

The Customer Logical Entity Attribute Relationship (CLEAR) Model for Business to Consumer Marketing Databases

Jeff Fowler
Decision Software, Inc.
4640 Forbes Blvd, Suite 310
Lanham, MD 20706
jfowler@dsoftware.biz

Originally Published April, 2002
Revised May, 2009

Abstract

This White Paper sets forth a particular style of marketing database design called CLEAR: Customer Logical Entity Attribute Relationship. This model has proven to be both intuitive for users and efficient in terms of storage requirements and performance. It presents a logical alternative to Star Schemas, and has been successfully employed in installations both large (100 million consumer records) and small (200,000 customers).

About the Author

Jeffrey Fowler began his career as a mainframe computer programmer in the early 1980's, and has been designing relational marketing databases since 1991. He founded Decision Software in 1989, and was the original developer of a campaign management software application known as TopDog®. His company continued to market TopDog until 2003, when it was replaced with their current MarketWide™ product. In addition to marketing software, DSI builds and maintain databases for a number of clients. Mr. Fowler is an occasional contributor of database articles to various industry publications, and has also given speeches on the subject.

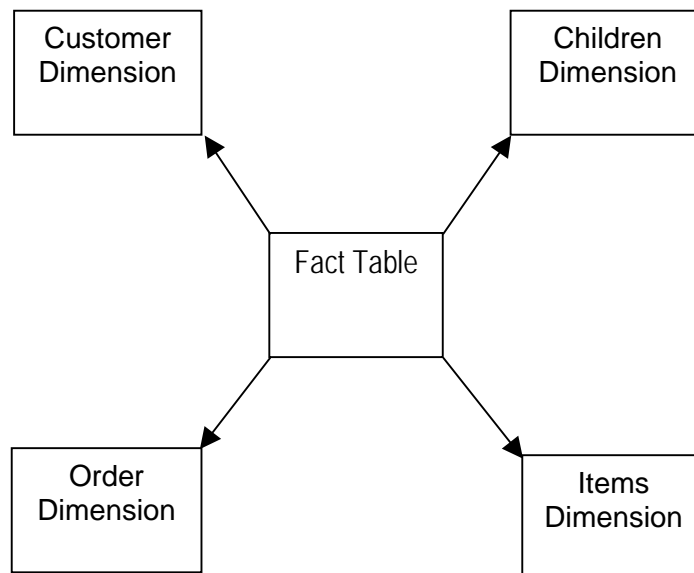
Context

This White Paper focuses strictly on marketing database design techniques for Business to Consumer (B-C) companies. The examples herein pertain to a consumer electronics company called Acme Corporation. Acme makes electronic games for children, and sends monthly catalogs to its customers and prospects, who respond by placing orders either over the telephone or from Acme's website for one or more items. While this White Paper assumed the use of Microsoft's SQL Server, many of the benefits cited apply to relational platforms such as Oracle, Sybase, and Informix.

Introduction

Relational databases used for marketing purposes have specific needs that mandate a different design approach than transaction processing systems. Generally speaking, a marketing database is read/only, and is updated in batch at periodic intervals. Whereas order entry systems read and write relatively small amounts of data - but do so quickly and frequently - marketing queries tend to be much lengthier in nature, scanning many rows and producing large result sets, which are often aggregated and/or sorted. While traditional database design methodologies are oriented towards tables, rows, and columns, a marketing database needs to embrace the logical concept of a **customer**, which is either a person or a business.

There are several schools of thought regarding marketing database design. One popular design approach is called the *Star Schema*, which derives its name from the appearance of its diagram. With the Star Schema methodology, a single "fact" table exists in the center of the star, and a number of "dimension" tables are created as the points. The fact table stores numeric items used in calculations such as dollar amounts and quantities, while the dimension tables store the non-numeric items used for filtering and sorting query results, such as dates or product codes.



Example of a Star Schema data model

This white paper sets forth an alternative design approach called CLEAR. As with any well-designed marketing database, the CLEAR model embraces the concept of the customer. A customer is defined as the entity that is being marketed to, which for B-C databases is a person within a household (as opposed to B-B marketing databases, which target contacts within businesses). While these two are similar, B-C marketing strategies usually differ from B-B, and therefore impose different design considerations.

In the CLEAR model, there are two types of tables: **entities** and **attributes**. An entity is defined as either:

- (a) a discrete set of data elements that pertains to a customer – such as address or demographic data;
- (b) a continuous set of data elements that pertains to things that customers have, such as vehicles or children;

- (c) a continuous set of data elements that pertains to things that customers do, such as place one or more orders for one or more items.

