Chapter 11
Temporal Data Warehousing

A *Temporal Data Warehousing* is a warehousing technique whereby the temporal (or historical) aspect of records is incorporated into the warehouse. For example, if a book is a dimension (in a bookshop setting), the book price changes overtime. If we would like to keep track the book prices in the data warehouse, then a temporal data warehouse technique must be used. Without this temporal or historical aspect of the book price, the analysis made from the data warehouse is only based on the current book price.

This chapter focuses on temporal data warehousing covering star schemas, which maintain a temporal aspect of the data. A temporal data warehousing is also known as *Slowly Changing Dimensions* (*SCD*). This chapter will also describe various techniques that can be used to deal with SCD.

1. The Bookshop Case Study

In order to learn the concept of temporal data warehousing, let's use the Bookshop example as a case study. A bookshop that has a number of branches in Melbourne would like to build a data warehouse to analyse their book sales. They have already stored all book sales transactions in an operational database. The management would particularly like to analyse their book sales performance from various perspectives, such as from a monthly basis, from a book basis, and from a branch basis. The first task to do this is to define a star schema.

A simple star schema for this bookshop case study is shown in Figure 11.1. It has three dimensions: Time_DIM, Branch_DIM, and Book_DIM, and the fact has one fact measure, called Number_Of_Books_Sold. With this star schema, the management is able to retrieve number of books sold based on the month, based on the branch, and based on the book.

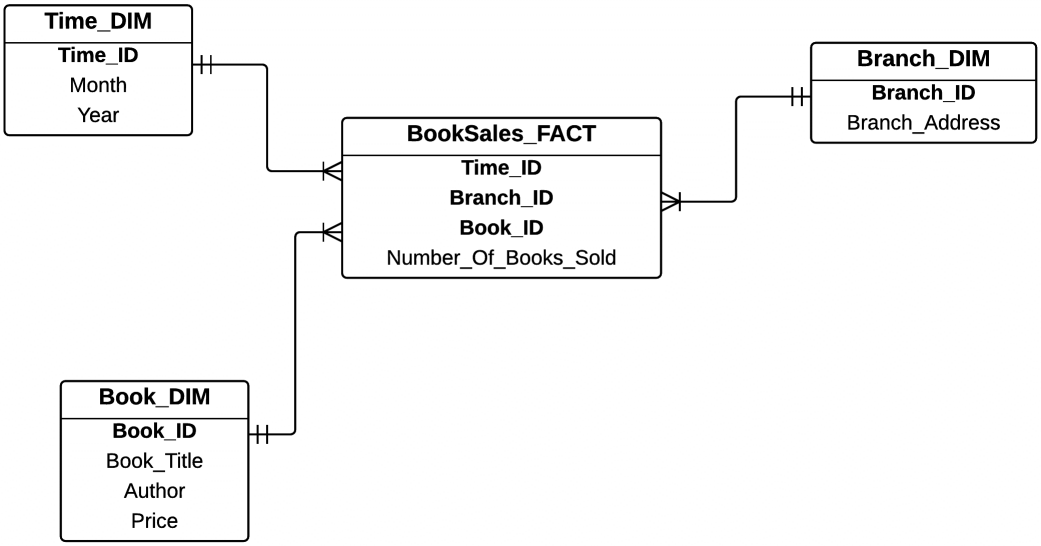

Figure 11.1. The Bookshop Star Schema

The BookSales_Fact Table would probably look like the following (Note that the sales figures are for illustrative purposes only, and the following entries in the tables are not necessarily comprehensive, either).

Table: **BookSales_Fact**

Time_ID	Branch_ID	Book_ID	Number_Of_Books_Sold
Mar2008	City	C1	5
Mar2008	City	H6	15
Mar2008	City	DV	23
Mar2008	City	…	
Mar2008	Chadstone	C1	15
Mar2008	Chadstone	H6	3
Mar2008	Chadstone	DV	2
Mar2008	Chadstone	…	
Mar2008	Camberwell	C1	1
Mar2008	Camberwell	H6	1
Mar2008	Camberwell	DV	2
Mar2008	Camberwell	…	
Mar2008	…	…	
…	…	…	
…	…	…	
Dec2007	City	C1	15
Dec2007	City	H6	6
Dec2007	City	DV	6
Dec2007	City	…	
Dec2007	Chadstone	C1	10
Dec2007	Chadstone	H6	8
Dec2007	Chadstone	DV	1
Dec2007	Chadstone	…	
Dec2007	Camberwell	C1	18
Dec2007	Camberwell	H6	3
Dec2007	Camberwell	DV	2
Dec2007	Camberwell	…	
Dec2007	…	…	
…	…	…	

The fact table can be joined with the dimensions to produce a more comprehensive report. In the example below, the BookSales_Fact table is joined with the Book_DIM table, and consequently the attributes in the Book_DIM table will appear in the report. Assume the Book_DIM table has the following data:

Table: **Book_DIM**

Book_ID	Book Title	Author	Price
C1	CSIRO Diet	CSIRO Team	$45.95
H6	Harry Potter 6	Rowling	$30.95
DV	Da Vinci Code	Dan Brown	$27.95
…	…	…	…

The new report might look like the following (Note that the three grey shaded columns are the additional columns appeared in the report):

Report 1

Time_ID	Branch_ID	Book_ID	Book_Title	Author	Price	Number_Of_ Books_Sold
Mar2008	City	C1	CSIRO Diet	CSIRO Team	$45.95	5
Mar2008	City	H6	Harry Potter 6	Rowling	$30.95	15
Mar2008	City	DV	Da Vinci Code	Dan Brown	$27.95	23
Mar2008	City		…			
Mar2008	Chadstone	C1	CSIRO Diet	CSIRO Team	$45.95	15
Mar2008	Chadstone	H6	Harry Potter 6	Rowling	$30.95	3
Mar2008	Chadstone	DV	Da Vinci Code	Dan Brown	$27.95	2
Mar2008	Chadstone		…			
Mar2008	Camberwell	C1	CSIRO Diet	CSIRO Team	$45.95	1
Mar2008	Camberwell	H6	Harry Potter 6	Rowling	$30.95	1
Mar2008	Camberwell	DV	Da Vinci Code	Dan Brown	$27.95	2
Mar2008	Camberwell		…			
Mar2008	…		…			
…	…		…			
…	…		…			
Dec2007	City	C1	CSIRO Diet	CSIRO Team	$45.95	15
Dec2007	City	H6	Harry Potter 6	Rowling	$30.95	6
Dec2007	City	DV	Da Vinci Code	Dan Brown	$27.95	6
Dec2007	City		…			
Dec2007	Chadstone	C1	CSIRO Diet	CSIRO Team	$45.95	10
Dec2007	Chadstone	H6	Harry Potter 6	Rowling	$30.95	8
Dec2007	Chadstone	DV	Da Vinci Code	Dan Brown	$27.95	1
Dec2007	Chadstone		…			
Dec2007	Camberwell	C1	CSIRO Diet	CSIRO Team	$45.95	18
Dec2007	Camberwell	H6	Harry Potter 6	Rowling	$30.95	3
Dec2007	Camberwell	DV	Da Vinci Code	Dan Brown	$27.95	2
Dec2007	Camberwell		…			
Dec2007	…		…			
…	…		…			

Pay attention to the Price column in Report 1. In this case, the Price column contains the CURRENT PRICE of each book, and it is likely to be the ORIGINAL PRICE. However, from time to time, a book is sold in a discounted price. The above sales report does not reflect that – and therefore, the analysis from this report can be misleading.

In order to incorporate the temporal values of the Price, we need to use a BRIGE TABLE to the Book Dimension, to store the *history* of book prices (which we call Book_Price_DIM). Note that Book_Price_DIM (the Bridge Table) is implemented as a Weak Entity, with a composite primary key of Book_ID, Start_Date, and End_Date (printed in bold in the Book_Price_DIM box, as shown in Figure 11.2).

