Automating Data Warehouse Conceptual Schema Design and Evaluation

Cassandra Phipps
ABB Inc.
650 Ackerman Rd.
Columbus, OH 43202
casie.phipps@us.abb.com

Karen C. Davis
ECECS Dept.
University of Cincinnati
Cincinnati, OH 45221-0030
karen.davis@uc.edu

Abstract

The popularity of data warehouses for analysis of data has grown tremendously, but much of the creation of data warehouses is done manually. We propose and illustrate algorithms for automatic conceptual schema development and evaluation. Our creation algorithm uses an enterprise schema of an operational database as a starting point for source-driven data warehouse schema design. Candidate conceptual schemas are created using the ME/R model, extended to note where additional user input can be used to further refine a schema. Our evaluation algorithm follows a user-driven requirements approach that utilizes queries to guide selection of candidate schemas most likely to meet user needs. In addition, we propose a guideline of manual steps to refine a conceptual schema to suit additional user needs, for example, the level of detail needed for date fields. The algorithms are illustrated using the TPC-H Benchmark schema and queries. Our algorithms provide a foundation for a software tool to create and evaluate data warehouse conceptual schemas.

1. Introduction

As conventional transaction processing systems have matured, becoming faster and more stable, the focus of organizational needs has changed. Increasing the value of transaction processing systems "means turning data into actionable information" [R96a]. Although traditional OnLine Transaction Processing (OLTP) systems may have some, or all, of the necessary data, it is not easily accessed by the user for analytical processing. The need for OnLine Analytical Processing (OLAP) gives rise to the data warehouse concept.

A data warehouse creation process consists of five steps: pre-development activities, architecture selection, schema creation, warehouse population, and data warehouse maintenance [M97, SC99]. The focus of this paper is the schema creation phase and its automation; while this phase includes conceptual, logical, and physical schema design, we only address conceptual design of a data warehouse here. The conceptual model allows a high-level design of entities and their relationships, represented in a user-friendly manner independent of implementation issues. A conceptual schema is a description of the data to be in the data warehouse that is understandable by end users to verify requirements, identify possible gaps, and conduct analysis for business goals.

We propose automated techniques to develop and evaluate candidate data warehouse conceptual schemas using source-driven and user-driven requirements gathering, respectively. *Source-driven* requirements gathering defines the requirements of a data warehouse using source data or a schema from an OLTP system. The benefits of using a source-driven requirements gathering approach are that minimal user time is required to start the project, and complete data is supplied since the existing data in the OLTP system provides the framework for the data warehouse. A disadvantage of this approach is that incomplete knowledge of user needs and reliance on only the OLTP database may not produce an adequate data warehouse schema. To alleviate this disadvantage we include an opportunity for user refinement in our design process. *User-driven* requirements gathering is a method based on investigating functions users perform. We focus on user needs as represented in a set of queries to be applied against the data warehouse. Having user-driven requirements prevents critical business needs from being overlooked. To illustrate the proposed algorithms, both source-driven and user-driven requirements are gathered from the TPC-H Benchmark [TPC99]. We treat the benchmark schema as an OLTP schema for source derivation and its queries as user requirements.

Section 2 discusses data models used to represent schemas and gives a brief overview of related work in data warehouse design. Section 3 presents and illustrates an automated approach to constructing candidate conceptual schemas for a data warehouse from an OLTP schema. Section 4 gives an algorithm for evaluating candidate schemas based on user requirements as well as a guideline for manual schema refinement. Section 5 discusses advantages and known limitations of our work along with topics for future work.

2. Related Work

The goal of creating a conceptual schema is

> to translate user requirements into an abstract representation understandable to the user, that is independent of implementation issues, but is formal and complete, so that it can be transformed into the next logical schema without ambiguities [TB99].

Requirements of a conceptual model for a data warehouse include a) providing a modeling construct to represent business facts and their properties, b) connecting temporal dimensions to facts, c) relating objects with their properties and associations, d) defining relationships between objects and business facts, and e) outlining dimensions and their respective hierarchies [TB99]. We

discuss our choice for a conceptual model and related automated approaches below.

2.1 Conceptual Model Selection

In the OLTP arena, Entity/Relationship-based models are most widely used, and in the OLAP arena dimensional modeling is most popular. An ER schema is a graphical representation of entities and their relationships to each other. ER-based models have been extended to include the dimensional functionality necessary in data warehousing [B99, FS99, TB99, HS00]. Dimensional models organize data based on business rules of an organization. Dimensional models are based on the idea that an organization's facts are central and the data unfolds around it. The most well-known dimensional model is the Star model [BH98]; other variations are the Snowflake model [BH98] and the Dimensional Fact Model [GM98a]. Further discussion about ER versus dimensional modeling [K95, R96b] and converting from an ER model to a dimensional model [K97, M98] is given elsewhere.

Conceptual and logical schemas for data warehouses can be built with ER-based or dimensional-based models. It is possible that the conceptual schema may be in one form and the logical another. For example, McGuff [M98] uses ER modeling for conceptual schemas and dimensional for the logical and physical designs, while Wu and Buchmann [WB97] do the opposite for schema creation. Table 1 gives a summary of models used by various authors for conceptual and logical schema creation. The Star model is popular for both conceptual and logical modeling, but may not be the best choice for end users since it does not show drill-down and roll-up paths [BH98]. We choose the ME/R model for conceptual schema creation and the Star model for logical schema creation. Translating from ME/R to Star form is straightforward; our algorithm and examples are not provided in this paper due to space limitations.