Attribute tables store descriptive information about entities or even other attributes. For example, a product attribute table might contain cost and descriptive information about the items that customers buy, while a manufacturer attribute table contains data pertaining to the manufacturers of those items. Attribute tables are sometimes referred to as *lookup* tables.

The Customer Entity

CLEAR requires that there must always be a unique, single field identifier for each customer. In the case of Acme Corporation, this field is called *Customer_Id*. The customer is the fundamental entity of the CLEAR model, and consequently every CLEAR marketing database begins with a table storing customer-level data. This base customer table stores one and only one row for each unique Customer_Id. CLEAR also requires *all* entity tables to carry the Customer_Id column.

There are generally two kinds of customer-level data: that which is used to analyze and/or market to the customer, and that which is used to actually solicit the customer. This latter category includes names, addresses, phone numbers, and email addresses. While these fields are critical to the marketer, they are seldom needed for query and selection purposes. They also tend to take up lots of room, and on occasion can slow queries. Therefore, it is recommended (but not required) that contact data be segregated from other "queryable" data and stored in a separate table, as shown in Figure 1.

Customer Table	Address Table
Customer_Id	Customer_Id
Gender	Address_Type
Income_Code	First_Name
Customer_Status	Last_Name
Model_Score	Address1
Birth_Dt	Address2
Occupation	City
Marital_Status	State
Length_of_Residence	Zipcode
No_Children	Plus4
Customer_Type	Phone_No
Mail_Flag	Fax_No
Telemarket_Flag	Email_Addr
Email_Flag	NCOA Date
State	
Zipcode	
Lifetime_Orders	
Lifetime_Value	
Order_Recency	

Figure 1 - Segregate name and address data from query data

Note that certain solicit fields (such as state or ZIP codes) may be used both in queries as well as to contact the customer. In these situations it is recommended that the data be redundantly stored in both tables. Often the presence or validity of solicit fields are needed in queries; for example, whether a customer has an email address or if a phone number is valid. Here a flag or indicator can be used rather than the actual field itself.

Other Entities

As mentioned previously, the CLEAR model requires the Customer_Id column to be stored in every entity table. This allows any non-customer entity to be joined back to the customer as well as the address table. Another reason for this practice is that Acme Corporation markets to people, not to products or items, and in the marketing database world, a Customer_Id *is* the customer.

Figure 2 – Customer_Id is replicated in every Entity table

Children Table	Orders Table	Items Table
Customer_Id	Customer_Id	Customer_Id
Child_Id	Order_No	Order_No
Child_Name	Order_Date	Line_No
Birthdate	Order_Amount	Product_Id
Gender	Order_Status	Quantity
	Payment_Type	Unit_Cost
	Ship_Date	Item_Status
	Ship_Type	Child_Id
	Ship_Amount	
	Shipto_Id	

Attributes

As mentioned, attribute tables describe or supply additional data pertaining to entities or even other attributes. By definition, an attribute table cannot contain a Customer ID. Shown in figure 3 are Acme's product and manufacturer attribute tables. Note that the manufacturer table in this example carries descriptive data that pertains to manufacturer codes, which itself is an attribute of a product. With CLEAR, the primary key for attribute tables is the attribute code itself (e.g., product id or manufacturer code), which generally become a foreign keys in entity tables. Using attribute codes as primary keys enforces their uniqueness.

Product Table	Manufacturer Table
Product_Id	Manufacturer_Code
Product_Description	Manufacturer_Name
Product_Category	Address1
Manufacturer_Code	Address2
	City
	State
	Zip
	Plus4
	Phone_No

Figure 3 – Attribute Tables

Referential Integrity

Whenever possible, referential integrity should be enforced with the CLEAR model. As such, children and orders cannot exist without a matching customer, and items cannot exist without a matching order. This rule occasionally must be broken for business or technical reasons; for example, data feeds may not be fully synchronized, or perhaps certain tables are to be updated more frequently than others.

The CLEAR design forms a natural hierarchy (or “tree”), with the customer always at the top. The non-customer entities form the branches of the tree, and the attribute tables form the leaves. In figure 4, Customer, Orders, and Items are on one branch, with Customer and Children forming a second.

