To appear in *Proc. Ninth Intl. Conf. Scientific and Statistical Database Management*

Large-Sample and Deterministic Confidence Intervals for Online Aggregation

Peter J. Haas
IBM Almaden Research Center
San Jose, CA 95120-6099
peterh@almaden.ibm.com

Abstract

The online aggregation system recently proposed by Hellerstein, et al. permits interactive exploration of large, complex datasets stored in relational database management systems. Running confidence intervals are an important component of an online aggregation system and indicate to the user the estimated proximity of each running aggregate to the corresponding final result. Large-sample confidence intervals contain the final result with a prespecified probability and rest on central limit theorems, while deterministic confidence intervals contain the final query result with probability 1. In this paper we show how new and existing central limit theorems, simple bounding arguments, and the delta method can be used to derive formulas for both large-sample and deterministic confidence intervals. To illustrate these techniques, we obtain formulas for running confidence intervals in the case of single-table and multi-table `AVG`*,* `COUNT`*,* `SUM`*,* `VARIANCE`*, and* `STDEV` *queries with join and selection predicates. Duplicate-elimination and* `GROUP-BY` *operations are also considered. We then provide numerically stable algorithms for computing the confidence intervals and analyze the complexity of these algorithms.*

1 Introduction

In a relational database management system (DBMS), "aggregation queries" are used to obtain statistical summaries of large, complex datasets. An aggregation query in a traditional DBMS is processed in batch mode: the query is submitted, the system processes a large volume of data over a long period of time, and, eventually, the final answer is returned. Although the usual purpose of an aggregation query is to get a rough picture of the data, the final answer is computed with painstaking precision, even in situations where an acceptably precise approximation might be available very quickly. Thus, the batch-mode approach often results in annoying, unnecessary delays for the user.

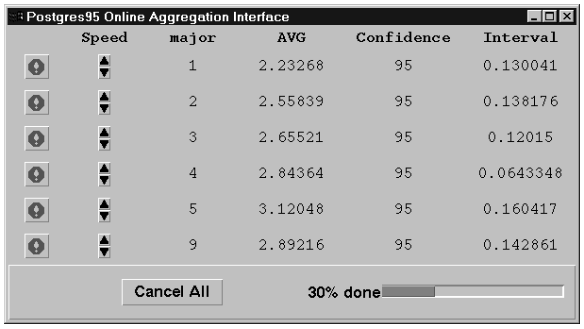

Figure 1: An online aggregation interface.

In order to address this problem, Hellerstein, et al. [6, 7] have proposed a system that supports *online* aggregation processing. In such a system, users can both observe the progress of their aggregation queries and control execution of these queries on the fly. Consider, for example, a single-table query of the form

```
SELECT AVG(final_grades) FROM grades
GROUP BY major;
```
(1.1)

A prototype of an online aggregation interface for this query is displayed in Figure 1. There is a row corresponding to each student group, that is, to each distinct value of `major` that appears in the table. The user does not have to specify the number of groups in advance– the groups are automatically detected by the system. For each group, the running average of all of the grades found so far (after 30% of the records in the table have been scanned) is displayed in the column entitled `AVG`. The "stop-sign" buttons can be used to abort the query for one or more groups, while allowing query processing to continue for the remaining groups. The arrow buttons in the column entitled `Speed` permit control of the relative rates at which the running averages for the different groups are updated.

As can be seen from the figure, an online aggregation interface is a more flexible and satisfying mechanism for data exploration than traditional batch-mode query processing. As indicated above, the user can modify a query on the fly using the stop-sign buttons. In addition, by controlling the relative speed at which the running aggregates are updated, the user can ensure that the most interesting information is obtained at the fastest rate. Finally, research on user interfaces indicates that the type of continuous feedback provided by an online aggregation interface is much more satisfying to a user than the traditional wristwatch icon or blinking cursor.

In an online aggregation system, the tuples of each input relation are retrieved in random order. This can be done, for example, by randomly permuting all or part of the data prior to querying, or in some cases simply by using Heap File access methods (when the attributes of interest are statistically independent of the location of tuples in the heap file). Since tuples are retrieved in random order, the tuples retrieved from a relation can be viewed as a simple random sample from the relation and the running aggregate can be viewed as a statistical estimator of the final query result. The estimated proximity of the running aggregate to the final result can therefore be indicated by means of an associated *running confidence interval*. Suppose that n records have been retrieved in random order and a running aggregate Y_n has been computed. For a prespecified *confidence parameter* $p \in (0,1)$, the idea is to display a *precision parameter* ε_n (also called a *confidence interval half-width*) such that Y_n is within $\pm\varepsilon_n$ of the final answer μ with probability approximately equal to p. Equivalently, the random interval $[Y_n - \varepsilon_n, Y_n + \varepsilon_n]$ contains μ with probability approximately equal to p. A large value of ε_n serves to warn the user that the records seen so far may not be sufficiently representative of the entire database, and hence the current value of the running aggregate may be far from the final result. The user can terminate processing of the aggregation query when ε_n decreases to a desired level; that is, a running confidence interval can help the user decide when to click on the stop-sign buttons. The running confidence intervals for the query in (1.1) are specified by the values in the rightmost two columns in Figure 1. For example, the running average of 2.23268 for the first group in Figure 1 is within ± 0.130041 of the final average with probability 95%.

An online aggregation system has an important advantage over existing DBMS's that support sampling: the online aggregation system does not require the user to specify stopping rules in advance. In existing systems, sampling typically stops when either a time limit elapses or the precision of the estimate equals a prespecified constant. Both types of stopping condition are unnecessarily rigid. The latter type of stopping condition, besides being hard to understand for casual users, is not even applicable to `GROUP-BY` queries as in Figure 1 since the number of groups is unknown *a priori*.

This paper is concerned with the problem of obtaining formulas for running confidence intervals and of computing these formulas efficiently. There are several types of running confidence intervals that can be constructed from n retrieved records:

(i) *Conservative* confidence intervals contain the final answer μ with a probability that is guaranteed to be greater than or equal to p. Such intervals can be based on extensions of Hoeffding's inequality [8] and are valid for all $n \geq 1$; see Haas [4].

(ii) *Large-sample* confidence intervals contain the final answer μ with a probability approximately equal to p and are based upon central limit theorems (CLT's). Such intervals are appropriate when n is small enough so that the records retrieved so far can be treated as a sample drawn with replacement but large enough so that approximations based on CLT's are accurate. Although large-sample confidence intervals must be used judiciously, the advantage of these intervals is that, when applicable, they are typically much shorter than conservative confidence intervals.

(iii) *Deterministic* confidence intervals contain μ with probability 1. Such intervals are typically useful when most of the records have been retrieved and n is very large. Unlike the other types of confidence interval, a deterministic confidence interval is typically of the form $[Y_n - \underline{\varepsilon}_n, Y_n + \overline{\varepsilon}_n]$ with $\underline{\varepsilon}_n \neq \overline{\varepsilon}_n$.

In an online aggregation system, it may be desirable to dynamically adjust the type of running confidence interval that is displayed based on the current value of n.

In Sections 2 and 3, we show how new and existing central limit theorems, simple bounding arguments, and the "delta method" can be used to obtain formulas for both large-sample and deterministic confidence intervals. To illustrate these techniques, we derive confidence-interval formulas for a variety of aggregation queries encountered in practice. For simple cases in which aggregates are computed for all of the tuples in a specified input relation, there is a well-known confidence-interval methodology; we present it here mainly for expository purposes and for completeness. Our primary focus is on more complicated cases in which aggregates are computed for the tuples in an output relation that is specified in terms of one or more input relations using selection, join, and projection operators. In this latter setting, methods for deriving confidence-interval formulas have not appeared previously. In Section 4 we provide numerically stable algorithms for computing the running confidence intervals discussed in Sections 2 and 3 and also analyze the complexity of these algorithms.

It is important to note that our results are applicable not only to online aggregation systems, but to any situation in which the answer to an aggregation query in a relational database is estimated based on simple random samples drawn from the input relations. As databases increase in size, such sampling-based estimation methods become an attractive alternative to a lengthy, exact computation.

2 Single-Table Queries

Consider a relation R containing m tuples, denoted $t_1, t_2, \dots, t_m$. In this section we consider queries of the form

```
SELECT op(expression)
FROM R
WHERE predicate;
```
(2.1)

where *op* is one of `COUNT`, `SUM`, `AVG`, `VARIANCE`, or `STDEV` (standard deviation), *expression* is an arithmetic expression involving the attributes of R, and *predicate* is an arbitrary predicate involving the attributes of R. A typical instance of such a query might look like

```
SELECT SUM(particulate_emissions *
           percent_fine_particulates)
FROM cities_emission_data
WHERE country = 'USA';
```

Unless stated otherwise, we assume throughout that *expression* is always equal to the SQL "*" construct whenever *op* is equal to `COUNT`; this means that the value of *expression* is taken as 1 for each tuple in R, so that the `COUNT` query simply computes the number of tuples that satisfy *predicate*. Another type of `COUNT` query that arises in practice has *expression* equal to *attribute*, where *attribute* is a fixed attribute of R; only tuples with non-`NULL` values of *attribute* contribute to the final count. Such a query can be transformed to a `COUNT(*)` query: replace *attribute* by * and adjoin the clause "*attribute* `IS NOT NULL`" to *predicate*. Finally, the `COUNT` operator is sometimes used in conjunction with the `DISTINCT` keyword; we discuss this situation in Section 2.4 below.

