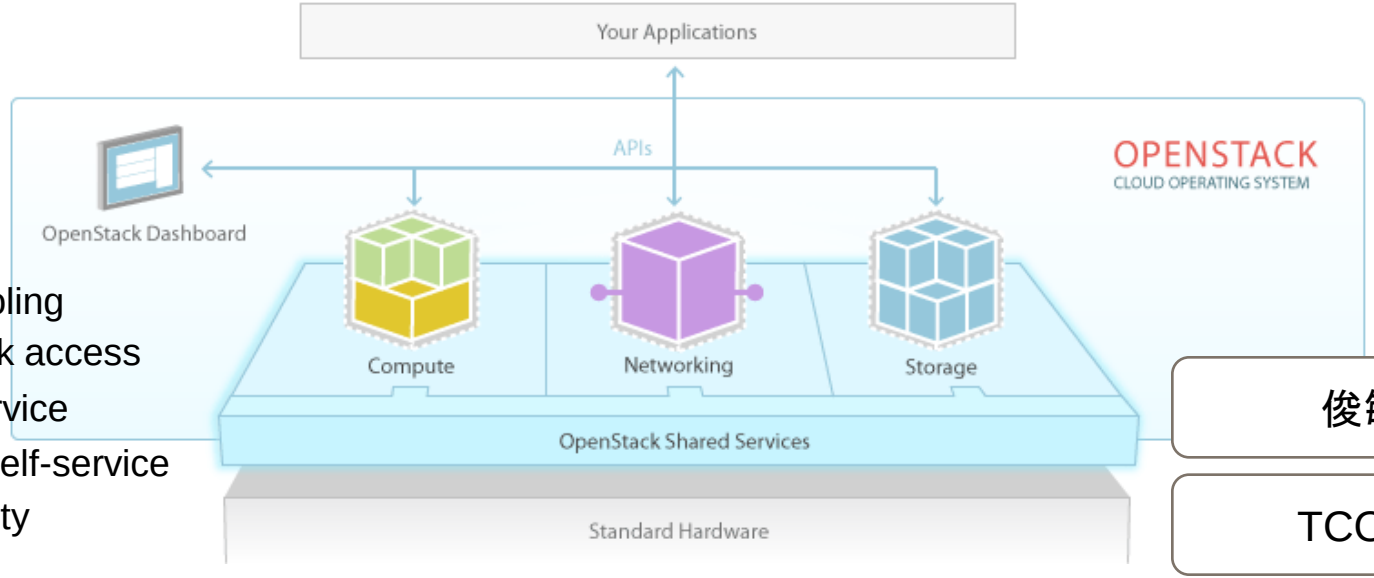
OpenStackにおけるMySQLの活用

Shinya Sugiyama / 杉山真也

MySQL Principal Sales Consult, MySQL Global Business Unit

OpenStackとは?

- OpenStackは大規模・マルチテナントクラウド環境への導入を目的とした
  オープンソース・クラウド・オペレーティングシステムです
- OpenStackはコンピュート・ストレージ・ネットワーク・アイデンティティ管理・オーケストレーション等
  を提供する分散サービス群で構成されています



Resource pooling
Broad network access
Measured service
On-demand self-service
Rapid elasticity

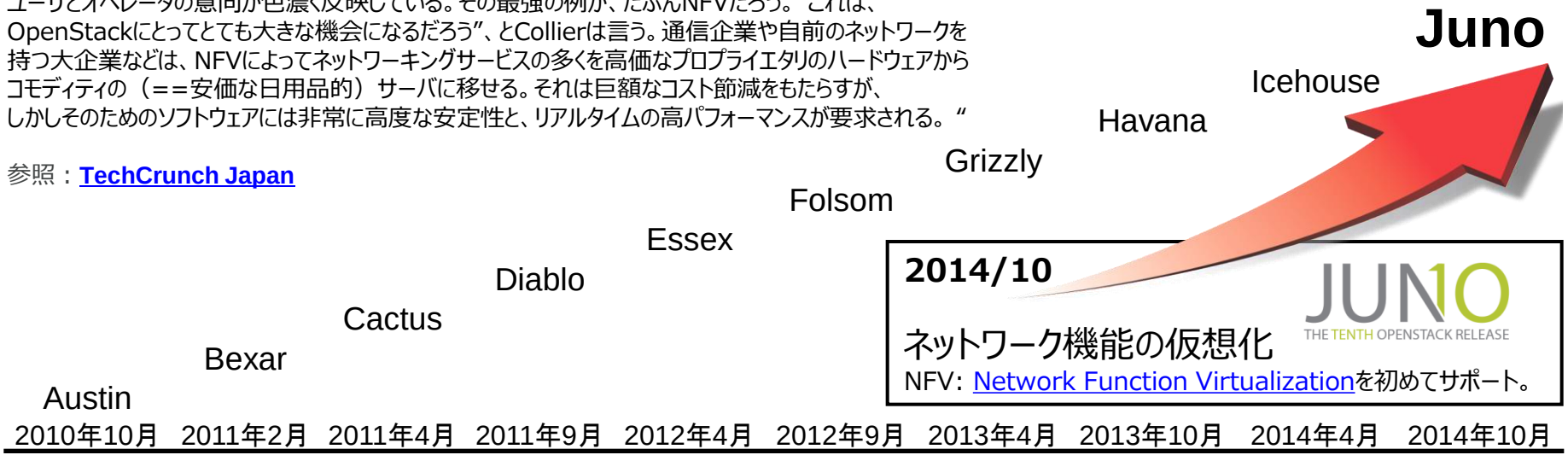
俊敏性

TCO削減

OpenStack's Juno Release Focuses on Big Data, Network Function Virtualization

"OpenStack FoundationのCOO Mark Collierによると、このリリースにはOpenStackのユーザとオペレータの意向が色濃く反映している。その最強の例が、たぶんNFVだろう。"これは、OpenStackにとってとても大きな機会になるだろう"、とCollierは言う。通信企業や自前のネットワークを持つ大企業などは、NFVによってネットワーキングサービスの多くを高価なプロプライエタリのハードウェアからコモディティの（＝＝安価な日用品的）サーバに移せる。それは巨額なコスト節減をもたらすが、しかしそのためのソフトウェアには非常に高度な安定性と、リアルタイムの高パフォーマンスが要求される。"

参照：[TechCrunch Japan]



NFV Subteam Focuses on Networking for Large Deployments

NFV represents a massive shift in how many networking and telco services are developed and deployed. An NFV development team was formed in May at the OpenStack Summit and has identified nine use cases to run NFV workloads on top of OpenStack environments. Initial features arrived in the Juno release, and additional NFV-related work will continue over coming releases.

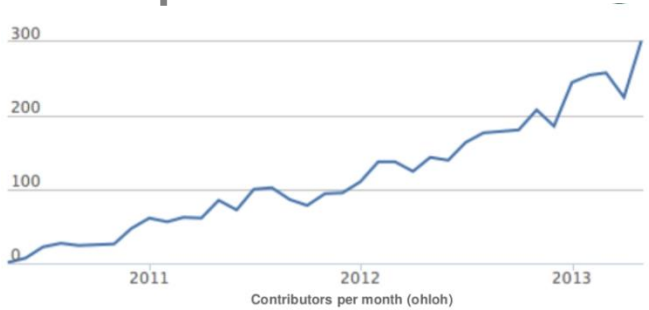
参照：[openstack.org]

OpenStack Momentum

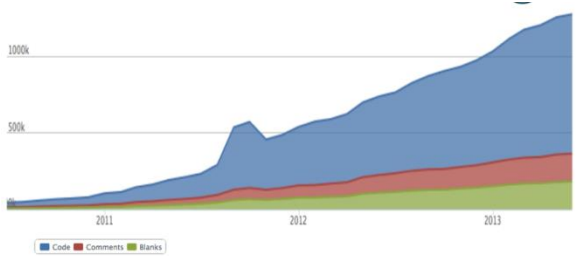
Community Growth

COMPANIES 231	COUNTRIES 121
INDIVIDUAL MEMBERS 10,149	
TOTAL CONTRIBUTORS 1,036	AVERAGE MONTHLY CONTRIBUTORS 238
CODE CONTRIBUTIONS 70,137	

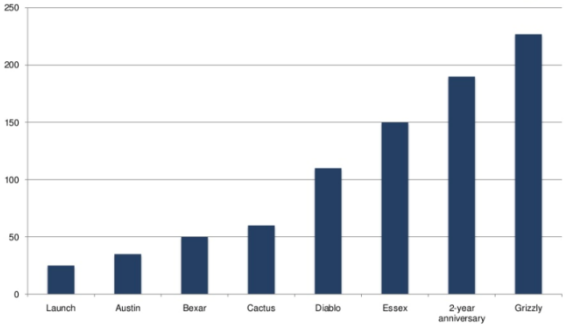
Developer Growth



Code Growth: 1M+ Lines



Participating Company Growth



参照： OpenStack

OpenStack Use Cases

- プライベートクラウド
  - オンプレミス
  - ホスティング
- パブリッククラウド
  - HP Public Cloud
  - Rackspace
- ハイブリッドクラウド
  - Rackspace APIs
  - AWS APIs
  - GCE APIs

“Enterprises should design private cloud services with a hybrid future in mind and make sure future integration & interoperability is possible.”

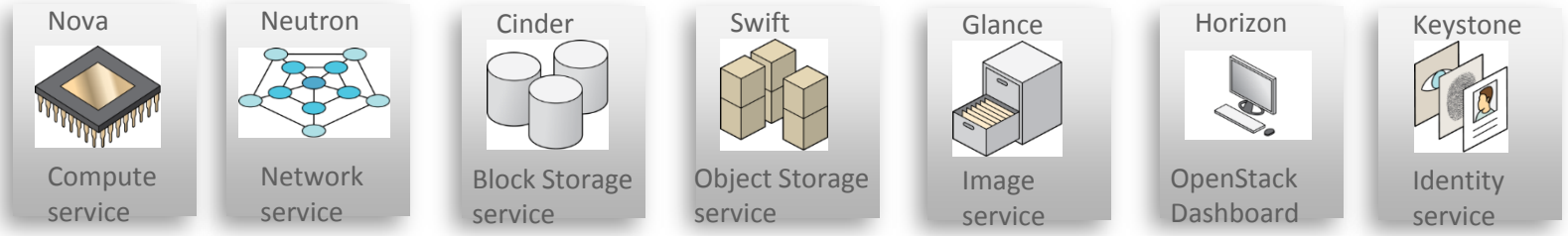
Gartner Group

その他; Think IT , OpenStackユーザ会 他

Oracle OpenStack for Oracle Linux & VM 提供開始

[Press Release: 2014 10 08]

- Oracle Linux と Oracle VMによるOpenStackクラウド本番環境の構築と管理
  - 高速、信頼性、スケーラブル、安全性
  - Linux, Windows, Solaris上で本番環境のワークロードを実現
- 無料ダウンロードおよび利用可能
- 「Oracle Linux and Oracle VM Premier Subscription」
  「Oracle Premier Support for Systems」が追加費用なしでサポートサービス提供
- OpenStackリリースIcehouseベースで以下を提供

