

# KYLIN-5529 Support adding base indexes by selecting "Base Aggregate Index" or "Base Detail Index".

## 1. Background

If you want to add a basic index, you can single-select "Basic Aggregate Index", and support both page and API addition methods.

## 2. Acceptance criteria

1. When adding a basic index to a page, you can add a basic aggregate index or a basic detail index separately.
2. Create model API supports adding basic aggregate index or basic detail index separately

## 3. Detailed design

- ### 3.1 When adding a basic index to a page, you can add a basic aggregate index or a basic detail index separately.

## model\_clone\_hs

最近修改: 2023-01-17 14:49:34 GMT+8

### 基本信息

### 数据特征

### Segment

### 索引

### 开发者

### 索引总览

### 优化建议

### 聚合组

### 明细索引

模型数据范围: 1992-01-01 00:00:00 GMT+8 至 1998-08-02 00:00:00 GMT+8

### + 索引

### 构建索引

### 删除

### 聚合组

### Index ID

### 存储

### 明细索引

000070...

711 KB

### 基础索引

0001

711 KB

## 1. Interface

POST kylin/api/index\_plans/base\_index


## 2. Current situation

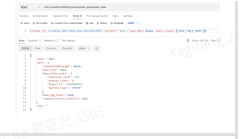
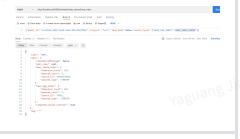
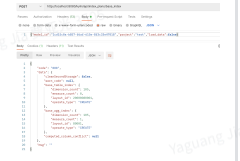
The interface can meet the requirements: **when adding a basic index to a page, it supports adding a basic aggregate index or a basic detail index separately**

## 3. Parameter usage example:

```
1 {"model_id":"1cd15c0a-b857-f6af-615e-8f3c33e47018","project":"test","load_data":  
2 "source_types":["BASE_TABLE_INDEX"]  
3 }
```

## 4. Test

source_types parameters	Result	Interface screenshot
source_types 只传 BASE_AGG_INDEX	Call interface success only add <b>base aggregate index</b>	
source_types 只传 BASE_TABLE_INDEX		

	Call interface success only add <b>basic detail index</b>	
source_types 传 <code>BASE_AGG_INDEX</code> 和 <code>BASE_TABLE_INDEX</code>	Call interface successfully added <b>basic aggregate index and basic detail index</b>	
source_types not transmitted	Call interface successfully added <b>basic aggregate index and basic detail index</b>	

## 5. Modify

Not yet available

## 5.2 Create Model API supports adding basic aggregate index or basic detail index separately

### 1. Interface

- POST `http://host:port/kylin/api/models`

[https://docs.kyligence.io/books/v4.6/zh-cn/developer/v4/model\\_api/model\\_management\\_api.cn.html#%E5%88%9B%E5%BB%BA%E6%A8%A1%E5%9E%8B](https://docs.kyligence.io/books/v4.6/zh-cn/developer/v4/model_api/model_management_api.cn.html#%E5%88%9B%E5%BB%BA%E6%A8%A1%E5%9E%8B)

### 2. Current situation

- with\_base\_index: true: both the basic aggregate index and the basic detail index are created
- with\_base\_index: false: both the basic aggregate index and the basic detail index are not created

```

1 //调用创建基础索引会同时创建
2 public void createAndAddBaseIndex(NDataModel model) {
3     checkIsNotCachedAndShared();
4     LayoutEntity agg = createBaseAggIndex(model);
5     LayoutEntity table = createBaseTableIndex(model);
6     List<LayoutEntity> baseLayouts = Lists.newArrayList();
7     baseLayouts.add(agg);
8     if (table != null) {
9         baseLayouts.add(table);
10    }
11    createAndAddBaseIndex(baseLayouts);
12 }
```

### 3. Modify

1. In order to support separate control over whether the basic aggregate index and the basic detail index are created

`IndexPlan` new method `createAndAddBaseIndex (NDataModel model, List< IndexEntity. Source > sources)` , add parameters to control the creation of the base index alone

