How do you keep John Doe anonymous when reviewing data on the larger picture? Techniques such as adding "noise" to the data help protect the privacy of the individual.

Inference Controls for Statistical Databases

Dorothy E. Denning, SRI International

Jan Schlörer, University of Ulm, Germany

The goal of statistical databases is to provide frequencies, averages, and other statistics about groups of persons (or organizations), while protecting the privacy of the individuals represented in the database. This objective is difficult to achieve, since seemingly innocuous statistics contain small vestiges of the data used to compute them. By correlating enough statistics, sensitive data about an individual can be inferred. As a simple example, suppose there is only one female professor in an electrical engineering department. If statistics are released for the total salary of all professors in the department and the total salary of all male professors, the female professor's salary is easily obtained by subtraction. The problem of protecting against such indirect disclosures of sensitive data is called the *inference problem.*

Over the last several decades, census agencies have developed many techniques for controlling inferences in population surveys. These techniques are applied before data are released so that the distributed data are free from disclosure problems. The data are typically released either in the form of microstatistics, which are files of "sanitized" records, or in the form of macrostatistics, which are tables of counts, sums, and higher order statistics.

Starting with a study by Hoffman and Miller,[1] computer scientists began to look at the inference problem in on-line, general-purpose database systems allowing both statistical and nonstatistical access. A hospital database, for example, can give doctors direct access to a patient's medical records, while hospital administrators are permitted access only to statistical summaries of the records. Up until the late 1970's, most studies of the inference problem in these systems led to negative results; every conceivable control seemed to be easy to circumvent, to severely restrict the free flow of information, or to be intractable to implement. Recently, the results have become more positive, since we are now discovering controls that can potentially keep security and information loss at acceptable levels for a reasonable cost.

This article surveys some of the controls that have been studied, comparing them with respect to their security, information loss, and cost. The controls are divided into two categories: those that place restrictions on the set of allowable queries and those that add "noise" to the data or to the released statistics. The controls are described and further classified within the context of a lattice model.

Lattice model

The lattice model consists of a lattice structure of logical tables. The tables are derived from a logical database of N records, where each record contains attribute values for an individual or organization. An example of a database of employee records is given by Employee (Sex, Department, Level, Degree, Years, Salary). We have purposely excluded an employee name or identifier, since the model does not allow retrieval of records by identifying keys. Note that some attributes, such as Sex, are nonnumeric, whereas others, such as Salary, are numeric.

Statistics are computed for subsets of records having common attribute values. A set of records is specified by a characteristic formula F, which informally, is any logical formula over the values of the attributes using the operators OR ($+$), AND ($\&$), and NOT ($\sim$). An example

of a formula for the employee database is $F=$ (Degree = MS) & [(Level = 5) + (Level = 6)], which specifies employees at level 5 or 6 with an MS degree.

The set of records whose values match a formula F is called the query set of F. A query set that can be specified using the values of m distinct attributes (but no fewer) is called an m-set. The query set for F in the above equation, for example, is a two-set.

Tables. Given any m attributes, a collection of related statistics over all possible combinations of values for the attributes can be represented as an m-dimensional table (or m-table for short), each dimension corresponding to one attribute. Figure 1 shows a two-table T_{AB} of counts broken down by two attributes A and B, where the domain of attribute A has four values and the domain of attribute B has five values. We use array notation to denote the statistics in a table. T_{AB} [2,4], for example, denotes the entry in row 2, column 4 of T_{AB}; that is, the number of records satisfying the formula $[(A = a_2)$ & $(B = b_4)]$, which is one. Table S_{AB} in Figure 2 is similar but contains sums over a sensitive attribute such as Salary.

Because an arbitrary formula F over m attributes $A_1,\ldots,A_m$ corresponds to a union of m-sets in the m-table over $A_1,\ldots,A_m$, statistics such as counts, sums, and other finite moments over F can be computed by adding the statistics for the m-sets in the m-table. The number of records satisfying the formula $[\sim(A = a_2)$ & $(B = b_4)]$ is given by $T_{AB}[1,4] + T_{AB}[3,4] + T_{AB}[4,4] = 37$. A statistic giving the total salary is similarly computed by adding entries in table S_{AB} of Figure 2. We will concentrate mainly on such "additive statistics."[2] Thus, an m-table over attributes $A_1,\ldots,A_m$ forms a basis for computing the statistics for all possible subsets of records over the m attributes. If each attribute A_i has a domain of size $|A_i|$, then the size of the m-table is

$$S_m = \prod_{i=1}^{m} |A_i|$$

and the total number of statistics that can be derived from the table is $2^{S_m} - 1$ (excluding the statistic for the empty set).

A table does not correspond to a physical structure of the database, but is rather a derived view of the database. A database with M attributes has 2^M such tables corresponding to all possible subsets of the attributes. There is exactly one M-table, where the records in each elementary M-set are indistinguishable. Each m-table partitions the N records of the database into s_m query sets.

Lattice structures. The set of all tables for a given statistical function forms a lattice structure. Figure 3 shows a detailed view of the lattice of counts over attributes A and B from Figure 1. Note that the statistics in the table T_A correspond to the row sums of those in T_{AB}; similarly, those in T_B correspond to the column sums of those in T_{AB}. The table T_{ALL} consists of a single statistic computed over all records in the database; it corresponds to the vector sum of either T_A or T_B. Thus, the zero- and one-tables of Figure 3 do not contain any new information that cannot be derived from the two-table T_{AB}.

If we add attributes C and D to the database, we get the lattice shown in Figure 4. In this case, all statistics in the lattice can be derived from those in the four-table T_{ABCD}. The table T_{AB}, for example, corresponds to the marginal sums of either T_{ABC} or T_{ABD}, which in turn are marginal sums of T_{ABCD}.

For any additive statistic, a table corresponds to the marginal sums of the tables directly below it in the lattice. The linear relations among the statistics of the tables are the primary cause of inference problems.

In many statistical applications, some attributes have hierarchical structures; e.g., cities are grouped by state. Such attribute hierarchies are readily incorporated into the lattice model.[3]

Sensitive statistics

The objective is to control the inference of sensitive statistics. The exact criterion for defining sensitivity is determined by the policies of the system. One criterion used by the US Census Bureau for economic data is the "n-respondent, k%-dominance" criterion, which defines a sensitive statistic to be one in which n or fewer records constitute more than k% of the total;[4] n and k are parameters of the database, and are usually kept secret. Here, we shall assume that a sensitive statistic is one with a query set of size one. Thus, the count $T_{AB}[2,4]$ in table T_{AB} is sensitive (similarly for the sum $S_{AB}[2,4]$). In practice, a statistic computed from a group of size two can also be classified as sensitive because a user with supplementary knowledge about one value can deduce the other from the statistic. The disclosure risk,

A \ B	b_1	b_2	b_3	b_4	b_5
a_1	0	5	14	7	0
a_2	6	2	8	1	23
a_3	13	18	2	27	4
a_4	9	0	17	3	6

T_{AB}

Figure 1. Two-table T_{AB} of counts.

A \ B	b_1	b_2	b_3	b_4	b_5
a_1	0	65	166	73	0
a_2	82	19	112	13	253
a_3	159	202	31	317	63
a_4	101	0	170	42	71

S_{AB}

Figure 2. Two-table S_{AB} of sums.

