Check for updates

Statistical Database Design

FRANCIS Y. CHIN and GULTEKIN OZSOYOGLU
University of Alberta

The security problem of a statistical database is to limit the use of the database so that no sequence of statistical queries is sufficient to deduce confidential or private information. In this paper it is suggested that the problem be investigated at the conceptual data model level. The design of a statistical database should utilize a statistical security management facility to enforce the security constraints at the conceptual model level. Information revealed to users is well defined in the sense that it can at most be reduced to nondecomposable information involving a group of individuals. In addition, the design also takes into consideration means of storing the query information for auditing purposes, changes in the database, users' knowledge, and some security measures.

Key Words and Phrases: security, compromisability, database design, protection, statistical database, conceptual database model
CR Categories: 3.72, 4.33

1. INTRODUCTION

The problem of enhancing the security of statistical databases (SDB) has been of growing concern in recent years. The security problem for a statistical database is to limit its use so that only statistical information is available and no sequence of queries is sufficient to deduce private or confidential information about any individual. When such information is obtained, the database is said to be *compromised* (or disclosure has occurred).

Several theoretical and practical studies of the security of statistical databases have been reported [10, 12, 13, 23, 27, 28] using different models and different statistical information such as COUNT, MAX, MIN, MEAN, MEDIAN, or SUM. In this paper we discuss some of the problems with these studies.

Problem 1. Statistical databases provide statistical information about groups of individuals in the real world. The assumption is that statistical information about a group of individuals conveys a meaningful aspect of that group of individuals. However, statistical information about an arbitrarily chosen group of individuals may not have a useful meaning attached to it. Some studies have set

Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the ACM copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Association for Computing Machinery. To copy otherwise, or to republish, requires a fee and/or specific permission.
This research was performed at the University of Alberta, Canada, and supported by a Canadian Natural Sciences and Engineering Research Council Grant.
Authors' present addresses: F. Y. Chin, Department of Electrical Engineering and Computer Sciences, University of California at San Diego, La Jolla, CA 92093; G. Ozsoyoglu, Department of Computer and Information Science, Cleveland State University, Cleveland, OH 44115.
© 1981 ACM 0362-5915/81/0300-0113 $00.75

forth questions (and given answers) with assumptions like "every possible combination of records can be requested" or "all possible medians of any sets of records are queriable" [16, 17, 27]. These types of assumptions quite naturally cause an explosion in the complexity of the problem. However, once a proper definition of the "statistical information" is used, and an analysis of the portion of the real world represented by an SDB is made for determining its statistical information, these combinatorially explosive possibilities usually can be reduced or even eliminated.

Problem 2. Although previous researchers have used the term "statistical database," they in fact meant "statistical files" and worked with records, record fields, etc. [4, 16, 31]. Databases are more than collections of records, and the information in databases may be highly complex. Databases contain a model of some portion of the real world; the security problem must be treated at that level. In all previous studies the SDB models used were closer to the physical than to the conceptual level of the database. Thus they encountered the problem of security at a very low level, that of physical records. Although these studies have contributed to our understanding of the problem, their SDB models were incomplete, and the results were usually negative in tone.

Problem 3. All previous studies (except [5, 35]) considered static databases in order to simplify the problem. The problem of SDB security should also be investigated for dynamic databases to capture the dynamics of the real world.

Problem 4. In the real world, users may be equipped with information other than what is explicit in records. If the database administrator (DBA) is aware of this information, effective security measures can be imposed easily. Example 1 illustrates this situation.

Example 1. Consider a database of employees of a certain computer manufacturing company in which the sum of salaries of employees is queriable. Assume the following information (which is not represented in the database and hence unknown to the database system) exists.

(a) Salary range of a new systems analyst with BS is \$[10K, 12K].
(b) Salary range of a new systems analyst with MS is \$[12K, 14K].

Now assume two new systems analysts are hired and information about them is inserted into the database. If the change in the sum of salaries of systems analysts is \$27K, then users can conclude that the new employees have MS degrees.

Most problems in SDB security can be removed by a good model of the real-world environment so that the DBA can take effective measures. Thus existing relationships and semantics of the information should always be considered for an effective SDB design.

Problem 5. Some database users may have access to some private or confidential information. When this happens, some mechanisms are needed for the DBA to decide (a) what other information has been disclosed by users, and (b) what protective measures should be taken. In other words, exact information revealed to users should be kept for auditing purposes. Some previous studies proposed

investigation of log trails for auditing [17, 21, 22]. However, for very large databases, the enormous amount of information in log trails is of little help for checking security (not to mention the "masking" of queries by users [15, 29]).

Let us now investigate previously suggested protection policies. In general, these protection policies impose restrictions on the database system. A "good" protection scheme should be *effective* (it should provide security to a reasonable extent), *feasible* (there should exist a way to enforce restrictions), and *efficient* and at the same time maintain the *richness* of the information revealed to users of the database [6].

(a) *Suppressing queries with very large and very small counts.* This policy does not mention the statistical information (Problem 1) and is shown to be ineffective [15, 30].
(b) *Limiting excessive overlap between queries* [16, 21, 27]. The same comment as (a) applies. Moreover, this policy is highly expensive (if feasible at all), and there is no systematic form of assurance that it guarantees security. Another drawback is that before the user gets to the necessary statistical information, he may be cut off from further access. In other words, the richness of the SDB is dependent on the users' queries.
(c) *Perturbing outputs, changing data records slightly, and replacing the database by a randomly chosen subset.* Although none of these policies really deal with Problems 1–4, they are mechanisms which make the job of the intruder harder. Perturbing outputs is proposed for manual off-line protection; the other two policies are not really investigated but briefly described in [1, 18, 20]. When properly tailored, these mechanisms may be effective. Recently, *random sampling* of records covered by the statistical query was proposed [14]. However, the proposed policy does not consider Problems 3–5.
(d) *Partitioning the records in the database* [5, 35]. This policy suffers from Problems 1, 2, and 4, and thereby has limited richness and effectiveness.

2. STATISTICAL DATABASE DESIGN

This section discusses the design of an SDB which employs several security constraints at the conceptual data model level. We list the desirable features of the SDB design in terms of the "goodness" criteria introduced in Section 1.

(1) *Effectiveness* of the protection. In order for the SDB system to be effective, the database should be equipped with the following information.
    (a) *A "good" conceptual model.* As a response to Problem 2 in Section 1, the SDB security should be elevated to the conceptual model level.
    (b) *Well-defined statistical information.* Statistical information must be well defined, and an analysis of the specific information and its statistical constituents should be made. This will help to reduce the size of the security problem (crystallize the complex relationships, define the information to be secured, etc.) and thus eliminate Problem 1 mentioned previously.
    For the real-world model, the statistical information revealed to users

will be only about predefined groups of individuals. The intersection of these groups of individuals will give a set of indivisible groups of individuals, and any statistical information about these indivisible groups of individuals will constitute *atomic information.* Thus we are no longer interested in giving out uncontrolled, random statistical information to users, which may easily be exploited, but rather well-defined information that can at most be reduced to atomic information.

(c) *Controlled changes in the database.* Dynamics, as well as statics, of the real world should be revealed to users (Problem 3). However, this should be done in a controlled manner, and the information revealed due to the changes in the environment should be recorded for auditing. (Note that the 1974 U.S. Privacy Act [34] necessitates the inclusion of changing aspects of the environment.)

(d) *Information about users' additional knowledge.* Users' additional knowledge should be maintained and kept up-to-date in the SDB. We assume that the DBA is correctly informed about users' additional knowledge of protected information.

(2) *Efficiency* of the protection. We now describe features of the SDB to improve the efficiency of the protection.

(a) Disjoint user groups are defined to utilize the fact that their initial knowledge may be substantially different from each other or that they may not necessarily have the same access authorization to different parts of the database.

(b) Different levels of statistical information should be revealed to different users. For example, some users may not be allowed to access certain detailed statistical information.

(c) For each group of individuals about which statistical information is to be revealed, allowable statistical query types are defined. This leads to different security constructs and mechanisms for different types of statistical information.

(3) *Richness* of the information revealed to users. Clearly, investigating the security problem at the conceptual model level provides the database designers with more control over the richness and usefulness of the SDB. However, atomic information should not be further decomposable by templates or by queries such as join, select, and project operators in a relational model [7, 8]. We also assume that the DBA should confirm the security and compatibility of any new view before granting access to it.

2.1 Constituents of Statistical Information

Statistics studies specific aspects of individuals in a population which may be conceptual or physical. The individuals in the population have *something in common* so that they altogether form the population. Most statistical methods can be viewed as ways of making inferences about a population. Such inferences are made after the examination of a "sample" from the population. A database may contain the whole population or a sample of the population. The user may or may not use the statistical information for statistical inferences. In any case

the central concept is the population concept. For the specific environment at hand, once the populations to be studied are found, then the individuals are no longer important, and two individuals with nothing in common will never be included in the answer of the same statistical query.

