MySQL

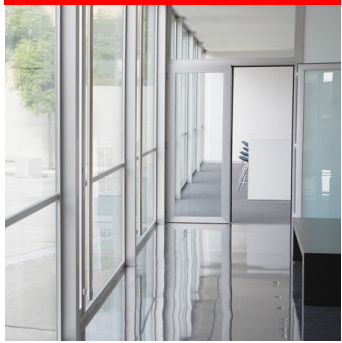
ORACLE

InnoDB パフォーマンスチューニング・新機能

Software Developer, Oracle
木下　靖文
April, 3rd, 2012

以下の事項は、弊社の一般的な製品の方向性に関する概要を説明するものです。また、情報提供を唯一の目的とするものであり、いかなる契約にも組み込むことはできません。以下の事項は、マテリアルやコード、機能を提供することをコミットメント（確約）するものではないため、購買決定を行う際の判断材料になさらないで下さい。オラクル製品に関して記載されている機能の開発、リリースおよび時期については、弊社の裁量により決定されます。

Agenda



- InnoDB とは？
- InnoDB アーキテクチャ
- InnoDB 機能・新機能
- InnoDB パフォーマンスチューニング

InnoDB とは？

InnoDB とは？

2001年からMySQLのコンポーネントとして提供されている、高可用性・ハイパフォーマンスのストレージエンジン。MySQL 5.5 からはデフォルトのストレージエンジンとなっている。主な特徴は…

- ACID特性に沿ったトランザクション処理・クラッシュリカバリ
- 行レベルロック、読み取り一貫性
- 表の構造は Clustered Index（≒索引構成表）
- 外部キー参照整合性のサポート

InnoDB の歴史（詳細略）

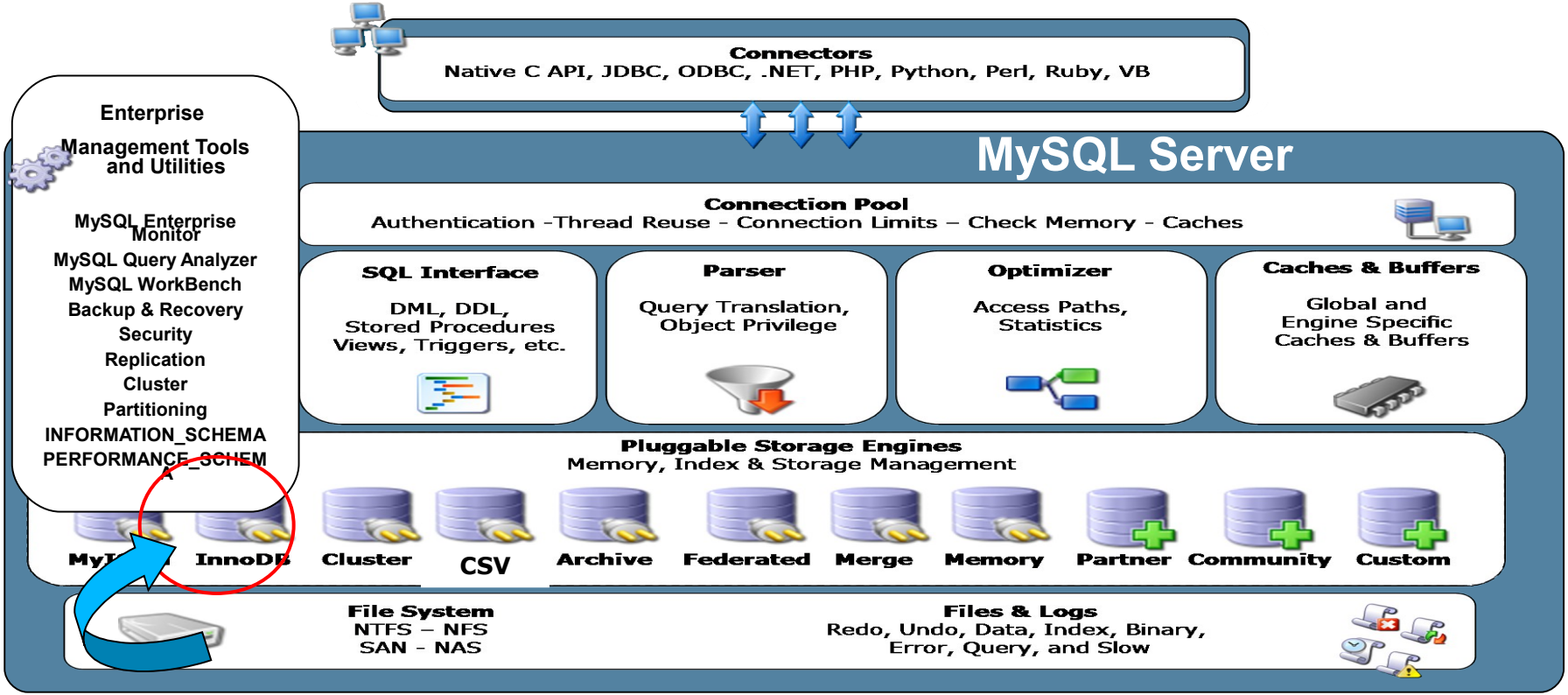
- Dr. Heikki Tuuri の手により制作される (1994/1)
- Innobase Oy 創業 (1995)
- MySQLに実装されオープンソースとして初リリース (2001/5)
- Oracle、Innobase Oy を買収 (2005/10)
- Sun、MySQL AB の買収を発表 (2008/1)
- Oracle、Sun の買収を発表 (2009/4)
- MySQL 5.5 で InnoDB がデフォルトのストレージエンジンとなる。

