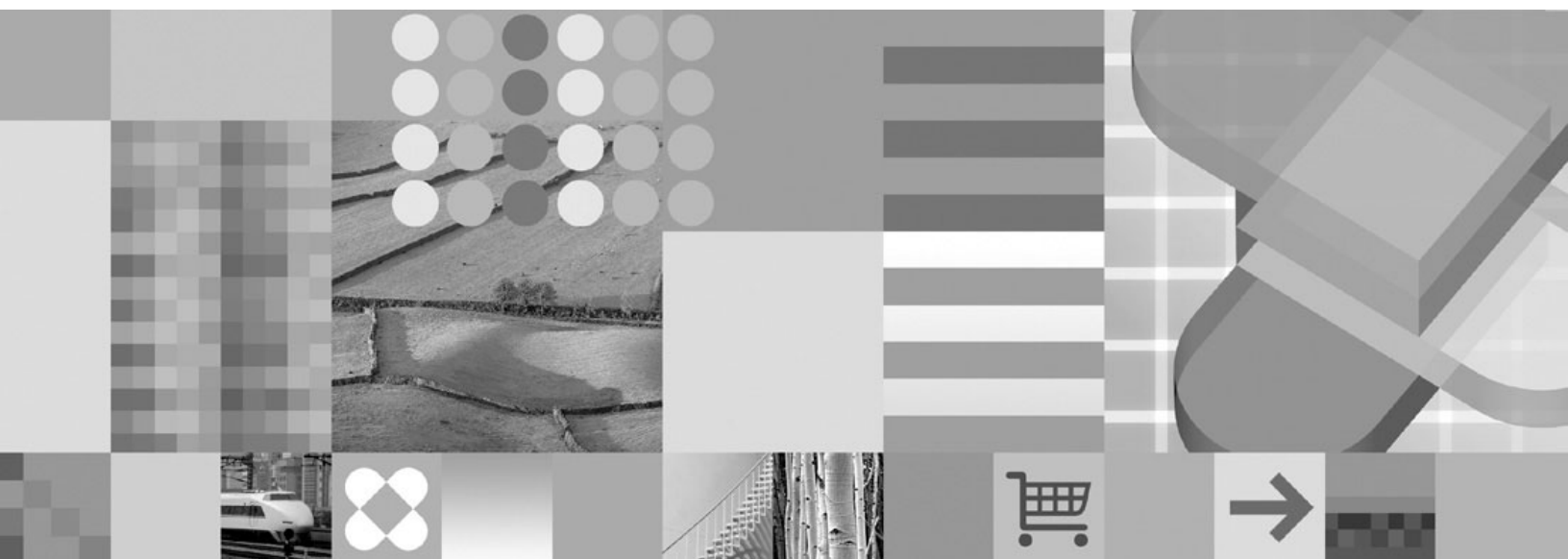




高可用性ユーザー・ガイド



高可用性ユーザー・ガイド

注記

本書および本書で紹介する製品をご使用になる前に、207 ページの『特記事項』に記載されている情報をお読みください。

本書は、バージョン 6、リリース 3 の IBM solidDB (プロダクト番号 5724-V17) および IBM solidDB Universal Cache (プロダクト番号 5724-W91)、および新しい版で明記されていない限り、以降のすべてのリリースおよびモディフィケーションに適用されます。

お客様の環境によっては、資料中の円記号がバックスラッシュと表示されたり、バックスラッシュが円記号と表示されたりする場合があります。

原典： SC23-9828-00
IBM solidDB
IBM solidDB Universal Cache
Version 6.3
High Availability User Guide

発行： 日本アイ・ビー・エム株式会社

担当： ナショナル・ランゲージ・サポート

第1刷 2009.2

© Solid Information Technology Ltd. 1993, 2008

目次

図	vii	トランザクション・ログのスペース不足	39
表	ix	管理タスクの概要	39
本書について	xi	HotStandby のリカバリーと保守の実行	40
図に関する規則	xi	特殊な構成: コストの低減か安全性の向上か	41
書体の規則	xii	コストの削減: N + 1 スペアと N + M スペアのシナリオ	42
構文表記法の規則	xiii	信頼性の向上: 2N + 1 スペアおよび 2N + M スペアのシナリオ	43
1 IBM solidDB HotStandby の紹介	1	solidDB HSB による N+1 (N+M) と 2N+1 (2N+M) の手法のサポート	43
HotStandby の動作の仕組み	2	スペアでの HAC の使用	44
HotStandby API (HSB 管理コマンド)	3	HotStandby 用の solidDB の構成	45
基本的な HotStandby サーバー・スキーム	3	2 次および 1 次ノード構成の定義 (Com セクション)	45
ハートビート	4	アプリケーションとサーバー間のタイムアウトの定義 (Com セクション)	45
トランザクション・ログと HotStandby	4	トランザクション持続性	47
サーバー HotStandby 状態	6	HotStandby 固有のパラメーターの構成	47
HotStandby と拡張レプリケーションの結合	6	1 次および 2 次 HotStandby 構成の定義	47
フェイルオーバー	7	切断された接続または使用不可の接続の検出に役立つ、HotStandby サーバー待ち時間の設定	48
サーバー・キャッチアップ	8	HotStandby データベース・コピー操作の名前と場所の定義	50
HotStandby でのレプリケーション・モード	10	2 次サーバー障害時の 1 次サーバーの動作の定義	50
サーバー状態の説明	13	パフォーマンス・チューニング	51
HotStandby によるパフォーマンスへの影響	15	安全性レベルと持続性レベルによるレプリケーション・パフォーマンスのチューニング	51
アダプティブ持続性	16	ネットコピーのパフォーマンスのチューニング (General セクション)	51
高可用性コントローラー (HAC)	19	HA コントローラーおよび HA マネージャーの構成	52
認識される障害	21	HAController セクション	53
データベース・サーバー・プロセスの制御	22	LocalDB セクション	54
外部参照エンティティ (ERE)	22	RemoteDB セクション	55
HAC でのネットワークング	24	ERE セクション	55
高可用性マネージャー	24	構成ファイルの例	55
HAC ロギング	25	solid.ini 構成ファイル	55
2 HotStandby の概要	27	solidhac.ini 構成ファイル	56
始める前に	27	HAManager.ini 構成ファイル	58
HotStandby のデモンストレーション	27	4 アプリケーションでの HotStandby の使用	59
HotStandby クイック・スタート手順	28	HotStandby サーバーに接続する 2 とおりの方法	59
HA コントローラーの開始と停止	30	接続タイプの選択	59
HA コントローラーを伴う HSB のクイック・スタート手順	31	透過接続の使用	60
開始シーケンスの要約	32	TC での障害透過性	60
3 HotStandby の管理および構成	35	透過接続でのロード・バランシング	60
必要な知識	35	透過接続情報の構文	64
HotStandby と solidDB 構成ファイル	36	TC 情報の属性の組み合わせ	65
HotStandby とアクセス権限	36	矛盾する TC 情報の処理	66
solidDB ツールと HotStandby	36	JDBC での透過接続の規定	67
データベース・マイグレーション (ディスク・ベースのサーバーのみ)	37	接続切り替えのためのプログラミング	69
相互運用性	37		
HotStandby での制限と警告	37		
インメモリー表	37		
ネットワーク分割と二重 1 次サーバー	38		

基本接続の使用	73
アプリケーションから 1 次サーバーへの再接続	73
2 次サーバーへの再接続	78
拡張レプリケーションの要件	78

5 HotStandby API コマンドの使用 . . . 81

サーバー状態の切り替え	81
切り替えの実行	81
切り替えの検証	84
フェイルオーバーの実行	84
PRIMARY ALONE 状態での新規 1 次サーバーの 実行	84
2 次サーバーのオンラインへの復帰	85
HotStandby 操作のシャットオフ	86
1 次サーバーと 2 次サーバーの同期	87
キャッチアップ	87
完全コピー	88
コピーの検証	91
ウォッチドッグを使用したサーバーの同期	91
ネットワークを介した 1 次データベースの 2 次 側へのコピー	91
2 次サーバー用の新規データベースの作成	92
2 次サーバー上の既存データベースの置き換え	94
ネットコピー状況の検証	95
1 次サーバーから指定されたディレクトリーへの データベース・ファイルのコピー	95
HotStandby サーバーの接続	97
HotStandby 状況の検査	97
切り替え状況情報の表示	98
接続状況情報の表示	98
通信情報の表示	99
ロール開始時刻の表示	99
HotStandby サーバー状態の検証	100
サーバー状態の組み合わせ	101
どちらのサーバーを 1 次サーバーにするかの選択	102
HotStandby サーバーから非 HotStandby サーバーへ の変更	103

6 障害が起きた場合の高可用性コントローラーの動作 . . . 105

1 次サーバーのデータベースの障害	105
2 次サーバーのデータベースの障害	106
1 次ノードの障害	106
2 次ノードの障害	107
HotStandby リンクの障害	108

7 HotStandby の使用によるサーバーのアップグレード . . . 111

コールドおよびホット・マイグレーション	111
HSB 互換バージョン間のマイグレーション	111
HSB 非互換バージョン間のマイグレーション	112
HSB 非互換バージョン間のマイグレーションの 準備手順	112
アップグレード後	115

付録 A. HotStandby 構成パラメーター 117

1 次および 2 次パラメーター値の調整の確認	117
1 次側の設定が 2 次側の設定より優先されるかど うかの決定	118
HotStandby 構成パラメーターの照会	119
HotStandby 構成パラメーターの変更	119
アクセス・モード	120
Cluster クラスタ	120
HotStandby セクション	120
高可用性コントローラーの構成パラメーター	124
高可用性マネージャーの構成パラメーター	129

付録 B. エラー・コード . . . 131

HotStandby のエラーおよび状況コード	131
高可用性コントローラーのエラー・コードと状況コ ード	140
solidDB データベース・エラー	142
solidDB 表エラー	145
solidDB 通信エラー	146

付録 C. HotStandby 管理コマンドの要約 . . . 147

HotStandby コマンド	147
高可用性コントローラーのコマンド	155

付録 D. サーバー状態遷移 . . . 157

HotStandby 状態遷移図	157
------------------	-----

付録 E. HotStandby システム・イベント . . . 165

付録 F. Watchdog サンプル . . . 167

Watchdog を使用した HotStandby 構成	167
Watchdog アプリケーションの動作	168
システム設計の問題	170
Watchdog 構成	171
サンプルの Watchdog アプリケーションの使用	172
障害の状態と Watchdog のアクション	173
1 次サーバーがダウンした場合	173
2 次サーバーがダウンした場合	175
Watchdog がダウンした場合	180
1 次サーバーと 2 次サーバーの間の通信リンク がダウンした場合	182
Watchdog と 1 次サーバーの間の通信リンクが ダウンした場合	185
Watchdog と 2 次サーバーの間の通信リンクが ダウンした場合	187
Watchdog と 1 次サーバーの間、および 1 次サ ーバーと 2 次サーバーの間の通信リンクがダウ ンした場合	189
Watchdog と 2 次サーバーの間、および 1 次サ ーバーと 2 次サーバーの間の通信リンクがダウ ンした場合	192
solid.ini 構成ファイルの Watchdog セクション	196

索引 . . . 201

特記事項. 207



1. HotStandby アーキテクチャー	2	17. 1 次サーバーがダウンしたシナリオと修復方 法	174
2. HotStandby サーバー・スキーム	4	18. 2 次サーバーがダウンしたシナリオと修復方 法	178
3. マスターおよびレプリカ・サーバー・スキーム での HotStandby	7	19. Watchdog がダウンしたシナリオと修復方法	181
4. 新しい 1 次側 (古い 2 次側)への HotStandby 切り替え	8	20. 1 次サーバーと 2 次サーバーの間のリンクが 切断されたシナリオと修復方法	183
5. サーバーのフェイルオーバーとキャッチアップ の例	9	21. Watchdog と 1 次サーバーの間のリンクが切 断されたシナリオと修復方法	186
6. 同期 HotStandby 構成	11	22. Watchdog と 2 次サーバーの間のリンクが切 断されたシナリオと修復方法	188
7. 高可用性コントローラーのアーキテクチャー	20	23. Watchdog と 1 次サーバーの間、および 1 次 サーバーと 2 次サーバーの間のリンクが切断 されたシナリオと修復方法	190
8. 外部参照エンティティのコンポーネント	23	24. Watchdog と 2 次サーバーの間、および 1 次 サーバーと 2 次サーバーの間のリンクが切断 されたシナリオと修復方法	194
9. 高可用性マネージャー	25		
10. 開始シーケンスの要約	33		
11. マスターのフェイルオーバー	79		
12. レプリカのフェイルオーバー	80		
13. 状態切り替え	82		
14. 手動での完全コピー手順	90		
15. HotStandby サーバー状態遷移	159		
16. Watchdog を使用した異機種混合 HotStandby 構成	171		

表

1. HotStandby の図に関する規則	xi	20. HAC 構成パラメーター: [HACController] セク ション	125
2. 書体の規則	xii	21. HAC 構成パラメーター: [LocalDB] セクショ ン	127
3. 構文表記法の規則	xiii	22. HAC 構成パラメーター: [RemoteDB] セクシ ョン	128
4. サーバー状態の説明	13	23. HAC 構成パラメーター: [ERE] セクション	129
5. インストール・シーケンスのステップ	34	24. 高可用性マネージャーの構成パラメーター	129
6. 管理タスク	40	25. HotStandby のエラーおよび状況コード	131
7. 接続タイプの選択	60	26. 高可用性コントローラーのエラー・コードと 状況コード	140
8. TC 情報の省略形	64	27. solidDB データベース・エラー	143
9. TC 情報の属性の可能な組み合わせ	66	28. solidDB 表エラー	145
10. 接続要求エラー	68	29. solidDB 通信エラー	146
11. 警告	69	30. 高可用性コントローラーのコマンド	155
12. 接続切り替え要求	70	31. サーバー状態遷移表	160
13. 通信リンク障害	70	32. HotStandby イベント	165
14. セッション状態の保存	71	33. Watchdog パラメーター	196
15. HOTSTANDBY_CONNECTSTATUS 状況値	76		
16. 接続状況の戻り値	99		
17. サーバー状態	101		
18. Cluster パラメーター	120		
19. HotStandby パラメーター	121		

本書について

IBM® solidDB® 高可用性 (HotStandby) コンポーネントを使用すると、データベース・システムの信頼性が向上し、ダウン時間が減少します。HotStandby は、「ホット・スタンバイ」アプローチを使用します。このアプローチでは、2 次データベース・サーバーが 1 次サーバーと並行して稼働し、データの正確な最新コピーを保持します。1 次データベース・サーバーに障害が発生すると、高可用性コントローラー (HAC) が 2 次側への切り替えを行います。この切り替えはアプリケーションに対して透過的に行われ、コミットされたトランザクションが失われることはなく、またパフォーマンスに対する影響も最小限にとどまります。切り替え時間は非常に短時間であり、ハードウェアおよびソフトウェアの環境の特性によって異なりますが、200 から 300 ミリ秒程度に収まります。

本書には、HotStandby コンポーネントのみに固有の情報が記載されています。solidDB データベースに関する一般的な管理および保守情報については、「*IBM solidDB 管理者ガイド*」を参照してください。

本書は、読者が、DBMS に関して一般的な知識を持っていること、また SQL と solidDB に精通していることを前提としています。

図に関する規則

本書では、サーバーの図をいくつか使用して、HotStandby 環境におけるいろいろなシナリオについて説明します。

以下の表に、サーバーの図の凡例を示します。

表 1. HotStandby の図に関する規則

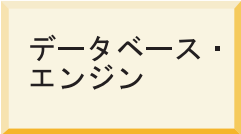


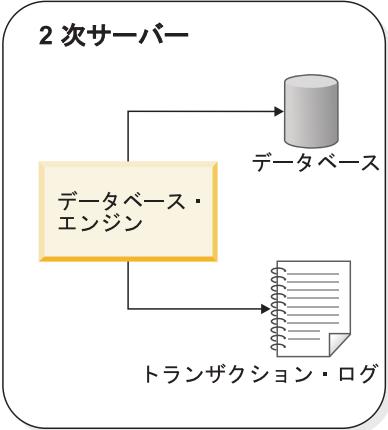

シンボル	説明
 データベース・エンジン	長方形は、実行元のプログラム、つまりデータベース・サーバー (エンジン) 自体を表します。
 データベース	シリンダーは、データを表し、これは一般にはディスクに格納されています。あるいは、一部または全部がメモリーに格納されている場合もあります。
 トランザクション・ログ	このシンボルは、データベース・リカバリーと HotStandby の両方で使用されるトランザクション・ログ (Txn ログ) を表します。

表 1. HotStandby の図に関する規則 (続き)

シンボル	説明
 <p>2次サーバー</p> <p>データベース・エンジン</p> <p>データベース</p> <p>トランザクション・ログ</p>	<p>角丸長方形は、データとトランザクション・ログを伴う完全なサーバーを表します。角丸長方形の内部に 2次サーバーまたは 1次サーバーという語句がある場合、そのサーバーは HotStandby サーバーです。</p>
 <p>1次サーバー</p> <p>データベース・エンジン</p> <p>トランザクション・ログ</p> <p>2次サーバー</p>	<p>簡略化のために、データベース内のデータを表すシンダーが省略される場合もあります。場合によっては、シンボルがさらに簡略化されて角丸長方形のみを示す場合もあります。どちらのアイコンも、サーバーの簡略化された表現です。</p>

書体の規則

solidDB の資料では、以下の書体の規則を使用します。

表 2. 書体の規則

フォーマット	用途
データベース表	このフォントは、すべての通常テキストに使用します。
NOT NULL	このフォントの大文字は、SQL キーワードおよびマクロ名を示しています。
solid.ini	これらのフォントは、ファイル名とパス式を表しています。
SET SYNC MASTER YES; COMMIT WORK;	このフォントは、プログラム・コードとプログラム出力に使用します。SQL ステートメントの例にも、このフォントを使用します。
run.sh	このフォントは、サンプル・コマンド行に使用します。
TRIG_COUNT()	このフォントは、関数名に使用します。
java.sql.Connection	このフォントは、インターフェース名に使用します。
LockHashSize	このフォントは、パラメーター名、関数引数、および Windows® レジストリー項目に使用します。
argument	このように強調されたワードは、ユーザーまたはアプリケーションが指定すべき情報を示しています。

表 2. 書体の規則 (続き)

フォーマット	用途
管理者ガイド	このスタイルは、他の資料、または同じ資料内の他の章の参照に使用します。新しい用語や強調事項もこのように記述します。
ファイル・パス表示	ファイル・パスは、UNIX [®] フォーマットで示します。スラッシュ (/) 文字は、インストール・ルート・ディレクトリーを表します。
オペレーティング・システム	資料にオペレーティング・システムによる違いがある場合は、最初に UNIX フォーマットで記載します。UNIX フォーマットに続いて、小括弧内に Microsoft [®] Windows フォーマットで記載します。その他のオペレーティング・システムについては、別途記載します。異なるオペレーティング・システムに対して、別の章を設ける場合があります。

構文表記法の規則

solidDB の資料では、以下の構文表記法の規則を使用します。

表 3. 構文表記法の規則

フォーマット	用途
INSERT INTO <i>table_name</i>	構文の記述には、このフォントを使用します。置き換え可能セクションには、このフォントを使用します。
solid.ini	このフォントは、ファイル名とパス式を表しています。
[]	大括弧は、オプション項目を示します。太字テキストの場合には、大括弧は構文に組み込む必要があります。
	垂直バーは、構文行で、互いに排他的な選択項目を分離します。
{ }	中括弧は、構文行で互いに排他的な選択項目を区切ります。太字テキストの場合には、中括弧は構文に組み込む必要があります。
...	省略符号は、引数が複数回繰り返し可能なことを示します。
⋮	3 つのドットの列は、直前のコード行が継続することを示します。

1 IBM solidDB HotStandby の紹介

この章では、IBM® solidDB HotStandby のコンポーネントについて説明します。HotStandby を使用すると、2 次サーバー (ホット・スタンバイ・サーバー) を 1 次サーバーと並行して稼働させ、1 次サーバー内のデータの最新コピーを保持できます。

サーバーの状態は、ウォッチドッグ と呼ばれるエンティティによって制御されます。solidDB HotStandby コンポーネントには、solidDB 高可用性コントローラー (HAC) と呼ばれるウォッチドッグ・インプリメンテーションが含まれています。

内部では、HAC は HotStandby のコマンドを使用してサーバーの状態を制御します。このようなソリューションでは、より信頼性の高いシステムのインプリメンテーションが可能です。データベース・サーバーに障害が起きても、サイトが完全に停止することはなくなります。HotStandby を使用すると、solidDB でサポートされているすべてのエンジン構成 (solidDB マスターまたはレプリカなど) では、わずか数百ミリ秒のうちに、障害を起こしたデータベースを 2 次データベースに置き換えることができます。

注:

「ホット・スタンバイ (hot standby)」という用語 (すべて英小文字の 2 ワード) は、最初のサーバーが障害を起こしたときに、2 番目のサーバーがいつでもその機能を引き継げるようにしておく一般的な技法を指しています。「HotStandby」(このように英大文字を使用した 1 ワード) は、この一般的な技法の solidDB 固有のインプリメンテーションを指しています。HotStandby の省略形は、HSB です。

同様に、ウォッチドッグは 2 つのデータベースの状態を監視し、必要な場合に状態を切り替えることができるテクノロジーを指しています。HAC は、solidDB のウォッチドッグ・インプリメンテーションです。付録の 1 つに提示されている Watchdog サンプルは、HSB API を使用して独自のウォッチドッグ・アプリケーション・プログラムを作成する方法を示すプログラミング例です。

以下に、HotStandby アーキテクチャーの概念図を示します。

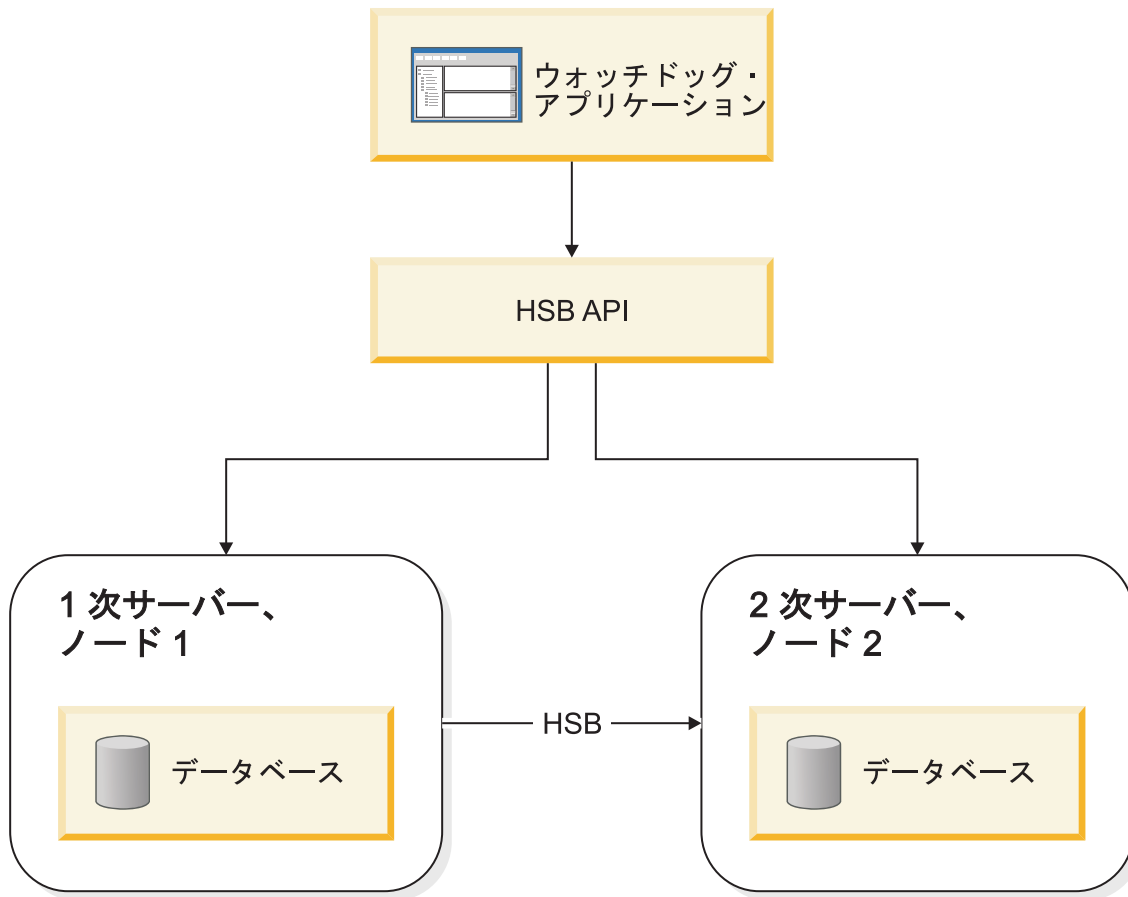


図 1. HotStandby アーキテクチャー

上の図に示すように、HotStandby は、以下のものを提供します。

- solidDB 高可用性コントローラー (HAC) と呼ばれるウォッチドッグ・アプリケーション。
- HotStandby API (HSB 管理コマンド)
- solidDB サーバー

HotStandby の動作の仕組み

HotStandby (HSB) は、1 次サーバーと 2 次サーバーという 2 つのノード間で、同期トランザクション・レプリケーションを行います。

1 次サーバー・ノード (1 次) には、アクティブ・データベースが入っています。2 次サーバー・ノード (2 次) には、アクティブ・データベースの正確な最新コピーが入っており、このノードは 1 次サーバーに障害が起きた場合、1 次サーバーに取って代わることができます。

2 次サーバーは 1 次サーバーから更新情報を受け取り、元の 1 次サーバーが障害を起こした場合は、いつでも 1 次サーバーとして後を引き継ぐことができます。それ以外の、2 次サーバーを使用することの利点は、クライアント

からの読み取り専用の要求 (例えば、SELECT ステートメント) に 2 次サーバーも応答できることです。これにより、ワークロードの一部を 1 台でなく 2 台のサーバーに分散できます。

HotStandby API (HSB 管理コマンド)

HotStandby サーバーの高可用性動作は、solidDB 管理コマンドのサブセットに基づいた API によって制御されます。

このコマンドのサブセットは、`hotstandby` (`hsb` と省略できます) がメイン・コマンドであることによって識別されます。これらのコマンドは、(`sql` のような) 任意の SQL 対応ツールか、ODBC や JDBC などのプログラマチック・インターフェースを使用して発行できます。HotStandby 管理コマンドの構文は、以下のとおりです。

```
admin command 'hotstandby hsb-command options';
```

または

```
admin command 'hsb hsb-command options';
```

後で、`hsb` 管理コマンドを使用して solidDB HSB サーバーの高可用性状態を制御する方法や、状態情報をリトリートする方法について説明します。これらのコマンドを手動で、またはプログラムで発行できることに注意してください。solidDB HA 管理ツールの HA コントローラーおよびウォッチドッグは、これらのコマンドをプログラムで使用します。

基本的な HotStandby サーバー・スキーム

基本的な HotStandby サーバー・スキームでは、2 つのデータベース・サーバー、つまり 1 次サーバー と 2 次サーバー が存在します。それぞれが独自のディスク・ドライブを持ち、データベースをそこに保管し、それぞれが独自のトランザクション・ログ (*Txn* ログ) を持ちます。

4 ページの図 2 は、基本的な HotStandby サーバー・スキームを示した図です。1 次サーバーは、それ自体のトランザクション・ログに書き込みを行い、そのログを 2 次サーバーへ転送して、2 次サーバーがそれ自体のデータベース・コピーに同じ変更を加えることができるようにします。2 次サーバーのトランザクション・ログは HSB にアクティブには関与しませんが、コミットされた後にメイン・データ表にまだ書き込まれていないデータを 2 次サーバーがリカバリーできるよう、保守されます。(ロギングとリカバリーの詳細については、「*solidDB* 管理者ガイド」を参照してください。)

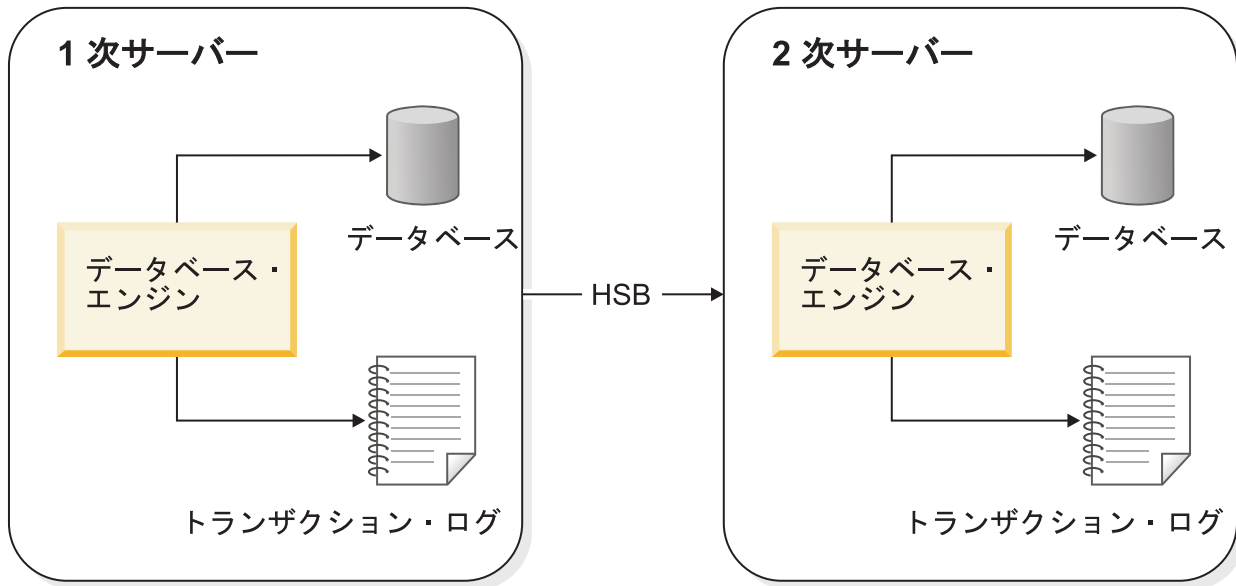


図 2. HotStandby サーバー・スキーム

ハートビート

solidDB HSB の内部では、ハートビート と呼ばれる技法を使用してサーバー間の接続がモニターされます。

アクティブ・サーバーとスタンバイ・サーバーの間では、キープアライブ・メッセージのシーケンスが送信されます。両方のサーバーは、自己の存在を示すそれらの単方向のメッセージを、絶えず相手側のサーバーへ送信します。メッセージは固定された時間間隔で送信されます。これに対し、相手側からのメッセージは、事前定義された時間枠内に到着することが予期されています。solidDB では、ハートビートの技法は *ping* と呼ばれます。

重要: solidDB では、ハートビートの技法は *ping* と呼ばれますが、*ping* 要求は送信されません。TCP/IP ネットワークで使用される Ping プロトコルと混同しないようにしてください。

トランザクション・ログと HotStandby

HotStandby は、1次サーバーのトランザクション・ログを使用します。このログには、サーバー上でコミットされたトランザクションのコピーが入っています。非 HotStandby サーバーでは、このトランザクション・ログはサーバーが異常なシャットダウンを起こした場合に、データをリカバリーするために使用されます。

HotStandby 1次サーバーでは、更新する必要があるデータを 2次サーバーに知らせるために、ログ・データが 2次サーバーへも送信されます。2次側のデータベースは絶えずロールフォワード・プロセスを実行し、このプロセスにより、ログ・データを受信して 2次側のデータ・コピーと 1次側のコピーとの同期を保ちます。

1次サーバーに障害が起きると、ウォッチドッグ・アプリケーションは 2次サーバーに 1次サーバーになるよう指示します。新しい 1次サーバーが稼働すると、ク

クライアントはそれに接続して作業を続行できます。クライアントには、1 次サーバーがダウンする前にコミットしたすべてのデータが表示されます。(クライアントは、元の 1 次サーバーがダウンした時点で開始されていて終了していなかったトランザクションを再開する必要があります。)

フェイルオーバーと切り替えの際にクライアントが HSB 環境で動作できるようにするため、透過接続 (TC) と呼ばれる特殊なタイプのクライアント接続を使用できます。詳細については、59 ページの『4 章 アプリケーションでの HotStandby の使用』を参照してください。

2 次サーバーに障害が起きた場合は、1 次サーバーが動作を続行できます。1 次サーバーはトランザクション・ログへのデータの書き込みを続行し、そのトランザクション・ログを、1 次サーバーと 2 次サーバーが互いに再接続して 1 次サーバーが 2 次サーバーへログを送信し終わるまで保持します。¹

障害を起こしたデータベース・サーバーが再び使用可能になった後、そのサーバーを新しい 2 次データベース・サーバーになるように構成できます (障害を起こしていないサーバーは、当然ながら、既に現行の 1 次サーバーとして機能しています)。

1 次サーバーが障害を起こしているサーバーならば、各サーバーはそれぞれの責任を逆にして、元の 2 次サーバーが 1 次サーバーとして動作を引き継ぎ、元の 1 次サーバーは、修復された後に新しい 2 次サーバーとしてシステムに復帰します。これらの逆転は、障害が起きるたびに発生する可能性があります。どちらのサーバーも 1 次側になりえることから、システムは長期にわたって複数の障害に耐えて存続でき、事実上無期限に動作を続行できます。

注意:

1 次サーバーが 2 次サーバーと長い期間、接続できなくなった場合は、トランザクション・ログが、使用可能なすべてのディスク・スペースを満たしてしまう可能性があります。これは、構成パラメーターの適切な設定によって回避できます。39 ページの『トランザクション・ログのスペース不足』を参照してください。

HSB は、ハードウェアおよびソフトウェアのアップグレード時に、ダウン時間の削減のために使用することさえできます。一方のサーバーを 1 次サーバーとして動作させたままにしておき、その間にもう一方をアップグレードします。

HotStandby は、速度と安全のバランスをカスタマイズするために役立てることもできます。HSB パラメーター **SafenessLevel** および **2SafeAckPolicy** は、2 次サーバ

1.

1 次サーバーがログを保持する時間の正確な長さは、solid.ini 構成パラメーター **CheckpointDeleteLog** および **BackupDeleteLog** の設定によって異なります。

1. **CheckpointDeleteLog=Y** の場合、1 次サーバーは、2 次サーバーがダウンした時刻と最新のチェックポイントのどちらか古い方の時点以降、すべてのトランザクション・ログを保持します。(チェックポイントの詳しい説明については、「*solidDB* 管理者ガイド」を参照してください。
2. **CheckpointDeleteLog=N** で **BackupDeleteLog=Y** の場合、1 次サーバーは 2 次サーバーがダウンした時刻と最新のバックアップのどちらか古い方の時点以降、すべてのトランザクション・ログを保持します。
3. **CheckpointDeleteLog=N** で **BackupDeleteLog=N** の場合、サーバーはログを無期限に保持します。

ーによるトランザクションの確認応答の方法を制御します。このパラメーターをロギングに関連した **DurabilityLevel** パラメーターと組み合わせることにより、速度と安全の組み合わせを指定できます。いくつかのパラメーター設定は、実際に非 HSB サーバー上のパフォーマンスを高めます。(詳細については、51 ページの『パフォーマンス・チューニング』で持続性レベルと安全性のパラメーターについての説明を参照してください。)

その一方で、**SafenessLevel** パラメーターの auto 値を使用することにより、持続性レベルとの関連で安全性レベルを動的に変更することもできます。

サーバー HotStandby 状態

HotStandby システムでは、各サーバーは可能な複数の状態のうち、そのサーバーの現行動作を説明しているいずれかの状態に置かれます。

例えば、1 次サーバーと 2 次サーバーが通信して同期を保っている場合、それらはそれぞれ、PRIMARY ACTIVE 状態と SECONDARY ACTIVE 状態にあります。別の例として、1 次サーバーと 2 次サーバーとの接続が失われた場合、1 次サーバーは自動的に PRIMARY UNCERTAIN 状態に切り替わります。その状態では、新しいトランザクションを受け入れません。ユーザーは (または、より一般的には HAC は) サーバーを、独立したサーバーとして動作する PRIMARY ALONE 状態に切り替えます。サーバーは新しいトランザクションを受け入れ、後で 2 次サーバーへ送信するために保管します。

HotStandby と拡張レプリケーションの結合

solidDB HotStandby コンポーネントは、solidDB 拡張レプリケーションと組み合わせて使用できます。拡張レプリケーションは、双方向の定期的な発生するデータ同期機能を提供し、これによりユーザーは、マスター・サーバーとレプリカ・サーバーが含まれている分散システムを作成できます。HotStandby を使用すると、分散システムのあらゆるデータベース・サーバーの可用性を高めることができます。

7 ページの図 3 は、1 つのマスター・データベースと 2 つのレプリカ・データベースを含んでいる単純な分散システムを示しています。各レプリカには、少なくともマスター・データベースのデータのサブセットが格納されています。それぞれのデータベース・サーバーは、HotStandby レプリケーションによってフォールト・トレラントになっています。拡張レプリケーションは、データベース・サーバー階層の 1 次サーバー間で発生します。いずれかの 1 次データベース・サーバーで問題が起きた場合、障害を起こしたノードは HotStandby フェイルオーバーを実行して、そのノードの 2 次サーバーを新しい 1 次サーバーにすることができます。これにより、新しい 1 次サーバーを使用して拡張レプリケーションを続行できるようになります。

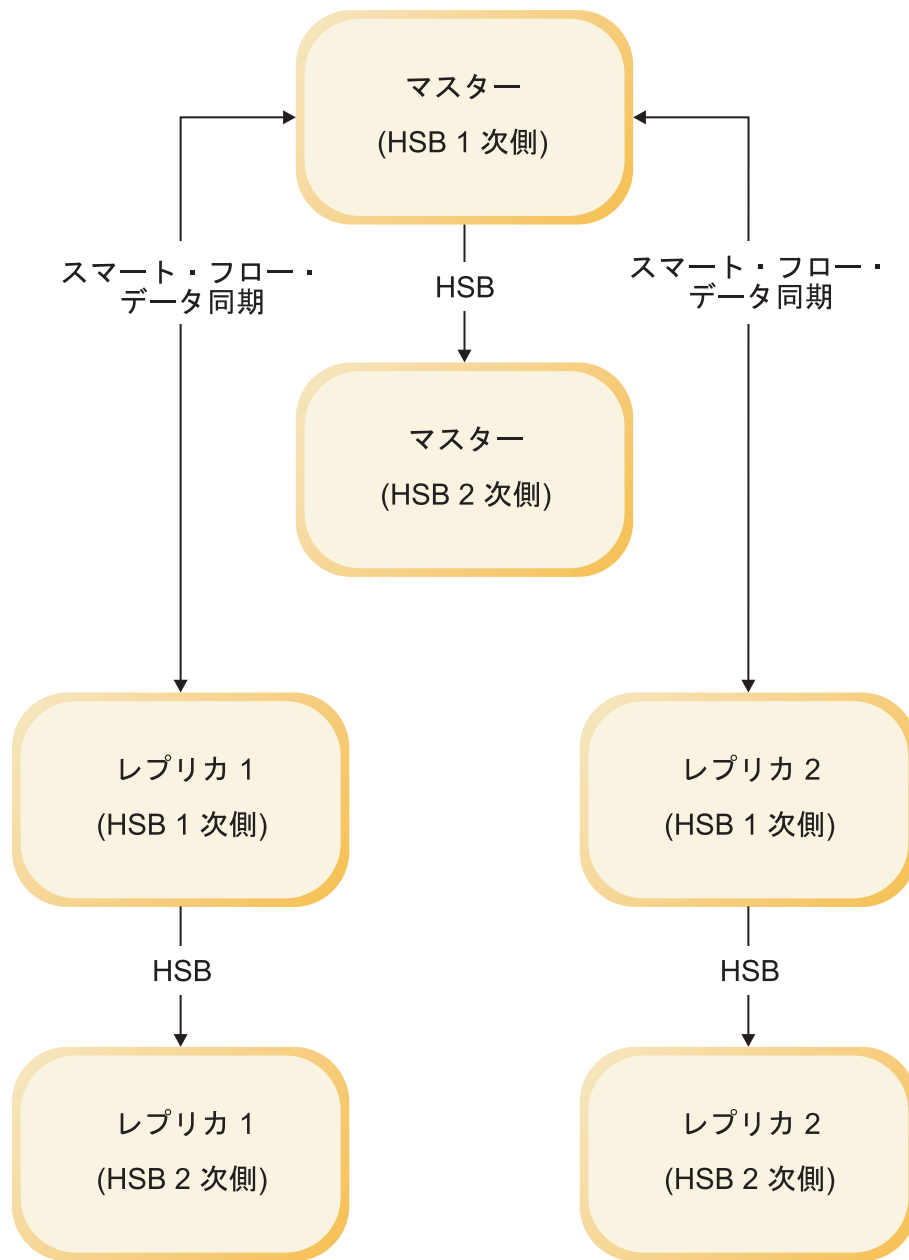


図3. マスターおよびレプリカ・サーバー・スキームでの HotStandby

フェイルオーバー

フェイルオーバーでは、2次サーバーが新しい1次サーバーに切り替えられます。

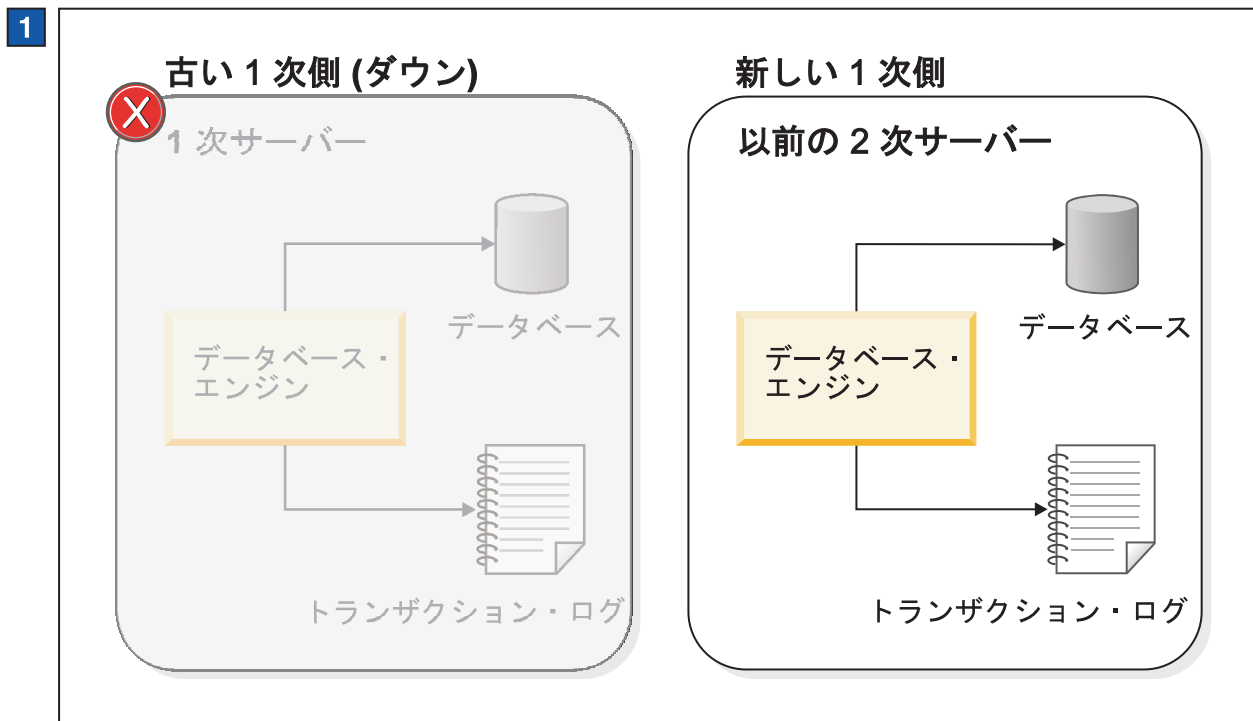
2次サーバーを新しい1次サーバーに切り替えるのには、いくつかの理由があります。

1. 1次側に障害が起きたとき
2. 1次側を管理したいとき
3. システム上に既存の1次側が存在しないときに1次側を選択する必要があるとき

2 次サーバーを新しい 1 次サーバーになるよう切り替えるには、2 次サーバー上で次のコマンドを発行します。

```
ADMIN COMMAND 'hotstandby set primary alone';
```

フェイルオーバーの場合、新しい 1 次側は、古い 1 次側データベースからの最新のコミットされたデータを含んでいます。1 次側のデータベース内でコミットされたすべてのものは、2 次側のデータベースから検出できることが保証されます。透過接続 (TC) を使用している場合は、フェイルオーバー時でも接続が失われません。ただし、進行中のトランザクションは打ち切られるので、再実行する必要があります。詳細については、60 ページの『TC での障害透過性』を参照してください。新しい 1 次側は単独で動作でき、引き続きトランザクションとデータを自身のデータベースとトランザクション・ログに書き込みます。



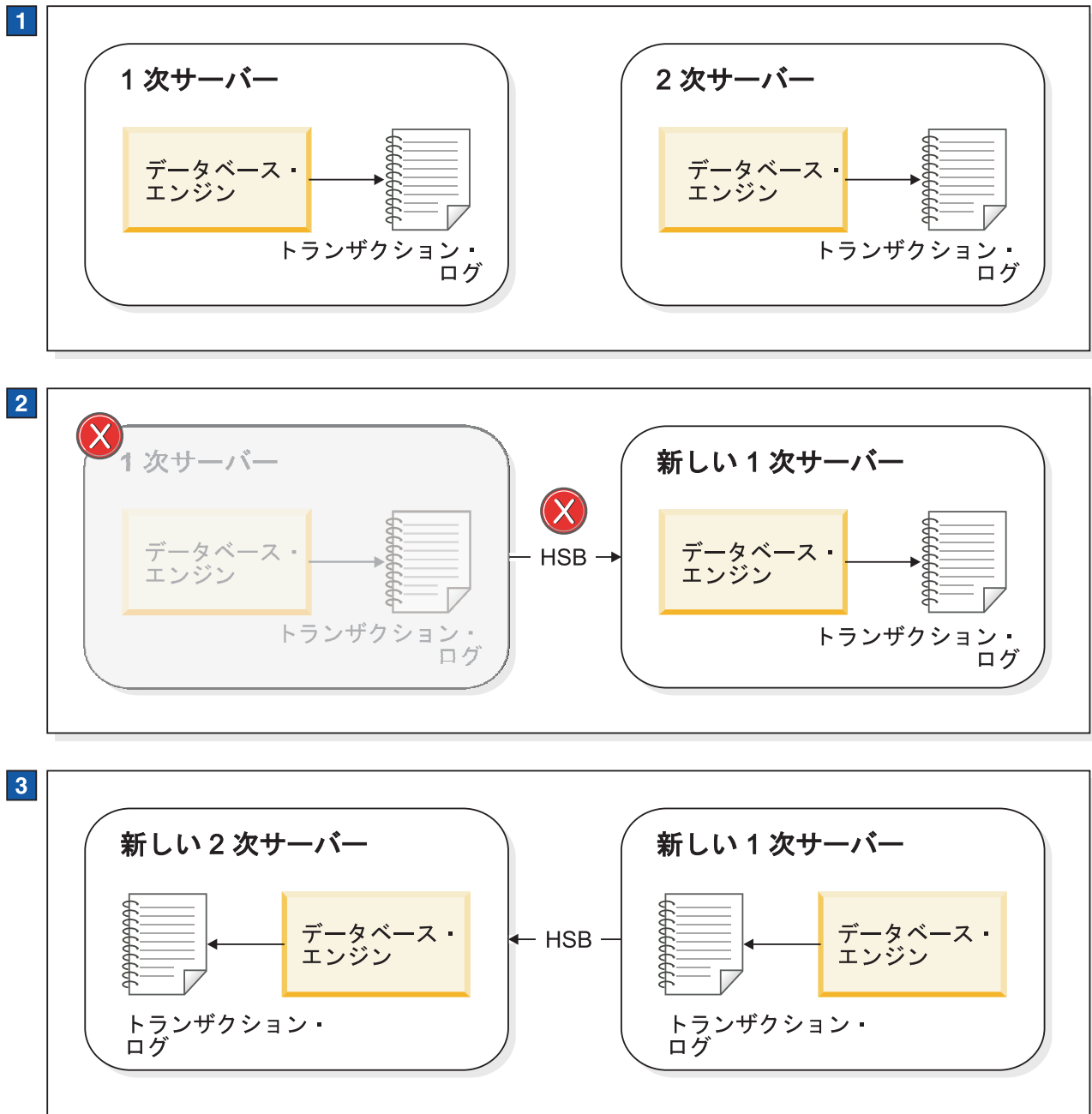
1. 当初 2 次側だったサーバーは、古い 1 次サーバーに障害が起きると新しい 1 次サーバーになります。

図 4. 新しい 1 次側 (古い 2 次側)への HotStandby 切り替え

サーバー・キャッチアップ

古い 1 次側は、オンラインに戻った後、(既存の 1 次側が存在する場合は) 新しい 2 次側になります。この段階で、新しい 2 次側内の情報は新しい 1 次側の情報より後れています。新しい 1 次側のデータベースには新しいトランザクションがコミットされているからです。新しい 2 次側を最新の状態にするために、サーバー同士が接続された後に、新しい 1 次側のトランザクション・ログ・データが自動的に新しい 2 次側へ送信されます。保留中のすべての変更がトランザクション・ログから

新しい 2 次側へ書き込まれるため、2 次側は 1 次側との同期を保持できます。サーバー・キャッチアップの図を図 5 に示します。



1. 通常の動作: 1 次サーバーが 2 次サーバーにデータを送信します。
2. 1 次サーバーで障害が発生すると、2 次サーバーが新しい 1 次サーバーとして後を引き継ぎます。新しい 1 次サーバーは、後で新しい 2 次サーバーにデータを送信できるように、そのトランザクション・ログにトランザクション情報を保存します。
3. 古い 1 次サーバーが新しい 2 次サーバーとして立ち上げられると、新しい 2 次側がキャッチアップできるように、新しい 1 次側のトランザクション・ログが 2 次側に送信されます。

図 5. サーバーのフェイルオーバーとキャッチアップの例

HotStandby でのレプリケーション・モード

1-Safe および 2-Safe レプリケーション

solidDB には、必要なパフォーマンスと耐久性のバランスを取るために、システムをチューニングするさまざまな選択肢があります。そのような選択肢の 1 つは、使用するレプリケーション・プロトコルの選択です。 **SafenessLevel** と呼ばれるシステム・パラメーターは、レプリケーション・プロトコルが同期式 (2-safe) か非同期式 (1-safe) かを決定します。

- 1-safe: トランザクションは、最初に 1 次側でコミットされてから、2 次側へコミットされます。
- 2-safe: トランザクションは、2 次側による確認応答が完了するまでコミットされません (デフォルト)。

この安全性レベルは、グローバル (サーバー)、セッション、およびトランザクションの 3 つのレベルで制御できます。

2-safe レプリケーションでの同期 HotStandby

1 次側と 2 次側が正確に同じデータを持つようにするために、solidDB は主に同期 HotStandby モデルを使用します。これは、「2-safe」レプリケーション方式と呼ばれ、データが 2 つの場所へ書き込まれた後に、データがコミットされたことがユーザーに通知されます。

したがって、1 次サーバーは、データが 1 次側に書き込まれた時点ではトランザクションがコミットされたことをユーザーに通知せず、2 次サーバーもデータをコミットする (または、少なくとも受信する) まで待ってから、初めてデータがコミットされたことをユーザーに通知します。²

1 次サーバーは、トランザクションに対する変更を 1 次側のデータベース内でコミットする前に、トランザクション・データを 2 次サーバーへ送信します。2 次サーバーは 1 次側に対し、データをコミットした (または少なくとも受信した) ことの確認応答を送信する必要があります。それが無い場合、1 次サーバーはタイムアウトになり、状態が PRIMARY ACTIVE から PRIMARY UNCERTAIN に変化します (これらの状態については、詳しく後述します)。その場合、1 次サーバーは、ロールバックもトランザクションのコミットもできません。HAC は 1 次サーバーを PRIMARY ALONE 状態に設定する場合があります、それによって 1 次側は、トランザクションの受信と、2 次側に依存しない動作を続行できます。1 次側は 2 次側へ送信した保留中のトランザクションをコミットし、新しいトランザクションの受け入れを再開します。

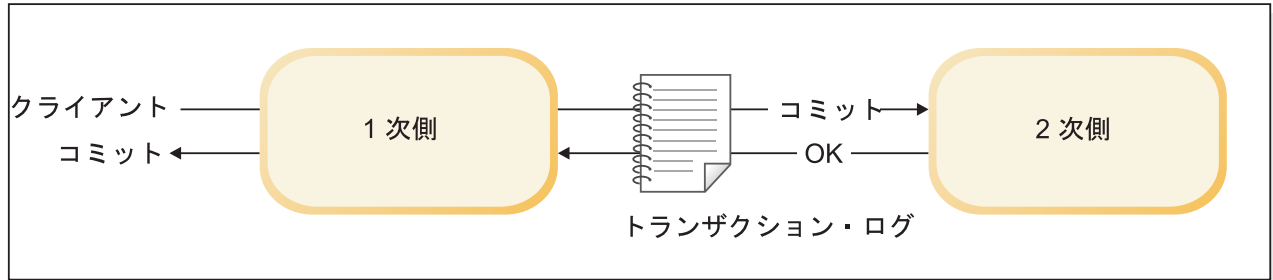
注:

2 次サーバーはトランザクション・ログ項目のコミット (または少なくとも受信) を完了すると同時に確認応答を送信します。この構成により、単一の障害が起きたときでも、トランザクションが失われずに済みます。さらに、システム全体の障害が

2. 2-safe と 1-safe のアルゴリズムの詳細については、「*Transaction Processing: Concepts and Techniques*」(Jim Gray および Andreas Reuter 共著、Morgan Kaufman, 1993) を参照してください。

発生した場合のデータベースのリカバリーを容易にするために、オプションとしてファイル・ベースのトランザクション・ログが保存されます。

1 次側と 2 次側の両方が稼働



1 次側が稼働、2 次側がダウン

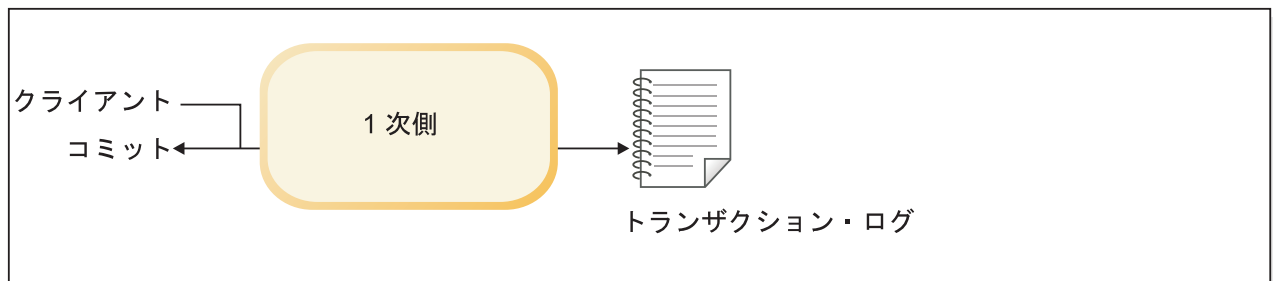


図 6. 同期 HotStandby 構成

データの送信での基本ステップ

以下に、同期レプリケーションでデータを送信する際のステップを示します。

- 1 次サーバーは、1 次側ノードのトランザクション・ログにデータを (レコード・レベルの形式で) 書き込みます。
- 1 次サーバーがコミット・ステートメントを検出すると、変更されたすべてのデータが 2 次サーバーへ送信されます。

注: トランザクションの開始後、しかも 1 次サーバーがデータを送信する前に 2 次サーバーが障害を起こした場合、1 次側はトランザクションをロールバックします。

- 2 次側は、コミット・メッセージの確認応答を行います。確認応答のタイミングは、**2SafeAckPolicy** 構成パラメーターの設定によって異なります。*2-safe received* という最高速の選択肢では、2 次側はコミット・メッセージの受信直後に 1 次側に確認応答を送信します。*2-safe durable* と呼ばれる最も安全な選択肢では、2 次側はトランザクションを実行して 2 次側自体のトランザクション・ログに永続的に書き込んだ後に、確認応答を送信します。

1 次側は、2 次側の確認応答を受信すると、ユーザーにデータがコミットされたことを通知します。

- 1 次側が 2 次側からの確認応答を (例えば、ネットワークやノードの障害のために) 受信できない場合、1 次側はタイムアウトになり、PRIMARY UNCERTAIN 状態に切り替わります。1 次側は、2 次側での最新のトランザクシ

ョンの状態が分からないので、トランザクション自体のロールバックやコミットができません。1次側には、以下のどれが起きたかが分かりません。

- トランザクションがコミットされる前に2次側がダウンした。
- 2次側は、既にトランザクションをコミットしたが、1次サーバーは例えばネットワーク障害のために、確認応答を受信しなかった。

サーバーが PRIMARY UNCERTAIN 状態にある間、ユーザーがコミットを試みる現行トランザクションと新規トランザクションはブロックされ、ユーザーにはサーバーが無応答のように感じられる場合があります。

5. HAC は、2次側がダウンしていること、またはネットワークが障害を起こしたことを検出した場合、1次サーバーを PRIMARY ALONE 状態に切り替えることができます。1次サーバーは、PRIMARY ALONE に設定されるとすぐに、2次側へ送信した保留中のトランザクションをコミットし、新しいトランザクションの受け入れを再開します。
6. 変更は、2次サーバーが復帰するか1次サーバーのディスク・スペースがなくなるまで、トランザクション・ログ・ファイルに蓄積されます。(サーバーがトランザクション・ログのディスク・スペース不足になった場合、1次側は読み取り専用モードに変わります。)
7. 2次サーバーが長時間にわたって稼働せず、サーバーがトランザクション・ログのディスク・スペース不足になる可能性があると思われる場合は、1次サーバーを PRIMARY ALONE から STANDALONE の状態に切り替えるとよいでしょう。そのような場合には、トランザクション・ログは、2次側との接続が失われた時点以降のすべてのトランザクションを保管しなくなり、したがって、単に1次側からトランザクション・ログを読み取るだけでは2次側が「キャッチアップ」することができなくなります。トランザクション・ログによって2次側を最新の状態にできない場合、2次側と1次側の同期をとる唯一の方法は、1次側のデータベース・ファイルを2次側にコピーすることです。これは、HotStandby の copy コマンドまたは HotStandby の netcopy コマンドによって行うことができます。copy および netcopy の詳細については、91ページの『ネットワークを介した1次データベースの2次側へのコピー』および95ページの『1次サーバーから指定されたディレクトリーへのデータベース・ファイルのコピー』を参照してください。
8. copy または netcopy を実行するには、1次側が PRIMARY ALONE 状態でなければなりません。copy/netcopy の後、1次サーバーはコマンドが成功したか失敗したかに関係なく、PRIMARY ALONE 状態のままです。
9. 1次側が再び2次側へのトランザクションの送信を開始するためには、1次サーバーを2次サーバーへ明示的に接続するために、97ページの『HotStandby サーバーの接続』で説明する hotstandby connect コマンドを使用する必要があります。1次サーバーが2次サーバーに接続した後、1次サーバーは PRIMARY ACTIVE 状態で動作します。

サーバー同士が接続した後、キャッチアップ(1次サーバーとの同期をとるために、保留中のすべての変更がトランザクション・ログから2次サーバーへ自動的に書き込まれること)の実行が開始されます。サーバー・キャッチアップの前に、1次サーバーと2次サーバーは情報を交換し、トランザクションが2次サーバー上で2回コミットされないよう、キャッチアップの開始位置を決定します。

1-safe レプリケーションでの非同期 HotStandby

オプションとして、1 次側から 2 次側への非同期レプリケーションを使用できます。これは、*1-safe* レプリケーションと呼ばれます。

1-safe レプリケーションでは、トランザクションが 1 次側でコミットされると、直ちにその確認応答が行われます。これにより、パフォーマンスが大幅に向上します。コミットの後、トランザクションは 2 次側へ非同期方式で送信されます。そのトレードオフとして、1 次側で障害が発生すると、転送中だった少数のトランザクションが失われる可能性があります。

2 つのレプリケーション方式のどちらも、動的に選択でき、セッションごと、またはトランザクションごとに選択することさえできます。1-safe レプリケーションに関連するレプリケーション遅延も、制御することができます。

サーバー状態の説明

HotStandby (HSB) ペアの両方のサーバーは、照会と操作が可能な特定の状態を公開します。

状態とその意味は、以下のリストに示すとおりです。

表 4. サーバー状態の説明

状態	説明
PRIMARY ACTIVE	サーバーは接続され、このサーバー (1 次サーバー) は読み取り/書き込みトランザクションを受け入れており、データを 2 次サーバーへ送信しています。2 次サーバーは、SECONDARY ACTIVE 状態でなければなりません。
PRIMARY ALONE	ピア・サーバーが相互接続されていません。ピアは立ち上がっている可能性があります、接続されておらず、したがってトランザクションを受け入れません (つまり、SECONDARY ALONE 状態である可能性があります)。 このサーバー (1 次サーバー) は、読み取り/書き込みトランザクションをアクティブに受け入れて実行しており、後で 2 次サーバーへ送信するためにトランザクションを収集しています。

表 4. サーバー状態の説明 (続き)

