# The Translation of Star Schema into Entity-Relationship Diagrams

**Michael Krippendorf and Il-Yeol Song**

College of Information Science and Technology
Drexel University, Philadelphia, PA 19104
E-mail: krippemp@duvm.ocs.drexel.edu
songiy@dunx1.ocs.drexel.edu

**Abstract.** The star schema is widely accepted as a proper underlying table structure for data warehouses, but enterprise data is frequently defined in terms of entity-relationship diagrams. A key issue for data warehouse designers is how to move from legacy OLTP designs into the star schema. In this paper, we examine the semantics and constraints of the star schema in some of its different forms and translate them into analogous ERDs. These different forms include the simple, multi-star, and snowflaked star schema, and several variations on facts and dimensions such as associative, multple fact and outboard dimension tables. We outline transformation rules for each case and show a series of examples. We believe that these heuristics can be used in mapping and maintaining consistency between OLTP databases and data warehouses.

## 1. Introduction

This paper examines the translation of star schemas into entity-relationship diagrams (ERD) [1]. A commonly used tool, entity-relationship diagramming is principally enlisted in the development of conceptual models for on-line transaction processing (OLTP) database management systems (DBMS). It is the tool of choice for relational database modeling, and it has also been applied to hierarchical, network, and to object-oriented database design. Further, it is the source from which table structure is translated. It is the defacto standard among DBMS semantic modeling techniques.

With the advent of data warehousing--the use of static and large scale read-only databases to support complex queries of decision support systems (DSS)--a new data model and a requisite table structure have evolved [4]. This model which consists of a simplified table structure minimizing joins and speeding the execution of queries is referred to as the star schema [7, 9].

This is a model used increasingly for the logical design of data warehouse table structures and in support of the mode of high data volume complex queries known as on-line analytical processing (OLAP).

We now consider two points:

1. Because sources of transactional data are most likely to come from databases of the relational type, it is likely that they have been modeled using ERD, thus providing a ready basis for translation into star schema, if a set of transformation rules or an algorithm to guide this task can be developed.

2. Because a large portion of the time spent in data warehousing is typically spent in the modeling stage, applying the widely employed and standardized technique of ERD could reduce development time. Further, if this process could be automated on the basis of transformation rules, development time might be reduced considerably.

There would seem to be value, then, in relating ERD to the foundational table structure of a rapidly growing database technology like data warehousing, viz., to star schema. An exploration of the first component of this translation process is the topic of this paper.

## 2. Distinct Models for Distinct Applications

For OLTP, where requirements include rapid, highly structured repetitive processing--look-ups, updates, deletes--on a continually changing and ever increasing store of transactions, the data model is typically characterized by minimal redundancy and maximal data integrity implemented through numerous highly normalized tables related through a comparatively complex schema. On the basis of the relational model, SQL queries must be relatively simple, often predefined and tested to generate output which accurately addresses the demands of business operations. Such demands might involve queries like the following. “List the customers who have a savings account with a balance greater than $5000 at each of our branch banks.” Or, “Calculate the average daily balance of all customers who have checking accounts at the downtown branch.”

In contrast, on-line analytical processing (OLAP) is designed to provide summary views of transactions according to user-defined business dimensions from the static stores of the data warehouse. Data may be historical and/or aggregate in nature. Queries are frequently ad hoc and unstructured and designed to address questions of a predictive or prospective character, examples of which might include the following: "What type of behavior characterizes customers who have closed savings accounts within less than one year over the period of the last five years?"[7]. Or, "Given the rates of saving and transaction history for all current account holders, how many of those account holders would be prospects for certificates of deposit, and of what duration?" Such queries tend to draw upon large volumes of data. Appropriate SQL statements addressed to a traditional data model would be complex, involving multiple selects, joins, and comparisons--and time consuming. In the OLAP environment, however, queries must be timely enough to permit the analyst to pursue a train of thought in an interactional manner. In order to respond to such demands, the data warehouse is best configured according to the star schema.[1] This structure reduces the multiple and often complex joins required to provide the same information from a relational design. Processing load and time can be reduced while providing information appropriate to the reporting objectives.

### 2.1 Star Schema Defined

The large and complex queries required in OLAP are best supported through a modeling technique that simplifies table structure and minimizes table joins. The *star schema*, so named for its radial design is such a model [9].

---

[1] Alternatively, it may be realized as a multidimensional database. MDDBs store transactional data in summary cross-tabular form. Such a topic, though related, is not addressed in this paper.

The star schema is the underlying model of data warehouse design in which a central fact table or tables containing quantitative measures of a unitary or transactional nature (such as sales, shipments, holdings, or treatments) is/are related to multiple dimensional tables which contain information used to group and constrain the facts of the fact table(s) in the course of a query. Fact and dimension tables are directly related, thus addressing the *pair-wise join problem*.

The pair-wise join problem is one inherent in the relational model that forms the basis for OLTP systems where efficiency and data integrity are maintained through many normalized tables. Joins between tables to answer queries are optimized for joins between two tables. Complex queries must often bridge multiple tables in circular paths, and often there are multiple paths to reference all of the data needed to answer a question. When this occurs, ambiguities as well as processing inefficiencies can result. That is, depending on the series of joins specified in the SELECT statement, different information can be returned in answer to what appeared to be the same formulation of a query. For the purposes of OLAP, this problem is redressed by use of the star schema which simplifies table structure and join paths through its table definitions, radial design, and differential treatment of table normalization [8, 10].

The *fact or major table* is the core table or sets of tables which form the center of a star schema. It contains multiple data items referred to as *facts*, quantitative measures of business activity or holdings. These facts consist of numeric data of a continuous and additive or "semi-additive" kind [7]. Examples of facts are amounts of sales, hours, numbers of units, etc. These facts can be constrained or grouped by attributes of dimension tables-- tallied, percentaged and compared. The unit of analysis, the granularity of the table, is a fundamental design consideration for fact tables and forms the basis for the aggregation of such facts--whether they are unitized on daily, weekly, or other summary levels, for instance. In the fact table of the simple star schema,

unique identifiers to records are provided by a primary key, a composite of the foreign keys related to a series of dimension tables which are arranged radially around the fact table (Fig. 1). Referential integrity is preserved through these constraints. Because of their transactional and frequently historical nature, fact tables typically include very large numbers of records.

The *dimension or minor tables* consist of a single primary key and numerous data items or attributes which are deemed useful to an organization for grouping and constraining the facts. These dimensions are usually those of time, location, or constituency, and could include periods, markets, regions, stores, customers, sales force, and products, among others. Dimensions may have multiple implicit hierarchies, e.g., a product dimension might contain information relative to package, brand, category and department, each of which could be seen as a subset of data with a different set of data dependencies[7]. Because of this hierarchy of dimensions and numerous attributes, dimension tables tend to be very wide, while containing relatively few records [7]. These structural characteristics are sometimes seen as the basis for modifying table design. Examples of star schemas are shown in Figures 1 through 13.

### 2.1.1 Variations on the Dimension Table

Variations on the structure of the dimension table result from normalizing dimensions (Figs. 7, 9, 11). Typically, at least in smaller data warehouses, dimension tables are highly denormalized while the fact table is highly normalized. The secondary dimension tables, variously referred to as outboard, outrigger, or outbound tables, result from normalization of dimension tables. Embedded dimensions (groups of functionally dependent attributes) are extracted into secondary tables, which contribute a foreign key to the primary dimension table. When this process is carried to its logical extreme, the result is the *snowflake schema* (Fig. 9).

The snowflake schema is a variation on the simple star schema in which all dimension tables are systematically normalized to 3NF to extract embedded dimensions of the dimensional hierarchy into a lattice of separate outboard tables [9]. Fact tables remain unaffected. Snowflake star schema improve query efficiencies for front-end OLAP tools which can exploit them [7].

### 2.1.2 Variations on the Fact Table

While the simplest case consists of a single fact table with multiple dimension tables, variations of the fact table also occur. Multiple fact tables result when data of different levels of aggregation or different sets of facts must be maintained in the data warehouse. Tables may be aggregated to reflect different levels of granularity such as unit sales, unit sales per week or per quarter, etc. Aggregation offers control of the size and speed of queries requiring the various levels of analysis.

Families of star schemas occur when there are multiple fact tables of different design sharing some, but not necessarily all dimension tables. Such designs are usually called for when distinctly different types of information must be kept at the fact level.