InnoDB の設計

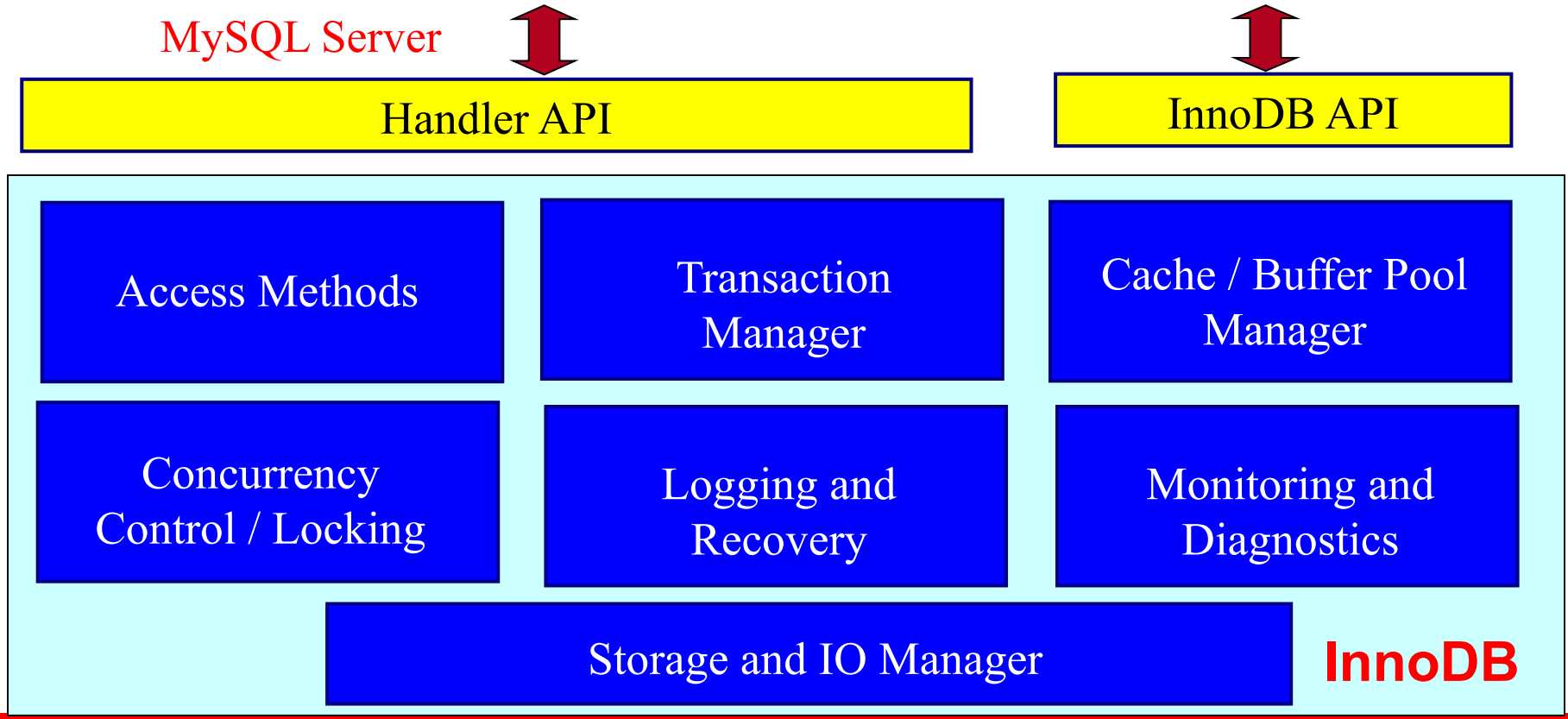
- Gray & Reuter著 “*Transaction Processing: Concepts & Techniques*” がモデル
  - ネクストキーロック
- Oracle DB のアーキテクチャのエミュレート
  - Multi-version concurrency control (MVCC)
  - Undo情報はログではなく、データとして格納 （ロールバックセグメント）
  - データ・索引の格納は、表毎のテーブルスペースへ
- 独自の機能／機構の追加
  - Doublewrite buffer
  - Change buffering (insert buffer)
  - Adaptive hash index