状態	説明
PRIMARY UNCERTAIN	<p>サーバー同士の接続が異常切断されており、AutoPrimaryAlone 構成パラメーターが「No」に設定されています。PRIMARY UNCERTAIN 状態では、確認応答されていないすべてのトランザクションが保留状態のままです。つまり、サーバーは HAC がサーバーを別の状態に変更するまで、それらのトランザクションをコミットもロールバックもしません。</p> <p>オペレーターに可能なアクションは、3 つあります。1 次サーバーを 2 次サーバーに再接続するか、1 次サーバーを PRIMARY ALONE 状態に設定するか、1 次サーバーを SECONDARY ALONE 状態に設定するかです。</p> <ol style="list-style-type: none"> 1. サーバーを 2 次サーバーに再接続した場合、トランザクションは 1 次サーバー上でコミットされます。 2. 状態を PRIMARY ALONE に変更した場合、オープン・トランザクションは 1 次サーバー上でコミットされます。 3. 状態を SECONDARY ALONE に変更した場合、オープン・トランザクションは保留中のままになります。それらはサーバーが別の状態に変更された後に、最終的に解決されます。例えば、サーバーが SECONDARY ACTIVE 状態に移された場合、ブロックされていたトランザクションはキャッチアップの結果に応じて、打ち切られるかコミットされます。サーバー状態が STANDALONE または PRIMARY ALONE に変更された場合、ブロックされていたトランザクションはコミットされます。 <p>2 次サーバーとの接続を失った 1 次サーバーを、PRIMARY UNCERTAIN でなく PRIMARY ALONE に自動的に移行させたい場合は、AutoPrimaryAlone 構成パラメーターの説明を参照してください。</p> <p>注:</p> <p>HAC は、常にサーバーを PRIMARY UNCERTAIN から SECONDARY ALONE に切り替えることにより、最大の安全性を得ることができます。これにより、1 次サーバーの二重化が防止されます。ただし、ユーザーがサーバー上のデータを更新することもできなくなります。(38 ページの『ネットワーク分割と二重 1 次サーバー』を参照してください。)</p>

表 4. サーバー状態の説明 (続き)

状態	説明
SECONDARY ACTIVE	ピア・サーバーは相互に接続されており、このサーバーは 1 次側からの着信トランザクション・ログ・データを受け入れています。それらのトランザクションは、2 次側が 1 次側と同じデータを持つよう、2 次側で実行されます (2 次側に障害が起きた場合でも、2 次側自体でデータをリカバリーできるよう、トランザクションも 2 次側自体のトランザクション・ログに書き込まれます)。さらに、クライアントは SECONDARY ACTIVE 状態のサーバー上で、読み取り専用トランザクションを実行できます。サーバーが SECONDARY ACTIVE 状態にある場合は、そのサーバーのピアは PRIMARY ACTIVE 状態にあることが必要です。
SECONDARY ALONE	2 次サーバーは、ピア・サーバーから切断されています。読み取り要求だけが受け入れられます。2 次サーバーか 1 次サーバー上で「HotStandby connect」コマンドを発行すると、サーバーをピアに接続できる場合があります。
STANDALONE	サーバーは HSB 状態 (1 次サーバーまたは 2 次サーバー) を持たず、通常のスタンドアロン・サーバーとして動作します。トランザクション・ログも通常の方法で処理および除去され、2 次サーバー用に保存されません。HSB 動作を再開するためには、サーバーを PRIMARY ALONE または SECONDARY ALONE に設定する必要があります。1 次サーバーは netcopy または copy 動作を実行して、2 次サーバーにデータベースの完全なコピーを送信する必要があります。
OFFLINE	サーバーは「netcopy listen モード」(「backupserver モード」とも呼ばれます) で始動しました。このモードでは、サーバーは PRIMARY ALONE 状態にあるサーバーからの着信 netcopy を待っています。サーバーは、netcopy を正常に完了すると、SECONDARY ALONE 状態に移行します。 OFFLINE 状態を直接監視することはできません。OFFLINE 状態のサーバーはクライアント接続を受け入れないからです。OFFLINE 状態のサーバーに接続しようとすると、エラー・コード 14552、「Server is in backup server mode, no connections are allowed」が返されます。OFFLINE 状態のサーバーは、netcopy 操作 (後述します) に対してのみ応答します。

HotStandby によるパフォーマンスへの影響

HotStandby (HSB) によってパフォーマンスが低下すると予想する人もいるかもしれませんが、常にそうなるとは限りません。事実、一部の構成では、HSB によってパフォーマンスが大幅に向上することさえあります。

HSB パフォーマンスの主要な要因は、以下のとおりです。

- 単一の障害でトランザクションを保存する必要がある場合の、アダプティブ持続性の使用
- 障害での小規模なトランザクションの損失を受容できる場合の、1-Safe レプリケーション・プロトコルの使用
- 2-Safe 確認応答ポリシー (2-safe レプリケーションが使用されます)、つまり、2次サーバーは1次サーバーからトランザクションを受信すると同時に受信の確認応答を行うのか、それともトランザクションをコミットし終わるまで待つのか。
- 2次サーバー上で読み取り専用トランザクションを実行する可能性
- サーバーの内部並列処理

要約すると、可能な最高の安全性レベルを必要とする場合を除いて、以下を行うことにより、パフォーマンスを高めることができます。

- アダプティブ・ロギングを使用する (**DurabilityLevel=2** を設定する)
- 「2-safe received」モードを使用する (**2SafeAckPolicy=1** を設定する)

読み取り専用の照会 (例えば要約報告書を生成する照会) がある場合は、それらを2次サーバーで実行して、ワークロードを両方のマシンに分散するとよいでしょう。

ここで示した「安全性が低い」方の設定 (アダプティブ持続性および 2-safe received モード) を使用した場合でも、2つ以上の障害がなければ、HSB によって保護されることに注意してください。安全性をほとんど犠牲にすることなく、非常に高いパフォーマンスが得られます。solidDB HotStandby は、高いパフォーマンスと安全性を同時に提供します。

当然ながら、実際のスループットと応答時間は多数の要因に依存しています。例えば、それらの要因には、ネットワークの速度、ネットワーク上の他のトラフィックの量、SQL ステートメントの複雑さ、および1トランザクション当たりのSQL ステートメント数が含まれますが、それだけに限りません。例えば、メモリー量とディスク速度なども当然、パフォーマンスに影響します。

アダプティブ持続性

solid.ini 構成ファイルで、リラックス持続性 (高速) が必要か、それともストリクト持続性 (安全) が必要か、あるいは、「アダプティブ持続性」という第3のオプションが必要かを指定できます。

- **ストリクト持続性:** トランザクションがコミットされると同時に、そのトランザクションがトランザクション・ログに書き込まれる場合、それを「ストリクト持続性」と呼びます。このタイプの持続性は、安全性を最大にします。
- **リラックス持続性:** トランザクションの書き込みを、サーバーがビジーでなくなるまで、または複数のトランザクションをまとめて書き込めるまで遅延できる場合、それを「リラックス持続性」(または「リラックス・ロギング」)と呼びます。HSB ペアに含まれていないサーバーでは、リラックス持続性を使用することは、サーバーが異常終了した場合、少数の最新のトランザクションを失う危険を冒すことを意味します。しかし、サーバーが HSB ペアに含まれている場合は、トランザクションのコピーがもう一方のサーバー (2次サーバー) にあり、しかも、1次サーバーがトランザクションをログに記録する前に障害を起こしたとしても、トランザクションは失われません。このため、リラックス持続性を HSB と一緒に使用した場合、リラックス持続性が原因で安全性が低下することは、まずあり

ません。逆に、リラックス持続性は、特にサーバーの負荷が多数の小さな書き込みトランザクションからなっている状況では、システムのパフォーマンスを向上させます。

- **アダプティブ持続性:** アダプティブ持続性は、HotStandby 1 次サーバーにのみ適用されます (また、これがデフォルトです)。アダプティブ持続性とは、単にサーバーが Primary Active 状態 (2 次サーバーへトランザクションを送信中) の場合はリラックス持続性を使用し、それ以外の状態ではストリクト持続性を使用することを意味します。これにより、HSB がアクティブのときは高いパフォーマンスを (安全性をほとんど失うことなく) 得ることができ、一方のサーバーのみが動作する場合でも、高い安全性が維持されます。アダプティブ持続性は、2-Safe レプリケーション (デフォルト) が使用されているときにのみ、効果的に機能を発揮できます。

アダプティブ持続性は、パフォーマンスを大幅に向上させる一方、障害のある状態でも高いデータの安全性を保証します。これはシステム全体のスループットを増し、待ち時間 (つまり、トランザクションがコミットされたことを通知されるまでユーザーが待たなければならない時間) を減らすことができます。

リラックス・ロギングと **DurabilityLevel** パラメーターの詳細については、「*solidDB 管理者ガイド*」を参照してください。

1-Safe レプリケーション

1-safe レプリケーションでは、1 次サーバーでコミット処理が完了した直後に、コミット・ステートメントに対する確認応答が行われます。コミットされたトランザクションは、アプリケーションへ制御が戻された後に、2 次サーバーへ非同期に伝送されます。

トランザクションの伝送にかかわる遅延は、数ミリ秒から数百ミリ秒の範囲です。1-safe レプリケーションを使用すると 1 次サーバーでの待ち時間が著しく減少するため、パフォーマンスが大幅に向上します。1-safe の不利な面は、障害が起きたとき、フェイルオーバーの際に少数のトランザクションが失われる可能性があることです。

1-safe レプリケーションは、サーバーに対してパラメーターと一緒に設定できません。

```
[HotStandby]
SafenessLevel=1safe ;values: 1safe, 2safe, auto; default is 2safe
```

安全性のレベルは、SET コマンドで動的に制御することもできます。

```
SET SAFENESS {1SAFE| 2SAFE| DEFAULT}
```

これは、現行セッションの安全性レベルを、変更されるまでの間、設定します。

```
SET TRANSACTION SAFENESS {1SAFE| 2SAFE| DEFAULT}
```

これは、現行トランザクションの安全性レベルを設定します。コミットの後、安全性レベルはセッションについて設定された値か開始値、またはシステム・デフォルト (これは 2-safe です) に戻ります。

オプション DEFAULT は、セッションの現行の設定を意味します。

SafenessLevel パラメーターが特殊値の「auto」（つまり「自動」）を持っているときは、安全性レベルをプログラムでの持続性の制御 (SET DURABILITY RELAXED など) によって制御することもできます。その場合、「strict」は「2-safe」に、「relaxed」は「1-safe」に対応します。

2-Safe 確認応答ポリシー

2-safe レプリケーションが有効 (デフォルト) の場合、1 次サーバーは 2 次サーバーからトランザクションを取得したことの確認応答を受信するまで、トランザクションが正常にコミットされたことをクライアントに通知しません。

solidDB では現在、次の 3 つの確認応答ポリシーを使用できます。

- 2-safe received: 2 次サーバーは、データを受信したときに確認応答を送信します (デフォルト)。
- 2-safe visible: 2 次サーバーはデータ・コピーの更新を完了し、変更が「可視」になりました。言い換えると、2 次サーバーに接続しているクライアント・アプリケーションは、更新を認識することができます。
- 2-safe durable: 2 次サーバーはデータを永続的にした時点で、つまり、データのコミットとディスクへのデータの書き込みを完了した時点で、確認応答を行います。

2-safe received は、より高速です。2-safe durable は、より安全です。これらの確認応答ポリシーは 1 次と 2 次のサーバーが両方ともアクティブである (つまり、両方ともトランザクションを適用している) ときにのみ適用されるので、2-safe received でも安全であると考えられることに注意してください。両方のサーバーが事実上同時に (互いに 1 秒以内に) 障害を起こした場合にのみ、トランザクションを失う危険があります。

2-safe received を使用すると、待ち時間が少なくなります。この待ち時間とは、コミットが開始されてから、ユーザーがコミットの確認を受け取るまでの時間です。**2SafeAckPolicy** は、全体のスループットにほとんど影響しません。

内部並列処理

システムで並列処理の利点を活用したい場合は、すべてのトランザクションを同じ接続によってサブミットするのではなく、複数の接続にトランザクションを分散することを考慮してください。

HotStandby (HSB) コンポーネントを使用した場合、書き込み操作を含んでいるすべてのトランザクションは、2 回 (1 回は 1 次サーバー上で、もう 1 回は 2 次サーバー上で) 実行されます。このため、状況によっては単一のトランザクションが、HSB を使用した場合に、HSB を使用しない場合の約 2 倍の時間を要することがあります。しかし、これは全体のスループットが 50% 低下することを意味するわけではありません。各サーバーの並列処理の割合は高く、2 次サーバーが 1 つのトランザクションを処理している間に、1 次サーバーは別のトランザクションを処理します。

システムで並列処理の利点を活用したい場合は、複数の接続にトランザクションを分散できます。ただし、並行して実行する照会の数が多いほどサーバーに必要なメモリーの量が増えるため、接続を追加して照会を並列に実行することが常にスルー

プットを高めるわけではないことに注意してください (特に、大量のメモリーを備えていないシステムの場合)。一度に実行する照会の最適の数を知るために、実際に試行が必要になる場合もあります。

2 次サーバーでの読み取り専用トランザクションの実行

クライアントは、2 次サーバーに接続して読み取り専用操作を行うことができます。

状況によっては、読み取り専用クライアントを 2 次サーバーに接続して、そこで読み取りを実行させることによって「負荷を分散」し、システム全体のパフォーマンスを向上させることができます。これは特に、多数のレコードを読み取る必要があり、レコードを変更する必要がない、報告書作成や「データウェアハウジング」の照会などの作業に便利です。

その他の安全性の要因

ホット・スタンバイの手法は、システムの単一部分の障害に対してユーザーを保護するように設計されています。しかし、両方のサーバーが同じ問題 (例えば、電源障害など) によって影響を受ける可能性がある場合は、ユーザーを保護しません。

DurabilityLevel を「relaxed」または「adaptive」に設定し、確認応答ポリシーが 2-safe received の場合は、両方のサーバーがほとんど同時にダウンすると、トランザクションを失う可能性があります。最低でも、それぞれのサーバーを別々の無停電電源装置 (UPS) に接続し、電源障害から保護してください。さらに、どのデータベース・システムでも同じですが、重要なデータはバックアップを作成し、なるべく別の場所に保存してください。HotStandby は、データのバックアップの代わりになりません。

ヒント:

バックアップは、HSB ペアのどちらのサーバー上でも (ADMIN COMMAND backup コマンドを使用して) 実行できます。多くの場合、バックアップの作成に使用可能なリソースをより多く備えているのは、2 次サーバーです。

高可用性コントローラー (HAC)

高可用性コントローラー (HAC) は、solidDB HotStandby 構成のための自動冗長性管理プログラムです。このプログラムはデータベース・サービスの可用性を維持するために、障害を検出し、スタンバイ装置へのフェイルオーバーを行い、必要なときは障害を起こしたプロセスを再開します。

障害の原因としては、データベース・ノードのハードウェアの問題、データベース・プロセスの失敗、または HSB リンクの切断が考えられます。HAC はサーバーの HSB 状態をモニターし、障害が起きた場合には障害の影響を受けていないサーバーが 1 次サーバーのロールを保持し、そのサーバーがトランザクションの負荷をいつでも受け入れられるようにします。

言い換えれば、HAC はウォッチドッグ・プログラムの役割を果たします。そのインプリメンテーションでは、サーバー状態をモニターするために、solidDB のイベント・メカニズムが使用されます。HSB サーバーの状態が変化したときはいつでも、

そのサーバーから HAC にイベントが送信され、その場合、HAC は、HotStandby 管理コマンド API を使用してアクションを実行する必要があるかどうかを推定します。

HAC のアーキテクチャーを下の図に示します。

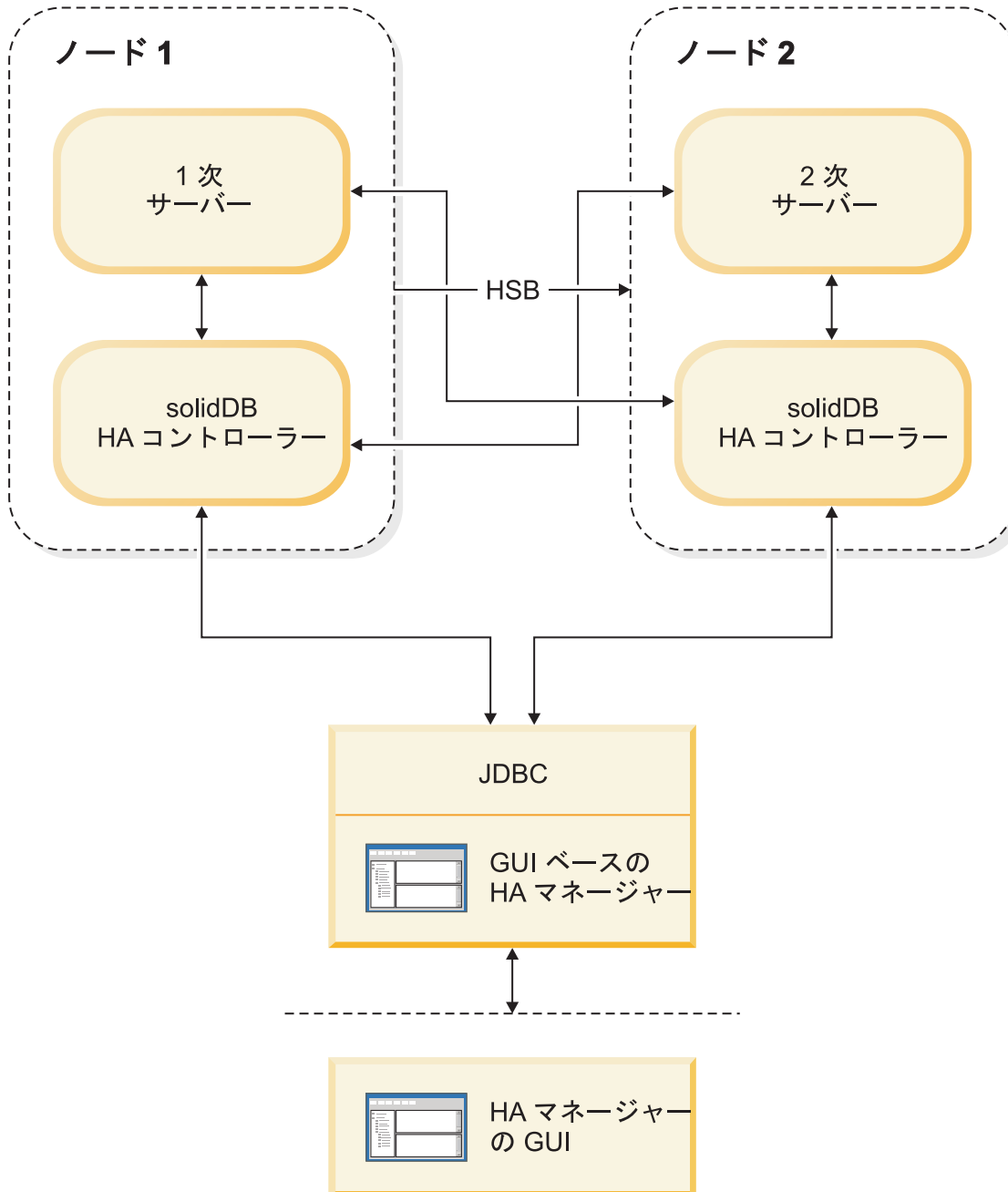


図7. 高可用性コントローラーのアーキテクチャー

HAC には、主に以下の 2 つの目的があります。

1. HAC をウォッチドッグとして使用して、データベース・サービスの可用性を自動的に維持することができます。この AUTOMATIC モードと呼ばれるモードでは、HAC は以下のアクションを実行します。

- データベース・サーバー・プロセスを開始し、再開し、終了します (オプション)。
 - 各サーバーとそれらの間の HSB リンクの状態をモニターします。
 - 実行する必要があるアクションを推定します。
 - アクションを実行します。
2. HAC は、HSB システムのモニターの役割も果たし、(別の誰かによって起動された) HSB サーバーの状態変更をモニターし、システムの状態を報告します。これは、ADMINISTRATIVE モードと呼ばれます。このモードでは、HACは HSB の状態遷移を実行せず、それ以外の HSB システムの変更も行いません。

HAC では、高可用性マネージャー (HAM) と呼ばれるグラフィカル・ユーザー・インターフェース (GUI) コンポーネントも使用できます。これは、サンプル・ディレクトリー 'samples¥hac¥' に含まれています。

高可用性コントローラーは、solidhac.ini 構成ファイルを通じて構成されます。HAC を開始する前に、このファイルを HAC の作業ディレクトリーに配置してください。製品パッケージ内に構成ファイル・テンプレートがあり、それに、使用可能なすべての構成パラメーター (コメント付き) と例が組み込まれています。HA マネージャーも、同様な方法で構成されます。

認識される障害

HAC は、HotStandby サーバーの正常性と状況をモニターします。データベース・プロセスの失敗やコンピューター・ノードの障害など、障害状態においては、HAC はフェイルオーバーやその他の必要な状態遷移を実行して、データベース・サービスの使用可能な状態をできる限り維持します。

考慮されるすべての障害について、それらの障害は、2 つの HSB サーバーの PRIMARY ACTIVE 状態および SECONDARY ACTIVE 状態によって表現される通常の完全な作動可能状態で発生することが想定されています。HAC は、単一の障害のみに対処します。言い換えれば、システムが前回の障害からリカバリーする前に障害が発生しないことを前提としています。ただし、HAC で処理できるいくつかの複数障害のシナリオが事前定義されています。

単一の障害に関する限り、HAC はデータベース・サービスをほとんど中断することなく維持します。複数の障害が発生した場合、HAC は、エラーのあるシステム状態 (二重 1 次サーバーなど) を回避しようと試みます。

HAC が処理できる障害は、以下のとおりです。

- 単一の障害
 - 1 次 (ACTIVE) データベース・サーバー・プロセスの失敗
 - 2 次 (ACTIVE) データベース・サーバー・プロセスの失敗
 - 1 次ノードの障害
 - 2 次ノードの障害
 - 外部参照エンティティーが使用されている場合、HAC は HotStandby リンクの障害、つまり 2 つの HotStandby データベース・プロセスの間で失われた接続も処理できます。外部参照エンティティーの詳細については、22 ページの『外部参照エンティティー (ERE)』を参照してください。

- 二重障害
 - 前回の障害からリカバリーする間、HAC は 1 次側と 2 次側のデータベース間の同期に関するエラーを認識します。
 - また、HAC は、もっと一般的でない障害にも対処します。例えば、サーバーが前回の障害の後、HSB リンクを確立しようとしているときに起きたサーバー・プロセスの失敗などです。

データベース・サーバー・プロセスの制御

HAC は、データベース・プロセスを開始し、失敗したプロセスを再開するよう構成することができます。

HAC インスタンスは、ローカル・データベース・サーバーとの接続を失うと、solidhac.ini 構成ファイル内で指定された開始スクリプトを呼び出します。このスクリプトは、ユーザーが提供します。スクリプトの例はパッケージに添付されています。

重要: HAC インスタンスは、開始スクリプトが終了する時点でサーバーが稼働しており、対応できるものと想定します。HAC はサーバーの始動の際に発生する障害を処理しないため、スクリプトはサーバーが接続を受け入れた場合以外、終了してはなりません。

データベース・サーバーが障害を起こしたか、それ以外の何らかの理由で消失した場合、HAC はデータベース・サーバーを再始動しますが、以下の 2 つの場合は例外です。

- データベース・プロセスがオペレーティング・システムのプロセス・リストから消失していない場合、または
- データベース・サーバーが HA マネージャーを使用してシャットダウンされた場合。

外部参照エンティティ (ERE)

処理を必要とする最も難しい障害状態の 1 つは、データベース・ノード間の通信リンクに障害が起き、どちらのデータベース・サーバーも相手側がダウンしたと想定する可能性がある状態です。この状態は、二重 1 次サーバー (「分離脳」) 状態につながる場合があり、後でデータベースの同期がとられたときに、トランザクションを失うおそれがあります。できるだけ solidDB HAC による誤った判断を避けるために、外部参照エンティティ (ERE、つまりネットワーク参照デバイス) を使用して、ネットワークの正常性を検査することが推奨されます。例えば、1 台のコンピュータでネットワーク・アダプターの障害が起きた場合、HAC はその状態を検出でき、正しいデータベース・ノードが 1 次側のデータベースとして (また、もう一方が 2 次側として) 機能を続行するよう設定することができます。

ERE を使用した場合、HAC は HotStandby ノード と ERE デバイスの間の物理リンクの状況を検査するために、ERE に対して ping を行います。最も近い ERE への物理リンクが動作していない場合、ローカル HAC はローカル・サーバーを SECONDARY ALONE 状態に設定します。最も近い物理リンクが動作しており、他のサーバーへの接続が使用可能でなければ、ローカル HAC はローカル・サーバーがサービスの提供を続行するサーバーであると判定して、ローカル・サーバーを PRIMARY ALONE に設定します。したがって、相手側の HotStandby ノードへの接

続と、最も近い ERE への接続を失っている HotStandby ノードは、2 次サーバーになります。このようにして、ネットワーク障害が起きた場合でも、2 つの 1 次サーバー (分離脳) の状態が防止されます。

ERE 用の HAC の詳しい構成方法については、52 ページの『HA コントローラーおよび HA マネージャーの構成』を参照してください。

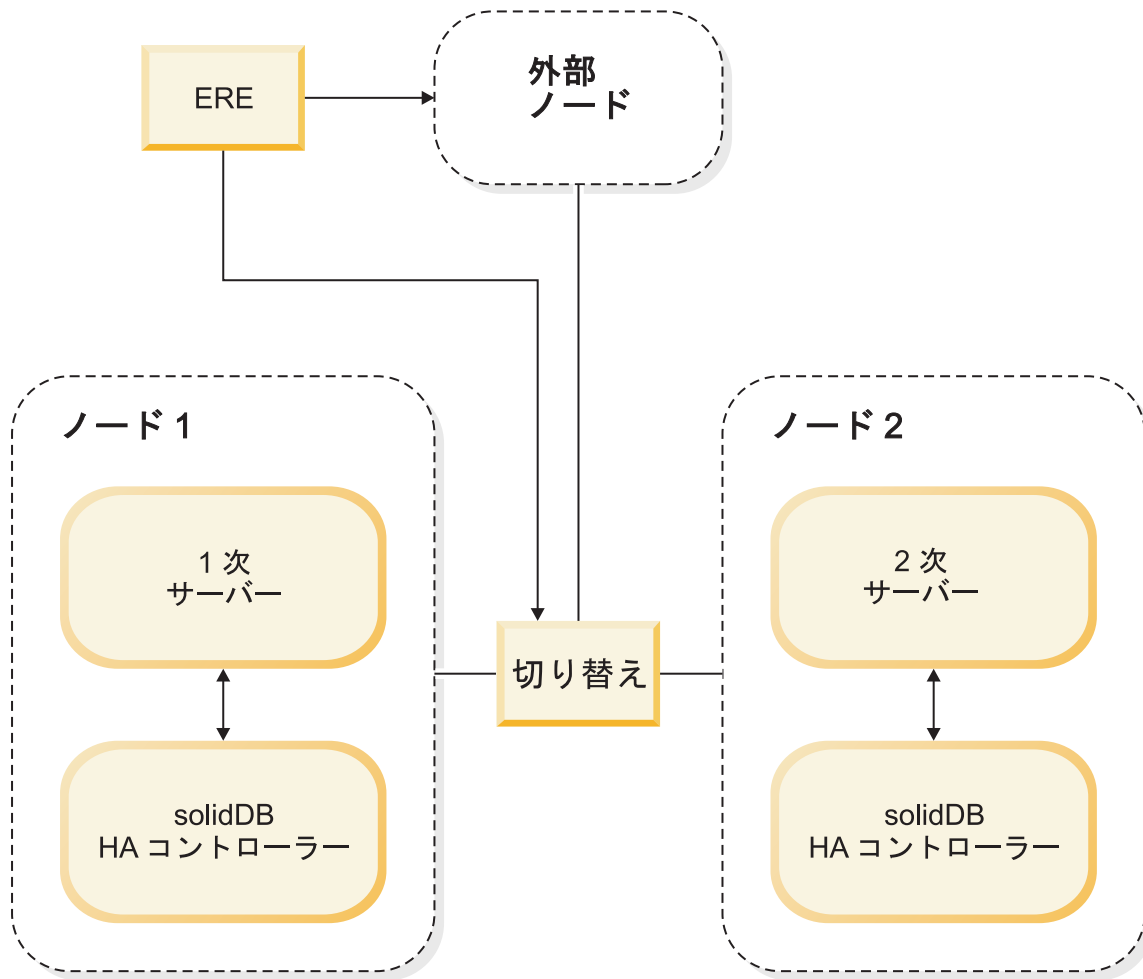


図 8. 外部参照エンティティのコンポーネント

上の図は、ERE に可能な以下の 2 つのロケーションを示しています。

- クラスタ・スイッチ
- クラスタの外部にある、ネットワーク内の任意のコンピューター。クラスタ内で冗長ネットワーク (つまり、重複するネットワーク・コントローラー、ケーブル およびスイッチ) を使用している場合は、ERE をクラスタの外部に定義することをお勧めします。

重要: HotStandby リンクが信頼できないと考えられる場合は (ERE を使用しているすべてのケースも含めて)、以下の HotStandby サーバー・パラメーターをファクトリー値に設定する必要があります。

HotStandby.AutoPrimaryAlone=no

ERE は、キープアライブ・メッセージが使用するのと同じ HSB リンクを使用する必要があります。

HAC でのネットワーキング

それぞれの HSB サーバーで、すべての通信に単一の論理ネットワーク・アクセス (IP アドレス) を使用することをお勧めします。

この推奨事項は、ネットワーク・インターフェース API レベルで透過的な、複数の (または予備の) 下位ネットワーク・コンポーネント (ネットワーク・インターフェース・カード、ケーブル、およびスイッチ) の使用を排除するものではありません。

サーバーは、さまざまなポート番号をさまざま目的に使用できます。

1 つの論理ネットワーク・アクセスだけを使用した場合、HAC (ERE を伴う) は HSB トランザクション・レプリケーションとデータベース・クライアント・アプリケーションの両方に影響を及ぼすネットワークの切断を検出します。別々のインターフェースを使用した場合、HAC は HSB リンクの正常性のみをモニターするため、データベース・クライアント通信での障害は HAC によって検出されません。

ただし、どのような理由であれ、ネットワーク・アクセスが信頼できないと考えられる場合は、下層にあるネットワーク・アクセスのインプリメンテーションに関係なく、HAC の ERE 機能を使用できます。

高可用性マネージャー

高可用性マネージャーは、HotStandby サーバーの状態と HAC の状態を表示する Java ベースのグラフィカル・インターフェース・ツールです。これはまた、例えば、HotStandby サーバーのロールの切り替えや HAC の一時中断と再開など、HAC を管理するための基本機能を提供します。

高可用性マネージャーは、HAManager.ini 構成ファイルを通じて構成されます。

高可用性マネージャーのスクリーン・ショットについては、以下を参照してください。

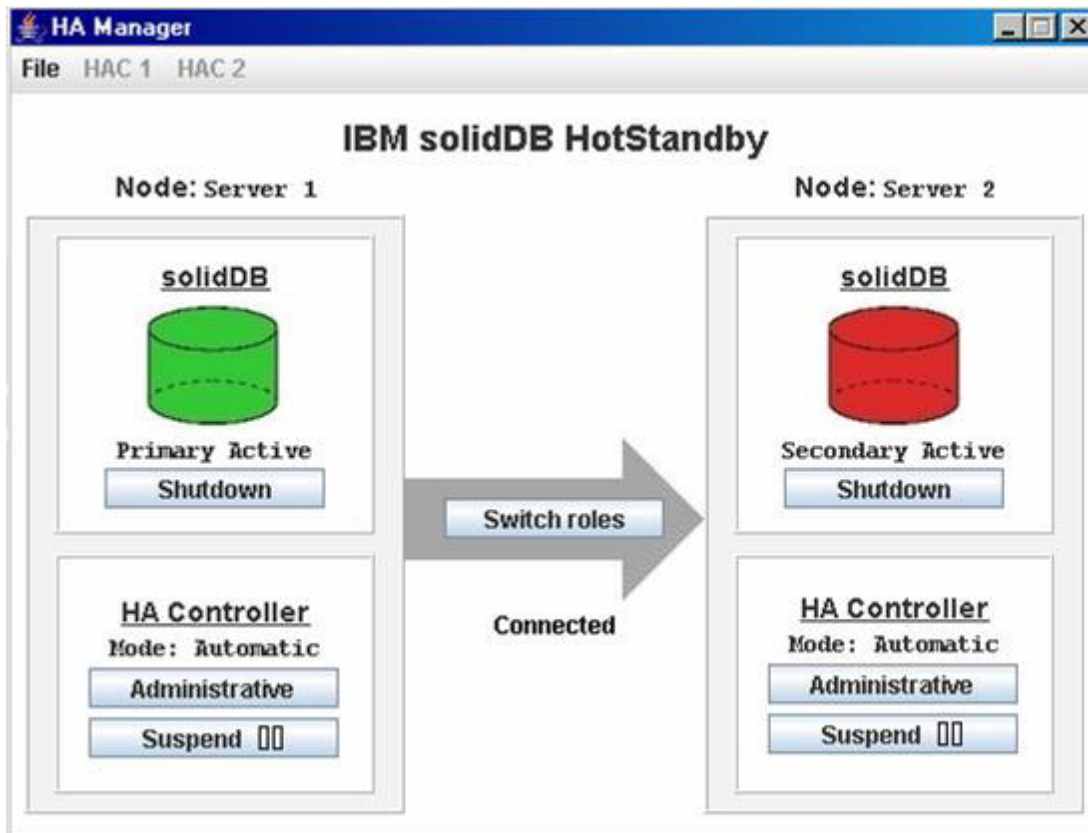


図9. 高可用性マネージャー

高可用性マネージャーの GUI を使用して、以下のアクションを実行できます。

- HotStandby サーバー間での切り替え
- AUTOMATIC と ADMINISTRATIVE の HAC モードの切り替え
 - ADMINISTRATIVE モードでは、HAC は HSB クラスターのみをモニターし、ユーザーは HSB サーバー上で管理タスクを実行できます。
 - AUTOMATIC モードでは、HAC はウォッチドッグとして機能し、障害を処理し、データベース・サービスの可用性を維持します。
- HotStandby データベース・サーバー・プロセスのシャットダウン
- HotStandby データベース・サーバー・プロセスの開始
- 高可用性コントローラーの一時中断
- 高可用性コントローラーの再開

注:

データベース・サーバー・プロセスのシャットダウンと開始を実行できるのは、以下の構成パラメーターが solidhac.ini ファイルの中で設定されている場合だけです。

HAController.EnableDBProcessControl=Yes

HAC ロギング

HAC は、ログ・レコードを HAC 作業ディレクトリーにある hacmsg.out ファイルに書き込みます。

ログには、以下に関する情報が入っています。

- 警告
- 致命的エラー
- 構成に関する情報
- 初期化に関する情報
- すべての入力イベント
- HotStandby の状態変更
- システム内の変更の原因となるユーザー・コマンド
- HAC の状態変更
- HAC のモード変更 (AUTOMATIC/ADMINISTRATIVE)
- システムの状態変更の原因となるイベント

HAC ログ・ファイルの最大サイズは 64 メガバイトです。サイズが限度を超えると、hacmsg.out は hacmsg.bak という名前に変更され、新しい hacmsg.out が作成されます。これらのファイルには、最大 128 メガバイトの最新のログが格納されます。

2 HotStandby の概要

このセクションでは、2 つの solidDB HotStandby サーバー (1 次サーバーと 2 次サーバー) をセットアップする段階的な手順を示します。

ここでは、既に solidDB HotStandby がインストールされていることを想定しています。必ず、製品に添付されていたインストールの説明に従ってください。

高可用性コントローラー (HAC) をセットアップする手順は、35 ページの『3 章 HotStandby の管理および構成』に記載されています。

始める前に

このタスクについて

HotStandby をセットアップする前に、以下の情報に注意してください。

1. 35 ページの『必要な知識』を参照してください。このセクションには、solidDB HotStandby コンポーネントの使用に関する重要な情報が記載されています。
2. HotStandby の機能について学習するには、solidDB パッケージに含まれているデモンストレーションを実行します。詳細については、以下のセクションを参照してください。
3. HotStandby のセットアップと構成に関する段階的な手順については、28 ページの『HotStandby クイック・スタート手順』を参照してください。
4. solidDB コマンド行 SQL エディター solsql に慣れておいてください。

HotStandby のデモンストレーション

インストールした solidDB ソフトウェア・パッケージには、HAC またはサンプルのウォッチドッグのインプリメンテーションを使用して HotStandby コンポーネントのデモンストレーションを実行するために必要な、すべてのファイルが含まれています。デモンストレーションは単純化されていますが、HotStandby コンポーネントの使用方法に関する理解を深めてくれます。

HAC のデモンストレーションは、`samples¥hac` に入っています。HSB/HAC のデモンストレーションの詳細な説明は、ファイル `samples¥hac¥readme.txt` に入っています。

HAC システムのセットアップ手順については、31 ページの『HA コントローラーを伴う HSB のクイック・スタート手順』を参照してください。

Watchdog のデモンストレーションの詳細な説明は、ファイル `samples¥hsb¥readme.txt` に入っています。

HotStandby クイック・スタート手順

このセクションでは、HotStandby のクイック・スタート手順について説明します。この手順を実行すると、HSB システムでアプリケーションのサービスを開始できる状態に到達します。

このタスクについて

この方式では、HSB サーバー以外の solidDB コンポーネントを想定していません。例えば、HAC は必要ありません。HAC についての同様な段階的な手順は、30 ページの『HA コントローラーの開始と停止』で説明されています。また、サンプルのウォッチドッグ・アプリケーションが solidDB HotStandby パッケージに含まれています。サンプルのウォッチドッグを使用するには、そのための構成設定を行う必要があります。

HSB サーバー・ペアをセットアップして実行するには、ネットワークで接続した 2 台のコンピューターが必要です。HotStandby サーバーを (他の solidDB コンポーネントなしに) セットアップするには、以下の手順を実行します。

1. 1 次ノードと 2 次ノードを構成します。

HotStandby を使用するには、最小でも solid.ini 構成ファイルの [HotStandby] セクションに以下のパラメーターを構成する必要があります。

- **HSBEnabled=Yes**

HSBEnabled パラメーターを省略するか、HotStandby 用に予定しているサーバーに「no」の値がある場合、サーバーは始動したときに非 HotStandby サーバーになります。

- **Connect=connect string for the opposite HSB server**

Connect パラメーターを省略した場合、サーバーは HotStandby サーバーとして始動しますが、サーバーを接続するためには、事前に ADMIN COMMAND を使用して接続ストリングを指定する必要があります。

ローカル・リカバリーを目的とする HotStandby のトランザクション・ロギングは、使用可能と使用不可のどちらにも設定できます。ロギングを使用不可にした場合、1 次サーバーはトランザクション・ログ情報の必要な部分をレプリケーションのためにメモリー内に保持します。トランザクション・ロギングを使用不可にすると、書き込みトランザクションのパフォーマンスが向上しますが、データの安全度が減少します。デフォルトでは、トランザクション・ロギングは使用可能に設定されます。これは、solid.ini 構成ファイルに以下のパラメーターを設定することにより、使用不可に設定できます。

```
[Logging] LogEnabled=no
```

上記のパラメーターのほかにも、使用できるパラメーターが複数あります。これらのパラメーターの詳細については、47 ページの『1 次および 2 次 HotStandby 構成の定義』を参照してください。

TCP/IP プロトコルを使用している場合は、アプリケーションと HotStandby サーバーの間のタイムアウト間隔を調整してもよいでしょう。詳細について

は、45 ページの『アプリケーションとサーバー間のタイムアウトの定義 (Com セクション)』を参照してください。

オプションとして、HotStandby 固有の構成パラメーター・オプションのデフォルト設定を変更することもできます。それらのパラメーターの詳細については、47 ページの『HotStandby 固有のパラメーターの構成』を参照してください。

2. 他の solidDB サーバーと同じ方法で、両方の HSB サーバーを始動します。

サーバーは、それぞれの独自の solid.ini ファイルから HotStandby 構成情報を読み取ります。始動後の両方のサーバーの (HSB) 状態は、SECONDARY ALONE です。

3. 1 次サーバーにするサーバーを選択し、以下のコマンドを発行することにより、選択したサーバーの状態を PRIMARY ALONE に切り替えます。

```
ADMIN COMMAND 'hotstandby set primary alone';
```

注: ADMIN コマンドでは、「hotstandby」の代わりに、省略形の「hsb」を使用してもかまいません。以下に例を示します。

```
ADMIN COMMAND 'hsb set primary alone
```

4. 1 次サーバーを 2 次サーバーに接続するために、どちらかのサーバーで以下のコマンドを発行します。

```
ADMIN COMMAND 'hsb connect'
```

接続が成功したかどうかを検証するために、以下のコマンドを入力します。

```
ADMIN COMMAND 'hsb state'
```

1 次サーバーは、状態が「PRIMARY ACTIVE」であることを応答する必要があります。しかし、1 次サーバーの状態が予期した PRIMARY ALONE でない何か別のものである場合は、例えば以下のコマンドを入力することにより、hsb connect の状況を検査できます。

```
ADMIN  
COMMAND 'hsb status connect'
```

結果が ACTIVE である場合、接続プロセスは、まだアクティブです。しかし、結果が BROKEN の場合には、接続する前に、両方のサーバーのデータベースを同期させる必要があります。

5. データベースの同期をとるには、1 次サーバーで以下のコマンドを実行します。

```
ADMIN COMMAND 'hsb netcopy'
```

データベース・コピー・プロセスの状況は、次のコマンドを入力して検査できます。

```
ADMIN COMMAND 'hsb status copy'
```

6. 結果が SUCCESS になると同時に、両方のデータベースが同期し、サーバー同士を接続できます。どちらかのサーバーで、以下のコマンドを再実行してください。

```
ADMIN COMMAND 'hsb connect'
```

ステップ 4 で説明したように、成功したかどうかを検証します。

7. アプリケーションの使用を開始します。

HA コントローラーの開始と停止

HAC を開始するには、正しく記入した `solidhac.ini` 構成ファイルを事前に HAC 作業ディレクトリーに入れておく必要があります。HAC の構成の簡単な説明については、31 ページの『HA コントローラーを伴う HSB のクイック・スタート手順』を参照してください。HAC の構成に関する詳しい説明は、52 ページの『HA コントローラーおよび HA マネージャーの構成』および 56 ページの『`solidhac.ini` 構成ファイル』にあります。

注:

使用しているプラットフォームに応じて、HAC バイナリーには、`solidhac` または `solidhac.exe` という名前が付いています。ここでは、分かりやすいよう、`solidhac.exe` を使用します。同様に、サンプルの IBM `solidDB` 開始スクリプトの名前は、`start_solid.sh` または `start_solid.bat` です。以下の説明では、`start_solid.bat` を使用します。

HA コントローラーのコマンド構文は、以下のとおりです。

```
solidhac.exe [-c working
              directory | -?]
```

? 引数、または `c` を除く他のすべての引数は、使用法のメッセージを出力します。

HAC の開始

`solidhac.exe` は、`c:%solid%hac` に置かれているとします。これは現行ディレクトリーでもあり、ここでは `c:%solid%run%server1` を HAC の作業ディレクトリーとして使用します。HA コントローラーは、以下のコマンドを使用して開始されます。

```
solidhac.exe -c c:%solid%run%server1
```

または

```
solidhac.exe
-c ..%run%server1
```

HAC は、開始すると、`solidhac.ini` 構成ファイル内で指定されているポートの `listen` を開始します。そのポートは、HAC と HAManager (Java™ GUI) の間でコマンドの転送に使用されます。さらに HAC は、ADMIN COMMAND を使用してそのポートへ送信された終了コマンドを受け入れます。

下記のコマンドを実行して、HAC を停止 (強制終了) することができます。このコマンドを発行するには、`solsql` などの SQL ツールを使用するか ODBC インターフェースを使用して、コマンドを HAC プロセスに送信します。`solidhac.ini` 構成ファイル内の `listen` パラメーターで述べたポートを使用してください。コマンドは以下のとおりです。

```
ADMIN COMMAND
'hacontroller shutdown'
```

HAC の管理の詳細については、24 ページの『高可用性マネージャー』を参照してください。

HA コントローラーを伴う HSB のクイック・スタート手順

このセクションでは、HAC を伴う HotStandby のクイック・スタート手順について説明します。

このタスクについて

この手順は、28 ページの『HotStandby クイック・スタート手順』の手順によく似ています。しかし、単に 2 つの HSB サーバーをセットアップする代わりに、この手順では、HA コントローラーによって可用性が保証された高可用性 HSB システムをセットアップする方法を説明します。以下の手順を実行すると、障害に対する耐久力のある HSB システムでアプリケーションにサービスを提供できます。

高可用性 HSB システムをセットアップして実行するには、ネットワークで接続した 2 台のコンピューターが必要です。HotStandby サーバーと HA コントローラー (1 つの HSB サーバーにつき 1 つのインスタンス) をセットアップするには、以下の手順を実行します。

1. HAC を使用するかしないかにかかわらず、HSB サーバーを同様な方法で構成します。詳細については、28 ページの『HotStandby クイック・スタート手順』を参照してください。
2. 両方の HAC を構成します。HAC は、作業ディレクトリーに置かれた `solidhac.ini` ファイルから、それ自体の構成を読み取ります。以下のリストには、必須構成パラメーターが含まれています。

[HAController] セクションの中:

- **Listen**=<listen address, 'tcp' and chosen port #>
- **Username**
- **Password**
- **DBUsername**
- **DBPassword**

[LocalDB] セクションの中:

- **Connect**=<connect address, protocol, ip/hostname, port #>
- **StartScript** (デフォルト値である **EnableDBProcessControl=Yes** の場合は必須)

[RemoteDB] セクションの中:

- **Connect**=<connect address, protocol, ip/hostname, port #>

HA コントローラーの構成の詳細については、52 ページの『HA コントローラーおよび HA マネージャーの構成』および 56 ページの『solidhac.ini 構成ファイル』を参照してください。

3. HA コントローラーを両方のノードで開始します。

HAC は自動的に、どちらのサーバーが新規 1 次サーバーになるべきかを各サーバーの前のロールに応じて判別し、ログの位置を検出します。HAC は、102 ページの『どちらのサーバーを 1 次サーバーにするかの選択』で説明されているメカニズムを使用します。一部の特殊な状況では、例えば、両方のサーバーが空のデータベースを使用して始動した場合などでは、いずれのサーバーも新規 1

次サーバーの候補として同等な適性を備えています。そのような状況では、HAC は solidhac.ini 内の [LocalDB] セクションで **PreferredPrimary** パラメーターが「Yes」に設定されている場合、ローカル・サーバーを選択します。

4. 2 番目のステップの後、一方のサーバーが PRIMARY ACTIVE 状態で、もう一方が SECONDARY ACTIVE 状態になるはずですが、必要であれば、ロールを切り替えるために、HAManager で「**Switch roles**」ボタンを押すか、2 次サーバーで以下のコマンドを実行します。

```
ADMIN COMMAND 'hsb switch primary'
```

あるいは、1 次サーバーで以下のコマンドを実行します。

```
ADMIN COMMAND 'hsb switch secondary'
```

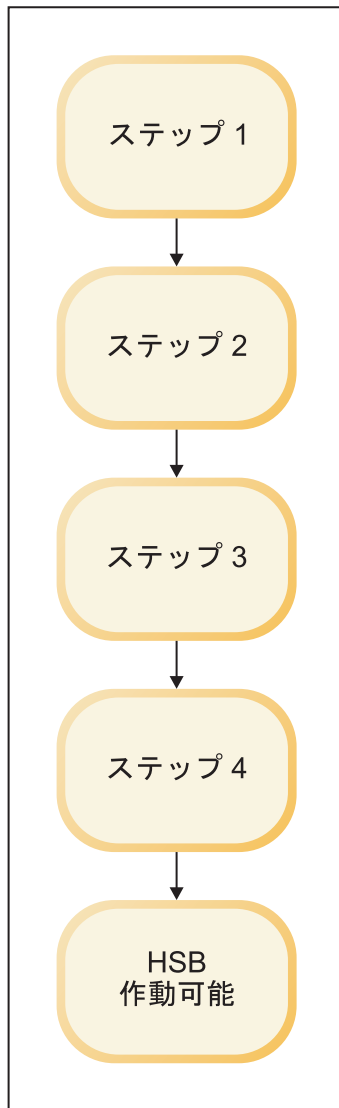
この時点以降、HAC は必要な場合にサーバーのロールを切り替えます。つまり、常に少なくとも 1 つのサーバーが、読み取りと書き込みのトランザクションを実行します。

5. アプリケーションの使用を開始します。

開始シーケンスの要約

以下の図と表は、HAC を伴うインストールと HAC を伴わないインストールのシーケンスを横並びに表したものです。図では、HAC のインストール・シーケンスは左側に、HAC を伴わないシーケンスは右側に示されています。図の後の表では、両方のインストール・タイプについて、図中で番号が付いている各ステップを説明しています。

HAC あり



HAC なし

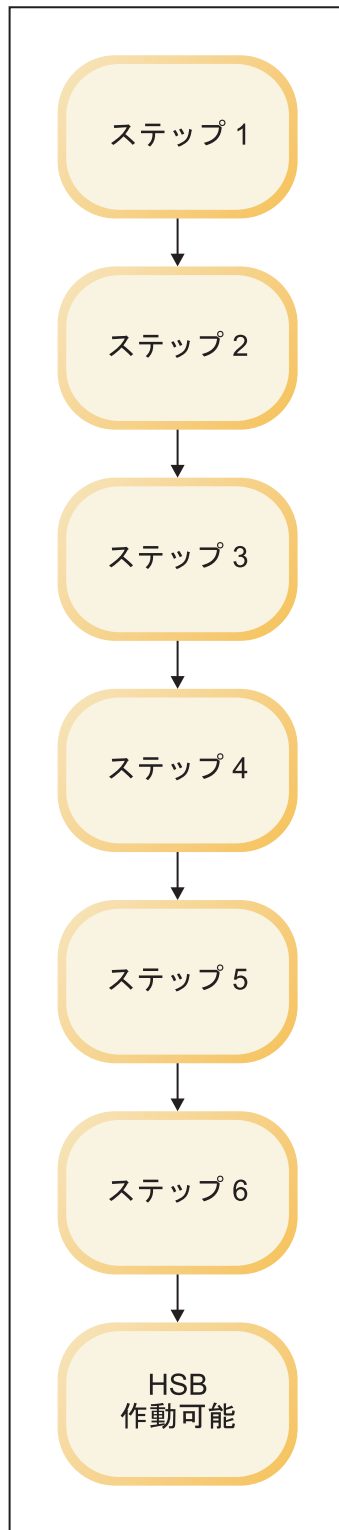


図 10. 開始シーケンスの要約

表 5. インストール・シーケンスのステップ

HAC を伴うインストール	HAC を伴わないインストール
ステップ 1. HSB サーバーを構成します。	
ステップ 2. HA コントローラーを構成します。 ステップ 3. 両方のノードで HA コントローラーを開始します。 ステップ 4. 必要であれば、HSB のロールを切り替えます。	ステップ 2. 両方のノードで HSB サーバーを始動します。 ステップ 3. データベース・サーバーの状態を PRIMARY ALONE に切り替えます。 ステップ 4. 1 次サーバーを 2 次サーバーに接続します。 ステップ 5. 接続に失敗した場合、1 次サーバーから 2 次サーバーへのデータベース・コピーを開始します。 ステップ 6. netcopy の後、再接続します。
HSB が作動可能になります。アプリケーションの使用を開始できます。	

3 HotStandby の管理および構成

このセクションでは、HSB サーバー、HA コントローラー・インスタンス、および HA マネージャーを含む HotStandby インストール済み環境を保守する方法について説明します。また、このセクションでは、HotStandby の機能をインプリメントおよび保守するためのパラメーターの設定についても説明します。

この説明は、「*solidDB 管理者ガイド*」の『*solidDB の構成*』セクションにある情報の補足です。

パラメーターは、*solid.ini* 構成ファイル内のセクション・カテゴリーに従ってグループ化されます。HotStandby を使用する場合は、*solid.ini* 構成ファイルの [HotStandby] セクションを構成する必要があります。

高可用性コントローラー (HAC) の構成は、*solidhac.ini* 構成ファイルに入っています。*solidhac.ini* 構成ファイル内のパラメーターも、さまざまなセクション・カテゴリーに従ってグループ化されます。

高可用性マネージャーの構成は、*HAManager.ini* 構成ファイルに入っています。*HAManager.ini* 内のパラメーターは、主に HA コントローラー・インスタンスの識別に使用されます。これにより、高可用性マネージャーはそれらのインスタンスにアクセスできます。

構成パラメーターは、以下の方法で変更できます。

- 手動で *solid.ini*、*solidhac.ini*、*HAManager.ini* の各構成ファイルを編集する。サーバーは始動のときにのみ構成ファイルを読み取ることにご注意ください。このため、構成ファイルに加えた変更は、対応するプログラムを次回に開始するときまで効果がありません。
- 稼働中の *solidDB* サーバーの設定を変更する場合は、*solidDB SQL* エディター *solsql* で **ADMIN COMMAND** を使用してそれを行うことができます。ADMIN COMMAND の構文を以下に示します。

```
ADMIN COMMAND 'parameter
                section_name.param_name=value';
```

例:

```
ADMIN COMMAND 'parameter hotstandby.2SafeAckPolicy=2'; ADMIN COMMAND
                'parameter com.listen="tcp sf_server 1315";
```

注意:

ADMIN COMMAND を使用してパラメーターを変更した場合、すべてではありませんが、一部のパラメーターは即時に有効になります。

必要な知識

このセクションでは、管理と保守を開始する前に HotStandby について知っておく必要があることを説明します。

HotStandby と solidDB 構成ファイル

HotStandby 機能を使用可能にするには、特殊な [HotStandby] セクションを solidDB 構成ファイル (solid.ini) の中に指定する必要があります。ライセンス・ファイルによって solidDB HotStandby コンポーネントの使用を許可されていないかもしれません。

[HotStandby] セクションに必要な最小の構成情報は、以下のとおりです。

- **HSBEnabled** を「yes」に設定する。
- サーバーに **Connect** パラメーターを設定する。この設定は、もう一方のサーバー (1 次サーバーまたは 2 次サーバー) に接続するために使用するネットワーク名を定義します。このパラメーターを solid.ini ファイルの中で設定しなかった場合、サーバーは HotStandby サーバーとして始動しません。サーバーが始動した後、ADMIN COMMAND を使用して、このパラメーターを設定または変更できることに注意してください。

各サーバーの **Connect** ストリングは、もう一方のサーバーの **Listen** ストリングに一致する必要があることに注意してください。 **Connect** パラメーターおよびその他のパラメーターの設定の詳細については、47 ページの『HotStandby 固有のパラメーターの構成』を参照してください。初めて HotStandby をセットアップする場合は、27 ページの『2 章 HotStandby の概要』を参照してください。

1 次サーバーおよび 2 次サーバーの solid.ini ファイルの例については、27 ページの『HotStandby のデモンストレーション』を参照してください。

HotStandby とアクセス権限

管理者は、HotStandby を実行するのに特殊なアクセス権限を必要としません。通常のアクセス権限で十分です。

管理の目的で、SYS_ADMIN_ROLE または SYS_CONSOLE_ROLE は HotStandby 管理コマンドを実行する必要があります。これらのコマンドは、次の solidDB SQL コマンドで実行されます。

```
ADMIN COMMAND 'hotstandby command_string';
```

例えば、solidDB SQL エディター (テレタイプ) を使用する場合は、次のようになります。

```
ADMIN COMMAND 'hotstandby status connect';
```

solidDB リモート制御 (テレタイプ) を使用して HotStandby コマンドを入力する場合は、hotstandby コマンド・ストリングのみを (引用符を付けずに) 入力します。以下に例を示します。

```
hotstandby status connect
```

solidDB ツールと HotStandby

solidDB で管理を行うために使用可能なすべてのツールは、HotStandby にも適用できます。

HotStandby 固有の管理コマンドは、solidDB SQL エディター および solidDB リモート制御 (solsql および solcon) で、それぞれ発行できます。さらに、solidDB

Speedloader (SOLLOAD)、solidDB エクスポート (SOLEXP)、および solidDB データ・ディクショナリー (SOLDD) を HotStandby と一緒に使用できます。これらのツールの説明については、「*solidDB 管理者ガイド*」の『solidDB データ管理ツールの使用』を参照してください。

高可用性マネージャーを使用して、solidDB HotStandby の状態をモニターし、HSB サーバーおよび HA コントローラーを制御することができます。HA マネージャーは、HA コントローラーとのみ連動させることができます。高可用性マネージャーについては、24 ページの『高可用性マネージャー』を参照してください。

データベース・マイグレーション (ディスク・ベースのサーバーのみ)

solidDB 4.x データベースは、以下のコマンド行パラメーターの 1 つを使用して、最新の solidDB フォーマットに変換できます。

- **-xconvert** は、データベース・ファイルを新しい構造に変換し、サーバーをシャットダウンします。
- **-xautoconvert** は、データベース・ファイルを変換し、実行を続けます。

必要なすべてのシステム表が、HotStandby 機能のための表も含めて、作成されます。変換後、変換したデータベースは旧製品バージョンで使用できなくなります。したがって、新リリースへのマイグレーションを行う前に、データベースとファイルのバックアップを作成することをお勧めします。

注:

IBM solidDB データベースは、HotStandby で使用されなくなっても、IBM solidDB との互換性を維持します。

相互運用性

HSB システム内のサーバーは、HSB 互換バージョンであることが必要です。一般に、隣接するバージョンは HSB 互換です。

旧バージョンとの HSB 互換性については、リリース・ノートを参照してください。

HotStandby での制限と警告

以下のトピックでは、HotStandby でインメモリー表を使用する場合の制限事項を示します。警告では、二重 1 次サーバーの回避とトランザクション・ログのスペース不足を扱っています。

インメモリー表

2 次サーバーに接続していて、インメモリー表からデータを読み取っている場合、トランザクション分離レベルは、たとえ REPEATABLE READ または SERIALIZABLE を指定してある場合でも、READ COMMITTED に設定されます (インメモリー表は、1 次と 2 次のどちらのサーバーでも SERIALIZABLE をサポートしません)。

ロード・バランシングを使用している場合、分離レベルはデフォルトで READ COMMITTED になります。ロード・バランシングと透過接続の詳細については、60 ページの『透過接続の使用』を参照してください。

ネットワーク分割と二重 1 次サーバー

環境によっては、両方のサーバーが PRIMARY ALONE 状態で動作する可能性があります。二重 1 次サーバー があると、深刻なりカバリー不能エラーにつながる可能性があります。

この状況では、いずれかのトランザクションを一方のサーバーだけがコミットした場合、サーバー同士を再同期させることができません。各データベースを「マージ」して、正しい情報が入った単一のデータベースを作成する方法が存在しないからです。実際には、二重 1 次サーバーの状況で「間違った 1 次サーバー」のデータベースにコミットされたトランザクションは、失われます。二重 1 次サーバーがあると、その他のエラーにもつながります。

二重 1 次サーバーの問題の原因として最も可能性が高いものは、「ネットワーク分割」です。この状況では、ネットワーク接続の全部ではなく一部が失われ、単一のネットワークが事実上複数の分離された小部分に分割された状態となり、それぞれがその部分の中で通信でき、他の部分と通信できなくなります。このため、両方のサーバーが互いに接続を失いますが、どちらも依然として稼働中であり、場合によっては一部のクライアントとの通信が維持されています。

二重 1 次サーバーのシナリオは、HSB システムの外部にあるノードで単一インスタンスのウォッチドッグを使用すれば、回避できます。そのようなウォッチドッグを使用すると、どちらのサーバーを 1 次サーバーに設定すればよいかを簡単に判断でき、クライアントが確実に 1 つの 1 次サーバーだけを認識するようにすることも簡単にできます。

HAC は、HSB サーバーを備えた同じノード内で稼働する 2 つのインスタンスから構成されていますが、HAC を ERE と一緒に使用している場合、二重 1 次サーバーの状態は事実上不可能です。

結果的に二重 1 次サーバーが発生した場合でも、誰かが元の 2 次サーバー上で (それが PRIMARY ALONE に切り替わった後に) 書き込み操作を実行できる場合を除いて、実際に矛盾するデータが生じることはありません。元の 2 次サーバーがネットワークのその他の部分から完全に切り離された場合、誰もそれに書き込むことはできず、元の 1 次サーバーは 2 次サーバーのスーパーセットとなるので、ユーザーは依然として、(サーバー同士を再接続して 2 次サーバーに元の 1 次サーバー上で行われた変更のキャッチアップを許可すれば) 整合性のある単一のデータ・セットを取得できます。

二重 1 次サーバーはまれなことではありますが、発生すると極めて危険なので、データが不整合にならないように最大の注意を払う必要があります。ERE を使用することを強くお勧めします。

一方または両方のサーバーの solid.ini ファイルで構成パラメーター

AutoPrimaryAlone=Yes を設定してあると、二重 1 次サーバーが生じる可能性が高くなります。**AutoPrimaryAlone=Yes** を使用することは、システムが障害に対して、より迅速に対応できることを意味しますが、同時に、二重 1 次サーバーを防止

するための独立した監視者 (HAC、ウォッチドッグ、または人間) がいなくなることも意味しています。ネットワークの信頼性にいくらかでも疑いがあるときは、**AutoPrimaryAlone** パラメーターをファクトリー値の No のままにしておいてください。

