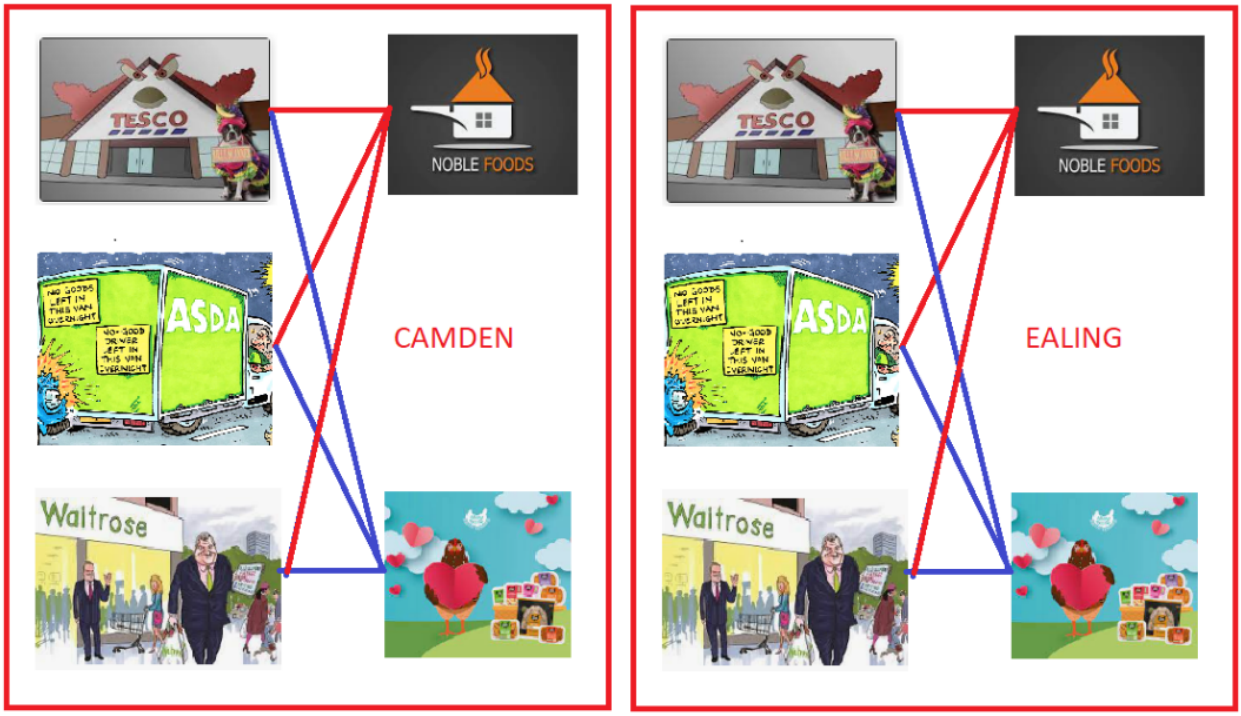
Resource Allocation based on DHondt Method and BigQuery

Problem statement:

We gave three supermarket brands (for example, Lidl, Tesco, and Asda) we denote them Shop 1, Shop 2, and Shop 3. They have branches in Camden, Ealing, Greenwich, Hounslow, Richmond upon Thames, Hammersmith and Fulham, Kensington and Chelsea, and the City of Westminster. They use two suppliers that have branches in the same London Boroughs, we have a historical probability of items being in the given supplier branch of London Borough. We know how many items each supplier can deliver per day.

We are given a total number of items demanded by every supermarket brand and information about when (date) and where (London Borough) the given supermarket can accept part of this total demand but the exact value is not provided. The supplier can deliver items to supermarkets from the same Borough only. We have weights or a probability that the total supply for the given location is going to a particular supermarket. The task is to find the date, location, and number of items that should be delivered to all three supermarkets.



We are given Demand Weights that could be weights of the objective function that are used in linear programming:

Table 1 Demand Weights defined by Optimization Objective in Linear Programming

Demand Weight		
Shop 1	Shop 2	Shop 3
0.2	0.3	0.5

The table below shows how many items are required by every supermarket in all considered locations during two days:

Table 2 Total Number of Demanded Items per Time Interval (2 days in our case)

Shop_Demand		
Shop 1	Shop 2	Shop 3
150	500	300

Information about when (date) and where (London Borough) the given supermarket is available to accept part of the above total demand (see the table called Shop_Demand above) but how much it will accept it depends on the supply ib the given Borough and

Demand Weights.
If it is "+" (True) then the supermarket is available to accept the items on the particular date and London Borough (location) and it does not accept otherwise (when it is "-" or False:

Table 3 Shop Demand Availability or Schedule when Shops update their Stocks

N	Date	London Borough	Shop Demand Availability		
			Shop 1	Shop 2	Shop 3
1	11/06/2022	Camden	-	-	-
2	11/06/2022	Ealing	-	-	+
3	11/06/2022	Greenwich	-	+	-
4	11/06/2022	Hounslow	-	+	+
5	11/06/2022	Richmond upon Thames	+	-	-
6	11/06/2022	Hammersmith and Fulham	+	-	+
7	11/06/2022	Kensington and Chelsea	+	+	-
8	11/06/2022	City of Westminster	+	+	+
9	12/06/2022	Camden	-	-	-
10	12/06/2022	Ealing	-	-	+
11	12/06/2022	Greenwich	-	+	-
12	12/06/2022	Hounslow	-	+	+
13	12/06/2022	Richmond upon Thames	+	-	-
14	12/06/2022	Hammersmith and Fulham	+	-	+
15	12/06/2022	Kensington and Chelsea	+	+	-
16	12/06/2022	City of Westminster	+	+	+
17	Total	-	150	500	300

The information about Supply includes the probability of supply over different London Borough (location) and the daily total amount of the supplied items (see Table below). The probability could be different for the different suppliers but in our example, it is the same for simplicity of presentation:

Table 4 Supply Items Availability

N	Date	London Borough	Supply			
			Supplier 1		Supplier 2	
			Probability	Daily Total	Probability	Daily Total
1	11/06/2022	Camden	0.118	255	0.118	255
2	11/06/2022	Ealing	0.157		0.157	
3	11/06/2022	Greenwich	0.118		0.118	
4	11/06/2022	Hounslow	0.220		0.220	
5	11/06/2022	Richmond upon Thames	0.078		0.078	
6	11/06/2022	Hammersmith and Fulham	0.192		0.192	
7	11/06/2022	Kensington and Chelsea	0.039		0.039	
8	11/06/2022	City of Westminster	0.078		0.078	
9	12/06/2022	Camden	0.118	510	0.118	255
10	12/06/2022	Ealing	0.157		0.157	
11	12/06/2022	Greenwich	0.118		0.118	
12	12/06/2022	Hounslow	0.220		0.220	
13	12/06/2022	Richmond upon Thames	0.078		0.078	
14	12/06/2022	Hammersmith and Fulham	0.192		0.192	
15	12/06/2022	Kensington and Chelsea	0.039		0.039	
16	12/06/2022	City of Westminster	0.078		0.078	
17	Total		2	765	2	510

The algorithm consists of 3 major steps:

STEP 1. Computation of the number of supplied items per day, Borough, and supplier given the probability of supply over different Boroughs and the total number of supplied items per day and supplier. In the simplest case, we just multiply the probability of items distributed to a given Borough by the total number of supplied items per day, and supplier. But sometimes we get no integer numbers after this multiplication that is why we apply DHondt method instead. For example, given date 11/06/2022 and supplier 1 (see Table 4) we have the total amount of items available equal to 255 and the following vector of probabilities: `[0.118, 0.157, 0.118, 0.220, 0.078, 0.192, 0.039, 0.078]` over London Boroughs. This information is required by DHondt method to find the number of items that can be supplied on 11/06/2022 by Supplier 1 to each London Borough. Therefore, for every combination of date and supply we need to run DHondt method or 4 times in our case (see Quantity column in Table 5).

