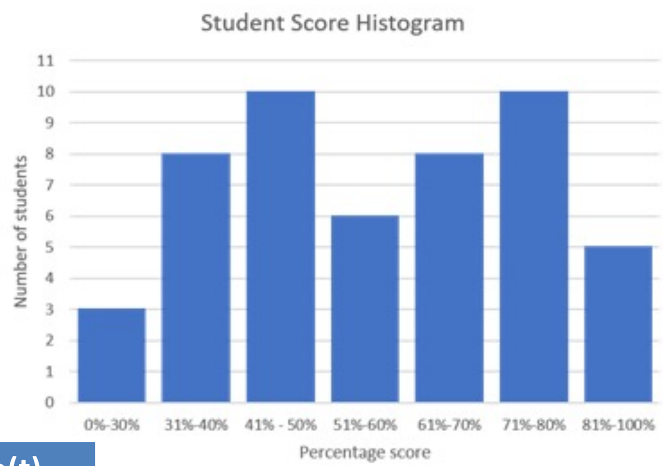
sample方法实现AQP

汇报人：李晓桐

- ✓ Design of Sample
- ✓ Analysis on Sample method
- ✓ Various methods of Sample
- ✓ Code for Sample

Review

- AQP = Approximate Query Process
- Goal: Accurate and Fast
- Histogram achieves the goal by divide data into various buckets
- Sample achieves the goal by sample from full data



index	value(t)
1	3
2	4
3	5
4	6
5	9
6	10
7	12
8	13
9	15
10	19

sample

sample index	sample value
6	10
3	5
5	9
3	5
9	15

Design of Sample: A simple example

- SELECT SUM(R.a) FROM R;
- R: <3, 4, 5, 6, 9, 10, 12, 13, 15, 19>
- size of full data N = 10
- sample size n = 5
- ground truth Q = 96
- How to sample?
- How to estimate?

index	value(t)
1	3
2	4
3	5
4	6
5	9
6	10
7	12
8	13
9	15
10	19

Design of Sample: A simple example

- Step 1: get sample
  - roll ten-sided die 5 times via approximate pseudorandom number generator (PRNG)
  - simple random sampling with replacement (SRSWR)
  - e.g sample index <6, 3, 5, 3, 9>
  - sample value<10, 5, 9, 5, 15>
- Step 2: calculate sum on sample
  - sum = 10+5+9+5+15 = 44
- Step 3: scale up estimation
  - estimation Y = 44 * N / n = 88

sample index	sample value
6	10
3	5
5	9
3	5
9	15

Analysis on Sample method

- ground truth = 96
- estimation = 88
- q-error = 96/88 = 1.09
- Important conclusion: Y is **unbias** estimation of ground truth
- Important conclusion: Error of Y is **bounded by variance**

index	value(t)
1	3
2	4
3	5
4	6
5	9
6	10
7	12
8	13
9	15
10	19

sample

sample index	sample value
6	10
3	5
5	9
3	5
9	15

Analysis on Sample method

- Y is **unbias** estimation of ground truth

$$E[Y] = E\left[\sum_{i=1}^{N} \frac{X_i t_i}{E[X_i]}\right] = \sum_{i=1}^{N} E\left[\frac{X_i t_i}{E[X_i]}\right] = \sum_{i=1}^{N} \frac{E[X_i] t_i}{E[X_i]} = \sum_{i=1}^{N} t_i.$$

$$= \quad Q = \sum_{j} t_j$$

Analysis on Sample method

- Error of Y is **bounded by variance**
- 95% chance our estimation is within ± 37.10
- q-error < 1.63

$$\sigma^2(Y) = E[(Y - E[Y])^2] = E[Y^2] - E^2[Y]$$

$$\begin{aligned}
\sigma^2(Y) &= E[Y^2] - E^2[Y] = E\left[\left(\sum_i \frac{X_i t_i}{\pi_i}\right)^2\right] - \left(\sum_i t_i\right)^2 \\
&= E\left[\sum_i \sum_j \frac{X_i t_i}{\pi_i} \frac{X_j t_j}{\pi_j}\right] - \sum_i \sum_j t_i t_j \qquad (2.3) \\
&= \sum_i \sum_j \frac{\pi_{ij} t_i t_j}{\pi_i \pi_j} - \sum_i \sum_j t_i t_j = \sum_i \sum_j \left(\frac{\pi_{ij}}{\pi_i \pi_j} - 1\right) t_i t_j .
\end{aligned}$$