トランザクション・ログのスペース不足

HotStandby を使用する場合、サーバーを PRIMARY ALONE 状態にするときは、トランザクション・ログのディスク・スペースが不足しないように注意する必要があります。非 HotStandby サーバーでは、頻繁にチェックポイントを指定した場合、トランザクション・ログが極端に大きくなることはありません。サーバーは各チェックポイントの後に、そのつど「古い」トランザクション・ログを削除するからです。

特に、非 HotStandby サーバーは、チェックポイントの前に発生したデータ変更を持つログを削除します。チェックポイントの詳細については、「*solidDB 管理者ガイド*」を参照してください。

しかし、PRIMARY ALONE 状態で動作している HotStandby サーバーでは、サーバーは 1 次側と 2 次側との接続が失われた時点以降に蓄積したトランザクション・ログを保持する必要があります。2 次側が長い時間にわたってダウンした場合、サーバーは、正常に稼働していれば各チェックポイントの後に放棄するはずの大量のトランザクション・ログ・データを、保持し続ける可能性があります。最悪の場合、2 次側が合理的な時間内に復帰できず、発生するすべてのトランザクションを保管するために十分なディスク・スペースがないと、1 次側のトランザクション・ログが、使用可能なすべてのディスク・スペースを埋め尽くす可能性があります。その場合、サーバーは読み取り専用モードに切り替わります。

これを防ぐために、[HotStandby] セクションでパラメーター **MaxLogSize** の適切な値を設定できます。指定した合計ログ・サイズに到達すると、サーバーは次のチェックポイントで自動的に STANDALONE 状態に切り替わります。(ただし、ディスクレス・サーバーではディスクへの書き込みが存在しないので、状態は PRIMARY ALONE のままです。)

STANDALONE 状態に設定されたサーバーは、1 次側と 2 次側の接続が失われた時点以降のすべてのトランザクション・ログを保持しなくなります。完全なトランザクション・ログがなければ、当然ながら、単に 1 次側を 2 次側に接続し、2 次側で古いログを読み取って「キャッチアップ」しても、システムの同期はとれません。copy または netcopy コマンドを使用して、データベース全体を 1 次側から 2 次側へコピーする必要があります。

HAC を使用している場合は、HAC が上記の状態を識別し、必要なアクションを自動的に実行して各サーバーを ACTIVE 状態に導きます。

管理タスクの概要

このセクションでは、HotStandby の使用時に実行しなければならない場合がある管理タスクについて説明します。

このセクションのトピックは、以下のとおりです。

表6. 管理タスク

トピック	説明	ページ
HotStandby のリカバリー・タスクと保守タスクの実行	システム障害 (通信リンクの切断または動作不能のホット・スタンバイ・サーバーが原因で起きたもの) の場合の HotStandby タスクについて説明します。これらのタスクには以下が含まれます。 <ul style="list-style-type: none"> サーバー状態の切り替え HotStandby 操作のシャットオフ 1 次サーバーと 2 次サーバーの同期 HotStandby サーバーの接続 	『HotStandby のリカバリーと保守の実行』
ネットワークを介した 1 次側データベースの新規 2 次側へのコピー	リモート・サーバーが HotStandby 構成に新規に追加されたものである (つまり、新規 2 次サーバーである) 場合、またはリモート・サーバーのデータが破損し、置き換える必要がある場合の、データベースのリモート (ネットワーク) コピーの作成方法を説明します。	91 ページの『ネットワークを介した 1 次データベースの 2 次側へのコピー』
HotStandby 状況の検査	1 次サーバーおよび 2 次サーバーの HotStandby 状況情報を検査する方法を説明します。	97 ページの『HotStandby 状況の検査』
HotStandby サーバー状態の検証	HotStandby サーバーの状態 (1 次、2 次、スタンドアロン) を検査する方法を説明します。	100 ページの『HotStandby サーバー状態の検証』
HotStandby サーバーから非 HotStandby サーバーへの変更	HotStandby 用に構成したサーバーを、通常の非 HotStandby サーバーに設定する方法を説明します。	103 ページの『HotStandby サーバーから非 HotStandby サーバーへの変更』

HotStandby のリカバリーと保守の実行

システム障害 (通信リンクの切断または動作不能のホット・スタンバイ・サーバーが原因で起きたもの) またはサーバー保守の場合、HotStandby タスクを実行しなければならないことがあります。これらのタスクには、サーバー状態の切り替え、HotStandby 操作のシャットオフ、1 次サーバーと 2 次サーバーの同期、および HotStandby サーバー同士の接続が含まれます。

1. 以下の操作の一部または全部を実行します。

- a. サーバー状態を切り替えます。

これには、1 次サーバーの PRIMARY ALONE 状態 (後で 2 次サーバーへ送信できるよう、トランザクション・ログへのトランザクションの蓄積を続行します) への切り替え、または HotStandby のシャットダウンが含まれます。
 - b. 1 次側と 2 次側のデータベースが同一になるよう、サーバー同士の同期をとります。
 - c. 何らかの理由で通信リンクが切断されている場合は、1 次サーバーを 2 次サーバーに接続します。
2. 以下のように、HAC と HA マネージャーで同じステップを実行できます。
 - a. HA マネージャーで「**Switch**」ボタンを押します。サーバーをシャットダウンする必要がある場合は、HA マネージャーで「**Shutdown**」ボタンを押します。
 - b. HAC インスタンスを ADMINISTRATIVE モードに設定するため、HA マネージャーで「**Administrative**」ボタンを押します。
 - c. HAC インスタンスを AUTOMATIC モードに設定するため、HA マネージャーで「**Automatic**」ボタンを押します。

次のタスク

これらのトピックは、以下のセクションで説明されています。1 次側または 2 次側データベースへのアプリケーションの再接続の詳細については 73 ページの『アプリケーションから 1 次サーバーへの再接続』を参照してください。

重要:

HAC を使用する場合は、HA マネージャーを使用して管理手順を実行するか、HAC インスタンスを ADMINISTRATIVE モードに設定してから、管理作業を開始します。

特殊な構成: コストの低減か安全性の向上か

solidDB の HotStandby ソリューションでは、1 次サーバーと 2 次サーバーのペアを使用して真のホット・スタンバイ機能を提供します。しかし、サーバーのペアを使用することは、すべてのお客様にとって最適であるとは限りません。瞬時に近いフェイルオーバーが必要ない場合は、すべての 1 次サーバーに 2 次サーバーを設けることは、その費用を正当化できないかもしれません。それとは反対に、一部のお客様は特別な信頼性を必要とし、すべての 1 次サーバー用に 2 次サーバーを購入するだけでなく、1 台以上の追加のスペア・サーバーを「スペアのスペア」として、購入する資金を持っている場合もあります。これにより、1 次サーバーに障害が起きて 2 次サーバーが取って代わったときに、元の 1 次サーバーを迅速に修復できない場合は、スペアを「新規 2 次サーバー」として使用できます。

お客様がコストの低減または信頼性の向上を実現できるよう、solidDB HotStandby (HSB) は、標準ホット・スタンバイ・モデルに代わるいくつかの代替モデルをサポートしています。標準モデルは、1 次サーバーと 2 次サーバーの数が同じ (「N」) であることから、「N+N」または「2N」モデルと呼ばれることもあります。代替モデルには、以下のものがあります。

- 「N + 1 スペア」または「N + M スペア」：これは、スタンドアロン用のスペア・ノードのシナリオです。この場合、N 台の「1 次サーバー」と 1 台以上のスペアが存在します。2 次サーバーは存在しません。障害を起こした「1 次」サーバーは、スペアによって置き換えられます。これは真の「ホット・スタンバイ」シナリオではなく、むしろ「ウォーム・スタンバイ」と呼ぶ方が当たっています。コンピューターは使用可能ですが、データベースのコピーを持っていないからです。
- 「2N + 1 スペア」または「2N + M スペア」：これは、HotStandby 用のスペア・ノードのシナリオです。この場合、N 個の HotStandby ペアが存在します。つまり、すべての 1 次サーバーごとに 1 台の 2 次サーバーが存在します。さらに、M 台のスペアが存在します。ここで、M は 1 以上で、通常は N 未満です。1 次サーバーまたは 2 次サーバーに障害が起きた場合は、スペアの 1 台が新規 2 次サーバーとして取り込まれます。これにより、1 次サーバーは、たとえ元のパートナーに障害が起きた場合でも、決して単独で長期間動作することはありません。

以下のトピックでは、N+M (または N+1) および 2 N+M (または 2N+1) の手法と、それらをインプリメントするのに役立つ solidDB 機能について説明します。

コストの削減: N + 1 スペアと N + M スペアのシナリオ

これらのシナリオでは、N 台の「1 次」サーバーがあり、そのそれぞれがスタンドアロン状態で作動します。つまり、2 次サーバーに接続されていません。さらに、M 台のスペア・サーバーが存在します。ここで、M は 1 以上で、通常は N 未満です。「1 次」サーバーに障害が起きた場合は、スペアのうちの 1 台がそれにとって代わります。データは元のサーバーからスペアにコピーされ、その後、元のサーバーはオフラインにされ、スペアが元のサーバーとして機能するよう構成されます。すべてのスペアが、すべての 1 次サーバーにとって代わることができる点に注意してください (特定の 1 次サーバー専用のスペアは存在しません)。また、フェイルオーバーがほとんど瞬時に行われるわけではないことにも注意してください。

この手法を、ここでは「N + 1」(単一スペア) または「N + M」(複数スペア) シナリオと呼びます。

この手法では、どこかに元のサーバーのデータのコピーを持っている必要があるため、元のサーバーのディスク・ドライブが損傷していて、データのバックアップも存在しない場合、この手法は使用できません。この N+M の手法は、以下の状況で最も便利です。

1. 予期できない障害でなく、定期保守を処理するために、スペア・ノードを使用している場合。
2. スペア・サーバーに素早くコピーでき、信頼できるバックアップを持っている場合。
 - a. 磁気テープ上か RAID ドライブ、またはその他の安全な場所にバックアップを持っている場合。または、
 - b. solidDB の拡張レプリケーション・テクノロジーを使用しており、障害を起こしたサーバーの拡張レプリケーションの「マスター」または「レプリカ」から、十分なデータをコピーまたは再作成できる場合。
3. データの個々の部分は、重大データでないか固有でない場合。

- a. 例えば、本当に必要なものが特定のデータでなくコンピューターの「計算能力」(負荷分散機能)であるなら、標準または「シード」データベースをコピーするか、クライアントからデータを取得して実行を続けるだけでよい場合もあります。
- b. 同様に、すべてのサーバーがほぼ同じデータを持っており、書き込み要求が少ないか 1 つもない「読み取り」要求のほとんどすべてに対応している場合(例えば、多数のサーバーを実行していて、そのすべてのサーバーが同じインターネット・ルーティング表または電話帳情報を使用している場合)は、どのコンピューターからでも有用なデータベースをコピーできます。

信頼性の向上: 2N + 1 スペアおよび 2N + M スペアのシナリオ

通常の solidDB HotStandby 動作には高い信頼性があります。1 次と 2 次の両方のサーバーがほとんど同時に障害を起こす可能性は、別々の信頼できる電源機構を使用していれば、非常に低くなります。しかし、その危険さえも減らしたい場合、または障害を起こしたサーバーを迅速に修復できない場合は、どうでしょう。理想的には、1 次サーバーが障害を起こし、それを 2 次サーバーで置き換えた場合(または 2 次サーバーが障害を起こした場合)、完全なサーバー・ペアで実行を続けられるよう、「旧」2 次サーバーに取って代わる「新規」2 次サーバーを持っていたいと考えるでしょう。

その状態を、 $2N + 1$ スペア(または $2N + M$ スペア)のシナリオと呼びます。その場合、 N 台の 1 次サーバーと N 台の 2 次サーバー、それに、障害を起こすか 1 次サーバーに変換された 2 次サーバーに取って代わる、少なくとも 1 台のスペアを使用します。各スペアは特定のサーバー(または HSB サーバー・ペア)専用ではなく、障害を起こしたサーバーをスペアで置き換えられるようにするためには、事前に何らかの構成が必要になります。

solidDB HSB による $N+1$ ($N+M$) と $2N+1$ ($2N+M$) の手法のサポート

スペア・サーバーは、置き換えられるサーバーに似たものにする必要があります。一般に、これは以下のことを意味します。

1. データをスペアにコピーする必要があります。
2. 置き換えられるサーバーと同じネットワーク・アドレスか、クライアント・プログラムが通信用に認識している別のアドレスで「listen」を行うよう、スペアに指示する必要があります。
3. さらに、 $2N+1$ ($2N+M$) のシナリオでは、新規 2 次サーバーと現行 1 次サーバーに相互の通信方法を指示する必要があります。言い換えれば、それぞれのサーバーに、相手との接続に使用するアドレスを指示する必要があります。

これらのニーズをサポートするために、solidDB は次の 2 つの機能を備えています。

- solidDB では、スペア・サーバーをシャットダウンすることなく、スペア・サーバーにデータをコピーできます。
- solidDB では、特定の構成パラメーターを動的に設定できます。

これらについて、以下で詳しく説明します。

1. solidDB の構成パラメーターを設定するには、通常ではサーバーをシャットダウンし、solid.ini 構成ファイルを更新し、サーバーを再始動しますが、一部の構成パラメーター（「com.listen」パラメーターや「hotstandby.connect」パラメーターなど）は、以下のような ADMIN コマンドを実行することによっても変更できます。

```
ADMIN COMMAND 'parameter com.listen="tcp SpareServer1 1315"'; ADMIN COMMAND 'hsb
parameter connect "tcp srvr27 1316"');
```

これは、最初にシャットダウンしなくても、スペアを別のサーバーに取って代わるよう動的に構成できることを意味しています。同様に、1 次サーバーに新規 2 次サーバーの Connect ストリングを指示することができます。

注意:

これらのコマンドを実行しても、更新されたパラメーター値は solid.ini ファイルに書き込まれません。したがって、サーバーが次の再始動時に新しい値を持つようにするには、上記のコマンドを実行するだけでなく、solid.ini ファイルも更新する必要があります。

重要:

スペア・サーバーは、1 次サーバーからのネットコピーを受信できるようにするために、**-x backupserver** コマンド行オプションを使用して始動してください。**-x backupserver** オプションの詳細については、92 ページの『2 次サーバー用の新規データベースの作成』を参照してください。また、「solidDB 管理者ガイド」のコマンド行オプションの説明も参照してください。

2. solidDB の「netcopy」コマンドを使用すると、既に稼働中のサーバーにデータベースをコピーできます。

- a. 「connect」パラメーターの新しい値を設定します。

```
ADMIN COMMAND 'hsb parameter
connect "tcp srvr27 1316"');
```

- b. netcopy コマンドを実行します。

```
ADMIN COMMAND 'hsb
netcopy';
```

- c. 以下のコマンドを実行して、現行の 1 次サーバーを新規 2 次サーバーに接続します。

```
ADMIN COMMAND 'hsb connect';
```

スペアでの HAC の使用

HAC は、スペアのシナリオではサポートに制限があります。HAC はスペア内で使用できますが、1 次サーバー・ノード内の HAC がスペア・サーバーの HSB 状態のモニターを開始するためには、事前に 1 次側の HAC の接続情報、**RemoteDB.Connect** を更新しておく必要があります。そのためには、solidhac.ini ファイルを更新し、1 次サーバー・ノード内の HAC を再開する必要があります。

同様に、スペア・ノード内の HAC は 1 次サーバーの接続情報を必要とします。それが事前に分かっていない場合は、その情報を solidhac.ini ファイルに追加する必要があります。その情報を solidhac.ini ファイルに追加した後、HAC を再開する必要があります。

HotStandby 用の solidDB の構成

1 次側と 2 次側の両方のノードの solid.ini ファイルには、HotStandby システムをセットアップするために必要なパラメーターが入っています。

2 次および 1 次ノード構成の定義 (Com セクション)

1 次または 2 次サーバーのネットワーク名は、通信プロトコルとサーバー名からなっています。ネットワーク名は、構成ファイルの [Com] セクションにある **Listen** パラメーターで定義されます。solid.ini ファイルは、solidDB プログラムの作業ディレクトリーか、**SOLIDDIR** 環境変数によって設定されたディレクトリーに置かれている必要があります。

1 次または 2 次サーバーは、1 つまたは複数のネットワーク名を使用できます。ネットワーク名のすべてのコンポーネントでは、大/小文字を区別しないことに注意してください。

solid.ini ファイルのエントリー

1 次ノード:

```
[Com]
;The Primary server listens to the network with this name
Listen = tcp 1320
```

2 次ノード:

```
[Com]
;The Secondary server listens to the network with this name
Listen = tcp 1321
```

Listen パラメーターの詳細については、「*solidDB 管理者ガイド*」を参照してください。

アプリケーションとサーバー間のタイムアウトの定義 (Com セクション)

このセクションでは、「Application Read Timeout」と「Connect Timeout」の設定を、solid.ini の **Connect** パラメーター、または ODBC の SQLConnect 機能の connect ストリングを使用して構成する方法について説明します。

これらのタイムアウト値は、そのサーバーと、クライアント・アプリケーション (solidDB SQL エディター (solsql)、solidDB リモート制御テレタイプ (solcon)、および HA マネージャーなど) の接続に適用されます。

アプリケーション読み取りタイムアウト・オプション

このオプションは、ロー・レベルのネットワーク RPC 読み取り操作での障害を検出します。そのタイムアウト設定は、TCP/IP プロトコル用にのみ機能する物理ネットワーク内での読み取りに適用されます。

この RPC 読み取りタイムアウト (ODBC および JDBC で接続タイムアウトと呼ばれます) の値は、以下のオプションを使用して、接続ストリングと listen ストリングの中で構成されます。

```
TCP
  -rnumber_of_milliseconds
  [machine_name]
  port_number
```

これを solid.ini ファイル内で指定するには、solid.ini の [Com] セクションで **Connect** パラメーターを使用します。以下に例を示します。

```
[Com] ;Set RPC read timeout to 1000
      milliseconds (one second) Connect=TCP -r1000 1313
```

RPC 読み取りタイムアウトのデフォルト値は、60000 ミリ秒 (60 秒) です。値ゼロ (0) は、無期限のタイムアウトを設定します。

ODBC を使用している場合は、SQLConnect 機能の接続ストリングの中でタイムアウトの設定を指定します。以下に例を示します。

```
SQLConnect (hdbc, "TCP -r1000
1313", SQL_NTS, "dba", SQL_NTS, "dba", SQL_NTS);
```

上記の例では、定数 SQL_NTS は前のストリング (servername、username、または password) が標準のヌル終了ストリングとして渡されたことを示しています。

注:

ウォッチドッグなどのクライアント・アプリケーションでは、RPC 読み取りタイムアウト (接続タイムアウトとも呼ばれます) を、接続パラメーターの中で **-r** オプションを使用して指定すると便利です。そうしなかった場合、特定のネットワーク障害タイプで無期限の待機が起きることがあります。

注:

[Com] セクション、[Watchdog] セクション、および [Hotstandby] セクション内の Connect パラメーターは、すべて目的が異なります。必ず、正しいものを編集してください。

接続パラメーターでの **-C** オプションの指定

接続タイムアウト (ログイン・タイムアウトとも呼ばれます) の値は、solid.ini ファイルの [Com] セクションおよび [Watchdog] セクションで使用される **Connect** パラメーターの中で指定できます。この接続タイムアウトは、TCP/IP プロトコルについてのみ機能します。

構文は以下のとおりです。

```
Parameter = tcp -cnumber-of-milliseconds
            [machine name]
            port_number
```

ここで、Parameter は **Connect** または **Listen** です。

接続タイムアウトに値を指定しなかった場合、サーバーはオペレーティング・システム固有のデフォルト値を使用します。

注:

ウォッチドッグなどのクライアント・アプリケーションでは、接続タイムアウト値を、Connect パラメーターの中で -c オプションを使用して指定すると便利です。そうしなかった場合、特定のネットワーク障害タイプで、障害が検出される前に無期限の待機が起きることがあります。

以下に例を示します。

アプリケーション・ノード:

```
[Com] ;The server listens to port 1320, and the Connection timeout  
is 1000 ms. Listen = tcpip -c1000 1320
```

トランザクション持続性

DurabilityLevel

パラメーター **DurabilityLevel** は、HotStandby と非 HotStandby の両方のサーバーに適用されます。このパラメーターには 3 つの値があり、リラックス持続性、アダプティブ持続性、ストリクト持続性にそれぞれ対応します。

アダプティブ持続性は、HotStandby でのみ使用されます。アダプティブ持続性の意味は、以下のとおりです。

- 1 次サーバーと 2 次サーバーが接続されていて、正常に機能している場合 (それぞれ、PRIMARY ACTIVE 状態と SECONDARY ACTIVE 状態にある場合)、サーバーはリラックス持続性を使用します。
- それ以外のすべての状態 (例えば、PRIMARY ALONE、STANDALONE など) では、サーバーはストリクト持続性を使用します。

「ストリクト」と「リラックス」の持続性の相違点に関する説明、または **DurabilityLevel** パラメーターの詳細については、「*solidDB 管理者ガイド*」を参照してください。

HotStandby 固有のパラメーターの構成

1 次サーバーと 2 次サーバーの両方のノードで、solid.ini ファイルに、HotStandby 固有の構成パラメーターを指定した [HotStandby] セクションが含まれています。

1 次および 2 次 HotStandby 構成の定義

HotStandby を使用可能にするために設定する必要がある solid.ini 構成パラメーターの最小セットは、以下のとおりです。

- **HSBEnabled**。このパラメーターは HSB をオンまたはオフにします。
- **Connect**。このパラメーターは、1 次サーバーまたは 2 次サーバーを定義するために使用するネットワーク名を定義します。このネットワーク名は、1 次サーバーが 2 次サーバーに (またはその逆に) 接続するために使用するプロトコルとサーバーの名前です。(厳密に言えば、**Connect** パラメーターは solid.ini ファイル内になくてもかまいません。このパラメーターがなくてもサーバーを始動でき、その後、ADMIN COMMAND を使用して Connect ストリングを指定できます。Connect ストリングが設定されていない場合、サーバーは接続を必要としない状態 (例えば、PRIMARY ALONE、SECONDARY ALONE、および STANDALONE) でのみ実行できます。

- **LogEnabled**. このパラメーターを設定する場合は、「yes」に設定してください。このパラメーターは、solid.ini ファイルの [HotStandby] セクションではなく、[Logging] セクションにあることに注意してください。

注:

solid.ini ファイルに [HotStandby] セクションが含まれていないか、[HotStandby] セクションに HSBEnabled=Yes が含まれていない場合、サーバーは非 HotStandby サーバーとして始動し、HotStandby レプリケーションは使用されません。

solid.ini ファイルの一部

ノード 1

```
[HotStandby] HSBEnabled = yes ;The server connects to the opposite server
;using the following connect string.
Connect = tcp machine2 1321 [Logging] LogEnabled=yes
```

ノード 2

```
[HotStandby] HSBEnabled = yes ;The server connects to the opposite server
;using the following connect string.
Connect = tcp machine1 1320 [Logging] LogEnabled=yes
```

切断された接続または使用不可の接続の検出に役立つ、HotStandby サーバー待ち時間の設定

HotStandby サーバーは、タイムアウト・パラメーターを使用して、既存の接続が切断されたか新規接続が確立されなかったと結論付けるまでの待ち時間を制御します。

このタスクについて

タイムアウト・パラメーターは、以下のとおりです。

- **HotStandby.PingTimeout**
- **HotStandby.PingInterval**
- **HotStandby.ConnectTimeout**

PRIMARY ACTIVE 状態または SECONDARY ACTIVE 状態の HotStandby サーバーは、もう一方のサーバーに接続しようとして、指定された時間内に応答を受信しなかった場合、PRIMARY UNCERTAIN、PRIMARY ALONE、または SECONDARY ALONE に変わります。

サーバーの待ち時間を制御するために、以下のことができます。

- **PingTimeout** パラメーターを設定して、サーバーが PRIMARY UNCERTAIN 状態に変わるまでの待ち時間を指定できます。
- **PingInterval** パラメーターを設定して、サーバーが正常であることを示す「ping」メッセージの送信間隔を指定できます。
- **ConnectTimeout** パラメーターを設定して、サーバーが (例えば、ADMIN COMMAND 'hotstandby connect' 操作などで) もう一方のサーバーに対して新規接続を確立しようとするときの、待ち時間を指定できます。します。

PingTimeout および PingInterval パラメーター [HotStandby]

[HotStandby] セクション内のオプションの **PingTimeout** パラメーターおよび **PingInterval** パラメーターは、ping 操作を制御します。

「ping」操作は、基本的に 1 つのデータベース・サーバーから別のデータベース・サーバーへ送信される「私は生きています」メッセージです。(一部のネットワーキング・ソフトウェアにも「ping」操作がありますが、[HotStandby] セクション内の **solidDBPingTimeout** 構成パラメーターは、solidDB サーバーの ping にのみ適用され、一般的なネットワーク ping には適用されません。)言い換えると、これは受動的なハートビート・システムを指しています。このパラメーターを設定した場合、1 次と 2 次の両方の HotStandby サーバーは定期的な間隔で、互いに「ping」メッセージを送信します。4 ページの『ハートビート』も参照してください。

- **PingTimeout** は、もう一方のサーバーがダウンしているかアクセス不能であると結論付けるまでのサーバーの待ち時間を指定します。デフォルトは 4000 (4 秒) です。
- **PingInterval** は、2 回の ping の間の間隔をミリ秒単位で指定します。デフォルトは 1000 (1 秒) です。

例えば、**PingInterval** が 10 秒の場合、各サーバーは互いに 10 秒ごとに ping メッセージを送信します。**PingTimeout** が 20 秒で、一方のサーバー (S1) が 20 秒以内に、もう一方 (S2) からの応答を聴かなかった場合、S1 は S2 がダウンしているかアクセス不能であると結論付けます。その後、サーバー S1 は別の状態に (例えば「PRIMARY ACTIVE」から「PRIMARY UNCERTAIN」に) 切り替わります。

上記のパラメーターの値が異なる場合は、1 次サーバー内で「hsb connect」コマンドの実行中に設定された値が優先されます。値は、切り替え時に変更されません。しかし、ADMIN COMMAND「パラメーター」コマンドで動的に変更できます。

PingTimeout をゼロに設定した場合、ping は使用不可になります。

ping は、ほとんどオーバーヘッドを必要とせず、solidDB サーバーは、欠落した ping メッセージに迅速に対応するようセットアップされています。**PingInterval** 値は、1 秒またはそれ以下のかかなり短い間隔に設定できます。

サーバーに障害が起きたときに、フェイルオーバーを素早く検出することが重要な場合は、**PingTimeout** 値を比較的短い時間に設定します。ただし、値が小さいほど、「誤報」が発生する可能性も高くなります。ネットワークに多量のトラフィックがあり、そのために ping メッセージを受信するまでに遅延が生じる場合は、誤報を避けるために、**PingTimeout** に大きな値を設定することが必要になる場合もあります。

ConnectTimeout パラメーター [HotStandby]

一部のネットワーク・インプリメンテーションでは、接続操作が不定の期間、応答しないことがあります。考えられる理由の 1 つは、リモート・マシンが既知のノードであっても、接続の試行中に使用不可であることです。接続タイムアウト値を指定することにより、HotStandby 接続操作でリモート・マシンに接続するまでの最大待ち時間を、秒単位で設定できます。

ConnectTimeout パラメーター (特定のプラットフォーム上でのみ役立つ) は特定の管理コマンドでのみ使用されます。それらのコマンドは、以下のとおりです。

- hotstandby connect
- hotstandby switch primary
- hotstandby switch secondary

接続タイムアウト値は、solid.ini ファイルの [HotStandby] セクションで、**ConnectTimeout** パラメーターを使用して、ミリ秒単位で設定します。単位はミリ秒です。デフォルトは 0 で、これはタイムアウトなしを意味します。これは、別の値に設定できます。以下に例を示します。

```
[HotStandby]
; Set ConnectTimeout to 20 seconds (20000 milliseconds).
ConnectTimeout=20000
```

HotStandby データベース・コピー操作の名前と場所の定義

[HotStandby] セクション内のオプションの **CopyDirectory** パラメーターは、HotStandby コピー操作のコピー先となるディレクトリーの名前と場所を定義します。

HotStandby コピー操作は、以下のコマンドで指定します。

```
ADMIN COMMAND 'hotstandby copy [directory_name]';
```

このパラメーターにはデフォルト値がないので、solid.ini ファイル内でディレクトリーを指定しなかった場合は、コピー・コマンドの中でそれを指定する必要があります。**CopyDirectory** パラメーターに相対パスを指定する場合は、1 次サーバーの solid.ini ファイルを保持するディレクトリーからの相対パスにします。

このパラメーターは、ADMIN COMMAND 'hotstandby netcopy' コマンドを使用して HotStandby データベース・コピー操作を行う場合は、必要ありません。これら 2 つの選択肢のうち、推奨されるのは、機能がより柔軟な netcopy コマンドです。

2 次サーバー障害時の 1 次サーバーの動作の定義

[HotStandby] セクションの **AutoPrimaryAlone** パラメーターを使用して、2 次サーバーとの接続が失われた後の 1 次サーバーを PRIMARY ALONE 状態に自動的に切り替えるか、それとも PRIMARY UNCERTAIN 状態に留めるかを制御できます。

AutoPrimaryAlone が Yes に設定されている場合、1 次サーバーは 2 次サーバーとの接続が失われたとき、PRIMARY ALONE 状態に自動的に切り替わります。これにより、1 次サーバーはトランザクションの受け入れを続行できます。

AutoPrimaryAlone が No に設定されている場合、2 次サーバーとの接続が失われた 1 次サーバーは、PRIMARY UNCERTAIN 状態に自動的に切り替わります。

デフォルトでは、**AutoPrimaryAlone** は No に設定されます。

```
[HotStandby]
AutoPrimaryAlone = No
```

PRIMARY UNCERTAIN 状態では、1 次サーバーは新規トランザクションの受け入れや現在アクティブなトランザクションのコミットを行うことができません。1 次サーバーは、HAC、Watchdog、またはシステム管理者が指示するまで、PRIMARY ALONE 状態に切り替わりません。

AutoPrimaryAlone が No に設定されている場合は、ADMIN COMMAND 'hotstandby set primary alone' コマンドを実行することにより、サーバーを PRIMARY ALONE 状態に設定できます。このコマンドは、構成ファイル内の **AutoPrimaryAlone** の値を変更しないことに注意してください。

デフォルトを Yes に変更した場合、1 次サーバーの状態は PRIMARY ACTIVE から PRIMARY ALONE に変更され、PRIMARY UNCERTAIN には変更されません。

パフォーマンス・チューニング

安全性レベルと持続性レベルによるレプリケーション・パフォーマンスのチューニング

通常操作時のデータ・レプリケーションのパフォーマンスは、持続性レベルと安全性レベルの設定によって異なります。また、2-safe レプリケーションを使用した場合、2-safe モードで使用される確認応答ポリシーは、アプリケーションによって感知される待ち時間に影響します。詳細については、15 ページの『HotStandby によるパフォーマンスへの影響』を参照してください。

ネットコピーのパフォーマンスのチューニング (General セクション)

hotstandby netcopy コマンドを使用すると、1 次サーバーのデータベースをリモート 2 次サーバーにコピーできます。このコマンドは、一方または両方のサーバーがディスクレスであるときに、データベースを 1 次サーバーから 2 次サーバーにコピーするためにも使用されます。ネットコピー用に 2 次サーバーに接続するために使用する接続ストリングは、solid.ini の [HotStandby] セクションで指定されます。

1 次サーバーのデータベース・ファイルは、ネットワーク・リンクを通じてコピーされます。netcopy の詳細については、91 ページの『ネットワークを介した 1 次データベースの 2 次側へのコピー』を参照してください。

solid.ini ファイルの [General] セクションにある以下のパラメーターは、ネットコピーのパフォーマンスを向上させる効果があります。

BackupBlockSize パラメーター [General]

solid.ini ファイルの [General] セクション内の **BackupBlockSize** パラメーターは、1 次サーバーのデータベース・ファイルを 2 次サーバーにコピーするときのブロック・サイズを増減することにより、ネットコピーのパフォーマンスを (また、当然ながらバックアップのパフォーマンスも) チューニングするために使用されます。一般的なルールとして、ブロック・サイズが大きいほどネットコピー/バックアップは高速になりますが、その代償として、ネットコピー/バックアップが行われている間、他の要求に対するサーバーの応答時間が低下する可能性があります。

デフォルトでは、**BackupBlockSize** パラメーターは 64K に設定されます。これは、別の値に設定できます。以下に例を示します。

```
[General]
BackupBlockSize = 32K
```

または

```
[General]
BackupBlockSize = 32768
```

BackupBlockSize の最小値はサーバーのブロック・サイズで、最大値は、現時点では 1GB であることに注意してください (「M」および「K」の接尾部がサポートされます。例えば、32K や 1M などです)。**BackupBlockSize** の値は、サーバーのデータベース・ブロック・サイズの倍数にしてください。

データベース・キャッチアップのパフォーマンスのチューニング

HotStandby では、障害を起こした 2 次サーバーがサービスに復帰し、1 次サーバーに接続した場合、「HotStandby データベース・キャッチアップ」と呼ばれる自動プロセスにより、HotStandby トランザクション・ログ・ファイルの内容が 1 次サーバーから 2 次サーバー・ノードへ引き続き送信されます。

solid.ini ファイルの [HotStandby] セクション内の **CatchupSpeedRate** パラメーターは、キャッチアップと現在のクライアント・データベース照会に対するサービスに使用するサーバーの時間を調整することにより、データベース・キャッチアップのパフォーマンスをチューニングするために使用されます。

CatchupSpeedRate に値 90 を割り当てた場合、サーバーはその時間の約 90% をキャッチアップに使用し、約 10% の時間をユーザーの照会に対する応答に使用します。以下に例を示します。

```
[HotStandby]
CatchupSpeedRate = 50
```

この数値が大きいくほど、高速なキャッチアップが実行されますが、ユーザー照会など、他のアクティビティに及ぼす影響が大きくなります。デフォルトでは、**CatchupSpeedRate** は 70 に設定されます。

HA コントローラーおよび HA マネージャーの構成

高可用性コントローラー (HAC) は、それぞれの HotStandby サーバー・ノードにデプロイされます。HAC 構成ファイル solidhac.ini は、HAC 作業ディレクトリーに配置する必要があります。

solidhac.ini 構成ファイル内のパラメーターは、以下のセクションの下にグループ化されています。

- HACController
- LocalDB
- RemoteDB
- ERE

すべての構成パラメーターは、56 ページの『solidhac.ini 構成ファイル』の solidhac.ini のファイル例の中に示されています。

HAController セクション

HAController セクションのパラメーターを以下に説明します。

- **Listen**

Listen パラメーターの値は、HAC がクライアント通信に使用するプロトコルとポートを指定します。HAC は開始時に、そのポートに対する **listen** を開始します。**listen** を開始できない場合、例えば、ポートが別のプロセスに使用されている場合は、HAC の終了直後に、情報がログ・ファイル (hacmsg.out) に書き込まれます。サポートされているプロトコルは、TCP/IP ('tcp') だけです。HAC の **listen** ポートは、HA マネージャーが HAC と通信するために使用します。また、HAC のシャットダウン・コマンドも、このポートへ送られます。

Listen は、必須パラメーターです。

- **StartInAutomaticMode**

HAC は、起動した後、AUTOMATIC と ADMINISTRATIVE のどちらかのモードになります。AUTOMATIC モードでは、HAC は自動的に可用性の最大化を試み、サーバーの HSB 状態を変更し、必要な場合はサーバー・プロセスを再開します。これに対し、ADMINISTRATIVE モードでは、HAC はサーバーの正常性のモニターだけを行います。**StartInAutomaticMode** は、HAC の初期モードを指定します。

StartInAutomaticMode のデフォルト値は 'Yes' です。

- **EnableDBProcessControl**

EnableDBProcessControl=Yes を設定すると HAC でローカル・サーバー・プロセスを管理できます。これにより、サーバーを自動的に始動でき、ユーザーにはデータベース・プロセスのシャットダウンと再開を行うコマンドが提供されます。

EnableDBProcessControl=Yes を設定した場合は、必ず **StartScript** パラメーターを使用する必要があります。

デフォルト値は 'No' です。

- **EnableAutoNetcopy**

EnableAutoNetcopy=Yes を設定すると、hsb connect コマンドで HSB リンクを確立できないときに、HAC でネットコピーを開始することができます。

EnableAutoNetcopy のデフォルト値は 'Yes' です。

- **RequiredConnectFailures**

サーバー状態が不明の場合、または HAC が何らかの理由でサーバー状態を確認する必要がある場合は、非ブロッキング SQLConnect (check) コマンドが使用されます。そのようなケースで非ブロッキング SQLConnects の実行が失敗した場合、問題のサーバーが無応答と見なされるまで、実行が複数回繰り返されます。接続を連続して試みる最大回数は、**RequiredConnectFailures** の値に相当します。

RequiredConnectFailures のデフォルト値は 2 です。

- **CheckTimeout**

連続した非ブロッキング `SQLConnect` コマンドのタイムアウト (ミリ秒単位)。値が小さすぎると、「誤判定」が起きることがあります。つまり、サーバーは稼働しているのに、タイムアウト期間内に応答できなかったために、障害を起こしているように見えます。

`CheckTimeout` のデフォルト値は 150 (ミリ秒) です。

- **CheckInterval**

連続した非ブロッキング `SQLConnect` コマンドの間隔。この値は、フェイルオーバー時間に影響しません。一般に、検査 (ポーリング) は障害の後に行われるか、システムの始動中に行われます。

`CheckInterval` のデフォルト値は、1000 (ミリ秒) です。

- **Username**

HAC ユーザー名。

`Username` は、必須パラメーターです。

- **Password**

`Username` を使用して識別されたユーザーのパスワード。

`Password` は、必須パラメーターです。

- **DBUsername**

ローカル HSB サーバーに接続するために使用される、HAC のユーザー名。データベース・ユーザーは、`SYS_ADMIN_ROLE` または `SYS_CONSOLE_ROLE` を持っている必要があります。

`DBUsername` は、必須パラメーターです。

- **DBPassword**

`DBUsername` を使用して識別されたユーザーのパスワード。

`DBPassword` は、必須パラメーターです。

LocalDB セクション

LocalDB セクションのパラメーターを以下に説明します。

- **Connect**

ローカル・データベース・サーバーの接続情報。HAC はローカル・サーバーに接続するときに、この情報を使用します。

`Connect` は、必須パラメーターです。

- **StartScript**

`StartScript` は、HAC がデータベースを制御するように構成されている場合 (`EnableDBProcessControl=Yes`)、必須パラメーターです。この値は、`EnableDBProcessControl=No` の場合には効果がありません。

- **PreferredPrimary**

PreferredPrimary=Yes を設定すると、どちらのサーバーも 1 次サーバーとして始動できる場合、ローカル HSB サーバーが 1 次サーバーに移行します。両方のサーバーが **PreferredPrimary** に同じ値を持っている場合は、'first' サーバーが新規 1 次サーバーになります。

PreferredPrimary のデフォルト値は 'No' です。

RemoteDB セクション

RemoteDB セクションのパラメーターを以下に説明します。

- **Connect**

リモート・データベース・サーバーの接続情報。HAC はリモート・ノードのサーバーに接続するときに、この情報を使用します。

Connect は、必須パラメーターです。

ERE セクション

ERE セクションのパラメーターを以下に説明します。

- **EREIP**

ERE の IP アドレス。

- **RequiredPingFailures**

サーバーが ERE から切断されていると HAC が結論付けるまでに、失敗が許される連続した ping 呼び出しの回数。

RequiredPingFailures のデフォルト値は 3 です。

構成ファイルの例

以下のセクションでは、solidDB に関連するさまざまな構成ファイルの例を示します。

solid.ini 構成ファイル

以下は、最初の HotStandby サーバー用の solidDB 構成ファイル (solid.ini) の例 (抜粋) です。

```
[Com]
; The first server listens to the network with this
; name
Listen = tcp 1320
[HotStandby]
HSBEnabled=yes
; The first server connects to the second server
; using the following connect string.
Connect = tcp 188.177.166.12 1321
AutoPrimaryAlone=No
[Logging]
LogEnabled=yes
```

以下は、2 番目の HotStandby サーバー用の solidDB 構成ファイル (solid.ini) の例 (抜粋) です。

```
[Com]
; The second server listens to the network using the following
; connect string.
Listen = tcp 1321
[HotStandby]
HSBEnabled=yes
; The second server connects to the first server
; using the following connect string.
Connect = tcp 188.177.166.11 1320
AutoPrimaryAlone=No
[Logging]
LogEnabled=yes
```

solidhac.ini 構成ファイル

以下は高可用性コントローラー構成ファイル (solidhac.ini) の例 (抜粋) です。

```
=====
; NOTE : Copy this file as solidhac.ini
;       to solidhac working directory
;
; solidDB High Availability Controller inifile
=====

[HAController]
;** HAC connect info
;** HAC clients, HA Manager, for example, use this information.
;** Mandatory
;** Listen=tcp 3135
Listen=

;** Setting StartInAutomaticMode=Yes starts HAC in AUTOMATIC mode.
;** In AUTOMATIC mode, solidhac automatically tries
;** to maximize the availability by changing the HSB states of the
;** server, and restarting the server processes when necessary.
;** In contrast, it can be in ADMINISTRATIVE mode
;** in which HAC only monitors the health of the servers.
;**
;** This is dynamically changeable parameter.
;** Optional
;** Values : Yes/No, default = Yes
StartInAutomaticMode=

;** Setting EnableDBProcessControl=Yes allows solidhac
;** manage local db process by automatically starting
;** the db, and by providing the user with commands to
;** shutdown and restart db process.
;**
;** Optional
;** Effective only when HAC is in AUTOMATIC mode.
;** Values : Yes/No, default = No
EnableDBProcessControl=

;** Setting EnableAutoNetcopy=Yes allows solidhac to initiate
;** netcopy when HSB link cannot be established with 'hsb connect'.
;**
;** Optional
;** Effective only when HAC is in AUTOMATIC mode.
;** Values : Yes/No, default = Yes
EnableAutoNetcopy=

;** When server state is unknown, or HAC needs, for some other reason, to
;** ensure the state of server, non-blocking SQLConnect command is used.
;** If the execution of non-blocking SQLConnect in such a case fails,
```



```

;** it is repeated multiple (RequiredConnectFailures) times before
;** the server in question is considered as non-responsive.
;**
;** Optional
;** Values : 1..n, default=2
RequiredConnectFailures=

;** Timeout in milliseconds for non-blocking SQLConnect commands.
;** Too short interval can cause 'false positives', server seems
;** to be failed because it wasn't able to respond within the timeout period.
;**
;** Optional
;** Values : 1..n, default=150 (milliseconds)
CheckTimeout=

;** Interval between consecutive non-blocking SQLConnect commands.
;** The value doesn't affect on failover time. Checking (polling)
;** takes place typically after failure, or during system startup.
;**
;** Optional
;** default = 1000 (milliseconds)
CheckInterval=

;** HAC user identification
;** Mandatory
Username=
Password=

;** HSB server user identification
;** Mandatory
DBUsername=
DBPassword=

[LocalDB]
;** soliddb connect info
;** Mandatory
;** Connect=tcp 2125
Connect=

;** The name of the script, which is used to initiate the db process.
;**
;** Optional, except if HAC controls db process (EnableDBProcessControl=Yes).
;** Value is not effective if EnableHACActions=No or EnableDBProcessControl=No
;** StartScript=/home/solid/start_solid.sh
StartScript=

;** Setting PreferredPrimary=Yes moves local HSB server to as Primary
;** in the case where either of the servers could start as Primary.
;** If both servers have PreferredPrimary=No, or no value, first
;** (new Primary) server wins.
;**
;** Optional
;** Value is not effective if EnableHACActions=No.
;** Values : Yes/No, default no
PreferredPrimary=

[RemoteDB]
;** soliddb connect info
;** Mandatory
;** Connect=tcp 192.168.3.123 2125
Connect=

[ERE]
;** IP address of an ERE

```

```

;** Optional
;** Connect=192.168.3.1
EREIP=

;** The number of consecutive ping calls that must
;** fail before HAC concludes that the server is
;** disconnected from ERE.
;**
;** Optional
;** Values : 1..n, default=3
;** RequiredPingFailures=10
RequiredPingFailures=

```

HAManager.ini 構成ファイル

以下は、高可用性マネージャー構成ファイル (HAManager.ini) の例 (抜粋) です。

```

=====
;solidDB High-Availability Manager
; Configuration file HAManager.ini
; V. 0.3
; 2008-21-01
=====
;** HA Controller connect info, e.g.
;Server1_name = Server 1
;Server1_host = node1.acme.com
;Server1_port = 2220
;Server2_name = Server 2
;Server2_host = node2.acme.com
;Server2_port = 2220

; All the following lines are mandatory.
Window_title = HA Manager
Header_text = SolidDB HA Manager

; Display names, host addresses and port numbers
; of the SolidHAC (HA Controllers) instances
;Server 1 HA Controller
;-----
Server1_name = Server 1
Server1_host = localhost
Server1_port = 1234
Server1_user = foo
Server1_pass = bar
;
;Server 2 HA Controller
;-----
Server2_name = Server 2
Server2_host = 192.168.0.1
Server2_port = 1234
Server2_user = foo
Server2_pass = bar

```

4 アプリケーションでの HotStandby の使用

この章では、HotStandby 構成において、障害と切り替えをアプリケーションでどのように処理すればよいかについて説明します。

HotStandby サーバーに接続する 2 とおりの方法

HotStandby 環境でアプリケーションをプログラムするために、2 とおりの方法があります。クライアントがそれぞれの HSB サーバーに明示的に接続する必要がある「基本接続」のほかに、クライアントが TC 接続と呼ばれる論理接続を 1 つだけ規定する「透過接続」(TC) が提供されます。

どちらの接続タイプも、solidDB ODBC ドライバーおよび JDBC ドライバーでサポートされています。接続タイプは、汎用接続ストリング (データ・ソース情報) を適宜に構築することによって選択されます。

基本接続

基本接続では、HotStandby 構成の各サーバーへの接続を、アプリケーションが特定のサーバー・アドレスを使用して別々に処理する必要があります。フェイルオーバーが発生した場合、アクティブ接続は失われ、アプリケーションは新規 1 次サーバーに再接続する必要があります。

透過接続

透過接続を使用すると、アプリケーションは特定のサーバーへの接続を処理する必要がなく、フェイルオーバーの場合でも再接続の処理を行う必要がありません。アプリケーションは、TC 接続 と呼ばれる論理接続 (ハンドル) を保守します。

接続ハンドルは、指定されたサーバー・セット内に PRIMARY ACTIVE、PRIMARY ALONE、STANDALONE のいずれかの状態にあるサーバーが 1 つでもある限り、フェイルオーバーや切り替えが何回起きても維持されます。フェイルオーバーおよび切り替えのとき、ドライバーは接続切り替え と呼ばれる内部操作を実行します。アプリケーションは、(障害透過性レベルに応じて) セッション状態を再構成しなければならない場合もあるので、接続切り替えに関する通知を受けます。TC を使用した場合は、1 次サーバーと 2 次サーバーの間で読み取り専用の負荷のバランスをとることができます。つまり、透過接続は、アプリケーションが多様なサーバーとそのアドレスについての処理を行わなくても済むようにします。

重要:

solsql などの solidDB ツールは、TC 接続をサポートしていません。

接続タイプの選択

以下の互換性マトリックスは、選択された接続情報に対してサポートされている機能を示すことにより、接続タイプの選択に役立ちます。

表 7. 接続タイプの選択

機能	スタンドアロン構成	HSB 構成
基本接続	可 (BC 情報)	可 (BC 情報)
透過的フェイルオーバー	不可	可 (TC 情報)
ロード・バランシング	不可	可 (TC 情報)

透過接続の使用

solidDB 透過接続を使用した場合、このドライバーは 2 つの HSB サーバーの存在を、ある程度、アプリケーションに対して隠蔽します。

このドライバーは、内部のアクティブ接続にマップされる単一の論理 TC 接続を提供します。理想的な事例では、1 次と 2 次の両方のサーバーがアクティブ状態で稼働しているとき、ドライバーはスタンバイ接続 (つまり 2 次サーバーへの接続) も維持します。この接続はイベント待機モードに設定され、HSB の状態変更イベントをいつでも受信できる状態になります。それらのイベントは、フェイルオーバーと切り替えに関してドライバーが使用する基本的な情報源です。スタンバイ接続は (例えば、Primary Alone 動作の場合などに) 失われることがあります。ドライバーは可能な限りそれを実現しようとします。スタンバイ接続は、クライアントに対して完全に透過的に処理されます。その一方で、接続切り替え (つまり、TC 接続から内部アクティブ接続へのマッピングの変更) が発生すると、それが特殊なエラー・コードによってクライアントへ通知されます。

重要:

solsql などの IBM solidDB ツールは、TC 接続をサポートしていません。

TC での障害透過性

障害透過性は、障害のマスキングを処理します。障害透過性レベルは、TC 情報の TF_LEVEL 属性で設定されます。

次の 3 つのレベルがあります。

1. NONE は、障害透過性を無効にします。これはデフォルト値です。
2. CONNECTION は、サーバー接続を保存します。つまり、フェイルオーバーまたは切り替えの場合に再接続を不要にします。
3. SESSION は、デフォルト以外の値を持つ大部分のセッション属性を保存します。さらに、準備済みステートメントも保存されます。ただし、オープン・カーソルはクローズされ、進行中のトランザクションは打ち切られます。保存されるセッション属性のリストについては、69 ページの『接続切り替えのためのプログラミング』を参照してください。

透過接続でのロード・バランシング

透過接続ドライバーは、2 つの方式を使用してトランザクションの負荷を方向付けます。1 つは読み取り集中負荷を処理する方式で、もう 1 つは書き込み集中負荷を処理する方式です。ロード・バランシングの目的で、論理 TC 接続は「ワークロード接続」と呼ばれる低い方のレベルのサーバー接続にマップされます。ワークロー

ド接続は、時間が経つと変化する場合があります、通常はアプリケーションの関心事ではありません。しかし、これが必要な場合は、何が現行のワークロード接続であるかを知るための方法があります。

静的ロード・バランシング構成

ロード・バランシング方式は、以下のとおりです。

1.

PREFERRED_ACCESS=READ_MOSTLY。この方式は、読み取り集中負荷を処理するためのものです。読み取り専用トランザクションは、2 次サーバーと 1 次サーバーの両方で実行できます。パラメーター

Cluster.ReadMostlyLoadPercentAtPrimary がゼロに設定された場合、読み取り専用負荷は、すべて 2 次サーバーで実行されます。

2.

PREFERRED_ACCESS=WRITE_MOSTLY。これはデフォルト値です。この方式は、書き込み集中負荷を処理するためのものです。すべてのトランザクションは、1 次サーバーで実行されます。これは、一般的な HotStandby 操作に対応しています。

PREFERRED_ACCESS=READ_MOSTLY に設定した場合、1 次サーバーはドライバーに、ワークロード接続用の接続先サーバーを指示します。負荷が 2 次サーバーへ誘導される場合、書き込み操作が発行されると 1 次サーバーへの引き渡しが行われ、トランザクションは 1 次サーバーで実行されます。トランザクションのコミット後、負荷は再び 2 次サーバーへ誘導されます。2 次サーバーに障害が起きると、接続は 2 次サーバーから 1 次サーバーへのフェイルオーバーが行われます。

さらに、新しいロード・バランシング構成パラメーターが導入されました。これを使用すると、ある一定量の読み取り専用負荷を 1 次サーバーへ誘導できます。

```
[Cluster]
ReadMostlyLoadPercentAtPrimary=<n>
```

ここで、 $n = [0 \dots 100]$ です。デフォルトは 50 です。

このパラメーターは、1 次サーバーへ誘導される、大部分が読み取りの負荷の合計パーセンテージを定義します。この値に基づいて、ワークロード割り当て済みサーバーが選択されます。デフォルトでは、接続の半分が 1 次サーバーを使用し、半分が 2 次サーバーを使用します。これは、ほとんどの混合負荷に望ましい値です。この値をゼロに設定した場合、すべての負荷が 2 次サーバーへ誘導されます。これは、極度の読み取り集中（または読み取り専用）アプリケーションに **PREFERRED_ACCESS=READ_MOSTLY** を使用しており、（それと同時に）書き込み集中アプリケーションに **PREFERRED_ACCESS=WRITE_MOSTLY** を使用している場合に適しています。

注:

ロード・バランシングは、**READ COMMITTED** の分離レベルでのみ機能します。サーバーの分離レベルの（始動）デフォルトが別の値に設定されている場合、**PREFERRED_ACCESS=READ_MOSTLY** を設定すると、そのセッションの分離レベ

ルが強制的に READ COMMITTED にされます。分離レベルは、より高いレベル (REPEATABLE READ など) に動的に再設定できますが、その場合、ロード・バランシングは使用不可になります。

注:

ロード・バランシングは、セッションが自動コミット・モードに設定された場合は、使用不可になります。

ロード・バランシングの動的制御

ワークロード割り当て済みサーバーが 2 次サーバーである場合は、プログラムで 1 次サーバーに変更できます。

セッション・レベルでは、以下のステートメントでワークロード接続サーバーを 1 次サーバーに変更できます。

- SET WRITE
- SET ISOLATION LEVEL REPEATABLE READ
- SET ISOLATION LEVEL SERIALIZABLE

ステートメントは、トランザクションの最初のステートメントの場合、直ちに有効になり、そうでない場合は次のトランザクションから有効になります。

トランザクション・レベルでは、以下のステートメントでワークロード接続サーバーを 1 つのトランザクションの間、1 次サーバーに変更できます。

- SET TRANSACTION WRITE
- SET TRANSACTION ISOLATION LEVEL REPEATABLE READ
- SET TRANSACTION ISOLATION LEVEL SERIALIZABLE

影響を受けるトランザクションは、このステートメントを使用して開始したトランザクションか、それ以外の場合は次のトランザクションです。トランザクションが 1 次サーバーで実行された後、ワークロード接続サーバーはセッション用に割り当てられたサーバーに戻ります。

SET [TRANSACTION] WRITE ステートメントの効果は、ステートメント SET [TRANSACTION] READ WRITE (SQL:1999) で元に戻すことができます。また、以下の分離レベル・ステートメントでも同じ効果があります。

- SET ISOLATION LEVEL READ COMMITTED
- SET TRANSACTION ISOLATION LEVEL READ COMMITTED

ロード・バランシングでのフェイルオーバー透過性

両方の透過的フェイルオーバーが設定されており (TF_LEVEL が NONE 以外)、しかもロード・バランシングが使用可能に設定されている場合 (PREFERRED_ACCESS=READ_MOSTLY)、適用されるフェイルオーバー・ポリシーは以下のとおりです。

1. 1 次サーバー障害: すべての負荷は PRIMARY ALONE 状態にある新規 1 次サーバーへ誘導されます。
2. 2 次サーバー障害: すべての負荷は 1 次サーバー (PRIMARY ALONE) へ誘導されます。

3. サーバー間の接続切断。サーバーは PRIMARY ALONE 状態と SECONDARY ALONE 状態: 2 次サーバー内で読み取り専用トランザクションの実行が進行中である場合、そのトランザクションも 2 次サーバー内で正常にコミットされます。それ以後のすべてのトランザクションは、(PRIMARY ALONE の) 1 次サーバーへ誘導されます。

通常のホット・スタンバイ操作が (PRIMARY ACTIVE 状態と SECONDARY ACTIVE 状態にあるサーバーを使用して) 再開されると、1 次サーバーと 2 次サーバーの間で再び負荷のバランスがとられます。

注:

TF_LEVEL=NONE (障害透過性なし) であっても、ある種の基本的なフェイルオーバー機能が使用可能です (2 次サーバーに障害が起きたときの 2 次サーバーから 1 次サーバーへのフェイルオーバー)。それ以外の障害は、結果として通信リンクの障害になります。したがって、TF_LEVEL=NONE とした場合、ほとんどの障害の事例で、アプリケーションは (同じ TC 情報で) 再接続する必要があります。再接続を回避するために、ロード・バランシングを使用するときは、常に障害透過性を使用可能に設定することをお勧めします。

ロード・バランシング下の実行の手順

すべての SQL ストアード・プロシージャは、プロシージャ宣言の中で SQL 標準節 *SQL Data Access Indication* によって読み取り専用プロシージャとして指定された場合を除き、1 次サーバーで実行されます。

```
<SQL-data-access-indication> ::=  
    NO SQL |  
    READS SQL DATA |  
    CONTAINS SQL |  
    MODIFIES SQL DATA
```

読み取り専用プロシージャおよび機能の不必要な引き渡しを回避するために、以下のいずれかの値を宣言できます。

- NO SQL
- READS SQL DATA
- CONTAINS SQL

MODIFIES SQL DATA (これはデフォルトです) のみがトランザクションの引き渡しを発生させます。

この節は、(オプションの) RETURNS 節とプロシージャ本体の間に置かれます。以下に例を示します。

```
"CREATE PROCEDURE PHONEBOOK_SEARCH  
(IN FIRST_NAME VARCHAR, LAST_NAME VARCHAR)  
RETURNS (PHONE_NR NUMERIC, CITY VARCHAR)  
READS SQL DATA  
BEGIN  
-- procedure_body  
END";
```

透過接続情報の構文

solidDB 透過接続を使用した場合、クライアントは TC 接続と呼ばれる唯一の論理接続を規定します。この接続は、TC 情報の中で指定されます。TC 情報は、透過的フェイルオーバーと両方の HSB 構成のロード・バランシングを規定します。

solidDB TC 情報の完全な構文は、以下のとおりです。

```
<solidDB TC Info> ::= {[<failure transparency level attribute>]  
[<preferred access attribute>] <connect target list>} | <cluster info>
```

```
<failure transparency level attribute> ::= TF_LEVEL={NONE |  
CONNECTION | SESSION}
```

```
<preferred access attribute> ::= PREFERRED_ACCESS={WRITE_MOSTLY |  
READ_MOSTLY}
```

```
<connect target list> ::= [SERVERS=]<connection string>[, <connection string > ...]
```

```
<cluster info> ::= CLUSTER <connect string>[, <connect string>...]
```

さらに、以下の省略形を使用できます。

表 8. TC 情報の省略形

省略形	対応する構文
TF	TF_LEVEL
CO	CONNECTION
SES	SESSION
PA	PREFERRED_ACCESS
RM	READ_MOSTLY
WM	WRITE_MOSTLY
S	SERVERS

障害透過性属性

障害透過性は TF_LEVEL 属性によって表され、障害のマスキングを処理します。次の 3 つのレベルがあります。

1. NONE は、障害透過性を無効にします。これはデフォルト値です。
2. CONNECTION は、サーバー接続を保存します。つまり、フェイルオーバーまたは切り替えの場合に再接続を不要にします。
3. SESSION は、デフォルト以外の値を持つ特定のセッション属性を保存します。さらに、準備済みステートメントも保存されます。ただし、オープン・カーソルはクローズされ、進行中のトランザクションは打ち切られます。

ロード・バランシング属性

優先アクセス属性 (PREFERRED_ACCESS) は、ロード・バランシングを適用するかどうかを示します。次の 2 つのレベルがあります。

1. WRITE_MOSTLY では、ワークロードがすべて 1 次サーバーへ誘導されます。これはデフォルト値です。
2. READ_MOSTLY では、ワークロードが (デフォルトでは) 2 次サーバーへ誘導されます。書き込みトランザクションは 1 次サーバーへ引き渡されます。

最後に、solidDB TC 情報は、サーバー・アドレスのリストを含んでいます。ドライバーはリストを左から右へスキャンして、1 次サーバーと 2 次サーバーを検出しようとします。したがって、望ましい構成をリストの冒頭に配置する必要があります。リストの残りの部分には、システムの存続期間中、いつかアクティブにされる可能性があるスペア・アドレスを含めることができます。リストは常に短くしておいてください。その理由は、エラー状態で、エラーがアプリケーションに返されるまでに長い時間を要する可能性があるからです。各アドレスは 1 つずつ試行され、その際、指定されたログイン・タイムアウトも関与します。リスト内のアドレスの数は無制限です。

属性 TF_LEVEL と PREFERRED_ACCESS のどれも指定されなかった場合 (または TF_LEVEL=NONE)、接続動作は基本接続にフォールバックします。複数の接続ストリングを指定した場合は、接続要求を受け入れるリスト上の最初のサーバーに対して接続が確立されます。

CLUSTER キーワードは、以下のシノニムとして使用できます。

```
TF_LEVEL=SESSION PREFERRED_ACCESS=READ_MOSTLY SERVERS=
```

例として、以下の TC 情報があるとします。

```
F_LEVEL=SESSION PREFERRED_ACCESS=READ_MOSTLY  
SERVERS=tcp srv1.acme.com 1315, tcp srv2.acme.com 1315
```

これは、以下のように置き換えることができます。

```
CLUSTER tcp srv1.acme.com 1315, tcp srv2.acme.com 1315
```

クライアント・サイド INI ファイル

```
[Data Sources]  
Cluster1=  
  TF_LEVEL=SESSION  
  PREFERRED_ACCESS=READ_MOSTLY  
  SERVERS=  
    tcp -c 1000 srv1.dom.acme.com 1315,  
    tcp srv2.dom.acme.com 1315,  
    tcp srv3.dom.acme.com 1316
```

ODBC での Connect ストリング

```
rc = SQLConnect(comHandle, "CLUSTER  
tcp -c 1000 srv1.dom.acme.com 1315,  
tcp srv2.dom.acme.com 1315,  
tcp srv3.dom.acme.com 1316", ...
```

