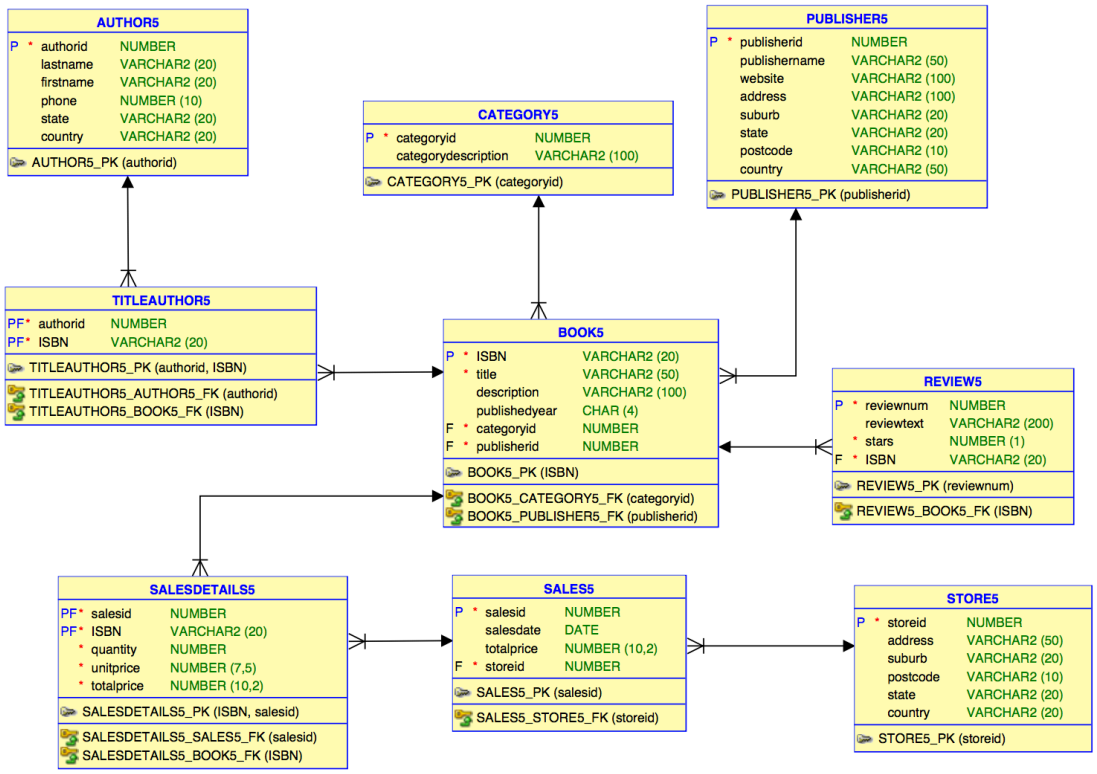
solution

# Tutorial 7
## Case Study # (BOOK SALES)

## 1. A Book Sales Case Study – Description

The following is an E/R diagram of an operational database. The system stores information about books, including the authors, publishers, book categories, as well as the reviews that each book has received. The 'stars' attribute in the Review entity records the star rating for each review (e.g. 5 stars for excellent to 1 star for poor, etc). One book may receive many reviews. For simplicity, it is assumed, as also shown in the E/R diagram, that a book will only have one category.

The E/R diagram also includes entities related to sales of books, and the stores which sale the books. Each store has many sales transactions (i.e. the Sales entity), and each sales transaction may include several books (i.e. the SalesDetails entity). The TotalPrice attribute in the SalesDetails entity is basically quantity multiplied by unitprice, whereas the TotalPrice attribute in the Sales entity is the total price for each sales transaction.



The tables form the operational database are as follows and they are in **DTANIAR** account:

DTANIAR.AUTHOR5 (**authorid**, lastname, firstname, phone, state, country)
DTANIAR.BOOK5(**ISBN**, title, description, publishedyear, categoryid, publisherid)
DTANIAR.TITLEAUTHOR5(**authorid**, ISBN)
DTANIAR.CATEGORY5(**categoryid**, categorydescription)
DTANIAR.PUBLISHER5(**publisherid**, publishername, website, address, suburb, state, postcode, country)
DTANIAR.REVIEW5(**reviewnum**, reviewtext, stars, ISBN)
DTANIAR.STORE5(**storied**, address, suburb, postcode, state, country)
DTANIAR.SALES5(**salesid**, salesdate, totalprice, storied)
DTANIAR.SALESDETAILS5(**salesid**, **ISBN**, quantity, unitprice, totalprice)

You are required to design a small data warehouse for analysis purposes. The analysis is needed for identifying at least the following questions:

- What are the *total sales* for each bookstore in a month?
- What is the *number of books* sold for each category?
- What is the book category that has the highest *total sales*?
- What is the *number of reviews* for each category?
- *How many* 5-star *reviews* for each category?

