ISOM 671: Managing Big Data (Assignment 2)

Name: Foster Mosden

Email: dmosden@emory.edu

Because the Query site we were instructed to use does not use the same flavor of (My)SQL we use in class, I had to write these first five queries a bit differently.

1.

```
SELECT
      COUNT(*) AS num_posts,
      MIN(CreationDate) AS min_date,
      MAX(Creation_Date) AS max_date,
      AVG(Score) AS avg_score
FROM
      posts;
```

num_posts	min_date	max_date	avg_score
21011	2013-10-08 21:29:51	2023-09-02 14:36:24	4

2.

```
 SELECT
    DATEPART(YEAR, CreationDate) AS Year,
    DATEPART(MONTH, CreationDate) AS Month,
    COUNT(*) AS Count
FROM
    Posts
GROUP BY
    DATEPART(YEAR, CreationDate),
      DATEPART(MONTH, CreationDate)
ORDER BY
    Year ASC,
      Month ASC;
```

Year	Month	Count
2013	10	1006
2013	11	408
2013	12	292
2014	1	245
2014	2	302
2014	3	262
2014	4	309
2014	5	237
2014	6	266
2014	7	305
2014	8	236
2014	9	158
2014	10	222
2014	11	160

3.

```
SELECT
    DATEPART(YEAR, CreationDate) as Year,
    DATEPART(MONTH, CreationDate) AS Month,
      PostTypeID,
      COUNT(*) AS cnt
FROM
    Posts
GROUP BY
    DATEPART(YEAR, CreationDate) as Year,
    DATEPART(MONTH, CreationDate) AS Month,
      PostTypeID
ORDER BY
    cnt DESC,
      Year ASC,
      Month ASC,
      PostTypeID ASC;
```

Year	Month	PostTypeID	cnt
2013	10	2	510
2013	10	1	350
2014	4	2	212
2018	12	2	206
2014	7	2	192
2013	12	2	180
2013	11	2	174
2014	2	2	171
2014	6	2	161
2018	1	2	158
2017	6	2	157
2014	1	2	154
2016	8	2	154
2021	1	2	154
2014	3	2	153
2015	8	2	153

October 2013 has the most original posts, at 350

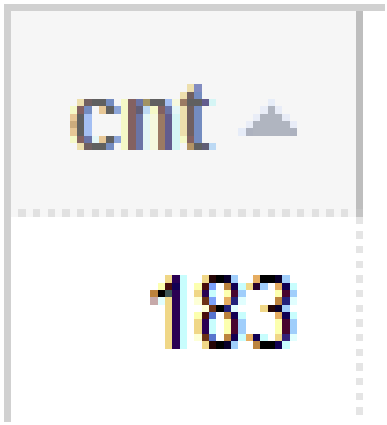
4.

```
SELECT
      name, COUNT(*) AS cnt
FROM
      Badges
GROUP BY Name
HAVING COUNT(*) >= 20
ORDER BY cnt DESC;
```

name	cnt
Autobiographer	7141
Student	4611
Supporter	3833
Popular Question	2827
Teacher	2261
Yearling	1959
Editor	1863
Notable Question	1840
Informed	1631
Scholar	1471
Nice Answer	968
Nice Question	766
Famous Question	742
Custodian	689
Revival	506
Citizen Patrol	452

5.

```
SELECT
      COUNT(DISTINCT ID) AS cnt
FROM
      Users
WHERE
         Location LIKE '%New York%'
      OR Location LIKE '%NY%'
      OR Location LIKE '%NYC%'
      OR Location LIKE '%New York City%'
      OR Location LIKE '%State of New York%'
      OR Location LIKE '%Empire State%'
      OR Location LIKE '%Big Apple%'
      OR Location LIKE '%NY State%'
      OR Location LIKE '%New Yorker%';
```



6.

```
SELECT
      c.category_name, p.product_name, p.list_price
FROM
      Categories c
      INNER JOIN
      Products p ON c.category_id = p.category_id
ORDER BY c.category_name ASC , p.product_name ASC;
```

category_name	product_name	list_price
Basses	Fender Precision	799.99
Basses	Hofner Icon	499.99
Drums	Ludwig 5-piece Drum Set with Cymbals	699.99
Drums	Tama 5-Piece Drum Set with Cymbals	799.99
Guitars	Fender Stratocaster	699.00
Guitars	Gibson Les Paul	1199.00
Guitars	Gibson SG	2517.00