Analysis on Sample method

- **Chebyshev Bounds**

$$\Pr[|Y-Q| \geq p^{-\frac{1}{2}}\sigma(Y)] \leq p$$

- **Hoeffding Bounds**

$$\Pr[|Y-E[Y]| \geq d] \leq 2\exp\left(-\frac{2d^2n^2}{\sum_i (hi_i - low_i)^2}\right)$$

- **Central limit Theorem**

定理 4.2 设 $X_1, X_2, \cdots, X_n, \cdots$ 为独立同分布的随机变量，$E(X_i)=a$，$\mathrm{Var}(X_i)=\sigma^2$，$0<\sigma^2<\infty$. 则对任何实数 x，有

$$\lim_{n\to\infty} P\left(\frac{1}{\sqrt{n}\sigma}(X_1+\cdots+X_n-na) \leqslant x\right) = \Phi(x) \quad (4.7)$$

这里 $\Phi(x)$ 是标准正态分布 $N(0,1)$ 的分布函数，即

$$\Phi(x) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x} e^{-t^2/2}\mathrm{d}t \quad (4.8)$$

注意 $X_1+\cdots+X_n$ 有均值 na，方差 $n\sigma^2$. 故

$$(X_1+\cdots+X_n-na)/(\sqrt{n}\sigma).$$

就是 $X_1+\cdots+X_n$ 的标准化，即使其均值变为 0 方差变为 1，以与 $N(0,1)$ 的均值方差符合.

Pros & Cons

- Simplicity
- Pervasiveness
- Extensive theory
- Immediacy
- Adaptivity
- Flexibility
- Insensitivity to dimension
- Ease of implementation

- Unsuitable for approximating the answer to queries that depend only upon a few tuples from the dataset
- Slower than histogram
- Sensitive to skew
- Do not support some important classes of aggregation queries

Pros

- PostgreSQL support various sample methods
- BERNOULLI sample: Scan the whole table and return N% of the total sampling records
- SYSTEM Sample: Scan the whole BLOCKS of the table and return N% of the total sampling blocks.

```
postgres=# select ctid,* from tbl TABLESAMPLE BERNOULLI (1)  limit 10;
  ctid    |  id  | loc  |  beginid  |   endid
----------+------+------+-----------+-----------
 (1,6)    |   76 | 2006 | 100034889 | 100035096
 (3,8)    |  218 | 3280 | 100105401 | 100105571
 (4,19)   |  299 | 708  | 100145769 | 100146449
 (6,25)   |  445 | 3195 | 100220431 | 100221192
 (7,3)    |  493 | 1867 | 100247252 | 100248048
 (9,5)    |  635 | 2125 | 100318350 | 100319087
 (10,51)  |  751 | 1151 | 100374936 | 100375883
 (12,20)  |  860 | 2302 | 100430532 | 100430674
 (12,33)  |  873 | 15   | 100438908 | 100439548
 (17,15)  | 1205 | 2540 | 100607913 | 100608198
(10 rows)
```

```
postgres=# select ctid,* from tbl TABLESAMPLE system (5)  limit 10;
  ctid    | id  | loc  |  beginid  |   endid
----------+-----+------+-----------+-----------
 (10,1)   | 701 | 2675 | 100348960 | 100349937
 (10,2)   | 702 | 4307 | 100349937 | 100350353
 (10,3)   | 703 | 475  | 100350353 | 100351093
 (10,4)   | 704 | 1611 | 100351093 | 100351171
 (10,5)   | 705 | 4307 | 100351171 | 100351692
 (10,6)   | 706 | 2841 | 100351692 | 100352448
 (10,7)   | 707 | 3680 | 100352448 | 100353372
 (10,8)   | 708 | 1085 | 100353372 | 100354108
 (10,9)   | 709 | 2137 | 100354108 | 100354314
 (10,10)  | 710 | 3381 | 100354314 | 100354905
(10 rows)

Time: 0.547 ms
```

https://www.alibabacloud.com/blog/postgresql-random-query-sampling-table-sample-method_599392

