も参照してください。

5.2.4. クラスターの状態

クラスターのステータスを確認するにはいくつかの方法があります。

- クラスターが実行中かどうかを確認します。
  - **pcs cluster status**
- クラスターとすべてのリソースを確認します。
  - **pcs status**
- クラスター、すべてのリソース、およびすべてのノード属性を確認します。
  - **pcs status --full**
- リソースのみを確認します。
  - **pcs resource**

pcs status --full コマンドを使用すると、必要な情報がすべて得られます。変更を監視するには、このコマンドを watch と一緒に実行できます。

```
# pcs status --full
```

変更を確認する場合は、別のウィンドウで **watch** コマンドを実行できます。

```
# watch pcs status --full
```

出力例とその他のオプションについては、クラスターのステータスの確認 を参照してください。

5.2.5. 残留物の検出

環境が次のテストを実行できる状態にあることを確認するには、以前のテストで残ったものを修正または削除する必要があります。

- **stonith** は、クラスター内のノードをフェンスするために使用します。
  - 検出: **[root@clusternode1]# pcs stonith history**

- 修正: **[root@clusternode1]# pcs stonith cleanup**

- 複数のプライマリーデータベース:
  - 検出: **clusternode1:rh2adm> hdbnsutil -sr_stateConfiguration | grep -i primary**
    同じプライマリーを持つすべてのノードを識別する必要があります。
  - 修正: clusternode1:rh2adm> は、**--force_full_replica** オプションを使用して間違ったプライマリーを再登録します。

- 移動により発生した場所の制約:
  - 検出: **[root@clusternode1]# pcs constraint location**
    警告セクションを確認します。
  - 修正: **[root@clusternode1]# pcs resource clear <clone-resource-which was moved>**

- セカンダリーレプリケーション関係:
  - 検出: プライマリーデータベースで **clusternode1:rh2adm> python ${DIR_EXECUTABLES}/python_support/systemReplicationStatus.py** を実行します。
  - 修正: セカンダリーデータベースを登録解除して再登録します。

- siteReplicationMode を確認します (すべての SAP HANA ノードで同じ出力が表示されます)
  - **clusternode1:rh2adm> hdbnsutil -sr_state --sapcontrol=1 |grep site.*Mode**

- Pcs プロパティー:
  - 検出: **[root@clusternode1]# pcs property config**
  - 修正: **[root@clusternode1]# pcs property set <key=value>**
  - **maintenance_mode** をクリア
  - **[root@clusternode1]# pcs property set maintenance-mode=false**

- **log_mode**:
  - 検出: **clusternode1:rh2adm> python systemReplicationStatus.py**
    通常は **log_mode** が必要であるという応答がレプリケーションのステータスで返されます。**log_mode** は、**hdbsql** を使用した **lnifile** の内容の確認 で説明されている方法で検出できます。
  - 修正: **log_mode** を normal に変更し、プライマリーデータベースを再起動します。

- CIB エントリー:
  - 検出: クラスター情報ベース内の SFAIL エントリー。
    CIB エントリーを検索して削除するには、クラスターの整合性の確認 を参照してください。

- クリーンアップ/クリア:
  - 検出: **[root@clusternode1]# pcs status --full**
    場合によっては、エラーや警告が表示されることがあります。リソースをクリーンアップ/クリアしても、すべてが正常であれば何も起こりません。次のテストを実行する前に、環境をクリーンアップできます。

- 修正の例:
  [root@clusternode1]# pcs resource clear <name-of-the-clone-resource>

  [root@clusternode1]# pcs resource cleanup <name-of-the-clone-resource>

これは、既存の環境に問題があるか確認する場合にも役立ちます。詳細は、便利なコマンド を参照してください。

5.3. テスト 1: プライマリーノードをアクティブな 3 番目のサイトを使用してフェイルオーバーする

テストの内容	3 番目のサイトの自動再登録。 クリア後に同期状態が SOK に変わる。
テストの前提条件	• DC1、DC2、DC3 上の SAP HANA が稼働している。 • クラスターがエラーや警告なく稼働している。
テストの手順	**[root@clusternode1]# pcs resource move <sap-clone-resource> <target-node>** コマンドを使用して、SAPHana リソースを移動します。
テストのモニタリング	3 番目のサイトで、**sidadm** として、表の最後に記載されているコマンドを実行します。(*) セカンダリーノードで root として **[root@clusternode1]# watch pcs status --full** を実行します。
テストの開始	クラスターコマンドを実行します。 **[root@clusternode1] pcs move resource SAPHana_RH2_02-clone** **[root@clusternode1]# pcs resource clear SAPHana_RH2_02-clone**
期待される結果	サイト 3 の monitor コマンドでは、プライマリーマスターが clusternode1 から clusternode2 に変更されます。 リソースをクリアすると、同期状態が **SFAIL** から **SOK** に変わります。
初期状態に戻す方法	テストを 2 回実行します。

(*)

```
remotehost3:rh2adm>
watch hdbnsutil -sr_state
[root@clusternode1]# tail -1000f /var/log/messages |egrep -e 'SOK|SWAIT|SFAIL'
```

詳細な説明:

- clusternode1 または clusternode2 で root としてクラスターの初期状態を確認します。

```
[root@clusternode1]# pcs status --full
Cluster name: cluster1
Cluster Summary:
  * Stack: corosync
  * Current DC: clusternode1 (1) (version 2.1.2-4.el8_6.6-ada5c3b36e2) - partition with
quorum
  * Last updated: Mon Sep  4 06:34:46 2023
  * Last change:  Mon Sep  4 06:33:04 2023 by root via crm_attribute on clusternode1
  * 2 nodes configured
  * 6 resource instances configured

Node List:
  * Online: [ clusternode1 (1) clusternode2 (2) ]

Full List of Resources:
  * auto_rhevm_fence1 (stonith:fence_rhevm):  Started clusternode1
  * Clone Set: SAPHanaTopology_RH2_02-clone [SAPHanaTopology_RH2_02]:
    * SAPHanaTopology_RH2_02 (ocf::heartbeat:SAPHanaTopology):  Started clusternode2
    * SAPHanaTopology_RH2_02 (ocf::heartbeat:SAPHanaTopology):  Started clusternode1
  * Clone Set: SAPHana_RH2_02-clone [SAPHana_RH2_02] (promotable):
    * SAPHana_RH2_02 (ocf::heartbeat:SAPHana):  Slave clusternode2
    * SAPHana_RH2_02 (ocf::heartbeat:SAPHana):  Master clusternode1
  * vip_RH2_02_MASTER (ocf::heartbeat:IPaddr2):  Started clusternode1

Node Attributes:
  * Node: clusternode1 (1):
    * hana_rh2_clone_state            : PROMOTED
    * hana_rh2_op_mode                : logreplay
    * hana_rh2_remoteHost             : clusternode2
    * hana_rh2_roles                  : 4:P:master1:master:worker:master
    * hana_rh2_site                   : DC1
    * hana_rh2_sra                    : -
    * hana_rh2_srah                   : -
    * hana_rh2_srmode                 : syncmem
    * hana_rh2_sync_state             : PRIM
    * hana_rh2_version                : 2.00.062.00
    * hana_rh2_vhost                  : clusternode1
    * lpa_rh2_lpt                     : 1693809184
    * master-SAPHana_RH2_02           : 150
  * Node: clusternode2 (2):
    * hana_rh2_clone_state            : DEMOTED
    * hana_rh2_op_mode                : logreplay
    * hana_rh2_remoteHost             : clusternode1
    * hana_rh2_roles                  : 4:S:master1:master:worker:master
    * hana_rh2_site                   : DC2
    * hana_rh2_sra                    : -
    * hana_rh2_srah                   : -
    * hana_rh2_srmode                 : syncmem
```

```
    * hana_rh2_sync_state              : SOK
    * hana_rh2_version                 : 2.00.062.00
    * hana_rh2_vhost                   : clusternode2
    * lpa_rh2_lpt                      : 30
    * master-SAPHana_RH2_02            : 100

Migration Summary:

Tickets:

PCSD Status:
  clusternode1: Online
  clusternode2: Online

Daemon Status:
  corosync: active/disabled
  pacemaker: active/disabled
  pcsd: active/enabled
```

この出力は、HANA がプライマリー SAP HANA サーバーである clusternode1 上で昇格されていること、およびクローンリソースの名前が SAPHana_RH2_02-clone であり、昇格可能であることを示しています。

テスト中にこれを別のウィンドウで実行して、変更を確認できます。

```
[root@clusternode1]# watch pcs status --full
```

- SAP HANA クローンリソースの名前を識別する別の方法は次のとおりです。

```
[root@clusternode2]# pcs resource
  * Clone Set: SAPHanaTopology_RH2_02-clone [SAPHanaTopology_RH2_02]:
    * Started: [ clusternode1 clusternode2 ]
  * Clone Set: SAPHana_RH2_02-clone [SAPHana_RH2_02] (promotable):
    * Promoted: [ clusternode2 ]
    * Unpromoted: [ clusternode1 ]
```

プライマリーサーバーの変更を確認するには、テストを開始する前に、別のターミナルウィンドウで remotehost3 のモニタリングを開始します。

```
remotehost3:rh2adm> watch 'hdbnsutil -sr_state | grep "primary masters"
```

出力は次のようになります。

```
Every 2.0s: hdbnsutil -sr_state | grep "primary masters"
remotehost3: Mon Sep  4 08:47:21 2023

primary masters: clusternode1
```

テスト中に、予想される出力は clusternode2 に変わります。

- 上記で検出したクローンリソースを clusternode2 に移動してテストを開始します。

```
[root@clusternode1]# pcs resource move SAPhana_RH2_02-clone clusternode2
```

remotehost3 上のモニターの出力は次のように変わります。

```
Every 2.0s: hdbnsutil -sr_state | grep "primary masters"
remotehost3: Mon Sep  4 08:50:31 2023

primary masters: clusternode2
```

Pacemaker は、クローンリソースを移動するための場所の制約を作成します。これは手動で削除する必要があります。以下を使用して制約を確認できます。

```
[root@clusternode1]# pcs constraint location
```

この制約は、次の手順を実行して削除する必要があります。

- クローンリソースをクリアして、場所の制約を削除します。

```
[root@clusternode1]# pcs resource clear SAPhana_RH2_02-clone
Removing constraint: cli-prefer-SAPHana_RH2_02-clone
```

- リソースをクリーンアップします。

```
[root@clusternode1]# pcs resource cleanup SAPHana_RH2_02-clone
Cleaned up SAPHana_RH2_02:0 on clusternode2
Cleaned up SAPHana_RH2_02:1 on clusternode1
Waiting for 1 reply from the controller
... got reply (done)
```

テスト結果

- remotehost3 上の “プライマリーマスター” モニターには、新しいプライマリーノードへの即時切り替えが表示されるはずです。
- クラスターのステータスを確認すると、元のセカンダリーがプロモートされ、元のプライマリーが再登録されて、**Clone_State** が **Promoted** から **Unknown**、**WAITINGFORLPA**、**DEMOTED** に変わります。
- セカンダリーは、フェイルオーバー後に初めて **SAPHana** モニターが起動するときに、**sync_state** を **SFAIL** に変更します。既存の場所の制約のため、リソースをクリアする必要があります。しばらくするとセカンダリーの **sync_state** が再び **SOK** に変わります。
- セカンダリーがプロモートされます。

初期状態に戻すには、次のテストを実行します。テストが終了したら、クラスターのクリーンアップ を実行してください。

5.4. テスト 2: プライマリーノードをパッシブな 3 番目のサイトを使用してフェイルオーバーする

テストの内容	3 番目のサイトの登録なし。 3 番目のサイトが停止している場合でも、フェイルオーバーが機能する。

テストの前提条件	• DC1、DC2 では SAP HANA が稼働しているが、DC3 では停止している。 • クラスターがエラーや警告なく稼働している。
テストの手順	**pcs move** コマンドを使用して SAPHana リソースを移動します。
テストの開始	クラスターコマンドを実行します。 **[root@clusternode1]# pcs move resource SAPHana_RH2_02-clone**
テストのモニタリング	3 番目のサイトで **sidadm** として **% watch hdbnsutil -sr_stateConfiguration** を実行します。 クラスターノード上で root として **[root@clusternode1]# watch pcs status** を実行します。
期待される結果	DC3 には何も変化がありません。レプリケーションは古い関係を維持します。
初期状態に戻す方法	新しいプライマリーに DC3 を再登録し、SAP HANA を起動します。

詳細な説明:

- clusternode1 または clusternode2 で root としてクラスターの初期状態を確認します。

```
[root@clusternode1]# pcs status --full
Cluster name: cluster1
Cluster Summary:
  * Stack: corosync
  * Current DC: clusternode1 (1) (version 2.1.2-4.el8_6.6-ada5c3b36e2) - partition with
quorum
  * Last updated: Mon Sep  4 06:34:46 2023
  * Last change:  Mon Sep  4 06:33:04 2023 by root via crm_attribute on clusternode1
  * 2 nodes configured
  * 6 resource instances configured

Node List:
  * Online: [ clusternode1 (1) clusternode2 (2) ]

Full List of Resources:
  * auto_rhevm_fence1 (stonith:fence_rhevm):  Started clusternode1
  * Clone Set: SAPHanaTopology_RH2_02-clone [SAPHanaTopology_RH2_02]:
    * SAPHanaTopology_RH2_02 (ocf::heartbeat:SAPHanaTopology):  Started clusternode2
    * SAPHanaTopology_RH2_02 (ocf::heartbeat:SAPHanaTopology):  Started clusternode1
```

```
  * Clone Set: SAPHana_RH2_02-clone [SAPHana_RH2_02] (promotable):
    * SAPHana_RH2_02 (ocf::heartbeat:SAPHana):  Slave clusternode2
    * SAPHana_RH2_02 (ocf::heartbeat:SAPHana):  Master clusternode1
  * vip_RH2_02_MASTER (ocf::heartbeat:IPaddr2):  Started clusternode1

Node Attributes:
  * Node: clusternode1 (1):
    * hana_rh2_clone_state            : PROMOTED
    * hana_rh2_op_mode                : logreplay
    * hana_rh2_remoteHost             : clusternode2
    * hana_rh2_roles                  : 4:P:master1:master:worker:master
    * hana_rh2_site                   : DC1
    * hana_rh2_sra                    : -
    * hana_rh2_srah                   : -
    * hana_rh2_srmode                 : syncmem
    * hana_rh2_sync_state             : PRIM
    * hana_rh2_version                : 2.00.062.00
    * hana_rh2_vhost                  : clusternode1
    * lpa_rh2_lpt                     : 1693809184
    * master-SAPHana_RH2_02           : 150
  * Node: clusternode2 (2):
    * hana_rh2_clone_state            : DEMOTED
    * hana_rh2_op_mode                : logreplay
    * hana_rh2_remoteHost             : clusternode1
    * hana_rh2_roles                  : 4:S:master1:master:worker:master
    * hana_rh2_site                   : DC2
    * hana_rh2_sra                    : -
    * hana_rh2_srah                   : -
    * hana_rh2_srmode                 : syncmem
    * hana_rh2_sync_state             : SOK
    * hana_rh2_version                : 2.00.062.00
    * hana_rh2_vhost                  : clusternode2
    * lpa_rh2_lpt                     : 30
    * master-SAPHana_RH2_02           : 100

Migration Summary:

Tickets:

PCSD Status:
  clusternode1: Online
  clusternode2: Online

Daemon Status:
  corosync: active/disabled
  pacemaker: active/disabled
  pcsd: active/enabled
```

この例のこの出力は、HANA がプライマリー SAP HANA サーバーである clusternode1 上でプロモートされていること、およびクローンリソースの名前が **SAPHana_RH2_02-clone** であり、プロモート可能であることを示しています。HANA の前にテスト 3 を実行すると、clusternode2 でテスト 3 がプロモートされる可能性があります。

- remotehost3 上のデータベースを停止します。

```
remotehost3:rh2adm> HDB stop
```

```
hdbdaemon will wait maximal 300 seconds for NewDB services finishing.
Stopping instance using: /usr/sap/RH2/SYS/exe/hdb/sapcontrol -prot NI_HTTP -nr 02 -
function Stop 400

12.07.2023 11:33:14
Stop
OK
Waiting for stopped instance using: /usr/sap/RH2/SYS/exe/hdb/sapcontrol -prot NI_HTTP -nr
02 -function WaitforStopped 600 2


12.07.2023 11:33:30
WaitforStopped
OK
hdbdaemon is stopped.
```

- remotehost3 上のプライマリーデータベースを確認します。

```
remotehost3:rh2adm> hdbnsutil -sr_stateConfiguration| grep -i "primary masters"

primary masters: clusternode2
```

- クラスターノード上のクラスター内の現在のプライマリーを確認します。

```
[root@clusternode1]# pcs resource | grep Masters
    * Masters: [ clusternode2 ]
```

- **sr_state** を確認して、SAP HANA System Replication 関係を確認します。

```
clusternode2remotehost3:rh2adm> hdbnsutil -sr_state

System Replication State
~~~~~~~~~~~~~~~~~~~~~~~~

online: true

mode: primary
operation mode: primary
site id: 2
site name: DC1

is source system: true
is secondary/consumer system: false
has secondaries/consumers attached: true
is a takeover active: false
is primary suspended: false

Host Mappings:
~~~~~~~~~~~~~~

clusternode1 -> [DC3] remotehost3
clusternode1 -> [DC1] clusternode1
clusternode1 -> [DC2] clusternode2
```

```
Site Mappings:
~~~~~~~~~~~~~~~
DC1 (primary/primary)
    |---DC3 (syncmem/logreplay)
    |---DC2 (syncmem/logreplay)

Tier of DC1: 1
Tier of DC3: 2
Tier of DC2: 2

Replication mode of DC1: primary
Replication mode of DC3: syncmem
Replication mode of DC2: syncmem

Operation mode of DC1: primary
Operation mode of DC3: logreplay
Operation mode of DC2: logreplay

Mapping: DC1 -> DC3
Mapping: DC1 -> DC2
done.
```

SAP HANA System Replication 関係には、依然として 1 つのプライマリー (DC1) があり、DC2 と DC3 にレプリケートされます。

ダウンしている remotehost3 上のレプリケーション関係は、次のコマンドを使用して表示できます。

```
remothost3:rh2adm> hdbnsutil -sr_stateConfiguration

System Replication State
~~~~~~~~~~~~~~~~~~~~~~~~

mode: syncmem
site id: 3
site name: DC3
active primary site: 1

primary masters: clusternode1
done.
```

オフラインの remotehost3 上のデータベースは、**global.ini** ファイル内のエントリーをチェックします。

- テストの開始: クラスター内でフェイルオーバーを開始し、**SAPHana-clone-resource** の例を移動します。

```
[root@clusternode1]# pcs resource move SAPHana_RH2_02-clone clusternode2
```

注記

SAPHana が clusternode2 でプロモートされている場合は、クローンリソースを clusternode1 に移動する必要があります。この例では、SAPHana が clusternode1 でプロモートされることを想定しています。

出力はありません。前のテストと同様に、場所の制約が作成され、次のように表示できます。

```
[root@clusternode1]# pcs constraint location
Location Constraints:
  Resource: SAPHana_RH2_02-clone
    Enabled on:
      Node: clusternode1 (score:INFINITY) (role:Started)
```

クラスターが再び正常に見える場合でも、この制約により、制約が削除されない限り、別のフェイルオーバーが回避されます。1 つの方法は、リソースをクリアすることです。

- リソースをクリアします。

```
[root@clusternode1]# pcs constraint location
Location Constraints:
  Resource: SAPHana_RH2_02-clone
    Enabled on:
      Node: clusternode1 (score:INFINITY) (role:Started)
[root@clusternode1]# pcs resource clear SAPHana_RH2_02-clone
Removing constraint: cli-prefer-SAPHana_RH2_02-clone
```

- リソースをクリーンアップします。

```
[root@clusternode1]# pcs resource cleanup SAPHana_RH2_02-clone
Cleaned up SAPHana_RH2_02:0 on clusternode2
Cleaned up SAPHana_RH2_02:1 on clusternode1
Waiting for 1 reply from the controller
... got reply (done)
```

- 現在のステータスを確認します。
  レプリケーションのステータスを表示するには 3 つの方法があり、同期している必要があります。まずは remotehost3 のプライマリーから始めます。

```
remotehost3clusternode2:rh2adm>  hdbnsutil -sr_stateConfiguration| grep -i primary
active primary site: 1
primary masters: clusternode1
```

出力には、サイト 1 または clusternode1 が表示されます。これは、プライマリーを clusternode2 に移動するテストを開始する前はプライマリーでした。

次に、新しいプライマリーのシステムレプリケーションステータスを確認します。

まず新しいプライマリーを検出します。

```
[root@clusternode1]# pcs resource | grep  Master
    * Masters: [ clusternode2 ]
```