InnoDB アーキテクチャ

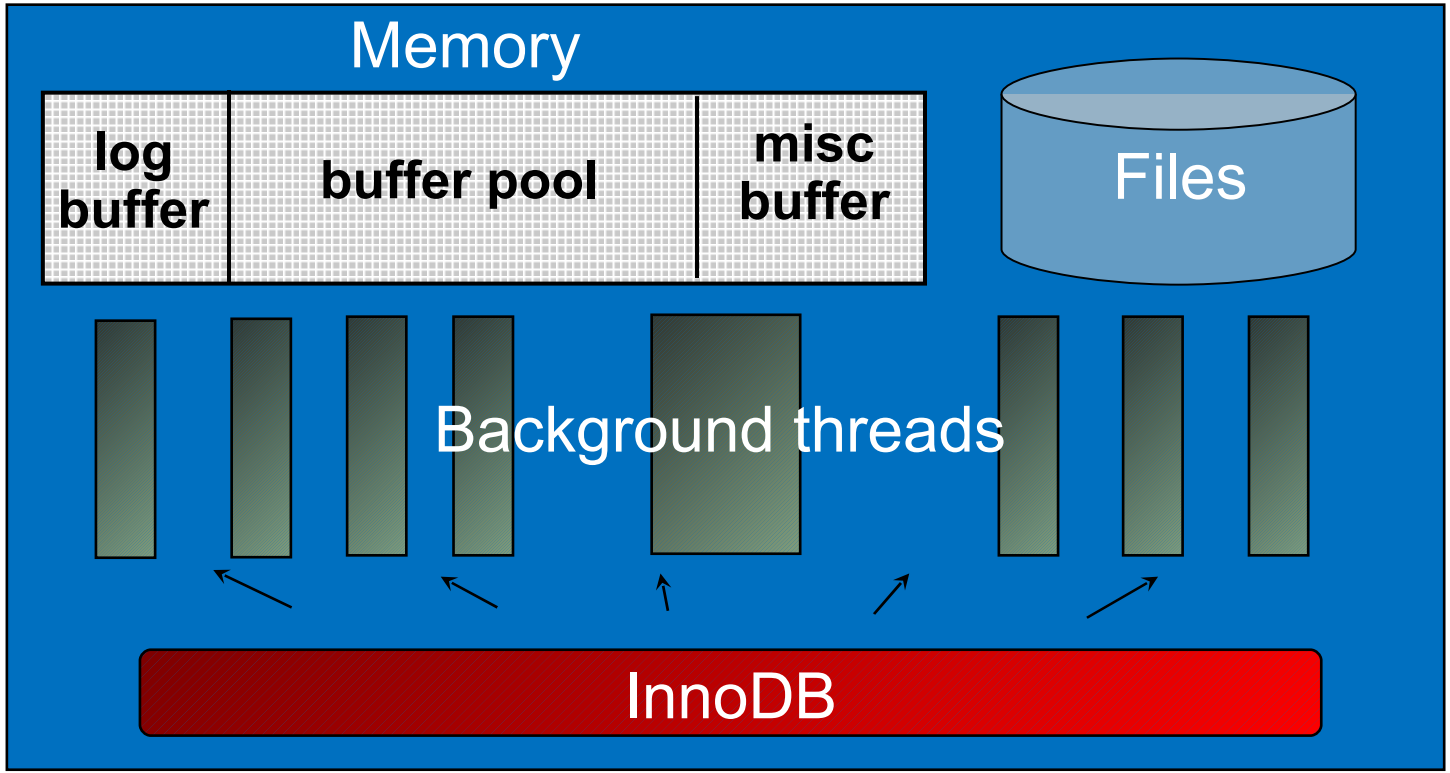
MySQL Server アーキテクチャ



InnoDB コンポーネントモデル



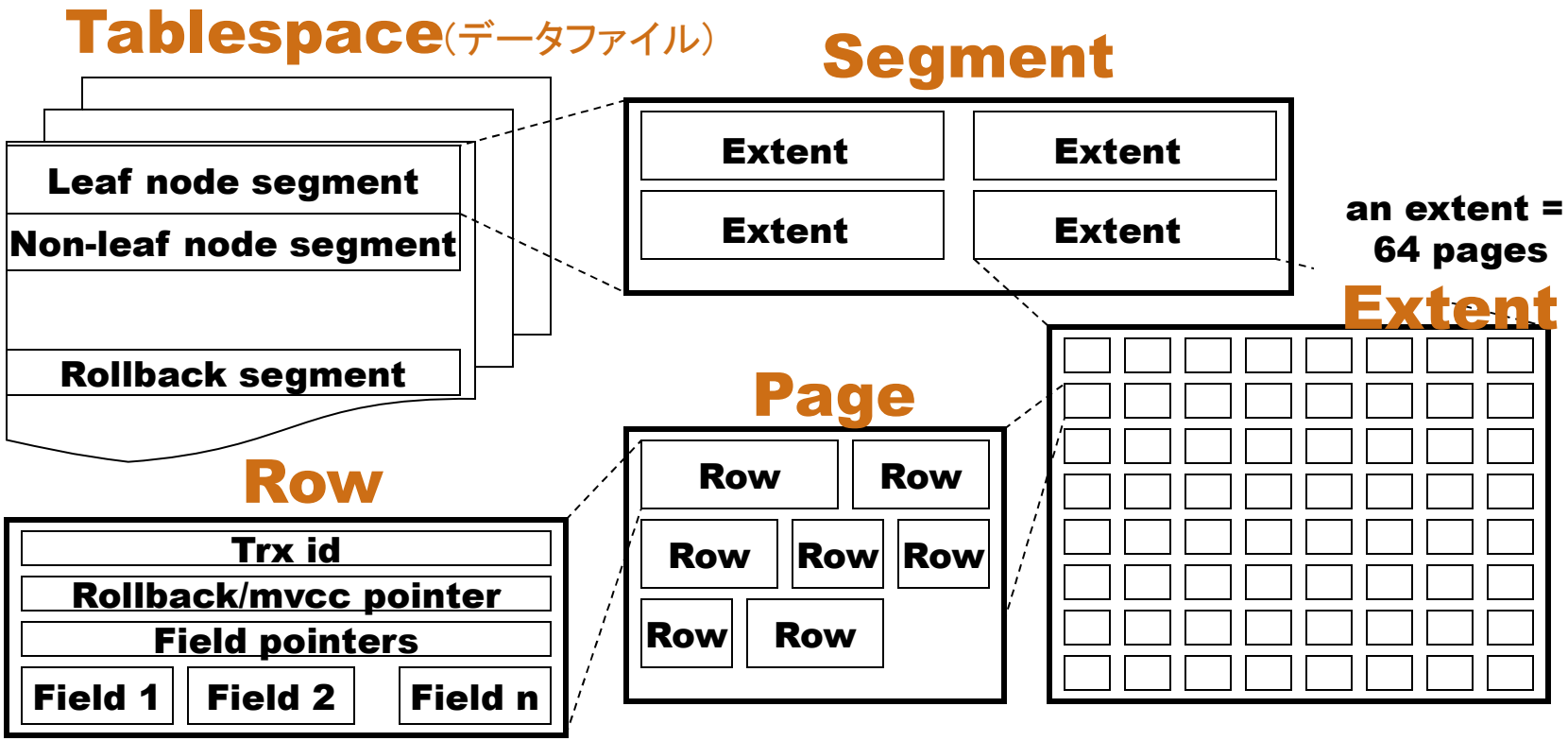
