# **STORYVERSE - MongoDB Aggregation**

## ****1. Filtering –**** $match

## Q1: Get all stories with more than 1000 views.

## db.stories.aggregate([

## { $match: { views: { $gt: 1000 } } }

## ])

## Q2: Find users from city "Mumbai".

## db.users.aggregate([

## { $match: { "profile.city": "Mumbai" } }

## ])

## ****2. Field Shaping –**** $project****,**** $addFields

## Q3: Show only story title and author name.

## db.stories.aggregate([

## { $project: { title: 1, authorName: 1, \_id: 0 } }

## ])

## Q4: Add a field estimatedReadTime assuming 200 words per minute.

## db.stories.aggregate([

## {

## $addFields: {

## estimatedReadTime: {

## $ceil: { $divide: ["$wordCount", 200] }

## }

## }

## }

## ])

## ****3. Aggregation –**** $group

## Q5: Count number of stories per genre.

## db.stories.aggregate([

## {

## $group: {

## \_id: "$genre",

## storyCount: { $sum: 1 }

## }

## }

## ])

## Q6: Get average rating per story.

## db.ratings.aggregate([

## {

## $group: {

## \_id: "$storyId",

## avgRating: { $avg: "$rating" }

## }

## }

## ])

## ****4. Array Handling –**** $unwind

## Q7: List each tag used in stories (1 per line).

## db.stories.aggregate([

## { $unwind: "$tags" },

## {

## $project: {

## title: 1,

## tag: "$tags"

## }

## }

## ])

## Q8: Show user and each story they bookmarked.

## db.users.aggregate([

## { $unwind: "$bookmarks" },

## {

## $project: {

## userId: "$\_id",

## storyId: "$bookmarks"

## }

## }

## ])

## ****5. Pagination & Sorting –**** $sort****,**** $skip****,**** $limit

Q9: Show top 5 most liked stories.

db.stories.aggregate([

{ $sort: { likes: -1 } },

{ $limit: 5 }

])

Q10: Get page 2 with 10 stories per page.

db.stories.aggregate([

{ $skip: 10 },

{ $limit: 10 }

])

## ****6. Analytics –**** $count****,**** $bucket****,**** $facet

## Q11: Count number of registered users.

## db.users.aggregate([

## { $count: "totalUsers" }

## ])

## Q12: Bucket stories by word count.

## db.stories.aggregate([

## {

## $bucket: {

## groupBy: "$wordCount",

## boundaries: [0, 500, 1000, 2000, 5000],

## default: "Above 5000",

## output: {

## count: { $sum: 1 }

## }

## }

## }

## ])

Q13: Run multiple stages together (short vs long stories).

## db.stories.aggregate([

## {

## $facet: {

## shortStories: [{ $match: { wordCount: { $lt: 1000 } } }],

## longStories: [{ $match: { wordCount: { $gt: 3000 } } }]

## }

## }

## ])

## ****7. Joins –**** $lookup

## Q14: Join stories with their author details.

## db.stories.aggregate([

## {

## $lookup: {

## from: "users",

## localField: "authorId",

## foreignField: "\_id",

## as: "authorInfo"

## }

## }

## ])

## Q15: Join comments with corresponding story titles.

## db.comments.aggregate([

## {

## $lookup: {

## from: "stories",

## localField: "storyId",

## foreignField: "\_id",

## as: "story"

## }

## }

## ])

## ****8. Advanced –**** $merge****,**** $replaceRoot****,**** $setWindowFields

Q16: Summarize total likes per user and store it in a new collection.

db.stories.aggregate([

{

$group: {

\_id: "$authorId",

totalLikes: { $sum: "$likes" }

}

},

{ $merge: "author\_like\_summary" }

])