Table 5 Supply Information with estimated “Quantity” and “ALL” Columns

N	Date	London Borough	Supply						ALL
			Supplier 1			Supplier 2			
			Probability	Quantity	Total	Probability	Quantity	Total	
1	11/06/2022	Camden	0.118	30	255	0.118	30	255	60
2	11/06/2022	Ealing	0.157	40		0.157	40		80
3	11/06/2022	Greenwich	0.118	30		0.118	30		60
4	11/06/2022	Hounslow	0.220	56		0.220	56		112
5	11/06/2022	Richmond upon Thames	0.078	20		0.078	20		40
6	11/06/2022	Hammersmith and Fulham	0.192	49		0.192	49		98
7	11/06/2022	Kensington and Chelsea	0.039	10		0.039	10		20
8	11/06/2022	City of Westminster	0.078	20		0.078	20		40
9	12/06/2022	Camden	0.118	60	510	0.118	30	255	90
10	12/06/2022	Ealing	0.157	80		0.157	40		120
11	12/06/2022	Greenwich	0.118	60		0.118	30		90
12	12/06/2022	Hounslow	0.220	112		0.220	56		168
13	12/06/2022	Richmond upon Thames	0.078	40		0.078	20		60
14	12/06/2022	Hammersmith and Fulham	0.192	98		0.192	49		147
15	12/06/2022	Kensington and Chelsea	0.039	20		0.039	10		30
16	12/06/2022	City of Westminster	0.078	40		0.078	20		60
17	Total			765	765		510	510	1275

STEP 2. Calculation of the maximum demand that can be satisfied or supplied. Using demand weights, demand availability, and the total possible number of supplied items per day and Borough (see 1)) we compute how many items can be supplied given day, Borough, and shop. Therefore, we find the maximum number of items that the shop located in the corresponding London Borough can get on the given date. In order to find it we first multiply every row of Shop Demand Availability (see Table 3, where "+" can be interpreted as 1 and "-" as 0) by Demand Weights and normalise every row of the result in such a way that weights in the row sum to 1. Then using this row normalised weights and the total possible supplied in the given London Borough and date we calculate the number of the items that can be delivered to the corresponding shop on the given date and London Borough. We can apply DHondt method again - we need to run it 16 times for every combination of date and London Borough, the output of the DHondt in this case is the number of items (in our case, the output is a vector of three values - one value per shop) for the corresponding shop given date and London Borough (see Table 6). Then we aggregate the number of items over all dates and London Borough for every shop. Therefore, we get the number of items that suppliers can provide to the corresponding entire supermarket network for all location and dates. then we compare these numbers with Shop Demand (see Table 2) and for every shop choose the smallest number (see Table 7). Therefore, we find the number of items that are required by the shops and can be delivered by suppliers subject to constrain (see Table 1-4) over all days (in our case two days) and London Borough (we selected eight London Boroughs).

Table 6 Distribution of Number of Available for Supply Items over Shops given Date and London Borough (see Supply AS Demand in red colour)

N	Date	London Borough	Shop Demand Availability			Demand Weight			Row Normalised Weight			Supply AS Demand		
			Shop 1	Shop 2	Shop 3	Shop 1	Shop 2	Shop 3	Shop 1	Shop 2	Shop 3	Shop 1	Shop 2	Shop 3
1	11/06/2022	Camden	-	-	-	0	0	0	0	0	0	0	0	0
2	11/06/2022	Ealing	-	-	+	0	0	0.5	0	0	1	0	0	80
3	11/06/2022	Greenwich	-	+	-	0	0.3	0	0	1	0	0	60	0
4	11/06/2022	Hounslow	-	+	+	0	0.3	0.5	0	0.375	0.625	0	42	70
5	11/06/2022	Richmond upon Thames	+	-	-	0.2	0	0	1	0	0	40	0	0
6	11/06/2022	Hammersmith and Fulham	+	-	+	0.2	0	0.5	0.285714	0	0.714286	28	0	70
7	11/06/2022	Kensington and Chelsea	+	+	-	0.2	0.3	0	0.4	0.6	0	8	12	0
8	11/06/2022	City of Westminster	+	+	+	0.2	0.3	0.5	0.2	0.3	0.5	8	12	20
9	12/06/2022	Camden	-	-	-	0	0	0	0	0	0	0	0	0
10	12/06/2022	Ealing	-	-	+	0	0	0.5	0	0	1	0	0	120
11	12/06/2022	Greenwich	-	+	-	0	0.3	0	0	1	0	0	90	0
12	12/06/2022	Hounslow	-	+	+	0	0.3	0.5	0	0.375	0.625	0	63	105
13	12/06/2022	Richmond upon Thames	+	-	-	0.2	0	0	1	0	0	60	0	0
14	12/06/2022	Hammersmith and Fulham	+	-	+	0.2	0	0.5	0.285714	0	0.714286	42	0	105
15	12/06/2022	Kensington and Chelsea	+	+	-	0.2	0.3	0	0.4	0.6	0	12	18	0
16	12/06/2022	City of Westminster	+	+	+	0.2	0.3	0.5	0.2	0.3	0.5	12	18	30
17	Total	-	150	500	300	-						210	315	600

Table 7 Available Demand Calculation

Super Market	Demand	Supply	Available Demand
Shop 1	150	210	150
Shop 2	500	315	315
Shop 3	300	600	300

STEP 3. Find weights for each shop using the corresponding column of Supply AS Demand (see Table 6) that is divided the corresponding total (for example, the total equal to 210, 315 and 600 for shop 1, 2 and 3, respectively, see Table 6). As a result we get “Column Normalise Weight” (see Table 8). After this we can multiply every column of “Column Normalised Weight” (see Table 8) that corresponds to the given shop by the corresponding value of “Available Demand” (see Table 7) to get the final allocation. As we mentioned before this approach give some float numbers therefore instead we apply DHondt method to every column of “Column Normalised Weight” (see Table 8) using the corresponding value of Available Demand (see Table 7). In our case we have three shops therefore we apply DHondt method three times and we get “Final Allocation” (see Table 8). Please note that, the following implementation in BigQuery gives slightly different numbers than what we get in Table 8 but it can be explained by difference in Python and BigQuery DHondt method implementations.

Table 8

N	Date	London Borough	Supply AS Demand			Column Normalised Weight			Final Allocation		
			Shop 1	Shop 2	Shop 3	Shop 1	Shop 2	Shop 3	Shop 1	Shop 2	Shop 3
1	11/06/2022	Camden	0	0	0	0	0	0	0	0	0
2	11/06/2022	Ealing	0	0	80	0	0	0.133333	0	0	40
3	11/06/2022	Greenwich	0	60	0	0	0.190476	0	0	60	0
4	11/06/2022	Hounslow	0	42	70	0	0.133333	0.116667	0	42	35
5	11/06/2022	Richmond upon Thames	40	0	0	0.190476	0	0	29	0	0
6	11/06/2022	Hammersmith and Fulham	28	0	70	0.133333	0	0.116667	20	0	35
7	11/06/2022	Kensington and Chelsea	8	12	0	0.038095	0.038095	0	5	12	0
8	11/06/2022	City of Westminster	8	12	20	0.038095	0.038095	0.033333	5	12	10
9	11/06/2022	Camden	0	0	0	0	0	0	0	0	0
10	12/06/2022	Ealing	0	0	120	0	0	0.2	0	0	60
11	12/06/2022	Greenwich	0	90	0	0	0.285714	0	0	90	0
12	12/06/2022	Hounslow	0	63	105	0	0.2	0.175	0	63	53
13	12/06/2022	Richmond upon Thames	60	0	0	0.285714	0	0	44	0	0
14	12/06/2022	Hammersmith and Fulham	42	0	105	0.2	0	0.175	31	0	52
15	12/06/2022	Kensington and Chelsea	12	18	0	0.057143	0.057143	0	8	18	0
15	12/06/2022	City of Westminster	12	18	30	0.057143	0.057143	0.05	8	18	15
17	Total	-	210	315	600	1	1	1	150	315	300

Our BigQuery implementation of DHondt algorithm was inspired by the following Python implementation:

```
def DHondt_AD_v2(nSeats: int, weights: np.ndarray) -> np.ndarray:
    #
    if nSeats == 0:
       seats = np.arrasy([0] * len(weights))
    else:
       quota = sum(weights) / (1 + nSeats)
       frac = [weight / quota for weight in weights]
       seats = np.floor(frac).astype(int)
       #
       n = nSeats - sum(seats)  # number of seats remaining to allocate
       #
       if n < 0: # we over allocate by 1 seat
          # I modified over allocation case - it could over allocate only by 1 seat
          min_weights = np.min(weights[np.nonzero(seats)])
          # find smallest weight where seats is non zero
          index = np.argmax(weights == min_weights)
          # find smallest index for min_weights
          seats[index] -= 1                      #
       elif n > 0: # We under allocate by n seats after initial allocation
            #
            # distribute the remaining number of seats to the elements of seats array
            # with the largest remainder
            remainders = frac - seats
            #
            n = int(n)
            # Find poistion of the first n largest reminders
            # in order to increase the number of seats
            index = np.argsort(remainders)[::-1][:n]
            #
            for i in index:
```

```
                seats[i] += 1
    #
    return seats
#
print("OK")
```

1. Inputs Definition

In this section we define inputs that illustrate our implementation of resource allocation algorithm (see Table 1-4 above).

```
demand_weight AS (
      SELECT  'shop_1' AS shop, 0.2 AS demand_weight UNION ALL
      SELECT  'shop_2',         0.3                  UNION ALL
      SELECT  'shop_3',         0.5
)
```