InnoDB ランタイムモデル



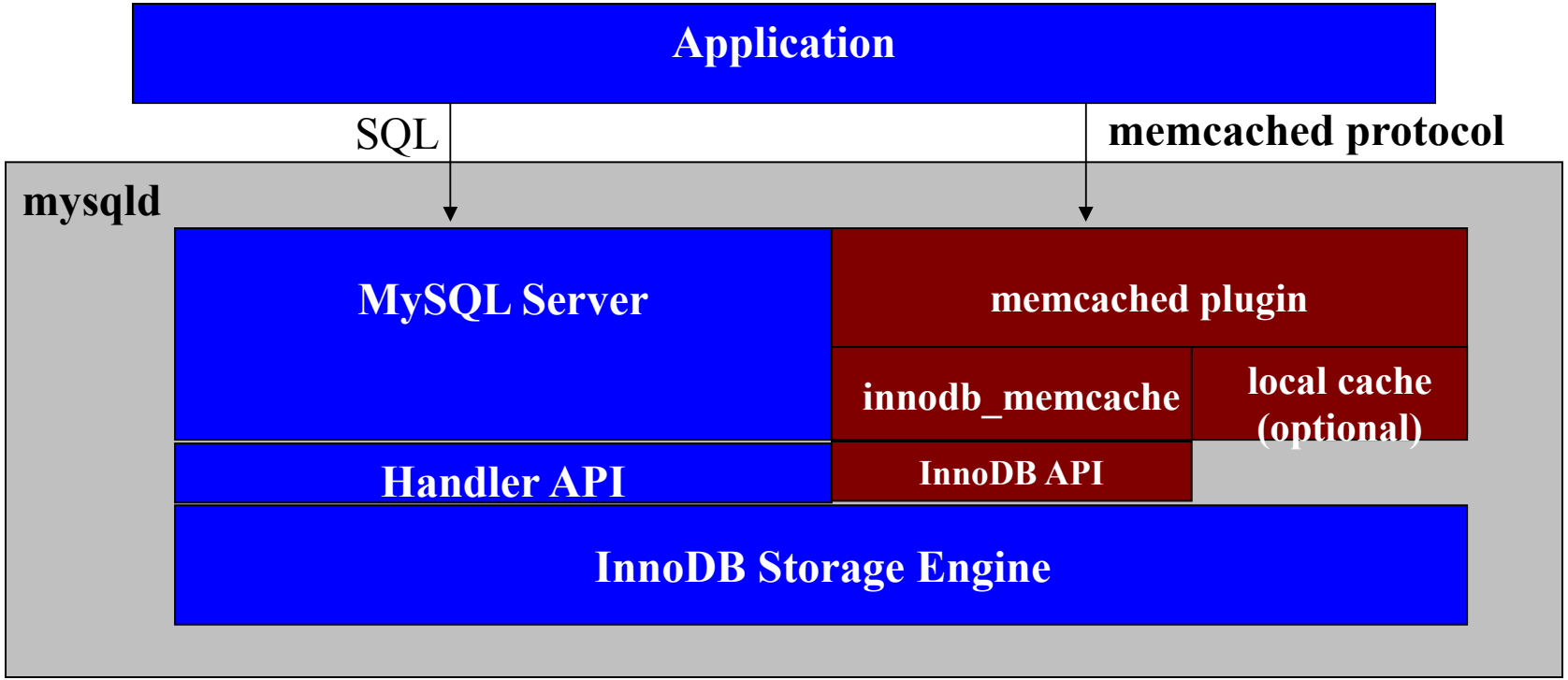
Threads:
-master
-read io
-write io
-purge
-monitor
-and more