We should also differentiate the quantitative properties of individuals for which statistical information is to be revealed and the defining characteristics of a population. For example, "sum of salaries of employees" is a quantity related with the employee population, but "sum of salaries of employees where salary >$12K" gives information about a different population. Not distinguishing this difference may cause protection problems. Similarly, for example, "number of employees" and "number of employees convicted of felony" give information about two different populations.

2.2 Conceptual Data Model for SDB Design

Although we propose that only aggregate information be revealed to users, this does not imply that there is only aggregate information in the database. On the contrary, the conceptual model of the SDB should be similar to the conceptual model of any other general-purpose database. There are several reasons for this requirement besides effectiveness of the protection and the richness of the SDB.

(1) For some users and at least for the DBA, the SDB is just a normal database, and these users should have access to all information in the database (not just aggregates).
(2) It is necessary to have total information about the environment in order to enforce integrity and validity in the database.

Thus the design of a secure SDB should be investigated with a conceptual data model. With the recently renewed interest in conceptual data models, over 30 different data models are mentioned in [24, 25]. From the security viewpoint, our concern is twofold. We are concerned about the structure of the conceptual model in order to define atomic information and to give out controlled statistical information. We are also concerned about the semantics of the conceptual model in order to successfully mirror the real-world environment so that (a) the security measures can easily and naturally be provided and (b) the database is still a highly rich and useful one for users. Thus we require a structured, semantic, and redundant conceptual model for the SDB. In this paper, the *data abstraction model* (D-A model) [32, 33] is used in the design of the SDB. The choice of the D-A model from among other structured, semantic, and redundant data models is motivated by the ease in applying protection measures without bringing many extra constructs and restrictions to the conceptual model. However, the SDB design may easily be modified for any other structured, redundant, and semantic data model, and Appendix A discusses two other data models, namely, the entity-relationship model [3] and the extended relational model [9], for their suitability as a conceptual model of the SDB. In this section, we summarize and modify the D-A model for SDB design. Our ultimate goals are to augment the conceptual data model with the population concept, to identify atomic information, and finally to propose a statistical security management facility.

Smith and Smith [32] introduce two kinds of database abstractions. *Aggrega-*

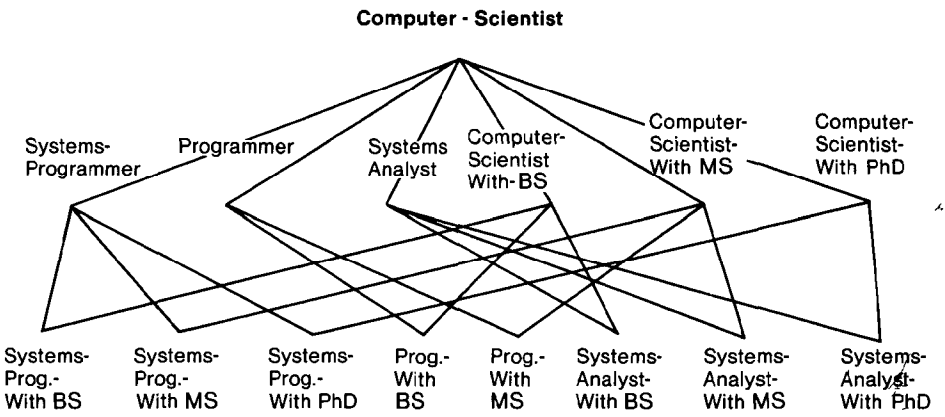

Fig. 1. Decomposition of the generic object Computer Scientist.

tion (naming relationships) is an abstraction which turns a relationship between objects into an aggregate object. *Generalization* (naming classes) is an abstraction which turns a class of objects into a generic object. All objects (individual, aggregate, generic) are given uniform treatment in the D-A model. The real world is modeled as a set of aggregation hierarchies intersecting with a set of generalization hierarchies. Abstract objects (i.e., generic and aggregate objects) occur only at the points of intersection. In the context of the relational model [7, 8], the D-A model is proposed as a conceptual model. Our aim is to modify the generalization hierarchy in such a way that all populations are identified in a systematic manner and a generic object in the hierarchy consists of a (group of) population(s). No modifications are proposed for the aggregation hierarchy.

Consider the same example of employees of a certain computer manufacturing company as used in Section 1. Figure 1 illustrates one particular decomposition of Computer Scientist into lower level generic objects. Notice that there are two mutually exclusive groups of partitions (also called clusters) of Computer Scientist; one group is {Programmer, Systems Programmer, Systems Analyst} and the other is based on the degree obtained. Now assume that we also have the "country in which PhD was obtained" information about Computer Scientists. Clearly, one may ask about the "population of U.S.-educated systems analysts with PhD." In [33] this information is kept as an attribute of objects in the generalization hierarchy, and there is no provision for further partitioning. The reason for this is that each abstract object is required to be explicitly *named using natural-language nouns* (e.g., Programmer, Systems Analyst), and these names help us to relate our understanding of the real world with its intended reflection in the relation definition. However, the generic object "U.S.-educated systems analyst with PhD" is certainly described by a phrase, not by a natural-language noun, and yet we are interested in this particular object and it has to exist in the hierarchy. Thus for statistical database design purposes we take more freedom at this point and use phrases to describe populations. Figure 2 contains the partitioning of Systems Programmer with PhD and Systems Analyst with PhD according to the attribute "country in which PhD is obtained."

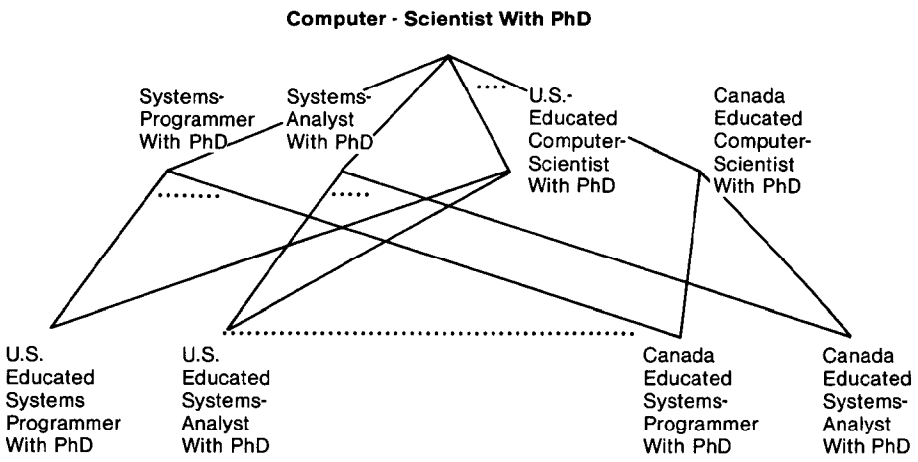

Fig. 2. Decomposition of the generic object Computer Scientist with PhD.

Now assume that we also have the "years of programming experience" information for Programmers. Populations using this information may be formed, such as "programmers with 5 years of programming experience" or "programmers with MS and 2 months of programming experience." At this stage a design decision problem appears. If the "years of experience in the company" information uniquely identifies many individuals by creating large numbers of populations with single individuals, the security is endangered. We assume that statistical database designers make the decomposition decisions using their knowledge of users' needs, i.e., if there are very many populations each with few individuals, then the designers will cut down the number of populations and still preserve a good model of the real world. However, this does not mean that initial design decisions cannot be changed; indeed, if a need arises, some mechanisms will be available to the DBA so that, with an assessment of the security of protected information, the decomposition of objects may be changed some time later. Figure 3 shows one particular design decision about the usage of "years of programming experience" information for decomposing the object Programmer.

In the SDB, statistical information about individuals in a population is made available to users. Clearly, each abstract object in the D-A model forms a population of individual objects. We call the smallest nondecomposable group of individuals an *atomic population* (A-population). For example, in Figure 3, SYSTEMS-PROGRAMMER-WITH-BS and CANADA-EDUCATED-SYSTEMS-PROGRAMMER-WITH-PhD are A-populations. In order to preserve the indivisibility property of A-populations, the following rule is applied.

Rule 1. Any population corresponding to any abstract object in the model is composed of mutually exclusive A-populations that explicitly exist in the model.

The restriction that A-populations explicitly exist in the conceptual model may bring limitations to the richness of the SDB. However, this restriction is needed to provide systematic assurance of the security of protected information in the SDB.

