Sensitivity of the Physical Database Design to Changes in Underlying Factors

By: Prashant Palvia

Palvia, P. "Sensitivity of the Physical Database Design to Changes in the Underlying Factors," Information & Management, Vol. 15, 1988, pp. 151-161.

Abstract:
Several articles have reported on the optimization of the complete physical database design or significant parts of it. However, few results are available on the sensitivity of the physical database design to change in underlying factors that influence the design decisions. Such tenths would be of significant practical value in both the design of the initial physical database and its subsequent restructuring. This paper considers underlying factors related to logical data structure, queries, computer system and physical implementation. An experimental design based on these factors is developed and experiments are conducted to determine the sensitivity effects, and these are reported and discussed.
Keywords: Database, Database design, Physical database, Initial database design, Database restructurings, Database monitoring, Sensitivity analysis, Performance evaluation, Optimization, Input-output analysis

Article:
1. Introduction
Sensitivity analysis refers to change in outputs or the final optimal solution due to changes in inputs. It has been applied in many complex settings in different business areas, e.g., in accounting [13], financial management and capital budgeting [12,21], and manufacturing [20). Currently, many computerized tools, such as IFPS [14] and LOTUS [17], facilitate the process of conducting this kind of "what-if" analysis. In addition, operations research has produced formal methods for conducting sensitivity analysis in special cases (e.g., linear programming problems occurring in manufacturing, resource allocation, and scheduling and in inventory management problems [10]). Here we consider the sensitivity of the physical database design to underlying variables. Results were obtained by conducting controlled experiments in a laboratory setting.

The design of physical databases is a complex activity. Many researchers have proposed methods to optimize either specific sub-problems in this area [9,11.18,19] or to optimize the entire problem in different contexts [3,6,7,15,16,22,23,24]. They consider many factors simultaneously in the design process. Research/experience does not generally provide information on the sensitivity of the physical database design to the underlying factors. Such results are of significant practical both at initial design and redesign. During the initial design (especially in the absence of optimization tools), the designer is aided in the knowledge of the relative importance of the various factors. This knowledge also helps at redesign time, because the designer is constantly faced with restructuring the physical database due to changing user and technical requirements.

What is meant by sensitivity analysis? As the design factors change, both the optimal physical database design itself and the operational cost of using the design may change; indeed the cost will almost always change. However, our main concern in this sensitivity analysis is to study the relative changes in the optimal physical database design.

2. Objectives and Underlying Factors in Physical Database Design
A generally accepted goal in physical database design is to minimize the operational costs of using the database. In this study, two operational costs are considered: that of storing the data and that of accessing it (as measured

by the number of page access from secondary memory). These are influenced by four major factors: A. logical data structure; B. database activity; C. computer system characteristics; and D. physical factors (including the physical design model).

The logical data structure (LDS) consists of entities and relationships between them. The LDS may be represented using a data structure diagram [1] or an entity—relationship diagram [5]. Figure 1 is an LDS for a personnel database, adapted from [3]. Relationships in this are self-explanatory. We assume that the LDS is given based on prior design activities. The operations on the database may be. queries or updates. Our work focuses queries (although some restricted updates can be considered). A query may *require* selected instances of only one entity (e.g., data about certain employees only) or data from several entities (e.g. data about certain departments and employees who work in them). The latter query is more complex and requires " traversing" several entities. The relevant computer system characteristics are the page size, pointer size, buffer size, and storage and access cost.

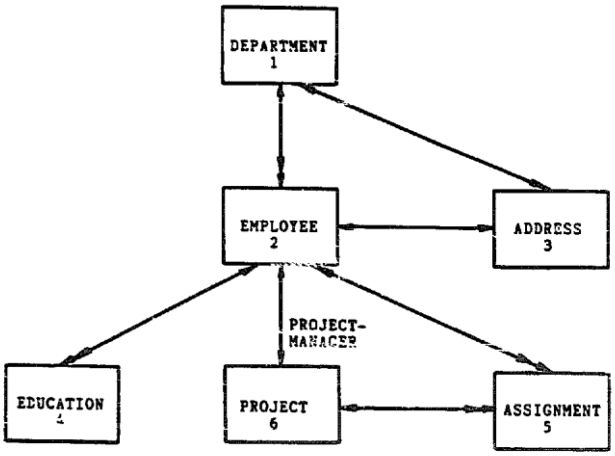


Fig. 1. Example of a logical data structure (LDS).

