MySQL Clusterによるクラスタ・データベースの実装と評価

Implementation and Evaluation of Cluster Database with MySQL Cluster

中山　陽太郎，林　　宏　祥

要　約　クラウドコンピューティング基盤やSaaS基盤上において，データベースサーバの可用性と拡張性への対応は，サービスレベル契約（Service Level Agreement．以降，SLA）要件の観点から重要な問題であるが，DBサーバのアクティブ/スタンバイ構成による対応やスケールアップによる対応では，信頼性の保障や運用管理の複雑化などの課題が生じる．また，商用のデータベース製品では，可用性と拡張性に対応したクラスタ製品もあるが，SaaS基盤として適用するためには，コストの増大が課題である．MySQL Clusterは，可用性と拡張性を同時に実現する機能性を備えたOSSのクラスタDBである．今回，MySQL Clusterの性能評価を実施した結果，MySQL Clusterが可用性と拡張性，コストパフォーマンスに優れたデータベース・クラスタであり，特に信頼性とリアルタイム性に高度なSLA要件が求められるシステムに適していることが確認できた．

Abstract　Guarantee of availability and scalability for a database server is important thing from the viewpoint of Service Level Agreement (SLA) in the cloud computing or SaaS infrastructures. There are some problems in guaranteeing the reliability and complexity in case of approaching to the active/standby or vertical scaling (scaling-up) of database server configuration. Though there are commercial database cluster products that enable both availability and scalability, adoption of them as SaaS infrastructure causes the cost increase. MySQL Cluster is the open source cluster database that can realize both the availability and scalability at the same time. As the result of this evaluation of the performance of MySQL Cluster, we confirm that MySQL Cluster is the database cluster that shows the advantage in availability, scalability and cost performance, and that also is suitable for the system that requires advanced SLA to the reliability and real-time nature.

1. はじめに

クラウド上にSaaSを利用してシステム構築を行うことがITシステム構築の選択肢の一つと考えられるようになってきた．SaaS基盤において，データベースシステムの可用性とトランザクション処理における拡張性への対応は，SLAの観点から重要な課題である．通常のデータベースの運用においては，アクティブ/スタンバイ型クラスタによる高可用性の対応と，ハードウェアリソースのスケールアップによる拡張性の対応を行い，それぞれ独立した要件として実現するのが普通である．商用データベース製品の中には，クラスタ対応したOracle社のOracle RAC（Real Application Cluster）のような可用性と拡張性を同時に実現可能なデータベース管理システム製品もあるが，共有ディスクなど高価なハードウェアが必要となり，SaaS基盤として適用するためには，ライセンスや運用管理なども含めたコストの増大が問題となる．これに対する解決策の一つとして，オープンソースソフトウェアを利用することが考

えられる．

MySQL Cluster は，MySQL 社[*1] によって開発されたオープンソースのクラスタ・データベースであり，これまでのオープンソースの DBMS では対応できていなかった可用性と拡張性を同時に実現する機能性を備えたクラスタ DB である．欧米においては既に大手通信キャリアなどの実績も多いが，国内では採用事例も少なく情報も充分ではない．そのため，標準的な手法に準拠した評価作業を実施し，システム適用のための標準的な参照モデルを提供していくことが課題となっている．

今回，MySQL Cluster 6.2[*2] が，実システムにおいてどの程度使えるのか，その性能と特性を明らかにすることを目的として性能評価を行った．評価ツールとしては，IPA[*3] によって OSS データベースを評価するために整備されたベンチマークツール（DBT-1）を使用し，MySQL Cluster のクラスタ DB としての性能を検証した．結果として，MySQL Cluster は，可用性と拡張性を同時に実現し，かつコストパフォーマンスに優れていることが実証された．

本稿では，MySQL Cluster のアーキテクチャを解説するとともに，特に拡張性における性能評価について報告する．2 章では，クラスタ・データベース技術の説明と OSS DBMS における課題について述べ，3 章で，MySQL Cluster の特徴的なアーキテクチャを概説する．4 章で今回実施した MySQL Cluster 6.2 の性能評価の結果について説明し，5 章では，MySQL Cluster 6.2 の性能評価と特性を踏まえたクラスタ構築時のポイントについて解説する．6 章でまとめとして，SaaS 基盤におけるクラスタ DB としての MySQL Cluster 6.2 の優位性について述べる．