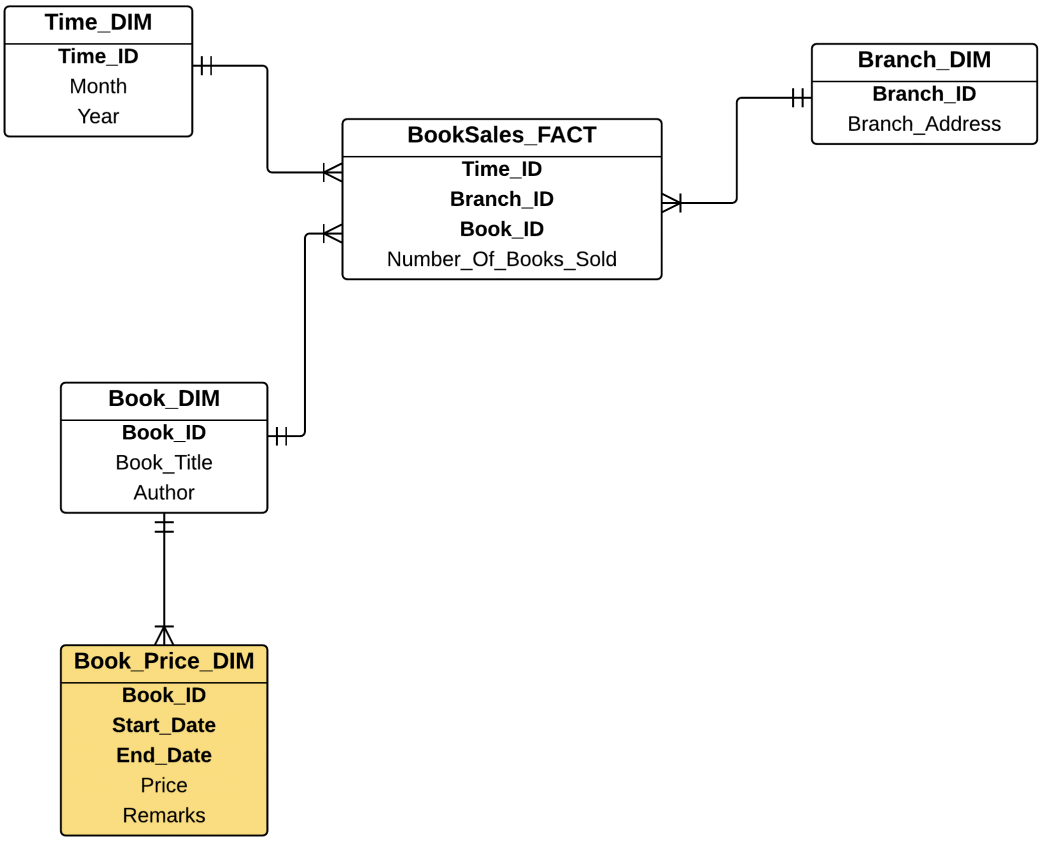

Figure 11.2. Temporal Data Warehousing Star Schema

Hence, the Book_DIM Table and Book_Price_DIM Table may look like the following:

Table: **Book_DIM**

Book_ID	Book Title	Author
C1	CSIRO Diet	CSIRO Team
H6	Harry Potter 6	Rowling
DV	Da Vinci Code	Dan Brown
…	…	…

Table: **Book_Price_DIM**

Book_ID	Start_Date	End_Date	Price	Remarks
C1	Jan2007	July2007	$45.95	Full Price
C1	Aug2007	Oct2007	$36.75	20% Discount
C1	Nov2007	Jan2008	$23.00	Half Price
C1	Feb2008	**Dec9999**	$45.95	Full Price
H6	Jan2007	Mar2007	$21.95	Launching
H6	Apr2007	Jan2008	$30.95	Full Price
H6	Feb2008	**Dec9999**	$10.00	End of Product Sale
DV	Jan2007	**Dec9999**	$27.95	Full Price
…	…	…	…	

In the Book_Price_DIM Table, a TEMPORAL attribute Price is managed, whereby for each price, the period (Start and End Dates) and the remarks are recorded. Note that we use Dec9999 to indicate that the end date is still open until now.

Since each book has a temporal feature (i.e. temporal attribute), the previous Report 1 can be revised to incorporate the correct sale price of the book. Hence, the new report (Report 2) would look like the following. Note that now we can understand why CSIRO Diet was sold more copies in December 2007, as it was half price. Also note that the sale of Harry Potter is already coming to an end (e.g. not much demand). For the DaVinci Code book, the price has been the full price.

Report 2

Time_ID	Branch_ID	Book_ID	Book_Title	Author	Price	Number_Of_Books_Sold
Mar2008	City	C1	CSIRO Diet	CSIRO Team	$45.95	5
Mar2008	City	H6	Harry Potter 6	Rowling	$10.00	15
Mar2008	City	DV	Da Vinci Code	Dan Brown	$27.95	23
Mar2008	City		…			
Mar2008	Chadstone	C1	CSIRO Diet	CSIRO Team	$45.95	15
Mar2008	Chadstone	H6	Harry Potter 6	Rowling	$10.00	3
Mar2008	Chadstone	DV	Da Vinci Code	Dan Brown	$27.95	2
Mar2008	Chadstone		…			
Mar2008	Camberwell	C1	CSIRO Diet	CSIRO Team	$45.95	1
Mar2008	Camberwell	H6	Harry Potter 6	Rowling	$10.00	1
Mar2008	Camberwell	DV	Da Vinci Code	Dan Brown	$27.95	2
Mar2008	Camberwell		…			
Mar2008	…		…			
…	…		…			
…	…		…			
Dec2007	City	C1	CSIRO Diet	CSIRO Team	$23.00	15
Dec2007	City	H6	Harry Potter 6	Rowling	$30.95	6
Dec2007	City	DV	Da Vinci Code	Dan Brown	$27.95	6
Dec2007	City		…			
Dec2007	Chadstone	C1	CSIRO Diet	CSIRO Team	$23.00	10
Dec2007	Chadstone	H6	Harry Potter 6	Rowling	$30.95	8
Dec2007	Chadstone	DV	Da Vinci Code	Dan Brown	$27.95	1
Dec2007	Chadstone		…			
Dec2007	Camberwell	C1	CSIRO Diet	CSIRO Team	$23.00	18
Dec2007	Camberwell	H6	Harry Potter 6	Rowling	$30.95	3
Dec2007	Camberwell	DV	Da Vinci Code	Dan Brown	$27.95	2
Dec2007	Camberwell		…			
Dec2007	…		…			
…	…		…			

2. Implementation of Temporal Data Warehousing

In the previous section, the concept of temporal data warehousing using bridge tables is discussed. This section will discuss the implementation details, using SQL.

Consider the E/R diagram of the operational database as shown in Figure 11.3.