In addition, many physical factors, e.g., the available access paths and the data processing strategies, also influence the physical database de-sign. Leading among the factors is the physical design model itself. This model prescribes the permissible alternative physical designs. Different commercial database management systems (DBMS's) use their own design models. Our work uses a generic and abstract physical design model.

3. The Generic Physical Design Model

The model used here is based on some well-recognized principles of physical design, as reported in many current models [2,3,16,19,23]. These bring out two fundamental principles for physically representing two entities that are related via a relationship. The first is well-known: store the two entities in separate independent files and indicate the relationship by some kind of pointers in the file records. The pointers may be linked-lists, inverted lists, or some combination and they may be direct or symbolic. The second principles is the concept of clustering or aggregation in which all instances of one entity that are related to another in a second entity are clustered with or near one another.

These alternatives yield substantially different physical designs. Our generic model captures the spirit of the two; our experimentation showed that details do not have a significant impact on the physical design. The model allows for five ways of representing two entities X and Y and the relationship between them:

a. Create two record types X and Y with X having pointers to Y.

b. Create two record types X and Y with Y having pointers to X.

c. Create two record types X and Y with both pointing to one another.

d. Create one record type X which will aggregate (cluster) the related Y instances together. Aggregating in the abstract model is realized by making the related Y instances part of the X record.

e. Create one record type Y that will aggregate the X instances.

Table 1

Entities	1	2	3	4	5	6	Explanation
Design 1	0	0	0	0	0	0	Unclustered flat-file design. All entities rooted, i.e. 6 files.
Design 2	0	1	0	2	2	0	1 clusters 2; 2 in turn clusters 4 &5. Entities 1, 3, 6 are rooted, i.e. 3 files in physical database.
Design 3	0	0	0	0	6	0	Only 6 clusters 5. All entities, except 5, rooted, i.e. 5 files.

Note that the pointers may be all symbolic or all direct. It is worthwhile to note that this model has strong parallels in commercial DBMS's. For example, hierarchical and network systems, generally incorporate pointers and/or aggregation. Aggregation is supported in IMS by permitting hierarchical segments in the same data set; in network systems, by storing MEMBER record types in the OWNER area using VIA SET and NEAR OWNER commands in the schema. Relational systems do not allow aggregator at a logical level; however, substantial efficiencies are achieved by its use at the physical level [4,8]. In fact, many relational systems are now beginning to support clustering (e.g. SQL and RBASE).

The total number of physical designs, using the abstract model, explode exponentially when the number of entities and relationships in the LDS is large. Fortunately, we can prune many of these options at an early stage. We found that while the optimal design is sensitive to the aggregation and pointer options, it is not very sensitive to the specific pointer option. Thus one of the three pointer options can be preselected for each related pair of entities, using judgment, guidelines, or analysis. With one, pointer option and two aggregation options, a physical design can be fully specified by indicating only die aggregations. A short-form notation can then be used icy represent a physical design. In this short-form, only the "aggregator" or "absorber" entity (i.e. the "physical parent") of each entity is named. A root entity does not have a physical parent; so its parent is numbered 0. Table 1 shows some designs for the six entity LDS.

Note that the first design has all entities stored in their own independent files (with appropriate pointers to provide the relationships). We call this a flat-file design.

4. Details of the Underlying Factors

A broad range of factors was included in this study. While there may be additional factors, we believe that we captured the most important ones. In most experimental studies, it is common to have factors at two levels (or occasionally three); our study uses three levels for most quantitative factors. The actual values for the levels *were* established to provide a practical range on each factor. The factors and factor levels are shown in Figure 2.

4.1. Logical Data Structure (LDS) Related Factors

The number of entities and relationships in the LDS determine the LDS size and complexity. Since the two are highly correlated, we used tile number of entities as a measure of LDS size and complexity. Since this is the most important factor representing the LDS, four levels were chosen: LDS with 4, 6, 8 and 10 entities.

4.2. Queries Related Factors

In addition to the number of queries, the structural characteristics of the queries may also affect the optimal physical design. Four "structural" factors were considered. For the factor: "average number of entities traversed per query", three sets of queries were constructed to vary the factor at three levels. In general, the number of entities traversed will not exceed the number of entities in the LDS; so three levels (low, medium, high) were used for the six-entity LDS: 1.67, 3.17 and 4.5 respectively. Similarly, query mixes were constructed to simulate the other factors and their levels. The factor: "proportion of entity instances addressed by queries"

depends on the operating environment. In a production/batch environment, queries address a large proportion of the entity instances, while an executive environment has queries addressing only a small proportion of the entity instances. The factor: "distribution of activities on the LDS" refers to the concentration or distribution of the queries over the entire LDS. Three levels were created by adjusting the frequencies of a basic set of queries.