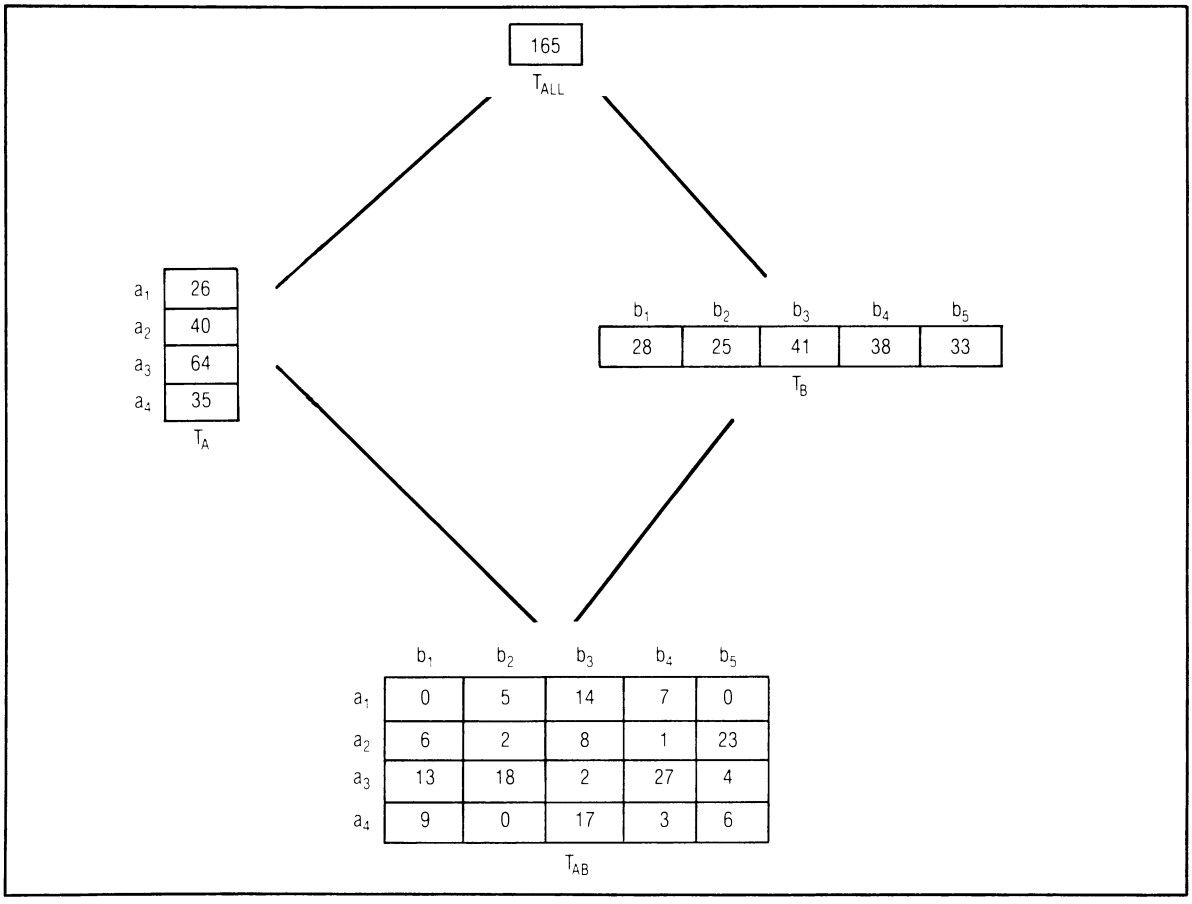


Figure 3. Lattice of tables over attributes *A* and *B*.

or identification risk, of a table is given by the number (or percentage) of sensitive cells in the table.

Personal disclosure (compromise) occurs when the user can infer a previously unknown sensitive statistic about an identifiable individual.[5,6] Disclosure can be either exact or approximate, positive, or negative.[5,7-11] Releasing counts for query sets of size zero always leads to negative disclosure because we can deduce that a particular individual does not have the associated properties. Releasing the count $T_{AB}[1,1] = 0$, for example, reveals that no individual has both a_1 and b_1.

Clearly, all sensitive statistics must be restricted (not permitted). In addition, we must restrict nonsensitive statistics that could lead to disclosure of sensitive ones. Such disclosures arise mainly from the linear relations in the lattice structure. For example, the sensitive count $T_{AB}[2,4]$ can be computed by subtracting the other entries in column 4 of the two-table from the column sum $T_B[4]$ in the one-table T_B; likewise for the sensitive sum $S_{AB}[2,4]$.

Permitting a count of one in a m-table T over attributes $A_1,\ldots,A_m$ does not usually lead to compromise when the individual is not also identified in some parent of T in the lattice. The values of $A_1,\ldots,A_m$ must be known in advance to identify the individual's cell in the table, and the count itself reveals no new information about the individual. To compromise, we must either obtain (directly or by inference) a count over $A_1,\ldots,A_m$,

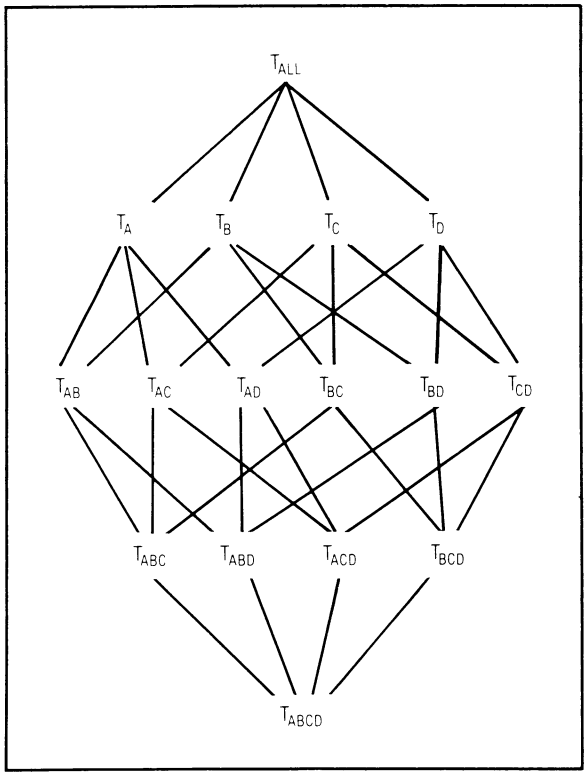


Figure 4. Lattice of tables over attributes *A*, *B*, *C*, and *D*.

A_{m+1}, where the value of A_{m+1} is unknown, or get a higher order statistic, such as a sum, over A_{m+1}. Nevertheless, we aim to protect all statistics over query sets of size one.

The lattice model has proved a powerful and effective security model for studying the inference problem and proposed solutions. It has provided a framework for estimating identification risks,[12,13] and for evaluating and comparing different controls. It has suggested new controls. Its table structure is closely tied to the publication format of many applications, e.g., Census data. Although the lattice model is inadequate as a stand-alone data model for a database system, it should be an important element of any complete data model when security is an issue.

Inference controls

To control inferences, information must be removed from tables with sensitive statistics. There are two general approaches to doing this: restriction and perturbation.

Restriction techniques aim to control inference of sensitive statistics by withholding additional nonsensitive ones. Figure 5 shows various strategies, classified vertically according to whether they restrict at the table level or cell level in the lattice, and horizontally according to whether they are memoryless, audit based, or a priori. Although some controls do not fit neatly in our classification system, the system is useful for comparative studies.

Table-level controls restrict complete m-tables of statistics, including the statistics for all m-sets over the associated attributes. Cell-level controls aim to restrict only the sensitive cells of an m-table, and just enough nonsensitive statistics over the associated attributes to prevent inference.

Memoryless controls attempt to determine whether release of a statistic could lead to compromise without keeping a record of previous queries or a list of permitted statistics. Audit-based controls, as the name suggests, keep an audit trail for determining whether release of a statistic, when correlated with previously released statistics, could lead to compromise. To the best of our knowledge, Fellegi[14] was the first to suggest this approach, though he did not propose a practical method for implementing it. A priori controls determine in advance a fixed set of statistics that can be released without causing compromise.

Figure 5 also indicates which restriction techniques analyze the characteristic formula of a query, as opposed to looking only at the size or composition of the query set. Controls that analyze the formula are potentially better equipped to interpret the semantics of a query and, therefore, to decide whether answering the query could lead to compromise. At the cell level, however, this analysis can be quite costly.

Perturbation techniques remove information from the tables by adding noise to the statistics. These techniques are generally used with some form of restriction technique, applied at either the table or cell level.

Inference controls are judged by three factors: security, information loss, and cost. Security is measured by the relative number of sensitive statistics that can be inferred by circumventing the control and by the difficulty of doing so. Information loss is measured by the number of nonsensitive statistics or tables of statistics that are unnecessarily restricted by the control, and by the amount of noise injected in permitted statistics. This measure is not entirely adequate, however, because it does not account for the relative importance of a statistic for a particular study. In general, the statistics in m-tables for small m values (i.e., those near the top of the lattice) are more valuable than those further down. Cost is determined by the initial implementation requirements, including any a priori computation, plus the overhead in query processing. Because higher levels of security usually imply higher levels of information loss and cost, the challenge is to find a control with the right balance for the given application and associated risks.

Perturbation techniques are judged by two additional factors, bias and consistency. These are discussed later when we describe different methods of perturbation. First, however, we describe methods of table restriction and cell restriction.

	TABLE LEVEL	CELL LEVEL
MEMORYLESS	Order* Relative Table Size* (s_m/N-criterion) Explicit Risk Estimation*	Query Set Size Implied Queries*
AUDIT BASED		Overlap Audit Expert
A PRIORI	Determine Transformability* (Data Swap)	Cell Suppression* Grouping or Rolling Up* Partitioning*

*Decision whether to answer a query involves analyzing the characteristic formula.

Figure 5. Classification of output restriction controls.