In the following subsections, we consider various special cases of the query in (2.1). Throughout, we denote by L_i the random index of the ith tuple retrieved from R; that is, the ith tuple retrieved from R is tuple t_{L_i}. We assume that all retrieval orders are equally likely, so that

$$P\{L_1 = i_1, L_2 = i_2, \dots, L_n = i_n\} = \begin{cases} (m-n)!/m! & \text{if } i_k \neq i_l \text{ for all } k \neq l, \\ 0 & \text{otherwise} \end{cases}$$

for $1 \le n \le m$ and $(i_1, i_2, \dots, i_n) \in \{1, 2, \dots, m\}^n$. When deriving large-sample confidence intervals, we assume as an approximation that $\{L_i \colon i \ge 0\}$ is a sequence of independent and identically distributed (i.i.d.) random variables with

$$P\{L_1 = i_1, \dots, L_n = i_n\} = \left(\frac{1}{m}\right)^n. \tag{2.2}$$

2.1 AVG and SUM Queries: No Selection Predicate

Suppose in (2.1) that *predicate* is empty and *op* is equal to `AVG` or `SUM`. Set $v(i)$ equal to ξ times the value of *expression* when applied to tuple t_i, where $\xi = 1$ if *op* is equal to `AVG` and $\xi = m$ if *op* is equal to `SUM`. Denote by $\mathcal{F}$ the set of real-valued functions defined on $\{1, 2, \dots, m\}$, and for $f \in \mathcal{F}$ set

$$\theta(f) = \frac{1}{m} \sum_{i=1}^{m} f(i). \tag{2.3}$$

With these definitions, the final answer μ to the query in (2.1) can be written as $\mu = \theta(v)$. After n tuples have been retrieved, the running aggregate is given by

$$\overline{Y}_n = T_n(v), \tag{2.4}$$

where

$$T_n(f) = \frac{1}{n} \sum_{i=1}^{n} f(L_i) \tag{2.5}$$

for $f \in \mathcal{F}$.

Under the large-sample assumption in (2.2), techniques for obtaining a large-sample confidence interval for $\overline{Y}_n$ are well known. Indeed, by the standard CLT for i.i.d. random variables, $\sqrt{n}(\overline{Y}_n - \mu)/\theta_2^{1/2}(v) \Rightarrow N(0,1)$ as $n \to \infty$, where

$$\theta_q(f) = \frac{1}{m} \sum_{i=1}^{m} \big(f(i) - \theta(f)\big)^q \tag{2.6}$$

for $q \ge 0$ and $f \in \mathcal{F}$, the symbol "$\Rightarrow$" denotes convergence in distribution, and $N(0,1)$ denotes a standardized (mean 0, variance 1) normal random variable. It follows from the strong law of large numbers for i.i.d. random variables that $T_{n,2}(v) \to \theta_2(v)$ almost surely (a.s.), where

$$T_{n,q}(f) = \frac{1}{n-1} \sum_{i=1}^{n} \big(f(L_i) - T_n(f)\big)^q \tag{2.7}$$

for $q \ge 0$ and $f \in \mathcal{F}$. A standard application of Slutsky's Theorem (cf Serfling [10, Section 1.5.4]) shows that

$$\frac{\sqrt{n}(\overline{Y}_n - \mu)}{T_{n,2}^{1/2}(v)} \Rightarrow N(0,1).$$

Thus, for large n, we have

$$\begin{aligned} P\{|\overline{Y}_n - \mu| \le \varepsilon\} &= P\left\{ \left| \frac{\sqrt{n}(\overline{Y}_n - \mu)}{T_{n,2}^{1/2}(v)} \right| \le \frac{\varepsilon\sqrt{n}}{T_{n,2}^{1/2}(v)} \right\} \\ &\approx 2\Phi\left(\frac{\varepsilon\sqrt{n}}{T_{n,2}^{1/2}(v)} \right) - 1, \end{aligned} \tag{2.8}$$

for $\varepsilon > 0$, where Φ is the cumulative distribution function of an $N(0,1)$ random variable. Denote by z_p the $(p+1)/2$ quantile of this distribution function, so that $\Phi(z_p) = (p+1)/2$. Then, setting the rightmost term in (2.8) equal to p and solving for ε, we see that a large-sample confidence interval is obtained by choosing

$$\varepsilon_n = \left(\frac{z_p^2 T_{n,2}(v)}{n}\right)^{1/2}. \tag{2.9}$$

This basic approach for deriving large-sample confidence intervals applies to all of the queries that we consider; only the variance constant in the CLT differs from query to query.

We now derive a deterministic confidence interval. The large-sample approximation in (2.2) is not needed for this purpose; instead, we suppose that there exist constants a and b, known *a priori*, such that $a \leq v(i) \leq b$ for $1 \leq i \leq m$. Such constants frequently can be obtained from the database system catalog, where the maximum and minimum values of various attributes are often stored for use by the DBMS query optimizer. Since $m\mu \leq n\overline{Y}_n + (m-n)b$, it follows that $\mu \leq \overline{Y}_n + \overline{\varepsilon}_n$, where

$$\overline{\varepsilon}_n = \frac{(m-n)(b-\overline{Y}_n)}{m}. \tag{2.10}$$

A similar argument shows that $\mu \geq \overline{Y}_n - \underline{\varepsilon}_n$, where

$$\underline{\varepsilon}_n = \frac{(m-n)(\overline{Y}_n - a)}{m} \tag{2.11}$$

2.2 VARIANCE and STDEV Queries: No Selection Predicates

Now suppose in (2.1) that *predicate* is empty and *op* is equal to `VARIANCE` or `STDEV`. Set $v(i)$ equal to the value of *expression* when applied to tuple t_i. If *op* equals `VARIANCE`, then the final answer σ^2 is given by $\sigma^2 = \theta_2(v)$, where $\theta_2(f)$ is defined by (2.6) for $f \in \mathcal{F}$. After n tuples have been retrieved, the corresponding running aggregate is given by

$$Z_n = \left(\frac{m-1}{m}\right) T_{n,2}(v) \tag{2.12}$$

where $T_{n,2}(f)$ is defined by (2.7). We choose this particular formula for the running aggregate because Z_n is unbiased for σ^2. If *op* is equal to `STDEV`, the final answer is equal to σ and the running aggregate is given by $\sqrt{Z_n}$.

In order to derive large-sample confidence intervals, we use the approximation in (2.2) and consider the modified estimator

$$Z'_n = \left(\frac{n-1}{n}\right) T_{n,2}(v).$$

When the large-sample approximation applies, m and n are large, so that $Z'_n \approx Z_n$. Following [10, Section 2.2], we establish CLT's for Z'_n and $\sqrt{Z'_n}$ using "delta-method" arguments based on Lemma 1 below; see [10, Section 2.2] for a general discussion of sample moments. For fixed $k \geq 1$, let $\{(X_{n,1},\ldots,X_{n,k})\colon n \geq 1\}$ and $(X_1,\ldots,X_k)$ be k-dimensional random vectors and $(c_1,\ldots,c_k)$ be a k-dimensional vector of constants.

Lemma 1 *Suppose that* $\sqrt{n}(X_{n,1}-c_1,\ldots,X_{n,k}-c_k) \Rightarrow (X_1,X_2,\ldots,X_k)$ *as* $n \to \infty$. *Let* $h\colon \Re^k \mapsto \Re$ *be a function differentiable at the point* $(c_1,c_2,\ldots,c_k)$ *with partial derivatives* $h_1,h_2,\ldots,h_k$. *Then*

$$\sqrt{n}\big(h(X_{n,1},X_{n,2},\ldots,X_{n,k}) - h(c_1,c_2,\ldots,c_k)\big) \Rightarrow \sum_{l=1}^{k} h_l(c_1,c_2,\ldots,c_k)X_l$$

as $n \to \infty$.

See Billingsley [1, p. 402] for a proof of this result. To obtain a CLT for Z'_n, fix $k=2$. Set $X_{n,1} = T_n(v^2)$ and $X_{n,2} = T_n(v)$ for $n \geq 1$, where $T_n(f)$ is defined in (2.5) and $v^2(i) = \big(v(i)\big)^2$ for $1 \leq i \leq m$. Since $\{\big(v^2(L_i), v(L_i)\big)\colon i \geq 1\}$ is a sequence of i.i.d. random vectors under the assumption in (2.2), it follows from the standard multivariate CLT [1, Theorem 29.5] that $\sqrt{n}\big(X_{n,1} - \theta(v^2), X_{n,2} - \theta(v)\big) \Rightarrow (X_1,X_2)$, where (X_1,X_2) is a bivariate normal random vector with mean $(0,0)$ and covariance matrix

$$\Sigma = \begin{pmatrix} \theta(v^4) - \theta^2(v^2) & \theta(v^3) - \theta(v^2)\theta(v) \\ \theta(v^3) - \theta(v^2)\theta(v) & \theta(v^2) - \theta^2(v) \end{pmatrix}.$$

Taking $h(x_1,x_2) = x_1 - x_2^2$ in Lemma 1, we find that

$$\sqrt{n}(Z'_n - \sigma^2) \Rightarrow X_1 - 2\theta(v)X_2$$

as $n \to \infty$. By standard properties of the bivariate normal distribution (see [1, p. 398]), the random variable $X_1 - 2\theta(v)X_2$ has a normal distribution with mean 0 and variance

$$\gamma = \big(1, -2\theta(v)\big)^t \Sigma \big(1, -2\theta(v)\big) = \theta_4(v) - \theta_2^2(v).$$

