Chapter 1

INTRODUCTION

1.1 The Present Problem

Conventionally, the answer to a database query is construed to be a set of all tuples that meet the stated criteria. Strict adherence to this notion in query evaluation has become increasingly unsatisfactory, because decision makers are rather prone to adopting exploratory strategies for information search so as to be able to get some answers quickly. Aggregations in databases are performed in batch mode, that is, when a query is submitted, the system processes a large volume of data over a long period of time, and eventually a final answer is returned. But aggregation processing, even in today's database systems roughly resembles the batch processing of the 1960's. When users submit an aggregation query, they are forced to wait without feedback while the system churns out through millions of records. Only after a significant period of time does the system respond with the final answer, which is usually a small set or a single value. A particularly frustrating aspect of this problem is that aggregation queries are typically used to get a rough picture of a large body
of data, yet they are computed with painstaking precision, even in situations where an acceptably precise approximation might be available very quickly. This classical approach is frustrating to users and has been abandoned in most other areas of computing.

Aggregation is an increasingly important operation in the relational database management systems of today. Queries involving aggregates are common in decision-support environments. Rapid feedback is essential and users cannot wait for the traditional blocking execution model to finish before getting some answers. Hence there is an urgent need to find a sound system that can instantaneously answer the database queries for the satisfaction of the end-users.

1.2 Motivation and Overview

Database management systems are progressively being employed to support end users in their decision-making. Such users have a three-fold requirement:

(i) the answers are summary data that characterize the datasets or are derived from or based on summary data,

(ii) imprecise answers can be tolerated, in other words, approximately correct answers suffice, specifically from aggregate queries, and

(iii) answers must be obtained quickly[7].

As such, database support for efficient computation of summary statistics in the form of aggregation queries becomes very important.
Some of the motivational examples, which would divert the attention from the conventional way of execution, are given here to facilitate rethinking:

```
SELECT R.A
FROM R
WHERE R.B IN
    (SELECT S.B
     FROM S)
```

is transformed into

```
SELECT R.A
FROM R, S
WHERE R.B = S.B
```

which can be evaluated as a single query.

For example,

```
SELECT R.A
FROM R
WHERE R.B = (SELECT AVG(S.B)
             FROM S);
```

is evaluated as two subqueries: the inner subquery is evaluated first, followed by the outer, which takes the result of the inner as a constant.
Processing nested queries with aggregates online pose interesting challenges that non-nested queries do not. First, it is not clear what forms the answers to a query, and how the results should be interpreted, given that the inner query is an aggregate query. Further, the query may be an enumerative query where answer tuples are returned, or an aggregate query where a single aggregate value is returned, and the outer and the inner query blocks may or may not be correlated. Second, it is not clear that how processing of the query can be optimally time-sliced across the outer and inner subqueries.

The present work is a novel effort at attacking these challenging issues and defining the future scope and direction for progressive query evaluation.

1.3 Problem Definition

For many Decision Support Systems, exact answers are not always required. Getting approximate answers in seconds is more useful than getting exact answers in minutes/hours.

In this thesis an ONline Aggregation Model, hereafter referred to as ONAM, is adopted as an effective approach to provide rapid online feedback for evaluating nested queries with aggregates. In this approach, two classes of nested queries, Type-A and Type-CA, are considered. The database may have the following schema:

ITD (Pval, Empid, Tax, Savings, Deductions, Earnings, Name, Address)

STUDENT (Pid, Income, Marks, Name, Address)

ITD10 (Pval, Earnings, Deductions, Savings, Tax, Policies, Name, Address)
Single Level Nesting with Multiple Aggregates

SELECT Pval, Tax, Name
FROM ITD
WHERE Pval >
(SELECT avg(Pval)
FROM ITD)
AND Tax >
(SELECT avg(Tax)
FROM ITD);

Aggregate Query with a Single Aggregate

SELECT avg(Marks)
FROM STUDENT
WHERE Income >
(SELECT avg(Income)
FROM STUDENT);

Aggregate Query with Multiple Aggregates

SELECT avg(Tax), avg(Policies)
FROM ITD
WHERE Pval >

(SELECT avg(Pval)
FROM ITD)

AND Deductions >

(SELECT avg(Deductions)
FROM ITD);

Enumerative Multilevel Query

SELECT Pval, Tax, Name
FROM ITD10
WHERE Tax >

(SELECT avg(Tax)
FROM ITD10
WHERE Pval >

(SELECT avg(Pval)
FROM ITD10));
Aggregative Multilevel Query

```
SELECT avg(Tax)
FROM ITD10
WHERE Deductions >
    (SELECT avg(Deductions)
    FROM ITD10)
WHERE Pval >
    (SELECT avg(Pval)
    FROM ITD10));
```

The **ONAM** approach works as follows. Instead of blocking the execution of the outer query block until the inner query block completes, the outer query block is allowed to proceed as soon as the inner query block produces some estimates for its answers. In other words, the inner query block will be evaluated progressively to provide estimates quickly so that the outer query block can proceed to evaluate progressively. In this way, users can have rapid feedback, that is, approximate answers to their nested queries. Subsequently, both queries can be evaluated concurrently: as the inner query estimates are refined progressively, the answers to the outer query block are also refined based on the inner query block's refined aggregates.