Table restriction techniques

Table-level controls restrict complete tables of statistics, namely those higher dimensional tables in the lattice that have, or are likely to have, a positive disclosure or identification risk (i.e., one or more cells corresponding to a single individual). If a given m-table over attributes $A_1,\ldots,A_m$ is restricted, then all statistics for the m-sets over these attributes are also restricted. Because studies of these controls have focused thus far on counts, we shall also focus on counts here.

Order control. This memoryless control restricts all m-tables of counts for $m > d$ where the threshold d is a parameter of the database. The parameter d is chosen so

that most, if not all, statistics in the permitted m-tables are nonsensitive (or are expected to be nonsensitive). The control is easily implemented by counting the number of attributes appearing in the characteristic formula of a query; if there are more than d attributes, the query is not answered. Choosing $d = 1$ for the lattice of Figure 3, for example, restricts the two-table T_{AB}, permitting the remaining three tables. For the complete lattice in Figure 4, only five of the 16 tables would be permitted.

Using the dimension of a table as a restriction criterion has the advantage of simplicity but also has the disadvantage of being poorly matched to the actual disclosure risk of a table.[2] This disadvantage occurs because an m-table partitions the entire database of N records into s_m groups, where s_m is the number of cells in the table. Thus, large tables are more likely to have identifications than smaller ones. For a database with 500 records, for example, a two-table with $10 \times 50 = 500$ cells would be more likely to have identifications than a two-table with $2 \times 3 = 6$ cells. The next control uses the table size as the restriction criterion.

Relative table-size control (s_m/N-criterion). This memoryless control restricts an m-table of counts if its relative size s_m/N exceeds a threshold $1/k$, where k is a parameter of the database chosen to reduce, but not necessarily eliminate, the possibility of disclosure.[2] Thus, an m-table is permitted if the average number of records falling into each cell is at least k.

The control is easily applied to a query by taking the product of the domain sizes for the attributes named in the characteristic formula to obtain the table size s_m. Figure 6 illustrates the effect of applying the criterion with $k = 10$ to the lattice in Figure 4, where the database has $N = 165$ records, and the domain sizes of the attributes are $|A| = 4$, $|B| = 5$, $|C| = 8$, $|D| = 2$. Here, eight of the 16 tables are permitted.

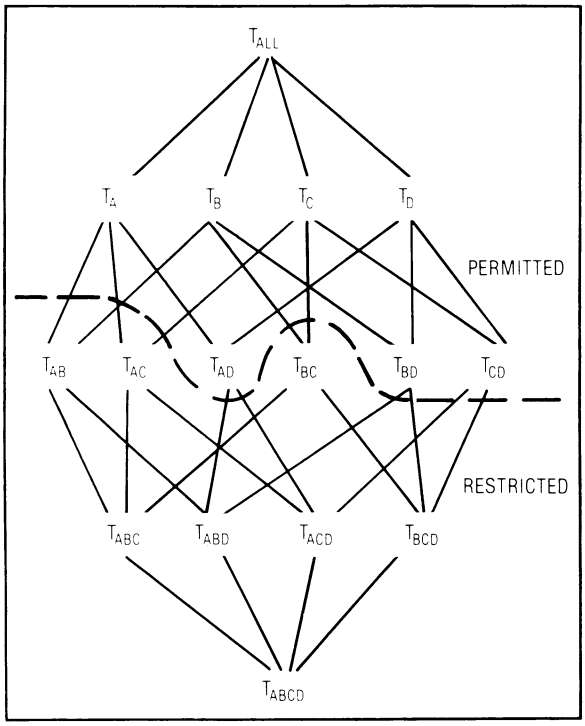


Figure 6. Relative table-size control.

The criterion goes back to an observation by Block and Olsson[12] that the identification risk in an m-table will be approximately $e^{-N/5^m}$ if all m attributes are independent and equidistributed. Although the attributes of real databases are more or less interdependent and nonuniformly distributed, data collected from 27 databases showed a strong relation between s_m/N and the risk.[13] Taking all databases together, we found an identification risk of approximately one percent for $k = 17$ ($s_m/N \approx 0.6$).

The relative table-size control does not recognize tables with identifications when the average cell size is k or more. The next control aims to recognize such tables by using frequency distributions.

Explicit risk estimation. Here, the decision of whether to restrict an m-table T is based on the frequency distributions of the attributes for the table. We have studied two restriction criteria.[2] The first estimates the number of identifications in all parents of T in the lattice; if any of these tables has z or more estimated identifications, then T is restricted, where the threshold z is a parameter of the database. The second estimates the number of identifications in T directly, restricting T if the number exceeds z. The first criterion has the advantage of providing a better estimate of the risk, but the disadvantage of requiring more computation. Both criteria are more closely related to the disclosure risk than relative table size but are also more costly.

The level of security provided by each of the preceding table controls is determined by the control's threshold. Unfortunately, setting this threshold to provide a high level of security can cause too many tables to be restricted. Often, a better strategy is to adjust the threshold for a somewhat lower level of security and use perturbation to control the remaining risk.

All the preceding table restriction techniques for counts are based on the attributes $A_1, \ldots, A_m$ named in formula F of the query. This basis suggests a possible approach for handling higher order statistics such as sums; if the statistical function is over an additional k attributes, $A_{m+1}, \ldots, A_{m+k}$, then we could apply the criterion for counts, using all $m + k$ attributes. Unfortunately, this approach would be overly restrictive. Suppose, for example, that the sums in the two-table S_{AB} of Figure 2 are over an attribute C with a domain size of 10. Then the two-table S_{AB} which has 20 cells, is likely to contain fewer sensitive statistics than the three-table of counts over A, B, and C, which has 200 cells. Yet, applying the criterion for two-tables of counts to two-tables of sums can be overly permissive, since sums contain more information than counts.[2]

Determine transformability (existence of data swap). Although the preceding controls can provide a high level of security, especially when combined with some perturbation technique, none guarantees there are no disclosures in permitted tables. Moreover, none even guaran-

tees that a sensitive statistic in a restricted table is safe from compromise. To prove that exact disclosure of a sensitive statistic in an $(m+1)$-table T is impossible, we must prove that the table is m-transformable; that is, at least one other $(m+1)$-table can be derived from the permitted ascendants of T in the lattice by swapping data in the records.[15,16] Because such a proof is costly (being an NP-complete problem), we could not apply such a criterion as a memoryless control at query processing time. Instead, we must apply it prior to answering any query and retain the information in the database. Even then, such a control is impractical for most applications.

To overcome these limitations, Reiss has proposed an approximate data swapping technique for off-line systems releasing microstatistics.[17] Here, a portion of the original database is replaced with a randomly generated database having approximately the same low-order statistics as the original database. But because the scheme modifies the original data, it is not suitable for general-purpose databases.

Cell restriction techniques

Here, the decision of whether to permit a statistic is determined by the query set for the statistic rather than by just the table with which the statistic is associated. With a properly chosen restriction criterion, we can then permit some (or even most) of the cells in tables that would otherwise be restricted by table restriction techniques. Thus, cell restriction potentially results in much less information loss than table restriction.

Query-set-size control. One of the earliest cell-level controls is a memoryless control that simply restricts statistics computed over extremely small or large query sets; that is, for a database of N records, sets smaller than k or larger than $N-k$ are restricted, where k is a parameter of the systems.[1,14,18] Statistics computed over small query sets must obviously be restricted, since they are sensitive. Why we should restrict statistics computed over large query sets is less obvious, since these statistics are clearly nonsensitive. A moments reflection, however, reveals that their release could disclose sensitive statistics. To see why, let $q(F)$ denote any additive statistic (e.g., count or sum) over the query set identified by formula F. Because the complement of a large query set F of size $|F|$ is a small query set $\sim F$ of size $N - |F|$, the sensitive statistic $q(\sim F)$ can be computed by subtracting the nonsensitive statistic $q(F)$ from the nonsensitive statistic $q(\text{ALL})$. Thus, the query set size control is effectively checking the sensitivity of both a given query set F and its "implied query set" $\sim F$.

A query-set-size control is trivial to implement. It can be valuable when combined with other protection techniques,[8,19] but, unfortunately, is easily subverted when used alone. The most powerful tools to subvert it are called *trackers*.[6,8,18,20,21] The basic idea is to pad small query sets with enough extra records to put them in the allowable range, and then subtract the effect of the padding. As an example, consider the sensitive count $q(F) = T_{AB}[2,4] = 1$ for the query set identified by $F = [(A=2)\ \&\ (B=4)]$ in Figure 1. Because counts are additive, $q(F)$ can be computed by padding F with some set R, e.g., $R = [(A=1)\ \&\ (B=4)]$, and subtracting out R; that is, $q(F) = q(F+R) - q(R) = 8-7 = 1$. The formula R is an example of a *union tracker*. (Because sums are additive, the same formula can be used to compute the sum $q(F) = S_{AB}[2,4]$ in Figure 2, where q is now interpreted to mean sum rather than count.)