(Here x^t denotes the transpose of the vector x; all vectors are assumed to be column vectors.) It follows from the strong law of large numbers for i.i.d. random variables that $T_{n,4}(v) - T_{n,2}^2(v) \to \gamma$ a.s. as $n \to \infty$. Arguing as in Section 2.1, we see that a large-sample confidence interval for Z'_n, and hence for Z_n, can be obtained by setting

$$\varepsilon_n = \left(\frac{z_p^2\big(T_{n,4}(v) - T_{n,2}^2(v)\big)}{n}\right)^{1/2}.$$

To obtain a large-sample confidence interval for a `STDEV` query, we repeat the above argument, this time using the function $h(x_1,x_2) = (x_1 - x_2^2)^{1/2}$ in Lemma 1. The resulting formula is

$$\varepsilon_n = \left(\frac{z_p^2\big(T_{n,4}(v) - T_{n,2}^2(v)\big)}{4nT_{n,2}(v)}\right)^{1/2}.$$

We now derive a deterministic confidence interval. As before, we assume the existence of *priori bounds*: $a \leq v(i) \leq b$

for $1 \le i \le m$. Set $D_n = (m-n)^{-1}\sum_{i=n+1}^m v(L_i)$ and $Q_n = \sum_{i=n+1}^m \big(v(L_i) - D_n\big)^2$. By Equation (1.5b) in Chan, Golub, and LeVeque [2], we have

$$\sigma^2 = \frac{n-1}{m-1}Z_n + \Big(1-\frac{n}{m}\Big)\bigg(\frac{1}{m-n}Q_n + \frac{n}{m}(\overline{Y}_n - D_n)^2\bigg),$$

where $\overline{Y}_n$ and Z_n are defined in (2.4) and (2.12), respectively. Using a well-known upper bound on the variance (see [4, Lemma 3] for a proof), we find that

$$Q_n \le (m-n)\frac{(b-a)^2}{4}. \tag{2.13}$$

Since

$$(\overline{Y}_n - D_n)^2 \le \Big((\overline{Y}_n - a)\vee(b-\overline{Y}_n)\Big)^2, \tag{2.14}$$

(where $x \vee y$ denotes the maximum of x and y) we have

$$\sigma^2 \le \frac{n-1}{m-1}Z_n + \lambda_n, \tag{2.15}$$

where

$$\lambda_n = \Big(1-\frac{n}{m}\Big)\bigg(\frac{(b-a)^2}{4} + \frac{n}{m}\Big((\overline{Y}_n - a)\vee(b-\overline{Y}_n)\Big)^2\bigg).$$

Equivalently, $\sigma^2 \le Z_n + \bar{\varepsilon}_n$, where

$$\bar{\varepsilon}_n = \lambda_n - \frac{m-n}{m-1}Z_n.$$

Using the trivial bounds $Q_n \ge 0$ and $(\overline{Y}_n - D_n)^2 \ge 0$, a similar argument shows that $\sigma^2 \ge Z_n - \underline{\varepsilon}_n$, where

$$\underline{\varepsilon}_n = \frac{m-n}{m-1}Z_n.$$

To obtain a deterministic confidence interval for `STDEV` queries, observe that, by (2.15),

$$\sigma \le \bigg(\frac{n-1}{m-1}Z_n + \lambda_n\bigg)^{1/2},$$

so that $\sigma \le \sqrt{Z_n} + \bar{\varepsilon}_n$, where

$$\bar{\varepsilon}_n = \bigg(\frac{n-1}{m-1}Z_n + \lambda_n\bigg)^{1/2} - \sqrt{Z_n}.$$

Similarly, we find that $\sigma \ge \sqrt{Z_n} - \underline{\varepsilon}_n$, where

$$\underline{\varepsilon}_n = \sqrt{Z_n} - \bigg(\frac{n-1}{m-1}Z_n\bigg)^{1/2}.$$

2.3 Queries with Selection Predicates

Suppose in (2.1) that *predicate* is nonempty and *op* is one of `COUNT`, `SUM`, `AVG`, `VARIANCE`, or `STDEV`. If *op* is equal to `COUNT`, then we can apply the formulas in (2.9)–(2.11) with $v(i) = m$ if tuple t_i satisfies *predicate* and $v(i) = 0$ otherwise. Similarly, when *op* is equal to `SUM`, we can apply formulas (2.9)–(2.11) with $v(i)$ equal to m times the value of *expression* when applied to tuple t_i if t_i satisfies *predicate* and $v(i)$ equal to 0 otherwise.

To handle the case in which *op* is equal to `AVG`, `VARIANCE`, or `STDEV`, set $v(i)$ equal to the value of *expression* when applied to tuple t_i (whether or not t_i satisfies *predicate*). Denote by S ($\subseteq R$) the set of tuples that satisfy *predicate*. Set $u(i) = 1$ if $t_i \in S$ and $u(i) = 0$ otherwise. The final answer to the `AVG`, `VARIANCE`, or `STDEV` query can be written as

$$\mu(S) = \frac{1}{|S|}\sum_{t_i \in S} v(i) = \frac{\theta(uv)}{\theta(u)}$$

$$\sigma^2(S) = \frac{1}{|S|}\sum_{t_i\in S}\big(v(i)-\mu(S)\big)^2 = \frac{\theta(uv^2)\theta(u) - \theta^2(uv)}{\theta^2(u)},$$

or $\sigma(S)$, respectively. (Here we define the function $u^q v^r \in \mathcal{F}$ for $q, r \ge 0$ by setting $(u^q v^r)(i) = \big(u(i)\big)^q\big(v(i)\big)^r$ for $1 \le i \le m$. We also denote by $|A|$ the number of elements in the set A.) After n tuples have been retrieved, the running aggregate for an `AVG`, `VARIANCE`, or `STDEV` query is given by

$$\overline{Y}_n(S) = \frac{T_n(uv)}{T_n(u)}, \tag{2.16}$$

$$Z_n(S) = \frac{T_n(uv^2)T_n(u) - T_n^2(uv)}{T_n^2(u)},$$

or $\sqrt{Z_n(S)}$, respectively.

As in previous sections, delta-method arguments can be used to establish large-sample confidence intervals. For example, consider the running aggregate $Z_n(S)$. To develop a CLT for $Z_n(S)$, set $X_{n,1} = T_n(uv^2)$, $X_{n,2} = T_n(uv)$, and $X_{n,3} = T_n(u)$. Also set

$$\theta_{q,r}(f,g) = \frac{1}{m}\sum_{i=1}^m \big(f(i) - \theta(f)\big)^q\big(g(i)-\theta(g)\big)^r$$

for $q, r \ge 0$ and $f, g \in \mathcal{F}$. By the standard multivariate CLT,

$$\sqrt{n}\big(X_{n,1} - \theta(uv^2), X_{n,2} - \theta(uv), X_{n,3} - \theta(u)\big) \Rightarrow (X_1, X_2, X_3),$$

where (X_1, X_2, X_3) is a multivariate normal random vector with mean $(0,0,0)$ and covariance matrix

$$\Sigma = \begin{pmatrix} \theta_2(uv^2) & \theta_{1,1}(uv^2, uv) & \theta_{1,1}(uv^2, u) \\ \theta_{1,1}(uv^2, uv) & \theta_2(uv) & \theta_{1,1}(uv, u) \\ \theta_{1,1}(uv^2, u) & \theta_{1,1}(uv, u) & \theta_2(u) \end{pmatrix}.$$

Taking $h(x_1, x_2, x_3) = (x_1 x_3 - x_2^2)/x_3^2$ in Lemma 1, we find that

$$\sqrt{n}\big(Z_n(S) - \sigma^2(S)\big) \Rightarrow N\big(0, \gamma/\theta^2(u)\big),$$

where

$$\begin{aligned}\gamma &= \theta_2(uv^2) - 4\rho_2\theta_{1,1}(uv^2, uv) \\ &\quad + (4\rho_2^2 - 2\rho_1)\theta_{1,1}(uv^2, u) + 4\rho_2^2\theta_2(uv) \\ &\quad + (4\rho_1\rho_2 - 8\rho_2^3)\theta_{1,1}(uv, u) + (2\rho_2^2 - \rho_1)^2\theta_2(u),\end{aligned}$$