# 2. Solution Model – using multi-fact

Based on the above requirements, it is clear that there are three fact measures:

1. Total sales,
2. Number of books sold, and
3. Number of reviews

*Total sales* and *number of books sold* would be useful for the management to understand how the book sales perform. *Number of reviews* would be useful for the marketing department to understand people's perception on certain books and would be able to use this information to launch any marketing campaign.

Potentially, there can be many dimensions. However, based on the above requirements, we limit the dimensions to the following four dimensions:

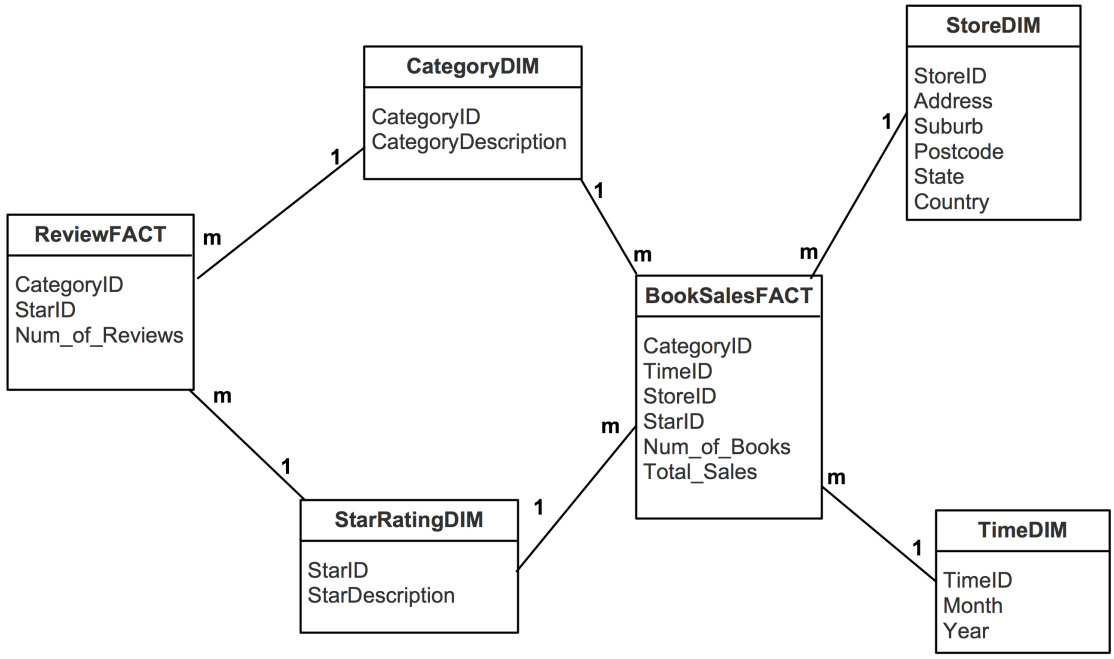
1. Store dimension,
2. Time dimension,
3. Book category dimension, and
4. Star rating dimension.

In summary, the first subject: "book sales" will be the first fact, with all the four dimensions (e.g. Store, Time, Category, and Star Rating dimensions). The fact measures are total sales, and number of books sold.

The second subject is "reviews", with only two dimensions: Category, and Star Rating dimensions. There is only one fact measure, which is number of reviews.

The star schema will then have two fact entities: Book sales fact, and Review fact. Note that the category and star rating dimensions are shared by the two facts.

The multi-fact star schema for this case study is as follows:




*Your tasks:*

a. Create a dimension table called TimeDim.

```
create table timedim as select distinct
to_char(salesdate, 'yyyymm') as timeid,
to_char(salesdate,'mm') as month,
to_char(salesdate,'yyyy') as year
from dtaniar.sales5;
```

b. Create a dimension table called StoreDim.

```
create table storedim as select * from dtaniar.store5;
```

c. Create a dimension table called CategoryDim.

```
create table categorydim as select * from dtaniar .category5;
```

d. Create a dimension table called StarRatingDim. This table will have 6 records (0 for 'Unknown', 1 for 'Poor', 2 for 'Not Good', 3 for 'Average', 4 for 'Good', 5 for 'Excellent')

```
Create Table StarRatingDim
( StarID Number(1),
  StarDescription Varchar2(15));

Insert Into StarRatingDim Values (0, 'Unknown');
Insert Into StarRatingDim Values (1, 'Poor');
Insert Into StarRatingDim Values (2, 'Not Good');
Insert Into StarRatingDim Values (3, 'Average');
Insert Into StarRatingDim Values (4, 'Good');
Insert Into StarRatingDim Values (5, 'Excellent');
```

