6

Star Schema Template

The ***star schema*** represents data as facts that are bound to dimensions. A ***fact*** measures the performance of a business or some aspect of a business; examples include sales, budget, revenue, profit, and inventory. A ***dimension*** specifies one of the bases for facts; examples include date, location, product, customer, and salesperson.

The star schema is the usual approach to data warehouse applications. A data warehouse takes the disjointed, functional applications of a business and integrates them, putting their data in one database and storing data in a common format for reporting purposes. The simple structure of the star schema makes it easier to write ad-hoc queries that mine data and gain insight into an enterprise. However, the simple structure cannot enforce constraints about data—that is the purpose of the functional applications that handle the day-to-day operations of a business.

The star schema is not limited to data warehouses and can also be used for functional applications with much reading and little writing.

There is one template for the star schema.

6.1 Star Schema Template

6.1.1 UML Template

Figure 6.1 shows the UML template for the star schema. A fact is surrounded by dimensions. The diagram happens to show eight dimensions, but there can be any number of dimensions. [Kimball-1998] suggests that a fact should have between five and fifteen dimensions for a well-designed star schema. Most dimensions are mandatory but some can be optional.

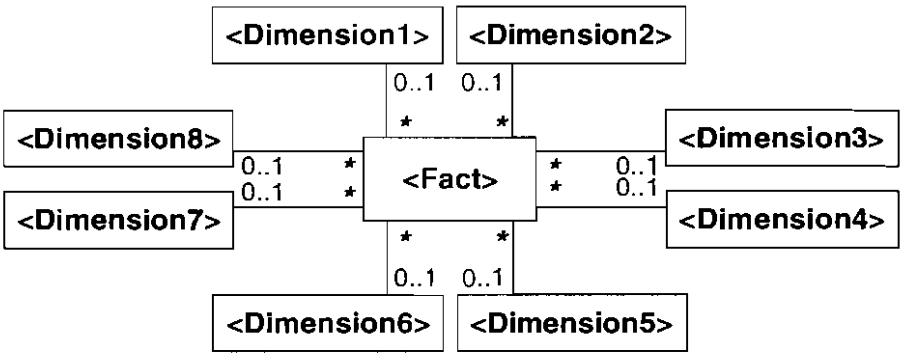


Figure 6.1 Star schema: UML template. Use when there must be a flexible structure for querying data and constraints on data are unimportant.

6.1.2 IDEF1X Template

Figure 6.2 restates Figure 6.1 with the IDEF1X notation. All the dimension IDs in *Fact* are foreign keys. The dimension foreign keys are specified to be mandatory to simplify database joins. There is no conflict between the UML and IDEF1X templates, as a conceptual NULL can be indicated with a special "NONE" record. The combination of dimensions identifies each fact and is used as the primary key to reduce fact table size.

