Oracle NoSQL Database at OOW 2017

- CON6544 – Oracle NoSQL Database Cloud Service
  - Monday 3:15 PM, Moscone West 3008
- CON6543 – Oracle NoSQL Database Introduction
  - Tuesday, 3:45 PM, Moscone West 3008
- **CON6545 – Oracle NoSQL Database Data Modelling**
  - **Wednesday, 11:00 AM, Moscone West 3008**
- HOL7611 – Oracle NoSQL Database – Cloud Service
  - Tuesday, 5:45 PM Hilton Union Square – Continental Ballroom 6
  - **Wednesday, 4:45 PM Hilton Union Square – Continental Ballroom 6**
- Demo Station – Moscone West

Data Modelling in Oracle NoSQL Database

October 1–5, 2017
SAN FRANCISCO, CA

Dave Rubin
Director of NoSQL Database Development

Oracle
October, 2017

ORACLE

Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Oracle NoSQL Database – Overview

- Highly available, horizontally scalable, distributed shared nothing database
- Predictable low latencies
- Automatic sharding
- Online elastic scale out/scale in
- Multi-model
  - Key/Value
  - Table
  - Document (ad-hoc JSON)
  - Property Graph
- Multi-datacenter support

Oracle NoSQL Database – Overview

- Security
  - Kerberos for authentication, SSL for confidentiality
  - Roles, groups, and privileges at the table level for authorization
- Time-to-live – Automatic aging of data
  - Default at table level
  - Override for each record if desired
- Streaming subscriptions
  - Subscribe to inserts, updates, deletes on a table
  - Delivered to client process via ReactiveStreams API
  - Horizontally scalable
  - Highly available via checkpointing

