Business Documentation

São Paulo, 14 de setembro de 2020

CONTROLE DE VERSÃO			
Autor	**Versão**	**Data**	**Descrição**
Lusandro Araújo	1.0	25/08/2020	Criação do documento.
Lusandro Araújo	1.1	27/08/2020	Ajuste nas informações.
Karin Esquina	1.2	14/08/2020	Modificação de documento, para formato documentação.

1. Introdução

Este documento tem como objetivo apresentar em detalhes os requisitos do “Projeto integrado - visões socioeconômicas” do ponto de vista técnico, exigido pela empresa Blueshift, como forma de treinamento do programa BlueShift Academy.

2. Solicitação

O objetivo do projeto foi, apresentar o quadro socioeconômico de nosso País, baseado no consumo de dados por APIs e repositórios públicos, disponibilizados pelo governo federal, em formato resumido de um Dashboard dimensional.

A entrega do projeto foi divida em duas partes. A primeira consistiu no consumo dos dados via APIs e repositórios, disponíveis nos portais do governo federal, através de um script em python para otimização do processo. Estes dados foram armazenados em um arquivo no *Blob Storage do Azure* e consumidos na ingestão do banco de dados *SQL Server*.

A segunda parte consistiu na modelagem dimensional, para construção do Datawarehouse e desenvolvimento dos dashboards, utilizando *PowerBi*.

Para que a visão socioeconomica fosse modulada, utilizamos as seguintes variáveis de análise:

- Renda média;
- Renda per capta;
- Índice de desemprego;
- Índice de IDH;
- Beneficiários do Bolsa Família;
- Beneficiários do Auxilio Emergencial.

3. Premissas da solução

3.1. Origem e especificação dos dados

Os dados foram retirados de APIs e repositórios públicos nos formatos ZIP e CSV, conforme abaixo:

- Renda Média: *http://atlasbrasil.org.br/2013/pt/consulta/*
- Renda per capta: *http://atlasbrasil.org.br/2013/pt/consulta/*
- Índice de desemprego: *http://atlasbrasil.org.br/2013/pt/consulta/*
- Índice de IDH: *http://atlasbrasil.org.br/2013/pt/consulta/*
- Bolsa Família: *http://www.portaltransparencia.gov.br/download-de-dados/bolsa-familia-pagamentos*
- Auxilio Emergencial: *http://www.portaltransparencia.gov.br/pagina-interna/603519-download-de-dados-auxilio-emergencial.*

3.2. Ambiente de desenvolvimento

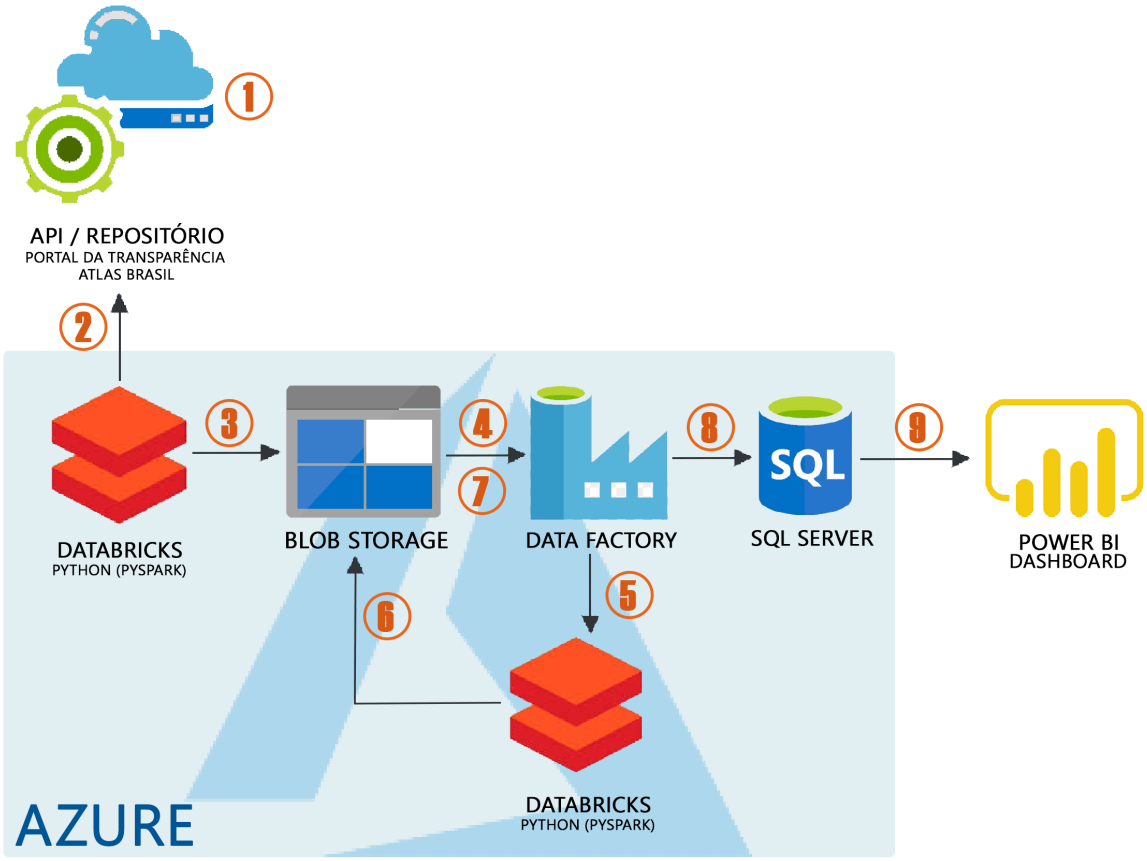
A empresa Blueshift disponibilizou todos os acessos, e ferramentas necessárias para o desenvolvimento da arquitetura definida na apresentação do Blueprint;

