Project

Daxeshkumar Patel

041086185

Algonquin College of Applied Arts and Technology

24W_CST2102_300 Database Analytics

Professor Jim Myronyk

April 3, 2024

TABLE OF CONTENT

PART 1: FOUR SQL QUERIES USING THE FEHILY DATABASE FROM CLASS

1. List the top 2 revenue generating authors (i.e., author id, author concatenated name, title_id, book revenue, total author revenue) for each publisher.

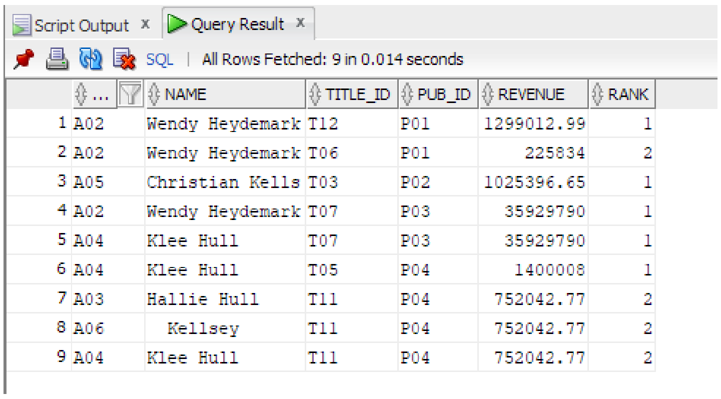
<u>EXPLANATION:-</u>

Main requirement our here is to show the top 2 revenue generating authors in the table having columns for author id, author concatenated name, title_id, book revenue, total author revenue. For doing so I have calculated the revenue **(t.price * t.sales) and alias it as a Revenue.** Then, to short revenue in **descending order and use Rank to give ranking** to each raw and use join to connect the tables. After that as required top 2 revenue generators for every publisher, I have partitioned the tanking by the pub_id. To show the result in requested format use **SELECT SELECT au_id, name, title_id, pub_id, Revenue, Rank and** use **WHERE** clause on **rank < =2 to filter the results further.**

<u>QUERY: -</u>

```
SELECT au_id, name, title_id, pub_id, Revenue, Rank

FROM

(

SELECT

a.au_id, a.fname || ' ' || a.lname AS name, at.title_id, p.pub_id,SUM(t.price * t.sales) AS Revenue,

RANK() OVER (PARTITION BY p.pub_id ORDER BY SUM(t.price * t.sales) DESC) AS Rank

FROM authors a

JOIN author_titles at ON a.au_id = at.au_id

JOIN titles t ON at.title_id = t.title_id

JOIN publishers p ON t.pub_id = p.pub_id

GROUP BY a.au_id, a.fname, a.lname, at.title_id, p.pub_id

)

WHERE Rank <= 2;
```

<u>OUTPUT: -</u>



Script Output | Query Result

SQL | All Rows Fetched: 9 in 0.014 seconds

	...	NAME	TITLE_ID	PUB_ID	REVENUE	RANK
1	A02	Wendy Heydemark	T12	P01	1299012.99	1
2	A02	Wendy Heydemark	T06	P01	225834	2
3	A05	Christian Kells	T03	P02	1025396.65	1
4	A02	Wendy Heydemark	T07	P03	35929790	1
5	A04	Klee Hull	T07	P03	35929790	1
6	A04	Klee Hull	T05	P04	1400008	1
7	A03	Hallie Hull	T11	P04	752042.77	2
8	A06	Kellsey	T11	P04	752042.77	2
9	A04	Klee Hull	T11	P04	752042.77	2

2.Use a subquery to count the number of books sold in the month closest to Christmas by each genre.

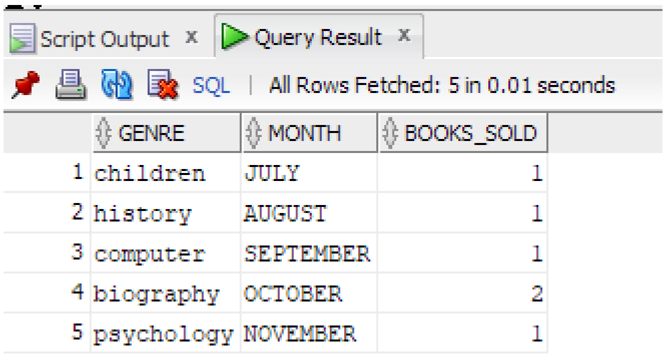
EXPLANATION:-

To achieve desired result It is mandatory to ger the details of books sold and filter the month from the date format. Hence, extracted the Month from the Date format of TITLES table and consider the total count as books sold per month by using **COUNT(*).** Then convert the months in to ascending order by using **EXTRACT(MONTH FROM MIN(PUBDATE)) ASC** for getting better understanding.