The ME/R (Multidimensional Entity-Relationship) model is similar to, but simpler than, the EVER and StarER models. Figure 1 is an ME/R example of a customer orders application. The central construct is a *fact node*, represented by a diamond, that contains business measures. A *level node*, represented by a rectangle, contains information about the measures. Each level or fact node has *attributes*, represented as ellipses, that are descriptors for this entity. These are the three main constructs that make up the ME/R model, and they are connected by various relationships. In the example, *Order* is the fact node. *Product*, *Customer*, *Day*, *Month*, *Week*, and *Year* are level nodes. A fact node is related to a level by a *dimension* edge that is an undirected line. The *has* edge connects facts or levels to their attributes and is also an undirected line. The *classification* edge is the relationship between levels. This is represented by a pitchfork-like symbol at one end of a directed line. All the levels of the fact node, related by classification edges, represent a *dimension* of the model. The dimension that represents the order date in our example is made up of a *Day* level, a classification of the *Month* level that in turn is a classification level of *Year*. The directed edge between the levels is important in that it shows hierarchical aggregation. This is especially useful when a level node can be a classification node for more than one other level as seen in the *Day* to *Week* relationship. In our example, we can roll-up from the *Day* level to the *Month* level, changing the level of aggregation shown for the fact attributes. *Week* is an alternate roll-up path from *Day*.

	Conceptual Design	Logical Design
ER	[M98]	[BH98]
EVER	[B99]	
StarER	[TB99]	
ME/R	[HS00]	
DWCDM	[FS99]	
MAC	[TK01]	
Star	[WB97, R96b, K96b, BH98, M94, M98]	[K96b, BE99, CD97, K97, BH98, M98]
Snowflake		[BE99, CD97, BH98]
DFM	[GR98, GR99]	

Table 1. Models for Conceptual and Logical Schemas

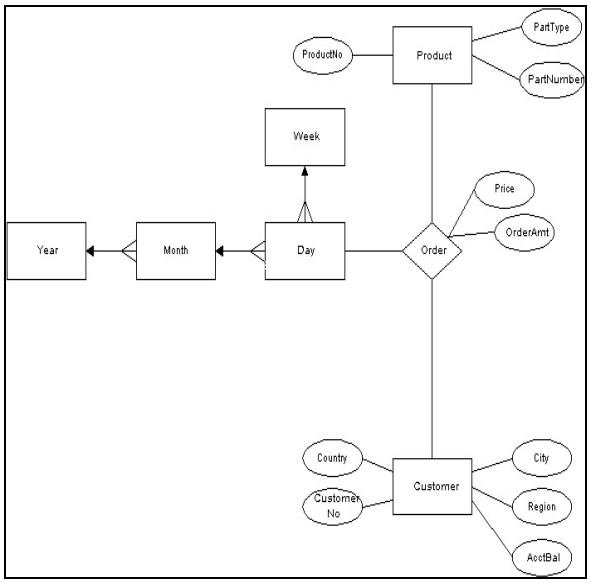

Figure 1. An Example ME/R Schema

With only three main constructs, the ME/R model is simple for users to read, and aggregation is shown through the hierarchy of levels representing a dimension. This model resembles the Star model, leading to ease of understanding by both users and logical modelers of the data warehouse. Details of additional ME/R modeling constructs are given elsewhere [HS00].

2.2 Automated Approaches to Schema Design

The initial determination of measures, facts, or events can prove to be the most difficult part of the design process, and is usually done manually. Different approaches to finding measures are suggested: (1) deriving the fact table of a Star schema by selecting the "many-to-many relationships in the ER model containing numeric and additive nonkey facts" [K97], (2) finding candidate measures by analyzing the business queries for data items indicating the performance of the business [BH98], (3) finding facts based on entities most frequently updated [GM98b], and (4) observing that "fact properties are usually numerical data, and can be summarized (or aggregated)" [TB99]. The third approach is implemented in a CASE tool [GR01]; we include the last approach in our algorithm in Section 3.

There have been some proposals for semi-automated or automated approaches to schema creation from OLTP systems. Automating logical schema creation and physical schema design have been proposed [TS97, CT98, GR98, BE99, TS99, S99, HL00, HS00, MK00]. None of the previous approaches include an automated mechanism for finding candidate measures. Our research is the first effort that addresses automation of creating an entire candidate conceptual schema from an OLTP schema, including the initial determination of facts or measures.

3. Conceptual Schema Creation

Our approach to schema generation has two basic premises. One is that numeric fields represent measures of potential interest to a business and the more numeric fields in an entity the more likely it is that the entity is an event or fact. The second premise is that the cardinality of a relationship determines how useful a related entity is to the schema being created. Any entity related with a many-relationship is of likely importance (and any of its entities may be as well), since it is potentially a higher level of abstraction.

Our algorithm for creating candidate conceptual schemas has five steps that result in candidate schemas centered around likely business events, with relationships to other entities forming the dimensions describing these events. Although there are instances where the events chosen have little or no meaning to users and additional commonly used calculations may need to be added, the bulk of the initial schema creation is automated. Schemas can be manually refined as described in Section 4.

Input to the algorithm is an ER schema represented here in table data structures. The steps of the algorithm are generalized as follows:

1. Find entities with numeric fields and create a fact node for each entity identified.
2. Create numeric attributes of each fact node, based on numeric fields in the entities.
3. Create date and or time levels (dimensions) with any date/time type fields per fact node.
4. Create a level (dimension) containing the remaining entity attributes (non-numeric, non-key, and non-date fields).
5. Recursively examine the relationships of the entities to add additional levels in a hierarchical manner (creating a dimension).

The output of the algorithm is a set of candidate ME/R schemas in tabular form. Four additional tables are created for use in logical schema creation that store fact node names for the various candidate schemas created, the names of each level of the fact nodes and the levels that are sub-levels of the level nodes, and the attributes of the facts and levels. These tables are not illustrated in this paper, only the graphical representations of candidate schemas are presented here.

The algorithm is given in Figure 2 with the corresponding steps labeled. The function calls are informally described here via an example using the TPC-H Benchmark schema [TPC99]. The TPC-H Benchmark is created specifically to benchmark query performance in a data warehouse environment, but here we treat the TPC-H schema as an OLTP schema since it resembles one in a normalized format. There are benefits from the decision to use the TPC-H benchmark. First, it is an industry benchmark example and thus not biased toward the schema creation algorithm. Second, order entry and/or part distribution is a common function of a wide range of businesses. Third, and most important for our purposes, a schema for source-driven requirements and queries for user-driven analysis are given. We reverse engineer an ER schema from the TPC-H schema and use it as input to our algorithm.