An associative fact table is a fact table which may be used to resolve a M:N relationship between two dimension tables (Fig 5.). Its nature as a fact table is defined by 1) its role as the relationship associating more than one dimension table, and 2) the inheritance of its primary key, the composite of the foreign keys contributed by each of the dimension tables.

While the variations summarized above are more commonplace than the idealized model of a single fact table surrounded by a few dimension tables, they all share the same key structure, that of the simple star schema in which the primary key of the fact table is the composite of the foreign keys of all related dimensions. But more complex relationships are possible.

### 2.1.3 The More Complex Case: Multi-Star Tables

The complex case or multi-star schema is one in which the concatenated set of foreign keys contributed by each of the related dimension tables does not uniquely identify tuples in the fact table(s). In such cases, additional keys must be added to provide a unique key for each record. Poe offers the example of a sales transaction database where the level of granularity is the individual item sale (Fig. 11). The keys of two dimension tables, SKU and Store, are not sufficient in combination to uniquely identify an individual transaction, because multiple items of the same SKU could be sold in the same store on the same day. The solution to this problem requires the addition of attributes sufficiently unique in their combination and with one or another foreign key to provide a primary key for all facts in the table. In the example, these attributes include a date, receipt number, and line item. The primary key still requires an inherited foreign key, in this case from Store_Id, because a given item could be recorded with the same line item number and receipt number on the same day, unlikely as this may seem. Such a transaction could not occur in the same store, however. Therefore, Store ID is the only key which distinguishes the transaction uniquely when used in combination with the fact table keys. In this case, then, a partial key must be added to the fact table to satisfy uniqueness constraints.

## 3. Analysis

On the basis of the issues summarized above, this section explores the relationship between ERD and star schema along empirical lines, that is, by searching for recurring patterns and conditions in evidence during the process of translating star schema to ERD. Examples of star schema were selected from [9]. They were then sorted into classes according to the distinctions enumerated above (simple case, star model with outrigger tables, snowflake schema, etc.) to form two general categories: standard and non-standard cases. The direction of translation is from star

schema to ERD. This seemed advisable because 1) star schema are simplifications in their own right with fewer tables and more direct associations, and 2) simple translations can be seen as *in vitro* structures permitting a more immediate identification of recognizable features and cues, 3) these features and cues become an important part of the lexicon for an ERD to star schema translation.

It is important to note that the translation from star schemas to ERDs is non-deterministic, viz., there may be more than one ERD translation. ERDs contain a degree of semantic information that is unavailable in table schema alone, and primary key design, redundancy issues, and performance requirements may inform the translation.

On the assumption that star schema are fundamentally of two different kinds, the standard and non-standard case, this analysis is divided into two sections. Each case is defined and significant distinctions are enumerated. Rules are then cited from the ERD modeling conventions. A translation or translations are conjectured and a transformation rule or rules are then posited. Finally, a series of examples are walked through as illustration and substantiation of the conjectures.

### 3.1 Description of the Standard Case

For the purposes of this paper, we will define the standard case to be that of the simple star schema. Recall that the simple star schema consists of a central fact table having as its primary key, the concatenation of the foreign keys of all related dimension tables which may be numerous. The nature of the relationships are M:1, facts to dimensions. Additional tables may be present, that is, the model may include a set of central fact tables of identical structure representing different levels of aggregation, and dimension tables may be normalized resulting in

outboard tables. The distinction of the standard case, however, is the multiply inherited structure of the composite key, a key which uniquely identifies each record. The standard case is therefore the simple case along with its various extensions which may include schema with associative, outboard, and snowflaked dimension tables.

### 3.2 Standard Case Translation

We may therefore state the following features and regularities with respect to the standard case. In its simplest form we know the following:

i) The schema consists of a fact table whose primary key is a) wholly inherited from some number of related dimension tables, b) whose value is the concatenation of the singular foreign keys contributed by each of those tables, and c) whose value uniquely identifies individual records in the fact table.

ii) The relationship between fact and dimensions tables is M:1, that is, there are many facts for a given dimension.

While we may see variations which involve multiple fact tables, or further normalization of dimension tables into outrigger tables, or of dimensional hierarchies into the snowflake schema, if the keyed relationships between the fact and directly related dimension tables remain the same, that is, they conform to items (i) and (ii), they are instances of the simple case.

A clue to the ERD translation in the case of the simple star schema is that the fact table has no primary key of its own, but inherits the primary keys of all related dimension tables which are concatenated to form its unique identifier. This situation corresponds to that of the relation table in a M:N relationship where the primary key consists of the primary key of participating entities. In all of these cases, exercises in translation suggest the same underlying entity-relation structure, that of the ternary M:N:P relationship. We know that n-ary relationships are types of

degree three or higher. In the simplest case, that of the ternary relation, we know the following as discussed in the literature [2, 3, 11].

A ternary relationship is a relationship among three entities such that for any given pair of entities, there is a concurrent relationship with a third entity. Cardinality for these relationships may be of the following types: 1:1:1, 1:M:1, 1:M:N, and M:N:P. Further, while ternary relations can be reduced to three binary relations, this set of relations is not equivalent to the ternary relation from which they are derived [5, 6]. That is, while instances of (a, b), (b, c), (a, c) may exist in each of the binary relations, a combination (a join) of these tuples is not necessarily equivalent to the ternary instance (a,b,c). Finally, we understand that, given the semantics of ternary relations implied through cardinality constraints, the FDs of the relation determine the keys of the different varieties of ternary relation as follows:

1) 1:1:1 - the primary key may be any one pair from the candidate keys A#B#, AC# or BC#
2) 1:M:1 - the primary key may be either A#B# or A#C#, the N:1 pairings
3) 1:M:N - the primary key is the composite of the keys of two entities with cardinality M:N
4) M:N:P the primary key is the composite of all keys of the participating entities

On the basis of the ternary relationship, we formulate the following conditions and rules for translation of a star schema of the standard case to an ERD.

### 3.2.1 Transformation Rules: The Standard Case

When the star schema is of the simple form, that is, conforms to observations (i) and (ii) above, translate the star schema into the form of an n-ary relation via the following steps:

1) Translate each dimension table into an entity
2) Translate each fact table as an n-ary relation to associate the principle dimension tables

3) Follow ERD representation conventions:
    a) Do not represent foreign keys in the relationship
    b) Identify any non-key attributes
    c) Identify primary keys of the dimension tables
    d) Add any non-key attributes of the dimension tables
4) Note cardinality constraints on the edges of the relations as M:N:P
5) Translate any outrigger or snowflaked tables into the appropriate form of binary relation noting constraints accordingly.

## 3.3 Example Translations for the Standard Case

Schema presented in [9] provide a series of examples to illustrate the analysis. Four of these are presented as figures in the Appendix.

### 3.3.1 Example: Simple Star Schema

The simple star schema in this most elementary case consists of single central fact table, Sales, keyed respectively by the three dimension tables Product, Period, and Market (Fig. 1). All relationships are M:1 , fact to dimension. Facts are clearly of a quantitative nature. Dimension attributes are types and kinds, and lend themselves to categorical interpretation (e.g., Size may be small, medium, or large.).

The ERD represented in Fig. 2 is the product of the observation that the fact table inherits its composite primary key as the concatenation of foreign keys from each of the three dimension tables. These tables are therefore made the associated entities with primary keys as noted. The central fact table becomes the ternary relation, Sales. Non-key attributes are left out for clarity of presentation.

### 3.3.2 Example: A Multiple Fact Table

In this second example, the star schema is like the first, but with an additional fact table from a previous period of time (Fig. 3). Primary keys are similarly the concatenation of the foreign keys of the three dimension tables. The ERD is similar to that of Fig. 2, but now reflecting dual ternary relationships. The second fact table is differentiated by name as shown in Fig. 4.

#### 3.3.3 Example: Use of an Associative Fact Table

A third example illustrates the case in which a fact table can be used to resolve a M:N relationship between two related dimensions of the product hierarchy (Fig. 5). In this respect, the associative fact table operates like a relation used to translate a M:N relation into two 1:M relations in binary modeling. The associative fact table works like an associative entity taking on the key values of the two entities it unites. The content of the table is a primary key, the concatenation of the foreign keys of the two related dimensions, which is used to relate the primary and secondary tables. In the ERD which supports M:N, this relational function is translated into its obvious counter-part, a binary M:N relation (Fig. 6).