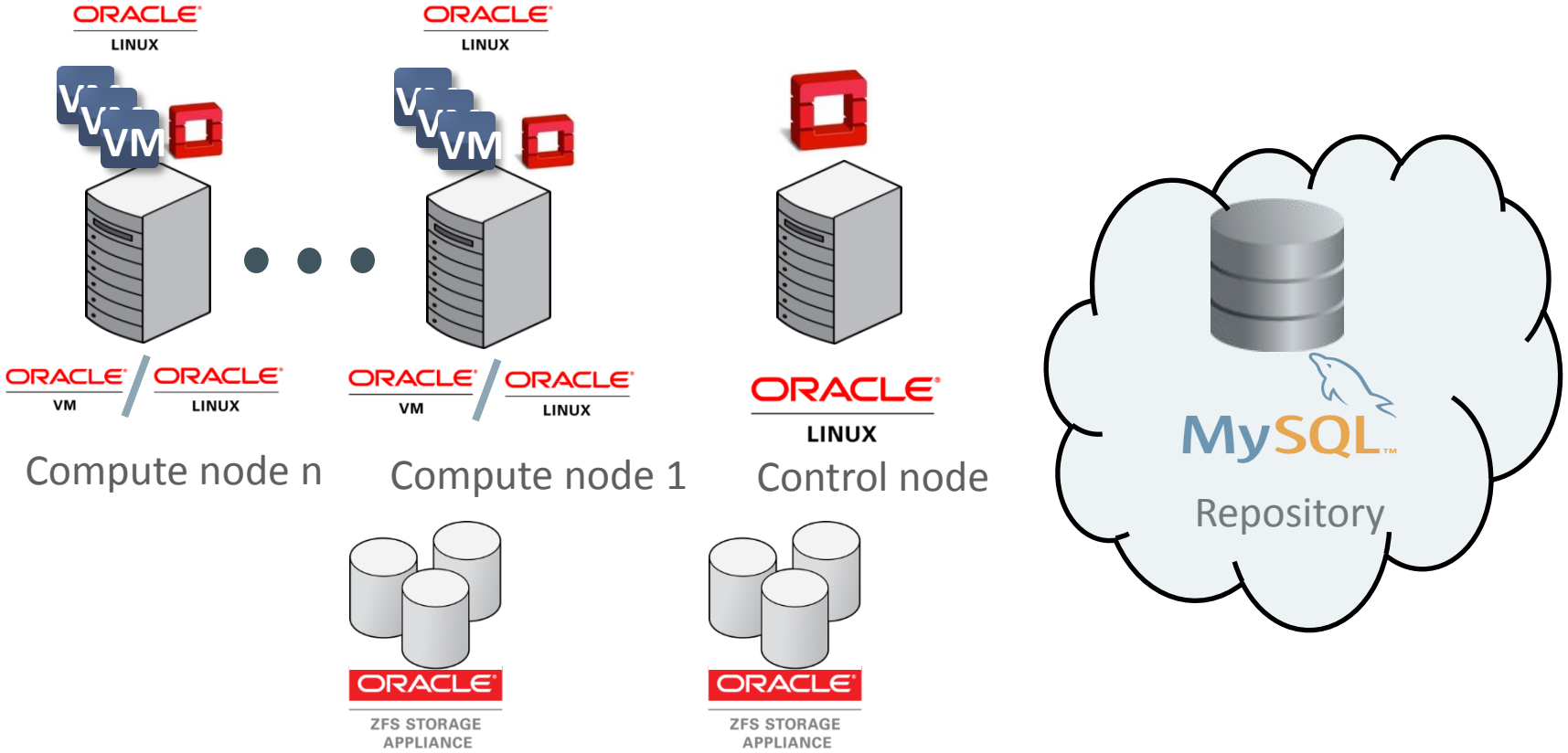


OpenStackに拡張された Oracleエンタープライズ・エンジニアリング

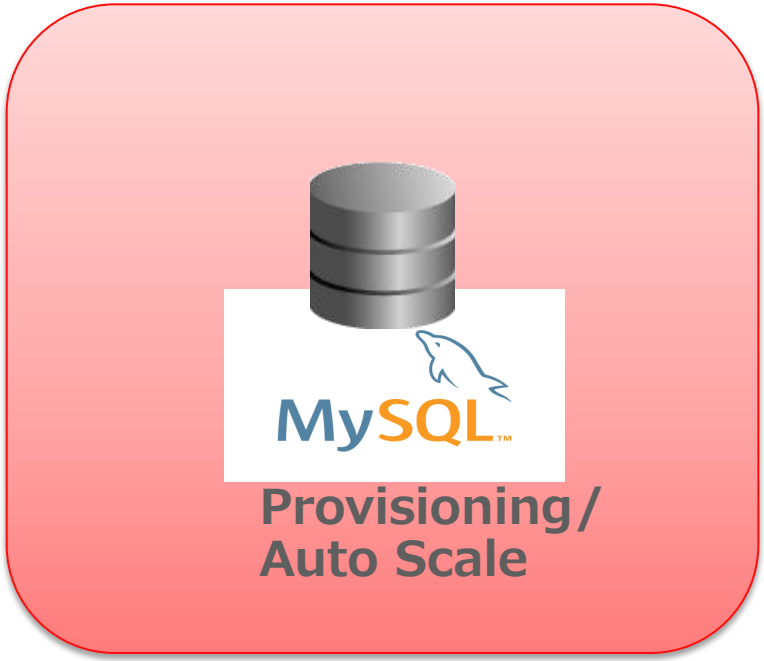
- MySQL Enterprise Editionとの統合が高スケーラビリティ・安全性・アップタイムの最大化を実現
- エンタープライズ級ストレージの実現を容易にするZFS Storage Appliance向けCinderプラグイン
- Oracleとサードパーティ・ハードウェアの両方で動作可能
- 最高レベルの性能と信頼性を実現するオラクル・エンジニアド・システムがOracle LinuxとOracle VMを強化

「Oracle OpenStack for Oracle Linux and VM」が OpenStack Cloudの優位性を実現

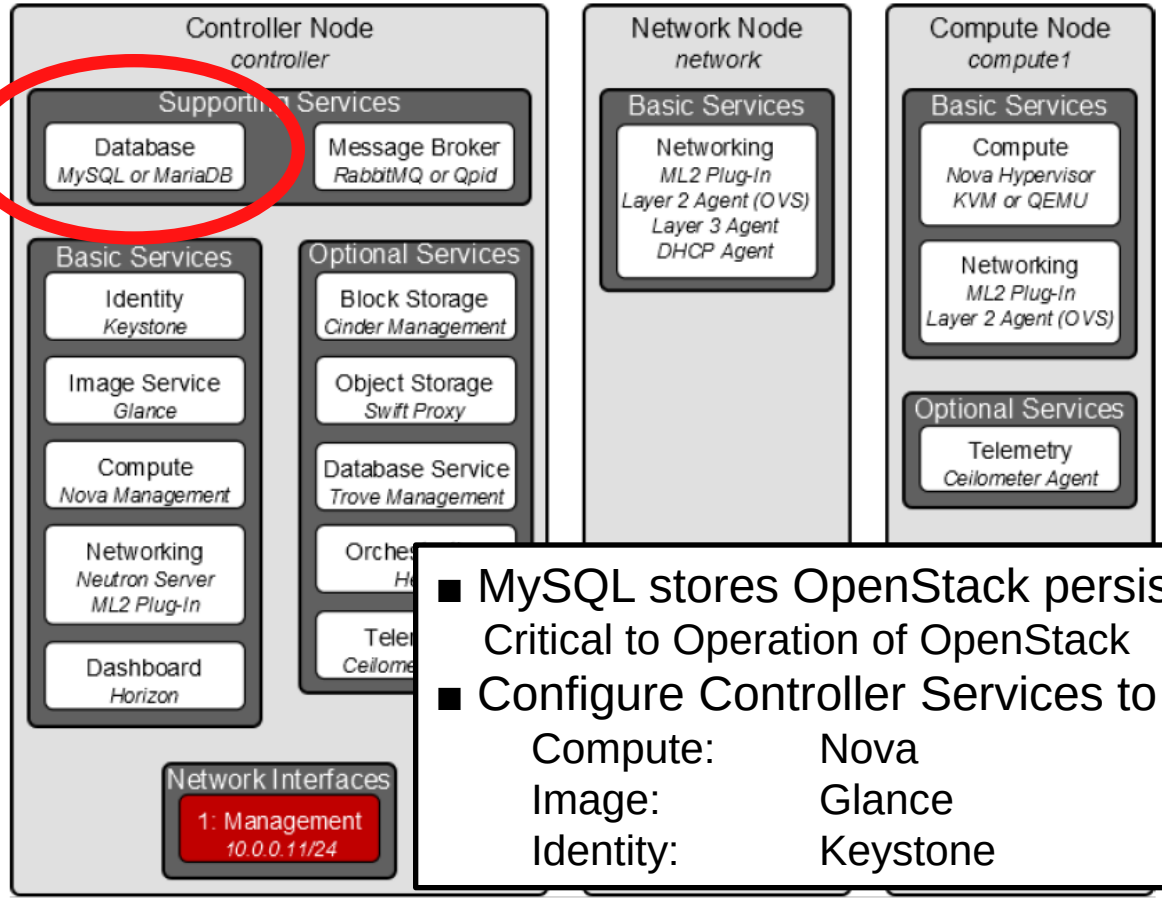
OracleがOpenStackクラウドの全パーツをサポート可能



OpenStackにおける、MySQLの主な役割
Controllerリポジトリ/Auto Scale & Provisioning



OpenStackのリポジトリーとして



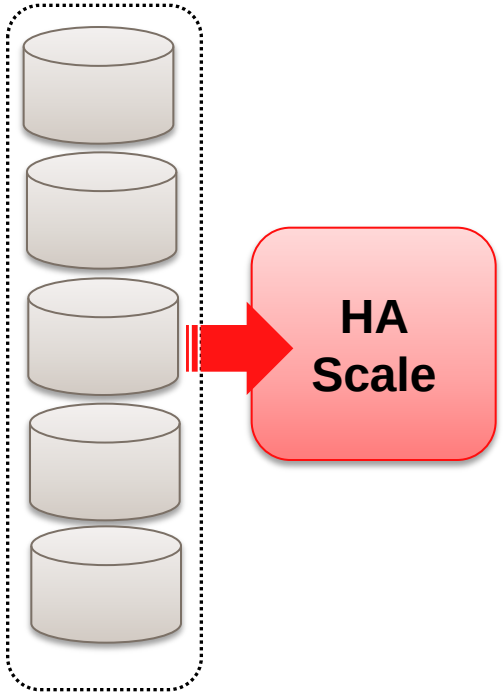
- MySQL stores OpenStack persistent data.
  Critical to Operation of OpenStack
- Configure Controller Services to use MySQL.
  - Compute: Nova
  - Image: Glance
  - Identity: Keystone

MySQL in OpenStack

OpenStackは可用性、コスト、性能、拡張性、管理性を必要とします。
MySQLは、管理情報リポジトリDBとして使用されています。

Keystone	認証
Nova	IaaS (例：EC2)
Cinder	ブロックストレージ
Glance	VMイメージ管理
Neutron	仮想ネットワーク管理

```
+--------------+------------------+
| TABLE_SCHEMA | Number of tables |
+--------------+------------------+
| cinder       |               19 |
| glance       |                8 |
| keystone     |               16 |
| nova         |              108 |
| ovs_neutron  |               29 |
+--------------+------------------+
5 rows in set (0.01 sec)
```



Those DBs are metadata, so not so high; however,
if data is increase update process will be slow down.
In that case ,delete old data or divide into partitions.
HA is one of the most important point.

管理情報リポジトリDB

基本的には、構成管理用のリポジトリとしての役割がメインなので、大規模なシステムを管理しない限りは現状それ程大きなデータベース負荷は発生しない事が想定される。しかし、Troveのように、データベースに継続的にアクセストークンを蓄積し続けるケースもあるので、データ量が多くなっても、処理が重たくならないようにするか、定期的に削除するなどの運用を検討する必要があります。

また、Ceilometer等を利用して課金の為などに、メータリング情報を収集する場合は、規模によってはインサートや参照に性能課題が発生するかもしれません。

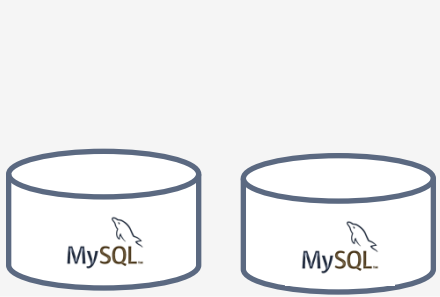
※MySQLの機能では、Table Partitioningも選択肢として検討しても良いかもしれません。

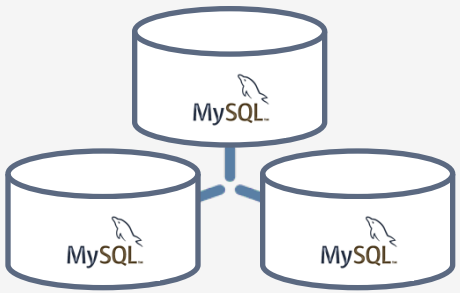
```
mysql> desc token;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| id       | varchar(64) | NO   | PRI | NULL    |       |
| expires  | datetime    | YES  | MUL | NULL    |       |
| extra    | text        | YES  |     | NULL    |       |
| valid    | tinyint(1)  | NO   |     | NULL    |       |
| trust_id | varchar(64) | YES  |     | NULL    |       |
| user_id  | varchar(64) | YES  |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
6 rows in set (0.01 sec)

mysql> select id,expires from token limit 0,10;
+----------------------------------+---------------------+
| id                               | expires             |
+----------------------------------+---------------------+
| 80c91125db88167f997c066eb8cc42e8 | 2014-11-26 06:59:50 |
| 030039767e1322c2044cf7a027c09e17 | 2014-11-27 01:26:07 |
| 05d5171229881d674b470dc2f384eeec | 2014-11-27 01:26:07 |
| 068869123f444ce2d0b80367443fe4f9 | 2014-11-27 01:26:07 |
| 07cfa8a6398417c731fa66e8df49ce81 | 2014-11-27 01:26:07 |
| 084b056e3cf9c00b90cac959506ab311 | 2014-11-27 01:26:07 |
| 084be2411e04da737c0ef917c477678a | 2014-11-27 01:26:07 |
| 0898f22665f48639a9456c61dd7547e2 | 2014-11-27 01:26:07 |
| 0a470b6703c8742750fec2720aad9375 | 2014-11-27 01:26:07 |
| 0bb52abeb199e440ccd9d46c69a94b6b | 2014-11-27 01:26:07 |
+----------------------------------+---------------------+
10 rows in set (0.00 sec)

mysql>
```

MySQLリポジトリのHA構成

```
$ nova list
+--------------------------------------+------------------+-----------+-------------- +----------------+------------------------------+
| ID                                   | Name             | Status    | Task State   | PowerState     | Networks                     |
+--------------------------------------+------------------+-----------+-------------- +----------------+------------------------------+
| 05682b91-81a1-464c-8f40-8 | demo-ins1      | ACTIVE |      -       | Running      | demo-net=192.168.1.3  |
+--------------------------------------+------------------+-----------+-------------- +----------------+------------------------------+
```

DRBD	MySQL Fabric	MySQL Cluster



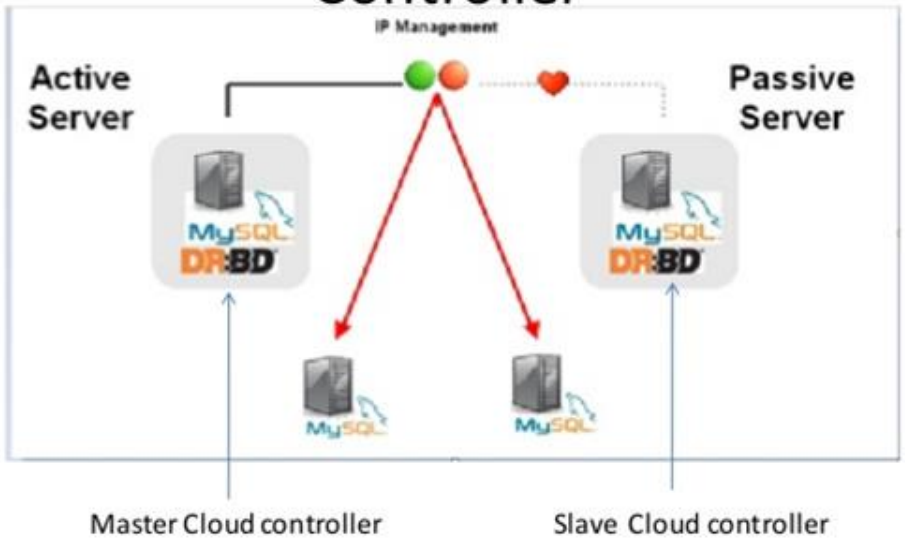


MySQL Cloud Controller: High Availability

DRBD

Clustering Mode	Auto-Failover	Failover Time	Scale-out
Active/Passive	**Yes**	**Secs +**	**No**

