1

MySQL 概要

日本 MySQL ユーザー会
MySQL Nippon Association
http://www.mysql.gr.jp/

2004-10-22

MyNA

2

MyNA History

黎明期

- **1997** とみた **(** 当時のハンドル名は民斗 **)** 氏による日本語対応パッチ **(ujis,sjis)** および日本語ドキュメント
- **1998 ML** スタート

本格稼働

- **2000 MyNA** に名称決定。**mysql.gr.jp** 取得。サーバー設置
- **2003** 初めてのユーザー会の勉強会
- **200401** 初めての **developer summit**
- **2004 MyNA** サーバー、クラックさる。さらに **MyNA** サーバー **disk** クラッシュ。
  泣きっ面に蜂状態。
- **200406** 新 **MyNA** キャラクターが内緒で登場
- **200408** やっと **MyNA** サーバーの **web** 戻ったよ。
- **200408 OSC** 参加
- **200409 LinuxWorld** 参加

MyNA

3

MyNA ではなにをやっているか？

- ML, Web
- 勉強会
  - 2004 年 11 月 20 日 (@ 東京) やります。
- イベント参加
  - 今 , 2004 年の LinuxWorld, OSC
- bug, 要望取りまとめ
- 執筆依頼があったら募集
- 飲み会

体制

- 大阪支部 (遠藤氏、まいぱぱ氏) 、北海道支部 (たてやん氏) など
- 会長 (とみた氏) 、副会長 (たてやん氏) 、Webmaster(つつい氏、さかい氏)
- サーバーは借りてます
- 金は集めてません

4

MySQL History

- 1979 Monty が UNIREG 書く (メモリ 32kB のマシンで動作)
- 1994 MySQL 開発開始。(Innobase Oy 設立)
- 1995 MySQL 1.0 誕生。既にマルチスレッド構造
- 1996 MySQL 3.11 インターネット上に公開。LIMIT 文
- 1996 Perl, PHP/FI インターフェース登場
- 1997 日本語対応 (ujis,sjis 。とみた氏) 。REPLACE 文
- 1998 MyNA ML スタート (このときはまだ MyNA ではない)
- 1998 Ruby インターフェース (とみた氏)
- 1999 HEAP, MyISAM テーブル
- 2000 MySQL AB 設立。(それまでは TcX Data Konsult AB)
- 2000 レプリケーション、RAID 、MERGE 、tcp-wrapper 、EMIC クラスター
- 2001 InnoDB(トランザクション) 、SSL 通信、クエリー・キャッシュ
- 2003 副問い合わせ、ストアドプロシジャ、OpenGIS
- 2004 NDB クラスター

5

MySQL 特徴 1

- 高速な動作
  - **MySQL** は素早いレスポンスを最も重要な目標として掲げています。マルチスレッドで処理を行います。
  - 主要 **DB** サーバにて行なったベンチマークテスト **(http://www.mysql.com/eweek/)** では、**MySQL(InnoDB)** は **Oracle** に匹敵する評価を得ています。
- 手軽に使える
  - インストール、運用はとても簡単に出来ます。
  - データはファイルに格納されているため、バックアップや移動を簡単に行なうことができます。
- 世界中に普及
  - 世界で最も人気のあるオープンソースデータベースです。
  - 米「 **Linux Journal** 」誌の読者投票では **6** 年連続で最高のデータベースに選ばれています。
- 多くのプラットフォームをサポート
  - **Linux, Microsoft Windows, FreeBSD, Sun Solaris, AIX, Mac OS X, HP-UX, QNX, Novell NetWare, SCO OpenUnix, SGI Irix, Dec OSF** といった様々なプラットフォームに対応しています。
  - 特に **Windows** 版は、初心者にとって心強い存在になっています。