Os ambientes de desenvolvimento utilizados para realização desse projeto foram:

- Azure Databricks Microsoft;
- Azure Data Factory;
- Azure Blob Storage (Containers);
- Microsoft Azure SQL Server;
- Power BI;

4. Modelo da arquitetura sugerida

A figura abaixo apresenta a arquitetura da solução com base no levantamento de requisitos e entendimento do negócio.



1. API Governo Federal (Portal da transparencia, Atlas Brasil);
2. Extração dos arquivos via **Databricks**, utilizando um script em Python;
3. Armazenamento dos aquivos no **Azure Blob Storage** em formato *.ZIP*;
4. Descompactação dos arquivos com o **Data Factory**, no formato *.CSV*;
5. Manipulação dos dados, utilizando **Databricks** com lingaguem Python;
6. Armazenamento dos aquivos no **Azure Blob Storage** em formato *.CSV* particionado;
7. Alimentação das tabelas SQL utilizando o **Data Factory**;
8. Criação das tabelas utilizando o **SQL Server**;
9. Consumo das tabelas e montagem dos Dashboards utilizando o **Power BI**.

5. Extração APIs

Foi desenvolvido um scirpt para extração dos dados das APIs de forma automatizada via Python, utilizando o Databricks disponível em: *https://github.com/karinesquina/ProjetoIntegrado*

O script foi desenvolvido respeitando a limitação de 30 consultas por minuto, impostas pelo portal. Abaixo a explicação do script completo.

Buscar os IDs das localidades no site do IBGE

```
data = datetime.now() #identifica a data e hora do start.
anomes = int(data.strftime('%Y%m')) #transforma a data em ano/mês em número inteiro (formato que o arquivo chega).
codibge=3500105
dtibge=202003 #primeiro mês de referência.
res2 = []

#Identificar os IDs dos municipios, de acordo com o site do IBGE.
res = []
u='https://servicodados.ibge.gov.br/api/v1/localidades/distritos' #lista de códigos de município, para ser usado dentro do r.
r=requests.get(u) #chama a u(url)
d=pd.read_json(r.content.decode('utf-8')) #lê o arquivo que voltar da solicitação do r, e salvar em um DF.
x=r.json() #Lê os dados json (transforma em dicinário).
res=pd.json_normalize(x) #insere os dados json dentro do DF. O Normalize serve para normalizar os dados, por exemplo, se tiver subarquivos, ele nivela.
id_ibge=res['municipio.id'] #pega a coluna município.id e insere em uma variável.
id_ibge=['520005005', '310010405']
```

Conectar com a API do site Portal Transparência

```
#Buscar as informações do bolsa família.
num = 0
iccsv=0
while dtibge <= anomes: #enquanto o dtibge for menor igual a data atual.
    for i in id_ibge: #para cada item do id ibge
        if num == 30: #se num = 30
            print('Aguardando....')
            time.sleep(60) #aguardar 60s
            num = 0 #num volta para 0

        num = num+1 #começa a rodar de novo
        headers = {
        'User-Agent': 'request',
        'chave-api-dados': '4d85f7d1d3dfca2ccc0602434ad67c93',
        } #credenciais fornecidas pela api
        url = "http://www.portaltransparencia.gov.br/api-de-dados/bolsa-familia-por-municipio?mesAno="+str(dtibge)+"&codigoIbge="+str(i)+"&pagina=1" #i = item da lista
        #print(url)
        try:
            response = requests.get(url, headers=headers)
            df=pd.read_json(response.content.decode('utf-8')) #transforma o arquivo recebido em DF.
            d = response.json()
            d = r
            df = pd.json_normalize(r)
            for i, c in df.iterrows(): #obtém o valor da próxima linha.
              res2.append(c)
```

Salvar os dados dentro do container do Azure

```
  except:
            print('erro: '+url)
        if iccsv==0: #se o iccsv for 0 o arquivo nao tem a primeira linha (cabecalho), ele vai inserir o cabecalho e salvar em uma pasta.
            df.to_csv('wasbs://projetointegrado@adfandreiablueshift.blob.core.windows.net//Bolsa Familia/API_Data/Teste_API', sep=';', encoding='utf-8',
header='true', index=False)
            iccsv ==1 #If roda apenas a primeira vez.
        else:
          df.to_csv('wasbs://projetointegrado@adfandreiablueshift.blob.core.windows.net//Bolsa Familia/API_Data/Teste_API', sep=';', encoding='utf-8',
header=False, index=False, mode='a')
          print('check')
          dtibge=dtibge+1 #quando terminar, ele vai para o proximo mês.
```

6. Extração de Repositórios

Desenvolvemos um script para buscar os arquivos dos respositórios, porém não o utilizamos, pois:

1. O site **Portal da Transparência**, não permite a extração automatizada de repositórios, somente das APIs;
2. O site **Dados.gov** estave fora do ar durante toda a consulta que fizemos;
3. O site **Atlas Brasil** fornece apenas os dados em formato de consulta. É possivel fazer o download em formato .csv após a consulta.