TC 情報の属性の組み合わせ

以下の表は、TC 情報の属性の可能な組み合わせを要約したもので、結果としての接続機能も示しています。

表 9. TC 情報の属性の可能な組み合わせ

PREFERRED_ACCESS:	TF_LEVEL: 指定されないか NONE	TF_LEVEL: CONNECTION	TF_LEVEL: SESSION
指定されない	<ul style="list-style-type: none"> フェイルオーバーまたは切り替えはサポートされない ロード・バランシングなし (基本接続)	<ul style="list-style-type: none"> 透過的フェイルオーバー (セッション状態は保存されない) 透過的な切り替え 1 次サーバーのみでのワークロード ロード・バランシングなし 	<ul style="list-style-type: none"> 透過的フェイルオーバー (セッション状態は保存される) 透過的な切り替え 1 次サーバーのみでのワークロード ロード・バランシングなし
WRITE_MOSTLY (デフォルト)	<ul style="list-style-type: none"> 透過的フェイルオーバーのサポートなし 透過的な切り替え 1 次サーバーのみでのワークロード ロード・バランシングなし 	<ul style="list-style-type: none"> 透過的フェイルオーバー (セッション状態は保存されない) 透過的な切り替え 1 次サーバーのみでのワークロード ロード・バランシングなし 	<ul style="list-style-type: none"> 透過的フェイルオーバー (セッション状態は保存される) 透過的な切り替え 1 次サーバーのみでのワークロード ロード・バランシングなし
READ_MOSTLY	<ul style="list-style-type: none"> 透過的フェイルオーバーのサポートなし 透過的な切り替え 2 次サーバーと 1 次サーバーでのワークロード ロード・バランシング 	<ul style="list-style-type: none"> 透過的フェイルオーバー (セッション状態は保存されない) 透過的な切り替え 2 次サーバーと 1 次サーバーでのワークロード ロード・バランシング 	<ul style="list-style-type: none"> 透過的フェイルオーバー (セッション状態は保存される) 透過的な切り替え 2 次サーバーと 1 次サーバーでのワークロード ロード・バランシング

矛盾する TC 情報の処理

TC 情報の属性が、使用可能にされた実際のサービスと矛盾する場合があります。そのような状況では、接続は認可されますが、SUCCESS_WITH_INFO の警告が発行されます。

これは、以下の場合に行われます。

- PREFERRED_ACCESS が指定されたが、HSB が使用可能でない。基本接続は使用可能である。
- TF_LEVEL が指定されたが、HSB が使用可能でない。基本接続は使用可能である。

JDBC での透過接続の規定

JDBC では、透過接続は 2 つの標準外の接続プロパティによって使用可能にされます。

障害透過性は、障害のマスキングを処理します。これは、HotStandby モードとクラスター透過モードの両方に等しく適用されます。障害透過性は、`solid_tf_level` 接続プロパティで使用可能にされます。値は、簡略記号 (NONE など) または数値 (NONE を表す 0) として指定できます。主として、簡略記号を使用してください。値は、ストリングとして与える必要があります。次の 3 つのレベルがあります。

1. NONE。 NONE を表す数値は 0 です。
2. CONNECTION。 CONNECTION を表す数値は 1 です。
3. SESSION。 SESSION を表す数値は 3 です。

値の詳細については、60 ページの『TC での障害透過性』を参照してください。

優先アクセス属性は、読み取り専用負荷を分散するかどうかを示します。優先アクセス属性は、`solid_preferred_access` 接続プロパティで使用可能にされます。値は、簡略記号または数値として指定できます。主として、簡略記号を使用してください。値は、ストリングとして与える必要があります。次の 2 つのレベルがあります。

1. WRITE_MOSTLY。 WRITE_MOSTLY は、接続を WRITE MOSTLY モードにも設定します。数値を指定してこれを行うことはできません。WRITE_MOSTLY を表す数値は 0 です。
2. READ_MOSTLY。 READ_MOSTLY を表す数値は 1 です。

接続の再接続タイムアウトをミリ秒単位で指定するために、TC/TF 固有のプロパティ `solid_tf1_reconnect_timeout` もあります。デフォルト値は 10,000ミリ秒 (10 秒) です。

サーバー・アドレスのリストは、拡張 JDBC 接続ストリングの一部として与えられます。

```
conStr= "jdbc:solid://host_name:port [,host_name:port].../user_name/password";
```

アドレス・リスト内のアドレスの数は、20 までに制限されています。

注意:

JDBC で透過接続を使用する場合は、ステートメント・オブジェクトの廃棄も明示的に処理する必要があります。ガーベッジ・コレクターは、参照されなかったステートメント・オブジェクトを検出しません。

JDBC での透過接続の使用

```
...
String conStr = "jdbc:solid://srv1.acme.com:1323,srv2-acme.com:1423/dba/dba";
Properties prop = new Properties();
prop.setProperty("solid_tf_level", "CONNECTION");
...
Connection c = DriverManager.getConnection(conStr, prop);
...
```

接続エラー処理

TC 接続の接続要求が発行された場合、少なくとも 1 つの該当するサーバーが検出されて、それに接続した場合、接続要求は成功したと見なされます。

サーバーは、PRIMARY ACTIVE、PRIMARY ALONE、STANDALONE のいずれかの状態になります。そうでない場合、接続の試みは失敗したと考えられます。アドレス・リストは 1 回だけスキャンされます。

接続要求の失敗には、さまざまな理由が存在する場合があります。その大部分は、以下のエラー・ケースによってマスクされます。

表 10. 接続要求エラー

SQLSTATE	ネイティブ・コード	メッセージ・テキストおよび説明
08001	25217	Client unable to establish a connection 説明: ドライバーは、該当するサーバーを見つけて接続するために、TC 接続情報を使用しました。その試みは、以下のいずれかの理由で失敗しました。 <ul style="list-style-type: none">• アドレス・リストにリストされたホストが見つからなかった• ホストは見つかったが、ログインがタイムアウトになった• ホストは見つかったが、ログインが拒否された• ホストは見つかったが、PRIMARY/STANDALONE 状態ではなかった
HY000	21307	Invalid connect info... 説明: 基本接続ストリングまたは TC 接続情報 (データ・ソース情報) の中で構文エラーが検出されました。
HY000	21300	Protocol ... not supported. 説明: TC 接続情報の冒頭にあるストリング「TC」のつづりに誤りがあります (または、基本接続ストリング内で誤ったプロトコル名が指定されています)。

警告とともに接続が受け入れられる場合もあります。

表 11. 警告

SQLSTATE	ネイティブ・コード	メッセージ・テキストおよび説明
0100	25218	Connected to Standalone or Primary Alone server. TF_LEVEL または PREFERRED_ACCESS にデフォルト以外の値を設定使用しましたが、使用可能なサーバーは 1 台だけです。この場合、障害透過性もロード・バランシングも使用できません。

接続切り替えのためのプログラミング

接続切り替え処理の原則

接続切り替えとは、ドライバーがアクティブ・サーバー接続を変更する状態を指しています。一般に、接続切り替えの理由は 2 次サーバーへのフェイルオーバーか、サーバー間の切り替えです。

具体的には、接続切り替えの必要性は、以下のいずれかのイベントから検出されます。

- PRIMARY ALONE (フェイルオーバー) または PRIMARY ACTIVE (切り替え) への状態変更に関する 2 次サーバーからのイベント。これは、接続切り替えを行う主な (また最高速の) モードです。
- 1 次サーバーでの状態変更の標識。
- アクティブ接続でのリンク障害。
- アクティブ接続での接続タイムアウト。

ドライバーは、以下の 2 つのステップで接続切り替えを実行します。

1. 接続切り替えの必要性が検出されます。ドライバーは、処理中の要求または以下の要求について、以下の接続切り替えエラーを返します。

表 12. 接続切り替え要求

SQLSTATE	ネイティブ・コード	メッセージ・テキストおよび説明
HY000	25216	<p>Connection switch, some session context may be lost</p> <p>説明: ドライバーは接続切り替えの必要性を検出しました。クライアントは、接続切り替えをファイナライズするために、トランザクション・ロールバック呼び出しを発行することを期待されます。このエラー・コードおよびメッセージは、ロールバック呼び出しが発行されるまで、連続するネットワーク要求呼び出しごとに受信されます。</p>

2. クライアント・プログラムはロールバック・コマンドを発行します (ODBC: SQL_ROLLBACK 付きの `SQLEndTran()`。JDBC: `Connection.rollback()`)。ロールバックが成功すると、新しいアクティブ接続が、使用される可能性がある TC 接続へマップされています。

注: 少数の連続する ODBC 呼び出しについて、接続切り替えエラーが返される場合があります。したがって、すべての ODBC ネットワーク要求で、このエラーに対して常にロールバックで応答するよう、対策を講じておく必要があります。トランザクションの最中にこれが発生した場合は、トランザクションを再実行する必要があります。

一方、新しいアクティブ接続を確立できない場合は、以下のエラー・コードが返されます。

表 13. 通信リンク障害

SQLSTATE	ネイティブ・コード	メッセージ・テキストおよび説明
08S01	14503	<p>Communication link failure</p> <p>説明: ドライバーは、新しいアクティブ接続の確立に失敗しました。TF 接続は失われ、クライアントは作業を続行するために、(データ・ソース情報を使用して) 再接続する必要があります。</p>

ロールバック呼び出しを受信後、ドライバーは新しいアクティブ接続を検出する少数の代替手段を使用します。最も単純なケースでは、その目的にスタンバイ接続が使用されます。その接続が正しい状態にない場合、ドライバーは適正なイベントが

到着するまで 2 秒間待機します。イベントが到着しない場合、およびその他のケースでは、ドライバーは TC 接続情報内のアドレス・リストにフォールバックし、接続シーケンスを最大 10 秒間繰り返します。すべての試みが失敗した場合、ドライバーは上記のエラーに戻ります。

エラーによる影響は、アプリケーションが検出したように、接続が失われることです。それ以後にその接続上で発行される要求は、同じエラーになります。

セッション状態の保存

ドライバーによって接続切り替えが実行された場合、一部のセッション・コンテキストが失われる可能性があり、クライアントはそれを再構成する必要があります。保存される状態の量は、透過的フェイルオーバーのレベルによって指示され、TC 情報の属性 TF_LEVEL で表されます。

基本的に、TF レベルの CONNECTION では、状態は保存されませんが、SESSION レベルでは、ほとんどのセッション状態が保存されます。セッション状態の保存は、必要なデータをドライバー内にキャッシングすることによって実施されます。透過性レベルを高くするほど、キャッシングを必要とする要求の応答時間が遅くなり、ドライバー内のメモリー使用量が増えます。

フェイルオーバーの場合は、TF レベルとは関係なく、以下のようになります。

- 現行トランザクションの更新情報は (トランザクション・ロールバックのために) 失われます。
- オープン・カーソルおよびその位置は失われます。

以下の表は、セッション状態の保存についての要約です。

表 14. セッション状態の保存

TF_LEVEL	保存される状態
CONNECTION	セッション状態は保存されません。

表 14. セッション状態の保存 (続き)

TF_LEVEL	保存される状態
SESSION	<p>準備済みステートメント</p> <ul style="list-style-type: none"> • 準備済みステートメントは保存されます。 <p>以下のステートメントの効果は保存されません。</p> <ul style="list-style-type: none"> • SET CATALOG • SET SQL INFO • SET SQL SORTARRAYSIZE • SET SQL CONVERTORSTOUNIONS • SET SQL SORTEDGROUPBY • SET SQL { OPTIMIZEROWS OPTIMISEROWS } • SET SIMPLIOPTIMIZERRULES • SET LOCK TIMEOUT <seconds> • SET OPTIMISTIC LOCK TIMEOUT • LOCK_TIMEOUT • SET IDLE TIMEOUT • SET STATEMENT MAXTIME • SET ISOLATION LEVEL • SET DURABILITY • SET SAFENESS • SET SCHEMA • SET SQL JOINPATHSPAN • SET SYNC USER • SET SYNC MODE <p>以下の標準 ODBC 属性は保存されます。</p> <ul style="list-style-type: none"> • SQL_ATTR_ACCESS_MODE <p>(SET READ ONLY、 SET READ WRITE)</p> <ul style="list-style-type: none"> • SQL_ATTR_CURRENT_CATALOG <p>(上記の SET CATALOG を複製する)</p> <ul style="list-style-type: none"> • SQL_ATTR_AUTOCOMMIT

追加のプロプラエタリー ODBC 属性

以下の読み取り専用 ODBC 属性は、アプリケーション・プログラマーが任意のフォーマットの TC 情報に使用できます。

- SQL_ATTR_PA_LEVEL

(整数、優先アクセス・レベル: 0=WRITE_MOSTLY、1=READ_MOSTLY)

ロード・バランシングを使用するかどうかを示す属性。

- SQL_ATTR_TC_WORKLOAD_CONNECTION

(ストリング、ワークロード接続のサーバー名)

現行のワークロード接続サーバー。Commit の前に照会された場合、この値はトランザクションがコミットされるサーバーを示します。

- SQL_ATTR_TF_LEVEL

(整数、TF レベル: 0=NONE、1=CONNECTION、3=SESSION)

障害透過性レベル。

- SQL_ATTR_TC_PRIMARY

(ストリング、1 次サーバー接続ストリング)

常に、現行の 1 次サーバーを示す値が存在します。

- SQL_ATTR_TC_SECONDARY

(ストリング、2 次サーバー接続ストリング)

この値は、以下の場合、ワークロード割り当て済みサーバーを示します。

1. PA=READ_MOSTLY で、しかも
2. 2 次サーバーが、指定されたワークロード・サーバーである場合 (これはデフォルトです)。

これ以外の場合は、空ストリングが返されます。

注: プロプラエタリー ODBC 属性を Windows の ODBC ドライバー・マネージャーで使用することはできません。Windows でプロプラエタリー ODBC 属性を使用する必要がある場合は、Solid[®] ODBC ドライバー・インポート・ライブラリー (solidimpodbca.lib または solidimpodbcu.lib) をアプリケーションと直接リンクする必要があります。

基本接続の使用

基本接続では、HotStandby 構成またはクラスター構成の各サーバーへの接続を、アプリケーションが特定のサーバー・アドレスを使用して別々に処理する必要があります。フェイルオーバーが発生した場合、アクティブ接続は失われ、アプリケーションは新規 1 次サーバーに再接続する必要があります。

透過性のない基本的な接続

```
Connect=tcp srv1.dom.acme.com 1315
```

アプリケーションから 1 次サーバーへの再接続

HotStandby 用のクライアント・アプリケーションの準備

1 次サーバーとの接続を失ったクライアント・プログラムは、新しい 1 次サーバー (旧 2 次サーバー) に再接続できなければなりません。クライアント・アプリケーションは、以下のことができるようにコーディングする必要があります。

1. 1 次サーバーを書き込みトランザクションにそれ以上使用できないことを認識する。
2. 他のサーバーに接続するか、前に作成された接続の使用に切り替える。
3. 現行の (中断された) トランザクションが失われるか打ち切られており、新しい 1 次サーバー上で再実行する必要があるかどうかを考慮する。

2 次サーバー・アドレスの取得

2 次データベース・サーバーの接続情報を取得する最も簡単な方法は、ADMIN COMMAND 'hotstandby cominfo' コマンドを使用することです。このコマンドは、HSB ペアの相手側サーバーの接続情報を返します。

このタスクについて

1. アプリケーションは、初めて 1 次サーバーに接続したとき、ADMIN COMMAND 'hotstandby cominfo' コマンドを実行して、結果を保管できます。cominfo コマンドが値を返した場合、それは 1 次サーバーと 2 次サーバーが現在接続されていることを意味するわけではありません。「cominfo」コマンドは、単に solid.ini 構成ファイルの **Connect** パラメーターで指定された値を返すか、hsb parameter connect コマンドで指定された最新の値を返します。1 次サーバーと 2 次サーバーの間の接続状態を確認する必要がある場合は、ADMIN COMMAND 'hotstandby status connect' を使用できます。
2. その後、1 次サーバーが障害を起こした場合、アプリケーションは保管されている情報を使用して、2 次サーバー (新しい 1 次サーバー) に接続できます。

クライアント・アプリケーションでの HotStandby サーバーの障害の検出

HotStandby (HSB) コンポーネントを使用するには、アプリケーションが、障害を起こした 1 次サーバーから 2 次 (新規 1 次) サーバーへ切り替える時期を知っている必要があります。これを行うには、いくつか方法があります。最良の方法は、単に呼び出した機能からの戻りコードを検査して、他のサーバーへ切り替える必要があることを示すエラーを受信したかどうかを調べることです。

また、サーバーの状態をモニターしてもかまいません (例えば、1 次サーバーを検査して、その状態が PRIMARY UNCERTAIN に変化したかどうかを調べます)。

別のサーバーへの切り替えを試みる必要があることを示すエラーには、以下のものがあります。

- 10013: トランザクションが読み取り専用である
- 10041: データベースが読み取り専用である
- 10047: レプリケーション・トランザクションが異常終了した
- 11002: ディスクが満杯である
- 11003: 構成を超過した
- 14501: 操作が失敗した
- 14502: 無効な rpc パラメーター
- 14503: 通信エラー
- 14506: サーバーがクローズされた (例えば、現在 HSB コピーまたはネットコピー操作のターゲットとなっているため)

- 14510 通信書き込み操作が失敗した
- 14511 通信読み取り操作が失敗した
- 14518: 接続が切断された
- 14519: ユーザーが拒否された (例えば、何らかの管理操作のため)
- 14529: 操作がタイムアウトになった
- 20009: セッション・エラー、書き込み操作が失敗した
- 21306: サーバーを検出できず、接続が失敗した
- 21308: 接続が切断された (書き込みがコード ... で失敗した)
- 21318: 操作が失敗した (通常でない戻りコード)

ODBC アプリケーション

以下のエラー・メッセージは、(例えば、操作不能なデータベース・サーバーが原因で) 接続を確立できなかった ODBC アプリケーションへ返されます。

- SQLState = 08001 - Client unable to establish connection

さらに、以下の solidDB 通信エラー・メッセージが生成されます。

- 21306 - Server '*server_name*' not found, connection failed.

照会の実行と結果の取得など、操作と操作の間で接続が (例えば、ネットワーク障害などのために) 失敗した場合、以下のエラー・メッセージが返されます。

- SQLState = 08S01 - Communication link failure

JDBC アプリケーション

以下のエラー・メッセージは、(例えば、操作不能なデータベースが原因で) 接続を確立できなかった JDBC アプリケーションへ返されます。

- SQLState = 08001 - Unable to connect to data source.

照会の実行と結果の取得など、操作と操作の間で接続が (例えば、ネットワーク障害などのために) 失敗した場合、以下のエラー・メッセージが返されます。

- SQLState = 08S01 - Communication link failure

注:

ODBC および JDBC は、同じエラー・コード (08001) に異なるエラー・メッセージを使用します。

新規 1 次サーバーへのアプリケーションの切り替え

アプリケーションは、「旧 1 次」サーバーへトランザクションを送信できないことを検出した後、旧 1 次サーバーと旧 2 次サーバーに対して、PRIMARY ACTIVE、PRIMARY ALONE、STANDALONE のいずれかの状態のサーバーを検出するまで、ポーリングを行う必要があります。

ポーリングは、アプリケーションがサーバーへの接続を試み、接続が確立されたときは、サーバーの状況を検査することによって行われます。接続が成功した場合、クライアントは SQL 関数 HOTSTANDBY_STATE を使用してサーバー状態を要求

できます。この SQL 関数については、『HOTSTANDBY_STATE 関数の使用』のセクションで説明されています。

注意:

切り替え後、オープンされているすべてのデータベース・オブジェクト (準備済みステートメント、オープン・カーソル、トランザクションなど) は、アクティブでなくなります。このため、それらのオブジェクトを再び初期化する必要があります。また、テンポラリー表またはトランジエント表 (solidDB メイン・メモリー・エンジン機能) を使用していた場合、それらの表は新規 1 次サーバーでは空になります。

HOTSTANDBY_CONNECTSTATUS 関数の使用

アプリケーションから 1 次サーバーに再接続したとき、接続状況情報を検証するために、HOTSTANDBY_CONNECTSTATUS 関数を使用できます。この関数は、管理コマンド `hotstandby status connect` と等価です。

この関数は引数を持たず、以下のいずれかの状況値を返します。

表 15. HOTSTANDBY_CONNECTSTATUS 状況値

状況	説明
CONNECTED	接続はアクティブです。この状況は、1 次と 2 次の両方のサーバーから返されます。
CONNECTING	1 次サーバーが 2 次サーバーに接続中です。この状況は、1 次と 2 次の両方のサーバーから返されます。
CATCHUP	1 次サーバーが 2 次サーバーに接続しましたが、トランザクション・ログはまだ完全にはコピーされていません。この状況は、1 次と 2 次の両方のサーバーから返されます。
BROKEN	接続は切断されています。この状況は、1 次と 2 次の両方のサーバーから返されます。

HOTSTANDBY_STATE 関数の使用

1 次および 2 次サーバーのアプリケーション・ポーリングをインプリメントするには、HOTSTANDBY_STATE 関数を使用できます。この関数は、管理コマンド `hotstandby state` と等価です。これは、アプリケーションがサーバーへ接続されたとき、アプリケーションが現行の HotStandby 状態を要求できるようにします。

注: この関数に引数はありません。この関数が返す可能性があるそれぞれの状態の説明については、100 ページの『HotStandby サーバー状態の検証』を参照してください。

サンプルの疑似コード

アプリケーションは、HSB 対応であろうとなかろうと、障害/異常終了を起こしたトランザクションをやり直せるエラー処理機能を備えている必要があります。

非 HSB 環境では、トランザクションは並行性競合 (オプティミスティック表) またはデッドロック (ペシミスティック表) のために異常終了する場合があります。アプリケーションは、それらのエラー状態をキャッチし、自動的にトランザクションを再試行するか、対話式ユーザーにトランザクションを再実行するよう依頼する必要があります。

障害を起こすか異常終了したトランザクションを処理するコードが既にアプリケーションに存在する場合、そのコードを拡張して HSB を使用するの、比較的簡単です。

非常に単純化した例では、非 HA 対応アプリケーション処理用の適正なエラー処理機能を備えたアプリケーション疑似コードは、以下のようになります。

```
BEGIN TRANSACTION
EXECUTE APPLICATION LOGIC
PREPARE & EXECUTE STATEMENTS
COMMIT TRANSACTION
IF ERROR OCCURRED
    IF ERROR == concurrency conflict or deadlock
        GO TO BEGIN TRANSACTION
    END IF
    other error handling
END IF ;
```

上記のアプリケーションを改良して HA 対応にするのは、非常に簡単です。アプリケーションが以下のことを実行できるよう、コードを追加する必要があります。

- 1 つだけでなく、2 つのサーバーのどちらにも接続でき、しかも、
- エラーが起きた場合は、現在の状態が PRIMARY ACTIVE、PRIMARY ALONE、STANDALONE のいずれかであるサーバーを検出する。

疑似コードは、以下のようになります。

```
BEGIN TRANSACTION
EXECUTE APPLICATION LOGIC
PREPARE & EXECUTE STATEMENTS
COMMIT TRANSACTION
IF ERROR OCCURRED
    IF ERROR == server unavailable for write transactions
        FIND CURRENT PRIMARY SERVER
        GO TO BEGIN TRANSACTION
    END IF
    IF ERROR == concurrency conflict or deadlock
        GO TO BEGIN TRANSACTION
    END IF
    IF ERROR == something else
        other error handling
    END IF
END IF
```

現行 1 次サーバーを検出するロジックも、非常に簡単です。単に両方のサーバーの現行状態を検査し (必要であれば再接続を試み)、どちらかが PRIMARY ACTIVE、PRIMARY ALONE、STANDALONE であれば、そのサーバーを現行 1 次サーバーとして設定します。どちらのサーバーも基準を満たさない場合は、しばらく待ってから現行サーバー状態の検査を再試行します。

2 次サーバーへの再接続

現行 2 次サーバー (稼働している場合) に接続したい場合もあります。アプリケーションは、2 次サーバーに対し、読み取り専用の照会をサブミットできます。これは、サーバー間でワークロードのバランスをとるのに役立つ場合があります。

アプリケーションは、読み取り専用モードでのみ、2 次サーバーのデータベースに接続できます。クライアントが (読み取り専用モードに限って) 2 次サーバーに接続できるようにするには、以下のように、各サーバーの `solid.ini` 構成ファイルの `HotStandby` セクションで、以下のパラメーター値を使用します。

- 1 次サーバー内で **Connect** パラメーター
- 2 次サーバー内で **Listen** パラメーター

次のコマンドを使用して、サーバーのパートナーの接続情報を取得することもできます。

```
ADMIN COMMAND 'hotstandby cominfo';
```

このように、現行 1 次サーバーに接続している場合は、`cominfo` 照会を使用して現行 2 次サーバーのアドレスを取得できます。

拡張レプリケーションの要件

拡張レプリケーション・システムのすべてのノードは、`solidDB HotStandby` コンポーネントによって可用性を高めることができます。

拡張レプリケーション・システムのマスター・データベースとレプリカ・データベースが同期データである場合、データベース・サーバー・ペアの 1 次サーバー同士の間で同期化が発生します。言い換えれば、マスターの 1 次サーバーは、レプリカの 1 次サーバーと通信します。7 ページの図 3 を参照してください。

データベース・サーバーは、どのような時点でもそのサーバーの 2 次サーバーへフェイルオーバーする可能性があり、データベース・サーバーが拡張レプリケーションを使用して別のサーバーとデータの同期をとっているときも、その例外ではありません。同期化の最中にフェイルオーバーが発生した場合、同期化メッセージの実行は停止し、そのプロセスはフェイルオーバー後に処理を再開する必要があります。エラーが起きた後に、同期化を再開する方法の詳細については、「[拡張レプリケーション・ユーザー・ガイド](#)」を参照してください。

拡張レプリケーション・マスター・データベースを格納しているサーバーを `solidDB HotStandby` でフォールト・トレラントにした場合、マスター・データベースのレプリカに両方のマスター・サーバーへの接続ストリングを知らせる必要があります。そのためには、それぞれのレプリカ・データベースで以下のステートメントを実行します。

```
SET SYNC CONNECT 'connect_string_to_server_1, connect_string_to_server_2'  
TO MASTER master_nodename
```

下の図で、灰色の矢印は当初の 1 次サーバーへの当初の接続を表し、黒の矢印は新規 1 次サーバー (旧 2 次サーバー) への新規接続を表しています。旧 1 次サーバーとの同期に失敗した場合は、代替接続が使用されます。

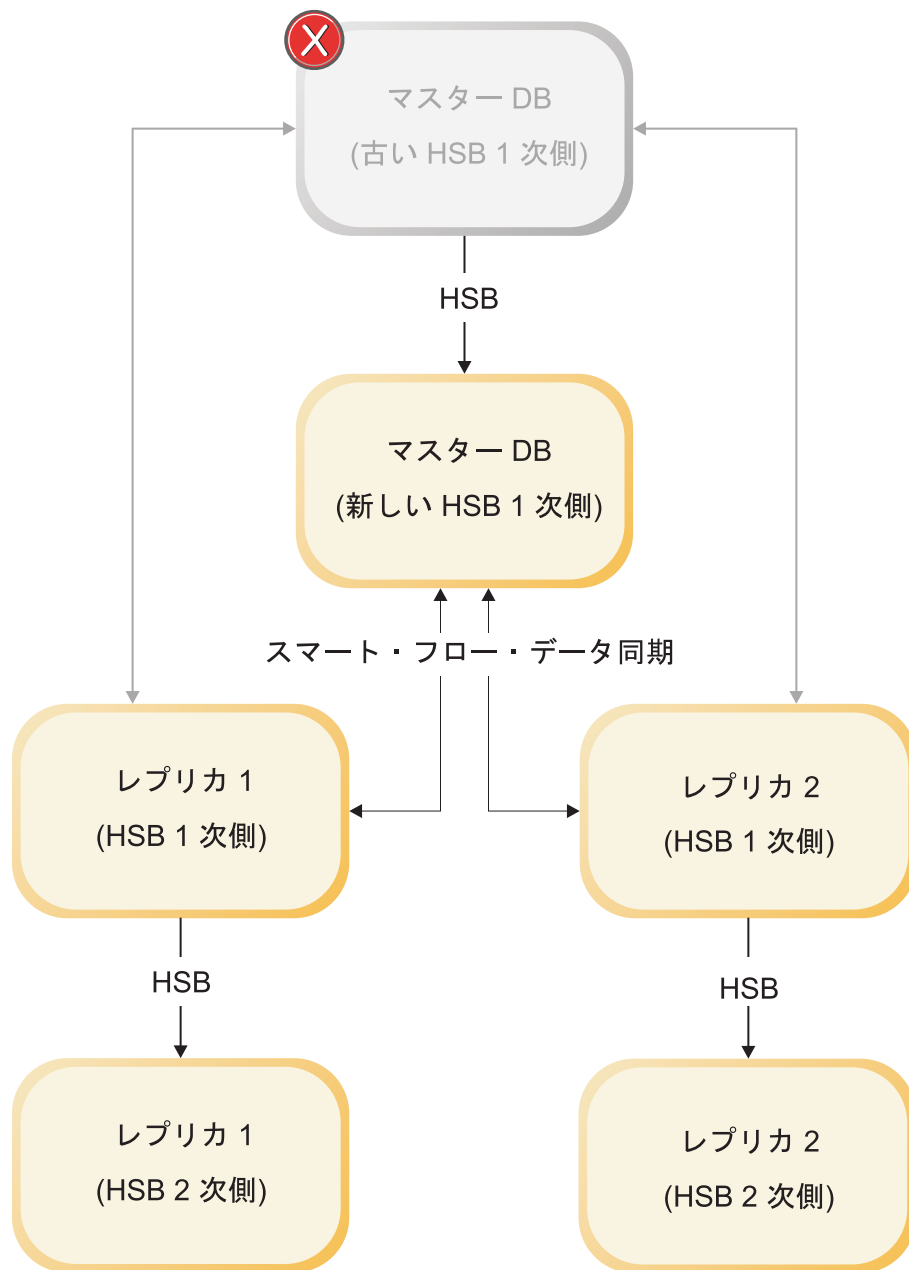


図 11. マスターのフェイルオーバー

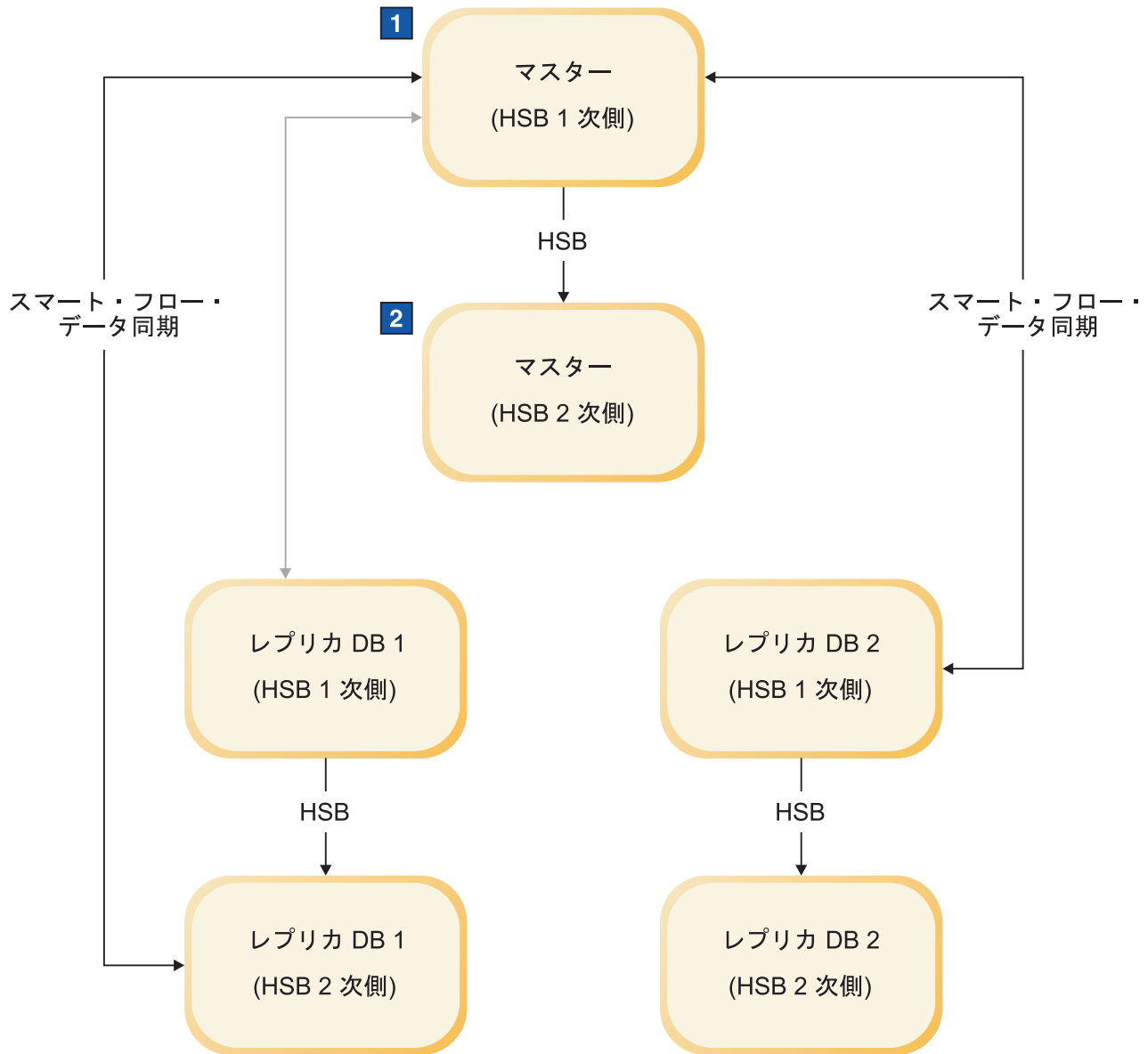
solidDB HotStandby を使用しているサーバーがレプリカ・データベースを格納しているサーバーであり、マスター・サーバーがリモート・プロシージャ・コール (CALL procedure AT node_name) を使用してレプリカでプロシージャを実行する (例えば、同期化を開始する) 場合、マスター・サーバーは、両方のレプリカ・サーバーへの接続ストリングを知らされている必要があります。一般に、マスター・サーバーはリモート・プロシージャ・コールを使用して、レプリカ・データベースとの同期化を開始します。マスターにレプリカ・サーバー・ペアへの接続ストリングを知らせるには、マスター・データベースで以下のステートメントを実行します。

```

SET SYNC CONNECT 'connect_string_to_server_1,
connect_string_to_server_2' TO REPLICAS replica_nodename
  
```

別の方法として、ステートメントをレプリカ・サーバー内に保存しておき、それを次回に同期をとるときにマスターへ伝搬させることもできます。その場合は、以下のステートメントを使用します。

```
SAVE SET SYNC CONNECT 'connect_string_to_server_1,
connect_string_to_server_2' TO REPLICIA replica_nodename
```



1. レプリカ・データベースでのフェイルオーバー
2. SET SYNC CONNECT 'tcp machine4 1315' REPLICIA TO replicaDB1

図 12. レプリカのフェイルオーバー

マスター・サーバーがレプリカ内でリモート・プロシージャー・コールを実行しない場合、上記のステートメントは必要ありません。

5 HotStandby API コマンドの使用

solidDB HotStandby の高可用性を管理するすべてのソフトウェアで、サーバー・プロセスのモニターと制御を行うために、HotStandby API (HSB API) が使用されます。製品に組み込まれている HA コントローラーは、そのようなプログラムの一例です。もう 1 つの例は、Watchdog サンプル・プログラムです。

HSB API は、SQL への構文拡張として提供されており、次のような標準外のステートメントの形態をとります。

```
ADMIN COMMAND hotstandby hsb-command options
```

または

```
ADMIN COMMAND hsb hsb-command options
```

HSB コマンドは、SQL に対応した任意の対話式ツール (solsql など) を介して発行するか、または ODBC や JDBC を通じてプログラマチックに発行することができます。

このセクションを使用して独自のアプリケーションをプログラムし、solidDB の高可用性を管理できます。これは、例えば、外部クラスター管理ソフトウェアへの統合をインプリメントする場合などに必要になります。

このセクションのトピックは、以下のとおりです。

サーバー状態の切り替え

HotStandby コンポーネントは、必要な場合、自動またはユーザーによる手動でのサーバー状態の切り替えを必要とします。

実動では、サーバー状態は自動的な状態切り替えによって (つまりフェイルオーバーの実行によって) 選択されます。これは、solidDB にインプリメントされている自動高可用性制御の責任で行われます。solidDB では、自動高可用性制御は高可用性コントローラー (HAC) によって処理されます。

切り替え とは、1 次サーバーと 2 次サーバーが、稼働中にロールを逆にすることを意味します。これは、さまざまな保守の目的で必要になる場合があります。

一方、フェイルオーバー は、1 次サーバーに障害が起きたときに、2 次サーバーが 1 次サーバーのロールを引き継ぐアクションです。

切り替えの実行

HAC では、2 次サーバーで ADMIN COMMAND 'hotstandby switch primary'; コマンドを発行するか、1 次サーバーで ADMIN COMMAND 'hotstandby switch secondary'; コマンドを発行することにより、サーバーのロールを逆にすることができます。

このコマンドは、2 つのサーバーが接続されている場合でも接続されていない場合でも使用できます。サーバー同士が接続されている場合は、単に状態が逆になり、

旧 2 次サーバーが新規 1 次サーバーに、旧 1 次サーバーが新規 2 次サーバーになります。サーバー同士が接続されていない場合は、旧 2 次サーバーが新規 1 次サーバーになり、もう一方のサーバーの状態は変更されません。

下の図は、サーバー同士が接続されているときに、コマンド `switch secondary` または `switch primary` を発行するとどうなるかを示したものです。コマンド `switch primary` は `SECONDARY` 状態 (例えば、`SECONDARY ACTIVE`) にあるサーバー上でのみ発行され、コマンド `switch secondary` は `PRIMARY` 状態 (例えば、`PRIMARY ACTIVE`) にあるサーバーでのみ使用されることに注意してください。

状態切り替え

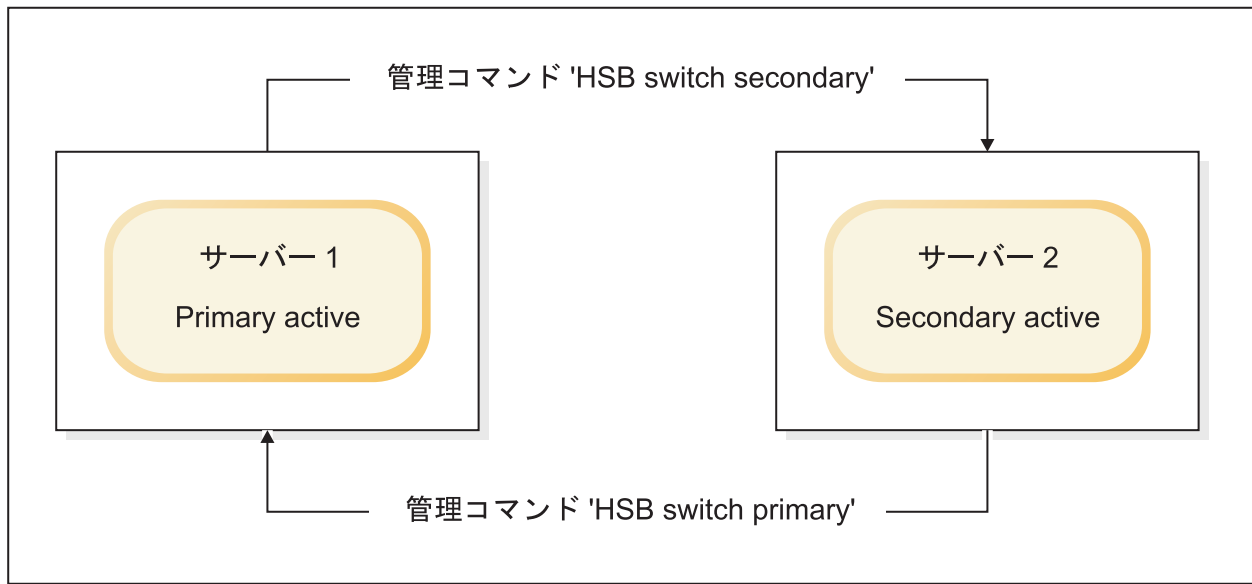


図 13. 状態切り替え

コマンド `hotstandby switch primary` を実行して 2 次サーバー (サーバー 2) を 1 次サーバーに切り替える場合、2 次サーバー (サーバー 2) がもう一方のサーバー (サーバー 1) に接続していないと、エラーが返されます。

2 つのサーバーが接続されている場合は、サーバー同士の状態が切り替わります。つまり、旧 1 次サーバー (サーバー 1) は新規 2 次サーバーになり、旧 2 次サーバー (サーバー 2) は新規 1 次サーバーになります。

旧 2 次サーバー (サーバー 2) がもう一方のサーバー (サーバー 1) に接続できない場合は、両方のサーバーが `SECONDARY ALONE` に切り替わります。(AutoPrimaryAlone 構成パラメーターが `yes` に設定されている場合でも、新規 1 次サーバーは `PRIMARY ALONE` でなく `SECONDARY ALONE` に切り替わることに注意してください。)

2 次サーバーの 1 次サーバーへの切り替え

`hotstandby switch primary` コマンドを実行すると、状態を切り替えるプロセスが開始されます。

切り替えプロセスが正常に開始された場合は、以下のメッセージが表示されます。
Started the process of switching the role to primary.

切り替えのとき、アクティブな書き込みトランザクションは、すべて打ち切られます。切り替えの状況は、コマンド `hotstandby status switch` を使用してモニターできます。詳細については、84 ページの『切り替えの検証』を参照してください。

SWITCH コマンドの後に COMMIT を発行すると、COMMIT は 'replicated transaction is aborted' というエラーで失敗します。

すべてのトランザクションは、切り替えのときに強制終了されます。ただし、「HSB switch」コマンドなどの ADMIN COMMAND (管理コマンド) は、トランザクション・コマンドでないので、ロールバックできないことに注意してください。

注:

ただし、管理コマンドは、新規トランザクションがまだ開かれていない場合、強制的に新規トランザクションを開始します。オープン・トランザクションが終了しないようにするには、またはトランザクションの開始時刻を予期したものと異なる時刻にするには、管理コマンドの後に COMMIT WORK を実行するとよいでしょう。

両方のサーバーが PRIMARY 状態 (例えば、両方が PRIMARY ALONE) になる構成エラーが起きた場合は、コマンド `hotstandby switch secondary` を使用して、どちらかのサーバーを SECONDARY 状態に戻すことができます。サーバーが同じデータを持っている場合は、両方のサーバー上で通常の操作が再開されます。しかし、サーバーが同じデータを持っていない場合、1 次サーバーは 2 次サーバーからの接続操作を拒否し、以下のメッセージを発行します。14525: HotStandby databases are not properly synchronized.

HotStandby レプリケーションは開始されません。その場合は、2 次サーバー側に 1 次データベースの完全コピーが必要です。最初に、どちらのデータベースが正しいかを判断する必要があります。14525 エラーが発生した場合、データベース状態は変化しません。両方のサーバーはコマンド発行以前と同じ状態のままになります。

1 次サーバーの 2 次サーバーへの切り替え

コマンド ADMIN COMMAND 'hotstandby switch secondary'; を発行することにより、1 次サーバーを SECONDARY 状態に切り替えることができます。

これは、2 つのサーバーの状態が既に切り替えられており、再び元の状態に切り替えたい場合に特に便利です。例えば、新規 2 次サーバーがサービスに復帰したときに、その状態を切り替えて 1 次サーバーに戻し、新規 1 次サーバーを 2 次サーバーに再び切り替えることができます。

`hotstandby switch secondary` を実行する場合、サーバー同士がまだ接続されていないれば、旧 1 次サーバーは旧 2 次サーバーに接続しようとします。

2 つのサーバーが接続されている場合は、サーバー同士の状態が切り替わります。つまり、旧 1 次サーバーは新規 2 次サーバーになり、旧 2 次サーバーは新規 1 次サーバーになります。

hotstandby switch secondary コマンドを実行すると、状態を切り替えるプロセスが開始されます。切り替えプロセスが正常に開始された場合は、以下のメッセージが表示されます。Started the process of switching the role to secondary.

任意の HotStandby サーバーの切り替え状況を検査して、切り替えが正常に実行されたかどうかを検証できます。詳細については、98 ページの『切り替え状況情報の表示』を参照してください。

切り替えの検証

1 次または 2 次サーバーでの切り替えプロセスの状況を検査できます。

以下のコマンドを発行します。ADMIN COMMAND 'hotstandby status switch'; このコマンドは、切り替えが 2 つのサーバー間で発生しなかったか、成功したか、まだ進行中か、あるいは切り替えが失敗したかを知らせる状況メッセージを表示します。

フェイルオーバーの実行

フェイルオーバーは、2 次サーバー側で、コマンド ADMIN COMMAND 'hotstandby set primary alone'; を実行することによって行われます。

サーバーは、以前に 1 次サーバーから受信した保留中のすべてのトランザクションを処理した後、新しい状態になります。これにより、失われるトランザクションがないこと、およびデータベース状態に障害直前の 1 次サーバーの状態が反映されることが保証されます。ただし、使用している安全性レベルが 1-safe の場合は、フェイルオーバーの際にいくつかのトランザクションが失われることがあります。

PRIMARY ALONE 状態での新規 1 次サーバーの実行

この状態では、2 次サーバーへの接続は切断されていますが、1 次サーバーを実行してトランザクション・ログの更新を継続できます。2 次サーバーが復帰した後、PRIMARY ALONE 状態のサーバーは 2 次サーバーへのトランザクションの送信を再開できます。

サーバーを PRIMARY ALONE 状態に設定するには、以下の 3 とおりの方法があります。

- 以下のコマンドを発行します。

```
ADMIN COMMAND 'hotstandby set primary  
alone';
```

- 以下のようにして、制御された切断を実行します。

```
ADMIN COMMAND 'hotstandby  
disconnect';
```

これは、1 次サーバーと 2 次サーバーのどちらかで実行します。制御されたシャットダウンを行うために、

```
ADMIN COMMAND 'shutdown';
```

を 2 次サーバー上で実行すると、2 次サーバーはシャットダウンの前に暗黙に切断され、1 次サーバーは安全に PRIMARY ALONE 状態に切り替わることに注意してください。

- 構成ファイル (solid.ini) の [HotStandby] セクションで、構成パラメーター **AutoPrimaryAlone** を「yes」に設定して、PRIMARY ALONE 状態をデフォルトにします。

PRIMARY ALONE 状態がデフォルトの場合、サーバーは 2 次サーバーへの接続が切断されたとき、自動的に PRIMARY ALONE 状態になります。それ以外の場合は、サーバーに障害が起きた後、サーバーの状態は PRIMARY UNCERTAIN のままになります。ただし、HAC、管理者、ウォッチドッグ・プログラムのいずれかがコマンド ADMIN COMMAND 'hotstandby set primary alone' を発行した場合は除きます。デフォルトでは、solid.ini ファイルの [HotStandby] セクションの **AutoPrimaryAlone** パラメーターは「no」に設定されます。これは、PRIMARY ACTIVE 状態で作動している 1 次サーバーが、2 次サーバーに障害が起きた場合、自動的に PRIMARY UNCERTAIN に切り替えられることを指定します。

タスクの結果

PRIMARY ALONE 状態は、以下のいずれかが発生するまで持続します。

- 2 次サーバーへの接続が成功した。
- サーバーにトランザクション・ログ用のスペースがなくなった。
- ログ・サイズの限度 (MaxLogSize) を超えた。
- 別のコマンドによって、サーバーが STANDALONE など、別の状態に切り替えられた。
- 1 次サーバーがシャットダウンされた。

注意:

2 次サーバーに PRIMARY ALONE 状態を命じると同時に 1 次サーバーのシャットダウンを行わないよう注意する必要があります。この 2 つの操作は矛盾しているため、2 次サーバーは SECONDARY ALONE 状態になります。これが現実の操作で偶然起きることは、ほとんどありません。ただし、システムのテスト中に、シャットダウンで 1 次サーバーの障害をシミュレートしたくなる場合があります。しかし、それはしないでください。シャットダウンは障害の代わりにはなりません。それは、1 次と 2 次の両方のサーバーが関与した複雑な分散操作になります。それをしないもう 1 つの理由は、1 次サーバーは、シャットダウンされて新規 2 次サーバーとして始動された後、新規 1 次サーバーにキャッチアップできなくなるからです。1 次サーバーのシャットダウンが本当に必要な場合、正しい手順は以下のとおりです。

1. 切り替えを実行します。
2. 新規 2 次サーバーをシャットダウンします。
3. 新規 1 次サーバーが自動的に PRIMARY ALONE 状態に切り替わります。

2 次サーバーのオンラインへの復帰

2 次サーバーをオンラインに復帰させるには、1 次サーバーを 2 次サーバーに接続します。詳細については、97 ページの『HotStandby サーバーの接続』を参照してください。

2 次サーバー・ノードをオンラインに復帰した後、キャッチアップが必要になる場合があります。1 次サーバー内の変更は、長期間にわたって蓄積されています。1

次サーバーが PRIMARY ALONE 状態に設定されていた間、1 次サーバーはトランザクション・ログにトランザクションとデータを書き込んでいました。2 次サーバーが再び 1 次サーバーに接続すると、同期をとるために、1 次サーバーに保留中の変更がトランザクション・ログから 2 次サーバーへ書き込まれます。変更が 2 次サーバーに書き込まれている間、2 次サーバーは SECONDARY ALONE 状態になり、1 次サーバーは PRIMARY ALONE 状態になります。(コマンド ADMIN COMMAND "hsb status connect" を発行すると、サーバーがキャッチアップを実行中であるかどうかを知らせるメッセージが表示されます。)

注: 1 次サーバーが、コマンド hotstandby set standalone を使用して STANDALONE 状態に設定されていた場合、2 次サーバーを SECONDARY ACTIVE 状態にするには、事前に 1 次サーバーから 2 次サーバーにデータベース全体をコピーしておく必要があります。87 ページの『1 次サーバーと 2 次サーバーの同期』を参照してください。

タスクの結果

2 次サーバーが、保留中の変更の処理を正常に終了した後、1 次サーバーと 2 次サーバーの状態は、それぞれ PRIMARY ACTIVE と SECONDARY ACTIVE に自動的に変更されます。

HotStandby 操作のシャットオフ

1 次サーバーの HotStandby 操作を一時的にシャットオフする必要がある場合があります。例えば、2 次サーバーのサービスをアップグレードのために休止しようとしているとき、1 次サーバーに、2 次サーバーのサービス休止中に蓄積されるトランザクション・ログを保管する十分なディスク・スペースがない場合などです。

このタスクについて

(詳細については、39 ページの『トランザクション・ログのスペース不足』を参照してください。)

1 次サーバーで HotStandby をシャットオフするには、以下の手順を実行します。

1. サーバーを切断します (現在、接続している場合)。
2. 以下の一連のコマンドを使用して、1 次サーバーを STANDALONE 状態に設定します。

```
ADMIN COMMAND 'hotstandby disconnect'; -- if servers are connected
ADMIN COMMAND 'hotstandby set standalone';
```

これにより、1 次サーバーは非 HotStandby サーバーであるかのように操作を続行できます。

注:

2 次サーバーへ送信するトランザクション・ログの保管を停止すると、もはや単に 1 次サーバーと 2 次サーバーを再接続するだけでは、キャッチアップできなくなります。代わりに、HotStandby 操作を再開するときに、手動でサーバー同士を同期させることが必要になります。詳細については、以下のセクションを参照してください。

次のタスク

このサーバーを、以後永続的に HotStandby サーバーとして使用しない場合は、103 ページの『HotStandby サーバーから非 HotStandby サーバーへの変更』を参照してください。

1 次サーバーと 2 次サーバーの同期

各サーバーが HSB レプリケーションを開始できるようにするためには、各サーバーのデータベースが同一でなければなりません。言い換えると、2 次データベースは 1 次データベースの正確なコピーでなければなりません。HotStandby システムの各データベースを同じものにするプロセスを、HotStandby 同期と呼びます。

1 次側と 2 次側の同期をとる必要があるのは、以下の場合です。

- 2 次側が新規で、開始するための 1 次側のデータベースのコピーをまだ持っていない場合。
- 2 次側がしばらくの間実行されておらず、そのデータ・コピーが最新のものでない場合。
- 1 次側と 2 次側の両方が同時に「Primary Alone」状態で実行されていたため、データが矛盾している場合。
- 2 次側のディスク・ドライブが破損したか、ファイルが壊れたため、置き換える必要がある場合。

サーバー上のデータの同期化には、主に 2 つの方法があります。キャッチアップとフル・コピーです。

キャッチアップ

キャッチアップを使用できるのは、サーバー同士の接続が切断されていた間、2 次サーバーに「未着」のすべてのトランザクションのコピーを 1 次サーバーが保管していた場合で、また、その場合に限られます。1 次サーバーがそれらのすべてのトランザクションを保管していた場合、1 次サーバーは 2 次サーバーに再接続したとき、自動的にそれらのトランザクションを 2 次サーバーへ転送し、2 次サーバーが 1 次サーバーに「キャッチアップ」できるようにします。

solidDB サーバーは、PRIMARY ALONE 状態にある間だけトランザクションを (2 次サーバーへ転送するために) 保管し、STANDALONE 状態にある間や非 HotStandby サーバーとして作動している間は、保管しません。したがって、最後に 2 次サーバーと接続していたとき以降に STANDALONE 状態にあったか非 HotStandby サーバーとして作動していたサーバーは、すべてのトランザクションを持っているわけではなく、キャッチアップを行うことができません。代わりに、(後述する) 完全コピーを行う必要があります。

明示的な「キャッチアップ」コマンドは存在しません。以下のコマンドを使用してサーバー同士を接続すると、サーバーは自動的にキャッチアップを試みます。

```
ADMIN COMMAND 'hotstandby connect';
```

1 次サーバーと 2 次サーバーを接続すると、それらのサーバーは自動的に、1 次サーバーがトランザクション・ログ内に 2 次サーバーへ送信するデータを持っているかどうかを検査します。そのデータが存在する場合、サーバーは自動的にキャッチアップを試みます。

キャッチアップ・プロセスの間、1 次サーバーと 2 次サーバーは PRIMARY ALONE 状態と SECONDARY ALONE 状態のままです。クライアントは、照会のサブミットとトランザクションのコミットを続行できます。キャッチアップ・プロセスは、クライアント・アプリケーションに意識されることがなく行われます。

1 次サーバーから 2 次サーバーへトランザクションをコピーした後であっても、1 次サーバーと 2 次サーバーのデータベースが同一でないことをサーバーが認識した場合は、エラー・メッセージが出ます。

キャッチアップが失敗した場合（または、例えば 1 次サーバーが STANDALONE 状態にあったなどの理由でキャッチアップが機能しないと前もって分かっている場合）は、完全コピーを実行する必要があります。

キャッチアップは、いずれかの時点で 2 次サーバーが既に SECONDARY ACTIVE 状態で稼働している場合にのみ適用されます。2 次サーバーが新品の場合は、たとえば 1 次サーバーがそれまで PRIMARY ALONE 状態で実行されており、その 1 次サーバー自体が始動した時点以降のすべてのトランザクションを保管している場合でも、2 次サーバーにデータベースの初期コピーを与えるために、完全コピーを行う必要があります。

キャッチアップ・プロセスの詳細は、85 ページの『2 次サーバーのオンラインへの復帰』に説明があります。

完全コピー

完全コピーとは、その名前が示すとおり、すべてのデータを 1 次サーバーから 2 次サーバーにコピーすることです。これは、単数または複数のデータベース・ファイル自体をコピーすることによって行います。

完全コピーは、以下の状況で使用されます。

- 2 次サーバーが新品で、1 次サーバーのデータベースの初期コピーを取得しようとしている場合。
- 1 次サーバーが、PRIMARY ALONE 状態でないときにトランザクションを書き込んだために、キャッチアップが可能でない場合。
- 2 次サーバーのデータベースが壊れたか欠落している場合。
- 2 次サーバーがディスクレスで、障害を起こした場合。2 次ディスクレス・サーバーを障害後に始動したときは、hotstandby netcopy コマンドでデータベースの完全なコピーを行う必要があります。ディスク・ベースの 2 次サーバーと異なり、2 次ディスクレス・サーバーは、トランザクション・ログを読み取って、作動不能だった間に発生した変更を適用することができません。
- 1 次サーバーはキャッチアップに必要なすべてのデータを持っているが、キャッチアップを行うと、単に現在のデータ・ファイルをコピーするよりも長時間かかると予想される場合。

注意:

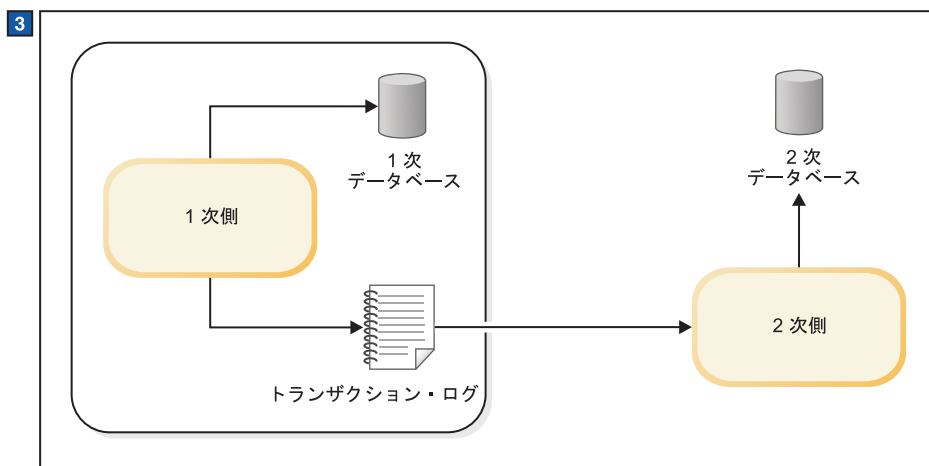
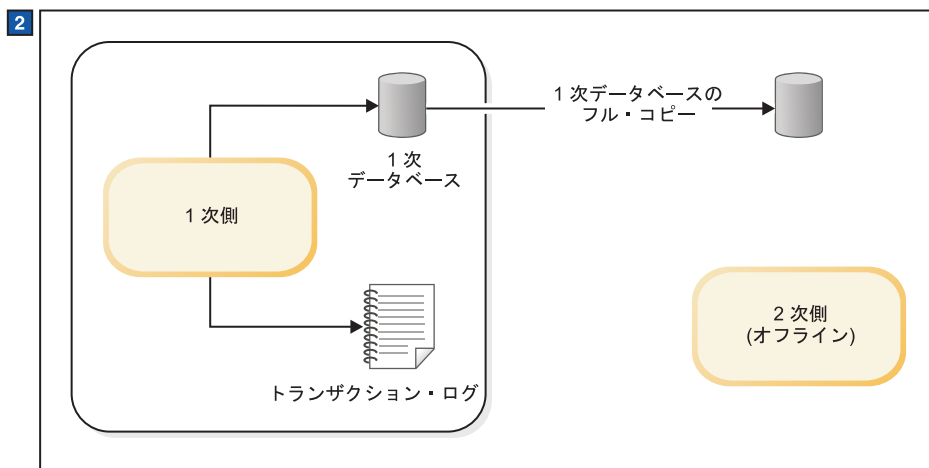
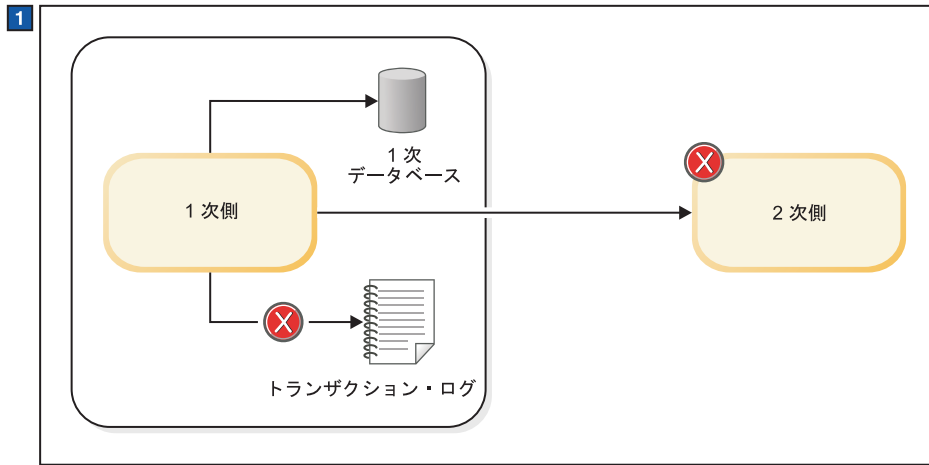
2 次サーバーに古いデータベース・ファイルがある場合、完全コピーを行うと、それらの古いファイルは上書きされます。何らかの理由で、1 次サーバー内になかったデータが 2 次サーバー上のファイルに含まれている場合 (例えば、両方のサーバーが同時に **PRIMARY ALONE** 状態で作動していた場合)、そのデータは失われます。