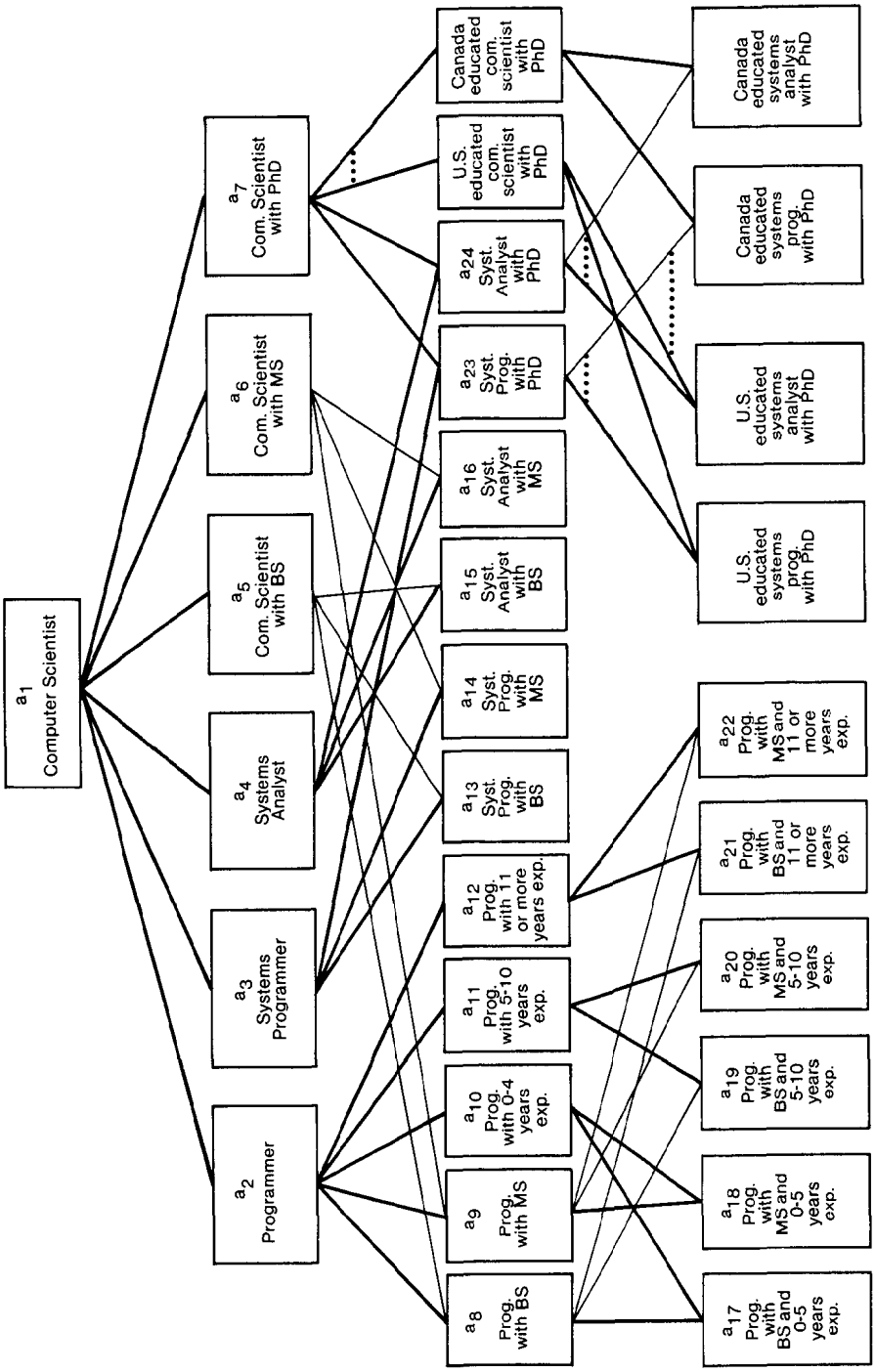

Fig. 3. Generalization hierarchy for Computer Scientist.

2.3 Statistical Information Related to Each Population

For each population we should define the following:

(a) the properties of the population[1] for which statistical information is to be revealed, e.g., SALARY or ABSENT-DAYS for the population EMPLOYEE;
(b) whether COUNT queries requesting the number of individuals in the population are permitted; and
(c) the allowable types of statistical information for each property of the population which may be one or more of MEAN, SUM, MAX, MIN, MEDIAN, K-LARGEST (order statistics), VARIANCE, STANDARD DEVIATION, k-MOMENT, $k = 2, 3, \ldots$.

Clearly, if, in Figure 3, SUM query for the SALARY property of PROGRAMMER and SYSTEMS-PROGRAMMER is allowed, then SUM information for the SALARY of SYSTEMS-ANALYST is deducible. Thus, unless individual security needs of populations require otherwise, we find the following two rules necessary for the uniformity of the revealed statistical information and richness of the database.

Rule 2. The allowable set of statistical query types should be identical for the same property of all populations in the same cluster. (Subpopulations created by a mutually exclusive decomposition of a population in the generalization hierarchy form a cluster.)

Consider Figure 3. Assume that SALARY is an attribute of all the objects in the hierarchy and SUM query is allowed for SALARY of Programmer. SUM query should also be allowed for SALARY of Systems Programmer and Systems Analyst.

Rule 3. The allowable set of statistical query types for a property of any population should be the subset of the allowable set of query types for the same property (if it exists) of its father population in the generalization hierarchy.

Consider Figure 3. Assume the statistical query SUM of SALARY is allowed in populations Programmer, Systems Programmer, and Systems Analyst and statistical query MEDIAN of SALARY is allowed in populations Computer Scientist with BS, MS, and PhD. Statistical queries SUM and MEDIAN are allowed for the population of Computer Scientist.

Since COUNT queries do not directly reveal information about protected properties of populations, applying protection measures down to A-populations may unnecessarily restrict the richness of the SDB. Thus a *security atom population* (SA-population) is defined to be the largest population such that no statistical information about any property of any of its proper subsets can be revealed to users. Notice that an SA-population contains one or more A-populations. The set of values to be protected for each property in an SA-population is called a *security atom value set* (SA-value set). The following example illustrates SA-populations.

---

[1] By "the property of a population" we mean "the property of individuals in the population."

Example 2. Consider Figure 3. Assume there are *only* two protected properties, SALARY and ABSENT-DAYS, and

(a) for all populations, COUNT query is allowed.
(b) SUM query for SALARY and ABSENT-DAYS is allowed for populations a_1, a_2, a_3, and a_4. MEDIAN query for SALARY is allowed for populations a_1, a_5, a_6, and a_7. Also, SUM query for ABSENT-DAYS is allowed for populations a_{10}, a_{11}, and a_{12}. Clearly, populations a_8, a_9, a_{13}, a_{14}, a_{15}, a_{16}, a_{23}, and a_{24} contain nondecomposable SALARY information revealed to users. Similarly, populations a_{10}, a_{11}, a_{12}, a_3, and a_4 contain nondecomposable ABSENT-DAYS information revealed to users. The intersections of these populations will give SA-populations a_{17}, a_{18}, a_{19}, a_{20}, a_{21}, a_{22}, a_{13}, a_{14}, a_{15}, a_{16}, a_{23}, and a_{24}. Notice that an SA-population may contain one or more A-populations.

2.4 Security Constraints

Dynamics of the real world or the existence of complex relationships between populations may lead the DBA to impose constraints on the security-related information in populations. The DBA should be able to state the conditions under which any statistical query about any protected property of a population may be reported to users. Since our aim is only to provide the DBA with the power to do so, we will in general distinguish three types of constraints. (Defining these constraints is very much dependent on the specific environment, and we are unable to give more detailed analysis and structural specifications of the constraints as done by [19] for semantic integrity constraints.)

(1) *Security atom constraints* (SA-constraints) apply to the SA-value set in an SA-population A and all populations that contain A. An example is: sum salary information must not include the salary x of employee a in SA-value set w until there is another employee hired or fired.
(2) *Global constraints* (type 1) apply to the individuals in a population A and individuals of all or some of the populations in the hierarchy that contain A. Consider Figure 3 and Example 1 given in the introduction. Assume user group u is allowed to access down to Systems Analyst in the hierarchy. Now the hiring of two new Systems Analysts with MS and with total salary \$27K should not be incorporated into the population Systems Analyst. However, if the range of salaries of Computer Scientists with MS include \$14K due to its other child populations, then the new change may be incorporated into the populations Computer Scientist with MS and Computer Scientist (if other constraints are also satisfied).
(3) *Global constraints* (type 2) apply to the individuals of a population A and to individuals of another population B in a different part of the hierarchy.