2. クラスタ・データベース技術の現状と OSS DBMS における課題

データベースにおけるクラスタ構成は，アクティブ/スタンバイ型（ホットスタンバイ型）とアクティブ/アクティブ型とに大別され，可用性と拡張性に関してそれぞれ特徴が異なる．アクティブ/スタンバイ型クラスタでは，稼働系と待機系の冗長構成によって，稼働系がダウンした場合に待機系へフェイルオーバされてサービスが継続される．それに対して，アクティブ/アクティブ型クラスタでは，サーバはすべて稼働系となり，稼働系サーバのいずれかがダウンした場合でも，残りの稼働サーバによってサービスを継続することが可能である．このため，アクティブ/スタンバイ型に比べて，より可用性の高いサービスが提供可能である．加えて，サーバの追加によるスケールアウトによって，サービス処理量の拡張や負荷分散に対応する．

アクティブ/アクティブ型は，実装方式によって共有ディスク型と非共有型とに区別され，それぞれシェアード・エブリシング，シェアード・ナッシングと呼ばれる[3], [4]．表 1 にそれぞれの特徴を示す．

商用データベースの一つである Oracle には，共有ディスク型クラスタ機能である Oracle RAC（Real Application Cluster）がオプションとして提供されている．Oracle RAC は，ミッションクリティカルな業務分野へ対応可能な可用性と拡張性を備えたクラスタ DB である．OSS DBMS においても，MySQL Cluster を含め，いくつかクラスタ対応機能が提供されているものもあるが，商用クラスタ製品に匹敵するほどの性能や信頼性を確立できていないのが現状である．

MySQL Cluster は，キャリアグレード要件と呼ばれる，通信事業に利用可能な高信頼性要件に対応しており，欧米での大手通信キャリアにおける実績も多い．しかしながら，日本国内

表1　クラスタ方式の特徴

クラスタ方式		特徴
アクティブ/スタンバイ型	フェイルオーバ型	・稼働系と待機系によるアクティブ／スタンバイ構成であり，稼働系のDBサーバのみ利用可能. ・負荷分散やスケールアウトによる拡張性に対応できない.
アクティブ/アクティブ型	共有ディスク型(シェアード・エブリシング)	・可用性や柔軟な拡張性の面で非共有型より優れている反面，リソース共有によるホットスポットが生じ易い. ・データベース設計が容易. ・追加ノード数は制約あり. ・共有ディスクが必要でありH/Wコストが高い. (例)Oracle RAC
	非共有型(シェアード・ナッシング)	・並列検索処理性能が高い．但し，データの分散状況がデータアクセスの効率に影響する. ・スケールアウトによるノード拡張性に優れている. 但し，DB再編成が必要になる. ・特殊なH/Wを必要としない. (例)IBM DB2, MySQL Cluster
	レプリケーション型	・検索処理の負荷分散に向く. ・同期レプリケーションと非同期レプリケーションがある. ・同期レプリケーションでは，更新性能が著しく低下.

においてはほとんど採用事例がなく，評価事例も充分なものが存在していない．国内におけるオープンソースDBMSの評価については，IPA OSSセンターによる「OSS性能・信頼性評価プロジェクト」によって推進されてきており，DBMSにおけるクラスタ性能評価についても「DB層～DBMSクラスタ評価編」[6]で報告されているが，可用性や効率性については，商用製品に比べまだ充分ではないことが分かる．また，実システム適用への参考となるような資料として適当なものは存在していないのが実情である．

今後，OSS DBの性能を向上させるため，OSS DBMSにおけるクラスタ技術に対する精度の高い評価を実施し，情報を共有していくことが重要と考える．

3. MySQL Cluster概要

3.1 MySQL Clusterのアーキテクチャ

MySQL Clusterは，アクティブ/アクティブ型クラスタアーキテクチャによって，拡張性と可用性を同時に実現するクラスタ・データベースであり，リアルタイム性と信頼性に優れ，サーバ障害を局所化して，システムの全面停止によるリスクを排除する[1],[2],[5]．レベル6.1までのMySQL Clusterは，メモリDBのため，データベース・サイズがメモリサイズに制約されていたが，6.2では，ディスクストレージに対応した．メモリ上に乗りきらないデータをディスク上に保持するため，メモリサイズに制限されていたDBサイズの制約が排除された．

