**Technical task\*\*\***

**DWH: IncreaseES + ContactsBDW**

**(Data WareHouse)**

**(IncreaseES, increase\_es, ies)**

**(Contacts, Big deals, Warehouse)**

**Internet Gold Rush**

The main goal for this project DWH is creating common database, based on IncreaseES data plus some of ContactsBDW data. DWH help to understand relation between promotion and really sale deals (one big sale deal and/or a lot of small/medium deals). Perhaps different channels are needed to promote large and small/medium deals.

**Main goals of DWH application to explore:**

- What are the best kind of promotion for really big/medium/small sale deals?

- Best methods of SEO for big/medium/small sale deals?

- Best methods for sale on C2C sites for big/medium/small sale deals?

**Requirements:**

- Working on computers.

- Main online Database based on IncreaseES data.

- Replicate data from ContactsBDW to IncreaseES. Tables: brands, measure\_type, products, types, customers, customers\_deals.

**Developer tools\*\*\***

- Python 3.8: Visual Studio Code 1.69.1.

- MongoDB ATLAS online 24\*7: MongoDB Compass.

- Django 3.1 > 4.1.2 – web-framework + ORM.

- Django REST framework

- PyMongo – 4.2.0 (3.11.2) > 3.12.3

- Djongo – 1.3.6 (SQLparse - 0.2.4)

- D3.js – library for reports.

- Java (1.8): basic programmer language. IDE: Eclipse IDE for Java Developers.

- PostgreSQL (11.16): basic database. Database administration tools: HeidiSQL, Dbeaver.

- Spring Boot (2.4.1): main framework.

- Hibernate (5.2.6.Final): framework for working with database.

- Application programming interface: Rest API, server Eureka,

- Maven (4.0.0): project management and building tool.

**DWH DBMS\*\*\***

Cloud MongoDB ATLAS – free, 24\*7, NoSQL.

**Algorithm\*\*\***

Primary micro service: Types, Brands tables without references parents.  
Secondary micro service: Customers, Customers\_deals, Products tables.

ContactsBDW\_DWH (java):

- eureka server.port=8761

- primary-service server.port=8081

- secondary-service server.port=8082

IncreaseES (python):

service server.port=8001

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Rest interface format**

**Test user:** kanvladi

**Test password:** \*\*\*\*\*\*\*\*

Same user and same password must be in IncreaseES (users and orm users) and ContactsBDW.

**IncreaseES** (python):

- <http://localhost:8000/>

**ContactsBDW** (java):

- [http://localhost:8080/](http://localhost:8000/)

**Eureka** server (java):

- http://localhost:8761/

Download and filling data of **Types** table:

- GET [http://localhost:8001/api/types](http://localhost:8000/api/types)

(IncreaseES <> DWH IncreaseES) (python)

- GET http://localhost:8081/api/types/2021-10-19 10:57:33

(DWH IncreaseES <> DWH ContactsBDW) (java)

- request.body: {"user\_id":"kanvladi", "pswd":"\*\*\*\*\*\*\*\*"}

Download and filling data of **Brands** table:

- GET [http://localhost:8001/api/brands](http://localhost:8000/api/types)

(IncreaseES <> DWH IncreaseES) (python)

- GET http://localhost:8081/api/brands/2021-10-19 10:57:33

(DWH IncreaseES <> DWH ContactsBDW) (java)

- request.body: {"user\_id":"kanvladi", "pswd":"\*\*\*\*\*\*\*\*"}

Download and filling data of C**ustomers** table:

- GET [http://localhost:8001/api/](http://localhost:8000/api/types)customers

(IncreaseES <> DWH IncreaseES) (python)

- GET http://localhost:8082/api/customers/2021-10-19 10:57:33

(DWH IncreaseES <> DWH ContactsBDW) (java)

- request.body: {"user\_id":"kanvladi", "pswd":"\*\*\*\*\*\*\*\*"}

Download and filling data of C**ustomers\_deals** table:

- GET [http://localhost:8001/api/](http://localhost:8000/api/types)customers\_deals

