store name, title name, quantity, sale amount, pulisher name, author name for all the sales

```sql
SELECT
   S.store_name,
   S.title_name,
   S.quantity,
   S.sale_amount,
   S.publisher_name,
   B.author_name
FROM Sales S
JOIN Books B ON S.title_name = B.title_name;
```

List of books that have not been sold

```sql
SELECT
   B.title_name
FROM Books B
LEFT JOIN Sales S ON B.title_name = S.title_name
WHERE S.title_name IS NULL;
```

List of authors who have not written any books

```sql
SELECT
   B.author_name
FROM Books B
LEFT JOIN Sales S ON B.title_name = S.title_name
WHERE S.title_name IS NULL
GROUP BY B.author_name;
```

2)

```
CREATE PROCEDURE CalculateTotalSalesAmount
   @AuthorName NVARCHAR(100)
AS
BEGIN
   DECLARE @TotalSalesAmount DECIMAL(10, 2)
   SELECT @TotalSalesAmount = SUM(S.sale_amount)
   FROM Sales S
   JOIN Books B ON S.title_name = B.title_name
   WHERE B.author_name = @AuthorName;
   IF @TotalSalesAmount IS NULL
   BEGIN
      PRINT 'Sale yet to gear up'
   END
   ELSE
   BEGIN
      PRINT 'Total Sales Amount for ' + @AuthorName + ': ' + CONVERT(NVARCHAR(30),
@TotalSalesAmount)
   END
END;
```

3)

```
WITH SaleAggregates AS (
    SELECT
        S.store_name,
        S.title_name,
        MAX(S.quantity) AS max_quantity
    FROM Sales S
    GROUP BY S.store_name, S.title_name
)

SELECT S.*
FROM Sales S
INNER JOIN SaleAggregates SA ON S.store_name = SA.store_name AND S.title_name = SA.title_name
WHERE S.quantity > SA.max_quantity;
```

4)

```
SELECT author_name, AVG(price) AS average_price
FROM Books
GROUP BY author_name;
```

5)

```
EXEC sp_help 'titles';
```

6)

```
CREATE PROCEDURE GetBookCountPricedLessThan
    @Price DECIMAL(10, 2)
AS
BEGIN
    DECLARE @BookCount INT
```

```
 SELECT @BookCount = COUNT(*)

  FROM Books

  WHERE price < @Price;


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

END;
```

7)

```
-- Create the trigger

CREATE TRIGGER PreventPriceUpdateBelow7

ON Books

INSTEAD OF UPDATE

AS

BEGIN

  -- Check if the price is being updated to a value less than 7

  IF EXISTS (

    SELECT 1

    FROM INSERTED

    WHERE price < 7

  )

  BEGIN

    RAISEERROR('Price cannot be updated to a value below 7.', 16, 1);

    ROLLBACK TRANSACTION;

  END

  ELSE

  BEGIN

    -- Allow the update if the price is not below 7

    UPDATE B

    SET B.price = I.price
    FROM Books AS B
      INNER JOIN INSERTED AS I ON B.book_id = I.book_id;
  END
END;
```

8)

```
SELECT title_name
FROM Books
WHERE CHARINDEX('e', title_name) > 0
  AND CHARINDEX('a', title_name) > 0;
```