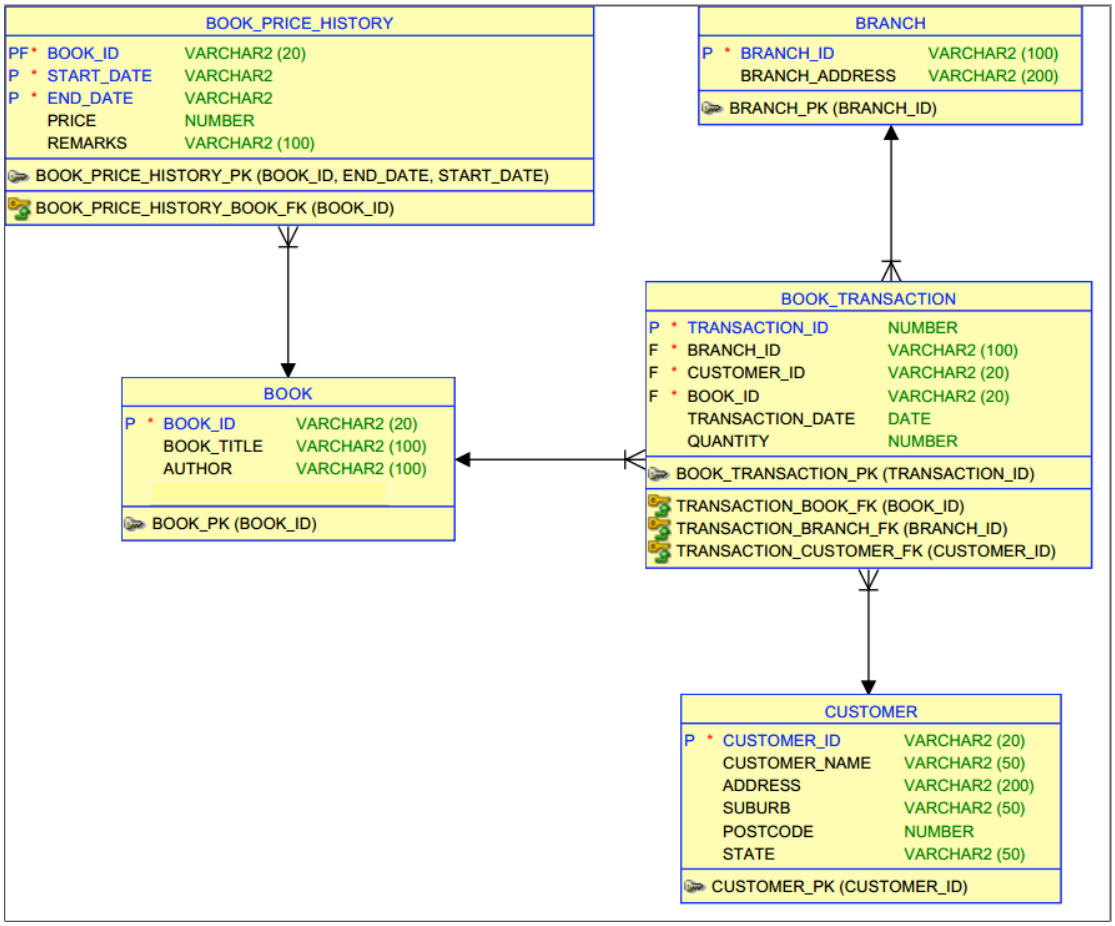

Figure 11.3. E/R Diagram of the Operational Database

The star schema is previously shown in Figure 11.2. The SQL commands to create the dimension tables are as follows:

```
-- BRANCH_DIM
create table BRANCH_DIM as
select *
from BRANCH;

-- BOOK_DIM
create table BOOK_DIM as
select *
from BOOK;

-- TIME_DIM
create table TIME_DIM as
select distinct
  to_char(TRANSACTION_DATE, 'MonYYYY') as TIME_ID,
  to_char(TRANSACTION_DATE, 'Mon') as MONTH,
  to_char(TRANSACTION_DATE, 'YYYY') as YEAR
from BOOK_TRANSACTION;

-- BOOK_PRICE_DIM (TEMPORAL DIMENSION)
create table BOOK_PRICE_DIM as
select distinct *
from BOOK_PRICE_HISTORY;
```

The SQL to create the fact table is as follows:

```
create table BOOK_SALES_FACT1 as
select to_char(T.TRANSACTION_DATE, 'MonYYYY') as TIME_ID,
       BK.BOOK_ID,
       BR.BRANCH_ID,
       sum(T.QUANTITY) as NUMBER_OF_BOOKS_SOLD
from BOOK_TRANSACTION T, BOOK BK, BRANCH BR
where T.BRANCH_ID = BR.BRANCH_ID
and   T.BOOK_ID = BK.BOOK_ID
group by
      to_char(T.TRANSACTION_DATE, 'MonYYYY'),
      BK.BOOK_ID,
      BR.BRANCH_ID;
```

Calculating Number_Of_Books_Sold is quite straightforward; that is sum of attribute quantity from the Book_Transaction table in the operational database.

Supposed we would like to have one additional fact measure, called **Total_Sales**. Therefore, the new star schema is shown in Figure 11.4.

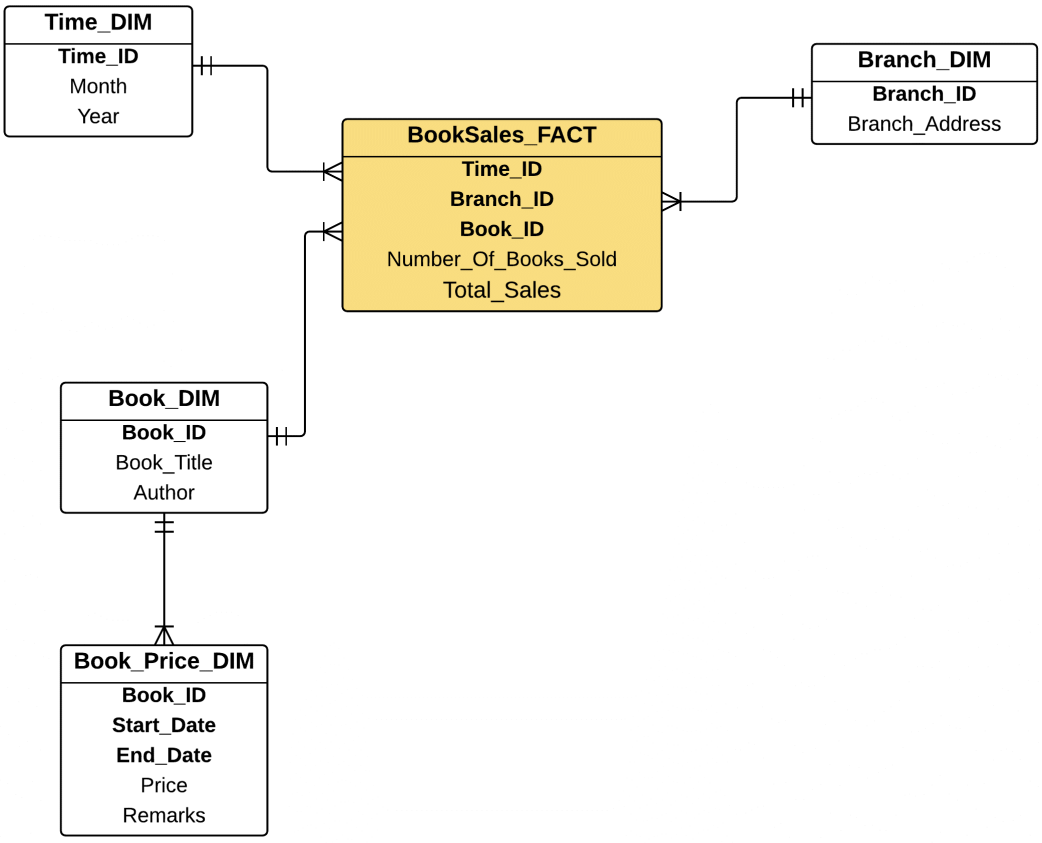

Figure 11.4. Star Schema with a new fact attribute: Total_Sales

The SQL command to create the second fact table (we call it Book_Sales_Fact2) is as follows:

```
create table BOOK_SALES_FACT2 as
select * from BOOK_SALES_FACT1;

alter table BOOK_SALES_FACT2
add (TOTAL_SALES NUMBER);
```

```
declare
  cursor PRICE_CURSOR is
    select *
    from BOOK_PRICE_DIM;
begin
  for ITEM in PRICE_CURSOR loop
    -- update value for TOTAL_SALES in BOOK_SALES_FACT2
    update BOOK_SALES_FACT2
    set    TOTAL_SALES = NUMBER_OF_BOOKS_SOLD * ITEM.PRICE
    where  BOOK_ID = ITEM.BOOK_ID
    and    to_date(TIME_ID, 'MonYYYY') >= to_date(ITEM.START_DATE, 'MonYYYY')
    and    to_date(TIME_ID, 'MonYYYY') <= to_date(ITEM.END_DATE, 'MonYYYY');
  end loop;
end;
/
```

For each record in the Book_Price_DIM table, update the Book_Sales_Fact2 table and calculate the total sales, by multiplying Number_Of_Books_Sold in the Book_Sales_Fact2 and the "correct" price of the Book_Price_DIM (where the Time_ID in the fact should be in between the start_date and end_date of the book price).

