Comp 1630

Relational Database Design & UML

EXERCISE 8 - SQL

1. List the title from the TITLES table, the order number and order date from the SALES table, and the store name from the STORES table. Display only the first **30** characters of the title. Display the order date in the format of **MMM DD YYYY**. There should be a row produced in the result set for each row in the titles table. Order the result set by the order number. The query should produce the result set listed below. (Hint: use **LEFT OUTER JOIN** statement)

Title	OrderNumber	OrderDate	StoreName
Net Etiquette	NULL	NULL	NULL
The Psychology of Computer Coo	NULL	NULL	NULL
The Gourmet Microwave	423LL922	Sep 14 1994	Bookbeat
The Busy Executive's Database	423LL930	Sep 14 1994	Bookbeat
The Busy Executive's Database	6871	Sep 14 1994	Eric the Read Books
.....			
Straight Talk About Computers	QQ2299	Oct 28 1993	Fricative Bookshop
Silicon Valley Gastronomic Tre	TQ456	Dec 12 1993	Fricative Bookshop
You Can Combat Computer Stress	X999	Feb 21 1993	Fricative Bookshop

(23 row(s) affected)

```
SELECT SUBSTRING(t.title,1,30)          AS Title,
       s.ord_num                        AS OrderNumber,
       CONVERT(CHAR(12),s.ord_date,109) AS OrderDate,
       st.stor_name                     AS StoreName
FROM                titles t
LEFT OUTER JOIN     sales s      ON t.title_id = s.title_id
LEFT OUTER JOIN     stores st    ON s.stor_id = st.stor_id
ORDER BY s.ord_num
```

2. Create a new table called ‘business_books’ containing the title ID, title, price, publisher ID, and publish date columns, as well as the data, from the TITLES table for those rows which are of type ‘business’.

(4 row(s) affected)

```
SELECT title_id,
       title,
       price,
       pub_id,
       pubdate
INTO   business_books
FROM   titles
WHERE  type = 'business'
```

3. List the publisher name and the total of books by each title type. Display the publisher name from the PUBLISHERS table, the title type and MIN price from the TITLES table, and the SUM of the quantity from the SALES table. (Hint: Use a **GROUP BY** statement)

PublisherName	Type	MinPrice	Qty
Algodata Infosystems	business	11.95	55
Algodata Infosystems	popular_comp	20.00	80
Binnet & Hardley	mod_cook	2.99	50
Binnet & Hardley	psychology	21.59	20
Binnet & Hardley	trad_cook	11.95	80
New Moon Books	business	2.99	35
New Moon Books	psychology	7.00	173

(7 row(s) affected)

```
SELECT p.pub_name    AS PublisherName,
       t.type        AS Type,
       MIN(t.price)  AS MinPrice,
       SUM(s.qty)    AS Qty
FROM          titles t
INNER JOIN    sales s      ON t.title_id = s.title_id
INNER JOIN    publishers p ON t.pub_id = p.pub_id
GROUP BY t.type, p.pub_name
```

4. Using the **UNION** command, calculate new prices for the books based on the year-to-date sales for each book in the TITLES table. If the year-to-date sales are less than $2500, add 15% to the price; if the year-to-date sales are greater than or equal to $2500 and less than or equal to $10000, add 10% to the price of the book; if the year-to-date sales are greater than $10000, add 5% to the price. Display the title id, year-to-date sales, price, and the new calculated price from the TITLES table. Order the result set by title id. The query should produce the result set listed below.

