

SoulCode Academy
BOOTCAMP – ENGENHARIA DE DADOS + PYTHON



**Relatório de *insights* no processo de ETL:
Setor imobiliário brasileiro**

GRUPO 9
Bruno Drumond
Edlaine Sá
Lucas Cursino
Nathaly Oliveira

SUMÁRIO

1.	Introdução.....	4
2.	Análise.....	4
3.	Base de dados.....	4
4.	Insights.....	4
5.	Extração de dados.....	5
5.1	BigQuery.....	5
5.2	Kaggle.....	5
6.	Armazenamento inicial.....	5
7.	Transformação.....	5
7.1	Colaboratory (Colab).....	5
7.2	Python.....	6
7.3	Pandas.....	6
7.3.1	Extração do dataframe direto do bucket-GCP.....	6
7.3.2	Instalação da biblioteca Pandas no Colab.....	6
7.3.3	Leitura do dataframe no colab.....	6
7.3.4	Backup do dataframe antes das normalizações.....	7
7.3.5	Exclusão (Drop) das colunas.....	7
7.3.6	Renomeando as colunas(tradução)	7
7.3.7	Verificando valores nulos.....	7
7.3.8	Verificando os tipos de imóveis do dataframe.....	7
7.3.9	Renomeando linhas (tradução).....	7
7.3.10	Transformando o formato da data.....	8
7.3.11	Plotagem.....	8
7.3.12	Dataframe sendo salvo diretamente em pasta específica (saída) do bucket....	9
7.4	PySpark.....	9
7.4.1	Instalação do PySpark e da biblioteca necessária para conexão com a Google Cloud Platform.....	9
7.4.2	Importação das bibliotecas e funções necessárias para análise	9
7.4.3	Início da conexão com a GCP.....	10
7.4.4	Iniciar sessão Spark	10
7.4.5	Lendo/abrindo o <i>dataframe</i> que será manipulado.....	10
7.4.6	Visualização do <i>dataframe</i>	10

7.4.7	Verifica se há diferença entre as colunas em questão e quantos são diferentes	11
7.4.8	Verifica os valores iguais entre as colunas em questão.....	11
7.4.9	Visualização das ocorrências dos tipos de propriedades e seu número de corências presentes no <i>dataframe</i>	11
7.4.10	Visualização do esquema do <i>dataframe</i>	11
7.4.11	Salvando o <i>dataframe</i> tratado em formato CSV direto no <i>bucket GCP</i>	12
8.	Carregamento final	12
9.	Spark SQL e BigQuery.....	13
9.1	Pré – Pandemia.....	13
9.2	Pandemia	13
9.3	Pandemia (não_nulos).....	14
10.	Data Studio.....	15
10.1	Valor médio de aluguel no brasil 2018 / 2020.....	16
10.2	Valor médio do m² por anúncios de vendas de imóveis: 2018/2020.....	17
10.3	Quantidade de anúncios por tipo de imóvel 2018/2020.....	18
11.	Conclusão.....	19
12.	Bibliografia.....	20

1. INTRODUÇÃO

O presente relatório tem como intuito mostrar o processo de ETL (Extração, Transformação e Limpeza) de dois bancos de dados nos respectivos anos de 2018 e 2020 com informações do Setor Imobiliário Brasileiro. O trabalho será realizado como projeto final na conclusão do curso de Engenharia de Dados + Python na SoulCode Academy - Turma BC6.

2. ANÁLISE

A análise tem como proposta mostrar o cenário imobiliário brasileiro antes e durante a pandemia de COVID-19, com foco no impacto da quantidade de anúncios, valores e tipo de imóvel mais ofertados.

3. BASE DE DADOS

Seguindo os parâmetros do projeto, foram necessárias extrações com dois tipos diferentes de formato de dados, sendo um em CSV e outro em JSON. Ambos detêm origens distintas sendo o primeiro *dataframe* - '**properati-data-public (ano 2018)**' - originário do **BigQuery** (disponibilizado na plataforma google em sua base de dados públicos) e o segundo – '**dataZAP (ano 2020)**' - proveniente do site **Kaggle**.