Implied queries control. Friedman and Hoffman[22] proposed to thwart tracker attacks by extending the concept of an implied query set to sets other than just complements, and by checking the sizes of these sets at query processing time. A set D is implied by F if $q(D)$ can be computed from $q(F)$ plus other statistics that may be permitted (primarily statistics over lower dimensional query sets). The implied query sets are determined by analyzing the formula F.

Unfortunately, we can have many implied query sets. If F is an arbitrary formula over m attributes, then we must check all cells in the m-table over these attributes, restricting F and the entire table of statistics if there is a single sensitive cell.[2,8,23] Thus, whereas the control is applied at the cell level, it effectively restricts at the table level, but is more costly than most table-level controls. If the syntax for F is restricted to logical AND, then by borrowing techniques from cell suppression (discussed later), we can reduce the number of cells that must be checked to 2^m and permit partial tables. This process usually, but not always, guarantees security. Note that for additive statistics, restricting the syntax to logical AND does not mean that statistics for formulas with OR and NOT cannot be computed. This condition occurs because the statistics for an AND syntax correspond to the cells in the m-tables, which form a basis for computing statistics over arbitrary subsets of records.

Overlap control. Here, an audit trail is used to restrict query sets having more than a small specified number of records in common. This approach thwarts tracker attacks, which employ overlapping query sets, but in so doing renders a database useless for statistical studies. Most studies require statistics for highly overlapping query sets—e.g., total salary of all individuals plus total salary of males and total salary of females. Releasing a statistic over the entire database would thus preclude releasing most other statistics. Moreover, an overlap control does not prevent many other types of attack under so-called "key-specified" queries.[24-26] Because of its excessive information loss, the control has no practical interest.

The audit expert. Recently, Chin and Ozsoyoglu have developed an algorithm that uses an audit trail to control inferences with sum queries.[27] Their audit expert records a complete history of all queries about a confidential attribute in a binary matrix H having N columns and at most $N-1$ linearly independent rows. Each column represents one individual in the database, and the rows represent a basis for the set of queries deducible from the previously answered queries; thus, each query that has been answered or could be deduced is expressed as a

linear combination of the rows of H. When a new query is asked, the matrix is updated so that if the query discloses the exact value for the j^{th} individual, updating the matrix introduces a row with all zeros except for a "1" in column j; thus the potential compromise is easily detected. The matrix can be udpated in $O(N^2)$ time, so the method may be practical for small databases.

The audit expert does not control approximate disclosure or disclosures through query sets larger than one. To do so would require an analysis comparable to that used for cell suppression, which would increase the cost considerably. It also leads to some information loss, the exact amount depending on the order in which queries are made. Even if queries are batched, it is difficult to minimize information loss, however, because the problem of finding a maximal set of statistics that can be released is NP-complete.[27]

Cell suppression. This a priori control is used by census agencies to protect data published in tabular form. The linear relations among all cells of a table (including the marginal sums in the parent tables) are analyzed to determine whether sensitive cells can be deduced (exactly or approximately) from those that are released; additional cells, called complementary suppressions, are suppressed for as long as possible.[4,10,11] The suppressed cells fall into hypercubes of size 2^m, where m is the size of the table. Some attempt is made in selecting cells for suppression to minimize information loss. Figure 7 shows how cell suppression can be used to protect the sensitive cell $T_{AB}[2,4]$.

Cell suppression can be expensive. For this reason, it is used as an a priori control rather than on a per query basis. Even determining a set of suppressions a priori can be infeasible if statistics are to be released over many attributes. Moreover, the dynamics of multipurpose database systems can rapidly outdate suppression patterns. Although this problem could be resolved by periodically computed and storing a complete set of tables for statistical purpose (e.g., by using the "box structures" employed by the Swedish Bureau of Statistics[28]), such a solution is not likely to be affordable.

Grouping or rolling up. As the name suggests, this technique merges attributes together.[9,10,14] Figure 8 shows how this strategy can be used to protect the sensitive cell $T_{AB}[2,4]$ in table T_{AB} by merging attribute values b_4 and b_5 or attribute values a_1 and a_2. Both lattices are permitted.

Grouping is often rejected for off-line systems because it hinders the comparability of different tables.[14] In on-line systems, users can choose from alternative groupings such as those in Figure 8. To be secure, however, the