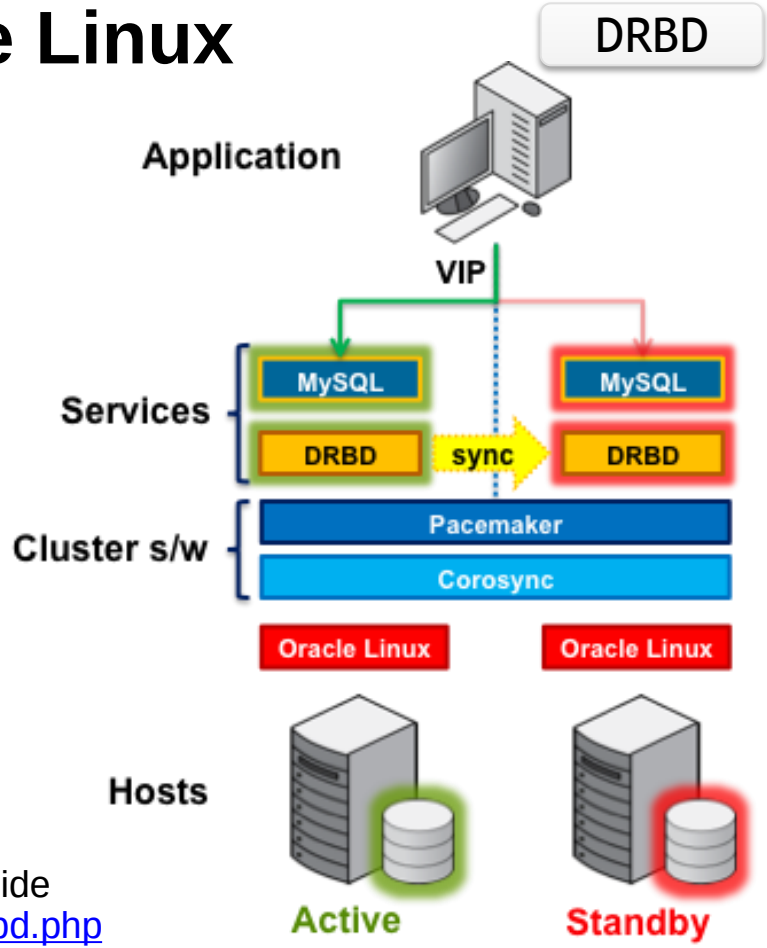
DBRD Architecture – MySQL-Cloud Controller



High Availability with Oracle Linux

Oracle Linux ＋ DRBD Stack

- 認定構成だからこそ実現できる、
  Oracleによるフルスタックサポート
  - Oracle Linux Unbreakable Enterprise
    Kernel R2に統合されたDRBD
  - Oracle Linux 6.2以上で使用可能
  - クラスタリングとフェイルオーバーのために、
    PacemakerとCorosyncを使用
- 分散ストレージを利用するため、共有ディスクやSAN不要
- 同期レプリケーションによってデータを失うリスクを回避
- オープンソースで実績の多いソリューション



※ホワイトペーパー：DRBD - Configuration and Deployment Guide
http://www.mysql.com/why-mysql/white-papers/mysql_wp_drbd.php

Configuring MySQL for HA on OpenStack

DRBD

Using DRBD, Pacemaker, Corosync

- MySQLを使用するようにDRBDを設定
- DRBDのデータディレクトリを使用するようにMySQLを構成
- Clusterノード間で共通のVIPを利用するように設定
- MySQLはVIPでListenするように設定
- OpenStack関連サービスはMySQLに設定されたVIPに接続

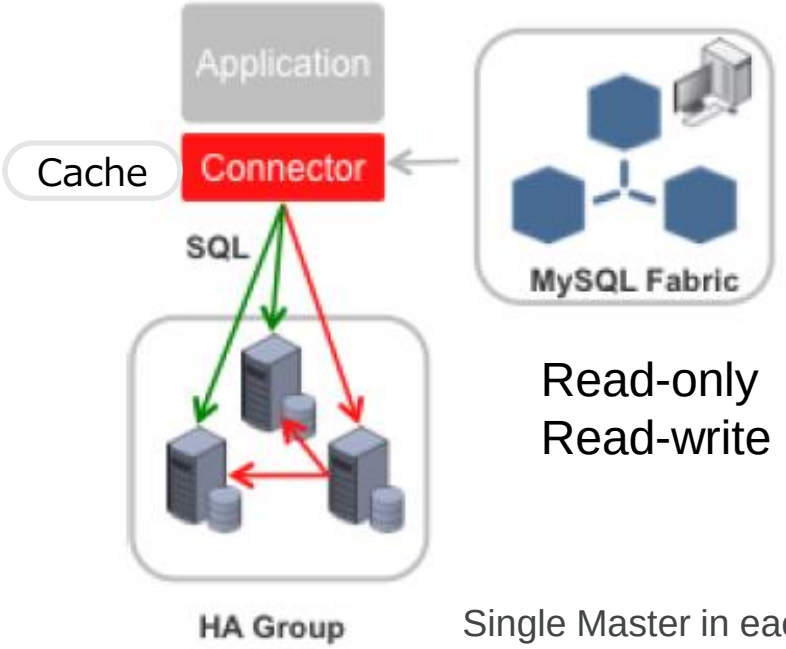
※　Scale Outは出来ないのでスケールアップで対応

MySQL Utilities - Fabric

Fabric

Clustering Mode	Auto-Failover	Failover Time	Scale-out
Master + Slaves	**Yes**	**Secs**	**Yes**

Scale-out using Sharding / Automatically Change Master



Read-only
Read-write

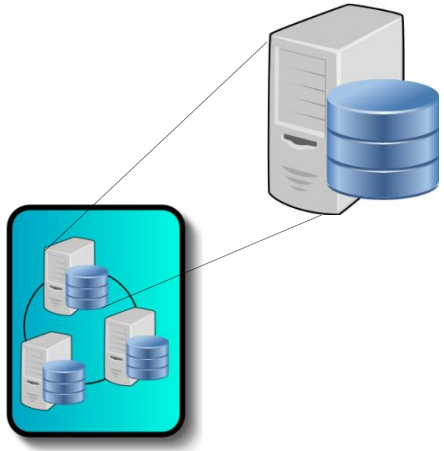
Single Master in each HA Group.

- 「フェールオーバー」と通信経路の自動再構成による高可用性
- 「シャーディング」による拡張性
- コネクタ
  - Python (Fabric 1.4)
  - Java (Fabric 1.4)
  - PHP (Fabric 1.4)
  - .NET (Fabric 1.5)
  - C (Fabric 1.5)
- アプリケーションでの分割キー
  - Range または Hash
  - シャード再構成も可能
  - シャード全体の更新も可能
- MySQL Utilities 1.4として提供

High-Availability Group Concept

Fabric

- **抽象概念**
  - サーバーセット
  - サーバー属性
- **コネクター属性**
  - 接続情報
  - モード:読み取り専用、読み書き、…
  - 重み付け: 負荷分散
- **管理属性**
  - **状態:サーバの状態/役割**



State: Primary
Mode: **Read-Write**
Host: server-1.example.com

server_uuid	address	status	mode	weight
83f2dd4f-46a1-11e4-94c4-e82aea9348c9	localhost:13007	SECONDARY	READ_ONLY	1.0
8426dd92-46a1-11e4-94c4-e82aea9348c9	localhost:13008	PRIMARY	READ_WRITE	1.0

MySQL Replication & MySQL Fabric HA

Fabric

& how this effects failover

- MySQLのレプリケーションは、HAグループで使用される初期実装です
  - PRIMARY = レプリケーションのマスターは全ての書き込みを受け取る
- Failover

  1）Fabricがマスターにおける障害を検知

  2)スレーブからマスター候補を選択しマスターに昇格します

  3)変更を更新し状況を保存

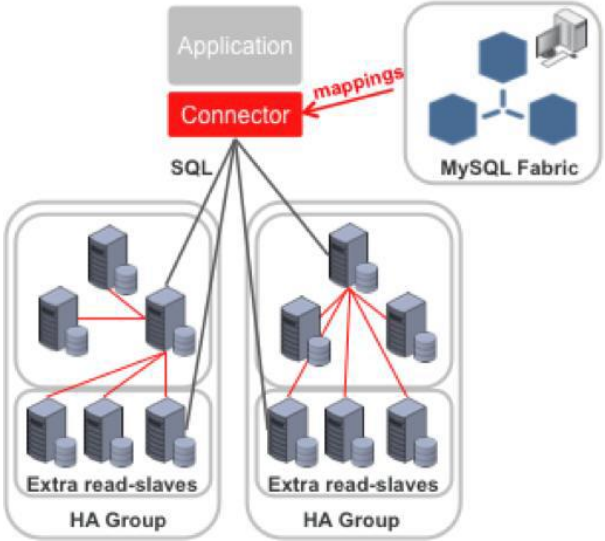
  4）ファブリック対応のコネクタに状態変化をプッシュ

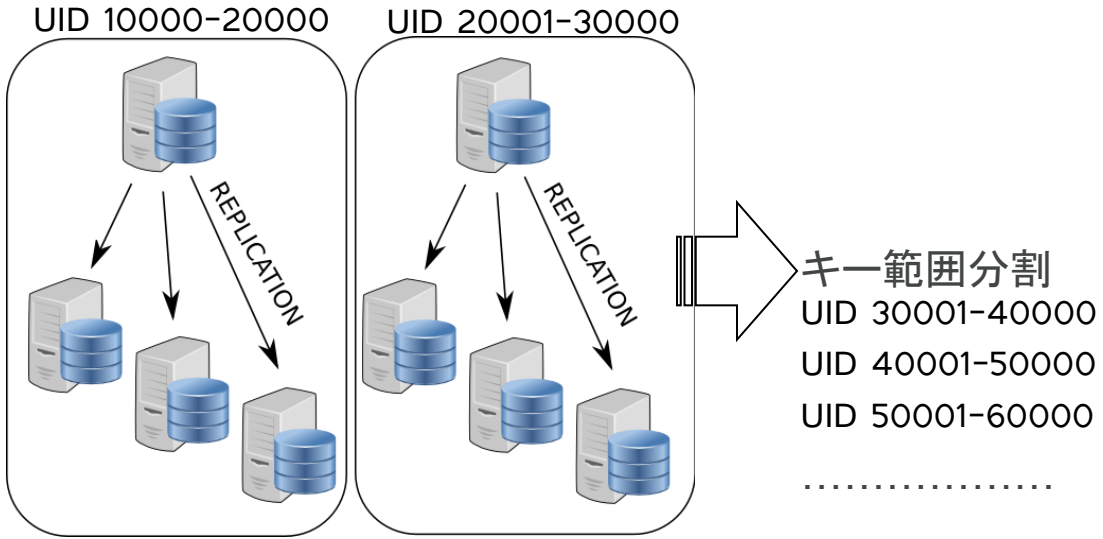
MySQL Fabricによる
データ可用性とスケールアウト

Sharding with Fabric

Fabric

- 書き込みスケーラビリティ　より多くの書き込みを処理することが可能
- 大規模なデータセット　大き過ぎるデータベース/単一サーバーに収まらないデータ
- 性能改善　小さなインデックスサイズ/ワーキングセットに分割





Connector API: Shard Specific Query

Fabric

- Provide tables in query
  - **Property:** tables
  - Fabric will compute map
- Provide sharding key
  - **Property:** key
  - Fabric will compute shard

```
conn.set_property(tables=['employees.employees','employees.titles'] key=emp_no)
cur = conn.cursor()
cur.execute("INSERT INTO employees VALUES(%s,%s,%s)",(emp_no,first_name,last_name))
cur.execute("INSERT INTO titles(emp_no, title, from_date)"
            " VALUES (%s, %s, CURDATE())",(emp_no, 'Intern'));
conn.commit()
```

Connector API: Global Update

Fabric

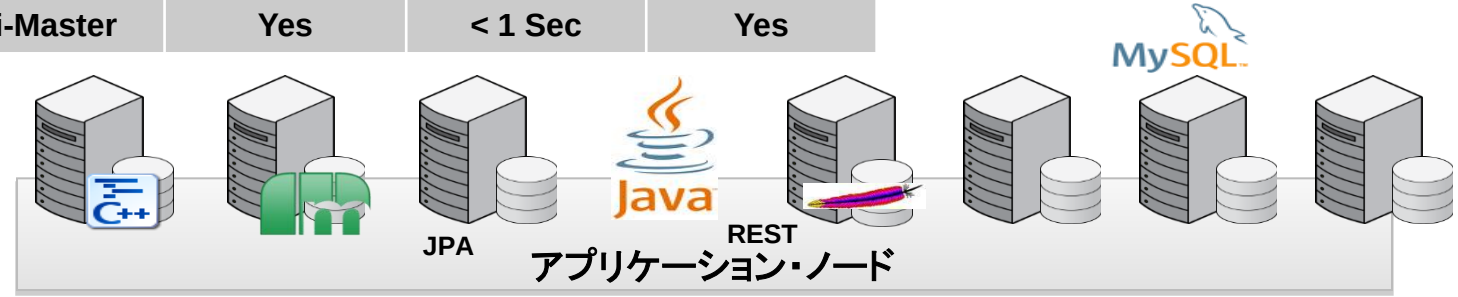
- Provide tables in query
  - **Property:** tables
  - Fabric will compute map
  - (Likely to not be needed)
- Set global scope
  - **Property:** scope
  - Query goes to global group

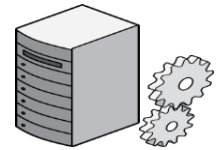
```
conn.set_property(tables=['employees.titles'], scope='GLOBAL')
cur = conn.cursor()
cur.execute("ALTER TABLE employees.titles ADD nickname VARCHAR(64)")
```

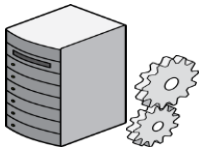
MySQL Cluster アーキテクチャ

Cluster

Clustering Mode	Auto-Failover	Failover Time	Scale-out
Multi-Master	Yes	< 1 Sec	Yes