QUERY:-

```
SELECT
  Genre, TO_CHAR(PUBDATE,'MONTH') AS Month,
  COUNT(*) as Books_Sold
FROM
  TITLES t
WHERE
  EXTRACT (MONTH FROM PUBDATE) = (
    SELECT MAX(EXTRACT(MONTH FROM PUBDATE))
    FROM TITLES
    WHERE Genre = t.genre )
GROUP BY
  Genre, TO_CHAR(PUBDATE,'MONTH')
ORDER BY
  EXTRACT(MONTH FROM MIN(PUBDATE)) ASC;
```

OUTPUT: -



All Rows Fetched: 5 in 0.01 seconds

	GENRE	MONTH	BOOKS_SOLD
1	children	JULY	1
2	history	AUGUST	1
3	computer	SEPTEMBER	1
4	biography	OCTOBER	2
5	psychology	NOVEMBER	1

3. Use the SUM(), RANK(), and LAG() analytic windowing functions with partitions to develop a meaningful query

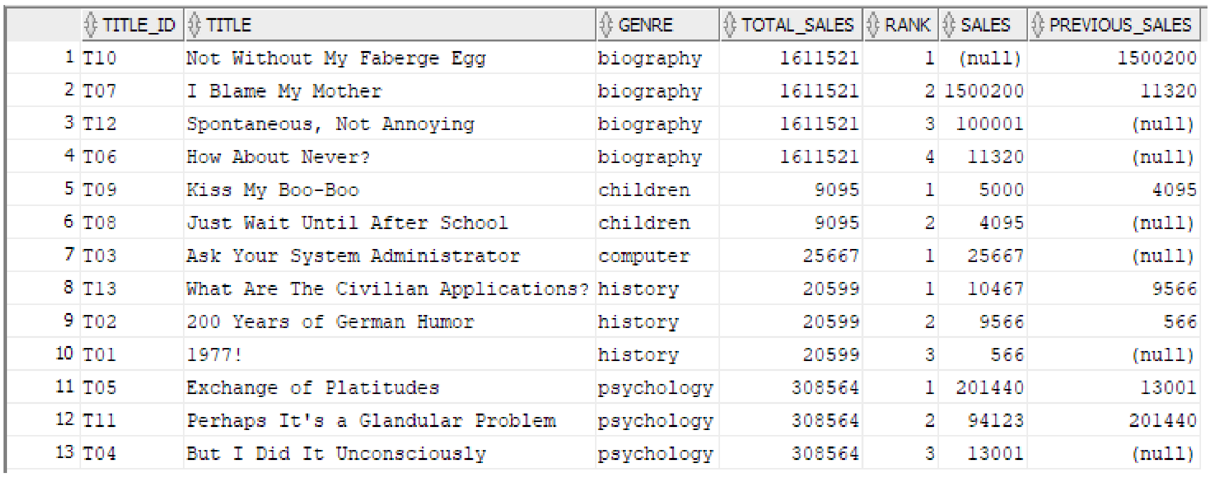
EXPLANATION:-

- The **SUM(Sales)** OVER (PARTITION BY Genre) calculates the total sales within each genre and sums up the sales values within that partitions itself.
- The **RANK()** OVER (PARTITION BY Genre ORDER BY SUM(Sales) DESC) give rank to each title within its genre based on the total sales. The higher the total sales, lover the rank would be.
- The **LAG(Sales)** OVER (PARTITION BY Genre ORDER BY Title_id) function retrieves the sales value of the previous title within the same genre.
- Belo query provides details of the total sales for each genre, rank titles based on their sales performance within each genre and compare the sales of each title with the previous titles within the same genre.

QUERY: -

```
SELECT

Title_id, Title, Genre, SUM(Sales) OVER (PARTITION BY Genre) AS Total_Sales,

RANK() OVER (PARTITION BY Genre ORDER BY SUM(Sales) DESC) AS Rank, Sales,

LAG(Sales) OVER (PARTITION BY Genre ORDER BY Title_id) AS Previous_Sales

FROM Titles

GROUP BY Title_id, Title, Genre, Sales;
```

OUTPUT: -



	TITLE_ID	TITLE	GENRE	TOTAL_SALES	RANK	SALES	PREVIOUS_SALES
1	T10	Not Without My Faberge Egg	biography	1611521	1	(null)	1500200
2	T07	I Blame My Mother	biography	1611521	2	1500200	11320
3	T12	Spontaneous, Not Annoying	biography	1611521	3	100001	(null)
4	T06	How About Never?	biography	1611521	4	11320	(null)
5	T09	Kiss My Boo-Boo	children	9095	1	5000	4095
6	T08	Just Wait Until After School	children	9095	2	4095	(null)
7	T03	Ask Your System Administrator	computer	25667	1	25667	(null)
8	T13	What Are The Civilian Applications?	history	20599	1	10467	9566
9	T02	200 Years of German Humor	history	20599	2	9566	566
10	T01	1977!	history	20599	3	566	(null)
11	T05	Exchange of Platitudes	psychology	308564	1	201440	13001
12	T11	Perhaps It's a Glandular Problem	psychology	308564	2	94123	201440
13	T04	But I Did It Unconsciously	psychology	308564	3	13001	(null)

4. Improve the performance of 1 of the above queries. Explain your approach and support the results with before and after explain plan results.