6

MySQL 特徴 2

- オープンソース
  - **MySQL** は、「オープンソースの商用データベース」です。公開当初からソースが公開され、社員だけではなく、全世界のボランティアの手で機能の拡張や修正が行なわれています。
- 多言語対応
  - データはもちろん、テーブル名やフィールド名にも日本語を使用できます。また、文字列処理関数の多くがマルチバイト対応され、便利に日本語を利用できます。
- 選択可能なライセンス形態
  - **GPL** とコマーシャルライセンスが選択できます。**GPL** が条件にあわない場合は、コマーシャルライセンスを選択して **GPL** を回避することができます。**(** 商用利用するかどうかは関係ありません。あくまでも **GPL** を承諾するか否か **)**
- 多種の対応言語
  - **C** 、**C++** 、**PHP** 、**Ruby** 、**Perl** 、**Python** 、**Java** 、**ODBC** 、**Tcl** などの多くの言語で **MySQL** を利用できます。
  - 特に **PHP** との組み合わせでよく使われており、**Linux + Apache + MySQL + PHP** の環境を表す **LAMP** という言葉までできているほどです。
  - **ANSI SQL 92** に準拠しています。

7

MySQL と MySQL AB

- **MySQL** を開発、メンテナンスしているのは、**MySQL AB** というスゥエーデンの会社です。開発者は世界中に何十人といます。
- ソースの保証
  - **MySQL AB** は、ソースの保証をしています。
  - ユーザーからの修正、提供パッチは必ず **AB** の人間がライセンスをチェックしています。
    - ライセンスや特許侵害における心配はありません。
- **MySQL** の向かうところ
  - バグが無いこと
  - 楽である、簡単であること
  - 使って楽しい
- **MySQL AB** 自体の売上は、毎年 **2,3** 倍で増えているらしい。
  - それだけ世界に認められているらしい。開発者も多いらしい
  - **NASA, Yahoo, google,** 多くのプロバイダー**, ...**
  - **ER/Studio ver.6** が **MySQL** に対応。

8

MySQL のライセンス

- **GPL** かコマーシャルライセンスか
  - **GPL** を選択するなら費用は **0** 円
  - **GPL** を選択しないならコマーシャルライセンスを購入すれば **OK**
  - **PHP** と合せて使用する場合は、特例のライセンスがあります。**(0** 円 **)**
  - 商用利用するかどうかは関係ありません。
  - **2000** 年にこのライセンスに変わりました。それ以前のライセンス条項とは別物です。古い情報で判断すると損ですよ。
- **OS** によるライセンスの違いはありません。
- **CPU** 数、接続クライアント数は無関係です。

9

MySQL の製品ラインナップ

- **MySQL**
  - 通常、**MySQL** と言えばこれ。
  - 高速検索と、トランザクションをサポートしています。
  - **NDB** クラスターは **MySQL 4.1** 本体に含まれています。**MySQL AB** が **Alzato** から引き継ぎました。
  - **MySQL-Max** は、**MySQL** の使用できる機能の全てを組み込んだバイナリを指します。**MySQL** と違う物ではありません。
  - **MS-Windows** 用 **(** ネイティブ **)** もずいぶん前からあります。
- **MaxDB**
  - **SAP DB** です。**MySQL AB** が **SAP** から引き継ぎました。
  - 現在、**MySQL** そのものとは統合されていません。将来的には一つにする予定らしいです。
-

10

MySQL の製品ラインナップ 2

- **MyODBC (Connector/ODBC)**
  - **ODBC** インターフェースです
  - **MS-Windows** 用と **Unix** 用があります。
- **Connector/J**
  - **JDBC** インターフェースです。
- **mysql-administrator**
  - **GUI** の管理ツールです。
  - **MS-Windows** 用と **Unix** 用があります。

11

MySQL の今と予定

- **4.0**
  - クエリ・キャッシュ、**SSL, tcp-wrapper, ujis, sjis**
- **4.1**
  - どんなに遅くとも今年中には **production release**
  - **utf8, sjis, ujis** サポート
  - 文字コードの自動変換
  - データベース、テーブル、フィールド単位に文字コードを変更できる
  - 副問い合わせ、**OpenGIS** サポート
  - **NDB** クラスター
- **5.0**
  - ストアド・プロシジャ
  - トリガー
  - ビュー

