Содержание

1) Сформировать в ADVANTA LINQ-запросы 3
Пример LINQ-запроса данных по проектам 3
Пример LINQ-запроса данных по контрольным точкам 4
2) Создать шаблоны таблиц данных в формате .xlsx 5
Пример шаблона для источника данных по проектам 6
Пример шаблона для источника данных по контрольным точкам 6
3) Создать модель в Analytic Workspace и сформировать скрипт в ETL-редакторе 7
Пример скрипта для проектов 8
Пример скрипта для контрольных точек 10
Пример модели 12
4) Создать виджеты для отображения данных и дашборды из виджетов 12
Пример дашборда «Здоровье портфеля» в Analytic Workspace на основе данных, полученных из ADVANTA 12

Интеграция ADVANTA с Analytic Workspace

Analytic Workspace — российская BI-система, позволяющая обрабатывать данные из разных источников и визуализировать их при помощи широкого набора встроенных инструментов. Благодаря интеграции возможно использование инструментария Analytic Workspace для построения дашбордов на основе данных из ADVANTA. Это дает дополнительные возможности для анализа данных и последующего формирования управленческих решений.

Ниже приведена подробная пошаговая инструкция по организации процесса обмена данными между двумя системами. Для реализации обмена у вас должен быть доступ к по крайней мере к демо-версиям данных продуктов.

1) Сформировать в ADVANTA LINQ-запросы

Интеграция с Analytic Workspace организована с использованием LINQ-запросов и Web-API ADVANTA. Таким образом, для того, чтобы загрузить данные в BI-систему, необходимо предварительно сформировать LINQ-запросы внутри ADVANTA для получения тех данных, которые в дальнейшем будут визуализированы.

Пример LINQ-запроса данных по проектам

```
  /* 1. перечень всех проектов (2 типа объектов - ИТ-проект и Организационный проект)
с UID  и всеми системными реквизитами (сроки начала/завершения - план, факт,
ответственные (ФИО), статус (текстом), последний базовый план)
- объект-родитель (название)
и пользовательскими реквизитами:
- жизненный цикл проекта
- описание
- масштаб проекта (текстовое значение)
- эффект
- последняя запись из справочника "Отчет о статусе" - все поля
- активная запись из справочника "Бюджет проекта" - только поле "Бюджет (план)"
*/

var projects = dataContext.Projects
      .Where(p => p is IT_proekt_d46b6d || p is
Organizacionnij_proekt_a3e2fb)
      .OrderBy(p => p.CreationDate)
      .Select(p => new { p.Id,
                         p.Name, ParentName = p.Parent.Name,
                         LifeCycleName =
p.Fields.Zhiznennij_cikl_proekta_b07f57.Name,
                         p.SystemStartDate, p.SystemEndDate,
p.ActualStartDate, p.ActualEndDate,
                         p.BaselinePlanStartDate, p.BaselinePlanEndDate,
                         BossFIO = String.Concat( p.Owner.LastName, " ",
p.Owner.FirstName), RPFIO = String.Concat( p.Responsible.LastName, " ",
```

```
p.Responsible.FirstName),
                           Description = p.Fields.Opisanie_caaaaa,
                           ProjectScaleName =
p.Fields.Masshtab_proekta_04412f.Name,
                           ProjectGainName = p.Fields.Effekt_b6c145.Name,
                           StatusReport_Date =
p.GetChildren<Otchet_o_statuse_440616>().OrderByDescending(o =>
o.Date).FirstOrDefault().Date,
                           StatusReport_Chto_sdelano_problemi_riski =
p.GetChildren<Otchet_o_statuse_440616>().OrderByDescending(o =>
o.Date).FirstOrDefault().Chto_sdelano_problemi_riski_4a11b8,
                           StatusReport_Prichina_problemi =
p.GetChildren<Otchet_o_statuse_440616>().OrderByDescending(o =>
o.Date).FirstOrDefault().Prichina_problemi_a3e564.Name,
                           StatusReport_StatusName =
p.GetChildren<Otchet_o_statuse_440616>().OrderByDescending(o =>
o.Date).FirstOrDefault().Status_e05ba3.Name,
                           BudjetSum =
p.GetChildren<Byudzhet_plan_e5bee7>().Where(b =>
b.Aktivnostj_versii_281a38.Id ==
Classifier_Aktivnostj_versii_625508.Aktivnaya_4f5ba775_Id).Sum(b =>
b.Summa_06ff93)
                         });
return projects;
```