Explanation: -

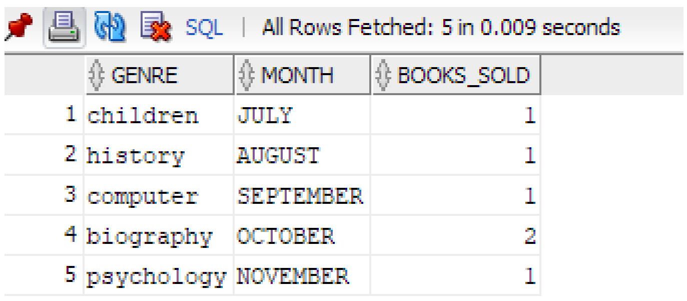
1.Subquery for Maximum genre per month: In place of recalculating the maximum month for each genre in the WHERE clause, calculate it once for each genre by using a subquery (genre_mx_month).

2.Join with Main Table: Join the main TITLES table with the result of the subquery based on the genre. This has ensured that we only perform the max month calculation one time only.

QUERY: -

```
SELECT t.Genre, TO_CHAR(t.PUBDATE, 'MONTH') AS Month, COUNT(*) AS Books_Sold

FROM TITLES t

JOIN (

  SELECT Genre, MAX(EXTRACT(MONTH FROM PUBDATE)) AS Mxmonth

  FROM TITLES

  GROUP BY Genre )

genre_mx_month ON t.Genre = genre_mx_month.genre

WHERE EXTRACT(MONTH FROM t.PUBDATE) = genre_mx_month.Mxmonth

GROUP BY t.Genre, TO_CHAR(t.PUBDATE, 'MONTH')

ORDER BY EXTRACT(MONTH FROM MIN(PUBDATE)) ASC;
```

