




Coherence Product Features – Querying the Cache
Oracle Coherence Workshop



Agenda

- Introduction to the QueryMap Interface
- Using Query Criteria to return data
- Understanding Filters
- Aggregation of Data
- Using Indexes
- Q&A




Copyright 2007 2

Objectives

After completing this lesson, you should be able to:

- Describe the QueryMap interface
- Understand how to return a set of data given a query criteria
- Describe the various filters available
- Understand how to aggregate data in a cache using InvocableMap interface
- Understand how to apply an index to help aggregation performance



Copyright 2007 3

Querying and Aggregating the Cache



Query of the Cache

- So far, we have only retrieved data by object ID
 - `myObject = NamedCache.get(id);`
- What if we want to do things like:
 - Queries other than by primary key, like this SQL statement:
 - `SELECT * FROM orders WHERE order_amount > 100`
 - Aggregations
 - `SELECT SUM(order_amount) FROM orders`
 - Queries with sorting
 - `SELECT id, name, order_amount FROM orders WHERE open_status = 1 ORDER BY order_amount`

Coherence doesn't support SQL, but there are powerful ways to query the cache!

QueryMap Interface

- Use `com.tangosol.util.QueryMap` interface to search for Values or Keys
- Use Filters to restrict searching and thus results
- Filtering occurs at Cache Entry Owner
 - ie: In Partitioned Topology, Primary Partitions do the filtering
- Use QueryMap interface to define Indexes to allow for search optimization
- Create Continuous View of entries based on a Filter with real-time events dispatch
 - Perfect for client applications "watching" data

QueryMap Interface



Copyright 2007

ORACLE
7

QueryMap Interface - methods

- **Set entrySet(Filter filter)**
 - Return a set view of the entries that satisfy the criteria expressed by the filter.
- **Set entrySet(Filter filter, Comparator comparator)**
 - As above but iteration over the set will occur in ascending order according to the comparator.
- **Set keySet(Filter filter)**
 - Return a set view of the keys contained in this map for entries that satisfy the criteria expressed by the filter.
- **void addIndex(ValueExtractor extractor, boolean fOrdered, Comparator comparator)**
 - Add an index to a QueryMap.
- **void removeIndex(ValueExtractor extractor)**
 - Remove an index from this QueryMap.

Copyright 2007

ORACLE
8

Example Object in Java

```
public class Trades implements ExternalizableLite{
    private int id;
    private String symbol;
    private BigDecimal price;

    public Person (...){}
    public String getSymbol(){
        return symbol;
    }
    public boolean isOpen(){
        if (...) return true;
        else return false;
    }
    public BigDecimal getPrice(){
        return price;
    }
}
```

We will perform a
query on these
methods

Copyright 2007

ORACLE
9

QueryMap Interface - Examples

- A set containing all of the open trades

```
NamedCache trades = CacheFactory.getCache("trades");
Set openTrades = trades.entrySet(new
    EqualsFilter("isOpen", BOOLEAN.TRUE));
```
- A set containing trades that have the symbol ORCL:

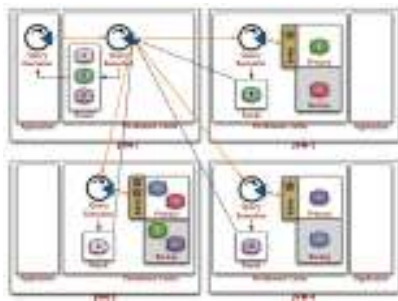
```
Set stockORCL = trades.entrySet( new
    LikeFilter("getSymbol", "ORCL%"));
```
- A set containing trades that have symbol ORCL and are open

```
Set openKeys = people.keySet(
    new AndFilter(
        new LikeFilter("getSymbol", "ORCL%"),
        new EqualsFilter("isOpen", BOOLEAN.TRUE)));
```

Indexes

- Each application using Coherence may suggest the same set of indexes when it starts
- There is no downside to an application blindly suggesting indexes regardless of whether another application has already suggested the same indexes
 - If an index has already been created, addIndex() is a no-op
- Indexes are maintained by Cache Entry Owners
 - ie: For Partitioned Topology, the Primary Partitions maintain their own indexes
- To sort entries (like an ORDER BY in SQL), specify that the index is sorted

Indexes



Index Examples

- Suggest an index for trades based on their portfolio. Queries using this index will be sorted (like an ORDER BY), and use natural ordering (hence the null).

```
trades.addIndex(  
    new ReflectionExtractor("getPortfolio"),  
    true, /*sort */  
    null); /* optional comparator */
```

- Suggest an index for trades based on their market. Don't use ordering

```
trades.addIndex(  
    new ReflectionExtractor("getMarket"),  
    false, /* do not sort */  
    null);
```

Features : InvocableMap Interface

- Execute processors against an Entry, a Collection or a Filter
- Executions occur in parallel (aka: Grid-style)
- No "workers" to manage!

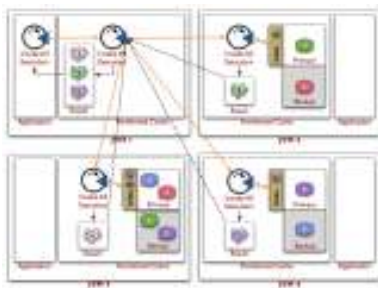
- Processors may return any value

```
trades.invoke(  
    new EqualsFilter("getSecurity", "ORCL"),  
    new StockSplit(2.0));
```

- Aggregate Entries based on a Filter

```
positions.aggregate(  
    new EqualsFilter("getSecurity", "ORCL"),  
    new SumFilter("amount"));
```

Features : InvocableMap Interface



Entry Aggregators

- `com.tangosol.util.InvocableMap.EntryAggregator` are agents that aggregate values from Entries
 - Sum, Average, Count, Max, Min, Distinct, GroupBy, Having...\
- Equivalent to “agents” executing services in parallel on the data in the cluster
- Aggregation...
 - must not mutate Entries
 - is for data extraction and aggregation only!

Entry Aggregators

- **Object aggregate(Collection keys, InvocableMap.EntryAggregator aggregator)**
 - Perform an aggregating operation against the entries specified by the passed keys
- **Object aggregate(Filter filter, InvocableMap.EntryAggregator aggregator)**
 - Perform an aggregating operation against the set of entries that are selected by the given Filter

Examples

- The total value of the open orders

```
BigDecimal result = orders.aggregate(  
    new EqualsFilter("isOpen", Boolean.TRUE),  
    new BigDecimalSum("getValue"));
```
- The categories of books on sale over \$25

```
Set categories = stock.aggregate(  
    new AndFilter(  
        new EqualsFilter("isOnSale", Boolean.TRUE),  
        new GreaterThanFilter("getPrice", 25)),  
    new DistinctValues("getCategory"));
```

True/False Quiz

- 1) To run a parallel query on each node of the cluster, your application needs to launch and manage worker threads.
- *FALSE. Coherence automatically performs QueryMap and EntryAggregators in parallel.*
- 2) Indexes improve the performance of queries.
- *TRUE.*
- 3) If you call addIndex() on an attribute that already has an index, an exception is thrown.
- *FALSE. It's a no-op.*

For More Information

<http://search.oracle.com>



or

<http://www.oracle.com/products/middleware/coherence/index.html>

Q&A



ORACLE IS THE **INFORMATION** COMPANY

ORACLE®