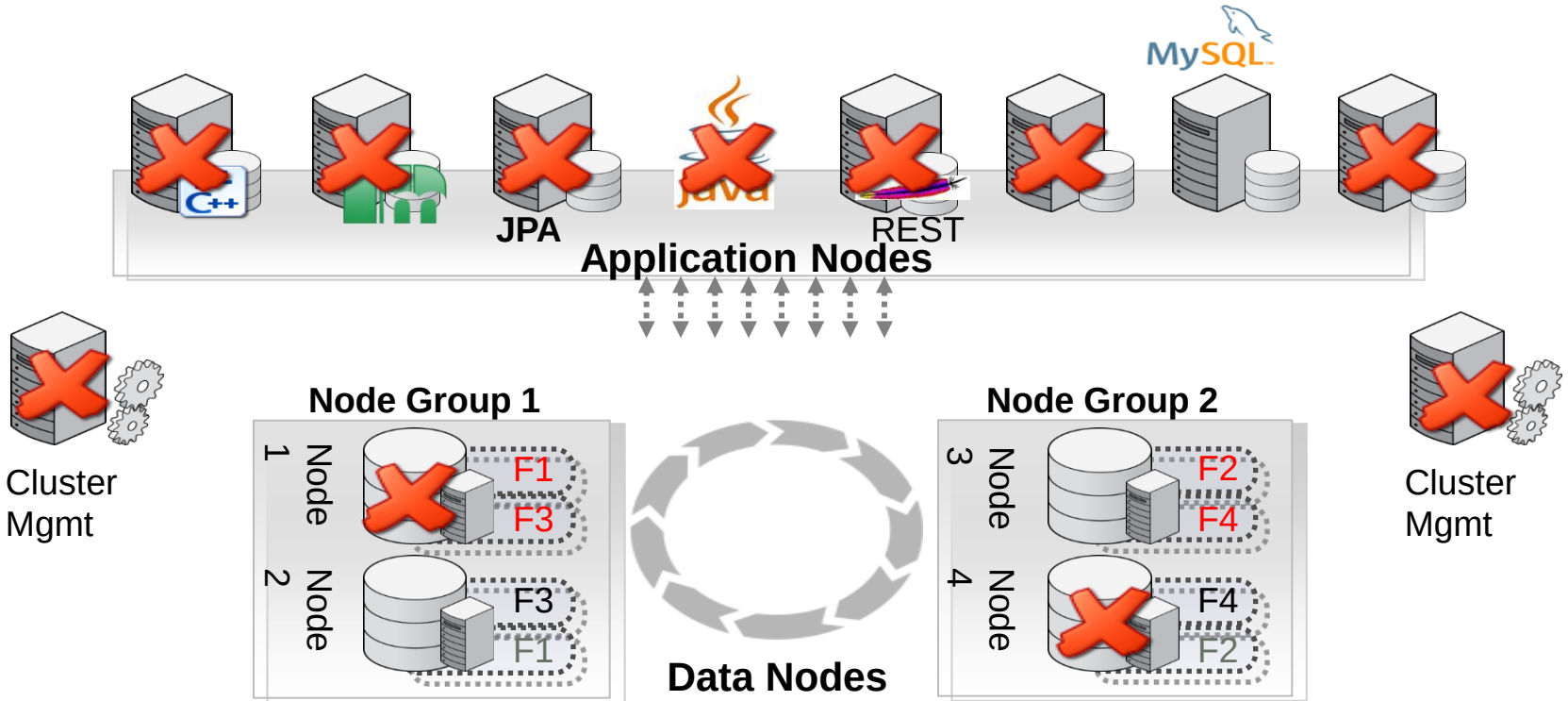






MySQL Cluster - Extreme Resilience

Cluster



MySQL Cluster の特徴

Cluster

参照更新性能の 高い拡張性	• 自動シャーディング、マルチマスタ • ACIDトランザクション、OLTPとリアルタイム分析
99.999% の可用性	• シェアードナッシング、単一障害点無し • 自動復旧、オンラインメンテナンス
リアルタイム	• インメモリ処理に最適化＋ディスク併用可能 • 低レイテンシ
SQL + NoSQL	• キー・バリュー型＋複雑なリレーショナルな処理 • SQL+Memcached+JavaScript+Java+HTTP/REST&C++
低コスト	• オープンソース＋商用版運用支援ツール • 特殊なハードウェア不要、管理監視ツール群、サポート

MySQL Clusterとは？

参照：MySQL Cluster Evaluation Guide

Cluster

- MySQLとは開発ツリーの異なる別製品
- 共有ディスクを使わずに、アクティブ-アクティブのクラスタ構成が組める
- 元々はSQLを使わないデータベースだったが、MySQLと統合されSQLも使えるようになった(NoSQL(KVS)とSQLの両方が使えるデータベース)
- AQL(Push Down Join)によりJOINの処理が高速化された
- 各テーブルのストレージエンジンを選択する事が出来る(InnoDB or NDB)

```
mysql> show variables like 'ndb_join_pushdown';
+-------------------+-------+
| Variable_name     | Value |
+-------------------+-------+
| ndb_join_pushdown | ON    |
+-------------------+-------+
1 row in set (0.00 sec)
```

適したシステム

- 高可用性が求められるシステム
- 同時多発的に大量のトランザクションが発生するシステム
- 読込み処理だけでなく、書込み処理に対しても拡張性が求められるシステム

Cluster

Handling Scheduled Maintenance

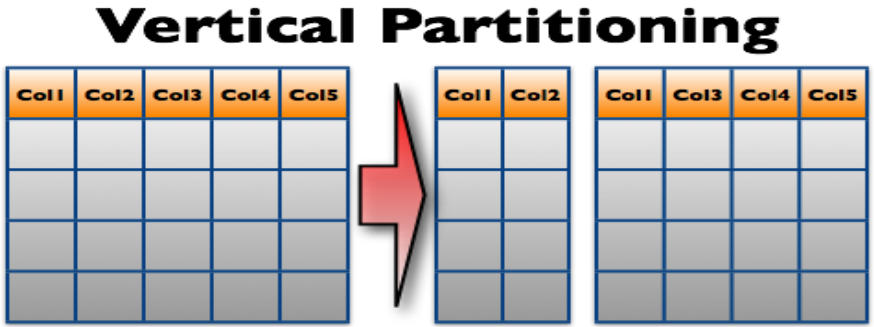
オンラインオペレーション

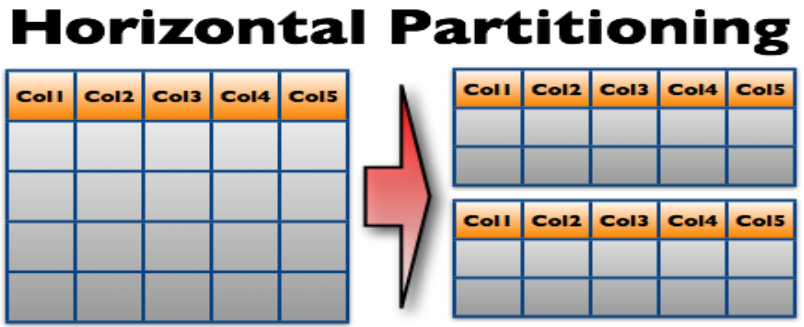
- クラスターのスケール(ノードの追加 & 削除)
- テーブルの再分割
- OSのアップグレードやパッチの適用
- MySQL Clusterのアップグレードやパッチの適用
- バックアップ
- リアルタイムでのオンラインスキーマ変更

MySQL Clusterによる
データ可用性とスケールアウト

Partitioning

- 垂直分割- 行, テーブルとインデックスのサイズを縮小
- 水平分割- 一つのテーブルを異なる行を持つ複数のテーブルに分割





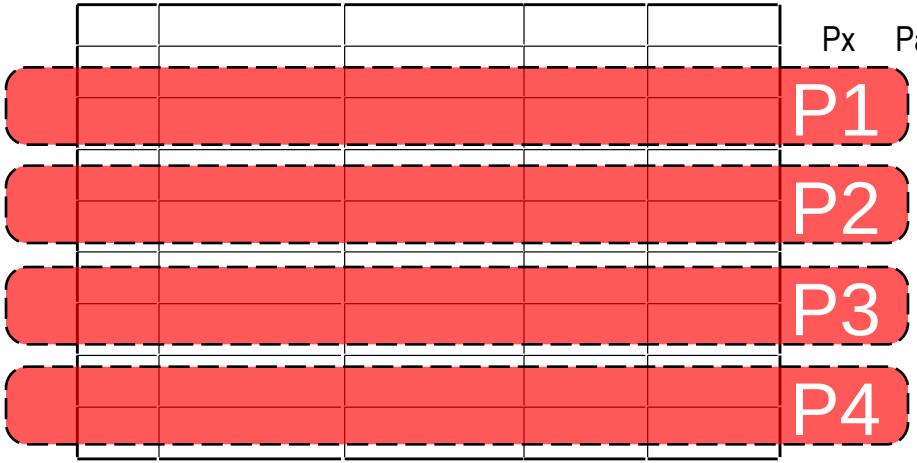
Automatic Data Partitioning

Nodes & Node Groups

Cluster

4 Partitions * 2 Replicas = 8 Fragments

Table T1



Data Node 1

Data Node 2

Data Node 3

Data Node 4

A fragment is a copy of a partition

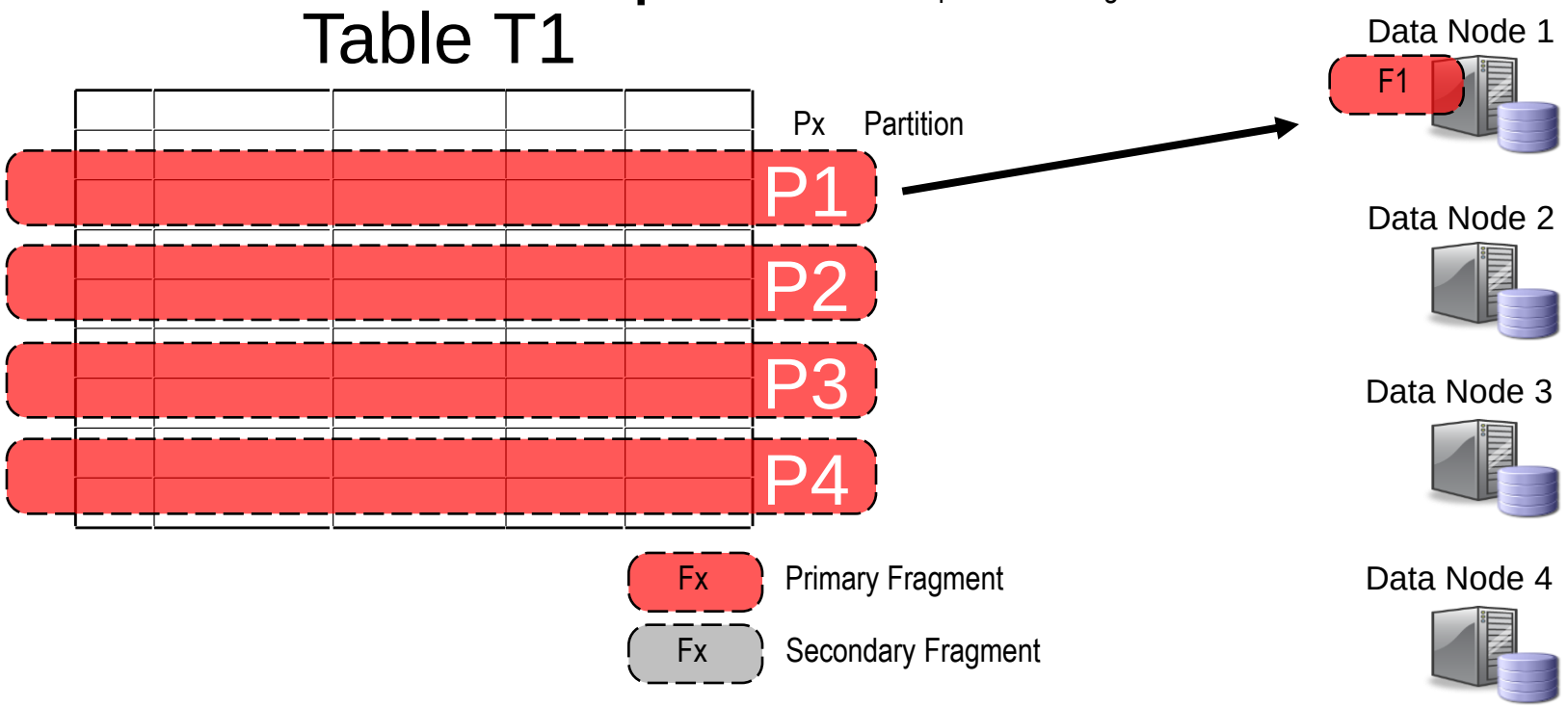
Number of fragments = # of partitions * # of replicas