ここでは不整合が発生しているため、remotehost3 を再登録する必要があります。テストを再度実行すると、プライマリーを元の clusternode1 に戻すのではないかと思うかもしれません。この場合、システムレプリケーションが機能しているかどうかを確認する 3 番目の方法があります。プライマリーノードで次のコマンドを実行します。

```
clusternode2:rh2adm> cdpy
clusternode2:rh2adm> python
```

```
${DIR_EXECUTABLES}/python_support/systemReplicationStatus.py
|Database |Host   |Port  |Service Name |Volume ID |Site ID |Site Name |Secondary
|Secondary |Secondary |Secondary |Secondary     |Replication |Replication |Replication
|Secondary    |
|         |       |      |             |          |        |          |Host      |Port      |Site ID   |Site Name |Active
Status |Mode        |Status      |Status Details |Fully Synced |
|-------- |------ |----- |------------ |--------- |------- |--------- |--------- |--------- |--------- |----
--------- |----------- |----------- |-------------- |------------ |
|SYSTEMDB |clusternode2 |30201 |nameserver   |        1 |      2 |DC2       |clusternode1    |
30201 |        1 |DC1       |YES           |SYNCMEM     |ACTIVE      |              |        True |
|RH2      |clusternode2 |30207 |xsengine     |        2 |      2 |DC2       |clusternode1    |    30207 |
1 |DC1       |YES           |SYNCMEM     |ACTIVE      |              |        True |
|RH2      |clusternode2 |30203 |indexserver  |        3 |      2 |DC2       |clusternode1    |    30203
|        1 |DC1       |YES           |SYNCMEM     |ACTIVE      |              |        True |

status system replication site "1": ACTIVE
overall system replication status: ACTIVE

Local System Replication State
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

mode: PRIMARY
site id: 2
site name: DC2
```

この出力に remotehost3 が表示されない場合は、remotehost3 を再登録する必要があります。登録する前に、プライマリーノードで次のコマンドを実行して、登録の進行状況を確認してください。

```
clusternode2:rh2adm> watch python
${DIR_EXECUTABLES}/python_support/systemReplicationStatus.py
```

これで、このコマンドを使用して、remotehost3 を再登録できます。

```
remotehost3:rh2adm> hdbnsutil -sr_register --remoteHost=clusternode2 --
remoteInstance=${TINSTANCE} --replicationMode=async --name=DC3 --remoteName=DC2
--operation
Mode=logreplay --online
adding site ...
collecting information ...
updating local ini files ...
done.
```

remotehost3 上のデータベースがまだ起動していない場合でも、システムレプリケーションステータスの出力で 3 番目のサイトを確認できます。登録は、remotehost3 でデータベースを起動することで完了できます。

```
remotehost3:rh2adm> HDB start


StartService
Impromptu CCC initialization by 'rscpCInit'.
  See SAP note 1266393.
OK
OK
```

```
Starting instance using: /usr/sap/RH2/SYS/exe/hdb/sapcontrol -prot NI_HTTP -nr 02 -
function StartWait 2700 2


04.09.2023 11:36:47
Start
OK
```

上記で開始されたモニターには、remotehost3 の同期がすぐに表示されます。

- 元に戻すには、テストを再度実行します。オプションのテストの 1 つは、プライマリーをノードに切り替えて、remotehost3 の **global.ini** で設定してデータベースを起動することです。データベースが起動する場合もありますが、再登録しない限り、システムレプリケーションステータスの出力には表示されません。
- 欠落しているエントリーがすぐに作成され、SAP HANA データベースが起動するとすぐにシステムレプリケーションが開始されます。
- これは、次を実行して確認できます。

```
sidadm@clusternode1% hdbnsutil -sr_state
sidadm@clusternode1% python systemReplicationStatus.py ; echo $?
```

- 詳細は、SAP HANA System Replication ステータスの確認 を参照してください。

5.5. テスト 3: プライマリーノードを 3 番目のサイトにフェイルオーバーする

テストの内容	プライマリーを 3 番目のサイトにフェイルオーバーします..3 番目のサイトがプライマリーになる。 セカンダリーが 3 番目のサイトに再登録される。
テストの前提条件	• DC1、DC2、DC3 上の SAP HANA が実行されています。 • クラスターがエラーや警告なく稼働している。 • System Replication が設定され、同期されている (**% python systemReplicationStatus.py** を確認してください)。
テストの手順	回復できるようにクラスターを **maintenance-mode** にします。 **% hdbnsuttil -sr_takeover** を使用して、3 番目のノードから HANA データベースをテイクオーバーします。
テストの開始	remotehost3:rh2adm> で SAP HANA コマンドを実行します: **hdbnsutil -sr_takeover**

テストのモニタリング	3 番目のサイトで **sidadm** として **% watch hdbnsutil -sr_state** を実行します。
期待される結果	• 3 番目のノードがプライマリーになります。 • セカンダリーノードは、プライマリーマスターを remotehost3 に変更します。以前のプライマリーノードを新しいプライマリーに再登録する必要があります。
初期状態に戻す方法	テスト 4: プライマリーノードを 1 番目のサイトにフェイルバックする を実行します。

詳細な説明:

- データベースの確認 を使用して、データベースが実行されているか確認し、レプリケーションのステータスを確認します。

```
clusternode2:rh2adm> hdbnsutil -sr_state | egrep -e "^mode:|primary masters"
```

出力は、たとえば次のようになります。

```
mode: syncmem
primary masters: clusternode1
```

この場合、プライマリーデータベースは clusternode1 です。このコマンドを clusternode1 で実行すると、次の結果が得られます。

```
mode: primary
```

このプライマリーノードでは、システムのレプリケーションステータスを表示することもできます。次のようになります。

```
clusternode1:rh2adm> cdpy
clusternode1:rh2adm> python systemReplicationStatus.py
|Database |Host   |Port  |Service Name |Volume ID |Site ID |Site Name |Secondary
|Secondary |Secondary |Secondary |Secondary     |Replication |Replication |Replication
|Secondary    |
|         |       |      |             |          |        |          |Host      |Port      |Site ID   |Site Name |Active
Status |Mode        |Status      |Status Details |Fully Synced |
|-------- |------ |----- |------------ |--------- |------- |--------- |--------- |--------- |--------- |---------- |----
--------- |----------- |----------- |-------------- |------------ |
|SYSTEMDB |clusternode1 |30201 |nameserver   |       1 |      1 |DC1       |remotehost3    |
30201 |        3 |DC3       |YES          |SYNCMEM     |ACTIVE      |               |        True |
|RH2      |clusternode1 |30207 |xsengine     |       2 |      1 |DC1       |remotehost3    |    30207 |
3 |DC3       |YES          |SYNCMEM     |ACTIVE      |               |        True |
|RH2      |clusternode1 |30203 |indexserver  |       3 |      1 |DC1       |remotehost3    |    30203
|        3 |DC3       |YES          |SYNCMEM     |ACTIVE      |               |        True |
|SYSTEMDB |clusternode1 |30201 |nameserver   |       1 |      1 |DC1       |clusternode2    |
30201 |        2 |DC2       |YES          |SYNCMEM     |ACTIVE      |               |        True |
|RH2      |clusternode1 |30207 |xsengine     |       2 |      1 |DC1       |clusternode2    |    30207 |
```

```
2 |DC2      |YES          |SYNCMEM     |ACTIVE      |            |       True |
|RH2      |clusternode1 |30203 |indexserver  |       3 |     1 |DC1      |clusternode2    |    30203
|       2 |DC2      |YES          |SYNCMEM     |ACTIVE      |            |       True |

status system replication site "3": ACTIVE
status system replication site "2": ACTIVE
overall system replication status: ACTIVE

Local System Replication State
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

mode: PRIMARY
site id: 1
site name: DC1
```

- これで適切な環境が整い、3 つのノードすべてで、システムレプリケーションステータスのモニタリングを別のウィンドウで開始できるようになりました。テストを開始する前に、3 つのモニターを開始する必要があります。テストが実行されると出力が変更されます。したがって、テストが完了するまでは実行し続けてください。
  古いプライマリーノードでは、テスト中、clusternode1 が別のウィンドウで実行されました。

```
clusternode1:rh2adm> watch -n 5 'python
/usr/sap/${SAPSYSTEMNAME}/HDB${TINSTANCE}/exe/python_support/systemReplicationSt
atus.py ; echo Status $?'
```

clusternode1 の出力は次のようになります。

```
Every 5.0s: python
/usr/sap/${SAPSYSTEMNAME}/HDB${TINSTANCE}/exe/python_support/systemReplicati...
clusternode1: Tue XXX XX HH:MM:SS 2023

|Database |Host   |Port  |Service Name |Volume ID |Site ID |Site Name |Secondary
|Secondary |Secondary |Secondary |Secondary     |
Replication |Replication |Replication    |Secondary    |
|        |   | |            |         |        |         |Host  |Port      |Site ID   |Site Name |Active Status |
Mode        |Status  |Status Details |Fully Synced |
|-------- |------ |----- |------------ |--------- |------- |--------- |--------- |--------- |--------- |----
--------- |
----------- |----------- |-------------- |------------ |
|SYSTEMDB |clusternode1 |30201 |nameserver   |       1 |  1 |DC1       |remotehost3    |
30201 |       3 |DC3   |YES   |
ASYNC       |ACTIVE  |             |       True |
|RH2   |clusternode1 |30207 |xsengine     |       2 |  1 |DC1       |remotehost3    |    30207 |
3 |DC3   |YES   |
ASYNC       |ACTIVE  |             |       True |
|RH2   |clusternode1 |30203 |indexserver  |       3 |  1 |DC1       |remotehost3    |    30203 |
3 |DC3   |YES   |
ASYNC       |ACTIVE  |             |       True |
|SYSTEMDB |clusternode1 |30201 |nameserver   |       1 |  1 |DC1       |clusternode2    |
30201 |       2 |DC2   |YES   |
SYNCMEM     |ACTIVE  |             |       True |
|RH2   |clusternode1 |30207 |xsengine     |       2 |  1 |DC1       |clusternode2    |    30207 |
2 |DC2   |YES   |
SYNCMEM     |ACTIVE  |             |       True |
|RH2   |clusternode1 |30203 |indexserver  |       3 |  1 |DC1       |clusternode2    |    30203 |
```

```
2 |DC2   |YES   |
SYNCMEM    |ACTIVE  |             |       True |

status system replication site "3": ACTIVE
status system replication site "2": ACTIVE
overall system replication status: ACTIVE

Local System Replication State
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

mode: PRIMARY
site id: 1
site name: DC1
Status 15
```

remotehost3 で同じコマンドを実行します。

```
remotehost3:rh2adm> watch -n 5 'python
/usr/sap/${SAPSYSTEMNAME}/HDB${TINSTANCE}/exe/python_support/systemReplicationSt
atus.py ; echo Status $?'
```

結果は次のようになります。

```
this system is either not running or is not primary system replication site
```

これは、テストでフェイルオーバーが開始された後に変更されます。出力は、テスト開始前のプライマリーノードの例と似ています。

2 番目のノードで、以下を開始します。

```
clusternode2:rh2adm> watch -n 10 'hdbnsutil -sr_state | grep masters'
```

これにより、現在のマスターの clusternode1 が表示され、フェイルオーバーの開始直後に切り替わります。

- すべてが正しく設定されていることを確認するには、**global.ini** も確認します。
- DC1、DC2、DC3 の **global.ini** を確認します。
  3 つのノードすべてで、**global.ini** に次の内容が含まれている必要があります。

```
[persistent]
log_mode=normal
[system_replication]
register_secondaries_on_takeover=true
```

以下を使用して **global.ini** を編集できます。

```
clusternode1:rh2adm>vim
/usr/sap/${SAPSYSTEMNAME}/SYS/global/hdb/custom/config/global.ini
```

- (オプション) クラスターを **maintenance-mode** にします。

```
[root@clusternode1]# pcs property set maintenance-mode=true
```

テスト中に、**maintenance-mode** を設定してもしなくても、フェイルオーバーが機能することがわかります。したがって、最初のテストは設定なしで実行できます。回復中にこれを設定する必要があります。ここでは、この設定があってもなくても機能することを示しています。これは、プライマリーにアクセスできない場合のオプションです。

- テストを開始します: DC3 にフェイルオーバーします。remotehost3 で次を実行してください。

```
remotehost3:rh2adm> hdbnsutil -sr_takeover
done.
```

テストが開始されました。以前に開始したモニターの出力を確認してください。clusternode1 では、システムレプリケーションステータスは remotehost3 および clusternode2 (DC2) との関係を失います。

```
Every 5.0s: python /usr/sap/RH2/HDB02/exe/python_support/systemReplicationStatus.py ;
echo Status $?                          clusternode1: Mon Sep  4 11:52:16 2023

|Database |Host   |Port  |Service Name |Volume ID |Site ID |Site Name |Secondary
|Secondary |Secondary |Secondary |Secondary     |Replication |Replication |Replic
ation                  |Secondary    |
|        |       |      |             |          |        |          |Host      |Port      |Site ID   |Site Name |Active
Status |Mode         |Status      |Status
 Details               |Fully Synced |
|-------- |------ |----- |------------ |--------- |------- |--------- |--------- |--------- |--------- |---
--------- |----------- |----------- |------
---------------------- |------------ |
|SYSTEMDB |clusternode1 |30201 |nameserver   |        1 |      1 |DC1       |clusternode2    |
30201 |        2 |DC2       |YES          |SYNCMEM     |ERROR       |Commun
ication channel closed |       False |
|RH2      |clusternode1 |30207 |xsengine     |        2 |      1 |DC1       |clusternode2    |    30207 |
2 |DC2       |YES          |SYNCMEM     |ERROR       |Commun
ication channel closed |       False |
|RH2      |clusternode1 |30203 |indexserver  |        3 |      1 |DC1       |clusternode2    |    30203
|        2 |DC2       |YES          |SYNCMEM     |ERROR       |Commun
ication channel closed |       False |

status system replication site "2": ERROR
overall system replication status: ERROR

Local System Replication State
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

mode: PRIMARY
site id: 1
site name: DC1
Status 11
```

クラスターは、この動作をまだ把握していません。システムレプリケーションステータスの戻りコードを確認すると、戻りコード 11 はエラーを意味し、何か問題があることがわかります。アクセスできる場合は、今すぐ **maintenance-mode** に入ることを推奨します。

remotehost3 が新しいプライマリーになり、clusternode2 (DC2) が remotehost3 上の新しいプライマリーとして自動的に登録されます。

remotehost3 のシステムレプリケーション状態の出力例:

```
Every 5.0s: python /usr/sap/RH2/HDB02/exe/python_support/systemReplicationStatus.py ;
echo Status $?                          remotehost3: Mon Sep  4 13:55:29 2023

|Database |Host   |Port  |Service Name |Volume ID |Site ID |Site Name |Secondary
|Secondary |Secondary |Secondary |Secondary     |Replication |Replication |Replic
ation    |Secondary    |
|         |       |      |             |          |        |          |Host      |Port      |Site ID   |Site Name |Active
Status |Mode        |Status      |Status
 Details |Fully Synced |
|-------- |------ |----- |------------ |--------- |------- |--------- |--------- |--------- |--------- |---
--------- |----------- |----------- |------
-------- |------------ |
|SYSTEMDB |remotehost3 |30201 |nameserver   |        1 |      3 |DC3       |clusternode2    |
30201 |        2 |DC2       |YES           |SYNCMEM     |ACTIVE      |
         |        True |
|RH2      |remotehost3 |30207 |xsengine     |        2 |      3 |DC3       |clusternode2    |    30207 |
2 |DC2       |YES           |SYNCMEM     |ACTIVE      |
         |        True |
|RH2      |remotehost3 |30203 |indexserver  |        3 |      3 |DC3       |clusternode2    |    30203
|        2 |DC2       |YES           |SYNCMEM     |ACTIVE      |
         |        True |

status system replication site "2": ACTIVE
overall system replication status: ACTIVE

Local System Replication State
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

mode: PRIMARY
site id: 3
site name: DC3
Status 15
```

戻りコード 15 も、すべてが正常であることを示していますが、clusternode1 が見つかりません。これは手動で再登録する必要があります。以前のプライマリー clusternode1 はリスト表示されないため、レプリケーション関係は失われます。

- **maintenance-mode** を設定します。
まだ行っていない場合は、次のコマンドを使用して、クラスターの 1 つのノードでクラスターに **maintenance-mode** を設定します。

```
[root@clusternode1]# pcs property  set maintenance-mode=true
```

このコマンドを実行すると、**maintenance-mode** がアクティブか確認できます。

```
[root@clusternode1]# pcs resource
  * Clone Set: SAPHanaTopology_RH2_02-clone [SAPHanaTopology_RH2_02]
(unmanaged):
    * SAPHanaTopology_RH2_02    (ocf::heartbeat:SAPHanaTopology):       Started
clusternode2node2 (unmanaged)
    * SAPHanaTopology_RH2_02    (ocf::heartbeat:SAPHanaTopology):       Started
clusternode1node1 (unmanaged)
  * Clone Set: SAPHana_RH2_02-clone [SAPHana_RH2_02] (promotable, unmanaged):
    * SAPHana_RH2_02    (ocf::heartbeat:SAPHana):       Slave clusternode2node2
(unmanaged)
```

```
    * SAPHana_RH2_02    (ocf::heartbeat:SAPHana):       Master clusternode1node1
(unmanaged)
  * vip_RH2_02_MASTER   (ocf::heartbeat:IPaddr2):       Started clusternode1node1
(unmanaged)
```

リソースは unmanaged を表示しています。これは、クラスターが **maintenance-mode=true** であることを示します。仮想 IP アドレスは引き続き clusternode1 で開始されます。この IP を別のノードで使用する場合は、maintanence-mode=true を設定する前に **vip_RH2_02_MASTER** を無効にしてください。

```
[root@clusternode1]# pcs resource disable vip_RH2_02_MASTER
```

- clusternode1 を再登録します。
clusternode1 の **sr_state** を確認すると、DC2 のみとの関係がわかります。

```
clusternode1:rh2adm> hdbnsutil -sr_state

System Replication State
~~~~~~~~~~~~~~~~~~~~~~~~

online: true

mode: primary
operation mode: primary
site id: 1
site name: DC1

is source system: true
is secondary/consumer system: false
has secondaries/consumers attached: true
is a takeover active: false
is primary suspended: false

Host Mappings:
~~~~~~~~~~~~~~

clusternode1 -> [DC2] clusternode2
clusternode1 -> [DC1] clusternode1


Site Mappings:
~~~~~~~~~~~~~~
DC1 (primary/primary)
    |---DC2 (syncmem/logreplay)

Tier of DC1: 1
Tier of DC2: 2

Replication mode of DC1: primary
Replication mode of DC2: syncmem

Operation mode of DC1: primary
Operation mode of DC2: logreplay
```

```
Mapping: DC1 -> DC2
done.
```

しかし、DC2 を確認すると、プライマリーデータベースサーバーは DC3 です。したがって、DC1 からの情報は正しくありません。

```
clusternode2:rh2adm> hdbnsutil -sr_state
```

DC1 でシステムレプリケーションステータスを確認すると、リターンコードは 12 ですが、これは不明です。したがって、DC1 を再登録する必要があります。

このコマンドを使用すると、以前のプライマリー clusternode1 を remotehost3 の新しいセカンダリーとして登録できます。

```
clusternode1:rh2adm> hdbnsutil -sr_register --remoteHost=remotehost3 --
remoteInstance=${TINSTANCE} --replicationMode=asyncsyncmem --name=DC1 --
remoteName=DC3 --operationMode=logreplay --online
```

登録が完了すると、remotehost3 上で 3 つのサイトすべてが複製され、ステータス (戻りコード) が 15 に変わります。

これが失敗した場合は、DC1 と DC3 のレプリケーション関係を手動で削除する必要があります。セカンダリーの登録 に記載されている手順に従ってください。

たとえば、次のように既存の関係をリストします。

```
clusternode1:rh2adm> hdbnsutil -sr_state
```

既存の関係を削除するには、次を使用できます。

```
clusternode1:rh2adm> hdbnsutil -sr_unregister --name=DC2`
```

通常、これは必要ないかもしれません。テスト 4 は、テスト 3 の後に実行することを想定したものです。回復手順は、テスト 4 を実行することです。

5.6. テスト 4: プライマリーノードを 1 番目のサイトにフェイルバックする

テストの内容	プライマリーがクラスターノードに戻る。 フェイルバックしてクラスターを再度有効にする。 3 番目のサイトをセカンダリーとして再登録する。

テストの前提条件	• SAP HANA プライマリーノードが 3 番目のサイトで稼働している。 • クラスターが部分的に稼働している。 • クラスターが **maintenance_mode** になっている。 • 以前のクラスターのプライマリーが検出可能である。
テストの手順	クラスターの予想されるプライマリーを確認します。 DC3 ノードから DC1 ノードにフェイルオーバーします。 以前のセカンダリーが新しいプライマリーに切り替わったか確認します。 新しいセカンダリーとして remotehost3 を再登録します。 クラスターを **maintenance_mode=false** に設定し、クラスターを引き続き動作させます。
テストのモニタリング	新しいプライマリーの起動時に、次のコマンドを実行します。 **remotehost3:rh2adm> watch python ${DIR_EXECUTABLES}/python_support/systemReplicationStatus.py** **[root@clusternode1]# watch pcs status --full** セカンダリーの起動時: **clusternode:rh2adm> watch hdbnsutil -sr_state**

テストの開始	クラスターの予想されるプライマリーを確認します: **[root@clusternode1]# pcs resource** VIP およびプロモートされた SAP HANA リソースを、新しい潜在的なプライマリーである同じノード上で実行する必要があります。 この潜在的なプライマリーでは、**sidadm** として実行します: **clusternode1:rh2adm> hdbnsutil -sr_takeover** 以前のプライマリーを新しいセカンダリーとして再登録します。 **clusternode1:rh2adm> hdbnsutil -sr_register \ --remoteHost=clusternode1 \ --remoteInstance=${TINSTANCE} \ --replicationMode=syncmem \ --name=DC3 \ --remoteName=DC1 \ --operationMode=logreplay \ --force_full_replica \ --online** **maintenance_mode=false** を設定すると、クラスターが引き続き動作します。
期待される結果	新しいプライマリーが SAP HANA を起動します。 レプリケーションステータスで、3 つのサイトすべてがレプリケートされたことが示されます。 2 番目のクラスターサイトが、新しいプライマリーに自動的に再登録されます。 DR サイトがデータベースの追加レプリカになります。
初期状態に戻す方法	テスト 3 を実行します。

詳細な説明:

- クラスターが **maintenance-mode** になっているか確認します。