The above method basically assumes that Book_Sales_Fact1 has been created. Then Book_Sales_Fact2 is created by copying from Book_Sales_Fact1 and then update it by adding with a new column called Total_Sales_Number.

However, if we do not assume that Book_Sales_Fact1 has been created before, we can create Book_Sales_Fact2 from scratch, using the following create table statement:

```
create table BOOK_SALES_FACT2 as
select
   to_char(T.TRANSACTION_DATE, 'MonYYYY') as TIME_ID,
   BK.BOOK_ID,
   BR.BRANCH_ID,
   sum(T.QUANTITY) as NUMBER_OF_BOOKS_SOLD,
   sum(T.QUANTITY * BP.PRICE) as TOTAL_SALES
from BOOK_TRANSACTION T, BOOK BK, BRANCH BR, BOOK_PRICE_HISTORY BP
where T.BRANCH_ID = BR.BRANCH_ID
and   T.BOOK_ID = BK.BOOK_ID
and   BK.BOOK_ID = BP.BOOK_ID
and   T.TRANSACTION_DATE >= to_date(BP.START_DATE, 'MonYYYY')
and   T.TRANSACTION_DATE <= to_date(BP.END_DATE, 'MonYYYY')
group by
   to_char(T.TRANSACTION_DATE, 'MonYYYY'),
   BK.BOOK_ID,
   BR.BRANCH_ID;
```

3. Temporal Attributes and Temporal Dimensions

3.1. Temporal Attributes

In the case study above, the book price, which is maintained in the Book_Price_DIM table, is called a **Temporal Attribute**. A temporal attribute is an attribute in which the value of that attribute has a life-time. In this example, each book price has a life-

time, and the life time is determined by the Start_Date and the End_Date attributes in the Book_Price_DIM table. For example, the book price of $45.95 of Book C1 is only valid between Jan-2007 and July-2007.

Table Book_Price_DIM is a Relational DBMS (RDBMS) implementation of a temporal data warehousing, which is basically implemented as a Bridge Table. The reason why the Book_Price_DIM is separated from the original Book_DIM is because one book may have many prices in different time periods. Because the relational model does not permit a nested table, the information about the prices of a book has to be separated into another table.

Table: **Book_Price_DIM**

Book_ID	Start_Date	End_Date	Price	Remarks
C1	Jan2007	July2007	$45.95	Full Price
C1	Aug2007	Oct2007	$36.75	20% Discount
C1	Nov2007	Jan2008	$23.00	Half Price
C1	Feb2008	Dec9999	$45.95	Full Price
H6	Jan2007	Mar2007	$21.95	Launching
H6	Apr2007	Jan2008	$30.95	Full Price
H6	Feb2008	Dec9999	$10.00	End of Product Sale
DV	Jan2007	Dec9999	$27.95	Full Price
…	…	…	…	

As stated in the previous section, when we produce a report that joins the fact table with the dimension tables (e.g. join the BookSales_Fact table with the Book_DIM and the Book_Price_DIM tables), we can produce Report 2 that contains all of the attributes from the BookSales_Fact table, with an addition of the Book_Title and Author attributes from the Book_DIM table, and the Price attribute from the Book_Price_DIM table. This report is produced by the following SQL statement:

```
Select
   F.Time_ID,
   F.Branch_ID,
   F.Book_ID,
   B.Book_Title,
   B.Author,
   P.Price,
   F.Number_Of_Books_Sold
From BookSales_Fact F, Book_DIM B, Book_Price_DIM P
Where F.Book_ID = B.Book_ID
And B.Book_ID = P.Book_ID
And to_date(F.Time_ID, 'MonYYYY') >= to_date(P.Start_Date, 'MonYYYY')
And to_date(F.Time_ID, 'MonYYYY') <= to_date(P.End_Date, 'MonYYYY');
```

This SQL command basically joins the three mentioned tables (i.e. BookSales_Fact, Book_DIM, and Book_Price_DIM), and checks if Time_ID (which is month and year) in the BookSales_Fact falls in the period of the book price, as stated by the Start_Date and End_Date in the Book_Price_DIM. As a result, the correct price will be displayed in the report, which matches with the month (i.e. Time_ID) of the fact record.

Report 2

Time_ID	Branch_ID	Book_ID	Book_Title	Author	Price	Number_Of_ Books_Sold
Mar2008	City	C1	CSIRO Diet	CSIRO Team	$45.95	5
Mar2008	City	H6	Harry Potter 6	Rowling	**$10.00**	15
Mar2008	City	DV	Da Vinci Code	Dan Brown	$27.95	23
Mar2008	City		…			
Mar2008	Chadstone	C1	CSIRO Diet	CSIRO Team	$45.95	15
Mar2008	Chadstone	H6	Harry Potter 6	Rowling	**$10.00**	3
Mar2008	Chadstone	DV	Da Vinci Code	Dan Brown	$27.95	2
Mar2008	Chadstone		…			
Mar2008	Camberwell	C1	CSIRO Diet	CSIRO Team	$45.95	1
Mar2008	Camberwell	H6	Harry Potter 6	Rowling	**$10.00**	1
Mar2008	Camberwell	DV	Da Vinci Code	Dan Brown	$27.95	2
Mar2008	Camberwell		…			
Mar2008	…		…			
…	…		…			
…	…		…			
Dec2007	City	C1	CSIRO Diet	CSIRO Team	**$23.00**	15
Dec2007	City	H6	Harry Potter 6	Rowling	$30.95	6
Dec2007	City	DV	Da Vinci Code	Dan Brown	$27.95	6
Dec2007	City		…			
Dec2007	Chadstone	C1	CSIRO Diet	CSIRO Team	**$23.00**	10
Dec2007	Chadstone	H6	Harry Potter 6	Rowling	$30.95	8
Dec2007	Chadstone	DV	Da Vinci Code	Dan Brown	$27.95	1
Dec2007	Chadstone		…			
Dec2007	Camberwell	C1	CSIRO Diet	CSIRO Team	**$23.00**	18
Dec2007	Camberwell	H6	Harry Potter 6	Rowling	$30.95	3
Dec2007	Camberwell	DV	Da Vinci Code	Dan Brown	$27.95	2
Dec2007	Camberwell		…			
Dec2007	…		…			
…	…		…			

However, a problem will arise if the granularity of time that is used by the book price and the fact is different. For example, if the price of Book_ID C1 is $23.00 but not from Nov2007 to Jan2008, but from Nov2007 to 15Jan2008 (consequently the price increased to $45.95 stated from 16Jan2008, instead of Feb2008), then the WHERE clause condition in the above SQL:

```
And to_date(F.Time_ID, 'MonYYYY') >= to_date(P.Start_Date, 'MonYYYY')
And to_date(F.Time_ID, 'MonYYYY') <= to_date(P.End_Date, 'MonYYYY');
```

will produce an incorrect report, because Time_ID 200701 for Jan2007 would match with two records in the Book_Price_DIM: one with price of $23.00 and the other with price of $45.95. Consequently, in the report, there will be two records for book C1 in Jan2008, in each branch: one with the price of $23.00 and the other with the price of $45.95; simply because there are two book prices for the month of January 2008. See Report 3 below.