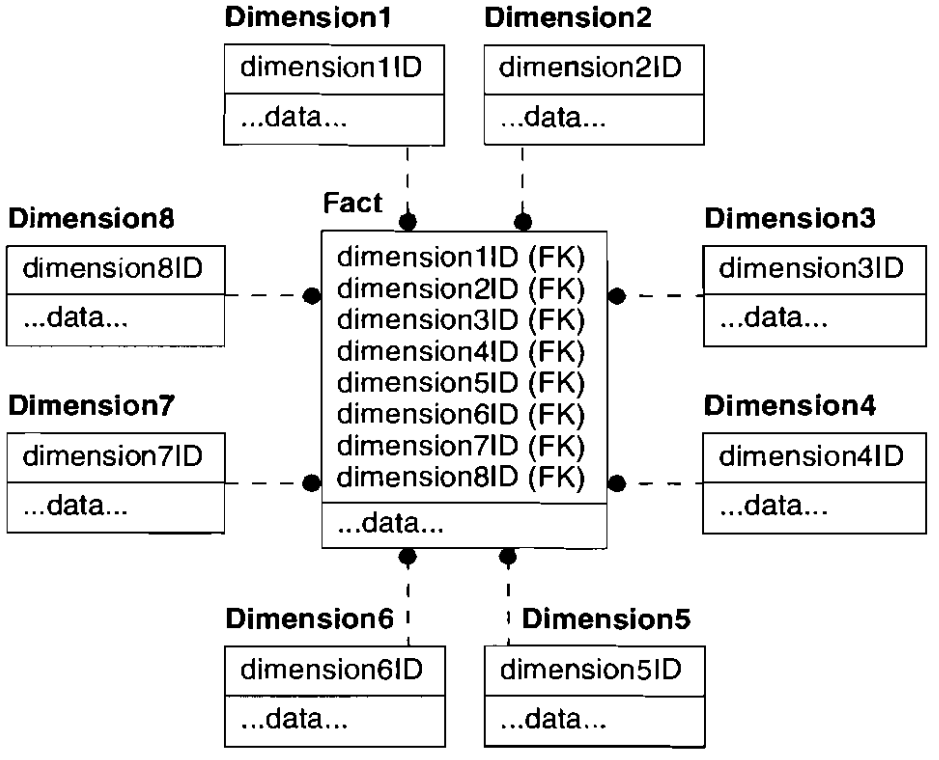


Figure 6.2 Star schema: IDEF1X template.

6.1.3 SQL Queries

Typically there are two kinds of queries for this template—querying facts and querying dimensions.

Figure 6.3 illustrates the first category of queries— selecting groups of facts and summarizing them for various combinations of dimensions. (Section 6.1.5 discusses the store sales example.) Such queries can involve massive amounts of data, so performance is always a concern. Data warehouses use special techniques to speed performance [Inmon-1993] [Kimball-1998]. The colon prefix denotes variable values that must be provided.

```
SELECT storeID, SUM(saleQuantity)
FROM Sale
   INNER JOIN Product AS P ON Sale.productID = P.productID
   INNER JOIN Date AS D ON Sale.dateID = D.dateID
   INNER JOIN Store AS S ON Sale.storeID = S.storeID
WHERE  P.productID = :aProductID AND
       D.fullDate = 'July 1, 2000'
GROUP BY storeID
ORDER BY storeID;
```

Figure 6.3 Star schema: SQL query. Summarize facts for a combination of dimensions.

The second kind of query searches dimension data to retrieve descriptive details (Figure 6.4). Such queries involve a straightforward search through a table or a few related tables.

```
SELECT storeName, streetAddress, cityName, stateName,
       postalCode
FROM Store
WHERE storeID = :aStoreID
```

Figure 6.4 Star schema: SQL query. Retrieve dimension data.

6.1.4 Sample Populated Tables

Figure 6.5 shows star schema tables populated with data. The values of the IDs are arbitrary, but internally consistent. Also for a real problem the dimension tables would have more descriptive attributes than the ones shown. The data is a subset of data for store sales and is covered further in the next section. In practice there are a modest number of dimension records (tens or hundreds per table) and a large number of facts (thousands or millions).

6.1.5 Examples

Figure 6.6 illustrates the star schema template with a store sales model. Sale is a fact that is surrounded by the dimensions of product, payment type, cashier, store, date, and customer.

In data warehouse terminology Figure 6.6 is called a snowflake schema—the dimensions are not shown as a single entity type, but rather as several associated entity types. For

Fact table

dimension1ID	dimension2ID	dimension3ID	dimension4ID	dimension5ID	dimension6ID	quantity	price	saletime
1	2	1	1	1	2	3	0.50	13:20
2	2	1	1	1	2	1	3.25	13:20
3	2	1	1	1	2	1.35	4.05	13:20
1	1	1	1	1	1	2	0.50	13:30
1	1	2	1	1	0	6	0.50	13:30
3	1	2	1	1	0	1.15	3.45	13:30

Dimension1 table

dimension1ID	name
1	16 oz can generic green beans
2	fresh pineapple
3	lean ground beef

Dimension2 table

dimension2ID	name
1	cash
2	credit card
3	debit card

Dimension3 table

dimension3ID	name
1	John Doe
2	Sally Smith

Dimension4 table

dimension4ID	name
1	primary store
2	secondary store

Dimension5 table

dimension5ID	date
1	January 1, 2010
2	January 2, 2010

Dimension6 table

dimension6ID	name
0	NONE
1	John Jones
2	Mary James

Figure 6.5 Star schema: Populated tables.

example, the *Product* dimension is associated with *Category* and *Industry*. When designing data warehouse tables, it is a common practice to denormalize dimensions and collapse their details. For example, *Industry* and *Category* could be folded into a *Product* table to reduce the number of tables and simplify the database.

