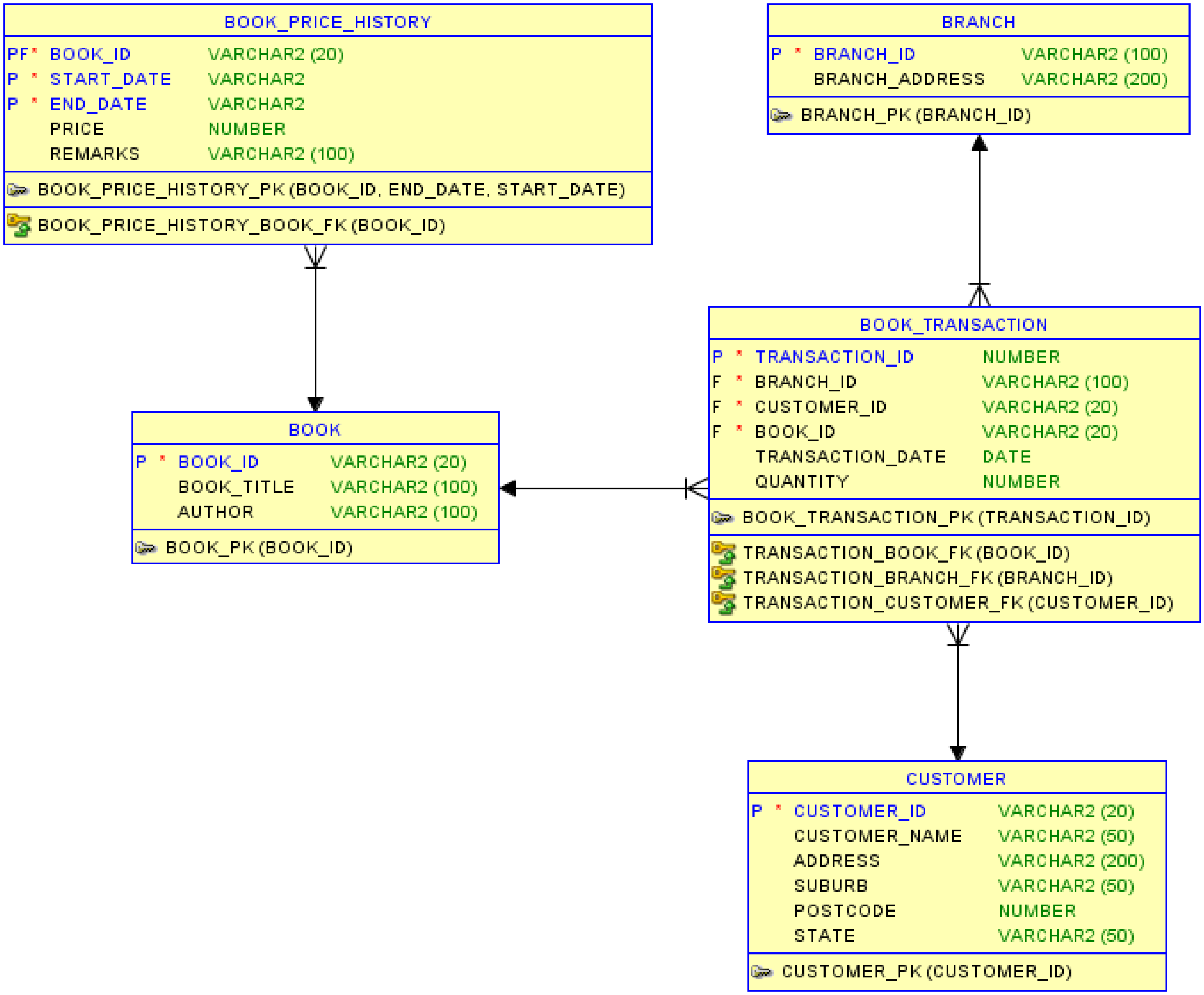
Laboratory 7

The Bookshop Case Study

SOLUTIONS

1. A Bookshop Case Study – Description

Border bookstore that has a number of branches in Melbourne would like to build a data warehouse to analyze their book sales. They have already stored all book sales transactions in an operational database. The management would particularly like to analyze their book sales performance from various perspectives, such as monthly basis, book basis, and branch basis. The following is an E/R diagram of the Bookshop system.