OUT PUT: -

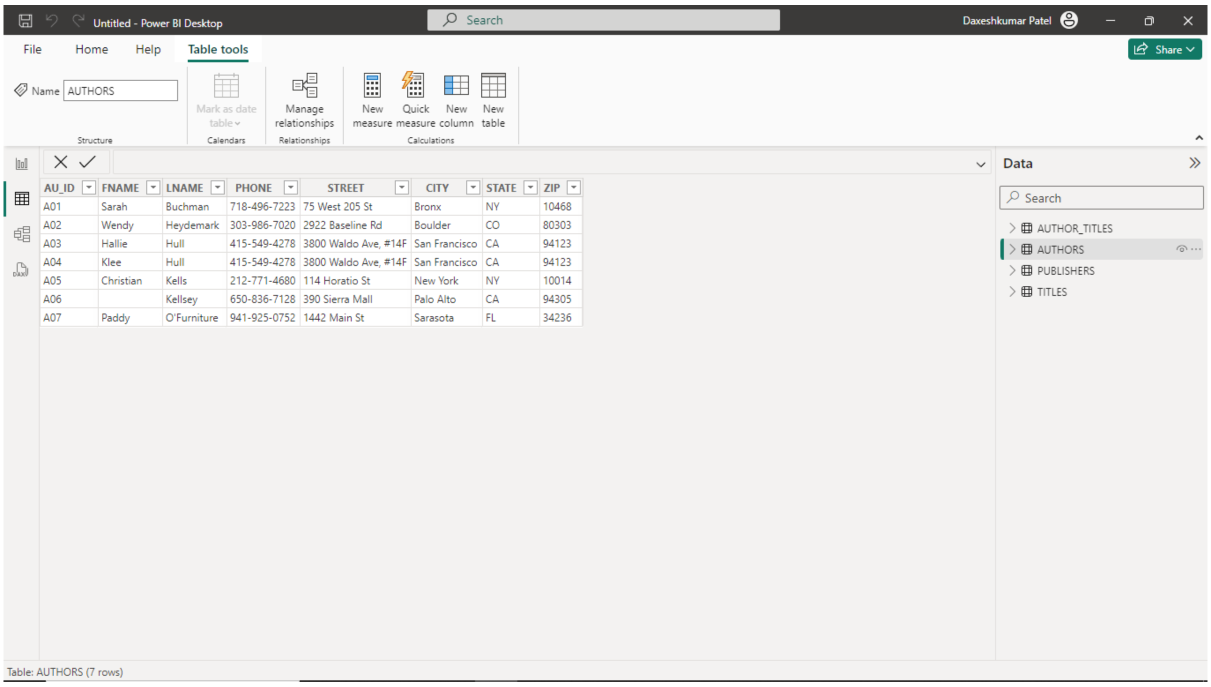


SQL | All Rows Fetched: 5 in 0.009 seconds

	GENRE	MONTH	BOOKS_SOLD
1	children	JULY	1
2	history	AUGUST	1
3	computer	SEPTEMBER	1
4	biography	OCTOBER	2
5	psychology	NOVEMBER	1

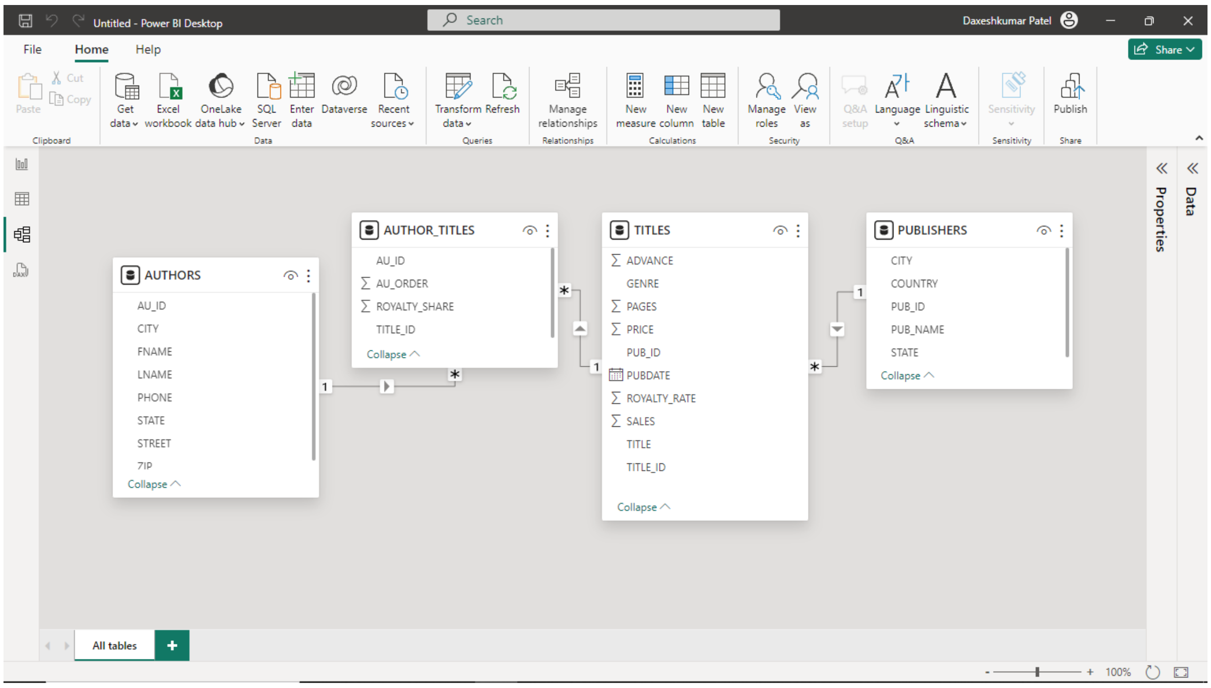
PART 2: EXPORT ORACLE DATA FROM ORACLE TO MS POWER BI AND MYSQL

2.1. Export the Fehily data from your user 'dax' schema and load it into MS Power BI.



AU_ID	FNAME	LNAME	PHONE	STREET	CITY	STATE	ZIP
A01	Sarah	Buchman	718-496-7223	75 West 205 St	Bronx	NY	10468
A02	Wendy	Heydemark	303-986-7020	2922 Baseline Rd	Boulder	CO	80303
A03	Hallie	Hull	415-549-4278	3800 Waldo Ave, #14F	San Francisco	CA	94123
A04	Klee	Hull	415-549-4278	3800 Waldo Ave, #14F	San Francisco	CA	94123
A05	Christian	Kells	212-771-4680	114 Horatio St	New York	NY	10014
A06		Kellsey	650-836-7128	390 Sierra Mall	Palo Alto	CA	94305
A07	Paddy	O'Furniture	941-925-0752	1442 Main St	Sarasota	FL	34236