The example shows six store dimensions. There could be additional dimensions including:

- promotional data (such as coupons)
- customer visit (enabling the grouping of products purchased by the customer in a visit)
- product placement (end of aisle, next to checkout, location on Web site)
- price range

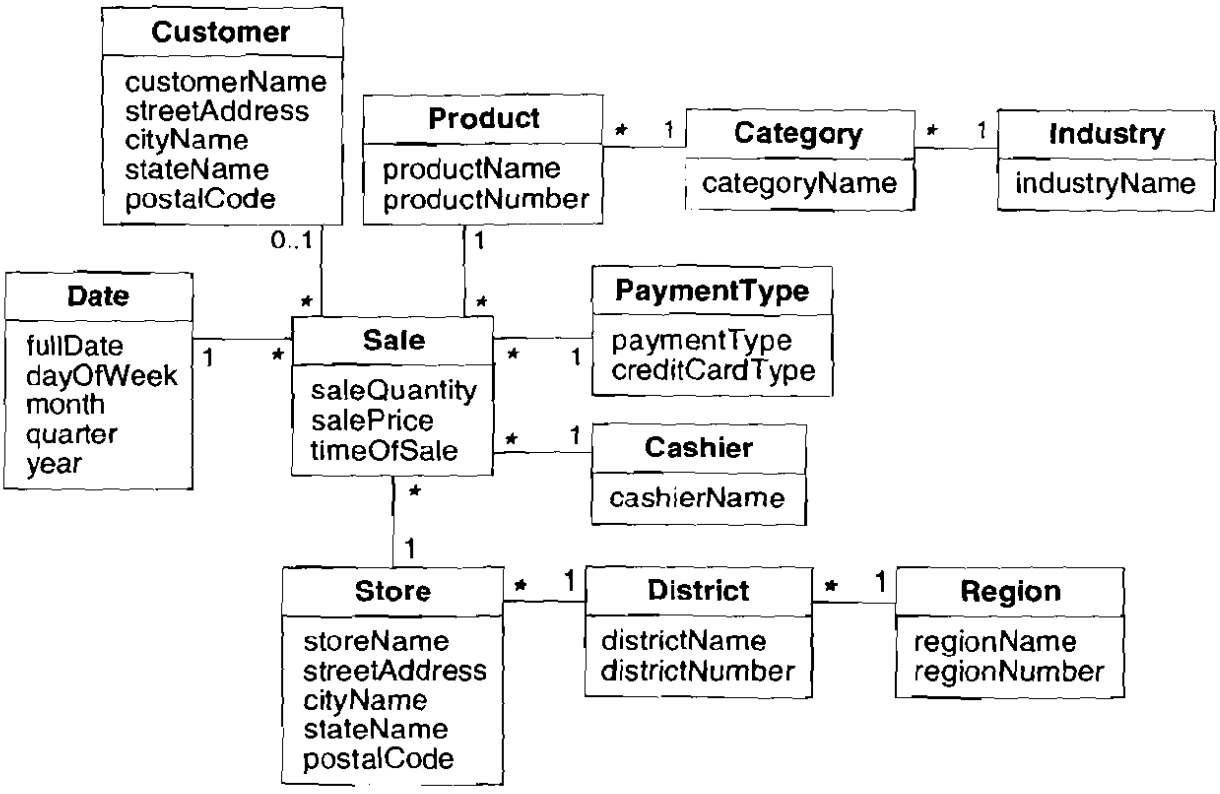


Figure 6.6 Star schema: Store sales model.

Note that *Customer* is optional in the store sales example; a person paying with cash may not be identifiable to the store. All other dimensions are mandatory.