```
[root@clusternode1]# pcs property config maintenance-mode
Cluster Properties:
 maintenance-mode: true
```

maintenance-mode が true でない場合は、以下で設定できます。

```
[root@clusternode1]# pcs property set  maintenance-mode=true
```

- システムのレプリケーションステータスを確認し、すべてのノード上のプライマリーデータベースを検出します。
まず、次を使用してプライマリーデータベースを検出します。

```
clusternode1:rh2adm> hdbnsutil -sr_state | egrep -e "^mode:|primary masters"
```

出力は、以下のようになります。

clusternode1 上

```
clusternode1:rh2adm> hdbnsutil -sr_state | egrep -e "^mode:|primary masters"
mode: syncmem
primary masters: remotehost3
```

clusternode2 上

```
clusternode2:rh2adm> hdbnsutil -sr_state | egrep -e "^mode:|primary masters"
mode: syncmem
primary masters: remotehost3
```

remotehost3 上

```
remotehost3:rh2adm> hdbnsutil -sr_state | egrep -e "^mode:|primary masters"
mode: primary
```

3 つのノードすべてで、プライマリーデータベースは remotehost3 です。

このプライマリーデータベースでは、システムレプリケーションステータスが 3 つのノードすべてでアクティブであり、戻りコードが 15 であることを確認する必要があります。

```
remotehost3:rh2adm> python
/usr/sap/${SAPSYSTEMNAME}/HDB${TINSTANCE}/exe/python_support/systemReplicationSt
atus.py
|Database |Host   |Port  |Service Name |Volume ID |Site ID |Site Name |Secondary
|Secondary |Secondary |Secondary |Secondary     |Replication |Replication |Replication
|Secondary    |
|         |       |      |             |          |        |          |Host      |Port      |Site ID   |Site Name |Active
Status |Mode        |Status      |Status Details |Fully Synced |
|-------- |------ |----- |------------ |--------- |------- |--------- |--------- |--------- |--------- |---
--------- |----------- |----------- |-------------- |------------ |
|SYSTEMDB |remotehost3 |30201 |nameserver   |        1 |      3 |DC3       |clusternode2    |
30201 |        2 |DC2       |YES           |SYNCMEM     |ACTIVE      |             |        True |
|RH2      |remotehost3 |30207 |xsengine     |        2 |      3 |DC3       |clusternode2    |    30207 |
2 |DC2       |YES           |SYNCMEM     |ACTIVE      |             |        True |
|RH2      |remotehost3 |30203 |indexserver  |        3 |      3 |DC3       |clusternode2    |    30203
|        2 |DC2       |YES           |SYNCMEM     |ACTIVE      |             |        True |
|SYSTEMDB |remotehost3 |30201 |nameserver   |        1 |      3 |DC3       |clusternode1    |
30201 |        1 |DC1       |YES           |SYNCMEM     |ACTIVE      |             |        True |
|RH2      |remotehost3 |30207 |xsengine     |        2 |      3 |DC3       |clusternode1    |    30207 |
1 |DC1       |YES           |SYNCMEM     |ACTIVE      |             |        True |
|RH2      |remotehost3 |30203 |indexserver  |        3 |      3 |DC3       |clusternode1    |    30203
|        1 |DC1       |YES           |SYNCMEM     |ACTIVE      |             |        True |

status system replication site "2": ACTIVE
status system replication site "1": ACTIVE
overall system replication status: ACTIVE

Local System Replication State
```

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

mode: PRIMARY
site id: 3
site name: DC3
[rh2adm@remotehost3: python_support]# echo $?
15
```

- 3 つの **sr_state** がすべて一貫しているか確認します。
3 つのノードすべてで、**hdbnsutil -sr_state --sapcontrol=1 |grep site.*Mode** を実行してください。

```
clusternode1:rh2adm>hdbnsutil -sr_state --sapcontrol=1 |grep  site.*Mode


clusternode2:rh2adm> hsbnsutil -sr_state --sapcontrol=1 | grep site.*Mode


remotehost3:rh2adm>hsbnsutil -sr_state --sapcontrol=1 | grep site.*Mode
```

出力はすべてのノードで同じになるはずです。

```
siteReplicationMode/DC1=primary
siteReplicationMode/DC3=async
siteReplicationMode/DC2=syncmem
siteOperationMode/DC1=primary
siteOperationMode/DC3=logreplay
siteOperationMode/DC2=logreplay
```

- 別のウィンドウでモニタリングを開始します。
clusternode1 で、以下を開始します。

```
clusternode1:rh2adm> watch "python
/usr/sap/${SAPSYSTEMNAME}/HDB${TINSTANCE}/exe/python_support/systemReplicationSt
atus.py; echo \$?"
```

remotehost3 で、以下を開始します。

```
remotehost3:rh2adm>watch "python
/usr/sap/${SAPSYSTEMNAME}/HDB${TINSTANCE}/exe/python_support/systemReplicationSt
atus.py; echo \$?"
```

clusternode2 で、以下を開始します。

```
clusternode2:rh2adm> watch "hdbnsutil -sr_state --sapcontrol=1 |grep  siteReplicationMode"
```

- テストを開始します。
clusternode1 にフェイルオーバーするには、clusternode1 で開始します。

```
clusternode1:rh2adm> hdbnsutil -sr_takeover
done.
```

- モニターの出力を確認してください。

clusternode1 のモニターは、次のように変更されます。

```
Every 2.0s: python systemReplicationStatus.py; echo $?
clusternode1: Mon Sep  4 23:34:30 2023

|Database |Host   |Port  |Service Name |Volume ID |Site ID |Site Name |Secondary
|Secondary |Secondary |Secondary |Secondary     |Replication |Replication |Replication
|Secondary    |
|         |       |      |             |          |        |          |Host      |Port      |Site ID   |Site Name |Active
Status |Mode         |Status      |Status Details |Fully Synced |
|-------- |------ |----- |------------ |--------- |------- |--------- |--------- |--------- |--------- |---
--------- |----------- |----------- |-------------- |------------ |
|SYSTEMDB |clusternode1 |30201 |nameserver   |       1 |      1 |DC1       |clusternode2    |
30201 |        2 |DC2       |YES          |SYNCMEM     |ACTIVE      |             |        True |
|RH2      |clusternode1 |30207 |xsengine     |       2 |      1 |DC1       |clusternode2    |    30207 |
2 |DC2       |YES          |SYNCMEM     |ACTIVE      |             |        True |
|RH2      |clusternode1 |30203 |indexserver  |       3 |      1 |DC1       |clusternode2    |    30203
|        2 |DC2       |YES          |SYNCMEM     |ACTIVE      |             |        True |

status system replication site "2": ACTIVE
overall system replication status: ACTIVE

Local System Replication State
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

mode: PRIMARY
site id: 1
site name: DC1
15
```

戻りコード 15 も重要です。

clusternode2 のモニターは、次のように変更されます。

```
Every 2.0s: hdbnsutil -sr_state --sapcontrol=1 |grep  site.*Mode
clusternode2: Mon Sep  4 23:35:18 2023

siteReplicationMode/DC1=primary
siteReplicationMode/DC2=syncmem
siteOperationMode/DC1=primary
siteOperationMode/DC2=logreplay
```

DC3 は失われているため、再登録する必要があります。

remotehost3 では、**systemReplicationStatus** がエラーを報告し、リターンコードが 11 に変更されます。

- クラスターノードが再登録されているか確認します。

```
clusternode1:rh2adm>  hdbnsutil -sr_state

System Replication State
~~~~~~~~~~~~~~~~~~~~~~~~

online: true
```

```
mode: primary
operation mode: primary
site id: 1
site name: DC1

is source system: true
is secondary/consumer system: false
has secondaries/consumers attached: true
is a takeover active: false
is primary suspended: false

Host Mappings:
~~~~~~~~~~~~~~~

clusternode1 -> [DC2] clusternode2
clusternode1 -> [DC1] clusternode1


Site Mappings:
~~~~~~~~~~~~~~~
DC1 (primary/primary)
    |---DC2 (syncmem/logreplay)

Tier of DC1: 1
Tier of DC2: 2

Replication mode of DC1: primary
Replication mode of DC2: syncmem

Operation mode of DC1: primary
Operation mode of DC2: logreplay

Mapping: DC1 -> DC2
done.
```

サイトマッピングには、clusternode2 (DC2) が再登録されたことが示されています。

- vip リソースを確認または有効にします。

```
[root@clusternode1]# pcs resource
  * Clone Set: SAPHanaTopology_RH2_02-clone [SAPHanaTopology_RH2_02]
(unmanaged):
    * SAPHanaTopology_RH2_02    (ocf::heartbeat:SAPHanaTopology):       Started
clusternode2 (unmanaged)
    * SAPHanaTopology_RH2_02    (ocf::heartbeat:SAPHanaTopology):       Started
clusternode1 (unmanaged)
  * Clone Set: SAPHana_RH2_02-clone [SAPHana_RH2_02] (promotable, unmanaged):
    * SAPHana_RH2_02    (ocf::heartbeat:SAPHana):       Master clusternode2 (unmanaged)
    * SAPHana_RH2_02    (ocf::heartbeat:SAPHana):       Slave clusternode1 (unmanaged)
  * vip_RH2_02_MASTER   (ocf::heartbeat:IPaddr2):       Stopped (disabled, unmanaged)
```

vip リソース **vip_RH2_02_MASTER** が停止されています。

再度開始するには、次のコマンドを実行します。

```
[root@clusternode1]# pcs resource enable vip_RH2_02_MASTER
Warning: 'vip_RH2_02_MASTER' is unmanaged
```

クラスターは、**maintenance-mode=false** でない限りリソースを開始しないため、この警告は正しいです。

- クラスター **maintenance-mode** を停止します。
maintenance-mode を停止する前に、別のウィンドウで 2 つのモニターを起動して、変更を確認する必要があります。

  clusternode2 で、次を実行します。

```
[root@clusternode2]# watch pcs status --full
```

  clusternode1 で、次を実行します。

```
clusternode1:rh2adm> watch "python
/usr/sap/${SAPSYSTEMNAME}/HDB${TINSTANCE}/exe/python_support/systemReplicationSt
atus.py; echo $?"
```

  これで、次のコマンドを実行して、clusternode1 の **maintenance-mode** の設定を解除できます。

```
[root@clusternode1]# pcs property set maintenance-mode=false
```

  clusternode1 のモニターには、すべてが期待どおりに実行されていることが表示されます。

```
Every 2.0s: pcs status --full
clusternode1: Tue Sep  5 00:01:17 2023

Cluster name: cluster1
Cluster Summary:
  * Stack: corosync
  * Current DC: clusternode1 (1) (version 2.1.2-4.el8_6.6-ada5c3b36e2) - partition with
quorum
  * Last updated: Tue Sep  5 00:01:17 2023
  * Last change:  Tue Sep  5 00:00:30 2023 by root via crm_attribute on clusternode1
  * 2 nodes configured
  * 6 resource instances configured

Node List:
  * Online: [ clusternode1 (1) clusternode2 (2) ]

Full List of Resources:
  * auto_rhevm_fence1   (stonith:fence_rhevm):   Started clusternode1
  * Clone Set: SAPHanaTopology_RH2_02-clone [SAPHanaTopology_RH2_02]:
    * SAPHanaTopology_RH2_02    (ocf::heartbeat:SAPHanaTopology):       Started
clusternode2
    * SAPHanaTopology_RH2_02    (ocf::heartbeat:SAPHanaTopology):       Started
clusternode1
  * Clone Set: SAPHana_RH2_02-clone [SAPHana_RH2_02] (promotable):
    * SAPHana_RH2_02    (ocf::heartbeat:SAPHana):       Slave clusternode2
    * SAPHana_RH2_02    (ocf::heartbeat:SAPHana):       Master clusternode1
  * vip_RH2_02_MASTER   (ocf::heartbeat:IPaddr2):       Started clusternode1
```

```
Node Attributes:
  * Node: clusternode1 (1):
    * hana_rh2_clone_state              : PROMOTED
    * hana_rh2_op_mode                  : logreplay
    * hana_rh2_remoteHost               : clusternode2
    * hana_rh2_roles                    : 4:P:master1:master:worker:master
    * hana_rh2_site                     : DC1
    * hana_rh2_sra                      : -
    * hana_rh2_srah                     : -
    * hana_rh2_srmode                   : syncmem
    * hana_rh2_sync_state               : PRIM
    * hana_rh2_version                  : 2.00.062.00
    * hana_rh2_vhost                    : clusternode1
    * lpa_rh2_lpt                       : 1693872030
    * master-SAPHana_RH2_02             : 150
  * Node: clusternode2 (2):
    * hana_rh2_clone_state              : DEMOTED
    * hana_rh2_op_mode                  : logreplay
    * hana_rh2_remoteHost               : clusternode1
    * hana_rh2_roles                    : 4:S:master1:master:worker:master
    * hana_rh2_site                     : DC2
    * hana_rh2_sra                      : -
    * hana_rh2_srah                     : -
    * hana_rh2_srmode                   : syncmem
    * hana_rh2_sync_state               : SOK
    * hana_rh2_version                  : 2.00.062.00
    * hana_rh2_vhost                    : clusternode2
    * lpa_rh2_lpt                       : 30
    * master-SAPHana_RH2_02             : 100

Migration Summary:

Tickets:

PCSD Status:
  clusternode1: Online
  clusternode2: Online

Daemon Status:
  corosync: active/disabled
  pacemaker: active/disabled
  pcsd: active/enabled
```

手動操作の後は、クラスターのクリーンアップ で説明されているように、クラスターをクリーンアップすることを推奨します。

- clusternode1 上の新しいプライマリーに remotehost3 を再登録します。
remotehost3 を再登録する必要があります。進行状況を監視するには、clusternode1 で開始してください。

```
con_cluster_cleanupclusternode1:rh2adm> watch -n 5 'python
/usr/sap/${SAPSYSTEMNAME}/HDB${TINSTANCE}/exe/python_support/systemReplicationSt
atus.py ; echo Status $?'
```

remotehost3 で、以下を開始してください。

```
remotehost3:rh2adm> watch 'hdbnsutil -sr_state --sapcontrol=1 |grep  siteReplicationMode'
```

これで、このコマンドを使用して、remotehost3 を再登録できます。

```
remotehost3:rh2adm> hdbnsutil -sr_register --remoteHost=clusternode1 --
remoteInstance=${TINSTANCE} --replicationMode=async --name=DC3 --remoteName=DC1
--operationMode=logreplay --online
```

clusternode1 のモニターは、次のように変更されます。

```
Every 5.0s: python
/usr/sap/${SAPSYSTEMNAME}/HDB${TINSTANCE}/exe/python_support/systemReplicationSt
atus.py ; echo Status $?                                                    clusternode1:
Tue Sep  5 00:14:40 2023

|Database |Host   |Port  |Service Name |Volume ID |Site ID |Site Name |Secondary
|Secondary |Secondary |Secondary |Secondary     |Replication |Replication |Replication
|Secondary    |
|         |       |      |             |          |        |          |Host      |Port      |Site ID   |Site Name |Active
Status |Mode         |Status       |Status Details |Fully Synced |
|-------- |------ |----- |------------ |--------- |------- |--------- |--------- |--------- |--------- |----
--------- |----------- |----------- |-------------- |------------ |
|SYSTEMDB |clusternode1 |30201 |nameserver   |        1 |      1 |DC1       |remotehost3    |
30201 |        3 |DC3       |YES          |ASYNC     |ACTIVE      |              |        True |
|RH2      |clusternode1 |30207 |xsengine     |        2 |      1 |DC1       |remotehost3    |    30207 |
3 |DC3       |YES          |ASYNC     |ACTIVE      |              |        True |
|RH2      |clusternode1 |30203 |indexserver  |        3 |      1 |DC1       |remotehost3    |    30203
|        3 |DC3       |YES          |ASYNC     |ACTIVE      |              |        True |
|SYSTEMDB |clusternode1 |30201 |nameserver   |        1 |      1 |DC1       |clusternode2    |
30201 |        2 |DC2       |YES          |SYNCMEM     |ACTIVE      |              |        True |
|RH2      |clusternode1 |30207 |xsengine     |        2 |      1 |DC1       |clusternode2    |    30207 |
2 |DC2       |YES          |SYNCMEM     |ACTIVE      |              |        True |
|RH2      |clusternode1 |30203 |indexserver  |        3 |      1 |DC1       |clusternode2    |    30203
|        2 |DC2       |YES          |SYNCMEM     |ACTIVE      |              |        True |

status system replication site "3": ACTIVE
status system replication site "2": ACTIVE
overall system replication status: ACTIVE

Local System Replication State
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

mode: PRIMARY
site id: 1
site name: DC1
Status 15
```

そして、remotehost3 のモニターは次のように変更されます。

```
Every 2.0s: hdbnsutil -sr_state --sapcontrol=1 |grep  site.*Mode
remotehost3: Tue Sep  5 02:15:28 2023

siteReplicationMode/DC1=primary
siteReplicationMode/DC3=syncmem
siteReplicationMode/DC2=syncmem
```

```
siteOperationMode/DC1=primary
siteOperationMode/DC3=logreplay
siteOperationMode/DC2=logreplay
```

ここで再び 3 つのエントリーがあり、remotehost3 (DC3) が再び clusternode1 (DC1) から複製されたセカンダリーサイトになります。

- すべてのノードが、clusternode1 上のシステムレプリケーションステータスの一部であるか確認します。
  3 つのノードすべてで、**hdbnsutil -sr_state --sapcontrol=1 |grep site.*Mode** を実行してください。

```
clusternode1:rh2adm> hdbnsutil -sr_state --sapcontrol=1 |grep
site.*ModesiteReplicationMode


clusternode2:rh2adm> hsbnsutil -sr_state --sapcontrol=1 | grep site.*Mode


remotehost3:rh2adm> hsbnsutil -sr_state --sapcontrol=1 | grep site.*Mode
```

すべてのノードで同じ出力が得られるはずです。

```
siteReplicationMode/DC1=primary
siteReplicationMode/DC3=syncmem
siteReplicationMode/DC2=syncmem
siteOperationMode/DC1=primary
siteOperationMode/DC3=logreplay
siteOperationMode/DC2=logreplay
```

- pcs ステータス --full および SOK を確認します。
  以下を実行します。

```
[root@clusternode1]# pcs status --full| grep sync_state
```

出力は PRIM または SOK のいずれかである必要があります。

```
 * hana_rh2_sync_state            : PRIM
   * hana_rh2_sync_state            : SOK
```

最後に、**sync_state** PRIM と SOK を含むクラスターのステータスは次のようになります。

```
[root@clusternode1]# pcs status --full
Cluster name: cluster1
Cluster Summary:
  * Stack: corosync
  * Current DC: clusternode1 (1) (version 2.1.2-4.el8_6.6-ada5c3b36e2) - partition with
quorum
  * Last updated: Tue Sep  5 00:18:52 2023
  * Last change:  Tue Sep  5 00:16:54 2023 by root via crm_attribute on clusternode1
  * 2 nodes configured
  * 6 resource instances configured

Node List:
```

```
  * Online: [ clusternode1 (1) clusternode2 (2) ]

Full List of Resources:
  * auto_rhevm_fence1   (stonith:fence_rhevm):   Started clusternode1
  * Clone Set: SAPHanaTopology_RH2_02-clone [SAPHanaTopology_RH2_02]:
    * SAPHanaTopology_RH2_02    (ocf::heartbeat:SAPHanaTopology):       Started
clusternode2
    * SAPHanaTopology_RH2_02    (ocf::heartbeat:SAPHanaTopology):       Started
clusternode1
  * Clone Set: SAPHana_RH2_02-clone [SAPHana_RH2_02] (promotable):
    * SAPHana_RH2_02    (ocf::heartbeat:SAPHana):       Slave clusternode2
    * SAPHana_RH2_02    (ocf::heartbeat:SAPHana):       Master clusternode1
  * vip_RH2_02_MASTER   (ocf::heartbeat:IPaddr2):       Started clusternode1