Fig. 2
Experimental factors and parameters of the base case.

Factor and factor level	Number of levels	Base case value
A. Number of entities in the LDS (4, 6, 8 and 10 entities)	4	6 entities
B. Number of queries (3, 6 and 9 queries)	3	6 queries
C. Average Number of entities traversed per query (low, medium and high)	3	Medium (3.17 entities)
D. Proportion of entity instances addressed by queries (low, medium and high)	3	Medium
E. Distribution of queries on LDS (low, medium and high concentr.)	3	Low concentration
F. Degree of conflict in queries (low, medium and high)	3	Medium
G. Access/storage costs (access, storage and combination of both)	3	Access costs
H. Page size (2000 and 4000 characters)	2	2000 characters
I. Processing Strategy (PS1 and PS2)	2	PS1 for one base case, PS2 for second base case
J. Access Path (sequential and random)	2	Random
K. Pointer type (symbolic/Direct)	2	Direct
L. Mandatory vs non-mandatory two-way pointer	2	Mandatory

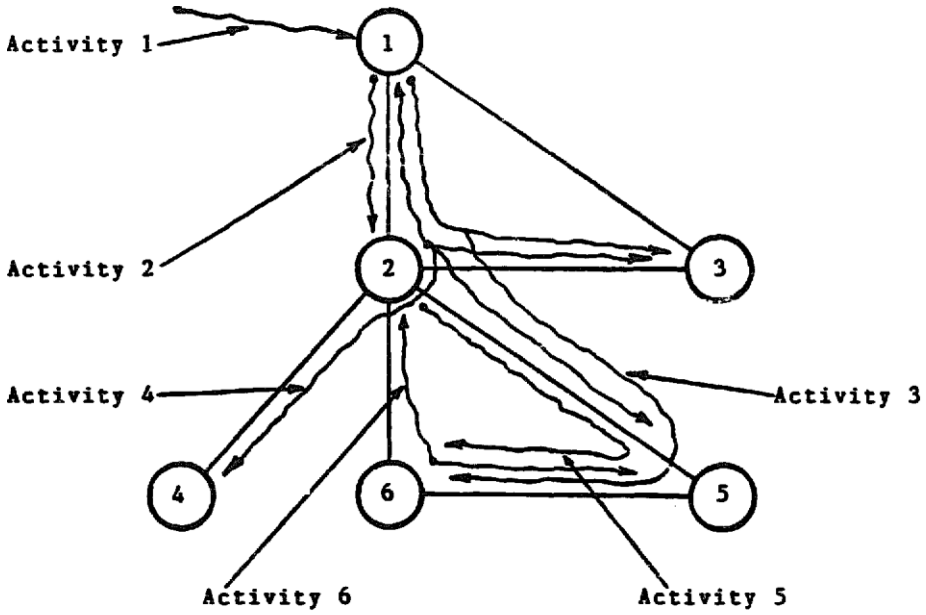
The last factor: "degree of conflict in queries" may make the physical design problem especially difficult. If all queries traversed the LDS in the same direction, there would be no conflict and it would be relatively easy to obtain the optimal design. Since a measure for the degree of conflict was not found in the literature, we devised our own method. In this method (see Figure 3), we focus on the number of queries along each relationship and split the queries into two classes, one for each direction along the relationship. For each relationship, let F1 be the sum of the frequencies of all queries in one direction and F2 be the sum of the frequencies of the queries in the other. Let FH be the greater of F1 and F2 and FL the smaller. Compute FH and FL for each relation-ship in the LDS. Let FHS be the sum of all FH's and FLS be the sum of all FL's. The ratio of FLS to FHS is then called the degree of conflict. This ratio varies between zero and one; a zero value indicates no conflict and a high value indicates high conflict. Again three sets of queries were created to indicate low, medium, and high conflict.

4.3. Computer System Related Factors

The computer system and the operating environment will determine the relative importance of storage and access costs. Three cases are considered: access costs alone, storage costs alone, and a realistic mix of both costs. For many computer systems, the access costs alone are of primary importance. The second computer system related factor is the page size and two levels for this factor are considered.