- `IndexEntity.Source.BASE_AGG_INDEX` : Individual control over creating base aggregate indexes
- `IndexEntity.Source.BASE_TABLE_INDEX` : Individual control to create a base detail index

2. To keep the same as before modification, `createAndAddBaseIndex (NDataModel model)` calls `createAndAddBaseIndex (model, Lists.newArrayList (IndexEntity.Source.BASE_AGG_INDEX, IndexEntity.Source.BASE_TABLE_INDEX))`; created by default

```
1 public void createAndAddBaseIndex(NDataModel model, List<IndexEntity.Source> sou
2     checkIsNotCachedAndShared();
3     List<LayoutEntity> baseLayouts = Lists.newArrayList();
4     if (sources.contains(IndexEntity.Source.BASE_AGG_INDEX)) {
5         LayoutEntity agg = createBaseAggIndex(model);
6         baseLayouts.add(agg);
7     }
8
9     if (sources.contains(IndexEntity.Source.BASE_TABLE_INDEX)) {
10        LayoutEntity table = createBaseTableIndex(model);
11        if (table != null) {
12            baseLayouts.add(table);
13        }
14    }
15    if (!baseLayouts.isEmpty()) {
16        createAndAddBaseIndex(baseLayouts);
17    }
18 }
19
20 public void createAndAddBaseIndex(NDataModel model) {
21     ArrayList<IndexEntity.Source> sources = Lists.newArrayList(IndexEntity.Sourc
22         IndexEntity.Source.BASE_TABLE_INDEX);
23     createAndAddBaseIndex(model, sources);
24 }
```

3. In order to meet the rules of the PRD convention:

现有参数保留：

- `with_base_index` - 可选 `boolean`，是否添加基础索引，基础索引包含模型全部维度和度量，随着模型变化自动更新，默认值：`false`

新增参数用于选择要添加的索引类型：

- `base_index_type` - 可选 `array[string]`，选择要添加的基础索引类型，可选值 `base_agg_index`，`base_table_index`
  - `[base_agg_index, base_table_index]` 两个都添加
  - `[base_agg_index]` 只添加基础聚合索引
  - `[base_table_index]` 只添加基础明细索引
  - `[]` 不添加基础索引




新参数 `base_index_type` 不与旧参数 `with_base_index` 强制绑定，可单独使用，当 `base_index_type` 存在时，以新参数为准，如下表：





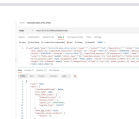
	没有配置 <code>base_index_type</code>	配置 <code>base_index_type</code>
<code>with_base_index = true</code>	两个都添加	按 <code>base_index_type</code> 添加
<code>with_base_index = false</code>	两个都不添加	按 <code>base_index_type</code> 添加
没有配置 <code>with_base_index</code>	两个都不添加	按 <code>base_index_type</code> 添加

手册不需要去掉旧参数 `with_base_index`，只需要解释说明新参数。

In `ModelService`，`modelRequest.isWithBaseIndex ()` is replaced by judging the **basic aggregate index and the basic detail index respectively**

## 4. Test

Parameter	Result	Interface screenshot
Single parameter		
<code>"with_base_index":true</code>	<b>Both the basic aggregate index and the basic detail index</b> are created, Consistent with before, as expected	
<code>"with_base_index":false</code>	<b>The basic aggregate index and the basic detail index</b> are not created, Consistent with before, as expected	
<code>"base_index_type":["BASE_TABLE_INDEX"]</code>	<b>Basic detail index</b> created, as expected	