Node Attributes:
  * Node: clusternode1 (1):
    * hana_rh2_clone_state              : PROMOTED
    * hana_rh2_op_mode                  : logreplay
    * hana_rh2_remoteHost               : clusternode2
    * hana_rh2_roles                    : 4:P:master1:master:worker:master
    * hana_rh2_site                     : DC1
    * hana_rh2_sra                      : -
    * hana_rh2_srah                     : -
    * hana_rh2_srmode                   : syncmem
    * hana_rh2_sync_state               : PRIM
    * hana_rh2_version                  : 2.00.062.00
    * hana_rh2_vhost                    : clusternode1
    * lpa_rh2_lpt                       : 1693873014
    * master-SAPHana_RH2_02             : 150
  * Node: clusternode2 (2):
    * hana_rh2_clone_state              : DEMOTED
    * hana_rh2_op_mode                  : logreplay
    * hana_rh2_remoteHost               : clusternode1
    * hana_rh2_roles                    : 4:S:master1:master:worker:master
    * hana_rh2_site                     : DC2
    * hana_rh2_sra                      : -
    * hana_rh2_srah                     : -
    * hana_rh2_srmode                   : syncmem
    * hana_rh2_sync_state               : SOK
    * hana_rh2_version                  : 2.00.062.00
    * hana_rh2_vhost                    : clusternode2
    * lpa_rh2_lpt                       : 30
    * master-SAPHana_RH2_02             : 100

Migration Summary:

Tickets:

PCSD Status:
  clusternode1: Online
  clusternode2: Online

Daemon Status:
  corosync: active/disabled
  pacemaker: active/disabled
  pcsd: active/enabled
```

- すべてが再び正常に動作することを確認するには、クラスターのステータスの確認 および データベースの確認 を参照してください。

第6章 便利なコマンド

以下の 3 つのセクションで、便利なコマンドを紹介します。ほとんどの場合、操作や設定の成功を確認するのに役立ちます。例を応答とともに記載しています。出力は、形式上の理由で調整されている場合があります。

注記

- このドキュメントにリストされているすべてのコマンドには、**<sid>adm** ユーザーが実行するものである場合、先頭に **>** が付いています。
- **root user** が実行するすべてのコマンドには、先頭に **#** が付いています。
- コマンドを実行するには、接頭辞の **>** または **#** を除きます。

6.1. SAP HANA コマンド

SAP HANA コマンドは、**<sid>adm** ユーザーによって実行されます。以下に例を示します。

```
[root@clusternode1]# su - rh2adm
clusternode1:rh2adm> cdpy
clusternode1:rh2adm> pwd
/usr/sap/RH2/HDB02/exe/python_support
clusternode1:rh2adm> python systemReplicationStatus.py -h
systemReplicationStatus.py [-h|--help] [-a|--all] [-l|--localhost] [-m|--multiTaget] [-s|--site=<site
name>] [-t|--printLandscapeTree] [--omitSecondaryActiveStatus] [--sapcontrol=1]
clusternode1:rh2adm> python landscapeHostConfiguration.py -h
landscapeHostConfiguration.py [-h|--help] [--localhost] [--sapcontrol=1]
clusternode1:rh2adm> hdbnsutil # run hdbnsutil without parameters to get help
```

6.1.1. **hdbclm** を使用した SAP HANA のインストール

3 番目のサイトのインストールは、2 番目のサイトのインストールと似ています。インストールは、root ユーザーとして **hdblcm** を使用して実行できます。以前に何もインストールされていないことを確認するには、**hdbuninst** を実行して、このノードに SAP HANA がまだインストールされていないかどうかを確認します。

HANA アンインストールの出力例:

```
[root@remotehost3]# cd /software/DATA_UNITS/HDB_SERVER_LINUX_X86_64
root@DC3/software/DATA_UNITS/HDB_SERVER_LINUX_X86_64# ./hdbuninst
Option 0 will remove an already existing HANA Installation
No SAP HANA Installation found is the expected answer
```

DC3 での HANA インストールの出力例:

```
----[root@remotehost3]# cd /software/DATA_UNITS/HDB_SERVER_LINUX_X86_64
# ./hdbuninst
Option 0 will remove an already existing HANA Installation
No SAP HANA Installation found is the expected answer
----
Example output of HANA installation:
[source,text]
```

```
----
[root@remotehost3]# ./hdblcm
1 install
2 server
/hana/shared is default directory
Enter Local Hostname [remotehost3]: use the default name
additional hosts only during Scale-Out Installation y default is n
ENTER SAP HANA System ID: RH2
Enter Instance Number [02]:
Enter Local Host Worker Group [default]:
Select System Usage / Enter Index [4]:
Choose encryption
Enter Location of Data Volumes [/hana/data/RH2]:
Enter Location of Log Volumes [/hana/log/RH2]:
Restrict maximum memory allocation? [n]:
Enter Certificate Host Name
Enter System Administrator (rh2adm) Password: <Y0urPasswd>
Confirm System Administrator (rh2adm) Password: <Y0urPasswd>
Enter System Administrator Home Directory [/usr/sap/RH2/home]:
Enter System Administrator Login Shell [/bin/sh]:
Enter System Administrator User ID [1000]:
Enter System Database User (SYSTEM) Password: <Y0urPasswd>
Confirm System Database User (SYSTEM) Password: <Y0urPasswd>
Restart system after machine reboot? [n]:
----
```

インストールが開始される前に、概要がリスト表示されます。

```
SAP HANA Database System Installation
  Installation Parameters
    Remote Execution: ssh
    Database Isolation: low
    Install Execution Mode: standard
    Installation Path: /hana/shared
    Local Host Name: dc3host
    SAP HANA System ID: RH2
    Instance Number: 02
    Local Host Worker Group: default
    System Usage: custom
    Location of Data Volumes: /hana/data/RH2
    Location of Log Volumes: /hana/log/RH2
    SAP HANA Database secure store: ssfs
    Certificate Host Names: remotehost3 -> remotehost3      System Administrator Home Directory:
/usr/sap/RH2/home
    System Administrator Login Shell: /bin/sh
    System Administrator User ID: 1000
    ID of User Group (sapsys): 1010
  Software Components
    SAP HANA Database
      Install version 2.00.052.00.1599235305
      Location: /software/DATA_UNITS/HDB_SERVER_LINUX_X86_64/server
    SAP HANA Local Secure Store
      Do not install
    SAP HANA AFL (incl.PAL,BFL,OFL)
      Do not install
    SAP HANA EML AFL
```

```
      Do not install
    SAP HANA EPM-MDS
      Do not install
    SAP HANA Database Client
      Do not install
    SAP HANA Studio
      Do not install
    SAP HANA Smart Data Access
      Do not install
    SAP HANA XS Advanced Runtime
      Do not install
  Log File Locations
    Log directory: /var/tmp/hdb_RH2_hdblcm_install_2021-06-09_18.48.13
    Trace location: /var/tmp/hdblcm_2021-06-09_18.48.13_31307.trc

Do you want to continue? (y/n):
```

y を入力してインストールを開始します。

6.1.2. hdbsql を使用した Inifile の内容の確認

```
clusternode1:rh2adm> hdbsql -i ${TINSTANCE} -u system -p Y0urP8ssw0rd

Welcome to the SAP HANA Database interactive terminal.

Type:  \h for help with commands
       \q to quit

hdbsql RH2=> select * from M_INIFILE_CONTENTS where section='system_replication'
FILE_NAME,LAYER_NAME,TENANT_NAME,HOST,SECTION,KEY,VALUE
"global.ini","DEFAULT","","","system_replication","actual_mode","primary"
"global.ini","DEFAULT","","","system_replication","mode","primary"
"global.ini","DEFAULT","","","system_replication","operation_mode","logreplay"
"global.ini","DEFAULT","","","system_replication","register_secondaries_on_takeover
","true"
"global.ini","DEFAULT","","","system_replication","site_id","1"
"global.ini","DEFAULT","","","system_replication","site_name","DC2"
"global.ini","DEFAULT","","","system_replication","timetravel_logreplay_mode","auto
"
"global.ini","DEFAULT","","","system_replication","alternative_sources",""
"global.ini","DEFAULT","","","system_replication","datashipping_logsize_threshold",
"5368709120"
"global.ini","DEFAULT","","","system_replication","datashipping_min_time_interval",
"600"
"global.ini","DEFAULT","","","system_replication","datashipping_parallel_channels",
"4"
"global.ini","DEFAULT","","","system_replication","datashipping_parallel_processing
","true"
"global.ini","DEFAULT","","","system_replication","datashipping_snapshot_max_retent
ion_time","300"
"global.ini","DEFAULT","","","system_replication","enable_data_compression","false"
"global.ini","DEFAULT","","","system_replication","enable_full_sync","false"
"global.ini","DEFAULT","","","system_replication","enable_log_compression","false"
"global.ini","DEFAULT","","","system_replication","enable_log_retention","auto"
"global.ini","DEFAULT","","","system_replication","full_replica_on_failed_delta_syn
```

```
c_check","false"
"global.ini","DEFAULT","","","system_replication","hint_based_routing_site_name",""
"global.ini","DEFAULT","","","system_replication","keep_old_style_alert","false"
"global.ini","DEFAULT","","","system_replication","logshipping_async_buffer_size","
67108864"
"global.ini","DEFAULT","","","system_replication","logshipping_async_wait_on_buffer
_full","true"
"global.ini","DEFAULT","","","system_replication","logshipping_max_retention_size",
"1048576"
"global.ini","DEFAULT","","","system_replication","logshipping_replay_logbuffer_cac
he_size","1073741824"
"global.ini","DEFAULT","","","system_replication","logshipping_replay_push_persiste
nt_segment_count","5"
"global.ini","DEFAULT","","","system_replication","logshipping_snapshot_logsize_thr
eshold","3221225472"
"global.ini","DEFAULT","","","system_replication","logshipping_snapshot_min_time_in
terval","900"
"global.ini","DEFAULT","","","system_replication","logshipping_timeout","30"
"global.ini","DEFAULT","","","system_replication","preload_column_tables","true"
"global.ini","DEFAULT","","","system_replication","propagate_log_retention","off"
"global.ini","DEFAULT","","","system_replication","reconnect_time_interval","30"
"global.ini","DEFAULT","","","system_replication","retries_before_register_to_alter
native_source","20"
"global.ini","DEFAULT","","","system_replication","takeover_esserver_without_log_ba
ckup","false"
"global.ini","DEFAULT","","","system_replication","takeover_wait_until_esserver_res
tart","true"
"global.ini","DEFAULT","","","system_replication","timetravel_call_takeover_hooks",
"off"
"global.ini","DEFAULT","","","system_replication","timetravel_log_retention_policy"
,"none"
"global.ini","DEFAULT","","","system_replication","timetravel_max_retention_time","
0"
"global.ini","DEFAULT","","","system_replication","timetravel_snapshot_creation_int
erval","1440"
"indexserver.ini","DEFAULT","","","system_replication","logshipping_async_buffer_si
ze","268435456"
"indexserver.ini","DEFAULT","","","system_replication","logshipping_replay_logbuffe
r_cache_size","4294967296"
"indexserver.ini","DEFAULT","","","system_replication","logshipping_replay_push_per
sistent_segment_count","20"
41 rows selected (overall time 1971.958 msec; server time 31.359 msec)
```

6.1.3. データベースの確認

データベースが実行されているかどうかを確認し、現在のプライマリーノードを検出します。

データベースインスタンスのリスト表示

```
clusternode1:rh2adm> sapcontrol -nr ${TINSTANCE} -function
GetSystemInstanceList

23.06.2023 12:08:17
GetSystemInstanceList
```

```
OK
hostname, instanceNr, httpPort, httpsPort, startPriority, features, dispstatus
node1, 2, 50213, 50214, 0.3, HDB|HDB_WORKER, GREEN
```

出力が緑色の場合、インスタンスは実行中です。

データベースプロセスのリスト表示

```
clusternode1:rh2adm> sapcontrol -nr ${TINSTANCE} -function GetProcessList
GetProcessList
OK
name, description, dispstatus, textstatus, starttime, elapsedtime, pid
hdbdaemon, HDB Daemon, GREEN, Running, 2023 09 04 14:34:01, 18:41:33, 3788067
hdbcompileserver, HDB Compileserver, GREEN, Running, 2023 09 04 22:35:40, 10:39:54, 445299
hdbindexserver, HDB Indexserver-RH2, GREEN, Running, 2023 09 04 22:35:40, 10:39:54, 445391
hdbnameserver, HDB Nameserver, GREEN, Running, 2023 09 04 22:35:34, 10:40:00, 445178
hdbpreprocessor, HDB Preprocessor, GREEN, Running, 2023 09 04 22:35:40, 10:39:54, 445306
hdbwebdispatcher, HDB Web Dispatcher, GREEN, Running, 2023 09 04 22:35:53, 10:39:41, 445955
hdbxsengine, HDB XSEngine-RH2, GREEN, Running, 2023 09 04 22:35:40, 10:39:54, 445394
```

通常、すべてのデータベースプロセスのステータスは **GREEN** です。

SAP HANA プロセスのリスト表示

```
clusternode1:rh2adm> HDB info
USER          PID     PPID  %CPU        VSZ        RSS COMMAND
rh2adm       1560     1559   0.0       6420       3136 watch -n 5 sapcontrol -nr 02 -functi
rh2adm       1316     1315   0.0       8884       5676 -sh
rh2adm       2549     1316   0.0       7516       4072  \_ /bin/sh /usr/sap/RH2/HDB02/HDB i
rh2adm       2579     2549   0.0      10144       3576      \_ ps fx -U rh2adm -o user:8,pi
rh2adm       2388        1   0.0     679536      55520 hdbrsutil  --start --port 30203 --vo
rh2adm       1921        1   0.0     679196      55312 hdbrsutil  --start --port 30201 --vo
rh2adm       1469        1   0.0       8852       3260 sapstart pf=/usr/sap/RH2/SYS/profile
rh2adm       1476     1469   0.7     438316      86288  \_ /usr/sap/RH2/HDB02/remotehost3/trace/
rh2adm       1501     1476  11.7    9690172    1574796      \_ hdbnameserver
rh2adm       1845     1476   0.8     410696     122988      \_ hdbcompileserver
rh2adm       1848     1476   1.0     659464     154072      \_ hdbpreprocessor
rh2adm       1899     1476  14.7    9848276    1765208      \_ hdbindexserver -port 30203
rh2adm       1902     1476   8.4    5023288    1052768      \_ hdbxsengine -port 30207
rh2adm       2265     1476   5.2    2340284     405016      \_ hdbwebdispatcher
rh2adm       1117        1   1.1     543532      30676 /usr/sap/RH2/HDB02/exe/sapstartsrv p
rh2adm       1029        1   0.0      20324      11572 /usr/lib/systemd/systemd --user
rh2adm       1030     1029   0.0      23256       3536  \_ (sd-pam)
```

SAP HANA ランドスケープ設定の表示

```
clusternode1:rh2adm>
/usr/sap/${SAPSYSTEMNAME}/HDB${TINSTANCE}/exe/Python/bin/python
/usr/sap/${SAPSYSTEMNAME}/HDB${TINSTANCE}/exe/python_support/landscapeHostConfiguration.
py;echo $?
| Host   | Host   | Host   | Failover | Remove | Storage   | Storage   | Failover | Failover | NameServer |
NameServer | IndexServer | IndexServer | Host   | Host   | Worker  | Worker  |
|        | Active | Status | Status   | Status | Config    | Actual    | Config   | Actual   | Config     | Actual     |
Config       | Actual       | Config | Actual | Config  | Actual  |
|        |        |        |          |        | Partition | Partition | Group    | Group    | Role       | Role       | Role       |
```

```
Role        | Roles  | Roles  | Groups  | Groups  |
| ------ | ------ | ------ | -------- | ------ | --------- | --------- | -------- | -------- | ---------- | ---------- | -----------
| ----------- | ------ | ------ | ------- | ------- |
| clusternode1 | yes    | ok     |          |        |         1 |         1 | default  | default  | master 1   | master     |
worker      | master      | worker | worker | default | default |

overall host status: ok
4
```

戻りコード:

- 0: Fatal
- 1: Error
- 2: Warning
- 3: Info
- 4: OK

プライマリーデータベースの検出

```
clusternode1:rh2adm> hdbnsutil -sr_state | egrep -e "primary masters|^mode"
```

セカンダリーでの確認の例:

```
clusternode1:rh2adm> hdbnsutil -sr_state | egrep -e "primary masters|^mode"
mode: syncmem
primary masters: clusternode1
```

現在のプライマリーでの確認の例:

```
clusternode1:rh2adm> hdbnsutil -sr_state | egrep -e "primary masters|^mode"
mode: primary

clusternode1:rh2adm>hdbnsutil -sr_state --sapcontrol=1 |grep  site.*Mode
siteReplicationMode/DC1=primary
siteReplicationMode/DC3=async
siteReplicationMode/DC2=syncmem
siteOperationMode/DC1=primary
siteOperationMode/DC3=logreplay
siteOperationMode/DC2=logreplay
```

データベースのバージョンの表示

SQL クエリーを使用した例:

```
hdbsql RH2=> select * from m_database
SYSTEM_ID,DATABASE_NAME,HOST,START_TIME,VERSION,USAGE
"RH2","RH2","node1","2023-06-22 15:33:05.235000000","2.00.059.02.1647435895","CUSTOM"
1 row selected (overall time 29.107 msec; server time 927 usec)
```

systemOverview.py を使用した例:

```
clusternode1:rh2adm> python ./systemOverview.py
| Section    | Name            | Status  | Value                                              |
| ---------- | --------------- | ------- | -------------------------------------------------- |
| System     | Instance ID     |         | RH2                                                |
| System     | Instance Number |         | 02                                                 |
| System     | Distributed     |         | No                                                 |
| System     | Version         |         | 2.00.059.02.1647435895 (fa/hana2sp05)              |
| System     | Platform        |         | Red Hat Enterprise Linux 9.2 Beta (Plow) 9.2 (Plow) |
| Services   | All Started     | OK      | Yes                                                |
| Services   | Min Start Time  |         | 2023-07-14 16:31:19.000                            |
| Services   | Max Start Time  |         | 2023-07-26 11:23:17.324                            |
| Memory     | Memory          | OK      | Physical 31.09 GB, Swap 10.00 GB, Used 26.38       |
| CPU        | CPU             | OK      | Available 4, Used 1.04                             |
| Disk       | Data            | OK      | Size 89.0 GB, Used 59.3 GB, Free 33 %              |
| Disk       | Log             | OK      | Size 89.0 GB, Used 59.3 GB, Free 33 %              |
| Disk       | Trace           | OK      | Size 89.0 GB, Used 59.3 GB, Free 33 %              |
| Statistics | Alerts          | WARNING | cannot check statistics w/o SQL connection         |
```

6.1.4. SAP HANA の起動と停止

オプション 1: HDB コマンド

```
clusternode1:rh2adm> HDB help
Usage: /usr/sap/RH2/HDB02/HDB { start|stop|reconf|restart|version|info|proc|admin|kill|kill-<sig>|term
}
  kill or kill-9 should never be used in a productive environment!
```

- データベースを起動します。

  ```
  clusternode1:rh2adm> HDB start
  ```

- データベースを停止します。

  ```
  clusternode1:rh2adm> HDB stop
  ```

オプション 2 (推奨):sapcontrol を使用

```
clusternode1:rh2adm> sapcontrol -nr ${TINSTANCE} -function StartSystem HDB

03.07.2023 14:08:30
StartSystem
OK
```

```
clusternode1:rh2adm> sapcontrol -nr ${TINSTANCE} -function StopSystem HDB

03.07.2023 14:09:33
StopSystem
OK
```

GetProcessList を使用して、HANA サービスの起動と停止を監視します。

```
clusternode1:rh2adm> sapcontrol -nr ${TINSTANCE} -function GetProcessList
```

6.1.5. SAP HANA System Replication のステータスの確認

SAP HANA System Replication のステータスを確認するには、さまざまな方法があります。

- プライマリーノードの `clusternode1:rh2adm> python systemReplicationStatus.py `
- **clusternode1:rh2adm> echo $? #** (systemReplicationStatus の戻りコード)
- **clusternode1:rh2adm> hdbnsutil -sr_state**
- **clusternode1:rh2adm> hdbnsutil -sr_stateConfiguration**

モニターとして実行される **systemReplicationStatus.py** の出力例:

```
clusternode1:rh2adm> watch -n 5 "python
/usr/sap/${SAPSYSTEMNAME}/HDB{TINSTACE}/exe/python_support/systemReplicationStatus.py;ech
o \$?"
 concurrent-fencing: true
Every 5.0s: python systemReplicationStatus.py;echo $?
hana08: Fri Jul 28 17:01:05 2023

|Database |Host   |Port  |Service Name |Volume ID |Site ID |Site Name |Secondary |Secondary
|Secondary |Secondary |Secondary     |Replication |Replication |Replication    |
|         |  | |             |         |        |          |Host  |Port      |Site ID   |Site Name |Active Status |Mode
|Status    |Status Details |
|-------- |------ |----- |------------ |--------- |------- |--------- |--------- |--------- |--------- |-------------
|----------- |----------- |-------------- |
|SYSTEMDB |hana08 |30201 |nameserver   |       1 |  1 |DC2       |hana09    |    30201 |       3 |DC3
|YES          |SYNCMEM     |ACTIVE    |             |
|RH2   |hana08 |30207 |xsengine     |       2 |  1 |DC2       |hana09    |    30207 |       3 |DC3   |YES
|SYNCMEM     |ACTIVE    |             |
|RH2   |hana08 |30203 |indexserver  |       3 |  1 |DC2       |hana09    |    30203 |       3 |DC3   |YES
|SYNCMEM     |ACTIVE    |             |
|SYSTEMDB |hana08 |30201 |nameserver   |       1 |  1 |DC2       |remotehost3    |    30201 |       2
|DC1   |YES          |SYNCMEM     |ACTIVE    |             |
|RH2   |hana08 |30207 |xsengine     |       2 |  1 |DC2       |remotehost3    |    30207 |       2 |DC1
|YES          |SYNCMEM     |ACTIVE    |             |
|RH2   |hana08 |30203 |indexserver  |       3 |  1 |DC2       |remotehost3    |    30203 |       2 |DC1
|YES          |SYNCMEM     |ACTIVE    |             |

status system replication site "3": ACTIVE
status system replication site "2": ACTIVE
overall system replication status: ACTIVE

