Article
Benjamin De Boe · Dec 15, 2021 4m read

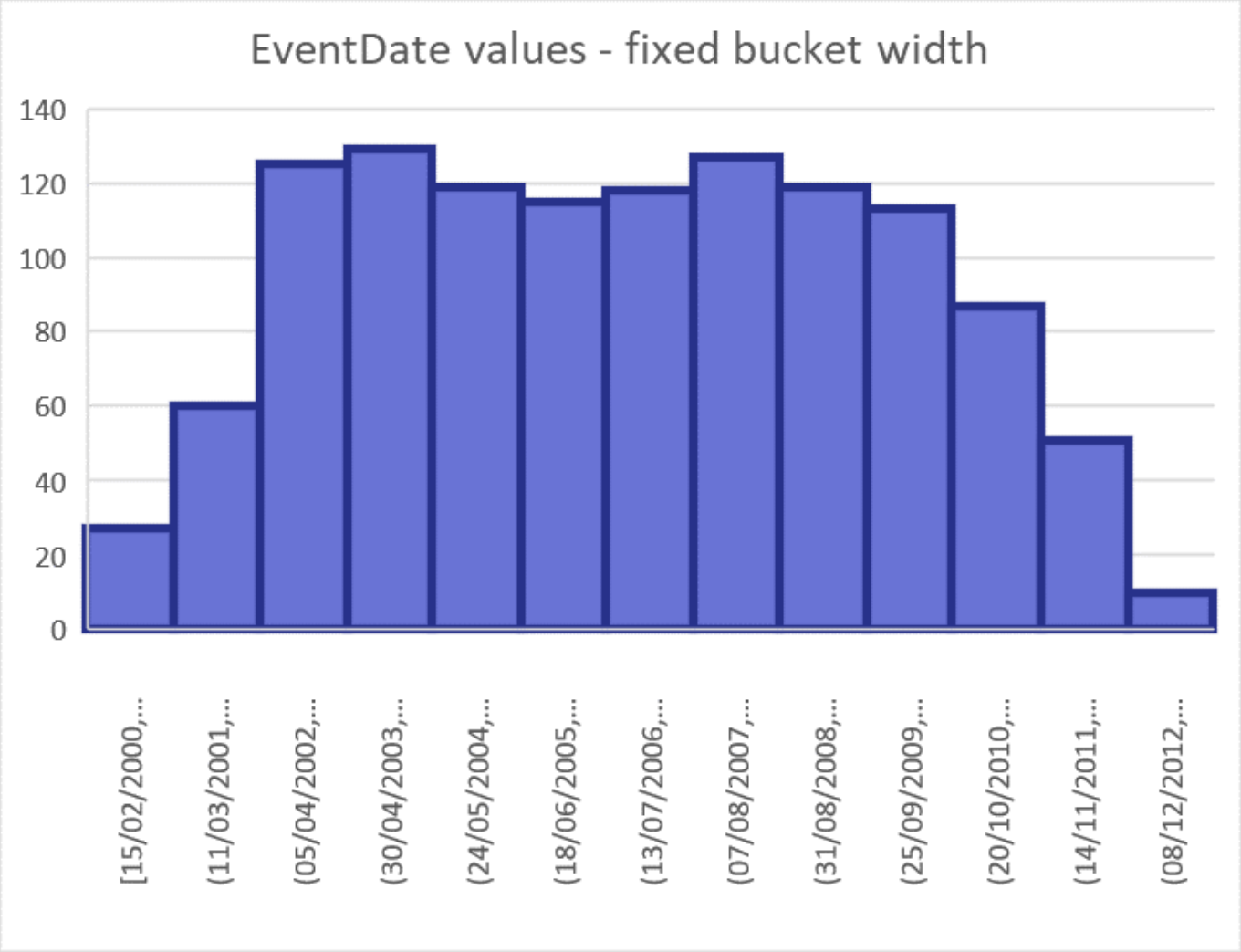
2021.2 SQL Feature Spotlight - Advanced Table Statistics

This is the third article in our short series around innovations in IRIS SQL that deliver a more adaptive, high-performance experience for analysts and applications querying relational data on IRIS. It may be the last article in this series for 2021.2, but we have several more enhancements lined up in this area. In this article, we'll dig a little deeper into additional table statistics we're starting to gather in this release: **Histograms**

What is a histogram?

A histogram is an approximate representation of the data distribution of a numerical field (or more broadly data that has a strict ordering). Knowing the smallest, largest and average value for such a field is helpful, but it doesn't tell you as much about how the data is distributed between those three points. That's where a histogram comes in, dividing the range of values into *buckets* and counting how many field values appear in each bucket.

This is a pretty flexible definition and you can still choose to take the size of the buckets such that the buckets are equally "wide" in terms of field values, or equally "large" in terms of number of sampled values covered. In the latter case, each bucket has the same percentage of values in it and therefore the buckets represent percentiles. The chart below plots a histogram for the EventData field in the Aviation Demo dataset, using the same bucket width expressed in number of days.



Why would I need a histogram?

Say you are querying this dataset for all events prior to 2004 in the state of California:

```
SELECT * FROM Aviation.Event WHERE EventDate < '2004-05-01' AND LocationCountry = 'California'
```

In our earlier article on Run Time Plan Choice, we already discussed how we capture the selectivity and potential outliers for a field like LocationCountry in the table statistics. But such statistics for individual field values aren't that practical for that < condition on EventDate. In order to calculate the selectivity of this condition, you would need to aggregate the selectivity of all possible EventDate values up to May 1st 2004, which can be a pretty demanding query all by itself rather than the kind of a quick estimate you can afford at query planning time. This is where histograms help.