7.

```
SELECT
	c.first_name,
	c.last_name,
	a.line1,
	a.city,
	a.state,
	a.zip_code
FROM
	Customers AS c
	JOIN
	Addresses AS a ON c.customer_id = a.customer_id
WHERE
	c.email_address = 'allan.sherwood@yahoo.com';
```

first_name	last_name	line1	city	state	zip_code
Allan	Sherwood	100 East Ridgewood Ave.	Paramus	NJ	07652
Allan	Sherwood	21 Rosewood Rd.	Woodcliff Lake	NJ	07677

8.

```
SELECT
	c.first_name,
	c.last_name,
	a.line1,
	a.city,
	a.state,
	a.zip_code
FROM
	Customers AS c
	JOIN
	Addresses AS a ON c.customer_id = a.customer_id
WHERE
	c.shipping_address_id = a.address_id;
```

first_name	last_name	line1	city	state	zip_code
Allan	Sherwood	100 East Ridgewood Ave.	Paramus	NJ	07652
Barry	Zimmer	16285 Wendell St.	Omaha	NE	68135
Christine	Brown	19270 NW Cornell Rd.	Beaverton	OR	97006
David	Goldstein	186 Vermont St.	San Francisco	CA	94110
Erin	Valentino	6982 Palm Ave.	Fresno	CA	93711
Frank Lee	Wilson	23 Mountain View St.	Denver	CO	80208
Gary	Hernandez	7361 N. 41st St.	New York	NY	10012

9.

```
SELECT
	c.last_name,
	c.first_name,
	o.order_date,
	p.product_name,
	oi.item_price,
	oi.discount_amount,
	oi.quantity
FROM
	Customers c
	INNER JOIN
	orders o ON c.customer_id = o.customer_id
	INNER JOIN
	order_items oi ON o.order_id = oi.order_id
	INNER JOIN
	products p ON oi.product_id = p.product_id
ORDER BY c.last_name , o.order_date , p.product_name;
```

last_name	first_name	order_date	product_name	item_price	discount_amount	quantity
Brown	Christine	2015-03-30 15:22:31	Gibson Les Paul	1199.00	359.70	2
Goldstein	David	2015-03-31 05:43:11	Washburn D10S	299.00	0.00	1
Goldstein	David	2015-04-03 12:22:31	Fender Stratocaster	699.00	209.70	1
Hernandez	Gary	2015-04-02 11:26:38	Tama 5-Piece Drum Set with Cymbals	799.99	120.00	1
Sherwood	Allan	2015-03-28 09:40:28	Gibson Les Paul	1199.00	359.70	1
Sherwood	Allan	2015-03-29 09:44:58	Gibson SG	2517.00	1308.84	1
Sherwood	Allan	2015-03-29 09:44:58	Rodriguez Caballero 11	415.00	161.85	1
Valentino	Erin	2015-03-31 18:37:22	Washburn D10S	299.00	0.00	1

10.

```
SELECT
	p1.product_name,
	p1.list_price
FROM
	Products p1
JOIN
	Products p2 ON p1.list_price = p2.list_price
WHERE
	p1.product_id <> p2.product_id
ORDER BY
	P1.product_name;
```

product_name	list_price
Fender Precision	799.99
Tama 5-Piece Drum Set with Cymbals	799.99

11.

```
SELECT
	'SHIPPED' AS ship_status, order_id, order_date
FROM
	Orders
WHERE
	ship_date IS NOT NULL
UNION SELECT
	'NOT SHIPPED' AS ship_status, order_id, order_date
FROM
	Orders
WHERE
	ship_date IS NULL
ORDER BY order_date;
```

ship_status	order_id	order_date
SHIPPED	1	2015-03-28 09:40:28
SHIPPED	2	2015-03-28 11:23:20
SHIPPED	3	2015-03-29 09:44:58
SHIPPED	4	2015-03-30 15:22:31
SHIPPED	5	2015-03-31 05:43:11
NOT SHIPPED	6	2015-03-31 18:37:22
SHIPPED	7	2015-04-01 23:11:12
NOT SHIPPED	8	2015-04-02 11:26:38