4.4. Physical Implementation Factors

implementation factor is the strategy used to process the queries. The prime processing strategy is to access files and bring records into main memory, as necessary, while traversing the query path: this is the PS1 strategy. A second strategy (PS2) is considered when maximum batching is allowed. It requires extremely large buffer sizes, so that all required records from each file are brought into main memory at the same time and the file need not be accessed further. Unless queries are simple or main memory is very large, real systems provide a limit on lookforward and the extent of batching. Thus the PS2 results should be interpreted as an upper bound; while the PSI results are more realistic.



Relationship	FH	FL
1-2	2	1
1-3	0	0
2-3	2	0
2-4	1	0
2-5	3	0
2-6	1	0
5-6	2	1
Total	11	2

Conflict = 2/11 = .182

Fig. 3. Measure of degree of conflict.

Another physical factors is the choice of random vs. sequential access paths. The first are more common in today's multi-user databases requiring fast online retrieval. Sequential access paths are evaluated for the sake of completeness and because batch-oriented systems will use sequential access. The pointer types can be symbolic or direct; mixing of these is permitted. Finally pointer directions can be either all two-way or an intelligent selection of one from the three pointer options available for each entity pair.

5. The Experimental Design

With the twelve experimental factors considered above, a full factorial design will require experimentation with 62, 208 cases based on 972 whole new problems. Clearly, it is infeasible to execute in its entirety. Further, since our experiments are deterministic, statistical methods cannot be used for developing a partial factorial design. Therefore, a methodology was developed to select the most important cases for experimentation. This was accomplished by defining a "reasonable case": the base case (Figure 2). To keep this base case reasonable and average, the value of each quantitative factor is set at the middle of the range, and the value of each qualitative factor is set to reflect current practice in database design.

The next step in the experimental methodology is to take each factor, one at a time, and change it to all of its possible values without changing other factors. Thus the sensitivity of each factor is evaluated individually, without regard to its interaction with other factors. If the optimal design is extremely sensitive to a particular factor, then another base problem is created by altering the value of this factor to a new value. The process is then repeated for the new base problem.

Fig. 4
Description of the forty experimental cases. *

	Difference from base case
Cases 1 and 21	No difference
Cases 2 and 22	LDS with 4 entities
Cases 3 and 23	LDS with 8 entities
Cases 4 and 24	LDS with 10 entities
Cases 5 and 25	3 queries
Cases 6 and 26	9 queries
Cases 7 and 27	Low contexts per query
Cases 8 and 28	High contexts per query
Cases 9 and 29	High proportion of entity instances addressed
Cases 10 and 30	Low proportion of entity instances addressed
Cases 11 and 31	Medium concentration of queries on LDS
Cases 12 and 32	High concentration of queries on LDS
Cases 13 and 33	Low conflict in queries
Cases 14 and 34	High conflict in queries
Cases 15 and 35	Consider only storage costs
Cases 16 and 36	Consider combination of storage and access costs
Cases 17 and 37	page size = 4000 characters
Cases 18 and 38	Sequential access paths
Cases 19 and 39	Symbolic pointers
Cases 20 and 40	Flexibility in pointers (i.e. best pointer directions)

**** Cases 1 to 20 use the PS1 processing strategy; and case 21 to 40 use the PS2 processing strategy. Case I and case 21 are the two base cases.***

Thus the philosophy of this experimental design is to *create* and recreate enough base problem to explore significantly different experimental regions. Figure 2 also lists the second base case generated in this manner. The two base cases are due to the two processing strategies: PS1 and PS2. in all there were 20 eases for each base case: a total of 40 cases. The forty cases are described in terms of their differences from the base cases in Figure 4.

6. Sensitivity Analysis Results sand Discussion

While there are many physical designs for each experimental case, we are only interested in the optimal design for the sensitivity analysis. This optimum was obtained by enumerating all possible designs, evaluating each for its operational cost and then selecting the least cost design. The evaluation was performed using a simulator writ-ten in FORTRAN, developed specifically for our physical design model. While a complete description of the simulator is beyond the scope of this paper, its main components and a line diagram are shown in Figure 5.

In order to quantify sensitivity, it is necessary to determine the relative difference between any two physical designs. Since such a measure was not found in the literature, we developed a measure: the *Physical Design Difference Measure* (PDDM). It is obtained by examining the physical parent of each entity in the two designs and counting the number of entities with different parents. Let this number be D. D is divided by the total number of entities, N, to obtain the PDDM. Consider the following two physical designs:

0 0 0 2 6 0
0 1 0 2 2 0

In these, entities 2 and 5 have different physical parents; thus D = 2. Since N 6; PDDM = 2/6 = 0.33.