Local System Replication State
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

mode: PRIMARY
site id: 1
site name: DC2
15
```

予想される戻りコードの結果は次のとおりです。

- 10: NoHSR

- 11: Error
- 12: Unknown
- 13: Initializing
- 14: Syncing
- 15: Active

ほとんどの場合、システムレプリケーションのチェックは戻りコード **15** を返します。別の表示オプションとして、**-t** (printLandscapeTree) を使用する方法があります。

現在のプライマリーでの出力の例:

```
clusternode1:rh2adm> python systemReplicationStatus.py -t
HANA System Replication landscape:
  DC1  ( primary )
    | --- DC3  ( syncmem )
    | --- DC2  ( syncmem )
```

hdbnsutil -sr_state の例:

```
[root@clusternode1]# su - rh2adm
clusternode1:rh2adm> watch -n 10 hdbnsutil -sr_state
Every 10.0s: hdbnsutil -sr_state                                    clusternode1: Thu Jun
22 08:42:00 2023


System Replication State
~~~~~~~~~~~~~~~~~~~~~~~~

online: true

mode: syncmem
operation mode: logreplay
site id: 2
site name: DC1

is source system: false
is secondary/consumer system: true
has secondaries/consumers attached: false
is a takeover active: false
is primary suspended: false
is timetravel enabled: false
replay mode: auto
active primary site: 1

primary masters: clusternode2

Host Mappings:
~~~~~~~~~~~~~~

clusternode1 -> [DC3] remotehost3
clusternode1 -> [DC1] clusternode1
clusternode1 -> [DC2] clusternode2
```

```
Site Mappings:
~~~~~~~~~~~~~~
DC2 (primary/primary)
    |---DC3 (syncmem/logreplay)
    |---DC1 (syncmem/logreplay)

Tier of DC2: 1
Tier of DC3: 2
Tier of DC1: 2

Replication mode of DC2: primary
[2] 0:ssh*
```

プライマリーでの **sr_stateConfiguation** の例:

```
clusternode1:rh2adm> hdbnsutil -sr_stateConfiguration

System Replication State
~~~~~~~~~~~~~~~~~~~~~~~~

mode: primary
site id: 2
site name: DC1
done.
```

セカンダリーでの **sr_stateConfiguration** の例:

```
clusternode1:rh2adm> hdbnsutil -sr_stateConfiguration

System Replication State
~~~~~~~~~~~~~~~~~~~~~~~~

mode: syncmem
site id: 1
site name: DC2
active primary site: 2

primary masters: clusternode1
done.
```

どのノードが現在のプライマリーであるかをセカンダリーデータベースで確認することもできます。フェイルオーバー中に 2 つのプライマリーデータベースが見つかった場合に、この情報は、どちらのプライマリーデータベースが間違っており、セカンダリーとして再登録する必要があるかを判断するために必要になります。

詳細は、Example: Checking the Status on the Primary and Secondary Systems を参照してください。

6.1.6. セカンダリーノードの登録

SAP HANA System Replication 環境のセカンダリーデータベースを登録するための前提条件:

- SAP HANA バックアップの作成

- プライマリーノードでの SAP HANA System Replication の有効化
- データベースキーのコピー
- セカンダリーノードの登録

登録例:

```
clusternode1:rh2adm> hdbnsutil -sr_register --remoteHost=clusternode2 --
remoteInstance=${TINSTANCE} --replicationMode=syncmem --name=DC1 --online
--operationMode not set; using default from global.ini/[system_replication]/operation_mode: logreplay
adding site ...
collecting information ...
updating local ini files ...
done.
```

登録すると、**global.ini** ファイルが自動的に更新されます。

更新前:

```
# global.ini last modified 2023-06-15 09:55:05.665341 by /usr/sap/RH2/HDB02/exe/hdbnsutil -
initTopology --workergroup=default --set_user_system_pw
[multidb]
mode = multidb
database_isolation = low
singletenant = yes

[persistence]
basepath_datavolumes = /hana/data/RH2
basepath_logvolumes = /hana/log/RH2
```

更新後:

```
# global.ini last modified 2023-06-15 11:25:44.516946 by hdbnsutil -sr_register --remoteHost=node2
--remoteInstance=02 --replicationMode=syncmem --name=DC1 --online
[multidb]
mode = multidb
database_isolation = low
singletenant = yes

[persistence]
basepath_datavolumes = /hana/data/RH2
basepath_logvolumes = /hana/log/RH2

[system_replication]
timetravel_logreplay_mode = auto
site_id = 3
mode = syncmem
actual_mode = syncmem
site_name = DC1
operation_mode = logreplay

[system_replication_site_masters]
1 = clusternode2:30201
```

6.1.7. sapcontrol GetProcessList

アクティブな SAP HANA データベースのプロセスの確認

```
clusternode1:rh2adm> sapcontrol -nr ${TINSTANCE} -function GetProcessList
clusternode1: Wed Jun  7 08:23:03 2023

07.06.2023 08:23:03
GetProcessList
OK
name, description, dispstatus, textstatus, starttime, elapsedtime, pid
hdbdaemon, HDB Daemon, GREEN, Running, 2023 06 02 16:59:42, 111:23:21, 4245
hdbcompileserver, HDB Compileserver, GREEN, Running, 2023 06 02 17:01:35, 111:21:28, 7888
hdbindexserver, HDB Indexserver-RH2, GREEN, Running, 2023 06 02 17:01:36, 111:21:27, 7941
hdbnameserver, HDB Nameserver, GREEN, Running, 2023 06 02 17:01:29, 111:21:34, 7594
hdbpreprocessor, HDB Preprocessor, GREEN, Running, 2023 06 02 17:01:35, 111:21:28, 7891
hdbwebdispatcher, HDB Web Dispatcher, GREEN, Running, 2023 06 02 17:01:42, 111:21:21, 8339
hdbxsengine, HDB XSEngine-RH2, GREEN, Running, 2023 06 02 17:01:36, 111:21:27, 7944
```

6.1.8. sapcontrol GetInstanceList

これにより、SAP HANA データベースのインスタンスのステータスがリスト表示されます。ポートも表示されます。3 つの異なるステータス名があります。

- GREEN (実行中)
- GRAY (停止)
- YELLOW (ステータスが現在変化中)

アクティブなインスタンスの例:

```
clusternode1:rh2adm> sapcontrol -nr ${TINSTANCE} -function GetSystemInstanceList
clusternode1: Wed Jun  7 08:24:13 2023


07.06.2023 08:24:13
GetSystemInstanceList
OK
hostname, instanceNr, httpPort, httpsPort, startPriority, features, dispstatus
remotehost3, 2, 50213, 50214, 0.3, HDB|HDB_WORKER, GREEN
```

停止したインスタンスの例:

```
clusternode1:rh2adm> sapcontrol -nr ${TINSTANCE} -function GetSystemInstanceList

22.06.2023 09:14:55
GetSystemInstanceList
OK
hostname, instanceNr, httpPort, httpsPort, startPriority, features, dispstatus
remotehost3, 2, 50213, 50214, 0.3, HDB|HDB_WORKER, GRAY
```

6.1.9. hdbcons の例

HDB コンソールを使用してデータベースに関する情報を表示することもできます。

- **hdbcons -e hdbindexserver 'replication info'**
- その他のオプションについては、**hdbcons -e hdbindexserver help** を参照

'replication info' の例:

```
clusternode1:rh2adm> hdbcons -e hdbindexserver 'replication info'
hdbcons -p `pgrep hdbindex` 'replication info'
SAP HANA DB Management Client Console (type '\?' to get help for client commands)
Try to open connection to server process with PID 451925
SAP HANA DB Management Server Console (type 'help' to get help for server commands)
Executable: hdbindexserver (PID: 451925)
[OK]
--
## Start command at: 2023-06-22 09:05:25.211
listing default statistics for volume 3
System Replication Primary Information
=====================================
System Replication Primary Configuration
 [system_replication] logshipping_timeout                = 30
 [system_replication] enable_full_sync                   = false
 [system_replication] preload_column_tables              = true
 [system_replication] ensure_backup_history              = true
 [system_replication_communication] enable_ssl           = off
 [system_replication] keep_old_style_alert               = false
 [system_replication] enable_log_retention               = auto
 [system_replication] logshipping_max_retention_size     = 1048576
 [system_replication] logshipping_async_buffer_size      = 268435456
 - lastLogPos                : 0x4ab2700
 - lastLogPosTimestamp       : 22.06.2023-07.05.25 (1687417525193952)
 - lastConfirmedLogPos       : 0x4ab2700
 - lastConfirmedLogPosTimestamp: 22.06.2023-07.05.25 (1687417525193952)
 - lastSavepointVersion      : 1286
 - lastSavepointLogPos       : 0x4ab0602
 - lastSavepointTimestamp    : 22.06.2023-07.02.42 (1687417362853007)
 2 session registered.
Session index 0
 - SiteID            : 3
 - RemoteHost        : 192.168.5.137
Log Connection
 - ptr               : 0x00007ff04c0a1000
 - channel           : {<NetworkChannelSSLFilter>={<NetworkChannelBase>={this=140671686293528,
fd=70, refCnt=2, idx=5, local=192.168.5.134/40203_tcp, remote=192.168.5.137/40406_tcp,
state=Connected, pending=[r---]}}}
 - SSLActive         : false
 - mode              : syncmem
Data Connection
 - ptr               : 0x00007ff08b730000
 - channel           : {<NetworkChannelSSLFilter>={<NetworkChannelBase>={this=140671436247064,
fd=68, refCnt=2, idx=6, local=192.168.5.134/40203_tcp, remote=192.168.5.137/40408_tcp,
state=Connected, pending=[r---]}}}
 - SSLActive         : false
Primary Statistics
 - Creation Timestamp            : 20.06.2023-13.55.07 (1687269307772532)
```

```
- Last Reset Timestamp            : 20.06.2023-13.55.07 (1687269307772532)
- Statistic Reset Count           : 0
- ReplicationMode                 : syncmem
- OperationMode                   : logreplay
- ReplicationStatus               : ReplicationStatus_Active
- ReplicationStatusDetails        :
- ReplicationFullSync             : DISABLED
- shippedLogPos                   : 0x4ab2700
- shippedLogPosTimestamp          : 22.06.2023-07.05.25 (1687417525193952)
- sentLogPos                      : 0x4ab2700
- sentLogPosTimestamp             : 22.06.2023-07.05.25 (1687417525193952)
- sentMaxLogWriteEndPosition      : 0x4ab2700
- sentMaxLogWriteEndPositionReqCnt: 0x1f6b8
- shippedLogBuffersCount          : 142439
- shippedLogBuffersSize           : 805855232 bytes
- shippedLogBuffersSizeUsed       : 449305792 bytes (55.76clusternode1:rh2adm>)
- shippedLogBuffersSizeNet        : 449013696 bytes (55.72clusternode1:rh2adm>)
- shippedLogBufferDuration        : 83898615 microseconds
- shippedLogBufferDurationMin     : 152 microseconds
- shippedLogBufferDurationMax     : 18879 microseconds
- shippedLogBufferDurationSend    : 7301067 microseconds
- shippedLogBufferDurationComp    : 0 microseconds
- shippedLogBufferThroughput      : 9709099.18 bytes/s
- shippedLogBufferPendingDuration : 80583785 microseconds
- shippedLogBufferRealThrougput   : 10073190.40 bytes/s
- replayLogPos                    : 0x4ab2700
- replayLogPosTimestamp           : 22.06.2023-07.05.25 (1687417525193952)
- replayBacklog                   : 0 microseconds
- replayBacklogSize               : 0 bytes
- replayBacklogMax                : 822130896 microseconds
- replayBacklogSizeMax            : 49455104 bytes
- shippedSavepointVersion         : 0
- shippedSavepointLogPos          : 0x0
- shippedSavepointTimestamp       : not set
- shippedFullBackupCount          : 0
- shippedFullBackupSize           : 0 bytes
- shippedFullBackupSizeNet        : 0 bytes (-nanclusternode1:rh2adm>)
- shippedFullBackupDuration       : 0 microseconds
- shippedFullBackupDurationComp   : 0 microseconds
- shippedFullBackupThroughput     : 0.00 bytes/s
- shippedFullBackupStreamCount    : 0
- shippedFullBackupResumeCount    : 0
- shippedLastFullBackupSize       : 0 bytes
- shippedLastFullBackupSizeNet    : 0 bytes (-nanclusternode1:rh2adm>)
- shippedLastFullBackupStart      : not set
- shippedLastFullBackupEnd        : not set
- shippedLastFullBackupDuration   : 0 microseconds
- shippedLastFullBackupStreamCount : 0
- shippedLastFullBackupResumeCount : 0
- shippedDeltaBackupCount         : 0
- shippedDeltaBackupSize          : 0 bytes
- shippedDeltaBackupSizeNet       : 0 bytes (-nanclusternode1:rh2adm>)
- shippedDeltaBackupDuration      : 0 microseconds
- shippedDeltaBackupDurationComp  : 0 microseconds
- shippedDeltaBackupThroughput    : 0.00 bytes/s
- shippedDeltaBackupStreamCount   : 0
```

```
 - shippedDeltaBackupResumeCount   : 0
 - shippedLastDeltaBackupSize      : 0 bytes
 - shippedLastDeltaBackupSizeNet   : 0 bytes (-nanclusternode1:rh2adm>)
 - shippedLastDeltaBackupStart     : not set
 - shippedLastDeltaBackupEnd       : not set
 - shippedLastDeltaBackupDuration  : 0 microseconds
 - shippedLastDeltaBackupStreamCount : 0
 - shippedLastDeltaBackupResumeCount : 0
 - currentTransferType             : None
 - currentTransferSize             : 0 bytes
 - currentTransferPosition         : 0 bytes (0clusternode1:rh2adm>)
 - currentTransferStartTime        : not set
 - currentTransferThroughput       : 0.00 MB/s
 - currentTransferStreamCount      : 0
 - currentTransferResumeCount      : 0
 - currentTransferResumeStartTime  : not set
 - Secondary sync'ed via Log Count : 1
 - syncLogCount                    : 3
 - syncLogSize                     : 62840832 bytes
 - backupHistoryComplete           : 1
 - backupLogPosition               : 0x4a99980
 - backupLogPositionUpdTimestamp   : 22.06.2023-06.56.27 (0x5feb26227e7af)
 - shippedMissingLogCount          : 0
 - shippedMissingLogSize           : 0 bytes
 - backlogSize                     : 0 bytes
 - backlogTime                     : 0 microseconds
 - backlogSizeMax                  : 0 bytes
 - backlogTimeMax                  : 0 microseconds
 - Secondary Log Connect time      : 20.06.2023-13.55.31 (1687269331361049)
 - Secondary Data Connect time     : 20.06.2023-13.55.33 (1687269333768341)
 - Secondary Log Close time        : not set
 - Secondary Data Close time       : 20.06.2023-13.55.31 (1687269331290050)
 - Secondary Log Reconnect Count   : 0
 - Secondary Log Failover Count    : 0
 - Secondary Data Reconnect Count  : 1
 - Secondary Data Failover Count   : 0
----------------------------------------------------------------------
Session index 1
 - SiteID              : 2
 - RemoteHost          : 192.168.5.133
Log Connection
 - ptr                 : 0x00007ff0963e4000
 - channel             : {<NetworkChannelSSLFilter>={<NetworkChannelBase>={this=140671506282520,
fd=74, refCnt=2, idx=0, local=192.168.5.134/40203_tcp, remote=192.168.5.133/40404_tcp,
state=Connected, pending=[r---]}}}
 - SSLActive           : false
 - mode                : syncmem
Data Connection
 - ptr                 : 0x00007ff072c04000
 - channel             : {<NetworkChannelSSLFilter>={<NetworkChannelBase>={this=140671463146520,
fd=75, refCnt=2, idx=1, local=192.168.5.134/40203_tcp, remote=192.168.5.133/40406_tcp,
state=Connected, pending=[r---]}}}
 - SSLActive           : false
Primary Statistics
 - Creation Timestamp              : 20.06.2023-13.55.49 (1687269349892111)
 - Last Reset Timestamp            : 20.06.2023-13.55.49 (1687269349892111)
```

```
- Statistic Reset Count          : 0
- ReplicationMode                : syncmem
- OperationMode                  : logreplay
- ReplicationStatus              : ReplicationStatus_Active
- ReplicationStatusDetails       :
- ReplicationFullSync            : DISABLED
- shippedLogPos                  : 0x4ab2700
- shippedLogPosTimestamp         : 22.06.2023-07.05.25 (1687417525193952)
- sentLogPos                     : 0x4ab2700
- sentLogPosTimestamp            : 22.06.2023-07.05.25 (1687417525193952)
- sentMaxLogWriteEndPosition     : 0x4ab2700
- sentMaxLogWriteEndPositionReqCnt: 0x1f377
- shippedLogBuffersCount         : 142326
- shippedLogBuffersSize          : 793939968 bytes
- shippedLogBuffersSizeUsed      : 437675200 bytes (55.13clusternode1:rh2adm>)
- shippedLogBuffersSizeNet       : 437565760 bytes (55.11clusternode1:rh2adm>)
- shippedLogBufferDuration       : 76954026 microseconds
- shippedLogBufferDurationMin    : 115 microseconds
- shippedLogBufferDurationMax    : 19285 microseconds
- shippedLogBufferDurationSend   : 2951495 microseconds
- shippedLogBufferDurationComp   : 0 microseconds
- shippedLogBufferThroughput     : 10446578.53 bytes/s
- shippedLogBufferPendingDuration : 73848247 microseconds
- shippedLogBufferRealThrougput  : 10875889.97 bytes/s
- replayLogPos                   : 0x4ab2700
- replayLogPosTimestamp          : 22.06.2023-07.05.25 (1687417525193952)
- replayBacklog                  : 0 microseconds
- replayBacklogSize              : 0 bytes
- replayBacklogMax               : 113119944 microseconds
- replayBacklogSizeMax           : 30171136 bytes
- shippedSavepointVersion        : 0
- shippedSavepointLogPos         : 0x0
- shippedSavepointTimestamp      : not set
- shippedFullBackupCount         : 0
- shippedFullBackupSize          : 0 bytes
- shippedFullBackupSizeNet       : 0 bytes (-nanclusternode1:rh2adm>)
- shippedFullBackupDuration      : 0 microseconds
- shippedFullBackupDurationComp  : 0 microseconds
- shippedFullBackupThroughput    : 0.00 bytes/s
- shippedFullBackupStreamCount   : 0
- shippedFullBackupResumeCount   : 0
- shippedLastFullBackupSize      : 0 bytes
- shippedLastFullBackupSizeNet   : 0 bytes (-nanclusternode1:rh2adm>)
- shippedLastFullBackupStart     : not set
- shippedLastFullBackupEnd       : not set
- shippedLastFullBackupDuration  : 0 microseconds
- shippedLastFullBackupStreamCount : 0
- shippedLastFullBackupResumeCount : 0
- shippedDeltaBackupCount        : 0
- shippedDeltaBackupSize         : 0 bytes
- shippedDeltaBackupSizeNet      : 0 bytes (-nanclusternode1:rh2adm>)
- shippedDeltaBackupDuration     : 0 microseconds
- shippedDeltaBackupDurationComp : 0 microseconds
- shippedDeltaBackupThroughput   : 0.00 bytes/s
- shippedDeltaBackupStreamCount  : 0
- shippedDeltaBackupResumeCount  : 0
```

```
 - shippedLastDeltaBackupSize      : 0 bytes
 - shippedLastDeltaBackupSizeNet   : 0 bytes (-nanclusternode1:rh2adm>)
 - shippedLastDeltaBackupStart     : not set
 - shippedLastDeltaBackupEnd       : not set
 - shippedLastDeltaBackupDuration  : 0 microseconds
 - shippedLastDeltaBackupStreamCount : 0
 - shippedLastDeltaBackupResumeCount : 0
 - currentTransferType             : None
 - currentTransferSize             : 0 bytes
 - currentTransferPosition         : 0 bytes (0clusternode1:rh2adm>)
 - currentTransferStartTime        : not set
 - currentTransferThroughput       : 0.00 MB/s
 - currentTransferStreamCount      : 0
 - currentTransferResumeCount      : 0
 - currentTransferResumeStartTime  : not set
 - Secondary sync'ed via Log Count : 1
 - syncLogCount                    : 3
 - syncLogSize                     : 61341696 bytes
 - backupHistoryComplete           : 1
 - backupLogPosition               : 0x4a99980
 - backupLogPositionUpdTimestamp   : 22.06.2023-06.56.27 (0x5feb26227e670)
 - shippedMissingLogCount          : 0
 - shippedMissingLogSize           : 0 bytes
 - backlogSize                     : 0 bytes
 - backlogTime                     : 0 microseconds
 - backlogSizeMax                  : 0 bytes
 - backlogTimeMax                  : 0 microseconds
 - Secondary Log Connect time      : 20.06.2023-13.56.21 (1687269381053599)
 - Secondary Data Connect time     : 20.06.2023-13.56.27 (1687269387399610)
 - Secondary Log Close time        : not set
 - Secondary Data Close time       : 20.06.2023-13.56.21 (1687269381017244)
 - Secondary Log Reconnect Count   : 0
 - Secondary Log Failover Count    : 0
 - Secondary Data Reconnect Count  : 1
 - Secondary Data Failover Count   : 0
----------------------------------------------------------------
[OK]
## Finish command at: 2023-06-22 09:05:25.212 command took: 572.000 usec
--
[EXIT]
--
[BYE]
```