Row	shop	demand_weight
1	shop_1	0.2
2	shop_2	0.3
3	shop_3	0.5

```
shop_demand   AS(
      SELECT  'shop_1' AS shop, 150 AS demand_quantity UNION ALL
      SELECT  'shop_2',         500                    UNION ALL
      SELECT  'shop_3',         300
)
```

Row	shop	demand_quantity
1	shop_1	150
2	shop_2	500
3	shop_3	300

```
demand AS (
      SELECT '11/06/2022' AS day, 'Camden'                 AS London_Borough,
             False AS Shop_1,  False AS Shop_2,  False  AS Shop_3 UNION ALL
      SELECT '11/06/2022' AS day, 'Ealing'                 AS London_Borough,
             False AS Shop_1,  False AS Shop_2,  True   AS Shop_3 UNION ALL
      SELECT '11/06/2022' AS day, 'Greenwich'              AS London_Borough,
             False AS Shop_1,  True  AS Shop_2,  False  AS Shop_3 UNION ALL
      SELECT '11/06/2022' AS day, 'Hounslow'               AS London_Borough,
             False AS Shop_1,  True  AS Shop_2,  True   AS Shop_3 UNION ALL
      SELECT '11/06/2022' AS day, 'Richmond upon Thames'   AS London_Borough,
             True  AS Shop_1,  False AS Shop_2,  False  AS Shop_3 UNION ALL
      SELECT '11/06/2022' AS day, 'Hammersmith and Fulham' AS London_Borough,
             True  AS Shop_1,  False AS Shop_2,  True   AS Shop_3 UNION ALL
      SELECT '11/06/2022' AS day, 'Kensington and Chelsea' AS London_Borough,
             True  AS Shop_1,  True  AS Shop_2,  False  AS Shop_3 UNION ALL
      SELECT '11/06/2022' AS day, 'City of Westminster'    AS London_Borough,
             True  AS Shop_1,  True  AS Shop_2,  True   AS Shop_3 UNION ALL
      SELECT '12/06/2022' AS day, 'Camden'                 AS London_Borough,
             False AS Shop_1,  False AS Shop_2,  False  AS Shop_3 UNION ALL
      SELECT '12/06/2022' AS day, 'Ealing'                 AS London_Borough,
             False AS Shop_1,  False AS Shop_2,  True   AS Shop_3 UNION ALL
      SELECT '12/06/2022' AS day, 'Greenwich'              AS London_Borough,
             False AS Shop_1,  True  AS Shop_2,  False  AS Shop_3 UNION ALL
      SELECT '12/06/2022' AS day, 'Hounslow'               AS London_Borough,
             False AS Shop_1,  True  AS Shop_2,  True   AS Shop_3 UNION ALL
      SELECT '12/06/2022' AS day, 'Richmond upon Thames'   AS London_Borough,
             True  AS Shop_1,  False AS Shop_2,  False  AS Shop_3 UNION ALL
      SELECT '12/06/2022' AS day, 'Hammersmith and Fulham' AS London_Borough,
             True  AS Shop_1,  False AS Shop_2,  True   AS Shop_3 UNION ALL
      SELECT '12/06/2022' AS day, 'Kensington and Chelsea' AS London_Borough,
             True  AS Shop_1,  True  AS Shop_2,  False  AS Shop_3 UNION ALL
      SELECT '12/06/2022' AS day, 'City of Westminster'    AS London_Borough,
             True  AS Shop_1,  True  AS Shop_2,  True   AS Shop_3
)
```

Row	day	London_Borough	Shop_1	Shop_2	Shop_3
1	11/06/2022	Camden	false	false	false
2	11/06/2022	Ealing	false	false	true
3	11/06/2022	Greenwich	false	true	false
4	11/06/2022	Hounslow	false	true	true
5	11/06/2022	Richmond upon Thames	true	false	false
6	11/06/2022	Hammersmith and Fulham	true	false	true
7	11/06/2022	Kensington and Chelsea	true	true	false
8	11/06/2022	City of Westminster	true	true	true
9	12/06/2022	Camden	false	false	false
10	12/06/2022	Ealing	false	false	true
11	12/06/2022	Greenwich	false	true	false
12	12/06/2022	Hounslow	false	true	true
13	12/06/2022	Richmond upon Thames	true	false	false
14	12/06/2022	Hammersmith and Fulham	true	false	true
15	12/06/2022	Kensington and Chelsea	true	true	false
16	12/06/2022	City of Westminster	true	true	true

```
-- SUPPLY
prob_of_supply_per_borough AS (
SELECT 'Camden'                 AS London_Borough, 0.118 AS supplier_1,
                                                   0.118 AS supplier_2 UNION ALL
SELECT 'Ealing'                 AS London_Borough, 0.157 AS supplier_1,
                                                   0.157 AS supplier_2 UNION ALL
SELECT 'Greenwich'              AS London_Borough, 0.118 AS supplier_1,
                                                   0.118 AS supplier_2 UNION ALL
SELECT 'Hounslow'               AS London_Borough, 0.220 AS supplier_1,
                                                   0.220 AS supplier_2 UNION ALL
SELECT 'Richmond upon Thames'   AS London_Borough, 0.078 AS supplier_1,
                                                   0.078 AS supplier_2 UNION ALL
SELECT 'Hammersmith and Fulham' AS London_Borough, 0.192 AS supplier_1,
                                                   0.192 AS supplier_2 UNION ALL
SELECT 'Kensington and Chelsea' AS London_Borough, 0.039 AS supplier_1,
                                                   0.039 AS supplier_2 UNION ALL
SELECT 'City of Westminster'    AS London_Borough, 0.078 AS supplier_1,
                                                   0.078 AS supplier_2
)
```

Row	London_Borough	supplier_1	supplier_2
1	Camden	0.118	0.118
2	Ealing	0.157	0.157
3	Greenwich	0.118	0.118
4	Hounslow	0.22	0.22
5	Richmond upon Thames	0.078	0.078
6	Hammersmith and Fulham	0.192	0.192
7	Kensington and Chelsea	0.039	0.039
8	City of Westminster	0.078	0.078

```
daily_total_supplied_quantity AS(
SELECT '11/06/2022' AS day, 255 AS supplier_1, 255 AS supplier_2 UNION ALL
SELECT '12/06/2022' AS day, 510 AS supplier_1, 255 AS supplier_2
)
```

Row	day	supplier_1	supplier_2
1	11/06/2022	255	255
2	12/06/2022	510	255

2. Daily Supplier Quantity per Day, Borough, and Supplier by DHondt

The output of this section is the column “daily_supplier_allocation” in the table called final_step_of_DHondt_daily_supplier (see column called `Quantity` in Table 5 above).

```
un_pivot_prob_of_supply_per_borough AS (
SELECT London_Borough, supplier, probability FROM prob_of_supply_per_borough
UNPIVOT(probability FOR supplier IN (supplier_1, supplier_2))
)
```

Row	London_Borough	supplier	probability
1	Camden	supplier_1	0.118
2	Camden	supplier_2	0.118
3	Ealing	supplier_1	0.157
4	Ealing	supplier_2	0.157
5	Greenwich	supplier_1	0.118
6	Greenwich	supplier_2	0.118
7	Hounslow	supplier_1	0.22
8	Hounslow	supplier_2	0.22
9	Richmond upon Thames	supplier_1	0.078
10	Richmond upon Thames	supplier_2	0.078
11	Hammersmith and Fulham	supplier_1	0.192
12	Hammersmith and Fulham	supplier_2	0.192
13	Kensington and Chelsea	supplier_1	0.039
14	Kensington and Chelsea	supplier_2	0.039
15	City of Westminster	supplier_1	0.078
16	City of Westminster	supplier_2	0.078

```
un_pivot_daily_total_supplied_quantity AS (
SELECT day, supplier, quantity FROM daily_total_supplied_quantity
UNPIVOT(quantity FOR supplier IN (supplier_1, supplier_2))
)
```