#### 3.3.4 Example: Variation on a Dimension with Two Outbound Tables

In Poe's example, Region and District dimensions are extracted from the Market dimension and added as secondary tables (Fig. 7). The simple translation of this case sees the secondary dimension tables transformed into entities and related to the entity, Market, in two 1:M relations (Fig. 8).

### 3.4 Description of the Non-Standard Case

The non-standard case is here defined as that of the multi-star schema. Recall that the multi-star schema presents a case where the set of inherited keys from related dimension tables is not sufficient to uniquely identify the records of the fact table. In such a case, additional keys must

be added to the fact table to satisfy the uniqueness constraint. These keys must be attributes commensurate with the granularity of the fact table and typically would include items such as transaction or serial numbers. In combination with one or another foreign key from related dimension tables, these keys form a primary key for the fact table of the multi-star schema. The nature of the relationships remains M:1, facts to dimensions. Additional tables may be present and dimension tables may be normalized to include associative, outboard, and snowflaked dimension tables.

### 3.5 Non-Standard Case Translation

We now state the following features and regularities with respect to the non-standard case. In its simplest form we know the following:

iii) The schema consists of a fact table whose primary key is a composite of a) a key or set of keys which uniquely identify instances at the appropriate level of granularity when used in combination with b) one or another foreign keys inherited from some number of related dimension tables.

iv) The relationship between fact and dimensions tables is M:1, that is, there are many facts for a given dimension.

While we may see variations which involve multiple fact tables, or further normalization of dimension tables into outrigger tables, or of dimensional hierarchies into the snowflake schema, if the primary key of the fact table is a composite of a partial key defined at the level of the unit or aggregate of the fact table, and one or another foreign keys contributed by related dimension tables, that is, the case conforms to items (iii) and (iv), then the schema is an example of the non-standard case.

Clues to the ERD translation of the non-standard case include 1) the inheritance of a key from an identifying relationship with a dimension table, and 2) the formation of a partial key to

uniquely identify records of the fact table in combination with this inherited key. These features suggest an ERD structure of a weak entity with a binary or n-ary identifying relationship type.

We recall that a weak entity type is distinguished by being related to an identifying owner entity and a weak entity cannot be identified without the contribution of some attribute from the identifying owner. The identifying relationship between a weak entity and its owner is one of total participation as the former cannot be uniquely identified without the latter [3]. Weak entities may also have multiple identifying relationships, thus providing for the case in which multiple dimension tables contribute keys.

On the basis of the weak entity type, we formulate the following conditions and rules for translation of a star schema of the non-standard case to an ERD.

### 3.5.1 Transformation Rule: The Non-Standard Case

When the star schema is of the multi-star form, that is, it conforms to observations (iii) and (iv) above, translate the star schema into the form of a binary or n-ary weak relationship type via the following steps:

1) Transform the dimension tables which contribute keys to the fact table into the identifying owners of the weak entity
2) Transform the fact table into a weak entity
3) Note the partial key which uniquely identifies instances of the weak entity
4) Associate the owner entities to the weak entity by way of an identifying relationship
5) Indicate total participation for the weak entity
6) Follow ERD representation conventions:
    a) Do not represent foreign keys in the relationship
    b) Identify any non-key attributes
    c) Identify primary keys of the dimension tables
    d) Add any non-key attributes of the dimension tables
7) Note cardinality constraints on the edges of the relations

8) Translate any outrigger or snowflaked tables into the appropriate form of binary relation noting constraints accordingly.
9) Where there are multiple identifying owners, assume an n-ary form of ownership and translate the star schema into the form of a weak entity with n-ary ownership.

### 3.5.2 Example: A Sales Transaction Database in the Form of a Multi-Star Schema

A Schema presented in Poe provides examples to illustrate the analysis. These are presented as Figures 9 and 10.[2]

The example is that of a sales transaction database. Tuples in the transaction table are uniquely identified, not by the simple concatenation of foreign keys as in the simple case, but by a primary key which is the composite of the partial key--Date, Receipt_Nbr, and Receipt_Line_Item--and the foreign key Store_Id and SKU_Id, inherited from the Store and SKU tables, respectively. Recall that the keys of two dimension tables, SKU and Store, are not sufficient in combination to uniquely identify an individual transaction because multiple items of the same SKU could be sold in the same store on the same day. The solution to this problem requires the addition of the attributes date, receipt number, and line item. The primary key still requires an inherited foreign key from Store_Id, because a given item could be recorded with the same line item number and receipt number on the same day. In addition to the multi-star configuration, two outbound tables are associated with SKU. This ERD translation of Poe's multi-star schema attempts to remain consistent with the original star schema diagram by showing the relationship between the two dimension tables and the fact table as a ternary relation of a weak entity type.

### 3.5.3 A Complex Example of a Multi-Star Schema

Poe also presents a more complex example of a multi-star schema for a reservation system which is described as a more realistic application of the model (Fig. 13).While Poe provides little or no discussion about this example, it provides a somewhat richer problem for testing our rules of transformation to an ERD.

This multi-star schema consists of three fact tables: Actuals, Bookings, and Promotions. Actuals and Bookings are similar tables , but differ in important respects. Both the key structure and the set of facts are somewhat different. The Actuals table has a primary key which is the composite of Chkout_date inherited from a dimension table, Period, and a unique identifier, Confirm_#. Bookings is more complex inheriting additional keys, Promo_id and Chkin_date from the Promo_type and Period dimension tables, respectively. Additionally, both of these fact tables have multiple foreign keys tying them to several other dimension tables including Facility, Frequent_Stayers, and Room. A third fact table, Promo_Schedule is used to tie dimension tables Promo_Type, Facilities, and Period together, inheriting its primary key, Promo_id, from the Promo_type table, two additional attributes, Start_date and End_date serve as facts.

There are a number of features which make this a more challenging case for translation. A number of decisions must be made along the way which foreground the design process in terms of analytical objectives and context dependence.

First, the occurrence of two similar fact tables, Actuals and Bookings, suggests that the objective is to provide comparisons of these two sets of facts to evaluate promotions and their targets, facilities in particular. A way to get at such data for comparisons is to create multiple fact

---

[2] Note that partial keys are shown in italics. Primary keys are underlined.

tables as discussed in Example 3.3.2, above. Multiple fact tables are characteristic of this particular data warehousing schema, but would not likely exist in a representative ERD.

Second, the Promo_Schedule table is again an artifact of this particular warehousing scheme permitting 1) the separation of promotion instances, Start_Date and End_Date (extracted and added to the Promo_Schedules table), from the Promotions dimensional attributes, Name, Type, and Description; and 2) the joining the three dimensions, Promo_Type, Facility, and Period. The Promo_Schedule table, then, would be dropped in the recovery of the ERD with Start_Date and End_Date moving into a Promotions table.

The third artifact of this particular multi-star schema involves the Period table which includes attributes for segmenting the data for comparison. This is a dimension of analytical interest created for the star schema and would likely not exit in an originating ERD.

On the basis of this analysis, we provide two somewhat different translations (Figs. 14, 15). The differences follow from our assumptions about Confirm_#, whether or not it is a unique identifier. In the first case (Fig. 14), we assume that the Bookings table must inherit a key based on the multi-star key structure. Bookings is depicted as a weak entity type with a partial key, Confirm_#, and an inherited key from Promotions, Promo_id. In short, Promotions stimulate Bookings in Facilities with Rooms that are targeted for such purposes and reserved by Frequent_Stayers. In a second possible translation (Fig. 15), the ERD is constructed assuming that Constraint_# is unique. In this case, related entities simply contribute foreign keys. This interpretation is suggested by the key structure of the Actuals table. If not for Chkout_date, Confirm_# would be the only key of this table, and we have eliminated the Period table which contributes Chkout_date from the translation.

In both cases, the Actuals and Period tables have been dropped as artifacts of the star schema design. Actual_room_rate is moved to the Bookings table. Bookings now contains both reservation and stay information. Frequent_Stayers, Facility, and Room tables are retained. Promotions contains the data of the Promo_type and becomes the owner of the weak entity Promo_Schedule. A key, PromoSched_# is added to the Promo_Schedule table and serves to uniquely identify any one instance of the multiple Promotions of a kind which may be offered at a given Facility.