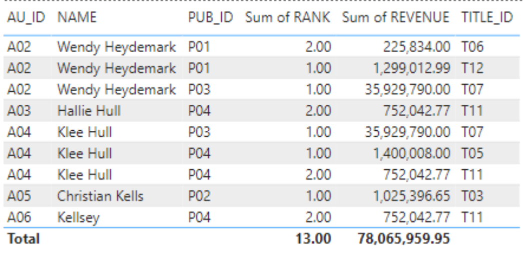
Table: AUTHORS (7 rows)



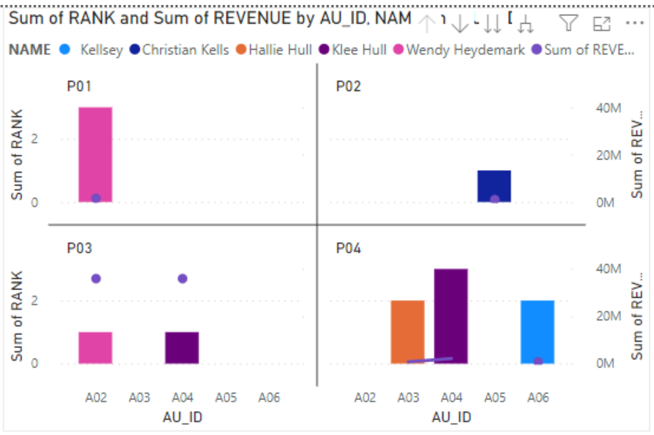
2.2. Reproduce the query results from Part1 in PBI

2.2.1. List the top 2 revenue generating authors (i.e., author id, author concatenated name, title_id, book revenue, total author revenue) for each publisher.

OUTPUT: -

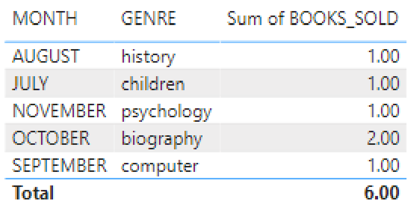


AU_ID	NAME	PUB_ID	Sum of RANK	Sum of REVENUE	TITLE_ID
A02	Wendy Heydemark	P01	2.00	225,834.00	T06
A02	Wendy Heydemark	P01	1.00	1,299,012.99	T12
A02	Wendy Heydemark	P03	1.00	35,929,790.00	T07
A03	Hallie Hull	P04	2.00	752,042.77	T11
A04	Klee Hull	P03	1.00	35,929,790.00	T07
A04	Klee Hull	P04	1.00	1,400,008.00	T05
A04	Klee Hull	P04	2.00	752,042.77	T11
A05	Christian Kells	P02	1.00	1,025,396.65	T03
A06	Kellsey	P04	2.00	752,042.77	T11
Total			**13.00**	**78,065,959.95**	