Buffer Pool:
-data,index
-undo
-and more

InnoDB データファイル内部構造



（開発中）memcached protocol での InnoDB へのNoSQLアクセス



InnoDB 機能・新機能

トランザクションとロック

- ACID特性に沿ったトランザクション
  - atomicity, consistency, isolation, durability
- ANSI/ISO SQL-standard トランザクション分離レベル
  - READ UNCOMMITTED (非推奨)
  - READ COMMITTED
  - REPEATABLE READ
  - SERIALIZABLE
- MVCC、行レベルロック
- デッドロック検知

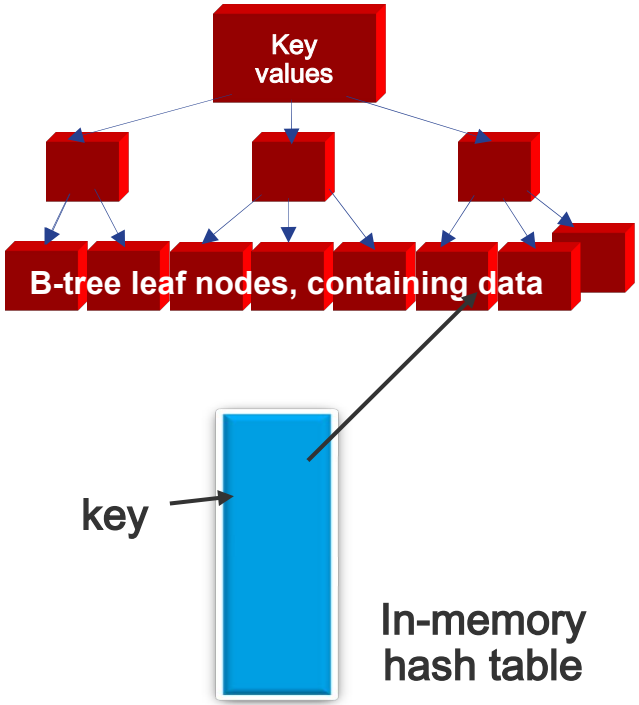
外部キーと参照整合性

- データの整合性の保護のために外部キーを定義することができる
- InnoDB は外部キーに違反する変更を回避します
- CREATE TABLE 文 の FOREIGN KEY 節のサポート
  - ON UPDATE
    CASCADE | SET NULL | RESTRICT | NO ACTION
  - ON DELETE
    CASCADE | SET NULL | RESTRICT | NO ACTION