$\rho_1 = \theta(uv^2)/\theta(u)$, and $\rho_2 = \theta(uv)/\theta(u)$. Next, set

$$\begin{aligned}G_n' &= T_{n,2}(uv^2) - 4R_{n,2}T_{n,1,1}(uv^2,uv)\\&\quad + (4R_{n,2}^2 - 2R_{n,1})T_{n,1,1}(uv^2,u) + 4R_{n,2}^2T_{n,2}(uv)\\&\quad + (4R_{n,1}R_{n,2} - 8R_{n,2}^3)T_{n,1,1}(uv,u)\\&\quad + (2R_{n,2}^2 - R_{n,1})^2T_{n,2}(u),\end{aligned}$$

where $T_n(f)$ and $T_{n,q}(f)$ are defined as in (2.5) and (2.7), respectively,

$$T_{n,q,r}(f,g) = \frac{1}{n-1}\sum_{i=1}^{n}\big(f(L_i)-T_n(f)\big)^q\big(g(L_i)-T_n(g)\big)^r$$

for $q,r \ge 0$ and $f,g \in \mathcal{F}$, $R_{n,1} = T_n(uv^2)/T_n(u)$, and $R_{n,2} = T_n(uv)/T_n(u)$. We then have $G_n'/T_n^2(u) \to \gamma'/\theta^2(u)$ a.s., and we obtain the formula

$$\varepsilon_n = \left(\frac{z_p^2 G_n'}{nT_n^2(u)}\right)^{1/2}.$$

Similar arguments lead to the formulas

$$\varepsilon_n = \left(\frac{z_p^2 G_n}{nT_n^2(u)}\right)^{1/2} \tag{2.17}$$

and

$$\varepsilon_n = \left(\frac{z_p^2 G_n'}{4nZ_n(S)T_n^2(u)}\right)^{1/2}$$

for `AVG` and `STDEV` queries, respectively, where

$$G_n = T_{n,2}(uv) - 2R_{n,2}T_{n,1,1}(uv,u) + R_{n,2}^2T_{n,2}(u).$$

Deterministic confidence intervals are derived as follows. Set $I_n = \sum_{i=1}^n u(L_i)$ and $B_n = \sum_{i=1}^m u(i) - I_n$ for $n \ge 1$. As usual, we assume the existence of *priori bounds*: $a \le v(i) \le b$ for $1 \le i \le m$. Observe that

$$(I_n + B_n)\mu(S) \le I_n\overline{Y}_n(S) + B_n b,$$

and, since $0 \le B_n \le m-n$,

$$\begin{aligned}\mu(S) &\le \frac{I_n\overline{Y}_n(S)}{I_n+B_n} + \frac{B_n b}{I_n+B_n}\\&\le (\overline{Y}_n(S)\vee 0) + \frac{I_n(\overline{Y}_n(S)\wedge 0)}{I_n+m-n} + \frac{B_n(b\vee 0)}{I_n+B_n}\end{aligned}$$

where $x \wedge y$ denotes the minimum of x and y. For $c > 0$ the function $f(x) = x/(c+x)$ is nondecreasing on $[0,\infty)$, so that

$$\frac{B_n}{I_n+B_n} \le \frac{m-n}{I_n+m-n}. \tag{2.18}$$

Thus, $\mu(S) \le \overline{Y}_n(S) + \overline{\varepsilon}_n$, where

$$\overline{\varepsilon}_n = \frac{(m-n)\Big(\big(b\vee 0\big) - \big(\overline{Y}_n(S)\wedge 0\big)\Big)}{I_n+m-n}.$$

Similar arguments show that $\mu(S) \ge \overline{Y}_n(S) - \underline{\varepsilon}_n$, where

$$\underline{\varepsilon}_n = \frac{(m-n)\Big(\big(\overline{Y}_n(S)\vee 0\big) - \big(a\wedge 0\big)\Big)}{I_n+m-n}.$$

The derivation of $\overline{\varepsilon}_n$ and $\underline{\varepsilon}_n$ for `VARIANCE` queries (see Appendix A) is similar to the derivation in Section 2.2 for `VARIANCE` queries with no selection predicate and yields the formulas

$$\overline{\varepsilon}_n = \lambda_n' - \frac{1}{I_n}Z_n(S) \tag{2.19}$$

and

$$\underline{\varepsilon}_n = \frac{m-n+1}{m-n+I_n}Z_n(S), \tag{2.20}$$

where

$$\begin{aligned}\lambda_n' &= \frac{m-n}{I_n+m-n}\frac{(b-a)^2}{4}\\&\quad + \frac{I_n\big((m-n)\wedge I_n\big)}{\Big(I_n + \big((m-n)\wedge I_n\big)\Big)^2}\\&\quad \times \Big(\big(\overline{Y}_n(S)-a\big)\vee\big(b-\overline{Y}_n(S)\big)\Big)^2.\end{aligned}$$

As in Section 2.2, we obtain corresponding formulas for `STDEV` queries:

$$\overline{\varepsilon}_n = \left(\frac{I_n-1}{I_n}Z_n(S) + \lambda_n'\right)^{1/2} - \sqrt{Z_n(S)}$$

and

$$\underline{\varepsilon}_n = \sqrt{Z_n(S)} - \left(\frac{I_n-1}{m-n+I_n}Z_n(S)\right)^{1/2}.$$

2.4 Duplicate-Elimination and GROUP-BY

The results in the previous subsections apply almost unchanged to queries of the form

```
SELECT op(DISTINCT expression)
FROM R
WHERE predicate;
```

where *op* is one of `COUNT`, `SUM`, `AVG`, `VARIANCE`, or `STDEV`. The `DISTINCT` keyword means that duplicate values of *expression* are eliminated before applying *op*. This keyword is usually used in conjunction with the `COUNT` operator, in which case *expression* is usually equal to a fixed attribute of R. To handle queries containing the `DISTINCT` keyword, define the set S as in Section 2.3. Set $U_i' = 1$ if $t_{L_i} \in S$ and $v(L_i) \ne v(L_j)$ for $1 \le j \le i-1$; otherwise, set $U_i' = 0$. The formulas in Section 2.3 then hold with $u(L_i)$ replaced by U_i', $uv(L_i)$ replaced by $U_i'v(L_i)$, and so forth. (When *predicate* is empty, take $S = R$ and apply the formulas in Section 2.3, modified as above.) This approach in effect bars a tuple t_i from entering the sample if there already exists a tuple t_j in the sample with $v(j) = v(i)$. As a result, the sample is statistically identical to a simple random sample from a "reduced" relation that is obtained from R by eliminating duplicates; cf Cochran [3, Section 2.12].

The results in the previous subsections also apply to queries of the form

```
SELECT op(expression)
FROM R
WHERE predicate
GROUP BY a_1, a_2, ..., a_J;
```

where $J \geq 1$, $a_1, a_2, \ldots, a_J$ are attributes of R, and *op* is one of `COUNT`, `SUM`, `AVG`, `VARIANCE`, or `STDEV`. The `GROUP BY` clause means that tuples of the result relation are divided into groups and *op* is computed separately for each group; tuples t_1 and t_2 belong to the same group if $t_1.a_j = t_2.a_j$ for $1 \leq j \leq J$. When the query contains a `GROUP BY` clause, the formulas in Section 2.3 can be applied to the running aggregate for each group. For the group with $a_j = x_j$ for $1 \leq j \leq J$ (where $x_1, x_2, \ldots, x_J$ are fixed), use the formulas in Section 2.3 with $u(i) = 1$ if tuple t_i satisfies *predicate* and $t_i.a_j = x_j$ for $1 \leq j \leq J$. It is also possible to obtain simultaneous confidence intervals for all of the running aggregates. For example, suppose that there are G (> 1) groups and the confidence-interval formulas are applied with confidence parameter $p' = 1 - (1-p)/G$. Then, by Bonferroni's inequality (See Miller [9, p. 8]), the probability that all of the G confidence intervals simultaneously contain their respective final answers is $\geq p$ for sufficiently large sample sizes. This approach to obtaining simultaneous confidence intervals is appropriate when the number of groups is not too large; more sophisticated approaches are available [9], but are beyond the scope of the current paper.

The above techniques can of course be combined to handle queries containing both the `DISTINCT` keyword and a `GROUP BY` clause. These techniques also generalize naturally to multi-table queries of the type discussed in Section 3 below.

2.5 Page-Level I/O

The above discussion has implicitly assumed that at each "retrieval step" exactly one tuple is retrieved from the input relation. Under a suitable reinterpretation, the confidence-interval formulas in the above subsections can be applied when a page of tuples is obtained at each retrieval step. Suppose that each page contains $B > 0$ tuples and consider, for example, a simple `AVG` query as in Section 2.1 in which v_0 is the function being averaged over all of the tuples in the input relation. The final answer to the aggregation query can be written as $\mu = (1/m)\sum_{i=1}^{m} v(i)$, where m is now interpreted as the number of pages in the relation and for $1 \leq i \leq m$ the function $v(i)$ is interpreted as the average value of v_0 over all of the tuples on page i. Similarly, the running aggregate $\overline{Y}_n$ (based on n retrieval steps) is given by (2.4) with v interpreted as above. Thus, the formulas for the large-sample and deterministic confidence intervals in (2.9)–(2.11) hold, provided that v is interpreted as above, n is interpreted as the number of retrieval steps, and pages are returned in random order. (No assumptions about the assignment of tuples to pages are needed. As n increases, however, the half-width of the large-sample confidence interval decreases most rapidly when tuples are assigned to pages randomly.) The other confidence-interval formulas developed in Sections 2.1–2.3 also hold under similar reinterpretations.