Пример LINQ-запроса данных по контрольным точкам

```
/*
2. перечень всех контрольных точек (1 тип объекта - KT0)
с UID  и всеми системными реквизитами (как и у проекта),
-  UID  -  проекта-родителя (непрямой родитель)
и пользовательские поля:
- описание (текст)
- результат (текст)
- последняя запись из справочника "Отчет о мероприятии" - все поля
*/
var projects = dataContext.KT0_1ff431_List
      .Select(p => new {
                    p.Id,
                    p.Name,
                    ProjectId =
(Guid?)p.GetParentHierarchy<Project>(false).Where(p => p is IT_proekt_d46b6d
|| p is Organizacionnij_proekt_a3e2fb).FirstOrDefault().Id,
                    // ProjectName =
p.GetParentHierarchy<Project>(false).Where(p => p is IT_proekt_d46b6d || p
is Organizacionnij_proekt_a3e2fb).FirstOrDefault().Name,
                    p.SystemStartDate, p.SystemEndDate, p.ActualStartDate,
p.ActualEndDate,
```

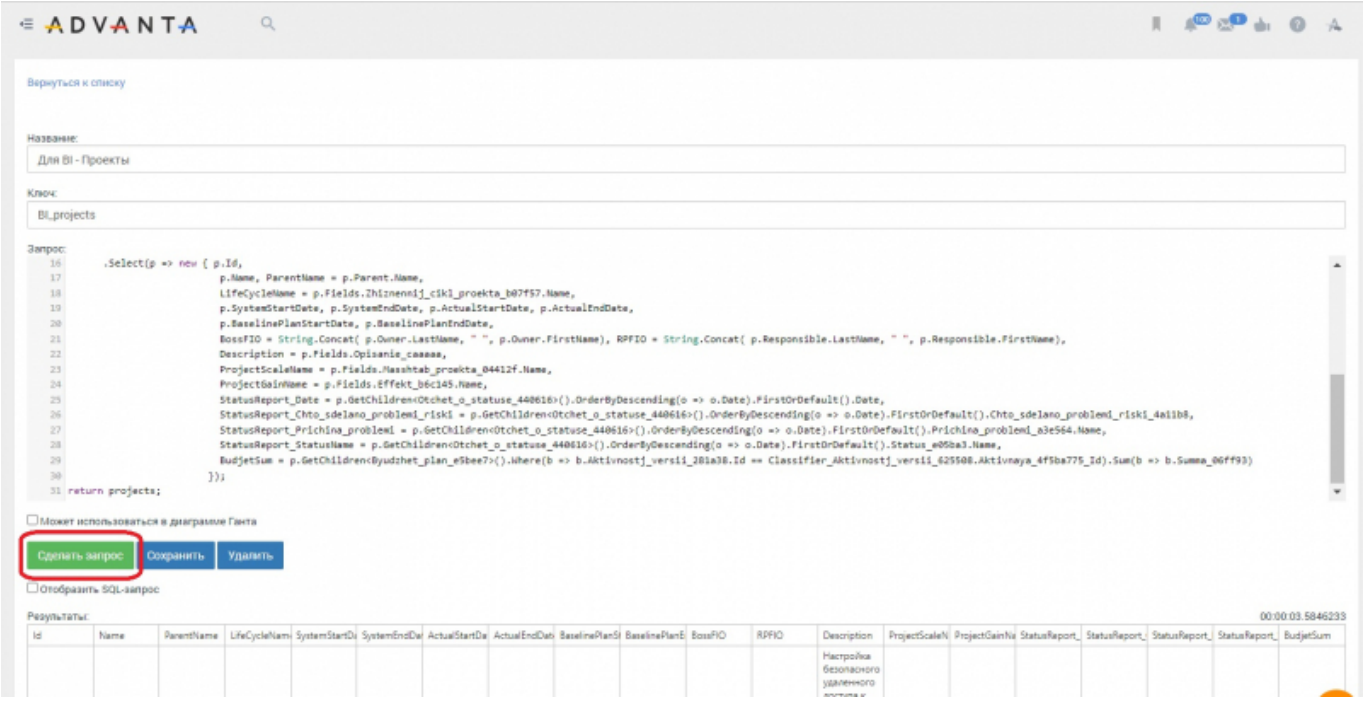
```
                    p.BaselinePlanStartDate, p.BaselinePlanEndDate,
                    BossFIO = String.Concat( p.Owner.LastName, " ",
p.Owner.FirstName), RPFIO = String.Concat( p.Responsible.LastName, " ",
p.Responsible.FirstName),
                    Description = p.Opisanie_caaaaa,
                    Result = p.Rezuljtat_8e716f,
                    StatusReport_Date =
p.GetChildren<Otchet_o_statuse_meropriyatiya_2bc942>().OrderByDescending(o
=> o.Date).FirstOrDefault().Date,
                    StatusReport_Chto_sdelano_problemi_riski =
p.GetChildren<Otchet_o_statuse_meropriyatiya_2bc942>().OrderByDescending(o
=> o.Date).FirstOrDefault().Chto_sdelano_problemi_riski_4a11b8,
                    StatusReport_Prichina_problemi =
p.GetChildren<Otchet_o_statuse_meropriyatiya_2bc942>().OrderByDescending(o
=> o.Date).FirstOrDefault().Prichina_problemi_a3e564.Name,
                    StatusReport_StatusName =
p.GetChildren<Otchet_o_statuse_meropriyatiya_2bc942>().OrderByDescending(o
=> o.Date).FirstOrDefault().Status_e05ba3.Name
      });
return projects;
```