MySQL Clusterのアーキテクチャを図1に示す．MySQL Clusterは，SQLを実行するSQLノードと，NDB（ネットワーク・データベース）エンジン*4と呼ばれるストレージエンジンから構成される．NDBエンジンは，Dataノードによって構成され，SQLサーバ群と独立した構成となっている．クライアントより実行されたSQLは，SQLノード上で実行計画が組み立てられ，Dataノードに対して実行される．

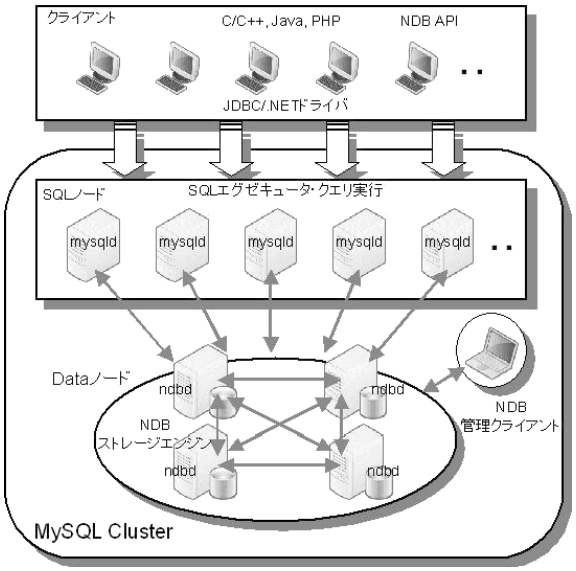


図1　MySQL Cluster アーキテクチャ

3. 2　MySQL Cluster における高可用性の実現

MySQL Cluster では，非共有型クラスタ DB でありながら，Data ノード間の内部同期レプリケーションによって可用性を実現している．MySQL Cluster のデータベースは，Data ノード数によって自動的に区分化（パーティション化）され，同じデータを持つ Data ノードの集合（ノードグループ）が構成される．また，レプリカ数によって複製の多重度を 1 から 4 までの値[*5] に指定できる．内部同期レプリケーションは，このノードグループ単位で行われ，Data ノード数，ノードグループ数，及びレプリカ数は次の関係を持つ．

総 Data ノード数＝ノードグループ数×レプリカ数

図 2 は，レプリカ（複製）数を 2 とした場合の Data ノードとノードグループの構成を表している．区分化されたデータ（データパーティション）は，内部的にプライマリとセカンダリ

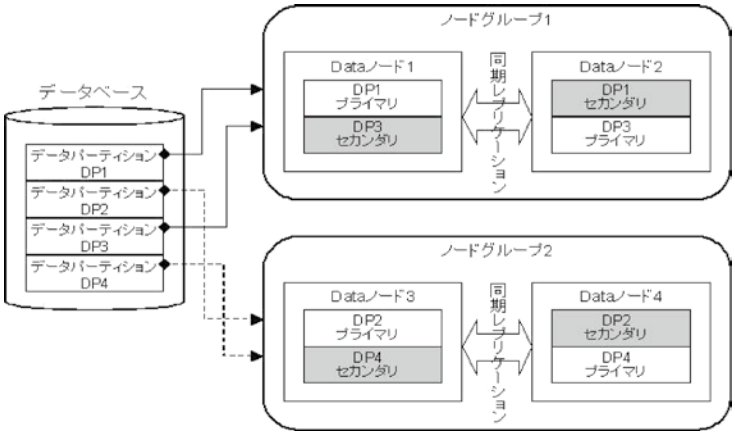


図2　Data ノードとノードグループの構成例（レプリカ数＝2 の場合）

の役割を与えられ，レプリケーション処理におけるマスター・スレーブとして機能する．Dataノードが障害停止した場合，障害側のDataノードのプライマリは，そのノードグループに属するDataノードのセカンダリにフェイルオーバされ，セカンダリが稼働を引き継ぐ．これによって，クラスタ全体のシステム停止を排除する．