7. Tratamento de dados via Python

Foi desenvolvido um script em python, no Databricks para fazer a manipulação dos dados, disponível em: *github.com/karinesquina/ProjetoIntegrado*

O script segue a seguinte linha de fluxo:

Conectar com o container do Azure

```
spark.conf.set(
  "fs.azure.account.key.adfandreiablueshift.blob.core.windows.net",
  "GHFhO6sVusVBxM4obc4GIqRufA+6mrYy/4PlBXsVgEnbkPuZ4vQ0sN09afisz/2fpDzh+NBfz6A16PYEjyvXHw==")

input_path = "wasbs://projetointegrado@adfandreiablueshift.blob.core.windows.net//Auxilio emergêncial/"
output_path = "wasbs://projetointegrado@adfandreiablueshift.blob.core.windows.net//Auxilio emergêncial/Output_AC"
```

Listar todos os arquivos dentro da pasta que contenham "data" e "csv" no nome.

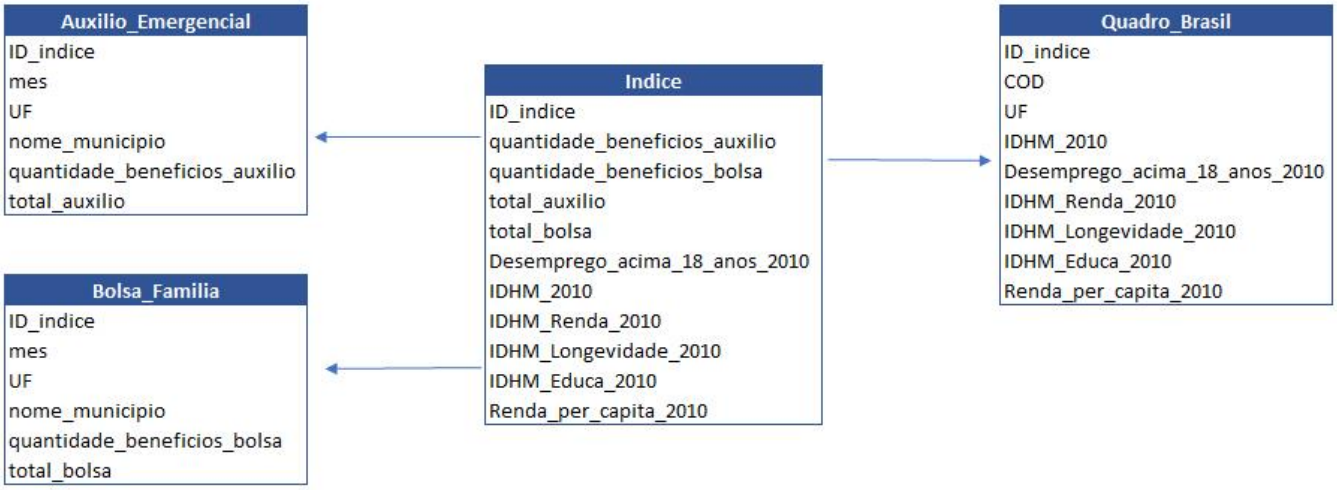
```
list_blob_files = dbutils.fs.ls(input_path)
list_name = []
for i in list_blob_files:
  if "data" in i.name and "csv" in i.name:
    list_name.append(i.path)
list_name
```

Através da lista pré-criada selecionar colunas, tratar dados, e salvar no container do Azure.

```
from pyspark.sql.functions import *

columns = ['MÊS DISPONIBILIZAÇÃO', 'UF', 'CÓDIGO MUNICÍPIO IBGE','NOME MUNICÍPIO', 'VALOR BENEFÍCIO']

df = spark.read.option("header","true").option("inferSchema", "true").option("delimiter", ";").option("charset", "ISO-8859-1").csv(list_name)
df = df[columns]
df = df.na.drop(subset=["UF"])
df = df.where(df.UF != 'None')

df = df.withColumn('VALOR BENEFÍCIO', regexp_replace('VALOR BENEFÍCIO', ',', '.'))
df = df.withColumnRenamed("MÊS DISPONIBILIZAÇÃO", "mes_disponibilizacao").withColumnRenamed("CÓDIGO MUNICÍPIO IBGE", "cod_municipio").withColumnRenamed("NOME
MUNICÍPIO", "nome_municipio").withColumnRenamed("VALOR BENEFÍCIO", "valor_beneficio")

df = df.withColumn("valor_beneficio", df["valor_beneficio"].cast("float"))

state = df.where(col("UF") == "AC")

state.write.option("header", "true").option("charset", "ISO-8859-1").format("com.databricks.spark.csv").save(output_path)
```