完全コピーを行うことができる (つまり、データベース・ファイルを 1 次サーバーから 2 次サーバーにコピーする) 2 つの HotStandby コマンドがあります。以下のどちらのコマンドを使用してもかまいません。

```
ADMIN COMMAND 'hotstandby netcopy'; ADMIN COMMAND 'hotstandby copy
[<directory_name>]';
```

netcopy 操作は、ネットワーク上で稼働してファイルを受信できる 2 次サーバーに、ネットワークを通じてデータベースをコピーします。**copy** 操作は、1 次サーバーに可視の指定されたディスク・ドライブ・ディレクトリーに、データベース・ファイルをコピーします。2 次サーバーは **copy** 操作の間、稼働してはなりません。通常、**netcopy** コマンドの方が **copy** コマンドより好ましいので、ほとんどの例で **copy** ではなく **netcopy** のみが示されています。

copy コマンドと **netcopy** コマンドについては、95 ページの『1 次サーバーから指定されたディレクトリーへのデータベース・ファイルのコピー』および 91 ページの『ネットワークを介した 1 次データベースの 2 次側へのコピー』に説明があります。



1. 2次サーバーが長時間ダウンしているために、1次サーバーが2次サーバー用のデータをトランザクション・ログに格納する処理を停止します。このログはまだローカル・リカバリーに使用されます。
2. 1次データベースが2次ノードにコピーされ、1次サーバーがトランザクション・ログへの書き込みを開始します。
3. データベースがコピーされ、1次サーバーがトランザクション・ログ・ファイルの変更を2次サーバーに送信します。

図 14. 手動での完全コピー手順

注:

上の図では、トランザクション・ログの使用が過度に単純化されています。図の最初の部分で、1 次サーバーと 2 次サーバーが接続されていないとき、1 次サーバーは実際にトランザクション・ログへのデータの書き込みを続行しますが、リカバリを行うのに十分なデータだけを保持し、2 次サーバーが接続の切断以降のすべての変更をキャッチアップするために十分なデータを保持するわけではありません。

コピーの検証

copy または netcopy 操作の状況を検証するには、1 次サーバーで以下のコマンドを発行します。

```
ADMIN COMMAND 'hotstandby status copy';
```

操作が netcopy であっても、キーワード「copy」（「netcopy」でなく）を使用することに注意してください。

このコマンドは、copy 操作が成功したのか、まだ進行中なのか、あるいは失敗したのかを知らせる状況メッセージを表示します。失敗の場合は、エラー・コードとエラー・メッセージで示します。

ウォッチドッグを使用したサーバーの同期

手動でサーバーの同期をとることができるコマンドを、ウォッチドッグ・プログラムで使用して、自動的にサーバーの同期をとることもできます。

キャッチアップが十分であれば、ウォッチドッグで行う必要があることは、2 次サーバーをモニターして 2 次サーバーが稼働したことを確認した後、1 次サーバーを 2 次サーバーに接続するコマンドを実行することだけです。完全コピーが必要な場合、ウォッチドッグは 1 次サーバーに netcopy (または copy) 操作を実行するよう指示できます。完全コピーでは、2 次サーバー上のすべてのデータが上書きされることに留意してください。

ネットワークを介した 1 次データベースの 2 次側へのコピー

データベース・ファイルのコピーを 1 次サーバーから 2 次サーバーへ送信するには、netcopy コマンドを使用します。2 次サーバーは、既に稼働している必要があります。

始める前に

重要: コマンドを実行するには、1 次サーバーが PRIMARY ALONE 状態であればなりません。

このタスクについて

netcopy を使用して 2 次サーバー用のデータベース・コピーを作成する状況には、主に以下の 2 つがあります。

- データベースを保持したことがない完全に新規の 2 次サーバー用に、データベースを作成する場合。(この方法は、ディスクレスの 2 次サーバーにデータベース

をコピーするときにも使用されます。そのような 2 次サーバーは、障害を起こした後、データベース全体を失っており、完全に新規の 2 次サーバーとして取り扱う必要があるからです。)

- 既存の 2 次側のデータベース (例えば、破損したもの) を置き換える場合。

`netcopy` を実行するためのコマンドは以下のとおりです。

```
ADMIN COMMAND 'hotstandby netcopy';
```

1 次側は、`netcopy` を行う場合、`solid.ini` の `[HotStandby]` セクションで指定された接続ストリングを使用します。

接続ストリングを定義する **Connect** パラメーターの詳細については、47 ページの『1 次および 2 次 HotStandby 構成の定義』を参照してください。

`hotstandby netcopy` コマンドを実行すると、1 次側のデータベースのコピーを送信する前に、データベース・チェックポイントが実行されます。

1 次側は `netcopy` の間も、トランザクションと接続の受け入れを続行します (ただし、サーバーの状態を変更する `ADMIN COMMAND` は、すべてリジェクトされます)。2 次側は、トランザクションと接続の受け入れを続行しません。`netcopy` が開始されると、2 次側は、オープンの接続またはトランザクションがある場合には、オープン・トランザクションをロールバックし、クライアントからの接続を切断してから、`netcopy` の受信を開始します。2 次側は、`netcopy` を受信している間、1 次サーバーとだけ通信します。

`netcopy` が正常に完了すると、2 次サーバーの状態は `SECONDARY ALONE` に変更されます (まだその状態になっていない場合)。

1 次サーバーは、`netcopy` 操作の間、`PRIMARY ALONE` 状態のままです。`netcopy` が正常に完了した後、1 次サーバーは引き続き同じ状態のままです。完全なホット・スタンバイ動作を再開できるようにするには、事前に 1 次サーバーと 2 次サーバーを接続する必要があり、それによって、1 次サーバーは `PRIMARY ACTIVE` 状態に設定されます。2 つのサーバーの接続方法については、97 ページの『HotStandby サーバーの接続』を参照してください。

2 次サーバー用の新規データベースの作成

通常、`solidDB` サーバーを始動すると、新規データベースを作成するかどうかを尋ねられます (まだデータベースが存在しない場合)。しかし、サーバーが 2 次サーバーである場合は、独自のデータベースを作成せずに、1 次サーバーのデータベースのコピーを使用してください。したがって、既存のデータベースがない 2 次サーバーを始動するときは、コマンド行パラメーターを指定して、1 次サーバーからデータベースのコピーを受信するまで待つよう指示する必要があります。そのコマンド行パラメーターは、**-x backupserver** です。例えば、以下のコマンドで 2 次サーバーを始動します。

```
solid -x backupserver
```

「-x」と「backupserver」の間のスペースは、なくてもかまいません。以下のようにしても同じことです。

```
solid -xbackupserver
```

-x backupserver コマンド行パラメーターは、サーバーに「ネットコピー *listen* モード」(「バックアップ *listen* モード」とも呼ばれます) に入るよう指示します。このモードでは、2 次サーバーに可能な唯一の操作は、1 次サーバーからデータベ

ース・コピーを受信することです。2 次サーバーは、それ以外のどのコマンドにも応答せず、実際に、`solsql` などのクライアント・プログラム、アプリケーション、またはウォッチドッグ・プログラムからの接続要求さえ受け入れません。

2 次データベースが存在する場合は、通常の方法でサーバーを始動でき、その場合、サーバーは `SECONDARY ALONE` 状態になります。

2 次サーバーを `-x backupserver` で始動した後、または 2 次サーバーが `SECONDARY ALONE` 状態にある場合は、1 次サーバー上で `netcopy` コマンドを実行できます。

最初に、1 次サーバーが `PRIMARY ALONE` 状態にあることを確認してください。次に、1 次サーバー上で以下のコマンドを発行します。

```
ADMIN COMMAND 'hsb netcopy';
```

1 次サーバー上で、`hotstandby netcopy` コマンドは `solid.ini` 構成ファイル内の `connect` パラメーターで定義された接続ストリングを使用して、2 次サーバーに接続します。接続後、ネットワーク・リンクを通じてデータベース・ファイルがコピーされます。ネットコピー `listen` モードでは、2 次サーバーは 1 次サーバーでの `hotstandby netcopy` コマンドを通じて新規データベース・コピーを受信した後、2 次側のデータベースを開くことだけを試みます。

以下に、2 次側のデータベース・コピーを作成する手順を示します。

1. `solid.ini` ファイルを、`HotStandby` 構成に有効になるように構成してあることを確認します。接続ストリングを定義する `Connect` パラメーターの詳細については、47 ページの『1 次および 2 次 `HotStandby` 構成の定義』を参照してください。

この接続ストリングは、1 次サーバーから 2 次サーバーに接続し、ネットワークを通じてデータベース・ファイルをコピーするために使用されます。

2. 1 次サーバーを始動します。
3. 以下のコマンドを実行して、2 次サーバーをネットコピー `listen` モードで始動します。

```
solid -x backupserver
```

あるいは、別の方法として、既存のデータベースを使用して 2 次サーバーを始動することもできます。

4. 1 次サーバーを `PRIMARY ALONE` 状態に設定します (まだその状態でない場合)。

```
ADMIN COMMAND 'hotstandby set primary alone';
```

5. 1 次サーバーで、以下のコマンドを発行します。

```
ADMIN COMMAND 'hotstandby netcopy';
```

6. `netcopy` が完了した後、2 つのサーバーを接続し、以下のコマンドで完全なホット・スタンバイ操作を開始 (または再開) できます。

```
ADMIN COMMAND 'hotstandby connect';
```

2 次サーバーは、ネットワーク・リンクを通じてデータベースの新規コピーを受信すると、その 2 次側のデータベースを開きます。サーバー同士が (`hsb connect` コ

マンドで) 接続した後、2 次サーバーは通常の SECONDARY ACTIVE 状態で稼働し、1 次サーバーからユーザー・トランザクションを受け入れることができる状態になります。

HAC を使用した場合、1 次サーバーのデータベースを 2 次サーバーにコピーする手順は、以下のとおりです。

1. 各サーバーが適正な接続パラメーターを備えていることを確認します。接続ストリングを定義する **Connect** パラメーターの詳細については、47 ページの『1 次および 2 次 HotStandby 構成の定義』を参照してください。
2. 1 次サーバー・ノードの HAC に、solidhac.ini 内で **PreferredPrimary=Yes** パラメーターが設定されており、2 次サーバー・ノードの HAC には設定されていないことを確認します。HAC の構成の詳細については、52 ページの『HA コントローラーおよび HA マネージャーの構成』を参照してください。
3. HAC のインスタンスを開始するか、オプションとして、HAC を AUTOMATIC モードに設定します。

注:

SECONDARY ALONE 状態にあるサーバーに netcopy が送信された場合、既存のデータベースはコピーされたデータベースによって上書きされます。このオプションは、データベース同士を再同期させる必要がある場合、または破損した 2 次側のデータベースを修復する必要がある場合に便利です。

2 次サーバー上の既存データベースの置き換え

netcopy は主に、それまでにデータベースがなかった 2 次サーバーにデータベースを送信するために使用されますが、それ以外の状況でも netcopy を使用できます。例えば、2 次サーバーのデータベースのコピーが (ディスク・ドライブの破損などによって) 壊れた場合、netcopy コマンドを使用して、2 次サーバーに 1 次サーバーのデータベースを送信できます。

始める前に

既存のデータベースを置き換える場合、2 次サーバーを「ネットコピー listen モード」にする必要はありません。つまり、2 次サーバーを **-x backupserver** で始動する必要はありません。

このタスクについて

1. 1 次サーバーが PRIMARY ALONE 状態で、2 次サーバーが SECONDARY ALONE 状態であることを確認します。
2. 1 次サーバーに以下のコマンドを発行します。

```
ADMIN COMMAND 'hotstandby netcopy';
```

netcopy が完了した後、1 次サーバーは引き続き PRIMARY ALONE 状態で、2 次サーバーは自動的に SECONDARY ALONE 状態に置かれる (まだその状態になかった場合) ことに注意してください。

2 次サーバーが SECONDARY ALONE 状態にあるときに netcopy を実行した場合、2 次サーバーにクライアントが (読み取り専用照会を行うために) 接続していると、2 次サーバーはすべてのオープン・トランザクションをロールバック

し、すべてのクライアント接続を切断します。 netcopy が完了した後、2 次サーバーは SECONDARY ALONE 状態のままです。

3. サーバーは自動的に接続しないので、以下のコマンドを発行する必要があります。

```
ADMIN COMMAND 'hotstandby connect';
```

ネットコピー状況の検証

netcopy コマンドを開始すると、バックグラウンドで非同期にネットコピーが実行されます。サーバーは netcopy の完了時にメッセージを表示しません。

このタスクについて

実際に、サーバーはネットワーク・エラーなどの問題で netcopy が失敗した場合でも、メッセージさえ表示しません。netcopy が正常に完了したかどうかを知るには、常に netcopy の状況を検証する必要があります。

- 1 次サーバーで、以下のコマンドを発行します。

```
ADMIN COMMAND 'hotstandby status copy';
```

注:

このコマンドでは、キーワードに「netcopy」でなく「copy」を使用します。コピー操作とネットコピー操作の両方に同じコマンドを使用します。

このコマンドは、netcopy が成功したのか、まだ進行中なのか、あるいは、失敗したのかを知らせる状況メッセージを表示します。失敗の場合は、エラー・コードとエラー・メッセージで示します。

1 次サーバーから指定されたディレクトリーへのデータベース・ファイルのコピー

2 次側のデータベースに使用されるディレクトリーが 1 次側に可視の場合は、hotstandby copy コマンドを使用してデータベースを 1 次側のディレクトリーから 2 次側のディレクトリーにコピーできます。

始める前に

このタスクを実行できるのは、1 次サーバーと 2 次サーバーがいくつかの同じディスク・ドライブを認識でき、したがって、いくつかの同じディレクトリーの読み取りと書き込みができる場合だけです。

注意:

hotstandby copy コマンドを使用する前に、必ず 2 次サーバーをシャットダウンしてください。1 次サーバーがデータベース・ファイルに書き込みを行っている間、2 次サーバーは、そのデータベース・ファイルにアクセスを試みてはなりません。

注: hotstandby copy コマンドを発行するとき、1 次サーバーは PRIMARY ALONE 状態にあることが必要で、コマンドの実行中 (およびコマンドの後)、1 次サーバーはその状態のままになります。

このタスクについて

hotstandby copy コマンドと hotstandby netcopy コマンドの主な相違点の 1 つは、netcopy コマンドが 2 次サーバーの実行中にのみ使用できるのに対し、copy コマンドは 2 次サーバーの実行中以外にのみ使用する必要があることです。パフォーマンスの面では、この 2 つのデータベース・コピー方式に大きな違いはありません。

1. hotstandby copy を使用してファイルをコピーするには、1 次サーバー側で次のコマンドを発行します。

```
ADMIN COMMAND 'hotstandby copy[directory_name]';
```

ここで、*directory_name* はファイルのコピー先にしたいディレクトリーの名前です。ディレクトリー名の形式は、オペレーティング・システムによって異なります。ディレクトリー名はオプションです。ディレクトリー名を指定しなかった場合、サーバーは solid.ini 構成ファイル内の **CopyDirectory** パラメーターで指定された値を使用します。

hotstandby copy コマンドを実行すると、データベースにチェックポイントが作成され、次に 1 次側のデータベースのコピーが作成された後、そのコピーが 2 次側へ送信されます。

サーバーは PRIMARY ALONE 状態にあるため、1 次側でのトランザクション処理は copy コマンドの間も通常どおり続行され、1 次側はトランザクションをトランザクション・ログに保管して、後で 2 次側へ転送できるようにします。

2. コピー操作の後も、2 次側はまだダウンしたままです。それを復帰させた後に、hotstandby connect コマンドを発行して、2 つのサーバーを接続する必要があります。

1 次サーバーのデータベースが管理コマンドの hotstandby connect を使用して 2 次側に接続されると、1 次と 2 次のサーバーは自動的に「キャッチアップ」を実行して、2 次サーバーを最新の状態にします。

2 次サーバーの始動とキャッチアップ

コピーが完了したら、新規にコピーしたデータベースを使用して 2 次サーバーを始動するする必要があります。

1. 通常と同じ方法でサーバーを始動します。つまり、オペレーティング・システムのプロンプトでコマンド「solid」を発行します。
2. 2 次サーバーを再始動した後、1 次サーバーで hotstandby connect コマンドを使用して、1 次サーバーを 2 次サーバーに接続します。

```
ADMIN COMMAND 'hotstandby connect';
```

hotstandby connect コマンドの詳細については、97 ページの『HotStandby サーバーの接続』に説明があります。

タスクの結果

1 次サーバーを 2 次サーバーに接続した後、1 次サーバーと 2 次サーバーは自動的にキャッチアップの実行を開始します。これは、1 次サーバーが 2 次サーバーのデータベースを最新のものにするために、1 次サーバーのトランザクション・ログを 2 次サーバーへコピーし、2 次サーバーは、トランザクション・ログをロールフ

ワードして、自身のデータベースのコピーを更新することを意味しています。

HotStandby サーバーの接続

このタスクについて

1 次サーバーが 2 次サーバーに接続するために使用する connect ストリングは、solid.ini 構成ファイルの [hotstandby] セクションで **Connect** パラメーターによって指定します。

1 次側ノードと 2 次側ノードの現行の接続設定は、次のコマンドを発行することによって表示できます。

```
ADMIN COMMAND 'hotstandby cominfo';
```

1 次側と 2 次側の間接続が切れているか、まだ確立されていない場合は、以下のコマンドを 1 次側または 2 次側のノードで発行する必要があります。

```
ADMIN COMMAND 'hotstandby connect';
```

例えば、netcopy を実行した後、通常はサーバー同士を接続します。

HotStandby サーバーには自動接続メカニズムがないため、サーバー間の接続が切れているときは、高可用性制御アプリケーションにこのコマンドを実行させてください。

このコマンドを発行した後、1 次サーバーと 2 次サーバーの接続が成功した場合は、確認メッセージが表示されます。1 次サーバーと 2 次サーバーが接続されていても、トランザクション・ログが 2 次側でまだ完全にはコピーされていない場合、ユーザーは 1 次サーバーから「Started the process of connecting the servers」というメッセージを受け取ります。

コマンドを実行したときに 1 次サーバーの状態が PRIMARY UNCERTAIN または PRIMARY ALONE であった場合、接続が成功すると、1 次サーバーの状態が PRIMARY ACTIVE に変わります。成功しないと、状態は PRIMARY UNCERTAIN または PRIMARY ALONE のままです。

次のタスク

1 次サーバーおよび 2 次サーバーで接続状態を照会する方法の詳細については、98 ページの『接続状況情報の表示』を参照してください。アプリケーションを 1 次サーバーに再接続する方法の詳細については、73 ページの『アプリケーションから 1 次サーバーへの再接続』を参照してください。

HotStandby 状況の検査

このセクションでは、1 次サーバーと 2 次サーバーの両方に対して要求できる HotStandby 状況情報について説明します。

状況を確認するには、1 次または 2 次サーバーで次のコマンドを発行します。

```
ADMIN COMMAND 'hotstandby status option';
```

ここで、*option* は、以下のいずれかを指定できます。

オプション	説明
catchup	サーバーがキャッチアップを実行するかどうかを指示します。キャッチアップは、1 次サーバーが 2 次サーバーに接続した後に行われます。キャッチアップの間、1 次サーバーは 2 次サーバーが変更を適用できるよう、蓄積したトランザクション・ログを送信します。可能な値は、「ACTIVE」と「NOT ACTIVE」です。
connect	前回サーバー同士を接続しようとして成功したかどうかを示します。
copy	前回の copy/netcopy の試みが成功したかどうかを示します。
switch	前回、サーバーを PRIMARY ACTIVE 状態または SECONDARY ACTIVE 状態に切り替えようとして成功したかどうかを示します。

例

```
ADMIN COMMAND 'hotstandby status catchup';
```

切り替え状況情報の表示

2 つの HotStandby サーバー間で、状態切り替えが発生したかどうかを検証する必要があります。が生じる場合があります。

このタスクについて

HotStandby 切り替え状況情報を検査するには、以下のようにします。

1 次または 2 次サーバーで、以下のコマンドを発行します。

```
ADMIN COMMAND 'hotstandby status switch';
```

- 以前に 2 つのサーバー間で切り替えが発生したことがない場合は、NO SERVER SWITCH OCCURRED BEFORE というメッセージが表示されます。
- 切り替えプロセスがまだアクティブである場合は、ACTIVE というメッセージが表示されます。
- 以前の切り替えプロセスのうち最新のものが正常に完了している場合は、SUCCESS というメッセージが表示されます。
- 最新の切り替えの試みが失敗している場合は、ERROR *number* というメッセージが表示されます。ここで、*number* は切り替えのときに発生したエラーのタイプを示しています。

接続状況情報の表示

1 次サーバーと 2 次サーバーの間の接続状況情報を照会できます。この機能は、アプリケーション・コード内で使用できる SQL 関数 HOTSTANDBY_CONNECTSTATUS と同じものです。

接続状況を確認するには、1 次サーバーまたは 2 次サーバーで次のコマンドを発行します。

```
ADMIN COMMAND 'hotstandby status connect';
```

可能な戻り値は、以下のとおりです。

表 16. 接続状況の戻り値

エラー・コード	テキスト	説明
0	CONNECTED	接続アクティブ。1 次と 2 次の両方のサーバーから返されます。
14007	CONNECTING	1 次サーバーが 2 次サーバーに接続中です。1 次と 2 次の両方のサーバーから返されます。
14008	CATCHUP	1 次サーバーが 2 次サーバーに接続しましたが、トランザクション・ログはまだ完全にはコピーされていません。1 次と 2 次の両方のサーバーから返されます。
14010	DISCONNECTING	サーバーは切断処理中です。
14537	BROKEN	接続が切れています。1 次と 2 次の両方のサーバーから返されます。

通信情報の表示

他のサーバーに接続するために使用した通信情報を照会できます。これは、solidDB 構成ファイル (solid.ini) の [HotStandby] セクションにある **Connect** パラメーター設定の値です。この情報をクライアント・アプリケーションで使用して、他のサーバーに接続できます。

通信情報を表示するには、1 次または 2 次サーバーで次のコマンドを発行します。

```
ADMIN COMMAND 'hotstandby cominfo';
```

ロール開始時刻の表示

場合によっては、いつサーバーが現在の 1 次サーバーまたは 2 次サーバーのロールに入ったかを知ることが重要になります。

この情報をリトリブするには、以下のような ADMIN COMMAND 'info' の 2 つの対応するオプションを使用します。

```
admin command 'info primarystarttime';
RC TEXT
-- ----
0 2005-06-09 14:22:18
admin command 'info secondarystarttime';
RC TEXT
-- ----
0 2005-06-09 18:24:44
```

報告される時刻は、ロールが 1 次サーバーまたは 2 次サーバーになった時刻です。STANDALONE 状態は、1 次サーバー・ロールの状態であると考えられます。

具体的には、1 次サーバー開始時刻は、以下のトランザクションが発生したときに設定されます。

- SECONDARY ALONE => PRIMARY ALONE
- SECONDARY ALONE => STANDALONE
- SECONDARY ACTIVE => PRIMARY ACTIVE

2 次サーバー開始時刻は、以下の時点で設定されます。

- サーバーが SECONDARY ALONE 状態で始動されたとき
- OFFLINE (-x backupserver で始動) => SECONDARY ALONE
- PRIMARY ALONE => SECONDARY ALONE
- STANDALONE => SECONDARY ALONE
- PRIMARY ACTIVE => SECONDARY ACTIVE

現在のロールが照会と矛盾する場合、照会は空ストリングを返します。例えば、ロールが SECONDARY で、コマンド 'info primarystarttime' が発行された場合は、空ストリングが返されます。

HotStandby サーバー状態の検証

HotStandby の管理または保守を行うときは、多くの場合、HotStandby サーバーの状態を確認する必要があります。

このタスクについて

HotStandby サーバーの現在の状態を確認するには、そのサーバーで以下の HotStandby コマンドを発行します。

```
ADMIN COMMAND 'hotstandby state';
```

次のタスク

コマンドが返す可能性がある状態については、13 ページの『サーバー状態の説明』を参照してください。

注:

1.

サーバーの solid.ini 構成ファイルが、そのサーバーを HotStandby サーバーにするように構成されている場合、そのサーバーは、始動された時点では SECONDARY ALONE 状態で始動します。

2.

上記の表に OFFLINE 状態がリストされていないことに注意してください。サーバーは、状態名「OFFLINE」に戻ることはできません。サーバーが OFFLINE 状

態になると、それに接続して照会 (例えば、ADMIN COMMAND 'hotstandby state' など) を発行できなくなるからです。

3.

HotStandby 用に構成されていないサーバー上で ADMIN COMMAND 'hotstandby state' を発行すると、以下のエラー・メッセージが返されます。

```
14527: This is a non-HotStandby Server
```

HAC および HA マネージャーを使用している場合、HA マネージャーは両方のサーバーの HSB 状態を表示します。この情報は、秒刻みで定期的に照会されます。

サーバー状態の組み合わせ

サーバー状態のすべての組み合わせが可能なわけではありません。例えば、2 次サーバーは 1 次サーバーが PRIMARY ACTIVE 状態の場合、SECONDARY ACTIVE 状態にのみなることができます。

以下の表は、関連付けられているサーバーが特定の状態にあるときに、HotStandby サーバーに可能となるサーバー状態を示しています。

表 17. サーバー状態

サーバーの状態	関連付けられているサーバーに可能な状態
PRIMARY ACTIVE	SECONDARY ACTIVE
PRIMARY ALONE	PRIMARY ALONE * PRIMARY UNCERTAIN SECONDARY ALONE STANDALONE *
PRIMARY UNCERTAIN	PRIMARY ALONE PRIMARY UNCERTAIN SECONDARY ALONE STANDALONE
SECONDARY ACTIVE	PRIMARY ACTIVE
SECONDARY ALONE	PRIMARY ALONE PRIMARY UNCERTAIN SECONDARY ALONE STANDALONE
STANDALONE	PRIMARY ALONE * PRIMARY UNCERTAIN SECONDARY ALONE STANDALONE *

* 一方のサーバーが PRIMARY ALONE 状態または STANDALONE 状態の場合、もう一方のサーバーが PRIMARY ALONE 状態または STANDALONE 状態にならないようにしてください。その理由は、両方のサーバーに別々の変更が加えられた場合、2 つのデータベースを 1 つにマージする方法がないためです。

どちらのサーバーを 1 次サーバーにするかの選択

両方のデータベースが障害を起こし、その障害からのリカバリーを試みる場合、状況によっては、どのサーバーを 1 次サーバーにすればよいか分からないことがあります。サーバー同士の互いの接続が失われる前に 1 次サーバーだったサーバーが、必ずしも、この時点で 1 次サーバーになるべきサーバーだとは限りません。

このタスクについて

どちらのサーバーを 1 次サーバーにすべきかを判別するには、各サーバー上で以下のコマンドを使用できます。

```
ADMIN COMMAND 'hsb logpos';
```

この機能は、値をストリングか 2 進値として返します。「より大きい」値を持つサーバー（より多くのトランザクションを受け入れていたサーバー）が、1 次サーバーになるべきサーバーです。両方のサーバーの `logpos` 値が等しい場合は、**PreferredPrimary** パラメーターによって、ローカル・サーバーが 1 次サーバーになるべきかどうか定義されます。

コマンドを使用するには、以下の手順を実行します。

1. 両方のサーバーが稼働中で、SECONDARY ALONE 状態であることが必要です。
2. 両方のサーバーに接続します。
3. 各サーバーで、以下を実行します。

```
ADMIN COMMAND 'hsb logpos';
```

成功した管理コマンドは、エラー・コード 0 とストリング、およびサーバーの前のロールを返します。（注: アプリケーションは、このストリングを、定義された構造を持たず、内部が見えない値と見なす必要があります。）

4. ストリング値を比較します。

例えば、C では `strcmp()` 関数を使用します。「より大きい」ストリングを返したサーバーが、新規 1 次サーバーとして選択されるべきサーバーです。ストリングが等しい場合は、**PreferredPrimary** パラメーターによって、ローカル・サーバーが 1 次サーバーになるべきかどうか定義されます。

5. 1 次サーバーになるサーバー上で、以下のコマンドを使用して 1 次サーバーを選択します。

```
ADMIN COMMAND 'hsb set primary alone';
```

6. 以下のコマンドを使用して、HotStandby サーバーを互いに接続します。

```
ADMIN COMMAND 'hsb connect';
```

7. 前記のコマンドが成功した場合、2 次サーバーは 1 次サーバーをキャッチアップし、HotStandby ペアが再び機能するようになります。コマンドが失敗した場合は、1 次サーバー上で以下のコマンドを発行することにより、各ノードを別々に同期させる必要があります。

```
ADMIN COMMAND 'hsb netcopy';
```

`netcopy` コマンドは、終了時に戻り値を返しません。代わりに、ユーザーが能動的に監視する必要があります。これは、以下のコマンドで行うことができます。

```
ADMIN COMMAND 'hsb status copy';
```

可能な戻り値は、ACTIVE、SUCCESS、または FAILED です。失敗の場合は、障害の理由も出力されます。同期化が行われた後、次のコマンドを発行してください。

```
ADMIN COMMAND 'hsb  
connect';
```

HotStandby ペアが再び機能するようになります。

注意:

この手順では、大きい方のストリング値を持つサーバーが他方のサーバーのスーパーセットになることは保証されません。2 つのサーバーのそれぞれが、他方が受け入れなかったトランザクションを受け入れた可能性は依然として存在し、例えば、両方のサーバーが PRIMARY ALONE 状態で稼働していた可能性があります。各サーバーは、どちらのサーバーも他方のスーパーセットでない可能性を検出するために、「connect」コマンドを実行するときに情報を比較します。どちらのサーバーもスーパーセットでない場合、Connect コマンドは失敗し、該当するエラー・メッセージが出力されます。

HotStandby サーバーから非 HotStandby サーバーへの変更

solid.ini ファイルの [HotStandby] セクションを編集することにより、1 次サーバーまたは 2 次サーバーを通常の非 HotStandby サーバーに変更できます。

1. **HSBEnabled** パラメーターを除去 (または、「no」に設定) します。
2. (オプション) **Connect** パラメーターを除去またはコメント化します。
3. solid.ini ファイル内のパラメーター設定を変更した後、変更を有効にするためには、サーバーを再始動する必要があります。

例

一時的に HotStandby サーバーとしての機能を停止し、後で HotStandby サーバーとしての機能を再開させたい場合は、solid.ini ファイルを変更しないでおき、代わりに、サーバーの状態を単に STANDALONE に変更することもできます。

6 障害が起きた場合の高可用性コントローラーの動作

このセクションでは、起こりうる障害のシナリオについて説明します。高可用性コントローラー (HAC) は、障害のシナリオを暗黙に処理します。ただし、人間の管理者または「ウォッチドッグ」と呼ばれるソフトウェア・プログラムは、さまざまな障害や初期化のシナリオ (要するに管理シナリオ) を処理できます。

ウォッチドッグは 1 次サーバーと 2 次サーバーをモニターする別個のプログラムで、必要な場合には、それらのサーバーの状態を変更するコマンドを発行します。1 次サーバーまたは 2 次サーバー自体が障害を起こした場合、またはそれらのサーバー間の通信リンクがダウンした場合にそれを判別できるよう、HAC を使用することをお勧めします。

リカバリーの目的は、障害を起こしたコンポーネントを操作に戻すことです。ときには、リカバリーの最中にさらに障害が発生します。通常、それらの障害が起きると、システムの可用性が制限されたまま (1 つのサーバーしか稼働せず)、人間の介入を待つ状態になります。よくあるリカバリー時の障害で、自動的に処理されないものは、以下のとおりです。

- 障害を起こしたデータベースが、再開が不可能なところまで壊れている
- キャッチアップを実行するのに十分な空きディスク・スペースがない

1 次サーバーのデータベースの障害

シナリオ

ノード 1 にある (PRIMARY ACTIVE 状態の) 1 次データベースが障害を起こします。

ノード 2 にある (SECONDARY ACTIVE 状態の) 2 次データベースが、ノード 1 の 1 次データベースへの接続障害を検出します。

リカバリー

1 次サーバーのデータベース障害からのリカバリーで、2 次サーバーは、1 次サーバーに置き換わります。リカバリーは、以下のように自動的に進行します。

1. 接続に障害が起きると、ノード 2 にある 2 次サーバーのデータベースは、自動的に SECONDARY ALONE 状態に移行します。
2. ノード 2 にある 2 次サーバーのデータベース上の HAC インスタンスは、ノード 1 にある 1 次サーバーのデータベースが障害を起こしたと結論付け、ノード 2 にある 2 次サーバーのデータベースを PRIMARY ALONE 状態に設定します。
3. 上記のタスクと並行して、ノード 1 にある 1 次サーバーのデータベース上の HAC インスタンスは、1 次サーバーのデータベースを再開し、そのデータベースは SECONDARY ALONE 状態に入ります。
4. ノード 2 にある 2 次サーバーのデータベース上の HAC インスタンスは、1 次サーバーと 2 次サーバーのデータベースの接続プロセスを開始します。

5. キャッチアップが行われます。

オプションとして、接続プロセスにはノード 2 にある 2 次サーバーのデータベースからノード 1 にある 1 次サーバーのデータベースへの netcopy 操作が含まれます。

2 次サーバーのデータベースの障害

シナリオ

ノード 2 にある (SECONDARY ACTIVE 状態の) 2 次データベースが障害を起こします。

ノード 1 にある (PRIMARY ACTIVE 状態の) 1 次サーバーのデータベースが、ノード 2 にある 2 次サーバーのデータベースへの接続障害を検出します。

リカバリー

2 次サーバーのデータベース障害からのリカバリーで、2 次サーバーが再始動されます。リカバリーは、以下のように自動的に進行します。

1. 接続に障害が起きると、ノード 1 にある 1 次サーバーのデータベースは自動的に PRIMARY UNCERTAIN 状態に移行するか、**AutoPrimaryAlone** パラメーターが使用可能に設定されている場合は PRIMARY ALONE 状態に移行します。
2. ノード 1 にある 1 次サーバーのデータベース上の HAC インスタンスは、ノード 2 にある 2 次サーバーのデータベースが障害を起こしたと結論付けます。
3. 1 次サーバーのデータベースがステップ 1 で PRIMARY UNCERTAIN 状態に設定された場合、HAC はこの時点でそれを PRIMARY ALONE 状態に設定します。
4. 上記のタスクと並行して、ノード 2 にある 2 次サーバーのデータベース上の HAC インスタンスは、2 次サーバーのデータベースを再開し、そのデータベースは SECONDARY ALONE 状態に入ります。
5. ノード 1 にある 1 次サーバーのデータベース上の HAC インスタンスは、1 次サーバーと 2 次サーバーのデータベースの接続プロセスを開始します。
6. キャッチアップが行われます。

オプションとして、接続プロセスには 1 次サーバーのデータベースから 2 次サーバーのデータベースへの netcopy 操作が含まれます。

1 次ノードの障害

シナリオ

- 1 次ノード (ノード 1) が障害を起こします。

ノード 2 にある (SECONDARY ACTIVE 状態の) 2 次サーバーのデータベースが、ノード 1 にある (PRIMARY ACTIVE 状態の) 1 次サーバーのデータベースへの接続障害を検出します。

リカバリー

1 次サーバー・ノードの障害からのリカバリーで、1 次サーバーが再始動されます。リカバリーは、以下のように自動的に進行します。

1. 接続に障害が起きると、2 次サーバーのデータベースは、自動的に SECONDARY ALONE 状態に移行します。
2. ノード 2 にある 2 次サーバーのデータベース上の HAC インスタンスは、ノード 1 にある 1 次サーバーのデータベースが障害を起こしたと結論付けます。
3. ノード 2 にある 2 次サーバーのデータベース上の HAC インスタンスは、2 次サーバーのデータベースを PRIMARY ALONE 状態に設定します。
4. 1 次サーバー・ノード (ノード 1) が再始動されます。
5. 1 次サーバーのデータベース (ノード 1) 上の HAC インスタンスが再始動されます。
6. 1 次サーバーのデータベース (ノード 1) 上の HAC インスタンスは、1 次サーバーのデータベースが稼働していないと結論付けます。
7. 1 次サーバーのデータベース (ノード 1) 上の HAC インスタンスは、1 次サーバーのデータベースを再開し、それを SECONDARY ALONE 状態に設定します。
8. ノード 2 にある 2 次サーバーのデータベース上の HAC インスタンスは、1 次サーバーと 2 次サーバーのデータベースの接続プロセスを開始します。
9. キャッチアップが行われます。

オプションとして、接続プロセスには 1 次サーバーのデータベースから 2 次サーバーのデータベースへの netcopy 操作が含まれます。

2 次ノードの障害

シナリオ

2 次ノード (ノード 2) が障害を起こします。

ノード 1 にある (PRIMARY ACTIVE 状態の) 1 次サーバーのデータベースが、ノード 2 にある (SECONDARY ACTIVE 状態の) 2 次サーバーのデータベースへの接続障害を検出します。

リカバリー

2 次サーバー・ノードの障害からのリカバリーで、2 次サーバーが再始動されません。リカバリーは、以下のように自動的に進行します。

1. 接続に障害が起きると、ノード 1 にある 1 次サーバーのデータベースは、自動的に PRIMARY UNCERTAIN 状態に移行します。
2. ノード 1 にある 1 次サーバーのデータベース上の HAC インスタンスは、ノード 2 にある 2 次サーバーのデータベースが障害を起こしたと結論付けます。
3. ノード 1 にある 1 次サーバーのデータベース上の HAC インスタンスは、1 次サーバーのデータベースを PRIMARY ALONE 状態に設定します。
4. 2 次サーバー・ノード (ノード 2) が再始動されます。

5. 2 次サーバーのデータベース (ノード 2) 上の HAC インスタンスが再始動されます。
6. 2 次サーバーのデータベース (ノード 2) 上の HAC インスタンスは、2 次サーバーのデータベースが稼働していないと結論付けます。
7. 2 次サーバーのデータベース (ノード 2) 上の HAC インスタンスは、2 次サーバーのデータベースを再開し、それを SECONDARY ALONE 状態に設定します。
8. ノード 1 にある 1 次サーバーのデータベース上の HAC インスタンスは、1 次サーバーと 2 次サーバーのデータベースの接続プロセスを開始します。
9. キャッチアップが行われます。

オプションとして、接続プロセスには 1 次サーバーのデータベースから新しい 2 次サーバーのデータベースへの netcopy 操作が含まれます。

HotStandby リンクの障害

シナリオ

HotStandby リンクが障害を起こします。

ノード 1 にある (PRIMARY ACTIVE 状態の) 1 次サーバーのデータベースと、ノード 2 にある (SECONDARY ACTIVE 状態の) 2 次サーバーのデータベースの両方が、相互の接続の障害を検出します。

リカバリー

HotStandby リンクの障害からのリカバリーで、各 HAC インスタンスは障害を起こしたのがネットワークなのか、それとも相手側サーバーなのかを知るために、外部参照エンティティ (ERE) に対して ping を実行します。リカバリーは、以下のように自動的に進行します。

1. 接続に障害が起きると、両方のデータベースは自動的にそれぞれ PRIMARY UNCERTAIN (ノード 1) と SECONDARY ALONE 状態 (ノード 2) の状態に移行します。
2. 両方の HAC インスタンスからリモート・サーバーへの直接接続は、失敗します。
3. 両方の HAC インスタンスは、オペレーティング・システムの ping ユーティリティを使用して、ERE に対して ping を実行します。
4. ノード 2 にある 2 次サーバーのデータベースが再開されます。
5. ping が失敗した場合、ローカル・サーバーはそのままであるか、SECONDARY ALONE 状態に設定されます。
6. ping が成功した場合、成功した HAC はリモート・データベース・サーバーに接続しようとしています。
7. リモート・データベース・サーバーへの接続の試みが失敗した場合、HAC はそのネットワーク接続の部分は操作可能であると結論付け、ローカル・サーバーを PRIMARY ALONE 状態に設定します。

8. 1 次サーバーのデータベース上の HAC インスタンスは、2 次サーバーのデータベースへの接続を再確立しようとします。
9. ネットワークが操作可能になり、接続が成功した場合、1 次サーバーのデータベースと 2 次サーバーのデータベースは自動的にそれぞれ、PRIMARY ACTIVE 状態と SECONDARY ACTIVE 状態に移行します。

7 HotStandby の使用によるサーバーのアップグレード

コールドおよびホット・マイグレーション

マイグレーションには、ソフトウェアのバージョンの更新が伴います。solidDB HotStandby のような高可用性システムでは、マイグレーションは「コールド」の場合と「ホット」場合があります。

コールド・マイグレーションは、従来のマイグレーション方法です。システム全体(両方のサーバー)をシャットダウンし、新しいソフトウェアと構成データを使用して再始動します。

solidDB の高可用性設計により、各サーバーのアップグレードを行っている間、システム全体をオフラインにしなくても、solidDB サーバーをアップグレードできます。一方のサーバーを更新している間、もう一方のサーバーを稼働させておくことができます。これは、ホット・マイグレーションと呼ばれます。

注意:

システム全体はダウンしませんが、基本接続を使用している場合、ユーザーまたはアプリケーションは一方のサーバーから切断し、もう一方のサーバーに接続しなければならない場合があります。透過接続を使用している場合は、ユーザーおよびアプリケーションに意識されることなく、ホット・マイグレーションを行うことができます。

HSB 互換バージョン間のマイグレーション

ソフトウェアのバージョンが HSB 互換の場合、それぞれのサーバーのバージョンが異なってもよく、それらのサーバー間で相互に通信できる場合があります。例えば、solidDB のどのメジャー・バージョンでも、すべてのリリースが HSB 互換です。

コールド・マイグレーション

コールド・マイグレーションは、ごく普通のもので、システム全体をシャットダウンし、新しいソフトウェアを使用して、シャットダウン前と同じロールでサーバーを再始動します。

ホット・マイグレーション

ホット・マイグレーションの基本的なアウトラインは、以下のとおりです。

1. 2 次サーバーを切断し、シャットダウンしてから 2 次サーバーをアップグレードします。
2. 旧 2 次サーバーを新規 1 次サーバー (PRIMARY ALONE 状態) になるように設定します。次に、旧 1 次サーバーをシャットダウンし、旧 1 次サーバーをアップグレードします。

3. 旧 1 次サーバーを新規 2 次サーバーとして復帰させます。新規 1 次サーバーと新規 2 次サーバーを接続し、新規 2 次サーバーを新規 1 次サーバーに「キャッチアップ」させます。

この手順により、サーバー同士が逆転することに注意してください。この一連の手順が終わると、当初 1 次サーバーだったサーバーが 2 次サーバーになり、また、その逆も同様です。

HSB 非互換バージョン間のマイグレーション

バージョンに互換性がない場合は、システムのアップグレードの際に特定のシナリオに従う必要があります。

具体的には、変換コマンド行パラメーターを使用する必要があります。

マイグレーションが可能であっても、このバージョンと HSB 非互換のバージョンは、3.1、3.7、および 4.0 です。3.1 より前のバージョンからのマイグレーションは、サポートされていません。

詳細な段階的な手順を以下に示します。

HSB 非互換バージョン間のマイグレーションの準備手順

この例では、アプリケーションがそれぞれの接続の状態を検知して、新規 1 次サーバーにフェイルオーバーすることを想定しています。このため、制御されたサーバー切り替えにより、アプリケーションは分断されません。ただし、オープン・トランザクションが切り替え中に打ち切られた場合は除きます。

1. アプリケーションが自動的にフェイルオーバーするように設計されていない場合は、接続が失われること、および新規 1 次サーバーに再接続する必要があることを、ユーザーに通知する必要があります。
2. システムとソフトウェアをアップグレードのために準備します。特に、以下のよう作業を行う必要があります。
 - a. アップグレード操作の一部で、それぞれの solidDB サーバーが単独で (特に PRIMARY ALONE 状態で) 作動することになるため、両方のコンポーネントが「健全」な状態であること、例えば、十分な空きディスク・スペースがあること、ネットワーク接続が信頼できること、電源障害に備えて UPS を装備していることなどを確認する必要があります。
 - b. それぞれのサーバーは、もう一方のサーバーがアップグレードされている間、少なくともある短い時間、PRIMARY ALONE 状態で作動します。サーバーは、PRIMARY ALONE 状態にある間、トランザクションをトランザクション・ログに保管します。もう一方のサーバーがアップグレードされている間 (そのサーバーが再始動後に「キャッチアップ」に要する時間も含む) に発生するすべてのトランザクションを保管するために、ログ・ファイル用の十分なディスク・スペースが使用可能でなければなりません。
 - c. アップグレード・ソフトウェアのコピーが各コンピューター上にあるか、すぐに入手可能であることを確認します。

3. 注意:

「ウォッチドッグ」プログラムがある場合は、そのウォッチドッグを一時的にオフにしてください。これは、アップグレード・プロセス中に発行するコマンドと競合するコマンドを、ウォッチドッグが発行しないようにするためです。

例えば、1 次サーバーを 2 次サーバーから切断した後、2 次サーバーをアップグレードする前にウォッチドッグがサーバー同士の再接続を試みないようにしたいからです。

コールド・マイグレーション手順

このタスクについて

マイグレーション・ステップは以下のとおりです。

1. サーバー同士を切断し、両方ともシャットダウンします。
2. 新バージョンのソフトウェアをインストールします。
3. solid.ini ファイルを更新します。
4. コマンド行パラメーター `-x autoconvert` を指定して 1 次サーバーを始動します。このパラメーターは、サーバーに既存のデータベースを新しいフォーマットに変換するよう指示します。
5. 1 次サーバーを PRIMARY ALONE 状態に設定します。
6. 1 次サーバーから 2 次サーバーへの 'hsb copy' または 'hsb netcopy' を実行します。
7. サーバー同士を接続します。

ホット・マイグレーション手順

このタスクについて

以下の手順で、S1 (「OP」) および S2 (「OS」) は、当初の 1 次サーバーと 2 次サーバーを表しています。各サーバーの状態は、このプロセスを進めるにつれて変化します。

1. S1: 1 次サーバーを 2 次サーバーから切断します。

```
ADMIN COMMAND 'hsb set broken';  
ADMIN COMMAND 'hsb status connect ping';
```

2. S2 OS: サーバー S2 (2 次サーバー) をシャットダウンします。

```
ADMIN COMMAND 'shutdown force';
```

3. S1 OP: サーバー S1 (1 次) サーバーに、PRIMARY ALONE 状態で作動するよう指示します (まだ自動的にその状態に切り替わっていない場合)。サーバー S1 (1 次サーバー) が PRIMARY ALONE 状態にあることを検証します。

```
ADMIN COMMAND 'hsb set primary alone';  
ADMIN COMMAND 'hsb state';
```

4. S2 OS: サーバー S2 (当初の 2 次サーバー) をアップグレードします。

ソフトウェアを更新するだけでなく、solid.ini ファイル内の構成パラメーターも更新してください。

5. S2 OS: `-x migratehsbg2` コマンド行スイッチを使用して、サーバー S2 (当初の 2 次サーバー) を起動します。

```
solid -x migratehsbg2
```

コマンド行スイッチには、2 つの効果があります。まず、サーバーに、既存のデータベースを受け入れて変換するよう指示します (-x autoconvert パラメーターと同じ効果)。また、新規 2 次サーバーが旧 1 次サーバーと古いレプリケーション・プロトコルで通信できるようにします。サーバーは SECONDARY ALONE 状態で起動する必要があります。

6. S1 OP: サーバー S1 (1 次サーバー) を検査し、まだ PRIMARY ALONE 状態であることを確認します。

ADMIN COMMAND 'hsb state';

7. S1 OP: hsb connect コマンドを実行して、1 次サーバーを 2 次サーバーに接続します。

ADMIN COMMAND 'hsb connect';

2 次サーバーが新しい方のバージョンのサーバーを実行している場合は、2 次サーバーから接続できないことに注意してください。

このステップにより、2 次サーバーがダウンしている間に発生したデータ変更により 2 次サーバーが「キャッチアップ」するプロセスが開始されます。

8. S1 OP: 続行する前に、「キャッチアップ」が完了するのを待ちます。キャッチアップが失敗した場合は、以下のことを行う必要があります。

- a. サーバー S2 (2 次サーバー) をシャットダウンします。
- b. S1 (1 次サーバー) から hsb copy を実行して、データベース全体をサーバー S2 にコピーします。

注: hsb netcopy でなく hsb copy を使用する必要があります。hsb netcopy は異なるサーバー・バージョン間では機能しないからです。

- c. 旧バージョンのサーバー (S2) でコピーをリカバリーします。
 - d. S2 (2 次サーバー) をシャットダウンします。
 - e. 前のステップに戻ります。
9. S1 OP: サーバー同士が接続され、キャッチアップされた後、以下を実行します。

ADMIN COMMAND 'shutdown force';

注意:

force オプションは、オープン・トランザクションをすべて打ち切ります。

サーバー S2 (2 次サーバー) がキャッチアップした後、それを新規 1 次サーバーにします。サーバー S1 (前の 1 次サーバー) を、アップグレードのためにシャットダウンします。

10. S2 OS: 新規 1 次サーバー S2 (旧 2 次サーバー) を PRIMARY ALONE 状態で作動するように設定します (まだ自動的にその状態に切り替わっていない場合)。サーバー S2 が PRIMARY ALONE 状態にあるかどうかを検証します。

ADMIN COMMAND 'hsb set primary alone';

ADMIN COMMAND 'hsb state';

11. S1 OP: サーバー S1 (当初の 1 次サーバー) をアップグレードします。

ソフトウェアを更新するだけでなく、solid.ini ファイル内の構成パラメーターも更新してください。

12. S1 OP: サーバー S1 上で、solidDB サーバーを OFFLINE 状態で再始動します。

```
solid -x backupserver
```

13. S2 OS: 新規 1 次サーバー S2 (旧 2 次サーバー) を検査し、まだ PRIMARY ALONE 状態であることを確認します。

```
ADMIN COMMAND 'hsb state';
```

14. S2 OS: データベースを新規 1 次サーバー (S2) から新規 2 次サーバー (S1) にネットコピーします。

```
ADMIN COMMAND 'hsb netcopy';
```

15. S2 OS: ネットコピーが成功したかどうかを検証します。

```
ADMIN COMMAND 'hsb status copy';
```

```
ADMIN COMMAND 'hsb connect';
```

hsb connect は新規 1 次サーバーを新規 2 次サーバーに接続し、新規 2 次サーバーがダウンしている間に発生したデータ変更、新規 2 次サーバーが「キャッチアップ」するプロセスが開始されます。

このステップが失敗した場合は、データベース全体を (hsb copy を使用して) 2 次サーバーにコピーしてから、ステップ 11 を再開します。

アップグレード後

新規 2 次サーバーが新規 1 次サーバーにキャッチアップした後、システムは完全に通常の状態に戻ります。新規 1 次サーバーと新規 2 次サーバーは両方ともアップグレードされ、最新の現行データを持ちます。いくつかテストの照会を行い、すべてが正しく作動していることを確認するとよいでしょう。

1. 1 次サーバーと 2 次サーバーが正しく機能していることをテストします。例えば、以下の一連の操作を選択してもかまいません。

1 次サーバー上で:

```
ADMIN COMMAND 'hsb state';  
ADMIN COMMAND 'hsb status catchup';
```

何かのタイプの読み取り専用照会を発行します。

2 次サーバー上で:

```
ADMIN COMMAND 'hsb state';  
ADMIN COMMAND 'hsb status catchup';
```

何かのタイプの読み取り専用照会を発行します。

2. 「ウォッチドッグ」プログラムを使用していた場合は、それを再開します。

注:

この方法は、ハードウェア、オペレーティング・システム、solidDB のどれをアップグレードする場合でも使用できます。

付録 A. HotStandby 構成パラメーター

このセクションでは、`solid.ini` 構成ファイル内のサーバー・サイド構成パラメーターについて説明します。サーバー・サイド・パラメーターは、`solid.ini` 構成ファイル内の `[Cluster]` セクションと `[HotStandby]` セクションのパラメーターからなっています。

さらに、このセクションでは、`solidhac.ini` 構成パラメーター内の高可用性コントローラー (HAC) 構成パラメーターと、`HAManager.ini` 構成パラメーター内の高可用性マネージャー構成パラメーターについても説明します。これらのパラメーターは、`solidDB HotStandby` に固有のものです。その他の `solid.ini` パラメーターについては、「*solidDB 管理者ガイド*」を参照してください。パラメーター名とセクション見出しのフォーマット設定方法については、「*solidDB 管理者ガイド*」を参照してください。これには、構成パラメーターに関する付録が付いています。その付録で、例えば、パラメーターの名前と値をフォーマット設定するときに従うべきルールが説明されています。`solid.ini` ファイルの `[HotStandby]` セクションは、それと同じルールに従っています。`solid.ini` ファイルの `[Watchdog]` セクションで使用できるパラメーターについては、`Watchdog` サンプルの項を参照してください。`[HotStandby]` セクション以外のセクションのパラメーターにも、`HotStandby` 機能に影響を及ぼすものがあることに注意してください。それらのパラメーターについては、47 ページの『`HotStandby` 固有のパラメーターの構成』と「*solidDB 管理者ガイド*」に説明があります。

1 次および 2 次パラメーター値の調整の確認

このセクションでは、1 次および 2 次サーバーでどのパラメーターを同一にする必要があるか、およびどのパラメーターを異ならせる必要があるかについて説明します。

1 次側および 2 次側の両方で特定のパラメーターを同一にする必要があります。この理由は、フェイルオーバーの後で、元の 2 次側が新規の 1 次側になり、それが古い 1 次側と同じ動作をしなければならないからです。注意すべき点は、同じ値を使用することが絶対要件ではないことです。異なる値が使用されたときに、サーバーで障害が起きるのではなく、クライアントで認識する動作が異なることがあります。

`[HotStandby]` セクションにないが、間接的に関係している一部のパラメーターも、1 次サーバーおよび 2 次サーバーの両方で同一にする必要があります。例えば、**DurabilityLevel** パラメーターは一般に、1 次側および 2 次側で同じでなければなりません。

1 次サーバーおよび 2 次サーバーで特定のパラメーターを異ならせる必要があります。この理由は、両サーバーを一意的に識別できるように、かつ相互に対話できるようにするためです。

以下の `HotStandby` パラメーターが 1 次側および 2 次側の両方で同じでなければなりません。

- [HotStandby]
 - **2SafeAckPolicy**
 - **AutoPrimaryAlone**
 - **ConnectTimeout**
 - **HSBEnabled**
 - **PrimaryAlone** (推奨されないが、使用した場合は同じにする必要がある)
- [IndexFile]
 - **FileSpec** は「互換性」が必要です。つまり、**FileSpec** パラメーターの数を同じにする必要と、対応する **FileSpec** パラメーターのサイズを一致させる必要があります。
 - **BlockSize**
- [Logging]
 - **BlockSize**

以下のパラメーターが異ならなければなりません。

- [HotStandby]
 - **Connect**

以下のパラメーターは、コンピューター上のディスク・ドライブ構成などの環境に基づいて、同じであるか異なる可能性があります。

- [General]
 - **BackupDirectory**
- [HotStandby]
 - **CopyDirectory**

HSB パフォーマンスに影響する、「非 HSB」パラメーターの設定もいくつかあります。例えば、solid.ini ファイルの [Logging] セクションにある **DurabilityLevel** パラメーターには、HotStandby でパフォーマンスを最適化できる設定があります。16 ページの『アダプティブ持続性』と、「solidDB 管理者ガイド」の **DurabilityLevel** の説明を参照してください。

1 次側の設定が 2 次側の設定より優先されるかどうかの決定

パラメーターによっては、1 次サーバーおよび 2 次サーバーの両方で同じである必要があります。同じ値を設定しないと、それぞれのサーバーが、そのサーバーの solid.ini ファイルに定義されている値を使用するようになることが予想されます。ただし、これは必ずしも当てはまりません。

2SafeAckPolicy のような、2 次側の動作を制御する一部のパラメーターの場合であっても、1 次側の値によって、その動作が決まります。原則的に、安全性および持続性のためのすべてのパラメーターが 1 次側で制御されます。例えば、1 次側は **2SafeAckPolicy** の値を読み取って、その値を 2 次側に使用するように送信します。2 次側の solid.ini ファイルに格納されている値は、2 次側が 1 次側になった場合のみ使用されます。

1 次側の値が優先されるパラメーターには以下のパラメーターが含まれます。

- **HotStandby.SafenessLevel**
- **HotStandby.2SafeAckPolicy**
- **Logging.DurabilityLevel**
- **HotStandby.NetcopyRpcTimeout**

コマンド 'hsb connect' の実行時点で、1 次側にある以下のパラメーターが優先されます。

- **HotStandby.PingTimeout**
- **HotStandby.PingInterval**

HotStandby 構成パラメーターの照会

標準パラメーター操作コマンドを使用して、HotStandby パラメーターの値およびプロパティを照会できます。

コマンドは以下のとおりです。

```
ADMIN COMMAND '[describe] parameter[section_name[.parameter_name]]';
```

以下に例を示します。

```
ADMIN COMMAND 'parameter logging.durabilitylevel';
RC TEXT
-- ----
0 Logging DurabilityLevel 3 3 2
```

```
ADMIN COMMAND 'parameter hotstandby.MaxLogSize';
RC TEXT
-- ----
0 HotStandby MaxLogSize 10000000 0 0
```

結果行に示されている 3 つの値は、左から以下のとおりです。

- **現行値** - 動的に設定されるか、デフォルト値またはファクトリー値から継承されます。
- **デフォルト値** - solid.ini ファイルから最初に読み取られるか、ファクトリー値から継承されます。
- **ファクトリー値** - 製品に事前設定されています。

HotStandby 構成パラメーターの変更

通常、パラメーターの値は、solid.ini 構成ファイルの値を変更し、サーバーを再始動することによって変更します。ただし、ほとんどの HotStandby パラメーターが ADMIN COMMAND でも変更可能です。

このタスクについて

コマンドは以下のとおりです。

```
ADMIN COMMAND 'parameter section_name.parameter_name=value [temporary]';
```

注: 推奨されないコマンド ADMIN COMMAND "hotstandby parameter ..." もあり、これが HotStandby パラメーターの変更で使用されることがあります。その構文は以下のとおりです。

```
ADMIN COMMAND 'hotstandby parameter parameter_name value';
```

アクセス・モード

パラメーターのアクセス・モードは、そのパラメーターの動的変更が可能であるかどうか、およびその変更がいつ有効になるかを指示します。

パラメーターの値が ADMIN コマンドで変更されたとき、その変更内容はすぐに適用される場合とそうでない場合があります、また次のサーバー始動時に適用される場合とそうでない場合があります。パラメーター値が solid.ini ファイルに書き込まれたとき、それは次のサーバー始動時に有効になります。

アクセス・モード値

可能なアクセス・モードは以下のとおりです。

- **RO** (読み取り専用): 値の変更は不可です。現行値は必ず始動時の値と同一です。
- **RW**: ADMIN COMMAND で変更可能です。変更内容はすぐに有効になります。
- **RW/Startup**: ADMIN COMMAND で変更可能です。変更内容は次のサーバー始動時に有効になります。
- **RW/Create**: ADMIN COMMAND で変更可能です。変更内容は新規データベースの作成時に有効になります。

パラメーター変更の保存

temporary オプションが使用されていないければ、パラメーターに対するすべての変更内容が次のチェックポイントで solid.ini ファイルに保存されます。保存は以下のコマンドで迅速に適用する場合があります。

```
ADMIN COMMAND 'save parameters [file_name]';
```

デフォルトでは、このコマンドはデフォルトの solid.ini ファイルを書き換えます。**file_name** オプションを使用することにより、異なる場所に出力を送信できます。

Cluster クラスタ

以下の構成パラメーターは、solid.ini 構成ファイルの [Cluster] セクションで設定できます。

表 18. Cluster パラメーター

パラメーター名	説明	ファクトリー値	アクセス・モード
ReadMostlyLoadPercentAtPrimary	1 次側に送信された読み取りロードのパーセント。	50	RW/Startup

HotStandby セクション

以下の構成パラメーターは、solid.ini 構成ファイルの [HotStandby] セクションで設定できます。

表 19. HotStandby パラメーター

パラメーター名	説明	ファクトリー値	アクセス・モード
1SafeMaxDelay	1-Safe レプリケーションでは、コミットされたトランザクションが 2 次側に送信されるまでの最大遅延 (ミリ秒)。	1000	RW
2SafeAckPolicy	<p>これは、2 次側が 1 次側からトランザクションを受信したときの 2 次側の確認応答タイミングを指定します。</p> <p>有効値は以下のとおりです。</p> <ul style="list-style-type: none"> • 1 = 2-safe received。2 次サーバーはデータを受信すると応答します。 • 2 = 2-safe visible。2 次サーバーはデータが「可視」であると、つまり 2 次側でトランザクションを実行したときに応答します。 • 3 = 2-safe durable。2 次サーバーはデータを永続的なものにしたとき、つまりデータをコミットしてそのデータをディスクに書き込んだときに応答します。 <p>当然、2-safe durable が最も安全な手法で、応答時間が最も速いのは 2-safe received です。ただし、実際には、ほとんどの場合に、2-safe received モードで、データの安全のための十分な保証が提供されるので、安全性と速度の間の最善の妥協が得られます。</p> <p>このパラメーターは、サーバーが 2-safe レプリケーションを使用している場合にのみ適用されます。 注: このパラメーターは 2 次サーバーの動作を制御しますが、設定は 1 次側で行います。2 次側の solid.ini ファイルの値は無視されます。</p> <p>118 ページの『1 次側の設定が 2 次側の設定より優先されるかどうかの決定』および 117 ページの『1 次および 2 次パラメーター値の調整の確認』を参照してください。</p>	1	RW
AutoPrimaryAlone	このパラメーターが Yes に設定されている場合、2 次側との接続が切れたときに、サーバーは自動的に PRIMARY ALONE 状態 (PRIMARY UNCERTAIN 状態ではなく) にされます。これを「yes」に設定する予定の場合は、38 ページの『ネットワーク分割と二重 1 次サーバー』にある最重要の警告を参照してください。	No	RW
CatchupSpeedRate	<p>サーバーはキャッチアップを実行しながら、引き続きクライアントからのデータベース要求の処理も行います。CatchupSpeedRate パラメーターは、アプリケーション要求への応答により重点を置き、キャッチアップの優先順位をより低くするため、またはその逆にするために使用できます。</p> <p>速度レートは、リンクおよび 2 次側スループットに従う最大利用可能速度のパーセントとして表されます。大きい数値ほど、キャッチアップにより重点を置き、クライアント要求の処理にあまり重点を置かないことを意味します。許可される値は 1 から 99 までです。</p>	70	RW