12.

```
SELECT
      c.category_name,
      COUNT(DISTINCT p.product_id) AS product_count,
      MAX(p.list_price) AS max_product_price
FROM
      categories c
      INNER JOIN
      products p ON c.category_id = p.category_id
GROUP BY c.category_id
ORDER BY product_count DESC;
```

category_name	product_count	max_product_price
Guitars	6	2517.00
Basses	2	799.99
Drums	2	799.99

13.

```
SELECT
      c.email_address,
      COUNT(o.order_id) AS num_orders,
      SUM((oi.item_price - oi.discount_amount) * oi.quantity) AS total_amt
FROM
      customers c
      INNER JOIN
      orders o ON c.customer_id = o.customer_id
      INNER JOIN
      order_items oi ON oi.order_id = o.order_id
GROUP BY c.customer_id , c.email_address
HAVING COUNT(o.order_id) > 1
ORDER BY total_amt DESC;
```

email_address	num_orders	total_amt
allan.sherwood@yahoo.com	3	2300.61
frankwilson@sbcglobal.net	3	1539.28
david.goldstein@hotmail.com	2	788.30

14.

```
SELECT
      c.email_address,
      COUNT(DISTINCT oi.item_id) AS num_products_ordered
FROM
      customers c
      INNER JOIN
      orders o ON c.customer_id = o.customer_id
      INNER JOIN
      order_items oi ON o.order_id = oi.order_id
GROUP BY c.customer_id , c.email_address
HAVING COUNT(DISTINCT oi.product_id) > 1
ORDER BY COUNT(DISTINCT oi.item_id);
```

email_address	num_products_ordered
david.goldstein@hotmail.com	2
allan.sherwood@yahoo.com	3
frankwilson@sbcglobal.net	3

15.

```
SELECT DISTINCT
      product_name, list_price
FROM
      products
WHERE
      list_price > (SELECT
            AVG(list_price)
      FROM
            products)
ORDER BY list_price DESC;
```

product_name	list_price
Gibson SG	2517.00
Gibson Les Paul	1199.00

16.

```
SELECT
      c.category_name
FROM
      categories c
WHERE
      NOT EXISTS( SELECT
            c.category_name
      FROM
            products p
      WHERE
            p.category_id = c.category_id);
```

category_name
Keyboards

17.

```
SELECT
      email_address, MAX(total_amt) AS biggest_order_amt
FROM
    (SELECT
      c.email_address,
      o.order_id,
      SUM((oi.item_price - oi.discount_amount) * oi.quantity) AS total_amt
     FROM
      customers c
     INNER JOIN orders o ON c.customer_id = o.customer_id
     INNER JOIN order_items oi ON o.order_id = oi.order_id
     GROUP BY c.email_address , o.order_id)
                                          AS order_totals
GROUP BY email_address;
```

email_address	biggest_order_amt
allan.sherwood@yahoo.com	1461.31
barryz@gmail.com	303.79
christineb@solarone.com	1678.60
david.goldstein@hotmail.com	489.30
erinv@gmail.com	299.00
frankwilson@sbcglobal.net	1539.28
gary_hernandez@yahoo.com	679.99

18.

```
SELECT
      product_name, discount_percent
FROM
      Products
WHERE
      discount_percent IN (SELECT
            discount_percent
      FROM
            Products
      GROUP BY discount_percent
      HAVING COUNT(*) = 1)
ORDER BY product_name;
```

product_name	discount_percent
Gibson SG	52.00
Hofner Icon	25.00
Rodriguez Caballero 11	39.00
Tama 5-Piece Drum Set with Cymbals	15.00
Washburn D10S	0.00
Yamaha FG700S	38.00