Automatic Data Partitioning

Cluster

Nodes & Node Groups

4 Partitions * 2 Replicas = 8 Fragments

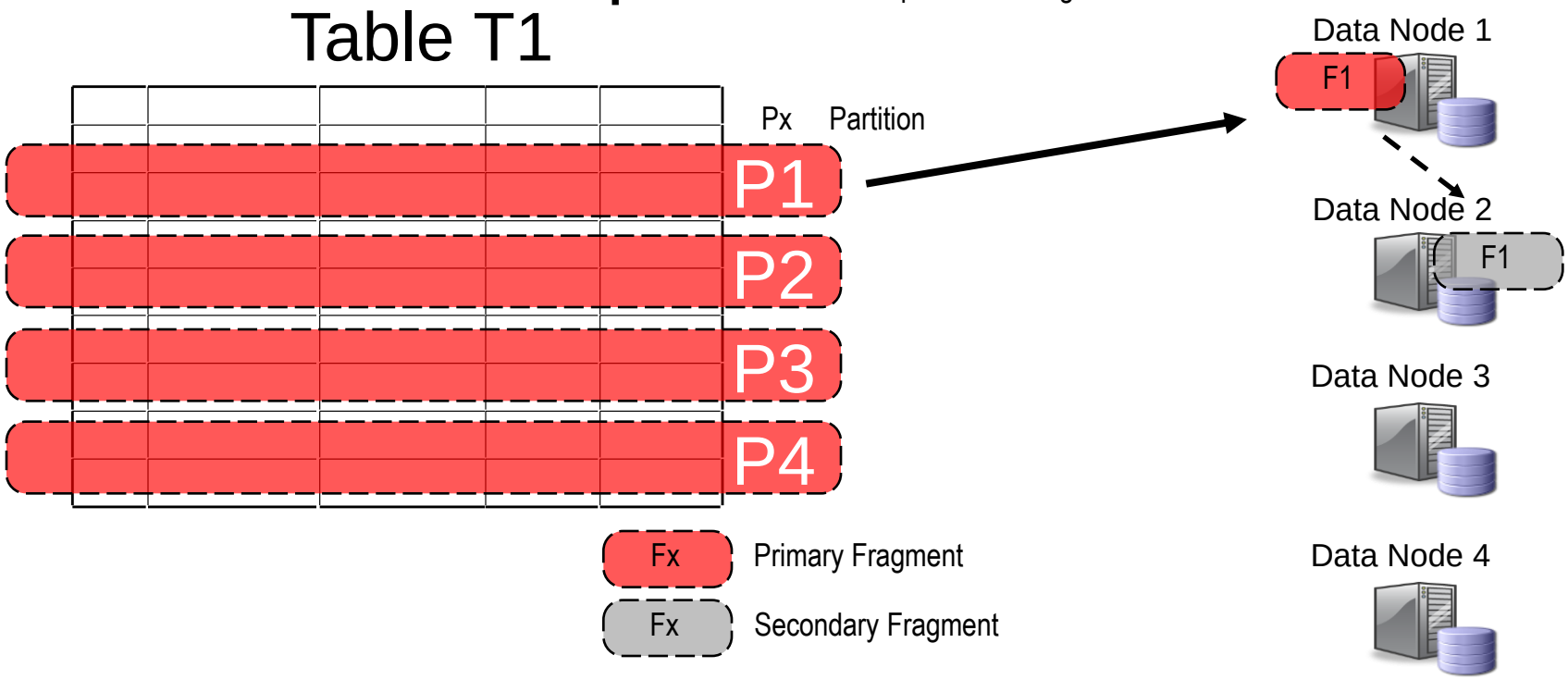


Automatic Data Partitioning

Cluster

Nodes & Node Groups

4 Partitions * 2 Replicas = 8 Fragments

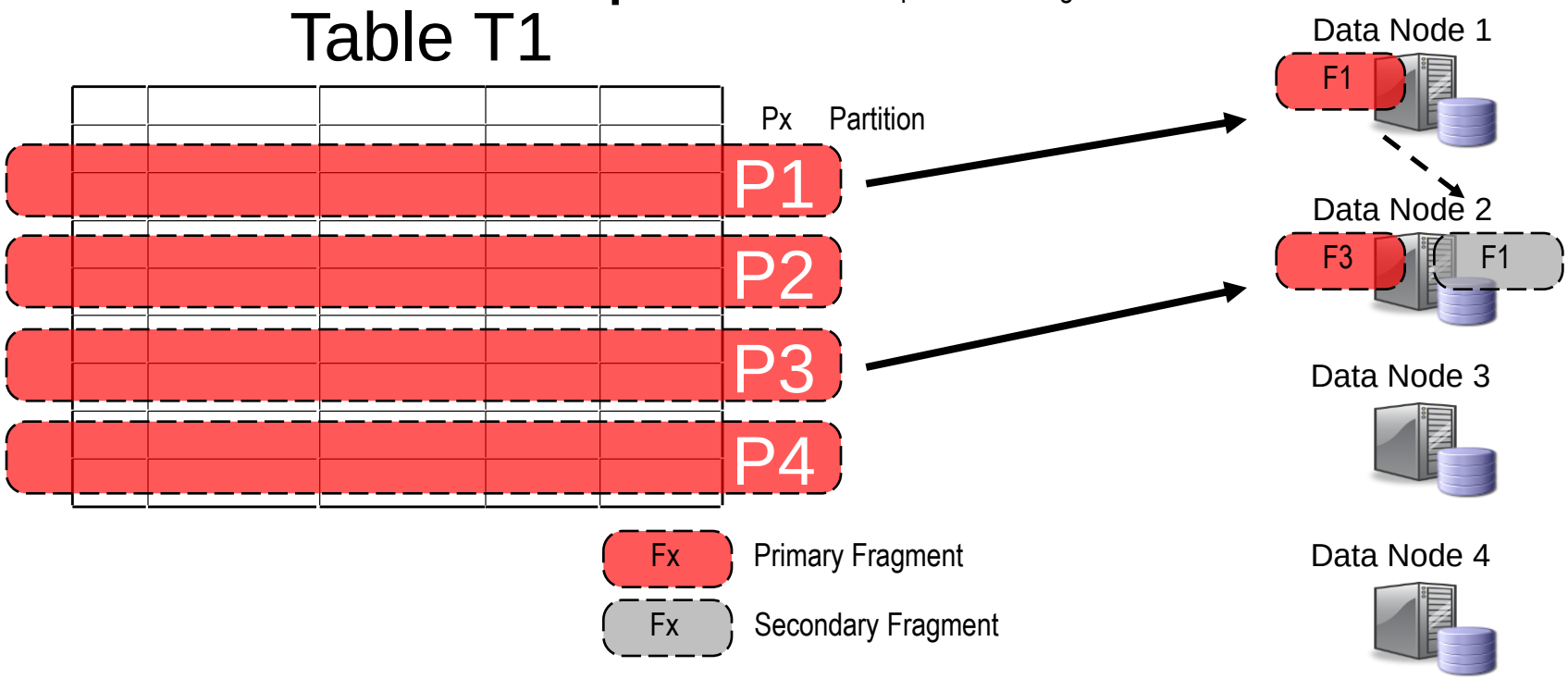


Automatic Data Partitioning

Cluster

Nodes & Node Groups

4 Partitions * 2 Replicas = 8 Fragments

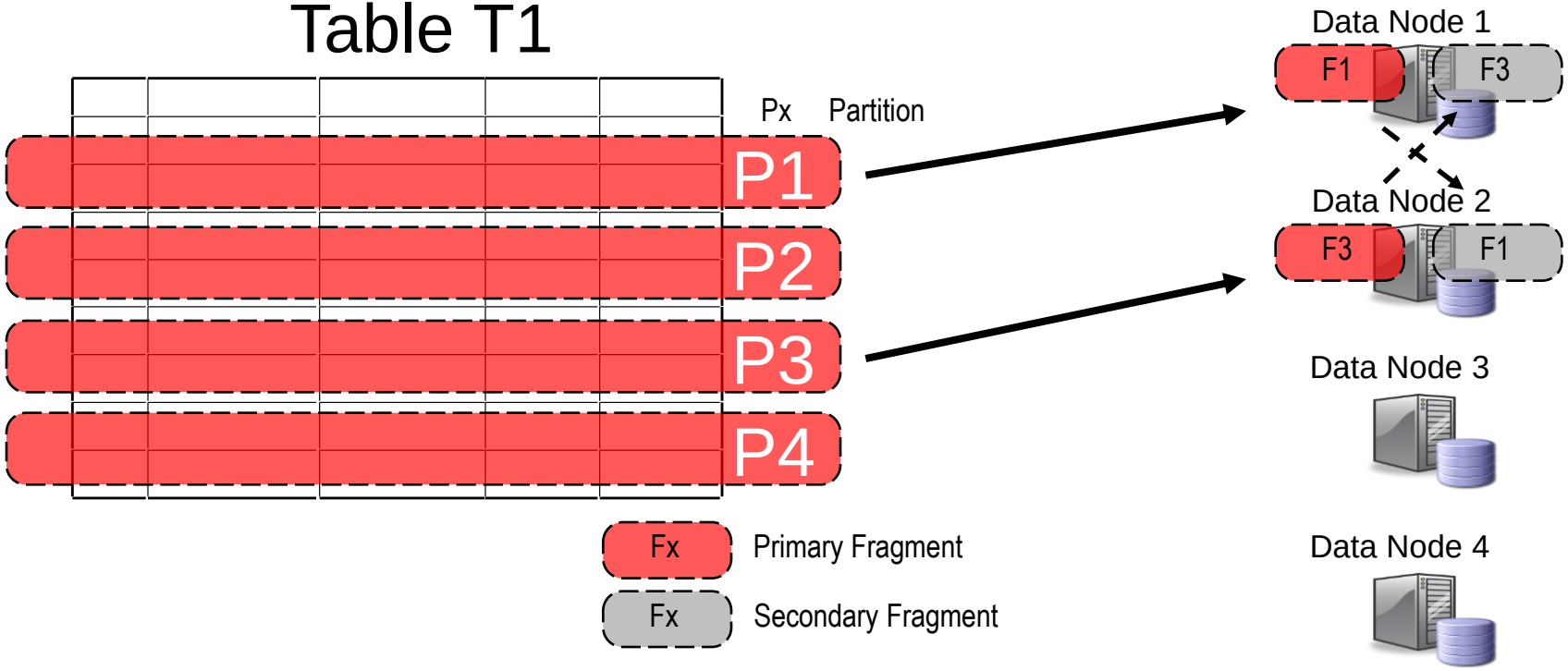


Automatic Data Partitioning

Cluster

Nodes & Node Groups

4 Partitions * 2 Replicas = 8 Fragments

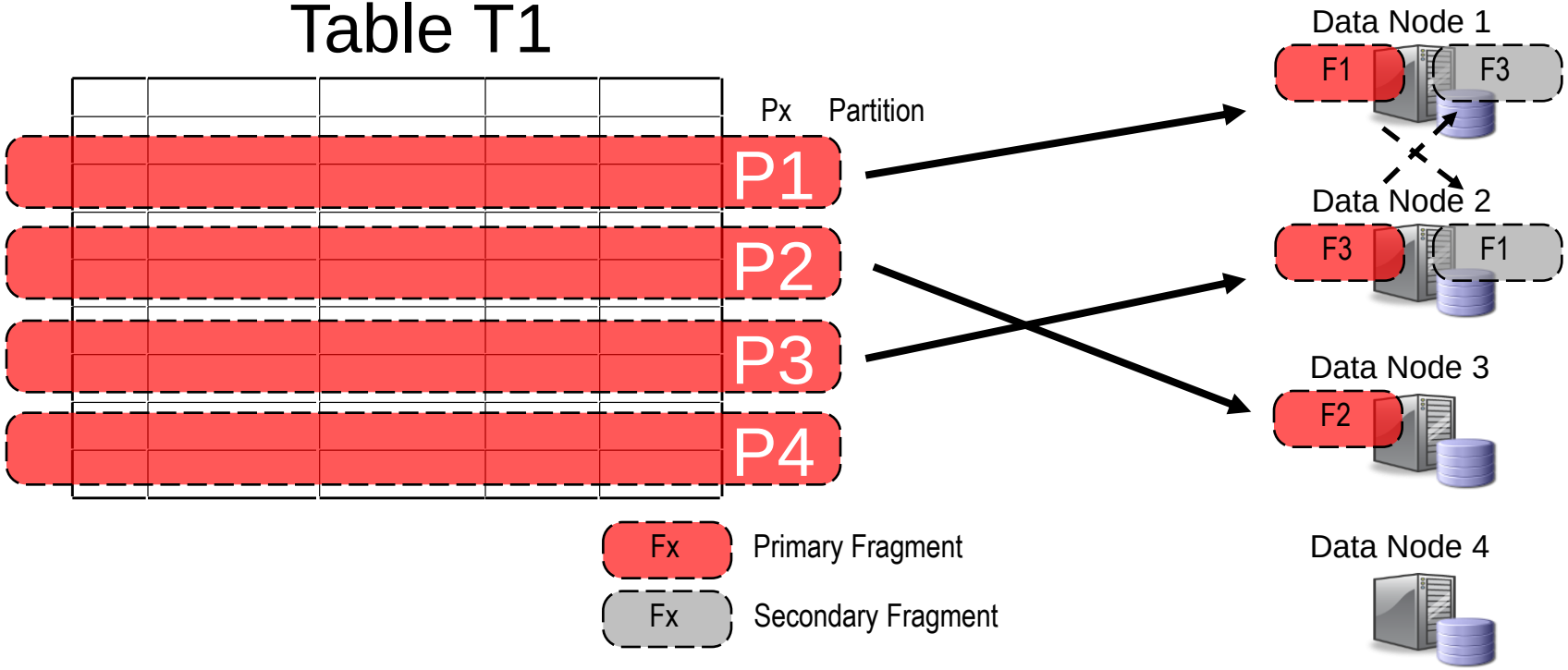


Automatic Data Partitioning

Cluster

Nodes & Node Groups

4 Partitions * 2 Replicas = 8 Fragments

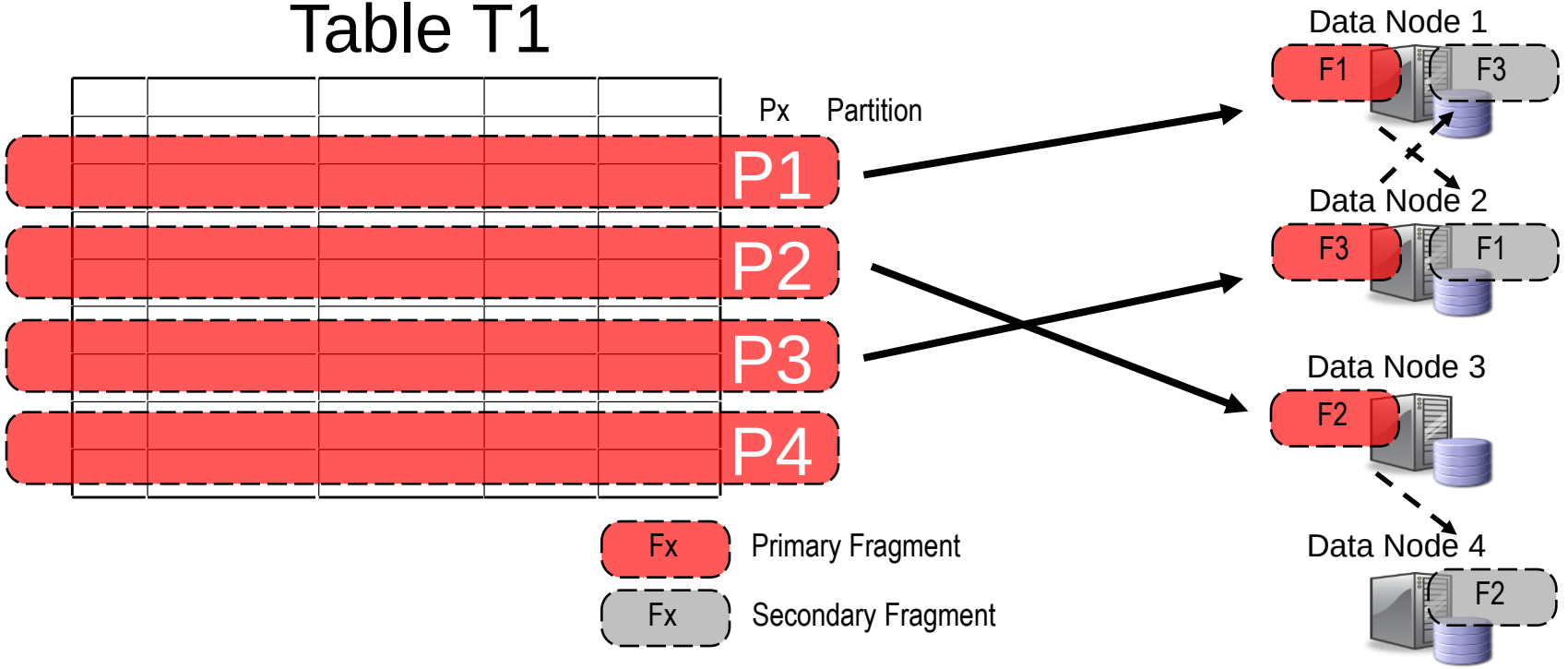


Automatic Data Partitioning

Cluster

Nodes & Node Groups

4 Partitions * 2 Replicas = 8 Fragments

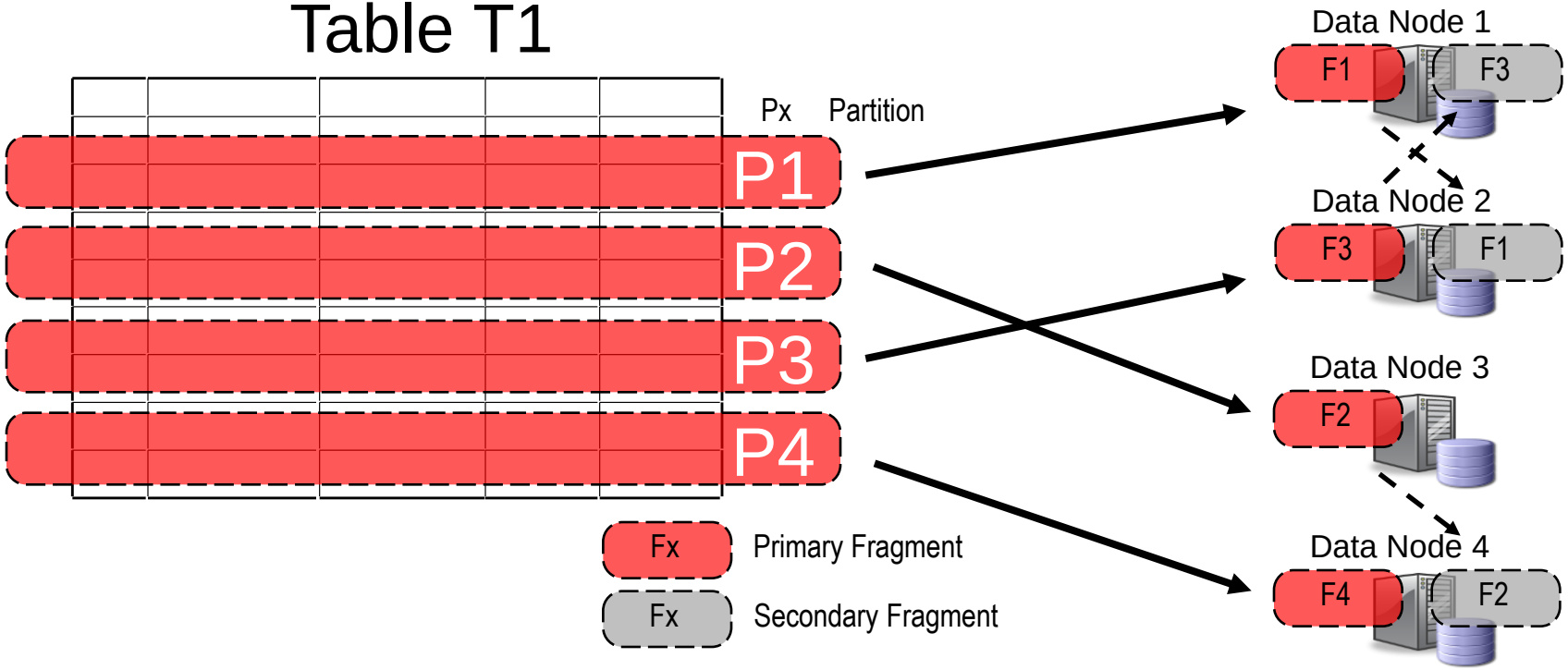


Automatic Data Partitioning

Cluster

Nodes & Node Groups

4 Partitions * 2 Replicas = 8 Fragments

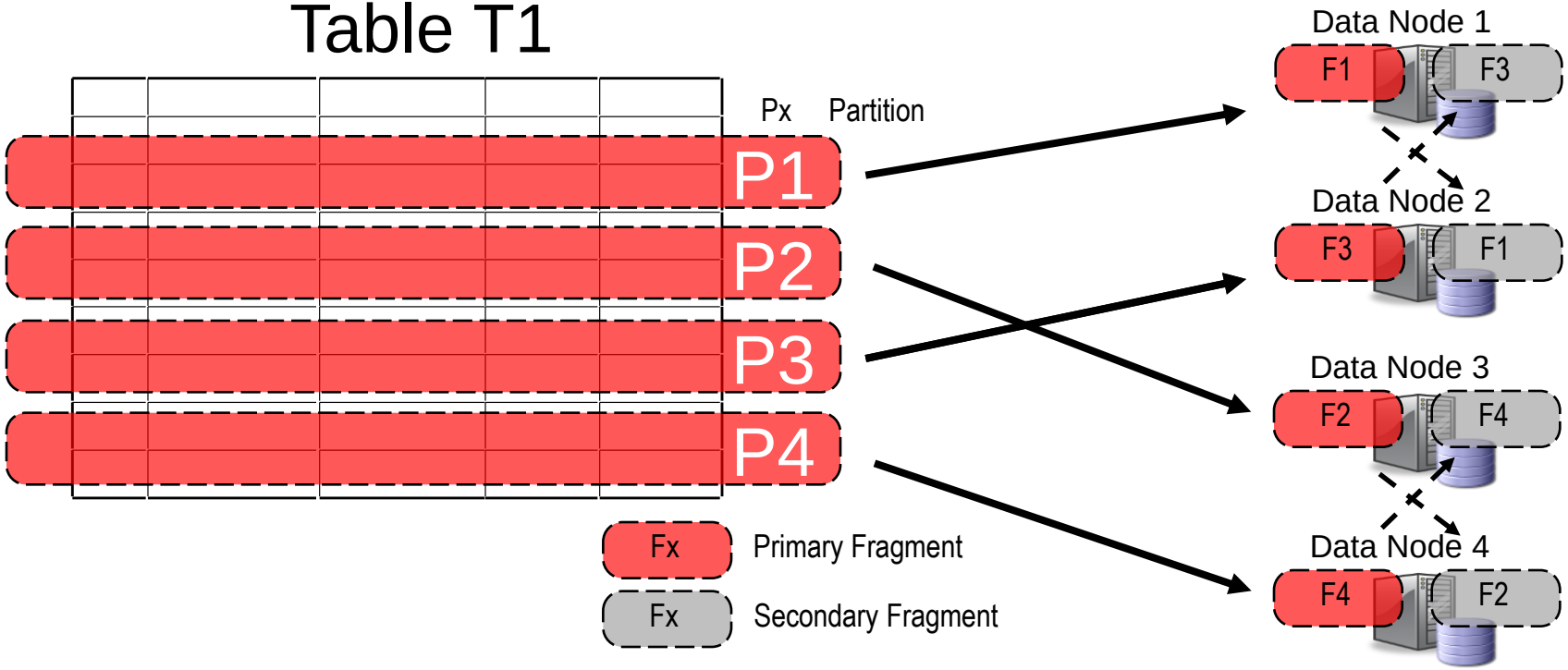