Step 1 orders the entities with numeric fields in descending order of number of numeric fields. Descending order is not necessary but the entities with the greatest number of numeric fields create candidate schemas that are generally better for answering user queries. By processing the entities in this order the candidates that are more likely to be useful are created first. In our example, the only numeric fields are of type decimal and integer. The result of this step is a list of tables in a ranked order of number of numeric fields, i.e., *LineItem* (5), *PartSupp* (2), *Part* (2), *Orders* (2), *Supplier* (1), and *Customer* (1), for a total of 6 candidate schemas.

Starting with the first entity, *LineItem*, we create an ME/R schema. The fact node is represented by a diamond shape, labeled *LineItem Event*, shown in Figure 3. This fact node becomes the focus of the candidate ME/R schema created on this iteration of the loop.

In Step 2, the numeric fields for *LineItem Event*, *L_LineNumber*, *L_Quantity*, *L_ExtendedPrice*, *L_Discount*, and *L_Tax* are added to the diamond as ellipses, indicating attributes, as shown in Figure 3.

Algorithm for Conceptual Schema Creation

```
Input Parameters:
        Table_Columns           // Table containing table name, column name, and column type
                                //      for every OLTP table.
        Table_Relations         // Table containing the OLTP schema relationships
In/Out Parameters:
        Fact_Node_Table         // Table defining the fact nodes of ME/R schemas and the
                                //      OLTP table name that is used to create the fact node.
        Fact_Attribute_Table    // Table defining the attributes of the fact nodes for the ME/R
                                //      schema.
        Level_Table             // Table defining the levels of the ME/R schema.
        Level_Attribute_Table   // Defines the attributes of the levels.
Variables: num_tables[]         // Array of table names from OLTP schema(s) with numeric
                                //      fields. Array is ordered in descending order of numeric
                                //      fields.
        fact_node               // Fact node name.
        num_field[]             // Array of numeric OLTP attribute field names.
        date_field[]            // Array of date OLTP attribute field names.
        other_field[]           // Array of OLTP non-key, non-numeric, non-date/time fields.

Method:
    num_tables[] := select_order_tables_numeric_fields (Table_Columns)                    (1)
    for each num_tables[j]
        fact_node := create_fact_node(num_tables[j], Fact_Node_Table)
        num_field[] := select_num_field (Table_Columns, num_tables[j])                    (2)
        for each num_field[m]
            create_fact_node_attribute (fact_node, num_field[m], Fact_Attribute_Table)
        end for loop
        date_field[v] := select_date_field  (Table_Columns, num_tables[j])                (3)
        if isempty(date_field[]) then
            create_review_levels(fact_node, Level_Table)
        else
            for each date_field[v]
                create_date_time_level (fact_node, date_field[v], Level_Table)
            end for loop
        end if
        if exists other_field_in_OLTP_table (Table_Columns, num_tables[j])                (4)
            create_level (fact_node, num_tables[j], Level_Table)
            other_field[] := select_other_fields (Table_Columns, num_tables[j])
            for each other_field[a])
                add_fields_to_level (fact_node, other_field [a], num_tables[j], Level_Attribute_Table)
            end for loop
        end if
        Walk_Relationships (num_tables[j], fact_node, Table_Columns, Table_Relations,     (5)
                            Level_Table, Level_Attribute_Table)
    end for loop
end algorithm
```

Figure 2. Algorithm for Conceptual Schema Creation

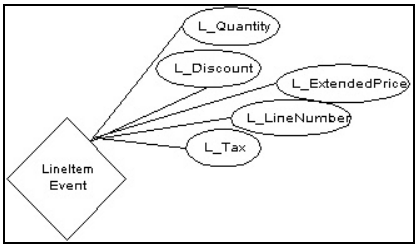

Figure 3. LineItem Event with Numeric Attributes

Next, as part of Step 3, we identify any date fields of this entity. *LineItem* has three date and/or time fields: *L_ShipDate*, *L_CommitDate*, and *L_ReceiptDate*. These fields become the date/time levels or dimensions of our ME/R schema. At this point in the automated process, the granularity of the date and time dimensions are unknown; user refinement is needed to determine how to represent the levels that make up the date dimension. In ME/R diagrams, dimensions are represented by a collection of

levels in a hierarchical manner. Instead of representing the date and time dimensions as rectangles normally used to represent levels in an ME/R diagram, we introduce a new notation, a hexagon, to indicate a portion of the schema where user refinement of date/time granularity is needed. The ME/R schema created so far as given in Figure 4.

Step 4 is only processed if there are attributes remaining in the entity that are not yet processed (i.e., not key, numeric, or date fields). The remaining fields are generally text fields. If there are text fields as part of this entity then a level (dimension) node is created with this entity's name. The level nodes are symbolized by rectangles. Each remaining field becomes an attribute of the level node. For our example, *LineItem* has five such attributes: *L_ReturnFlag*, *L_LineStatus*, *L_ShipInstruct*, *L_ShipMode*, and *L_Comment*. The new level, *LineItem,* and its attributes are in Figure 5.