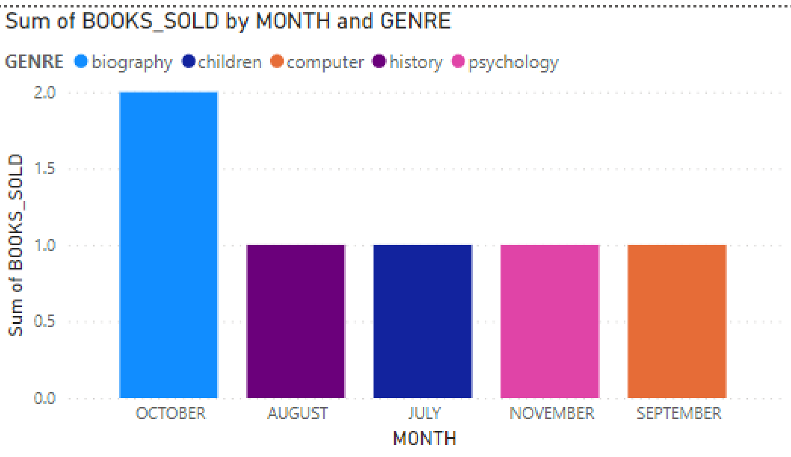


2.2.2. Use a subquery to count the number of books sold in the month closest to Christmas by each genre.

OUTPUT: -

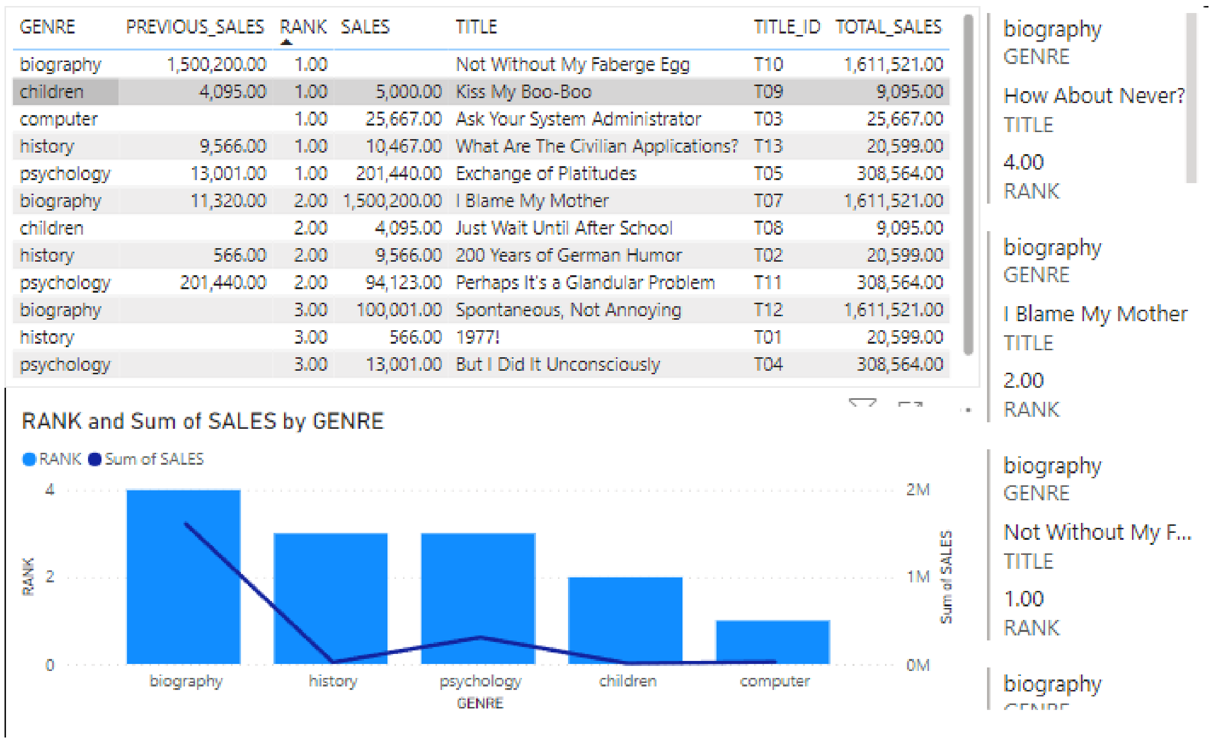


MONTH	GENRE	Sum of BOOKS_SOLD
AUGUST	history	1.00
JULY	children	1.00
NOVEMBER	psychology	1.00
OCTOBER	biography	2.00
SEPTEMBER	computer	1.00
Total		**6.00**