While these translations appear coherent on the basis of features derived in the analysis above, they appear somewhat unrealistic as they suggest that the DBMS is designed solely to track promotions and their effect on bookings by frequent customers. But of course, this was the purpose of the data warehousing scheme on which these ERD translations were based. The OLTP system from which such data would be drawn would be more comprehensive than what is suggested by the ERDs constructed from the star schema in Poe's example. This more realistic case indicates that star schemas only account for a particular subset of data and data structures originating in the production DBMS, and that a much larger understanding of the reservation system is needed to arrive at an accurate understanding of the ERD. This example proves to be a useful check on the rules derived in the simpler examples above, and suggests that caution must be exercised in moving between these different domains.

## 4. Summary and Conclusion

Our analysis has proceeded via four steps which involved:

1) Evaluating a series of star schema to extract a number of formal features such as key configurations and table relations
2) Identifying analogous structures in ER diagramming
3) Positing rules for transforming archetypal instances of star schemas into ERDs

4) Constructing representative ERDs to determine if semantics would be preserved

In the simple case using simple examples, these operations proceeded smoothly, while in the non-standard case of the multi-star schema, it was seen that design objectives of the data warehousing strategy required that some decision-making and qualification be introduced into the process. Star schemas only account for a particular subset of data and data structures originating in the production DBMS. A much larger understanding of a system is needed to arrive at an accurate understanding of the ERD, and ultimately, the design for a data warehouse must be based on business needs and interests.

To this point, the analysis has been informative. It has provided us with some features for navigating the landscape, cues for moving from star schema to likely ERD structures. The heuristics developed in this paper are necessary for the translation from ERD to star schema as well. The payoff which was suggested in the introduction--the time and cost savings resulting from a technique or possibly an algorithm to translate existing designs of the structure of information within businesses, ERDs--must be pursued by examining transformations from the ERD to star schema. We believe, however, that the heuristics discussed in this paper are necessary and will be useful in that process, and ultimately in mapping and maintaining consistency between OLTP databases and data warehouses.

**References**


1) Chen, P. *The Entity-Relationship Model -- Toward a Unified View of Data. QACM Transactions on Database Systems,* Vol.1, No. 1 (1976), pp. 9-36.
2) Date, C.J. *An Introduction to Database Systems, 5th Ed*. Menlo Park, CA: Addison-Wesley Publishing (1991).
3) Elmasri, R. and S.B. Navathe. *Fundamentals of Database Systems, 2nd Ed*. Menlo Park, CA: Addison-Wesley Publishing (1994).

4) Inmon, W. H. *Building the Data Warehouse, 2nd Ed*. New York: John Wiley and Sons, Inc. (1996).

5) Jones, T.H. and I.Y. Song. *Analysis of Binary/Ternary Cardinality Combinations in Entity-Relationship Modeling.* Data and Knowledge Engineering, Vol 19, No. 1 (1996), pp. 39-64.

6) Jones, T.H. and I.Y. Song. *Binary Representations of Ternary Relationships in ER Conceptual Modeling.* In Proceedings of the 16th International Conference on Object-Oriented and Entity-Relationship Approach, Dec 12-15, 1995, pp. 216-225.

7) Kimball, R.*The Data Warehouse Toolkit: Practical Techniques for Building Dimensional Data Warehouses*. New York, NY: John Wiley and Sons (1997).

8) Livingston, G. and B. Rumsby. *Database Design for Data Warehouses: The Basic Requirements* in Planning and Designing the Data Warehouse, R. Barquin and H. Edelstein (Eds.), Upper Saddle River, NJ: Prentice Hall (1997).

9) Poe, V. *Building a Data Warehouse for Decision Support.* Upper Saddle River, NJ: Prentice Hall (1996).

10) Red Brick Systems. *Star Schemas and STARjoin Technology*. Los Gatos, CA: Red Brick Systems (1995) [www.redbrick.com].

11) Teory, T.J. *Database Modeling and Design: The Fundamental Principles, 2nd Ed*. San Francisco, CA: Morgan Kauffman (1994).

## Appendix: Examples

### Example 3.3.1: A Simple Star Schema for a Sales Data Warehouse

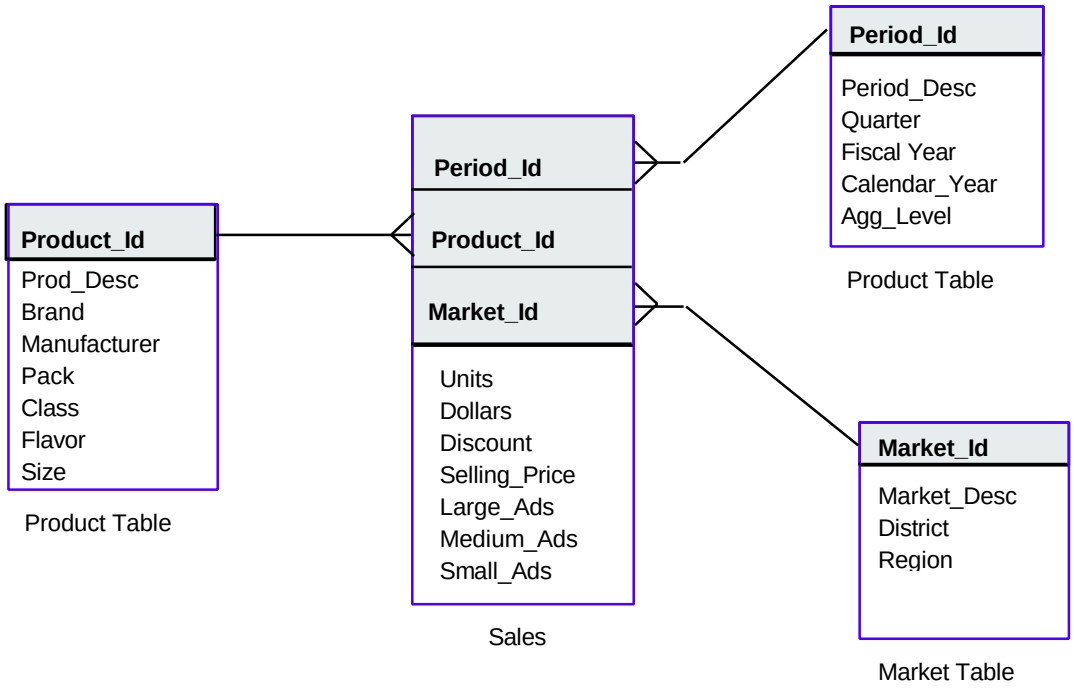


Fig 1. Star Schema for a sales database showing relationships of primary keys ( Poe, Fig. 7-2, p. 125)

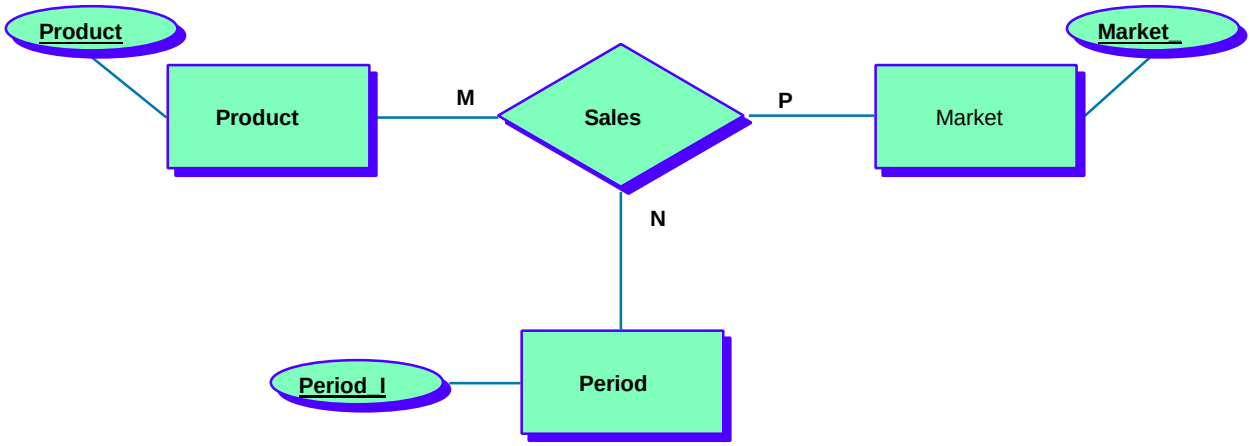


Fig. 2 An ERD for a sales database realized as a ternary relationship and showing key attributes