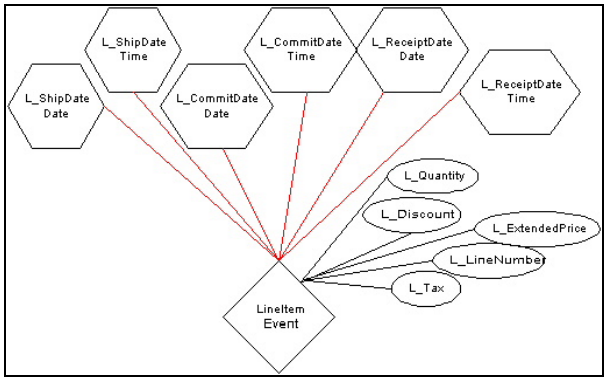

Figure 4. Date Levels of LineItem Event

Step 5 is the most complicated step, called *Walk_Relations,* where recursive traversal of relationships is conducted. The rest of the TPC-H schema entities (*Customer, Orders, Nation, Region, Supplier, Part,* and *PartSupplier*) may be used here. In this step, the relationships of the fact node/event entity are evaluated. Every connected entity is made into a level node with its attributes. If this level entity is part of the many side of the relationship, its relationships are also evaluated and processed by calling the sub-procedure again. This step identifies the relationship names of interest; *Order_LineItem* and *PartSupp_LineItem*; these relationships are processed one at a time within the sub-procedure loop.

The relationship names (*Order_LineItem* and *PartSupp_LineItem*) are used to find the entities that make up dimensions for *LineItem Event*. Starting with the *Order_LineItem* relationship name the first relationship is found, with the entity *Orders*. A level node is created for *Orders* along with its attributes.

At this point we have added our first level not derived from the *LineItem* entity. We now need to determine if any of the OLTP entities related to *Orders* should be included in this schema. We use relationship cardinality to indicate how many entity instances are related to another entity instance by the relationship. For the *Orders_LineItem* relationship for the *LineItem* entity, the cardinality is on the many side of the relationship with *Orders*. In other words, one *Order* can have many *LineItem* instances. *Walk_Relations* is recursively called with *Orders* now being the parameter passed. For the same reason, *Customer*, *Nation*, and *Region* become levels in the hierarchy, shown in Figure 5. In the example, all of the relationships are many-to-one, thus we continue to recursively call the *Walk_Relations* procedure. If we encounter a relationship that is not many-to-one or many-to-many, we do not include it in the schema. If the cardinality of *Orders* to *Customer* had not been a many-to-one relationship, the *Customer*, *Nation*, and *Region* entitities would not have been visited and added to the schema. As is, the sub-leveling ends with *Region* because it has no relationships not already evaluated.

The last relationship with *LineItem* is to *PartSupp*. This is a many-to-one relationship, so the relationships of *PartSupp* are recursively processed as well. *PartSupp* is a little different from other entities in this example because it has multiple sub-levels. The other difference is that the *Nation* and *Region* levels already exist from the relationship to the *Customer* level so that we do not have to duplicate part of the schema. The first complete candidate schema (for one iteration of the outermost loop in 2) is given in Figure 5.

Now that we have completed the first iteration of the algorithm, we create another candidate schema in the second iteration. Steps 1 through 5 are reapplied with *PartSupp* as the event entity. Since the *PartSupp* entity is on the one-side of the relationship with the *LineItem* entity, we do not delve any lower on that dimension path. The completed candidate schema is shown in Figure 7. This candidate schema has fewer levels in its dimensions. Only the relationship with the *Supplier* table leads to sub-levels. Another new symbol, the cloud shape, is introduced to denote that a level may be needed and is not automatically derived from the OLTP schema. In data warehouses, events tend to be measured by date or time periods. Although the *PartSupp* entity does not have any date/time fields, the final design in the data warehouse probably will. This dimension may be created by the grain requirements of the user for analysis or by the refresh requirements used to capture data from the OLTP system to store into the data warehouse. Dimensions represented by the cloud are added to any fact node with no date or time fields.

Following the sequence of the algorithm, the other entities with numeric fields are examined and candidate schemas are produced. These are given in Figures 7 through 10. Not all of the generated conceptual schemas may prove to be useful to the user; the 6 candidate

conceptual schemas automatically created by our algorithm are evaluated against user requirements in the next section.