"base_index_type":["BASE_AGG_INDEX"]	<b>Basic aggregate index</b> creation, as expected	
"base_index_type": ["BASE_TABLE_INDEX","BASE_AGG_INDEX"]	<b>Both the basic aggregate index and the basic detail index</b> are created, In line with expectations	
"base_index_type":[]	<b>The basic aggregate index and the basic detail index</b> are not created, as expected	
Two-parameter		
"with_base_index": <b>true</b> ,"base_index_type": ["BASE_TABLE_INDEX"]	<b>Basic detail index</b> created, as expected	
"with_base_index": <b>false</b> ,"base_index_type": ["BASE_TABLE_INDEX"]	<b>Basic detail index</b> created, as expected	

## 5.3 Edit model API supports adding basic aggregate index or basic detail index separately

### 1. Acceptance criteria

Edit model modification logic: refer to the PRD mentioned, also need to support: separately add "basic aggregate index" or "basic detail index"

产品逻辑和当前保持一致：

- 初次保持模型，默认添加“基础聚合索引”+“基础明细索引”，用户可以分别取消勾选
- 再次编辑模型，不可删除已经添加过的基础索引，可以添加未添加过的基础索引
- 开启分层存储时，必须添加基础明细索引，不推荐添加基础聚合索引

### 2. Interface

```
PUT /api/models/semantic
```

### 3. Current situation

There are two switches to modify the base index

- createIfNotExistTableLayout: Basic Detail Index
- createIfNotExistAggLayout: Basic Aggregate Index



```

@Transaction(project = 0)
public BuildBaseIndexResponse updateBaseIndex(String project, CreateBaseIndexRequest request,
        boolean createIfNotExistTableLayout, boolean createIfNotExistAggLayout, boolean isAuo) {
    aclEvaluate.checkProjectWritePermission(project);
    // update =

```

- createIfNotExist Controls the assignment of two switches at the same time

<pre> 82         long curBaseTableLayout = getBaseTableLayout(); 83         boolean needCreateBaseTable = createIfNotExist; 84         if (exist(preBaseTableLayout) &amp;&amp; notExist(curBaseTableLayout)) { 85             needCreateBaseTable = true; 86         } 87     } 88     Long curExistBaseAggLayout = getBaseAggLayout(); 89     boolean needCreateBaseAgg = createIfNotExist; 90     if (exist(preBaseAggLayout) &amp;&amp; notExist(curExistBaseAggLayout)) { 91         needCreateBaseAgg = true; 92     } 93 } </pre>	<pre> 95         long curBaseTableLayout = getBaseTableLayout(); 96         boolean needCreateBaseTable = 97             updateBaseIndexTypes.contains(IndexEntity.Source.BASE_TABLE_INDEX); 98         if (exist(preBaseTableLayout) &amp;&amp; notExist(curBaseTableLayout)) { 99             needCreateBaseTable = true; 100         } 101     } 102     Long curExistBaseAggLayout = getBaseAggLayout(); 103     boolean needCreateBaseAgg = 104         updateBaseIndexTypes.contains(IndexEntity.Source.BASE_AGG_INDEX); 105     if (exist(preBaseAggLayout) &amp;&amp; notExist(curExistBaseAggLayout)) { 106         needCreateBaseAgg = true; 107     } 108 } </pre>
--	---

## 4. Modify logic

### 1. CreateIfNotExist in BaseIndexUpdateHelper modified to updateBaseIndexTypes

<pre> 43         private String project; 44         private String modelId; 45         private boolean createIfNotExist; 46         private boolean needUpdate; 47         private boolean secondStorageEnabled = false; 48     } </pre>	<pre> 48         private String project; 49         private String modelId; 50         private List&lt;IndexEntity.Source&gt; updateBaseIndexTypes; 51         private boolean needUpdate; 52         private boolean secondStorageEnabled = false; 53     } </pre>
--	---