Doublewrite Buffer

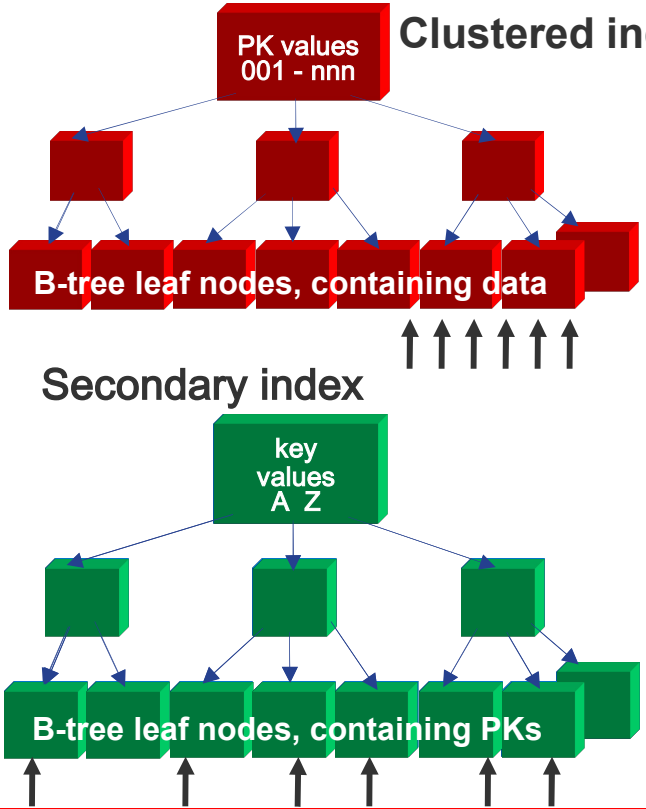
- クラッシュ時の partial write(書き込み途中状態での終了)によるデータページ破壊への対策機能
- データページ書き込み時にはシステムファイルのdoublewrite buffer領域にも同じ内容を事前に書き込む
- クラッシュリカバリ時にページ破壊が検出された場合、doublewrite buffer内からもそのページを検索する
  - 100%ではない（IOスケジューリングやディスクキャッシュの上書きの影響で、doublewrite bufferにまだ書き込まれていないか他の内容で既に上書きされている場合は修復不可）
  - ページ単位の書き込みIOのAtomicity(原子性)が下位層で100%確保できている場合は有効にする必要はない

Adaptive Hash Index



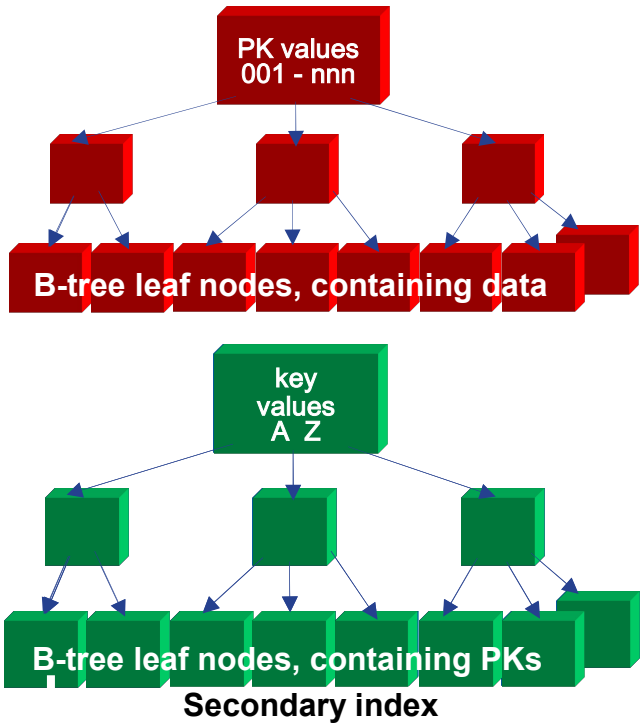
- Buffer poolのメモリ上にある一意キー索引のリーフページの「各レコードのアドレス」はAdaptive Hash Indexにも登録され、一意キー検索の際に B+tree 検索をスキップする事が出来る
- インメモリデータベースの機構に近い

Change Buffering



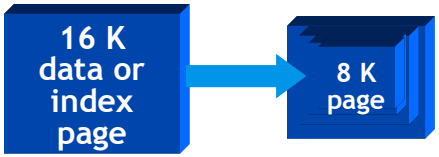
- セカンダリインデックスのページがbuffer pool に無い場合、そこに対する INSERT、DELETE(mark)、PURGE(delete)処理を後回しにすることができる
- セカンダリインデックスのページ読み込みを待つことなく次の処理が可能
- innodb_change_buffering='inserts'を指定すると従来のinsert bufferと同等の動作(5.5~)

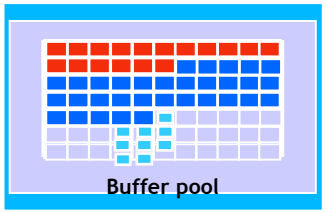
Fast Index Creation (5.1-InnoDB-Plugin~)

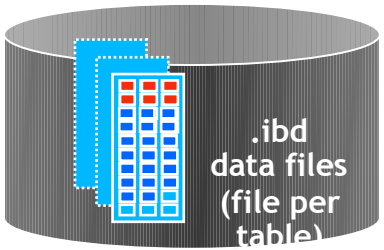


- ADD INDEX: 内部的な表の再作成は行わない
  - 索引作成のために必要なデータを読み込み、ソートしてから索引を作成する
- DROP INDEX: ほぼ、データディクショナリの変更のみ