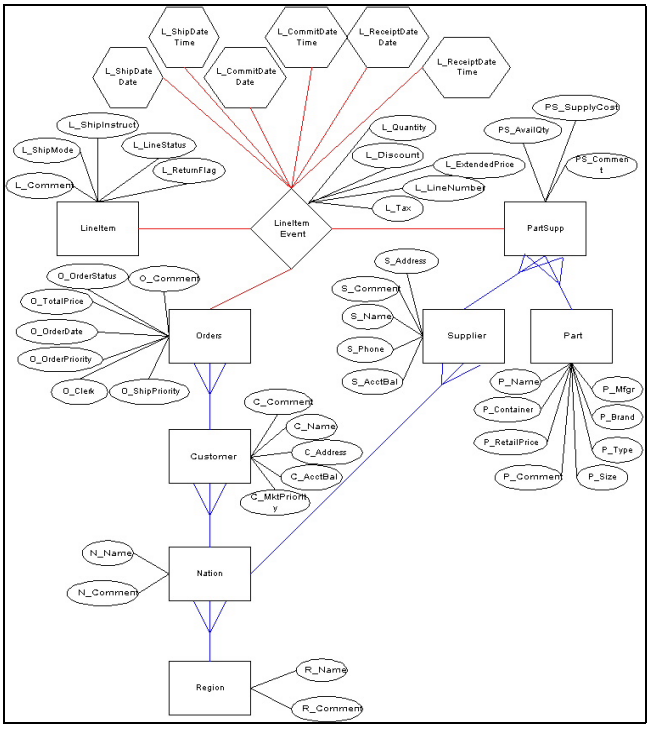

Figure 5. Candidate Schema 1: LineItem Event

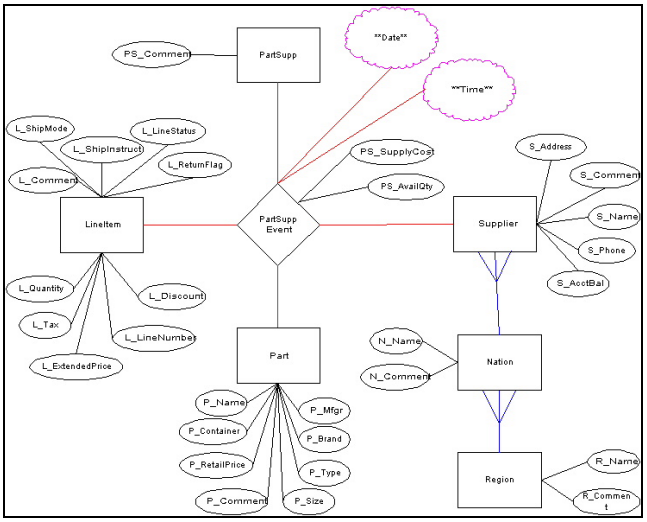

Figure 6. Candidate Schema 2: PartSupp Event

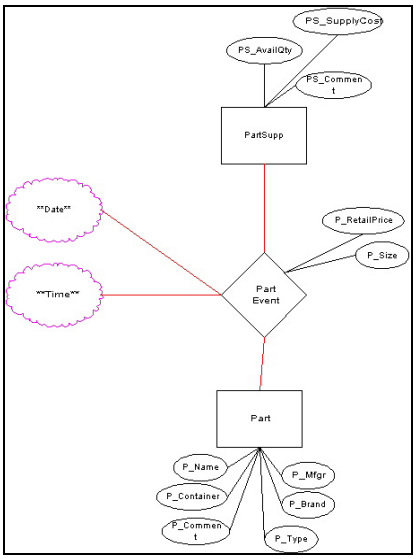

Figure 7. Candidate Schema 3: Part Event

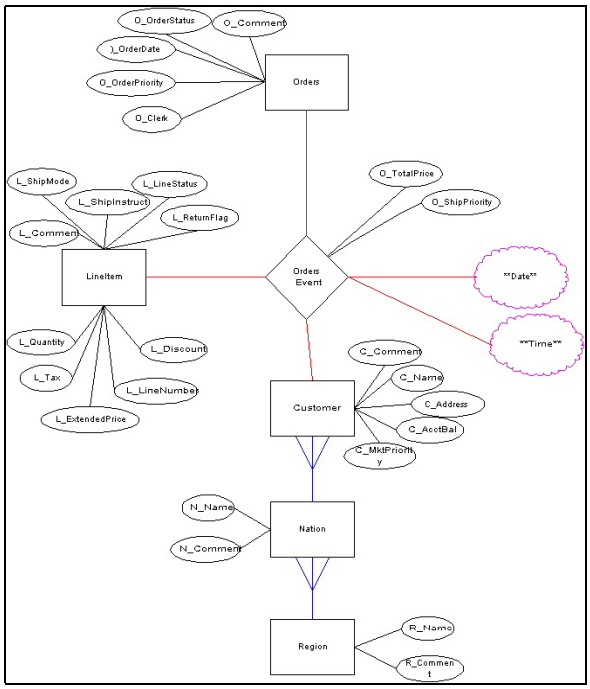

Figure 8. Candidate Schema 4: Orders Event

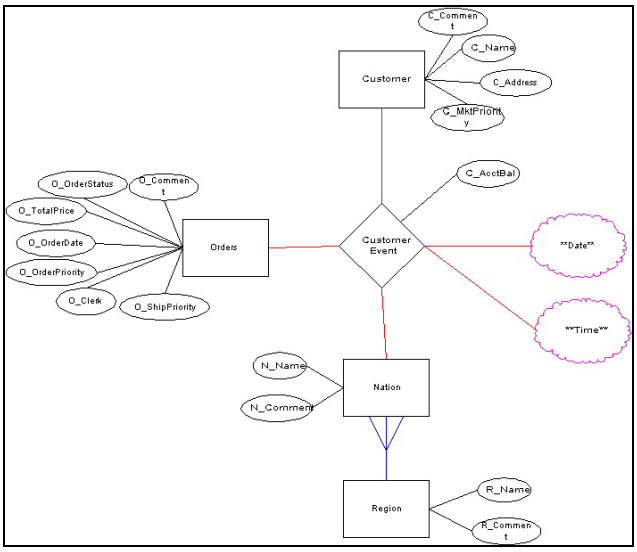

Figure 9. Candidate Schema 5: Customer Event

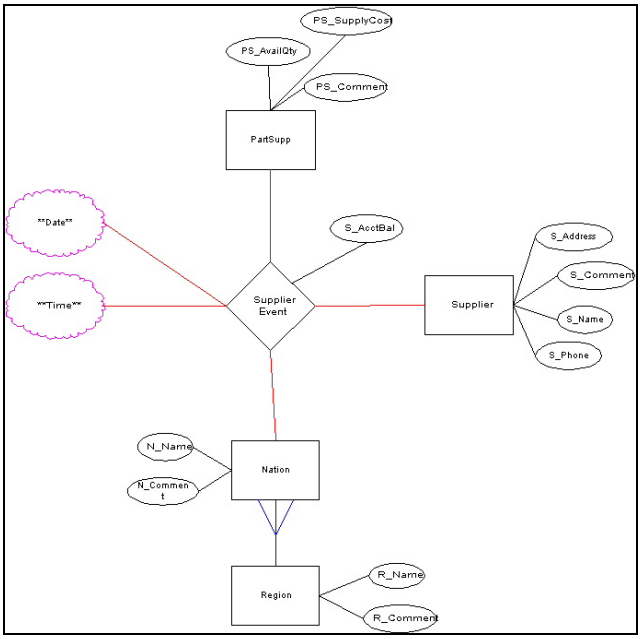

Figure 10. Candidate Schema 6: Supplier Event

4. Candidate Schema Selection and Refinement

The TPC-H Benchmark queries are used here to evaluate which candidate schemas best meet users' needs. There are two aspects of a query that are used to determine if a candidate schema can answer a query: the tables in the FROM clause and the numeric fields in the SELECT clause. If a candidate schema does not contain the table(s) in the FROM clause it cannot answer the query because the fields of that table are not in the schema either. It is unnecessary to check for every field in the query SELECT statement because the candidate schema generation algorithm dictates that every field in a table of the OLTP system is in the schema. The numeric fields from the SELECT clause are essentially the measures that need to be attributes of the fact node to answer the query. In order to compare the candidate schemas for satisfying the queries, we create a table. For our example, Table 2 represents the 22 queries and the 6 candidate conceptual schemas. In the table, "X" shows that the candidate schema completely meets the query requirement, and "P" means that the schema partially answers the query because a numeric value of interest is not in a fact node but is in one of the dimensions. A blank entry means that the query is not answerable by this schema.