BOOK_PRICE_HISTORY		
PF*	BOOK_ID	VARCHAR2 (20)
P *	START_DATE	VARCHAR2
P *	END_DATE	VARCHAR2
	PRICE	NUMBER
	REMARKS	VARCHAR2 (100)
	BOOK_PRICE_HISTORY_PK (BOOK_ID, END_DATE, START_DATE)	
	BOOK_PRICE_HISTORY_BOOK_FK (BOOK_ID)	

BRANCH		
P *	BRANCH_ID	VARCHAR2 (100)
	BRANCH_ADDRESS	VARCHAR2 (200)
	BRANCH_PK (BRANCH_ID)	

BOOK_TRANSACTION		
P *	TRANSACTION_ID	NUMBER
F *	BRANCH_ID	VARCHAR2 (100)
F *	CUSTOMER_ID	VARCHAR2 (20)
F *	BOOK_ID	VARCHAR2 (20)
	TRANSACTION_DATE	DATE
	QUANTITY	NUMBER
	BOOK_TRANSACTION_PK (TRANSACTION_ID)	
	TRANSACTION_BOOK_FK (BOOK_ID)	
	TRANSACTION_BRANCH_FK (BRANCH_ID)	
	TRANSACTION_CUSTOMER_FK (CUSTOMER_ID)	

BOOK		
P *	BOOK_ID	VARCHAR2 (20)
	BOOK_TITLE	VARCHAR2 (100)
	AUTHOR	VARCHAR2 (100)
	BOOK_PK (BOOK_ID)	

CUSTOMER		
P *	CUSTOMER_ID	VARCHAR2 (20)
	CUSTOMER_NAME	VARCHAR2 (50)
	ADDRESS	VARCHAR2 (200)
	SUBURB	VARCHAR2 (50)
	POSTCODE	NUMBER
	STATE	VARCHAR2 (50)
	CUSTOMER_PK (CUSTOMER_ID)	

