パブリッククラウド環境におけるデータベースの冗長化構成の構築方法

技術を知る

データベースをクラウドリフトする際には、クラウド特有の要素を考慮し、可用性と信頼性を確保する必要があります。本記事では、クラウド環境の例として Microsoft Azure を利用したデータベースの冗長化構成を取り上げ、クラウド特有の構築ポイント、構成例、および障害発生時の対応について説明します。

1. Azure 利用時の構築設計ポイント

オンプレミス環境では、サーバー、ストレージ、ネットワーク機器といったハードウェアに加え、OS、ハイパーバイザー、仮想化ソフトウェアなどの全ての要素を自社で調達・構築・管理する必要があります。
一方、クラウド環境では、ハードウェアの調達、設置、設定といった物理的な作業は不要となり、仮想環境の基盤もクラウド事業者が管理します。この変化に伴い、クラウドサービスではクラウド事業者と利用者の間に「責任分界点」が設定され、インフラ基盤の責任範囲が明確に定められるようになりました。そのため、Azure などのクラウド環境でシステムを構築する場合、この責任分界点を意識し、以下の 2 点に対応することが重要になります。

サービスレベルアグリーメント（SLA）

Azure は、ネットワークやストレージなどの可用性が担保された状態で提供されています。これらの稼働率や Azure の各サービスの可用性については SLA が公開されており、要件定義時に確認しておく必要があります。これを踏まえ、ネットワークやストレージなどの監視タイムアウト時間を調整するといった、各種タイムアウトの設定を検討することが推奨されます。

可用性ゾーン

Azure は、リージョン内に複数の可用性ゾーン（Availability Zone、以降 AZ）を提供しています。AZ は物理的に分離されたデータセンターであり、異なる AZ にリソースを配置することで、単一障害点の回避に役立ちます。特に高可用性と事業継続性が求められるシステムでは、AZ 障害時にもシステムを稼働させ続けるために、同一リージョン内の別の AZ にデータベースを配置（複製）する、いわゆるマルチ AZ 構成が推奨されます。AZ を適切に活用することで障害耐性が向上し、サービスの信頼性を高めることができます。

2. Azure マルチ AZ 環境でのデータベース構築方法

ここでは、データベースとして Fujitsu Enterprise Postgres（以降 Enterprise Postgres）を利用する場合について説明します。
Enterprise Postgres はエンタープライズ環境における可用性向上のため、データベース多重化機能を搭載しています。この機能の管理は、Mirroring Controller（以降 MC）と呼ばれる機構が行います。

データベース多重化における重要な要件の一つに、スプリットブレインの防止があります。スプリットブレインとは、データベースサーバー間のネットワークで異常が発生し、互いのサーバーの状態を把握できなくなることで、両方のデータベースサーバーが同時にプライマリサーバーとして動作してしまう状態を指します。この状態になると、それぞれのサーバーが独立してデータの書き込みを受け付けるため、データの不整合や矛盾が発生し、システムの信頼性を損なうおそれがあります。そのため、データベース多重化機能には、データベースサーバーが相互の状態を正確に判断できない場合に、第三者としてデータベースサーバーの状態を客観的に判断し、必要に応じて異常なデータベースサーバーを隔離する機能が備わっています。この機能をインストールしたサーバーを「裁定サーバー」と呼びます。裁定サーバーを利用することで、異常なサーバーをシステムから隔離（フェンシング）し、スプリットブレインを防止できます。

データベース多重化機能が検知可能な異常としては、他にも OS やサーバーのダウンおよび無応答、データベースプロセスのダウンおよび無応答、ディスク障害、ネットワーク異常、ストリーミングレプリケーション異常が挙げられます。これを踏まえ、Enterprise Postgres を利用したマルチ AZ 構成の例を紹介します。

構成例

図 1 は、裁定サーバーをデータベースとは異なる AZ に配置した、3 つの AZ を利用した構成の例です。この構成により、データベースが存在する AZ で障害が発生した場合でも、裁定サーバーによる監視・制御が実行でき、スプリットブレインを防止できます。なお、裁定サーバーの可用性を確保するために、裁定サーバーの冗長化を推奨します。