> Данный код запросов приведен в качестве примера. На вашей инсталяции необходимо будет указать наименования сущностей, объектов, справочников, реквизитов, соответствующих сформированному контексту LINQ-запросов.

2) Создать шаблоны таблиц данных в формате .xlsx

Данные шаблоны необходимы для формирования структуры хранения данных в AW в соответствии со структурой получаемых их ADVANTA данных.

Для создания шаблона необходимо **выполнить созданный LINQ-запрос** в ADVANTA, скопировать шапку полученной таблицы в Excel и сохранить файл в формате .xlsx.



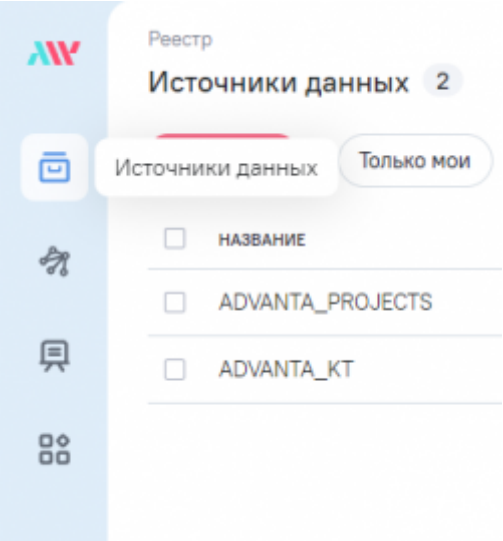
Пример шаблона для источника данных по проектам

Id	Name	ParentName	LifeCycleName	SystemStartDate	SystemEndDate	ActualStartDate	ActualEndDate