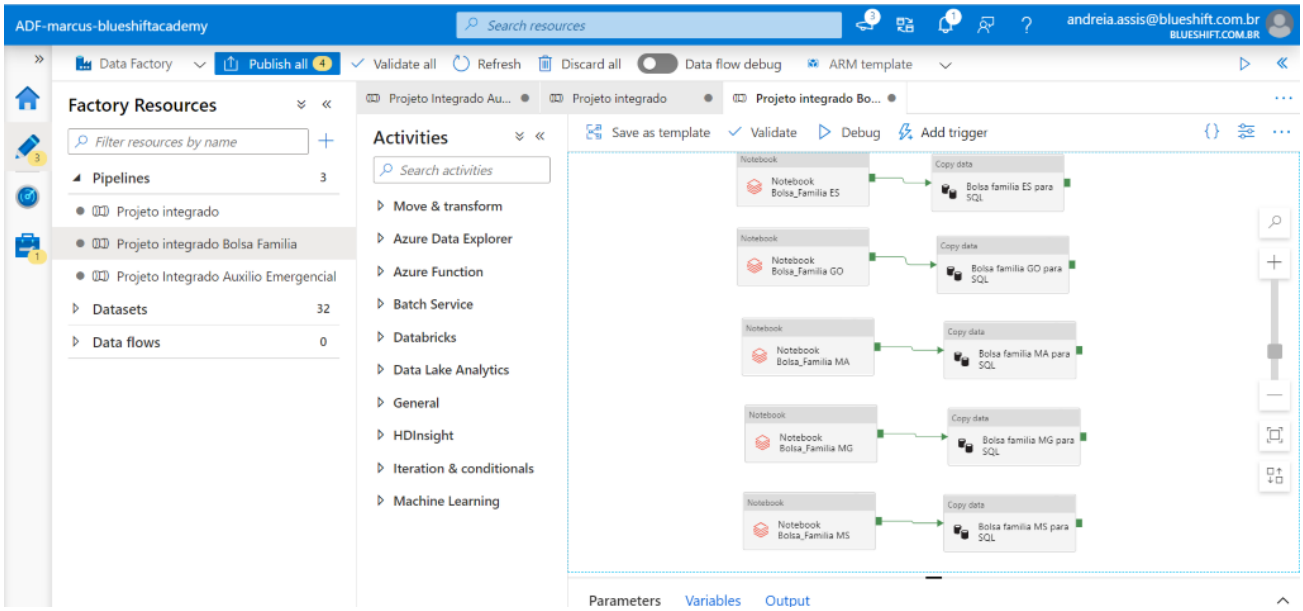
8. Ingestão no SQL Server

Para atender a demanda solicitada inicialmente, foi criado o seguinte modelo relacional.



Para que as tabelas fossem alimentadas com os dados manipulados pelo Python, utilizamos o Data factory com o fluxo seguinte:

1. Importação do Notebook do Databricks;
2. Exportação para a tabela SQL.



8.1. Modelagem tabelas

As tabelas foram modeladas seguindo o padão abaixo:

8.1.1. Auxilio Emergencial

Para alimentar a tabela auxilio emergencial, devido ao grande volume de dados, particionamos os mesmos por Estado. Dessa maneira, criamos uma tabela para cada Estado, contemplando as seguintes colunas:

```
CREATE TABLE [dbo].[Auxilio_Emergencial_SP](
    [Mes_Disponibilizacao] [int] NOT NULL,
    [UF] [varchar](2) NULL,
    [cod_municipio] [int] NULL,
    [nome_municipio] [varchar](500) NULL,
    [valor_beneficio] [float] NULL
)
```

Após a alimentação das *27 tabelas*, referente a todos os Estados, criamos um “View” onde pudemos fazer a soma de valores recebidos, e a contagem da quantidade de beneficiários, conforme o script abaixo:

```
create view [dbo].[Auxilio_Emergencial] as
Select Mes_Disponibilizacao, UF, Nome_municipio, count(valor_beneficio) as ‘quantidade_beneficios_auxilio’, sum(valor_beneficio) as 'total_auxilio' from [dbo].[Auxilio_Emergencial_AC]
group by Mes_Disponibilizacao, UF, Nome_municipio
union all
Select Mes_Disponibilizacao, UF, Nome_municipio, count(valor_beneficio) as ‘quantidade_beneficios_auxilio’, sum(valor_beneficio) as 'total_auxilio' from [dbo].[Auxilio_Emergencial_AL]
group by Mes_Disponibilizacao, UF, Nome_municipio
union all
...
```

8.1.2. Bolsa Família

Para alimentar a tabela auxilio emergencial, devido ao grande volume de dados, particionamos os mesmos por Estado. Dessa maneira, criamos uma tabela para cada Estado, contemplando as seguintes colunas:

```
CREATE TABLE [dbo].[Auxilio_Emergencial_SP](
    [Mes_Disponibilizacao] [int] NOT NULL,
    [UF] [varchar](2) NULL,
    [nome_municipio] [varchar](500) NULL,
    [valor_parcela] [float] NULL
)
```

Após a alimentação das *27 tabelas*, referente a todos os Estados, criamos um “View” onde pudemos fazer a soma de valores recebidos, e a contagem da quantidade de beneficiários, conforme o script abaixo:

```
    create view [dbo].[Bolsa_View] as
    Select Mes, Nome_municipio, UF, count(Valor_parcela) as 'quantidade_beneficios_bolsa', sum(Valor_parcela) as 'total_bolsa' from Bolsa_Familia_AC
    group by Mes, Nome_municipio, UF
    union all
    Select Mes, Nome_municipio, UF, count(Valor_parcela) as ''quantidade_beneficios_bolsa', sum(Valor_parcela) as 'total_bolsa' from Bolsa_Familia_AL
    group by Mes, Nome_municipio, UF
    union all
…
```

