Q1.

(1). As the table shows below:

Location	Time	Item	SUM(Quantity)
Sydney	2005	PS2	1400
Sydney	2005	ALL	1400
Sydney	2006	PS2	1500
Sydney	2006	Wii	500
Sydney	2006	ALL	2000
Sydney	ALL	PS2	2900
Sydney	ALL	Wii	500
Sydney	ALL	ALL	3400
Melbourne	2005	XBox 360	1700
Melbourne	2005	ALL	1700
Melbourne	ALL	XBox 360	1700
Melbourne	ALL	ALL	1700
ALL	2005	PS2	1400
ALL	2005	XBox 360	1700
ALL	2005	ALL	3100
ALL	2006	PS2	1500
ALL	2006	Wii	500
ALL	2006	ALL	2000
ALL	ALL	PS2	2900
ALL	ALL	Wii	500
ALL	ALL	XBox 360	1700
ALL	ALL	ALL	5100

(2).

SELECT Location, Time, Item, SUM(Quantity)

FROM Sales

GROUP BY Location, Time, Item

UNION ALL

SELECT Location, Time, ALL, SUM(Quantity)

FROM Sales

GROUP BY Location, Time

UNION ALL

SELECT Location, ALL, Item, SUM(Quantity)

FROM Sales

GROUP BY Location, Item

UNION ALL

SELECT Location, ALL, ALL, SUM(Quantity)

FROM Sales

GROUP BY Location

UNION ALL

SELECT ALL, Time, Item, SUM(Quantity)

FROM Sales

GROUP BY Time, Item

UNION ALL

SELECT ALL, Time, ALL, SUM(Quantity)

FROM Sales

GROUP BY Time

UNION ALL

SELECT ALL, ALL, Item, SUM(Quantity)

FROM Sales

GROUP BY Item

UNION ALL

SELECT ALL, ALL, ALL, SUM(Quantity)

FROM Sales

(3). The *ice-berg cube* shows below:

Location	Time	Item	SUM(Quantity)
Sydney	ALL	PS2	2900
Sydney	2006	ALL	2000
Sydney	ALL	ALL	3400
ALL	ALL	PS2	2900
ALL	2005	ALL	3100
ALL	2006	ALL	2000
ALL	ALL	ALL	5100

(4).

Denote Location: L, Time: T, Item: I

The function is $h(L,T,I) = 12*L + 4*T + I$

The sparse multi-dimensional array shows below:

ArrayIndex	Value
17	1400
16	1400
21	1500
23	500
20	2000
13	2900
15	500
12	3400
30	1700
28	1700
26	1700
24	1700
5	1400
6	1700
4	3100
9	1500
11	500

8	2000
1	2900
3	500
2	1700
0	5100

Q2.

Step1:

	P1	P2	P3	P4	P5
P1	1.00	0.10	0.41	0.55	0.35
P2	0.10	1.00	0.64	0.47	0.98
P3	0.41	0.64	1.00	0.44	0.85
P4	0.55	0.47	0.44	1.00	0.76
P5	0.35	0.98	0.85	0.76	1.00

Cluster P2 and Cluster P5 have the max similarity.

Step2:

	P1	P2&P5	P3	P4
P1	1.00	0.477	0.41	0.55
P2&P5		1.00	0.823	0.737
P3			1.00	0.44
P4				1.00

Sim(25, 1) = 2 * (0.98 + 0.10 + 0.35) / 6 = 0.4766666

Sim(25, 3) = 2 * (0.98 + 0.64 + 0.85) / 6 = 0.8233333

Sim(25, 4) = 2 * (0.98 + 0.47 + 0.76) / 6 = 0.7366666

Cluster P2&P5 and Cluster P3 have the max similarity.

Step3:

	P1	P2&P3&P5	P4
P1	1.00	0.555	0.55
P2&P3&P5		1.00	0.69

P4				1.00

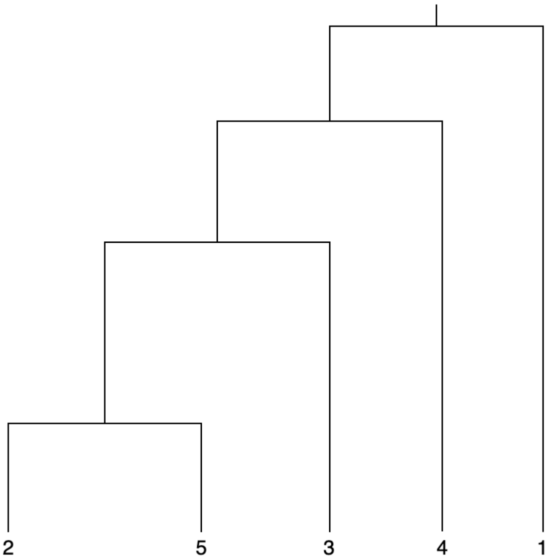
Sim(235, 1) = 2 * (0.64 + 0.98 + 0.10 + 0.85 + 0.41 + 0.35) / 12 = 0.555

Sim(235, 4) = 2 * (0.64 + 0.98 + 0.47 + 0.85 + 0.44 + 0.76) / 12 = 0.69

Cluster P2&P3&P5 and Cluster P4 have the max similarity.

Step4:

So, as the three steps showed before, the final result showed as dendrogram is:



Q3.

(1).

Lines 8- 9 and lines 12 - 14 are new added.

```
1.  Initialize k centers C = [c1, c2, . . . , ck];
2.  canStop ← false;
3.  while canStop = false do
4.      Initialize k empty clusters G = [g1, g2, . . . , gk];
5.      for each data point p ∈ D do
6.          cx ← NearestCenter(p, C);
7.          gcx .append(p);
8.      previous_C ← C;
9.      C ← [];
10.     for each group g ∈ G do
11.         ci ← ComputeCenter(g);
12.         C.append(ci);
13.     if previous_C == C do
14.         canStop ← True;
15. return G;
```

(2).

Conclusion 1:

For the *cost(g_i),* we compute the derivative of c_i.

Denote S is the total number in one cluster.

$$\frac{\partial Cost(g_i)}{\partial c_i} = \sum_{p \in g_i} 2(p - c_i) = 0$$

$$\sum_{p \in g_i} p = S \cdot c_i$$

$$c_i = \frac{1}{S} \sum_{p \in g_i} p$$

Then we got when c_i is the mean of all points in same cluster, the distance is the minimum.

Conclusion 2:

And apparently when centers are fixed, for each point find the nearest centers can minimise the total cost.

So we can analyse the cost from the beginning of the pseudo-code:

We denote the beginning cost is C0, after line 5- 7 the cost is denoted as C1, C2 is denoted at the end of iteration(i.e. Line 8- 9).

It's not difficult to find that C1 is smaller than C0 as we find nearest centre for each point. (Conclusion 1).

And C2 is smaller than C1 because we re-compute the centres by using the mean of all points in one cluster which we found on stage Line 5 - 7.(Conclusion 2)

In conclusion, C2 < C1 < C0, therefore, the cost of k clusters never increases at each iteration.

(3).

The conclusion of the second question is that the total cost of clustering will never increases.

The total possible clusters are **k^n / k! ,** k is the number of clusters and n is the number of points, this is finite which means the loop is finite and the cost is keep decreasing, so K-Means will always converge.

In addition, because of the randomly K and initial centers, we are not guarantee this algorithm will converge at the global minima, but the algorithm will always converge at local minima.