# Similarity Matrix for Collaborative Filtering

## Problems:

1. Similarity Matrix is too large to fit in the memory.
   1. Use key value pairs like ((position1, position2), similarity) to calculate the final result.
2. If I use key value pairs, how can I find ?
   1. Use the idea of combination. I will show it in detail in step 3).

## Steps:

1. Calculate the mean rate of every movie and using mean\_ri.
   1. The form of original data is a tuple, (user ID, movie ID, rate, time).
   2. If I want to calculate means, we do not need user ID and time, so that we map it to (movie ID, (rate, 1)).
   3. Then, I reduce the data by key, and I get (movie ID, (total rate, times))
   4. Finally, I get (movie ID, mean)
2. In the formula, I only need to calculate , so I first calculate every using minus\_mean.
   1. I can get minus\_mean (user ID, movie ID, rate –mean rate).
3. Find and calculate and without using matrix.
   1. Basically, means that a user, say x, both rates the movie i and movie j. Therefore, when we calculate , we need to find all rates that user both rates the movie i and movie j, but we do not need to know who is this user.
   2. In detail, I can map my minus\_mean (user ID, movie ID, rate–mean rate) to (user ID, (movie ID, rate–mean rate)) and group it by key.
   3. I get (user ID, a list of [(movie ID, rate–mean rate), ……]).
   4. Sort the list by movie ID, cause movie ID must be in ascending order.
      1. Reason: for example, the combination of [1, 2, 3] are (1, 2), (1, 3), (2, 3).

I do not want that I have an element (i, j) that i > j.

It is very easy to find (i, j), if i < j.

* 1. Discard user IDs. I get a list of [(movie ID, rate–mean rate), ……].

Use combination to combine the list to get ((movie ID\_1, movie ID\_2), (rate\_1, rate\_2)) (movie ID\_1 and movie ID\_2 are rated by one user)

1. The I can get the Similarity Matrix using the formula