```
Add jar /usr/lib/hive-hcatalog/share/hcatalog/hive-hcatalog-core-2.3.6-amzn-2.jar;
create database if not exists amz_review;
Show databases;
use amz_review;
create external table amz_review.amz_review_dump (json_dump string)
location 's3a://mybucketashu1/tables/';
select * from amz_review.amz_review_dump limit 10;

create external table amz_review.amz_review_col (
    reviewerid string,
    asin string,
    reviewername string,
    helpful array<int>,
    reviewtext string,
    overall double,
    summary string,
    unixreviewtime bigint)
    row format serde 'org.apache.hive.hcatalog.data.JsonSerDe'
    with serdeproperties ('paths'= '')
    location 's3a://mybucketashu1/tables/';

    Select * from amz_review.amz_review_col limit 5;


    Select count(*) from amz_review_col;
     Select count(DISTINCT asin) am_products from amz_review_col;

     /*How many reviews are posted on a yearly basis? \*/
     Select rate_year , count(1) cnt from (
Select year(from_unixtime(unixreviewtime)) rate_year from amz_review_col )T group
by rate_year order by rate_year desc;

 /*// which are the most popular products */
select asin amz_product,count(1) r_cnt from amz_review_col group by
asin order by r_cnt desc limit 10;

 /*// For a maximum helpful review */
Select product , sum(help_rate) as help_rt from (
Select asin product , CASE WHEN helpful[0]=0 then 0.00 else
ROUND(helpful[0]/helpful[1],2) end as help_rate FROM amz_review_col)T
group by product order by help_rt desc limit 10;


Select product , sum(help_rate) as help_rt from (
Select asin product , CASE WHEN helpful[0]=0 then 0.00 else
ROUND(helpful[0]/helpful[1]*overall,2) end as help_rate FROM amz_review_col)T
group by product order by help_rt desc limit 10;
```

```
 /*// product B00DR0PDNE is definately a popular product as it comes in all 3
analysis */

 /* */
  /*For a minimum helpful review */
Select product , sum(help_rate) as help_rt from (
Select asin product , CASE WHEN helpful[0]=0 then 0.00 else
ROUND(helpful[0]/helpful[1],2) end as help_rate FROM amz_review_col)T
group by product order by help_rt asc limit 10;


 /* Product with minimum review  */
 Select amz_product , min(review_cnt) min_r_cnt from (
Select asin amz_product,count(1) review_cnt from amz_review_col group by asin
)T group by amz_product order by min_r_cnt asc limit 1;


 /* Product with maximum  review  */
Select amz_product , max(review_cnt) max_r_cnt from (
Select asin amz_product,count(1) review_cnt from amz_review_col group
by asin)T group by amz_product order by max_r_cnt desc limit 1;

/* For average review */
Select amz_product , avg(review_cnt) max_r_cnt from (
Select asin amz_product,count(1) review_cnt from amz_review_col group by asin
)T group by amz_product;



/* Now lets Partitioning and Bucketing the data */

set hive.exec.dynamic.partition=true;
set hive.exec.dynamic.partition.mode=nonstrict;
set hive.exec.max.dynamic.partitions=1000;
set hive.exec.max.dynamic.partitions.pernode=1000;

create external table if not exists amz_review_yr_mnth_part (
   reviewerid string,
   asin string,
   reviewername string,
   helpful array<int>,
   reviewtext string,
   overall double,
   summary string,
   unixreviewtime bigint) partitioned by
(yr int, mnth int)
location 's3a://mybucketashu1/tables/';


insert overwrite table amz_review_yr_mnth_part partition(yr, mnth)
```

```
select    reviewerid,
          asin,
          reviewername,
          helpful,
          reviewtext,
          overall,
          summary,
          unixreviewtime,
          year(from_unixtime(unixreviewtime)) as yr,
          month(from_unixtime(unixreviewtime)) as mnth
from      amz_review_col;

/* checking the time taken by both queries */
select  overall, count(*) as review_count from amz_review_yr_mnth_part
where  yr = 2004 and mnth = 1  group by overall   order by review_count desc;
/* here time taken 4 sec */

select  overall,  count(*) as review_count from amz_review_col
where year(from_unixtime(unixreviewtime)) = 2004 and
month(from_unixtime(unixreviewtime)) = 1
 group by overall order by review_count desc;
 /* here time taken is 115.881 */
 /* lets rank product */
 Select asin as Product , RANK() OVER(order by overall)  from
amz_review_yr_mnth_part;

 /* lets use dense rank */
 Select asin as Product , DENSE_RANK() OVER(order by overall)  from
amz_review_yr_mnth_part ;

 /* lets start bucketing */

 set hive.exec.dynamic.partition.mode=nonstrict;
set hive.exec.dynamic.partition=true;
set hive.enforce.bucketing=true;
set hive.exec.max.dynamic.partitions=1000;
set hive.exec.max.dynamic.partitions.pernode=1000;

create external table if not exists
amz_review_clustered_yr_mnth
(reviewerid string, asin string, reviewername string, helpful
array<int>, reviewtext string,
overall double, summary string, unixreviewtime bigint) partitioned by
(yr int, mnth int)
clustered by (reviewerid) into 4 buckets
location 's3a://mybucketashu1/tables/';



insert overwrite table
```

```
amz_review_clustered_yr_mnth partition(yr,
mnth)
select reviewerid, asin, reviewername, helpful, reviewtext,
overall, summary, unixreviewtime, yr, mnth
from amz_review_yr_mnth_part
```