4. MySQL Clusterの性能評価

ここでは，今回実施した性能評価について解説する．

4.1 DBT-1ベンチマークツールによる性能検証

性能評価ツールとして，IPAによって整備されたTPC-Wベースの DBT-1を使用し，プラットフォームとして，日立BS320（ブレードサーバ8台構成）を使用した．今回の評価環境を表2に示す．DBT-1は，オンライン書店をシミュレートするTPC-Wをベースとした簡易版の効率測定ツールである．クライアントからの商品の照会や購入など，いくつかのトランザクション処理パターンが実行される．DBT-1では，クラスタ環境で実行するための仕組みは提供されていないため，今回はSSH[*6]を利用し，各SQLノードに対して同期をとってクライアントを起動する仕組みを組み込んで効率測定を実施した．

表2　評価環境

ハードウェア	日立BS320 (Blade Symphony) 8ブレード構成 CPU: Xeon 5160 3.0GHz(Dual Core)x2 /ブレード メモリ: 8GB (4GBx2) 内蔵ディスク: 73GB SAS x 2 (10000r/min,RAID1構成) アーキテクチャ: x86_64
ソフトウェア	OS: Redhat Enterprise Linux AS 4.0 Update6 (x86_64) DBMS: MySQL 5.1.22

測定では，SQLノードとDataノードの台数を増加させながら，同時に関連するパラメタの設定を変えて，最適な構成を導出した．今回は，特に顕著な効率改善の見られた，SQLノード8台に対してDataノードを2台，4台，6台と増やした場合のスケーラビリティについて解説する．SQLノード8台，Dataノード6台によるMySQL Clusterのクラスタ構成を図3に示す．

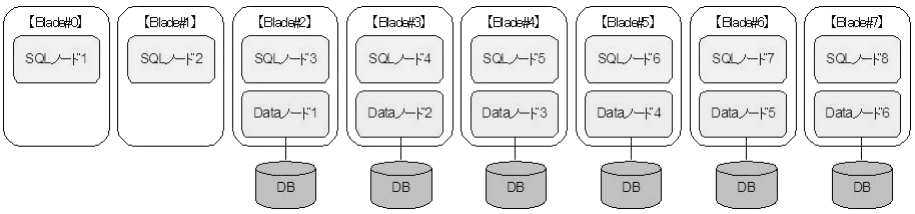


図3　SQLノード8台/Dataノード6台によるクラスタ構成

4.2 DBT-1性能測定結

測定結果を図4に示す．縦軸は，単位時間当たりのトランザクション量（BT値）を，横軸は，仮想ユーザ数（EU数）を示している．グラフのBT値とEU数は，SQLノードごとの各値の総和である．グラフの各線は，SQLノードを8台に固定し，Dataノードを2台（下から2番

目のグラフ），4 台（3 番目），6 台（4 番目）と増やした場合のスループットの推移を表している．また，一番下のグラフは，改善前（SQL ノード 8 台，Data ノード 6 台）のものである．

今回測定を行った中で，最大スループット（BT 値）が得られた構成は，SQL ノード 8 台・Data ノード 6 台の構成であった[*7]．使用したブレードサーバは最大 8 サーバ構成のため，SQL ノード八つの内六つは，Data ノードと同一サーバ上に配置する構成となっている（図 3 参照）．図 4 のグラフより，Data ノードを 2，4，6 台と増やすことに伴い，チューニングを行っていない改善前のグラフを除いて BT 値がリニアにスケールしている．また，改善後の Data ノード 6 台のグラフでは，クラスタ全体の仮想ユーザ数 9600（SQL ノードあたりの仮想ユーザ数 1200 × 8 ノード＝9600）を過ぎるあたりまで理論値どおりの性能を示している．それ以降，横ばいとなり，BT 値がほぼ飽和に達していることが分かる．