**Example 3.3.2: Simple Star Schema: Use of Multiple Fact Tables.**

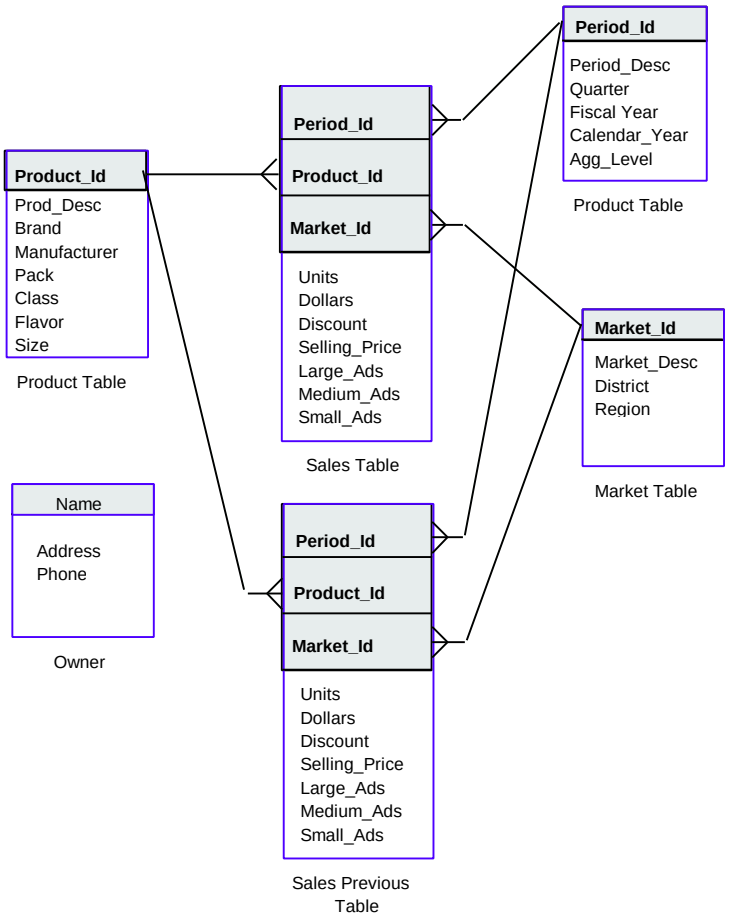


Fig. 3 Star schema for a sales database with multiple (2) fact tables (Poe, Fig. 7-3, p. 126)

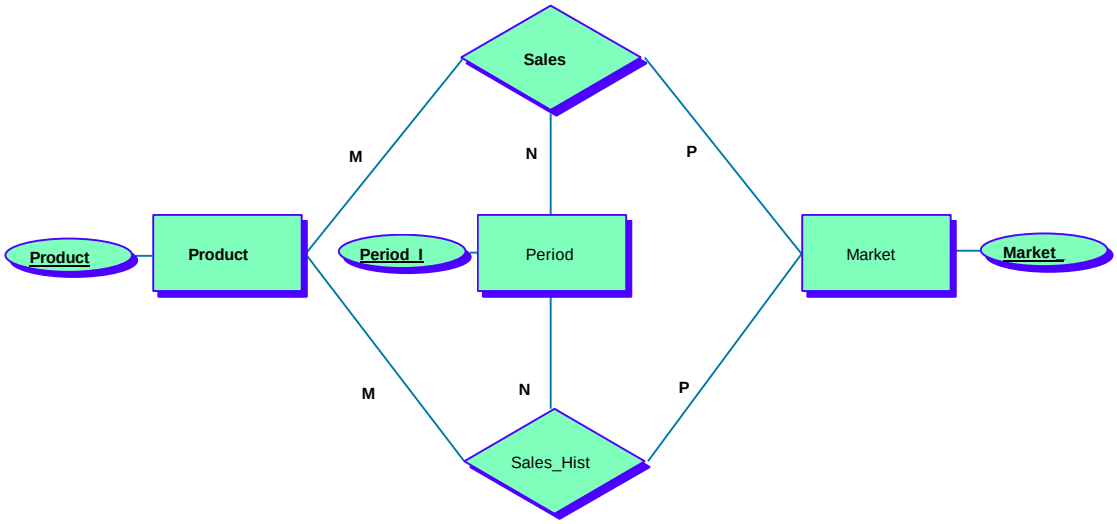


Fig. 4 An ERD translation of a multi-fact table

## Example 3.3.3: Use of a Fact Table to Resolve a M:N Relationship

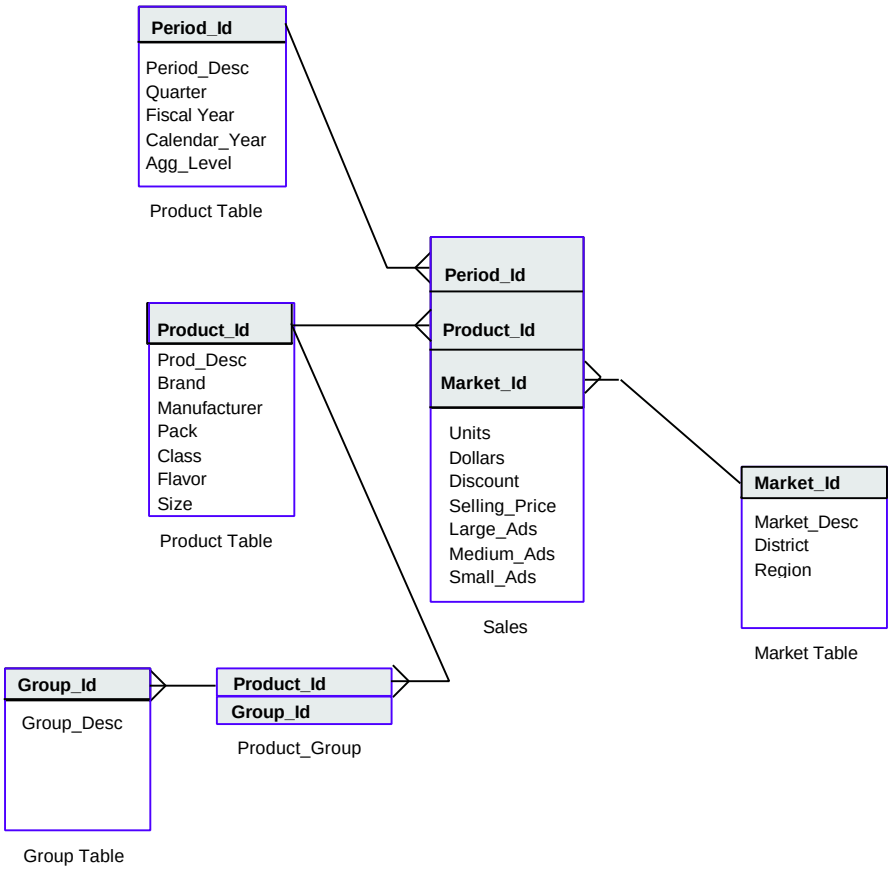


Fig. 5. Use of a fact table to resolve a M:N relationship (Poe, Figure 7-4, p. 127).