Example 3. Consider the database of employees, projects, and assignments in Figure 4. It is known that at least 10 programmers and 1 systems analyst with more than 10 years working experience are involved in the database development project. Assume we also know that a project leader must be a systems analyst with a PhD. Now if COUNT queries of SYSTEMS-ANALYST-WITH-MORE-THAN-10-YEARS-EXPERIENCE and ASSIGNMENT-IN-DATA-BASE-PROJECT return 1 and 11, respectively, and if we know a systems analyst

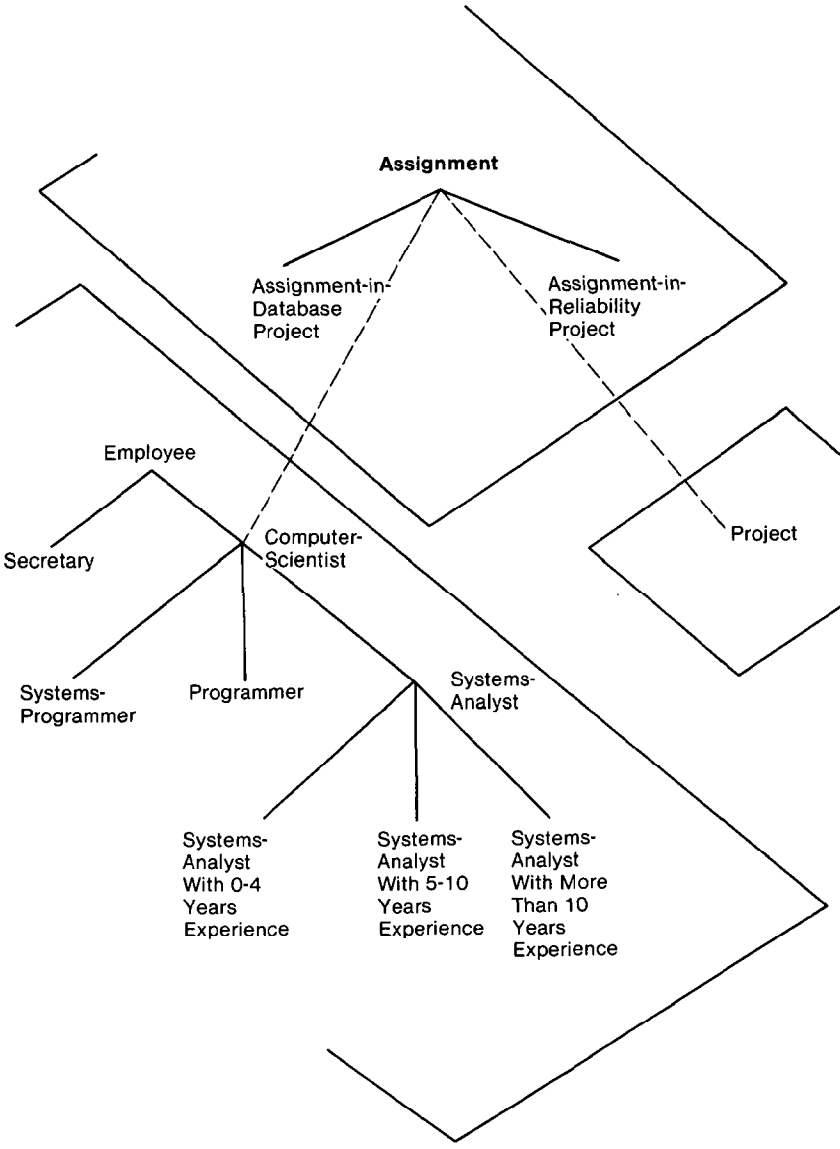

Fig. 4. Database of employees, projects, and assignments.

with more than 10 years experience, then we disclose that he is the project leader of the database development project and also has a PhD. To prevent this disclosure, type 2 global constraints applied to SYSTEMS-ANALYST-WITH-MORE-THAN-10-YEARS-EXPERIENCE and ASSIGNMENT-IN-DATA-BASE-PROJECT may state that if COUNT information of these two populations are smaller than a_1 and $10 + a_2$, respectively, where a_1 and a_2 are properly chosen small integer constants, then COUNT information of both of the populations are not revealed to users.

Example 4. Consider Figure 4. Assume two programmers are hired. Now if COUNT information of both PROGRAMMER and ASSIGNMENT-IN-DATA-BASE-PROJECT increase by two then the new programmers are assigned to the database development project. If this information is to be protected, then type 2 global constraints applied to child populations of ASSIGNMENT may state that new assignments in child populations of ASSIGNMENT are reported only when there are new assignments in two or more projects.

3. A STATISTICAL SECURITY MANAGEMENT FACILITY

We now propose a statistical security management facility (SSMF) with three principal components.

(1) A population definition construct (PDC).
(2) A user knowledge construct (UKC).
(3) A constraint enforcer and checker (CEC).

The PDC of a population contains information about the population, related constraints, changes of population, etc., in order to achieve effective protection. The UKC of a user group is designed to record users' additional knowledge and SA-constraints. Finally, the CEC consists of several algorithms designed to keep the PDCs and UKCs up-to-date, to enforce the security constraints, and to help the DBA in security-related decision problems.

3.1 Population Definition Construct

For each population P, there is one PDC which contains the following information:

(a) description of the population and its parent, child, and sibling populations;
(b) lowest permissible user group level;
(c) information as to how changes are included in P;
(d) allowable statistical query types for each property of P;
(e) global constraints of P;
(f) if P is an SA-population, then description of SA-constraints for each SA-value set of P.

(a), (d), and (f) are self-explanatory.

(b) Assume user groups are classified by numbers such that user groups with higher numbers have more access power to the database than the user groups with lower numbers. The lowest permissible user group level is a number n such that user groups with number $m \geq n$ can access that population.

(c) Changes due to the dynamics of the real world may be processed in many ways. How these changes are handled is described in (c) and (f).

(e) Global constraints may be static or dynamic. They may evolve and change as the DBA modifies them, for example, a manager changes companies and thus extends his knowledge, and the DBA should take necessary action. For each global constraint, the PDC contains the description of the constraint and a call for a routine in the case of violation of the constraint. Figure 5 contains the PDC of Programmer in the generic hierarchy described in Figure 3.

```
Population PROGRAMMER
     description [Phrase],
     parent populations [COMPUTER-SCIENTIST],
     child populations [(PROGRAMMER-WITH-BS, PROGRAMMER-WITH-MS),
               (PROGRAMMER-WITH-0-4-YEARS-EXPERIENCE, PROGRAMMER-
               WITH-5-10-YEARS-EXPERIENCE, PROGRAMMER-WITH-11-OR-
               MORE-YEARS-EXPERIENCE)],-
     other populations in the same cluster [SYSTEMS-PROGRAMMER,
               SYSTEMS-ANALYST],
     lowest permissible user group level 2,
     allowable query COUNT,
     changes processed in PAIRS,
     protected property SALARY,
         allowable query SUM,
     protected property ABSENT-DAYS,
         allowable query MEDIAN,
     global constraints
         constraint 1
               description [Phrase],
               call VIOL-CS1,
         constraint 2
               description [Phrase],
               call VIOL-CS2,
end.
```

Fig. 5. The PDC of Programmer.

3.2 User Knowledge Construct

For each user group u, the UKC records the users' additional knowledge about individuals in the SDB. Figure 6 contains the UKC of user group u for the generic hierarchy described in Figure 3.

Assume user group u is at the third level, which can access all populations in the hierarchy.

Users in user group u can identify the individuals that are updated, inserted, or deleted from the population PROGRAMMER (e.g., the newly inserted, deleted, or updated programmer in the population PROGRAMMER is known by the user group u). For each population, this information is defined after the keyword "identifiable dynamics" in the UKC. Clearly, protection measures to be applied should be different for a user group which identifies only inserted individuals of a population and a user group which identifies both inserted and deleted individuals of the same population. (There may be other variations; for example, users in user group w may identify updated individuals when the update is from Systems Programmer to Systems Programmer, etc.)

Each SA-population contains one SA-value set for each of its properties. Dynamics of an SA-population (i.e., inserted, deleted, updated individuals) are recorded in a list called the *change sequence* in the order of occurrences of changes. (This list may be kept separately if the expected number of changes is large.) Depending on the type of statistical information revealed, the change sequence is used in several procedures to decide whether the security of individuals and the protected information are in danger.

For security purposes, changes may be processed in groups, say triplets. In such cases some individuals may be waiting to be processed; these individuals are

```
USER GROUP U [user-id, user-id,...,user-id],
     user group level 3;
     population COMPUTER-SCIENTIST,
         identifiable dynamics INSERTION, DELETION, UPDATE,
     population PROGRAMMER,
         identifiable dynamics INSERTION, DELETION, UPDATE,
     population PROGRAMMER-WITH-0-4-YEARS-EXPERIENCE
                        [SA-POPULATION],
        identifiable dynamics INSERTION, DELETION, UPDATE,
        protected property SALARY,
             security atom constraint: {JOHN DOE} is not
                    included,
             change sequence parameters
                active individuals set {(STEVE HART,
                     ROCK HO,20),...,(JOHN GRAY,,3)},
                reachability constant 0.1,
                largest reachability set size 20,
             known value set {(JOHN SO),...,(ALAN POE)},
             known global upper bound $34K,
             known global lower bound $8K,
             known upper bounds set {(IAN MUNROE,$18K),...,
                     (GEORGE HO,$20K)},
             known lower bounds set {Ø},
             change sequence {[(JIM JOE,INSERT),
                     (JACK YU,DELETE),(OLD MEDIAN,$15K),
                     (NEW MEDIAN,$14K)],...,
                     [(PHILIP HO,DELETE),(JACK FU,DELETE),
                     (NEW MEDIAN,$18K)]},
        protected property ABSENT-DAYS,
             security atom constraint: {STEVE HUDSON} is not
                     included,
             change sequence parameters
                active individuals set {(STEVE HART,,15),...
                     ,(JOHN GRAY,,7)},
                reachability constant 0.2,
                largest reachable set size 15,
             known value set {0},
             known global upper bound 90,
             known global lower bound 0,
             known upper bounds set {Ø},
             known lower bounds set {Ø},
             change sequence {[(JIM JOE,INSERT)],
                   [(JACK YU,DELETE),(CHEN TU,DELETE),
                   (OLD SUM,450),(NEW SUM,345)],...},
     population PROGRAMMER-WITH-5-10-YEARS-EXPERIENCE
                        [SA-POPULATION],

end.
```

Fig. 6. The UKC of user group u for the generic hierarchy described in Figure 3.

described and maintained in SA-constraints. For each SA-value set, users may know global upper or lower bounds of the property values of individuals, and upper or lower bounds for some specific individuals. For example, in Figure 6, users in user group u know that the salary of programmer Ian Munroe is less than $18K. Several parameters related to the change sequence are defined and described in Section 4.2.

