Problem 1

Assume d1, d2, d3, ... dn are instances of DrivingDistance

$(\pi_{d1.city1,\ d1.city2}(\sigma_{d1.city1 = \text{'Palo Alto'} \wedge d1.city2 = \text{'Washington DC'}}(d1))\ )$

$\cup$

$(\pi_{d1.city1,\ d2.city2}(\sigma_{d1.city1 = \text{'Palo Alto'} \wedge d2.city2 = \text{'Washington DC'}}(d1 \bowtie_{d1.city2 = d2.city1} d2)))$

$\cup$

$(\pi_{d1.city1,\ d3.city2}(\sigma_{d1.city1 = \text{'Palo Alto'} \wedge d3.city2 = \text{'Washington DC'}}((d1 \bowtie_{d1.city2 = d2.city1} d2) \bowtie_{d2.city2 = d3.city1} d3)))$

$\cup$

...

$\cup$

$(\pi_{d1.city1,\ dn.city2}(\sigma_{d1.city1 = \text{'Palo Alto'} \wedge dn.city2 = \text{'Washington DC'}}((d1 \bowtie_{d1.city2 = d2.city1} d2) \bowtie_{d2.city2 = d3.city1} d3)\ \ldots$

$\bowtie_{dn-1.city2 = dn.city1} dn))$

/*

In this question, I get the idea from the SQL Problem 3 where I first see if the two city can be drive directly, if not, I check if I can have only one intermediate city, if not, I check if two intermediate city is enough until I check all n cities, if I still get no tuples then the cities is unreachable

*/

Problem 2

Answer : $\pi_{city1, city2}(\sigma_{d1.city1 = 'Palo Alto'}(DrivingDistance))$

R: $\pi_{city1, city2}(DrivingDistance - Answer)$

While NotEmpty(R) Do

{

Temp: $\pi_{Answer.city1, R.city1, R.city2}$ $(Answer \bowtie_{Answer.city2 = R.city1} R)$

Answer: $Answer \cup \pi_{Answer.city1, R.city2}(Temp)$

R: $R - \pi_{R.city1, R.city2}(Temp)$

}

Answer: $\sigma_{d2.city2 = 'Washington DC'}$ Answer

/*

In this problem, I use two set, Answer which contains the temp answer, and R which contains the tuples left. And a temporary set Temp which store the tuples to be delete.

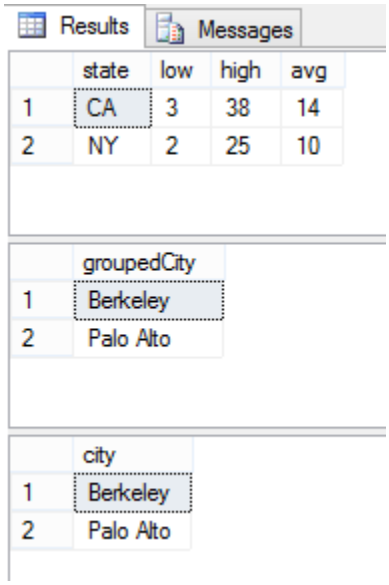
I first initialized Answer to be all the tuples with city1 = “Palo Alto” and removes those tuples from R. Then, for each iteration, I join the Answer’s city 2 with city 1 in R, and union with Answer. Then, I remove those joined tuples from R so that R will have fewer tuples until all tuples has been removed, the loop ends

Finally, I select city2 = “DC” to get the final answer

*/

SQL Result

Problem 1:



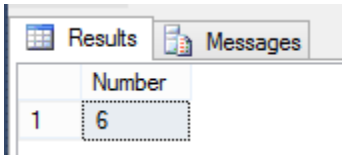
Results | Messages

	state	low	high	avg
1	CA	3	38	14
2	NY	2	25	10

	groupedCity
1	Berkeley
2	Palo Alto

	city
1	Berkeley
2	Palo Alto

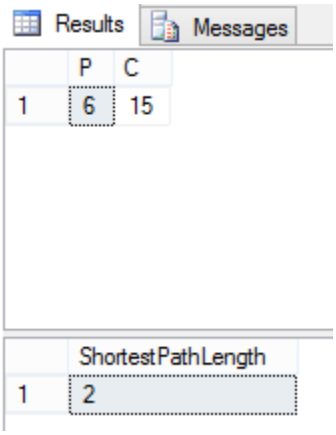
Problem 2:



Results | Messages

	Number
1	6

Problem 3:



Results | Messages

	P	C
1	6	15

	ShortestPathLength
1	2