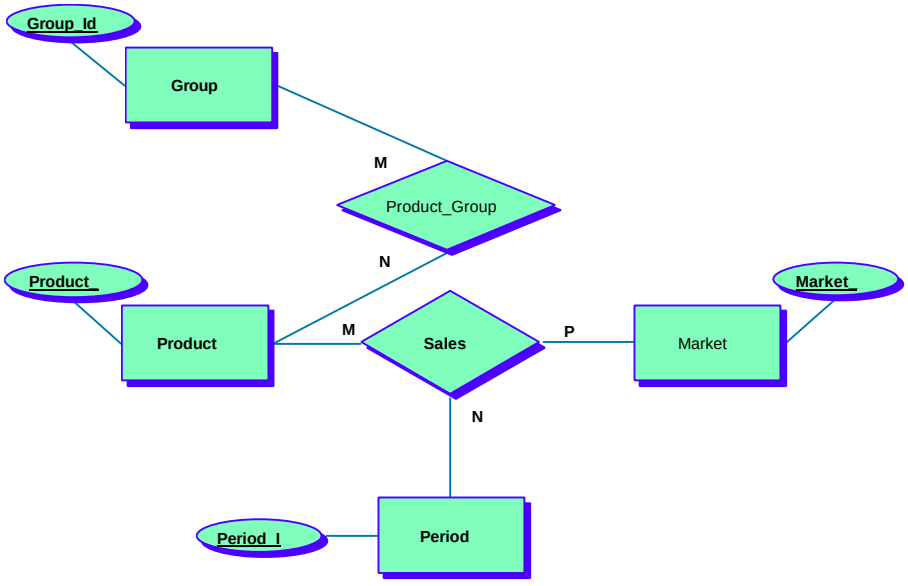


Fig. 6. ERD for a sales database with a relation used to manage a M:N relationship

## Example 3.3.4: Variation on a Dimension Table--Addition of Outboard Tables

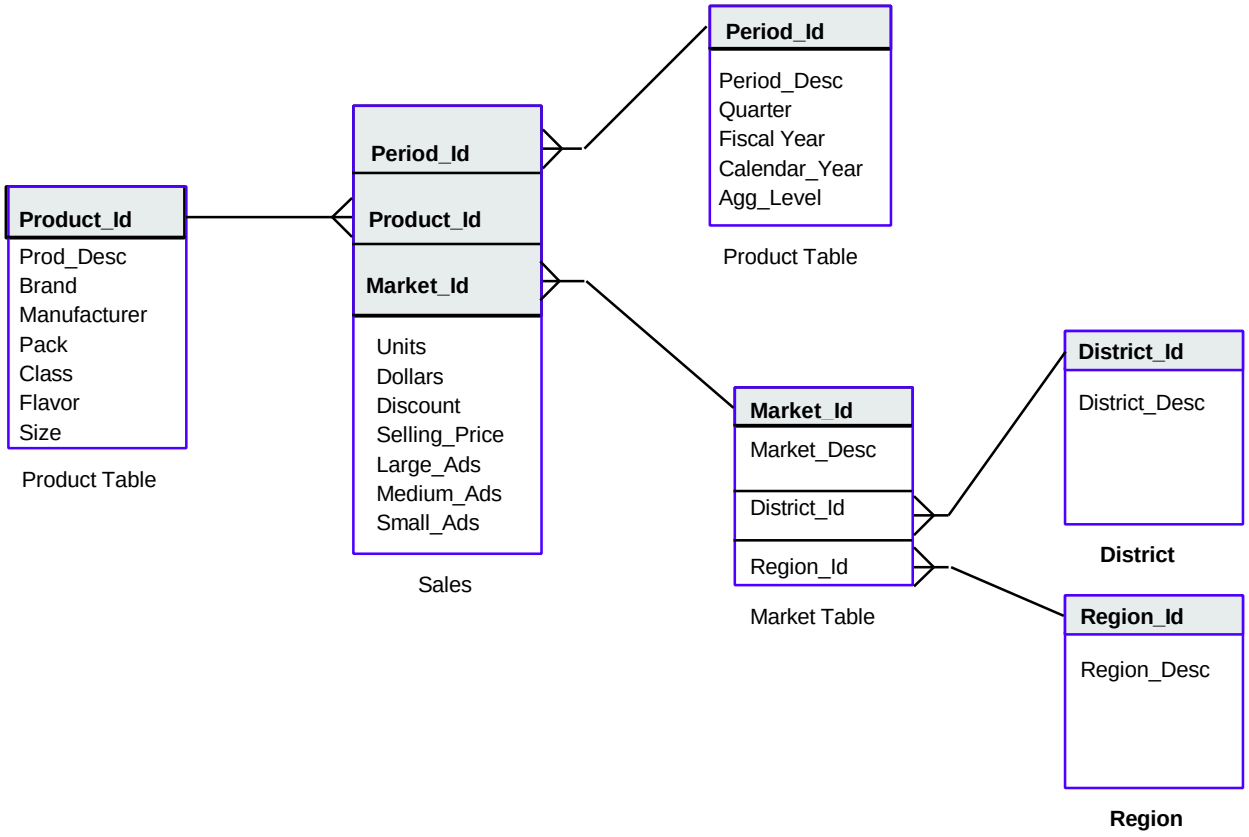


Fig 7. Star schema for a sales database showing two outboard tables of a dimension (Poe, Fig. 7-5, p. 128)

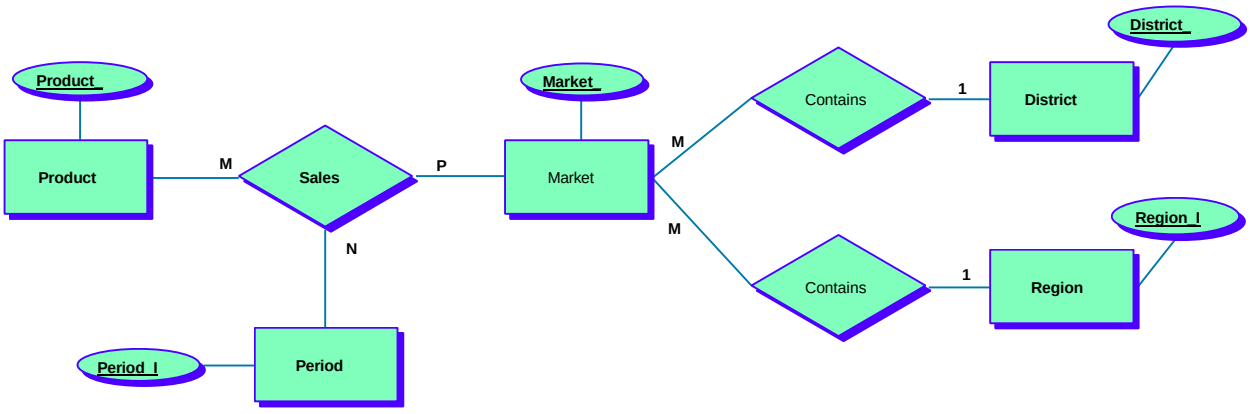


Fig. 8 An ERD showing the translation of outbound tables of a dimension.

**Example 3.3.4: Snowflake Schema: Normalization of Dimension Tables.**

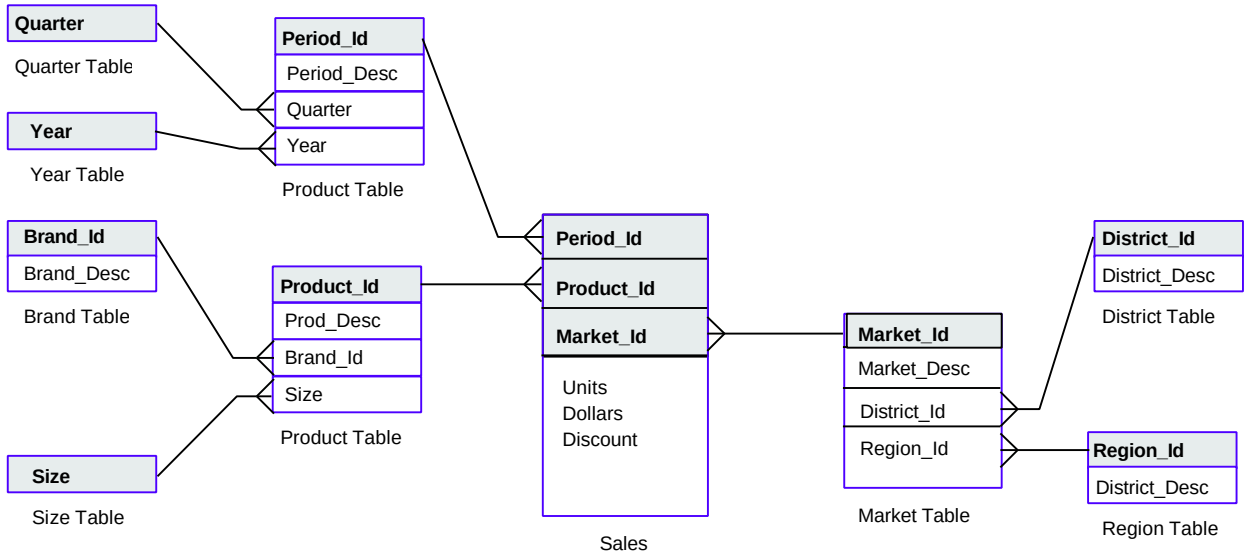


Fig 9. A snowflake schema (Poe, Fig. 7-6, p. 129).