Cons

- unsuitable for approximating the answer to queries that depend only upon a few tuples from the dataset
- When query is highly selective (e.g 10 / 1 million), 1% sample is unlikely to contribute to accuracy
- ps: Histogram is widely used in commercial database systems

```
SELECT histogram_bounds FROM pg_stats
WHERE tablename='tenk1' AND attname='stringu1';

                              histogram_bounds
--------------------------------------------------------------------------------
 {AAAAAA,CQAAAA,FRAAAA,IBAAAA,KRAAAA,NFAAAA,PSAAAA,SGAAAA,VAAAAA,XLAAAA,ZZAAAA}
```

https://www.postgresql.org/docs/current/row-estimation-examples.html

Cons

- slower than histogram
- 5% sample → 20* speed up
- ps: histogram offers a faster speed O(1)

```
selectivity = mcv_selectivity + histogram_selectivity * histogram_fraction
            = 0.01833333 + 0.298387 * 0.96966667
            = 0.307669

rows        = 10000 * 0.307669
            = 3077  (rounding off)
```


https://www.postgresql.org/docs/current/row-estimation-examples.html

Cons

- sensitive to skew
- e.g ⟨3,4,5,6,9,10,12,13,15,10^99⟩
- sample 1: <3,4,5,6,9> → Y = 54
- sample 2: <10,12,13,15,10^99> → Y = 2*10^99 + 100

Cons

- Do not support some important classes of aggregation queries
- e.g SELECT SUM(R.a) FROM R WHERE R.b NOT IN (SELECT S.c FROM S)
- Hard to decide scope of sample

Various methods of Sample

- Simple Random Sampling With Replacement
- Simple Random Sampling Without Replacement
- Bernoulli and Poisson Sampling
  - Bernoulli: each tuple i is sampled w.p. p_i in which p_i is same for each i
  - Poisson: p_i can be different
- Stratified Sampling
  - group data into m different subclasses
  - perform sample in each subclass

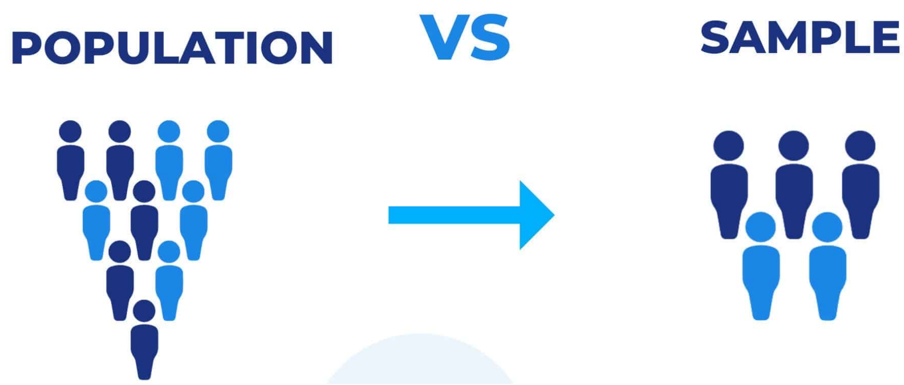
Code Example

```python
# get samples
sample_size = 5
sample_list = []
for i in range(sample_size):
    sample_idx = np.random.randint(0, len(full_data))
    sample_item = full_data[sample_idx]
    sample_list.append(sample_item)

# get estimation
est = get_sum_est(sample_list, full_size)

print(sample_list)
print(get_q_err(est, ground_truth))
```

```
[12, 4, 3, 15, 13]
1.0212765957446808
```