表 19. HotStandby パラメーター (続き)

パラメーター名	説明	ファクトリー値	アクセス・モード
Connect	<p>Connect パラメーターは、ペアのもう一方の HotStandby サーバーの「アドレス」を示します。</p> <p>HotStandby セクションの Connect スtringの形式は、[Com] セクションの Listen パラメーターの形式と同じです (詳細は、「IBM solidDB 管理者ガイド」を参照してください)。</p> <p>このパラメーターを HotStandby 用のサーバーで省略した場合は、ADMIN COMMAND を使用して、このパラメーターを動的に設定できます。サーバーが Connect Stringを持つまで、サーバーに可能な状態は、接続に関係しない状態、つまり PRIMARY ALONE、SECONDARY ALONE、および STANDALONE のみです。</p> <p>Connect パラメーターは、HSBEnabled パラメーターが「yes」に設定されていなければ無視されます。</p>	ファクトリー値なし。	RW
ConnectTimeout	<p>接続タイムアウト値を指定することにより、HotStandby 接続操作でリモート・マシンに接続するまでの最大待ち時間を、秒単位で設定できます。</p> <p>ConnectTimeout パラメーター (特定のプラットフォーム上でのみ役立つ) は特定の管理コマンドでのみ使用されます。それらのコマンドは、以下のとおりです。</p> <ul style="list-style-type: none"> • hotstandby connect • hotstandby switch primary • hotstandby switch secondary <p>例えば、タイムアウトを 30 秒 (30000 ミリ秒) に設定するには、次のようにします。</p> <pre>[HotStandby] ConnectTimeout=30000</pre> <p>PingTimeout も参照してください。</p>	<p>0 (タイムアウトなし)</p> <p>単位: 1 ms</p>	RW

表 19. HotStandby パラメーター (続き)

パラメーター名	説明	ファクトリー値	アクセス・モード
CopyDirectory	<p>[HotStandby] セクションの CopyDirectory パラメーターは、ユーザーが以下のコマンドを実行したときに実行される HotStandby コピー操作の名前および場所を定義します。</p> <pre>ADMIN COMMAND 'hotstandby copy';</pre> <p>例えば、パラメーターを次のようにすることがあります。</p> <pre>[HotStandby] CopyDirectory=C:%solidDB%secondary%dbfiles</pre> <p>CopyDirectory パラメーターに相対パスを指定する場合は、1 次サーバーの solid.ini ファイルを保持するディレクトリーからの相対パスにします。</p> <p>このパラメーターにファクトリー値はないため、ディレクトリーが solid.ini ファイルに指定されていないときは、copy コマンドで指定する必要があります。</p> <p>より柔軟な解決策として ADMIN COMMAND 'hotstandby netcopy' をデータベースのコピーに使用する方法が推奨される点に注意してください。</p>	ファクトリー値なし。	RW
HSBEnabled	<p>このパラメーターを yes に設定した場合、サーバーは 1 次または 2 次の HotStandby サーバーとして動作できます。このパラメーターを no に設定した場合、サーバーは HotStandby サーバーとして動作できません。</p> <p>このパラメーターを Yes に設定することは、サーバーの最初の始動時にサーバーのデフォルトの初期状態を SECONDARY ALONE に明示的に定義することになります。有効値は「yes」および「no」です。</p> <p>HotStandby を使用するには、Connect パラメーターも指定する必要があります。それには、このパラメーターを solid.ini ファイルで設定するか、または ADMIN COMMAND を使用してそれを設定します。</p>	no	RO
MaxLogSize	ディスク・ベース HSB ログの最大サイズ。ファクトリー値は無制限です。	0 単位: 1 バイト k=KB m=MB	
MaxMemLogSize	ファイル・ベース・ロギングが使用不可 (Logging.LogEnabled=No) の場合は、2 次側に送信される前のトランザクションを保持するインメモリー・ログのサイズ。値は、インメモリー・ログがフルになるまでに、サーバーが PRIMARY ALONE 状態を継続できる時間に影響します。	8M 単位: 1 バイト k=KB m=MB	RO
NetcopyRpcTimeout	ネットコピー操作のデータ伝送確認応答タイムアウト (ミリ秒)	30000 単位: 1 ms	RW

表 19. HotStandby パラメーター (続き)

パラメーター名	説明	ファクトリー値	アクセス・モード
PingInterval	<p>1 次側および2 次側は、まだ接続済みであることを確認するために、定期的な時間間隔で相互に「ping」メッセージを送信します。(これらの ping は 1 次側から 2 次側に送信されるトランザクション情報とは無関係です。)</p> <p>値は サーバーが送信する 2 つの連続 ping 間の間隔 (ミリ秒) に等しくなります。</p>	<p>1000 (1 秒)</p> <p>単位: 1 ms</p>	RW
PingTimeout	<p>このパラメーターは、サーバーの待ち時間の長さを指定し、この時間が経過すると、もう一方のサーバーがダウンしたか、アクセス不能であると結論付けます。</p> <p>指定された時間 (ミリ秒) が過ぎると、サーバーは接続が切れたと結論付けて、それに応じて状態を変更します。</p> <p>49 ページの『PingTimeout および PingInterval パラメーター [HotStandby]』を参照してください。</p> <p>ConnectTimeout も参照してください。</p>	<p>4000 (4 秒)</p> <p>単位: 1 ms</p>	RW
PrimaryAlone	<p>このパラメーターは推奨されません。</p> <p>AutoPrimaryAlone パラメーターを使用してください。</p>	No	RW
SafenessLevel	<p>このパラメーターは、レプリケーション・プロトコルの安全性レベルを設定します。</p> <p>「auto」値を使用することにより、持続性レベルに関連して安全性レベルを動的に変更することを許可できます。SafenessLevel を「auto」に設定し、SET DURABILITY コマンドまたは DurabilityLevel パラメーターを使用することにより、持続性をリラックスに設定した場合には、安全性レベルは 1-safe に、持続性レベルをストリクトに設定した場合には、安全性レベルは 2-safe に設定されます。ただし、DurabilityLevel を 2 (アダプティブ持続性) に設定した場合は、「auto」設定は効力を持たず、安全性レベルは常に 2-safe になります。</p>	<p>可能な値:</p> <p>1safe、2safe、および auto</p>	RW

高可用性コントローラーの構成パラメーター

以下の表は、solidhac.ini 構成ファイルにある、高可用性コントローラーの構成パラメーターの説明です。この構成ファイルは、異なる 3 つのセクションに分かれています。以下の各表は、これらのセクション分けに従っています。パラメーターは構成ファイルにあるとおりの順序で並んでいます。

[HAController] セクション

表 20. HAC 構成パラメーター: [HAController] セクション

パラメーター名	説明	ファクトリー値
Listen	HAC listen ポートのプロトコルおよびポート。このポートは、HAC にコマンドを送信するために高可用性マネージャーによって使用されます。	tcp 3135
StartInAutomaticMode	<p>このパラメーターは、HAC が自動モードで実行を開始するかどうかを定義します。これは動的に変更可能なパラメーターです。</p> <p>AUTOMATIC モードの場合、HAC は、サーバーの状態をモニターし、必要な場合にデータベースに関連したアクションを実行することにより、データベースの可用性を自動的に維持しようと試みます。対照的に、ADMINISTRATIVE モードでは、HAC はシステム・コンポーネントの状態をモニターするのみです。</p> <p>PreferredPrimary パラメーターも参照してください。</p> <p>可能な値は、「Yes」および「No」です。</p>	Yes
EnableDBProcessControl	<p>このパラメーターは、データベースを自動的に開始し、さらにデータベース・プロセスのシャットダウンと再開のためのコマンドをユーザーに提供することにより、ローカル・データベース・プロセスを HAC で管理できるようにします。</p> <p>この値は、HAC が ACTIVE モードの場合にのみ有効です。</p> <p>可能な値は、「Yes」および「No」です。</p> <p>これはオプション・パラメーターです。</p>	No

表 20. HAC 構成パラメーター: [HAController] セクション (続き)

パラメーター名	説明	ファクトリー値
EnableAutoNetcopy	<p>このパラメーターは、必要な場合に HAC がネットコピーを開始できるかどうかを定義します。</p> <p>可能な値は、「Yes」および「No」です。</p> <p>これはオプション・パラメーターです。</p>	No
RequiredConnectFailures	<p>このパラメーターは、接続試行が何回連続して失敗した後に、サーバーで障害が起きたと見なすかその回数を定義します。</p> <p>可能な値は 1 から無制限の数値です。</p> <p>これはオプション・パラメーターです。</p>	1
CheckTimeout	<p>このパラメーターは、CHECK モードの SQL コマンドに対してミリ秒単位のタイムアウトを定義します。</p> <p>可能な値は 1 ミリ秒から無制限のミリ秒単位です。</p>	50
CheckInterval	<p>このパラメーターは、連続するサーバー状態検査の間隔をミリ秒単位で定義します</p> <p>可能な値は 1 ミリ秒から無制限のミリ秒単位です。</p> <p>これはオプション・パラメーターです。</p>	1000
Username	<p>HAC のユーザー名。</p> <p>これは必須パラメーターです。</p>	
Password	<p>HAC のパスワード。</p> <p>これは必須パラメーターです。</p>	
DBUsername	<p>HotStandby サーバーのユーザー名。</p> <p>これは必須パラメーターです。</p>	

表 20. HAC 構成パラメーター: [HAController] セクション (続き)

パラメーター名	説明	ファクトリー値
DBPassword	HotStandby サーバーのパスワード。 これは必須パラメーターです。	

[LocalDB] セクション

表 21. HAC 構成パラメーター: [LocalDB] セクション

パラメーター名	説明	ファクトリー値
Connect	このパラメーターは、HAC がローカル・データベースに接続できるようにします。ローカル・データベースはネットワークとして定義されます。ネットワーク名は、通信プロトコルおよびサーバー名で構成されています。 これは必須パラメーターです。	tcp 2125
StartScript	このパラメーターは、データベース・プロセスの開始に使用されるスクリプトの名前を宣言します。このパラメーターはオプションですが、データベース・プロセスを HAC で制御する場合は設定することをお勧めします。HAC は、ACTIVE モードで、かつ EnableDBProcessControl パラメーターが「Yes」に設定されていれば、この指定されたスクリプトを自動的に使用します。 これはオプション・パラメーターです。	/home/solid/start_solid.sh

表 21. HAC 構成パラメーター: [LocalDB] セクション (続き)

パラメーター名	説明	ファクトリー値
PreferredPrimary	<p>このパラメーターは、両方のサーバーの logpos 値が等しい場合にローカル・サーバーが 1 次側になるかどうかを定義します。両方のサーバーの値が「No」であるか、または値がない場合には、最初のサーバーが 1 次側になります。</p> <p>この値は、StartInActiveMode パラメーターの値が「Yes」の場合にのみ有効です。</p> <p>可能な値は、「Yes」および「No」です。</p> <p>これはオプション・パラメーターです。</p>	No

[RemoteDB] セクション

表 22. HAC 構成パラメーター: [RemoteDB] セクション

パラメーター名	説明	ファクトリー値
Connect	<p>このパラメーターは、HAC がリモート・データベースに接続できるようにします。ローカル・データベースは、通信プロトコル、リモート・サーバー IP アドレス、およびそのポートを指定することによって定義されます。</p> <p>これは必須パラメーターです。</p>	tcp 192.168.3.123 2125

[ERE] セクション

表 23. HAC 構成パラメーター: [ERE] セクション

パラメーター名	説明	ファクトリー値
EREIP	<p>このパラメーターは、外部参照エンティティの IP アドレスを識別します。</p> <p>RequiredPingFailures パラメーターも参照してください。</p> <p>これはオプション・パラメーターです。</p>	EREIP=192.168.3.1
RequiredPingFailures	<p>このパラメーターは、Ping 呼び出しが何回か連続して失敗したため、サーバーは ERE から切断され、クライアント・ネットワークから分離されていると HAC が結論付ける Ping 呼び出しの連続失敗の最大数を定義します。</p> <p>このパラメーターは ERE でのみ使用可能です。</p> <p>可能な値は 1 から無制限の数値です。</p> <p>これはオプション・パラメーターです。</p>	3

高可用性マネージャーの構成パラメーター

HAManager.ini 構成ファイルにある、高可用性マネージャーの構成パラメーター。

表 24. 高可用性マネージャーの構成パラメーター

パラメーター名	説明
Header_text	<p>このパラメーターは、HA マネージャーのヘッダー・テキストを定義します。この値は、ユーザー・インターフェースに表示されません。</p> <p>これは必須パラメーターです。</p>
Server1_host Server2_host	<p>これらの 2 つのパラメーターは、HAC インスタンスのホスト名を定義します。</p>

表 24. 高可用性マネージャーの構成パラメーター (続き)

パラメーター名	説明
Server1_name Server2_name	これらの 2 つのパラメーターは、HAC インスタンスの名前を定義します。
Server1_pass Server2_pass	これらの 2 つのパラメーターは、HAC インスタンスのパスワードを定義します。
Server1_port Server2_port	これらの 2 つのパラメーターは、HAC インスタンスのポートを定義します。
Server1_user Server2_user	これらの 2 つのパラメーターは、HAC インスタンスのユーザー名を定義します。
Window_title	このパラメーターは、HA マネージャーのウィンドウ・タイトルを定義します。この値は、ユーザー・インターフェースに表示されます。 これは必須パラメーターです。

付録 B. エラー・コード

このセクションでは、HotStandby に関連するエラー・コードについて説明します。他のサーバー・エラー・コードの大部分は、「*solidDB 管理者ガイド*」で説明されています。

このセクションで説明するエラーには、ADMIN COMMAND 結果セットの RC 列の値と、ODBC または JDBC ドライバーのエラー・コードとして返されるエラーが含まれています。例えば、すべての「*solidDB HotStandby エラー*」と「*solidDB 高可用性コントローラー・エラー*」は ADMIN COMMAND 結果セットの値であり、すべての「*solidDB 通信エラー*」は、ドライバーが返すものです。

このセクションに記載されている表のエラー・カテゴリーは、以下のとおりです。

HotStandby のエラーおよび状況コード

solidDB HotStandby エラー (14009 - 147xx, 307xx) は、特定の HotStandby コマンドを使用したときに発生します。それらのコマンドは、solidDB SQL 拡張機能です。

表 25. HotStandby のエラーおよび状況コード

エラーまたは状況コード	説明
14003	ACTIVE コマンドの結果セット内にこの状況を返すことがある ADMIN COMMAND: <ul style="list-style-type: none">ADMIN COMMAND 'hotstandby status switch'ADMIN COMMAND 'hotstandby status catchup'ADMIN COMMAND 'hotstandby status copy' 意味: 切り替えプロセス、キャッチアップ・プロセス、コピーまたはネットコピー・プロセスが、まだアクティブです。
14007	CONNECTING コマンドの結果セット内にこの状況を返すことがある ADMIN COMMAND: <ul style="list-style-type: none">ADMIN COMMAND 'hotstandby status connect' 意味: 1 次サーバーおよび 2 次サーバーは接続プロセスを実行中です。

表 25. HotStandby のエラーおよび状況コード (続き)

エラーまたは状況コード	説明
14008	<p>CATCHUP</p> <p>コマンドの結果セット内にこの状況を返すことがある ADMIN COMMAND:</p> <ul style="list-style-type: none"> • ADMIN COMMAND 'hotstandby status connect' <p>意味: 1 次サーバーが 2 次サーバーに接続しましたが、トランザクション・ログはまだ完全にはコピーされていません。このメッセージを返すのは、1 次サーバーだけです。</p>
14009	<p>以前にサーバー切り替えは発生しませんでした。</p> <p>コマンドの結果セット内にこの状況を返すことがある ADMIN COMMAND:</p> <ul style="list-style-type: none"> • ADMIN COMMAND 'hotstandby status switch' <p>意味: サーバー間に切り替えプロセスが発生したことはありません。</p>
14501	<p>操作が失敗しました。</p> <p>意味: 操作が失敗し、サーバーはシャットダウンしようとしています。失敗の原因としては、非 HotStandby サーバーにコマンドを発行したか、コマンドを適用できない 1 次サーバーまたは 2 次サーバーにコマンドを発行したことが考えられます。</p> <p>コマンドの結果セット内にこの状況を返すことがある ADMIN COMMAND:</p> <ul style="list-style-type: none"> • ADMIN COMMAND 'hotstandby switch primary' • ADMIN COMMAND 'hotstandby switch secondary' • ADMIN COMMAND 'hotstandby cominfo' • ADMIN COMMAND 'hotstandby connect' • ADMIN COMMAND 'hotstandby status switch' • ADMIN COMMAND 'hotstandby set standalone' • ADMIN COMMAND 'hotstandby copy' • ADMIN COMMAND 'hotstandby netcopy'
14502	<p>RPC パラメーターが無効です。</p> <p>意味: HSB 接続ストリング内で提供された接続情報の一部にエラーがあり、別のサーバーへの接続が失敗しました。</p> <p>コマンドの結果セット内にこの状況を返すことがある ADMIN COMMAND:</p> <ul style="list-style-type: none"> • ADMIN COMMAND 'hotstandby status connect'

表 25. HotStandby のエラーおよび状況コード (続き)

エラーまたは状況コード	説明
14503	<p>通信エラー、接続の切断。</p> <p>意味: 通信エラーがあったため、もう一方のサーバーを検出できませんでした。もう一方のサーバーへの接続に失敗しました。</p> <p>コマンドの結果セット内にこの状況を返すことがある ADMIN COMMAND:</p> <ul style="list-style-type: none"> • ADMIN COMMAND 'hotstandby status connect'
14520	<p>サーバーは HotStandby 2 次サーバーであり、接続は許可されません。</p>
14522	<p>HotStandby コピー・ディレクトリーが指定されませんでした。</p> <p>意味: コピー・ディレクトリーが指定されていません。</p> <p>コマンドの結果セット内にこの状況を返すことがある ADMIN COMMAND:</p> <ul style="list-style-type: none"> • ADMIN COMMAND 'hotstandby copy' <p>この問題を解決するには、ディレクトリーをコマンドの一部として指定します。以下に例を示します。</p> <p>ADMIN COMMAND 'hotstandby copy ¥Secondary¥dbfiles¥'</p> <p>あるいは、solid.ini 構成ファイル内で CopyDirectory パラメーターを設定します。</p>
14523	<p>切り替えプロセスは、既にアクティブです。</p> <p>意味: 切り替えプロセスは、HotStandby サーバー内で既にアクティブです。現行の切り替えを完了する必要があるだけなら、待ってください。2 回目の切り替え (つまり、元の構成に戻すこと) を実行しようとしている場合は、最初の切り替えが完了するまで待ってから、2 回目の切り替えを開始できます。</p> <p>コマンドの結果セット内にこの状況を返すことがある ADMIN COMMAND:</p> <ul style="list-style-type: none"> • ADMIN COMMAND 'hotstandby switch primary' • ADMIN COMMAND 'hotstandby switch secondary' • ADMIN COMMAND 'hotstandby status switch'

表 25. HotStandby のエラーおよび状況コード (続き)

エラーまたは状況コード	説明
14524	<p>HotStandby データベースの基本データベースの 1 つが異なっており、データベースのタイム・スタンプが互いに異なっています。</p> <p>意味: データベースが、異なるシード・データベースに由来しています。データベースの同期をとる必要があります。1 次サーバーのデータベースを 2 次サーバーにネットコピーしなければならない可能性があります。</p> <p>コマンドの結果セット内にこの状況を返すことがある ADMIN COMMAND:</p> <ul style="list-style-type: none"> • ADMIN COMMAND 'hotstandby connect' • ADMIN COMMAND 'hotstandby status switch'
14525	<p>HotStandby のデータベース同士が正しく同期していません。</p> <p>意味: データベース同士が正しく同期していません。データベースの同期をとる必要があります。データベース・サーバーの 1 つ (2 次サーバーにしようとしている方) を、コマンド行パラメーター -x backupserver を指定して始動し、1 次サーバーのデータベースを 2 次サーバーにネットコピーする必要があります。</p> <p>コマンドの結果セット内にこの状況を返すことがある ADMIN COMMAND:</p> <ul style="list-style-type: none"> • ADMIN COMMAND 'hotstandby connect' • ADMIN COMMAND 'hotstandby status switch'
14526	<p>引数が無効です。</p> <p>意味: HotStandby ADMIN COMMAND で使用した引数が、不明または無効です。</p> <p>すべての HotStandby コマンドが、ADMIN COMMAND の結果セット内にこのエラーを返す可能性があります。</p> <p>注: 以下の HotStandby コマンドでは、指定された 1 次サーバーまたは 2 次サーバーを切り替えに適用できない場合、この引数無効エラーは構文エラーです。</p> <ul style="list-style-type: none"> • ADMIN COMMAND 'hotstandby switch primary' • ADMIN COMMAND 'hotstandby switch secondary'

表 25. HotStandby のエラーおよび状況コード (続き)

エラーまたは状況コード	説明
14527	<p>これは HotStandby サーバーではありません。</p> <p>意味: HotStandby 用に構成されていないサーバーに対してコマンドが実行されました。</p> <p>コマンドの結果セット内にこの状況を返すことがある ADMIN COMMAND:</p> <ul style="list-style-type: none"> • ADMIN COMMAND 'hotstandby connect' • ADMIN COMMAND 'hotstandby status switch' • ADMIN COMMAND 'hotstandby switch primary' • ADMIN COMMAND 'hotstandby switch secondary' • ADMIN COMMAND 'hotstandby state'
14528	<p>両方の HotStandby データベースが 1 次データベースです。</p> <p>意味: 両方のデータベースが 1 次です。これは、変更内容が競合する可能性があるため致命的エラーです。両方のデータベースは、システムによって自動的に 2 次状態へ落とされます。どのデータベースが真の 1 次データベースであるかを決定し、データベース同士の同期をとる必要があります。</p> <p>コマンドの結果セット内にこの状況を返すことがある ADMIN COMMAND:</p> <ul style="list-style-type: none"> • ADMIN COMMAND 'hotstandby connect' • ADMIN COMMAND 'hotstandby status switch'
14535	<p>サーバーは、既に 1 次サーバーです。</p> <p>意味: 1 次サーバーに切り替えようとしているサーバーは、既にいずれかの PRIMARY 状態にあります。</p> <p>コマンドの結果セット内にこの状況を返すことがある ADMIN COMMAND:</p> <ul style="list-style-type: none"> • ADMIN COMMAND 'hotstandby switch primary'
14536	<p>サーバーは、既に 2 次サーバーです。</p> <p>意味: 2 次サーバーに切り替えようとしているサーバーは、既にいずれかの SECONDARY 状態にあります。</p> <p>コマンドの結果セット内にこの状況を返すことがある ADMIN COMMAND:</p> <ul style="list-style-type: none"> • ADMIN COMMAND 'hotstandby switch secondary'

表 25. HotStandby のエラーおよび状況コード (続き)

エラーまたは状況コード	説明
14537	<p>HotStandby 接続が切断されています。</p> <p>意味: このコマンドは、1 次と 2 次の両方のサーバーから返されます。</p> <p>コマンドの結果セット内にこの状況を返すことがある ADMIN COMMAND:</p> <ul style="list-style-type: none"> • ADMIN COMMAND 'hotstandby status connect' <p>この問題の原因として考えられることの 1 つは、2 次サーバーの solid.ini ファイル内で Connect ストリングが正しくないことです。ネットコピー操作が成功するのに、接続コマンドが失敗する場合は、Connect ストリングを調べてください。(ネットコピーでは、2 次サーバーが 1 次サーバーへの別個の接続を開く必要がないので、2 次サーバー上の Connect ストリングに誤りがあっても、成功する場合があります。)</p>
14538	<p>サーバーが HotStandby 1 次サーバーではありません。</p> <p>意味: このコマンドを発行するには、サーバーが HotStandby 1 次サーバーである必要があります。</p> <p>コマンドの結果セット内にこの状況を返すことがある ADMIN COMMAND:</p> <ul style="list-style-type: none"> • ADMIN COMMAND 'hotstandby copy copy_directory' • ADMIN COMMAND 'hotstandby netcopy' • ADMIN COMMAND 'hotstandby connect' • ADMIN COMMAND 'hotstandby set primary alone' • ADMIN COMMAND 'hotstandby set standalone'

表 25. HotStandby のエラーおよび状況コード (続き)

エラーまたは状況コード	説明
14539	<p>操作が拒否されました。</p> <p>このエラー・コードは、以下のいずれかの状態が発生したときに発生します。</p> <ul style="list-style-type: none"> ユーザーが 1 次サーバーに対して netcopy コマンドを発行しましたが、2 次サーバーであるはずのサーバーが実際には 2 次サーバー状態でないか、「ネットコピー listen モード」になっていません。(1 次サーバーと「2 次」サーバーの両方が PRIMARY ALONE 状態の可能性があります。) <p>この問題を解決するには、-x backupserver コマンド行オプションを指定して「2 次サーバー」を再始動し、再度 1 次サーバーに対して netcopy コマンドを発行してみます。</p> <p>重要: 両方のサーバーが PRIMARY ALONE 状態だった場合、しかも、両方のサーバーが PRIMARY ALONE 状態だったときにトランザクションを実行していた場合は、それぞれに相手側が持っていないデータが存在する可能性があります。これは深刻なエラーであり、netcopy を実行して両方を同期状態に戻すと、既に「2 次」サーバーにコミットされているトランザクションが上書きされる結果になります。</p> <ul style="list-style-type: none"> このメッセージは、コールバック関数を使用し、コールバック関数がシャットダウンを拒否するか、バックアップ・コマンドまたはネットコピー・コマンドの受け入れを拒否したときに生成される場合があります。 <p>リンク・ライブラリー・アクセスを使用している場合は、SSCSetNotifier 関数を使用して「コールバック」関数を指定できます。コールバック関数は、サーバーがシャットダウンまたはネットコピー操作の実行を命令したときに、通知を受け取ります。何らかの理由でアプリケーションがコマンドに従わない場合、コールバックはコマンドをキャンセルする値を返すことがあります。その状態では、エラー 14539 が表示されます。</p> <p>問題を解決するには、クライアント・コードが中断を望まない操作を終了するまで待つから、コマンド (例えば、シャットダウンまたはネットコピー) を再試行します。</p>
14540	サーバーは、既に HotStandby サーバーではありません。
14541	solid.ini 内の HotStandby 構成が ADMIN COMMAND 'HSB SET STANDALONE' と競合しています。
14542	サーバーが backupserver モードです。操作が拒否されました。
14543	コマンドが無効です。データベースは HotStandby データベースですが、solid.ini 構成ファイル内に HotStandby セクションが見つかりませんでした。

表 25. HotStandby のエラーおよび状況コード (続き)

エラーまたは状況コード	説明
14544	操作が失敗しました。このコマンドは、ディスクレス・サーバーではサポートされていません。
14545	1 次サーバーは、そのロールが <code>primary broken</code> の場合、 <code>PRIMARY ALONE</code> にのみ設定できます。
14546	切り替えが失敗しました。サーバーまたはリモート・サーバーを <code>PRIMARY ALONE</code> から 2 次サーバーに切り替えることができません。切り替えの前に、まずキャッチアップを実行してください。 意味: このコマンドは、 <code>PRIMARY ALONE</code> 状態のローカルまたはリモート 1 次サーバーから <code>SECONDARY</code> への状態切り替えが実行され、1 次サーバーと 2 次サーバーが同期していないことが検出された場合に返されます。1 次サーバーを 2 次サーバーに接続し、キャッチアップ・プロセスが完了するのを待ってから、2 次サーバーを 1 次サーバーに切り替える必要があります。 このエラーを返す HotStandby コマンド: <ul style="list-style-type: none"> • <code>ADMIN COMMAND 'hotstandby switch secondary'</code>
14547	-R オプション (読み取りタイムアウト) の値がなかったか、無効でした。
14548	切り替えが失敗しました。STANDALONE のサーバーを <code>SECONDARY</code> に切り替えることができませんでした。 意味: このコマンドは、 <code>STANDALONE</code> 状態のローカルまたはリモート 1 次サーバーから <code>SECONDARY</code> への状態切り替えが実行され、1 次サーバーと 2 次サーバーが同期していないことが検出された場合に返されます。1 次サーバーを 2 次サーバーに接続し、キャッチアップが完了するのを待ってから、2 次サーバーを 1 次サーバーに切り替える必要があります。 このエラーを返す HotStandby コマンド: <ul style="list-style-type: none"> • <code>ADMIN COMMAND 'hotstandby switch secondary'</code>
14549	HotStandby トランザクションがアクティブです。 意味: HotStandby 接続が切断されている場合、1 次サーバーはシャットダウンの前に、 <code>ALONE</code> モードに設定されるか、 <code>SECONDARY</code> モードに切り替えられている必要があります。
14550	Hotstandby 接続パラメーターを変更できるのは、1 次サーバーが 2 次サーバーに接続されていないときだけです。
14551	START AFTER COMMIT ステートメントの最大数に到達しました。
14552	サーバーはバックアップ・サーバー・モードにあり、接続は許されません。

表 25. HotStandby のエラーおよび状況コード (続き)

エラーまたは状況コード	説明
14700	<p>接続が拒否されました。両方のサーバーが PRIMARY ロールです。</p> <p>意味: コマンド 'hsb connect' は、両方のノードのロールが同じ場合に、このエラーを返します。</p>
14701	<p>14701: 接続が拒否されました。両方のサーバーが SECONDARY ロールです。</p> <p>意味: コマンド 'hsb connect' は、両方のノードのロールが同じ場合に、このエラーを返します。</p>
14702	<p>14702: 操作が失敗しました。キャッチアップがアクティブです。</p> <p>意味: サーバーがキャッチアップを実行中に、1 次サーバー上で以下のいずれかのコマンドを発行すると、このエラーが発生します。'hsb switch secondary'、'hsb set secondary alone'、'hsb set standalone'、'hsb connect'、'hsb copy'、または 'hsb netcopy'。</p> <p>サーバーがキャッチアップを実行中に、2 次サーバー上で以下のいずれかのコマンドを発行すると、このエラーが発生します。'hsb switch primary'、'hsb set secondary alone'、'hsb set primary alone'、'hsb set standalone'、または 'hsb connect'。</p>
14703	<p>操作が失敗しました。コピーがアクティブです。</p> <p>意味: 1 次サーバーがコピーまたはネットコピーを実行中に、以下のコマンドがこのエラーを返します。'hsb switch secondary'、'hsb set secondary alone'、'hsb set standalone'、'hsb connect'、'hsb disconnect'、'hsb copy'、または 'hsb netcopy'。</p>
14704	<p>HotStandby のコピーまたはネットコピーは、1 次サーバーが PRIMARY ALONE 状態であるときにのみ許可されます。</p> <p>意味: このエラーは、サーバーが PRIMARY ACTIVE 状態にあり、コマンド 'hsb copy' または 'hsb netcopy' が発行された場合に返されます。</p>
14705	<p>この状態で STANDALONE に設定することは許されません。</p> <p>意味: サーバーが PRIMARY ACTIVE 状態にあるときに 'hsb set standalone' コマンドを発行すると、このメッセージを受け取ります。</p>
14706	<p>HotStandby に無効な読み取りスレッド・モードです。モード 2 のみがサポートされています。</p>
14707	<p>STANDALONE 状態で許可されない操作。</p>
14708	<p>キャッチアップが失敗しました。キャッチアップ位置をログ・ファイルから検出できませんでした。</p>
14709	<p>HotStandby は使用可能に設定されていますが、接続ストリングが定義されていません。</p>

表 25. HotStandby のエラーおよび状況コード (続き)

エラーまたは状況コード	説明
14710	HotStandby 管理コマンドが着信管理コマンドと競合しています。
14711	サーバーがシャットダウン操作の途中であるために失敗しました。
14712	サーバーが 2 次サーバーです。この操作には 1 次サーバーを使用してください。
30787	pri_dologskip:bad type, log pos, log size このエラーは、HSB 1 次サーバーで失敗した操作を指しています。このエラーは、失敗した操作とそのログ内の位置、およびログ・サイズを返します。レプリケーション・ログ内の操作はスキップされます。
30788	pri_hsblogcopy_write:bad type, log pos, log size このエラーは、HSB 1 次サーバーで失敗した操作を指しています。レプリケーション・ログ・ファイルへの書き込みは失敗します。このエラーは、失敗した操作とそのログ内の位置、およびログ・サイズを返します。
30789	ホット・スタンバイ・レプリケーション・ログ・ファイルを開くことができませんでした。
30790	HotStandby ログ用のメモリーを割り振ることができませんでした。最大ログ・サイズは <i>logsize</i> です。 このエラーは、hotstandby を使用するディスクレス・データベースに関するものです。それらのシステムでは、hotstandby ログはメモリーに書き込まれます。このエラーは、ログ・ファイルに、より多くのメモリーを割り振ろうとして失敗した場合に発行されます。

高可用性コントローラーのエラー・コードと状況コード

solidDB 高可用性コントローラー・エラー (17xxx) は、特定の高可用性コントローラー・コマンドを使用しているときに発生します。

表 26. 高可用性コントローラーのエラー・コードと状況コード

エラーまたは状況コード	説明
17501	HAC はシャットダウン操作の途中です。 コマンドの結果セット内にこの状況を返すことがある ADMIN COMMAND: <ul style="list-style-type: none"> ADMIN COMMAND 'hac shutdown' 意味: 高可用性コントローラーがシャットダウン操作の途中です。

表 26. 高可用性コントローラーのエラー・コードと状況コード (続き)

エラーまたは状況コード	説明
17502	<p>コマンドが失敗しました。HAC は中断状態です。</p> <p>コマンドの結果セット内にこの状況を返すことがある ADMIN COMMAND:</p> <ul style="list-style-type: none"> • ADMIN COMMAND 'hac suspend' • ADMIN COMMAND 'hac gethsbdstate' • ADMIN COMMAND 'hac getdbstate' • ADMIN COMMAND 'hac shutdowndb' • ADMIN COMMAND 'hac restartdb' • ADMIN COMMAND 'hac switchdb' • ADMIN COMMAND 'hac statemachinestate' • ADMIN COMMAND 'hac getereip' • ADMIN COMMAND 'hac pingere' <p>意味: コマンドの実行に失敗し、高可用性コントローラーは中断状態です。</p>
17503	<p>無効なコマンド。'hac commands' を入力してヘルプを表示してください。</p> <p>意味: 正しくない admin command 構文です。</p>
17504	<p>HAC は、既に稼働しています。</p> <p>コマンドの結果セット内にこの状況を返すことがある ADMIN COMMAND:</p> <ul style="list-style-type: none"> • ADMIN COMMAND 'hac resume'
17506	<p>この HSB 状態では、切り替えは許可されません。</p> <p>コマンドの結果セット内にこの状況を返すことがある ADMIN COMMAND:</p> <ul style="list-style-type: none"> • ADMIN COMMAND 'hac switchdb'
17507	<p>コマンドを実行できません。</p> <p>コマンドの結果セット内にこの状況を返すことがある ADMIN COMMAND:</p> <ul style="list-style-type: none"> • ADMIN COMMAND 'hac shutdowndb'
17509	<p>データベース・サーバーの再始動が失敗しました。詳細については、solmsg.out を参照してください。</p> <p>コマンドの結果セット内にこの状況を返すことがある ADMIN COMMAND:</p> <ul style="list-style-type: none"> • ADMIN COMMAND 'hac restartdb'

表 26. 高可用性コントローラーのエラー・コードと状況コード (続き)

エラーまたは状況コード	説明
17510	データベース・サーバーに接続できません。 コマンドの結果セット内にこの状況を返すことがある ADMIN COMMAND: <ul style="list-style-type: none"> ADMIN COMMAND 'hac switchdb'
17511	データベース・サーバーがシャットダウンされませんでした。 コマンドの結果セット内にこの状況を返すことがある ADMIN COMMAND: <ul style="list-style-type: none"> ADMIN COMMAND 'hac restartdb'
17513	切り替えが失敗しました。 コマンドの結果セット内にこの状況を返すことがある ADMIN COMMAND: <ul style="list-style-type: none"> ADMIN COMMAND 'hac switchdb'
17514	ERE IP が構成ファイル内で指定されていません。 コマンドの結果セット内にこの状況を返すことがある ADMIN COMMAND: <ul style="list-style-type: none"> ADMIN COMMAND 'hac getereip' ADMIN COMMAND 'hac pingere'
17516	HAC は、既にアクティブです。 コマンドの結果セット内にこの状況を返すことがある ADMIN COMMAND: <ul style="list-style-type: none"> ADMIN COMMAND 'hac setactive' ADMIN COMMAND 'hac pingere'
17517	HAC は、既にパッシブです。 コマンドの結果セット内にこの状況を返すことがある ADMIN COMMAND: <ul style="list-style-type: none"> ADMIN COMMAND 'hac setpassive'

solidDB データベース・エラー

solidDB データベース・エラー (10002 - 10050) は solidDB によって検出され、クライアント・アプリケーションへ送信されます。これらのエラーには、管理アクションを必要とするものもあります。

表 27. solidDB データベース・エラー

エラー・コード	説明
Database Error 10002	<p>操作が失敗しました。</p> <p>意味: 予期しなかったエラーで接続操作が失敗しました。最も考えられることは、サーバー同士が正しく同期していないことです。</p>
Database Error 10013	<p>トランザクションが読み取り専用です。</p> <p>意味: 読み取り専用を設定されたトランザクションの内部で書き込みを試みたか、サーバーが (例えば状態切り替え中で) 一時的に読み取り専用モードに設定されています。更新可能なトランザクションは許可されません。</p>
Database Error 10019	<p>バックアップは、既にアクティブです。</p> <p>意味: 既にバックアップまたはコピーが進行中であるときに、それを開始しようとしてしました。</p>
Database Error 10024	<p>バックアップ・ディレクトリー 「<i>directory_name</i>」が正しくありません。</p> <p>意味: バックアップまたはコピー・ディレクトリーが空ストリングであるか、ドット (現行ディレクトリーにバックアップまたはコピーが作成されることを示します) です。</p>
Database Error 10030	<p>バックアップまたはコピー・ディレクトリー 「<i>directory_name</i>」は存在しません。</p> <p>意味: バックアップまたはコピー・ディレクトリーが見つかりません。バックアップまたはコピー・ディレクトリーの名前を確認してください。</p>
Database Error 10045	<p>この操作を HotStandby 2 次サーバーに対して実行することはできません。</p> <p>意味: この操作を HotStandby 2 次サーバーに対して実行することはできません。</p> <p>要求した操作を成功させるには、サーバーが 1 次サーバーでなければなりません。</p>
Database Error 10046	<p>操作が失敗しました。データ・ディクショナリー操作がアクティブです。</p> <p>意味: データ・ディクショナリー操作が現在進行中です。</p>

表 27. solidDB データベース・エラー (続き)

エラー・コード	説明
Database Error 10047	複製されたトランザクションが異常終了しました。 意味: トランザクションが、例えば状態切り替えなどで、異常終了しました。サーバー状態が 1 次サーバーから 2 次サーバーへ切り替えられるとき、すべてのアクティブ・トランザクションは異常終了します。
Database Error 10048	複製されたトランザクションにデータ・ディクショナリー変更が含まれており、通常の更新操作は許可されません。 意味: HotStandby モードでは、データ・ディクショナリー操作は制限されています。例えば、CREATE TABLE を通常の更新操作と混在させることはできません。 このメッセージは、バージョン 4.1 以降では廃止されました。バージョン 4.1 以降では、HSB を使用している間、トランザクション内に DML 操作と DDL 操作を混在させることができます。
Database Error 10049	リモート・サーバーが 2 次サーバーではありません。 意味: コマンド内で指定したサーバーは、SECONDARY 状態ではありません。
Database Error 10050	複製された操作で BLOB 列を更新しました。 意味: BLOB 列を 2 次サーバーへ複製することはできません。
Database error 10078	ユーザーがトランザクションをロールバックしました。
Database error 10079	filespec を除去できません。ファイルは、既に使用中です。
Database error 10080	HotStandby 2 次サーバーが、1 次サーバーから受け取った操作を実行できません。 意味: このエラーの原因として考えられることは、データベースが HotStandby の copy または netcopy コマンドで 1 次サーバーから作成されなかったことです。

表 27. solidDB データベース・エラー (続き)

エラー・コード	説明
Database error 10081	データベース・ファイルが不完全であるか破損しています。 意味: ファイルがホット・スタンバイの 2 次サーバー上にある場合は、hotstandby copy または hotstandby netcopy コマンドを使用してファイルを 1 次サーバーから再度送信してください。
Database error 10082	バックアップが異常終了しました。
Database error 10083	既に 2 次サーバーへコミットが送信されているため、HSB トランザクションを中止することができませんでした。
Database error 10084	表がロックされていません。
Database error 10085	チェックポイントは使用不可です。
Database error 10087	HotStandby はメイン・メモリー表に対しては許可されません。
Database error 10088	指定したロック・タイムアウト値が大きすぎます。
Database error 10089	操作が失敗しました。サーバーは HSB PRIMARY UNCERTAIN モードです。

solidDB 表エラー

solidDB データベース表エラー (13xxx) は、エラーのある SQL ステートメントが原因で発生し、solidDB によって検出されます。管理アクションは必要ありません。

表 28. solidDB 表エラー

エラー・コード	説明
Table Error 13068	サーバーのシャットダウンが進行中です。 意味: サーバーのシャットダウンが進行中のため、この操作を完了することはできません。
Table Error 13123	表「table_name」が空ではありません。 意味: この操作は、表が空の場合にのみ実行できます。例えば、ディスク・ベースからインメモリー (またはその逆) への表の変更は、表が空のときのみ行うことができます。
Table Error 13167	インメモリー表のみをトランジエント表とすることができます。 意味: ディスク・ベースのトランジエント表を作成することはできません。例えば、以下の SQL ステートメントは、このエラー・メッセージを生成します。 CREATE TRANSIENT TABLE t1 (i INT) STORE DISK;

表 28. *solidDB* 表エラー (続き)

エラー・コード	説明
Table Error 13170	<p>インメモリー表のみをテンポラリー表とすることができます。</p> <p>意味: ディスク・ベースのテンポラリー表を作成することはできません。例えば、以下の SQL ステートメントは、このエラー・メッセージを生成します。</p> <pre>CREATE TEMPORARY TABLE t1 (i INT) STORE DISK;</pre>

solidDB 通信エラー

solidDB 通信エラー (21306、21308) は、ネットワーク・エラーが原因で発生します。これらのエラーは、管理アクションを必要とします。

表 29. *solidDB* 通信エラー

エラー・コード	説明
Communication Error 21306	<p>サーバー「<i>server_name</i>」が見つからず、接続が失敗しました。</p> <p>意味: 2 次サーバーが見つかりませんでした。</p> <ul style="list-style-type: none"> • サーバー稼働しているかどうかを確認してください。 • ネットワーク名が有効かどうかを確認してください。 • サーバーが指定されたネットワーク名を listen しているかどうかを確認してください。
Communication Error 21308	<p>「Connection is broken (%s '%s' operation failed with code %d).」</p> <p>例:</p> <p>「Connection is broken (TCP/IP 'Write' operation failed with code 7).」</p> <p>エラー 21306 の推奨処置を参照してください。</p>

付録 C. HotStandby 管理コマンドの要約

このセクションでは、HotStandby で使用可能な管理コマンドを要約を示します。

例えば、solidDB SQL エディター (solsql) または solidDB リモート制御 (solcon) などの使用ツールは、コマンドをどのように入力しなければならないかに影響します。

solidDB SQL エディター (テレタイプ) の使用では、下に示した構文を使用してコマンドを入力します。

```
ADMIN COMMAND 'hotstandby switch primary';
```

これらのコマンドを solidDB リモート制御 (テレタイプ) で入力する場合は、コマンドのみを (引用符なし、「ADMIN COMMAND」なしで) 次のように入力します。

```
hotstandby switch primary
```

'hotstandby' を 'hsb' に省略して次のように入力できる点にも注意してください。

```
hsb switch primary
```

下の表では、可能な最短形式が使用されています。「ADMIN COMMAND」および引用符は省略されて、省略形 'hsb' および 'hac' が使用されています。

solidDB リモート制御 (テレタイプ) または solidDB SQL エディターの詳細については、「*solidDB 管理者ガイド*」の『solidDB データ管理ツールの使用』を参照してください。

ADMIN COMMAND は、コマンドに構文エラーがなければ必ず戻りコード (0) を返します。コマンドの実際の結果コードは、結果セットの「RC」フィールドに含まれます。

HotStandby コマンド

コマンド	説明
hsb cominfo	<p>もう一方のサーバーとの接続に使用される通信情報 (接続ストリング) を返します。これは通常は、solid.ini 構成ファイルの HotStandby セクションにある Connect パラメーターの値ですが、以下のコマンドで設定された可能性もあります。</p> <pre>ADMIN COMMAND 'hsb parameter connect connect_string';</pre> <p>この情報は、他のサーバーに接続するためにアプリケーションで使用することができます。</p>

コマンド	説明
hsb connect	<p>1 次サーバーおよび 2 次サーバー間の接続が切れるか、またはまだ確立していない場合、このコマンドは 1 次サーバーを 2 次サーバーに接続して、HotStandby レプリケーションを開始します。サーバー間には自動接続メカニズムがないため、このコマンドは、サーバー間の接続に常に必要です。正常に接続した後、1 次サーバーの状態は自動的に PRIMARY ALONE から PRIMARY ACTIVE に設定されます。正常に接続できなかった場合、状態は PRIMARY ALONE のままです。</p> <p>このコマンドは、1 次サーバーまたは 2 次サーバーのいずれでも実行できます。</p> <p>注: このコマンドを実行したとき、1 次サーバーおよび 2 次サーバーが接続済みであるが、トランザクション・ログがまだ完全には 2 次側へコピーされていないければ、キャッチアップがアクティブであることを示すメッセージ「Catchup is active」が表示されます。</p>
hsb copy [<i>directory_name</i>]	<p>注: このコマンドは推奨されません。代わりに hsb netcopy コマンドを使用することをお勧めします。</p> <p>hsb copy コマンドを使用して、1 次側から最初に 2 次データベースを作成できます。このコマンドは、データベースを 1 次ノードのローカル・ディレクトリーに (および 2 次ノードのローカル・ディレクトリーにも) コピーします。コピーの完了後、2 次サーバーを始動できます。1 次サーバーを 2 次サーバーに接続した後、1 次サーバーは、トランザクション・ログを 2 次サーバーにコピーすることによって、自動的に 2 次サーバーを最新の状態にします。</p> <p>このコマンドを使用して、1 次データベースを、1 次ノードのローカル・ディレクトリーにある 2 次データベースと (それが相当の期間オフラインになっていたときに) 同期化することもできます。87 ページの『1 次サーバーと 2 次サーバーの同期』を参照してください。</p> <p>オプションの <i>directory_name</i> を指定した場合、データベース・ファイルはそのディレクトリーにコピーされます。指定しなければ、solid.ini 構成ファイルの [Hotstandby] セクションにある copydirectory パラメーターで指定したディレクトリーにコピーされます。hsb copy コマンドは solid.ini 構成ファイルもログ・ファイルもコピーしないため、このディレクトリーは通常のバックアップ・ディレクトリーとは異なるディレクトリーにすることをお勧めします。</p> <p>1 次側は、PRIMARY ALONE 状態の場合にのみ hsb copy コマンドを実行できます。コマンドの実行中およびその後で、サーバーは PRIMARY ALONE 状態のままです。コマンドが完了した後で、2 次サーバーを始動して、2 つのサーバーを接続できます。</p>

コマンド	説明
hsb disconnect	<p>これは、HSB ペアのもう一方のメンバーから安全に切断するようサーバーに通知します。このコマンドは、1 次サーバーまたは 2 次サーバーのいずれでも有効です。このコマンドを使用する典型的な理由は、両方のサーバーのいずれかをアップグレードする前にそれらを切断することです。(もう一方サーバーは、それが引き続きクライアント要求へ応答できるように PRIMARY ALONE 状態に設定できます。)</p> <p>このコマンドによって通常は、両方のサーバーが「単体 (Alone)」モードになります。つまり、1 次サーバーは PRIMARY ACTIVE から PRIMARY ALONE に切り替わり、2 次サーバーは SECONDARY ACTIVE から SECONDARY ALONE に切り替わります。</p> <p>このコマンドは 1 次側および 2 次側の両方で有効です。</p> <p>次の shutdown コマンドを使用するときには以下の点にご注意ください。</p> <pre>ADMIN COMMAND 'shutdown';</pre> <p>サーバーはシャットダウンする前に、制御された方式で切断を行います。2 次側がシャットダウン (および切断) した場合、1 次側は PRIMARY ALONE 状態になっても安全なことを認識してそれを実行します。</p>
hsb logpos	<p>2 つのこれらのコマンドは、Watchdog プログラム (または人) が 2 つのサーバーのいずれかを 1 次側に切り替え、いずれかを 2 次側に切り替えるべきかを決定するために使用できます。(サーバー同士の互いの接続が失われる前に 1 次サーバーだったサーバーが、必ずしも、この時点で 1 次サーバーになるべきサーバーだとは限りません。)この手法では、どのサーバーが「先」にいるか、(つまり、いずれがより多くのトランザクションを受け入れたか)、そして HSB 接続の確立前に 1 次側にするべきかを検出します。</p> <p>このコマンドの使用方法の詳細については、102 ページの『どちらのサーバーを 1 次サーバーにするかの選択』を参照してください。</p> <p>標準的な出力を下に示してあります。</p> <pre>ADMIN COMMAND 'hsb logpos'; RC TEXT -- ---- 0 0000000000000000000000000871:PRIMARY</pre> <p>出力はログ操作 ID およびサーバーの以前のロール (この例では、Primary だった) から構成されています。</p>

コマンド	説明
hsb netcopy	<p>このコマンドは、1 次データベースまたはディスクレス・インメモリー・データを 2 次サーバーへ、solid.ini の [HotStandby] セクションにある Connect パラメーターを使ってコピーするために使用します。接続ストリングを使用して 2 次サーバーに接続すると、データベース・ファイルがネットワーク・リンク経由でコピーされます。</p> <p>このコマンドを使用して、1 次データベースを、長時間オフラインになっていた 2 次データベースと同期化することができます。87 ページの『1 次サーバーと 2 次サーバーの同期』を参照してください。</p> <p>このコマンドを使用して、新規 2 次データベースを作成することもできます。この理由として考えられるのは、2 次データベースの破損、新規 HotStandby 構成用の初期 2 次データベースの作成、または既存の構成への新規 2 次データベースの追加があります。詳細については、91 ページの『ネットワークを介した 1 次データベースの 2 次側へのコピー』を参照してください。</p> <p>このコマンドを発行するには、1 次サーバーが PRIMARY ALONE 状態であればなりません。</p> <p>コマンドの完了後 (正常でも、そうでなくても)、1 次サーバーは同じ状態のままです。</p> <p>コピーが正常に完了した場合、2 次サーバーは自動的に SECONDARY ALONE 状態に切り替わります。</p> <p>netcopy コマンドの後には通常、1 次サーバーおよび 2 次サーバーを接続するために「connect」コマンドが続きます。2 次サーバーに接続した後、1 次サーバーは、トランザクション・ログをコピーすることによって、自動的に 2 次サーバーを最新の状態にします。</p>
hsb parameter	<p>(推奨されない) このコマンドを使用すると、HSB 固有のパラメーター、例えば、AutoPrimaryAlone、Connect、および PingTimeoutなどを設定できます。それぞれのパラメーターの詳細については、117 ページの『付録 A. HotStandby 構成パラメーター』を参照してください。</p> <p>いくつかのパラメーターの 1 つの値を設定すると、そのコマンドはすぐに有効になりますが、solid.ini 構成ファイルには、シャットダウンが実行されるまで書き込まれません。</p> <p>このコマンドの構文は以下のとおりです。</p> <pre>ADMIN COMMAND 'hsb parameter param_name param_value';</pre> <p>このコマンドでは等号記号を使用しないことに注意してください。したがって、以下のような他の類似の (推奨される) コマンドとは異なります。</p> <pre>ADMIN COMMAND 'parameter hotstandby.param_name = param_value';</pre>

コマンド	説明
hsb role	<p>注: このコマンドは推奨されません。代わりに <code>hsb state</code> を使用してください。</p> <p>以下のロールのいずれかを結果セットで返します。</p> <ul style="list-style-type: none"> • PRIMARY (接続済みサーバーが通常の 1 次サーバーの場合)。このロールでは、1 次サーバーのトランザクションが 2 次サーバーに送信されます。 • PRIMARY NOHSBLOG (1 次サーバーがトランザクションを受け入れ、データベースに格納することを示す)。ただし、それらのトランザクションが後で 2 次側へ送信できるようにログに格納されることはありません。2 次側と 1 次側の再同期を行うには、1 次側にあるデータベース全体を 2 次サーバーにコピーする必要があります。 • PRIMARY BROKEN (1 次サーバーで 2 次サーバーとの間の接続が切れた場合)。1 次サーバーでは読み取り専用トランザクションのみを実行できます。 • PRIMARY ALONE (1 次サーバーが単独で動作している場合)。2 次側との接続は切れていますが、後で 2 次側に送信できるように 1 次側でトランザクションを受け入れ、トランザクション・ログに追加します。 • PRIMARY CATCHUP (キャッチアップが進行中の場合)。キャッチアップ中に、<code>'hsb connect'</code> コマンドが 1 次側で発行されるとその後で、1 次側は自動的にトランザクション・ログの変更を 2 次サーバーへ送信します。キャッチアップ・プロセスの完了後、サーバーのロールは自動的に PRIMARY へ切り替わります。1 次側は、そのロールが接続前に PRIMARY ALONE だった場合には、引き続きトランザクションを受け入れます。 • SECONDARY (接続済みサーバーが通常の 2 次サーバーの場合)。つまり、サーバーは 1 次側からトランザクションを受信して適用します。 • SECONDARY BROKEN (2 次サーバーで 1 次サーバーとの間の接続が切れた場合)。 • SECONDARY CATCHUP (<code>'hsb connect'</code> コマンドが 1 次サーバーで発行された後で、2 次サーバーが、1 次サーバーからの変更内容のキャッチアップ中の場合)。キャッチアップ・プロセスの完了後、2 次側のロールは自動的に SECONDARY へ切り替わります。 <p>ADMIN COMMAND <code>'hsb role'</code> が、HotStandby 用に構成されていないサーバーで発行された場合は、エラー・メッセージ「14527: This is a non-HotStandby Server.」が返されます。</p> <p>このコマンドは、SQL 関数 <code>HOTSTANDBY_ROLE</code> と同じ情報を返します。</p>

コマンド	説明
hsb set primary alone	<p>HotStandby 1 次サーバーを無条件に PRIMARY ALONE 状態に設定します。このコマンドは、PRIMARY ACTIVE、SECONDARY ACTIVE、SECONDARY ALONE、および STANDALONE の状態で有効です。</p> <p>このコマンドを使用して、高速フェイルオーバーをインプリメントできます。2 次側が SECONDARY ACTIVE 状態のとき、サーバーは、このコマンドを受信しても、1 次側との通信をまったく試みません。代わりに、すぐに PRIMARY ALONE 状態に切り替えます。この動作は、1 次側の障害に関する情報が、2 次側でその障害を検出する前に HAC に達する場合に利用できません (遅延は PingTimeout および PingInterval パラメーターに従います)。</p> <p>ただし、例えば、正しくない障害検出などのため、このコマンドが 2 次側で実行されたときに、1 次側が「活動中」で、PRIMARY ACTIVE 状態であった場合、1 次側は自動的に PRIMARY UNCERTAIN 状態に強制されます。その後で、トランザクションをまったく失うことなく、SECONDARY ALONE 状態に移して再接続することができます。</p> <p>注: フェイルオーバーを実行する代替方法として、hsb switch primary コマンドを使用することがあります。</p> <p>PRIMARY ALONE 状態では、2 次サーバーとの接続は切れますが、この状態では、1 次サーバーがトランザクション・ログの更新を継続して実行できます。PRIMARY ALONE 状態は、1 次サーバーがシャットダウンするまで、2 次サーバーとの接続が正常に行われるまで、あるいはサーバーでトランザクション・ログ用のスペースが使い尽くされるまで持続します。</p> <p>PRIMARY ALONE 状態に設定した場合、サーバーは、他のサーバーとの接続の再確立を自動的に試みることはないことに注意してください。</p> <p>重要: このコマンドをサーバー上で実行する前に、ペアのもう一方のサーバーがまだ PRIMARY ALONE 状態 (または STANDALONE 状態) になっていないことを確認するようにしてください。「二重の 1 次」を回避することは非常に重要です (38 ページの『ネットワーク分割と二重 1 次サーバー』を参照)。</p> <p>コマンド 'hsb switch primary' も参照してください。</p>
hsb set secondary alone	<p>このコマンドは、サーバーの状態を SECONDARY ALONE に設定します。このコマンドが使用可能なのは、サーバーが現在、PRIMARY ALONE、PRIMARY UNCERTAIN、STANDALONE のいずれかの状態になっている場合です。</p>
hsb set standalone	<p>このコマンドが発行された場合、1 次サーバーの状態は STANDALONE になります。サーバーは 2 次サーバーのためのトランザクションの格納を停止します。1 次側 (STANDALONE) は引き続き、読み取り/書き込みトランザクションを受け入れることができます。このオプションは、2 次サーバーが相当の期間オフラインで、トランザクション・ログが大きくなりすぎている可能性がある場合に 1 次サーバーで役立ちます。このコマンドが使用可能なのは、サーバーが現在、PRIMARY ALONE または SECONDARY ALONE の状態になっている場合です。</p>

コマンド	説明
hsb state	<p>以下のいずれかの状態を結果セットで返します。</p> <ul style="list-style-type: none"> • PRIMARY ACTIVE (接続済みサーバーが通常の 1 次サーバーの場合)。この状態では、1 次サーバーのトランザクションが 2 次サーバーに送信されます。 • STANDALONE (1 次サーバーがトランザクションを受け入れ、データベースに格納するが、それらのトランザクションを 2 次側へ転送するための格納ではないことを示す)。 • PRIMARY UNCERTAIN (1 次サーバーで 2 次サーバーとの間の接続が切れ、PRIMARY ALONE などの別の状態にまだ切り替わっていない場合)。1 次サーバーでは読み取り専用トランザクションのみを実行できます。 • PRIMARY ALONE (1 次サーバーが単独で動作している場合)。2 次側との接続は切れていますが、2 次側に転送できるようにトランザクションを受け入れ、1 次側のトランザクション・ログに格納します。 • SECONDARY ACTIVE (接続済みサーバーが通常の 2 次サーバーの場合)。つまり、サーバーは 1 次側からトランザクションを受信して適用します。 • SECONDARY ALONE (2 次サーバーで 1 次サーバーとの間の接続が切れた場合)。 <p>ADMIN COMMAND 'hsb state' が、HotStandby 用に構成されていないサーバーで発行された場合は、エラー・メッセージ「14527: This is a non-HotStandby Server.」が返されます。</p> <p>このコマンドは、SQL 関数 HOTSTANDBY_STATE と同じ情報を返します。この関数の詳細については、75 ページの『新規 1 次サーバーへのアプリケーションの切り替え』の『HOTSTANDBY_STATE 関数の使用』セクションを参照してください。</p> <p>管理操作およびトラブルシューティング操作の実行中に起きる HotStandby 状態遷移の概要については、157 ページの『付録 D. サーバー状態遷移』を参照してください。</p>
hsb status option	<p>HotStandby 状況情報を返します。オプションには以下のいずれかを使用できます。</p> <ul style="list-style-type: none"> • catchup • connect • copy • switch <p>詳細については、下記の個々のコマンド/オプション (例えば、'hsb status catchup') の説明を参照してください。</p> <p>status コマンドは、正常に開始してから長時間かかる操作の結果に関する情報を提供するためのコマンドです。このコマンドは、最後に正常に開始した操作の状況を返します。例えば、正しくない状態などの原因により、操作の開始が失敗した場合、status コマンドは、その操作ではなく、その前に実行された操作の状況を返します。</p>

