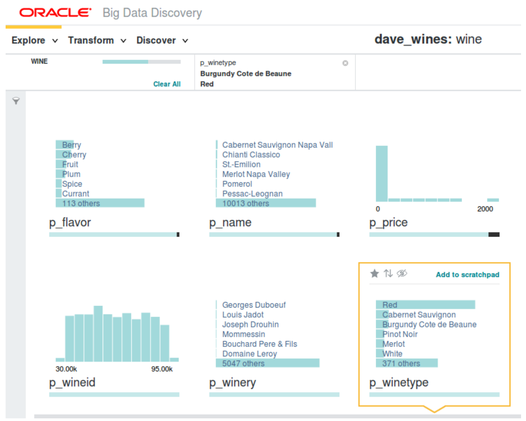
Applying Streaming Algorithms to Data at Rest

Mashhood Ishaque, Ligia Nistor and Kevin Backhouse
{mashhood.ishaque, ligia.nistor, kevin.backhouse}@oracle.com
Oracle Corporation, Cambridge, MA, USA

Guided Navigation

- Customers use Oracle Big Data Discovery for finding:
  - Most frequent elements in their datasets
  - Number of elements of a particular kind



Guided Navigation [5] (multiple properties)

! We have implemented streaming algorithms on gigabytes of data at rest, with dramatic performance improvements.

Previous Approaches

✖ Count number of distinct elements using Count aggregator (written in EQL – Endeca Query Language)

```
RETURN "RESULTS" AS
 SELECT COUNT(1) AS "C"
 FROM "Base" "Base"
 GROUP BY members("tags") AS "tags"
 ORDER BY "C" DESC
 PAGE(0, 10)
```

- Uses sorting and grouping: expensive and requires extra materializations of intermediary tables
- Implemented using in-memory hash table: need 2367 MB to represent 150 million unique elements!

✖ Use sampling to get approximation of number of distinct values

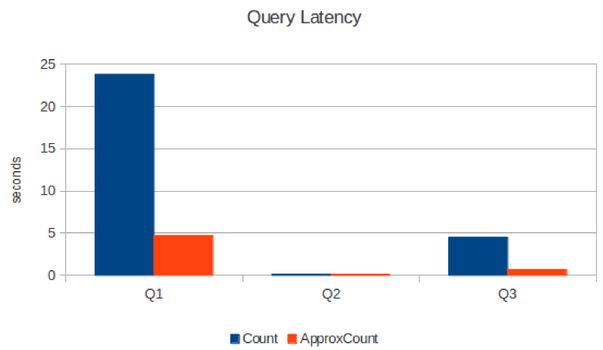
- It does not give accurate answer if number of distinct elements is small
- Very likely that the sample will not contain all different kinds of elements

SpaceSaving and FrequentK

- Internal *FrequentK* operator based on SpaceSaving [1,2] algorithm
- Finds the top K most frequently occurring elements, ordered by frequency
- Implemented by ApproxCount operator in EQL

```
RETURN "RESULTS" AS
 SELECT APPROXCOUNT(1) AS "A"
 FROM "Base" "Base"
 GROUP BY members("tags") AS "tags"
 ORDER BY "A" DESC
 PAGE(0, 10)
```

Performance and accuracy results



Query Latency for ApproxCount

- Each query returns the top 10 elements that satisfy that query
- First query looks for the dominant hashtags in the Twitter dataset
- Second query returns the countries of the tweets
- Third query looks for the most popular Twitter mentions. The results are shown in the table below:

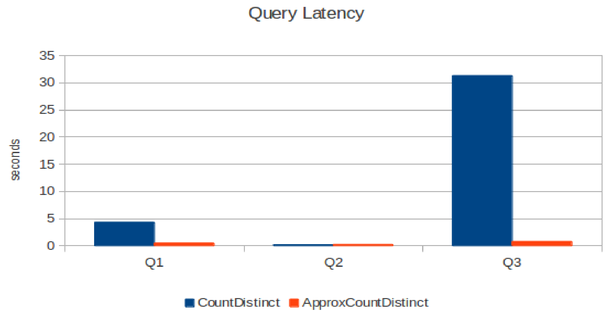
	Count		FrequentK	
#	Value	Count	Value	Count
1	YouTube	752808	YouTube	752808
2	justinbieber	278185	justinbieber	278185
3	wenyunchao	202077	wenyunchao	202077
4	NiallOfficial	169866	NiallOfficial	169866
5	Harry_Styles	113752	Harry_Styles	113752
6	Real_Liam_Payne	102755	Real_Liam_Payne	97534
7	getglue	93209	getglue	86281
8	zaynmalik	87139	zaynmalik	78585
9	JawabJUJUR	73287	JawabJUJUR	67838
10	Louis_Tomlinson	70140	foursquare	63386

Query Accuracy for FrequentK (Twitter mentions)

HyperLogLog and ApproxCountDistinct

- We implemented the HyperLogLog algorithm [3,4] as an aggregator called ApproxCountDistinct in EQL, for counting the number of distinct elements
- The aggregator is very inaccurate when there is a small number of distinct values ➔ in this case we default to our CountDistinct aggregator

Performance and accuracy results



Query Latency for ApproxCountDistinct

Query	CountDistinct	ApproxCountDistinct	Error (%)
Q1	3548212	3530581	0.49
Q2	249	249	0
Q3	23703655	23956593	1.06

Query Accuracy for ApproxCountDistinct

Benefits

- We only sort a constant number of values (a few thousand)
- We give accurate counts when there is a small number of distinct values
- We return approximate counts when there is a large number of distinct values, but with a provably small error
- We use well studied algorithms that have been shown to perform well in practice
- Streaming algorithms are usually embarrassingly parallel

References

1. Radu Berinde, Piotr Indyk, Graham Cormode, and Martin J. Strauss. Space-optimal heavy hitters with strong error bounds. 2010
2. Ahmed Metwally, Divyakant Agrawal, and Amr El Abbadi. Efficient computation of frequent and top-k elements in data streams. 2005
3. Philippe Flajolet, Eric Fusy, Olivier Gandouet, and Frederic Meunier. Hyperloglog: The analysis of a near-optimal cardinality estimation algorithm. 2007
4. Stefan Heule, Marc Nunkesser, and Alex Hall. Hyperloglog in practice: Algorithmic engineering of a state of the art cardinality estimation algorithm. 2013
5. Daniel Tunkelang. Faceted Search. Synthesis Lectures on Information Concepts, Retrieval, and Services. 2009