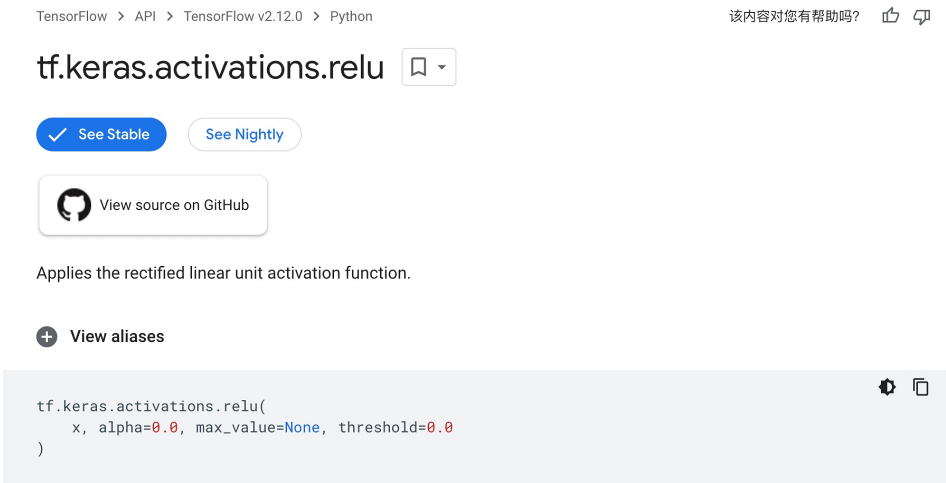
Take home message

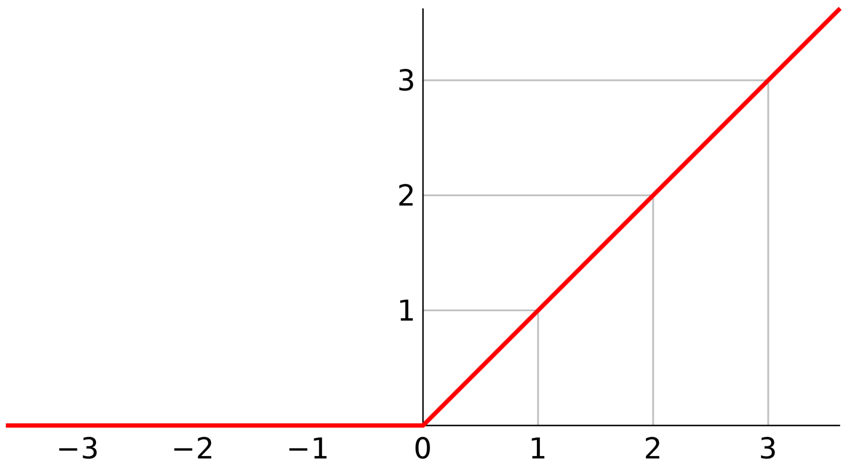
- Basic idea of Sample
- Accuracy of Sample is guaranteed by math
- Pros & Cons: sample is easy but fails in some situation
- various methods of sample
- Code example



Evaluation

- $loss = MSLE + relu(all\_online\_time - 5)/5 + relu(all\_offline\_time - 180)/180$





https://www.tensorflow.org/api_docs/python/tf/keras/activations/relu

https://zh.wikipedia.org/wiki/%E7%BA%BF%E6%80%A7%E6%95%B4%E6%B5%81%E5%87%BD%E6%95%B0

Reference

- Cormode, G. (2011). Synopses for Massive Data: Samples, Histograms, Wavelets, Sketches. Foundations and Trends in Databases, 4(1-3), 1–294. doi:10.1561/1900000004
- 概率论与数理统计（陈希孺）：file:///Users/xiaotong/Downloads/%E6%A6%82%E7%8E%87%E8%AE%BA%E4%B8%8E%E6%95%B0%E7%90%86%E7%BB%9F%E8%AE%A1%EF%BC%88%E9%99%88%E5%B8%8C%E5%AD%BA%EF%BC%89.pdf
- https://www.alibabacloud.com/blog/postgresql-random-query-sampling-table-sample-method_599392
- https://www.postgresql.org/docs/current/row-estimation-examples.html