19.

```
SELECT
	list_price,
	FORMAT(list_price, 1),
	CONVERT(list_price , UNSIGNED),
	CAST(list_price AS UNSIGNED),
	date_added,
	CAST(date_added AS DATE),
	DATE_FORMAT(CAST(date_added AS DATE), '%Y-%m'),
	CAST(date_added AS TIME)
FROM
	products;
```

list_price	FORMAT(list_price, 1)	CONVERT(list_price , UNSIGNED)	CAST(list_price AS UNSIGNED)	date_added	CAST(date_added AS DATE)	DATE_FORMAT(CAST(date_added AS DATE), '%Y-%m')	CAST(date_added AS TIME)
699.00	699.0	699	699	2014-10-30 09:32:40	2014-10-30	2014-10	09:32:40
1199.00	1,199.0	1199	1199	2014-12-05 16:33:13	2014-12-05	2014-12	16:33:13
2517.00	2,517.0	2517	2517	2015-02-04 11:04:31	2015-02-04	2015-02	11:04:31
489.99	490.0	490	490	2015-06-01 11:12:59	2015-06-01	2015-06	11:12:59
299.00	299.0	299	299	2015-07-30 13:58:35	2015-07-30	2015-07	13:58:35
415.00	415.0	415	415	2015-07-30 14:12:41	2015-07-30	2015-07	14:12:41
799.99	800.0	800	800	2015-06-01 11:29:35	2015-06-01	2015-06	11:29:35

20.

```
SELECT
	card_number,
	LENGTH(card_number) AS card_num_length,
	RIGHT(card_number, 4) AS final_four,
	CONCAT('XXXX-XXXX-XXXX-', RIGHT(card_number, 4)) AS formatted_num
FROM
	Orders;
```

card_number	card_num_length	final_four	formatted_num
4111111111111111	16	1111	XXXX-XXXX-XXXX-1111
4012888888881881	16	1881	XXXX-XXXX-XXXX-1881
4111111111111111	16	1111	XXXX-XXXX-XXXX-1111
378282246310005	15	0005	XXXX-XXXX-XXXX-0005
4111111111111111	16	1111	XXXX-XXXX-XXXX-1111
6011111111111117	16	1117	XXXX-XXXX-XXXX-1117
5555555555554444	16	4444	XXXX-XXXX-XXXX-4444

21.

```
SELECT
      n.`name` AS neighborhood_name
FROM
      neighborhoods n
      LEFT JOIN
      users u ON n.id = u.neighborhood_id
GROUP BY n.`name`
HAVING COUNT(u.id) = 0;
```

22.

```
SELECT
      u.`name` AS `name`, r.distance AS distance_traveled
FROM
      users u
      LEFT JOIN
      rides r ON u.id = r.passenger_user_id
GROUP BY u.id
ORDER BY r.distance DESC;
```

23.

```
SELECT
      CONCAT(CAST((COUNT(e.id) * 100 / d.total_employees) AS SIGNED), '%')
AS percentage_over_100K,
      d.`name` AS department_name,
      COUNT(e.id) AS number_of_employees
FROM
      departments d
      JOIN (SELECT
      department_id, COUNT(id) AS total_employees
      FROM
      employees
      GROUP BY department_id
      HAVING COUNT(id) >= 10) AS emps_100K ON d.id =
emps_100K.department_id
      JOIN
      employees e ON d.id = e.department_id AND e.salary > 100000
GROUP BY d.`name`
ORDER BY percentage_over_100K DESC
```

```
LIMIT 3;
```

24.

```
SELECT
      df.`date` AS `date`,
      ad.paying_customer AS paying_customer,
      ROUND(AVG(df.downloads), 2) AS average_downloadss
FROM
      user_dimension ud
      INNER JOIN
      account_dimension ad ON ud.account_id = ad.account_id
      INNER JOIN
      download_facts df ON df.user_id = ud.user_id
GROUP BY df.`date` , ad.paying_customer
ORDER BY df.`date` , ad.paying_customer;
```

25.

```
SELECT
      COUNT(DISTINCT t1.user_id) AS num_of_upsold_customers
FROM
      transactions t1
      JOIN
      transactions t2 ON t1.user_id = t2.user_id
      AND t1.product_id != t2.product_id
      AND t1.id != t2.id
      AND DATE(t1.created_at) != DATE(t2.created_at);
```