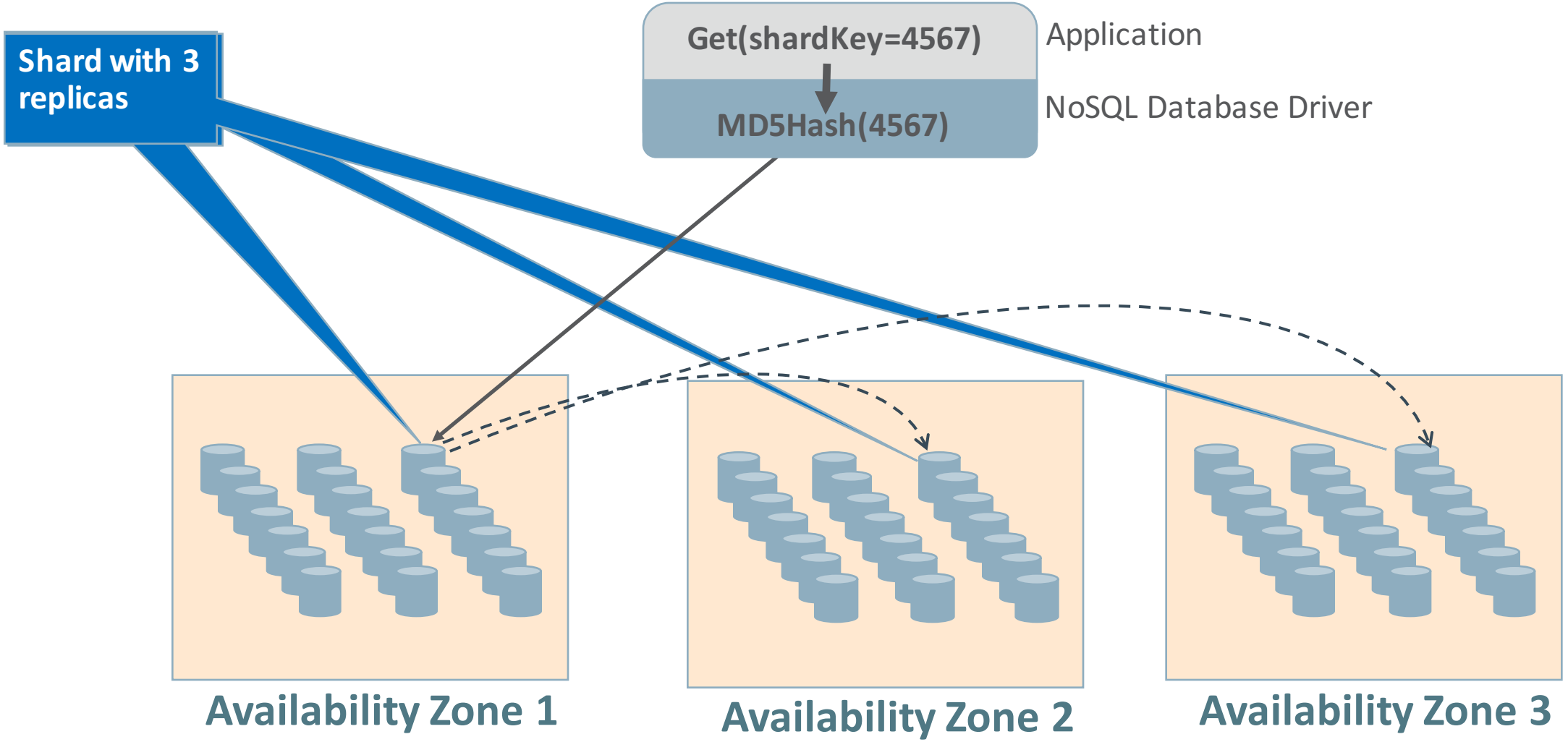
Oracle NoSQL Database – Overview

- Large object streaming
  - Split large objects into 256k chunks
  - Spray across cluster for writes
  - Re-assemble for reads
- Scalar datatypes
  - Integer, binary, boolean, double, enum, float, long, number, string, timestamp
- Non-scalar datatypes
  - Array, map, record, JSON
- Parent-child table traversal
  - Child records live on same shard with parent

Oracle NoSQL Database – Overview

- Rich secondary indexing
  - Secondary indexes updated atomically with primary data
  - Full JSON path expression support
  - Indexes on non-scalar datatypes
  - Range scans, forward or reverse
- Access via API or SQL
  - Get, put, scan APIs for raw key/value pairs and tables
  - SQL for rich access to JSON, more complex filtering expressions
  - Support for conjunctions and disjunctions in filter expressions

Oracle NoSQL Database – Distributed Systems Concepts



Oracle NoSQL Database – Data Modeling Concepts

- Data distribution and shard keys
- ACID Transactions
- Non-scalar datatypes versus child tables
- Workload characteristics
  - Read/write mix
  - Durability and consistency tradeoffs
- Flexibility versus performance and cost
  - Scalar versus non-scalar attributes
  - Ad-hoc JSON versus fixed schema

Data Distribution – Shard Keys Matter

Similar to Oracle Database Hash Partitioning

- Determines how data is distributed and ultimately scaled out
- Choose a shard key that has large cardinality
  - Gender is bad (very small range of values)
  - ID is good (scales out with number of users)

```
CREATE TABLE user( id INTEGER, surname STRING, familiarName STRING,
       gender ENUM (male, female),
       PRIMARY KEY (SHARD(id))
```

- ACID transactions are shard local

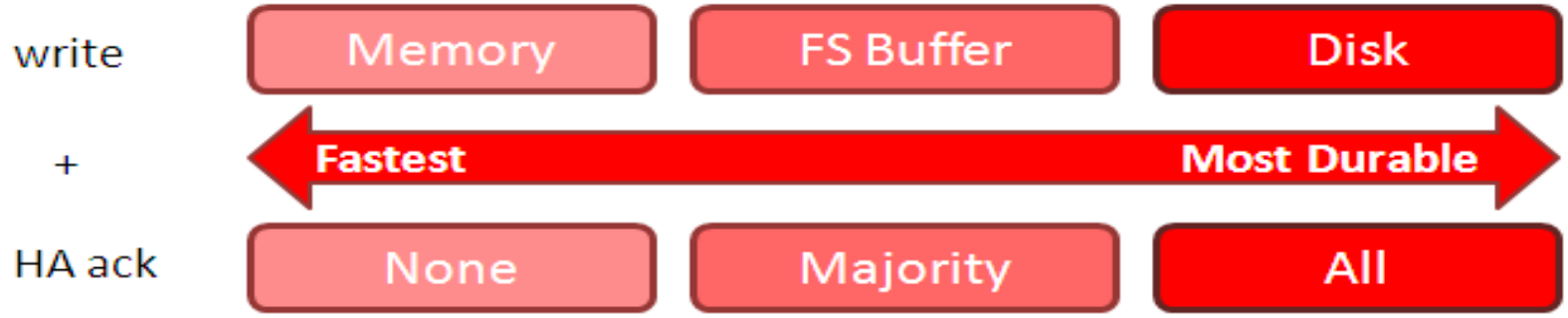
```
CREATE TABLE user.folders.inbox(folderID INTEGER, msgID INTEGER,
       PRIMARY KEY(folderID))

CREATE TABLE user.folders.deleted(folderID INTEGER, msgID INTEGER,
       PRIMARY KEY(folderID))
```