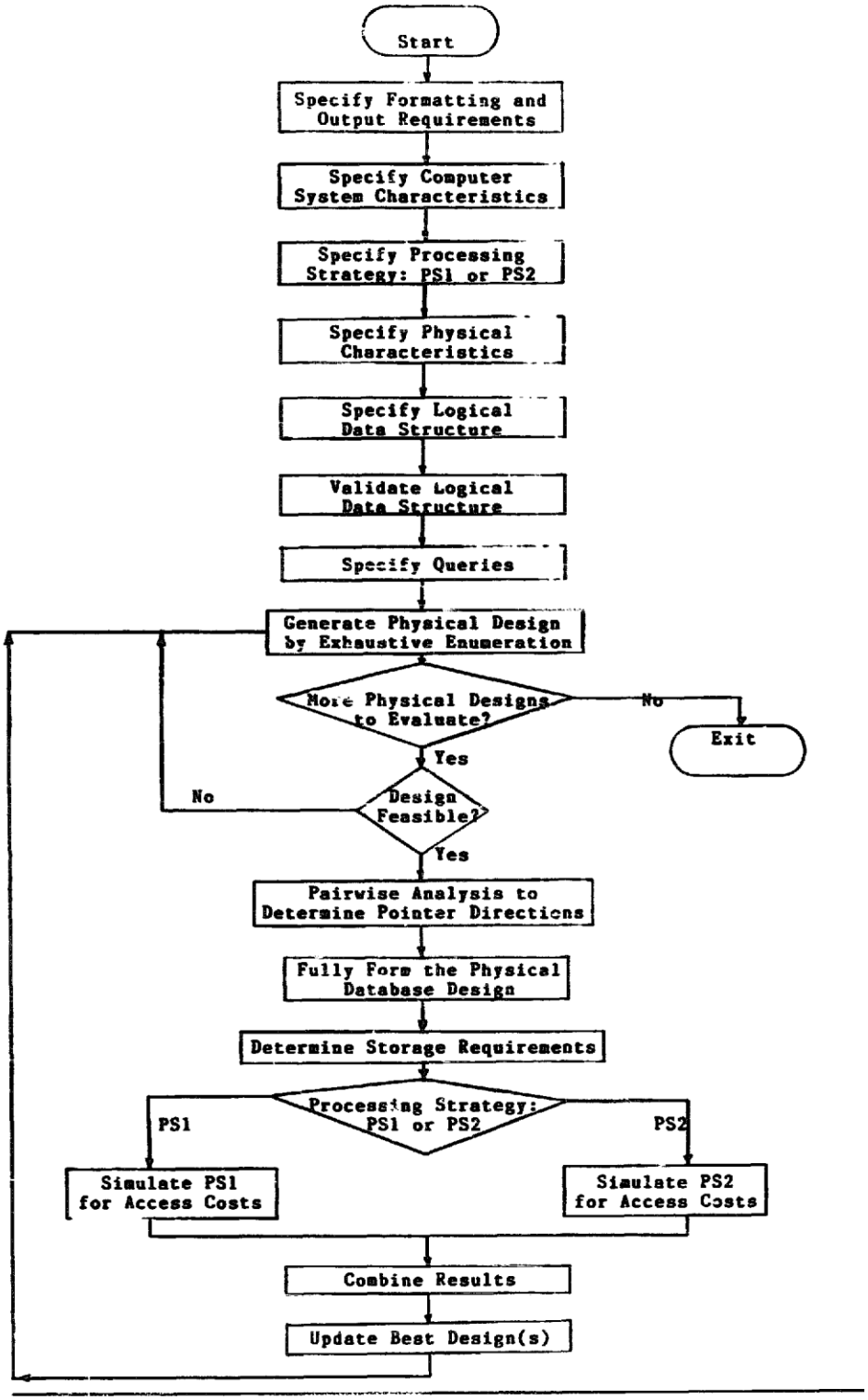


Fig. 5. Main components and line diagram of the simulator.

The process for determining sensitivity is therefore as follows: 1. Obtain the optimal design for each using the simulator, 2. Compare each case's optimal design with that of the base case and compute the PDDM value.