3.3 Constraint Enforcer and Checker

The CEC is composed of several algorithms. It utilizes PDCs and UKCs to perform the following two basic tasks:

(a) For each statistical query, it is invoked to find the global and SA-constraints by tracing the related PDC and the UKC, and to enforce these constraints by executing the related procedures (thus altering the answer to the user's statistical query, if necessary).
(b) For each change (i.e., insertion, deletion, or update of individuals) in the populations of the D-A model, it is invoked to modify the constraints, to decide whether to process (i.e., to include into users' statistical queries) the change for each SA-value set, and (if the change is processed) to modify the related change sequence and its parameters for each user group u.

In addition to the above, the CEC helps the DBA in several security-related decision problems by providing lists of individuals whose security is threatened under events described below.

(1) *Changes in user groups.* User groups may join or decompose, or users may move from one user group to another. In these cases additional knowledge of a user group may increase, and further disclosed information is then decided by the CEC using the UKC of the user group and change sequences.
(2) *Changes in the conceptual model* such as decomposing a population or repartitioning a population. In these cases the CEC finds the SA- and A-populations, rearranges UKCs, modifies security measures, and reports disclosures.
(3) *Changes in users' additional knowledge* such as a modified known value set or an updated known upper bounds set. In these cases further disclosures are decided by tracing the change sequences and considering possible inferences discussed in Section 4.

Another job of the CEC is to modify and maintain several security measures related to the change of sequence of each SA-value set in order to give the DBA a measure of how secure the system really is at a particular time. Some of these measures are discussed in Section 4.2.2.

The general scheme of the SSMF is depicted in Figure 7. The CEC utilizes the conceptual model, PDCs, and UKCs to enforce security constraints and modify statistical queries. Individual insertions, deletions, and updates into populations in the conceptual model are intercepted, and modifications of security constraints and change sequences and their parameters are carried out by the CEC. In the case of disclosures, the DBA is notified, and security-related reports such as the values of security measures, the number of constraints in effect, the number of individuals not included in statistical queries, and the introduced error are reported by the CEC.

4. PROTECTION REQUIREMENTS FOR DIFFERENT STATISTICAL QUERIES

In this section we investigate the possible security constraints for different statistical queries. First, inferences available to users are identified, and then related security constraints to enforce security are briefly described. We distinguish three different inferences by users.

(a) *Type S inferences* due to the hierarchical structure of the conceptual model.

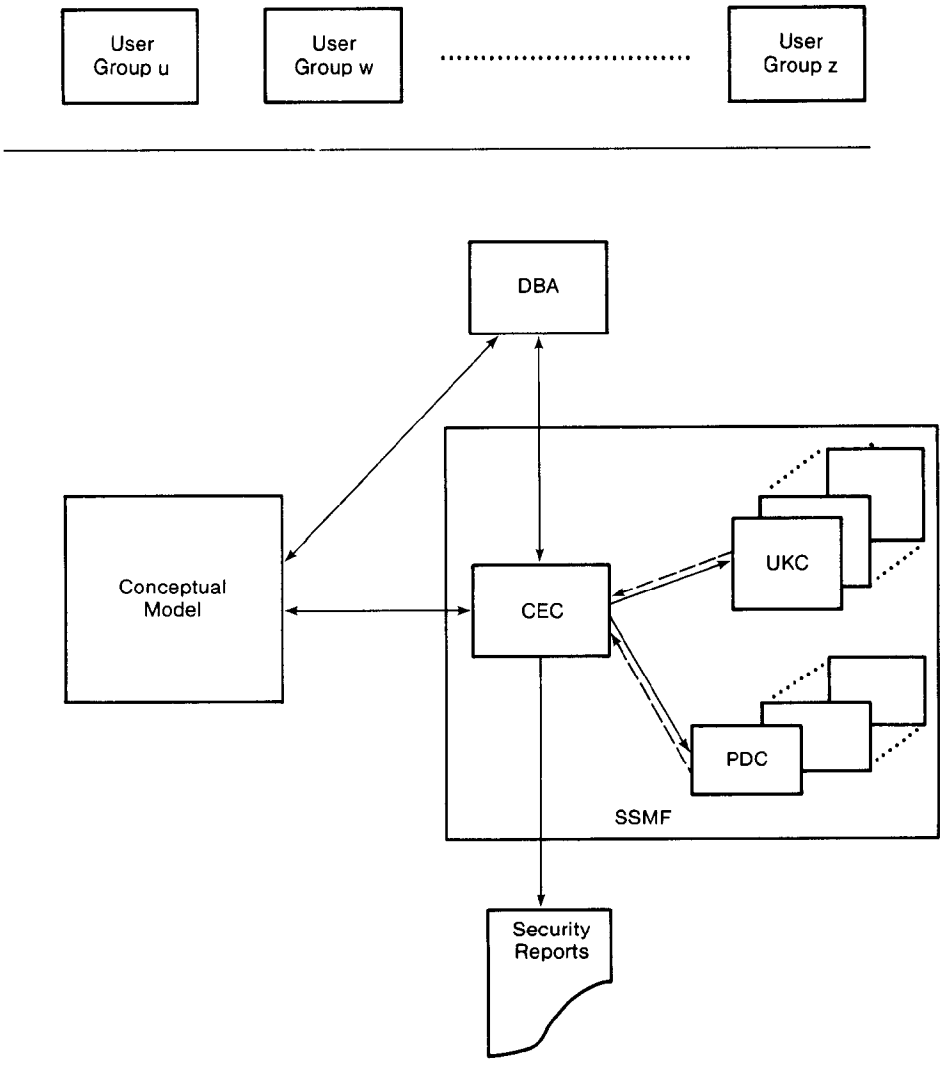

Fig. 7. Statistical database model.

(b) *Type D inferences* due to the dynamics of the real world.
(c) *Type R inferences* due to existing relationships between individuals in different populations or in the same population.

Disclosures in Examples 1, 3, and 4 are due to type R inferences. Type R inferences are dependent on the specific environment. In this paper we assume that global constraints are defined by the DBA to prevent disclosures due to type R inferences. That is, the DBA is responsible for identifying type R inferences and applying protection measures to prevent compromise. This decision is motivated by the simplicity and efficiency of SDB design. Another approach may be to define formally type R inferences and use a theorem prover to decide about the inferred knowledge and the disclosed information. In [26], this approach

which uses a question-answering system to enhance the security of the SDB, is outlined.

In what follows we consider only type S and type D inferences. The following are suggested schemes for a sample of different types of statistical queries; the others can be derived similarly.

4.1 Count Queries

Assume only COUNT queries are allowed and individuals in populations are identifiable. Assume Systems Analyst with BS is decomposed into two subpopulations as "Systems Analyst with BS and convicted of felony" and "Systems Analyst with BS and not convicted of felony." It is well known [22] that

$$\left[\begin{array}{l}\text{COUNT(Systems Analyst with BS)} = \text{COUNT(Systems Analyst with BS and convicted of felony}\\ \text{John Doe is a Systems Analyst with BS}\end{array}\right]$$

$\rightarrow$ John Doe is convicted of felony.[2]

Similarly,

$$\left[\begin{array}{l}\text{COUNT(Systems Analyst with BS and not convicted of a felony)} = 0\\ \text{John Doe is a Systems Analyst with BS}\end{array}\right] \rightarrow \text{John Doe is convicted of felony.}$$

Thus for type S inferences we need the following global constraints.

(a) in population SYST-ANALYST-BS-CONV-FELONY,

if COUNT(any superpopulation of SYST-ANALYST-BS-CONV-FELONY) − COUNT(SYST-ANALYST-BS-CONV-FELONY) $\leq a$
then individuals creating above difference are not reported in COUNT queries.

(b) in population SYST-ANALYST-BS-NOT-CONV-FELONY,

if COUNT(SYST-ANALYST-BS-NOT-CONV-FELONY) $\leq a$
then COUNT(SYST-ANALYST-BS-NOT-CONV-FELONY) is not answered.

The above disclosure type and constraint have been proposed and discussed widely in recent studies [15, 28, 29]. If introduced in a controlled environment, they constitute a viable protection procedure.

The constraints described above can be easily modified to consider the users' knowledge described in the UKC by changing a to $(a + x)$ where x is the size of the user group's knowledge in the population.

Type D inferences may be similarly avoided; for example, only when there are $(a_1 + x)$ insertions or $(a_2 + x)$ deletions from either populations of SYST-ANALYST-BS-CONV-FELONY or SYST-ANALYST-BS-NOT-CONV-FELONY are changes reported to user group u, where insertion and deletion of SYST-ANALYST-BS are identifiable.