As an example of how the evaluation algorithm works, consider query Q1:

```
SELECT l_returnflag, l_linestatus, Sum(l_quantity) as
      sum_qty, Sum(l_extendedprice) as sum_base_price,
      Sum(l_extendedprice * (1 – l_discount)) as
      sum_disc_price, Sum(l_extendedprice * (1 – l_discount)
      * (1 + l_tax)) as sum_charge, Avg(l_quantity) as
      avg_qty, Avg(l_extendedprice) as avg_price,
      Avg(l_discount) as avg_disc, Count(*) as count_order
FROM          lineitem
WHERE   l_shipdate <= date '1998-12-01' – interval
      '[DELTA]' day(3)
GROUP BY      l_returnflag, l_linestatus
ORDER BY      l_returnflag, l_linestatus;
```

Starting with Candidate Schema 1, since *LineItem Event* is the fact node and all numeric fields in the SELECT statement are attributes of the fact node, this schema satisfies the query (an "X" entry). Doing the same steps for Candidate Schema 2, *PartSupp Event*, *LineItem* is again included in the schema, however, the numeric fields are not in the fact node but are in a level node (a "P" entry for partial match). Candidate Schemas 3 through 6 are processed in this manner with only Candidate Schema 4, *Orders Event*, found to be a partial answer to the query. Evaluating the remaining queries for each schema in the same manner yields Table 2.

The evaluation algorithm identifies whether various candidate schemas meet the requirements of the user queries. The resulting table can be automatically processed using a graph covering algorithm or manually evaluated to determine which candidate schemas to keep and which can be discarded. We describe a manual process to illustrate tradeoffs between candidate schemas and motivate our refinement guidelines.

There are two types of queries that yield results that may require manual refinement before evaluation of the table: a query with numeric SELECT fields from multiple OLTP entities, and queries with no numeric fields. Any query that has numeric SELECT fields from more than one OLTP table results in only a partial solution to the query. Q10 is an example of this. Query Q10 has three numeric fields, *L_ExtendedPrice, L_Discount,* and *C_AcctBal*. Because these fields span different OLTP entities, none of our candidate schemas can directly answer the query. Any candidate schema that meets the FROM criteria evaluates to a "P" for partial. Manual

evaluation is needed to determine if the schema can indeed answer the query adequately or whether minor changes to the candidate schema may allow it to completely answer the query. One such change may be the movement of a numeric field from a level to a fact node.

For queries with no numeric fields in the SELECT statement, the schemas that include all the entities in the FROM statement of the query evaluate as direct answers (i.e., "X") to the query requirements. This evaluation of the candidate schemas is correct because the queries only require textual or date type information. With no requirement for numerical data all data can come from any candidate schema with the appropriate levels. Query Q12 is an example of this type of query. The FROM section of Query Q12 has two tables, *Orders* and *Lineitem*. Candidate Schemas 1 and 4 have data from these tables and both meet the user requirements of the query.

	1	2	3	4	5	6
Q1	X	P		P		
Q2	P	P				
Q3	X			P		
Q4	X			X		
Q5	X					
Q6	X					
Q7	X					
Q8	X					
Q9	P					
Q10	P			P		
Q11	P	X				P
Q12	X			X		
Q13	X			X	X	
Q14	X	P				
Q15	X	P				
Q16	P	X				
Q17	X	P				
Q18	X			P		
Q19	X	P				
Q20	X	P				
Q21	X					
Q22	P			P	X	

Table 2. Candidate Schema Evaluation

Table 2 gives us several possibilities for schemas that meet our data warehouse requirements as defined by the user queries. Candidate Schemas 3 and 6 can be eliminated since they do not answer any queries that cannot be answered by other candidate schemas. Candidate Schema 4, while satisfying many queries, does not satisfy any that are not satisfied by Candidate Schema 1, and can thus be discarded. Candidate Schema 2 is promising; it is a stronger schema than Candidate Schema 1 for a few of the queries ("X" instead of "P" for Q11 and Q16). Candidate Schema 5 may not be needed, but it answers Q22 which can only be partially answered by Candidate Schema 1. Further analysis is needed to decide if Candidate Schema 5 can be dropped or is needed as part of the data warehouse for specific purposes. The analysis may result in modifications to the query or modifications to Candidate Schema 1. Additional steps to further refine schemas based on user input or knowledge of a designer are discussed below.

Most of the refinement steps require little knowledge of the existing OLTP database schema; knowledge of the user needs is more important. We identify seven manual steps for conceptual schema refinement (numbering starts with 6 because the automated conceptual design steps ended with 5).

6. If user queries are known, eliminate unnecessary candidate schemas.
7. Inspect measures in each fact node. Are they indeed measures or attributes?
8. What is the necessary grain of date/time information?
9. Are other calculated fields necessary?
10. Can schemas be merged?
11. Can any fields be eliminated as not necessary?
12. Is there any data required that did not exist in the OLTP database?

There may be other additions dictated by need and situation but the steps above address many of the changes that need to be made manually. We provide some examples, but these examples rely on our interpretation of the TPC-H information. The following is for illustrative purposes only and is not meant as a comprehensive solution for finishing the automatically generated schemas.

Because user queries to be answered by the data warehouse are known, we are able to eliminate several of the candidate schemas in Step 6. The selection of the candidate schemas is considered a manual step, although it is facilitated by the evaluation table, because the determination of which schemas to keep is subjective and should be considered by a designer.