Пример шаблона для источника данных по контрольным точкам

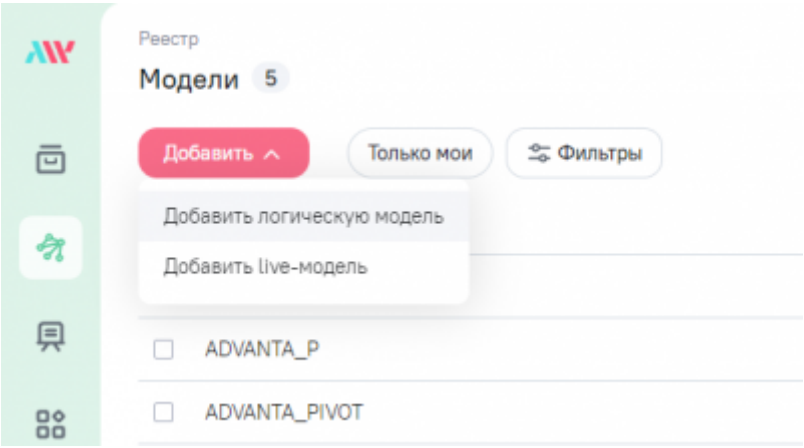
Id	Name	ProjectId	SystemStartDate	SystemEndDate	ActualStartDate	ActualEndDate	BaselinePlanStartDate

Загрузите данные шаблоны в Analytic Workspace, в раздел «Источники данных» — здесь необходимо **создать источник, выбрать формат «Файл» и загрузить сформированный шаблон**.

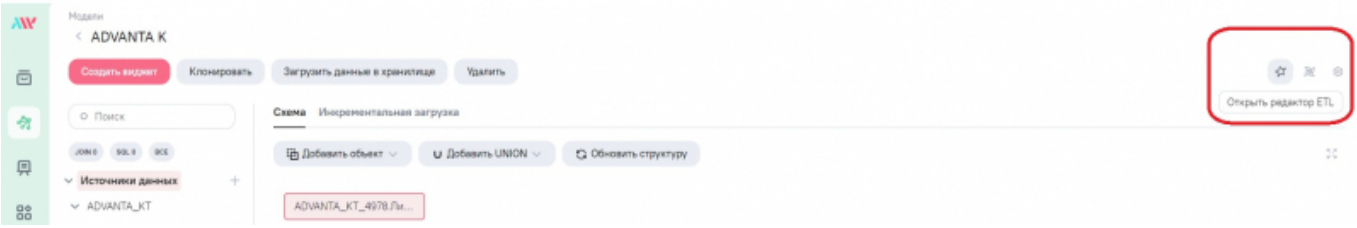


3) Создать модель в Analytic Workspace и сформировать скрипт в ETL-редакторе

После создания источника необходимо в соответствующем разделе Analytic Workspace **создать модель**, которая и будет служить витриной данных для созданного источника. Необходимо создать столько же моделей, сколько предполагается использовать LINQ-запросов.



Сразу после создания необходимо перейти в режим редактирования модели и зайти в **ETL-редактор**.



В открывшемся редакторе необходимо ввести скрипт для обработки получаемых из ADVANTA данных. Скрипт написан на Python.