3 Multi-Table Queries

Consider K (> 1) relations $R_1, R_2, \ldots, R_K$. For $1 \leq k \leq K$, denote the tuples in R_k by $t_{k,1}, t_{k,2}, \ldots, t_{k,m_k}$, where m_k is the number of tuples in R_k. In this section we consider queries of the form

```
SELECT op(expression)
FROM R_1, R_2, ..., R_K
WHERE predicate;
```
(3.1)

where *op* is one of `COUNT`, `SUM`, or `AVG`, *expression* is an arithmetic expression involving the attributes of $R_1, R_2, \ldots, R_K$, and *predicate* is an arbitrary predicate involving these attributes. As before, *expression* is empty if *op* is equal to `COUNT`. Usually, *predicate* is a conjunction of join and selection predicates. A typical instance of such a query might look like

```
SELECT SUM(emissions
            * calibration_factor)
FROM CO2_data, instrument
WHERE CO2_data.instrument_id
        = instrument.id
AND CO2_data.state = 'CA';
```
(3.2)

In the following subsections, we consider special cases of the query in (3.1). For simplicity, we assume that at each retrieval step exactly one tuple is retrieved from each of the K input relations. We also assume that, for each input relation R_k, tuples are retrieved in random order, independently of the retrieval order for the other relations. (In Section 3.4 we discuss other retrieval methods.) Denote by $L_{k,i}$ the random index of the ith tuple retrieved from relation R_k. When deriving large-sample confidence intervals, we assume as an approximation that, for each $1 \leq k \leq K$, $\{ L_{k,i} \colon\ i \geq 0 \}$ is a sequence of i.i.d. random variables with

$$P\{ L_{k,1} = i_1, \ldots, L_{k,n} = i_n \} = \left(\frac{1}{m_k} \right)^n. \tag{3.3}$$

3.1 SUM and COUNT Queries

We focus first on the case in which *op* is equal to `COUNT` or `SUM`. Set $v(i_1, i_2, \ldots, i_K)$ equal to $m_1 m_2 \cdots m_K$ times the value of *expression* when applied to tuples $t_{1,i_1}, t_{2,i_2}, \ldots, t_{K,i_K}$ if these tuples jointly satisfy the predicate; otherwise, set $v(i_1, i_2, \ldots, i_K) = 0$. (As before, we assume that if *op* is equal to `COUNT`, then the value of *expression* is always equal to 1.) Denote by $\tilde{\mathcal{F}}$ the set of real-valued functions defined on

$$\Lambda = \{1,2,\ldots,m_1\} \times \cdots \times \{1,2,\ldots,m_K\},$$

and for $f \in \tilde{\mathcal{F}}$ set

$$\tilde{\theta}(f) = \frac{1}{m_1 m_2 \cdots m_K} \sum_{i_1=1}^{m_1} \sum_{i_2=1}^{m_2} \cdots \sum_{i_K=1}^{m_K} f(i_1, i_2, \ldots, i_K). \quad (3.4)$$

With these definitions, the final answer $\tilde{\mu}$ to the query in (3.1) can be written as

$$\tilde{\mu} = \tilde{\theta}(v). \quad (3.5)$$

Suppose that n_k tuples have been retrieved from relation R_k for $1 \le k \le K$, where $1 \le n_k \le m_k$. Setting $\boldsymbol{n} = (n_1, n_2, \ldots, n_K)$, the running aggregate for a `COUNT` or `SUM` query is given by $\tilde{Y}_{\boldsymbol{n}} = \tilde{T}_{\boldsymbol{n}}(v)$, where

$$\tilde{T}_{\boldsymbol{n}}(f) = \frac{1}{n_1 n_2 \cdots n_K} \sum_{i_1=1}^{n_1} \sum_{i_2=1}^{n_2} \cdots \sum_{i_K=1}^{n_K} f(L_{1,i_1}, L_{2,i_2}, \ldots, L_{K,i_K}) \quad (3.6)$$

for $f \in \tilde{\mathcal{F}}$. To develop large-sample confidence intervals, we assume that n records have been retrieved from each input relation, so that the running aggregate is $\tilde{Y}_n = \tilde{T}_n(v)$, where

$$\tilde{T}_n(f) = \frac{1}{n^K} \sum_{i_1=1}^{n} \sum_{i_2=1}^{n} \cdots \sum_{i_K=1}^{n} f(L_{1,i_1}, L_{2,i_2}, \ldots, L_{K,i_K}). \quad (3.7)$$

for $f \in \tilde{\mathcal{F}}$. We also assume that (3.3) holds.

An appropriate expression for ε_n can be derived from Theorem 3.12 below. Set

$$\tilde{\theta}(f;k,j) = (m_1 \cdots m_{k-1} m_{k+1} \cdots m_K)^{-1} \times \sum_{i_1=1}^{m_1} \cdots \sum_{i_{k-1}=1}^{m_{k-1}} \sum_{i_{k+1}=1}^{m_{k+1}} \cdots \sum_{i_K=1}^{m_K} f(i_1, \ldots, i_{k-1}, j, i_{k+1}, \ldots, i_K) \quad (3.8)$$

and

$$\tilde{\theta}_q(f) = \sum_{k=1}^{K} \left(\frac{1}{m_k} \sum_{j=1}^{m_k} \Big(\tilde{\theta}(f;k,j) - \tilde{\theta}(f) \Big)^q \right) \quad (3.9)$$

for $q \ge 0$, $1 \le k \le K$, $1 \le j \le m_k$, and $f \in \tilde{\mathcal{F}}$. Also set

$$\tilde{T}_n(f;k,j) = \frac{1}{n^{K-1}} \sum_{i_1=1}^{n} \cdots \sum_{i_{k-1}=1}^{n} \sum_{i_{k+1}=1}^{n} \cdots \sum_{i_K=1}^{n} f(L_{1,i_1}, \ldots, L_{k-1,i_{k-1}}, L_{k,j}, L_{k+1,i_{k+1}}, \ldots, L_{K,i_K}) \quad (3.10)$$

and

$$\tilde{T}_{n,q}(f) = \sum_{k=1}^{K} \left(\frac{1}{(n-1)} \sum_{j=1}^{n} \Big(\tilde{T}_n(f;k,j) - \tilde{T}_n(f) \Big)^q \right). \quad (3.11)$$

Theorem 3.12 *Suppose that $0 < \tilde{\theta}_2(f) < \infty$. Then $\sqrt{n}\big(\tilde{T}_n(f) - \tilde{\theta}(f)\big) \Rightarrow X$ as $n \to \infty$, where X has a normal distribution with mean 0 and variance $\tilde{\theta}_2(f)$. Moreover, $\tilde{T}_n(f) \to \tilde{\theta}(f)$ and $\tilde{T}_{n,2}(f) \to \tilde{\theta}_2(f)$ a.s. as $n \to \infty$.*

The proof of this result is contained in the proof of Theorem 6 in Haas, Naughton, Seshadri, and Swami [5]. Taking $f = v$ in Theorem 3.12, an argument as in Section 2.1 yields the formula

$$\varepsilon_n = \left(\frac{z_p^2 \tilde{T}_{n,2}(v)}{n} \right)^{1/2}.$$

To obtain a deterministic confidence interval, we suppose as before that there exist *a priori* bounds: $a \le v(i_1, i_2, \ldots, i_K) \le b$ for $(i_1, i_2, \ldots, i_K) \in \Lambda$. Set

$$h(\boldsymbol{n}) = \prod_{k=1}^{K} m_k - \prod_{k=1}^{K} n_k$$

for $\boldsymbol{n} = (n_1, n_2, \ldots, n_K)$. An argument almost identical to that in Section 2.1 shows that $\tilde{Y}_{\boldsymbol{n}} - \underline{\varepsilon}_{\boldsymbol{n}} \le \tilde{\mu} \le \tilde{Y}_{\boldsymbol{n}} + \overline{\varepsilon}_{\boldsymbol{n}}$, where

$$\overline{\varepsilon}_{\boldsymbol{n}} = \frac{h(\boldsymbol{n})(b - \tilde{Y}_{\boldsymbol{n}})}{\prod_{k=1}^{K} m_k} \quad (3.13)$$

and

$$\underline{\varepsilon}_{\boldsymbol{n}} = \frac{h(\boldsymbol{n})(\tilde{Y}_{\boldsymbol{n}} - a)}{\prod_{k=1}^{K} m_k}. \quad (3.14)$$

The formulas in (3.13) and (3.14) reduce to the formulas in (2.10) and (2.11), respectively, when $K = 1$.

Using prior knowledge about the data, it is sometimes possible to tighten the bounds in (3.13) and (3.14). To this end, we introduce the following notation. For a subset $H = \{k(1), k(2), \ldots, k(r)\} \subseteq \{1,2,\ldots,K\}$ with $k(1) < k(2) < \cdots < k(r)$, set

$$\Lambda(H) = \Big\{ 1,2,\ldots,m_{k(1)} \Big\} \times \cdots \times \Big\{ 1,2,\ldots,m_{k(r)} \Big\}.$$

For an r-vector $q = (q_1, q_2, \ldots, q_r) \in \Lambda(H)$, denote by $\tilde{\theta}(v;H,q)$ the average of the function v over all vectors $(i_1, i_2, \ldots, i_K) \in \Lambda$ such that $i_{k(j)} = q_j$ for $1 \le j \le r$. When $K = 4$, $H = \{1,3\}$, and $q = (5,7)$, for example, we have

$$\tilde{\theta}(v;H,q) = \frac{1}{m_2 m_4} \sum_{i_2=1}^{m_2} \sum_{i_4=1}^{m_4} v(5, i_2, 7, i_4).$$

Next, set $J(\boldsymbol{n}) = \{k\colon n_k < m_k\}$. Finally, choose $a(\boldsymbol{n})$ and $b(\boldsymbol{n})$ such that

$$a(\boldsymbol{n}) \le \min_{q \in \Lambda(J(\boldsymbol{n}))} \tilde{\theta}\big(v; J(\boldsymbol{n}), q\big) \quad (3.15)$$

and

$$b(\boldsymbol{n}) \ge \max_{q \in \Lambda(J(\boldsymbol{n}))} \tilde{\theta}\big(v; J(\boldsymbol{n}), q\big). \quad (3.16)$$

Then (3.13) and (3.14) hold with a and b replaced by $a(\boldsymbol{n})$ and $b(\boldsymbol{n})$, respectively. In some commonly occurring situations it is possible to choose $a(\boldsymbol{n})$ and $b(\boldsymbol{n})$ such that $a(\boldsymbol{n}) > a$ and/or $b(\boldsymbol{n}) < b$, so that the length of resulting deterministic confidence interval is decreased.