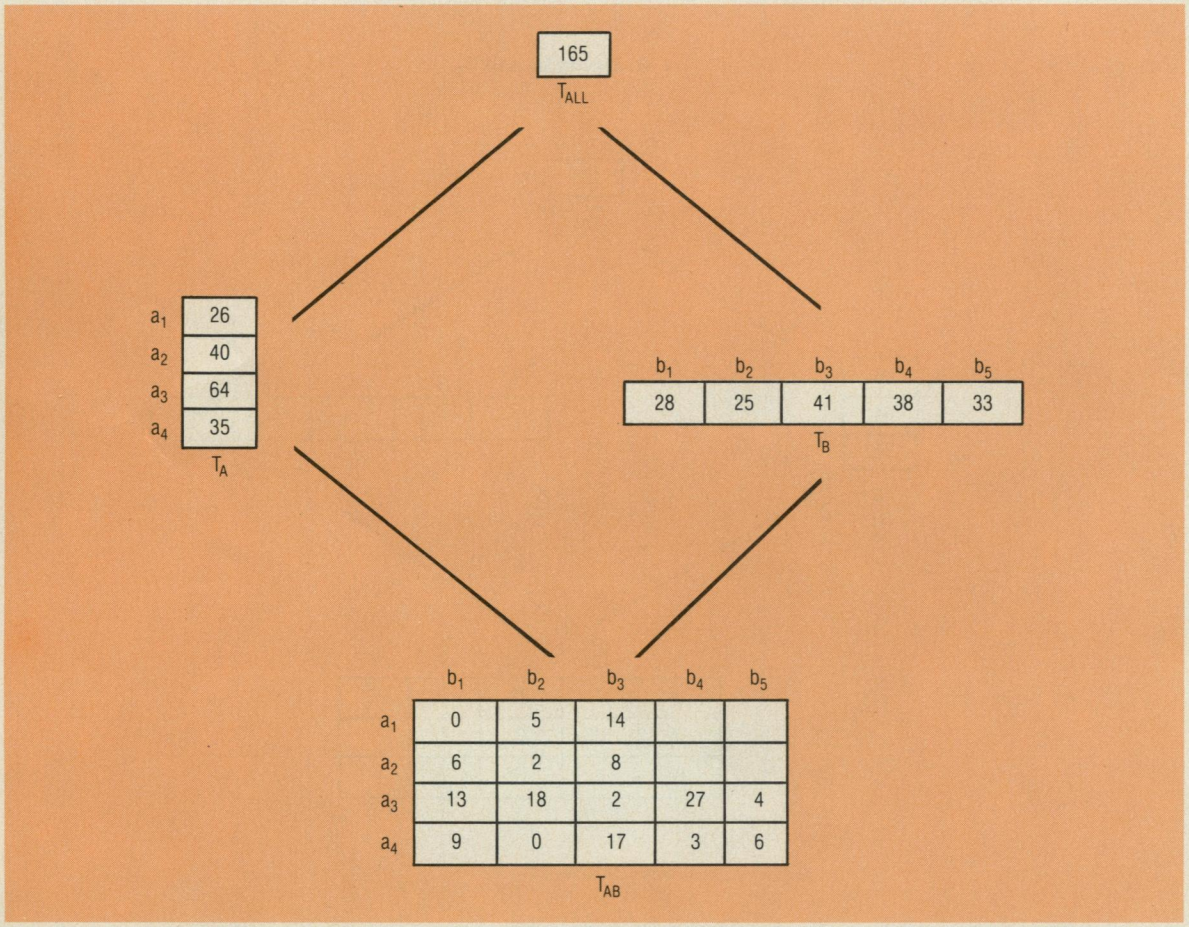
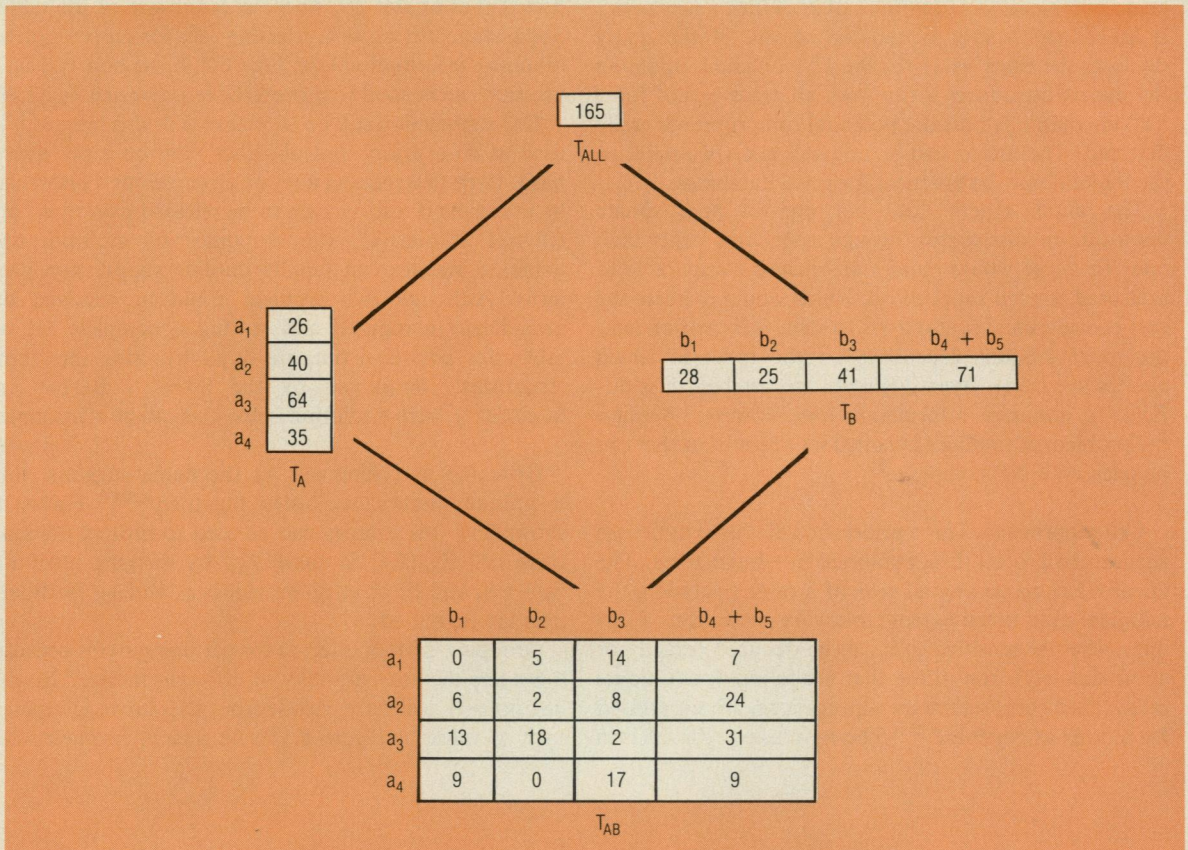
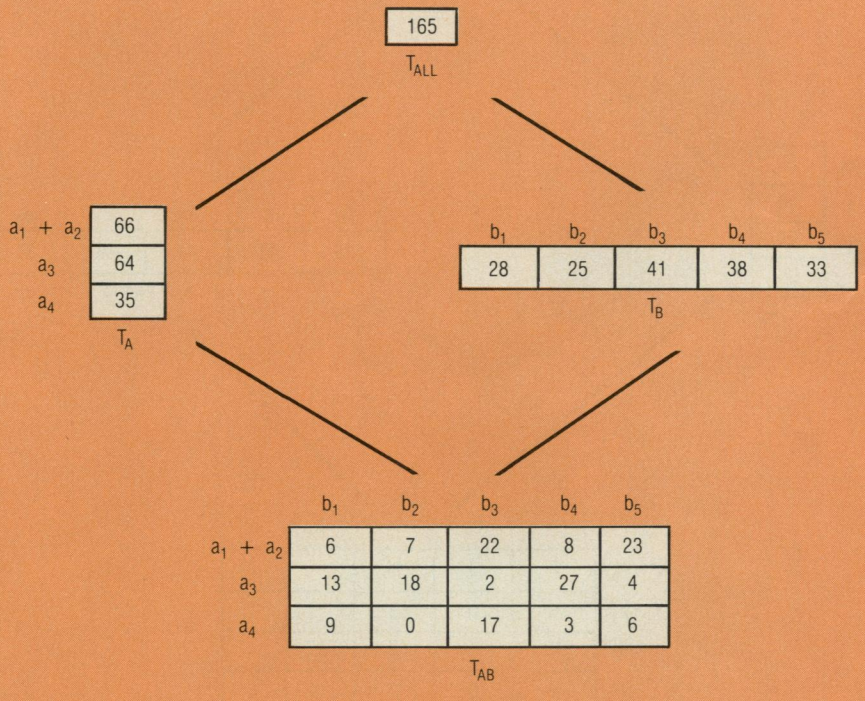


Figure 7. Cell suppression.



a. Grouping b_4 and b_5.



b. Grouping a_1 and a_2.

Figure 8. Grouping or rolling up.

groupings must be determined in advance and have a hierarchical strucure; arbitrary groupings are not allowed.[29]

Partitioning. This technique is similar to grouping but is applied to attribute values rather than attributes, and alternative groupings are now allowed.[30-33] Using the p values of the attributes, the database is partitioned into a set of mutually exclusive, nonoverlapping atomic populations (A-populations), where no A-population consists of a single record. A statistic with query set F is permitted only if F is an A-population or union of A-populations. Either lattice in Figure 8 can be used as a basis for protecting the sensitive cell $T_{AB}[2,4]$. Unlike grouping, however, both lattices are not permitted.

One of the greatest drawbacks to partitioning is the potential information loss.[34] With the partitioning of Figure 8a, for example, we must restrict the statistics $T_B[4]$ and $T_B[5]$ in the one-table T_B of column sums, because the query sets for these statistics would subdivide an A-population. In general, we may have to restrict many statistics for the cells in one-tables, even though these statistics seldom lead to disclosure.

Dummy records, which pad A-populations of size one, can be inserted in the database to reduce the number of groups that must be merged (and thereby the information loss), and to enable record processing in pairs to thwart compromises due to database updates. As the techniques are now formulated, however, this insertion would bias statistics such as counts and means. Because of its high security, partitioning is an interesting concept, but further study is needed to determine whether it can be practical.

Comparison of output restriction techniques

Figure 9 compares the various table and cell restriction techniques. For each control, we give a rough estimate of the control's relative security, information loss, and cost. Although some estimates are based on hard data, others are more speculative, so the diagrams should not be interpreted too strictly.

In general, the most efficient mechanisms have the least security or greatest information loss, but acceptable solutions can still be found to the inference problem. For example, a criterion for a memoryless table restriction that is based on relative table size (s_m/N), combined with a simple size restriction and perturbation technique, seems to strike a reasonable balance between security and information loss at low cost.

Perturbation techniques

By introducing noise into the statistics, perturbation techniques try to permit more statistics than can be permitted with restriction techniques alone. Like restriction techniques, they are judged by their security, cost, and information loss. In this context, information loss refers to the variance of the error in the perturbed statistics rather than the number of restricted statistics or tables of statistics.

Two other factors are important when evaluating perturbation techniques: bias and consistency. Bias refers to the difference between a true statistic and the expectation of its perturbed estimate. The bias should be zero or at least as small as possible.

Consistency refers to the lack of contradictions or paradoxes in the perturbed statistics. Contradictions arise, for example, when repetitions of the same query yield different results, or when an additive statistic corresponding to a row or column sum differs from that obtained by adding the statistics in the row or column of the table (i.e., the linear equations do not hold). Contradictions also arise when an average avg(F) for formula F differs from the computed average sum(F)/count(F). Paradoxes arise when negative values are returned for counts or, for example, when a total of 31.72 children is returned.

Unfortunately, the goals of consistency and statistical quality of perturbed statistics can conflict, making a perfect consistency probably unrealizable.[9,11,35,36] Yet some consistency is needed for user acceptance and security. Users of statistical databases often react negatively to perturbation schemes, particularly when they are inconsistent. Inconsistencies can also lead to compromise. For example, when repetitions of the same query give different responses, the true statistic can be estimated by averaging the responses.

Perturbation techniques can be classified according to whether they are record based or result based. Given a query $q(F)$, record-based techniques perturb the input to the statistical function for q. Perturbation is accomplished either by taking a sample of the records satisfying F and estimating $q(F)$ from the sample or by perturbing the data used to compute $q(F)$ as they are extracted from the records satisfying F. The cost and variance of both approaches are proportional to the query set size.