12

MySQL の活動フィールド

- 適材適所
  - 機能があるというだけで製品を選んでいませんか?
  - 全ての機能が本当に必要ですか?
  - 同時 **100** セッションの確立は、**70msec.** 程度で完了します。
    - これだけ速ければコネクションプーリング自体いらないよね。今までの観念が壊れます。
  - 必要なものだけを、少ない投資で得られればそれでいいのでは?
    - うん千 **(** 百 **)** 万円　対　数万円 **(0** 円 **)**
  - すべての **DB** を **MySQL** に置き換えれるとは思ってません。が、効果的に使えればいいのではないですか? 所詮道具です。
- **MySQL** はソースもバイナリも **Web** で公開されています。
  - ぜひダウンロードして試してください。
- **MySQL AB** という会社の保証付き

13

MySQL の手軽さ1

- 設定ファイルなしでも動きます。
- バイナリインストール
  - **Unix** も **Windows** もバイナリパッケージを展開するだけ。
    - バイナリをコピーすれば動くということです。
  - **MS-Windows** 用 **MySQL** はレジストリを操作しません。
- **CPU** を増やすだけで性能 **UP**
  - **CPU** を追加したとき特別な変更作業は発生しません。
  - もともとマルチスレッドなので無理がない
- データファイルはコピーして違う機械に持っていくだけで **OK**
  - 機種依存性はほとんど無い
- バージョンアップはバイナリを差し替えるだけ
  - 古いファイルはそのまま使用できます。フルダンプの必要はありません。
- フルダンプや差分の記録は **SQL** 文で記録される
  - その記録を違う環境に持って行って、その **SQL** を実行すれば、たやすく同じ状況が再現できる

14

MySQL の手軽さ2

- 多くのバイナリ配布のソフトが最初から **MySQL** 対応
  - ほとんどの **Linux** ディストリビューションには **MySQL** のバイナリパッケージがあります。
  - **MacOSX** にも付属しています。
  - **Perl,PHP,Ruby** なども、だれかがバイナリを作っています
- 多くのレンタルサーバーにも **MySQL** がインストール済み
  - 海外は多いです
- 情報量
  - 日本語書籍 **24** 冊 **(** 以上 **?)**
    - 翔泳社「**MySQL** 徹底 ***** 」、ソフトバンクパブリッシング「実践 **MySQL4** 」
  - 日本語マニュアル
  - 雑誌記事や **web** の記事が増えてきた
    - **Web+DB vol22** は **MySQL BF** による濃い内容
    - 昔に比べればかなり使われてきてきたのかも
    - 間違った記事もある **(** 普及のしるしか? **)**

15

個人的に MySQL の気に入っている点

- 楽で簡単
- シンプル
  - 必要なものだけがあればいい
  - ディスクイメージじゃなくて **SQL** 文で出る
  - データベース **=directory,** テーブル **=file**
- **Unix/Linux** に慣れた者にはうれしいインターフェース
  - コマンドの組み合わせで動作することが多い
  - **mysqlbinlog filename | mysql -h**
  - **tail -f** 詳細ログファイル **| logger**
- バグ対応が速い

16

マルチストレージエンジン

- **MySQL** 独特の特徴
- 最適化
  - 使い分けを行うことで、特別なチューニングを施すことなく、簡単によりよい状態にすることが可能。
- 拡張性
  - 今後新しい考え、手法が登場したとき、簡単に新しい機能を追加できる。しかも今までの部分に影響はでない。
- 互換性
  - 前のストレージエンジンのファイルは引き続き使用可能。
- 保守性、信頼性
  - バグの切り分けのしやすさ。他のエンジンに影響を及ぼしません。