Table: **Book_Price_DIM**

Book_ID	Start_Date	End_Date	Price	Remarks
C1	Jan2007	July2007	$45.95	Full Price
C1	Aug2007	Oct2007	$36.75	20% Discount
C1	Nov2007	15Jan2008	$23.00	Half Price
C1	16Jan2008	Dec9999	$45.95	Full Price
H6	Jan2007	Mar2007	$21.95	Launching
H6	Apr2007	Jan2008	$30.95	Full Price
H6	Jan2008	Dec9999	$10.00	End of Product Sale
DV	Jan2007	Dec9999	$27.95	Full Price
…	…	…	…	

Report 3: An Incorrect Report

Time_ID	Branch_ID	Book_ID	Book_Title	Author	Price	Number_Of_Books_Sold
…	…		…			
Jan2008	City	C1	CSIRO Diet	CSIRO Team	$23.00	25
Jan2008	City	C1	CSIRO Diet	CSIRO Team	$45.95	25
Jan2008	City	H6	Harry Potter 6	Rowling	$30.95	10
Jan2008	City	DV	Da Vinci Code	Dan Brown	$27.95	7
Jan2008	City		…			
Jan2008	Chadstone	C1	CSIRO Diet	CSIRO Team	$23.00	30
Jan2008	Chadstone	C1	CSIRO Diet	CSIRO Team	$45.95	30
Jan2008	Chadstone	H6	Harry Potter 6	Rowling	$30.95	15
Jan2008	Chadstone	DV	Da Vinci Code	Dan Brown	$27.95	5
Jan2008	Chadstone		…			
Jan2008	Camberwell	C1	CSIRO Diet	CSIRO Team	$23.00	20
Jan2008	Camberwell	C1	CSIRO Diet	CSIRO Team	$45.95	20
Jan2008	Camberwell	H6	Harry Potter 6	Rowling	$30.95	5
Jan2008	Camberwell	DV	Da Vinci Code	Dan Brown	$27.95	5
Jan2008	Camberwell		…			
Jan2008	…		…			
…	…		…			

This problem can only be solved if the granularity of Time_ID in the fact is the same as the Start_Date and End_Date in the Book_Price_DIM. If Time_ID is on the month level, then the period of the book price must be exclusively based on month. On the other hand, if the period of the book price is at a date granularity level, then the Time_ID in the fact must also be at a date granularity level. If the level of granularity is different, the abovementioned problem will happen. Another way to "solve" this problem is by not allowing the report to join with table Book_Price_DIM. In other words, the Book_Price_DIM is purely treated as additional information in the data warehouse, in case if the management wants to drill down certain books.

The above problem is due to the report that joins the fact table with the Book_Price_DIM table; there is no problem with the records in the fact table itself. One might ask what happened with the Total_Sales attribute in the fact table. The calculation for the Total_Sales is still correct, as shown in the following SQL statement:

```
create table BOOK_SALES_FACT2 as
select
   to_char(T.TRANSACTION_DATE, 'MonYYYY') as TIME_ID,
   BK.BOOK_ID,
   BR.BRANCH_ID,
   sum(T.QUANTITY) as NUMBER_OF_BOOKS_SOLD,
   sum(T.QUANTITY * BP.PRICE) as TOTAL_SALES
from BOOK_TRANSACTION T, BOOK BK, BRANCH BR, BOOK_PRICE_HISTORY BP
where T.BRANCH_ID = BR.BRANCH_ID
and   T.BOOK_ID = BK.BOOK_ID
and   BK.BOOK_ID = BP.BOOK_ID
and   T.TRANSACTION_DATE >= BP.START_DATE
and   T.TRANSACTION_DATE <= BP.END_DATE
group by
   to_char(T.TRANSACTION_DATE, 'MonYYYY'),
   BK.BOOK_ID,
   BR.BRANCH_ID;
```

This SQL basically joins the four tables in the operational database, namely Book_Transaction, Book, Branch, and Book_Price_History. While joining these tables, it gets the correct book price by comparing transaction_date with start_date and end_date of the book price. In this case, we assume that start_date and end_date, as well as transaction_date are of a date granularity. Once this is joined, the grouping is then based on month. The individual book total sales is correct, and the grouping is then based on month. Hence, the problem of incorrect report is not due to the incorrect fact, but because of the join between the fact and the dimension that stores the temporal attribute.

3.2. Temporal Dimensions

If a temporal attribute is an attribute where the value has a life-time, a **Temporal Dimension** is a dimension where the record of the dimension has a life time. For example, if Book ID C1 appeared in 2007, and disappeared in 2008, and then re-appeared again in 2009, then the book dimension needs a Temporal Dimension. Or if a branch opens and closes several times, then the branch dimension needs a Temporal Dimension. However, these two examples are not realistic, because very rarely books appear and disappear, and branches open and close. So, in order to highlight Temporal Dimensions, we are going to use more realistic case study.

A Calendar shop has a number of branches. It sells different types of calendars and diaries. Some branches are seasonal; they open a stall in major shopping malls, and these stalls are only open in certain months (some stalls open from October to February, some others may open from November to January). This company also has a small number of permanent shops, which open all year round.

This Calendar shop company wants to analyse their sales, similar to the Bookshop case study – the star schema for the Calendar shop has three dimensions: Time_DIM, Branch_DIM, and Merchandise_DIM (instead of Book_DIM in the Bookshop case study).

For the Merchandise_DIM, if we want to keep track the changes in prices, like we did with the Book_Price_DIM, we could have merchandise price as a Temporal Attribute, in the Merchandise_Price_DIM.

For the Branch_DIM, because some branches have a certain life-time, it needs a Temporal Dimension for the branch, and it is called the Branch_History_DIM. In Branch_History_DIM, there is no temporal attribute. A temporal dimension means that the entire branch record has a temporal element, indicated by the Start and End dates.

The star schema for the Calendar shop is shown in Figure 11.5.

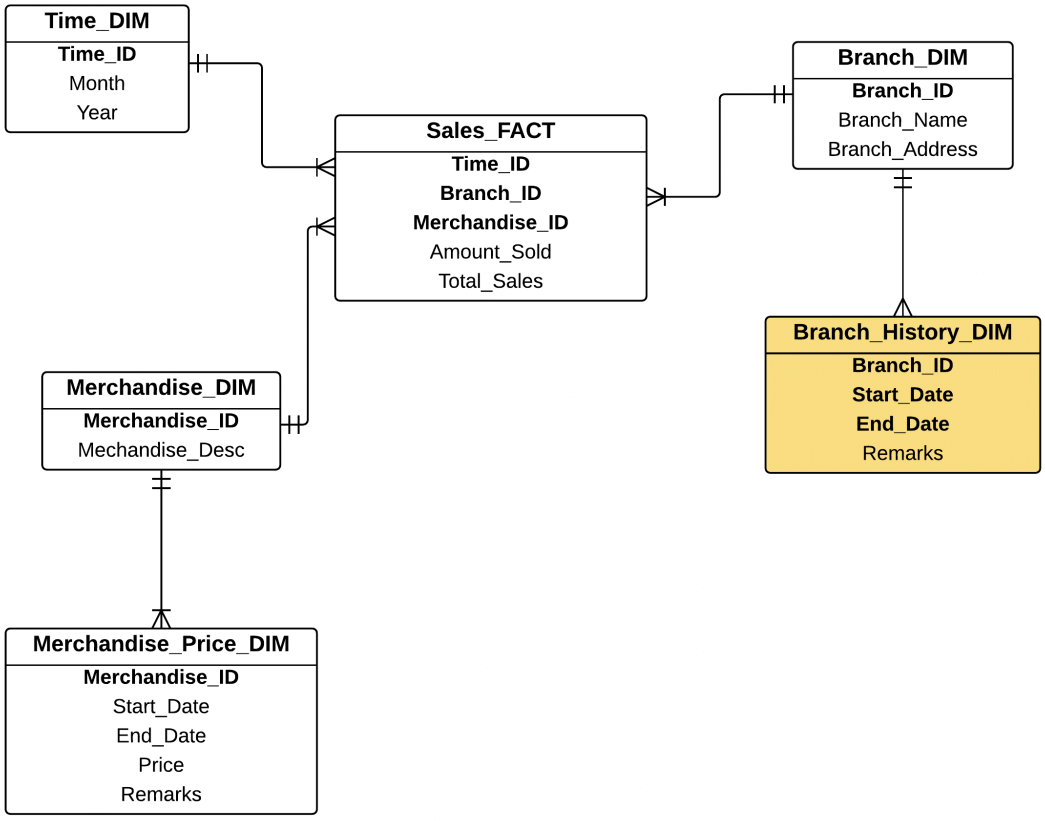

Figure 11.5. Star Schema for the Calendar Shop Case Study

The sample records of the Branch_DIM and Branch_History_DIM are as follows:

Table: **Branch_DIM**

Branch_ID	Branch_Name	Branch_Address
MEL	Melbourne_Central	88 Lonsdale Street
CH	Chadstone_Mall	109 Dandenong Road
DOW	Doncaster_Westfield	75 Doncaster Road
	…	…

Table: **Branch_History_DIM**

Branch_ID	Start_Date	End_Date	Remarks
MEL	Jan0000	Dec9999	Main shop
CH	Oct2007	Mar2008	
CH	Oct2008	Feb2009	
CH	Oct2009	Feb2010	
DOW	Nov2007	Feb2008	
DOW	Nov2008	Feb2009	
DOW	Oct2009	Feb2010	
…	…	…	

In the Branch_History_DIM, the start_date and end_date refers to the branch ID, not to other attributes in the Branch_DIM. This means, for example, the Chadstone branch (Branch ID CH) opens several times in the last few years, as indicated by the start_date and end_date; whereas the Melbourne Central shop (Branch ID MEL) opens all year round. The Remarks attribute is only used to give some remarks for each occurrence of the branch entry in the Branch_History_DIM table.

If we compare Temporal Attribute and Temporal Dimension in this Calendar shop case study, both look very similar. The only difference is that in case of the temporal attribute, there is a temporal attribute in the history dimension table (e.g. book price in the Book_Price_DIM table), whereas in the Temporal Dimension case, there is no such attribute. The attributes in the temporal dimension are only the key identifier of the parent dimension, and possibly any remarks attributes.

4. Slowly Changing Dimensions

Temporal data warehousing is often known as "*Slowly Changing Dimensions*" or *SCD*. Slowly changing dimensions are dimensions where the records of these dimensions change slowly over a period of time. Using the bookshop case study earlier in this chapter, the Book dimension has a price information, and it is common that the price of a book changes "slowly" over time; and that's why the Book dimension is an example of a slowly changing dimension. It is called "slowly" because then changes of price are not, for example, done every hour, or even every week. Over a longer period of time, such as months or even years, the price of a book is changed.

The topic of slowly changing dimensions is different from attributes or records that are "rapidly" changed, such as the location attribute of a moving taxi, which records changes very frequently (e.g. the change of location coordinate is recorded every 1 minute in the operational database), or the price of shares in the stock market, which

may change every a few second, etc. The topic of these rapid changes of records or values is often studied in the area of *Real-Time Data Warehousing* or *Stream Data Warehousing*, which is not the scope of this chapter.

Originally, there are three different types of treatment of SCD, called Type 1, Type2, and Type 3. Each of these types treats an implementation of SCD differently. However, lately, data warehousing practitioners have added with new types, called Type 0, Type 4, and Type 6, which enrich the implementation options for SCD.

4.1. SCD Type 0 and Type 1

SCD Type 0 and Type 1 are quite similar; they do not actually record the history of changes in the dimension.

In **Type 0**, the dimension stores the "Original or Initial" value of the records, when the data warehousing is built. If the value of the dimension attributes changes, the changes are not recorded. The records remain the same as when the records were first inserted into the data warehouse.

Using the Bookshop case study, these values are the "Full Price". For many books, the original or initial price was the full price; but for some books, the price listed initially may not be the full price. So, BookDim table using SCD Type 0, which lists the full prices, is shown as follows:

Table: **Book_DIM (SCD Type 0)**

Book_ID	Book Title	Author	Price
C1	CSIRO Diet	CSIRO Team	$45.95
H6	Harry Potter 6	Rowling	$30.95
DV	Da Vinci Code	Dan Brown	$27.95
…	…	…	…

If the book prices change after that, the new price will not be recorded in the data warehouse.

For other systems, it is more desirable to store the original or initial value, rather than the so called full price as in this example.

Because SCD Type 0 does not record the history of book price, the star schema does not have any temporal dimension, hence, the star schema for the Bookshop case study is the one shown in Figure 11.1.

In **Type 1** does not record the history of changes either. It only records the latest value of the record. Using the Book_DIM example, the book price in the Book_DIM will be the latest price. This means that when there is a change of price, the old price will be overwritten by the new price in the Book_DIM table. The following is the contents of the Book_DIM table using SCD Type 1. In this case, the $10.00 price of Book ID H6 is the latest price. Note that the other two books in this example have the same price as the full price in SCD Type 0, but that does not mean that the prices were never changed. This Book_DIM table only tells us that these are the current price of the books.

Table: **Book_DIM (SCD Type 1)**

Book_ID	Book Title	Author	Price
C1	CSIRO Diet	CSIRO Team	$45.95
H6	Harry Potter 6	Rowling	**$10.00**
DV	Da Vinci Code	Dan Brown	$27.95
…	…	…	…

Like SCD Type 0, SCD Type 1 star schema does not maintain a temporal dimension, and hence the star schema is the one shown in Figure 11.1.

Because SCD Type 0 and Type 1 do not maintain the history of book prices, if we need to produce a report that joins Book_DIM with BookSales_Fact tables, we need to be careful not to draw an association between the book price from Book_DIM and the Time_ID from the BookSales_Fact, because the book price in the report may not necessarily the book price at that particular Time_ID. In such a report, the column title for the book price can be changed to "original book price" (for SCD Type 0), or "latest book price" (for SCD Type 1), in order to avoid any misunderstanding in interpreting the report.

4.2. SCD Type 2

SCD Type 2 keeps track the history, but not separating the history from the main dimension; instead the new records keep added to the dimension. Using the Book_DIM dimension as an example, when the price of a book is changed, it creates "another book" with the same details, except with the new Book ID, and of course with the new price. In addition to the start_date and end_date, it also has a Current_Flag, to indicate whether a record is the current record or a past record. Any additional information, such as the remarks, may also be included. Hence, the Book_DIM table for SCD Type 2 is as follows:

Table: **Book_DIM (SCD Type 2)**

Book_ID	Book Title	Author	Start_Date	End_Date	Price	Remarks	Current _Flag
C1_1	CSIRO Diet	CSIRO Team	Jan2007	July2007	$45.95	Full Price	N
C1_2	CSIRO Diet	CSIRO Team	Aug2007	Oct2007	$36.75	20% Discount	N
C1_3	CSIRO Diet	CSIRO Team	Nov2007	Jan2008	$23.00	Half Price	N
C1_4	CSIRO Diet	CSIRO Team	Feb2008	Dec9999	$45.95	Full Price	Y
H6_1	Harry Potter 6	Rowling	Jan2007	Mar2007	$21.95	Launching	N
H6_2	Harry Potter 6	Rowling	Apr2007	Jan2008	$30.95	Full Price	N
H6_3	Harry Potter 6	Rowling	Feb2008	Dec9999	$10.00	End of Product Sale	Y
DV_1	Da Vinci Code	Dan Brown	Jan2007	Dec9999	$27.95	Full Price	Y
…			…	…	…		