First you need to create the operational database using the following SQL:

```
--CUSTOMER
create table CUSTOMER as
select CUSTOMERID as CUSTOMER_ID, name as CUSTOMER_NAME, ADDRESS,
SUBURB, POSTCODE, STATE
from  DTANIAR.CUSTOMER4;
alter table CUSTOMER add constraint CUSTOMER_PK primary key
( CUSTOMER_ID ) ;

--BOOK
create table BOOK
  (
    BOOK_ID    varchar2(20) not null ,
    BOOK_TITLE varchar2(200),
    AUTHOR     varchar2(200)
  ) ;
alter table BOOK add constraint BOOK_PK primary key ( BOOK_ID ) ;
insert into BOOK values('C1', 'CSIRO Diet', 'CSIRO Team');
insert into BOOK values('H6', 'Harry Potter 6', 'Rowling');
insert into BOOK values('DV', 'Da Vinci Code', 'Dan Brown');

--BOOK PRICE HISTORY
create table BOOK_PRICE_HISTORY
  (
    BOOK_ID    varchar2(20) not null ,
    START_DATE varchar2(10) null ,
    END_DATE   varchar2(10) not null ,
    PRICE      number ,
    REMARKS    varchar2(100)
  ) ;
alter table BOOK_PRICE_HISTORY add constraint BOOK_PRICE_HISTORY_PK
primary key ( BOOK_ID, START_DATE, END_DATE ) ;
alter table BOOK_PRICE_HISTORY add constraint BOOK_PRICE_HISTORY_BOOK_FK
foreign key ( BOOK_ID ) references BOOK ( BOOK_ID ) ;
insert into BOOK_PRICE_HISTORY values('C1', 'Jan2007', 'Jul2007', 45.95,
'Full Price');
insert into BOOK_PRICE_HISTORY values('C1', 'Aug2007', 'Oct2007', 36.75,
'20% Discount');
insert into BOOK_PRICE_HISTORY values('C1', 'Nov2007', 'Jan2008', 23.00,
'Half Price')
insert into BOOK_PRICE_HISTORY values('C1', 'Feb2008', 'Now', 45.95,
'Full Price');
insert into BOOK_PRICE_HISTORY values('H6', 'Jan2007', 'Mar2007', 21.95,
'Launching');
insert into BOOK_PRICE_HISTORY values('H6', 'Apr2007', 'Feb2008', 30.95,
'Full Price');
insert into BOOK_PRICE_HISTORY values('H6', 'Jan2008', 'Now', 10.00,
'End of Product Sale');
```

```
insert into BOOK_PRICE_HISTORY values('DV', 'Jan2007', 'Now', 27.95,
'Full Price');
--BRANCH
create table BRANCH
  (
    BRANCH_ID      varchar2(100) not null ,
    BRANCH_ADDRESS varchar2(200)
  ) ;
alter table BRANCH add constraint BRANCH_PK primary key ( BRANCH_ID ) ;
insert into BRANCH values('City', 'VIC3622');
insert into BRANCH values('Chadstone', 'Chadstone VIC3234');
insert into BRANCH values('Camberwell', 'Camberwell VIC2451');

--TRANSACTION
create table BOOK_TRANSACTION
  (
    TRANSACTION_ID   number not null ,
    BRANCH_ID        varchar2 (100) not null ,
    CUSTOMER_ID      varchar2 (20) not null ,
    BOOK_ID          varchar2 (20) not null ,
    TRANSACTION_DATE date ,
    QUANTITY         number
  ) ;
alter table BOOK_TRANSACTION add constraint BOOK_TRANSACTION_PK primary
key ( TRANSACTION_ID ) ;
alter table BOOK_TRANSACTION add constraint TRANSACTION_BOOK_FK foreign
key ( BOOK_ID ) references BOOK ( BOOK_ID ) ;
alter table BOOK_TRANSACTION add constraint TRANSACTION_BRANCH_FK
foreign key ( BRANCH_ID ) references BRANCH ( BRANCH_ID ) ;
alter table BOOK_TRANSACTION add constraint TRANSACTION_CUSTOMER_FK
foreign key ( CUSTOMER_ID ) references CUSTOMER ( CUSTOMER_ID ) ;
create sequence BOOK_TRANSACTION_TRANSACTION_I start with 1 ;
create or replace trigger BOOK_TRANSACTION_TRANSACTION_I before
  insert on BOOK_TRANSACTION for each row when (new.TRANSACTION_ID is
null)
begin
  :new.TRANSACTION_ID := BOOK_TRANSACTION_TRANSACTION_I.NEXTVAL;
end;
/
insert into BOOK_TRANSACTION values(null, 'City', 'Cus1', 'C1',
to_date('Mar 2008', 'Mon YYYY'), 2);
insert into BOOK_TRANSACTION values(null, 'City', 'Cus2', 'C1',
to_date('Mar 2008', 'Mon YYYY'), 3);
insert into BOOK_TRANSACTION values(null, 'City', 'Cus2', 'H6',
to_date('Mar 2008', 'Mon YYYY'), 10);
insert into BOOK_TRANSACTION values(null, 'City', 'Cus3', 'H6',
to_date('Mar 2008', 'Mon YYYY'), 5);
insert into BOOK_TRANSACTION values(null, 'City', 'Cus3', 'DV',
to_date('Mar 2008', 'Mon YYYY'), 10);
insert into BOOK_TRANSACTION values(null, 'City', 'Cus4', 'DV',
to_date('Mar 2008', 'Mon YYYY'), 13);
```