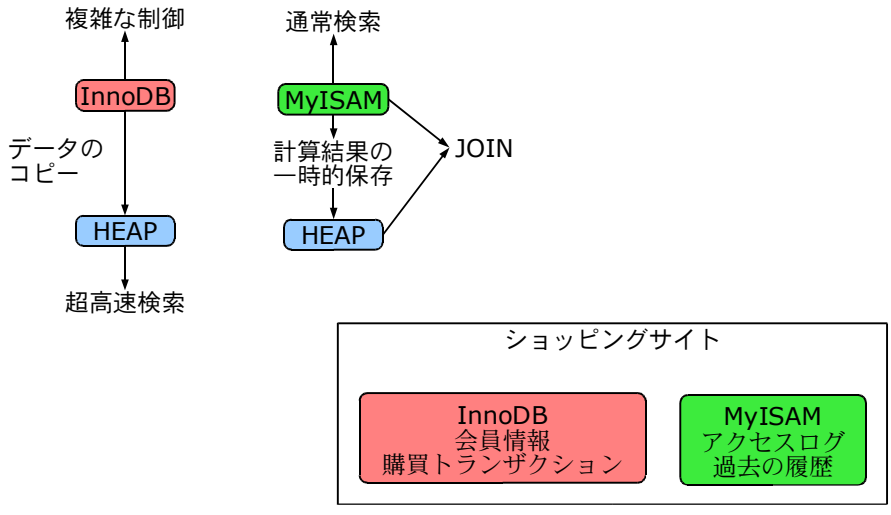
17

各ストレージエンジン概要

	MyISAM	InnoDB	HEAP	NDB
保存メディア	disk	disk	memory	memory
ロック	テーブル	行レベル	テーブル	行レベル
トランザクション	なし	あり	なし	あり
記録量上限	2^64 バイト/1テーブル	64T バイト/全体	メモリ上限	メモリ上限
特徴	速い	トランザクション	速い	クラスター
主な用途	検索が多い場面	トランザクション	一時的な記録 高速な検索	クラスター

18

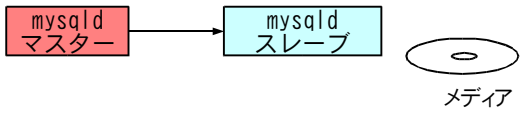
各ストレージエンジン使い分け例



19

レプリケーション使用例 1

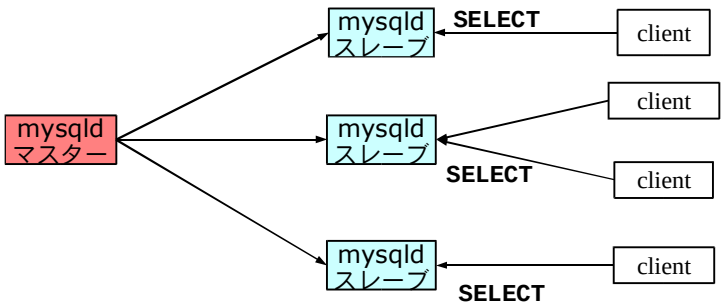
- バックアップ、あるいはスタンバイ目的
  - スレーブサーバーにマスターサーバーのコピーを作成します。
  - スレーブサーバーを常時接続しておかず、ある時間だけ接続して切断するという運用も可能です。
    - バックアップメディアに落とす時の負荷の軽減
    - 世代管理
  - 他のリライアントソフトと組み合わせれば、 Active-Stanby の構成をとって、障害時に自動的に切り替えることが可能です。



20

レプリケーション使用例 2

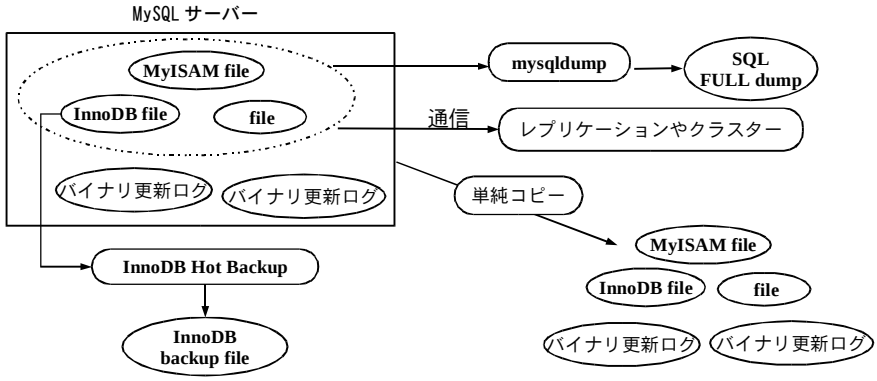
- 負荷の分散
  - スレーブサーバーを大量に作り、検索の負荷を分散させることが可能です。