Example 1 Consider the query in (3.2), and denote by m_1 and m_2 the number of tuples in the `instrument` and `CO2_data` relations, respectively. Suppose that, according to information in the system catalog, the calibration factor for each measuring instrument falls between 0 and p and the maximum recorded value of `emissions` is q units. Then the function v satisfies $a \le v(i,j) \le b$ for $(i,j) \in \Lambda$, where $a = 0$ and $b = m_1 m_2 pq$. Now suppose that $n_1 = m_1$ records have been retrieved from the `instrument` relation and $n_2 < m_2$ records from the `CO2_data` relation. Also suppose that the `id` attribute is a key for the `instrument` relation. Setting $\boldsymbol{n} = (n_1, n_2)$, we can take $a(\boldsymbol{n}) = 0$ and $b(\boldsymbol{n}) = b/m_1 = m_2 pq$. $\square$

3.2 AVG Queries

We now consider a query of the form (3.1) with *op* equal to `AVG`. Set $v(i_1, i_2, \ldots, i_K)$ equal to the value of *expression* when applied to tuples $t_{1,i_1}, t_{2,i_2}, \ldots, t_{K,i_K}$ (whether or not these tuples jointly satisfy *predicate*). In analogy to the single-table case, denote by S the subset of $R_1 \times R_2 \times \cdots \times R_K$ such that $(t_{1,i_1}, t_{2,i_2}, \ldots, t_{K,i_K}) \in S$ if and only if tuples $t_{1,i_1}, t_{2,i_2}, \ldots, t_{K,i_K}$ jointly satisfy *predicate*. Set $u(i_1, i_2, \ldots, i_K) = 1$ if $(t_{1,i_1}, t_{2,i_2}, \ldots, t_{K,i_K}) \in S$ and $u(i_1, i_2, \ldots, i_K) = 0$ otherwise. If *predicate* is empty, then the final answer to the query can be written in the form (3.5) and the confidence-interval formulas of Section 3.1 apply. If *predicate* is not empty, then the final answer $\tilde{\mu}(S)$ can be written in the form

$$\tilde{\mu}(S) = \frac{\tilde{\theta}(uv)}{\tilde{\theta}(u)}, \tag{3.17}$$

where $\tilde{\theta}(f)$ is defined as in (3.4) and

$$uv(i_1, \ldots, i_K) = u(i_1, \ldots, i_K) v(i_1, \ldots, i_K)$$

for $(i_1, i_2, \ldots, i_K) \in \Lambda$. When n_k tuples have been retrieved from relation R_k for $1 \le k \le K$, the running aggregate is given by $\tilde{Y}_{\boldsymbol{n}}(S) = \tilde{T}_{\boldsymbol{n}}(uv)/\tilde{T}_{\boldsymbol{n}}(u)$, where $\boldsymbol{n} = (n_1, n_2, \ldots, n_K)$ and $\tilde{T}_{\boldsymbol{n}}(f)$ is defined by (3.6) for $f \in \tilde{\mathcal{F}}$.

To develop large-sample confidence intervals, we assume as in Section 3.1 that n records have been retrieved from each input relation and that (3.3) holds. The running aggregate is therefore given by $\tilde{Y}_n(S) = \tilde{T}_n(uv)/\tilde{T}_n(u)$, where $\tilde{T}_n(f)$ is defined by (3.7). Define $\tilde{\theta}(f;k,j)$, $\tilde{\theta}_q(f)$, $\tilde{T}_n(f;k,j)$, and $\tilde{T}_{n,q}(f)$ as in (3.8)–(3.11). Set

$$\tilde{\theta}_{q,r}(f,g) = \sum_{k=1}^{K} \left(\frac{1}{m_k} \sum_{l=1}^{m_k} \left(\tilde{\theta}(f;k,l) - \tilde{\theta}(f)\right)^q \left(\tilde{\theta}(g;k,l) - \tilde{\theta}(g)\right)^r \right)$$

and

$$\tilde{T}_{n,q,r}(f,g) = \sum_{k=1}^{K} \Bigg((n-1)^{-1} \times \sum_{j=1}^{n} \Big(\tilde{T}_n(f;k,j) - \tilde{T}_n(f)\Big)^q \Big(\tilde{T}_n(g;k,j) - \tilde{T}_n(g)\Big)^r \Bigg) \tag{3.18}$$

for $q, r \ge 0$ and $f, g \in \tilde{\mathcal{F}}$. The following result extends Theorem 3.12.

Theorem 3.19 *Let $\tilde{f}_1, \tilde{f}_2, \ldots, \tilde{f}_d \in \tilde{\mathcal{F}}$ and suppose that $0 < \tilde{\theta}_2(\tilde{f}_i) < \infty$ for $1 \le i \le d$. Then*

$$\sqrt{n}\Big(\tilde{T}_n(\tilde{f}_1) - \tilde{\theta}(\tilde{f}_1), \ldots, \tilde{T}_n(\tilde{f}_d) - \tilde{\theta}(\tilde{f}_d)\Big) \Rightarrow (X_1, X_2, \ldots, X_d)$$

as $n \to \infty$, where $(X_1, X_2, \ldots, X_d)$ has a multivariate normal distribution with mean vector $(0,0,\ldots,0)$ and covariance matrix $\tilde{\Sigma}$ satisfying

$$\tilde{\Sigma}(i,j) = \begin{cases} \tilde{\theta}_{1,1}(\tilde{f}_i, \tilde{f}_j) & \text{if } i \ne j; \\ \tilde{\theta}_2(\tilde{f}_i) & \text{if } i = j. \end{cases}$$

Moreover, $\tilde{S}_n \to \tilde{\Sigma}$ a.s. as $n \to \infty$, where

$$\tilde{S}_n(i,j) = \begin{cases} \tilde{T}_{n,1,1}(\tilde{f}_i, \tilde{f}_j) & \text{if } i \ne j; \\ \tilde{T}_{n,2}(\tilde{f}_i) & \text{if } i = j. \end{cases}$$

The first assertion of the theorem is a straightforward consequence of Theorem 3.12 and the Cramér-Wold Theorem (see Theorem 29.4 in [1]). The proof of the second assertion uses an inductive argument similar to the proof of a.s. convergence in [5, Theorem 6].

Taking $\tilde{f}_1 = uv$ and $\tilde{f}_2 = u$ in Theorem 3.19 and taking $h(x,y) = x/y$ in Lemma 1, we find that $\sqrt{n}\big(\tilde{Y}_n(S) - \tilde{\mu}(S)\big) \Rightarrow N\big(0, \tilde{\gamma}/\tilde{\theta}^2(u)\big)$, where

$$\tilde{\gamma} = \tilde{\theta}_2(uv) - 2\tilde{\rho}\tilde{\theta}_{1,1}(uv,u) + \tilde{\rho}^2 \tilde{\theta}_2(u)$$

and $\tilde{\rho} = \tilde{\theta}(uv)/\tilde{\theta}(u)$. Moreover, $\tilde{G}_n/\tilde{T}_n^2(u) \to \gamma$ a.s. as $n \to \infty$, where

$$\tilde{G}_n = \tilde{T}_{n,2}(uv) - 2\tilde{R}_n \tilde{T}_{n,1,1}(uv,u) + \tilde{R}_n^2 \tilde{T}_{n,2}(u)$$

and $\tilde{R}_n = \tilde{T}_n(uv)/\tilde{T}_n(u)$. Arguing as in Section 2.1, we obtain the final result

$$\varepsilon_n = \left(\frac{z_p^2 \tilde{G}_n}{n \tilde{T}_n^2(u)} \right)^{1/2}$$

To obtain a deterministic confidence interval, define $h(\boldsymbol{n})$, $a(\boldsymbol{n})$, and $b(\boldsymbol{n})$ as in Section 3.1 for $\boldsymbol{n} = (n_1, n_2, \ldots, n_K)$. Also set

$$\tilde{I}_{\boldsymbol{n}} = \sum_{i_1=1}^{n_1} \sum_{i_2=1}^{n_2} \cdots \sum_{i_K=1}^{n_K} u(L_{1,i_1}, L_{2,i_2}, \ldots, L_{K,i_K}).$$

Then, by an argument almost identical to that in Section 2.3, $\tilde{Y}_{\boldsymbol{n}}(S) - \underline{\varepsilon}_{\boldsymbol{n}} \le \tilde{\mu}(S) \le \tilde{Y}_{\boldsymbol{n}}(S) + \overline{\varepsilon}_{\boldsymbol{n}}$, where

$$\overline{\varepsilon}_{\boldsymbol{n}} = \frac{h(\boldsymbol{n})\Big(\big(b(\boldsymbol{n}) \vee 0\big) - \big(\tilde{Y}_{\boldsymbol{n}}(S) \wedge 0\big)\Big)}{\tilde{I}_{\boldsymbol{n}} + h(\boldsymbol{n})}$$

and

$$\underline{\varepsilon}_{\boldsymbol{n}} = \frac{h(\boldsymbol{n})\Big(\big(\tilde{Y}_{\boldsymbol{n}}(S) \vee 0\big) - \big(a(\boldsymbol{n}) \wedge 0\big)\Big)}{\tilde{I}_{\boldsymbol{n}} + h(\boldsymbol{n})}.$$