Row	day	supplier	quantity
1	11/06/2022	supplier_1	255
2	11/06/2022	supplier_2	255
3	12/06/2022	supplier_1	510
4	12/06/2022	supplier_2	255

```
-- START DHondt_daily_supplier
initial_step_of_DHondt_daily_supplier AS
(
SELECT day,
       London_Borough,
       d.supplier,
       probability,
       quantity,
       probability * (quantity + 1)        AS fraction,
       FLOOR(probability * (quantity + 1)) AS res,

       SUM( FLOOR(probability * (quantity + 1)) ) OVER (PARTITION BY day, d.supplier)
            AS total_res,
       quantity -
            SUM( FLOOR(probability * (quantity + 1)) )
            OVER (PARTITION BY day, d.supplier)  AS n,
       ROW_NUMBER() OVER (
                    PARTITION BY day, d.supplier
                    ORDER BY (       probability * (quantity + 1) -
                                FLOOR(probability * (quantity + 1)) ) DESC)
                    AS th_largest_reminder,
       ROW_NUMBER() OVER (
                    PARTITION BY day,
                    d.supplier ORDER BY probability)
```

```
                    AS smallest_weight
FROM un_pivot_daily_total_supplied_quantity AS d
INNER JOIN un_pivot_prob_of_supply_per_borough AS p ON d.supplier = p.supplier
),
final_step_of_DHondt_daily_supplier as (
SELECT *,
       CAST(CASE WHEN (n = -1) AND (smallest_weight      = 1) THEN res - 1
                 WHEN (n >  0) AND (th_largest_reminder <= n) THEN res + 1
                 ELSE res
            END AS INT64) AS daily_supplier_allocation
FROM initial_step_of_DHondt_daily_supplier
),
-- END   DHondt_daily_supplier
```

Row	day	London_Borough	supplier	probability	quantity	fraction	res	total_res	n	th_largest_reminder	smallest_weight	daily_supplier_allocation
1	11/06/2022	Kensington and Chelsea	supplier_1	0.039	255	9.984	9.0	252.0	3.0	1	1	10
2	11/06/2022	Richmond upon Thames	supplier_1	0.078	255	19.968	19.0	252.0	3.0	2	2	20
3	11/06/2022	City of Westminster	supplier_1	0.078	255	19.968	19.0	252.0	3.0	3	3	20
4	11/06/2022	Camden	supplier_1	0.118	255	30.208	30.0	252.0	3.0	5	4	30
5	11/06/2022	Greenwich	supplier_1	0.118	255	30.208	30.0	252.0	3.0	6	5	30
6	11/06/2022	Ealing	supplier_1	0.157	255	40.192	40.0	252.0	3.0	7	6	40
7	11/06/2022	Hammersmith and Fulham	supplier_1	0.192	255	49.152	49.0	252.0	3.0	8	7	49
8	11/06/2022	Hounslow	supplier_1	0.22	255	56.32	56.0	252.0	3.0	4	8	56
9	11/06/2022	Kensington and Chelsea	supplier_2	0.039	255	9.984	9.0	252.0	3.0	1	1	10
10	11/06/2022	Richmond upon Thames	supplier_2	0.078	255	19.968	19.0	252.0	3.0	2	2	20
11	11/06/2022	City of Westminster	supplier_2	0.078	255	19.968	19.0	252.0	3.0	3	3	20
12	11/06/2022	Camden	supplier_2	0.118	255	30.208	30.0	252.0	3.0	5	4	30
13	11/06/2022	Greenwich	supplier_2	0.118	255	30.208	30.0	252.0	3.0	6	5	30
14	11/06/2022	Ealing	supplier_2	0.157	255	40.192	40.0	252.0	3.0	7	6	40
15	11/06/2022	Hammersmith and Fulham	supplier_2	0.192	255	49.152	49.0	252.0	3.0	8	7	49
16	11/06/2022	Hounslow	supplier_2	0.22	255	56.32	56.0	252.0	3.0	4	8	56

17	12/06/2022	Kensington and Chelsea	supplier_1	0.039	510	19.929	19.0	507.0	3.0	1	1	20
18	12/06/2022	Richmond upon Thames	supplier_1	0.078	510	39.858	39.0	507.0	3.0	2	2	40
19	12/06/2022	City of Westminster	supplier_1	0.078	510	39.858	39.0	507.0	3.0	3	3	40
20	12/06/2022	Camden	supplier_1	0.118	510	60.297999999999995	60.0	507.0	3.0	5	4	60
21	12/06/2022	Greenwich	supplier_1	0.118	510	60.297999999999995	60.0	507.0	3.0	6	5	60
22	12/06/2022	Ealing	supplier_1	0.157	510	80.227	80.0	507.0	3.0	7	6	80
23	12/06/2022	Hammersmith and Fulham	supplier_1	0.192	510	98.112000000000009	98.0	507.0	3.0	8	7	98
24	12/06/2022	Hounslow	supplier_1	0.22	510	112.42	112.0	507.0	3.0	4	8	112
25	12/06/2022	Kensington and Chelsea	supplier_2	0.039	255	9.984	9.0	252.0	3.0	1	1	10
26	12/06/2022	Richmond upon Thames	supplier_2	0.078	255	19.968	19.0	252.0	3.0	2	2	20
27	12/06/2022	City of Westminster	supplier_2	0.078	255	19.968	19.0	252.0	3.0	3	3	20
28	12/06/2022	Camden	supplier_2	0.118	255	30.208	30.0	252.0	3.0	5	4	30
29	12/06/2022	Greenwich	supplier_2	0.118	255	30.208	30.0	252.0	3.0	6	5	30
30	12/06/2022	Ealing	supplier_2	0.157	255	40.192	40.0	252.0	3.0	7	6	40
31	12/06/2022	Hammersmith and Fulham	supplier_2	0.192	255	49.152	49.0	252.0	3.0	8	7	49
32	12/06/2022	Hounslow	supplier_2	0.22	255	56.32	56.0	252.0	3.0	4	8	56

3. Row Normalised Shop Weights per Day, Borough, and Shop

In this section we compute `Row Normalised Weight` (see Table 6) that we need to compute the number of items supplier can provide to the given shop, date and London Borough.

```
un_pivot_demand AS (
SELECT day, London_Borough, shop, indicator FROM demand
UNPIVOT(indicator FOR shop IN (shop_1, shop_2, shop_3))
)
```

Row	day	London_Borough	shop	indicator
1	11/06/2022	Camden	shop_1	FALSE
2	11/06/2022	Camden	shop_2	FALSE
3	11/06/2022	Camden	shop_3	FALSE
4	11/06/2022	Ealing	shop_1	FALSE
5	11/06/2022	Ealing	shop_2	FALSE
6	11/06/2022	Ealing	shop_3	TRUE
7	11/06/2022	Greenwich	shop_1	FALSE
8	11/06/2022	Greenwich	shop_2	TRUE
9	11/06/2022	Greenwich	shop_3	FALSE
10	11/06/2022	Hounslow	shop_1	FALSE
11	11/06/2022	Hounslow	shop_2	TRUE
12	11/06/2022	Hounslow	shop_3	TRUE
13	11/06/2022	Richmond upon Thames	shop_1	TRUE
14	11/06/2022	Richmond upon Thames	shop_2	FALSE
15	11/06/2022	Richmond upon Thames	shop_3	FALSE
16	11/06/2022	Hammersmith and Fulham	shop_1	TRUE
17	11/06/2022	Hammersmith and Fulham	shop_2	FALSE
18	11/06/2022	Hammersmith and Fulham	shop_3	TRUE
19	11/06/2022	Kensington and Chelsea	shop_1	TRUE
20	11/06/2022	Kensington and Chelsea	shop_2	TRUE
21	11/06/2022	Kensington and Chelsea	shop_3	FALSE
22	11/06/2022	City of Westminster	shop_1	TRUE
23	11/06/2022	City of Westminster	shop_2	TRUE
24	11/06/2022	City of Westminster	shop_3	TRUE
25	12/06/2022	Camden	shop_1	FALSE
26	12/06/2022	Camden	shop_2	FALSE
27	12/06/2022	Camden	shop_3	FALSE
28	12/06/2022	Ealing	shop_1	FALSE
29	12/06/2022	Ealing	shop_2	FALSE
30	12/06/2022	Ealing	shop_3	TRUE
31	12/06/2022	Greenwich	shop_1	FALSE
32	12/06/2022	Greenwich	shop_2	TRUE
33	12/06/2022	Greenwich	shop_3	FALSE
34	12/06/2022	Hounslow	shop_1	FALSE
35	12/06/2022	Hounslow	shop_2	TRUE
36	12/06/2022	Hounslow	shop_3	TRUE
37	12/06/2022	Richmond upon Thames	shop_1	TRUE
38	12/06/2022	Richmond upon Thames	shop_2	FALSE
39	12/06/2022	Richmond upon Thames	shop_3	FALSE
40	12/06/2022	Hammersmith and Fulham	shop_1	TRUE
41	12/06/2022	Hammersmith and Fulham	shop_2	FALSE
42	12/06/2022	Hammersmith and Fulham	shop_3	TRUE
43	12/06/2022	Kensington and Chelsea	shop_1	TRUE
44	12/06/2022	Kensington and Chelsea	shop_2	TRUE
45	12/06/2022	Kensington and Chelsea	shop_3	FALSE
46	12/06/2022	City of Westminster	shop_1	TRUE
47	12/06/2022	City of Westminster	shop_2	TRUE
48	12/06/2022	City of Westminster	shop_3	TRUE

```
deman_weight_un_pivot_demand AS (
SELECT day,
       London_Borough,
       u.shop,
       indicator,
```

```
       demand_weight,
       CASE WHEN indicator THEN demand_weight ELSE 0 END AS indicator_weight,

FROM un_pivot_demand AS u
INNER JOIN demand_weight AS d ON d.shop=u.shop
)
```