21

複数のバックアップ

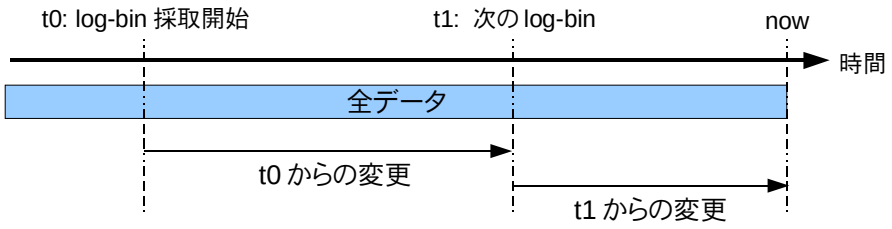
- mysqldump による SQL 文でのダンプ
- レプリケーションやクラスター
- InnoDB Hot Backup
- バイナリ更新ログファイル (ある時点からの変更点の記録)
- ファイルの単純コピー



22

MySQL のログファイル

- バイナリ更新ログ (**`--log-bin`**)
  - ログを取り始める時点の全データに対する変更点だけを SQL 文で記録
- 詳細ログ (**`--log`**)
  - いつどのユーザーが接続してきてどのような操作をしたか詳細に記録
- スロークエリーログ (**`--log-slow-queries`**)
  - 処理に時間のかかったクエリを記録



InnoDB 概要

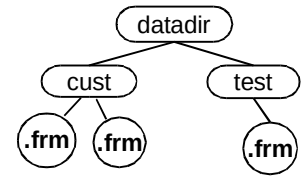
- 行レベルロック
  - 複数のトランザクションが同じレコードを更新、**SELECT** した場合、**SELECT** は待たされない
  - ロックエスカレーションはおきない
- マルチ・バージョニング
- 4 つのトランザクション分離レベル
  - **REPEATABLE READ** が標準
  - ただし、**REPEATABLE READ** において、phantom read は起きない
  - **SET TRANSACTION ISOLATION LEVEL**
  - **--transaction-isolation=**
- Foreign Key
- Hot Backup ツールの存在
- レプリケーションは InnoDB でも動作。

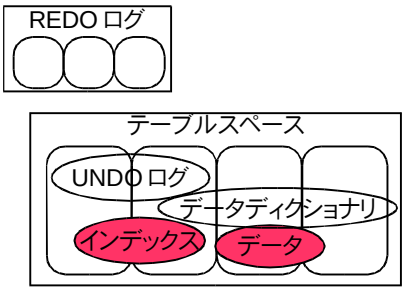
InnoDB ファイル構造

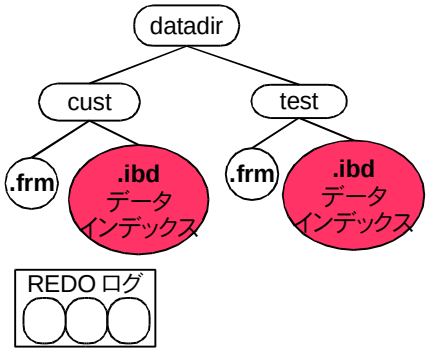
- 2 通りの記録方式
  - 1mysqld につき、どちらか一つの方法だけが採用できる。
  - 全ての InnoDB 型のテーブルを同じ領域（テーブルスペース）に保存するか
  - テーブル単位にデータを別のファイルに保存するか (**innodb_file_per_table**)
- REDO ログはデータ（テーブルスペース）とは別のファイルに記録される
  - 循環して利用される
- テーブルスペースを構成するファイルは、どのディレクトリに置いてもかまわない。また、ばらばらのディレクトリに置いてもかまわない

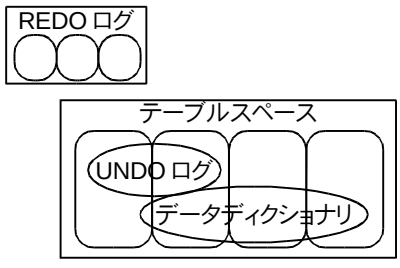
InnoDB ファイル構造 2

- 2 つの違い









InnoDB のレコードの管理

- レコードはセカンダリー・インデックスとクラスタード・インデックスにより素早く見つけられます。
- クラスタード・インデックス
  - プライマリー・キーに従ってレコードを保存しています。
  - インデックスのリーフはレコードです。
    - といっても、レコードの論理的な並び順に、レコードを物理的に記録するわけではありません。また、新たなレコードの挿入のたびにインデックスを全て作成し直すわけでもありません。
- セカンダリー・インデックス
  - セカンダリー・インデックスは、プライマリー・キーの値をインデックス化しています。
- なぜかよく聞かれる質問ですが
  - 追記型ではありません。