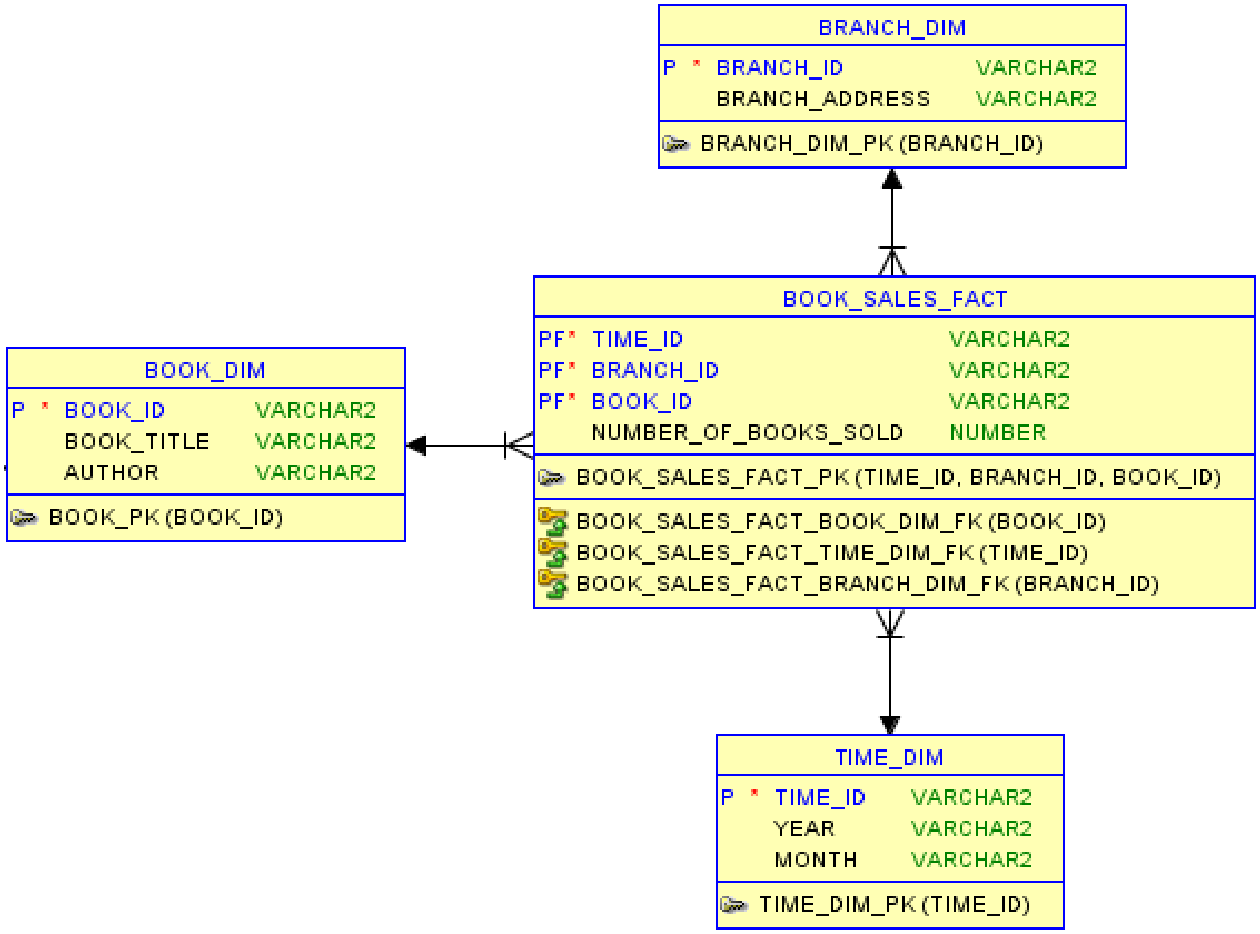
```
insert into BOOK_TRANSACTION values(null, 'Chadstone', 'Cus4', 'C1',
to_date('Mar 2008', 'Mon YYYY'), 10);
insert into BOOK_TRANSACTION values(null, 'Chadstone', 'Cus5', 'C1',
to_date('Mar 2008', 'Mon YYYY'), 5);
insert into BOOK_TRANSACTION values(null, 'Chadstone', 'Cus4', 'H6',
to_date('Mar 2008', 'Mon YYYY'), 3);
insert into BOOK_TRANSACTION values(null, 'Chadstone', 'Cus3', 'DV',
to_date('Mar 2008', 'Mon YYYY'), 2);
insert into BOOK_TRANSACTION values(null, 'Camberwell', 'Cus3', 'C1',
to_date('Mar 2008', 'Mon YYYY'), 1);
insert into BOOK_TRANSACTION values(null, 'Camberwell', 'Cus2', 'H6',
to_date('Mar 2008', 'Mon YYYY'), 1);
insert into BOOK_TRANSACTION values(null, 'Camberwell', 'Cus1', 'DV',
to_date('Mar 2008', 'Mon YYYY'), 2);
insert into BOOK_TRANSACTION values(null, 'City', 'Cus4', 'C1',
to_date('Dec 2007', 'Mon YYYY'), 10);
insert into BOOK_TRANSACTION values(null, 'City', 'Cus3', 'C1',
to_date('Dec 2007', 'Mon YYYY'), 5);
insert into BOOK_TRANSACTION values(null, 'City', 'Cus2', 'H6',
to_date('Dec 2007', 'Mon YYYY'), 5);
insert into BOOK_TRANSACTION values(null, 'City', 'Cus2', 'H6',
to_date('Dec 2007', 'Mon YYYY'), 1);
insert into BOOK_TRANSACTION values(null, 'City', 'Cus5', 'DV',
to_date('Dec 2007', 'Mon YYYY'), 6);
insert into BOOK_TRANSACTION values(null, 'Chadstone', 'Cus4', 'C1',
to_date('Dec 2007', 'Mon YYYY'), 5);