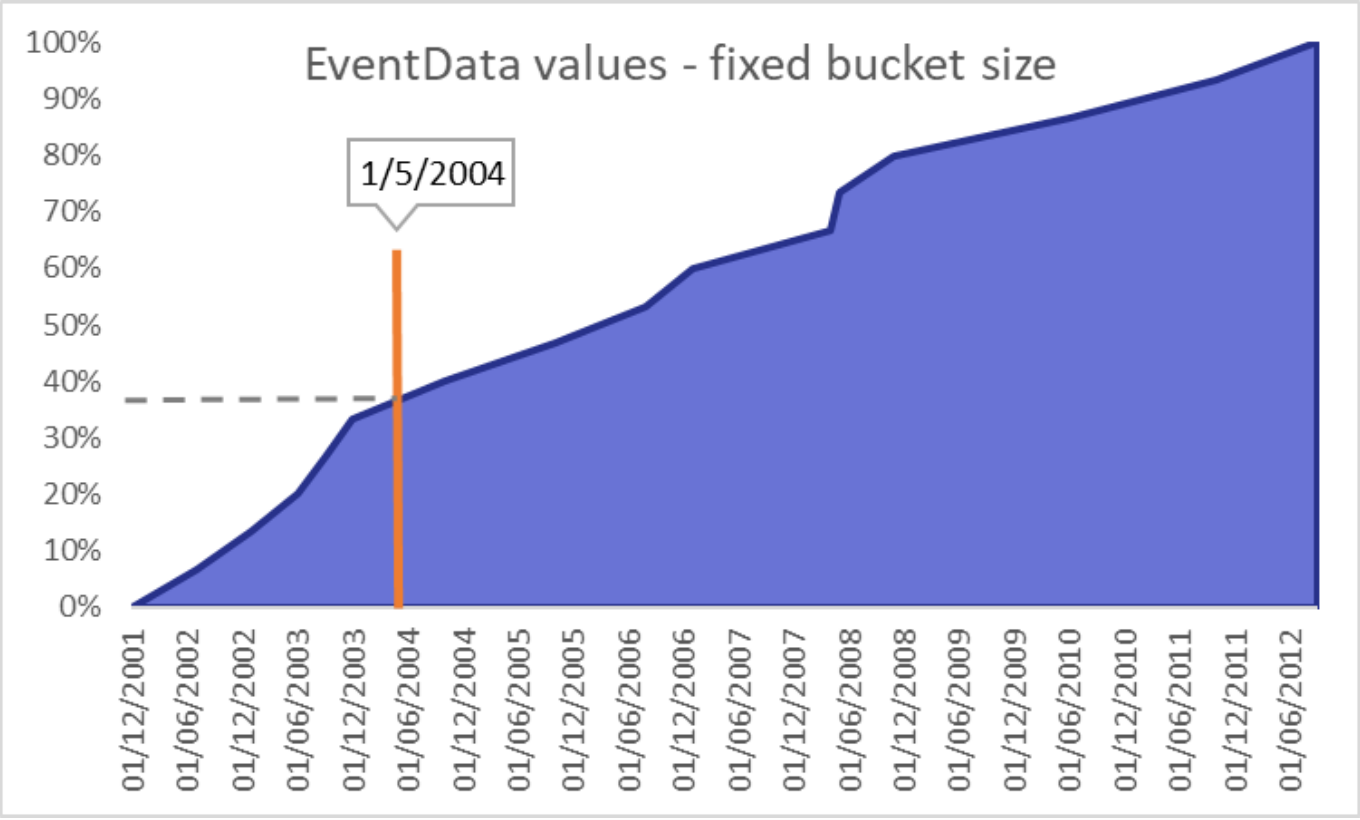
Let's look at our histogram data for the EventDate value distribution, this time dividing the data into 16 buckets of equal size, each holding 6,667% of the data. This way things translate more easily into percentiles and selectivity numbers we can use for query cost estimations. To read this table, let's look at the fourth row: 20% of the values (3 buckets of 6,667% each) precede this bucket's lower bound of June 22, 2003, and it holds a 6,667% further values, up to September 19, 2003.

Bucket	Percentile	Value
0	0%	21/12/2001
1	7%	02/07/2002
2	13%	19/01/2003

3	20%	22/06/2003
4	27%	19/09/2003
5	33%	30/12/2003
6	40%	01/10/2004
7	47%	01/10/2005
8	53%	20/08/2006
9	60%	14/01/2007
10	67%	02/04/2008
11	73%	14/05/2008
12	80%	29/11/2008
13	87%	01/06/2010
14	93%	30/10/2011
15	100%	26/09/2012

Our cutoff date used in the example query above (May 1, 2004) is in the fifth bucket, and has between 33% and 40% of the values preceding that date. As buckets get smaller, we can consider the distribution *within* them to be approximately uniform and simply interpolate between the lower and higher bounds, which in this case leads to a selectivity of about 37%, which we can use in our query cost estimation.

Here's another way to visualize our use of histograms, plotting it as a cumulative distribution chart. We can see how the line drawn for May 1, 2004 on the X axis (the values), translates to 37% on the Y axis.



The above example uses a range condition with just an upper bound for clarity, but the approach obviously works just as well when using a lower bound or interval condition (e.g. using the BETWEEN predicate).

Starting with 2021.2, we're collecting histograms as part of table statistics for any collated field, including strings, and use it for estimating range selectivity as part of RTPC. Many real-world queries involve some range condition on date (and other) fields, so we're quite optimistic this IRIS SQL enhancement will benefit query planning for many of our customers and, as always, are eager to hear your experiences.

#Relational Tables #SQL #InterSystems IRIS