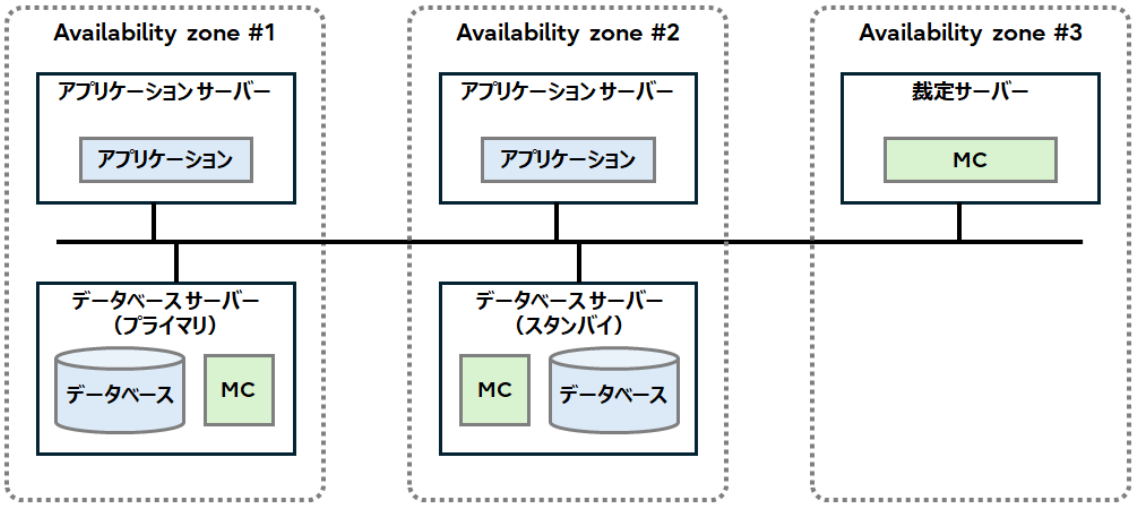

図 1：裁定サーバーを別 AZ に配置する構成

2 つの AZ で構成する場合は、裁定サーバーの代わりに、各データベースサーバー内で裁定処理を行う「裁定コマンド」の利用をご検討ください。詳細は、以下のオンラインマニュアルを参照してください。

- Fujitsu Enterprise Postgres 17 クラスタ運用ガイド（データベース多重化編）

ここまでは構成について説明しました。次の章では、Azure 環境における障害対応について、注意すべきポイントや具体的な手順などを説明します。

3. Azure でのデータベース障害への対策

クラウド環境への移行におけるシステム障害対策では、クラウドならではの特徴を考慮する必要があります。特に Azure のようなパブリッククラウド環境では、インフラの物理的な構成がブラックボックス化されているため、オンプレミス環境のように詳細な原因究明や切り分けが困難になる場合があります。そのため、Azure 環境では、従来のシステム運用における「信頼性」「可用性」「保守性」の考え方をクラウドの特性に合わせて見直し、クラウド固有のリスクを考慮・検証した上で、システム全体の安定稼働を確保する必要があります。

本章では、Azure 環境におけるデータベースの障害対応について、クラウドならではの視点で 対策を解説します。以下に、AZ の障害が発生した場合のスプリットブレインを防止する 仕組みと、フェンシング方法について解説します。

3.1 スプリットブレインを防止する仕組み

スプリットブレインは、例えば、データベースのフェイルオーバー後に、意図せず元のデータベースサーバー（プライマリ）が再起動されたり、遮断されていたネットワークが復旧したりした場合などに発生しやすくなります。そのため、フェイルオーバー時に確実にフェンシングを実行することが、スプリットブレイン防止の重要なポイントです。データベース多重化機能には、フェイルオーバー時に障害が発生したデータベースサーバーをフェンシングする仕組みが備わっています。

以下に、AZ#1 で AZ 障害が発生した場合を例に、その仕組みを説明します。

1. **異常検知**：AZ#2 の MC が、AZ#1 のデータベースサーバーに対する死活監視で異常を検知
2. **裁定依頼**：AZ#2 の MC から裁定サーバーへ、AZ#1 のデータベースサーバーの状態確認を依頼
3. **裁定**：裁定サーバーが AZ#1 のデータベースサーバーの状態を確認し、正常/異常を判断
4. **フェンシング**：裁定サーバーが AZ#1 のデータベースサーバーを異常と判断した場合、フェンシングを実行
5. **自動縮退**：フェンシング成功で、自動縮退が実行され、スタンバイ側のデータベースがプライマリに昇格。フェンシング失敗で、自動縮退は中断