Since there are multiple cases for each factor, the combination of the PDDM value provides the sensitivity of the factor. For multiple cases, the higher PDDM value is used as measure of the sensitivity of the factor (as this reflects the maximum change that can occur in the physical design due to a change in the factor level). The sensitivity rating (and PDDM) can range between 0 and 1; where 0 represents insensitivity and 1 is maximum

sensitivity. The sensitivity rating is classified as "high" if the rating is 0.5 or greater, "medium" between 0.25 and 0.5, and "low" otherwise.

Fig. 6.
Sensitivity of the physical database due to underlying factors.

	Factors not likely to change after initial design	Factors likely to change during database life
High sensitivity	Pointer type (symbolic vs direct)	Number of queries Degree of conflict in queries Access and Storage
Medium sensitivity	Page size Access path	Number of contexts per activity Proportion of entity instances addressed
Low sensitivity	Number of entities in the LDS Pointer directions (two way vs flexible)	Distribution of activities on the LDS

The most sensitive factor is the processing strategy. Since strategy PS1 is used in most DBMS's, we discuss the sensitivity effects of this strategy. These effects, as shown in Figure 6, are split into two categories: those that are less likely to change after initial database design, and those that are likely to change during the database life.

6.1. LDS Related Factors

Changes in LDS factors will change the optimal design because the very character of the problem changes. It may seem unfair to conduct sensitivity analysis if the changes significantly alter the problem definition. Therefore, we conduct sensitivity analysis by changing the LDS size without changing its basic structure (e.g. entities are merely dropped from or added to an existing LDS). During the database life, the LDS may not change much, and if it does, the changes are not likely to be structural.

Our results show that when such changes are made to the LDS (with minimal changes in the queries), then the optimal physical design is only moderately affected. The changes that occur in the physical design are limited to the storage of the entities that are added to or dropped from the LDS.

Guidelines for physical database design based on the logical data structure have been proposed in [3,22]. The guidelines in [3] suggest that (a) a 1 : 1 relationship should be represented by pointers and (b) a ' : M relationship should not be represented I by the "M" entity absorbing the "1" entity. While these guidelines generally apply to the PS2 processing strategy, they are not fully supported in realistic PS1 cases. 'There were several PS1 *cases* that violated these guidelines. Further these guidelines are "static", because they only consider the LDS, which is unlikely to change; they do not consider the effect of queries, which are more likely to change. Our experiments showed that these guidelines 3TC effective as long as the queries are few ann. Ample (i.e., focusing only on one or a few entities). This is generally not the case in large multi-user databases, which have a large number of activities (i.e., queries and updates) and many of the activities are fairly complex. We, therefore, infer that pure LDS based guidelines have limited applicability in physical database design; they are applicable only when the activities on the data-base are few and simple.

6.2. Query Related Factors

The query related factors are the "dynamic" ones in the sense that they are likely to change during the database life. Further, generally all of them have relatively high sensitivity effects. One of the important conclusions is that it is not only the number of activities, but also the "structure" of the activities affect the optimal physical database design. 'The query structural factors include: the number of entities addressed by the query, the degree

of conflict between queries, the proportion of entity instances addressed by the queries, and the distribution of queries on the LDS.

It is intuitively obvious that the number of queries will be a critical factor, as the queries cause accesses to the database and minimizing accesses is an important criterion in physical database design. With fewer queries, LDS characteristics dominate; however different design options are more attractive with larger number of queries. As to the number of entities per query. when queries focus on a single entity alone, the unclustered flat-file design is fairly good. However, as the number of entities per query increase, different clusterings are desirable. For example, if "Employee" entity instances are only required via their corresponding "Department" entity instances, then it is best to cluster Employee into Department. The proportion of entity instances addressed by the queries has a volume effect, in that the effects of clustering are multiplied. The degree of conflict between queries makes the physical design problem especially difficult. For example, consider two conflicting queries: one from the Department entity to the Employee entity and the other from the Employee entity to the Department entity. For the first query, clustering Employee near Department will be advantageous, while the reverse will be true for the second query. Thus the design choices are sensitive to which queries "win out" in the conflict.

As stated earlier, designers have developed intuitive guidelines for physical database design based on the logical data structure alone; but this view offers a microscopic perspective of the design problem. Two guidelines suggested by the author in [22] are: a. the entity with the higher outdegree should absorb the entity with the lower one, and b. if an instance of one entity compared to another is relatively small, then the larger one should absorb the smaller. The first guideline is voided, if the activity is directed predominantly through the entity with the lower outdegree; the second guideline is overridden if the activity is predominantly from the smaller instance entity. Based on these results, LDS based design guidelines are not always adequate; they must be augmented by information about the queries.

