Join Synopses for Approximate Query Answering

by

Swarup Acharya, Phillip B. Gibbons,
Viswanath Poosala, Sridhar Ramaswamy

Presented by,
Jeevan Kumar Gogineni
Saranya Gottipati
In this presentation we deal with...........

- Traditional Query processing
- The Problem with Joins
- The AQUA System
- Join Synopses
- Space Allocation
- Improved Accuracy Measures
- Maintenance policy
- Experimental results
- Something that were missing in this paper.
- Conclusion
Traditional Query processing

- Focused on Exact Answers
- For Larger Databases it took lot of time

So What we need?

- For complex aggregate queries based on statistical summaries of the full data, it is often advantageous to provide fast, approximate answers.
- Less access to Base relation

What motivated them to take approximate querying

- full precision of the exact answer is not needed, e.g., a total, average, or percentage
The Problem with Joins

- **Non-Uniform Result Sample:** In general, the join of two uniform random base samples is *not a uniform random sample of the output of the join*.

  The probability of any joined tuples to be in the former should be the same as their probability in the later.

- **Small Join output size:** The join of two random samples typically has very few tuples, even when the actual join selectivity is fairly high. This can lead to both inaccurate answers and very poor confidence bounds since they critically depend on the query result size.

*Def: Base samples:*- uniform random samples of each base relation

*TPC-D* represents a broad range of decision support (DS) applications that require complex, long running queries against large complex data structures.
The Aqua System

- The goal of Aqua is to improve response times for queries by avoiding accesses to the original data altogether.
- Aqua maintains smaller-sized statistical summaries, called synopses, on the warehouse and uses them to answer queries.

A data warehouse is a repository of an organization’s electronically stored data. Data warehouses are designed to facilitate reporting and analysis.
**SQL Query:**

```sql
SELECT N_NAME, COUNT(*), AVG(O_TOTALPRICE)
FROM REGION, NATION, CUSTOMER, ORDER
WHERE C_CUSTKEY = O_CUSTKEY AND
  C_NATIONKEY = N_NATIONKEY AND
  N_REGIONKEY = R_REGIONKEY AND
  R_NAME = 'ASIA'
GROUP BY N_NAME;
```

---

**Actual Answer**

<table>
<thead>
<tr>
<th>N_NAME</th>
<th>COUNT(*)</th>
<th>AVG(O_TOTALPRICE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHINA</td>
<td>18490</td>
<td>144683.3265</td>
</tr>
<tr>
<td>INDIA</td>
<td>17639</td>
<td>144225.7318</td>
</tr>
<tr>
<td>INDONESIA</td>
<td>17775</td>
<td>145205.5079</td>
</tr>
<tr>
<td>JAPAN</td>
<td>18219</td>
<td>144787.2551</td>
</tr>
<tr>
<td>VIETNAM</td>
<td>17913</td>
<td>143588.3381</td>
</tr>
</tbody>
</table>

5 row(s) processed.

**Time Taken (secs) = 655.93**

---

**Approximate Answer**

<table>
<thead>
<tr>
<th>n_name</th>
<th>count(*)</th>
<th>err1</th>
<th>avg(O_totalprice)</th>
<th>err2</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHINA</td>
<td>18120</td>
<td>3700</td>
<td>151200</td>
<td>9000</td>
</tr>
<tr>
<td>INDIA</td>
<td>17840</td>
<td>3700</td>
<td>143100</td>
<td>8800</td>
</tr>
<tr>
<td>INDONESIA</td>
<td>17060</td>
<td>3700</td>
<td>151800</td>
<td>9100</td>
</tr>
<tr>
<td>JAPAN</td>
<td>19340</td>
<td>3700</td>
<td>147200</td>
<td>8900</td>
</tr>
<tr>
<td>VIETNAM</td>
<td>17780</td>
<td>3700</td>
<td>141200</td>
<td>9000</td>
</tr>
</tbody>
</table>

5 row(s) processed.

**Time Taken (secs) = 15.95**
Join Synopses

- Effective solution for producing approximate join aggregates of good quality
- Our main contribution is to show that by computing samples of the results of a small set of distinguished joins, we can obtain random samples of all possible joins in the schema- distinguished joins as join synopses.
- Nodes correspond to Relations and whose edges correspond to every possible 2-way foreign key join for the schema.
- **Foreign Key Join Definition**
- Key result we prove is that there is a one-one correspondence between a tuple in a relation and a tuple in the output of any foreign key join involving and the relations corresponding to one or more of its descendants in the graph.
- A sample S.r of a relation ‘r’ can be used to produce another relation ζ( S.r ) called a join synopsis of ‘r’ that can be used to provide random samples of any join involving ‘r’ and one or more of its descendants.
Definition 4.1 Foreign Key Join: A 2-way join $r_1 \Join r_2$, $r_1 \neq r_2$, is a foreign key join if the join attribute is a foreign key in $r_1$ (i.e., a key in $r_2$). For $k \geq 3$, a $k$-way join is a $k$-way foreign key join if there is an ordering $r_1, r_2, \ldots, r_k$ of the relations being joined that satisfies the following property: for $i = 2, 3, \ldots, k$, $s_{i-1} \Join r_i$ is a 2-way foreign key join, where $s_{i-1}$ is the relation obtained by joining $r_1, r_2, \ldots, r_{i-1}$.
Join Synopses – Important Statements

- The subgraph of G on the k nodes in any K-way foreign key join must be a connected subgraph with a single root node.