[2] $\rightarrow$ means "implies" or "imply."

4.2 Sum and Count Queries

Assume only SUM and COUNT queries are allowed for all populations (i.e., each SA-population contains only one A-population), and consider Figure 3. Assume users in user group u cannot identify if a systems programmer or a systems analyst is hired or fired, but they can always identify if a programmer is hired or fired. Now any changes in populations Systems Analyst or Systems Programmer can be reported to user group u immediately, but care must be taken in reporting changes in the population Programmer. In what follows we assume insertions and deletions into a population are identifiable and consider only type D inferences. Later we discuss type S inferences.

Assume a new programmer with salary x is hired. Querying the population Programmer immediately before and after the change reveals the salary x of the new programmer. On the other hand, if changes are processed in large batches, the error introduced in SUM queries may reduce the usefulness of the statistical information. (In general, changes in a partition can be processed in batches with size $t \geq 3$ for better security. This policy however introduces an error in statistical queries, which is dependent on the values of records with changes, and for large t it may reduce the usefulness of the statistical information.) Thus we assume that changes are processed in *pairs*.

We assume that the individuals deleted from a population are not normally reinserted into the same population, or if they are reinserted, they have independent protected property values. This assumption is realistic since every change is approved by the DBA and any continuous insertion and deletion of the same individual by a malicious user can be detected by the DBA.

If updates are not identifiable, then an update operation can be replaced by a pair of insertion and deletion operations. In what follows we assume that updates are not identifiable, and later we mention identifiable updates.

4.2.1 *The Information Graph.* Assume two individuals with protected property values x and y are both to be inserted into or both to be deleted from the population A. Querying A before and after the change, one can obtain the information $x + y = c_1$ where c_1 is a constant. Similarly, when one individual with property value u and another with property value v are deleted and inserted, respectively, into A, then the information $u - v = c_2$ is obtained, where c_2 is a constant. In [5], this information is characterized by an undirected labeled graph called the *information graph* such that vertices correspond to individuals and s- and d-edges represent equations with sums and differences, respectively.

Using the information graph, it is proved [5] that if no protected property value is known by users and if each population starts with an even number of individuals, the SDB is secure for COUNT and SUM queries. Clearly, this result should incorporate users' knowledge in order to be meaningful. If individuals with known (or suspected) property values are processed in pairs only with known individuals, then the SDB will still be secure, since unknown values will be separated from known values in the equations in pairs. Thus in order to prevent disclosures due to type D inferences, we have

(a) populations with an even number of individuals, and
(b) for each SA-value set in each UKC, there is an SA-constraint which delays

the processing of (known and/or unknown) individuals with recent changes until another change occurs (see SA-constraint in Figure 6).

Clearly, if each population has an even number of individuals, then the size difference between a population and any of its parent populations must be either zero or at least two. Thus type S inferences cannot cause any disclosure.

A population A may contain several SA-populations, and thus several individuals may be waiting for inclusion in statistical answers of A. To prevent that, when t unknown (known) individuals are waiting, they may be included in the statistical answers of A. This can be specified by global constraints in A and its parent populations. The bound t must be large enough so that the information revealed to users will be practically useless for all disclosure purposes.

However, the above-described model has the following restrictions: (a) there is an even number of individuals in each population; (b) changes in a population must wait for some time until there is another change; and (c) updates are assumed to be unidentifiable. In [5], incorporation of dummy individuals is suggested to remove these restrictions.

4.2.2 *Security Measures Related to the Information Graph.* The information graph has another usage. As more and more changes are made, the users get more and more information. Moreover, there is always a danger of being unable to assess what users know; and users' knowledge of one property value of an individual x is sufficient to disclose all other property values of individuals that are in the same connected component with x in the related information graph. Thus a measure of security may be defined in terms of the number of connected components of the information graph. The *reachability set* Rs is defined as a subset of the individuals in population A such that they are in the same connected component. We also define

$$\textit{reachability constant } w = \frac{\textit{number of reachability sets in the information graph}}{\textit{number of vertices in the information graph}}$$

Clearly $0 < w \leq 1$, and $w \cong 1$ implies relatively "more" security (more connected components in the information graph) and $w \cong 0$ implies relatively "less" security. Another security measure may be *largest reachable set size* z, i.e., $z = \max |\text{Rs}|$ for all reachability sets Rs. The two described measures of security, w and z, are not controllable in the model described in Section 4.2.1. However, there are ways that dummy individuals may be used to control and change the sizes of reachability sets and, thus, to control the security measures w and z [5].

Change sequence parameters (see Figure 6) are described as reachability constant w and largest reachable set size z. For each change sequence, an active individuals set is also maintained. (A vertex is *inactive* if it corresponds to an individual deleted from the population; it is called *active* if it corresponds to an individual previously inserted but not yet deleted from the population.) An active individuals set contains one set element for each connected component which has one or two active individuals. Each set element contains the names of the active individuals and the number of vertices in that connected component (i.e., the reachability set size). For example, in Figure 6, the information graph of the

population of Programmer with 0–4 years of experience contains a connected component with two active vertices for individuals Steve Hart and Rock Ho, and the number of vertices in that connected component is 20.

4.3 Median and Count Queries

Assume only MEDIAN and COUNT queries are allowable for all populations in the conceptual data model. Both type S and type D inferences may lead users to obtain upper or lower bounds for protected property values if insertions, deletions, or updates are identifiable. The following examples illustrate the possible inferences.

Example 5. (Type S Inference). Consider Figure 3. Assume individuals are identifiable and there is only one Programmer with MS whose salary is x. Thus queries about Programmer with MS are not permitted. However, the following information is obtainable.

$$\text{MEDIAN(PROGRAMMER-WITH-BS, SALARY)} = a$$
$$\text{MEDIAN(PROGRAMMER, SALARY)} = a_1$$
$$\text{COUNT(PROGRAMMER-WITH-BS)} = n$$
$$\text{COUNT(PROGRAMMER)} = n + 1.$$

Now

$$a_1 > a \rightarrow x > a \qquad \text{and} \qquad a_1 < a \rightarrow x < a.$$

Thus we have an inference about the salary x of the only programmer with MS.

Example 6. (Type D Inference). Consider Figure 3 and assume changes are identifiable.

$$\text{MEDIAN(SYSTEMS-ANALYST-WITH-BS, SALARY)} = a$$
$$\text{COUNT(SYSTEMS-ANALYST-WITH-BS)} = m.$$

Assume a new Systems Analyst with BS and salary x is hired:

$$\text{MEDIAN(SYSTEMS-ANALYST-WITH-BS, SALARY)} = a_1$$
$$\text{COUNT(SYSTEMS-ANALYST-WITH-BS)} = m + 1.$$

Now

$$a_1 > a \rightarrow x > a \qquad \text{and} \qquad a_1 < a \rightarrow x < a.$$

We would like to avoid the above inferences.

4.3.1 *Processing Changes in Pairs.* We first consider type D and then discuss type S inferences. Consider population A with protected property B, and assume all changes are processed in pairs.

$$\text{MEDIAN}(A, B) = a \qquad \text{COUNT}(A) = n.$$

Two individuals with property B values x and y are added to A; then

$$\text{MEDIAN}(A, B) = a_1 \qquad \text{COUNT}(A) = n + 2.$$

Now

(a) n is odd:

$$a_1 < a \rightarrow x, y \leq a_1$$
$$a_1 > a \rightarrow x, y \geq a_1.$$

(b) n is even:

$$a_1 < a \rightarrow \text{at least one of } x, y \leq a_1$$
$$a_1 > a \rightarrow \text{at least one of } x, y \geq a_1.$$

Clearly, (b) is better than (a) in the sense that it does not allow an upper or lower bound inference for any of the property values x, y.

Now consider

$$\begin{matrix} \text{MEDIAN}(A,B) = a \\ \\ \text{COUNT}(A) = n \end{matrix} \left\{ \begin{matrix} \text{an individual with property} \\ x \text{ is added to } A \\ \text{an individual with property} \\ y \text{ is deleted from } A \end{matrix} \right\} \begin{matrix} \text{MEDIAN}(A,B) = a_1 \\ \\ \text{COUNT}(A) = n \end{matrix}$$

and we have the following inferences.

(a) n is odd:

$$a_1 < a \rightarrow (x \leq a_1)\&(y \geq a)$$
$$a_1 > a \rightarrow (x \geq a_1)\&(y \leq a).$$

(b) n is even:

$$a_1 < a \rightarrow x < y$$
$$a_1 > a \rightarrow x > y.$$

Thus if every population always has an even number of individuals, processing changes in pairs prevents any direct inference of the property values of individuals. We can also use this result for type S inferences by having the global constraint that the difference in size between any population and its parent population should at least be 2. Actually this requirement is always satisfied if all populations start with an even number of individuals.