Пример скрипта для проектов

```
# ---------------------------------------------------------------------------
---------
# Скрипт для обработки модели * * *
# ---------------------------------------------------------------------------
---------

import requests
import datetime
from pyspark.sql import Row

# python - m pip install requests, matplotlib, pandas

from pandas import json_normalize

def after_all(df, spark, app, *args, **kwargs):
    print(df.schema)

    LOGIN = '* * *'
    PASSWORD = '* * *'
    DOMAIN = 'https://* * *.ru'

    AUTH = {
    'Login': LOGIN,
    'Password': PASSWORD,
    }

    session = requests.Session()
    # cookies = session.cookies.get_dict() # {}

    # авторизация
    response = session.post(url=DOMAIN+'/api/auth/login', json=AUTH,
verify=False)

    cookies = session.cookies.get_dict()

    # print(cookies)

    # получение LINQ-запроса
    LINQ = {
        'DataSourceKey': 'BI_projects',
        # нужен другой запрос для перечня проектов в модель list1
        # 'DataSourceKey': 'BI_projects',
        # и еще один для перечня КТ из этих проектов в модель list1_ywrq
        # 'DataSourceKey': 'BI_milestones',
        'PageSize': 100,
    }

    r = session.post(url=DOMAIN+'/api/queries/get', cookies=cookies,
```

```
json=LINQ, verify=False)

    if not r.ok:
        raise Exception(f'Ошибка в {r.url}. HTTP {r.status_code}: {r.text}')

    data = r.json()

    #Создадим список, который послужит основной для создания DataFrame
    rows = []
    for line in data:
        rows.append(Row(
            id=line['Id'],
            name=line['Name'],
            parentname=line['ParentName'],
            lifecyclename=line['LifeCycleName'],
            #
systemstartdate=str(datetime.datetime.fromisoformat(line['SystemStartDate'])
.replace(tzinfo=datetime.timezone.utc)),
            #
systemenddate=str(datetime.datetime.fromisoformat(line['SystemEndDate']).rep
lace(tzinfo=datetime.timezone.utc)),
            #
actualstartdate=str(datetime.datetime.fromisoformat(line['ActualStartDate'])
.replace(tzinfo=datetime.timezone.utc)),
            #
actualenddate=str(datetime.datetime.fromisoformat(line['ActualEndDate']).rep
lace(tzinfo=datetime.timezone.utc)),
            #
baselineplanstartdate=str(datetime.datetime.fromisoformat(line['BaselinePlan
StartDate']).replace(tzinfo=datetime.timezone.utc)),
            #
baselineplanenddate=str(datetime.datetime.fromisoformat(line['BaselinePlanEn
dDate']).replace(tzinfo=datetime.timezone.utc)),

            systemstartdate=line['SystemStartDate'],
            systemenddate=line['SystemEndDate'],
            actualstartdate=line['ActualStartDate'],
            actualenddate=line['ActualEndDate'],
            baselineplanstartdate=line['BaselinePlanStartDate'],
            baselineplanenddate=line['BaselinePlanEndDate'],

            bossfio=line['BossFIO'],
            rpfio=line['RPFIO'],
            description=line['Description'],
            projectscalename=line['ProjectScaleName'],
            projectgainname=line['ProjectGainName'],

            #
statusreport_date=str(datetime.datetime.fromisoformat(line['StatusReport_Dat
e']).replace(tzinfo=datetime.timezone.utc)),
            statusreport_date=line['StatusReport_Date'],
```

```
statusreport_chto_sdelano_problemi_riski=line['StatusReport_Chto_sdelano_pro
blemi_riski'],
statusreport_prichina_problemi=line['StatusReport_Prichina_problemi'],
            statusreport_statusname=line['StatusReport_StatusName'],
            budjetsum=line['BudjetSum']
        ))

    return spark.createDataFrame(rows)
```