Result-based techniques perturb the result $q(F)$ after it has been correctly computed; the perturbation typically involves some form of rounding. The cost is a small constant, and the variance is usually a constant proportional to the square of the rounding base. Because errors are confined to a known interval, rounding is often more acceptable to users than record-based perturbation, where errors are confined only by confidence intervals.

Record-based perturbation

Random sample queries. In random sample queries, a statistic $q(F)$ is computed from a random sample drawn from the query set for F.[8,19]. When the database's system fetches each record satisfying F, it applies a selection function that determines whether the record is used to compute the statistic. The set of selected records forms a sampled query set, which is then used to compute the statistic. The sample size is chosen to provide a balance between security and information loss. The resulting statistics are unbiased.

To provide consistency and prevent averaging attacks, the selection function should be generated from the

characteristic formula F of the query such that equivalent queries return the same result. For this approach to work, we must either couple the rounding process to some normal form of the characteristic formula, which cannot be done efficiently for an arbitrary formula, or couple it to the query set, which is less effective.[3] A possible solution is to restrict the syntax of a characteristic formula, e.g., to logical AND, which would allow easy reduction to a normal form.[2,3,36]

Sampling introduces uncertainty into the composition of query sets so that record identifications become difficult if not impossible. Random sample queries are equivalent to a special form of random data perturbation, which hides the highest percentage of records for a given variance.[36]

Random data perturbation. Let

$$\text{sum}(F) = \sum_{i=F} x_i$$

be a true sum over the values x_i in a query set satisfying F. Random data perturbation replaces input x_i to the computation with $x_i' = x_i + e_i$.[35,36] The error e_i is randomly chosen from a distribution with expected value of zero; thus, the perturbed sum is unbiased. The variance in e_i is chosen to provide a balance between security and information loss.

The technique can also be used with counts, where the x_i for records in F are interpreted as "1's" since count(F) can be written as

$$\sum_{i=F} 1$$

This application can lead to inconsistencies, however, in the form of negative counts. Similar problems arise with other statistics, such as correlation coefficients.

Although random perturbation schemes can be vulnerable to averaging attacks when e_i is generated randomly at query processing time, Beck's scheme[35] thwarts such attacks without introducing large errors.

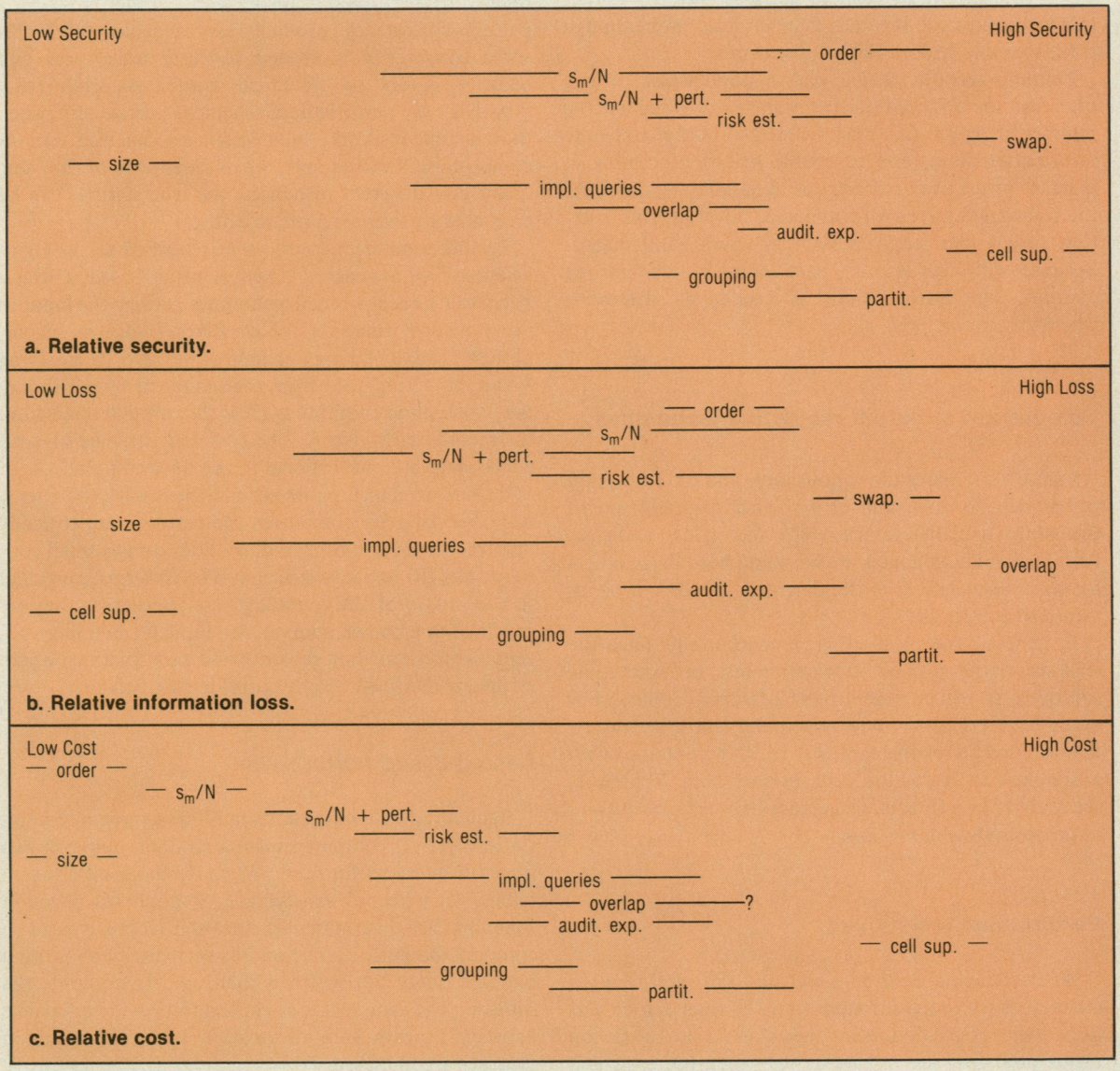
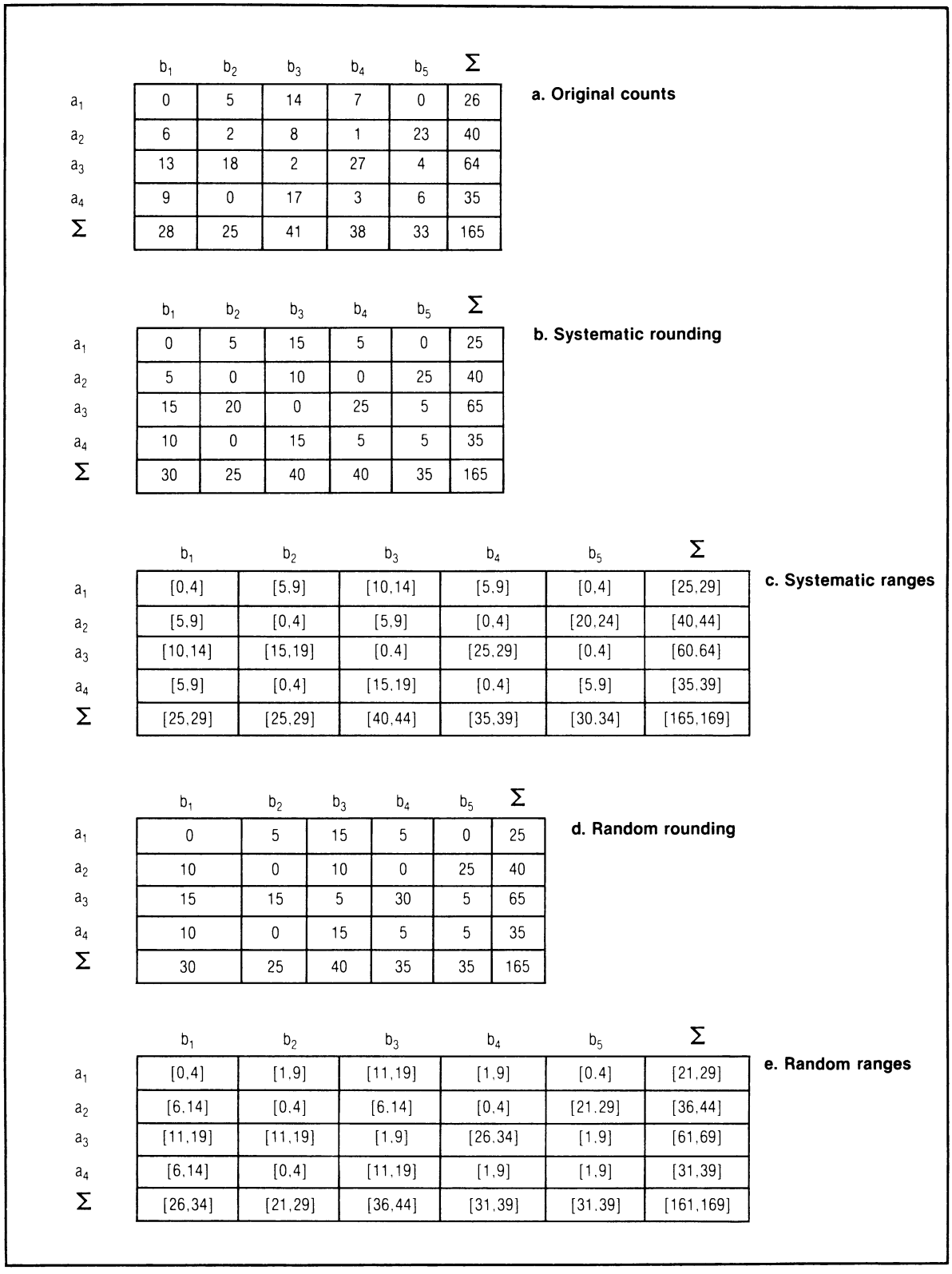