2.2.3. Use the SUM(), RANK(), and LAG() analytic windowing functions with partitions to develop a meaningful query

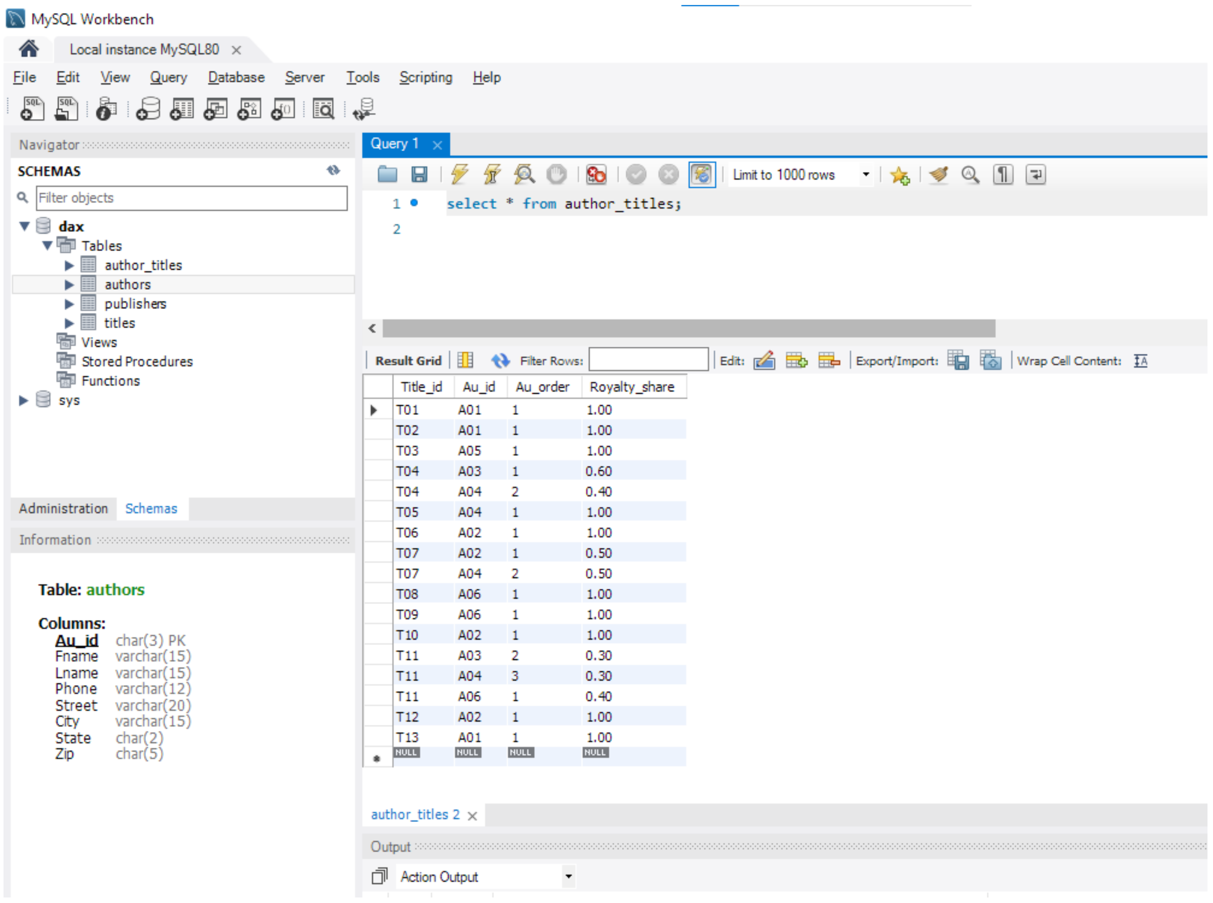
<u>OUTPUT: -</u>



GENRE	PREVIOUS_SALES	RANK	SALES	TITLE	TITLE_ID	TOTAL_SALES
biography	1,500,200.00	1.00		Not Without My Faberge Egg	T10	1,611,521.00
children	4,095.00	1.00	5,000.00	Kiss My Boo-Boo	T09	9,095.00
computer		1.00	25,667.00	Ask Your System Administrator	T03	25,667.00
history	9,566.00	1.00	10,467.00	What Are The Civilian Applications?	T13	20,599.00
psychology	13,001.00	1.00	201,440.00	Exchange of Platitudes	T05	308,564.00
biography	11,320.00	2.00	1,500,200.00	I Blame My Mother	T07	1,611,521.00
children		2.00	4,095.00	Just Wait Until After School	T08	9,095.00
history	566.00	2.00	9,566.00	200 Years of German Humor	T02	20,599.00
psychology	201,440.00	2.00	94,123.00	Perhaps It's a Glandular Problem	T11	308,564.00
biography		3.00	100,001.00	Spontaneous, Not Annoying	T12	1,611,521.00
history		3.00	566.00	1977!	T01	20,599.00
psychology		3.00	13,001.00	But I Did It Unconsciously	T04	308,564.00

biography	GENRE
How About Never?	TITLE
4.00	RANK
biography	GENRE
I Blame My Mother	TITLE
2.00	RANK
biography	GENRE
Not Without My F...	TITLE
1.00	RANK
biography	GENRE

2.3. Export the Fehily data from your user ‘dax’ schema and load it into MySQL.



```
select * from author_titles;
```

Table: authors

Columns:

Au_id	char(3) PK
Fname	varchar(15)
Lname	varchar(15)
Phone	varchar(12)
Street	varchar(20)
City	varchar(15)
State	char(2)
Zip	char(5)

Title_id	Au_id	Au_order	Royalty_share
T01	A01	1	1.00
T02	A01	1	1.00
T03	A05	1	1.00
T04	A03	1	0.60
T04	A04	2	0.40
T05	A04	1	1.00
T06	A02	1	1.00
T07	A02	1	0.50
T07	A04	2	0.50
T08	A06	1	1.00
T09	A06	1	1.00
T10	A02	1	1.00
T11	A03	2	0.30
T11	A04	3	0.30
T11	A06	1	0.40
T12	A02	1	1.00
T13	A01	1	1.00
NULL	NULL	NULL	NULL

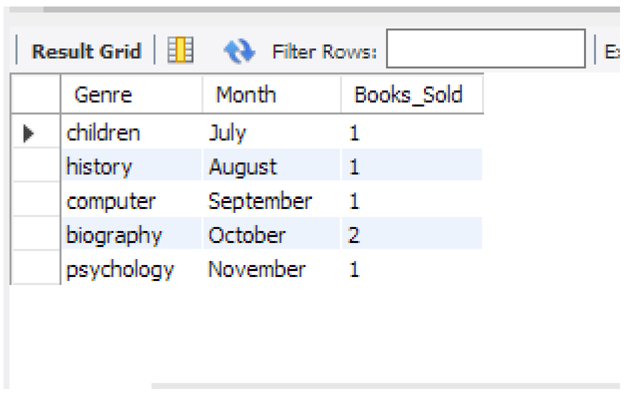
2.4.Reproduce 2 query results from Part1 in MySQL.