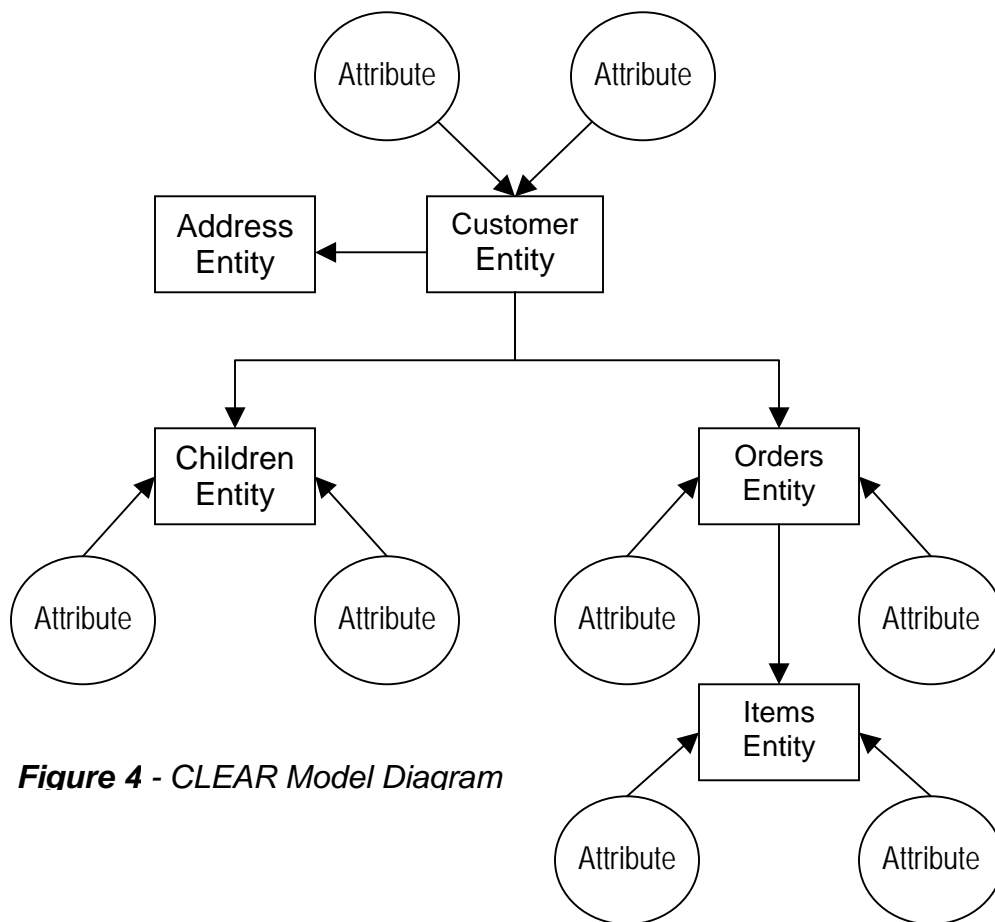


Figure 4 - CLEAR Model Diagram

Indexes and Keys

A common school of thought in database design is the assumption that every table must have a single field (frequently called an “ID column”) as its primary unique key. As such, the primary key for the Customer table might be Customer_Id, for the Orders table Order_Id, and for Items an Item_Id. Under this design methodology, joining two tables such as Customer and Orders involves matching the primary key in the Customer table to a foreign key (Customer_Id) that is added to Orders. Further, linking the Items table to either Customers or Orders requires adding *two* foreign keys (Customer_Id and Order_Id) to the Items table. In addition to being an unwieldy practice for marketing databases, this technique causes two problems:

1. Because SQL Server creates a unique clustered index on the primary key column(s), tables are physically sorted in ID column order, NOT in customer sequence. Since each table has its own ID column, every table is sorted differently, and joins between any two tables involve a read of one table's primary key and a read of the other table's foreign key (nonclustered index), followed by a seek to retrieve the corresponding data. This tends to have more overhead than straight matches between two primary keys. Further, GROUP BY operations involving Customer_Id (frequently used in marketing database queries in order to aggregate customer transactions) perform more slowly, since transactions belonging to the same customer are not stored contiguously.
2. ID columns allow unnatural and undesirable situations to exist, such as duplicate customer rows, an order being repeated for the same customer, or an item repeated within the same order.

The CLEAR model uses composite (multi-field) primary keys for entity tables, created based on granularity of the data, and the first field in every such key must be Customer_Id. As shown in figure 5, the order of fields in Acme's primary keys follows the natural hierarchy of their data. With this approach, SQL Server sorts entity tables in customer sequence, and joins between tables are performed by matching primary key to primary key, thus yielding excellent performance.

Note that Acme's Customer, Orders, and Items tables are located on the same branch of the tree, and figure 5 shows that the further along the branch we move, the more fields there are in the composite key. This lets us quickly determine the granularity of the data. Note further that CLEAR databases allow *any number or combination of tables on the same branch to be joined*, without the addition of ID columns as foreign keys. Because SQL Server clustered indexes do not take additional storage space (they are implicitly stored with the table data) and because ID columns are not used, the CLEAR model is efficient in both performance as well as storage.