Figure 9. Comparison of restriction techniques, where s_m/N is the relative table size (+ pert. is with perturbation), swap is determine transformability (data swap), and size is query set size.

Rounding techniques

Systematic rounding and systematic ranges. In systematic rounding,[9,32,36-38] we round a statistic to the closest integer multiple of a fixed rounding base b. Figure 10b illustrates for the counts in Figure 4, where the rounding base b is five. For clarity, we show the counts in Figure 4 as a single table in Figure 10a, where the zero- and one-tables are displayed as marginal sums for the two-table, a common publication format for tables. Note that some



	b_1	b_2	b_3	b_4	b_5	Σ
a_1	0	5	14	7	0	26
a_2	6	2	8	1	23	40
a_3	13	18	2	27	4	64
a_4	9	0	17	3	6	35
Σ	28	25	41	38	33	165

a. Original counts

	b_1	b_2	b_3	b_4	b_5	Σ
a_1	0	5	15	5	0	25
a_2	5	0	10	0	25	40
a_3	15	20	0	25	5	65
a_4	10	0	15	5	5	35
Σ	30	25	40	40	35	165

b. Systematic rounding

	b_1	b_2	b_3	b_4	b_5	Σ
a_1	[0,4]	[5,9]	[10,14]	[5,9]	[0,4]	[25,29]
a_2	[5,9]	[0,4]	[5,9]	[0,4]	[20,24]	[40,44]
a_3	[10,14]	[15,19]	[0,4]	[25,29]	[0,4]	[60,64]
a_4	[5,9]	[0,4]	[15,19]	[0,4]	[5,9]	[35,39]
Σ	[25,29]	[25,29]	[40,44]	[35,39]	[30,34]	[165,169]

c. Systematic ranges

	b_1	b_2	b_3	b_4	b_5	Σ
a_1	0	5	15	5	0	25
a_2	10	0	10	0	25	40
a_3	15	15	5	30	5	65
a_4	10	0	15	5	5	35
Σ	30	25	40	35	35	165

d. Random rounding

	b_1	b_2	b_3	b_4	b_5	Σ
a_1	[0,4]	[1,9]	[11,19]	[1,9]	[0,4]	[21,29]
a_2	[6,14]	[0,4]	[6,14]	[0,4]	[21,29]	[36,44]
a_3	[11,19]	[11,19]	[1,9]	[26,34]	[1,9]	[61,69]
a_4	[6,14]	[0,4]	[11,19]	[1,9]	[1,9]	[31,39]
Σ	[26,34]	[21,29]	[36,44]	[31,39]	[31,39]	[161,169]

e. Random ranges

Figure 10. Output perturbation.

statistics are inconsistent. For example, the sum of the rounded values in column 4 of the table, namely 35, does not equal the rounded row sum, namely 40. Systematic rounding can also be biased, although the average bias approaches zero if the database is sufficiently large.

Systematically rounded output is often taken literally, and is thus often misinterpreted. This tendency can be avoided by giving systematic ranges of the form $[kb,(k+1)b-1]$ instead.[39] This technique is illustrated in Figure 10c. Systematic ranges are also easier to analyze for security than systematic rounding.

Systematic rounding always returns the same answer to the same query, thereby thwarting attacks based on averaging different answers to the same query. Under certain conditions, however, systematic rounding can be circumvented directly by derounding or indirectly using trackers.[9,36,37,40] Because many of these attacks require using logical OR in the characteristic formula, we have another argument for possibly restricting the syntax of queries to logical AND.

Random rounding and random ranges. This technique,[9,36-38] illustrated in Figure 10d, randomly rounds a statistics either to the next higher or the next lower multiple of the rounding base.

Unlike systematic rounding, random rounding is unbiased. To provide consistency and prevent averaging attacks, the decision whether to round up or down should be determined by the query in such a way that equivalent queries always give the same response. This requirement is similar to that for random sample queries and seems to need a restricted syntax for characteristic formulas.

For a given rounding base, random rounding has about the same (small) variance as systematic rounding but is somewhat less vulnerable to derounding and tracker attacks.[36] By analogy to systematic rounding, random rounding can be replaced by random ranges to avoid misinterpretations. Figure 10e illustrates. If users will accept a restricted syntax for characteristic formulas, then random ranges is probably the best strategy overall for perturbing results on a per-query basis.

Controlled rounding. This method of rounding provides an even smaller variance and greater consistency than random ranges by requiring the marginal sums of rounded statistics to equal their rounded sum.[41-43] For example, the table in Figure 10b could be modified to meet the requirements for controlled rounding by replacing the column 4 sum of 40 with 35. Unfortunately, controlled rounding is now too expensive to apply on a per-query basis. It is, however, attractive as an a priori control for off-line publication of tables.

Perturbing arithmetic means. Providing consistency of arithmetic means with counts and sums is a difficult problem for which no solution is entirely satisfactory.[36] For a given query avg(F), one approach is to return the computed values s/c, where s is the perturbed statistic for sum(F) and c is the perturbed statistic for count(F). This approach provides acceptable results with random sample queries.[19,36] It can lead to questionable results, however, with rounding techniques or with small query sets. Another approach is to compute averages correctly and return the computed product $c * \text{avg}(F)$ for sum(F). This approach is usually preferable with rounded counts, although additivity of sums is lost.[9,36]

Dynamic databases. The dynamics of a database introduce additional security risks if the users with statistical access are allowed to update the database.[30-33] To control these risks, techniques such as record-based perturbation and partitioning may be necessary. If, however, the users are not allowed to update the database, the dynamics of the database can probably be ignored. Note that any form of output perturbation, including rounding, renders small changes to the database less visible.

Although no single control alone satisfies the conflicting goals of high security, low information loss, and efficiency, a scheme that combines simple restriction criteria, such as query-set size and relative table size, with simple perturbation techniques can provide an acceptable level of security for many applications without being overly restrictive or costly. Threat monitoring (audit trails)[1] can also be used to determine whether a user has attempted to obtain restricted data, thereby reducing the need for large perturbations or tight restriction criteria. For applications requiring extremely high levels of security or low levels of information loss, a priori techniques such as cell suppression and controlled rounding can be used—although at the price of increased cost. The best strategy for a particular application will depend on its objectives and risks.

There is a paradoxical tradeoff between the power of the query language and the amount of obtainable information. If the syntax of the query language is restricted to logical AND, the only statistics released are those corresponding to table cells in the lattice; statistics for query sets defined by logical OR and NOT are not released. With such a restricted syntax, however, the database can release more table cells, and more accurate statistics for these cells. Because additive statistics for arbitrary formulas can be computed from the table cells, more information is effectively released than with a free syntax, where controls must be tighter. ■

Acknowledgments

We are grateful to the referees for helpful suggestions. This research was partially supported by NSF grant MCS 80-15484, and by Stiftung Volkswagenwerk.

References

1. L. J. Hoffman and W. F. Miller, "Getting a Personal Dossier from a Statistical Data Bank," *Datamation,* Vol. 16, No. 5, May 1970, pp. 74-75.
2. D. E. Denning, J. Schlörer, and E. Wehrle, *Memoryless Inference Controls for Statistical Databases,* Computer Sciences Dept., Purdue University, West Lafayette, Ind., 1982.