TitleID	YTDSales	Price	NewPrice
BU1032	4095	19.99	21.99
BU1111	3876	11.95	13.16
BU2075	18722	2.99	3.14
BU7832	4095	19.99	21.99
.....			
TC3218	375	20.95	24.09
TC4203	15096	11.95	12.55
TC7777	4095	14.99	16.49

(16 row(s) affected)

```
SELECT title_id                                  AS TitleID,
       ytd_sales                                 AS YTDSales,
       price                                     AS Price,
       CONVERT(DECIMAL(5,2),(price * 1.15))      AS NewPrice
FROM   titles
WHERE  ytd_sales < 2500
UNION
SELECT title_id,
       ytd_sales,
       price,
       CONVERT(DECIMAL(5,2),(price * 1.10))
FROM   titles
WHERE  ytd_sales BETWEEN 2500 AND 10000
```

```
UNION
SELECT title_id,
       ytd_sales,
       price,
       CONVERT(DECIMAL(5,2),(price * 1.05))
FROM   titles
WHERE  ytd_sales > 10000
ORDER BY title_id
```

5. List the AVG and SUM of the price by type for rows with a price that is NOT NULL from the TITLES table. At the end of the report, show the AVG and SUM of the price for all types. The query should produce the result set listed below. (Hint: Use the **GROUP BY WITH ROLLUP** statement)

Type	Average	Sum
business	13.73	54.92
mod_cook	11.49	22.98
popular_comp	21.475	42.95
psychology	13.504	67.52
trad_cook	15.9633	47.89
NULL	**14.7662**	**236.26**

(6 row(s) affected)

```
SELECT type          AS Type,
       AVG(price)    AS Average,
       SUM(price)    AS Sum
FROM   titles
WHERE  price IS NOT NULL
GROUP BY type WITH ROLLUP
```

6. For each unique store ID, list the store ID, store name, and SUM of the cost calculated as (quantity * price), but only for those stores with a cost between $500 and $1500. Obtain the store ID and name from the STORES table, the quantity from the SALES table, and the price from the TITLES table. Order the result set by store ID. The query should produce the result set listed below.

StoreID	StoreName	Cost
7067	News & Brews	1486.30
7131	Doc-U-Mat: Quality Laundry and Books	1400.15
7896	Fricative Bookshop	604.40
8042	Bookbeat	1232.00

(4 row(s) affected)

```
SELECT st.stor_id                AS StoreID,
       st.stor_name              AS StoreName,
       SUM (s.qty * t.price)     AS Cost
FROM          stores st
INNER JOIN    sales s      ON st.stor_id = s.stor_id
INNER JOIN    titles t     ON s.title_id = t.title_id
GROUP BY      st.stor_id,
              st.stor_name
HAVING SUM (s.qty * t.price) BETWEEN 500.00 AND 1500.00
ORDER BY st.stor_id
```

7. For each store ID, list the SUM of the quantity from the SALES table and the MIN price from the TITLES table. Generate a final total of the qty SUM and MIN price. The query should produce the result set listed below.

StoreID	Qty	Min
6380	8	10.95
7066	125	10.95
7067	90	10.95
7131	130	2.99
7896	60	2.99
8042	80	2.99
NULL	**493**	**2.99**

(7 row(s) affected)

```
SELECT s.stor_id    AS StoreID,
       SUM(s.qty)   AS Qty,
       MIN(t.price) AS Min
FROM         sales s
INNER JOIN   titles t  ON s.title_id = t.title_id
GROUP BY s.stor_id WITH ROLLUP
```

8. Using the INSERT INTO command, insert a new title into the TITLES table with a title ID of 'ZZ1234', a title of 'Microsoft SQL Server', a book type of 'computer', a publisher ID of '0877', a price of $89.99, and a publish date of 'September 29, 2008'. Check your results.

```
INSERT INTO titles
(      title_id,
       title,
       type,
       pub_id,
       price,
       pubdate      )
VALUES
(      'ZZ1234',
       'Microsoft SQL Server',
       'computer',
       '0877',
       89.99,
       'Sep 29 2008' )
```

9. Using the UPDATE command, increase the price by **10%** for the title created in question 8. Check your results.

```
UPDATE titles
SET price = (price * 1.10)
WHERE title_id = 'ZZ1234'
```

10. Delete the title created in question 8. Check your results.

```
DELETE FROM titles
WHERE title_id = 'ZZ1234'
```