**SQL Shit 4**

x = column(s)

a = decimal between 0 and 1

* CUME\_DIST() OVER (ORDER BY x):
* This function calculates the cumulative distribution of a value within a group of values
* In other words, CUME\_DIST calculates the relative position of a specified value in a group of values.
* Assuming ascending order, the CUME\_DIST of a value in row *r* is defined as the number of rows with values less than or equal to that value in row *r,* divided by the number of rows evaluated in the partition or query result set.
* This function is similar to the PERCENT\_RANK function.
* The PARTITION BY (can put before order, after over) clause divides the FROM clause result into partitions, to which the function is applied. If the PARTITION BY isn’t specified, CUME\_DIST treats all query result set rows as a single group.
* The ORDER BY clause determines the logical order in which the operation occurs. CUME\_DIST requires the ORDER BY clause.
* CUM\_DIST won’t accept the ‘rows or range’ clause of the OVER syntax.
* Returns a range of values >0 and <= 1.
* CUME\_DIST includes NULL values by default and treats these values as the lowest possible values.
* Nondeterministic.
* PERCENT\_RANK() OVER (ORDER BY x):
* Calculates the relative rank of a row within a group of rows in SQL Server.
* Use PERCENT\_RANK to evaluate the relative standing of a value within a query result set or partition.
* Similar to CUME\_DIST.
* Again, can use PARTITION BY to divide results set produced by FROM clause. If not specified, the function treats all rows of the query results set as a single group.
* ORDER BY is required.
* The ‘rows or range’ clause of the OVER syntax cannot be specified
* Returns a range of values >0 and <=1 .
* NULL values are included by default and are treated as the lowest possible values.
* Nondeterministic
* PERCENTILE\_DISC(a) WITHIN GROUP (ORDER BY x) OVER():