8.1.3. Quadro Brasil

Os dados recebidos foram inseridos em uma tabela, contendo as seguintes colunas:

```
CREATE TABLE [dbo].[Quadro_Brasil](
    [COD] [int] NULL,
    [UF] [varchar](500) NULL,
    [Desemprego_acima_18_anos_2010] [float] NULL,
    [IDHM_2010] [float] NULL,
    [IDHM_Renda_2010] [float] NULL,
    [IDHM_Longevidade_2010] [float] NULL,
    [IDHM_Educa_2010] [float] NULL,
    [Renda_per_capita_2010] [float] NULL
)
```

8.1.4. Índice

A tabela Índice foi criada para receber os valores a serem utilizados nos dashboards.

```
CREATE TABLE [dbo].[Quadro_Brasil](
    [ID_indice] [int] NOT NULL,
    [quantidade_beneficio_auxilio] [int] NULL,
    [quantidade_beneficio_bolsa] [int] NULL,
    [total_auxilio] [int] NULL,
    [total_bolsa] [int] NULL,
    [Desemprego_acima_18_anos_2010] [float] NULL,
    [IDHM_2010] [float] NULL,
    [IDHM_Renda_2010] [float] NULL,
    [IDHM_Longevidade_2010] [float] NULL,
    [IDHM_Educa_2010] [float] NULL,
    [Renda_per_capita_2010] [float] NULL
)
```

Todos os scripts estão disponíveis no endereço abaixo:
github.com/karinesquina/ProjetoIntegrado

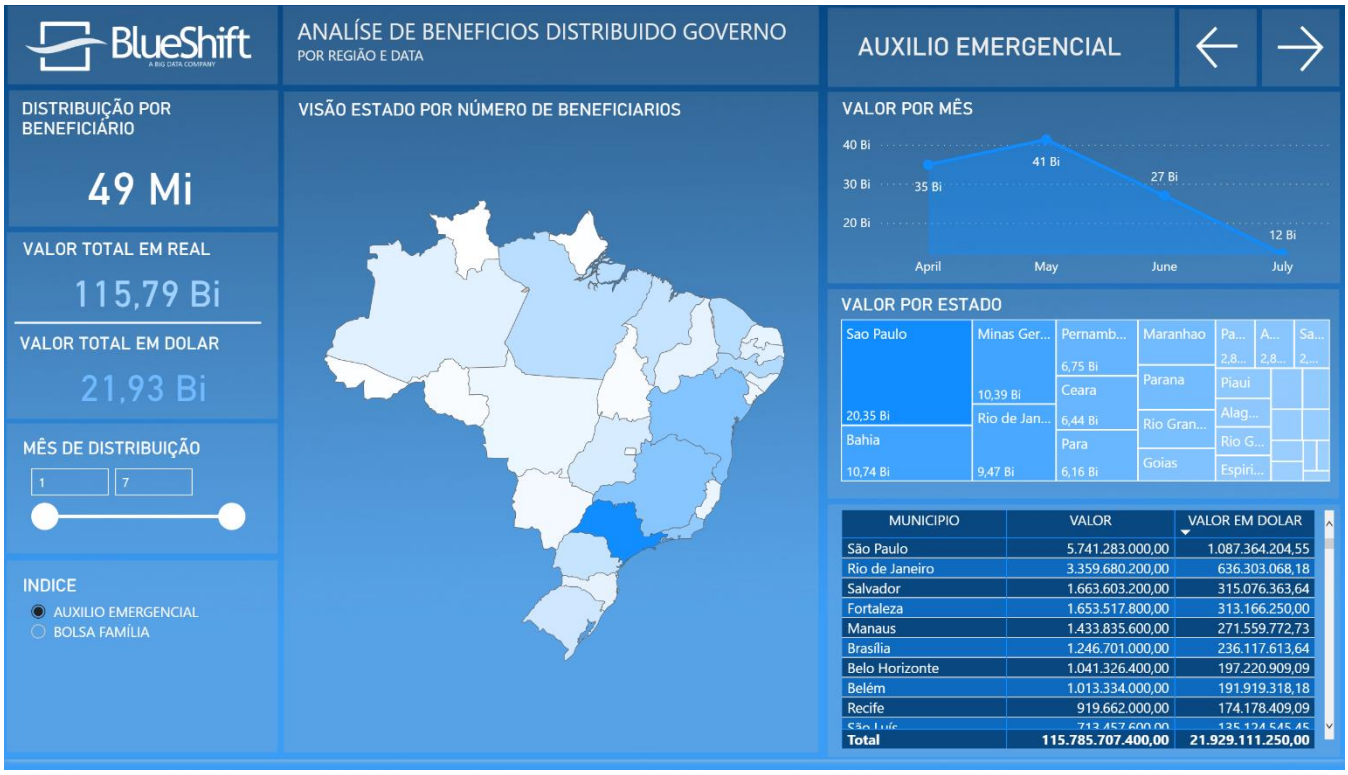
9. Dashboards

Abaixo os dashboards com os resultados obtidos durante a pesquisa.