Figure 6.7 shows another example for processing an insurance application on a property. Various events occur as an application is processed and they must all be tracked. The star schema can store the events but does not enforce constraints such as the order of the processing. (That is the purpose of the functional applications.) The star schema can answer questions regarding:

- the status of each application (the latest event type that has been processed)
- the average time for processing between each event as an application progresses
- the fastest employees
- the fastest offices

A property may have more than one owner and hence there can be multiple applicants. For example, a husband and wife may own a property. Thus there is a many-to-many relationship between *ApplicationEvent* and *Applicant*. Many-to-many relationships are troublesome for a star schema and the *Applicants* dimension groups together the multiple owners of a property to finesse the issue. The owners of a property may have unequal ownership.

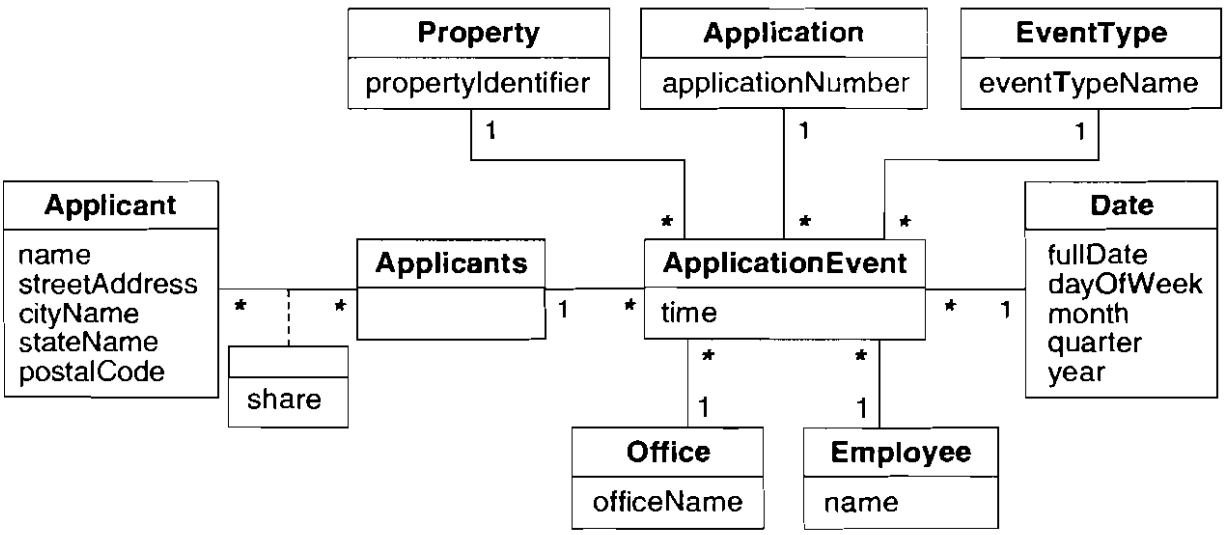
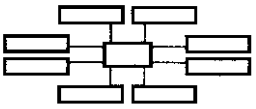


Figure 6.7 Star schema: Application processing model.

6.2 Chapter Summary

The star schema template is pervasive for data warehouse applications and sometimes occurs for functional applications. Table 6.1 summarizes the star schema template.

Table 6.1 Summary of the Star Schema Template

Template	Synopsis	UML diagram	Use when	Frequency
Star schema	Represents data as facts that are bound to dimensions.		There must be a flexible structure for querying data.	Occasional (frequent for data warehouse)

Note: Consider when there must be a flexible structure for querying data and constraints on data are unimportant.

Bibliographic Notes

[Blaha-2001] has a further explanation about data warehouses. Chapter 4 of [Fowler-1997] also discusses the star schema. Inmon and Kimball are prominent authors in the data warehouse community and have written excellent books.

References

[Blaha-2001] Michael Blaha. *A Manager's Guide to Database Technology: Building and Purchasing Better Applications*. Upper Saddle River, NJ: Prentice Hall, 2001.

[Fowler-1997] Martin Fowler. *Analysis Patterns: Reusable Object Models*. Boston, Massachusetts: Addison-Wesley, 1997.

[Inmon-1993] W. H. Inmon. *Building the Data Warehouse*. New York, New York: Wiley-QED, 1993.

[Kimball-1998] Ralph Kimball, Laura Reeves, Margy Ross, and Warren Thornthwaite. *The Data Warehouse Lifecycle Toolkit*. New York, New York: Wiley, 1998.