(IncreaseES <> DWH IncreaseES) (python)

- GET http://localhost:8082/api/customers\_deals/2021-10-19 10:57:33

(DWH IncreaseES <> DWH ContactsBDW) (java)

- request.body: {"user\_id":"kanvladi", "pswd":"\*\*\*\*\*\*\*\*"}

Download and filling data of P**roducts** table:

- GET [http://localhost:8001/api/](http://localhost:8000/api/types)products

(IncreaseES <> DWH IncreaseES) (python)

- GET http://localhost:8082/api/products/2021-10-19 10:57:33

(DWH IncreaseES <> DWH ContactsBDW) (java)

- request.body: {"user\_id":"kanvladi", "pswd":"\*\*\*\*\*\*\*\*"}

Download and filling data of L**og\_dwh** table:

- POST [http://localhost:8001/api/](http://localhost:8000/api/types)log\_dwh

(IncreaseES <> DWH IncreaseES) (python)

- POST http://localhost:8082/api/log\_dwh

(DWH IncreaseES <> DWH ContactsBDW) (java)

- request.body: {"table\_name":"types", "user\_id":"kanvladi", "pswd":"\*\*\*\*\*\*\*\*"}

**Structure DB**

**Django Rest Framework ORM**

**class Types(models.Model):**

#class Meta:

# unique\_together = (('type\_name', 'dataset'),)

id\_orig = models.IntegerField(unique = True) # id from source, original database

type\_name = models.CharField(max\_length=30) # types of goods or services

desc = models.CharField(max\_length=300, blank=True, null=True) # description

date\_create = models.DateField(null=True) # date creating record

dataset = models.IntegerField(null=True) # 1 - live, 0 – archive/deleted, -1/-2/-3 - reserv

last\_user = models.CharField(max\_length=20, blank=True, null=True) # last user

last\_changed = models.DateTimeField(null=True) # date and time last changed

**class Customers(models.Model):**

#class Meta:

# unique\_together = (('dataset', 'type\_name', 'first\_name', 'middle\_name', 'last\_name'),)

id\_orig = models.IntegerField(unique = True) # id from source, original database

type\_name = models.ForeignKey(Types, on\_delete=models.PROTECT)

first\_name = models.CharField(max\_length=20, blank=True, null=True)

middle\_name = models.CharField(max\_length=20, blank=True, null=True)

last\_name = models.CharField(max\_length=20, blank=True, null=True)

TIN = models.CharField(max\_length=30, blank=True, null=True) # INN

mobile\_p = models.CharField(max\_length=20, blank=True, null=True)

station\_p = models.CharField(max\_length=20, blank=True, null=True)

address = models.CharField(max\_length=50, blank=True, null=True)

desc = models.CharField(max\_length=300, blank=True, null=True) # description

date\_create = models.DateField(null=True) # date creating record

dataset = models.IntegerField(null=True) # 1 - live, 0 – archive/deleted, -1/-2/-3 - reserv

last\_user = models.CharField(max\_length=20, blank=True, null=True) # last user

last\_changed = models.DateTimeField(null=True) # date and time last changed

**class Customers\_deals(models.Model):**

#class Meta:

# unique\_together = (('dataset', 'id\_customer', 'name\_good', 'price', 'quantity', 'sum', 'date\_deal'),)

id\_orig = models.IntegerField(unique = True) # id from source, original database

id\_customer = models.ForeignKey(Customers, on\_delete=models.PROTECT)

name\_good = models.CharField(max\_length=30, blank=True, null=True)

price = models.BigIntegerField(null=True)

quantity = models.IntegerField(null=True)

sum = models.BigIntegerField(null=True)

date\_deal = models.DateField(null=True) # date creating record

details = models.CharField(max\_length=300, blank=True, null=True) # description

dataset = models.IntegerField(null=True) # 1 - live, 0 – archive/deleted, -1/-2/-3 - reserv

last\_user = models.CharField(max\_length=20, blank=True, null=True) # last user

last\_changed = models.DateTimeField(null=True) # date and time last changed

**class Brands(models.Model):**

#class Meta:

# unique\_together = (('dataset', 'brand\_name'),)

id\_orig = models.IntegerField(unique = True) # id from source, original database

brand\_name = models.CharField(max\_length=30, blank=True, null=True)

desc = models.CharField(max\_length=300, blank=True, null=True) # description

date\_create = models.DateField(null=True) # date creating record

dataset = models.IntegerField(null=True) # 1 - live, 0 – archive/deleted, -1/-2/-3 - reserv

last\_user = models.CharField(max\_length=20, blank=True, null=True) # last user

last\_changed = models.DateTimeField(null=True) # date and time last changed

**class Products(models.Model):**

#class Meta:

# unique\_together = (('dataset', 'type\_name', 'brand\_name', 'product\_name', 'measure'),)

id\_orig = models.IntegerField(unique = True) # id from source, original database

type\_name = models.ForeignKey(Types, on\_delete=models.PROTECT)

brand\_name = models.ForeignKey(Brands, on\_delete=models.PROTECT)

product\_name = models.CharField(max\_length=30, blank=True, null=True)

measure = models.CharField(max\_length=10, blank=True, null=True)

price = models.BigIntegerField(null=True)

desc = models.CharField(max\_length=300, blank=True, null=True) # description

date\_create = models.DateField(null=True) # date creating record

dataset = models.IntegerField(null=True) # 1 - live, 0 – archive/deleted, -1/-2/-3 - reserv

last\_user = models.CharField(max\_length=20, blank=True, null=True) # last user

last\_changed = models.DateTimeField(null=True) # date and time last changed

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# Last download data by tables

**class Last\_download(models.Model):**

class Meta:

unique\_together = (('table\_name', 'dataset'),)

table\_name = models.CharField(max\_length=30) # types of goods or services

last\_download = models.DateTimeField(null=True) # date and time last download

desc = models.CharField(max\_length=300, blank=True, null=True) # description

date\_create = models.DateField(null=True) # date creating record

dataset = models.IntegerField(null=True) # 1 - live, 0 – archive/deleted, -1/-2/-3 - reserv

last\_user = models.CharField(max\_length=20, blank=True, null=True) # last user

last\_changed = models.DateTimeField(null=True) # date and time last changed

# Log download data by tables

**class Log\_dwh(models.Model):**

#class Meta:

# unique\_together = (('actions', 'dataset', 'last\_changed'),)

actions = models.CharField(max\_length=40) #

desc = models.CharField(max\_length=300, blank=True, null=True) # description

date\_create = models.DateField(null=True) # date creating record

dataset = models.IntegerField(null=True) # 1 - live, 0 – archive/deleted, -1/-2/-3 - reserv

last\_user = models.CharField(max\_length=20, blank=True, null=True) # last user

last\_changed = models.DateTimeField(null=True) # date and time last changed

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Spring Boot**

CREATE TYPE measure\_type AS ENUM

('Единица','Комплект','Метр','М\_погон','Кг','Коробка','Прочее','');

\*\*\*\*\*

**Fields create\_date and last\_changed of all table must be always difined.**

\*\*\*\*\*

-- Download data log from ContactsBDW to IncreaseES DWH

CREATE TABLE log\_dwh (

id serial,

actions varchar(50), -- Example: Types table downloaded.

description text,

date\_create date, -- date creating record = actions date

dataset smallint, -- 1 - live, 0 – archive/deleted, -1/-2/-3 - reserv

last\_user varchar(20), -- last user

last\_changed timestamp, -- date and time actions

CONSTRAINT log\_dwh\_c UNIQUE (dataset, actions, last\_changed)

)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Types of goods, services\*\*\***

Новые телефоны

БУ телефоны

Аксессуары моб.

Аксессуары комп.

Аксессуары проч. - ТВ, батарейки, калькуляторы,

Текстиль - Постель, Полотенца, Халаты

Сумки, Рюкзаки

Сантехник

Сайты

Прочее