Automatic Data Partitioning

Cluster

Nodes & Node Groups

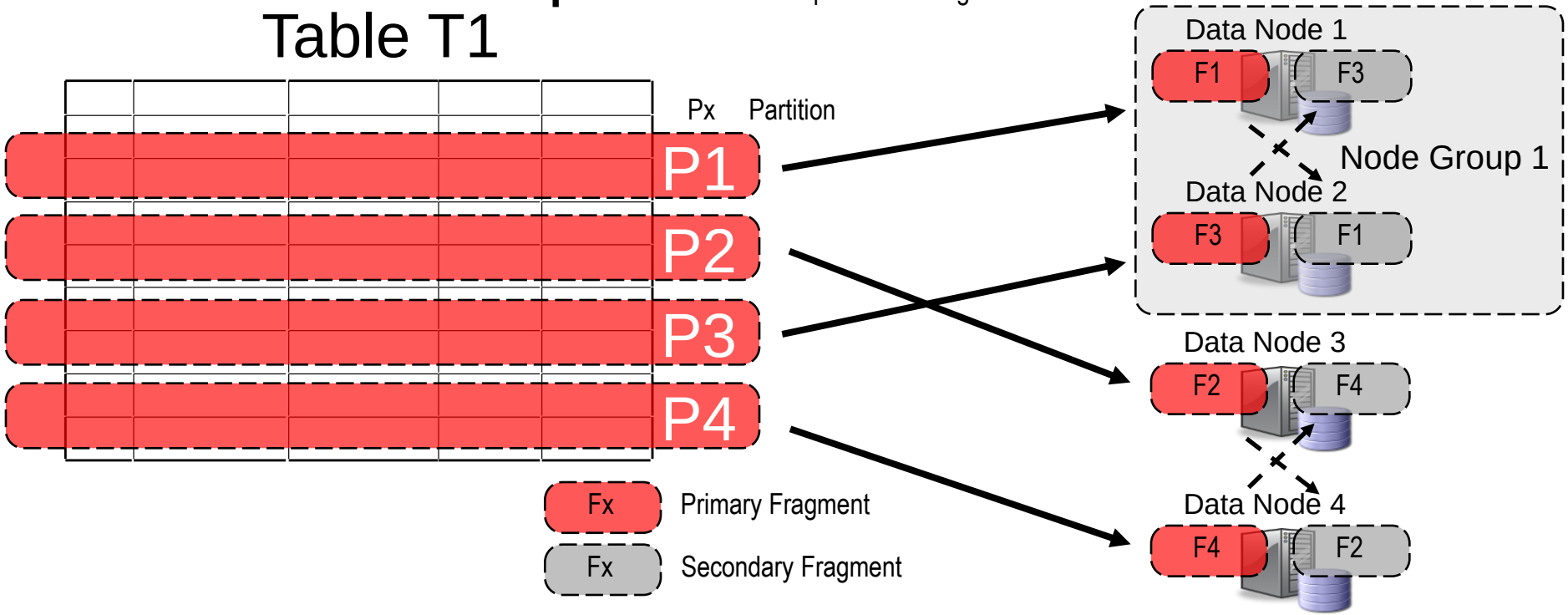
4 Partitions * 2 Replicas = 8 Fragments



Automatic Data Partitioning

Nodes & Node Groups

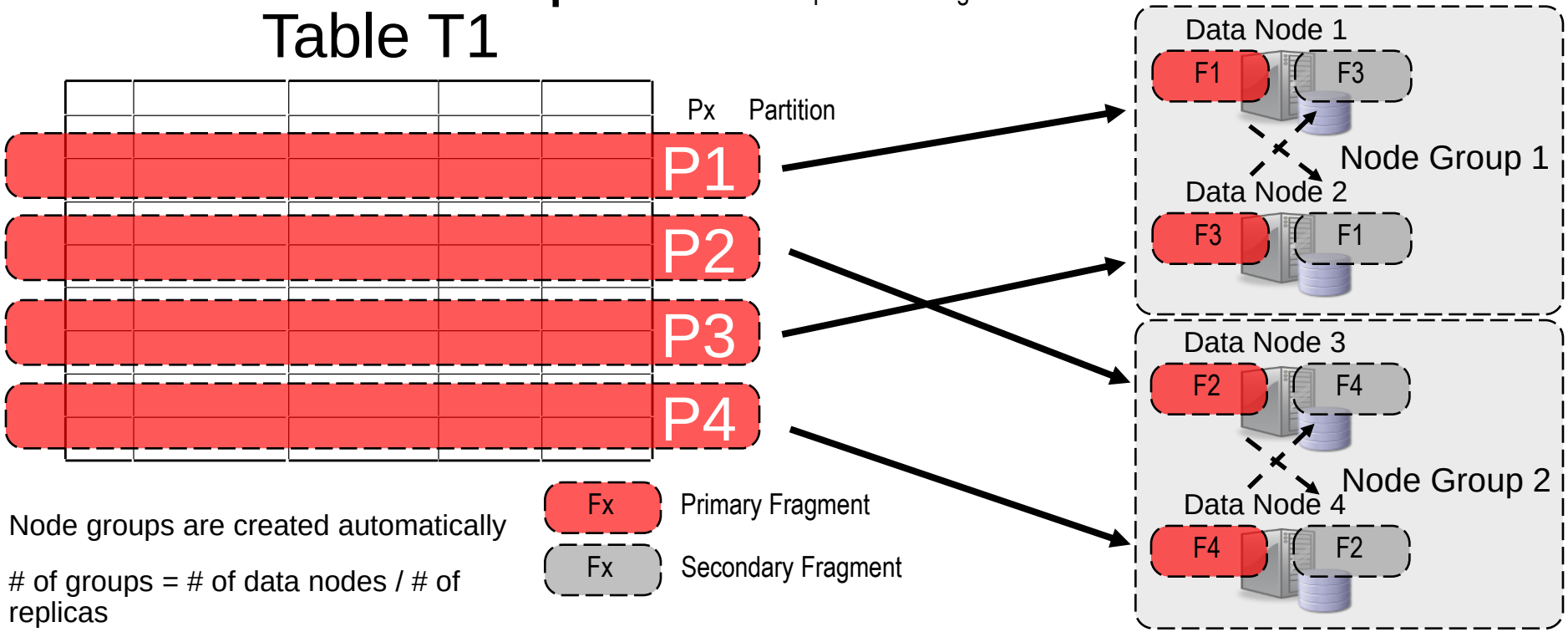
4 Partitions * 2 Replicas = 8 Fragments



Automatic Data Partitioning

Nodes & Node Groups

4 Partitions * 2 Replicas = 8 Fragments



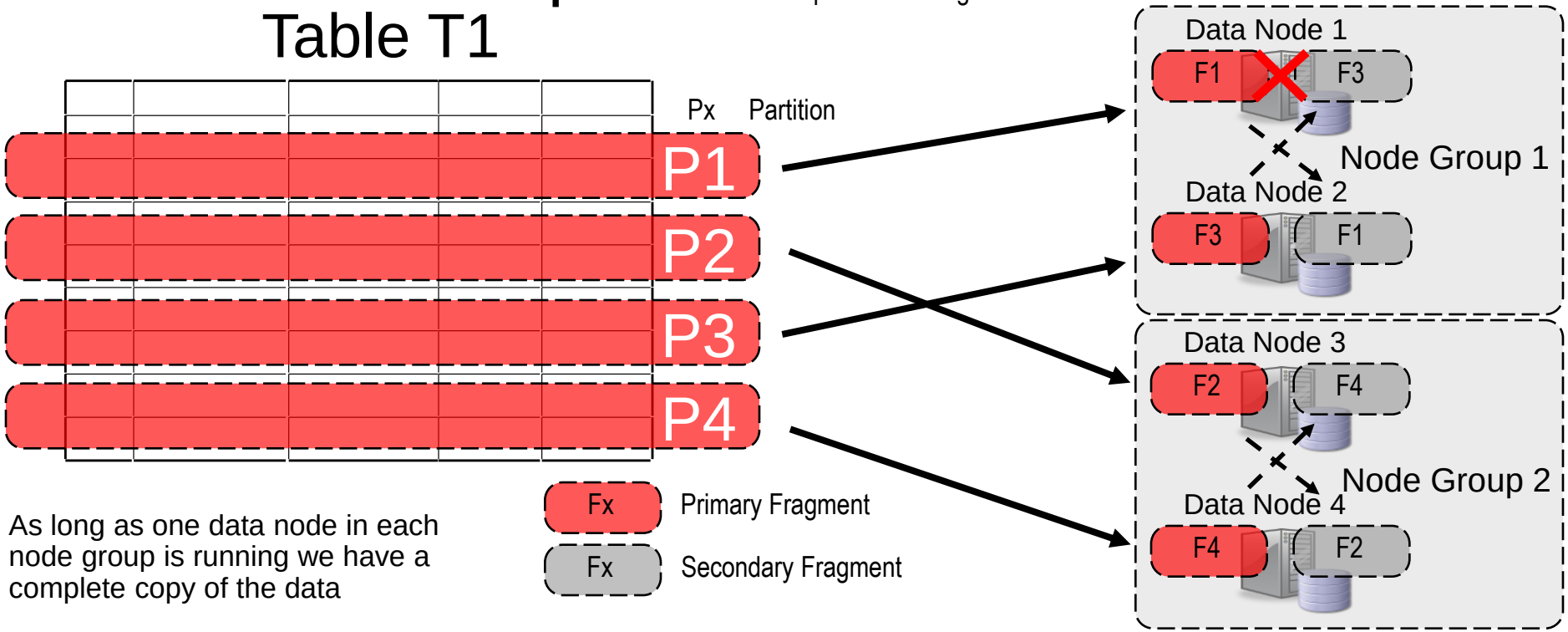
Node groups are created automatically

# of groups = # of data nodes / # of replicas

Automatic Data Partitioning

Nodes & Node Groups

4 Partitions * 2 Replicas = 8 Fragments

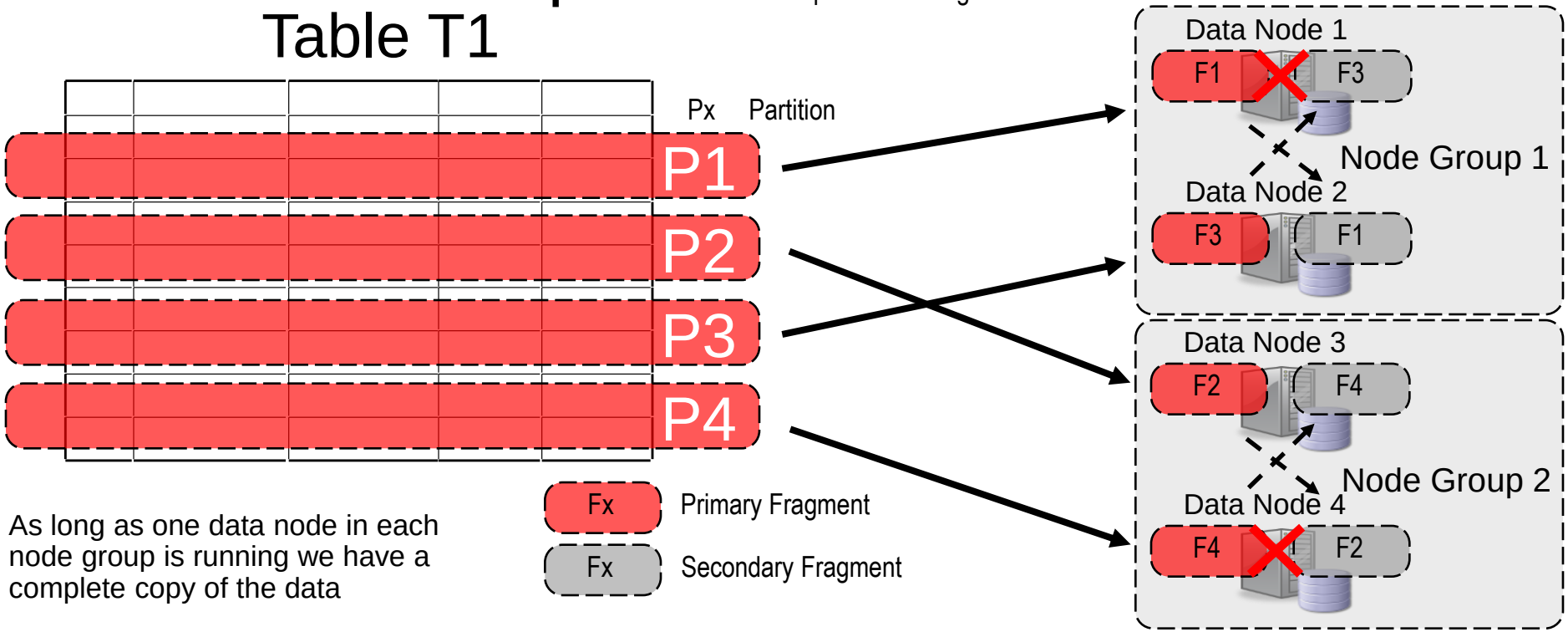


As long as one data node in each node group is running we have a complete copy of the data

Automatic Data Partitioning

Nodes & Node Groups

4 Partitions * 2 Replicas = 8 Fragments

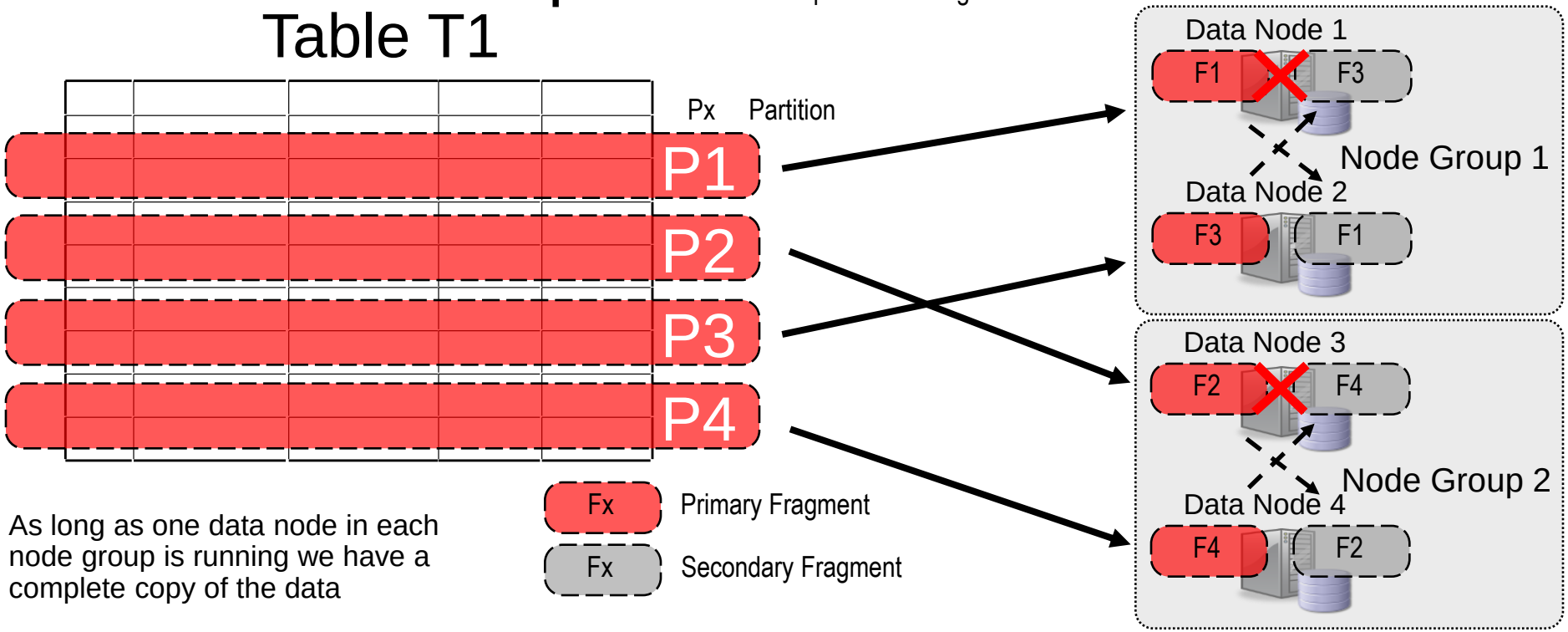


As long as one data node in each node group is running we have a complete copy of the data

Automatic Data Partitioning

Nodes & Node Groups