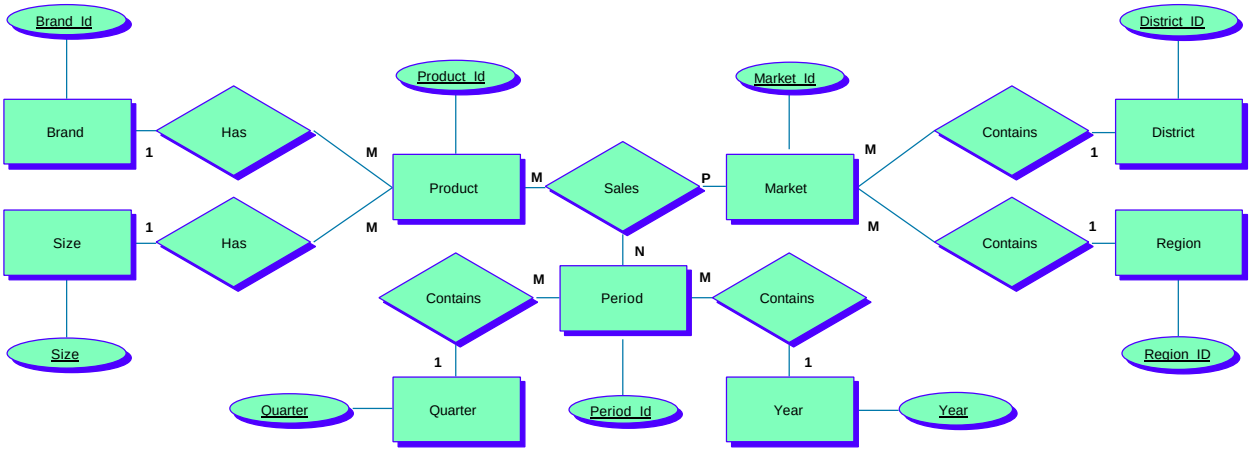


Fig. 10 An ERD showing the translation of a snowflake schema.

## Example 3.5.2: A Complex Multi-star Schema for a Reservation System

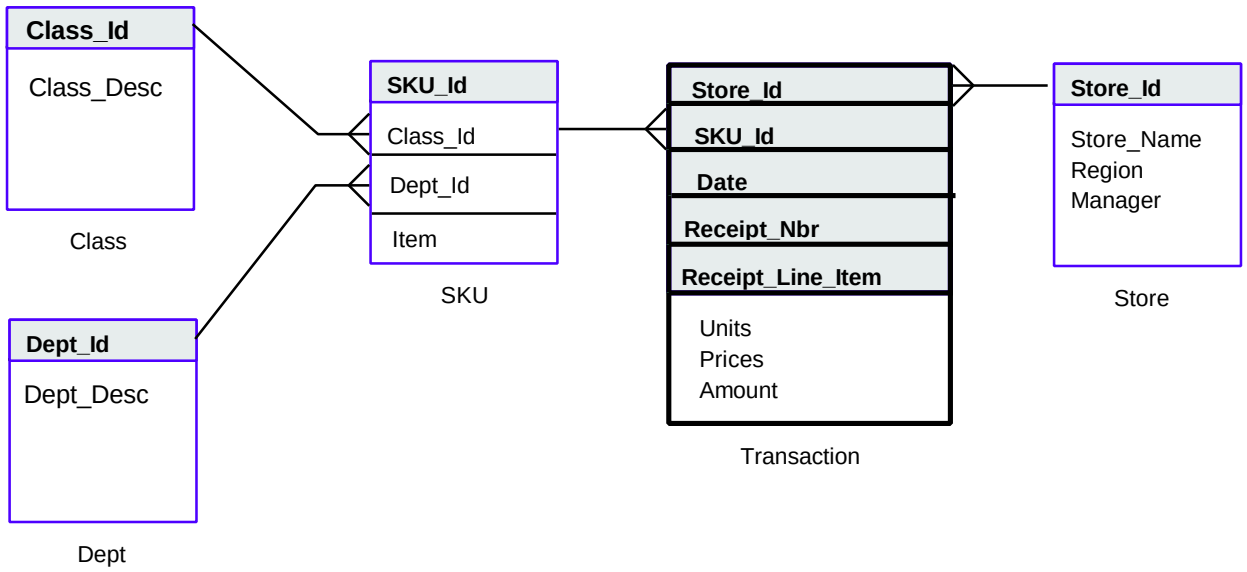


Fig. 11. Retail sales database as a multi-star schema with two secondary dimension tables (Poe, Fig. 7-8, p. 131)

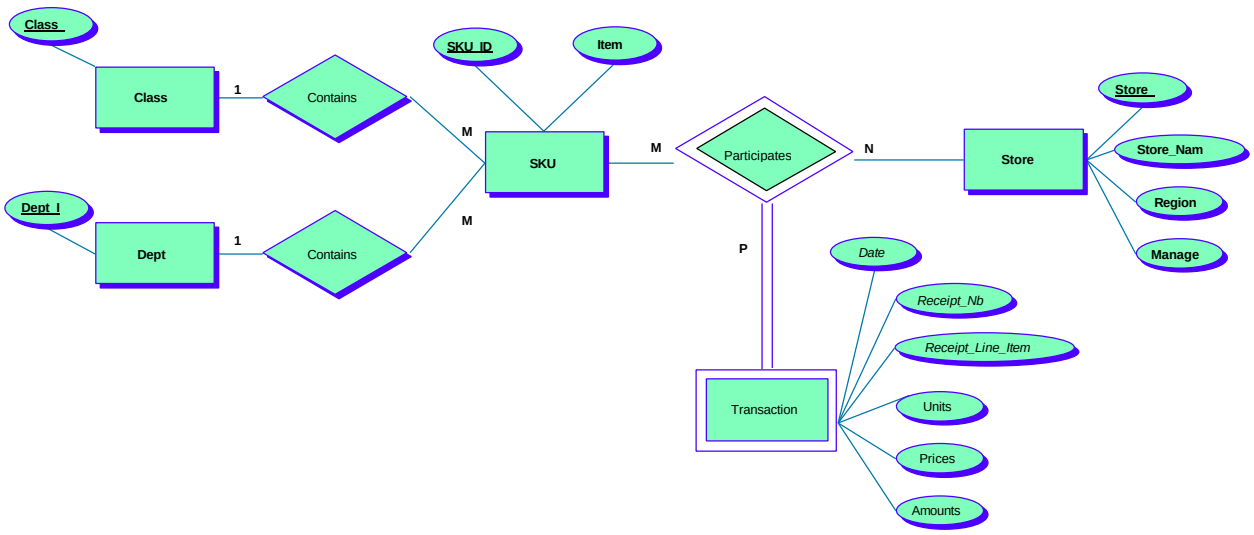


Fig 12. An ERD translation of Poe's multi-star schema for a sales transaction database