Row	day	London_Borough	shop	indicator	demand_weight	indicator_weight
1	11/06/2022	Camden	shop_1	FALSE	0.2	0
2	11/06/2022	Camden	shop_2	FALSE	0.3	0
3	11/06/2022	Camden	shop_3	FALSE	0.5	0
4	11/06/2022	Ealing	shop_1	FALSE	0.2	0
5	11/06/2022	Ealing	shop_2	FALSE	0.3	0
6	11/06/2022	Ealing	shop_3	TRUE	0.5	0.5
7	11/06/2022	Greenwich	shop_1	FALSE	0.2	0
8	11/06/2022	Greenwich	shop_2	TRUE	0.3	0.3
9	11/06/2022	Greenwich	shop_3	FALSE	0.5	0
10	11/06/2022	Hounslow	shop_1	FALSE	0.2	0
11	11/06/2022	Hounslow	shop_2	TRUE	0.3	0.3
12	11/06/2022	Hounslow	shop_3	TRUE	0.5	0.5
13	11/06/2022	Richmond upon Thames	shop_1	TRUE	0.2	0.2
14	11/06/2022	Richmond upon Thames	shop_2	FALSE	0.3	0
15	11/06/2022	Richmond upon Thames	shop_3	FALSE	0.5	0
16	11/06/2022	Hammersmith and Fulham	shop_1	TRUE	0.2	0.2
17	11/06/2022	Hammersmith and Fulham	shop_2	FALSE	0.3	0
18	11/06/2022	Hammersmith and Fulham	shop_3	TRUE	0.5	0.5
19	11/06/2022	Kensington and Chelsea	shop_1	TRUE	0.2	0.2
20	11/06/2022	Kensington and Chelsea	shop_2	TRUE	0.3	0.3
21	11/06/2022	Kensington and Chelsea	shop_3	FALSE	0.5	0
22	11/06/2022	City of Westminster	shop_1	TRUE	0.2	0.2
23	11/06/2022	City of Westminster	shop_2	TRUE	0.3	0.3
24	11/06/2022	City of Westminster	shop_3	TRUE	0.5	0.5
25	12/06/2022	Camden	shop_1	FALSE	0.2	0
26	12/06/2022	Camden	shop_2	FALSE	0.3	0
27	12/06/2022	Camden	shop_3	FALSE	0.5	0
28	12/06/2022	Ealing	shop_1	FALSE	0.2	0
29	12/06/2022	Ealing	shop_2	FALSE	0.3	0
30	12/06/2022	Ealing	shop_3	TRUE	0.5	0.5
31	12/06/2022	Greenwich	shop_1	FALSE	0.2	0
32	12/06/2022	Greenwich	shop_2	TRUE	0.3	0.3
33	12/06/2022	Greenwich	shop_3	FALSE	0.5	0
34	12/06/2022	Hounslow	shop_1	FALSE	0.2	0
35	12/06/2022	Hounslow	shop_2	TRUE	0.3	0.3
36	12/06/2022	Hounslow	shop_3	TRUE	0.5	0.5
37	12/06/2022	Richmond upon Thames	shop_1	TRUE	0.2	0.2
38	12/06/2022	Richmond upon Thames	shop_2	FALSE	0.3	0
39	12/06/2022	Richmond upon Thames	shop_3	FALSE	0.5	0
40	12/06/2022	Hammersmith and Fulham	shop_1	TRUE	0.2	0.2
41	12/06/2022	Hammersmith and Fulham	shop_2	FALSE	0.3	0
42	12/06/2022	Hammersmith and Fulham	shop_3	TRUE	0.5	0.5
43	12/06/2022	Kensington and Chelsea	shop_1	TRUE	0.2	0.2
44	12/06/2022	Kensington and Chelsea	shop_2	TRUE	0.3	0.3
45	12/06/2022	Kensington and Chelsea	shop_3	FALSE	0.5	0
46	12/06/2022	City of Westminster	shop_1	TRUE	0.2	0.2
47	12/06/2022	City of Westminster	shop_2	TRUE	0.3	0.3
48	12/06/2022	City of Westminster	shop_3	TRUE	0.5	0.5

```
row_normalised_deman_weight_un_pivot_demand AS (
 SELECT
       day,
       London_Borough,
       shop,
```

```
        indicator,
        demand_weight,
        indicator_weight,
        CASE WHEN SUM(indicator_weight) OVER(PARTITION BY day, London_Borough) > 0
                  THEN indicator_weight / SUM(indicator_weight)
                                          OVER(PARTITION BY day, London_Borough)
                  ELSE 0
        END AS row_normalise_weight
FROM deman_weight_un_pivot_demand
WHERE indicator_weight > 0
)
```

Row	day	London_Borough	shop	indicator	demand_weight	indicator_weight	row_normalise_weight
1	11/06/2022	City of Westminster	shop_1	TRUE	0.2	0.2	0.2
2	11/06/2022	City of Westminster	shop_2	TRUE	0.3	0.3	0.3
3	11/06/2022	City of Westminster	shop_3	TRUE	0.5	0.5	0.5
4	11/06/2022	Ealing	shop_3	TRUE	0.5	0.5	1
5	11/06/2022	Greenwich	shop_2	TRUE	0.3	0.3	1
6	11/06/2022	Hammersmith and Fulham	shop_1	TRUE	0.2	0.2	0.285714286
7	11/06/2022	Hammersmith and Fulham	shop_3	TRUE	0.5	0.5	0.714285714
8	11/06/2022	Hounslow	shop_2	TRUE	0.3	0.3	0.375
9	11/06/2022	Hounslow	shop_3	TRUE	0.5	0.5	0.625
10	11/06/2022	Kensington and Chelsea	shop_1	TRUE	0.2	0.2	0.4
11	11/06/2022	Kensington and Chelsea	shop_2	TRUE	0.3	0.3	0.6
12	11/06/2022	Richmond upon Thames	shop_1	TRUE	0.2	0.2	1
13	12/06/2022	City of Westminster	shop_1	TRUE	0.2	0.2	0.2
14	12/06/2022	City of Westminster	shop_2	TRUE	0.3	0.3	0.3
15	12/06/2022	City of Westminster	shop_3	TRUE	0.5	0.5	0.5
16	12/06/2022	Ealing	shop_3	TRUE	0.5	0.5	1
17	12/06/2022	Greenwich	shop_2	TRUE	0.3	0.3	1
18	12/06/2022	Hammersmith and Fulham	shop_1	TRUE	0.2	0.2	0.285714286
19	12/06/2022	Hammersmith and Fulham	shop_3	TRUE	0.5	0.5	0.714285714
20	12/06/2022	Hounslow	shop_2	TRUE	0.3	0.3	0.375
21	12/06/2022	Hounslow	shop_3	TRUE	0.5	0.5	0.625
22	12/06/2022	Kensington and Chelsea	shop_1	TRUE	0.2	0.2	0.4
23	12/06/2022	Kensington and Chelsea	shop_2	TRUE	0.3	0.3	0.6
24	12/06/2022	Richmond upon Thames	shop_1	TRUE	0.2	0.2	1

4. Available Demand

In this section we compute `Supply AS Demand` (see Table 6) and `Avalaible Demand` (see Table 7).

```
-- START DHondt_supply_AS_demand
initial_step_of_DHondt_supply_AS_demand AS (
SELECT
        r.day,
        r.London_Borough,
```

```
       shop,
       row_normalise_weight,
       daily_supplier_allocation,


       row_normalise_weight * (daily_supplier_allocation + 1)        AS fraction,
       FLOOR(row_normalise_weight * (daily_supplier_allocation + 1)) AS res,

       SUM( FLOOR(row_normalise_weight * (daily_supplier_allocation + 1)) )
            OVER (PARTITION BY r.day, r.London_Borough) AS total_res,
       daily_supplier_allocation  -
            SUM( FLOOR(row_normalise_weight * (daily_supplier_allocation + 1)) )
                 OVER (PARTITION BY r.day, r.London_Borough)  AS n,
       ROW_NUMBER() OVER (
                    PARTITION BY r.day, r.London_Borough
                    ORDER BY ( row_normalise_weight * (daily_supplier_allocation + 1) -
                    FLOOR(row_normalise_weight * (daily_supplier_allocation + 1)) ) DESC)
                    AS th_largest_reminder,
       ROW_NUMBER() OVER (PARTITION BY r.day, r.London_Borough
                           ORDER BY row_normalise_weight) AS smallest_weight
FROM row_normalised_deman_weight_un_pivot_demand AS r
INNER JOIN (
              SELECT  day,
                      London_Borough,
                      SUM(daily_supplier_allocation) AS daily_supplier_allocation
              FROM final_step_of_DHondt_daily_supplier
              GROUP BY day, London_Borough
           ) as s ON r.day = s.day AND r.London_Borough = s.London_Borough
WHERE row_normalise_weight > 0
),
final_step_of_DHondt_supply_AS_demand  as (
SELECT *,
       CAST(CASE WHEN (n = -1) AND (smallest_weight      = 1) THEN res - 1
                 WHEN (n >  0) AND (th_largest_reminder <= n) THEN res + 1
                 ELSE res
            END AS INT64) AS allocation
FROM initial_step_of_DHondt_supply_AS_demand
ORDER BY day, London_Borough, shop
),
-- END DHondt_supply_AS_demand
```