3.3 VARIANCE and STDEV Queries

In this section we consider queries of the form (3.1) with *op* equal to `VARIANCE` or `STDEV`. Define the set S and the functions u, v as in Section 3.2. The final answer to the `VARIANCE` or `STDEV` query can be written as

$$\begin{aligned}\tilde{\sigma}^2(S) &= \frac{1}{|S|}\sum_{(t_{1,i_1},\ldots,t_{K,i_K})\in S}\big(v(i_1,\ldots,i_K)-\tilde{\mu}(S)\big)^2\\ &= \frac{\tilde{\theta}(uv^2)\tilde{\theta}(u)-\tilde{\theta}^2(uv)}{\tilde{\theta}^2(u)}\end{aligned}$$

or $\tilde{\sigma}(S)$, respectively, where $\tilde{\theta}(f)$ is defined by (3.4) and $\tilde{\mu}(S)$ is defined by (3.17). After n_k tuples have been retrieved from relation R_k for $1 \le k \le K$, the running aggregates for an `VARIANCE` or `STDEV` query are given by

$$\tilde{Z}_{\boldsymbol{n}}(S) = \frac{\tilde{T}_{\boldsymbol{n}}(uv^2)\tilde{T}_{\boldsymbol{n}}(u)-\tilde{T}^2_{\boldsymbol{n}}(uv)}{\tilde{T}^2_{\boldsymbol{n}}(u)}$$

or $\sqrt{\tilde{Z}_{\boldsymbol{n}}(S)}$, respectively, where $\boldsymbol{n} = (n_1, n_2, \ldots, n_K)$ and $\tilde{T}_{\boldsymbol{n}}(f)$ is defined by (3.6). When $n_1 = n_2 = \cdots = n_K = n$, write $\tilde{Z}_n(S)$ instead of $\tilde{Z}_{\boldsymbol{n}}(S)$.

For the case $n_1 = n_2 = \cdots = n_K = n$, large-sample confidence intervals can be developed by the same arguments as in the single-table case, except that the CLT in Theorem 3.19 is used instead of the standard multivariate CLT. The resulting formulas are given by

$$\varepsilon_n = \left(\frac{z_p^2\tilde{G}'_n}{n\tilde{T}_n^2(u)}\right)^{1/2}.$$

and

$$\varepsilon_n = \left(\frac{z_p^2\tilde{G}'_n}{4n\tilde{Z}_n(S)\tilde{T}_n^2(u)}\right)^{1/2}$$

for `VARIANCE` and `STDEV` queries, respectively, where

$$\begin{aligned}\tilde{G}'_n &= \tilde{T}_{n,2}(uv^2) - 4\tilde{R}_{n,2}\tilde{T}_{n,1,1}(uv^2,uv)\\ &\quad+(4\tilde{R}^2_{n,2}-2\tilde{R}_{n,1})\tilde{T}_{n,1,1}(uv^2,u)\\ &\quad+(4\tilde{R}_{n,1}\tilde{R}_{n,2}-8\tilde{R}^3_{n,2})\tilde{T}_{n,1,1}(uv,u)\\ &\quad+(2\tilde{R}^2_{n,2}-\tilde{R}_{n,1})^2\tilde{T}_{n,2}(u)+4\tilde{R}^2_{n,2}\tilde{T}_{n,2}(uv),\end{aligned}$$

$\tilde{R}_{n,1} = \tilde{T}_n(uv^2)/\tilde{T}_n(u)$, $\tilde{R}_{n,2} = \tilde{T}_n(uv)/\tilde{T}_n(u)$, and $\tilde{T}_{n,2}(f)$ and $\tilde{T}_{n,1,1}(f,g)$ are defined as in (3.11) and (3.18), respectively.

As with large-sample confidence intervals, the arguments for deterministic confidence intervals closely parallel the arguments for the single-table case and lead to the formulas

$$\begin{aligned}\overline{\varepsilon}_{\boldsymbol{n}} &= \tilde{\lambda}'_{\boldsymbol{n}} - \frac{1}{\tilde{I}_{\boldsymbol{n}}}\tilde{Z}_{\boldsymbol{n}}(S)\\ \underline{\varepsilon}_{\boldsymbol{n}} &= \frac{h(\boldsymbol{n})+1}{h(\boldsymbol{n})+\tilde{I}_{\boldsymbol{n}}}\tilde{Z}_{\boldsymbol{n}}(S)\end{aligned}$$

for `VARIANCE` queries and

$$\begin{aligned}\overline{\varepsilon}_{\boldsymbol{n}} &= \left(\frac{\tilde{I}_{\boldsymbol{n}}-1}{\tilde{I}_{\boldsymbol{n}}}\tilde{Z}_{\boldsymbol{n}}(S)+\tilde{\lambda}'_{\boldsymbol{n}}\right)^{1/2} - \sqrt{\tilde{Z}_{\boldsymbol{n}}(S)}\\ \underline{\varepsilon}_{\boldsymbol{n}} &= \sqrt{\tilde{Z}_{\boldsymbol{n}}(S)} - \left(\frac{\tilde{I}_{\boldsymbol{n}}-1}{h(\boldsymbol{n})+\tilde{I}_{\boldsymbol{n}}}\tilde{Z}_{\boldsymbol{n}}(S)\right)^{1/2}.\end{aligned}$$

for `STDEV` queries, where

$$\begin{aligned}\tilde{\lambda}'_{\boldsymbol{n}} &= \frac{h(\boldsymbol{n})}{\tilde{I}_{\boldsymbol{n}}+h(\boldsymbol{n})}\frac{\big(b(\boldsymbol{n})-a(\boldsymbol{n})\big)^2}{4}\\ &\quad+\frac{\tilde{I}_{\boldsymbol{n}}\big(h(\boldsymbol{n})\wedge\tilde{I}_{\boldsymbol{n}}\big)}{\Big(\tilde{I}_{\boldsymbol{n}}+\big(h(\boldsymbol{n})\wedge\tilde{I}_{\boldsymbol{n}}\big)\Big)^2}\\ &\quad\times\Big(\big(\tilde{Y}_{\boldsymbol{n}}(S)-a(\boldsymbol{n})\big)\vee\big(b(\boldsymbol{n})-\tilde{Y}_{\boldsymbol{n}}(S)\big)\Big)^2\end{aligned}$$

and $a(\boldsymbol{n})$, $b(\boldsymbol{n})$, $h(\boldsymbol{n})$, and $\tilde{I}_{\boldsymbol{n}}$ are defined as in Sections 3.1 and 3.2.

3.4 Page-Level I/O and Indexes

In the above sections we have assumed that exactly one tuple is obtained from each of the K input relations at each retrieval step. Under a suitable reinterpretation (as in Section 2.5), the confidence-interval formulas developed above apply to the case in which a page of tuples is obtained at each retrieval step. Moreover, the formulas also can be made to apply when one or more of the input relations has a combined index on its join and selection attributes, and this index is used during join processing. For example, suppose that there are two input relations R_1 and R_2 having common attribute a and that the aggregate of interest is the average of the function $v_0(\cdot,\cdot)$ over all elements of $R_1 \bowtie R_2$, where $R_1 \bowtie R_2$ is the equijoin of R_1 and R_2 on attribute a. At the ith retrieval step, tuple t_{L_i} is retrieved from R_1, the index on R_2 is used to retrieve all of the tuples in R_2 that join with t_{L_i}, and these tuples are used to update the running average. For $1 \le i \le m_1$ denote by H_i the set of all indices j such that $1 \le j \le m_2$ and $t_{2,j}.a = t_{1,i}.a$. The final answer to the aggregation query can then be written as

$$\tilde{\mu} = \frac{(1/m)\sum_{i=1}^m u(i)v(i)}{(1/m)\sum_{i=1}^m u(i)},$$

where $m = m_1$, $u(i) = |H_i|$, and $v(i) = |H_i|^{-1}\sum_{j\in H_i} v_0(i,j)$. (Here we take $0/0 = 0$.) Moreover, the running aggregate can be written in the form (2.16), where u and v are defined as above and n is the number of retrieval steps. Thus, for example, the formula (2.17) for large sample confidence intervals applies, provided that u, v, and n are interpreted as above and that tuples are retrieved from R_1 in random order. Other complex retrieval schemes for various aggregation queries can be handled in a similar manner.

4 Computational Considerations

The discussion in Sections 2 and 3 shows that to compute large-sample and deterministic confidence intervals for single-table queries, it is necessary to update quantities of the form $T_n(f)$, $T_{n,2}(f)$, $T_{n,4}(f)$, and $T_{n,1,1}(f,g)$ at each retrieval step. Similarly, the quantities $\tilde{T}_n(f)$, $\tilde{T}_{n,2}(f)$, and $\tilde{T}_{n,1,1}(f,g)$ must be updated at each retrieval step in order compute large-sample and deterministic confidence intervals for multi-table queries. It is of course desirable to update all of these quantities in an efficient and numerically stable manner. In this section we discuss several updating methods and their complexity. For ease of exposition, we assume that one tuple is retrieved from each input relation at a retrieval step; the modifications for page-level and index-assisted retrieval are straightforward and the complexity results are similar. We focus on the time and space requirements over and above what is required to compute the running aggregate itself.