e. Create a fact table called ReviewFact.

```
create table reviewfact as
select
   b.CATEGORYID,
   r.STARS as starid,
   count(*) as num_of_review
from dtaniar.book5 b, dtaniar.review5 r
where b.isbn=r.isbn
group by b.CATEGORYID, r.STARS;
```

f. Create a tempfact table called TempBookWithStar. (NOTE: Books without review should be included in the result, and it should get a ZERO in the Star).

```
Create table TempBookWithStar as
Select
  B.ISBN,
  B.CategoryID,
  NVL(R.Stars, 0) As Star
From dtaniar.book5 B, dtaniar.review5 R
Where B.ISBN = R.ISBN(+);

--select * from TempBookWithStar where ISBN='0316465186';
Book 0316465186 is the one that does not have any reviews, and the star is 0
```

g. Create a tempfact table called TempBookWithAvgStar. (NOTE: Calculate the average of stars by each book and each category using TempBookWithStar. The average of stars should be an integer either 0 or 1,2,3,4,5).

```
Create table TempBookWithAvgStar as
Select ISBN, CategoryID, Round(Avg(Star)) As Avg_Star
From TempBookWithStar
Group By ISBN, CategoryID;
```

h. Create the final fact table called BookSalesFact. (NOTE: starid in the fact table is the average of stars).

```
Create Table BookSalesFact as
Select
  T.CategoryID,
  To_Char(S.SalesDate, 'YYYYMM') As TimeID,
  S.StoreID,
  T.Avg_star as StarID,
  sum(sd.quantity) as Num_of_Books,
  Sum(SD.TotalPrice) As Total_Sales
From
  TempBookWithAvgStar T, dtaniar.Sales5 S, dtaniar.SalesDetails5 SD
Where T.ISBN=sd.ISBN
and sd.SALESID= s.SALESID
Group by
  T.CategoryID,
  To_Char(S.SalesDate, 'YYYYMM'),
  S.StoreID,
  T.Avg_star;
```

Hints:

1. Please make sure there are 6 required records in StarRatingDim. Think about that can you create it by coping REVIEW table or create it manually.
2. Be careful to create the table TempBookWithStar. The tempfact table should include the books without reviews, and the stars should be a ZERO. Thus, we need to use outer join to create the table.
3. After creating the TempBookWithStar, display the table and observe the contents of the table. Try to find the books without reviews.
4. After creating the TempBookWithAvgStar, display the table and observe the contents of the table. So far, we have the average of stars by each book.
5. Now, you are ready to create table BookSalesFact using the average of stars created by the table TempBookWithAvgStar, and other attributes in the operation database.

Now, answer the questions using the dimension tables and fact tables you have created.

- What are the *total sales* for each bookstore in a month?

```
select s.STOREID, t.MONTH, sum(f.total_sales) as total_sales
from storedim s, timedim t, booksalesfact f
where s.STOREID=f.STOREID
and f.timeid = t.timeid
group by s.STOREID, t.MONTH
order by s.STOREID, t.MONTH;
```

- What is the *number of books* sold for each category?

```
select c.CATEGORYID, c.CATEGORYDESCRIPTION,
       sum(f.num_of_books) as total_num_books
from booksalesfact f, categorydim c
where f.categoryid = c.categoryid
group by c.CATEGORYID, c.CATEGORYDESCRIPTION
order by c.CATEGORYID, c.CATEGORYDESCRIPTION;
```

- What is the book category that has the highest *total sales*?

```
select * from (
select c.CATEGORYID, c.CATEGORYDESCRIPTION,
       sum(f.num_of_books) as total_num_books
from booksalesfact f, categorydim c
where f.categoryid = c.categoryid
group by c.CATEGORYID, c.CATEGORYDESCRIPTION
order by total_num_books desc)
where rownum =1;
```

- What is the *number of reviews* for each category?

```
select c.CATEGORYID, c.CATEGORYDESCRIPTION,
       sum(f.num_of_review) as number_of_reviews
from reviewfact f, categorydim c
where f.categoryid = c.categoryid
group by c.CATEGORYID, c.CATEGORYDESCRIPTION
order by c.CATEGORYID, c.CATEGORYDESCRIPTION;
```

- *How many* 5-star *reviews* for each category?

```
Select c.CATEGORYID, c.CATEGORYDESCRIPTION,
       sum(f.NUM_OF_REVIEW) as number_of_5star_reviews
from reviewfact f, categorydim c, starratingdim s
where f.categoryid = c.categoryid
and s.starid = f.starid
and s.STARID=5
group by c.CATEGORYID, c.CATEGORYDESCRIPTION
order by c.CATEGORYID, c.CATEGORYDESCRIPTION;
```