Row	day	London_Borough	shop	row_normalise_weight	daily_supplier_allocation	fraction	res	total_res	n	th_largest_reminder	smallest_weight	allocation
1	11/06/2022	City of Westminster	shop_1	0.2	40	8.2	8	40	0	3	1	8
2	11/06/2022	City of Westminster	shop_2	0.3	40	12.3	12	40	0	2	2	12
3	11/06/2022	City of Westminster	shop_3	0.5	40	20.5	20	40	0	1	3	20
4	11/06/2022	Ealing	shop_3	1	80	81	81	81	-1	1	1	80
5	11/06/2022	Greenwich	shop_2	1	60	61	61	61	-1	1	1	60
6	11/06/2022	Hammersmith and Fulham	shop_1	0.285714286	98	28.28571	28	98	0	2	1	28
7	11/06/2022	Hammersmith and Fulham	shop_3	0.714285714	98	70.71429	70	98	0	1	2	70
8	11/06/2022	Hounslow	shop_2	0.375	112	42.375	42	112	0	2	1	42
9	11/06/2022	Hounslow	shop_3	0.625	112	70.625	70	112	0	1	2	70
10	11/06/2022	Kensington and Chelsea	shop_1	0.4	20	8.4	8	20	0	2	1	8
11	11/06/2022	Kensington and Chelsea	shop_2	0.6	20	12.6	12	20	0	1	2	12
12	11/06/2022	Richmond upon Thames	shop_1	1	40	41	41	41	-1	1	1	40
13	12/06/2022	City of Westminster	shop_1	0.2	60	12.2	12	60	0	3	1	12
14	12/06/2022	City of Westminster	shop_2	0.3	60	18.3	18	60	0	2	2	18
15	12/06/2022	City of Westminster	shop_3	0.5	60	30.5	30	60	0	1	3	30
16	12/06/2022	Ealing	shop_3	1	120	121	121	121	-1	1	1	120
17	12/06/2022	Greenwich	shop_2	1	90	91	91	91	-1	1	1	90
18	12/06/2022	Hammersmith and Fulham	shop_1	0.285714286	147	42.28571	42	147	0	2	1	42
19	12/06/2022	Hammersmith and Fulham	shop_3	0.714285714	147	105.7143	105	147	0	1	2	105
20	12/06/2022	Hounslow	shop_2	0.375	168	63.375	63	168	0	2	1	63
21	12/06/2022	Hounslow	shop_3	0.625	168	105.625	105	168	0	1	2	105
22	12/06/2022	Kensington and Chelsea	shop_1	0.4	30	12.4	12	30	0	2	1	12
23	12/06/2022	Kensington and Chelsea	shop_2	0.6	30	18.6	18	30	0	1	2	18
24	12/06/2022	Richmond upon Thames	shop_1	1	60	61	61	61	-1	1	1	60

```
available_demand AS (
SELECT
       s.shop,
       demand_quantity,
       supply_quantity,
       CASE WHEN demand_quantity < supply_quantity
                 THEN demand_quantity
                 ELSE supply_quantity
       END  AS available_demand
FROM  shop_demand as s
INNER JOIN (
              SELECT
                      shop,
                      SUM(allocation) AS supply_quantity
              FROM final_step_of_DHondt_supply_AS_demand
              GROUP BY shop

            ) AS f ON s.shop = f.shop
)
```

Row	shop	demand_quantity	supply_quantity	available_demand
1	shop_1	150	210	150
2	shop_2	500	315	315
3	shop_3	300	600	300

5. Final Allocation by DHondt

It is the final section where compute the output similar to "`Final Allocation`" (see Table 8).

```
pre_final AS
(
    SELECT
            day,
            London_Borough,
            f.shop,
            allocation * 1.0 / SUM(allocation)
                               OVER(PARTITION BY f.shop) AS column_normalised_weight,
            available_demand
    FROM  final_step_of_DHondt_supply_AS_demand AS f
    INNER JOIN available_demand  AS a on f.shop = a.shop
)
```

Row	day	London_Borough	shop	column_normalised_weight	available_demand
1	11/06/2022	City of Westminster	shop_1	0.038095238	150
2	11/06/2022	City of Westminster	shop_2	0.038095238	315
3	11/06/2022	City of Westminster	shop_3	0.033333333	300
4	11/06/2022	Ealing	shop_3	0.133333333	300
5	11/06/2022	Greenwich	shop_2	0.19047619	315
6	11/06/2022	Hammersmith and Fulham	shop_1	0.133333333	150
7	11/06/2022	Hammersmith and Fulham	shop_3	0.116666667	300
8	11/06/2022	Hounslow	shop_2	0.133333333	315
9	11/06/2022	Hounslow	shop_3	0.116666667	300
10	11/06/2022	Kensington and Chelsea	shop_1	0.038095238	150
11	11/06/2022	Kensington and Chelsea	shop_2	0.038095238	315
12	11/06/2022	Richmond upon Thames	shop_1	0.19047619	150
13	12/06/2022	City of Westminster	shop_1	0.057142857	150
14	12/06/2022	City of Westminster	shop_2	0.057142857	315
15	12/06/2022	City of Westminster	shop_3	0.05	300
16	12/06/2022	Ealing	shop_3	0.2	300
17	12/06/2022	Greenwich	shop_2	0.285714286	315
18	12/06/2022	Hammersmith and Fulham	shop_1	0.2	150
19	12/06/2022	Hammersmith and Fulham	shop_3	0.175	300
20	12/06/2022	Hounslow	shop_2	0.2	315
21	12/06/2022	Hounslow	shop_3	0.175	300
22	12/06/2022	Kensington and Chelsea	shop_1	0.057142857	150
23	12/06/2022	Kensington and Chelsea	shop_2	0.057142857	315
24	12/06/2022	Richmond upon Thames	shop_1	0.285714286	150

```
-- START DHondt_final
initial_step_of_DHondt_final AS (
SELECT
       day,
       London_Borough,
       shop,
       column_normalised_weight,
       available_demand,

       column_normalised_weight * (available_demand + 1)           AS fraction,
       FLOOR( column_normalised_weight * (available_demand + 1)  ) AS res,

       SUM( FLOOR( column_normalised_weight * (available_demand + 1)) )
            OVER (PARTITION BY shop) AS total_res,
       available_demand -
            SUM( FLOOR( column_normalised_weight * (available_demand + 1)) )
            OVER (PARTITION BY shop)   AS n,
       ROW_NUMBER() OVER (
                    PARTITION BY shop
                    ORDER BY ( column_normalised_weight * (available_demand + 1) -
                    FLOOR(  column_normalised_weight * (available_demand + 1)  )) DESC)
                    AS th_largest_reminder,
       ROW_NUMBER() OVER (
                    PARTITION BY shop
                    ORDER BY column_normalised_weight)
                    AS smallest_weight
FROM pre_final
),
final_step_of_DHondt_final  as (
SELECT *,
       CAST(CASE WHEN (n = -1) AND (smallest_weight      = 1) THEN res - 1
                 WHEN (n >  0) AND (th_largest_reminder <= n) THEN res + 1
                 ELSE res
            END AS INT64) AS allocation
FROM initial_step_of_DHondt_final
),
-- END DHondt_final
```