insert into BOOK_TRANSACTION values(null, 'Chadstone', 'Cus3', 'C1',
to_date('Dec 2007', 'Mon YYYY'), 5);
insert into BOOK_TRANSACTION values(null, 'Chadstone', 'Cus2', 'H6',
to_date('Dec 2007', 'Mon YYYY'), 4);
insert into BOOK_TRANSACTION values(null, 'Chadstone', 'Cus1', 'H6',
to_date('Dec 2007', 'Mon YYYY'), 4);
insert into BOOK_TRANSACTION values(null, 'Chadstone', 'Cus4', 'DV',
to_date('Dec 2007', 'Mon YYYY'), 1);
insert into BOOK_TRANSACTION values(null, 'Camberwell', 'Cus1', 'C1',
to_date('Dec 2007', 'Mon YYYY'), 9);
insert into BOOK_TRANSACTION values(null, 'Camberwell', 'Cus3', 'C1',
to_date('Dec 2007', 'Mon YYYY'), 9);
insert into BOOK_TRANSACTION values(null, 'Camberwell', 'Cus2', 'H6',
to_date('Dec 2007', 'Mon YYYY'), 3);
insert into BOOK_TRANSACTION values(null, 'Camberwell', 'Cus1', 'DV',
to_date('Dec 2007', 'Mon YYYY'), 2);
insert into BOOK_TRANSACTION values(null, 'Chadstone', 'Cus4', 'DV',
to_date('Dec 2007', 'Mon YYYY'), 1);
insert into BOOK_TRANSACTION values(null, 'Camberwell', 'Cus1', 'C1',
to_date('Dec 2007', 'Mon YYYY'), 9);
insert into BOOK_TRANSACTION values(null, 'Camberwell', 'Cus3', 'C1',
to_date('Dec 2007', 'Mon YYYY'), 9);
insert into BOOK_TRANSACTION values(null, 'Camberwell', 'Cus2', 'H6',
to_date('Dec 2007', 'Mon YYYY'), 3);
insert into BOOK_TRANSACTION values(null, 'Camberwell', 'Cus1', 'DV',
to_date('Dec 2007', 'Mon YYYY'), 2);
```

```
commit;
```