Note that the same book has a different Book_ID for different book price and period. Normally, the Book_ID attribute is implemented as a Surrogate Key; but in this example, we just added with a sequence number to the original Book_ID to

differentiate the same book in different time period. Because of these multiple Book_ID for the same book, the report that joins the BookSales_Fact and the Book_DIM will look like as follows:

Report 3 (SCD Type 2)

Time_ID	Branch_ID	Book_ID	Book_Title	Author	Price	Number_Of_ Books_Sold
Mar2008	City	**C1_4**	CSIRO Diet	CSIRO Team	$45.95	5
Mar2008	City	**H6_3**	Harry Potter 6	Rowling	$10.00	15
Mar2008	City	**DV_1**	Da Vinci Code	Dan Brown	$27.95	23
Mar2008	City		…			
Mar2008	Chadstone	**C1_4**	CSIRO Diet	CSIRO Team	$45.95	15
Mar2008	Chadstone	**H6_3**	Harry Potter 6	Rowling	$10.00	3
Mar2008	Chadstone	**DV_1**	Da Vinci Code	Dan Brown	$27.95	2
Mar2008	Chadstone		…			
Mar2008	Camberwell	**C1_4**	CSIRO Diet	CSIRO Team	$45.95	1
Mar2008	Camberwell	**H6_3**	Harry Potter 6	Rowling	$10.00	1
Mar2008	Camberwell	**DV_1**	Da Vinci Code	Dan Brown	$27.95	2
Mar2008	Camberwell		…			
Mar2008	…		…			
…	…		…			
…	…		…			
Dec2007	City	**C1_3**	CSIRO Diet	CSIRO Team	$23.00	15
Dec2007	City	**H6_2**	Harry Potter 6	Rowling	$30.95	6
Dec2007	City	**DV_1**	Da Vinci Code	Dan Brown	$27.95	6
Dec2007	City		…			
Dec2007	Chadstone	**C1_3**	CSIRO Diet	CSIRO Team	$23.00	10
Dec2007	Chadstone	**H6_2**	Harry Potter 6	Rowling	$30.95	8
Dec2007	Chadstone	**DV_1**	Da Vinci Code	Dan Brown	$27.95	1
Dec2007	Chadstone		…			
Dec2007	Camberwell	**C1_3**	CSIRO Diet	CSIRO Team	$23.00	18
Dec2007	Camberwell	**H6_2**	Harry Potter 6	Rowling	$30.95	3
Dec2007	Camberwell	**DV_1**	Da Vinci Code	Dan Brown	$27.95	2
Dec2007	Camberwell		…			
Dec2007	…		…			
…	…		…			

Note that for example, the first record in Report 3, the Book_ID is C1_4, because this is the Book_ID of CSIRO Diet book in March 2008.

Because SCD Type 2 only changes the original Book_DIM dimension, the star schema looks similar to that of Figure 11.1, but Book_DIM has additional attributes. See Figure 11.6.

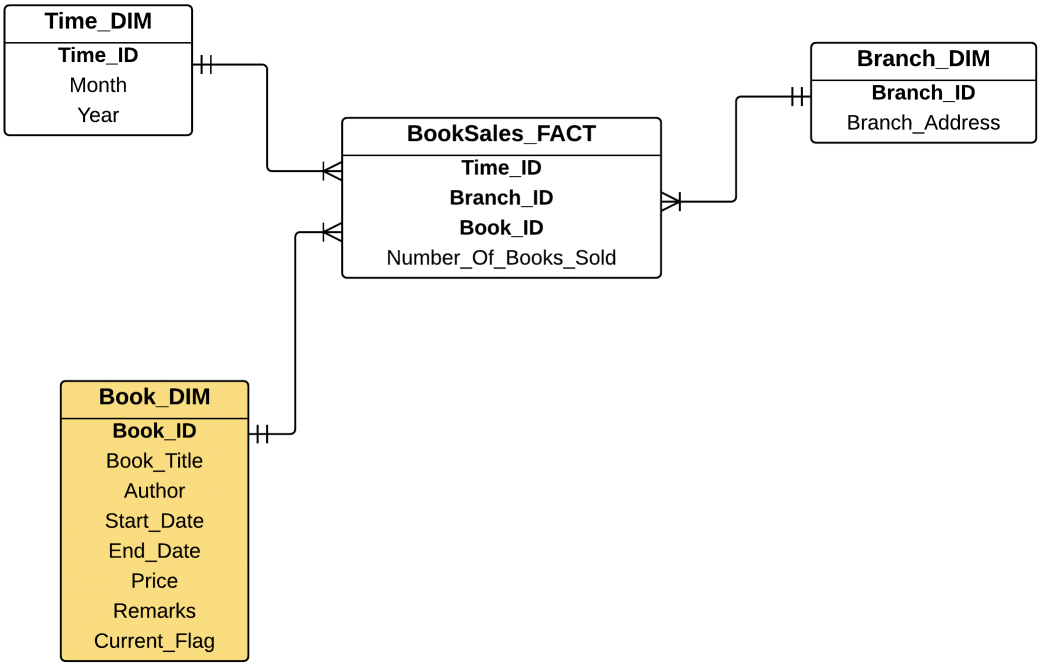

Figure 11.6. The Bookshop Star Schema with SCD Type 2 for Book_DIM

4.3. SCD Type 3

SCD Type 3 is a simplification of Type 2. Unlike Type 2 that maintains multiple records of the same book, Type 3 does not have multiple records for the same book. One book has one entry in the Book_DIM table. For the price, it only records the current price and the previous price. In other words, it does not maintain the entire history of price changes; it only keeps the last two prices. Therefore, the Book_DIM table for SCD Type 3 looks as follows. Note that for the book that does not have any previous price, a Null value will be recorded in the Previous Price column.

Table: **Book_DIM (SCD Type 3)**

Book_ID	Book Title	Author	Current_Price	Previous_Price
C1	CSIRO Diet	CSIRO Team	**$45.95**	**$23.00**
H6	Harry Potter 6	Rowling	**$10.00**	**$30.95**
DV	Da Vinci Code	Dan Brown	**$27.95**	**Null**
…	…	…	…	

The main rationale for adopting SCD Type 3 is that most of the analysis will be based on the current price and at most one past price, which is the previous price. Perhaps, this is for comparison with the trend when the price was the previous price. It is assumed that analysing the complete history is not necessary.

Additionally, although we could store when the price was changed, the original SCD Type 3 does not record this. It only records the current and the previous prices. Without keeping the date when the price was changed, it is not possible then to correlate the book price with the Time_ID information in the BookSales_Fact table. Consequently, the report that can be produced will need to particularly pay attention

to the potential mismatch between the information in the book price column (from the Book_DIM table) and the Time_ID column (from the BookSales_Fact table)

The star schema for SCD Type 3 is shown in Figure 11.7.