- There is a 1-1 correspondence between tuples in a relation r1 and tuples in any -way foreign key join with source relation r1.

- The joining tuples in any relation other than the source relation will not in general be a uniform random sample of . So we need Distinct join synopses for each node/relation.

- Join Synopses definition
Definition 4.2 Join synopses: For each node $u$ in $G$, corresponding to a relation $r_1$, define $J(u)$ to be the output of the maximum foreign key join $r_1 \bowtie r_2 \bowtie \cdots \bowtie r_\kappa$ with source $r_1$. (If $u$ has no descendants in $G$, then $\kappa = 1$ and $J(u) = r_1$.) Let $S_u$ be a uniform random sample of $r_1$. Define a join synopsis, $J(S_u)$, to be the output of $S_u \bowtie r_2 \bowtie \cdots \bowtie r_\kappa$. The join synopses of a schema consists of $J(S_u)$ for all $u$ in $G$. ■
Allocation

- Optimal strategy for allocating the available space among the various join synopses when certain properties of the query work load are known.

- Discuss heuristic allocation when such properties of work load are not known.
Optimal Allocation

- Characterize a set $S$, of queries with selects, aggregates, group bys, and foreign key joins.
- For each relation $R_i$, we determine the fraction $f_i$, of the queries in $S$ for which $R_i$ is either the source relation in the foreign key join or the sole relation in a query without joins.
- Minimizing the average relative error bounds reduces the average relative errors over a collection of aggregate queries like COUNT, SUM and AVERAGE.
- Error bounds is inversely proportional to the $\sqrt{n}$, where $n$ is the number of tuples in the join sample.
- Thus the average relative error over the queries is proportional to
  $$\sum [f_i] / [\sqrt{n_i}]$$
  where $n_i$ is the number of tuples allocated to the join sample for source relation $R_i$
- Error bounds is inversely proportional to the $\sqrt{n}$, where $n$ is the number of tuples in the join sample.
Hueristic Allocation

- There are three strategies for allocating the available space among various join synopses, namely,
  - EqJoin
  - CubeJoin
  - PropJoin

- The allocation strategies using base samples are similar to the ones above. These are called as EqBase, CubeBase, and PropBase which are from base samples.
Improved Accuracy Measures

- Several popular methods for deriving confidence bounds for approximate answers
- Queries with foreign key joins can be treated as queries without joins
The Algorithm for Join Synopses is very simple.

If there is a deleted tuple, we have to **remove** it from the synopses. If there is an **added** tuple, we’ll decide with random probability ‘p’ whether it’s needed to be in the synopses, and if yes, we’ll add it with an appropriate join.
Experimental Evaluation

1. Join Synopses Accuracy

These graphs demonstrate the advantages of schemes based on join synopses over base sampling schemes for approximate join aggregates. Even with a summary size of only 0.1%, join synopses are able to provide fairly accurate aggregate answers.

(a) Date Interval = [1/1/94, 1/1/95]

(b) Summary Size = 1.5%
Experimental Evaluation

\[
\text{select avg(\text{l\_extendedprice}) from customer, order, lineitem, supplier, nation, region}
\]
\[
\text{where c\_custkey = o\_custkey and o\_orderkey = l\_orderkey and l\_suppkey = s\_suppkey}
\]
\[
\text{and c\_nationkey = s\_nationkey and s\_nationkey = n\_nationkey and n\_regionkey = r\_regionkey}
\]
\[
\text{and r\_name = [region] and o\_orderdate \geq \text{DATE [startdate]} and o\_orderdate < \text{DATE [enddate]}}
\]

Query Execution Time

This experiment demonstrates that it is possible to use join synopses to obtain extremely fast approximate answers with minimal loss in accuracy.
Even for extremely small sizes, the join synopsis is able to track the actual aggregate value quite closely despite significant changes in the data distribution.

Shows that maintenance of join synopses is very inexpensive.
Something Missing in this paper

- Accurately approximating answers to group-by, rank and set valued queries.
- The formula for developing Space allocation was not complete in the paper.
- This paper relates only to part of aggregate queries and it's not specified, why and how the problem with other types of queries can be solved.
Hellerstein proposed a framework for approximate answers of aggregation queries called *online aggregation*. The base data is scanned in random order at query time and the approximate answer is continuously updated as the scan proceeds.

- Fully accurate answer
- It is not affected by database updates
- This work involves accessing original data at query time, thus being more costly.
- Here a large fraction of the data needs to be processed before the errors become tolerable
Conclusion

- We focused on important problem of computing approximate answers to aggregates computed on multi-way joins especially foreign key join.
- We have shown that schemes based on join synopses provide better performance than schemes based on base samples for computing approximate join aggregates.
- Join synopses can be maintained efficiently during updates to the underlying data.
Join Synopses for Approximate Query Answering

Questions?

Thank you