Row	day	London_Borough	shop	column_normalised_weight	available_demand	fraction	res	total_res	n	th_largest_reminder	smallest_weight	allocation
1	11/06/2022	nsington and Chels	shop_1	0.038095238	150	5.752381	5	147	3	2	1	6
2	11/06/2022	ity of Westminste	shop_1	0.038095238	150	5.752381	5	147	3	3	2	6
3	12/06/2022	ity of Westminste	shop_1	0.057142857	150	8.628571	8	147	3	4	3	8
4	12/06/2022	nsington and Chels	shop_1	0.057142857	150	8.628571	8	147	3	5	4	8
5	11/06/2022	mersmith and Ful	shop_1	0.133333333	150	20.13333	20	147	3	8	5	20
6	11/06/2022	hmond upon Than	shop_1	0.19047619	150	28.7619	28	147	3	1	6	29
7	12/06/2022	mersmith and Ful	shop_1	0.2	150	30.2	30	147	3	6	7	30
8	12/06/2022	hmond upon Than	shop_1	0.285714286	150	43.14286	43	147	3	7	8	43
9	11/06/2022	nsington and Chels	shop_2	0.038095238	315	12.0381	12	315	0	7	1	12
10	11/06/2022	ity of Westminste	shop_2	0.038095238	315	12.0381	12	315	0	8	2	12
11	12/06/2022	ity of Westminste	shop_2	0.057142857	315	18.05714	18	315	0	5	3	18
12	12/06/2022	nsington and Chels	shop_2	0.057142857	315	18.05714	18	315	0	6	4	18
13	11/06/2022	Hounslow	shop_2	0.133333333	315	42.13333	42	315	0	4	5	42
14	11/06/2022	Greenwich	shop_2	0.19047619	315	60.19048	60	315	0	3	6	60
15	12/06/2022	Hounslow	shop_2	0.2	315	63.2	63	315	0	2	7	63
16	12/06/2022	Greenwich	shop_2	0.285714286	315	90.28571	90	315	0	1	8	90
17	11/06/2022	ity of Westminste	shop_3	0.033333333	300	10.03333	10	299	1	8	1	10
18	12/06/2022	ity of Westminste	shop_3	0.05	300	15.05	15	299	1	7	2	15
19	11/06/2022	mersmith and Ful	shop_3	0.116666667	300	35.11667	35	299	1	5	3	35
20	11/06/2022	Hounslow	shop_3	0.116666667	300	35.11667	35	299	1	6	4	35
21	11/06/2022	Ealing	shop_3	0.133333333	300	40.13333	40	299	1	4	5	40
22	12/06/2022	mersmith and Ful	shop_3	0.175	300	52.675	52	299	1	1	6	53
23	12/06/2022	Hounslow	shop_3	0.175	300	52.675	52	299	1	2	7	52
24	12/06/2022	Ealing	shop_3	0.2	300	60.2	60	299	1	3	8	60

```
pivot_final AS(
SELECT day,
       London_Borough,
       IFNULL(CAST(allocation_shop_1 AS INT64), 0)  AS allocation_shop_1,
       IFNULL(CAST(allocation_shop_2 AS INT64), 0) AS allocation_shop_2,
       IFNULL(CAST(allocation_shop_3 AS INT64), 0) AS allocation_shop_3
FROM
(
  -- #1 from_item
  SELECT
    day,
    London_Borough,
    shop,
    CAST(allocation AS INT64) AS allocation
  FROM final_step_of_DHondt_final
)
PIVOT
(
  -- #2 aggregate
  AVG(allocation) AS allocation
  -- #3 pivot_column
  FOR shop in ('shop_1', 'shop_2', 'shop_3')
)
ORDER BY day, London_Borough
)
```

Row	day	London_Borough	allocation_shop_1	allocation_shop_2	allocation_shop_3
1	11/06/2022	City of Westminster	6	12	10
2	11/06/2022	Ealing	0	0	40
3	11/06/2022	Greenwich	0	60	0
4	11/06/2022	Hammersmith and Fulham	20	0	35
5	11/06/2022	Hounslow	0	42	35
6	11/06/2022	Kensington and Chelsea	6	12	0
7	11/06/2022	Richmond upon Thames	29	0	0
8	12/06/2022	City of Westminster	8	18	15
9	12/06/2022	Ealing	0	0	60
10	12/06/2022	Greenwich	0	90	0
11	12/06/2022	Hammersmith and Fulham	30	0	53
12	12/06/2022	Hounslow	0	63	52
13	12/06/2022	Kensington and Chelsea	8	18	0
14	12/06/2022	Richmond upon Thames	43	0	0

