Question A

Assumption:
For each observation, the **dash_start_time** and the **dash_end_time** are of the same day

Query:

```
SELECT WEEKDAY(dash_start_time) AS day_of_week
        , SUM(total_pay)*60/SUM(TIMESTAMPDIFF((minute, dash_start_time,
dash_end_time) AS avg_hourly_earnings
FROM Dash
GROUP BY day_of_week
```

Note:

1. WEEKDAY is the MySQL function that returns the day of week number for a given datetime (0=Monday, …, 6=Sunday). Some other functions do the similar work, e.g. DAYOFWEEK() or DATE_FORMAT(*datetime*, '%a%'). Other SQL languages may use different functions.
2. Here we calculate time difference between dash_start_time and dash_end_time in minute then divide it by 60 so that we can get the hour number in decimal.

When **dash_start_time** and **dash_end_time** are of different days for a record, e.g. overnight or even more than 1 day (although it is unlikely to be true in reality, but it could happen technically). In that case, we may have to write a function that calculates how many hours were allocated in each day-of-week (see the table below). Then we assign the **total_pay** to each day of week proportionally assuming it's evenly distributed over that period. Lastly, we can simply calculate the average hourly pay by each day-of-week

dash_start_time	dash_end_time	total_pay	Mon	Tue	Wed	Thu	Fri	Sat	Sun
1/1/16 7:00 AM	1/5/16 9:40 AM	46.78	24	9.67	0	0	17	24	24

Question B

Assumptions:

1. For each observation in ***Dash***, the **dash_start_time** and the **dash_end_time** are of the same day
2. **Dash.id** is the foreign key references **Dasher.id** (primary key)

Query:

```
SELECT SUM(total_pay)*60/SUM(TIMESTAMPDIFF((minute, dash_start_time,
dash_end_time) AS avg_hourly_earnings
FROM Dash d1 JOIN Dasher d2
ON d1.dasher_id = d2.id
WHERE submarket_id = 3 AND DATE_FORMAT(dash_start_time,'%H') >= 11 AND
DATE_FORMAT(dash_end_time,'%H') < 14
```

Explanation:

- Do the inner join of Dash and Dasher
- Filter by the conditions for submarket_id and lunch time
- Calculate the average earnings per hour using the same logic as in question a.

Note:

DATE_FORMAT(datetime,'%H') returns the hour number in 24-hour format for a given datetime

Question C

Assumptions:

We don't need to assume that for each observation the **dash_start_time** and the **dash_end_time** are of the same day for the case as the average hourly pay by a dasher is calculated by his/her total amount of earnings divided by the total amount of online time.

Query:

```
WITH Dash_Latest_Hourly_Pay AS
(SELECT dash_id
            , SUM(total_pay)*60/SUM(TIMESTAMPDIFF((minute, dash_start_time,
dash_end_time) AS avg_hourly_earnings
 FROM Dash
 WHERE TIMESTAMPDIFF(day, now(), dash_start_time) <= 30
 GROUP BY dash_id
)
SELECT dash_id, email_address
FROM (SELECT dash_id
                    , ROW_NUMBER() OVER(ORDER BY avg_hourly_earnings ASC) AS
rank_number
          FROM Dash_Latest_Hourly_Pay) temp
JOIN Dasher
```

```
ON temp.dash_id = Dasher.id
WHERE temp.rank_number <= (SELECT FLOOR(COUNT(*)/2)
                           FROM Dash_Latest_Hourly_Pay)
```

Explanation:

- Use common table expression (CTE) to create a view named as ***Dash_Latest_Hourly_Pay*** where the average hourly pay for the last 30 days (based on the difference of the current day of system and the **dash_start_time**) is calculated for each dasher
- Get the rank number for each dasher in ***Dash_Latest_Hourly_Pay*** based on average hourly pay in ascending order using a window function **ROW_NUMBER() OVER ()**
- Join Dasher to get the corresponding email address for each dasher
- Select dashers whose rank numbers are smaller than or equal to 50% of the total number of rows of ***Dash_Latest_Hourly_Pay***

Note:

Since we just want to get the bottom 50^{th} percentile of the target dashers instead of calculating the median of total pay per hour, we don't need to deal with the odd-or-even-number issue. Also since ROW_NUMBER() function randomly assigns rank numbers to the ties, we can safely select the records with rank numbers <= FLOOR(COUNT(*)/2) without worrying the cases such as more than 50% dashers had exactly the same hourly pay, which rarely happens but in theory exists.