ACID Transactions

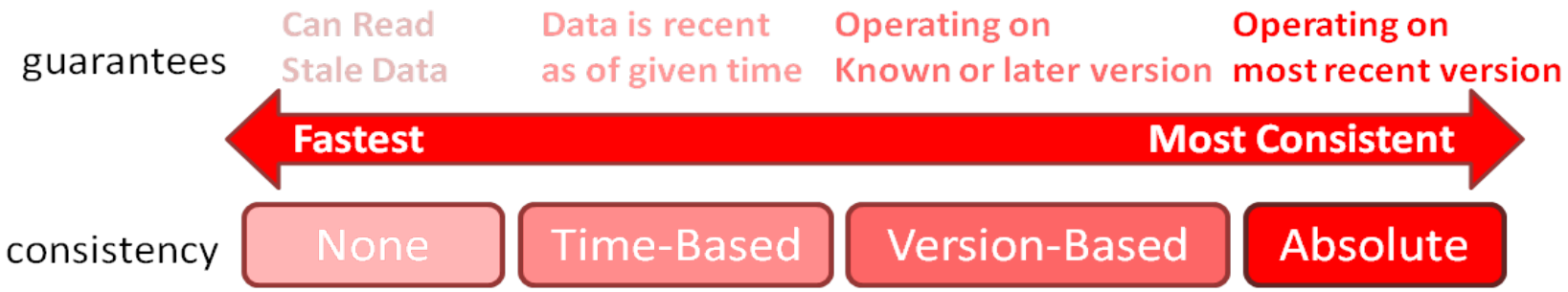
- Shard local only – Choose your shard key wisely
  - Single writes are ACID be default
  - Collection level transactions via API call – All records must have the same shard key
    - Example – Multi-select ten emails, move from inbox to another folder (shard key is userID)
      - Must be atomic
      - Must exhibit consistent reads when UI is refreshed
- Relaxed consistency
  - To favor latency over data recency
  - Increase throughput - scale the reads across replicas
- Relaxed durability
  - To favor latency over data recoverability

ACID Transactions – Tune-ability

- Configurable **D**urability Policy



- Configurable **C**onsistency Policy



Non-Scalar Datatypes versus Child Tables

Similar to master-detail relationships in the Oracle Database

- Non-scalar datatypes
  - Embedded objects
  - Modeling 1 to N relationships
    - A person with multiple addresses (home, office, bill-to, ship-to)
- Non-scalar types supported in Oracle NoSQL Database
  - JSON document – Most flexible, highest cost
  - Arrays – Good flexibility, less costly
  - Records – ”Fixed format” document… Good tradeoff for JSON
- Convenient and easy to use
- Not the best for extreme velocity updates

Non-Scalar Datatypes versus Child Tables

- Child Tables
  - Modeling 1 to N relationships
    - A sensor with 1000 events per second

```
CREATE TABLE sensor(sensorID INTEGER, sensorType INTEGER,
       PRIMARY KEY(sensorID))

CREATE TABLE sensor.sensorEvents (eventTime TIMESTAMP, eventType INTEGER,
              eventValue DOUBLE
       PRIMARY KEY(eventTime, eventType))
```

- Very efficient for extreme velocity data
  - Everything is an insert – Optimal for “append only” storage system
- Not as flexible as non-scalar datatypes

Secondary Indexes

- Very useful for JSON documents
  - When there is no “natural” primary key
  - JSON path expressions and arrays are supported
  - Utilized by rich JSON SQL via heuristics
- Each secondary will cause overhead for writes
  - Balancing replica ack durability may be an acceptable tradeoff
  - May not be suitable for very low write latency sensitive applications
- Primary key encoding can alleviate secondary index overhead
  - Key prefix searches
    - Shard local if full shard is specified in filter expression
    - B-tree prefix scan at storage layer

Workload Characteristics

- High volume data ingest, limited simple queries, temporal data
  - Use TTL to delete old data very efficiently
  - Favor child tables over non-scalar datatypes
  - Utilize primary keys for queries
    - Key prefixing for partition pruning – Attributes for query as leading columns in primary key
    - Key only scans as much as possible
    - Embed time in primary, range scan on primary key columns
- High volume reads, limited writes
  - Singleton primary key reads whenever possible
    - Key prefixed range scans also perform very well (using entire shard key)
  - Secondary index scans good when primary key not usable
  - Loosely consistent reads when possible
  - Avoid non-indexed table scans

Summary - Flexibility versus Performance and Cost

Knowledge is power

- Scalar, non-scalar, and parent/child tables for 1 to N relationships
  - Scalar attributes perform best, least flexible
  - Keep arrays and maps as small as possible
    - Arrays of scalars are preferable
    - Maps are expensive – contain attribute names redundantly
    - Records – less expensive than maps. Embedded objects with fixed schema.
  - Parent child tables
    - Very efficient for write-heavy workloads – artifact of log structured storage
    - More flexible for fine grained authorization
    - More challenging for queries
- Ad-hoc JSON
  - Extremely flexible
  - Schema redundancy
    - Higher cost on storage and compute

Oracle NoSQL Database Cloud Service – Coming Soon

- Fully managed, multi-tenant, provisioned throughput service
- Buy, connect, and go
  - Purchase cloud credits
  - Write your application
  - Connect to the service, create table with reads/sec, writes/sec, GB storage
  - Start writing and reading data
- Scaling the service is our problem
- Maintaining predictable latencies is our problem
- Maintaining high availability is our problem
- **You focus on delivering business value to your customers**

Safe Harbor Statement

The preceding is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.


ORACLE®