1) print the store name, title name,, quantity, sale amount, publisher name, author name for all the sales. Also print those books which have not been sold and authors who have not written.

select

st.stor\_name 'Store Name',

t.title 'Title Name',

s.qty 'Quantity',

s.qty \* t.price 'Sale Amount',

p.pub\_name 'Publisher Name',

CONCAT(au.au\_fname, ' ', au.au\_lname) 'Author Name',

'Sold' 'Status'

from sales s

join stores st on s.stor\_id = st.stor\_id

join titles t on s.title\_id = t.title\_id

join publishers p on t.pub\_id = p.pub\_id

left join titleauthor ta on t.title\_id = ta.title\_id

left join authors au on ta.au\_id = au.au\_id

union all

select

t.title 'Title Name',

p.pub\_name 'Publisher Name',

null 'Quantity',

null 'Sale Amount',

null 'Store Name',

CONCAT(au.au\_fname, ' ', au.au\_lname) 'Author Name',

'Not Sold' 'Status'

from titles t

join publishers p on t.pub\_id = p.pub\_id

left join titleauthor ta on t.title\_id = ta.title\_id

left join authors au on ta.au\_id = au.au\_id

union all

select

null 'Store Name',

null 'Title Name',

null 'Quantity',

null 'Sale Amount',

null 'Publisher Name',

CONCAT(au.au\_fname, ' ', au.au\_lname) 'Author Name',

'Not Written' 'Status'

from authors au

left join titleauthor ta on au.au\_id = ta.au\_id;

2) Create a stored procedure that will take the author name and print the total sales amount for all the books authored by him/her

alter PROCEDURE get\_total\_sales\_by\_author (@author\_name VARCHAR(60))

AS

BEGIN

DECLARE @total\_sales DECIMAL(10,2);

SET @total\_sales = (SELECT sum(s.qty )\* sum(t.price) FROM sales s

INNER JOIN titles t ON s.title\_id = t.title\_id

INNER JOIN titleauthor ta ON t.title\_id = ta.title\_id

INNER JOIN authors a ON ta.au\_id = a.au\_id

WHERE a.au\_fname + ' ' + a.au\_lname = @author\_name)

IF @total\_sales IS NULL OR @total\_sales = 0

BEGIN

PRINT 'Sale yet to gear up';

END

ELSE

BEGIN

PRINT 'The total sales amount for ' + @author\_name + ' is ' + CAST(@total\_sales AS VARCHAR);

END;

END;

EXEC get\_total\_sales\_by\_author @author\_name = 'Ann Ringer';

3) print the details of the sale when the sale quantity is greater than the sale quantity of all the same titles sold in the same store

select

s.stor\_id 'Store ID',

s.title\_id 'Title ID',

t.title 'Title',

s.qty 'Sale Quantity',

s.ord\_date 'Sale Date'

from sales s

left join (select

s.stor\_id,

s.title\_id,

max(s.qty) AS max\_qty

from sales s

group by s.stor\_id, s.title\_id)

max\_sale\_qty on s.stor\_id = max\_sale\_qty.stor\_id

and s.title\_id = max\_sale\_qty.title\_id

and s.qty > max\_sale\_qty.max\_qty

join titles t on s.title\_id = t.title\_id;

4) Print the average price of every author's books withthe author's full name

select

concat(au.au\_fname, ' ', au.au\_lname) 'Author Name',

AVG(t.price) 'Average Price'

from authors au

join titleauthor ta on au.au\_id = ta.au\_id

join titles t on ta.title\_id = t.title\_id

group by au.au\_id, au.au\_fname, au.au\_lname;

5) Print the schema of the titles table and locate all the constrants

sp\_columns 'titles' -- Get the schema of the titles table

sp\_helpconstraint 'titles' -- Locate the constraints for the titles table

6) Create a procedure that will take the price and prints the count of book that are priced less than that

create procedure proc\_CountBooksPricedLessThan

@Price decimal(10, 2)

as

begin

declare @BookCount int

select @BookCount = count(\*)

from titles

where price < @Price

-- Print the count

PRINT 'Count of books priced less than ' + CAST(@Price AS VARCHAR) + ': ' + CAST(@BookCount AS VARCHAR)

END

EXEC proc\_CountBooksPricedLessThan 20.00

7) Find a way to ensure that the price of books are not updated if the price is below 7

CREATE TRIGGER check\_price\_before\_update

ON titles

instead of insert

AS

BEGIN

DECLARE

@title\_id varchar(6),

@title varchar(60),

@type char(12),

@pub\_id char(4),

@price money,

@advance money,

@royalty int,

@ytd\_sales int,

@notes varchar(200),

@pubdate datetime,

@new\_price DECIMAL(10,2);

SET @new\_price = (SELECT price FROM inserted);

IF @new\_price < 7

BEGIN

PRINT 'The price cannot be updated to below 7';

END;

ELSE

BEGIN

insert into titles values(@title\_id,@title,@type,@pub\_id,@price,@advance,@royalty ,@ytd\_sales ,@notes,@pubdate)

END;

END;

insert into titles values('AU1099','Learn From Failures',

'psychology','0599',6.00,15000.00,25,333,

'Here you can face fear of Failures','2023-10-30 00:00:00:000')

8) print the books that have 'e' and 'a' in their name

select title

from titles

where title like '%e%' and title like '%a%';