コマンド	説明
hsb status catchup	<p>サーバーがキャッチアップを行っているかどうか、つまり 2 次側が 1 次側のトランザクション・ログを読み取ったときに、その変更内容を適用するかどうかを示します。</p> <p>可能な値は以下のとおりです。</p> <ul style="list-style-type: none"> • ACTIVE • NOT ACTIVE
hsb status connect	<p>返される状況情報は以下のとおりです。</p> <ul style="list-style-type: none"> • CONNECTED - 接続がアクティブ。この情報は、1 次サーバーおよび 2 次サーバーの両方から返されます。 • CONNECTING - 1 次サーバーおよび 2 次サーバーが相互に接続しています。この情報は、1 次サーバーおよび 2 次サーバーの両方から返されません。 • CATCHUP - 1 次サーバーが 2 次サーバーに接続していますが、1 次 HotStandby のデータベース・ログが 2 次サーバーに完全にはコピーされていません。この情報は、1 次サーバーおよび 2 次サーバーの両方から返されます。 • • BROKEN - 1 次および 2 次サーバー間の接続が切れました。この情報は、1 次サーバーおよび 2 次サーバーの両方から返されます。 <p>注: このコマンドは、SQL 関数 HOTSTANDBY_CONNECTSTATUS と同じ情報を返します。この関数の詳細については、75 ページの『新規 1 次サーバーへのアプリケーションの切り替え』の『HOTSTANDBY_CONNECTSTATUS 関数の使用』セクションを参照してください。</p>
hsb status copy	<p>このコマンドを使用すると、最後の hsb copy または hsb netcopy コマンドの結果を確認できます。この status コマンドは、copy ではなく netcopy の結果を確認する場合でも、必ずキーワード「copy」を使用することに注意してください。</p> <p>返される状況情報は以下のとおりです。</p> <ul style="list-style-type: none"> • SUCCESS - コピーが正常に完了しました。 • ACTIVE - コピー・プロセスがまだアクティブです。 • ERROR <i>number</i> - コピーがエラー・コード <i>number</i> で失敗しました。
hsb status switch	<p>HotStandby 切り替えの状況情報を返します。返される状況情報は以下のとおりです。</p> <ul style="list-style-type: none"> • ACTIVE - コピー・プロセスがまだアクティブです。 • SUCCESS - コピーが正常に完了しました。 • ERROR <i>number</i> - コピーがエラー・コード <i>number</i> で失敗しました。 • NO SERVER SWITCH OCCURRED BEFORE - 前に切り替えが起きていません。

コマンド	説明
hsb switch primary	<p>データベース・サーバーを PRIMARY に切り替えます。このコマンドは切り替えプロセスを開始します。開始したプロセスは、hsb status switch コマンドを使用してモニターできます。</p> <p>このコマンドの実行時にサーバー間が接続している場合には、単に状態を逆にするだけです。つまり、古い 1 次側が PRIMARY ACTIVE から SECONDARY ACTIVE に変更され、2 次サーバーが SECONDARY ACTIVE から PRIMARY ACTIVE に切り替わります。</p> <p>サーバー間が接続していない状態で、サーバーが SECONDARY ALONE 状態の場合には、サーバーを 1 次に切り替えると、結局それは PRIMARY ALONE 状態になります。新しい 1 次サーバーは、もう一方のサーバーに接続して PRIMARY ACTIVE 状態への切り替えを自動的に試行することはありません。</p> <p>このコマンドは SECONDARY ACTIVE および SECONDARY ALONE 状態の両方で使用可能なため、フェイルオーバーの実行に使用できます。ただし、サーバーは 1 次側との通信を必ず試みるため、ネットワーク・タイムアウトになる可能性があります。したがって、この方式は 'hsb set primary alone' コマンドを使用する場合よりも低速です。一方、この方式は「二重の 1 次」に対するより優れた安全対策になります。</p> <p>'hsb set primary alone' コマンドも参照してください。</p>
hsb switch secondary	<p>データベースを 2 次の状態に切り替えます。すべてのアクティブな書き込みトランザクションが終了します。</p> <p>注: 接続済みデータベース・サーバーが 1 次サーバーである場合は、それが 2 次サーバーになります。古い 2 次サーバーが使用可能な場合は、その古い 2 次サーバーが 1 次に切り替わります (hsb switch primary コマンドを参照)。</p> <p>注: switch コマンドがオープン・トランザクション内部で発行された場合には (トランザクションの開始後で COMMIT ステートメントの実行前の Microsoft Windows)、COMMIT ステートメントを発行すると、その COMMIT はエラー 'replicated transaction is aborted' で失敗します。切り替え中に、switch ステートメントが実行されたトランザクションを含む、すべてのトランザクションがロールバックされます。ADMIN COMMAND はトランザクション・コマンドではないため、切り替え自体は正常に行われます (つまり、ロールバックされません)。(注: ただし、管理コマンドは、新規トランザクションがまだオープン状態でなければ、新規トランザクションの開始を実際には強制します。)</p>

高可用性コントローラーのコマンド

表 30. 高可用性コントローラーのコマンド

コマンド	説明
hac shutdown 省略形: hac sd	このコマンドは HAC プロセスを終了します。

付録 D. サーバー状態遷移

この章では、可能な状態遷移 (例えば、OFFLINE から SECONDARY ALONE への遷移) について説明します。

それぞれのサーバー状態の説明が 13 ページの『サーバー状態の説明』にあります。

HotStandby 状態遷移図

このセクションの図は、起きる可能性のある状態遷移、およびそれらが起きることのある環境を示しています。

例えば、次のようにコマンド 'hsb set primary alone' を実行することによって、サーバーの状態を PRIMARY UNCERTAIN から PRIMARY ALONE に変更できます。

```
ADMIN COMMAND 'hsb Set Primary Alone';
```

この遷移図を使う際には、以下の点に注意してください。

1. コマンドの完全な構文は示してありません。例えば、以下のように示してあります。

```
'hsb set primary alone'
```

これの完全な構文は以下のとおりです。

```
ADMIN COMMAND 'hsb Set Primary Alone';
```

2. 'hsb copy' に示されている状態遷移パスは 'hsb netcopy' にも適用されます。
3. コマンドによっては、実行すると失敗することがあります。コマンドが成功または失敗するときは、両方の可能性が示されます。コマンドが失敗したときにどうなるかを分岐によって説明しようとする場合は、'Disconnect' 失敗 のように、「失敗」という単語が付きます。
4. 一部の状態では、動作は **AutoPrimaryAlone** という名前の solid.ini 構成パラメーターの設定によって決まります。多くの場合、このパラメーターを表すのに省略形「APA」を使用します。
5. 図で「イベント」を指している場合、それは内部生成の通知を指しています。これらは、例えば、SQL コマンドの CREATE EVENT に説明されている、ユーザーが通知して待機可能な「イベント」と同じではありません。
6. 図の左上近くに、「'-x backupserver' で始動」とあります。新規 2 次サーバーを始動して、それに「netcopy」コマンド経由で 1 次サーバーからデータベースのコピーを取得したい場合には、そのサーバーを (オペレーティング・システムのコマンド行から) コマンド行オプション -x backupserver で始動します。これはサーバーに対して、1 次側からの netcopy を待機するよう指示しています。サーバーは netcopy の受信を待機する一方で、その状態 (またはロール) に関する照会に回答しないことに注意してください。例えば、以下のコマンドを発行した場合、

```
ADMIN COMMAND 'hsb state';
```

サーバーは応答しないので、実際には、それが状態「OFFLINE」を返すことを確認するわけではありません。

7. 「rpc」は「リモート・プロシージャ・コール」を表します。「rpc broken」は、1 次および 2 次側が、明示的な切断 (Disconnect) なしで、相互接続が失われたことを意味します。接続は、ネットワークで障害が起きた場合、または一方のサーバーがクラッシュした場合などに失われる可能性があります。
8. 矢印がその開始元の同じ状態にループバックする場合、それはその状態が変化しないことを意味します。例えば、PRIMARY ALONE 状態のサーバーが、もう一方のサーバーに接続を試みたが失敗した場合に、状態は PRIMARY ALONE のままです。

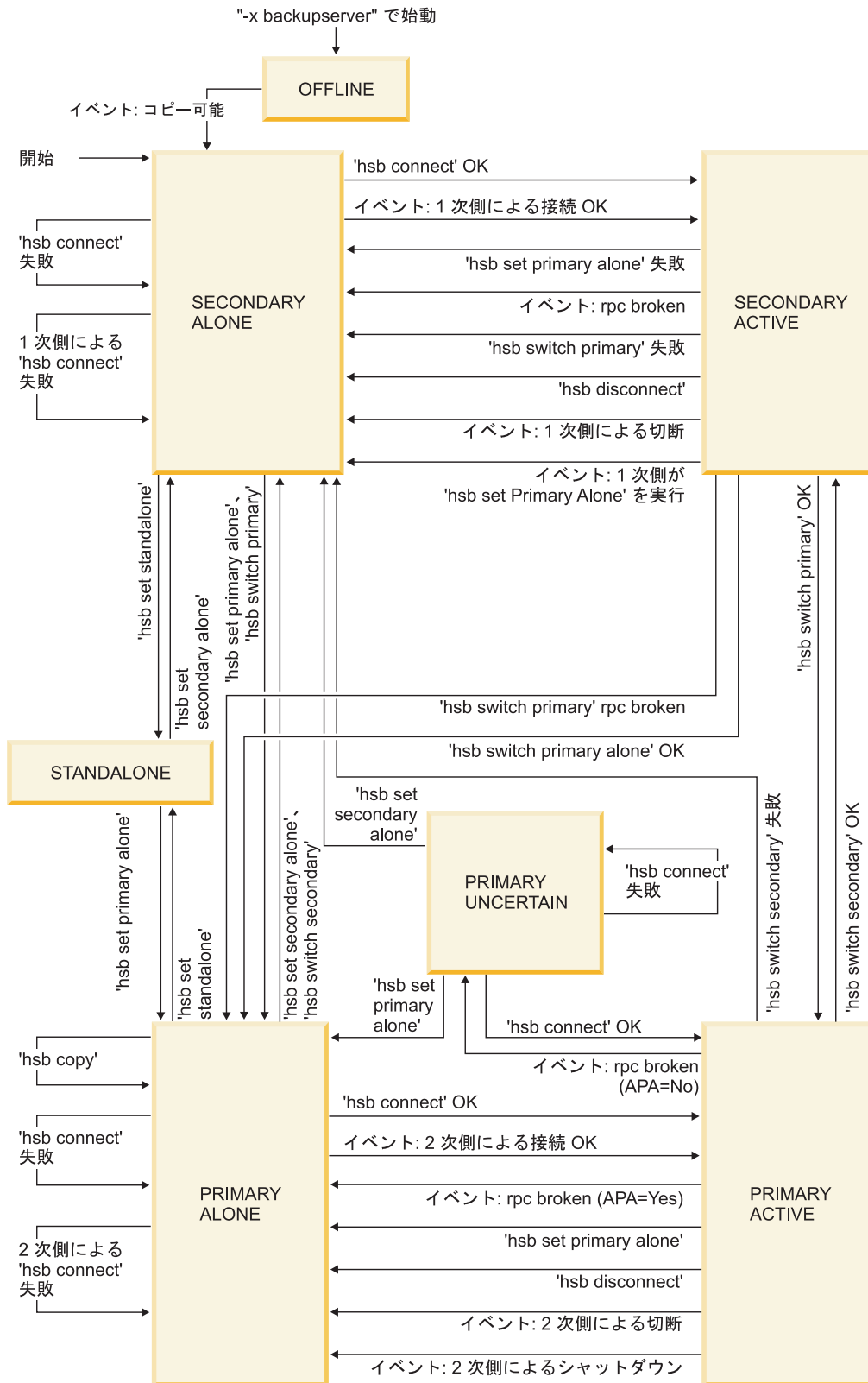


図 15. HotStandby サーバー状態遷移

以下の表は、サーバー状態と、HotStandby コマンドでサーバー状態を変更できる方法を示しています。

表 31. サーバー状態遷移表

サーバー状態	この条件が起きた場合、またはこの HSB コマンドが発行された場合...	サーバーは以下の状態になる...	コマンドが正常でない場合は、以下の状態である...
OFFLINE	1 次サーバーが ADMIN COMMAND 'hotstandby netcopy' を実行した場合、2 次サーバーの状態は、データベースのコピー後に SECONDARY ALONE に変更されます。	SECONDARY ALONE	未変更
PRIMARY ACTIVE	AutoPrimaryAlone = Yes の場合、HotStandby タイムアウト (自動)。 注: HSB タイムアウトは、2 次サーバーがダウンしたか、または 1 次および 2 次間の接続が切れた場合に自動的に起きます。	PRIMARY ALONE	(適用外)
PRIMARY ACTIVE	AutoPrimaryAlone = No の場合、HotStandby タイムアウト (自動)。 注: HSB タイムアウトは、2 次サーバーがダウンしたか、または 1 次および 2 次間の接続が切れた場合に自動的に起きます。	PRIMARY UNCERTAIN	(適用外)
PRIMARY ACTIVE	1 次サーバーで ADMIN COMMAND 'hotstandby set standalone'	STANDALONE	未変更
PRIMARY ACTIVE	1 次サーバーで ADMIN COMMAND 'hotstandby switch secondary' または 2 次サーバーで ADMIN COMMAND 'hotstandby switch primary'	SECONDARY ACTIVE	SECONDARY ALONE
PRIMARY ACTIVE	1 次サーバーで ADMIN COMMAND 'hotstandby disconnect'	PRIMARY ALONE	PRIMARY ALONE

表 31. サーバー状態遷移表 (続き)

サーバー状態	この条件が起きた場合、またはこの HSB コマンドが発行された場合...	サーバーは以下の状態になる...	コマンドが正常でない場合は、以下の状態である...
PRIMARY ALONE	<p>1 次サーバーで ADMIN COMMAND 'hotstandby copy' または ADMIN COMMAND 'hotstandby netcopy'</p> <p>1 次サーバーの状態は変更されないことに注意してください。サーバーは PRIMARY ALONE 状態のままです。状態を PRIMARY ACTIVE に変更するには、「connect」コマンドを ADMIN COMMAND 'hotstandby connect'; のように発行する必要があります。</p> <p>注: ディスクレス・サーバーを使用して、2 次サーバーへのファイル・アクセスを行わない場合は、copy ではなく、netcopy を使用してください。</p>	PRIMARY ALONE	PRIMARY ALONE
PRIMARY ALONE	<p>1 次サーバーで ADMIN COMMAND 'hotstandby connect'</p> <p>注: 上記のコマンドは、2 次サーバー (今現在修正されている)、または障害が起きた 2 次側以外のサーバーに接続するために使用します。</p>	PRIMARY ACTIVE (キャッチアップの完了後)	未変更
PRIMARY ALONE	<p>1 次サーバーで ADMIN COMMAND 'hotstandby set standalone', またはトランザクション・ログがフル</p>	STANDALONE	未変更
PRIMARY ALONE	<p>1 次サーバーで ADMIN COMMAND 'hotstandby set secondary alone' または ADMIN COMMAND 'hotstandby switch secondary'</p>	SECONDARY ALONE	SECONDARY ALONE

表 31. サーバー状態遷移表 (続き)

サーバー状態	この条件が起きた場合、またはこの HSB コマンドが発行された場合...	サーバーは以下の状態になる...	コマンドが正常でない場合は、以下の状態である...
PRIMARY UNCERTAIN	1 次サーバーで ADMIN COMMAND 'hotstandby set primary alone' at the Primary server	PRIMARY ALONE	未変更
PRIMARY UNCERTAIN	1 次サーバーで ADMIN COMMAND 'hotstandby connect' 注: 上記のコマンドは、2 次サーバー (今現在修正されている) に接続するため、または障害が起きた 2 次側以外のサーバーに接続するために使用します。	PRIMARY ACTIVE	未変更
PRIMARY UNCERTAIN (2 次側との接続で HSB タイムアウトが起きた)	1 次サーバーで ADMIN COMMAND 'hotstandby set standalone'	STANDALONE	未変更
PRIMARY UNCERTAIN	1 次サーバーで ADMIN COMMAND 'hotstandby set secondary alone' または ADMIN COMMAND 'hotstandby switch secondary'	SECONDARY ALONE	未変更
SECONDARY ACTIVE	HotStandby タイムアウト (自動) 注: HSB タイムアウトは、2 次サーバーがダウンしたか、または 1 次および 2 次の間接続が切れた場合に自動的に起きます。	SECONDARY ALONE	(適用外)
SECONDARY ACTIVE	1 次サーバーで ADMIN COMMAND 'hotstandby switch secondary' または 2 次サーバーで ADMIN COMMAND 'hotstandby switch primary'	PRIMARY ACTIVE	未変更
SECONDARY ACTIVE	2 次サーバーで ADMIN COMMAND 'hotstandby set primary alone'	PRIMARY ALONE	未変更
SECONDARY ACTIVE	2 次または 1 次 サーバーで ADMIN COMMAND 'hotstandby disconnect'	SECONDARY ALONE	SECONDARY ALONE

表 31. サーバー状態遷移表 (続き)

サーバー状態	この条件が起きた場合、またはこの HSB コマンドが発行された場合...	サーバーは以下の状態になる...	コマンドが正常でない場合は、以下の状態である...
SECONDARY ALONE	2 次または1 次 サーバーで ADMIN COMMAND 'hotstandby connect'	SECONDARY ACTIVE	未変更
SECONDARY ALONE	2 次サーバーで ADMIN COMMAND 'hotstandby set standalone'	STANDALONE	未変更
SECONDARY ALONE	2 次サーバーで ADMIN COMMAND 'hotstandby set primary alone' または ADMIN COMMAND 'hotstandby switch primary'	PRIMARY ALONE	未変更

付録 E. HotStandby システム・イベント

この付録では、HSB 固有のイベントのみ扱います。他のタイプのイベントについては、「solidDB SQL ガイド」など、他のマニュアルを参照してください。

HotStandby 操作はそれぞれがイベントを生成します。これらのイベントをモニターするには、ウォッチドッグ・アプリケーションなどのアプリケーションを使用することができます。

イベントは、特定のアクションがサーバーで起きたことをシグナル通知する名前付きオブジェクトです。ストアード・プロシージャの特殊ステートメントがイベントの受信に必要です。HotStandby イベントは、solidDB によって作成されサポートされる他のイベントと異なる点はまったくありません。それらのイベントは、ストアード・プロシージャのイベントを受信するように登録済みのユーザーに送信されます。イベントの通知、登録、および待機の詳細については、「solidDB SQL ガイド」の『ストアード・プロシージャ、イベント、トリガー、およびシーケンス』と、やはり「solidDB SQL ガイド」にある『solidDB SQL 構文』を参照してください。

HotStandby に現在使用可能なイベントを以下の表にリストしてあります。ほとんどのイベントが 5 つのパラメーターを含みますが、そのすべてのパラメーターが必ずしも使用されるわけではない点に注意してください。

表 32. HotStandby イベント

HSB イベント	イベント・パラメーター	イベントの原因
SYS_EVENT_HSBCONNECTSTATUS	ENAME WVARCHAR, POSTSRVTIME TIMESTAMP, UID INTEGER, NUMDATAINFO INTEGER, TEXTDATA WVARCHAR TEXTDATA の場合、可能な有効値は以下のとおりです。 TEXTDATA = { CONNECTED CONNECTING CATCHUP BROKEN}	1 次および 2 次サーバー間の接続状況の変更

表 32. HotStandby イベント (続き)

HSB イベント	イベント・パラメーター	イベントの原因
SYS_EVENT_HSBSTATESWITCH	ENAME WVARCHAR, POSTSRVTIME TIMESTAMP, UID INTEGER, NUMDATAINFO INTEGER, TEXTDATA WVARCHAR TEXTDATA の場合、可能な有効値は以下のとおりです。 TEXTDATA = { PRIMARY ACTIVE PRIMARY ALONE PRIMARY UNCERTAIN SECONDARY ACTIVE SECONDARY ALONE STANDALONE }	それぞれの状態切り替えで状態切り替えイベントが送信される。
SYS_EVENT_NETCOPYEND	ENAME WVARCHAR, POSTSRVTIME TIMESTAMP, UID INTEGER, NUMDATAINFO INTEGER, TEXTDATA WVARCHAR どのパラメーターも使用されません。	HotStandby NETCOPY 操作が終了した。 このイベントは、ユーザーがリンク・ライブラリー・アクセスを使用している場合にのみユーザーによってキャッチされます。
SYS_EVENT_NETCOPYREQ	ENAME WVARCHAR, POSTSRVTIME TIMESTAMP, UID INTEGER, NUMDATAINFO INTEGER, TEXTDATA WVARCHAR どのパラメーターも使用されません。	HotStandby NETCOPY が要求された。 ユーザー・アプリケーションのコールバック関数がゼロ以外を返した場合、ネットコピー (netcopy) は実行されません。 このイベントは、ユーザーがリンク・ライブラリー・アクセスを使用している場合にのみユーザーによってキャッチされます。

付録 F. Watchdog サンプル

このセクションでは、solidDB インストール済み環境に組み込まれているサンプルで使用可能な Watchdog サンプル・アプリケーションについて説明します。

ウォッチドッグは 1 次サーバーと 2 次サーバーをモニターおよび制御する別個のプログラムです。ウォッチドッグは、両方のホット・スタンバイ・サーバーをモニターし、必要なときはそれらの状態を切り替えます。これにより、データベース管理者によるサーバーのモニター作業の必要性が軽減されます。

solidDB には、必要に合わせてカスタム・ウォッチドッグを構築するための基礎として使用できる、サンプルのウォッチドッグが用意されています。このサンプル・アプリケーションは、Watchdog と呼ばれます。プログラミングを開始する前に、Watchdog サンプルの以下の機能に注意してください。

- Watchdog は、ウォッチドッグの例として使用するためのものです。
- Watchdog は、ポーリングを使用して最新のサーバー状態を把握します。
- Watchdog は、1 スレッドのプログラムです。
- Watchdog は、ODBC を通じて HSB API を使用します。この API インプリメンテーションは、お客様独自のウォッチドッグ・アプリケーションのモデルとして使用できます。
- Watchdog には、ユーザー・インターフェースがありません。

Watchdog を使用する場合は、Watchdog の現行作業ディレクトリーにある solidDB 構成ファイル (solid.ini) の [WatchDog] セクションを構成する必要があります。Watchdog が 1 次サーバーまたは 2 次サーバーと同じディレクトリーで実行されている場合は、1 つの solid.ini ファイルをサーバーと Watchdog とで共用することになります。Watchdog が別のディレクトリーで実行されている場合、Watchdog はそれ自体の solid.ini ファイルを持ちます。

さらに、この付録には、Watchdog に固有の solid.ini 構成パラメーターの説明が含まれています。これらのパラメーターは、solid.ini 構成ファイルの [WatchDog] セクションで設定されます。独自のウォッチドッグ・プログラムを作成する場合は、これらパラメーターをどれも使用する必要はありません。

その他の solid.ini パラメーターについては、「*solidDB* 管理者ガイド」を参照してください。

Watchdog を使用した HotStandby 構成

HotStandby 構成では 1 次サーバー、2 次サーバー、および Watchdog を異なるマシンに常駐させることができ、それらは、170 ページの『システム設計の問題』の例に示すように、異なるオペレーティング・システムおよび API を使用できます。異機種混合構成のインプリメントの詳細については、170 ページの『システム設計の問題』を参照してください。

1 次と 2 次のデータベース間のすべての通信は (障害を起こしたシステムをサービスに復帰させ、1 次と 2 次のデータベースを再同期させることも含め)、例えば TCP/IP など、既存の通信レイヤー内で行われます。HotStandby は、共用ディスクや FTP 転送など、補助ストレージや転送方式を必要としません。

重要: 2 次サーバーが常駐するマシン上で Watchdog を実行している場合は必ず、パラメーター **AutoPrimaryAlone** を no に設定してください。その状況では、**AutoPrimaryAlone** を no に設定することは、非常に重要です。この設定は、1 次サーバーが 2 つになる潜在的なエラーを防止するからです。1 次サーバーは PRIMARY ALONE 状態に置かれる場合があり、サーバー障害時の Watchdog は、2 次サーバーを PRIMARY ALONE 状態に切り替える可能性があります。このエラーは、ユーザーが偶然、旧 2 次サーバーを新規 1 次サーバーになるように設定した場合にも発生することがあります。二重 1 次サーバーの詳細については、38 ページの『ネットワーク分割と二重 1 次サーバー』を参照してください。

Watchdog アプリケーションの動作

Watchdog サンプル・アプリケーションは、1 次サーバーがダウンしたとき、ユーザーに通知します。通常モードでは、Watchdog は 1 次サーバーと 2 次サーバーの両方で、hotstandby status connect コマンドを使用してサーバーの接続状況を検査します。

Watchdog プログラムは、サーバー間で、この検査を一定の間隔で実行します。その間隔時間は、Watchdog の solid.ini 構成ファイル内の **PingInterval** パラメーターで設定します。

Watchdog は、指定された回数だけポーリングを試みても、1 次サーバーか 2 次サーバー、またはその両方のノードから応答を受信しなかった場合、HotStandby システムに問題があるという結論に到達します。その試行回数は、Watchdog 構成ファイル (solid.ini 内の [Watchdog] セクション) の **NumRetry** パラメーターで設定されます。

Watchdog は、1 次サーバーと 2 次サーバーが互いに接続されているかどうかを監視します。1 次サーバーまたは 2 次サーバーが接続成功の状況を Watchdog に返した場合、それは 1 次サーバーと 2 次サーバーがまだ接続されていることを意味します。逆に、エラーが返された場合、1 次サーバーと 2 次サーバーは、もはや接続されていません。

Watchdog 構成ファイル内の **AutoSwitch** パラメーターが YES に設定されている場合、Watchdog は、1 次サーバーに障害が起きたときの自動的なサーバー状態の切り替えにも責任を負います。例えば、1 次サーバーがダウンした場合、Watchdog は 2 次サーバーを新規 1 次サーバーになるように切り替え、PRIMARY ALONE 状態にします。**AutoSwitch** パラメーターが NO に設定されている場合、Watchdog はサーバー状態自体を変更せず、代わりに Watchdog ログにメッセージを書き込み、ユーザーにサーバー状態を切り替えるよう通知します。

モニターを続行するために、Watchdog は障害モードに切り替わります。これは、接続が機能するよう、障害を起こしたサーバーの検査を継続することを意味します。

障害モード

Watchdog サンプル・アプリケーションが HotStandby の 1 次サーバーと 2 次サーバーが接続していることを認識した場合、Watchdog は通常モードのままです。いずれかのサーバーが障害を起こした場合、またはそれらのサーバー間の通信リンクに障害がある場合、Watchdog は何らかのアクションを実行します。そのアクションでサーバーの接続に失敗した場合、Watchdog は障害モードに入ります。

Watchdog は障害モードに入った後、システム管理者が 1 次サーバーまたは 2 次サーバーの問題を修正するのを待ちます。その間に 2 番目の障害が発生した場合、Watchdog はその障害を処理しません。この Watchdog での制限は、意図的なものです。状況によっては、連続した障害により、また、外見上では適切に見える応答さえ、2 つの 1 次サーバー (PRIMARY ALONE または STANDALONE 状態) というエラーを起こす可能性があります。これは特に、ネットワーク内に短時間の障害があっても、データベース・サーバー自体には障害がない場合に当てはまります。2 つの 1 次サーバーを生成する例は、『複数障害用の Watchdog のコーディング』で提供されています。

障害モードの間、Watchdog は 1 次サーバーと 2 次サーバーの両方に対してポーリングを行います。両方のサーバーに接続できる場合は、両方のサーバーに hotstandby state コマンドを送信し、それらのサーバーと通信できるかどうか、およびそれらがどのような状態にあるかを調べます。

Watchdog は、両方のサーバーと通信できる場合、solid.ini のパラメーター **DualSecAutoSwitch** に基づいて、次に何をすべきかを決めます。

DualSecAutoSwitch = Yes で、両方のサーバーが 2 次サーバーである場合、Watchdog はその 2 つの 2 次サーバーのうちの 1 つを新規 1 次サーバーとして自動的に選択し、それを 1 次サーバーに切り替えます。 **DualSecAutoSwitch** = No の場合は、システム管理者が一方のサーバーを 1 次サーバーに切り替える必要があります。 **DualSecAutoSwitch** は、Watchdog が「通常」モードであっても「障害」モードであっても適用されることに注意してください。

複数障害用の Watchdog のコーディング

Watchdog で複数の障害を処理するには、以下の 2 とおりの方法があります。以下のことができます。

- それぞれの障害 (および Watchdog による自動応答) の後、状態を検査するために、手操作 (人) による介入を義務付けることができます。手操作による介入には、サーバーの再始動やネットワーク問題の修正などが必要になることがあります。これは、1 次サーバーが 2 つになる危険を少なくするために Watchdog が使用する手法です。
- 長い時間をかけて、複数の障害を処理できるウォッチドッグ・アプリケーションを作成できます。

この方式では、以下の例に示すように、1 次サーバーが 2 つになる危険がありません。

二重 1 次サーバー

この例では、Server1 が初期の 1 次サーバーで、Server2 が初期の 2 次サーバーです。

1. ネットワーク障害が発生し、Server1 がアクセス不能になります。
2. Watchdog は Server2 を SECONDARY から PRIMARY ALONE に切り替えます。
3. 2 番目のネットワーク障害が発生し、Server2 がアクセス不能になります。
4. 最初のネットワーク障害が修復され、Server1 が再びアクセス可能になります。
5. Watchdog は、Server1 がアクセス可能で Server2 がそうでないことを認識し、Server1 を PRIMARY ALONE に切り替えます。
6. 2 番目のネットワーク障害が修正され、Server2 が再びアクセス可能になります。
7. この時点で、Server1 と Server2 の両方が PRIMARY ALONE 状態になります。

システム設計の問題

HotStandby をどのように構成するか (ローカルか、リモートか、1 つ以上の異なる場所か、インターネット上か、Watchdog プログラムと一緒に) によって、システムの信頼性と効率が影響を受けることがあります。このセクションでは、それらの問題について説明します。

以下の図は、異機種混合システムの 1 つの例を示しており、ここでは、1 次サーバーと 2 次サーバーが使用するハードウェアとオペレーティング・システムさえ異なっています。

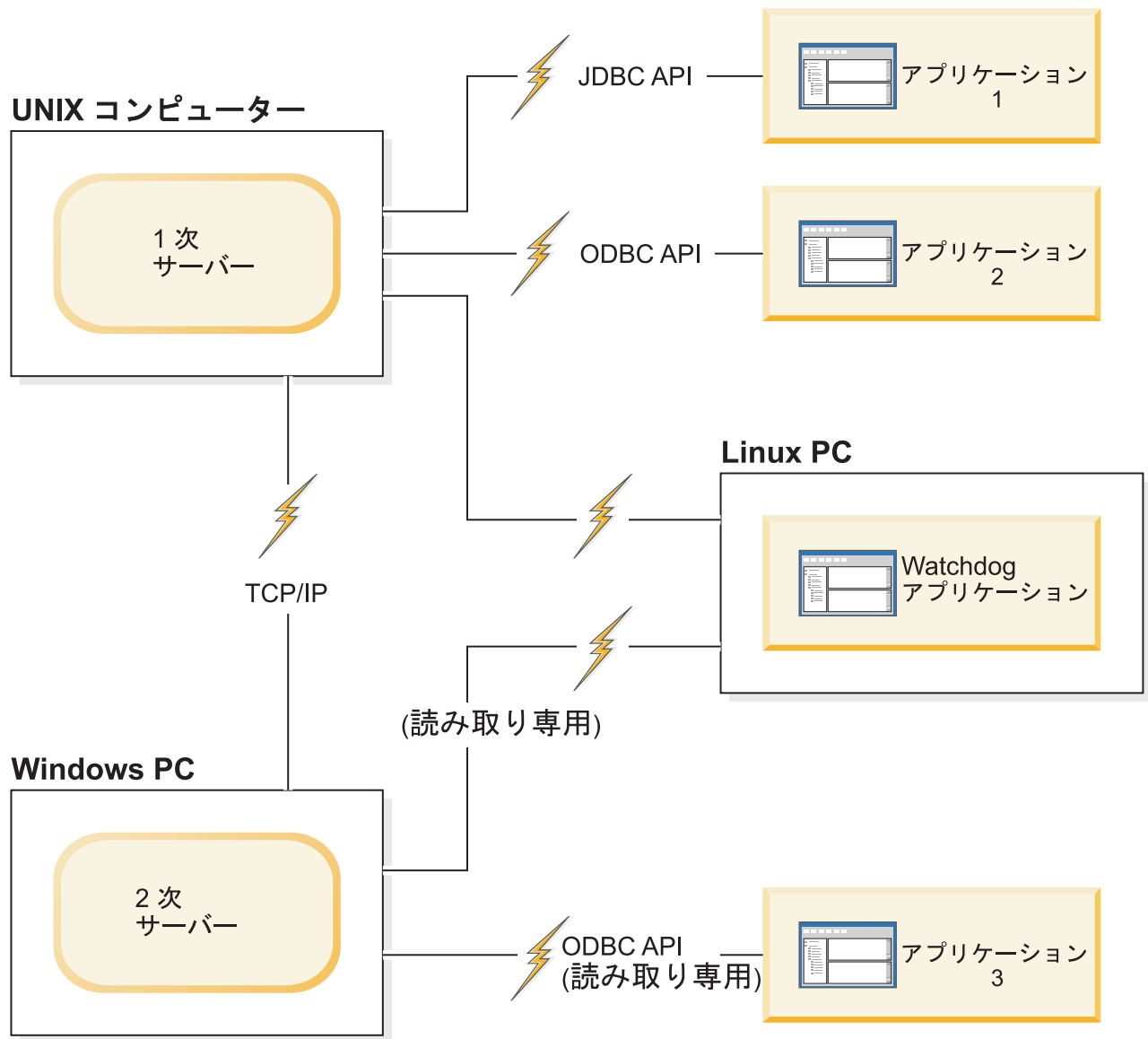


図 16. Watchdog を使用した異機種混合 HotStandby 構成

Watchdog 構成

サーバー状態のモニターの効率と精度を高めるために、Watchdog を HotStandby 構成とは別のコンポーネントとして使用することをお勧めします。

2 台のマシンだけが使用可能で、Watchdog プログラミングを別のマシンで実行できない場合は、Watchdog を 2 次サーバーが常駐するマシンで実行し、1 次サーバーと 2 次サーバーの両方の構成ファイル (solid.ini) でパラメーター **AutoPrimaryAlone** を no に設定してください。このパラメーターを「no」に設定することは、きわめて重要です。なぜなら、これは 1 次サーバーが 2 つになる潜在的なエラーを防止するからです。

注意:

両方のサーバーが書き込みを許可する状態 (PRIMARY ALONE または STANDALONE) にあり、両方のサーバーのデータベースが独立して更新された場合、2 つのデータベースを再同期させることができなくなります。両方のサーバーが同時に PRIMARY ALONE 状態または STANDALONE 状態に置かれることを、Watchdog が許可しないようにしてください。38 ページの『ネットワーク分割と二重 1 次サーバー』を参照してください。

1 次サーバーが失敗した場合、Watchdog は 2 次サーバーを切り替えて、新規 1 次サーバーにすることができます。

Watchdog を 2 次サーバーと同じマシンに配置することには、いくつかの欠点があります。それらの欠点は、以下のとおりです。

- この構成では、Watchdog と 1 次サーバーの間の通信リンクがダウンしただけで、1 次サーバーと 2 次サーバーの間で誤った切り替えが起きる可能性があります。
- 通信リンクは「Single Point of Failure」になります。つまり、単一の障害でシステム全体が使用不可になる可能性があります。(ほとんどの HotStandby 構成では、少なくとも 2 つの障害が存在しなければ、システム全体は使用不可になりません。)
- ネットワーク障害が存在し、2 次サーバー・マシンが 1 次サーバー・マシンと通信できない場合でも、ユーザーとアプリケーションは 1 次サーバーに引き続きアクセスでき、理論的には 1 次サーバーを使用して操作を続行できます。しかし、1 次サーバーはトランザクションの受け入れを停止します。ウォッチドッグが 1 次サーバーに、例えば、PRIMARY ALONE 状態に切り替えるなどの方法で操作を続行するよう通知できないからです。

サンプルの Watchdog アプリケーションの使用 このタスクについて

初期には、両方のサーバーが始動して接続された後に Watchdog を開始してください。

Watchdog を開始するには、Watchdog の現行作業ディレクトリーに移動し、プロンプトで以下のコマンドを発行します。

```
watchdog
```

solid.ini ファイルで (1 次サーバーと 2 次サーバーとしてサービスを提供できる) connect1 サーバーと connect2 サーバーのユーザー名とパスワードを指定しなかった場合、Watchdog はそれらを入力するためのプロンプトを出します。

タスクの結果

開始した後、Watchdog は両方のサーバーに ping を実行し、どちらが 1 次サーバーであるかを検査します。Watchdog は、再試行回数を超えた後にサーバー障害を検出した場合以外、通常モードのままです。Watchdog が最後の再試行をサーバーへ送信した後に障害が発生した場合、Watchdog は障害モードに切り替わります。1 次サーバーと 2 次サーバーの両方が始動して再接続されると、Watchdog は通常モードに切り替わります。

障害の状態と Watchdog のアクション

このセクションでは、よくある具体的な障害のシナリオで、標準的なウォッチドッグ・プログラムがどのように機能するかについて説明します。

これらのシナリオは、どちらかのサーバーの障害か、1 次サーバーと 2 次サーバーの間の通信リンクの切断、あるいはサーバーの 1 つとウォッチドッグとの間の通信リンクの切断というコンテキストでのシナリオです。

これらのコマンドは、人間の管理者とソフトウェア・プログラムのどちらが発行してもかまいませんが、分かりやすくするために、コマンドが Watchdog サンプルによって発行されることを想定しています。

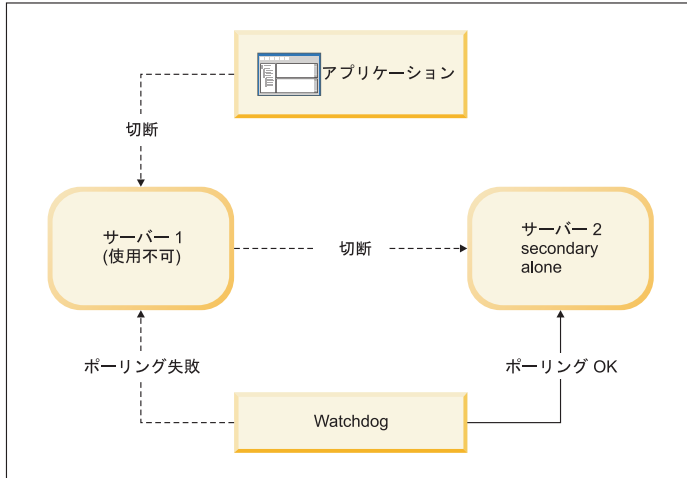
1 次サーバーがダウンした場合 シナリオ

1 次サーバーへのすべての接続が切断されました。

修復方法

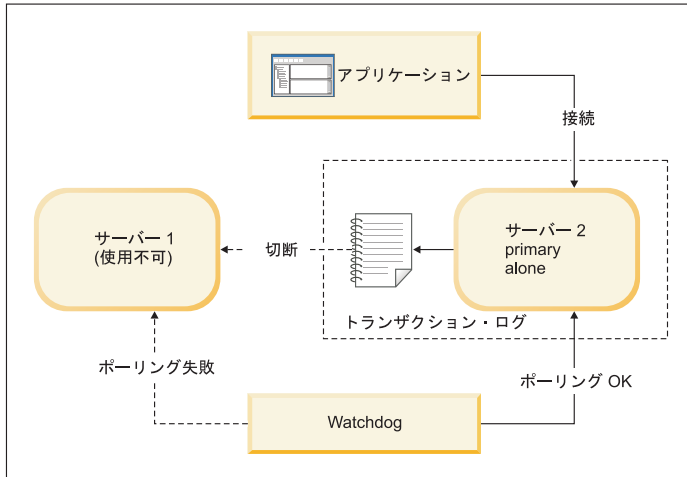
1 次サーバーがダウンしたときは、2 次サーバーを切り替えて新規 1 次サーバーとし、新規 1 次サーバーを PRIMARY ALONE 状態に設定します。後で、旧 1 次サーバーを新規 2 次サーバーにすることができます。

シナリオ



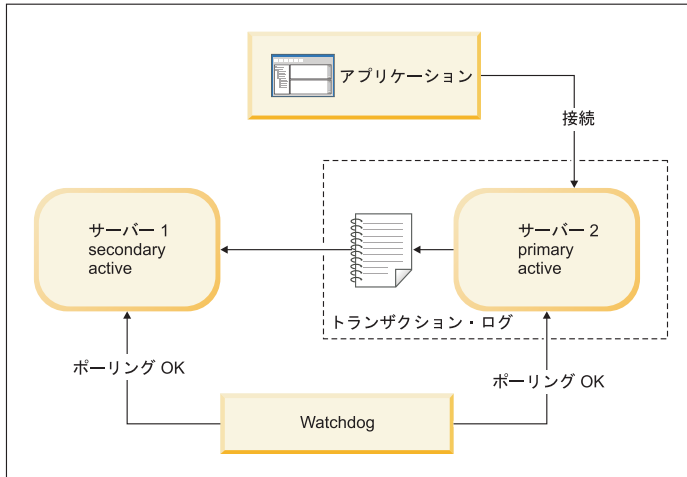
フェイルオーバー/修復方法

1



解決方法

2



3

図 17. 1 次サーバーがダウンしたシナリオと修復方法

1. Watchdog がサーバー 2 に指示します。

HSB SET PRIMARY ALONE

アプリケーションがサーバー 1 からサーバー 2 に切り替えます。

HSB SET STANDALONE

- サーバー 1 が修正された後に、サーバー 1 が 2 次サーバーとして復帰します。

Watchdog がサーバー 2 に指示します。HSB COPY または NETCOPY HSB CONNECT

- トランザクション・ログが満杯である場合は、PRIMARY ALONE サーバーを STANDALONE に切り替えなければならないことがあります。その場合は、サーバーを再接続する前に HSB COPY または HSB NETCOPY を実行する必要があります。トランザクション・ログが満杯でない場合は、COPY/NETCOPY コマンドをスキップする必要があります。

症状

アプリケーションが 1 次サーバーに接続できません。また、ウォッチドッグによるポーリングが 1 次サーバーで失敗します。2 次サーバーの HSB 状態は SECONDARY ALONE です。

1 次サーバーがダウンしたときのリカバリーの方法 このタスクについて

「HotStandby」(2 次サーバー) が 1 次サーバーに置き換わることができるようにするには、以下の手順を実行します。

- 以下のコマンドを使用して、新規 1 次サーバーを PRIMARY ALONE 状態に設定します。

```
ADMIN COMMAND 'hotstandby set primary alone';
```

- アプリケーションを新規 1 次サーバーに再接続します。
- アプリケーションの使用を開始します。
- 旧 1 次サーバーを修正し、新規 2 次サーバーとして始動します。
- 必要であれば、以下のコマンドを使用して、データベースを新規 1 次サーバーから新規 2 次サーバーにコピーします。

```
ADMIN COMMAND 'hotstandby netcopy';
```

詳細については、87 ページの『1 次サーバーと 2 次サーバーの同期』を参照してください。

- 以下のコマンドを使用して、新規 1 次サーバーを新規 2 次サーバーに再接続します。

```
ADMIN COMMAND 'hotstandby connect';
```

2 次サーバーがダウンした場合 シナリオ

2 次サーバーへのすべての接続が切断されました。この原因としては、2 次サーバー内の問題またはネットワーク内の障害により、1 次サーバーと Watchdog がどちらも 2 次サーバーと通信できなくなったことが考えられます。このセクションでは、2 次サーバーに障害があるとしていますが、実際には、2 次サーバーかネット

ワークに問題があることも考えられます。

修復方法

標準的な修復方法は、1 次サーバーを PRIMARY ALONE 状態に切り替えることです。2 次サーバーが再び稼働したら、1 次サーバーとの同期をとります。

2 次サーバーとの接続に問題が見つかったと、1 次サーバーは以下のことを行います。

1. オープン・トランザクションを一時的に中断し、トランザクションのコミットもロールバックも行いません (1 次サーバーは、クライアントにエラー・メッセージ、または「成功」メッセージを送信しません)。しかも、
2. 自身の状態を自動的に PRIMARY ACTIVE から PRIMARY UNCERTAIN に切り替えます。

一般に、ウォッチドッグは、2 次サーバーが使用不可であることを確認した後、1 次サーバーを PRIMARY UNCERTAIN から PRIMARY ALONE に切り替えます。1 次サーバーが PRIMARY ALONE 状態に切り替えられた後、1 次サーバーはトランザクションの受け入れを続行でき、トランザクションを 2 次サーバーへ送信するために保存します。その後、2 次サーバーが復帰した時点で、2 次サーバーにトランザクション・ログを送信し、2 次サーバーが 1 次サーバーに「キャッチアップ」することができるようになります。

1 次サーバーは、PRIMARY ALONE 状態に設定された後、オープン・トランザクションをコミットします。2 次サーバーがまだコミットしていないトランザクションを 1 次サーバーがコミットする可能性を回避するために、トランザクションは、あたかも 2 次サーバーへ送信されなかったかのように、トランザクション・ログ内に保持されます。2 次サーバーが復帰してキャッチアップを開始すると、1 次サーバーは上記のトランザクション・ログを送信し、2 次サーバーは各トランザクションを検査します。トランザクションが重複している場合 (つまり、2 次サーバーが、障害を起こす前にそのトランザクションを既にコミットしてある場合)、重複するトランザクションは 2 次サーバー上で再実行されません。

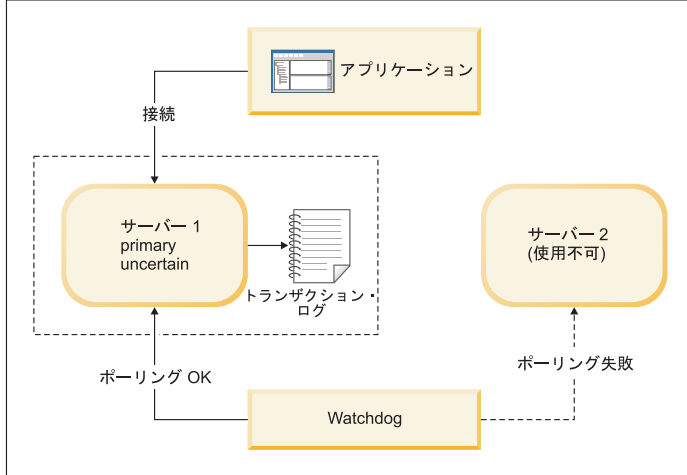
ウォッチドッグまたはシステム管理者は、1 次サーバーを PRIMARY ALONE 状態にするか、それとも代替アクションを選択するかを慎重に選択する必要があります。1 次サーバーを PRIMARY ALONE 状態に切り替える以外のアクションをウォッチドッグまたはシステム管理者が選択する場合は、2 次サーバーおよび 1 次サーバーが同じデータを持っていない可能性 (つまり、両方のサーバーがトランザクションのロールバックを済ませていない可能性) を考慮に入れる必要があります。障害を起こした 2 次サーバーが実際にデータをコミットし、コミット後、1 次サーバーへ確認を送信する前に破損したために、1 次サーバーがコミットしていないことも考えられます。その状態では、2 次サーバーは実際には 1 次サーバーより遅れているのではなく、1 次サーバーの「先」にいる可能性があります。

また、いつもそうですが、ウォッチドッグまたは管理者は、両方のサーバーが同時に PRIMARY ALONE 状態に入らないように注意する必要があります。

下の図は、3 つのフレームに分かれています。最初のフレームは、1 次サーバーとウォッチドッグが 2 次サーバーとの接続を失ったシナリオを示しています。その次のフレームは、問題を完全に解決できるまでの間、システムを機能させておくため

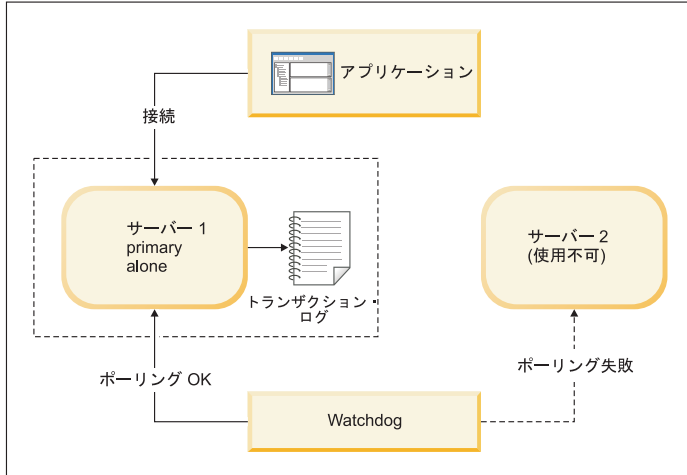
の対応の方法を示しています。3番目のフレームは、問題が解決した後、つまり、壊れたサーバーが修正された後か通信が復元された後の最終状態を示しています。

シナリオ



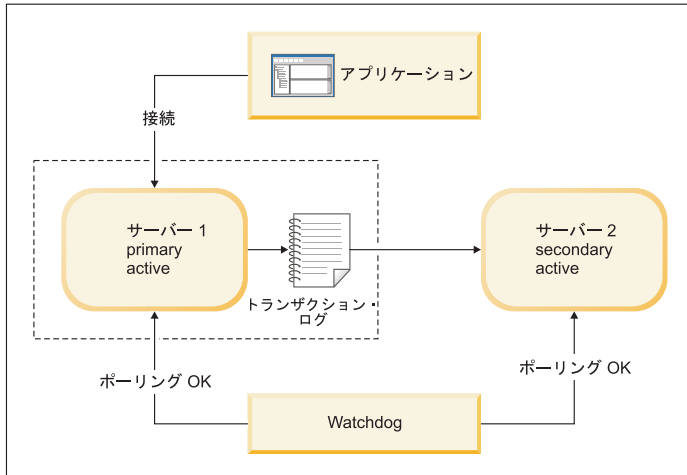
フェイルオーバー/修復方法

1



解決方法

2



3

図 18. 2 次サーバーがダウンしたシナリオと修復方法

1. Watchdog がサーバー 1 に指示します。

HSB SET PRIMARY ALONE

HSB SET STANDALONE

2. サーバー 2 が復帰した後に、Watchdog がサーバー 1 に指示します。

HSB COPY または

NETCOPY HSB CONNECT

3. トランザクション・ログが満杯である場合は、PRIMARY ALONE サーバーを STANDALONE に切り替えなければならないことがあります。その場合は、サーバーを再接続する前に HSB COPY または HSB NETCOPY を実行する必要があります。トランザクション・ログが満杯でない場合は、COPY/NETCOPY コマンドをスキップする必要があります。

症状

ウォッチドッグのポーリングが 2 次サーバーで失敗します。1 次サーバーの状態は、PRIMARY ALONE か PRIMARY UNCERTAIN です。

2 次サーバーがダウンしたときのリカバリーの方法

このタスクについて

1 次サーバーが 2 次サーバーから独立して作動し、トランザクションの受信を続行できるようにするには、以下の手順を実行します。

1. 1 次サーバーが PRIMARY UNCERTAIN 状態にある場合は、以下のコマンドを使用して、1 次サーバーを PRIMARY ALONE に設定します。

```
ADMIN COMMAND 'hotstandby set primary alone';
```

2. 2 次サーバーが修復されて再始動されるか、2 次サーバーのネットワーク接続が再確立された後 (またはその両方の後)、以下のコマンドを使用して 1 次サーバーの状態を検査します。

```
ADMIN COMMAND 'hotstandby state';
```

3. 1 次サーバーの状態が PRIMARY ALONE の場合は、以下のコマンドを使用して、1 次サーバーを 2 次サーバーに再接続します。

```
ADMIN COMMAND 'hotstandby connect';
```

4. 以前に 1 次サーバーの状態を STANDALONE に変更した場合は、以下のようになります。

- a. 以下のコマンドを使用して、データベースを新規 1 次サーバーから新規 2 次サーバーにコピーします。

```
ADMIN COMMAND 'hotstandby netcopy';
```

- b. 詳細については、87 ページの『1 次サーバーと 2 次サーバーの同期』を参照してください。

5. 以下のコマンドを使用して、1 次サーバーを 2 次サーバーに再接続します。

```
ADMIN COMMAND 'hotstandby connect';
```

2 次サーバーがダウンした場合の追加シナリオ

アプリケーションが 1 次サーバーからエラー・メッセージ 10047 または 14537 を受け取った場合は、以下のようになります。

- 2 次サーバーの状態が新規 1 次サーバーに切り替えられているかどうかを検査するために、2 次サーバーに接続を試みます。

- 2次サーバーの状態が1次サーバーのいずれかの状態 (PRIMARY ACTIVE または PRIMARY ALONE) でない場合は、173ページの『1次サーバーがダウンした場合』のシナリオを参照してください。

Watchdog がダウンした場合

このセクションでは、Watchdog に障害が起きた場合にどうなるかを説明します。

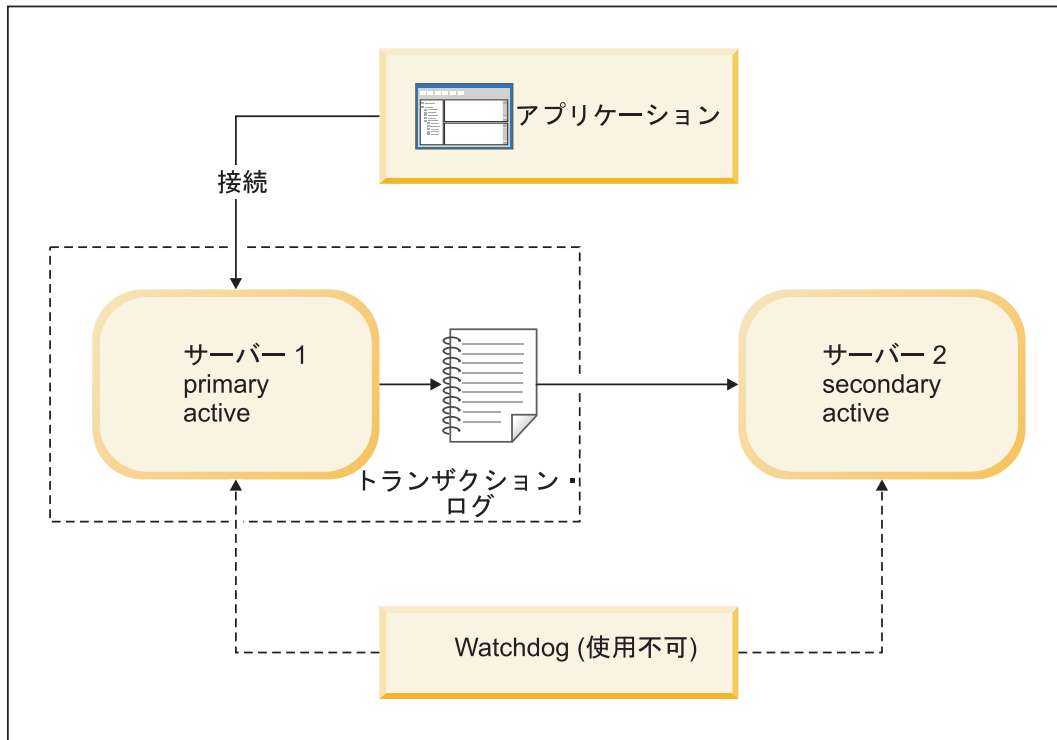
シナリオ

Watchdog へのすべての接続が切断されました。

修復方法

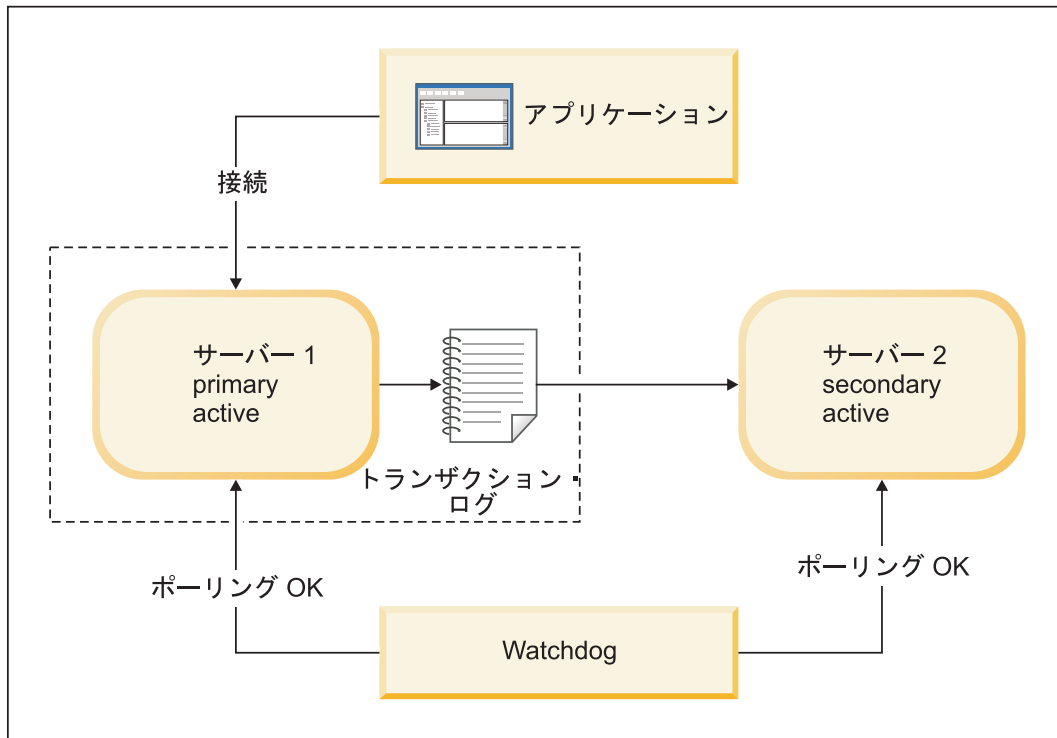
手操作による介入が必要です。Watchdog が起動した時点で、必ず1次サーバーと2次サーバーを検査して、それらの状態を確認してください。

シナリオ



解決方法

1



1. Watchdog を復帰させるか、ネットワークを修正します。

図 19. Watchdog がダウンしたシナリオと修復方法

症状

Watchdog プロセスがダウンするか、Watchdog から両方のサーバーへのネットワーク接続が使用不可になります。

追加シナリオ

各サーバーが状態を変更し、1 つのサーバーがもはや機能しなくなった場合は、このセクションにある該当するシナリオの説明を参照してください。

ウォッチドッグがダウンした場合のリカバリーの方法 このタスクについて

Watchdog へのすべての接続が切断されたシナリオからリカバリーを行うには、以下の手順を実行します。

1. 1 次サーバーと 2 次サーバーが通常操作を続行できるようにします。
2. Watchdog が起動した後、以下のコマンドで Watchdog に各サーバーの状態を検査させます。

```
ADMIN COMMAND 'hotstandby state';
```

1 次サーバーと 2 次サーバーの間の通信リンクがダウンした場合 シナリオ

1 次サーバーと 2 次サーバーの間の接続が切断されました。

1 次サーバーは、自身を PRIMARY UNCERTAIN 状態に切り替えます。
(AutoPrimaryAlone が Yes に設定されている場合、サーバーは自身を PRIMARY ALONE 状態に切り替えます。)

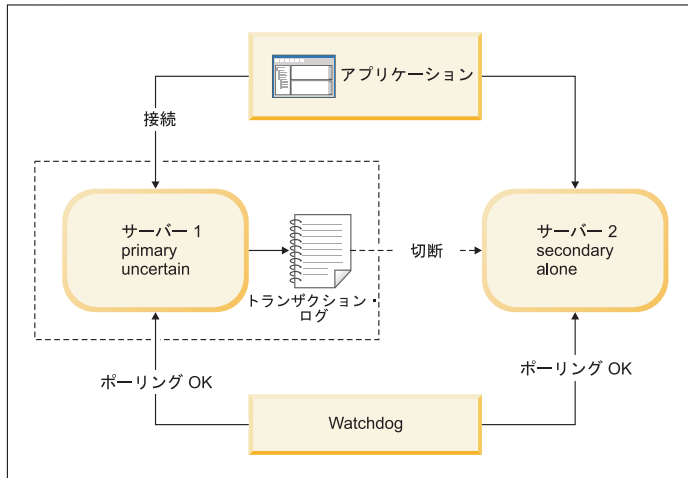
注: 1 次サーバーが 2 次サーバーにコミット・メッセージを送信し、2 次サーバーの障害を検出した場合、1 次サーバーがどのように進行するかは Watchdog または管理者に依存します。その理由は、2 次サーバーが障害を起こす前にトランザクションが 2 次サーバー内でコミットされたかロールバックされたかを、1 次サーバーは検出できないからです。

1 次サーバーは Watchdog または管理者からコマンドを受け取るまで、トランザクションを受け入れなくなります。この段階で、1 次サーバーに操作を続行させるために、Watchdog または管理者は 1 次サーバーを PRIMARY ALONE 状態に設定できます。

修復方法

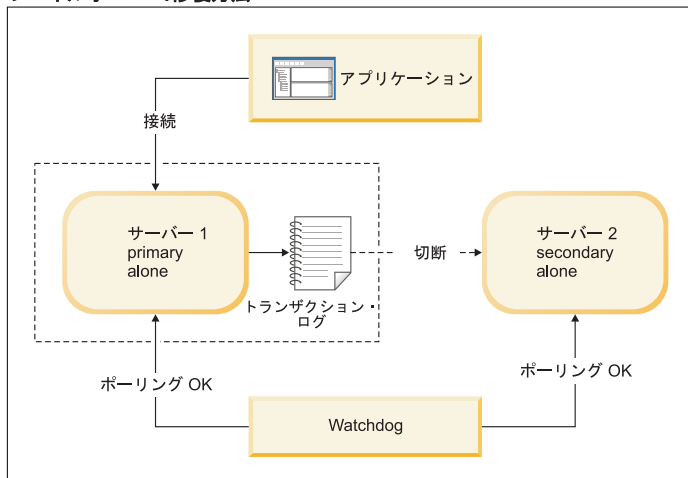
1 次サーバーは、2 次サーバーへのリンクがダウンした場合でも、操作を続行できます。1 次サーバーがまだ PRIMARY ALONE 状態でない場合は、1 次サーバーを PRIMARY ALONE 状態に切り替えてください。1 次サーバーと 2 次サーバーの間のリンクが復元された後、データベース同士の同期をとります。

