1. Question asked in DropBox: [LEVEL: EASY]
   Write a query that calculates the difference between the highest salaries found in the marketing and engineering departments. Output just the absolute difference in salaries.
   Tables: db_employee, db_dept

Tables:

db_employee

id:	int
first_name:	varchar
last_name:	varchar
salary:	int
department_id:	int

db_dept

id:	int
department:	varchar

id	first_name	last_name	salary	department_id
10301	Keith	Morgan	27056	2
10302	Tyler	Booth	32199	3
10303	Clifford	Nguyen	32165	2
10304	Mary	Jones	49488	3
10305	Melissa	Lucero	27024	3
10306	Ashley	Li	28516	4
10307	Joseph	Solomon	19945	1

id	department
1	engineering
2	human resource
3	operation
4	marketing
5	sales
6	customer care

Solution:

```
SELECT
    ABS(MAX(a.salary) - MAX(b.salary)) AS sal_difference
FROM
  db_employee AS a,
  db_employee AS b
WHERE
  a.department_id = 4 and b.department_id = 1;
```

2. Question asked in Microsoft: [LEVEL: EASY]

We have a table with employees and their salaries, however, some of the records are old and contain outdated salary information. Find the current salary of each employee assuming that salaries increase each year. Output their id, first name, last name, department ID, and current salary. Order your list by employee ID in ascending order.
Table: ms_employee_salary

Table:

id:	int
first_name:	varchar
last_name:	varchar
salary:	int
department_id:	int

id	first_name	last_name	salary	department_id
1	Todd	Wilson	110000	1006
1	Todd	Wilson	106119	1006
2	Justin	Simon	128922	1005
2	Justin	Simon	130000	1005
3	Kelly	Rosario	42689	1002
4	Patricia	Powell	162825	1004
4	Patricia	Powell	170000	1004
5	Sherry	Golden	44101	1002

Solution:

```
SELECT DISTINCT id, first_name, last_name,department_id, MAX(salary) as Current_Salary
FROM
  ms_employee_salary
GROUP BY
  id, first_name, last_name, department_id
ORDER BY
  id
```

3. Question asked in Lyft: [LEVEL: EASY]

Find the last time each bike was in use. Output both the bike number and the date-timestamp of the bike's last use (i.e., the date-time the bike was returned). Order the results by bikes that were most recently used.
Table: dc_bikeshare_q1_2012

duration:	varchar
duration_seconds:	int
start_time:	datetime
start_station:	varchar
start_terminal:	int
end_time:	datetime
end_station:	varchar
end_terminal:	int
bike_number:	varchar
rider_type:	varchar
id:	int

duration	duration_seconds	start_time	start_station	start_terminal	end_time	end_station	end_terminal	bike_number	rider_type	id
0h 10m 47sec.	647	2012-03-25 10:30:00	17th & Corcoran St NW	31214	2012-03-25 10:40:00	Calvert St & Woodley Pl NW	31106	W00576	Registered	326188
0h 11m 45sec.	705	2012-03-28 18:59:00	Rosslyn Metro / Wilson Blvd & Ft Myer Dr	31015	2012-03-28 19:11:00	21st & M St NW	31212	W00011	Registered	345585
0h 7m 45sec.	465	2012-03-12 22:30:00	3rd & H St NE	31616	2012-03-12 22:37:00	Florida Ave & R St NW	31503	W01215	Registered	251919
0h 4m 27sec.	267	2012-03-12 20:11:00	14th & G St NW	31238	2012-03-12 20:15:00	14th & Rhode Island Ave NW	31203	W00455	Registered	251426
0h 10m 2sec.	602	2012-02-03 09:06:00	Lamont & Mt Pleasant NW	31107	2012-02-03 09:16:00	17th & Rhode Island Ave NW	31239	W00300	Registered	105965
0h 24m 59sec.	1499	2012-03-30 19:35:00	Eastern Market Metro / Pennsylvania Ave & 7th St SE	31613	2012-03-30 20:00:00	Massachusetts Ave & Dupont Circle NW	31200	W01352	Registered	357661
0h 13m 45sec.	825	2012-03-10 16:44:00	North Capitol St & F St NW	31624	2012-03-10 16:58:00	Thomas Circle	31241	W00089	Registered	240483

Solution:

```
SELECT
  DISTINCT bike_number,
    end_time
FROM
  dc_bikeshare_q1_2012
ORDER BY
  end_time DESC
```

4. Question asked Amazon: [LEVEL: MEDIUM]

Write a query that'll identify returning active users. A returning active user is a user that has made a second purchase within 7 days of any other of their purchases. Output a list of user_ids of these returning active users.
Table: amazon_transactions
Tables:

id:	int
user_id:	int
item:	varchar
created_at:	datetime
revenue:	int

id	user_id	item	created_at	revenue
1	109	milk	2020-03-03	123
2	139	biscuit	2020-03-18	421
3	120	milk	2020-03-18	176
4	108	banana	2020-03-18	862
5	130	milk	2020-03-28	333
6	103	bread	2020-03-29	862
7	122	banana	2020-03-07	952
8	125	bread	2020-03-13	317
9	139	bread	2020-03-30	929
10	141	banana	2020-03-17	812
11	116	bread	2020-03-31	226
12	128	bread	2020-03-04	112

```sql
SELECT DISTINCT user_id
FROM (
    SELECT *,
    LEAD(created_at) OVER(partition by user_id order by created_at) AS next_purchase
    FROM amazon_transactions) sbqry
WHERE DATEDIFF(next_purchase, created_at) <= 7;
```