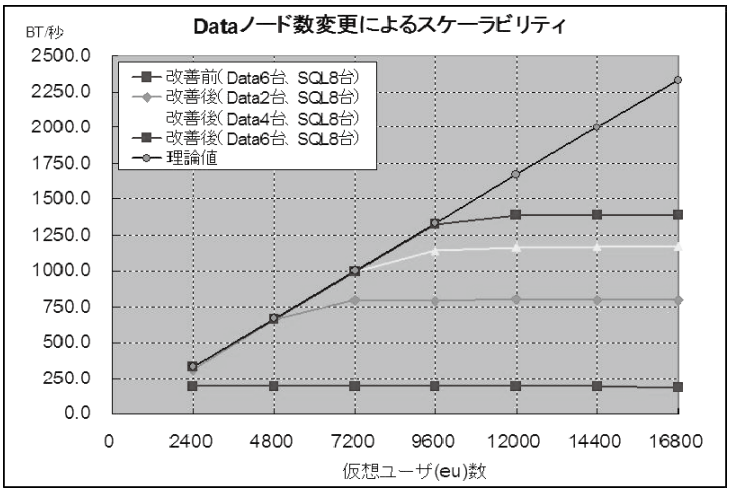


図 4　スループットの推移

4. 3　パラメタチューニングと SQL 実行計画の変更

ここでは，改善前の状態（図 4 の一番下のグラフ）に対して実施したパラメタチューニングと SQL 実行計画の変更について説明する．まず，DBT-1 実行時のシステム統計情報（CPU，メモリ，スワップ，ページング，入出力）を確認したところ，BT 値が伸びない理由はシステムリソース不足が原因ではないことが分かった．次に，DBT-1，MySQL Cluster のパラメタチューニングを行った．すなわち，SQL ノード，Data ノードをそれぞれ 1 台とし，特定の仮想ユーザ数で表 3 にあるパラメタを変更しながら DBT-1 を実行し，効果を測定した．

その結果，最も効果があったのは，⑥「STRAIGHT_JOIN 句指定」であった．MySQL Cluster は，Data ノードから得た統計情報を基に実行計画を立てるが，全文検索となるような SQL が多発する条件では，実行計画が最適とならないケースがある．今回，クエリに時間を要している箇所を特定し，STRAIGHT_JOIN 句により最適なテーブルの結合の順序を保証することで効率改善を行った．

STRAIGT_JOIN 句指定による SQL 実行計画の最適化は以下のように行った．表 4 では，SQL 実行計画の改善前と改善後の SQL 文を示している．この SQL では，item テーブルと author テーブルの結合を行っており，それぞれ 10000 件と 2500 件のデータが格納されている．

表3　効率改善項目一覧

#	パラメタ	Default	初期値	適用値
①	engine_condition_pushdown	ON	ON	ON
②	ndb_force_send	ON	ON	OFF
③	ndb_use_exact_count	ON	ON	OFF
④	table_open_cache	64	64	2048
⑤	query_cache_limit	1048576	1048576	33554432
⑥	STRAIGHT_JOIN句指定	なし	なし	あり
⑦	author, country MyISAM化	なし	なし	あり
⑧	join_buffer_size	131072	131072	1306624
⑨	DBT-1 dbconnection	100	100	50

条件として，author のカラム a_lname に対して，文字列の中間一致検索が行われる．実行計画改善前の SQL では，item が外部表（駆動表），author が内部表として結合処理が行われていた．STRAIGHT_JOIN による改善後の SQL では，author が外部表，item が内部表となる．これによって，SQL ノードと Data ノード間でのデータ転送の負荷が軽減されたためにスループット性能が向上した．

MySQL Cluster では，ネットワーク経由でのノード間の通信量を抑制できるかどうかが効率に大きく影響する．特に，SQL による大量検索処理では，SQL ノードと Data ノード間で大量のデータ転送が発生することになるため，これをどう削減するかが効率改善への重要なポイントと考えられる．

表4　STRAIGHT_JOIN による SQL 実行計画の改善例

```
[改善前]
SELECT i_id, i_title, a_fname, a_lname
FROM item, author WHERE i_a_id=a_id AND a_lname LIKE '%%%s%%' ORDER BY i_title ASC;

[改善後]
SELECT STRAIGHT_JOIN i_id, i_title, a_fname, a_lname
FROM author, item WHERE i_a_id=a_id AND a_lname LIKE '%%%s%%' ORDER BY i_title ASC;
```

4.4　Data ノードにおける CPU ボトルネックの原因と改善

ここでは，BT 値が最大となる SQL ノード 8 台，Data ノード 6 台の構成（図 4 の下から 4 番目のグラフ）で，ユーザ数 9600（1200 × 8 SQL ノード）以降，BT 値が伸びない原因を分析する．図 5 は SQL ノード，図 6 は Data ノードの CPU リソース（4CPU の合計）の使用状況を示している[*8]．それぞれユーザ数 1500 から CPU 使用率の伸びに変化がないことが分かる．その時点での SQL ノード（図 5）の CPU リソースの使用状況をみると，user は 20％未満であり，まだ充分な余裕があることが分かる．一方 Data ノード（図 6）では，idle が 50％弱という高い状況ではあるが，user は 30％弱であり，CPU 一つ分については，CPU リソースを使い切っていることが分かる．