2. Solution Model 1 – No bridge

The only one measurable fact to be included in the fact table is “Number of Books Sold”, which is calculated by *total quantity* of all transactions. The dimensions are Branch, Time and Book.

The star schema for this case study is as follows:



Your tasks:

a. Create a dimension table called BOOK_DIM.

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

b. Create a dimension table called TIME_DIM. Year and Month are extracted from Transaction Date.

```
-- 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;
```

```
select * from TIME_DIM;
```

c. Create a dimension table called BRANCH_DIM.

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

d. Create fact table (called it BOOK_SALES_FACT1).

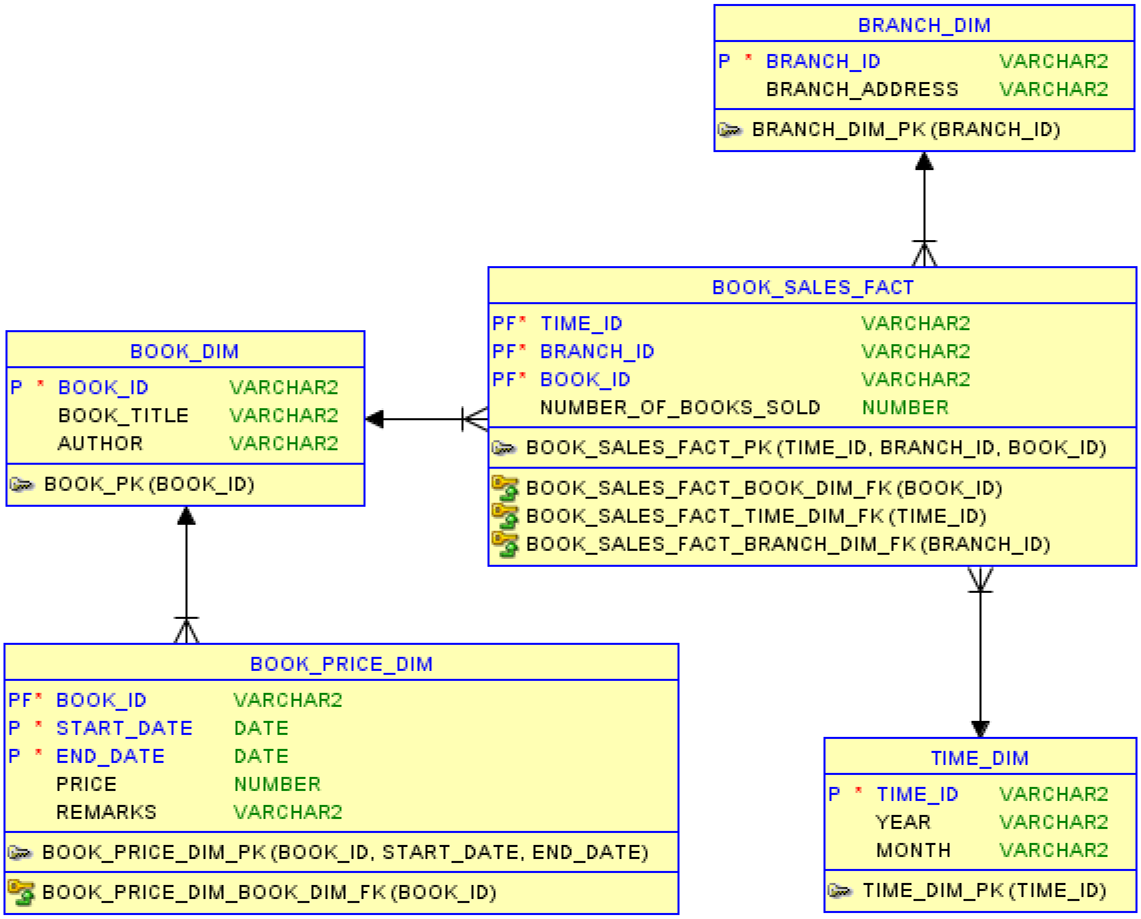
```
-- BOOK_SALES_FACT1
create table BOOK_SALES_FACT1 as
select to_char(T.TRANSACTION_DATE, 'MonYYYY') as TIME_ID,
BK.BOOK_ID, BR.BRANCH_ID,
       sum(T.QUANTITY) as NUMBER_OF_BOOKS_SOLD
from BOOK_TRANSACTION T, BOOK BK, BRANCH BR
where T.BRANCH_ID = BR.BRANCH_ID
and   T.BOOK_ID = BK.BOOK_ID
group by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

e. Display (and observe) the contents of the fact table (BOOK_SALES_FACT1).