6 Summary

Below we put all above scripts in one place. You should be able to get the final result if you run the following script:

```
WITH
demand_weight AS (
      SELECT  'shop_1' AS shop, 0.2 AS demand_weight UNION ALL
      SELECT  'shop_2',         0.3                  UNION ALL
      SELECT  'shop_3',         0.5
),
shop_demand   AS(
      SELECT  'shop_1' AS shop, 150 AS demand_quantity UNION ALL
      SELECT  'shop_2',         500                    UNION ALL
      SELECT  'shop_3',         300
),
demand AS (
      SELECT '11/06/2022' AS day, 'Camden'                 AS London_Borough,
             False AS Shop_1,  False AS Shop_2,  False  AS Shop_3 UNION ALL
      SELECT '11/06/2022' AS day, 'Ealing'                 AS London_Borough,
             False AS Shop_1,  False AS Shop_2,  True   AS Shop_3 UNION ALL
      SELECT '11/06/2022' AS day, 'Greenwich'              AS London_Borough,
             False AS Shop_1,  True  AS Shop_2,  False  AS Shop_3 UNION ALL
      SELECT '11/06/2022' AS day, 'Hounslow'               AS London_Borough,
             False AS Shop_1,  True  AS Shop_2,  True   AS Shop_3 UNION ALL
      SELECT '11/06/2022' AS day, 'Richmond upon Thames'   AS London_Borough,
             True  AS Shop_1,  False AS Shop_2,  False  AS Shop_3 UNION ALL
```

```
      SELECT '11/06/2022' AS day, 'Hammersmith and Fulham' AS London_Borough,
             True  AS Shop_1,  False AS Shop_2,  True   AS Shop_3 UNION ALL
      SELECT '11/06/2022' AS day, 'Kensington and Chelsea' AS London_Borough,
             True  AS Shop_1,  True  AS Shop_2,  False  AS Shop_3 UNION ALL
      SELECT '11/06/2022' AS day, 'City of Westminster'    AS London_Borough,
             True  AS Shop_1,  True  AS Shop_2,  True   AS Shop_3 UNION ALL
      SELECT '12/06/2022' AS day, 'Camden'                 AS London_Borough,
             False AS Shop_1,  False AS Shop_2,  False  AS Shop_3 UNION ALL
      SELECT '12/06/2022' AS day, 'Ealing'                 AS London_Borough,
             False AS Shop_1,  False AS Shop_2,  True   AS Shop_3 UNION ALL
      SELECT '12/06/2022' AS day, 'Greenwich'              AS London_Borough,
             False AS Shop_1,  True  AS Shop_2,  False  AS Shop_3 UNION ALL
      SELECT '12/06/2022' AS day, 'Hounslow'               AS London_Borough,
             False AS Shop_1,  True  AS Shop_2,  True   AS Shop_3 UNION ALL
      SELECT '12/06/2022' AS day, 'Richmond upon Thames'   AS London_Borough,
             True  AS Shop_1,  False AS Shop_2,  False  AS Shop_3 UNION ALL
      SELECT '12/06/2022' AS day, 'Hammersmith and Fulham' AS London_Borough,
             True  AS Shop_1,  False AS Shop_2,  True   AS Shop_3 UNION ALL
      SELECT '12/06/2022' AS day, 'Kensington and Chelsea' AS London_Borough,
             True  AS Shop_1,  True  AS Shop_2,  False  AS Shop_3 UNION ALL
      SELECT '12/06/2022' AS day, 'City of Westminster'    AS London_Borough,
             True  AS Shop_1,  True  AS Shop_2,  True   AS Shop_3
),
```

```
-- SUPPLY
prob_of_supply_per_borough AS (
SELECT 'Camden'                AS London_Borough, 0.118 AS supplier_1,
                                                  0.118 AS supplier_2 UNION ALL
SELECT 'Ealing'                AS London_Borough, 0.157 AS supplier_1,
                                                  0.157 AS supplier_2 UNION ALL
SELECT 'Greenwich'             AS London_Borough, 0.118 AS supplier_1,
                                                  0.118 AS supplier_2 UNION ALL
SELECT 'Hounslow'              AS London_Borough, 0.220 AS supplier_1,
                                                  0.220 AS supplier_2 UNION ALL
SELECT 'Richmond upon Thames'  AS London_Borough, 0.078 AS supplier_1,
                                                  0.078 AS supplier_2 UNION ALL
SELECT 'Hammersmith and Fulham' AS London_Borough, 0.192 AS supplier_1,
                                                  0.192 AS supplier_2 UNION ALL
SELECT 'Kensington and Chelsea' AS London_Borough, 0.039 AS supplier_1,
                                                  0.039 AS supplier_2 UNION ALL
SELECT 'City of Westminster'   AS London_Borough, 0.078 AS supplier_1,
                                                  0.078 AS supplier_2
),
daily_total_supplied_quantity AS(
SELECT '11/06/2022' AS day, 255 AS supplier_1, 255 AS supplier_2 UNION ALL
SELECT '12/06/2022' AS day, 510 AS supplier_1, 255 AS supplier_2
),
un_pivot_demand AS (
SELECT day, London_Borough, shop, indicator FROM demand
UNPIVOT(indicator FOR shop IN (shop_1, shop_2, shop_3))
```

```
),
deman_weight_un_pivot_demand AS (
SELECT day,
       London_Borough,
       u.shop,
       indicator,
       demand_weight,
       CASE WHEN indicator THEN demand_weight ELSE 0 END AS indicator_weight,

FROM un_pivot_demand AS u
INNER JOIN demand_weight AS d ON d.shop=u.shop
),
row_normalised_deman_weight_un_pivot_demand AS (
 SELECT
       day,
       London_Borough,
       shop,
       indicator,
       demand_weight,
       indicator_weight,
       CASE WHEN SUM(indicator_weight) OVER(PARTITION BY day, London_Borough) > 0
                 THEN indicator_weight / SUM(indicator_weight)
                                         OVER(PARTITION BY day, London_Borough)
                 ELSE 0
       END AS row_normalise_weight
FROM deman_weight_un_pivot_demand
WHERE indicator_weight > 0
),
un_pivot_prob_of_supply_per_borough AS (
SELECT London_Borough, supplier, probability FROM prob_of_supply_per_borough
UNPIVOT(probability FOR supplier IN (supplier_1, supplier_2))
),
un_pivot_daily_total_supplied_quantity AS (
SELECT day, supplier, quantity FROM daily_total_supplied_quantity
UNPIVOT(quantity FOR supplier IN (supplier_1, supplier_2))
),
-- START DHondt_daily_supplier
initial_step_of_DHondt_daily_supplier AS
(
SELECT day,
       London_Borough,
       d.supplier,
       probability,
       quantity,
       probability * (quantity + 1)        AS fraction,
       FLOOR(probability * (quantity + 1)) AS res,

       SUM( FLOOR(probability * (quantity + 1)) ) OVER (PARTITION BY day, d.supplier)
            AS total_res,
       quantity -
            SUM( FLOOR(probability * (quantity + 1)) )
            OVER (PARTITION BY day, d.supplier)  AS n,
       ROW_NUMBER() OVER (
                    PARTITION BY day, d.supplier
```

```
                    ORDER BY (       probability * (quantity + 1) -
                                FLOOR(probability * (quantity + 1)) ) DESC)
                    AS th_largest_reminder,
       ROW_NUMBER() OVER (
                    PARTITION BY day,
                    d.supplier ORDER BY probability)
                    AS smallest_weight
FROM un_pivot_daily_total_supplied_quantity AS d
INNER JOIN un_pivot_prob_of_supply_per_borough AS p ON d.supplier = p.supplier
),
final_step_of_DHondt_daily_supplier as (
SELECT *,
       CAST(CASE WHEN (n = -1) AND (smallest_weight      = 1) THEN res - 1
                 WHEN (n >  0) AND (th_largest_reminder <= n) THEN res + 1
                 ELSE res
            END AS INT64) AS daily_supplier_allocation
FROM initial_step_of_DHondt_daily_supplier
),
-- END   DHondt_daily_supplier
-- START DHondt_supply_AS_demand
initial_step_of_DHondt_supply_AS_demand AS (
SELECT
       r.day,
       r.London_Borough,
       shop,
       row_normalise_weight,
       daily_supplier_allocation,


       row_normalise_weight * (daily_supplier_allocation + 1)        AS fraction,
       FLOOR(row_normalise_weight * (daily_supplier_allocation + 1)) AS res,

       SUM( FLOOR(row_normalise_weight * (daily_supplier_allocation + 1)) )
            OVER (PARTITION BY r.day, r.London_Borough) AS total_res,
       daily_supplier_allocation  -
            SUM( FLOOR(row_normalise_weight * (daily_supplier_allocation + 1)) )
                 OVER (PARTITION BY r.day, r.London_Borough)  AS n,
       ROW_NUMBER() OVER (
                    PARTITION BY r.day, r.London_Borough
                    ORDER BY ( row_normalise_weight * (daily_supplier_allocation + 1) -
                    FLOOR(row_normalise_weight * (daily_supplier_allocation + 1)) ) DESC)
                    AS th_largest_reminder,
       ROW_NUMBER() OVER (PARTITION BY r.day, r.London_Borough
                            ORDER BY row_normalise_weight) AS smallest_weight
FROM row_normalised_deman_weight_un_pivot_demand AS r
INNER JOIN (
              SELECT  day,
                      London_Borough,
                      SUM(daily_supplier_allocation) AS daily_supplier_allocation
              FROM final_step_of_DHondt_daily_supplier
              GROUP BY day, London_Borough
            ) as s ON r.day = s.day AND r.London_Borough = s.London_Borough
WHERE row_normalise_weight > 0
),
```

```
final_step_of_DHondt_supply_AS_demand  as (
SELECT *,
       CAST(CASE WHEN (n = -1) AND (smallest_weight      = 1) THEN res - 1
                 WHEN (n >  0) AND (th_largest_reminder <= n) THEN res + 1
                 ELSE res
            END AS INT64) AS allocation
FROM initial_step_of_DHondt_supply_AS_demand
ORDER BY day, London_Borough, shop
),
-- END DHondt_supply_AS_demand
available_demand AS (
SELECT
       s.shop,
       demand_quantity,
       supply_quantity,
       CASE WHEN demand_quantity < supply_quantity
                 THEN demand_quantity
                 ELSE supply_quantity
       END  AS available_demand
FROM  shop_demand as s
INNER JOIN (
              SELECT
                      shop,
                      SUM(allocation) AS supply_quantity
              FROM final_step_of_DHondt_supply_AS_demand
              GROUP BY shop

           ) AS f ON s.shop = f.shop
),
pre_final AS
(
    SELECT
            day,
            London_Borough,
            f.shop,
            allocation * 1.0 / SUM(allocation) OVER(PARTITION BY f.shop)
                      AS column_normalised_weight,
            available_demand
    FROM  final_step_of_DHondt_supply_AS_demand AS f
    INNER JOIN available_demand  AS a on f.shop = a.shop
),
-- START DHondt_final
initial_step_of_DHondt_final AS (
SELECT
       day,
       London_Borough,
       shop,
       column_normalised_weight,
       available_demand,

       column_normalised_weight * (available_demand + 1)           AS fraction,
       FLOOR( column_normalised_weight * (available_demand + 1)  ) AS res,

       SUM( FLOOR( column_normalised_weight * (available_demand + 1)) )
```

```
            OVER (PARTITION BY shop) AS total_res,
       available_demand -
            SUM( FLOOR( column_normalised_weight * (available_demand + 1)) )
            OVER (PARTITION BY shop)   AS n,
       ROW_NUMBER() OVER (
                    PARTITION BY shop
                    ORDER BY ( column_normalised_weight * (available_demand + 1) -
                    FLOOR(  column_normalised_weight * (available_demand + 1)  )) DESC)
                    AS th_largest_reminder,
       ROW_NUMBER() OVER (
                    PARTITION BY shop
                    ORDER BY column_normalised_weight)
                    AS smallest_weight
FROM pre_final
),
final_step_of_DHondt_final  as (
SELECT *,
       CAST(CASE WHEN (n = -1) AND (smallest_weight      = 1) THEN res - 1
                 WHEN (n >  0) AND (th_largest_reminder <= n) THEN res + 1
                 ELSE res
            END AS INT64) AS allocation
FROM initial_step_of_DHondt_final
),
-- END DHondt_final
pivot_final AS(
SELECT day,
       London_Borough,
       IFNULL(CAST(allocation_shop_1 AS INT64), 0)  AS allocation_shop_1,
       IFNULL(CAST(allocation_shop_2 AS INT64), 0) AS allocation_shop_2,
       IFNULL(CAST(allocation_shop_3 AS INT64), 0) AS allocation_shop_3
FROM
(
  -- #1 from_item
  SELECT
    day,
    London_Borough,
    shop,
    CAST(allocation AS INT64) AS allocation
  FROM final_step_of_DHondt_final
)
PIVOT
(
  -- #2 aggregate
  AVG(allocation) AS allocation
  -- #3 pivot_column
  FOR shop in ('shop_1', 'shop_2', 'shop_3')
)
ORDER BY day, London_Borough
)

select * from pivot_final
```