Table Compression (5.1-InnoDB-Plugin~)







- 表毎に圧縮ページサイズを指定する
  CREATE TABLE t ...
  KEY_BLOCK_SIZE=8 (default)
- 圧縮後のサイズが圧縮ページに収まるかどうかでページの容量が決まる
- 不要な再圧縮を避けるため、ページに対する変更は圧縮差分データを圧縮データに付け加える形で行う（圧縮ページサイズに達した場合にのみ再圧縮する）

モニタリング と 診断

- SHOW ENGINE INNODB STATUS
  - BACKGROUND THREAD
  - SEMAPHORES
  - TRANSACTIONS
  - FILE I/O
  - INSERT BUFFER AND ADAPTIVE HASH INDEX
  - LOG
  - BUFFER POOL AND MEMORY
  - ROW OPERATIONS
- innodb monitor による上記情報の.errファイルへの周期的な出力
- Status variables: 48
- Information schema tables: 18

INNODB SHOW STATUS の出力例

```
=====================================
111003 17:20:42 INNODB MONITOR OUTPUT
=====================================
Per second averages calculated from the last 37 seconds
-----------------
BACKGROUND THREAD
-----------------
srv_master_thread loops: 3 srv_active, 0 srv_shutdown, 150 srv_idle
srv_master_thread log flush and writes: 153
----------
SEMAPHORES
----------
OS WAIT ARRAY INFO: reservation count 3
OS WAIT ARRAY INFO: signal count 3
Mutex spin waits 0, rounds 0, OS waits 0
RW-shared spins 3, rounds 90, OS waits 3
RW-excl spins 0, rounds 0, OS waits 0
Spin rounds per wait: 0.00 mutex, 30.00 RW-shared, 0.00 RW-excl
------------
TRANSACTIONS
------------
Trx id counter 503
```

InnoDB Status Variables

```
mysql> show global status like "innodb%";
  +---------------------------------+---------+
  | Variable_name                   | Value   |
  +---------------------------------+---------+

1.| Innodb_buffer_pool_pages_data     | 148     |
2.| Innodb_buffer_pool_pages_dirty    | 0       |
3.| Innodb_buffer_pool_pages_flushed  | 0       |
4.| Innodb_buffer_pool_pages_free     | 2668    |
5.| Innodb_buffer_pool_pages_misc     | 0       |
6.| Innodb_buffer_pool_pages_total    | 2816    |
……
18.| Innodb_data_fsyncs                | 3       |
19.| Innodb_data_pending_fsyncs        | 0       |
20.| Innodb_data_pending_reads         | 0       |
21.| Innodb_data_pending_writes        | 0       |
22.| Innodb_data_read                  | 4608000 |
```

モニタリング と 診断（新）

- Performance Schema (5.5~) を利用
  - “performance_schema.events_waits_summary_*” ビュー
  - InnoDBの内部イベント待ち（mutex、rw_lock、thread、IO）を監視可能
  - 「待ち時間」と「待った回数」の情報
  - 有効にするとパフォーマンスに多少の影響あり

- InnoDB Metrics Table (5.6~)
  - “information_schema.innodb_metrics” ビュー
  - 約200個のInnoDBの基礎動作に関するモニターカウンターの動作を innodb_monitor_* グローバル変数を用いて個別に制御
  - 軽量

Performance Schema 出力例

```
mysql> SELECT EVENT_NAME, COUNT_STAR, SUM_TIMER_WAIT, AVG_TIMER_WAIT
-> FROM EVENTS_WAITS_SUMMARY_BY_EVENT_NAME
-> WHERE EVENT_NAME like "%innodb%"
-> order by COUNT_STAR DESC;
```

EVENT_NAME	COUNT_STAR	SUM_TIMER_WAIT	AVG_TIMER_WAIT
buf_pool_mutex	1925253	264662026992	137468
buffer_block_mutex	720640	80696897622	111979
kernel_mutex	243870	44872951662	184003
purge_sys_mutex	162085	12238011720	75503
trx_undo_mutex	120000	11437183494	95309
rseg_mutex	102167	14382126000	140770
fil_system_mutex	97826	15281074710	156206
log_sys_mutex	80034	35446553406	442893
dict_sys_mutex	80003	6249472020	78115

（バージョン毎の変更点など）

InnoDB in MySQL 5.5

- パフォーマンス・スケーラビリティ改善
  - マルチ buffer pool
  - マルチ rollback segment
  - purgeスケジューリングの改善
  - insert buffering に加え delete buffering と purge buffering も
  - Linuxでのネイティブ非同期 I/O
  - Windows版の性能調整
- モニタリング
  - Performance schema への対応
- UTF-32 サポート