Пример скрипта для контрольных точек

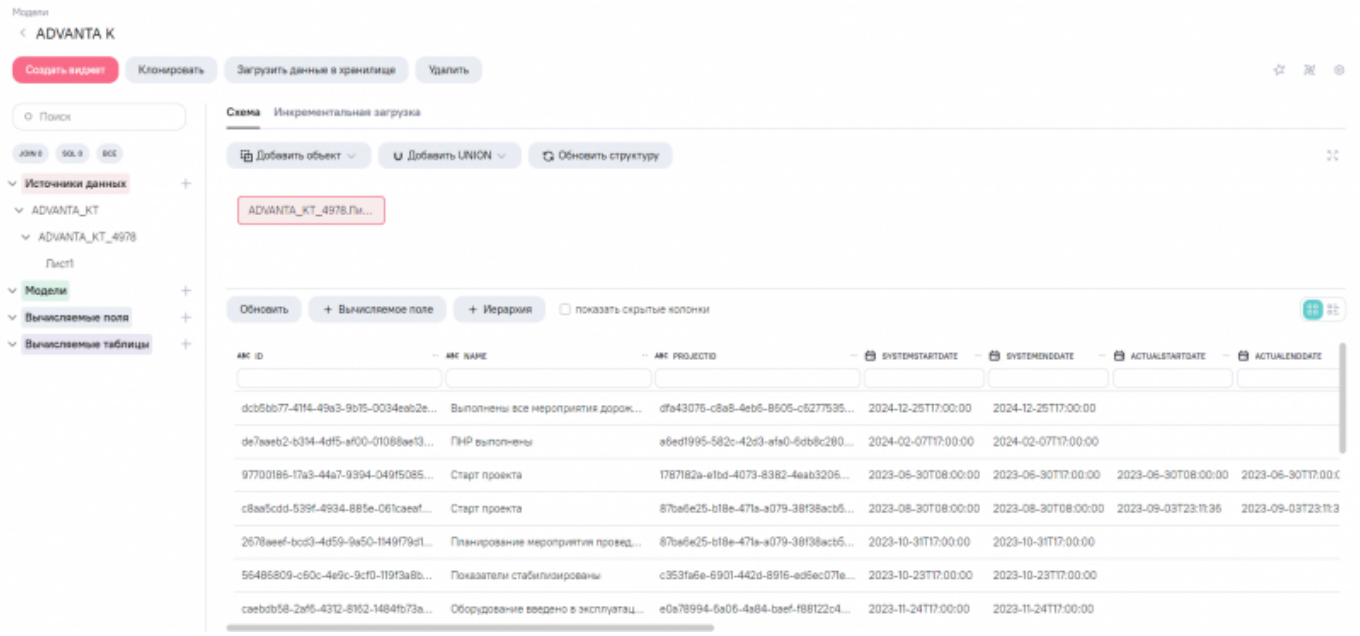
```
# # ---------------------------------------------------------------------------
-----------
# Скрипт для обработки модели * * *
# ---------------------------------------------------------------------------
---------

import requests
import datetime
from pyspark.sql import Row
from pandas import json_normalize

def after_all(df, spark, app, *args, **kwargs):

    LOGIN = '* * *'
    PASSWORD = '* * *'
    DOMAIN = 'https://* * *.ru'

    AUTH = {
    'Login': LOGIN,
    'Password': PASSWORD,
    }

    session = requests.Session()
    # cookies = session.cookies.get_dict() # {}

    # авторизация
    response = session.post(url=DOMAIN+'/api/auth/login', json=AUTH,
verify=False)

    cookies = session.cookies.get_dict()

    # print(cookies)

    # получение LINQ-запроса
    LINQ = {
        'DataSourceKey': 'BI_milestones',
```

```
        'PageSize': 1000,
    }

    r = session.post(url=DOMAIN+'/api/queries/get', cookies=cookies,
json=LINQ, verify=False)

    if not r.ok:
        raise Exception(f'Ошибка в {r.url}. HTTP {r.status_code}: {r.text}')

    data = r.json()

    #Создадим список, который послужит основной для создания DataFrame
    rows = []
    for line in data:
        print(line)
        rows.append(Row(
            id=line['Id'],
            name=line['Name'],
            projectid=line['ProjectId'],
            systemstartdate=line['SystemStartDate'],
            systemenddate=line['SystemEndDate'],
            actualstartdate=line['ActualStartDate'],
            actualenddate=line['ActualEndDate'],
            baselineplanstartdate=line['BaselinePlanStartDate'],
            baselineplanenddate=line['BaselinePlanEndDate'],
            bossfio=line['BossFIO'],
            rpfio=line['RPFIO'],
            description=line['Description'],
            result=line['Result'],
            statusreport_date=line['StatusReport_Date'],
statusreport_chto_sdelano_problemi_riski=line['StatusReport_Chto_sdelano_pro
blemi_riski'],
statusreport_prichina_problemi=line['StatusReport_Prichina_problemi'],
            statusreport_statusname=line['StatusReport_StatusName']
        ))

    return spark.createDataFrame(rows)
```

> Символами * * * в примерах скриптов обозначены автоматически формируемые данные (например, номер модели) или данные, имеющие отношение к конкретной учетной записи в ADVANTA (например, логин, пароль и домен).