For single-table queries, $T_n(f)$, $T_{n,2}(f)$, $T_{n,4}(f)$ can be updated using the numerically stable recursions

$$nT_n(f) = (n-1)T_{n-1}(f) + f(L_n),$$

$$(n-1)T_{n,2}(f) = (n-2)T_{n-1,2}(f) + \frac{\Delta_n^2(f)}{n(n-1)}, \qquad (4.1)$$

$$(n-1)T_{n,3}(f) = (n-2)T_{n-1,3}(f) + \frac{3T_{n-1,2}(f)\Delta_n(f)}{n(n-1)} - \frac{(n-2)\Delta_n^3(f)}{n^2(n-1)^2},$$

and

$$(n-1)T_{n,4}(f) = (n-2)T_{n-1,4}(f) + \frac{4T_{n-1,3}(f)\Delta_n(f)}{n(n-1)} + \frac{6T_{n-1,2}(f)\Delta_n^2(f)}{n^2(n-1)^2} + \frac{(n^2-3n+3)\Delta_n^4(f)}{n^3(n-1)^3},$$

where $\Delta_n(f) = (n-1)\big(T_{n-1}(f) - f(L_n)\big)$ for $n \geq 1$. The recursions for $T_n(f)$ and $T_{n,2}(f)$ are well known [2]; the remaining two recursion formulas appear to be new. We also have the well-known recursion

$$(n-1)T_{n,1,1}(f,g) = (n-2)T_{n-1,1,1}(f,g) + \frac{\Delta_n(f)\Delta_n(g)}{n(n-1)}$$

It follows from the above results that the large-sample and deterministic confidence intervals for single-table queries discussed in this paper can be updated in constant time and require a small, constant amount of storage.

In the case of a K-table query, recall the definitions of $\tilde{T}_n(f)$ and $\tilde{T}_n(f;k,j)$ from (3.7) and (3.10), respectively, and observe that $n^{-1}\sum_{j=1}^n \tilde{T}_n(f;k,j) = \tilde{T}_n(f)$ for $1 \leq k \leq K$ and $n \geq 1$. Thus, the quantity $\tilde{T}_{n,2}(f)$ defined by (3.11) can be viewed as the sum of K terms, each of the same mathematical form as $T_{n,2}(f)$. A recursion of the form (4.1) can therefore be used to update each of the K terms. The key difference between computing $T_{n,2}(f)$ and $\tilde{T}_{n,2}(f)$ is that, in the latter case, quantities of the form $\tilde{T}_n(f;k,j)$ must be stored and updated at each retrieval step. This updating can be done as follows. For $n \geq 1$, denote by J_n the set of index vectors $(i_1, i_2, \ldots, i_K)$ such that (i) $i_k = n$ for some $1 \leq k \leq K$ and (ii) $f(L_{1,i_1}, \ldots, L_{K,i_K}) > 0$. Initialize the computation by retrieving a tuple from each input relation and setting $y = f(L_{1,1}, L_{2,1}, \ldots, L_{K,1})$, $w_1 = w_2 = \cdots = w_K = 0$, $q_{1,1} = q_{2,1} = \cdots = q_{K,1} = y$, and $n = 1$. At each subsequent retrieval step, execute the algorithm given in Figure 2. After the algorithm has executed, the variable z is equal to $\tilde{T}_{n,2}(f)$, the variable y/n is equal to $\tilde{T}_n(f)$, and each variable $q_{k,j}/n^{K-1}$ is equal to $\tilde{T}_n(f;k,j)$. It can be seen that after n retrieval steps, the cumulative computation cost is roughly $O(KM_n)$, where $M_n = \sum_{i=1}^n |J_n|$. The cumulative storage required is $O(V_n)$, where $V_n = \sum_{k=1}^K \sum_{j=1}^n B_n(f;k,j)$ and

$$B_n(f;k,j) = \begin{cases} 1 & \text{if } \tilde{T}_n(f;k,j) \neq 0; \\ 0 & \text{otherwise.} \end{cases}$$

(Here we assume that sparse storage techniques are used when appropriate.) Thus, after n retrieval steps the worst-case cumulative computation cost is $O(Kn^K)$ and the worst-case cumulative storage requirement is $O(Kn)$.

We conjecture that, in many cases, the computation and storage costs will not be a problem. For K input relations

```
n := n + 1;
y := y((n − 1)/n)^(K−1);
for k = 1, 2, ..., K
{
  q_{k,n} = 0;
  w_k := w_k((n − 1)/n)^(2(K−1)) + y²/(n(n − 1));
  x_k = y;
}
for (i_1, ..., i_K) ∈ J_n
{
  for k = 1, 2, ..., K
  {
    j := i_k;
    x_k := x_k − q_{k,j}/n^(K−1);
    w_k := w_k − (x_k − (n − 1)n^(1−K) q_{k,j})/n
               × (x_k − (n − 1)n^(1−K) q_{k,j})/(n − 1);
    q_{k,j} := q_{k,j} + f(L_{1,i_1}, ..., L_{K,i_K});
    w_k := w_k + (x_k − (n − 1)n^(1−K) q_{k,j})/n
               × (x_k − (n − 1)n^(1−K) q_{k,j})/(n − 1);
    x_k := x_k + q_{k,j}/n^(K−1);
  }
}
y := x_1;
z := Σ_{k=1}^K w_k/(n − 1);
```

Figure 2: An Updating Algorithm.

$R_1, R_2, \ldots, R_K$, the usual goal is to estimate an aggregate that is evaluated over all elements of the Cartesian product $W = R_1 \times R_2 \times \cdots \times R_K$ that satisfy a join predicate and perhaps also a selection predicate; denote by W_0 the subset of W over which the aggregate is evaluated. Typically, the user will abort the query when the running confidence interval is sufficiently short. If $|W_0|/|W|$ is small, then M_n and V_n are small as long as n is not extremely large, so that the cumulative computation and storage costs are small. If $|W_0|/|W| \approx 1$, then the cumulative computation cost is $\approx O(Kn^K)$ and the cumulative storage requirement is $\approx O(Kn)$ after the nth retrieval step. In this case, however, n is usually small: many nonzero evaluations of the function f are obtained at each retrieval step, so that typically the running confidence interval narrows rapidly and the user can abort the query after a small number of steps. A similar discussion applies to the quantity $\tilde{T}_{n,1,1}(f,g)$.

Acknowledgements

The author wishes to thank Joe Hellerstein for introducing him to the problem of online aggregation and for providing Figure 1. The author is also grateful to Frank Olken and Sean Wang for their helpful comments and to an anonymous referee for recommending the inclusion of computational results as in Section 4.

References

[1] P. Billingsley. *Probability and Measure*. Wiley, New York, second edition, 1986.

[2] T. F. Chan, G. H. Golub, and R. J. LeVeque. Algorithms for computing the sample variance: Analysis and recommendation. *Amer. Statist.*, 37:242–247, 1983.

[3] W. G. Cochran. *Sampling Techniques*. Wiley, New York, third edition, 1977.

[4] P. J. Haas. Hoeffding inequalities for join-selectivity estimation and online aggregation. IBM Research Report RJ 10040, IBM Almaden Research Center, San Jose, CA, 1996.

[5] P. J. Haas, J. F. Naughton, S. Seshadri, and A. N. Swami. Selectivity and cost estimation for joins based on random sampling. *J. Comput. System Sci.*, 52:550–569, 1996.

[6] J. M. Hellerstein. The case for online aggregation. Technical Report UCB//CSD-96-908, EECS Computer Science Division, University of California, Berkeley, CA, 1996.

[7] J. M. Hellerstein, P. J. Haas, and H. J. Wang. Online aggregation. In *Proc. 1997 ACM SIGMOD Intl. Conf. Managment of Data*, pages 171–182. ACM Press, 1997.

[8] W. Hoeffding. Probability inequalities for sums of bounded random variables. *J. Amer. Statist. Assoc.*, 58:13–30, 1963.

[9] R. G. Miller, Jr. *Simultaneous Statistical Inference*. Springer-Verlag, New York, second edition, 1981.

[10] R. J. Serfling. *Approximation Theorems of Mathematical Statistics*. Wiley, New York, 1980.

A Derivation of Deterministic Confidence Interval for $Z_n(S)$

Set $D_n(S) = \sum_{i=n+1}^m uv(L_i)/\sum_{i=n+1}^m u(L_i)$ and $Q_n(S) = \sum_{i=n+1}^m u(L_i)\big(v(L_i) - D_n(S)\big)^2$. Using Equation (1.5b) in [2], we find that

$$\sigma^2 \le \frac{I_n - 1}{I_n + B_n} Z_n(S) + \frac{1}{I_n + B_n} Q_n(S) + \frac{I_n B_n}{(I_n + B_n)^2} \big(\overline{Y}_n(S) - D_n(S)\big)^2.$$

In analogy with (2.13) and (2.14), $Q_n(S) \le B_n(b-a)^2/4$ and

$$\big(\overline{Y}_n(S) - D_n(S)\big)^2 \le \Big(\big(\overline{Y}_n(S) - a\big) \vee \big(b - \overline{Y}_n(S)\big)\Big)^2.$$

The function $f(x) = cx/(c+x)^2$ achieves its maximum value over $[0,\infty)$ at the point $x = c$ and has no other global or local maxima. For $a > 0$, therefore, the function f achieves its maximum value over the interval $[0,a]$ at the point $x = a \wedge c$. Thus,

$$\frac{I_n B_n}{(I_n + B_n)^2} \le \frac{I_n\big((m-n) \wedge I_n\big)}{\Big(I_n + \big((m-n) \wedge I_n\big)\Big)^2}.$$

Combining the above results, we find that $\sigma^2 \le Z_n(S) + \overline{\varepsilon}_n$, where $\overline{\varepsilon}_n$ is given by (2.19). A similar argument shows that $\sigma^2 \ge Z_n(S) - \underline{\varepsilon}_n$, where $\underline{\varepsilon}_n$ is given by (2.20).