InnoDB in MySQL 5.6.2（開発版）

- パフォーマンス・スケーラビリティ改善
  - kernel_mutex の分割
  - マルチ purge thread
  - Change buffer 処理量の調整
  - データディクショナリのメモリ消費量を調整可能に
    （※既存オプション table_definition_cache の設定値に連動する）
  - 索引の統計情報をシステムファイル内に保存することが可能
    （再起動後に同じ索引に対する統計情報の再取得を回避可能）
  - MRR(Multi-Range Read)/ICP(Index Condition Pushdown) への対応

InnoDB in MySQL 5.6.2（開発版）

- モニタリング
  - Information schema : InnoDB Metrics Table
  - Information schema : InnoDB システムテーブル
  - Information schema : buffer pool の内容

InnoDB in MySQL 5.6.3（開発版）

- パフォーマンス・スケーラビリティ改善
  - トランザクションログファイルのサイズ制限増加（4GB → 512GB）
  - UNDO (rollback segment) 専用データファイルのサポート
  - ファイル伸張時の競合の改善
  - デッドロック検知性能の改善
  - スレッドのスケジューリングの改善
  - 高速なチェックサム方式を選択可
  - buffer pool に現在どのページが保持されているかの情報のダンプとその情報に基づいてページをロードする機能

InnoDB in MySQL 5.6.3（続き）

- データページ破壊を検出した場合、その表が壊れている扱いをし、即クラッシュはしないように変更
- 索引のキーの長さ制限の拡張（オプション）（767 → 3072）

InnoDB in MySQL 5.6.4（開発版）

- パフォーマンス・スケーラビリティ改善
  - リードオンリートランザクションの最適化
- InnoDB Full-Text Search（全文検索）
  - トランザクション対応
  - Partitioned inverted index
  - FTS index cache for Doc ID/Position
  - CJK は未サポート

InnoDB
パフォーマンスチューニング

InnoDB以外の設定

- Linuxの場合
  - IOスケジューラ： elevator = noop（for RAID/SSD）
    （HDDやローエンドRAIDの場合はdeadlineも試して決めても良い）
  - マウントオプション： noatime

- my.cnf は my-innodb-heavy-4G.cnf をベースに
  - max_connections、table_open_cache を十分な大きさに
    （リソース{ulimit -u -n で確認}が足りなければ勝手に小さくされる。起動ログで確認。）
  - tmpdir に高速で十分な容量のストレージを指定
    （Fast Index Creationは一時ファイルを使用するため。）
  - query_cache_type = false
    （更新系主体の処理の場合 mutex競合の元でしかなくなるので無効に）

InnoDBの設定

- 基本
  - innodb_file_per_table = true
    （デフォルトの値ではないが、ほぼ必須）
  - innodb_flush_method = O_DIRECT
    （データファイルアクセスにOSのキャッシュを無駄に消費しないため）
  - innodb_buffer_pool_size
    （可能な限り大くする。が、下記ログファイル全体をOSがキャッシュできる程度の余裕を持たせると更新系の性能に利点が大きい。）
  - innodb_log_file_size * innodb_log_files_in_group
    （最大リカバリ時間に影響するので考慮しつつ、大きくする。大きくしすぎてOSのキャッシュに乗らなくなると性能が悪くなる場合もある。）

InnoDBの設定

- 【必要に応じた】高度な設定
  - innodb_buffer_pool_instances = 2以上 (5.5~)
    （buf_pool->mutex の競合が多く見られる場合）
  - innodb_purge_threads = 1以上 [default: 0 (5.5)]
    （purgeが間に合わずに、History Listが肥大化する場合）
  - innodb_checksum_algorithm = 'crc32' (5.6~)
    （プロファイリングでチェックサムの計算が重い場合。過去バージョンとの互換性に注意）
  - innodb_undo_directory (5.6~)
    （書き込みが最も多いこのシステムデータファイルを書き込みが高速なストレージに置く）
  - innodb_sync_array_size = [CPU*2くらい] (5.6~)
    （mutex/rw_lockのイベント待ち処理の並列性に影響。）

InnoDBの設定

- SSDを意識した設定
  - Linux native AIO を利用 (5.5~)
    （SSDはアクセスが速いので、InnoDBのAIOエミュレーションでは少し非効率）
  - innodb_flush_neighbors = false (現状5.6~)
  - innodb_random_read_ahead = false
  - innodb_read_ahead_threshold = 0
    （シーケンシャルアクセスの利点は無いので、周辺ページをついでにIOする必要は無い）
  - innodb_page_size = (4K | 8K | [16K]) (5.6.4~)
    （SSDのページサイズ＝ファイルシステムのブロックサイズ　とした上で、InnoDBのページサイズも合わせることが可能。しかし、効果は未検証）

Q&A

Hardware and Software

ORACLE

Engineered to Work Together


ORACLE®