Step 7 analyzes the measures of a fact node. Some numeric fields are actually attributes (descriptors) rather than measures. Fields such as width, length, and other physical properties are generally attributes used as limiting criteria in a query rather than as measures of the business. In *LineItem Event*, the *L_LineNumber* field is likely used to sequence the items that make up the order. *L_LineNumber* is non-additive, and would not be summed for each item of an order, which is a good indicator that it is an attribute rather than a measure. As part of Step 7, the *L_LineNumber* field is moved from the fact table and placed in the level node or dimension table.

Step 8, determining the grain of data necessary, resolves date and/or time measures (illustrated by hexagons and clouds in our ME/R schemas). For example, if the user wants to see part inventory levels at the various suppliers on a weekly basis, a level representing week is

necessary. Alternate rollup paths for dates could be added at this point, as well.

In Step 9, any calculated fields used on a regular basis may be added to the schema so that the calculation is done at the warehouse rather than by end user tools. For example, in many of the TPC-H queries, *LineItem* measures are combined to produce a revenue field (*sum(L_ExtendedPrice * (1 – L_Discount))*. Another important aspect of Step 9 is adding count fields; this can be done in queries or a counter field could be added to the fact node.

Step 10 is based on the merging of facts and dimensions as described by Ballard et al. [BH98] and Kimball [K96a, K98]. The merging of either facts or dimensions requires knowledge of not only user requirements but also of the OLTP system. If any schemas have a common dimension (a dimension having all of the same levels as a dimension in another schema) they can be merged into fact trees. By sharing a dimension, users gain a more complete view of their order data in one schema. Schema merging relies on the date level granularity being the same and the facts sharing a date.

The second form of schema merging is fact merging, where facts from two candidate conceptual schemas can be merged to a single schema. Merging some of our candidate schemas (or parts of) may allow for additional queries to be answered. This may be the case where a table is not part of a many relationship to another table and some information from that table fulfills a query. In our evaluation table (Table 2), Candidate Schema 4 partially answers Q1, Q3, Q10, Q18 and Q22. If we decide to add the numeric fields from *LineItem* to the *Order Event* we could answer Q1, Q3 and Q18 completely and would now have more of the numeric SELECT fields from Q10 and Q22. This form of candidate schema merging is more likely when a one-to-one relationship exists. Automating the identification of possibilities for merging is a topic for future work.

Step 11 removes fields that do not hold information of interest. In a few entities, such as *Region*, it is unlikely that the comment field holds information of value in an OLAP scenario and can be removed.

The last manual step, Step 12, addresses instances where user requirements reference data that is not stored in the enterprise database. For example, a car lot may see a decline is sales on rainy days and wants to track the weekly sales based on weather conditions. If the OLTP system does not have data about the weather, an outside source of information is integrated into the data warehouse. Because the data is not in the OLTP data sources we use as input, this step is not automated in our algorithm.

Schemas selected based on satisfying user queries and refined based on manual guidelines provide a basis for physical data warehouse design and implementation.

5. Discussion

This paper contributes an algorithm to derive data warehouse candidate conceptual schemas from an OLTP schema. Our algorithm uses numeric fields and relationships between entities as the basis to create ME/R schemas. The basic concept can be used with most models such as UML, ER, and relational, as long as the source schema contains some notion of entities, their relationships, and relationship cardinalities. The algorithm is applicable for any data model where the data types can be partitioned into numeric, date/time, and textual data types. A second contribution is an algorithm to evaluate candidate conceptual schemas using user queries. Candidate conceptual schemas can be selected and possibly modified using our manual refinement guidelines to fulfill user needs. The algorithms are illustrated using the TPC-H Benchmark schema and queries.

A potential limitation of the work is that it requires an enterprise-wide initial schema and queries that span business operations. If these are not readily available, then manual requirements gathering is required before our algorithms could be applied; however, this would have to be done regardless of the chosen design methodology. Alternately, data marts could be created using our algorithms if enterprise-wide information is not available.

Enhancements to the creation algorithm include:

- allowing specification of count measures and measures that are stored in non-numeric fields prior to automatic schema generation since this cannot be automated and is currently only considered in post-processing manual refinement,
- allowing 1-to-1 relationships to create merged fact nodes,
- considering the impact of many-to-many relationships [PJD94, SMRE01], and
- expanding its scope to include historical, summarized, and consolidated data rather than just OLTP data.

Extensions to the evaluation algorithm include identifying opportunities for schema merging based on queries.

While we have made headway in the automation of conceptual schema generation and evaluation, there is additional work to be done to evaluate and implement the proposed algorithms. Our work is illustrated using the TPC-H schema and queries, but additional case studies using other application scenarios are needed. Our work provides a basis for implementation in a software tool, but that remains for future work.

References

[B99] L. Bækgaard, "Event-Entity-Relationship Modeling in Data Warehouse Environments," *Proceedings of the ACM DOLAP99 Workshop*, Missouri, November 2-6, 1999.

[BH98] C. Ballard , D. Herreman, D. Schau, R. Bell, E. Kim, and A. Valencic, *Data Modeling Techniques for Data*

Warehousing, IBM Redbook, IBM International Technical Support Organization, Feb-26-1998, ISBN No. 0738402451.

[BE99] M. Boehnlein and A. Ulbrich-vom Ende, "Deriving Initial Data Warehouse Structures from the Conceptual Data Model of the Underlying Operational Information Systems," *Proceedings of the ACM DOLAP99 Workshop*, Missouri, November 2-6, 1999.

[CD97] S. Chaudhuri and U. Dayal**,** "An Overview of Data Warehousing and OLAP Technology," *ACM SIGMOD Record*, Vol. 26, No. 1, pp. 65-74, March 1997.

[CT98] L. Cabbibo and R. Torlone, "A Logical Approach to Multidimensional Databases," *Proceedings of the EDBT Conference,* 1998.