4 Partitions * 2 Replicas = 8 Fragments



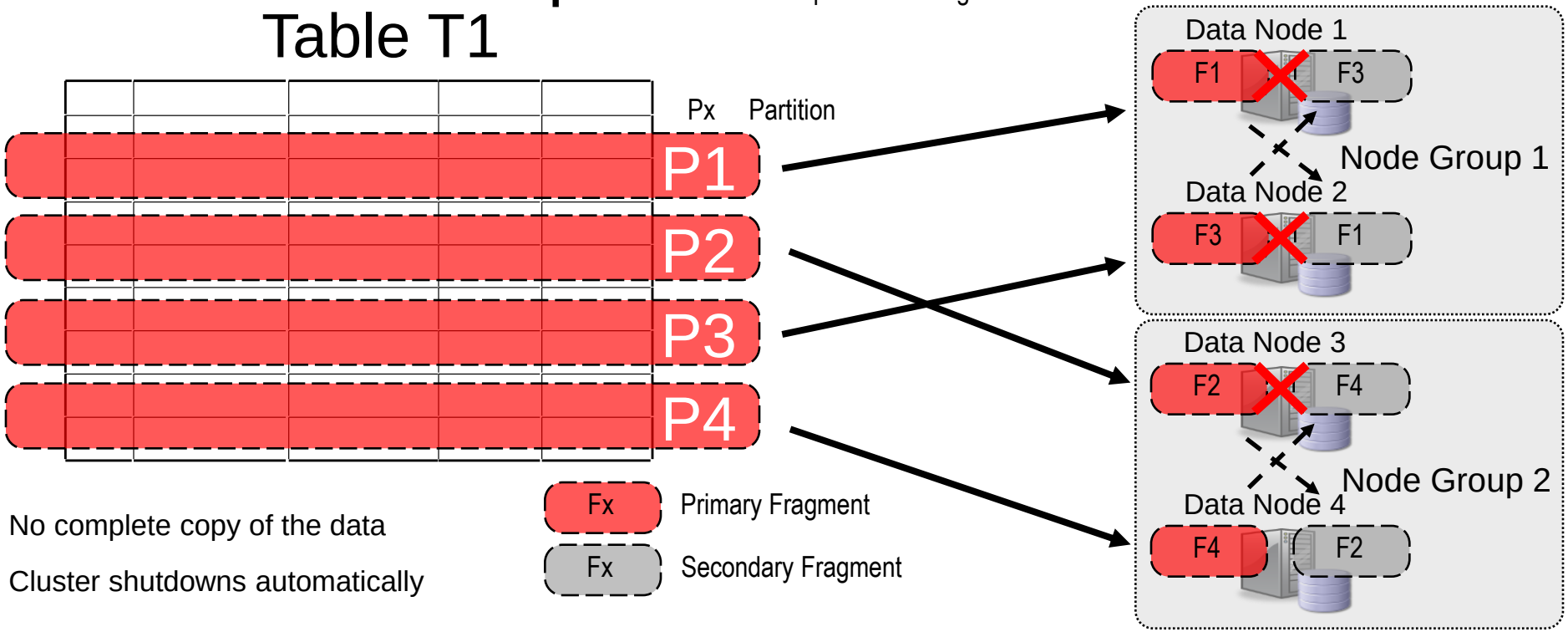
As long as one data node in each node group is running we have a complete copy of the data

Automatic Data Partitioning

Cluster

Nodes & Node Groups

4 Partitions * 2 Replicas = 8 Fragments



No complete copy of the data

Cluster shutdowns automatically

User Defined Data Partitioning

Cluster

- ユーザー定義パーティショニング
  - データとインデックスに適用されます
- キーによるパーティショニング
  - Hashによるパーティショニングに類似
- 特定のオプションを使用してソースをコンパイル
  - 範囲
  - リスト
  - ハッシュ

※ Scale Outはノード追加とデータのリロケーションにて対応

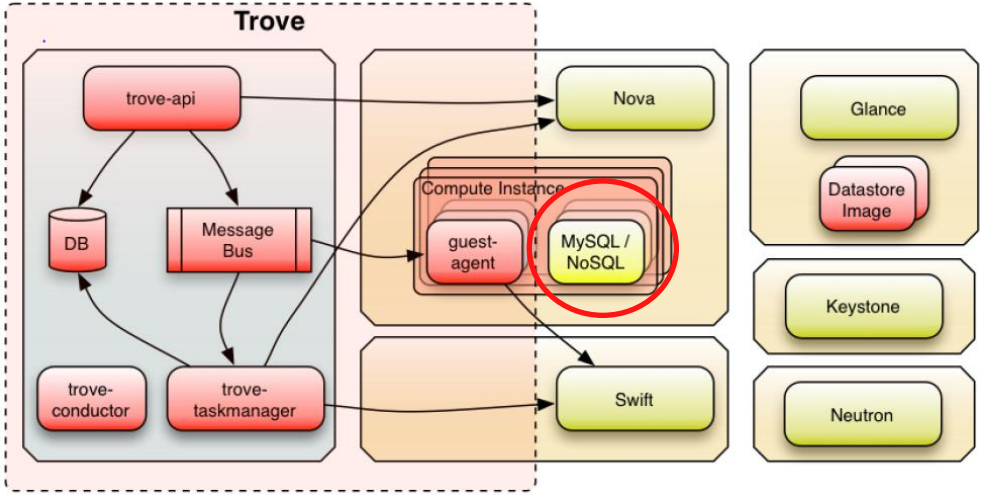
Auto Scale and Provisioning

MySQL & OpenStack Trove

Trove
Community Edition

- Database as a Service for OpenStack
- Provision & Manage Multiple Database Instances
- Single Tenant Database in Compute (Nova) Instance
- User/Database Management
- Features a fully functional REST API

- Spin up Instances
- Create Replicas
- Resize instances
- Add Users & Databases / manage Grants
- Manage Database Backups
- Change DB configuration