4.3.2 *Processing Changes in Triplets.* Having populations with an even number of individuals and processing changes in pairs still have the following deficiency. Let the median value a be the average of two protected property values u and v, $u \leq v$; and two individuals with protected property values x and y are added into the population. Then

$$x, y \notin [u, v]\&(a_1 < a) \rightarrow x, y < a_1 \quad \text{or} \quad x, y \notin [u, v]\&(a_1 > a) \rightarrow x, y > a_1.$$

Thus for large population size n and $a_1 < a$, it is highly probable that $x, y < a_1$, or similarly for large n and $a_1 > a$, we have $x, y > a_1$ with high probability. If this deficiency and the requirement of even population size are not tolerable, then changes are processed in triplets. The following inferences are possible. Assume $\text{MEDIAN}(A, B) = a$ and $\text{COUNT}(A) = n$.

(a) Individuals with property values x, y, z are added to population A

$$\text{MEDIAN}(A, B) = a_1 \qquad \text{COUNT}(A) = n + 3.$$

For even or odd n:

$$a_1 < a \rightarrow \text{at least two of } x, y, z \leq a_1$$
$$a_1 > a \rightarrow \text{at least two of } x, y, z \geq a_1.$$

(b) An individual with property value x is deleted from A and individuals with property values y, z are added to A

$$\text{MEDIAN}(A, B) = a_1 \qquad \text{COUNT}(A) = n + 1.$$

For even or odd n:

$$a_1 < a \rightarrow \text{at least one of } y, z \leq a_1$$
$$a_1 > a \rightarrow \text{at least one of } y, z \geq a_1.$$

Other changes (i.e., one addition and two deletions or three deletions) result in similar inferences.

Similarly, to prevent disclosures due to type S inferences, we can have a global constraint that the size difference between any population and its parent population should be at least 3, or the individuals creating this difference are not reported. The hierarchical structure of the conceptual model should also be taken into account for type D inferences. Consider Figure 3. Assume three Systems Analysts with BS and with salaries x, y, z are hired and median salaries of populations Systems Analyst with BS, Systems Analyst, and Computer Scientist have changed from a_1, a_2, a_3 to b_1, b_2, b_3 where $b_1 < a_1$, $b_2 < a_2$, $b_3 < a_3$. Now if the DBA learns that user group u had in fact known the value of $z > \max\{a_1, a_2, a_3\}$, then x, $y \leq \min\{b_1, b_2, b_3\}$ is revealed. Thus this inference should be recorded into the UKC of user group u.

Changes (whether processed in pairs or triplets) should be recorded into the related change sequence for auditing and for other tasks of the CEC described in Section 3.3. Also individuals with known (or suspected) property values should be processed only with known individuals to avoid direct inferences about protected property values.

5. CONCLUSION

We have considered the design of a statistical database which utilizes a statistical security management facility to enforce several security constraints at the conceptual data model level. Users are allowed to query the database with different statistical queries such as COUNT, MEDIAN, or SUM. Changes in the database are verified by the DBA. Any information involving few individuals (therefore risking disclosure of confidential or private information) is recorded. Several security measures are also described to help the DBA assess how secure the database is.

APPENDIX A. SUITABILITY OF TWO OTHER DATA MODELS FOR SDB DESIGN

In this appendix we briefly consider two other data models for their suitability as a conceptual model of the SDB, namely, the entity-relationship model [3] and the extended relational model [9]. It should be noted that our aim is to investigate the needed modifications (i.e., rules and constructs) in order to define populations

clearly and to analyze and control inferences. Thus the data models are only briefly described, and other issues such as expressive power, semantics, and naturalness of the data models are not discussed.

A1. The Entity-Relationship (E-R) Model

The E-R model adopts the view that the real world consists of *entities* and *relationships*. An entity is a thing that can be distinctly identified. A relationship is an association among entities. Entities are classified into different entity sets, such as EMPLOYEE, PROJECT, and DEPARTMENT. Similarly, relationships are classified into relationship sets, such as PROJECT-WORKER and DEPARTMENT-EMPLOYEE.

In the SDB, entities and relationships correspond to individuals, and each entity set or relationship set is a population. Statistical information about values in attribute-value pairs of entities or relationships are revealed to users. However, in order to define A-populations and to enforce security constraints, we need additional rules and constructs as described in what follows.

(1) Some means are needed to identify the A-populations that a population contains. For example, the fact that entity set MALE-PERSON is a subset of the entity set PERSON should be easily accessible to the SDB. This is needed, for example, when constraints applied to individuals in an A-population are also applied to all populations that include the same A-population (e.g., SA-constraints).
(2) For the richness of the SDB and systematic application of security constraints, Rule 1 of the D-A model should also be used: Each entity set or relationship set must be composed of some mutually exclusive entity sets or relationship sets.
(3) The protection mechanism of the SDB should easily locate all populations that contain a given A-population. This may be needed, for example, while processing insertions, deletions, or updates of individuals. In the E-R model, locating all populations containing a given A-population requires additional structures or rules.

Finally, in the E-R model, the job of defining the allowable types of statistical queries in a systematic manner relies on the DBA.

A2. The Extended Relational Model (RM/T)

In RM/T, there are *entities* and *entity types* classified by whether they

(1) fill a subordinate role in describing entities of some other type, in which case they are called *characteristic*;
(2) fill a superordinate role in interrelating entities of other types, in which case they are called *associative*;
(3) neither of the above, in which case they are called *kernel*.

Using these entity types, the semantic structures defined are characteristic tree, association graph, Cartesian aggregation, generalization, and cover aggregation. *Cartesian aggregation* is the aggregation abstraction of the D-A model. In what follows we look at each of the new semantic structures of RM/T and

discuss related SDB issues as to how to obtain populations and indivisible A-populations, control inferences, and employ security constraints.

(a) *Characteristic Tree.* The characteristic entity types that provide description of a given kernel entity type form a characteristic tree. Example 7 is from [9].

Example 7. EMPLOYEEs (a kernel entity type) have a JOB-HISTORY (characteristic entity type subordinate to EMPLOYEE) whose immediate properties are DATE-ATTAINED-POSITION and NAME-OF-POSITION (see Figure 8). This information is augmented by SALARY-HISTORY (characteristic entity type subordinate to JOB-HISTORY) whose immediate properties are DATE-OF-SALARY-CHANGE and NEW-SALARY. The mapping between entities in the parent and child nodes is one-to-many, for example, one EMPLOYEE has many JOB-HISTORY entities and one JOB-HISTORY entity has many SALARY-HISTORY entities.

Each of EMPLOYEE, JOB-HISTORY, and SALARY-HISTORY nodes forms a population. A-populations of these populations may be formed by decomposing them (as generalization abstractions) using their properties, their mapping to the parent nodes, etc. For example, SALARY-HISORY may be decomposed using (a) EMPLOYEE individuals, (b) NAME-OF-POSITION of JOB-HISTORY, (c) date, and (d) salary ranges.

Type S inferences in the characteristic tree may occur when the decomposition of a population is effected by its parent node(s). These inferences may identify the existence of entities in parent populations as described in Example 8.

Example 8. Assume SALARY-HISTORY is decomposed using employee job types. Then

COUNT (SALARY-HISTORY-OF-TECHNICIAN) = 1
→ There is only one technician in the company.

Type S inferences may be avoided by constraints which consider the size of A-populations, decomposition rules, and one-to-many mappings of parent populations in the characteristic tree.

Type D inferences in the characteristic tree may occur owing to the fact that any new entity in node v creates new entities in all the characteristic entity type nodes of the subtree with root v. For example, new EMPLOYEE individual creates new JOB-HISTORY and SALARY-HISTORY individuals. This fact brings extra limitations in processing changes. Otherwise processing changes may be achieved by methods discussed in Section 4, i.e., processing changes in pairs or triplets, even-size populations, dummy individuals, etc.

(b) *Association Graph.* An associative entity interrelates entities of other types, and this interrelation is represented by the association graph in the RM/T. If the association among individual entities is to be protected, then type 2 global constraints can be applied to the associative entities or the entities that they interrelate.

(c) *Generalization.* Codd [9] renames the generalization abstraction of the D-A model as *unconditional generalization inclusion* (UGI), and also describes

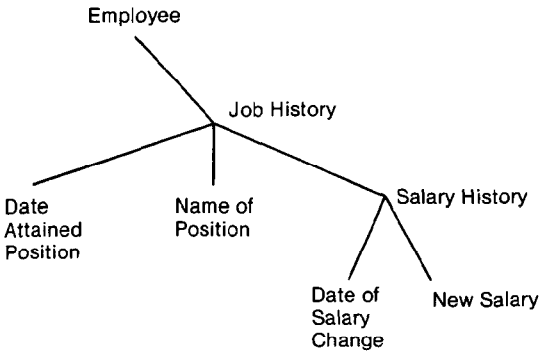

Fig. 8. A characteristic tree.

an abstraction called *alternative generalization inclusion* (AGI), which is an alternative or conditional inclusion of entities of an entity type into some other entity types. Clearly, the only structural difference between the UGI and the AGI is that the AGI decomposes a population P into mutually exclusive populations that are one level above P in the hierarchy. Thus controlling inferences for the AGI are the same as the UGI.

(d) *Cover Aggregation.* Cover aggregation is an aggregation in which a subset of entities of the same type forms another entity with a different entity type. For example, a CONVOY-OF-SHIPS is a cover entity of entity type SHIP; a CLUB that some people belong to forms a cover aggregate of PEOPLE.