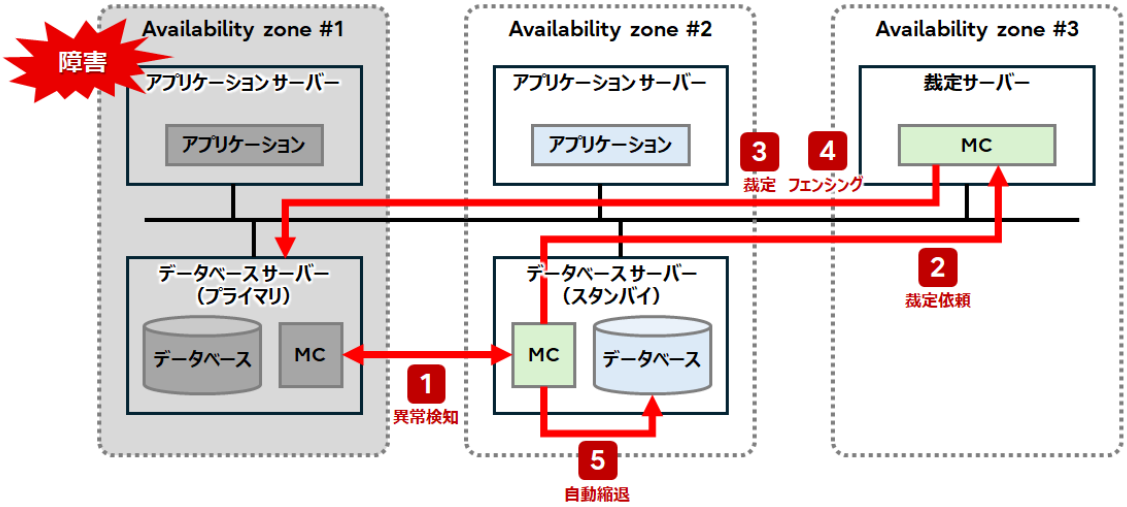

図 2：AZ#1 に障害が発生した場合

AZ#2 で AZ 障害が発生した場合も、同様の流れでフェンシングを実行できます。
なお、フェンシング処理については、システム要件に応じて適切なフェンシング方法を決定し、コマンドを作成する必要があります。このコマンドを「フェンシングコマンド」と呼びます。具体的な内容については、以下で例を用いて解説します。

3.2 フェンシング方法

クラウド環境におけるフェンシングとしては、データベースのフェンシングとネットワークのフェンシングという 2 つの手法が考えられます。

データベースのフェンシング

データベースのフェンシングでは、対象のデータベースサーバーを特定し、停止または隔離するなどの手法が考えられます。クラウド環境におけるデータベースの停止や切り離しの手段については、事前に検討・検証が必要です。具体的な手順は以下のとおりです。

処理	内容
1. フェンシング対象データベースサーバーの特定	裁定サーバーからフェンシングコマンドに渡される引数「サーバー識別子」を基に、フェンシング対象のデータベースサーバーのホスト名または IP アドレスを特定します。
2. データベースサーバーの状態確認	フェンシング対象のデータベースサーバーの状態を、例えば Azure CLI コマンド（例：az vm get-instance-view）で確認し、フェンシング実行可否を判断します。稼働中または状態が取得できない場合は、次の「処理 3」に進みます。停止状態の場合は、フェンシングは不要なため、復帰値 0（MC による縮退処理を継続）で終了します。
3. フェンシングの実行（対象のサーバー停止）	フェンシング対象のデータベースサーバーを停止するため、Azure CLI コマンド（例：az vm stop）を使用します。
4. データベースサーバーの状態確認	再度、Azure CLI コマンドなどでデータベースサーバーの状態を確認します。この処理は、データベースサーバーが停止するまでリトライします。停止状態の場合は、復帰値 0 で終了します。停止状態が確認できない場合は、復帰値 0 以外（MC による縮退処理を中断）で終了します。

ネットワークのフェンシング

ネットワークのフェンシングにおいては、切り離すべきネットワーク範囲の検討が重要です。障害時には、そのネットワークを隔離します。例えば、AZ へのアクセスにロードバランサーを利用している場合、ロードバランサーの設定を変更し、障害が発生した AZ へのトラフィックを遮断する、といった手法が考えられます。このように、障害が発生した AZ へのデータ流入を防ぐことで、正常な AZ のみで業務を継続させることが可能になります。

おわりに

本記事では、Azure 環境におけるデータベースの可用性を最大限に高めるための冗長化構成と設計指針について解説しました。クラウド環境において、冗長化はビジネス継続性を担保するための不可欠な要素であり、障害発生時には確実なフェンシングを行い、サービスを継続することが重要です。
本記事が、Azure 上で Enterprise Postgres をご活用いただく上で、システム構築のヒントとなれば幸いです。

2025 年 5 月 12 日