```
select * from BOOK_SALES_FACT1;
```

3. Solution Model 2 – add a Temporal Bridge

In order to incorporate the temporal values of book price, we add a temporal/bridge table: BOOK_PRICE_DIM to store the history of book prices. The Star Schema would look like this:



Your tasks:

a. Create a dimension table called BOOK_PRICE_DIM.

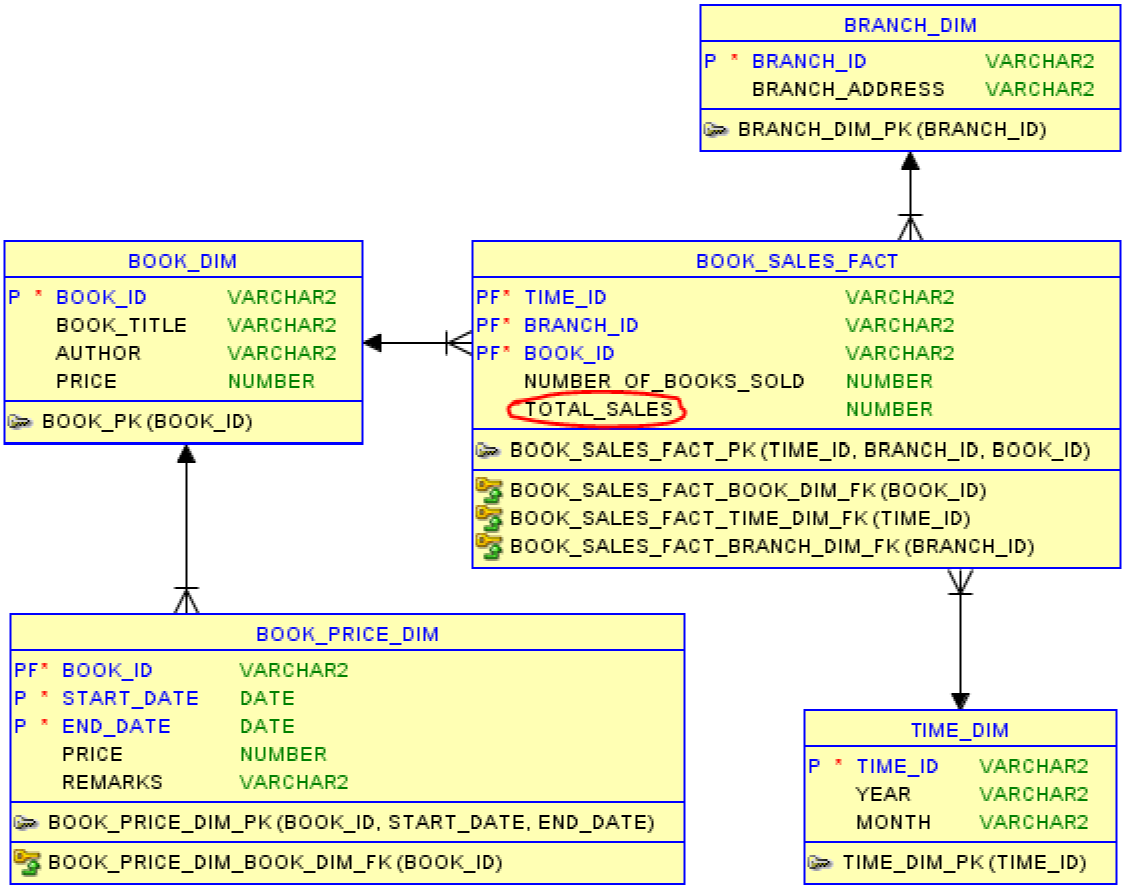
```
-- BOOK_PRICE_DIM (TEMPORAL DIMENSION)
create table BOOK_PRICE_DIM as select distinct * from
BOOK_PRICE_HISTORY;
select * from BOOK_PRICE_DIM;
```

b. **Challenge:** Create the “Correct Book Sale” Report as shown in Lecture 4 Notes page 5/page 10. Hint: Use **Case When** to handle END_DATE = ‘Now’. (Don’t waste time on this, you can come back to this task after finishing Task 3).