help の例:

```
clusternode1:rh2adm> hdbcons  -e hdbindexserver help
SAP HANA DB Management Client Console (type '\?' to get help for client commands)
Try to open connection to server process with PID 451925
SAP HANA DB Management Server Console (type 'help' to get help for server commands)
Executable: hdbindexserver (PID: 451925)
[OK]
--
## Start command at: 2023-06-22 09:07:16.784
Synopsis:
help [<command name>]: Print command help
   - <command name> - Command name for which to display help
```

```
Available commands:
ae_tableload - Handle loading of column store tables and columns
all - Print help and other info for all hdbcons commands
authentication - Authentication management.
binarysemaphore - BinarySemaphore management
bye - Exit console client
cd - ContainerDirectory management
cfgreg - Basis Configurator
checktopic - CheckTopic management
cnd - ContainerNameDirectory management
conditionalvariable - ConditionalVariable management
connection - Connection management
context - Execution context management (i.e., threads)
converter - Converter management
cpuresctrl - Manage cpu resources such as last-level cache allocation
crash - Crash management
crypto - Cryptography management (SSL/SAML/X509/Encryption).
csaccessor - Display diagnostics related to the CSAccessor library
ddlcontextstore - Get DdlContextStore information
deadlockdetector - Deadlock detector.
debug - Debug management
distribute - Handling distributed systems
dvol - DataVolume management
ELF - ELF symbol resolution management
encryption - Persistence encryption management
eslog - Manipulate logger on extended storage
event - Event management
exit - Exit console client
flightrecorder - Flight Recorder
hananet - HANA-Net command interface
help - Display help for a command or command list
hkt - HANA Kernal Tracer (HKT) management
indexmanager - Get IndexManager information, especially for IndexHandles
itab - Internaltable diagnostics
jexec - Information and actions for Job Executor/Scheduler
licensing - Licensing management.
log - Show information about logger and manipulate logger
machine - Information about the machine topology
mm - Memory management
monitor - Monitor view command
mproxy - Malloc proxy management
msl - Mid size LOB management
mutex - Mutex management
numa - Provides NUMA statistics for all columns of a given table, broken down by column
constituents like dictionary, data vector and index.
nvmprovider - NVM Provider
output - Command for managing output from the hdbcons
page - Page management
pageaccess - PageAccess management
profiler - Profiler
quit - Exit console client
readwritelock - ReadWriteLock management
replication - Monitor data and log replication
resman - ResourceManager management
rowstore - Row Store
runtimedump - Generate a runtime dump.
```

```
savepoint - Savepoint management
semaphore - Semaphore management
servicethreads - Thread information M_SERVICE_THREADS
snapshot - Snapshot management
stat - Statistics management
statisticsservercontroller - StatisticsServer internals
statreg - Statistics registry command
syncprimi - Syncprimitive management (Mutex, CondVariable, Semaphore, BinarySemaphore,
ReadWriteLock)
table - Table Management
tablepreload - Manage and monitor table preload
trace - Trace management
tracetopic - TraceTopic management
transaction - Transaction management
ut - UnifiedTable Management
version - Version management
vf - VirtualFile management
x2 - get X2 info
[OK]
## Finish command at: 2023-06-22 09:07:16.785 command took: 209.000 usec
--
[EXIT]
--
[BYE]
```

6.1.10. SAP HANA バックアップの作成

SAP HANA System Replication を使用する場合は、先にプライマリーシステムでバックアップを作成する必要があります。

ユーザー **<sid>adm** としてこれを実行する方法の例:

```
clusternode1:rh2adm> hdbsql -i ${TINSTANCE} -u system -d SYSTEMDB "BACKUP DATA USING
FILE ('/hana/backup/')"
clusternode1:rh2adm> hdbsql -i ${TINSTANCE} -u system -d ${SAPSYSTEMNAME} "BACKUP
DATA USING FILE ('/hana/backup/')"
```

6.1.11. プライマリーノードでの SAP HANA System Replication の有効化

SAP HANA System Replication はプライマリーノードで有効にする必要があります。これを行うには、先にバックアップを実行する必要があります。

```
clusternode1:rh2adm> hdbnsutil -sr_enable --name=DC1
nameserver is active, proceeding ...
successfully enabled system as system replication source site
done.
```

6.1.12. セカンダリーノードへのデータベースキーのコピー

セカンダリーデータベースをセカンダリーとして登録するには、データベースキーをプライマリーデータベースからセカンダリーデータベースにコピーする必要があります。

以下に例を示します。

-

```
clusternode1:rh2adm> scp -rp
/usr/sap/${SAPSYSTEMNAME}/SYS/global/security/rsecssfs/data/SSFS_${SAPSYSTEMNAME}.DAT
remotehost3:/usr/sap/${SAPSYSTEMNAME}/SYS/global/security/rsecssfs/data/SSFS_${SAPSYSTEMN
AME}.DAT
clusternode1:rh2adm> scp -rp
/usr/sap/${SAPSYSTEMNAME}/SYS/global/security/rsecssfs/key/SSFS_${SAPSYSTEMNAME}.KEY
remotehost3:/usr/sap/${SAPSYSTEMNAME}/SYS/global/security/rsecssfs/key/SSFS_${SAPSYSTEMN
AME}.KEY
```

6.1.13. SAP HANA System Replication のセカンダリーノードの登録

まず、データベースキーがセカンダリーノードにコピーされていることを確認してください。次に、登録コマンドを実行します。

```
clusternode1:rh2adm> hdbnsutil -sr_register --remoteHost=remotehost3 --
remoteInstance=${TINSTANCE} --replicationMode=syncmem --name=DC1 --remoteName=DC3 --
operationMode=logreplay --online
```

パラメーターの説明:

- **remoteHost**: ソース (プライマリー) データベースを実行しているアクティブノードのホスト名
- **remoteInstance**: データベースのインスタンス番号
- **replicationMode**: 次のオプションのいずれか
  - **sync**: ハードディスク同期
  - **async**: 非同期レプリケーション
  - **syncmem**: メモリー同期
- **name**: このレプリケーションサイトのエイリアス
- **remoteName**: ソースデータベースのエイリアス名
- **operationMode**: 次のオプションのいずれか
  - **delta_datashipping**: データが定期的に送信されます。テイクオーバーに少し時間がかかります。
  - **logreplay**: ログがリモートサイトですぐに再実行されます。テイクオーバーが早くなります。
  - **logreplay_readaccess**: 2 番目のサイトへの追加の logreplay 読み取り専用アクセスが可能です。

6.1.14. SAP HANA データベースの log_mode の確認

log_mode の設定には 2 つのオプションがあります。

- **log_mode=overwrite**
- **log_mode=normal**: これはデフォルト値であり、データベースインスタンスがプライマリーとして実行されている場合にも必要です。SAP HANA Multitarget System Replication を使用する場合は、**log_mode=normal** を使用する必要があります。**log_mode** を確認する最良の方法

は、**hdbsql** を使用することです。

間違った **overwrite** エントリーを含む例:

```
clusternode1:rh2adm> hdbsql -i ${TINSTANCE} -d ${SAPSYSTEMNAME} -u system
Password:

Welcome to the SAP HANA Database interactive terminal.

Type:  \h for help with commands
       \q to quit

hdbsql RH2=> select * from m_inifile_contents where key='log_mode'
FILE_NAME,LAYER_NAME,TENANT_NAME,HOST,SECTION,KEY,VALUE
"global.ini","DEFAULT","","","persistence","log_mode","normal"
"global.ini","HOST","","node2","persistence","log_mode","overwrite"
2 rows selected (overall time 46.931 msec; server time 30.845 msec)

hdbsql RH2=>exit
```

この場合、次の 2 つの **global.ini** ファイルがあります。

- **DEFAULT**
  - **/usr/sap/${SAPSYSTEMNAME}/SYS/global/hdb/custom/config/global.ini**
- **HOST**
  - **/hana/shared/${SAPSYSTEMNAME}/HDB${TINSTANCE}/${HOSTNAME}/global.ini**
    HOST 値は **DEFAULT** 値を上書きします。データベースを起動する前に両方のファイルを確認してから、**hdbsql** を再度使用して正しい設定を確認することもできます。**log_mode** は、**global.ini** ファイルを編集することで変更できます。

以下に例を示します。

```
clusternode1:rh2adm> vim
/hana/shared/${SAPSYSTEMNAME}/HDB${TINSTANCE}/${HOSTNAME}/global.ini
# global.ini last modified 2023-04-06 16:15:03.521715 by hdbnameserver
[persistence]
log_mode = overwrite
```

```
# global.ini last modified 2023-04-06 16:15:03.521715 by hdbnameserver
[persistence]
log_mode = normal
```

global.ini ファイルを確認または更新した後、**log_mode** 値を確認します。

```
clusternode1:rh2adm> hdbsql -d ${SAPSYSTEMNAME} -i ${TINSTANCE} -u SYSTEM;
hdbsql RH2=> select * from m_inifile_contents where section='persistence' and key='log_mode'
FILE_NAME,LAYER_NAME,TENANT_NAME,HOST,SECTION,KEY,VALUE
"global.ini","DEFAULT","","","persistence","log_mode","normal"
"global.ini","HOST","","node2","persistence","log_mode","normal"
2 rows selected (overall time 60.982 msec; server time 20.420 msec)
```

また、このセクションからわかるように、このパラメーターは **[persistence]** セクションで設定する必要があります。ログモードを **overwrite** から **normal** に変更する場合は、データベースを確実に回復できるように、完全なデータバックアップを作成することを推奨します。

6.1.15. プライマリーデータベースの検出

プライマリーノードを識別するには、たとえば次の方法があります。

- **pcs status | grep Promoted**
- **hdbnsutil -sr_stateConfiguration**
- **systemReplicationStatus.py**

オプション 1 - 次の **systemReplicationStatus.py** スクリプトとフィルターの例は、すべてのノード上のプライマリーデータベースの場所を返します。

```
clusternode1:rh2adm>
/usr/sap/${SAPSYSTEMNAME}/HDB${TINSTANCE}/exe/Python/bin/python
/usr/sap/${SAPSYSTEMNAME}/HDB${TINSTANCE}/exe/python_support/systemReplicationStatus.py
--sapcontrol=1 | egrep -e
"3${TINSTANCE}01/HOST|PRIMARY_MASTERS"| head -1 | awk -F"=" '{ print $2 }'
```

出力:

```
clusternode2
```

オプション 2 - 次の例では、すべてのノードに対して同様の方法で **systemReplicationStatus** を表示します。

```
rh2adm>hdbnsutil -sr_state --sapcontrol=1 | grep site.*Mode
```

出力:

```
siteReplicationMode/DC1=primary
siteReplicationMode/DC3=async
siteReplicationMode/DC2=syncmem
siteOperationMode/DC1=primary
siteOperationMode/DC3=logreplay
siteOperationMode/DC2=logreplay
```

6.1.16. プライマリーのテイクオーバー

プライマリーノードとセカンダリーノードの確認については、レプリケーションステータスの確認 セクションを参照してください。また、以下の点にも注意してください。

- クラスターを **maintenance-mode** にします。
- セカンダリーノードでテイクオーバーを開始します。

クラスターの **maintenance-mode** を有効にする例:

```
[root@clusternode1]# pcs property set maintenance-mode=true
```

新しいプライマリーとなるセカンダリーで、**<sidadm>** ユーザーとして次のコマンドを実行します。

```
clusternode1:rh2adm> hdbnsutil -sr_takeover
```

このセカンダリーがプライマリーになり、他のアクティブなセカンダリーデータベースが、新しいプライマリーに再登録されます。前のプライマリーは、セカンダリーとして手動で再登録する必要があります。

6.1.17. 以前のプライマリーをセカンダリーとして再登録する

クラスターが停止しているか、**maintenance-mode** になっていることを確認してください。以下に例を示します。

```
clusternode2:rh2adm> hdbnsutil -sr_register --remoteHost=remotehost3 --
remoteInstance=${TINSTANCE} --replicationMode=syncmem --name=DC2 --online --
remoteName=DC3 --operationMode=logreplay --force_full_replica --online
```

この例では、完全なレプリケーションを使用しています。SAP HANA システム管理者は、完全なレプリケーションが必要になる場合を理解しておく必要があります。

6.1.18. フェイルオーバーからの回復

SAP HANA System Replication のステータスの確認 および プライマリーノードの検出 を参照してください。重要なのは、情報の整合性を確保することです。ノードが **systemReplicationStatus.py** の出力に含まれておらず、ノードのシステムレプリケーションの状態が異なる場合は、このノードを再登録する必要があるかどうかをデータベース管理者に確認してください。

これを解決する 1 つの方法は、このサイトを新しいセカンダリーとして 再登録 することです。

場合によっては、セカンダリーインスタンスがまだ起動していないことがあります。その後、もう一度再登録する前に、このサイトの登録を解除してください。セカンダリー DC1 の登録を解除する例:

```
clusternode1:rh2adm> hdbnsutil -sr_unregister --name=DC1
```

DC1 の再登録例:

```
clusternode1:rh2adm> hdbnsutil -sr_register --name=DC1 --remoteHost=node2 --remoteInstance=02
--replicationMode=sync --operationMode=logreplay --online
```

データベースを 起動 し、実行中 かどうかを確認する必要があります。最後に レプリケーションステータスを確認 します。

6.2. PACEMAKER コマンド

6.2.1. クラスターの起動と停止

すべてのノードでクラスターを起動するには、次のコマンドを実行します。

```
# pcs cluster start -all
```

再起動後、サービスが有効になっている場合にのみ、クラスターが自動的に起動します。このコマンドは、クラスターが起動したかどうか、およびデーモンの自動起動が有効になっているかどうかを確認するのに役立ちます。

```
# pcs cluster status
```

クラスターの自動起動は次の方法で有効にできます。

```
# pcs cluster enable --all
```

その他のオプションは以下のとおりです。

- クラスターを停止する
- ノードをスタンバイ状態にする
- クラスターを **maintenance-mode** にする

詳細は、**pcs cluster** のヘルプを参照してください。

```
# pcs cluster stop --all
# pcs cluster help
```

6.2.2. クラスターを **maintenance-mode** にする

変更を加えるときに Pacemaker クラスターによる干渉を避けるには、クラスターを **maintenance-mode** にすることで、クラスターを "フリーズ" できます。

```
# pcs property set maintenance-mode=true
```

maintenance-mode を確認する簡単な方法は、リソースが unmanaged かどうかを確認することです。

```
# pcs resource
  * Clone Set: SAPHanaTopology_RH2_02-clone [SAPHanaTopology_RH2_02] (unmanaged):
    * SAPHanaTopology_RH2_02    (ocf:heartbeat:SAPHanaTopology):        Started clusternode1
(unmanaged)
    * SAPHanaTopology_RH2_02    (ocf:heartbeat:SAPHanaTopology):        Started clusternode2
(unmanaged)
  * Clone Set: SAPHana_RH2_02-clone [SAPHana_RH2_02] (promotable, unmanaged):
    * SAPHana_RH2_02    (ocf:heartbeat:SAPHana):        Unpromoted clusternode1 (unmanaged)
    * SAPHana_RH2_02    (ocf:heartbeat:SAPHana):        Promoted clusternode2 (unmanaged)
  * vip_RH2_02_MASTER   (ocf:heartbeat:IPaddr2):        Started clusternode2 (unmanaged)
```

maintenance-mode のクラスターは、リソースステータスの変更を更新しません。その間に、クラスターリソースを更新してリソース状態を検出します。

```
# pcs resource refresh
```

これにより、何か問題がある場合にその問題が示され、**maintenance-mode** が終了するとすぐに、クラスターによる修復アクションが実行されます。

次のコマンドを実行して **maintenance-mode** を解除します。

```
# pcs property set maintenance-mode=false
```

これで、クラスターが引き続き動作します。設定に問題があった場合、クラスターはその問題にすぐに対処します。

6.2.3. クラスターのステータスの確認

以下に、クラスターのステータスを確認するいくつかの方法を示します。

- クラスターが実行中かどうかを確認します。

```
# pcs cluster status
```

- クラスターとすべてのリソースを確認します。

```
# pcs status
```

- クラスター、すべてのリソース、およびすべてのノード属性を確認します。

```
# pcs status --full
```

- リソースのみを確認します。

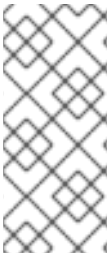
```
# pcs resource status --full
```

- **Stonith** の履歴を確認します。

```
# pcs stonith history
```

- 場所の制約を確認します。

```
# pcs constraint location
```



注記

フェンシングを設定してテストする必要があります。可能な限り自動化されたソリューションを実現するには、クラスターを常にアクティブにする必要があります。そうすることで、再起動後にクラスターが自動的に起動できるようになります。実稼働環境では、再起動を無効にすると、クラッシュ後などに手動で介入できるようになります。デーモンのステータスも確認してください。

以下に例を示します。

```
# pcs status --full
Cluster name: cluster1
Status of pacemakerd: 'Pacemaker is running' (last updated 2023-06-22 17:56:01 +02:00)
Cluster Summary:
  * Stack: corosync
  * Current DC: clusternode2 (2) (version 2.1.5-7.el9-a3f44794f94) - partition with quorum
  * Last updated: Thu Jun 22 17:56:01 2023
  * Last change:  Thu Jun 22 17:53:34 2023 by root via crm_attribute on clusternode1
  * 2 nodes configured
```

```
  * 6 resource instances configured
Node List:
  * Node clusternode1 (1): online, feature set 3.16.2
  * Node clusternode2 (2): online, feature set 3.16.2
Full List of Resources:
  * h7fence (stonith:fence_rhevm):  Started clusternode2
  * Clone Set: SAPHanaTopology_RH2_02-clone [SAPHanaTopology_RH2_02]:
    * SAPHanaTopology_RH2_02 (ocf:heartbeat:SAPHanaTopology):  Started clusternode1
    * SAPHanaTopology_RH2_02 (ocf:heartbeat:SAPHanaTopology):  Started clusternode2
  * Clone Set: SAPHana_RH2_02-clone [SAPHana_RH2_02] (promotable):
    * SAPHana_RH2_02 (ocf:heartbeat:SAPHana):  Promoted clusternode1
    * SAPHana_RH2_02 (ocf:heartbeat:SAPHana):  Unpromoted clusternode2
  * vip_RH2_02_MASTER (ocf:heartbeat:IPaddr2):  Started clusternode1
Node Attributes:
  * Node: clusternode1 (1):
    * hana_rh2_clone_state            : PROMOTED
    * hana_rh2_op_mode                : logreplay
    * hana_rh2_remoteHost             : clusternode2
    * hana_rh2_roles                  : 4:P:master1:master:worker:master
    * hana_rh2_site                   : DC1
    * hana_rh2_sra                    : -
    * hana_rh2_srah                   : -
    * hana_rh2_srmode                 : syncmem
    * hana_rh2_sync_state             : PRIM
    * hana_rh2_version                : 2.00.059.02
    * hana_rh2_vhost                  : clusternode1
    * lpa_rh2_lpt                     : 1687449214
    * master-SAPHana_RH2_02           : 150
  * Node: clusternode2 (2):
    * hana_rh2_clone_state            : DEMOTED
    * hana_rh2_op_mode                : logreplay
    * hana_rh2_remoteHost             : clusternode1
    * hana_rh2_roles                  : 4:S:master1:master:worker:master
    * hana_rh2_site                   : DC2
    * hana_rh2_sra                    : -
    * hana_rh2_srah                   : -
    * hana_rh2_srmode                 : syncmem
    * hana_rh2_sync_state             : SOK
    * hana_rh2_version                : 2.00.059.02
    * hana_rh2_vhost                  : clusternode2
    * lpa_rh2_lpt                     : 30
    * master-SAPHana_RH2_02           : 100
Migration Summary:
Tickets:
PCSD Status:
  clusternode1: Online
  clusternode2: Online
Daemon Status:
  corosync: active/enabled
  pacemaker: active/enabled
  pcsd: active/enabled
```

6.2.4. リソースの状態の確認

pcs resource を使用して、すべてのリソースのステータスを確認します。これにより、リソースのリストと現在のステータスが出力されます。

以下に例を示します。

```
# pcs resource
  * Clone Set: SAPHanaTopology_RH2_02-clone [SAPHanaTopology_RH2_02]:
    * Started: [ clusternode1 clusternode2 ]
  * Clone Set: SAPHana_RH2_02-clone [SAPHana_RH2_02] (promotable):
    * Promoted: [ clusternode1 ]
    * Unpromoted: [ clusternode2 ]
  * vip_RH2_02_MASTER (ocf:heartbeat:IPaddr2):  Started clusternode1
```