[FS99] E. Franconi and U. Sattler, "A Data Warehouse Conceptual Data Model for Multidimensional Aggregation," *Proceedings of the Intl. DMDW Workshop*, Heidelberg, Germany, June 14-15, 1999.

[GM98a] M. Golfarelli, D. Maio, and S. Rizzi, "The Dimensional Fact Model: a Conceptual Model for Data Warehouses," *International Journal of Cooperative Information Systems*, Vol. 7, Nos. 2 and 3, pp. 215-247, 1998.

[GM98b] M. Golfarelli, D. Maio, and S. Rizzi. "Conceptual Design of Data Warehouses from E/R Schemes," *Proceedings of the 31st Hawaii International Conference on System Sciences (HICSS-31)*, Vol. VII, Kona, Hawaii, pp. 334-343, 1998.

[GR98] M. Golfarelli and S. Rizzi, "A Methodological Framework for Data Warehouse Design," *Proceedings of the ACM DOLAP98 Workshop*, Washington, D.C., pp. 3-9, 1998.

[GR99] M. Golfarelli and S. Rizzi, **"**Designing the Data Warehouse: Key Steps and Crucial Issues," *Journal of Computer Science and Information Management*, Vol. 2, No. 1, pp. 1-14, 1999.

[GR01] M. Golfarelli and S. Rizzi, "WanD: A CASE Tool for Data Warehouse Design," *Demo Proceedings of the 17th International Conference on Data Engineering (ICDE 2001)*, Heidelberg, Germany, pp. 7-9, 2001.

[HL00] B. Husemann, J. Lectenborger, and G. Vossen, "Conceptual Data Warehouse Design," *Proceedings of the Intl. Workshop on DMDW 2000,* Stockholm, Sweden, June 5-6, 2000.

[HS00] K. Hahn, C. Sapia, and M. Blaschka, "Automatically Generating OLAP Schemata from Conceptual Graphical Models," *Proceedings of the ACM DOLAP 2000 Workshop*, pp 9-16, 2000.

[K95] R. Kimball, "Data Warehousing Gets the Data Out," *DBMS Magazine,* Sept. 1995.

[K96a] R. Kimball, "Factless Fact Tables," *DBMS Magazine,* Sept. 1996.

[K96b] R. Kimball, *The Data Warehouse Toolkit*, John Wiley and Sons, Inc., New York, 1996.

[K97] R. Kimball, "A Dimensional Modeling Manifesto," *DBMS Magazine*, Aug. 1997.

[K98] R. Kimball, "Bringing Up Supermarts," *DBMS Magazine,* Jan. 1998.

[M97] S. Mahajan, "Building a Data Warehouse Using Oracle OLAP Tools," Oracle Technical Report, *ACTA Journal*, Sept. 1997.

[M98] F. McGuff, "Designing the Perfect Data Warehouse," 1998.
http://members.aol.com/fmcguff/dwmodel/index.htm

[M94] MicroStrategy Incorporated, "Relational OLAP: An Enterprise-Wide Data Delivery Architecture," white paper, 1994.
http://www.dmreview.com/master.cfm?NavID=61&WhitePaperID=136

[MK00] D. Moody and M.A.R. Kortink, "From Enterprise Models to Dimensional Models: A Methodology for Data Warehouse and Data Mart Design," *Proceedings of the Intl. Workshop on DMDW 2000,* Stockholm, Sweden, June 5-6, 2000.

[PJD94] T.B. Pedersen, C.S. Jensen, and C.E. Dyreson, "A Foundation for Capturing and Querying Complex Multidimensional Data," *Information Systems,* Vol. 19, No. 4, pp. 33-54, 1994.

[R96a] N. Raden, "Modeling the Data Warehouse," Archer Decision Sciences, Inc., 1996.
http://netmar.com/~nraden/iw0196_1.htm

[R96b] N. Raden, "Technology Tutorial – Modeling A Data Warehouse – Value to an Organization Means Turning Data into Actionable Information," *InformationWeek*, Jan. 1996, Issue 564. http://www.techweb.com

[S99] H. Schouten, "Analysis and Design of Data Warehouses," *Proceedings of the Intl. Workshop DMDW'99*, Heidelberg, Germany, June 14-15, 1999.

[SC99] J. Srivastava and P. Chen, "Warehouse Creation — A Potential Roadblock to Data Warehousing," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 11, No. 1, pp. 118–126, January/February 1999.

[SMRE01] I.-Y. Song, W. Rowen, C. Medsker, and E. Ewen, "An Analysis of Many-to-Many Relationships between Fact and Dimension Tables in Dimensional Modeling," *Proceedings of the Intl. Workshop on DMDW'01,* Interlaken, Switzerland, June 4, 2001.

[TK01] A. Tsois, N. Karayannidis, and T. Sellis, "MAC: Conceptual Data Modeling for OLAP," *Proceedings of the Intl. Workshop on DMDW 2001,* Interlaken, Switzerland, June 4, 2001.

[TS97] D. Theodoratos and T. Sellis, "Data Warehouse Configuration," *Proceedings of the 23rd VLDB Conference*, Greece, 1997.

[TS99a] D. Theodoratos and T. Sellis, "Designing Data Warehouses," Data and Knowledge Engineering (DKE), *Elsevier Science*, Vol. 31, Oct. 1999, pp. 279 – 301.

[TPC99] Transaction Processing Performance Council, *TPC Benchmark™H (Decision Support) Standard Specification Revision 1.3.0 (TPC-H)***,** June 1999. http://www.tpc.org/

[TB99] N. Tryfona, F. Busborg, and J. G. B. Christiansen, "StarER: A Conceptual Model for Data Warehouse Design," *Proceedings of the ACM DOLAP99 Workshop*, Missouri, November 2-6, 1999.

[WB97] M.C. Wu and A.P. Buchmann, "Research Issues in Data Warehousing," *Intl. Conference on Databases in Office, Engineering and Science (BTW'97)*, Ulm, Germany, March, 1997.