Precisely, the **ONAM** approach seeks to address the shortcomings mentioned earlier. The present work concentrates in evaluating the queries with more than one aggregate per one level of nesting and multiple levels of nesting.

For the first problem, the answer space to the query begins with a superset of the
final answers and is refined as the aggregates from the inner query block are refined. For the intermediary answers to be meaningful, they have to be interpreted with the aggregates from the inner query. For enumerative queries, a priority-based evaluation strategy is proposed to present answers that are certainly in the final answer space first, before presenting those answers whose validity may be affected, as the inner query aggregates are refined.

For the second problem, a *multi-threaded evaluation model* is proposed where the different query blocks are evaluated concurrently in a multi-threaded fashion. The time-slice across two query blocks is *nondeterministic*, in the sense that the user controls the relative rate at which these subqueries are being evaluated.

Nesting of query blocks is a very interesting and powerful feature of SQL. It is well known now that nested queries could be rewritten as multiple separate queries. However, it is the nested queries with aggregates, which require special attention. These queries are run against the databases very frequently, and hence, form an important class of queries.

Most of the earlier approaches have presented the whole problem perspective, rather simplistically by assuming trivial problems for demonstration. As an illustration, in progressive query evaluation, multiple aggregates and multi-level nesting have not been given adequate treatment. Obviously, more research needs to be carried out to address the inherently complex issues involved in it. This research needs to focus on specific algorithms to improve scan performance and on caching the intermediate results in order to make effective use of the cached results for further evaluation. Fast initial response is the key here. More exploration could also be carried out in relating progressive query evaluation to real-time systems.
1.4 Organization of Thesis

In Chapter 2, an overview of the literature produced so far on Nested queries is presented. A historical perspective of the various evolutions in databases related to query processing is provided in this chapter. Specifically, this chapter has been divided into three subsections. In the first subsection, it provides the information about the traditional query execution model, its application area and the problems associated with it, in a large volume of data. The second subsection deals with the overview of numerous other works that have indirectly contributed to the growth of this area of research. Random sampling from relational databases is just one of them. The third subsection summarizes the work that has been carried out so far on Online Aggregation, elaborates the early approximate answers, and the work carried out on single level of nesting. Finally, the chapter narrows down to the area yet to be taken up and identifies the specific issue of study for this dissertation.

Chapter 3 describes the proposed Online Aggregation Model to provide rapid online feedback in the case of Nested Queries with Multiple Aggregates as well as Multi Level Nested Queries with Multiple Aggregates. It consists of six subsections. The first subsection deals with the description of the solution to provide the rapid online feedback. The second subsection gives the information about the framework. It develops a three-step approach while evaluating the queries with multiple aggregates and provides the essential condition for sufficient estimation. The third subsection deals with the optimistic approach followed for the refinement of the answer set. It presents a three-step methodology for the inclusion of desired set of tuples and removal of unwanted tuples, if any. This is applied for queries with multiple levels of nesting.

The description of the multi-criterion partitioning approach for the execution of
remainder query is the content of fourth subsection. If any one of the calculated bounds becomes greater than its current running aggregate, then a remainder query is generated and the partitioning approach is applied again on the newly generated tuples. This is elaborated by way of using the activation of an outer thread during the execution of nested queries. The fifth subsection deals with the key performance metrics of the proposed model. It treats with the performance goals like response time versus accuracy, response time versus completion and pacing of results. The final subsection summarizes the conclusions of this chapter.

Chapter 4 gives an account of the proposed model and its implementation for the queries with multiple aggregates for a single level of nesting. This chapter consists of six subsections. The first subsection gives the overview of the approach that has been adopted in the new aggregation model. The second subsection deals with the various types of queries with multiple aggregates for single level of nesting. It also presents an overview of online evaluation. The third subsection introduces the new approach, that is ONAM with its objectives and performance goals. The next subsection deals with the evaluation strategies for the inner query block as well as the outer query block, and the associated Multi-threaded evaluation model. It also concentrates on the Multi-Criterion Partitioning approach adopted to display the set of the most likely tuples in the model. The following section discusses the approach as applied to the refinement of the answer space. All the related algorithms intended for the implementation of the ONAM for the queries involving multiple aggregates for a single level of nesting, are given in the later subsection. The final subsection summarizes the conclusions of this chapter with its performance considerations by taking various parameters like number of partitions, single and multiple relations in subqueries.