6.2.5. リソース設定の確認

次のコマンドで、現在のリソース設定が表示されます。

```
# pcs resource config
Resource: vip_RH2_02_MASTER (class=ocf provider=heartbeat type=IPaddr2)
  Attributes: vip_RH2_02_MASTER-instance_attributes
    ip=192.168.5.136
  Operations:
    monitor: vip_RH2_02_MASTER-monitor-interval-10s
      interval=10s
      timeout=20s
    start: vip_RH2_02_MASTER-start-interval-0s
      interval=0s
      timeout=20s
    stop: vip_RH2_02_MASTER-stop-interval-0s
      interval=0s
      timeout=20s
Clone: SAPHanaTopology_RH2_02-clone
  Meta Attributes: SAPHanaTopology_RH2_02-clone-meta_attributes
    clone-max=2
    clone-node-max=1
    interleave=true
  Resource: SAPHanaTopology_RH2_02 (class=ocf provider=heartbeat type=SAPHanaTopology)
    Attributes: SAPHanaTopology_RH2_02-instance_attributes
      InstanceNumber=02
      SID=RH2
    Operations:
      methods: SAPHanaTopology_RH2_02-methods-interval-0s
        interval=0s
        timeout=5
      monitor: SAPHanaTopology_RH2_02-monitor-interval-10
        interval=10
        timeout=600
      reload: SAPHanaTopology_RH2_02-reload-interval-0s
        interval=0s
        timeout=5
      start: SAPHanaTopology_RH2_02-start-interval-0s
        interval=0s
        timeout=600
      stop: SAPHanaTopology_RH2_02-stop-interval-0s
        interval=0s
```

```
        timeout=600
  Clone: SAPHana_RH2_02-clone
    Meta Attributes: SAPHana_RH2_02-clone-meta_attributes
      clone-max=2
      clone-node-max=1
      interleave=true
      notify=true
      promotable=true
    Resource: SAPHana_RH2_02 (class=ocf provider=heartbeat type=SAPHana)
      Attributes: SAPHana_RH2_02-instance_attributes
        AUTOMATED_REGISTER=true
        DUPLICATE_PRIMARY_TIMEOUT=300
        HANA_CALL_TIMEOUT=10
        InstanceNumber=02
        PREFER_SITE_TAKEOVER=true
        SID=RH2
      Operations:
        demote: SAPHana_RH2_02-demote-interval-0s
          interval=0s
          timeout=3600
        methods: SAPHana_RH2_02-methods-interval-0s
          interval=0s
          timeout=5
        monitor: SAPHana_RH2_02-monitor-interval-251
          interval=251
          timeout=700
          role=Unpromoted
        monitor: SAPHana_RH2_02-monitor-interval-249
          interval=249
          timeout=700
          role=Promoted
        promote: SAPHana_RH2_02-promote-interval-0s
          interval=0s
          timeout=3600
        reload: SAPHana_RH2_02-reload-interval-0s
          interval=0s
          timeout=5
        start: SAPHana_RH2_02-start-interval-0s
          interval=0s
          timeout=3200
        stop: SAPHana_RH2_02-stop-interval-0s
          interval=0s
          timeout=3100
```

これには、インストールおよび設定されたリソースエージェントの設定に使用されるすべてのパラメーターがリスト表示されます。

6.2.6. SAPHana リソースオプションの **AUTOMATED_REGISTER=true**

このオプションを SAPHana リソースで使用すると、Pacemaker はセカンダリーデータベースを自動的に再登録します。

最初のテストではこのオプションを使用することを推奨します。**AUTOMATED_REGISTER=false** を使用した場合、管理者はセカンダリーノードを手動で再登録する必要があります。

6.2.7. リソースの処理

リソース管理には複数のオプションがあります。詳細は、利用可能なヘルプを参照してください。

```
# pcs resource help
```

使用されているリソースエージェントをリスト表示します。

```
# pcs resource config | grep "type=" | awk -F"type=" '{ print $2 }' | sed -e "s/)//g"
```

出力例:

```
IPaddr2
SAPHanaTopology
SAPHana
```

特定のリソースエージェントの説明と設定パラメーターを表示します。

```
# pcs resource describe <resource agent>
```

例 (出力なし):

```
# pcs resource describe IPaddr2
```

リソースエージェント **IPaddr2** の例 (出力付き):

```
Assumed agent name 'ocf:heartbeat:IPaddr2' (deduced from 'IPaddr2')
ocf:heartbeat:IPaddr2 - Manages virtual IPv4 and IPv6 addresses (Linux specific version)

This Linux-specific resource manages IP alias IP addresses. It can add an IP alias, or remove one. In
addition, it can implement Cluster Alias IP functionality if invoked as a clone resource.  If used as a
clone, "shared address with a trivial, stateless (autonomous) load-balancing/mutual exclusion on
ingress" mode gets applied (as opposed to "assume resource uniqueness" mode otherwise). For that,
Linux
firewall (kernel and userspace) is assumed, and since recent distributions are ambivalent in plain
"iptables" command to particular back-end resolution, "iptables-legacy" (when present) gets
prioritized
so as to avoid incompatibilities (note that respective ipt_CLUSTERIP firewall extension in use here is,
at the same time, marked deprecated, yet said "legacy" layer can make it workable, literally, to this
day) with "netfilter" one (as in "iptables-nft"). In that case, you should explicitly set clone-node-max
>= 2, and/or clone-max < number of nodes. In case of node failure, clone instances need to be re-
allocated on surviving nodes. This would not be possible if there is already an instance on those
nodes,
and clone-node-max=1 (which is the default).  When the specified IP address gets assigned to a
respective interface, the resource agent sends unsolicited ARP (Address Resolution Protocol, IPv4)
or NA
(Neighbor Advertisement, IPv6) packets to inform neighboring machines about the change. This
functionality is controlled for both IPv4 and IPv6 by shared 'arp_*' parameters.

Resource options:
  ip (required) (unique): The IPv4 (dotted quad notation) or IPv6 address (colon hexadecimal notation)
      example IPv4 "192.168.1.1". example IPv6 "2001:db8:DC28:0:0:FC57:D4C8:1FFF".
  nic: The base network interface on which the IP address will be brought online.  If left empty, the
      script will try and determine this from the routing table.  Do NOT specify an alias interface in
```

```
     the form eth0:1 or anything here; rather, specify the base interface only. If you want a label,
     see the iflabel parameter.  Prerequisite:  There must be at least one static IP address, which is
     not managed by the cluster, assigned to the network interface. If you can not assign any static IP
     address on the interface, modify this kernel parameter:  sysctl -w
     net.ipv4.conf.all.promote_secondaries=1 # (or per device)
  cidr_netmask: The netmask for the interface in CIDR format (e.g., 24 and not 255.255.255.0)  If
     unspecified, the script will also try to determine this from the routing table.
  broadcast: Broadcast address associated with the IP. It is possible to use the special symbols '+' and
     '-' instead of the broadcast address. In this case, the broadcast address is derived by
     setting/resetting the host bits of the interface prefix.
  iflabel: You can specify an additional label for your IP address here. This label is appended to your
     interface name.  The kernel allows alphanumeric labels up to a maximum length of 15 characters
     including the interface name and colon (e.g. eth0:foobar1234)  A label can be specified in nic
     parameter but it is deprecated. If a label is specified in nic name, this parameter has no effect.
  lvs_support: Enable support for LVS Direct Routing configurations. In case a IP address is stopped,
     only move it to the loopback device to allow the local node to continue to service requests, but
     no longer advertise it on the network.  Notes for IPv6: It is not necessary to enable this option
     on IPv6. Instead, enable 'lvs_ipv6_addrlabel' option for LVS-DR usage on IPv6.
  lvs_ipv6_addrlabel: Enable adding IPv6 address label so IPv6 traffic originating from the address's
     interface does not use this address as the source. This is necessary for LVS-DR health checks to
     realservers to work. Without it, the most recently added IPv6 address (probably the address
added
     by IPaddr2) will be used as the source address for IPv6 traffic from that interface and since that
     address exists on loopback on the realservers, the realserver response to pings/connections will
     never leave its loopback. See RFC3484 for the detail of the source address selection.  See also
     'lvs_ipv6_addrlabel_value' parameter.
  lvs_ipv6_addrlabel_value: Specify IPv6 address label value used when 'lvs_ipv6_addrlabel' is
enabled.
     The value should be an unused label in the policy table which is shown by 'ip addrlabel list'
     command. You would rarely need to change this parameter.
  mac: Set the interface MAC address explicitly. Currently only used in case of the Cluster IP Alias.
     Leave empty to chose automatically.
  clusterip_hash: Specify the hashing algorithm used for the Cluster IP functionality.
  unique_clone_address: If true, add the clone ID to the supplied value of IP to create a unique
address
     to manage
  arp_interval: Specify the interval between unsolicited ARP (IPv4) or NA (IPv6) packets in
     milliseconds.  This parameter is deprecated and used for the backward compatibility only. It is
     effective only for the send_arp binary which is built with libnet, and send_ua for IPv6. It has no
     effect for other arp_sender.
  arp_count: Number of unsolicited ARP (IPv4) or NA (IPv6) packets to send at resource initialization.
  arp_count_refresh: For IPv4, number of unsolicited ARP packets to send during resource
monitoring.
     Doing so helps mitigate issues of stuck ARP caches resulting from split-brain situations.
  arp_bg: Whether or not to send the ARP (IPv4) or NA (IPv6) packets in the background. The default
is
     true for IPv4 and false for IPv6.
  arp_sender: For IPv4, the program to send ARP packets with on start. Available options are:  -
     send_arp: default  - ipoibarping: default for infiniband interfaces if ipoibarping is available  -
     iputils_arping: use arping in iputils package  - libnet_arping: use another variant of arping
     based on libnet
  send_arp_opts: For IPv4, extra options to pass to the arp_sender program. Available options are
vary
     depending on which arp_sender is used.  A typical use case is specifying '-A' for iputils_arping
     to use ARP REPLY instead of ARP REQUEST as Gratuitous ARPs.
  flush_routes: Flush the routing table on stop. This is for applications which use the cluster IP
```

```
      address and which run on the same physical host that the IP address lives on. The Linux kernel
may
      force that application to take a shortcut to the local loopback interface, instead of the
      interface the address is really bound to. Under those circumstances, an application may,
somewhat
      unexpectedly, continue to use connections for some time even after the IP address is
deconfigured.
      Set this parameter in order to immediately disable said shortcut when the IP address goes away.
  run_arping: For IPv4, whether or not to run arping for collision detection check.
  nodad: For IPv6, do not perform Duplicate Address Detection when adding the address.
  noprefixroute: Use noprefixroute flag (see 'man ip-address').
  preferred_lft: For IPv6, set the preferred lifetime of the IP address. This can be used to ensure that
      the created IP address will not be used as a source address for routing. Expects a value as
      specified in section 5.5.4 of RFC 4862.
  network_namespace: Specifies the network namespace to operate within. The namespace must
already
      exist, and the interface to be used must be within the namespace.

Default operations:
  start:
    interval=0s
    timeout=20s
  stop:
    interval=0s
    timeout=20s
  monitor:
    interval=10s
    timeout=20s
```

クラスターが停止すると、すべてのリソースも停止します。クラスターが **maintenance-mode** になると、すべてのリソースは現在のステータスのままになりますが、監視または管理されなくなります。

6.2.8. maintenance mode のクラスタープロパティーの処理

定義されているすべてのプロパティーをリスト表示します。

```
[root@clusternode1] pcs property
Cluster Properties:
 cluster-infrastructure: corosync
 cluster-name: cluster1
 concurrent-fencing: true
 dc-version: 2.1.5-7.el9-a3f44794f94
 hana_rh2_site_srHook_DC1: PRIM
 hana_rh2_site_srHook_DC2: SFAIL
 have-watchdog: false
 last-lrm-refresh: 1688548036
 maintenance-mode: true
 priority-fencing-delay: 10s
 stonith-enabled: true
 stonith-timeout: 900
```

データベースを再設定するには、設定が完了するまで変更を無視するようにクラスターに指示する必要があります。以下を使用してクラスターを **maintenance-mode** にできます。

```
# pcs property set maintenance-mode=true
```

maintenance-mode を確認します。

```
# pcs resource
  * Clone Set: SAPHanaTopology_RH2_02-clone [SAPHanaTopology_RH2_02] (unmanaged):
    * SAPHanaTopology_RH2_02 (ocf:heartbeat:SAPHanaTopology):  Started clusternode1
(unmanaged)
    * SAPHanaTopology_RH2_02 (ocf:heartbeat:SAPHanaTopology):  Started clusternode2
(unmanaged)
  * Clone Set: SAPHana_RH2_02-clone [SAPHana_RH2_02] (promotable, unmanaged):
    * SAPHana_RH2_02 (ocf:heartbeat:SAPHana):  Promoted clusternode1 (unmanaged)
    * SAPHana_RH2_02 (ocf:heartbeat:SAPHana):  Unpromoted clusternode2 (unmanaged)
  * vip_RH2_02_MASTER (ocf:heartbeat:IPaddr2):  Started clusternode1 (unmanaged)
```

すべてのリソースが "unmanaged" であることを確認します。

```
[root@clusternode1]# pcs status
Cluster name: cluster1
Status of pacemakerd: 'Pacemaker is running' (last updated 2023-06-27 16:02:15 +02:00)
Cluster Summary:
  * Stack: corosync
  * Current DC: clusternode2 (version 2.1.5-7.el9-a3f44794f94) - partition with quorum
  * Last updated: Tue Jun 27 16:02:16 2023
  * Last change:  Tue Jun 27 16:02:14 2023 by root via cibadmin on clusternode1
  * 2 nodes configured
  * 6 resource instances configured

              *** Resource management is DISABLED ***
  The cluster will not attempt to start, stop or recover services

Node List:
  * Online: [ clusternode1 clusternode2 ]

Full List of Resources:
  * h7fence (stonith:fence_rhevm):  Started clusternode2 (unmanaged)
  * Clone Set: SAPHanaTopology_RH2_02-clone [SAPHanaTopology_RH2_02] (unmanaged):
    * SAPHanaTopology_RH2_02 (ocf:heartbeat:SAPHanaTopology):  Started clusternode1
(unmanaged)
    * SAPHanaTopology_RH2_02 (ocf:heartbeat:SAPHanaTopology):  Started clusternode2
(unmanaged)
  * Clone Set: SAPHana_RH2_02-clone [SAPHana_RH2_02] (promotable, unmanaged):
    * SAPHana_RH2_02 (ocf:heartbeat:SAPHana):  Promoted clusternode1 (unmanaged)
    * SAPHana_RH2_02 (ocf:heartbeat:SAPHana):  Unpromoted clusternode2 (unmanaged)
  * vip_RH2_02_MASTER (ocf:heartbeat:IPaddr2):  Started clusternode1 (unmanaged)

Daemon Status:
  corosync: active/enabled
  pacemaker: active/enabled
  pcsd: active/enabled
```

maintenance-mode の設定を解除すると、リソースは managed に戻ります。

```
# pcs property set maintenance-mode=false
```

6.2.9. move を使用した SAPHana リソースのフェイルオーバー

SAP HANA データベースをフェイルオーバーする簡単な例は、**pcs resource move** コマンドを使用することです。以下に示すように、クローンリソース名を使用してリソースを移動する必要があります。

```
# pcs resource move <SAPHana-clone-resource>
```

この例では、クローンリソースは **SAPHana_RH2_02-clone** です。

```
[root@clusternode1]# pcs resource
  * Clone Set: SAPHanaTopology_RH2_02-clone [SAPHanaTopology_RH2_02]:
    * Started: [ clusternode1 clusternode2 ]
  * Clone Set: SAPHana_RH2_02-clone [SAPHana_RH2_02] (promotable):
    * Promoted: [ clusternode1 ]
    * Unpromoted: [ clusternode2 ]
  * vip_RH2_02_MASTER (ocf:heartbeat:IPaddr2):  Started clusternode1
```

リソースを移動します。

```
# pcs resource move SAPHana_RH2_02-clone
Location constraint to move resource 'SAPHana_RH2_02-clone' has been created
Waiting for the cluster to apply configuration changes...
Location constraint created to move resource 'SAPHana_RH2_02-clone' has been removed
Waiting for the cluster to apply configuration changes...
resource 'SAPHana_RH2_02-clone' is promoted on node 'clusternode2'; unpromoted on node
'clusternode1'
```

制約が残っているかどうかを確認します。

```
# pcs constraint location
```

リソースをクリアすることで、フェイルオーバー中に作成された場所の制約を削除できます。以下に例を示します。

```
[root@clusternode1]# pcs resource clear SAPHana_RH2_02-clone
```

"Migration Summary" に警告やエントリーが残っているかどうかを確認します。

```
# pcs status --full
```

stonith の履歴を確認します。

```
# pcs stonith history
```

必要に応じて、stonith の履歴をクリアします。

```
# pcs stonith history cleanup
```

Pacemaker バージョン 2.1.5 より前のバージョンを使用している場合は、Is there a way to manage constraints when running pcs resource move? を参照し、残っている制約を確認してください。

6.2.10. フェイルオーバーと同期状態の監視

すべての Pacemaker のアクティビティーは、クラスターノードの **/var/log/messages** ファイルに記録

されます。他にも多くのメッセージがあるため、SAP リソースエージェントに関連するメッセージを確認するのが難しい場合があります。SAP リソースエージェントに関連するメッセージのみをフィルターするコマンドエイリアスを設定できます。

エイリアスの例 **tmsl**:

```
# alias tmsl='tail -1000f /var/log/messages | egrep -s "Setting master-
rsc_SAPHana_${SAPSYSTEMNAME}_HDB${TINSTANCE}|sr_register|WAITING4LPA|PROMOTED|
DEMOTED|UNDEFINED|master_walk|SWAIT|WaitforStopped|FAILED|LPT"'
```