### 2. To reuse the model API created in 5.2, the processing logic gets the value of updateBaseIndexTypes

<pre> 2999         copyModel.init(modelManager.getConfig(), project, 3000             modelManager.getCCRelatedModels(copyModel)); 3001         BaseIndexUpdateHelper baseIndexUpdater = new 3002             BaseIndexUpdateHelper(originModel, 3003                 request.isWithBaseIndex()); 3004         preProcessBeforeModelSave(copyModel, project); </pre>	<pre> 3032         copyModel.init(modelManager.getConfig(), project, 3033             modelManager.getCCRelatedModels(copyModel)); 3034         BaseIndexUpdateHelper baseIndexUpdater = new 3035             BaseIndexUpdateHelper(originModel, 3036                 needHandleBaseIndexType(request)); 3037         preProcessBeforeModelSave(copyModel, project); </pre>
--	---

- Modify the constructor of BaseIndexUpdateHelper

<pre> 54         NIndexPlanManager indexPlanManager = 55             NIndexPlanManager.getInstance(KylinConfig.getInstanceFromEnv(), 56                 model.getProject()); 57         IndexPlan indexPlan = indexPlanManager.getIndexPlan(model.getId()); 58         @@ -61,7 +72,7 @@ public BaseIndexUpdateHelper(NDataModel model, boolean createIfNotExist) { 59             if (needUpdate) { 60                 project = model.getProject(); 61                 modelId = model.getId(); 62                 this.createIfNotExist = createIfNotExist; 63                 preBaseAggLayout = getBaseAggLayout(); 64                 preBaseTableLayout = getBaseTableLayout(); 65             } </pre>	<pre> 64         + public BaseIndexUpdateHelper(NDataModel model, List&lt;IndexEntity.Source&gt; 65             updateBaseIndexTypes) { 66             NIndexPlanManager indexPlanManager = 67                 NIndexPlanManager.getInstance(KylinConfig.getInstanceFromEnv(), 68                     model.getProject()); 69             IndexPlan indexPlan = indexPlanManager.getIndexPlan(model.getId()); 70             if (needUpdate) { 71                 project = model.getProject(); 72                 modelId = model.getId(); 73                 this.updateBaseIndexTypes = updateBaseIndexTypes; 74                 preBaseAggLayout = getBaseAggLayout(); 75                 preBaseTableLayout = getBaseTableLayout(); 76             } </pre>
--	--

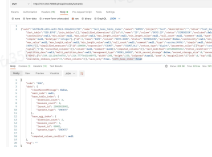
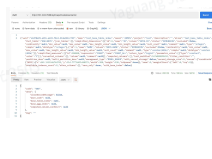

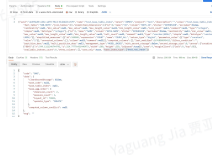
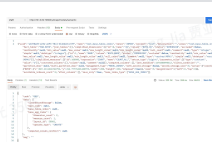
- Added constructor to be compatible with external calls
  - createIfNotExist: true, create **basic aggregate index and basic detail index**
  - createIfNotExist: false, do not create **basic aggregate index and basic detail index**

```

public BaseIndexUpdateHelper(NDataModel model, boolean createIfNotExist) {
57
58 +     this(model,
59 +         createIfNotExist
60 +         ? Lists.newArrayList(IndexEntity.Source.BASE_AGG_INDEX,
61 +             IndexEntity.Source.BASE_TABLE_INDEX)
62 +         : Lists.newArrayList());
63 + }

```

## 5. Test interface

Parameter	Result	Interface screenshot
Regression parameters		
"with_base_index":true	Both the basic aggregate index and the basic detail index are created,  In line with expectations	
"with_base_index":false	The basic aggregate index and the basic detail index are not created,  In line with expectations	
Added base_index_type parameter test		
"base_index_type": ["BASE_TABLE_INDEX"]	Basic detail index created, as expected	
"base_index_type": ["BASE_AGG_INDEX"]	Basic aggregate index creation, as expected	
Multiple edit model test		
"base_index_type" when the basic detail index already exists: ["BASE_AGG_INDEX"]	The basic aggregate index is created, and the existing basic detail index is not affected, as expected	
"base_index_type" when the underlying aggregate index already exists: ["BASE_TABLE_INDEX"]	The basic detail index is created, and the existing basic aggregate index is not affected, as expected	