シナリオ



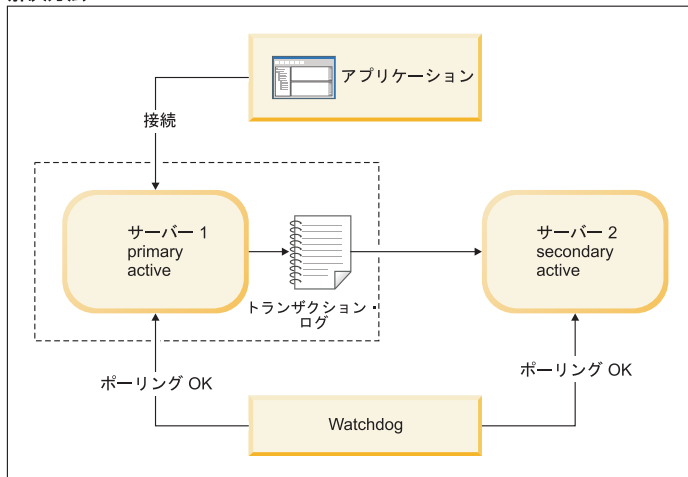
フェイルオーバー/修復方法

1



解決方法

2



3

図 20. 1 次サーバーと 2 次サーバーの間のリンクが切断されたシナリオと修復方法

1. Watchdog がサーバー 1 に指示します。

HSB SET PRIMARY ALONE

HSB SET STANDALONE

2. 1 次サーバーと 2 次サーバーの間の接続が修正された後に、Watchdog がサーバー 1 に指示します。

HSB COPY または

NETCOPY HSB CONNECT

3. トランザクション・ログが満杯である場合は、PRIMARY ALONE サーバーを STANDALONE に切り替えなければならないことがあります。その場合は、サーバーを再接続する前に HSB COPY または HSB NETCOPY を実行する必要があります。トランザクション・ログが満杯でない場合は、COPY/NETCOPY コマンドをスキップする必要があります。

症状

- 1 次サーバーは 2 次サーバーと接続しておらず、状態は PRIMARY UNCERTAIN または PRIMARY ALONE です。

1 次サーバーと 2 次サーバーの間の通信リンクがダウンした場合のリカバリーの方法

このタスクについて

- 1 次サーバーと 2 次サーバーの間の接続が切断されたシナリオからリカバリーを行うには、以下の手順を実行します。

1. 1 次サーバーと 2 次サーバーの間のネットワーク接続を修正します。
2. 以下のコマンドを使用して、1 次サーバーの状態を検査します。

```
ADMIN COMMAND 'hotstandby state';
```

3. 1 次サーバーの状態が PRIMARY ALONE の場合は、以下のコマンドを使用して、1 次サーバーを 2 次サーバーに再接続します。

```
ADMIN COMMAND 'hotstandby connect';
```

4. 1 次サーバーの状態が STANDALONE の場合は、以下の手順を実行します。
 - a. データベースを 1 次サーバーから 2 次サーバーにコピーします。詳細については、87 ページの『1 次サーバーと 2 次サーバーの同期』を参照してください。

コマンド ADMIN COMMAND 'hotstandby netcopy'; を使用する前に、2 次サーバーが稼働中であり、netcopy を受信する準備が整っていることを確認してください。また、1 次サーバーの状態を PRIMARY ALONE に設定してあることも確認してください。

- b. 以下のコマンドを使用して、1 次サーバーを 2 次サーバーに再接続します。

```
ADMIN COMMAND 'hotstandby connect';
```

1 次サーバーと 2 次サーバーの間の通信リンクがダウンした場合の追加シナリオ

アプリケーションが 1 次サーバーからエラー・メッセージ 10047 または 14537 を受け取った場合は、以下のようになります。

- 2 次サーバーが新規 1 次サーバーとして切り替えられているかどうかを検査するために、2 次サーバーに接続を試みます。
- 旧 2 次サーバーが新規 1 次サーバーとして切り替えられていない場合は、173 ページの『1 次サーバーがダウンした場合』のシナリオを参照してください。

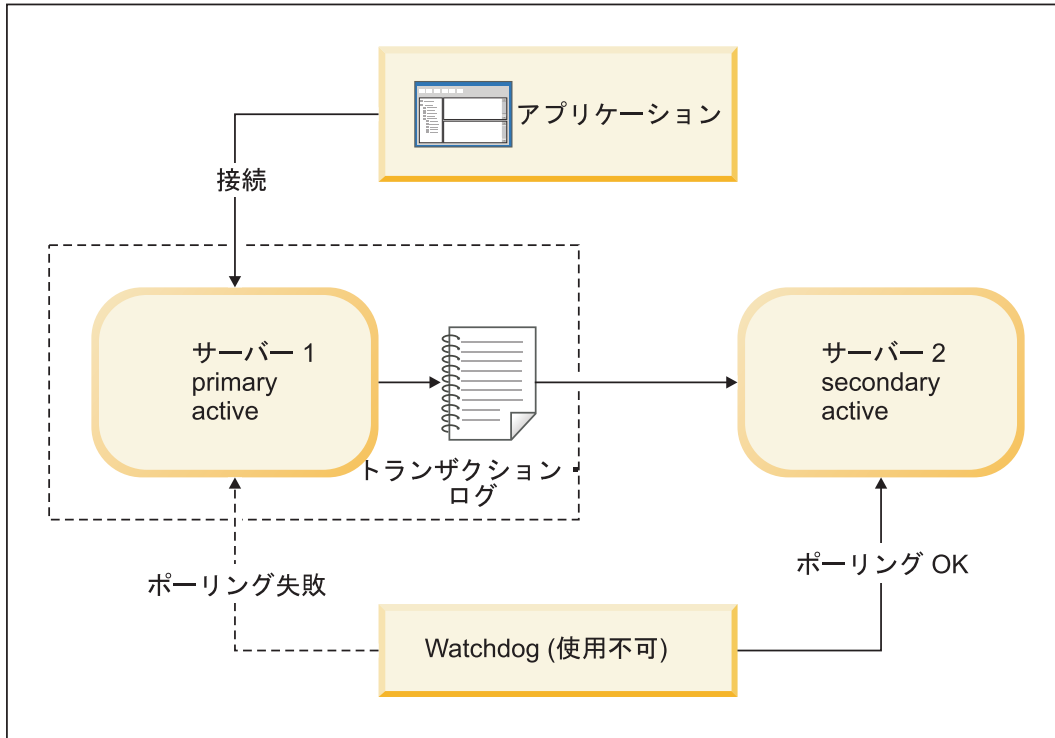
Watchdog と 1 次サーバーの間の通信リンクがダウンした場合 シナリオ

Watchdog と 1 次サーバーの間の接続が切断されました。

修復方法

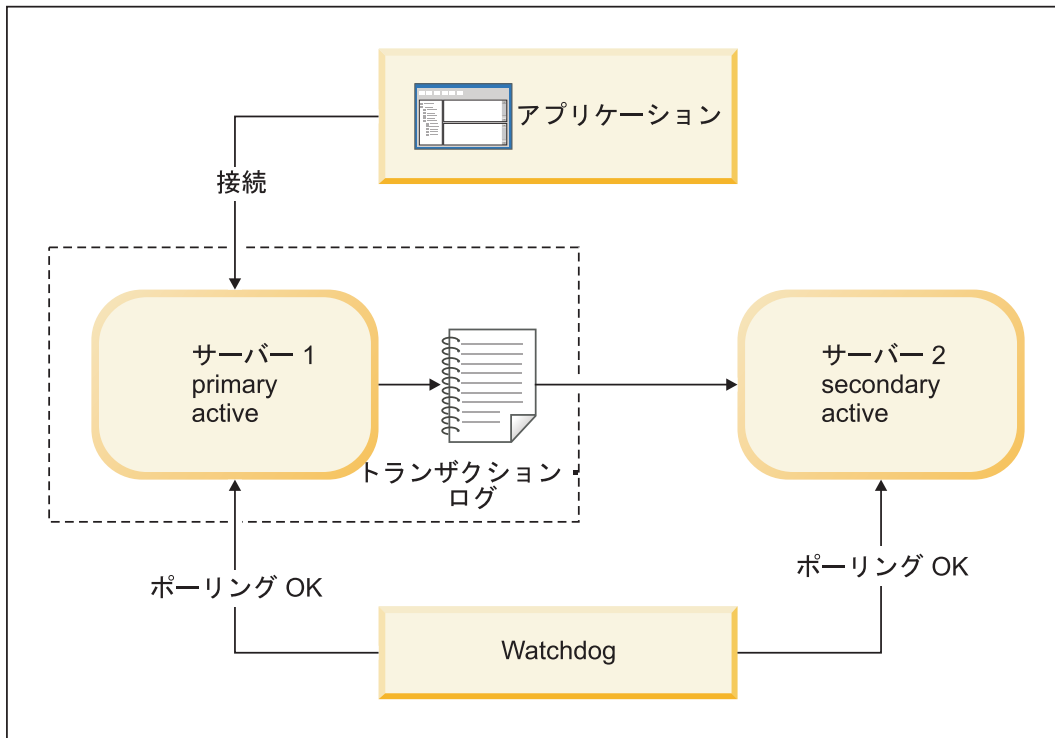
1 次サーバーおよび 2 次サーバーは、Watchdog から 1 次サーバーへのリンクがダウンした場合でも、操作を続行できます。Watchdog から 1 次サーバーへのリンクが修正されたら、必ず 1 次サーバーと 2 次サーバーの状態を検査してください。

シナリオ



解決方法

1



1. サーバー 1 との Watchdog のネットワーク接続を修正します。

図 21. Watchdog と 1 次サーバーの間のリンクが切断されたシナリオと修復方法

症状

Watchdog のポーリングが 1 次サーバーで失敗します。しかし、2 次サーバーの状態は SECONDARY ACTIVE として報告されます。これは、1 次サーバーには問題がない可能性が大きく、単に Watchdog と 1 次サーバーとの接続が失われたことを意味しています。

追加シナリオ

各サーバーの状態が変更され、1 つのサーバーがもはや機能しなくなった場合は、このセクションにある該当するシナリオの説明を参照してください。

Watchdog と 1 次サーバーの間の通信リンクがダウンした場合のリカバリーの方法

このタスクについて

Watchdog と 1 次サーバーの間の接続が切断されたシナリオからリカバリーを行うには、以下の手順を実行します。

1. 1 次サーバーと 2 次サーバーが通常操作を続行できるようにします。
2. Watchdog と 1 次サーバーの間のネットワーク接続を修正します。
3. ネットワークが接続されたら、以下のコマンドで Watchdog に各サーバーの状態を検査させます。

```
ADMIN COMMAND 'hotstandby state';
```

Watchdog と 2 次サーバーの間の通信リンクがダウンした場合

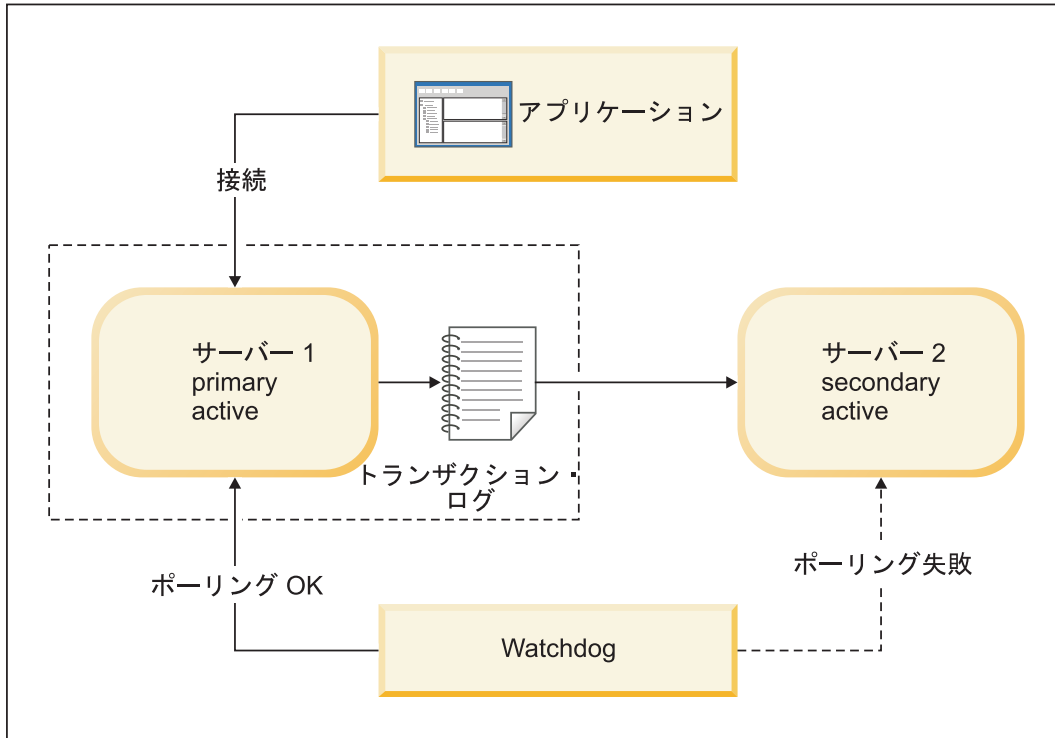
シナリオ

Watchdog と 2 次サーバーの間の接続が切断されました。

修復方法

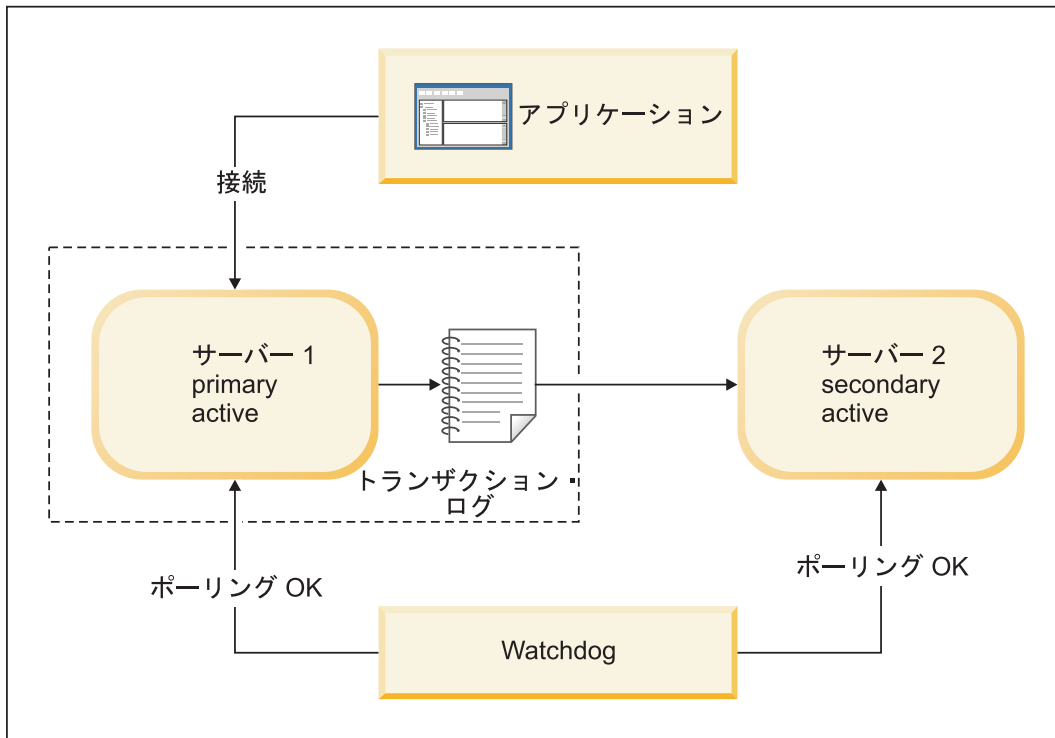
1 次サーバーおよび 2 次サーバーは、Watchdog から 2 次サーバーへのリンクがダウンした場合でも、操作を続行できます。Watchdog から 2 次サーバーへのリンクが修正された時点で、必ず 1 次サーバーと 2 次サーバーを検査して、それらの状態を確認してください。

シナリオ



解決方法

1



1. サーバー 2 との Watchdog のネットワーク接続を修正します。

図 22. Watchdog と 2 次サーバーの間のリンクが切断されたシナリオと修復方法

症状

Watchdog のポーリングが 2 次サーバーで失敗します。

追加シナリオ

各サーバーの状態が変更され、1 つのサーバーがもはや機能しなくなった場合は、このセクションにある該当するシナリオの説明を参照してください。

Watchdog と 2 次サーバーの間の通信リンクがダウンした場合のリカバリーの方法

このタスクについて

Watchdog と 2 次サーバーの間の接続が切断されたシナリオからリカバリーを行うには、以下の手順を実行します。

1. 1 次サーバーと 2 次サーバーが通常操作を続行できるようにします。
2. Watchdog と 2 次サーバーの間のネットワーク接続を修正します。
3. ネットワークが接続されたら、以下のコマンドで Watchdog に各サーバーの状態を検査させます。

```
ADMIN COMMAND 'hotstandby state';
```

Watchdog と 1 次サーバーの間、および 1 次サーバーと 2 次サーバーの間の通信リンクがダウンした場合

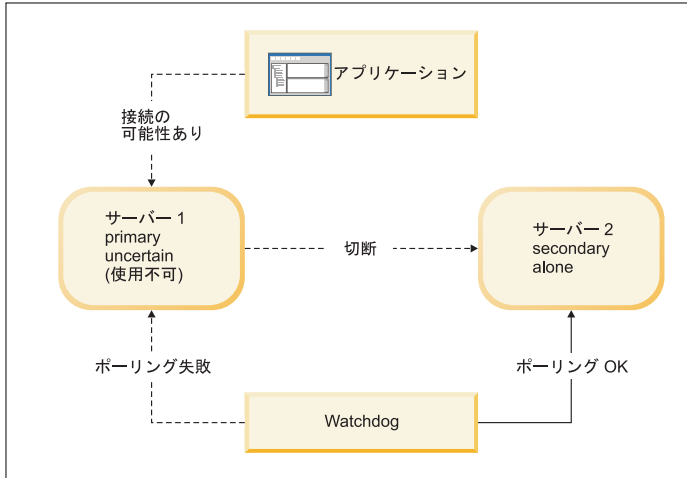
シナリオ

Watchdog と 1 次サーバーの間、および 1 次サーバーと 2 次サーバーの間の接続が切断されました。

修復方法

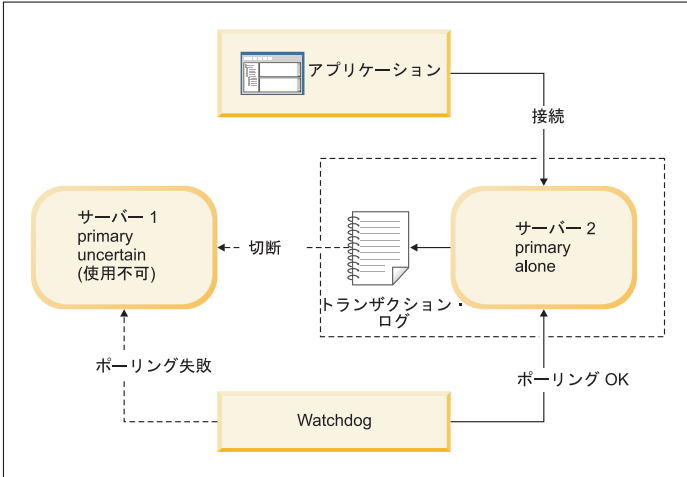
Watchdog が 1 次サーバーのモニターを続行するためには、2 次サーバーを新規 1 次サーバーになるように切り替え、その新規 1 次サーバーを PRIMARY ALONE 状態に設定します。後で、新規 2 次サーバーをセットアップし、1 次サーバーと同期させます。

シナリオ



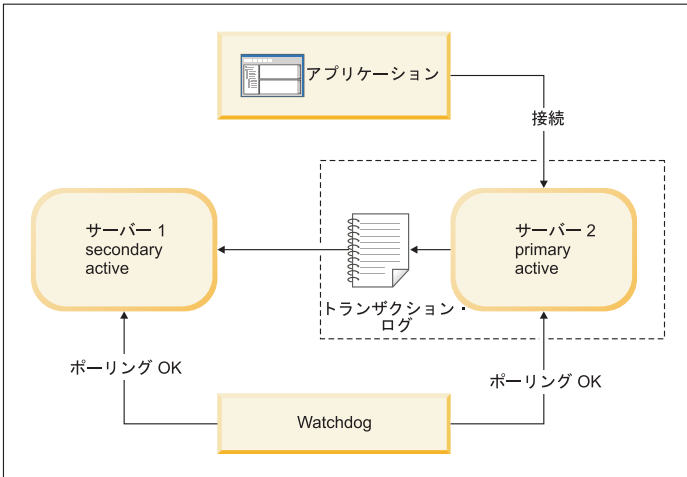
フェイルオーバー/修復方法

1



解決方法

2



3

図 23. Watchdog と 1 次サーバーの間、および 1 次サーバーと 2 次サーバーの間のリンクが切断されたシナリオと修復方法

1. サーバー 1 のロールは Primary Uncertain です。ただし、Watchdog から見たサーバー 1 は使用不可であり、Primary Uncertain ではありません。

Watchdog がサーバー 2 に指示します。

HSB SET PRIMARY ALONE

アプリケーションがサーバー 1 からサーバー 2 に切り替えます。

2. 両方のサーバーが自身を 1 次サーバーと認識しています。プログラムまたは管理者 (手操作による介入) がサーバー 1 を Primary Uncertain から Primary Alone に切り替えると、アクティブな 1 次サーバーが 2 つ存在することになり、両方がデータを更新する可能性があり、その差違は解決不可能です。

ネットワーク接続が修正された後に、Watchdog がサーバー 1 に指示します。

HSB SWITCH SECONDARY

Watchdog がサーバー 2 に指示します。

HSB COPY または

NETCOPY HSB CONNECT

3. トランザクション・ログが満杯である場合は、PRIMARY ALONE サーバーを STANDALONE に切り替えなければならないことがあります。その場合は、サーバーを再接続する前に HSB COPY または HSB NETCOPY を実行する必要があります。トランザクション・ログが満杯でない場合は、COPY/NETCOPY コマンドをスキップする必要があります。

症状

Watchdog のポーリングが 1 次サーバーで失敗します。2 次サーバーおよび 1 次サーバーがお互いの接続を失います。このため、サーバー 2 は SECONDARY ALONE 状態になり、1 次サーバーは (接続できる場合) 状態が PRIMARY UNCERTAIN または PRIMARY ALONE であることを報告します。

このシナリオの始めには、アプリケーションが旧 1 次サーバーに接続している可能性もあることを想定しています。しかし、旧 1 次サーバーが PRIMARY UNCERTAIN 状態にあるので、アプリケーションは更新を行うことができません。サーバー 1 に接続しているアプリケーションが通信リンクを失い、もはや旧 1 次サーバーの存在を認識できなくなっている可能性もあることに注意してください。

Watchdog と 1 次サーバーの間、および 1 次サーバーと 2 次サーバーの間の通信リンクがダウンした場合のリカバリーの方法

Watchdog と 1 次サーバーの間、および 1 次サーバーと 2 次サーバーの間の接続が切断されたシナリオからリカバリーを行うには、ホット・スタンバイ・サーバー (2 次サーバー) を 1 次サーバーに置き換えるために必要な手順を実行します。

このタスクについて

2 次サーバーが 1 次サーバーに置き換わることができるようにするには、以下の手順を実行します。

1. 1 次サーバーが PRIMARY UNCERTAIN 状態にあるか、2 次サーバーおよびアプリケーションから切り離されている場合は、以下のコマンドを使用して、2 次サーバーを PRIMARY ALONE 状態に設定します。

```
ADMIN COMMAND 'hotstandby set primary alone';
```

2. アプリケーションを新規 1 次サーバーに再接続します。
3. 旧 1 次サーバーへのネットワーク接続または切断された接続を修正します。
4. サーバーの状態を検査します。この時点で、両方のサーバーが稼働している必要があります。
5. 新規 1 次サーバーが (例えば、接続を修正中に新規 1 次サーバーのトランザクション・ログが満杯になったために) STANDALONE 状態にある場合は、以下の手順を実行します。

- a. 以下のコマンドを使用して、新規 1 次サーバーを PRIMARY ALONE 状態に設定します。

```
ADMIN COMMAND 'hotstandby set primary alone';
```

- b. データベースを新規 1 次サーバーから新規 2 次サーバーにコピーします。詳細については、87 ページの『1 次サーバーと 2 次サーバーの同期』を参照してください。

6. 新規 1 次サーバーが PRIMARY ALONE 状態にある場合は、以下のようになります。

- a. 以下のコマンドを使用して、旧 1 次サーバーが新規 2 次サーバーになるように切り替えます。

```
ADMIN COMMAND 'hotstandby switch secondary';
```

7. 以下のコマンドを使用して、新規 1 次サーバーを新規 2 次サーバーに再接続します。

```
ADMIN COMMAND 'hotstandby connect';
```

Watchdog と 1 次サーバーの間、および 1 次サーバーと 2 次サーバーの間の通信リンクがダウンした場合の追加シナリオ

アプリケーションが新規 1 次サーバーからエラー・メッセージ 10047 または 14537 を受け取った場合は、以下のようになります。

- 2 次サーバーが新規 1 次サーバーになるように切り替えられたかどうかを検査するために、旧 2 次サーバーに接続を試みます。
- 旧 2 次サーバーが新規 1 次サーバーになるように切り替えられていない場合は、元の 1 次サーバーを PRIMARY ALONE 状態にして、トランザクションを再実行します。

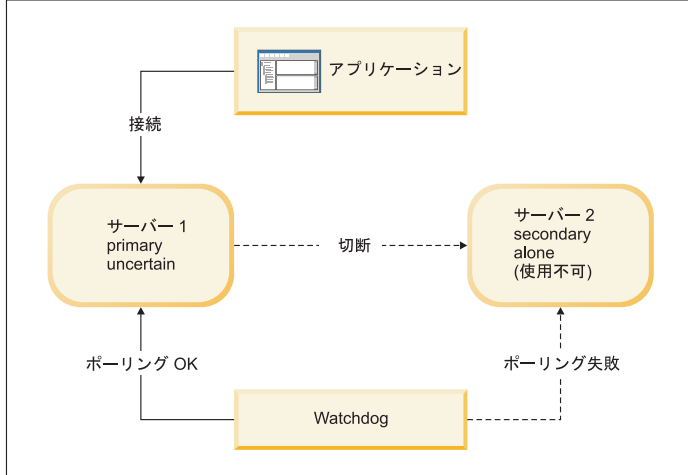
Watchdog と 2 次サーバーの間、および 1 次サーバーと 2 次サーバーの間の通信リンクがダウンした場合シナリオ

Watchdog と 2 次サーバーの間の接続、および 1 次サーバーと 2 次サーバーの間の接続が切断されました。

修復方法

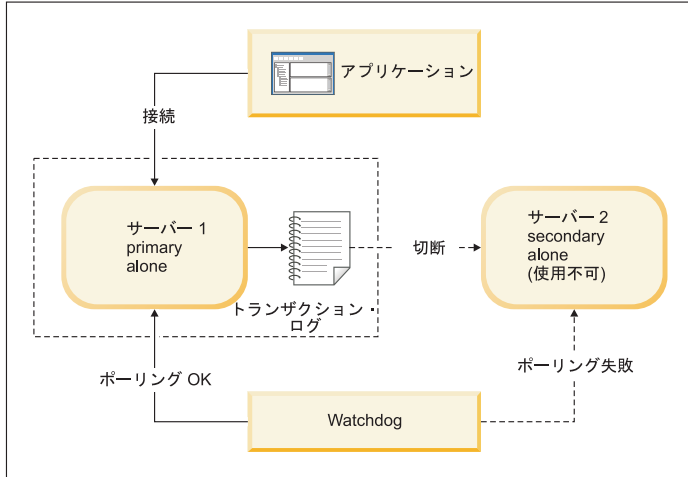
1 次サーバーは、2 次サーバーおよび Watchdog へのリンクがダウンした場合でも、操作を続行できます。1 次サーバーを PRIMARY ALONE 状態に切り替えます (まだ PRIMARY ALONE 状態でない場合)。後で 2 次サーバーが再び稼働したときに、1 次サーバーとの同期をとります。

シナリオ



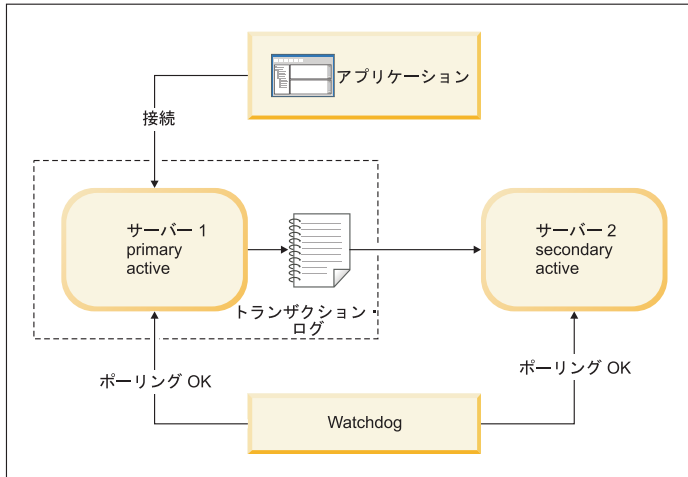
フェイルオーバー/修復方法

1



解決方法

2



3

図 24. Watchdog と 2 次サーバーの間、および 1 次サーバーと 2 次サーバーの間のリンクが切断されたシナリオと修復方法

1. サーバー 2 は自身のロールを Secondary Alone と見なしますが、Watchdog はサーバー 2 を認識できないため、サーバー 2 を使用不可と見なします。

Watchdog がサーバー 1 に指示します。

HSB SET PRIMARY ALONE

2. 2 次サーバーとの接続が修正された後に、Watchdog がサーバー 1 に指示します。

HSB COPY

NETCOPY HSB CONNECT

3. トランザクション・ログが満杯である場合は、PRIMARY ALONE サーバーを STANDALONE に切り替えなければならないことがあります。その場合は、サーバーを再接続する前に HSB COPY または HSB NETCOPY を実行する必要があります。トランザクション・ログが満杯でない場合は、COPY/NETCOPY コマンドをスキップする必要があります。

症状

Watchdog によるポーリングは 2 次サーバーで失敗し、1 次サーバーは 2 次サーバーと接続しておらず、状態が PRIMARY UNCERTAIN または PRIMARY ALONE に切り替えられます。

Watchdog と 2 次サーバーの間、および 1 次サーバーと 2 次サーバーの間の通信リンクがダウンした場合のリカバリーの方法 このタスクについて

Watchdog と 2 次サーバーの間の接続、および 1 次サーバーと 2 次サーバーの間の接続が切断されたシナリオからリカバリーを行うには、以下の手順を実行します。

1. 接続の修正を試みます。
2. 接続が修正された後、コマンド ADMIN COMMAND 'hotstandby state' を使用して 1 次サーバーの状態を検査します。
3. 1 次サーバーの状態が STANDALONE の場合は、以下の手順を実行します。
 - a. 両方のサーバーが稼働していることを確認します。
 - b. 以下のコマンドを使用して、1 次サーバーの状態を PRIMARY ALONE に設定します。

```
ADMIN COMMAND 'hotstandby set primary alone';
```

- c. 以下のコマンドを使用して、データベースを 1 次サーバーから 2 次サーバーにコピーします。

```
ADMIN COMMAND 'hotstandby netcopy';
```

詳細については、87 ページの『1 次サーバーと 2 次サーバーの同期』を参照してください。

4. 以下のコマンドを使用して、1 次サーバーを 2 次サーバーに再接続します。

```
ADMIN COMMAND 'hotstandby connect';
```

Watchdog と 2 次サーバーの間、および 1 次サーバーと 2 次サーバーの間の通信リンクがダウンした場合の追加シナリオ

アプリケーションが 1 次サーバーからエラー・メッセージ 10047 または 14537 を受け取った場合は、以下のようにします。

- 2 次サーバーが 1 次サーバーになるように切り替えられたかどうかを検査するために、2 次サーバーに接続を試みます。
- 2 次サーバーが新規 1 次サーバーとして切り替えられていない場合は、当初の 1 次サーバーを PRIMARY ALONE 状態にして、トランザクションを再実行します。

solid.ini 構成ファイルの Watchdog セクション

Watchdog 用の solid.ini ファイルにある [Watchdog] 構成セクションは、Watchdog 固有のパラメーターを指定するための場所です。

重要: solid.ini ファイルの [Watchdog] セクションのパラメーターは、solidDB によってすべてが事前定義されるわけではありません。どのように Watchdog を書いたか、およびそこで solid.ini ファイルからパラメーター情報を読み取るかどうかに応じて、ここで定義するパラメーターと定義済みのパラメーターを任意に組み合わせで使用できます。パラメーターを無視することもできます。ここに示したパラメーターは、solidDB で提供されるサンプルの C 言語 Watchdog プログラム用のものです。

表 33. Watchdog パラメーター

[Watchdog]	説明	ファクトリー値
AutoSwitch	<p>AutoSwitch パラメーターを yes に設定すると、Watchdog が自動的に以下を行います。</p> <ol style="list-style-type: none"> 1. 2 次サーバーで障害が起きた場合は、Watchdog が 1 次サーバーに、PRIMARY UNCERTAIN 状態を継続するのではなく、PRIMARY ALONE 状態へ切り替えるよう指示します。 2. 1 次サーバーで障害が起きた場合は、Watchdog が自動的に以下のコマンドを送信します。 <pre>'hsb switch primary' 'hsb set primary alone'</pre> <p>その目的は、元の 2 次側を新規の 1 次側にすることです。</p> <p>以下に例を示します。</p> <pre>[Watchdog] AutoSwitch = NO</pre> <p>このパラメーターはオプションです。</p>	Yes
Connect1	<p>[Watchdog] セクションの Connect1 パラメーターを使用すると、Watchdog アプリケーションは、1 次サーバーまたは 2 次サーバーに接続できます。これは、Connect1 サーバーのプロトコルおよびネットワーク・アドレスを定義する必須パラメーターです。</p> <p>以下に例を示します。</p> <pre>connect1 = tcp primarymachine 1313</pre>	なし

表 33. Watchdog パラメーター (続き)

[Watchdog]	説明	ファクトリー値
Connect2	<p>[Watchdog] セクションの Connect2 パラメーターを使用すると、Watchdog アプリケーションは、1 次サーバーまたは 2 次サーバーに接続できます。これは、Connect2 サーバーのプロトコルおよびネットワーク・アドレスを定義する必須パラメーターです。</p> <p>以下に例を示します。</p> <pre>connect2 = tcp secondarymachine 1313</pre>	なし
DualSecAutoSwitch	<p>DualSecAutoSwitch = Yes で、両方のサーバーが 2 次サーバーである場合、Watchdog はその 2 つの 2 次サーバーのうちの 1 つを新規 1 次サーバーとして自動的に選択し、それを 1 次サーバーに切り替えます。DualSecAutoSwitch = No の場合は、システム管理者が一方のサーバーを 1 次サーバーに切り替える必要があります。DualSecAutoSwitch は、Watchdog が「通常」モードであっても「障害」モードであっても適用されることに注意してください。</p>	Yes
NumRetry	<p>[Watchdog] セクションの NumRetry パラメーターを使用すると、Watchdog が 2 次または 1 次サーバーへの接続試行を何回行った後で、接続試行を応答障害またはエラーであると見なすかの回数を指定できます。</p> <p>以下に例を示します。</p> <pre>[Watchdog] NumRetry = 3</pre> <p>再試行は元の試行に加える回数です。再試行回数を 3 に設定した場合、試行の総数は 4 です。再試行はすぐに行われることに注意してください。Watchdog は障害があるときには再試行の間で (PingTimeout などの) 時間間隔を待機しません。</p> <p>このパラメーターはオプションです。</p>	0
Password1 Password2	<p>下の Username1 および Username2 パラメーターの説明を参照してください。</p>	ファクトリー値なし。

表 33. Watchdog パラメーター (続き)

[Watchdog]	説明	ファクトリー値
<p>Pessimistic</p>	<p>このパラメーターを Yes に設定すると、Watchdog の反応を高速化できます。</p> <p>Pessimistic = No の場合、Watchdog はサーバーとの接続を検査しますが、いずれかのサーバーが問題の存在を検出してその状態を (例えば、PRIMARY UNCERTAIN へ) 変更するまで、実際には動作しません (例えば、サーバーの状態を PRIMARY ALONE へ変更しません)。</p> <p>Pessimistic = Yes の場合、Watchdog は、それ自体といずれかのサーバーとの接続が失われるとすぐに動作します。つまり、Watchdog は、残りのサーバーが状態を変更するのを待機しません。これにより反応時間の高速化が可能ですが、例えば、ネットワークの問題に起因する誤報の可能性が高くなることもあります。</p> <p>Pessimistic = Yes の場合、Watchdog は以下のように反応します。1 次サーバーとの接続が失われると、Watchdog は 2 次サーバーを 1 次サーバーに切り替えます。2 次サーバーとの接続が失われると、Watchdog は 1 次サーバーを PRIMARY ALONE に設定します。</p> <p>注意: Pessimistic = Yes に設定すると、余分な切り替えが生じたり、1 次側が二重になることさえ場合によってはあります。このパラメーターは、サーバーに比べ障害が起きそうにもないネットワークでなければ、Yes には設定しないでください。</p> <p>オプションのコマンド行スイッチ「-p」を使用することにより、ベシミスティック動作をオンにすることもできます。</p>	<p>No</p>
<p>PingInterval</p>	<p>[Watchdog] セクションの PingInterval パラメーターを使用すると、通常の Watchdog モードで接続状況情報の照会を行うミリ秒単位の間隔を指定できます。サーバーの障害を検出するために、Watchdog は hotstandby status connect コマンドを、毎回の PingInterval ミリ秒の後で、1 次サーバーと 2 次サーバーの両方に送信します。</p> <p>以下に例を示します。</p> <pre>[Watchdog] PingInterval = 5000</pre> <p>このパラメーターはオプションです。</p> <p>Watchdog 用の PingInterval パラメーターは、サーバー用の PingTimeout パラメーターとは異なることに注意してください。</p> <p>注意: 以前のサンプル Watchdog では、PingInterval をミリ秒単位ではなく秒単位で指定する必要がありました。古い solid.ini ファイルを使用している場合は、それを更新してください。</p>	<p>1000 (1 秒)</p>

表 33. Watchdog パラメーター (続き)

[Watchdog]	説明	ファクトリー値
Username1 Username2	<p>[Watchdog] セクションの Username および Password パラメーターはオプションです。それらは、connect1 サーバーの使用が許可されるユーザー名とパスワードを指定します。</p> <p>以下に例を示します。</p> <pre>[Watchdog] Username1 = Tom Password1 = dr17xy Username2 = Jerry Password2 = M89tvv</pre> <p>セキュリティ上の理由から、これらのパラメーターが solid.ini 構成ファイルに指定されていない場合、Watchdog は開始時にそれらを求めるプロンプトを出します。</p>	ファクトリー値なし。
WatchdogLog	<p>[Watchdog] セクションの WatchdogLog パラメーターを使用すると、Watchdog ログのファイル名を指定できます。Watchdog ログは、現行作業ディレクトリーに作成されます。これは、Watchdog のメッセージを記録して、Watchdog コマンド発行の必要性を管理者に警告するために使用します。</p> <p>以下に例を示します。</p> <pre>[Watchdog] WatchdogLog = Watchdog.log</pre> <p>空白または特定の句読記号など、特殊文字を含まない限り、ファイル名を引用符で囲む必要はありません。</p> <p>このパラメーターはオプションです。</p>	Watchdog.log

以下のパラメーターを使用した場合、

```
[Logging]
DurabilityLevel
```

DurabilityLevel パラメーター値は 1 次サーバーのみに影響します。2 次サーバーのロギング・モードは、[HotStandby] セクションの **2SafeAckPolicy** パラメーターに従います。

索引

日本語, 数字, 英字, 特殊文字の順に配列されています。なお, 濁音と半濁音は清音と同等に扱われています。

[ア行]

- アクセス権限 36
- アクセス・モード
 - RO (読み取り専用) 120
 - RW (読み取り/書き込み) 120
 - RW/Create 120
 - RW/Startup 120
- アプリケーション
 - 基本接続の使用 73
 - 新規 1 次サーバーへの切り替え 75
- アプリケーションから 1 次サーバーへの再接続 73
- インメモリー表 37

[カ行]

- 開始シーケンス 32
- 外部参照エンティティ
 - 構成 124
 - 定義 22
- 基本接続 73
- キャッチアップ 87
- 切り替え
 - 切り替え状況情報の表示 98
 - 接続状況情報の表示 99
- 切り替え状況情報の表示 98
- 切り替えプロセスの検証
 - ADMIN COMMAND 'hotstandby status switch' 84
- 権限
 - アクセス 36
- 現行値 119
- 現行の接続設定の表示 97
- 高可用性コントローラー
 - 開始 30
 - 構成 30, 52
 - サンプル 27
 - 障害が起きた場合の動作 105
 - セットアップ 30
 - 停止 30
 - ロギング 26
 - solidhac.ini 30
- 高可用性制御
 - 原理 81
- 高可用性マネージャー
 - 構成 52
 - スクリーン・ショット 24

- 高可用性マネージャー (続き)
 - 定義 24
- 構成
 - アプリケーションとサーバー間のタイムアウト 45
 - 2 次および 1 次サーバー・ノードの構成 45
- コピー
 - データベース内容 92, 94, 95
 - 手順の検証 91, 95
 - 1 次からローカル 2 次へ 95
- コピー手順の検証 95

[サ行]

- サーバー
 - 接続 97
- サーバー状態
 - 検証 100
 - サーバー状態の切り替え 81
 - OFFLINE 13
 - PRIMARY ACTIVE 13
 - PRIMARY ALONE 13, 84
 - PRIMARY UNCERTAIN 13
 - SECONDARY ACTIVE 13
 - SECONDARY ALONE 13
 - STANDALONE 13
- サーバーのアップグレード
 - コールドおよびホット・マイグレーション 111
 - コールド・マイグレーション 113
 - 準備 112
 - ホット・マイグレーション 113
 - HSB 互換バージョン間のマイグレーション 111
 - HSB 非互換バージョン間のマイグレーション 112
- サーバー名 ネットワーク名 45
- サンプル
 - 高可用性コントローラー 27
 - Watchdog 27
- 障害透過性
 - 接続タイプの選択
 - CONNECTION 60
 - NONE 60
 - SESSION 60
- 状況 97
 - 切り替え状況情報の表示 98
 - 接続状況情報の表示 99
 - 通信情報の表示 99
 - リスト 68, 69, 99
 - HotStandby 97
- 状態
 - サーバー状態の検証 100
 - OFFLINE 157
 - PRIMARY ACTIVE 157

状態 (続き)

PRIMARY ALONE 157
PRIMARY UNCERTAIN 157
STANDALONE 85, 103, 157

状態切り替え

検証 84

状態切り替えの検証 84

状態の切り替え

検証 84

接続

基本 73

接続タイプの選択 59

透過的フェイルオーバー 60

接続切り替え

透過接続 69

接続状況情報の検証

HOTSTANDBY_CONNECTSTATUS 75

接続状況情報の表示 99

接続設定

表示 97

[タ行]

チェックポイント 4

ツール

solidDB データ管理ツール 36

通信情報の表示 99

データ管理ツール 36

データベース

インメモリー表 37

コピー手順の検証 95

内容のコピー 92, 94, 95

データベース内容のコピー 92, 94, 95

デフォルト値 119

透過接続 60

等号

パラメーター値を設定するときの使用 35

どちらのサーバーを 1 次サーバーにするかの選択 102

トランザクション分離レベル

インメモリー表 37

トランザクション・ログ

スペース不足 39

トランザクション・ログのスペース不足 39

[ナ行]

二重 1 次サーバー 38

ネットコピー ネットコピー listen モード 51

ネットコピー listen モード 92

パフォーマンスのチューニング 51

ADMIN COMMAND 'hotstandby netcopy' 92

ネットワーク分割 38

ネットワーク分割と二重 1 次サーバー 38

ネットワーク名 45

ネットワークを介した 1 次側データベースの 2 次側へのコピー 91

[ハ行]

バックアップ 4

バックアップ listen モード

参照: ネットコピー listen モード

パラメーター

高可用性コントローラー

CheckInterval 124

CheckTimeout 124

Connect [LocalDB] 124

Connect [RemoteDB] 124

DBPassword 124

DBUsername 124

EnableAutoNetcopy 124

EnableDBProcessControl 124

EREIP 124

Listen 124

Password 124

PreferredPrimary 124

RequiredConnectFailures 124

RequiredPingFailures 124

StartInAutomaticMode 124

StartScript 124

Username 124

AutoPrimaryAlone 50, 81, 84

AutoSwitch 196

BackupBlockSize 51

BackupDeleteLog 4

CatchupSpeedRate 52

CatchupStepsToSkip 52

CheckInterval 53, 124

CheckpointDeleteLog 4

CheckTimeout 53, 124

Connect 36, 47, 54, 55, 103

Connect [LocalDB] 124

Connect [RemoteDB] 124

Connect1 196

Connect2 196

ConnectTimeout 48, 50

CopyDirectory 50

DBPassword 53, 124

DBUsername 53, 124

DualSecAutoSwitch 196

DurabilityLevel 47

EnableAutoNetcopy 53, 124

EnableDBProcessControl 53, 124

EREIP 55, 124

Header_text 129

HSBEnabled 36, 47, 103

Listen 45, 53

LogEnabled 47

NumRetry 196

Password 53, 124

パラメーター (続き)

- Password1 196
- Password2 196
- Pessimistic 196
- PingInterval 48, 49, 196
- PingTimeout 48, 49
- PreferredPrimary 54, 124
- ReadMostlyLoadPercentAtPrimary 61
- RequiredConnectFailures 53, 124
- RequiredPingFailures 55, 124
- Server1_host 129
- Server1_name 129
- Server1_pass 129
- Server1_port 129
- Server1_user 129
- Server2_host 129
- Server2_name 129
- Server2_pass 129
- Server2_port 129
- Server2_user 129
- StartInAutomaticMode 53, 124
- StartScript 54, 124
- Username 53, 124
- Username1 196
- Username2 196
- WatchdogLog 196
- Window_title 129
- ファクトリー値 119
- 分割
 - ネットワーク 38
- 保管モード 120

[ラ行]

- リカバリー
 - 保守 40
- リカバリーと保守の実行 40
- ロード・バランシング
 - 透過接続 60
 - 動的制御 62
- ロード・バランシング方式
 - PREFERRED_ACCESS=READ_MOSTLY 61
 - PREFERRED_ACCESS=WRITE_MOSTLY 61
- ロギング
 - 参照：高可用性コントローラー

[数字]

- 1 次サーバーと 2 次サーバーの同期 87
- 1SafeMaxDelay (パラメーター) 120
- 2 次サーバー
 - オンラインへの復帰 85
- 2 次サーバー障害時の 1 次サーバーの動作の定義 50
- 2 次サーバーのオンラインへの復帰 85
- 2 次サーバー用の新規データベースの作成 92

2SafeAckPolicy (パラメーター) 120

A

- ADMIN COMMAND 'hotstandby cominfo'
 - 接続設定の表示 97
- ADMIN COMMAND 'hotstandby connect'
 - HotStandby サーバーの接続 97
- ADMIN COMMAND 'hotstandby copy'
 - データベース内容のコピー 95
- ADMIN COMMAND 'hotstandby netcopy'
 - データベース内容のコピー 92, 94
 - 2 次へのコピー 92
- ADMIN COMMAND 'hotstandby set primary alone'
 - PRIMARY ALONE 状態でのサーバーの実行 84
- ADMIN COMMAND 'hotstandby set standalone'
 - HotStandby 操作のシャットオフ 86
- ADMIN COMMAND 'hotstandby state'
 - サーバー状態の検証 100
- ADMIN COMMAND 'hotstandby status connect'
 - 接続状況情報の表示 99
- ADMIN COMMAND 'hotstandby status copy'
 - コピー手順の検証 91, 95
- ADMIN COMMAND 'hotstandby status switch'
 - 切り替えプロセスの検証 84
- ADMIN COMMAND 'hotstandby status'
 - HotStandby 状況の照会 97
- ADMIN COMMAND 'hotstandby switch primary'
 - サーバー状態の切り替え 81
- ADMIN COMMAND 'hotstandby switch secondary'
 - サーバー状態の切り替え 81
- autoconvert 113
- AutoPrimaryAlone (パラメーター) 50, 84, 120, 171
 - および 'hotstandby switch' コマンド 81
- AutoSwitch (パラメーター) 196

B

- BackupBlockSize (パラメーター) 51
- BackupDeleteLog (パラメーター) 4

C

- CatchupSpeedRate (パラメーター) 52, 120
- CatchupStepsToSkip (パラメーター) 52
- CheckInterval (パラメーター) 124
- CheckpointDeleteLog (パラメーター) 4
- CheckTimeout (パラメーター) 124
- CLUSTER 64
- Connect (パラメーター) 36, 47, 103, 120
- Connect [LocalDB] (パラメーター) 124
- Connect [RemoteDB] (パラメーター) 124
- Connect1 (パラメーター) 196
- Connect2 (パラメーター) 196
- ConnectTimeout (パラメーター) 48, 50, 120

CopyDirectory (パラメーター) 50, 120

D

DBPassword (パラメーター) 124

DBUsername (パラメーター) 124

DualSecAutoSwitch (パラメーター) 196

DurabilityLevel (パラメーター) 47

E

EnableAutoNetcopy (パラメーター) 124

EnableDBProcessControl (パラメーター) 124

ERE

参照：外部参照エンティティ

EREIP (パラメーター) 124

G

GUI

参照：高可用性マネージャー

H

HA コントローラー 81

HA マネージャーのパラメーター

Header_text 129

Server1_host 129

Server1_name 129

Server1_pass 129

Server1_port 129

Server1_user 129

Server2_host 129

Server2_name 129

Server2_pass 129

Server2_port 129

Server2_user 129

Window_title 129

HAC の障害のシナリオ

1 次サーバーのデータベースの障害 105

1 次ノードの障害 106

2 次サーバーのデータベースの障害 106

2 次ノードの障害 107

HotStandby リンクの障害 108

HAC パラメーター

HACController 53, 54, 55

HAC を伴う HotStandby

クイック・スタート 31

構成 31

セットアップ 31

Header_text (パラメーター) 129

HotStandby

オフにする 103

管理 35

クイック・スタート 28

HotStandby (続き)

構成 28, 35, 45, 52

セットアップ 28

操作のシャットオフ 86

hotstandby copy 157

hotstandby netcopy 157

HotStandby イベント

SYS_EVENT_HSBCONNECTSTATUS 165

SYS_EVENT_HSBSTATESWITCH 165

SYS_EVENT_NETCOPYEND 165

SYS_EVENT_NETCOPYREQ 165

HotStandby 状況 97

HotStandby 操作のシャットオフ 86

HotStandby の管理 98, 99

サーバー状態の切り替え 81

HotStandby の構成

ネットコピーのパフォーマンス 51

HOTSTANDBY_CONNECTSTATUS

SQL 関数 99

HOTSTANDBY_CONNECTSTATUS (SQL 関数) 75

HOTSTANDBY_STATE (SQL 関数) 75

hsb status catchup 147

hsb status connect 147

hsb status copy 147

hsb status switch 147

HSBEnabled (パラメーター) 36, 47, 103, 120

L

Listen (パラメーター) 45, 124

LogEnabled (パラメーター) 47

logpos (hotstandby コマンド) 102

logpos (管理コマンド) 147

M

MaxLogSize (パラメーター) 120

MaxMemLogSize (パラメーター) 120

migratehsbg2 113

N

netcopy 157

1 次側は PRIMARY ALONE 状態でなければならない 10

NetcopyRpcTimeout (パラメーター) 120

NumRetry (パラメーター) 196

O

OFFLINE (状態) 157

P

Password (パラメーター) 124
Password1 (パラメーター) 196
Password2 (パラメーター) 196
Pessimistic (パラメーター) 196
ping 49
PingInterval (パラメーター) 48, 49, 120, 196
PingTimeout (パラメーター) 48, 49, 120
PreferredPrimary (パラメーター) 124
PRIMARY ACTIVE (状態) 157
PRIMARY ALONE 84
PRIMARY ALONE 状態
PRIMARY ALONE 状態での新規 1 次サーバーの実行 84
PRIMARY ALONE 状態でのサーバーの実行 84
PRIMARY ALONE (状態) 157
PRIMARY UNCERTAIN (状態) 157
PrimaryAlone (パラメーター) 120

R

READ COMMITTED
トランザクション分離レベル 37
ReadMostlyLoadPercentAtPrimary (パラメーター) 61, 120
REPEATABLE READ
トランザクション分離レベル 37
RequiredConnectFailures (パラメーター) 124
RequiredPingFailures (パラメーター) 124
RO
アクセス・モード 120
RW
アクセス・モード 120
RW/Create
アクセス・モード 120
RW/Startup
アクセス・モード 120

S

SERIALIZABLE
トランザクション分離レベル 37
Server1_host (パラメーター) 129
Server1_name (パラメーター) 129
Server1_pass (パラメーター) 129
Server1_port (パラメーター) 129
Server1_user (パラメーター) 129
Server2_host (パラメーター) 129
Server2_name (パラメーター) 129
Server2_pass (パラメーター) 129
Server2_port (パラメーター) 129
Server2_user (パラメーター) 129
SET TRANSACTION WRITE 62
SET WRITE 62
solidDB SpeedLoader
定義 36

solidDB エクスポート
定義 36
solidDB データ管理ツール 36
solidDB データ・ディクショナリー
定義 36
solidhac.ini 30
ERE セクション 55
HAController セクション 53
LocalDB セクション 54
RemoteDB セクション 55
SQL 関数
HOTSTANDBY_CONNECTSTATUS 75, 99
HOTSTANDBY_STATE 75
STANDALONE (状態) 85, 103, 157
StartInAutomaticMode (パラメーター) 124
StartScript (パラメーター) 124
SYS_EVENT_HSBCONNECTSTATUS (イベント) 165
SYS_EVENT_HSBSTATESWITCH (イベント) 165
SYS_EVENT_NETCOPYEND (イベント) 165
SYS_EVENT_NETCOPYREQ (イベント) 165

T

TC 情報 64
構文 64
属性の組み合わせ 65
プロプラエタリー ODBC 属性 72
矛盾の処理 66
JDBC 構文 67
TC 接続 60
TF 接続 60

U

Username (パラメーター) 124
Username1 (パラメーター) 196
Username2 (パラメーター) 196

W

Watchdog サンプル 27
WatchdogLog (パラメーター) 196
Window_title (パラメーター) 129

[特殊文字]

-x autoconvert 113
-x backupserver (コマンド行オプション) 92
-x backupserver (コマンド) 92
-x migratehsbg2 113
=
パラメーター値を設定するときの等号の使用 35

特記事項

Copyright © Solid Information Technology Ltd. 1993, 2008

All rights reserved.

Solid Information Technology Ltd. または International Business Machines Corporation の書面による明示的な許可がある場合を除き、本製品のいかなる部分も、いかなる方法においても使用することはできません。

本製品は、米国特許 6144941、7136912、6970876、7139775、6978396、および 7266702 により保護されています。

本製品は、米国輸出規制品目分類番号 ECCN=5D992b に指定されています。

本書は米国 IBM が提供する製品およびサービスについて作成したものです。

本書に記載の製品、サービス、または機能が日本においては提供されていない場合があります。日本で利用可能な製品、サービス、および機能については、日本 IBM の営業担当員にお尋ねください。本書で IBM 製品、プログラム、またはサービスに言及していても、その IBM 製品、プログラム、またはサービスのみが使用可能であることを意味するものではありません。これらに代えて、IBM の知的所有権を侵害することのない、機能的に同等の製品、プログラム、またはサービスを使用することができます。ただし、IBM 以外の製品とプログラムの操作またはサービスの評価および検証は、お客様の責任で行っていただきます。

IBM は、本書に記載されている内容に関して特許権 (特許出願中のものを含む) を保有している場合があります。本書の提供は、お客様にこれらの特許権について実施権を許諾することを意味するものではありません。実施権についてのお問い合わせは、書面にて下記宛先にお送りください。

〒106-8711

東京都港区六本木 3-2-12

日本アイ・ビー・エム株式会社

法務・知的財産

知的財産権ライセンス渉外

以下の保証は、国または地域の法律に沿わない場合は、適用されません。IBM およびその直接または間接の子会社は、本書を特定物として現存するままの状態を提供し、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとします。国または地域によっては、法律の強行規定により、保証責任の制限が禁じられる場合、強行規定の制限を受けるものとします。

この情報には、技術的に不適切な記述や誤植を含む場合があります。本書は定期的に見直され、必要な変更は本書の次版に組み込まれます。IBM は予告なしに、随時、この文書に記載されている製品またはプログラムに対して、改良または変更を行うことがあります。

本書において IBM 以外の Web サイトに言及している場合がありますが、便宜のため記載しただけであり、決してそれらの Web サイトを推奨するものではありません。それらの Web サイトにある資料は、この IBM 製品の資料の一部ではありません。それらの Web サイトは、お客様の責任でご使用ください。

IBM は、お客様が提供するいかなる情報も、お客様に対してなんら義務も負うことのない、自ら適切と信ずる方法で、使用もしくは配布することができるものとします。

本プログラムのライセンス保持者で、(i) 独自に作成したプログラムとその他のプログラム (本プログラムを含む) との間での情報交換、および (ii) 交換された情報の相互利用を可能にすることを目的として、本プログラムに関する情報を必要とする方は、下記に連絡してください。

IBM Canada Limited
Office of the Lab Director
8200 Warden Avenue
Markham, Ontario
L6G 1C7
CANADA

本プログラムに関する上記の情報は、適切な使用条件の下で使用することができますが、有償の場合もあります。

本書で説明されているライセンス・プログラムまたはその他のライセンス資料は、IBM 所定のプログラム契約の契約条項、IBM プログラムのご使用条件、またはそれと同等の条項に基づいて、IBM より提供されます。

この文書に含まれるいかなるパフォーマンス・データも、管理環境下で決定されたものです。そのため、他の操作環境で得られた結果は、異なる可能性があります。一部の測定が、開発レベルのシステムで行われた可能性があります。その測定値が、一般に利用可能なシステムのものと同じである保証はありません。さらに、一部の測定値が、推定値である可能性があります。実際の結果は、異なる可能性があります。お客様は、お客様の特定の環境に適したデータを確かめる必要があります。

IBM 以外の製品に関する情報は、その製品の供給者、出版物、もしくはその他の公に利用可能なソースから入手したものです。IBM は、それらの製品のテストは行っておりません。したがって、他社製品に関する実行性、互換性、またはその他の要求については確認できません。IBM 以外の製品の性能に関する質問は、それらの製品の供給者にお願いします。

IBM の将来の方向または意向に関する記述については、予告なしに変更または撤回される場合があります、単に目標を示しているものです。

本書には、日常の業務処理で用いられるデータや報告書の例が含まれています。より具体性を与えるために、それらの例には、個人、企業、ブランド、あるいは製品などの名前が含まれている場合があります。これらの名称はすべて架空のものであり、名称や住所が類似する企業が実在しているとしても、それは偶然にすぎません。

著作権使用許諾:

本書には、様々なオペレーティング・プラットフォームでのプログラミング手法を例示するサンプル・アプリケーション・プログラムがソース言語で掲載されています。お客様は、サンプル・プログラムが書かれているオペレーティング・プラットフォームのアプリケーション・プログラミング・インターフェースに準拠したアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。このサンプル・プログラムは、あらゆる条件下における完全なテストを経ていません。従って IBM は、これらのサンプル・プログラムについて信頼性、利便性もしくは機能性があることをほめかしたり、保証することはできません。

それぞれの複製物、サンプル・プログラムのいかなる部分、またはすべての派生的創作物にも、次のように、著作権表示を入れていただく必要があります。

© (お客様の会社名) (西暦年)。このコードの一部は、IBM Corp. のサンプル・プログラムから取られています。

© Copyright IBM Corp. _年を入れる_。 All rights reserved.

商標

IBM、IBM ロゴ、ibm.com[®]、Solid、solidDB、InfoSphere[™]、DB2[®]、Informix[®]、および WebSphere[®] は、International Business Machines Corporation の米国およびその他の国における商標です。これらおよび他の IBM 商標に、この情報の最初に現れる個所で商標表示 (® または ™) が付されている場合、これらの表示は、この情報が公開された時点で、米国において、IBM が所有する登録商標またはコモン・ロー上の商標であることを示しています。このような商標は、その他の国においても登録商標またはコモン・ロー上の商標である可能性があります。現時点での IBM の商標リストについては、「Copyright and trademark information」(www.ibm.com/legal/copytrade.shtml) をご覧下さい。

Java およびすべての Java 関連の商標およびロゴは Sun Microsystems, Inc. の米国およびその他の国における商標です。

Linux[®] は、Linus Torvalds の米国およびその他の国における商標です。

Microsoft および Windows は、Microsoft Corporation の米国およびその他の国における商標です。

UNIX は、The Open Group の米国およびその他の国における登録商標です。

他の会社名、製品名およびサービス名等はそれぞれ各社の商標です。



Printed in Japan

SC88-5817-00



日本アイ・ビー・エム株式会社
〒106-8711 東京都港区六本木3-2-12