1. **EXECUTE QUERY**

|  |  |
| --- | --- |
| **Input: queryId** | **Output: [shopperId]** |
| **queryId**: query051  **indexName:** shoppers  **queryBody**: {"condition":"OR","rules":[{"id":"firstName","field":"firstName","type":"string","input":"text","operator":"equal","value":"Nguyen"},{"id":"birthMonth","field":"birthMonth","type":"integer","input":"number","operator":"equal","value":11}]} | [shopper051, shopper052, shopper053] |
| **http://35.237.224.155:5000/method/query/execute/query051** | |

1. **EXECUTE SEGMENTATION**

|  |  |
| --- | --- |
| **Input: segmentationId** | **Output: [shopperId]** |
| **segmentationId:** segmentation051  **queryId:** ["query051", “query052”] | [shopper052, shopper053] |
| **http://35.237.224.155:5000/method/segmentation/execute/segmentation051** | |

1. **EXECUTE PERSIONALIZATION**

|  |  |
| --- | --- |
| **Input: shopperId & personalizationId** | **Output: point** |
| **shopperId**: shopper052  **lastName**: Dino  **birthMonth**: 11 | 150 |
| **personalizationId**: personalization051  **scoredQuerieIds**:  [  { "queryIds" : "query053"  "point" : 100 },  { "queryIds" : "query051"  "point" : 50 }  ] |
| **http://35.237.224.155:5000/method/personalization/execute/shopper052/personalization051** | |

1. **DEMO SEGMENTATION**

|  |  |
| --- | --- |
| **Input: assignmentId** | **Output: [assigned\_content]** |
| **assignmentId:** assignment051  **contentId**: content051  **triggerId:** trigger051  **competitionPoolId:** comp051  **segmentationId**: segmentation052  **validFrom**: 2019-06-11T01:00:00Z  **validTo**: 2019-06-13T01:00:00Z  **expireAfter:** 1 | [ {  "shopperId": "shopper052",  "contentId": "content051",  "segmentationId": "segmentation052",  "competitionPoolId": "comp051",  "createTime": "2019-06-11T03:12:13Z",  "validTo": "2019-06-12T03:12:13Z"  } ] |
| **http://35.237.224.155:5000/method/demoSegmentation/assignment051** | |

1. **Goal:** assigning a content to shoppers in a segmentation. Since there are many shoppers in a segmentation, thus a list of assigned\_content will be returned, each assigned\_content has shopper and content information, as well as some other metadata.
2. **How it works:**

From assignmentId, we can get:

* **contentId:** the content needed to be assigned
* **competitionPoolId:** type of assignment, normally consists of a set of related contents
  + **frequencyNumber, frequencyUnit**: maximum frequency that a content will be sent out. At max there are frequencyNumber content/frequencyUnit will be sent. For example, maximum 10 contents/2 days.
  + **limit**: the maximum number of content sent to each shopper.
  + **overwriteFlag**: true – allowing overwriting for assigned\_content with nearest valid time.
* **segmentationId:** only considering shoppers in this segmentation.
* **validFrom:** time to start this assignment.
* **validTo**: time to end this assignment.
* **expireAfter:** time to end after creating assigned\_content

Working mechanism:

* Select shoppers from segmentation.
* Check **4 conditions**, if satatisfied, create an assigned\_content for each shopper. assigned\_content will store related information, in which, **createTime** is the time assigned\_content created, and **validTo** will be calculated by: validTo(assigned\_content) = createTime + expireAfter(assignment).
* **4 conditions:**
  + check\_Limit\_Content: content condition, check in all assigned\_content how many contentId have been sent, if limit is reached, stop sending.
  + check\_Frequency\_Content: content condition, check in related assigned\_content the frequency of sending a contentId, if reaching **frequencyNumber content/frequencyUnit**, stop sending.
  + check\_Limit\_CompetionPool: competion pool condition, check in related assigned\_content how many contentId have been sent for a shopperId. If **limit** is reached, stop sending. Here we also check the overwriteFlag as mentioned above.
  + check\_Frequency\_CompetionPool: competion pool condition, check in related assigned\_content the frequency of sending a contentId for a shopperId, if reaching **frequencyNumber content/frequencyUnit**, stop sending

1. **DEMO PERSIONALIZATION**

|  |  |
| --- | --- |
| **Input: shopperId & triggerId** | **Output: assigned\_content** |
| **shopperId**: shopper052  **lastName**: Dino  **birthMonth**: 11 | {  "shopperId": "shopper052",  "contentId": "content052",  "personalizationId": "personalization052",  "competitionPoolId": "comp053",  "createTime": "2019-06-11T05:21:03Z",  "validTo": "2019-06-12T05:21:03Z"  } |
| **triggerId:** trigger051 |
| **http://35.237.224.155:5000/method/demoPersonalizationTrigger/shopper052/trigger051** | |

1. **Goal:** from shopperId và triggerId, assigning most suitable content to a shopper.
2. **How it works:**

From triggerId, check and get all assignment has the triggerId that are valid. In all assignment with personalization, get the personalizationId, calculate the points, select assignment with highest score. Then check with 4 conditions similar to demo segmentation. Assignment that satisfied with highest score will be chosen to create assigned\_content.