9.1. Auxilio Emergicial

Baseado nas informações retiradas dos bancos de dados extraidos, obtivemos o seguinte quadro de resultados:

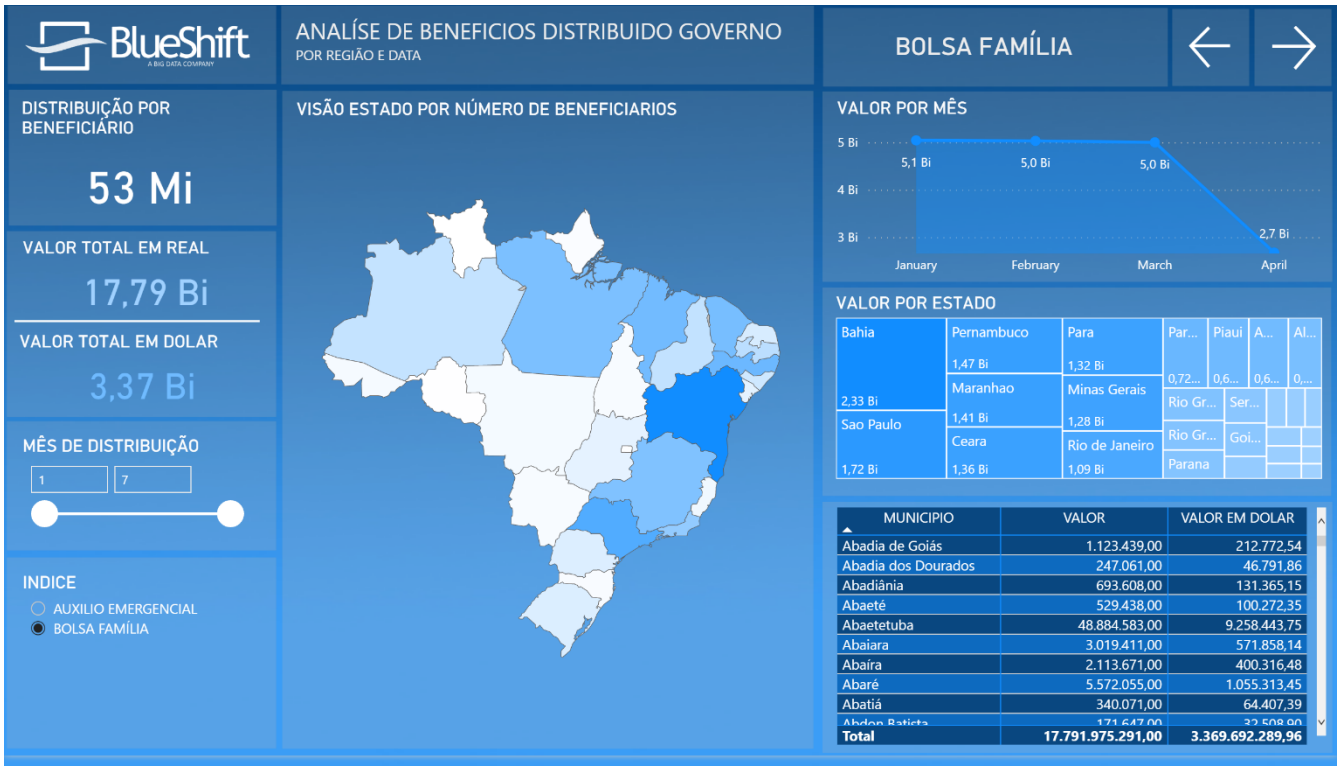
- Visões dos Estados pela quantidade de beneficiários, e total de Auxílio Emergencial recebido. Apresentado por mês, contemplando os meses de abril, maio, junho e julho de 2020;
- 10 municipios que mais receberam Auxílio Emergencial, separados por Estado, e por mês;
- Todos os valores foram convertidos para o dólar, cotado no dia da análise (14/09/2020 – R$5,27).



9.2. Bolsa Família

Baseado nas informações retiradas dos bancos de dados extraidos, obtivemos o seguinte quadro de resultados respondendo as seguintes questões:

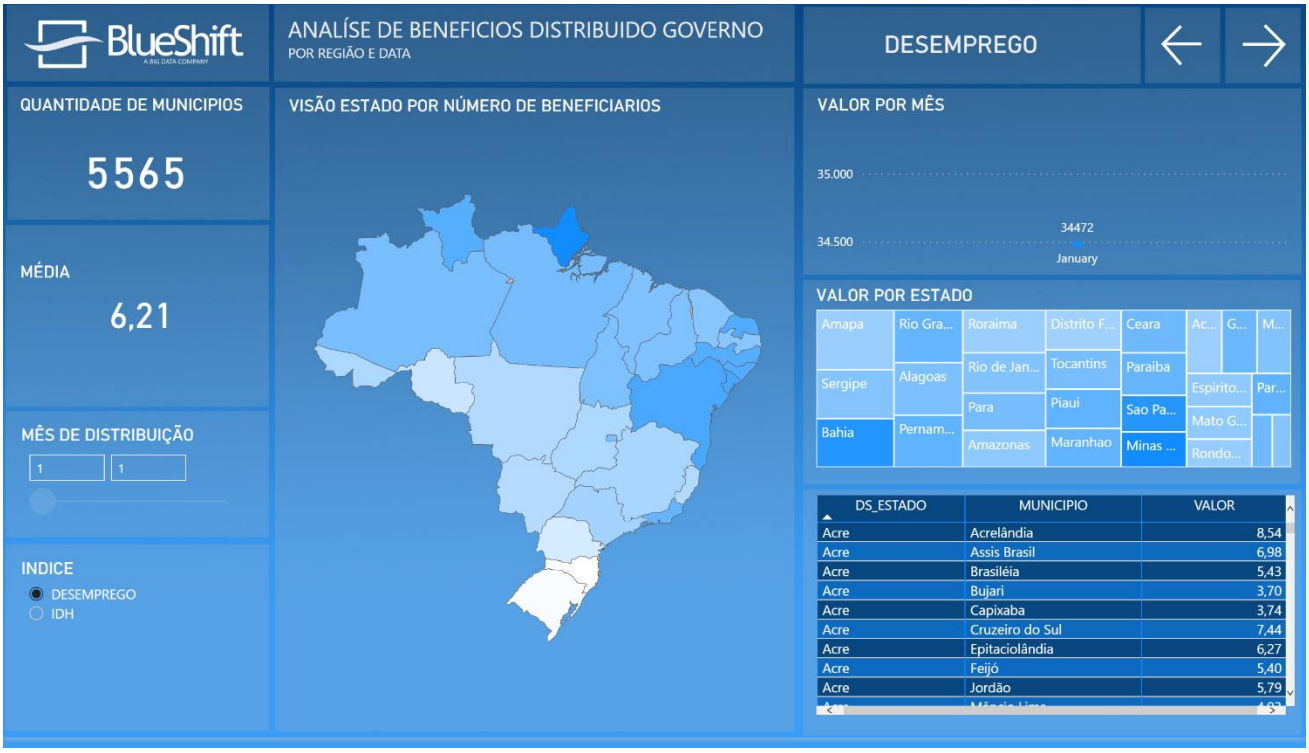
- Visões dos Estados pela quantidade de beneficiários, e total de auxílio Bolsa Família recebido. Apresentado por mês, contemplando todos os meses do ano de 2019 e os meses de janeiro, fevereiro, março, abril, maio, junho e julho de 2020;
- 10 municípios que mais receberam o auxílio Bolsa Família, separados por Estado, e por mês;
- Todos os valores foram convertidos para o dólar, cotado no dia da análise (14/09/2020 – R$5,27).



9.3. Índice de Desemprego

Nossa pesquisa foi baseada nas informações disponibilizadas pelo ***Portal Atlas de Desenvolvimento Humano***, onde os números se baseam no censo de 2010 e são atualizados mensalmente, por uma amostra de 1000 elementos. Dessa maneira podemos identificar em nosso dashboard:

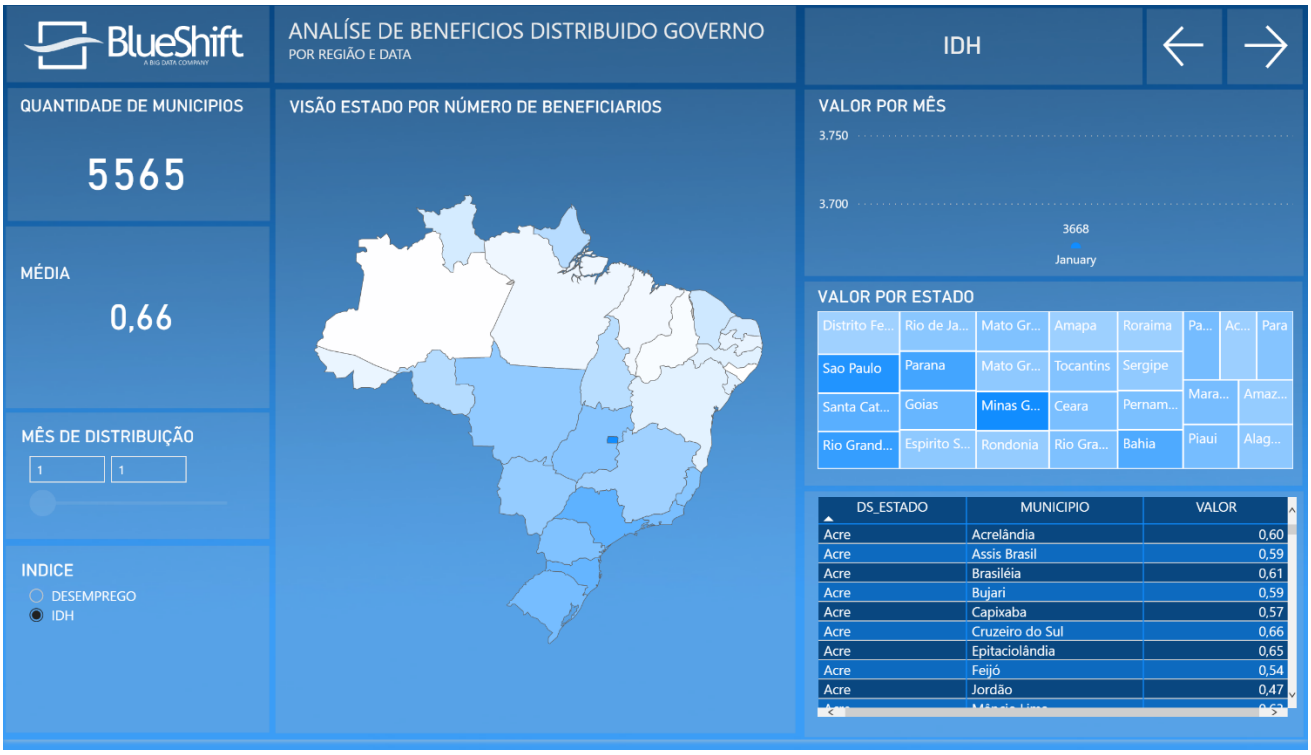
- Porcentagem de desempregados acima de 18 anos, no mês de julho de 2020, apresentados por município;
- Visão geral dos Estados, pela porcentagem do índice de desemprego no mês de julho de 2020;
- Média de desemprego nacional.