MySQL Cluster 5.1 では，Data ノードで稼働する NDB エンジンの実行スレッド数は一つである．その実行スレッドが全 SQL ノードからの要求を受け付け，データ処理を行った後，要求元の SQL ノードへデータを送信する役割を持つ．そのため，SQL ノードを増やしたことで，Data ノードの負荷が増加し，実行スレッドが絶え間なく稼働する状況となり，ユーザ数 1500 以上では，BT 値が頭打ちになっていることが分かる．

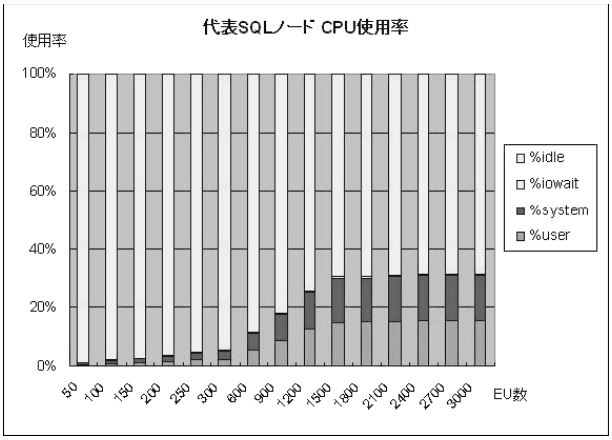


図 5　SQL ノードの CPU リソースの使用状況

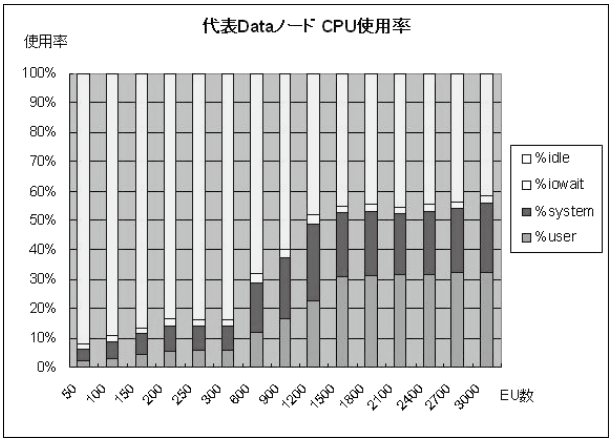


図 6　Data ノードの CPU リソースの使用状況

つまり，レベル 5.1 の MySQL Cluster では，NDB エンジンの実行プロセスが一つであるため，マルチコア環境において CPU リソースを有効に利用できないと言える．NDB エンジンの複数実行スレッド対応は現在開発中であり，CGE（キャリアグレードエディション）6.4 で提供される予定となっている[*9]．次期バージョンで，今回発覚した実行スレッドの負荷によるボトルネックが解消されることが期待できる．

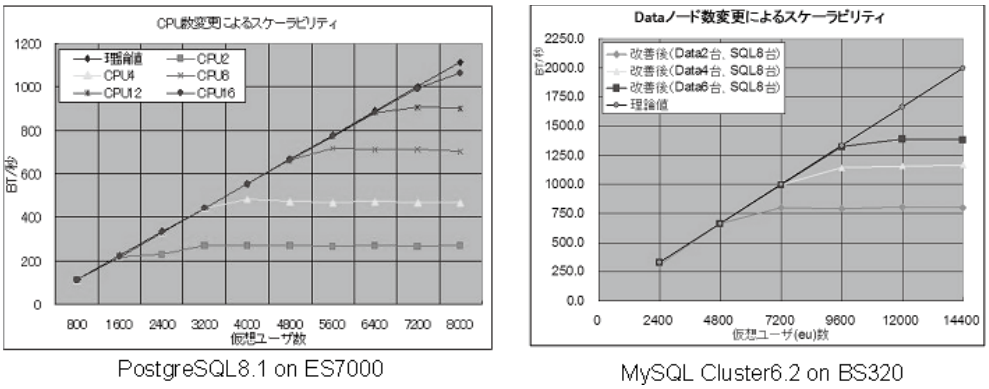
5.　MySQL Cluster 性能評価の考察

5. 1　スループット性能に関する考察― MySQL Cluster と PostgreSQL との比較

DBT-1 性能評価では，ブレードサーバ BS320 を使用し，MySQL Cluster のノード数の増加にともなって，効率がスケールすることを検証した．ここでは，今回測定を行った MySQL Cluster の性能について，OSS DBMS の一つである PostgreSQL と比較することによって，その特性を検討する．