Customer Key	Orders Key	Items Key
Customer_Id	Customer_Id, Order_No	Customer_Id, Order_No, Line_No

Figure 5 - Primary keys for Customer, Orders, and Items tables

Other benefits of CLEAR composite keys are:

- Marketing queries routinely include or exclude customers based on criteria, causing subqueries such as "Customer_Id IN (another table)" and/or "Customer_Id NOT IN (another table)." For customer inclusion or exclusion, CLEAR keys produce excellent results.
- Customer aggregates such as dollars within a given time period commonly create "GROUP BY Customer_Id" clauses. Again, because CLEAR-style clustered indexes always start with Customer_Id, performance for these aggregates is greatly benefited.
- When standard (nonclustered) indexes are built such as for a foreign key, SQL Server stores the contents of the clustered index along with each data value as shown in figure 6. Although composite primary keys increase storage demands for nonclustered indexes, their use often results in overall improvements on query performance. That is because if all fields needed to satisfy a query can be found in an index, *many database systems satisfy the query using the index alone*, without accessing the underlying data table. Index queries such as these tend to be extremely fast.

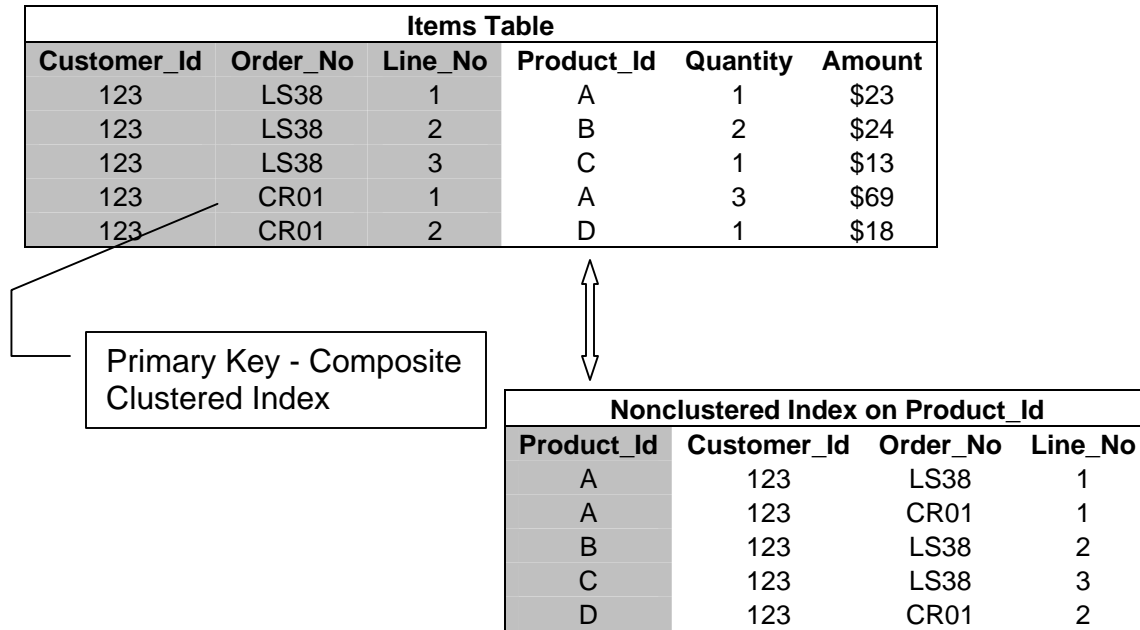


Figure 6 – SQL Server Nonclustered Index Storage

Normalization

Marketing databases are typically read/only, and are updated in batch during off-hours. Thus, database design should emphasize ease and performance of queries rather than efficiency of transaction processing (adds, deletes, updates). A well-designed CLEAR database tends to have a smaller number of “wide” tables rather than a large number of “skinny” tables; i.e., fewer tables with lots of columns rather than many tables with fewer columns. This concept is known as denormalizing a database. As such, entity tables having a relatively few number of columns should be considered, with thought given to moving the data onto the next table located along the same branch having greater granularity. As an example, if Acme’s Orders table contained only a single field – say the order date - not included in the primary key, it could be moved onto the Items table, allowing the Orders table to be eliminated.

Conclusion

The CLEAR model produces a high-performance, intuitive database that addresses the needs of marketers. It employs a straightforward, “common sense” design methodology, allowing tables to be joined without the addition of ID columns as foreign keys. The use of composite keys helps maintain data integrity, and improves performance for customer includes, excludes, and aggregate functions. Query performance overall benefits whenever full table scans are avoided in favor of index queries. Lastly, the CLEAR design focuses on simplicity and ease of use from the perspective of a marketing user, rather than convenience for the database administrator.