9.4. Índice de Desenvolvimento Humano

Nossa pesquisa foi baseada nas informações disponibilizadas pelo ***Portal Atlas de Desenvolvimento Humano***, onde os números se baseam no censo de 2010 e são atualizados mensalmente, por uma amostra de 1000 elementos. Dessa maneira podemos identificar em nosso dashboard:

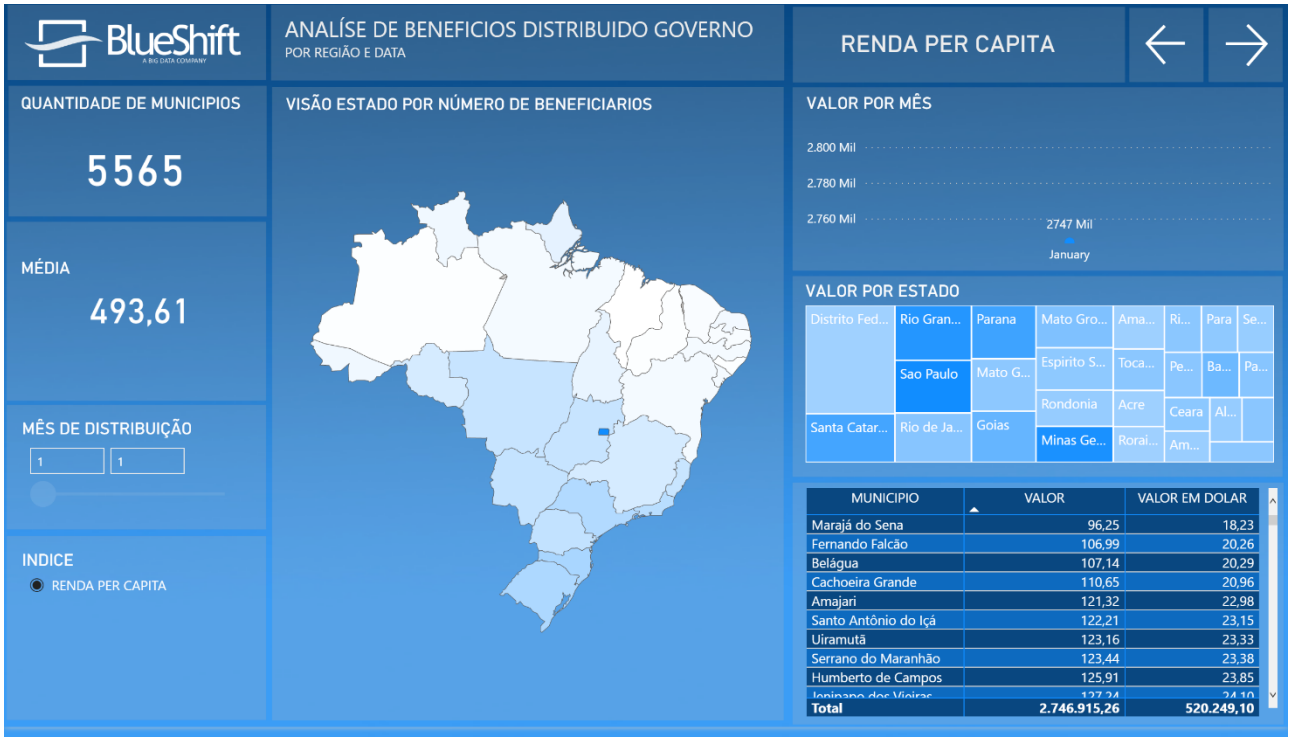
- Índice de desenvolvimento humano, por Estado, atualizado no mês de julho de 2020, apresentado por município;
- IDH do Brasil, no mês de julho de 2020;
- Municípios com maior avaliação de IDH, por Estado, no mês de julho de 2020.



9.5. Renda Per Capta

Nossa pesquisa foi baseada nas informações disponibilizadas pelo ***Portal Atlas de Desenvolvimento Humano***, onde os números se baseam no censo de 2010 e são atualizados mensalmente, por uma amostra de 1000 elementos. Dessa maneira podemos identificar em nosso dashboard:

- Renda per capta, por Estado, atualizada no mês de julho de 2020, apresentada por município;
- Renda média da população, apresentada por Estado, atualizada no mês de julho de 2020;
- Todos os valores foram convertidos para o dólar, cotado no dia da análise (14/09/2020 – R$5,27).



Todos os dashoboards estão disponíveis no endereço abaixo:
github.com/karinesquina/ProjetoIntegrado