3. D. E. Denning, *A Security Model for Statistical Databases,* Computer Sciences Dept., Purdue University, West Lafayette, Ind., 1983.
4. L. H. Cox, "Suppression Methodology and Statistical Disclosure Control," *J. American Statistical Assoc.,* Vol. 75, No. 370, June 1980, pp. 377-385.
5. M. I. Haq, "Insuring Individual's Privacy from Statistical Data Base Users," *AFIPS Conf. Proc.,* Vol. 44, 1975 NCC, AFIPS Press, Montvale, N.J., pp. 941-946.
6. J. Schlörer, "Disclosure from Statistical Databases: Quantitative Aspects of Trackers," *ACM Trans. Database Systems,* Vol. 5, No. 4, Dec. 1980, pp. 467-492.
7. T. Dalenius, "Towards a Methodology for Statistical Disclosure Control," *Statistisk Tidskrift,* Vol. 15, 1977, pp. 429-444.
8. D. E. Denning, *Cryptography and Data Security,* Addison-Wesley, Reading, Mass., 1982.
9. I. P. Fellegi and J. L. Phillips, "Statistical Confidentiality: Some Theory and Applications to Data Dissemination," *Annals Economic Social Measurement,* Vol. 3, No. 2, Apr. 1974, pp. 399-409.
10. L. Olsson, "Protection of Output and Stored Data in Statistical Databases," *ADB-Information,* Vol. 4, 1975, Statistika Centralbyran, Stockholm, Sweden.
11. E. Rapaport and B. Sundgren, "Output Protection in Statistical Databases," S/SYS-E04, Nat'l. Central Bureau of Statistics, Stockholm, Sweden (invited paper, Warsaw meeting of the Int'l Statistical Inst., Oct. 1975).
12. H. Block and L. Olsson, "Bakvagsidentifiering," *Statistisk Tidskrift,* Vol. 14, 1976, pp. 135-144.
13. J. Schlörer and L. Zick, *Empirical Investigations on the Identification Risk in Statistical Databases,* Klinische Dokumentation, Universitat Ulm, W. Germany, June 1982.
14. I. P. Fellegi, "On the Question of Statistical Confidentiality," *J. American Statistical Assoc.,* Vol. 67, No. 337, Mar. 1972, pp. 7-10.
15. T. Dalenius and S. P. Reiss, "Data-Swapping—A Technique for Disclosure Control," *Proc. Section Survey Research Methods,* American Statistical Assoc., 1979, pp. 191-196.
16. J. Schlörer, "Security of Statistical Databases: Multidimensional Transformation," *ACM Trans. Database Systems,* Vol. 6, No. 1, Mar. 1981, pp. 95-112.
17. S. P. Reiss, "Practical Data-Swapping: The First Steps," *Proc. 1980 Symp. Security and Privacy,* Apr. 1980, pp. 38-45.
18. J. Schlörer, "Identification and Retrieval of Personal Records from a Statistical Data Bank," *Methods Information Mediation,* Vol. 14, No. 1, Jan. 1975, pp. 7-13.
19. D. E. Denning, "Secure Statistical Databases under Random Sample Queries," *ACM Trans. Database Systems,* Vol. 5, No. 8, Sept. 1980, pp. 291-315.
20. D. E. Denning, P. J. Denning, and M. D. Schwartz, "The Tracker: A Threat to Statistical Database Security," *ACM Trans. Database Systems,* Vol. 4, No. 1, Mar. 1979, pp. 76-96.
21. D. E. Denning and J. Schlörer, "A Fast Procedure for Finding a Tracker in a Statistical Database," *ACM Trans. Database Systems,* Vol. 5, No. 1, Mar. 1980, pp. 88-102.
22. A. D. Friedman and L. J. Hoffman, "Towards a Fail-Safe Approach to Secure Databases," *Proc. 1980 Symp. Security and Privacy,* Apr. 1980, pp. 18-21.
23. D. E. Denning, "Restricting Queries That Might Lead to Compromise," *Proc. 1981 Symp. Security and Privacy,* Apr. 1981, pp. 33-40.
24. D. Dobkin, A. K. Jones, and R. J. Lipton, "Secure Databases: Protection Against User Inference," *ACM Trans. Database Systems,* Vol. 4, No. 1, Mar. 1979, pp. 97-106.
25. G. I. Davida et al., "Data Base Security," *IEEE Trans. Software Engineering,* Vol. SE-4, No. 6, Nov. 1978, pp. 531-533.
26. R. A. DeMillo, D. Dobkin, and R. J. Lipton, "Even Databases That Lie Can Be Compromised," *IEEE Trans. Software Engineering,* Vol. SE-4, Vol. 1, Jan. 1978, pp. 73-75.
27. F. Y. Chin and G. Ozsoyoglu, "Auditing and Inference Control in Statistical Databases," *IEEE Trans. Software Engineering,* Vol. SE-8, No. 6, Nov. 1982, pp. 574-582.
28. B. Sundgren, "RAM—A Framework for a Statistical Production System," S/SYS-E02, Nat'l Central Bureau Statistics, Stockholm, Sweden, 1978.
29. J. Schlörer, "Confidentiality of Statistical Records: A Threat Monitoring Scheme for On Line Dialogue," *Methods Information Mediation,* Vol. 15, No. 1, 1976, pp. 36-42.
30. F. Y. Chin and G. Ozsoyoglu, "Security in Partitioned Dynamic Statistical Databases," *Proc. IEEE Compsac,* 1979, pp. 594-601.
31. F. Y. Chin and G. Ozsoyoglu, "Statistical Database Design," *ACM Trans. Database Systems,* Vol. 6, No. 1, Mar. 1981, pp. 113-139.
32. F. Y. Chin and G. Ozsoyoglu, "Update Handling Techniques in Statistical Databases," *Proc. First LBL Workshop Statistical Database Management,* Lawrence Berkeley Laboratory, Dec. 1981.

Use order form on p. 144A

33. C. T. Yu and F. Y. Chin, "A Study on the Protection of Statistical Databases," *Proc. ACM Sigmod Int'l Conf. Management of Data,* 1977, pp. 169-181.

34. J. Schlörer, *Information Loss in Partitioned Statistical Databases,* Klinische Dokumentation, Universitat Ulm, W. Germany, 1983.

35. L. L. Beck, "A Security Mechanism for Statistical Databases," *ACM Trans. Database Systems,* Vol. 5, No. 3, Sept. 1980, pp. 316-338.

36. J. Schlörer, *Query Based Output Perturbations to Protect Statistical Databases,* Klinische Dokumentation, Universitat Ulm, W. Germany, Oct. 1982.

37. J. O. Achugbue and F. Y. Chin, "The Effectiveness of Output Modification by Rounding for Protection of Statistical Databases," *Infor,* Vol. 17, No. 3, Mar. 1979, pp. 209-218.

38. M. S. Nargundkar and W. Saveland, "Random Rounding to Prevent Statistical Disclosure," *Proc. American Statistical Assoc.,* 1972, pp. 382-385.

39. V. Alagar, *Complexity of Compromising Statistical Databases,* Dept. of Computer Science, Concordia University, Montreal, Canada, Nov. 1980.

40. J. Schlörer, *Security of Statistical Databases: Ranges and Trackers,* Klinische Dokumentation, Universitat Ulm, W. Germany, Nov. 1981.

41. I. P. Fellegi, "Controlled Random Rounding," *Survey Methodology,* Vol. 1, No. 2, 1975, pp. 123-133.

42. L. H. Cox and L. R. Ernst, *Controlled Rounding,* US Bureau of the Census, Washington, D.C., Jan. 1981.

43. T. Dalenius, "A Simple Procedure for Controlled Rounding," *Statistisk Tidskrift,* Vol. 3, 1981, pp. 202-208.

Dorothy E. Denning is a computer scientist at SRI International. Her primary research interests are cryptography and data security. She has been an associate professor of computer sciences at Purdue University, an assistant research mathematician at the University of Michigan Radio Astronomy Observatory, and a systems programmer and instructor at the University of Rochester. Denning is chairman of the ACM Special Interest group on Operating Systems. She is on the editorial board of *ACM Transactions on Computer Systems* and the advisory board *Computer Security Journal.* She has served as an IEEE distinguished visitor and is the author of *Cryptography and Data Security,* Addison-Wesley, 1982.

Denning received a BA and MA in mathematics from the University of Michigan and a PhD in computer science from Purdue University.

Jan Schlörer is with the Klinische Dokumentation of the University of Ulm, which functions as a department for medical computing, statistics, and documentation. His research interests include statistical database security, especially as it relates to medical databases. Schlörer has also worked at the Insitut fur Medizinische Informatik at the University Gieben.

Schlorer received an MD from the University of Marburg, West Germany. He entered medical computing and statistics in 1973 and has since studied mathematics and computer science at the University of Ulm.