Затем необходимо опубликовать скрипт и обновить модель. После обновления данные из ADVANTA автоматически загрузятся в модель.

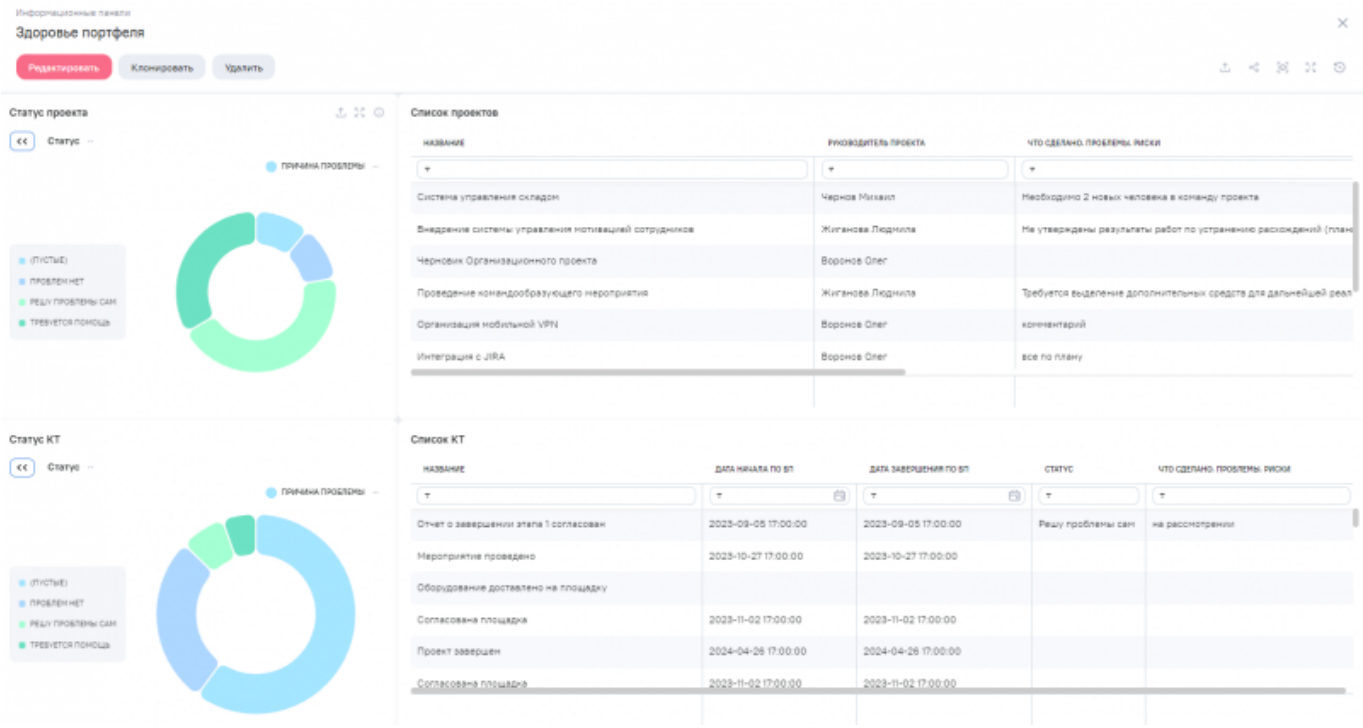
Пример модели



4) Создать виджеты для отображения данных и дашборды из виджетов

Для реализации этих шагов необходимо ознакомиться с возможностями системы Analytic Workspace. В этом вам помогут база знаний, обучающий курс и специалисты, контакты которых вы можете найти на сайте Analytic Workspace.

Пример дашборда «Здоровье портфеля» в Analytic Workspace на основе данных, полученных из ADVANTA



Оставить заявку на демонстрацию или внедрение ADVANTA

Узнать подробнее о системе Analytic Workspace

From:
https://wiki.a2nta.ru/ - **Wiki [3.x]**

Permanent link:
https://wiki.a2nta.ru/doku.php/product/api/integration_examples/aw

Last update: **07.08.2024 05:55**