Code Summary: Computed Pattern

In this example, we implemented a roll-up operation which results in a summary document for each book type. The summary document includes the number of books and average number of authors for each product type.

The `$sum` operator increments the count of books for each type. The `$size` operator gets the number of authors in the `authors` array which is then averaged for the type using the `$avg` operator.

```
var roll_up_product_type_and_number_of_authors_pipeline = [
  {
    $group: {
      _id: "$product_type",
      count: {
        $sum: 1,
      },
      averageNumberOfAuthors: {
        $avg: {
          $size: "$authors",
        },
      },
    },
  },
]
```

Finally, we run the aggregation by using the aggregate method on the books collection, passing in the roll up pipeline as an argument.

```
db.books.aggregate(roll_up_product_type_and_number_of_authors_pipeline)
```

The resulting documents look like this:

```
[
  { _id: 'audiobook', count: 1, averageNumberOfAuthors: 1 },
  { _id: 'ebook', count: 1, averageNumberOfAuthors: 3 },
  { _id: 'book', count: 1, averageNumberOfAuthors: 3 }
]
```