For the SDB design, cover aggregate entity types partition the group of entities resulting in smaller populations. Intersection of these smaller populations gives A-populations of covered individuals.

APPENDIX B. IMPLEMENTATION CONSIDERATIONS

In this appendix we comment on implementation and maintenance issues of security-related structures.

To answer a statistical query, conventional database query processing is performed to obtain the answer, and then related security constraints (i.e., global and SA-constraints) are enforced. To retrieve SA-constraints, the user group of the user is determined and the related UKC is accessed. To retrieve global constraints, the PDC of the population described in the user's query is accessed. Thus the additional time overhead to the conventional database query processing involves two accesses and the processing of security constraint routines.

To process changes (i.e., insertion, deletion, or updates of individuals), in addition to the conventional database query processing, the related PDC is accessed to modify global constraints, and then, for each user group, the UKC is retrieved to update SA-constraints, and the user group's knowledge and change sequence and its parameters.

In what follows we discuss possible operations on the known value set (KVS), the known upper bounds set (KUBS), and change sequences and their parameters.

The operations on the KVS are set membership check, insertion, and deletion. Clearly, the KVS is a dictionary [2]. Thus a balanced tree (a 2-3 tree or an AVL

tree) may be used to process an operation of the KVS in $O(\log m)^3$ time, where m is the number of individuals in the population.

The KUBS of a property of an SA-population for the user group u is maintained for the following operations:

(1) search for
   (a) a known upper bound value of a given individual in the population;
   (b) all individuals in the population such that the difference between their property values and the known upper bound values are less than a certain constant.
(2) Insertion of a known upper bound value into the KUBS;
(3) deletion of a known upper bound value from the KUBS;
(4) update of a known upper bound value in the KUBS.

Operation (1b) may be required to assess how "accurate" the user's knowledge is. All of these operations can be processed in $O(\log m)$ time, where m is the number of individuals in the population, using techniques described in [2].

The change sequence is consulted during checking for disclosure, and modified during a change in the population. The operations on the change sequence are:

(1) inserting a new change into the change sequence and modifying the change sequence parameters;
(2) finding a certain reachability set from the change sequence;
(3) finding all reachability sets from the change sequence.

Operations 1, 3, and 2 can be processed in $O(\log k)$, $O(k + t \log s)$, and $O(t \log s)$ time, respectively, where k is the size of the active individuals set, t is the number of tuples in the change sequence, and s is the maximum number of active individuals at any time during the construction of the change sequence.

ACKNOWLEDGMENT

The authors would like to thank Jeffrey Sampson for his help in improving the readability of this paper.

REFERENCES

(Note. Reference [11] is not cited in the text.)

1. ACHUGBUE, J.D., AND CHIN, F.Y. The effectiveness of output modification by rounding for protection of statistical databases. *INFOR 17,* 3 (1979), 209–218.
2. AHO, A.V., HOPCROFT, J.E., AND ULLMAN, J.D. *The Design and Analysis of Computer Algorithms.* Addison-Wesley, Reading, Mass., 1976.
3. CHEN, P.P-S. The entity-relationship model—Toward a unified view of data. *ACM Trans. Database Syst. 1,* 1 (March 1976), 1-36.
4. CHIN, F.Y. Security in statistical databases for queries with small counts. *ACM Trans. Database Syst. 3,* 1 (March 1978), 92-104.
5. CHIN, F.Y., AND OZSOYOGLU, G. Security in partitioned dynamic statistical databases. *Proc. IEEE COMPSAC Conf.,* 1979, pp. 594-601.
6. CHIN, F.Y., AND OZSOYOGLU, G. Security of statistical databases. Dep. Computing Science, Univ. Alberta, Edmonton, Alberta, Canada, 1979.
7. CODD, E.F. A relational model of data for large shared data banks. *Commun. ACM 13,* 6 (June

[3] Base of the logarithm is always 2. O-notation is described in [2].

1970), 377–387.
8. CODD, E.F. Recent investigations in relational database systems. Information Processing 74, North-Holland Pub. Co., Amsterdam, 1974, pp. 1017–1021.
9. CODD, E.F. Extending the database relational model to capture more meaning. *ACM Trans. Database Syst. 4,* 4 (Dec. 1979), 397–434.
10. DAVIDA, G., AND ROCHELEAU, R. Compromising a database using MEAN queries of variable length. TR-CS-77-2, Univ. Wisconsin, Milwaukee, Wisc., 1976.
11. DEMILLO, R.A., AND DOBKIN, D. Recent progress in secure computation. *IEEE 2nd Int. Conf. Computer Software and Applications,* 1978.
12. DEMILLO, R., DOBKIN, D., AND LIPTON, R.J. Even databases that lie can be compromised. *IEEE Trans Softw. Eng. SE-4,* 1 (1978), 73–75.
13. DENNING, D.E. Are statistical databases secure? Tech. Rep., Computer Sciences Dep., Purdue Univ., W. Lafayette, Ind., 1977.
14. DENNING, D.E. Secure statistical databases with random sample queries. Computer Sciences Dep., Purdue Univ., W. Lafayette, Ind., 1979.
15. DENNING, D.E., DENNING, P.J., AND SCHWARTZ, M.D. The tracker: A threat to statistical database security. *ACM Trans. Database Syst. 4,* 1 (March 1979), 76–96.
16. DOBKIN, D., JONES, A.K., AND LIPTON, R.J. Secure databases: Protection against user inference. *ACM Trans. Database Syst. 4,* 1 (March 1979), 97–106.
17. DOBKIN, D., LIPTON, R.J., AND REISS, S.P. Aspects of the database security problem. *Proc. Conf. Theoretical Computer Science,* Waterloo, Canada, 1977.
18. FELLEGI, I.P., AND PHILLIPS, J.L. Statistical confidentiality: Some theory and applications to data dissemination. *Ann. Econ. Soc. Meas. 3,* 2 (1972), 399–409.
19. HAMMER, M.M., AND MCLEOD, D.J. Semantic integrity in a relational database system. *Proc. Very Large Databases,* 1975, pp. 25–47.
20. HANSEN, M.H. Insuring confidentiality of individual records in data storage and retrieval for statistical purposes. *Proc. AFIPS 1971 FJCC,* vol. 39, AFIPS Press, Arlington, Va., pp. 579–585.
21. HOFFMAN, L.J. *Modern Methods for Computer Security and Privacy.* Prentice-Hall, Englewood Cliffs, N.J., 1977.
22. HOFFMAN, L.J., AND MILLER, W.F. Getting a personel dossier from a statistical data bank. *Datamation 16,* 5 (May 1970), 74–75.
23. KAM, J.B., AND ULLMAN, J.D. A model of statistical databases and their security. *ACM Trans. Database Syst. 2,* 1 (March 1977), 1–10.
24. KERSCHBERG, L., KLUG, A., AND TSICHRITZIS, D. A taxonomy of data models. *Proc. Very Large Databases,* 1976, pp. 43–63.
25. NIJSSEN, G.M. (ED.) *IFIP Working Conf. Modelling in Data Base Management Systems, Proceedings,* North-Holland, 1976.
26. OZSOYOGLU, G., AND CHIN, F.Y. Enhancing the security of statistical databases with a question-answering system and a kernel design. Tech. Rep., Dep. Computing Science, Univ. Alberta, Edmonton, Alberta, Canada, 1979.
27. REISS, S.P. Security in databases: A combinatorial study. *J. ACM 26,* 1 (Jan. 1979), 45–57.
28. Schlörer, J. Identification and retrieval of personnel records from a statistical data bank. *Methods Inf. in Med. 14,* 1 (Jan. 1975), 7–13.
29. SCHLÖRER, J. Confidentiality of statistical records: A threat monitoring scheme for on-line dialogue. *Methods Inf. in Med. 15,* 1 (Jan. 1976), 36–42.
30. SCHLÖRER, J. Union tracker and open statistical databases. Rep. TB-IMSD 1/78, Inst. Medizinische Statistik und Dokumentation, Univ. Giessen, Giessen, W. Germany, June 1978.
31. SCHWARTZ, M.D., DENNING, D.E., AND DENNING, P.J. Linear queries in statistical databases. CSD-TR-216, Computer Sciences Dep., Purdue Univ., W. Lafayette, Ind., 1976.
32. SMITH, J.M., AND SMITH, D.C.P. Database abstractions: Aggregation. *Commun. ACM 20,* 6 (June 1977), 405–413.
33. SMITH, J.M., AND SMITH, D.C.P. Database abstractions: Aggregation and generalization. *ACM Trans. Database Syst. 2,* 2 (June 1977), 105–133.
34. PRIVACY ACT OF 1974. Title 5, United States Code, Section 552a (Public Law 93-579), 1974.
35. YU, C.T., AND CHIN, F.Y. A study on the protection of statistical data bases. *Proc. ACM SIGMOD Int. Conf. Management of Data,* Toronto, Canada, 1977, pp. 169–181.

Received May 1979; revised February 1980; accepted May 1980