6.3. Computer System Related Factors

The design goal of minimizing storage and/or access costs has a significant bearing on the optimal physical design. The storage cost for a given physical design depends only on the design itself, while the access cost depends both on the design and the queries. For this reason, the "minimizing storage costs" objective function yields the same optimal design, irrespective of the other factor values, as long as the LDS contents and structure remain the same. On the other hand, when access costs are important for design, then different designs are optimal based on other factor values. Earlier, we suggested (Figure 6), that this factor may change during the life of the database. It may change if the physical resources become scarce (then minimizing storage costs becomes more important), the costs of the resources may change, or the users may have higher expectations (then minimizing access times and access costs becomes more important).

Another factor: " page size" has some effect on sensitivity. Page size has an effect on the clusterings, because it dictates the amount of data that can be brought into memory at once; thus it affects the size of the clusterings.

6.4 Physical Implementation Factors

The processing strategy has a very high sensitivity effect. Of the remaining physical factors, " mandatory two-way vs. intelligently selected flexible pointers" has low sensitivity effect on the optimal design. The "selected pointer option" reduces the operating cost of the database as it allows more flexibility in the direction of pointers. The low sensitivity can be easily explained as mandatory two-way pointers automatically includes one-way pointers of the selection option.

The "symbolic vs. direct pointer" factor showed a high sensitivity effect. As can be expected, direct pointers yield fewer page accesses, because they directly retrieve records, as opposed to traveling via access paths when using symbolic pointers. Since direct pointers give direct address of the related record, the advantages due to clustering are not very significant. Finally, the "random access path vs. sequential access path" factor had a

medium sensitivity effect. One would expect that absorptions would be less desirable with sequential access paths, as one would have to scan through much more unnecessary data in order to get to the required data.

6.5. Effect of Processing Strategies

All of the above sensitivity effects are for processing strategy PS1. We now summarize the results for strategy PS2. One observation was that the optimal physical designs obtained using the two processing strategies are not very different when the queries are few and simple. With an increasing number of more complex queries, the optimal designs with PS1 been to change, but the optimal designs with PS2 do not change much.

The PS2 strategy accesses each required file only once. Thus, if the file sizes are small, the accesses will also be few. We can thus hypothesize that the objective of minimizing accesses using PS2 is strongly correlated to the objective of minimizing storage. The latter objective depends solely on the logical data contents and structure, and does not change with changing queries; hence the low sensitivity.

For these reasons, almost all factors have low sensitivity effect on the physical database design. The only exceptions were the degree of conflict among activities and the relative importance of storage and access. Another observation is the suitability of the flat-file design when using the PS2 strategy. The flat-file option is used by many designers for physical database design. Our experiments show that the flat-file design is a very good option in the PS2 case. Of the twenty PS2 cases, the flat-file design was optimal in one case, and in fourteen cases the operational cost difference between the optimal and flat-file design was less than 10%. This observation can be directly attributed to our hypothesis, as the flat-file design is good from the standpoint of minimizing storage. On the other hand, the fiat-file design is not always good for strategy PS1. For example, in nineteen of the twenty PS1 cases, the difference between the operational cost of the flat-file and optimal design was more than 20%.

In essence, with die PS2 strategy, the physical design is only mildly sensitive to the underlying factors and the flat-file design is a very good design choice. The flat-file design may not, however, be a good physical design choice for the strategy PS1. Further, in the PS1 case, there are some designs which can be disastrous from a performance standpoint. Thus, care and evaluation is required in physical database design with normal processing strategies; and the activity factors are to be given proper attention in the physical database design process.

7. Implications

The above results have managerial/technical implications both at initial database design as well as during the operating life of the database. All factors identified earlier need to be given proper attention at initial design time. If a design tool is used in creating the initial database design, it must at least address the factors identified as having relatively high sensitivities. The design tools can be fairly comprehensive or rudimentary, and this analysis can help in the selection of the proper tool. If an experimental/benchmarking methodology is used in the selection of the database design, again this analysis would help in focusing on the proper design variables and creating appropriate benchmarks.

During the operating life of the database, those factors that are more likely to change and have high sensitivities may be carefully monitored. Statistical data may be captured on these factors. When significant changes are detected in the factor values, that may be an indication of reevaluating the database design for possible changes.