PostgreSQL は，単体の DBMS であり，SMP サーバ上で CPU 数を増加（スケールアップ）させることによってスループットの向上を図るのに対して，MySQL Cluster では，並列分散アーキテクチャにより，ノード数の追加（スケールアウト）によってスループットの向上を図

るというアーキテクチャ上の違いがある.



PostgreSQL8.1 on ES7000　　MySQL Cluster6.2 on BS320

図7　PostgreSQL と MySQL Cluster の DBT-1 測定値

図7のグラフ左側は，2005年度にIPAによって実施されたUnisys ES7000でのPostgreSQL8.1のDBT-1による効率測定結果[7]であり，右側は，今回実施したBS320でのMySQL Cluster 5.1の測定結果である．ES7000 PostgreSQLでは，CPU数16でBT値＝約985（ユーザ数8000）が上限となるのに対して，BS320 MySQL Clusterでは，SQLノード8台，Dataノード数6台でBT値＝1387（ユーザ数12000）までスケールしている*10．これよりMySQL Clusterでは，小型サーバをクラスタ化しスケールアウトすることによって，ハイエンドサーバと同等以上のスループット性能であることが確認できた.

また，可用性については，PostgreSQLは単体サーバであるため，高可用性対応としてホットスタンバイ構成による二重化を行う必要があるのに対して，MySQL Clusterでは，本来のクラスタ構成によって高可用性も同時に対応している．このためMySQL Clusterは，コストパフォーマンスにも優れていると言える.

5.2　MySQL Clusterの用途，及びDB設計に関する考察

ここでは，性能評価で得られたデータを基に，MySQL Clusterの適するシステム，及び留意すべき機能特性としてメモリストレージとディスクストレージの使い分けについて考察と指針を示す．今回の測定により，MySQL Clusterは，より実システムに近いベンチマークモデルにおいても優れていることを実証し，通信系キャリア以外のシステムにも充分適用可能であることが分かった．今回得られた評価結果とMySQL Clusterの特性を考慮し，適合する処理としない処理を簡単にまとめたものを表5に示す.

表5　MySQL Clusterに適合する処理と適合しない処理

適合する処理	適合しない処理
● 単純なデータモデル ● 無停止性とリアルタイム性が必要なシステム ● データ更新(UPDATE)が多いシステム	● 複雑な問合せ，表結合の多い問合せ ● 非索引データによる全件検索 ● バッチ処理のような大量更新処理

また，MySQL Cluster 6.2では，ディスクストレージ対応によって，従来のメモリストレージにおけるDBサイズの制限が大幅に緩和され，可用性が向上した*11．メモリストレージとデ

ィスクストレージは，テーブルごとの属性として指定が可能であり，使い分けと物理設計の指針については，表6のように考えられる．

表6　メモリストレージとディスクストレージの使い分け

メモリストレージが有効なケース	ディスクストレージが有効なケース
● 参照系マスターテーブル ● データの増加率が低く，更新頻度の高いテーブル （注）更新頻度の高いテーブルとは，データ追加ではなく既存データ内容の変更が多いことを指す．	● データ増加率の高いテーブル ● データ容量の大きなテーブル （注）メモリストレージとディスクストレージの効率比較を実施した結果，全データがメモリ上にのる場合は，両者で効率上の差はない．

MySQL Clusterでは，特に信頼性とリアルタイム性に高度なSLA要件が求められるシステムに向いていると言える．また，複雑なSQLについても，ストレージエンジンをうまく使い分けることで対応することも可能であり，ストレージエンジンの長所を生かした適切な設計を行うことが重要である．

6. お わ り に

今回の性能評価によって，MySQL Clusterでは非共有型クラスタの技術により，ハイエンドサーバと同等以上のパフォーマンスを実現できることが確認できた．MySQL Clusterでは拡張性と同時に可用性を実現し，かつ共有ディスクのような高価なハードウェアを必要としないため，コストパフォーマンス的にも優れていることが実証された．また，可用性の評価については，評価内容について具体的な説明は省いたが，クラスタを構成するSQLノード，Dataノードの障害停止をシミュレートし，MySQL Clusterのシステム全体が停止しないことを確認している．