tsml の出力例:

```
[root@clusternode1]# tmsl
Jun 22 13:59:54 clusternode1 SAPHana(SAPHana_RH2_02)[907482]: INFO: DEC: Finally
get_SRHOOK()=SOK
Jun 22 13:59:55 clusternode1 SAPHana(SAPHana_RH2_02)[907482]: INFO: DEC: secondary with
sync status SOK ==> possible takeover node
Jun 22 13:59:55 clusternode1 SAPHana(SAPHana_RH2_02)[907482]: INFO: DEC:
hana_rh2_site_srHook_DC1=SWAIT
Jun 22 13:59:55 clusternode1 SAPHana(SAPHana_RH2_02)[907482]: INFO: DEC:
hana_rh2_site_srHook_DC1 is empty or SWAIT. Take polling attribute: hana_rh2_sync_state=SOK
Jun 22 13:59:55 clusternode1 SAPHana(SAPHana_RH2_02)[907482]: INFO: DEC: Finally
get_SRHOOK()=SOK
Jun 22 13:59:55 clusternode1 SAPHana(SAPHana_RH2_02)[907482]: INFO: DEC:
saphana_monitor_secondary: scoring_crm_master(4:S:master1:master:worker:master,SOK)
Jun 22 13:59:55 clusternode1 SAPHana(SAPHana_RH2_02)[907482]: INFO: DEC:
scoring_crm_master: sync(SOK) is matching syncPattern (SOK)
Jun 22 14:04:06 clusternode1 SAPHana(SAPHana_RH2_02)[914625]: INFO: DEC:
hana_rh2_site_srHook_DC1=SWAIT
Jun 22 14:04:06 clusternode1 SAPHana(SAPHana_RH2_02)[914625]: INFO: DEC:
hana_rh2_site_srHook_DC1 is empty or SWAIT. Take polling attribute: hana_rh2_sync_state=SOK
Jun 22 14:04:06 clusternode1 SAPHana(SAPHana_RH2_02)[914625]: INFO: DEC: Finally
get_SRHOOK()=SOK
Jun 22 14:04:09 clusternode1 SAPHana(SAPHana_RH2_02)[914625]: INFO: DEC:
hana_rh2_site_srHook_DC1=SWAIT
Jun 22 14:04:09 clusternode1 SAPHana(SAPHana_RH2_02)[914625]: INFO: DEC:
hana_rh2_site_srHook_DC1 is empty or SWAIT. Take polling attribute: hana_rh2_sync_state=SOK
Jun 22 14:04:09 clusternode1 SAPHana(SAPHana_RH2_02)[914625]: INFO: DEC: Finally
get_SRHOOK()=SOK
Jun 22 14:04:09 clusternode1 SAPHana(SAPHana_RH2_02)[914625]: INFO: DEC: secondary with
sync status SOK ==> possible takeover node
Jun 22 14:04:09 clusternode1 SAPHana(SAPHana_RH2_02)[914625]: INFO: DEC:
hana_rh2_site_srHook_DC1=SWAIT
Jun 22 14:04:09 clusternode1 SAPHana(SAPHana_RH2_02)[914625]: INFO: DEC:
hana_rh2_site_srHook_DC1 is empty or SWAIT. Take polling attribute: hana_rh2_sync_state=SOK
Jun 22 14:04:09 clusternode1 SAPHana(SAPHana_RH2_02)[914625]: INFO: DEC: Finally
get_SRHOOK()=SOK
Jun 22 14:04:09 clusternode1 SAPHana(SAPHana_RH2_02)[914625]: INFO: DEC:
saphana_monitor_secondary: scoring_crm_master(4:S:master1:master:worker:master,SOK)
Jun 22 14:04:09 clusternode1 SAPHana(SAPHana_RH2_02)[914625]: INFO: DEC:
scoring_crm_master: sync(SOK) is matching syncPattern (SOK)
Jun 22 14:08:21 clusternode1 SAPHana(SAPHana_RH2_02)[922136]: INFO: DEC:
hana_rh2_site_srHook_DC1=SWAIT
Jun 22 14:08:21 clusternode1 SAPHana(SAPHana_RH2_02)[922136]: INFO: DEC:
hana_rh2_site_srHook_DC1 is empty or SWAIT. Take polling attribute: hana_rh2_sync_state=SOK
```

```
Jun 22 14:08:21 clusternode1 SAPHana(SAPHana_RH2_02)[922136]: INFO: DEC: Finally
get_SRHOOK()=SOK
Jun 22 14:08:23 clusternode1 SAPHana(SAPHana_RH2_02)[922136]: INFO: DEC:
hana_rh2_site_srHook_DC1=SWAIT
Jun 22 14:08:23 clusternode1 SAPHana(SAPHana_RH2_02)[922136]: INFO: DEC:
hana_rh2_site_srHook_DC1 is empty or SWAIT. Take polling attribute: hana_rh2_sync_state=SOK
Jun 22 14:08:23 clusternode1 SAPHana(SAPHana_RH2_02)[922136]: INFO: DEC: Finally
get_SRHOOK()=SOK
Jun 22 14:08:24 clusternode1 SAPHana(SAPHana_RH2_02)[922136]: INFO: DEC: secondary with
sync status SOK ==> possible takeover node
Jun 22 14:08:24 clusternode1 SAPHana(SAPHana_RH2_02)[922136]: INFO: DEC:
hana_rh2_site_srHook_DC1=SWAIT
Jun 22 14:08:24 clusternode1 SAPHana(SAPHana_RH2_02)[922136]: INFO: DEC:
hana_rh2_site_srHook_DC1 is empty or SWAIT. Take polling attribute: hana_rh2_sync_state=SOK
Jun 22 14:08:24 clusternode1 SAPHana(SAPHana_RH2_02)[922136]: INFO: DEC: Finally
get_SRHOOK()=SOK
Jun 22 14:08:24 clusternode1 SAPHana(SAPHana_RH2_02)[922136]: INFO: DEC:
saphana_monitor_secondary: scoring_crm_master(4:S:master1:master:worker:master,SOK)
Jun 22 14:08:24 clusternode1 SAPHana(SAPHana_RH2_02)[922136]: INFO: DEC:
scoring_crm_master: sync(SOK) is matching syncPattern (SOK)
Jun 22 14:12:35 clusternode1 SAPHana(SAPHana_RH2_02)[929408]: INFO: DEC:
hana_rh2_site_srHook_DC1=SWAIT
Jun 22 14:12:35 clusternode1 SAPHana(SAPHana_RH2_02)[929408]: INFO: DEC:
hana_rh2_site_srHook_DC1 is empty or SWAIT. Take polling attribute: hana_rh2_sync_state=SOK
Jun 22 14:12:36 clusternode1 SAPHana(SAPHana_RH2_02)[929408]: INFO: DEC: Finally
get_SRHOOK()=SOK
Jun 22 14:12:38 clusternode1 SAPHana(SAPHana_RH2_02)[929408]: INFO: DEC:
hana_rh2_site_srHook_DC1=SWAIT
Jun 22 14:12:38 clusternode1 SAPHana(SAPHana_RH2_02)[929408]: INFO: DEC:
hana_rh2_site_srHook_DC1 is empty or SWAIT. Take polling attribute: hana_rh2_sync_state=SOK
Jun 22 14:12:38 clusternode1 SAPHana(SAPHana_RH2_02)[929408]: INFO: DEC: Finally
get_SRHOOK()=SOK
Jun 22 14:12:38 clusternode1 SAPHana(SAPHana_RH2_02)[929408]: INFO: DEC: secondary with
sync status SOK ==> possible takeover node
Jun 22 14:12:39 clusternode1 SAPHana(SAPHana_RH2_02)[929408]: INFO: DEC:
hana_rh2_site_srHook_DC1=SWAIT
Jun 22 14:12:39 clusternode1 SAPHana(SAPHana_RH2_02)[929408]: INFO: DEC:
hana_rh2_site_srHook_DC1 is empty or SWAIT. Take polling attribute: hana_rh2_sync_state=SOK
Jun 22 14:12:39 clusternode1 SAPHana(SAPHana_RH2_02)[929408]: INFO: DEC: Finally
get_SRHOOK()=SOK
Jun 22 14:12:39 clusternode1 SAPHana(SAPHana_RH2_02)[929408]: INFO: DEC:
saphana_monitor_secondary: scoring_crm_master(4:S:master1:master:worker:master,SOK)
Jun 22 14:12:39 clusternode1 SAPHana(SAPHana_RH2_02)[929408]: INFO: DEC:
scoring_crm_master: sync(SOK) is matching syncPattern (SOK)
Jun 22 14:14:01 clusternode1 pacemaker-attrd[10150]: notice: Setting
hana_rh2_clone_state[clusternode2]: PROMOTED -> DEMOTED
Jun 22 14:14:02 clusternode1 pacemaker-attrd[10150]: notice: Setting
hana_rh2_clone_state[clusternode2]: DEMOTED -> UNDEFINED
Jun 22 14:14:19 clusternode1 pacemaker-attrd[10150]: notice: Setting
hana_rh2_clone_state[clusternode1]: DEMOTED -> PROMOTED
Jun 22 14:14:21 clusternode1 SAPHana(SAPHana_RH2_02)[932762]: INFO: DEC:
hana_rh2_site_srHook_DC1=SWAIT
Jun 22 14:14:21 clusternode1 SAPHana(SAPHana_RH2_02)[932762]: INFO: DEC:
hana_rh2_site_srHook_DC1 is empty or SWAIT. Take polling attribute: hana_rh2_sync_state=SOK
Jun 22 14:14:21 clusternode1 SAPHana(SAPHana_RH2_02)[932762]: INFO: DEC: Finally
get_SRHOOK()=SOK
```

```
Jun 22 14:15:14 clusternode1 SAPHana(SAPHana_RH2_02)[932762]: INFO: DEC:
hana_rh2_site_srHook_DC1=SWAIT
Jun 22 14:15:22 clusternode1 pacemaker-attrd[10150]: notice: Setting
hana_rh2_sync_state[clusternode1]: SOK -> PRIM
Jun 22 14:15:23 clusternode1 pacemaker-attrd[10150]: notice: Setting
hana_rh2_sync_state[clusternode2]: PRIM -> SOK
Jun 22 14:15:23 clusternode1 SAPHana(SAPHana_RH2_02)[934810]: INFO: ACT site=DC1, setting
SOK for secondary (1)
Jun 22 14:15:25 clusternode1 pacemaker-attrd[10150]: notice: Setting
hana_rh2_clone_state[clusternode2]: UNDEFINED -> DEMOTED
Jun 22 14:15:32 clusternode1 pacemaker-attrd[10150]: notice: Setting
hana_rh2_sync_state[clusternode2]: SOK -> SFAIL
Jun 22 14:19:36 clusternode1 pacemaker-attrd[10150]: notice: Setting
hana_rh2_sync_state[clusternode2]: SFAIL -> SOK
Jun 22 14:19:36 clusternode1 SAPHana(SAPHana_RH2_02)[942693]: INFO: ACT site=DC1, setting
SOK for secondary (1)
Jun 22 14:23:49 clusternode1 SAPHana(SAPHana_RH2_02)[950623]: INFO: ACT site=DC1, setting
SOK for secondary (1)
Jun 22 14:28:02 clusternode1 SAPHana(SAPHana_RH2_02)[958633]: INFO: ACT site=DC1, setting
SOK for secondary (1)
Jun 22 14:32:15 clusternode1 SAPHana(SAPHana_RH2_02)[966683]: INFO: ACT site=DC1, setting
SOK for secondary (1)
Jun 22 14:36:27 clusternode1 SAPHana(SAPHana_RH2_02)[974736]: INFO: ACT site=DC1, setting
SOK for secondary (1)
Jun 22 14:40:40 clusternode1 SAPHana(SAPHana_RH2_02)[982934]: INFO: ACT site=DC1, setting
SOK for secondary (1)
```

フィルターを使用すると、どのようなステータス変化が発生しているかを理解しやすくなります。詳細がない場合は、メッセージファイル全体を開くと、すべての情報を確認できます。

フェイルオーバー後、リソースをクリアできます。場所の制約が残っていないことも確認してください。

6.2.11. クラスターの整合性の確認

インストール中、設定が最終的に完了する前にリソースが起動することがあります。これにより、Cluster Information Base (CIB) のエントリーが発生し、誤った動作が発生する可能性があります。これは簡単に確認でき、設定の完了後に手動で修正することもできます。

SAPHana リソースを起動すると、欠落しているエントリーが再作成されます。間違ったエントリーには pcs コマンドでは対処できないため、手動で削除する必要があります。

CIB エントリーを確認します。

```
# cibadmin --query
```

クラスターメンバーが DC1 と DC2 であり、ノード間の同期状態が SOK と報告される場合、DC3 と SFAIL のエントリーはクラスター情報ベースに存在しないはずです。

対応するエントリーを確認する例:

```
# cibadmin --query |grep '"DC3"'
# cibadmin --query |grep '"SFAIL"'
```

このコマンドは、クラスター内の任意のノードで root ユーザーとして実行できます。通常、コマンドの出力は空です。設定にまだエラーがある場合、出力は次のようになります。

```
        <nvpair id="SAPHanaSR-hana_rh1_glob_sec" name="hana_rh1_glob_sec" value="DC3"/>
```

これらのエントリーは、次のコマンドを使用して削除できます。

```
# cibadmin --delete --xml-text '<...>'
```

上の例のエントリーを削除するには、次のように入力する必要があります。出力には二重引用符が含まれるため、テキストを一重引用符で囲む必要があることに注意してください。

```
# cibadmin --delete --xml-text '        <nvpair id="SAPHanaSR-hana_rh1_glob_sec"
name="hana_rh1_glob_sec" value="DC3"/>'
```

削除された CIB エントリーがないことを確認します。返される出力は空である必要があります。

```
# cibadmin --query |grep 'DC3"'
```

6.2.12. クラスターのクリーンアップ

フェイルオーバーテスト中に、以前のテストの制約やその他の残留物が残ることがあります。次のテストを開始する前に、クラスターからこれらをクリアする必要があります。

クラスターのステータスで障害イベントがないか確認します。

```
# pcs status --full
```

"Migration Summary" にクラスターの警告またはエントリーが表示された場合は、リソースをクリアしてクリーンアップする必要があります。

```
# pcs resource clear SAPHana_RH2_02-clone
# pcs resource cleanup SAPHana_RH2_02-clone
```

出力:

```
Cleaned up SAPHana_RH2_02:0 on clusternode1
Cleaned up SAPHana_RH2_02:1 on clusternode2
```

以前のフェイルオーバーなどで使用された不要な場所の制約があるかどうかを確認します。

```
# pcs constraint location
```

既存の制約をさらに詳しく確認します。

```
# pcs constraint --full
```

リソース移動後の場所の制約の例:

```
      Node: hana08 (score:-INFINITY) (role:Started) (id:cli-ban-SAPHana_RH2_02-clone-on-hana08)
```

この場所の制約をクリアします。

```
# pcs resource clear SAPHana_RH2_02-clone
```

制約が制約リストから消えていることを確認します。まだ残っている場合は、制約 ID を使用して明示的に削除します。

```
# pcs constraint delete cli-ban-SAPHana_RH2_02-clone-on-hana08
```

フェンシングを使用して複数のテストを実行する場合は、**stonith** の履歴をクリアすることもできます。

```
# pcs stonith history cleanup
```

pcs コマンドはすべて root ユーザーとして実行します。残留物の検出 も参照してください。

6.2.13. その他のクラスターコマンド

さまざまなクラスターコマンドの例

```
# pcs status --full
# crm_mon -1Arf # Provides an overview
# pcs resource # Lists all resources and shows if they are running
# pcs constraint --full # Lists all constraint ids which should be removed
# pcs cluster start --all # This will start the cluster on all nodes
# pcs cluster stop --all # This will stop the cluster on all nodes
# pcs node attribute # Lists node attributes
```

6.3. RHEL および一般的なコマンド

6.3.1. 現在のステータスの検出

環境の現在のステータスを知るには、いくつかの手順に従う必要があります。環境の監視 を参照してください。また、次のことを行うことを推奨します。

- **/var/log/messages** を確認し、ログの確認を容易にするために モニタリング用のエイリアス を使用します。
- 場合によっては、適切な操作を継続するために、クラスターを以前のアクティビティーからクリーンアップする必要があります。残留物を検出 し、必要に応じてクリアします。

6.3.2. yum info

```
# yum info resource-agents-sap-hana
Last metadata expiration check: 2:47:28 ago on Tue 06 Jun 2023 03:13:57 AM CEST.
Installed Packages
Name         : resource-agents-sap-hana
Epoch        : 1
Version      : 0.162.1
Release      : 2.el9_2
Architecture : noarch
Size         : 174 k
```

```
Source       : resource-agents-sap-hana-0.162.1-2.el9_2.src.rpm
Repository   : @System
Summary      : SAP HANA cluster resource agents
URL          : https://github.com/SUSE/SAPHanaSR
License      : GPLv2+
Description  : The SAP HANA resource agents interface with Pacemaker to allow
             : SAP instances to be managed in a cluster environment.
```

6.3.3. RPM バージョン表示

```
# rpm -q resource-agents-sap-hana
resource-agents-sap-hana-0.162.1-2.el9_2.noarch
```

6.3.4. モニタリング用のエイリアス

これをシェルプロファイルに追加できます。この例では、root のエイリアスは **<sid>adm** のエイリアスに依存しているため、これもすでに定義されている必要があります。

- root (~/**.bashrc** に追加):

```
# export ListInstances=$(/usr/sap/hostctrl/exe/saphostctrl -function ListInstances| head -1 )
export sid=$(echo "$ListInstances" |cut -d " " -f 5| tr [A-Z] [a-z])
export SID=$(echo $sid | tr [a-z] [A-Z])
export Instance=$(echo "$ListInstances" |cut -d " " -f 7 )
alias crmm='watch -n 1 crm_mon -1Arf'
alias crmv='watch -n 1 /usr/local/bin/crmmv'
alias cglo='su - ${sid}adm -c cglo'
alias cdh='cd /usr/lib/ocf/resource.d/heartbeat'
alias gtr='su - ${sid}adm  -c gtr'
alias hdb='su - ${sid}adm  -c hdb'
alias hdbi='su - ${sid}adm  -c hdbi'
alias hgrep='history | grep $1'
alias hri='su - ${sid}adm  -c hri'
alias hris='su - ${sid}adm  -c hris'
alias killnode="echo 'b' > /proc/sysrq-trigger"
alias lhc='su - ${sid}adm  -c lhc'
alias pit='ssh pitunnel'
alias python='/usr/sap/${SID}/HDB${Instance}/exe/Python/bin/python'
alias srstate='su - ${sid}adm  -c srstate'
alias shr='watch -n 5 "SAPHanaSR-monitor --sid=${SID}"'
alias sgsi='su - ${sid}adm  -c sgsi'
alias srm='su - ${sid}adm  -c srm'
alias srs='su - ${sid}adm  -c srs'
alias sapstart='su - ${sid}adm  -c sapstart'
alias sapstop='su - ${sid}adm  -c sapstop'
alias tma='tmux attach -t `tmux ls | grep -v atta| head -1 |cut -d " " -f 1`'
alias tm='tail -100f /var/log/messages |grep -v systemd'
alias tms='tail -1000f /var/log/messages | egrep -s "Setting master-
rsc_SAPHana_${SID}_HDB${Instance}|sr_register|WAITING4
LPA|EXCLUDE as possible takeover
node|SAPHanaSR|failed|${HOSTNAME}|PROMOTED|DEMOTED|UNDEFINED|master_walk
|SWAIT|WaitforStop
ped|FAILED"'
alias tmss='tail -1000f /var/log/messages | grep -v systemd| egrep -s "secondary with sync
```

```
status|Setting master-rsc_SAPHa
na_${SID}_HDB${Instance}|sr_register|WAITING4LPA|EXCLUDE as possible takeover
node|SAPHanaSR|failed|${HOSTNAME}|PROMOTED|DE
MOTED|UNDEFINED|master_walk|SWAIT|WaitforStopped|FAILED"'
alias tmm='tail -1000f /var/log/messages | egrep -s "Setting master-
rsc_SAPHana_${SID}_HDB${Instance}|sr_register|WAITING4
LPA|PROMOTED|DEMOTED|UNDEFINED|master_walk|SWAIT|WaitforStopped|FAILED|LPT
|SOK|SFAIL|SAPHanaSR-mon"| grep -v systemd'
alias tmsl='tail -1000f /var/log/messages | egrep -s "Setting master-
rsc_SAPHana_${SID}_HDB${Instance}|sr_register|WAITING
4LPA|PROMOTED|DEMOTED|UNDEFINED|master_walk|SWAIT|WaitforStopped|FAILED|LP
T|SOK|SFAIL|SAPHanaSR-mon"'
alias vih='vim /usr/lib/ocf/resource.d/heartbeat/SAPHanaStart'
alias vglo='su - ${sid}adm  -c vglo'
```

- **<sid>adm** (~/**.customer.sh** に追加):

```
alias tm='tail -100f /var/log/messages |grep -v systemd'
alias tms='tail -1000f /var/log/messages | egrep -s "Setting master-
rsc_SAPHana_${SAPSYSTEMNAME}_HDB${TINSTANCE}|sr_register|WAITING4LPA|EXCL
UDE as possible takeover
node|SAPHanaSR|failed|${HOSTNAME}|PROMOTED|DEMOTED|UNDEFINED|master_walk
|SWAIT|WaitforStopped|FAILED"'
alias tmsl='tail -1000f /var/log/messages | egrep -s "Setting master-
rsc_SAPHana_${SAPSYSTEMNAME}_HDB${TINSTANCE}|sr_register|WAITING4LPA|PRO
MOTED|DEMOTED|UNDEFINED|master_walk|SWAIT|WaitforStopped|FAILED|LPT"'
alias sapstart='sapcontrol -nr ${TINSTANCE} -function StartSystem HDB;hdbi'
alias sapstop='sapcontrol -nr ${TINSTANCE} -function StopSystem HDB;hdbi'
alias sgsi='watch sapcontrol -nr ${TINSTANCE} -function GetSystemInstanceList'
alias spl='watch sapcontrol -nr ${TINSTANCE} -function GetProcessList'
alias splh='watch "sapcontrol -nr ${TINSTANCE} -function GetProcessList| grep
hdbdaemon"'
alias srm='watch "hdbnsutil -sr_state --sapcontrol=1 |grep site.*Mode"'
alias srs="watch -n 5 'python
/usr/sap/${SAPSYSTEMNAME}/HDB${TINSTANCE}/exe/python_support/systemReplicationSt
atus.py ; echo Status \$?'"
alias srstate='watch -n 10 hdbnsutil -sr_state'
alias hdb='watch -n 5 "sapcontrol -nr ${TINSTANCE} -function GetProcessList| egrep -s
hdbdaemon\|hdbnameserver\|hdbindexserver "'
alias hdbi='watch -n 5 "sapcontrol -nr ${TINSTANCE} -function GetProcessList| egrep -s
hdbdaemon\|hdbnameserver\|hdbindexserver;sapcontrol -nr ${TINSTANCE} -function
GetSystemInstanceList "'
alias hgrep='history | grep $1'
alias vglo="vim /usr/sap/${SAPSYSTEMNAME}/SYS/global/hdb/custom/config/global.ini"
alias vgloh="vim
/hana/shared/${SAPSYSTEMNAME}/HDB${TINSTANCE}/${HOSTNAME}/global.ini"
alias hri='hdbcons -e hdbindexserver "replication info"'
alias hris='hdbcons -e hdbindexserver "replication info" | egrep -e
"SiteID|ReplicationStatus_"'
alias gtr='watch -n 10
/usr/sap/${SAPSYSTEMNAME}/HDB${TINSTANCE}/exe/Python/bin/python
/usr/sap/${SAPSYSTEMNAME}/HDB${TINSTANCE}/exe/python_support/getTakeoverRecom
mendation.py --sapcontrol=1'
alias lhc='/usr/sap/${SAPSYSTEMNAME}/HDB${TINSTANCE}/exe/Python/bin/python
/usr/sap/${SAPSYSTEMNAME}/HDB${TINSTANCE}/exe/python_support/landscapeHostConfi
guration.py;echo $?'
```

-

第7章 参考資料

7.1. RED HAT

- Support Policies for RHEL High Availability Clusters - Management of SAP HANA in a Cluster
- RHEL HA Add-On を使用した SAP HANA Scale-Up System Replication の自動化
- 障害発生によるリソースの移動
- Is there a way to manage constraints when running pcs resource move?

7.2. SAP

- SAP HANA Administration Guide for SAP HANA Platform
- Disaster Recovery Scenarios for Multitarget System Replication
- SAP HANA System Replication Configuration Parameter
- Example: Checking the Status on the Primary and Secondary Systems
- General Prerequisites for Configuring SAP HANA System Replication
- Change Log Modes
- Failed to re-register former primary site as new secondary site due to missing log
- Checking the Status with landscapeHostConfiguration.py
- How to Setup SAP HANA Multi-Target System Replication
- SAP HANA Multitarget System Replication