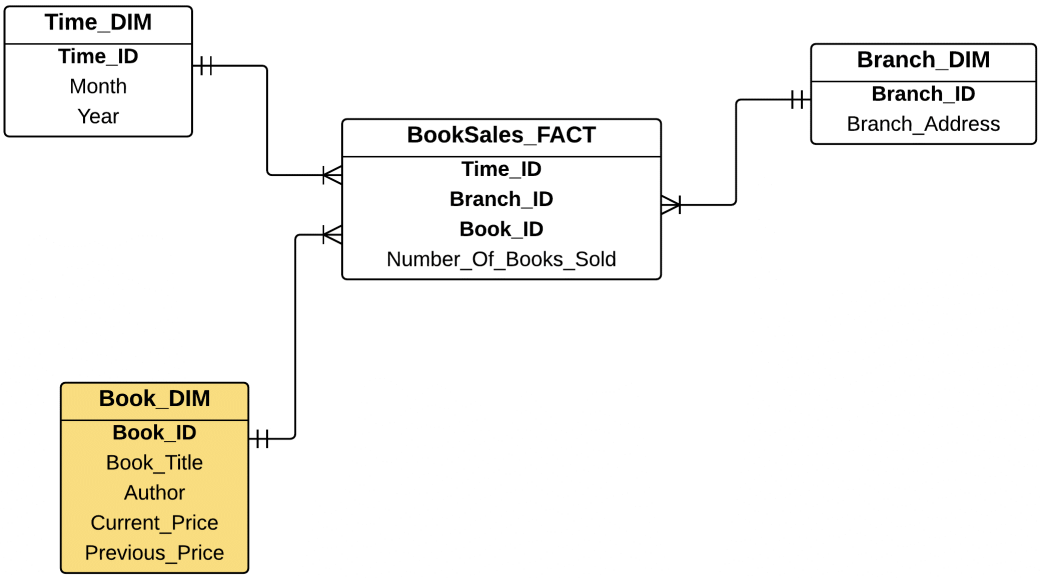

Figure 11.7. The Bookshop Star Schema with SCD Type 3 for Book_DIM

4.4. SCD Type 4

SCD Type 4 is a new way to treat SCD that was not in the original SCD theory. In SCD Type 4, we create a new dimension to maintain the history of attribute value change. Basically, the temporal dimension that is described in the beginning of this chapter is SCD Type 4.

In Type 4, the original Book_DIM is kept without the price attribute. The price attribute (and the start_date and end_dates) are separated into another table; this is the Book_Price_DIM table.

Hence, the Book_DIM and the Book_Price_DIM tables are as follows:

Table: **Book_DIM**

Book_ID	Book Title	Author
C1	CSIRO Diet	CSIRO Team
H6	Harry Potter 6	Rowling
DV	Da Vinci Code	Dan Brown
…	…	…

Table: **Book_Price_DIM (SCD Type 4)**

Book_ID	Start_Date	End_Date	Price	Remarks
C1	Jan2007	July2007	$45.95	Full Price
C1	Aug2007	Oct2007	$36.75	20% Discount
C1	Nov2007	Jan2008	$23.00	Half Price
C1	Feb2008	Dec9999	$45.95	Full Price
H6	Jan2007	Mar2007	$21.95	Launching
H6	Apr2007	Jan2008	$30.95	Full Price
H6	Feb2008	Dec9999	$10.00	End of Product Sale
DV	Jan2007	Dec9999	$27.95	Full Price
…	…	…	…	

The main advantage of Type 4 is that we do not need to have a different Book_ID for the same book. Additionally, the entire history of changes is kept. As shown earlier, this method will guarantee that the report that joins the information from the BookSales_Fact table and the dimension tables will be accurate reflecting the correct book price at certain Time_ID.

The star schema for SCD Type 4 is shown previous in Figure 11.2.

4.5. SCD Type 6

SCD Type 6 is actually a combination between Type 2 and Type 3. In Type 3, only the current price and the previous price are recorded; not the entire history. In Type 2, the entire history of changes is maintained, but a separate identifier (e.g. Surrogate Key is needed).

In Type 6, a separate identifier for the same book is not needed (like Type 3), but the entire history is kept (like Type 2). Hence, SCD Type 6 for the Book_DIM table is as follows:

Table: **Book_DIM (SCD Type 6)**

Book_ID	Book Title	Author	Start_Date	End_Date	Current _Price	Previous _Price	Remarks	Current _Flag
C1	CSIRO Diet	CSIRO Team	Jan2007	July2007	$45.95	Null	Full Price	N
C1	CSIRO Diet	CSIRO Team	Aug2007	Oct2007	$36.75	$45.95	20% Discount	N
C1	CSIRO Diet	CSIRO Team	Nov2007	Jan2008	$23.00	$36.75	Half Price	N
C1	CSIRO Diet	CSIRO Team	Feb2008	Dec9999	$45.95	$23.00	Full Price	Y
H6	Harry Potter 6	Rowling	Jan2007	Mar2007	$21.95	Null	Launching	N
H6	Harry Potter 6	Rowling	Apr2007	Jan2008	$30.95	$21.95	Full Price	N
H6	Harry Potter 6	Rowling	Feb2008	Dec9999	$10.00	$30.95	End of Product Sale	Y
DV	Da Vinci Code	Dan Brown	Jan2007	Dec9999	$27.95	Null	Full Price	Y
…			…	…		…		

In Type 6, there is no need to maintain a separate history table. The history itself is kept in the original dimension table. The star schema for SCD Type 6 is shown in Figure 11.8. Note that the Book_DIM table has a composite key comprising of Book_ID, Start_Date, and End_Date.

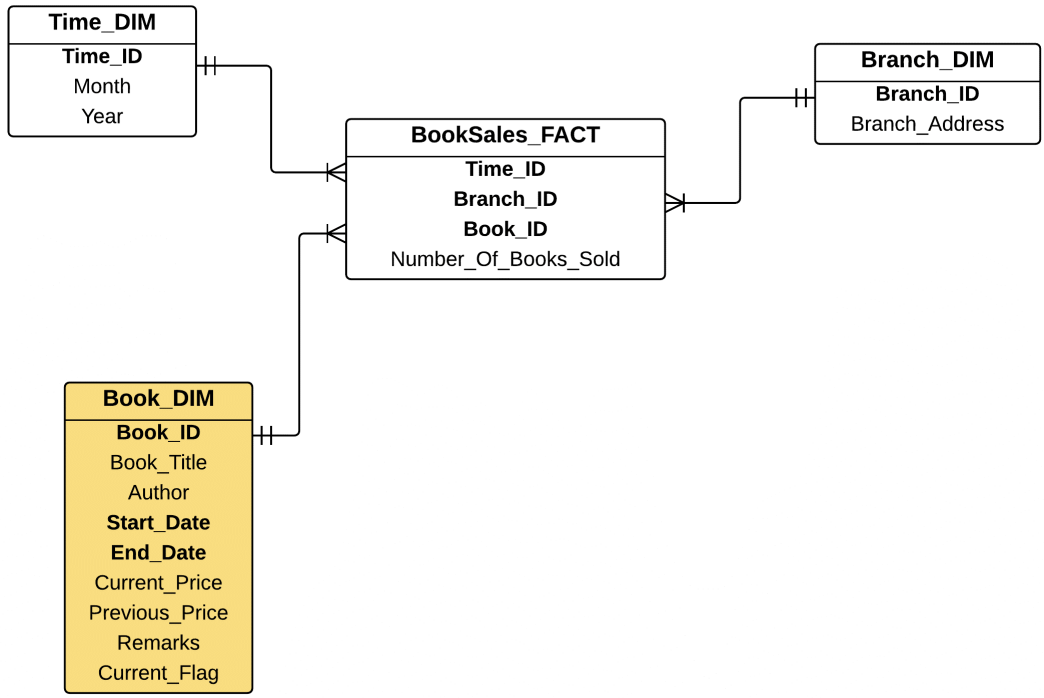

Figure 11.8. The Bookshop Star Schema with SCD Type 6 for Book_DIM

5. Summary

In this chapter, we focus on incorporating historical data in the data warehouse. This is called *Temporal Data Warehousing*. A temporal data warehousing uses the concept of the Bridge Table (or a Weak Entity), where the history is maintain in a bridge table.

Maintaining the history of certain attributes is important in order to make associative analysis more accurate when analysing the reports produced by the fact and dimensions. However, certain degree of caution when joining the fact table and the temporal dimension, especially when the level of granularity of time between the fact and the temporal dimension is not the same.

Temporal data warehousing is also known as *Slowly Changing Dimensions* (SCD). In this chapter, various treatment and types for SCD are presented. Different types will server different purposes of the data warehousing.