Chapter 5 extends the work carried out on the progressive evaluation of queries
with multiple aggregates and multiple levels of nesting. Various subsections of this chapter further detail the proposed methodology to carry out the work. The first subsection of this chapter gives the evaluation strategies for single aggregate on a single relation, multiple aggregates on a single relation, and multiple aggregates on multiple relations. The next section deals with the queries involving multiple levels of nesting. In this section it takes two classes of outer queries. One is Aggregate outer query and the other is Enumerative outer query. It also elucidates the partitioning approaches employed for displaying the initial set of tuples. There are some issues related to the assurance of providing quality results, scheduling of the execution of various threads and handling the effect when some threads complete their evaluation prior to their inner threads in the context of multiple levels of nesting. All these are taken care of in this section.

In the process of running queries of multiple aggregates with multiple levels of nesting, there may be a requirement for the refinement of answer spaces. This case may be due to two reasons: first, some tuples may not enter the final answer space even though they are present in the calculation of the final aggregate, second, some tuples may enter the final answer space even though they may not be present in the calculation of the final aggregate. The contribution of the next subsection is the refinement of answer tuples for the case mentioned. The Algorithms subsection provides the various stepwise procedures developed to implement the proposed model for queries with multiple aggregates and multiple levels of nesting. It includes segments of algorithms to carry out each plan. The final subsection of this chapter concludes with the complexities allied with the time and the space. It includes remarks about the issues of null values, and the future work that can be carried out in this direction for joins and correlated queries.
Chapter 6 highlights the implementation of the proposed approach ONAM, in order to establish its validity. All the graphical user interfaces are presented in this chapter. There are five subsections in this chapter. The first subsection describes the evaluation strategy of ONAM. The second subsection presents the study of the single-level enumerative nested queries with multiple aggregates whose outer queries do not contain aggregates. The implementation of a single-level nested enumerative queries offers a self-explanatory and user-friendly interface. It assumes that the least or no technical expertise is required to operate with the interface. The experiments are evaluated with the multi-criterion priority-based partitioning strategy. The initial response times to get the first approximate answers are taken and compared with the traditional blocking execution model. The parameters that influence the proposed model like the % of processing completed, running aggregates, confidence levels and intervals are experimented with various proportions.

The third subsection deals with the aggregate query with a single aggregate, where the outer query and the inner query have the aggregate function. ONAM's optimistic approach in dealing with aggregate queries as well as the interface for evaluating the aggregate query with a single aggregate are explained in this section. This subsection also deals with an aggregate query with multiple aggregates, where the outer and the inner queries contain more than one aggregate function. The queries, which operate on only a single relation, have been implemented. The user interface shown in this section can be used for queries that operate on multiple relations also. The fourth subsection deals with the most widely used class of queries, namely, the enumerated queries, and the aggregation queries with multiple levels of nesting. In the proposed implementation three levels of nesting are shown and they can be extended to any desired levels of nesting. In these sections both the user interfaces shown are self-explanatory. The user interface provides clear information about each level when
that particular level is selected. It gives the amount of processing completed for each level. Particularly these queries are the combinations of the above-mentioned classes of queries. The last subsection concludes with the essence of good user interface.

The penultimate Chapter deals with the results of the implementation of the queries followed by an in-depth discussion and analysis of the results. In ONAM, the key performance metrics constitute the response time and throughput for useful estimations to an answer, rather than the response time and throughput of a completely accurate answer. Half-width is also one of the metrics, which evaluates the accuracy of the given answer. Most of the graphs shown are against the initial response time or half-width. The graphs include the queries like Single Level Nested Enumerative Queries, Single Level Nested Aggregate Queries with a Single Aggregate and with Multiple Aggregates and Multi Level Nested Queries. Finally, it concludes with the performance and impact of various parameters on each other.

Finally, the last Chapter sums up the ONAM statement that the approximate answers in seconds are more effective than the accurate answers in minutes/hours in decision support environments. It concludes with the performance considerations of all the implementations of the thesis under different classifications like queries with multiple aggregates and multiple levels of nesting and reiterates the contributions of the thesis. This chapter includes the directions for modifications into DBMS necessary to implement online aggregation model. It also tries to converge some existing technologies of displays into online aggregation model. It also offers certain directions for future research and enhancements.