Provisioning by Trove

Trove
Community Edition

Trove用管理DB

```
$ mysql -u root -p
mysql> CREATE DATABASE trove;
mysql> GRANT ALL PRIVILEGES ON trove.* TO trove@'localhost' IDENTIFIED BY 'TROVE_DBPASS';
mysql> GRANT ALL PRIVILEGES ON trove.* TO trove@'%' IDENTIFIED BY 'TROVE_DBPASS'
```

```
+----+--------+-------------+-----------------------+--------+------------+------+
| id | name | datastore | datastore_version | status | flavor_id | size |
+----+--------+-------------+-----------------------+--------+------------+------+
```

TroveによるDBインスタンス作成

```
$ trove create 名前 2 --size=2 --databases DBNAME ¥
--users USER:PASSWORD --datastore_version mysql-5.6  --datastore mysql
```

Troveは2014年にリリースされたIce Houseからサポートしている。
Troveは標準でMySQLに対応している。詳細：https://wiki.openstack.org/wiki/Trove

商用版では、他にvFablic Data Directorのように更に高機能なソフトもある。

Server Provisioning

Fabric

MySQL Fabricは、OpenStackなどのクラウドフレームワーク同様に、
ベアメタル及び仮想環境においてMySQLの管理を可能とし、
柔軟にMySQLをスケールアウトさせる事が出来ます。
また、MySQLファブリックは、新しいサーバーの利用開始する前に、
透過的にMySQLとレプリケーションを設定します。

Managing read load **Read load increases**	**Managing shards** **A shard is hot or full**	**Managing server failures** **A server in a group fails/Slave is promoted**
Problem: Read servers overloaded Solution: Add more read Servers	Problem: slower queries Solution: Split the shard	Problem: Availability is affected Solution: Add replacement Node

Elastic Scalability with Replication

Fabric

SCALABILITY & PERFORMANCE

- マルチスレッドスレーブ
- バイナリーログのグループコミット
- 最適化された行ベースのレプリケーション

SELF HEALING

- 自動フェイルオーバー、スイッチオーバー
- グローバルトランザクション識別子(GTID)
- クラッシュセーフなスレーブ&バイナリーログ
- レプリケーションイベントチェックサム

DEV/OPS AGILITY

- 遅延レプリケーション
- リモートからのバイナリーログバックアップ
- 情報ログイベント

Self Healing MySQL Cloud Replication

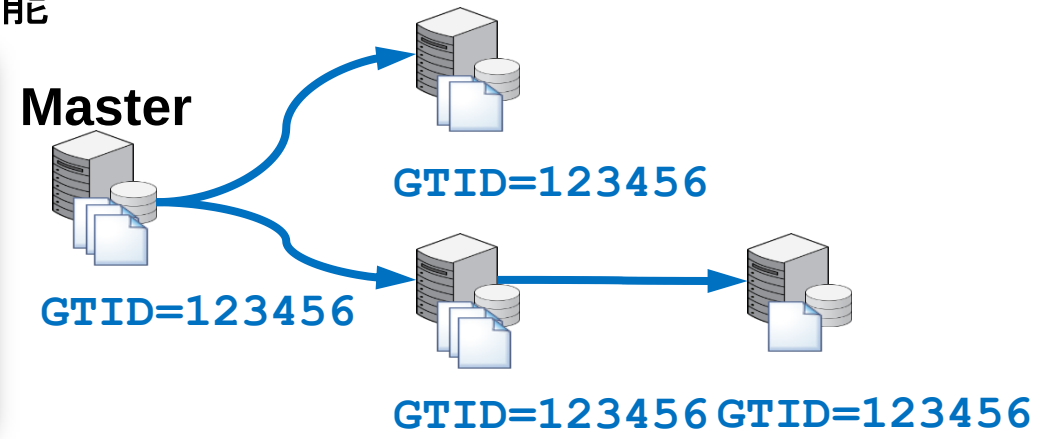
Fabric

グローバルトランザクション識別子(GTID)

- シンプルにトレース出来 & インスタンス間で容易にレプリケーションを比較可能
  - バイナリーログに書き込まれた各トランザクションの一意の識別子
- 自動的にフェールオーバーする為の最新データを持つスレーブを識別
- N階層のレプリケーションを展開可能

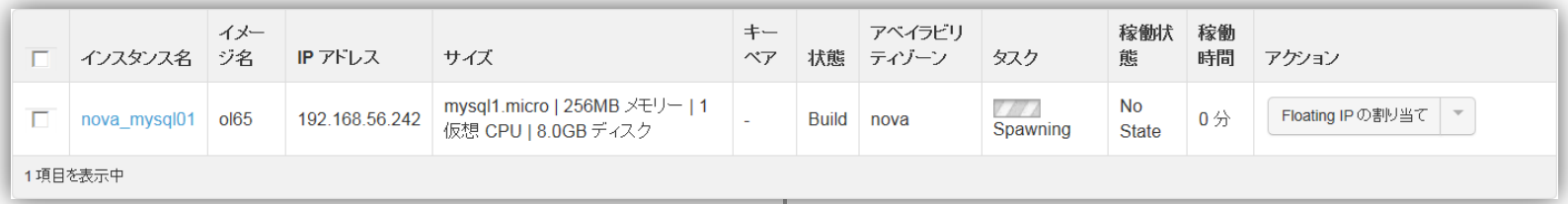
```
root@localhost [mysql]>SELECT @@GLOBAL.GTID_EXECUTED;
+-----------------------------------------+
| @@GLOBAL.GTID_EXECUTED                  |
+-----------------------------------------+
| 3edaa0b8-3e39-11e4-9df1-080027f5bf08:1-109 |
+-----------------------------------------+
1 row in set (0.00 sec)

root@localhost [mysql]>
```

```
root@localhost [mysql]> SELECT @@GLOBAL.GTID_EXECUTED\G
*************************** 1. row ***************************
@@GLOBAL.GTID_EXECUTED: 3edaa0b8-3e39-11e4-9df1-080027f5bf08:1-108,
cf80b01f-364f-11e4-aa59-0800270e2d1e:1-9
1 row in set (0.00 sec)

root@localhost [mysql]>
```



Server Provisioning – OpenStack Nova Integration

Fabric



	インスタンス名	イメージ名	IP アドレス	サイズ	キーペア	状態	アベイラビリティゾーン	タスク	稼働状態	稼働時間	アクション
☐	nova_mysql01	ol65	192.168.56.242	mysql1.micro | 256MB メモリー | 1 仮想 CPU | 8.0GB ディスク	-	Build	nova	Spawning	No State	0 分	Floating IP の割り当て

1 項目を表示中

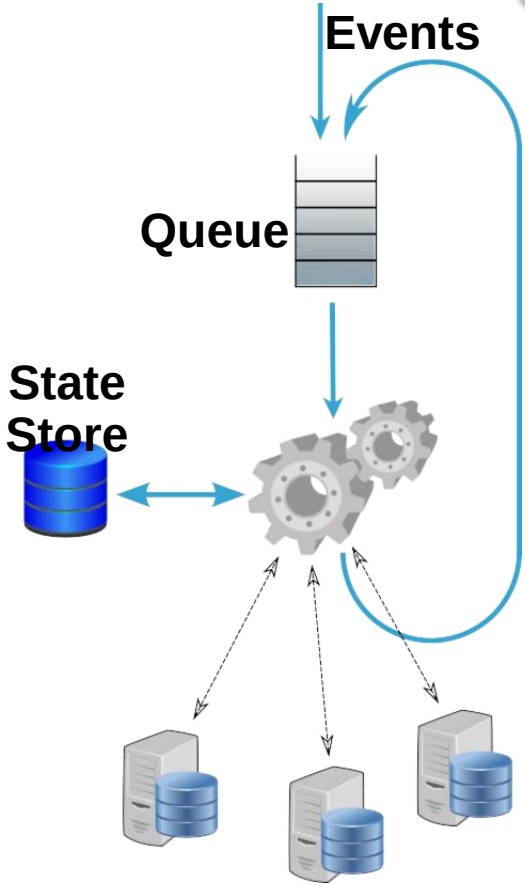
```
> mysqlfabric provider register my_stack ¥
 my_user my_password ¥
 http://8.21.28.222:5000/v2.0/ ¥
 --tenant=my_user_role ¥
 --provider_type=OPENSTACK

> mysqlfabric machine create my_stack ¥
 --image id=8c92f0d9-79f1-4d95-b398-86bda7342a2d ¥
 --flavor name=m1.small

> mysqlfabric machine list my_stack
```

- Fabric creates new machines, & MySQL Servers
  1. Initially using OpenStack Nova
     - Other frameworks on the way (OpenStack Trove, AWS,…)
- Server setup
  2. Clones slave
  3. Sets up replication
  4. Performs custom operations

MySQL Fabric executor

- イベント駆動
  - Events will trigger execution of procedures
  - Procedures can trigger events themselves
  - Each step of a procedure is called a *job*
- 手順
  - Written in Python
  - Interacts with servers
  - Write state changes into state store
  - Lock manager for conflict resolution
    - Conservative two-phase locking strategy
    - Avoid deadlocks

参照: [MySQL Fabric – adding Scaling to MySQL]



MySQL Fabric: Goals & Features

Fabric

- Connector API Extensions
  - Support Transactions
  - Support full SQL
- Fabric-Aware Connectors at GA:
  - PHP + Doctrine, Python, Java + Hibernate, .NET, C (pre-GA)
- Decision logic in connector
  - Reducing latency &network load
- Load Balancing
  - Read-Write Split
  - Distribute transactions
- Global Updates
  - Global data
  - Schema updates
- Sharding Functions
  - Range
  - (Consistent) Hash
- Shard Operations
  - Shard move
  - Shard split
- Server Provisioning
  - Cloud frameworks integration

MySQL Fabric – Current Limitations

Fabric

- Routing is dependent on Fabric-aware connectors
  - Currently Java (+ Hibernate), PHP (+ Doctrine), Python, .NET & C (labs)
- MySQL Fabric node is a single (non-redundant process)
  - HA Maintained as connectors continue to route using local caches
- Establishes asynchronous replication
  - Manual steps to switch to semisynchronous

- Sharding not completely transparent to application (must provide shard key – column from application schema)
- No cross-shard joins or other queries
- Management is through CLI, MySQL protocol or XML/RPC API
  - No GUI

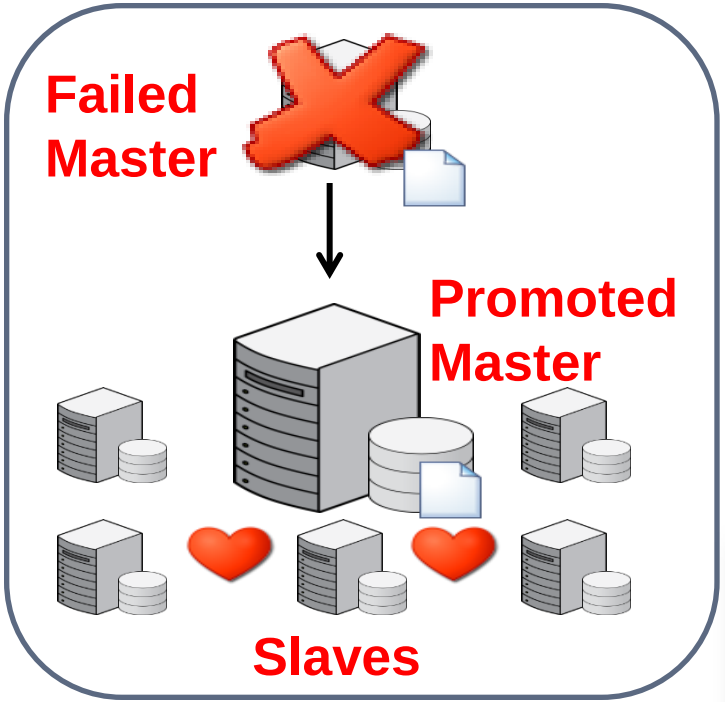
MySQL Fabric Resources

Fabric

- Download and try
  http://dev.mysql.com/downloads/fabric/
- Documentation
  http://dev.mysql.com/doc/mysql-utilities/en/fabric.html
- MySQL Fabric on the web
  http://www.mysql.com/products/enterprise/fabric.html
- Forum (MySQL Fabric, Sharding, HA, Utilities)
  http://forums.mysql.com/list.php?144
- Tutorial: MySQL Fabric - adding High Availability and Scaling to MySQL
  http://www.clusterdb.com/mysql-fabric/mysql-fabric-adding-high-availability-and-scaling-to-mysql
- White Paper: MySQL Fabric - A Guide to Managing MySQL High Availability and Scaling Out
  http://www.mysql.com/why-mysql/white-papers/mysql-fabric-product-guide
- Webinar Replays
  http://www.mysql.com/news-and-events/on-demand-webinars/#en-20-41

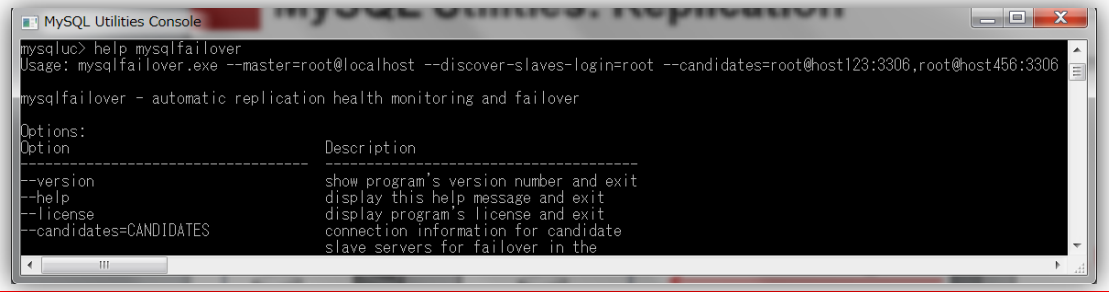
MySQL Utilities: Replication

Option: Utilities



- Enabling self-healing replication clusters
- Automated failover & recovery
  - **`mysqlfailover`** Utility
- Switchover & administration
  - **`mysqlrpladmin`** Utility

- Integrated HA to tolerate instance and region failures



Get Started Today!

MySQL Enterprise Edition Trial

30日間トライアル

Oracle Software Delivery Cloud

http://edelivery.oracle.com/

製品パックを選択： “MySQL Database”

事例紹介：http://www.mysql.com/why-mysql/case-studies/#ja-5-0

製品マニュアル： http://dev.mysql.com/doc/index-enterprise.html

Contact a MySQL Sales Rep

[MySQL お問い合わせ窓口]
電話： 0120-065556
【受付時間】平日 9:00-12:00/13:00-18:00
(祝日及び年末年始休業日を除きます)
メール：MySQL-Sales_jp_grp@oracle.com
URL： http://www.mysql.com/about/contact/

有難うございました