2.4.1.Use a subquery to count the number of books sold in the month closest to Christmas by each genre.

QUERY:-

```
SELECT
  Genre, MONTHNAME(PUBDATE) AS Month,
  COUNT(*) as Books_Sold
FROM
  TITLES t
WHERE
  MONTH(PUBDATE) =
(
    SELECT MAX(MONTH(PUBDATE))
    FROM TITLES
    WHERE Genre = t.genre
  )
GROUP BY
  Genre, MONTHNAME(PUBDATE)
ORDER BY
  MONTH(MIN(PUBDATE)) ASC;
```

OUTPUT: -



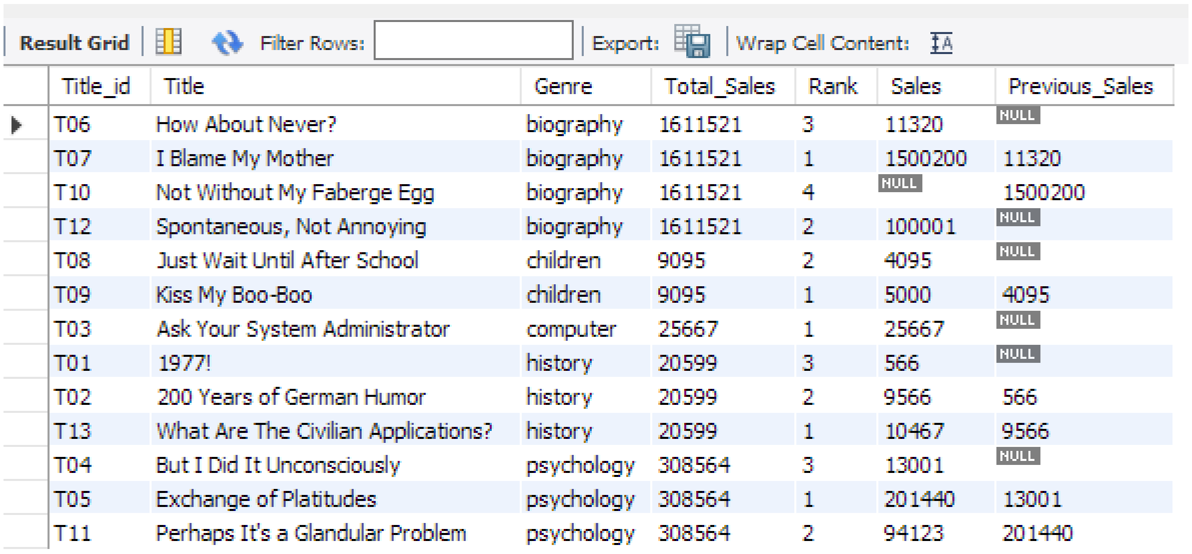
Genre	Month	Books_Sold
children	July	1
history	August	1
computer	September	1
biography	October	2
psychology	November	1

2.4.2. Use the SUM(), RANK(), and LAG() analytic windowing functions with partitions to develop a meaningful query

QUERY: -

```
SELECT

 Title_id, Title, Genre,

SUM(Sales) OVER (PARTITION BY Genre) AS Total_Sales,

RANK() OVER (PARTITION BY Genre ORDER BY Sales DESC) AS ` Rank` ,

Sales,

LAG(Sales) OVER (PARTITION BY Genre ORDER BY Title_id) AS Previous_Sales

FROM Titles;
```

OUTPUT: -



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Title_id	Title	Genre	Total_Sales	Rank	Sales	Previous_Sales
T06	How About Never?	biography	1611521	3	11320	NULL
T07	I Blame My Mother	biography	1611521	1	1500200	11320
T10	Not Without My Faberge Egg	biography	1611521	4	NULL	1500200
T12	Spontaneous, Not Annoying	biography	1611521	2	100001	NULL
T08	Just Wait Until After School	children	9095	2	4095	NULL
T09	Kiss My Boo-Boo	children	9095	1	5000	4095
T03	Ask Your System Administrator	computer	25667	1	25667	NULL
T01	1977!	history	20599	3	566	NULL
T02	200 Years of German Humor	history	20599	2	9566	566
T13	What Are The Civilian Applications?	history	20599	1	10467	9566
T04	But I Did It Unconsciously	psychology	308564	3	13001	NULL
T05	Exchange of Platitudes	psychology	308564	1	201440	13001
T11	Perhaps It's a Glandular Problem	psychology	308564	2	94123	201440