今後，企業レベルにおけるクラウドの利用が本格化するのに従い，SaaS基盤におけるDBサービスには，企業内に構築されている既存の基幹系システムと同等のレベルを保障できるものであることが要求されると考えられる．このような高度なSLA要件に対し，MySQL ClusterをSaaS基盤に利用し，高度なデータベースサービスを提供することが期待される．

＊1　MySQL社は2008年1月にSun Microsystemsに買収され，また2009年4月にSun MicrosystemsはOracle社によって買収されたため，現在MySQL Clusterは，Oracle社のオープンソース製品である．
＊2　MySQL Cluster 6.2は，2008年5月にリリースされた．また，2009年4月にMySQL Cluster 7.0がリリースされた．
＊3　独立行政法人情報処理推進機構
＊4　NDBエンジンは，Ericson社によって，テレコム/IP環境向け高可用性クラスタ・データベースNDB（Network Database）として開発され，2003年にMySQL社に買収された．
＊5　レプリカ数1は単なるストライピングとなり，多重化による可用性は保障されない．
＊6　SSH（Secure SHell）：ネットワークセキュリティに対応したリモート操作用のシェル．
＊7　理論的には，SQLノードとDataノードを増加するほど，トランザクション処理量が増加することが推測される．
＊8　図5，図6のグラフは，SQLノード，Dataノードのそれぞれ代表ノードのデータ．
＊9　MySQL Clusterでは，CGEと呼ばれるリリースバージョンで管理される．詳しくは，マニュアルを参照のこと．http://dev.mysql.com/doc/refman/5.1/en/mysql-cluster-cge.html.
＊10　H/W環境（CPU，メモリサイズ，ディスク性能）が異なるため単純に比較はできないことに注意．
＊11　メモリストレージでは，データ容量がメモリストレージのサイズを越えた場合，システムを停止し，メモリの増加及び再設定作業を行う必要があった．

参考文献
［1］ MySQL AB, “Benchmarking Highly Scalable MySQL Clusters”, Technical White Paper, MySQL AB, 2007.
［2］ Ronstrom M., Oreland J., “Recovery Principles of MySQL Cluster 5.1”, VLDB, PP.1108-1115, 2005.
［3］ Stonebraker M., “The Case for Shared Nothing”, IEEE Database Engineering, Vol.9, No.1, 1986.
［4］ 中山陽太郎,「PostgreSQLによる共有ディスク・クラスタPG-CALSの設計と実装」, ユニシス技報, 日本ユニシス, 通巻94号, 2007年11月
［5］ 中山陽太郎, 岩下忠資,「詳解！ MySQL 5.1 第4章：MySQL Cluster 5.1を使ってみよう」, WEB+DB PRESS Vol.45, 技術評論社, 2008年
［6］「OSS性能・信頼性評価/障害解析ツール開発」DB層～DBMSクラスタ評価編～, OSS技術開発・評価コンソーシアム, 日本OSS推進フォーラム, 2005年
［7］「DBT-1によるパッチを適用したPostgreSQL8.1.2の32ビットマシン（IA32）でのCPUスケーラビリティに関する考察（チューニング有り）」, IPA OSS iPedia（オープンソース情報データベース）, 2007年2月
http://ossipedia.ipa.go.jp/capacity/EV0612250287/（URL存在確認：2009/8/10）

執筆者紹介
中 山　陽 太 郎（Yotaro Nakayama）
1988年日本ユニシス(株)入社. Unisys汎用機Ⅸシリーズのデータベース管理システム主管業務を経て, Unisys製オープン系データベースの開発保守等に従事. 2006年よりOSS DBのクラスタ機能開発, 評価を実施. 現在, 日本ユニシス(株)総合技術研究所先端技術部データエンジニアリング室に所属.

林　　宏　祥（Hiroyoshi Hayashi）
1990年日本ユニシス・ソフトウェア(株)入社. 1992年から, Ⅸシリーズ・メインフレームのデータベース管理システムの主管作業に従事. 2006年からは, OSS DBの調査および評価を実施. 現在, ユニアデックス(株)ソフトウェアサービス事業グループ システムサービス統括部ミドルウエアサービス部に所属.