## Example 3.5.3: A Complex Multi-Star Schema for a Reservation System

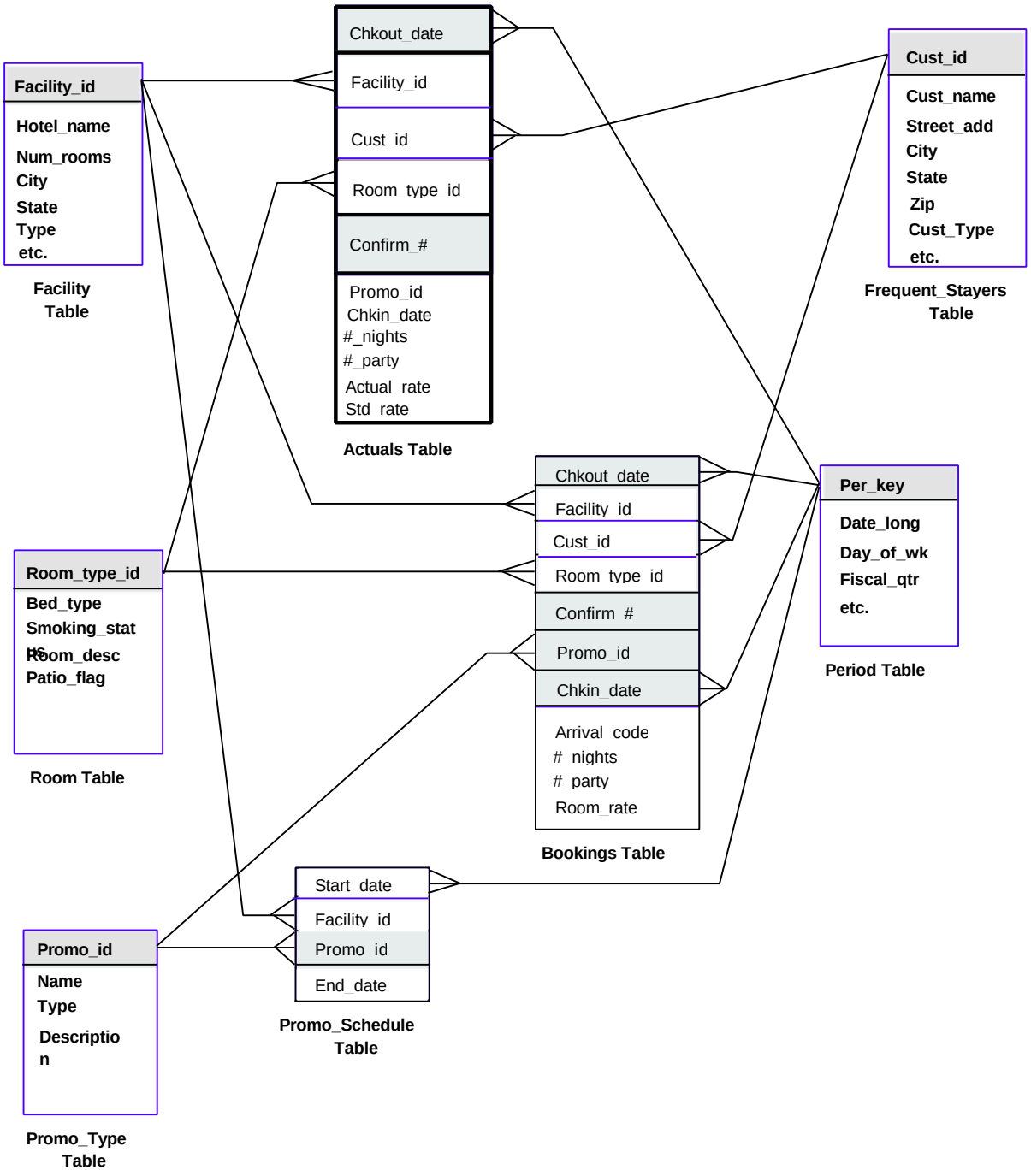


Figure 13. A Multi-Star Schema for a Reservation Database (Poe, p. 140)

## Example 3.5.3: A Complex Multi-star Schema for a Reservation System

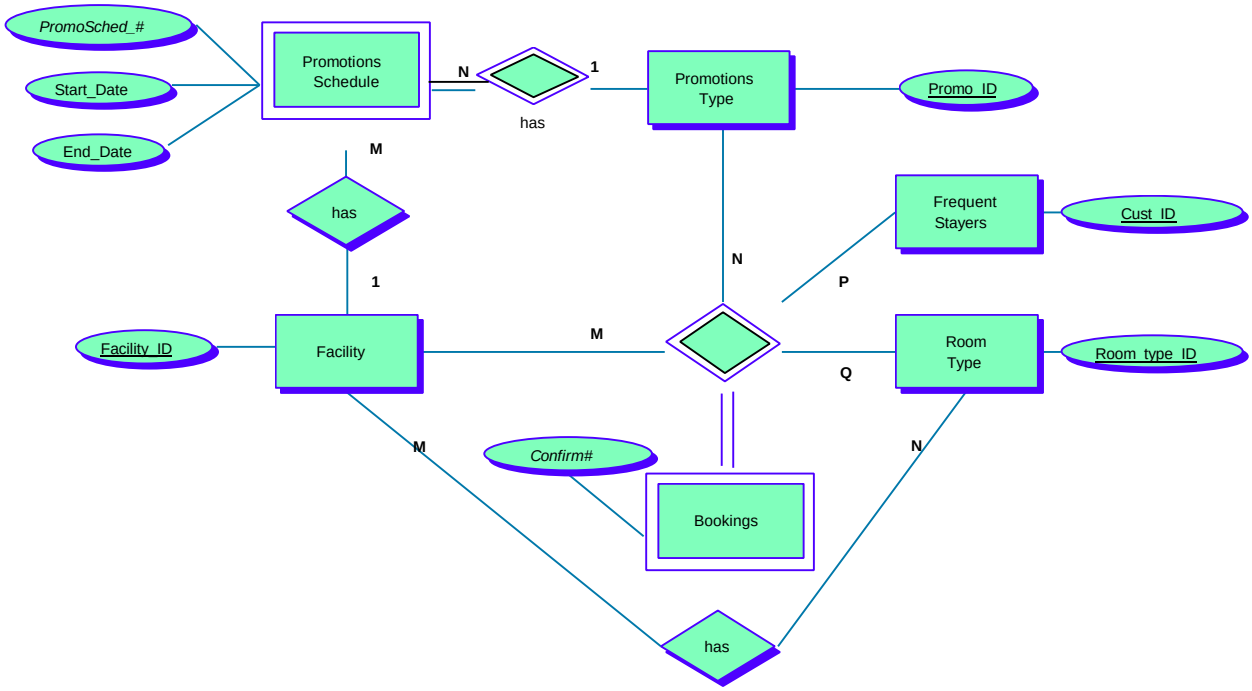


Fig. 14. An ERD translation of Poe's schema for a reservation database where there is no unique ID for Bookings

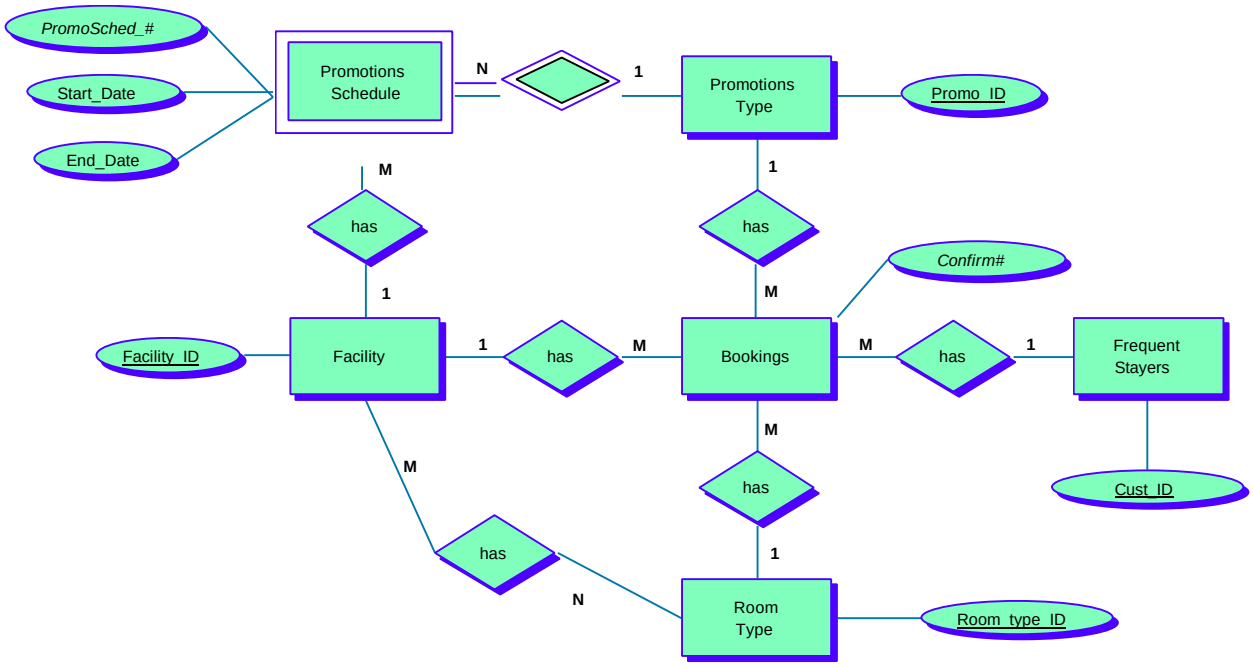


Fig. 15. An ERD translation of Poe's schema for a reservation database where there is a unique ID for Bookings