```
-- BOOK SALES REPORT 2 (CORRECT PRICE REPORT)
select F.TIME_ID as "Month", F.BRANCH_ID as "Branch", F.BOOK_ID
as "Book ID",
       B.BOOK_TITLE as "Book Title", B.AUTHOR, BP.PRICE,
       F.NUMBER_OF_BOOKS_SOLD as "No of Books Sold"
from BOOK_SALES_FACT1 F, BOOK_PRICE_DIM BP, BOOK_DIM B
where F.BOOK_ID = B.BOOK_ID
and   BP.BOOK_ID = B.BOOK_ID
and   to_date(F.TIME_ID, 'MonYYYY') >= to_date(BP.START_DATE,
'MonYYYY')
and   to_date(F.TIME_ID, 'MonYYYY') <= case BP.END_DATE when
'Now' then SYSDATE
      else to_date(BP.END_DATE, 'MonYYYY')
      end
order by F.TIME_ID desc, F.BRANCH_ID desc, F.NUMBER_OF_BOOKS_SOLD
asc;
```

3. Solution Model 3 – add a new Fact: Total Sales

Although we can produce the “Correct Book Sale” Report with Solution Model 2, the calculation will be complex and performance is not optimal for any reports that involve history book price. Therefore, it’s better to calculate Total Sales using the “correct price” beforehand and store it as a Fact measure. The Star Schema would look like this:



Your tasks:

a. Create a new Fact table: BOOK_SALES_FACT2 by coping BOOK_SALES_FACT1.

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

b. Add Column TOTAL_SALES (NUMBER) to BOOK_SALES_FACT2.

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

c. Use the PRICE_CURSOR (by selecting all data from BOOK_PRICE_DIM) to populate data for column TOTAL_SALES in BOOK_SALES_FACT2. Pay attention to the current book price in BOOK_PRICE_DIM, they will have END_DATE equals to ‘Now’ instead of a normal date (MonYYYY).

```
declare
  cursor PRICE_CURSOR is select * from BOOK_PRICE_DIM;
  VALID_END_DATE DATE;
begin
```

```
    for ITEM in PRICE_CURSOR loop
      -- Change END_DATE = 'Now' to valid date value
      if ITEM.END_DATE = 'Now' then
        VALID_END_DATE := SYSDATE;
      else
        VALID_END_DATE := to_date(ITEM.END_DATE, 'MonYYYY');
      end if;

      -- 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') <= VALID_END_DATE;
  end loop;
end;
/
select * from BOOK_SALES_FACT2;
```

d. **Challenge:** Recreate BOOK_SALES_FACT2 without using **Cursor**.

```
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 <= case BP.END_DATE when 'Now' then
SYSDATE
      else to_date(BP.END_DATE, 'MonYYYY')
      end
group by to_char(T.TRANSACTION_DATE, 'MonYYYY'), BK.BOOK_ID,
BR.BRANCH_ID;
select * from BOOK_SALES_FACT2;
```