8. Conclusions

The paper discussed the sensitivity effects of factors that influence the physical database de-sign. These factors were classified into four broad categories: logical, query, computer system, and physical factors. An important conclusion is that the logical data structure factors are important in physical database design, and equally important are the activity (query) related factors. Consequently, any guidelines developed for physical database design should consider both activities and the logical data structure. Further, activities become even more

important during database re-design, as they are more likely to change than any other factor during the life of the database.

The knowledge edge of the sensitivity effects is important, both at initial physical design and subsequent restructuring of the database. It is hoped that this study will trigger future ones, as well as result in the documentation of current experience on the subject. The cumulative and corroborated knowledge will be extremely useful for the data administrator and the database designer.

References

[1] Bachman, C.W. Data Structure Diagrams. *Data Base* 1, 2, 1969.
[2] Batory, D.S. and Gotlieb, C.C. A Unifying Model of Physical Databases. *ACM TODS, Vol.* 7, No. 4, December 1982.
[3] Carlis, J.V. *Investigation in the Modeling and Design of Large, Logically Complex, Multi-User Databases.* Ph.D. Thesis, University of Minnesota, 1980.
[4] Chamberlin, D.D., et al. History and Evaluation of System R. *Communications of the ACM.* October, 1981.
[51 Chen, P.P.S. The Entity-Relationship Model - Towards a Unified View of Data. *ACM TODS 1,* 1, March 1976.
[6] Gambino, J. and Gerritsen R. A Data Base Design Decision Support System. *Proc 3rd International Conference on Very Large Databases,* Tokyo, Japan, 1977. ACM, *New* York.
[7] Gerritsen, R. Tools for the Automation of Database Design. *Proceedings of the New York Symposium OA Database Design,* 1978. Springer-Verlag, New Yorlt.
[8] Guttman, A., Stonebroker, M. Using a Relational Database Management System for Computer Aided Design Data. *Bull. IEEE Comput. Soc Tech. Comm. Database Engg.* June 1982.
[9] Hammer, M. and Niamir, B. A Heuristic Approach to Attribute partitioning. *Proc. ACM SIGMOD,* Boston, MA, 1979.
[10] Hillier, F.S. and Lieberman, G.J. *Introduction to Operations Research.* Holden-Day, Inc. 4th Ed. 1986.
[11] Hoffer, JA. and _____ . Optimal Performance of Invert/kJ Files. *Operations Research, 30,* Z Match-April 1982.
[12] Hsaio, F.S.T. and Smith, LW. An Analytical Approach to Sensitivity Analysis of the Internal Rate of Return Model. *Journal of Finance,* 33, May 1978.
[13] Huefner, R.J. Analyzing and Reporting Sensitivity Data. *T7se Accounting Review,* October, 1971.
[14] *IFPS,* EXECUCOM Systems Corporation. P.O. Box 9758. Austin, TX 78766.
[15] Jain, H.K. A Comprehensive Model for the Storage Structure Design of CODASYL Databases. *Info. Systems.* Vol 9, 3/4, 1984.
[16] Katz, R.H. and Wong, E. Resolving Conflicts in Global Storage Design through Replication. *ACM TODS,* 8, 1, March 1983.
[17] *LOTUS 1-2-3,* Lotus Development Corporation, 55 Cam-bridge Parkway, Cambridge, MA 02142.
[18] March, S.T. and Severance, D.G. The Determination of Efficient Record Segmentation and Blocking Factors for Shared Data Files. *ACM TODS Z* 3. September. 1977-
[19] March, S.T. Techniques for Record Structuring. *ACM Computing Surveys,* March, 1983.
[20] Mehra, S. and Watson, HJ. Analyzing Raw Material Requirements in a Fluctuating Environment: An Input-Output Approach *Proceedings,* Southwest Decision Science Institute, March 1979.
[21] Myers, S.C. *Modern Developments in Financial Management.* Praeger Publications, Inc., NY, 1976.
[22] Palvia, P. *An Analytical Investigation into Record Structuring and Physical Database Design of Generalized Logical Data Structures.* Ph.D. Thesis, University of Minnesota, 1984.
[23] Schkolnick, M. A Clustering Algorithm for Hierarchical Structures. *ACM TODS,* 2, 1, March 1977.
[24] Whang, K.Y., Weiderhold, G. and Sagalowicz D. Physical Design of Network Databases Using the Property of Separability. *Proc. 8th Int'l Conference on Very Large Databases.* Sept., 1982.