InnoDB のレコード

- InnoDB は、レコードの値と共に、各種情報をレコードに付加して保存します。
- Raw ID
- Transaction ID
- Roll Pointer
- 削除フラグ
- ロック情報
- 各フィールドポインター
- 各フィールドの値

InnoDB のバックアップ

- **mysqldump**
- ファイルのコピー
  - my.cnf ファイル、テーブルスペース、 REDO ログ、 .frm ファイル、 .ibd ファイルをコピーします
- InnoDB Hot Backup
  - mysqld を停止する必要はありません
  - InnoDB に関するファイルを読み込んでバックアップファイルを作成します。
  - mysqld を動作させているマシン上で実行します。
  - 有料です (MySQL 本体には含まれておらず、別売されています)

InnoDB のフラグメンテーション

- フラグメンテーションは発生しますが、それほど大きな問題にはなりません。
  - セカンダリーインデックスやクラスタードインデックス、内部構造など
  - もしきれいにしたい場合は、
    - 一度 InnoDB のデータ全てを mysqldump で吸い上げて InnoDB 領域を作成し直すか、
    - **ALTER TABLE** で MyISAM に一度変更して、**ALTER TABLE** で InnoDB に戻すか、
    - InnoDB Hot Backup を使用してバックアップを取り、バックアップから戻す

InnoDB そのほか

- ロックを明示的に指定したい場合は、
  - **SELECT ... LOCK IN SHARE MODE**
  - **SELECT ... FOR UPDATE**
  - **LOCK TABLES** は使用しない
- フィールドの頭の部分にだけインデックスを作成できない
- BLOB, TEXT は 4G 未満であること
- フィールド数は 1000 まで
- 1 レコードの最大長 (BLOB,TEXT は除く) は、 8KByte まで
- mysql データベース以下の権限テーブルを InnoDB 型にしてはいけない

31

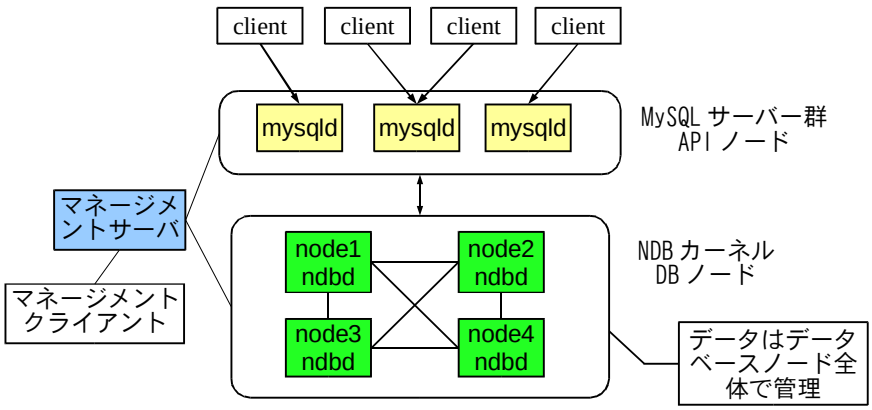
NDB クラスター概要

- Alzato から MySQL AB に
- MySQL 4.1 のソースに組み込まれています。
-
- データはメモリー内にあります。
  - 高速な動作
  - ログ (データベースのイメージ) は **disk** に記録されています
- **NDB** 型のテーブル (ストレージエンジン) 。 **TYPE=ndbcluster**
- **HotBackup** 機能
- 障害時のノードの切り替えは一瞬。
- ノードのメンテナンスは楽。
  - ノードを落としてまたたち上げるだけで復旧。
-
- **NDB** 型は **READ COMMITTED** のみ。
- 行レベルロック

32

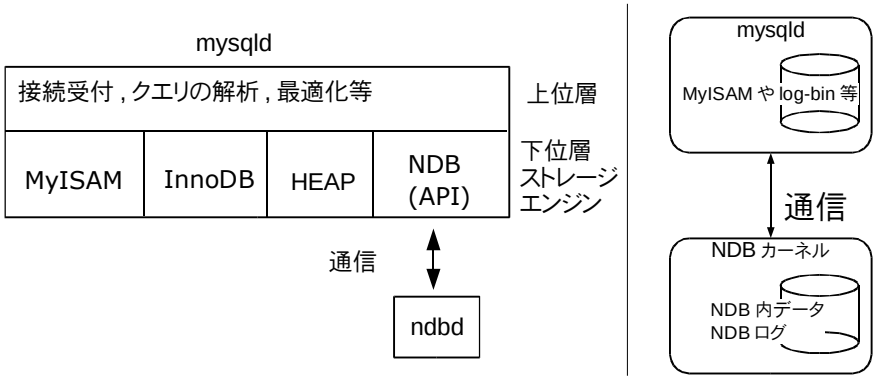
NDB クラスター構造

- 3 層。それぞれの層でスケールアップ可能
- 1DB ノード = 1ndbd
- 1API ノード = 1mysqld(etc...)
- 各ノードはどのハードにインストールしてもかまわない



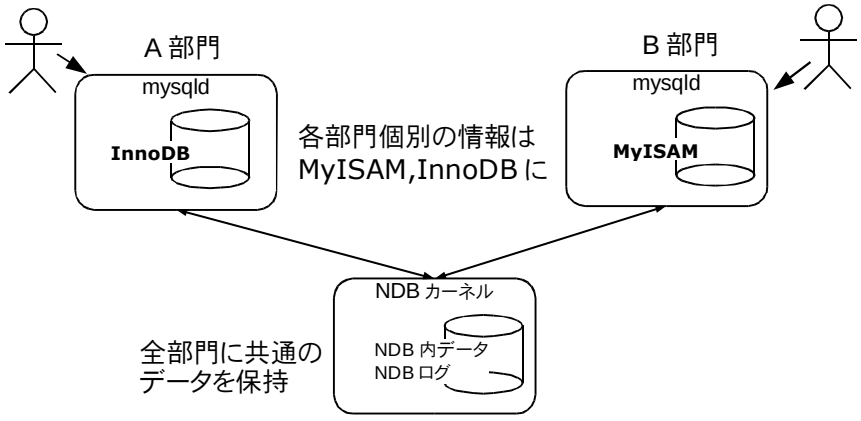
NDB クラスター構造 2

- mysqld にしてみれば、 ndbd は別プロセス
- ndbd にしてみれば、 mysqld は NDB API をもったソフト
- MyISAM,InnoDB は個々の mysqld が管理
- NDB のデータは、全ての mysqld,ndbd で共通



NDB クラスターのちょっとした使い方

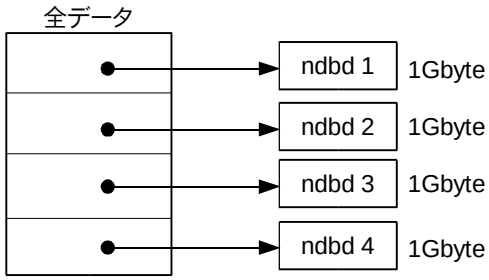
- たんなる冗長性、拡張性、可用性、高速性以外の目的での使い方
- 以下の特徴を逆手にとって、データアクセスの分離
  - MyISAM,InnoDB は個々の mysqld が管理
  - NDB のデータは、全ての mysqld,ndbd で共通



35

Data Fragment

- 全てのデータ (NDB 型のデータ) は、全ての DB ノードに分割されて保持管理されます。
- メリット
  - 各 ndbd に分割することで、素早い動作が期待できる。
  - ndbd 搭載のマシンを増やすだけで、使えるメモリー量 (保存できるデータ量) が増える
- データベースのイメージログは各 ndbd が動作している機械上に作成されます。



36

Data Replica

- ある ndbd が管理しているデータは、別の ndbd に複製されます。
  - 安全性が高まります。
  - ndbd が仮に一つおちたとしてもサービスは続けられます。
    - primary と secondary の切り替えは一瞬
  - ndbd のメンテナンス時もサービスが提供可能になります。
- 同じデータを持っている複数の ndbd を、node group と呼びます
- コピーは同期