4. INSIGHTS

Concluimos que os seguintes *insights* seriam fundamentais para conduzir as prospecções:

- Quantidade de anúncios de VENDA por Estado (UF) antes e durante a pandemia;
- Valor médio do ALUGUEL por Estado (UF) antes e durante a pandemia;

- Valor do m² por Estado (UF) nos anúncios de Vendas antes e durante a pandemia.

5. EXTRAÇÃO DE DADOS:



Google BigQuery

5.1 BigQuery

O *dataframe* '**properati-data-public**' possui formato *JSON*, com dados do ano de 2018 e foi extraído do **Google BigQuery** (disponibilizado na plataforma google em sua base de dados públicos).

5.2 Kaggle



O *dataframe* '**dataZAP**' em formato *CSV*, com dados do ano de 2020 foi extraído do site de banco de dados **Kaggle**.

6. ARMAZENAMENTO INICIAL



Google Cloud Storage

Após o download dos *dataframes*, ambos foram armazenados em seu formato bruto, em um **Data Lake (Bucket - Google Cloud Storage)** para as manipulações seguintes.

7. TRANSFORMAÇÃO

7.1 Colaboratory (Colab)



O Colaboratory ou "Colab" é um produto do Google Research, área de pesquisas científicas do Google. O Colab permite que qualquer pessoa escreva e execute código *Python* pelo navegador e é especialmente adequado para *machine learning*, análise de dados e educação.



7.2 Python

Python é uma linguagem de programação com propósito geral usada bastante em *data science*. Através dela faremos as manipulações em **Pandas e PySpark**.



7.3 Pandas

Os *dataframes* foram extraídos do *Bucket* para realização da primeira manipulação na Biblioteca *PANDAS* através do *Colab*.

Abaixo os códigos e manipulações utilizados para a transformação (normalização) dos dados no dataframe '**dataZAP**':

7.3.1 Extração do *dataframe* direto do *Bucket*

```
1 pip install gcsfs
```

```
1 import os
2 from google.cloud import storage
3 serviceAccount = '/content/Chave_Ingestao_Apache.json'
4 os.environ['GOOGLE_APPLICATION_CREDENTIALS'] = serviceAccount
```

7.3.2 Instalação da Biblioteca *Pandas* no *Colab*

```
1 import pandas as pd
```

7.3.3 Leitura do *dataframe* no *Colab*

```
1 df = pd.read_csv('gs://projeto-final-grupo09/entrada_dados/dataZAP.csv', sep=";")
2 df.head(1)
```

	account.licenseNumber	account.name	imvl_type	listing.acceptExchange	listing.address.city
0	04268-J-SP	ADI Assessoria e Imóveis Ltda	apartamentos	False	São Paulo

7.3.4 Backup do dataframe antes das normalizações

```
1 dfback = df
```

7.3.5 Exclusão (*Drop*) das Colunas

```
1 df.drop(['listing.acceptExchange', 'listing.address.confidence'], axis=1, inplace = True)
2
```

7.3.6 Renomeando as colunas (Tradução)

```
1 df.rename(columns={'account.licenseNumber': 'numero.licença'}, inplace = True)
```

7.3.7 Verificando valores nulos

```
1 df.isnull().sum()
```

7.3.8 Verificando os tipos de imóvel no dataframe

```
1 sorted(pd.unique(df['tipo_imovel']))

['apartamentos',
 'casas',
 'casas-de-condominio',
 'casas-de-vila',
 'cobertura',
 'flat',
 'quitinetes',
 'studio']
```

7.3.9 Renomeando linhas (tradução)

```
2 df['residencial_comercial'].replace('RESIDENTIAL', 'Residencial', inplace=True)
3 df['residencial_comercial'].replace('RESIDENTIAL|COMMERCIAL', 'Comercial', inplace=True)
4 print(df['residencial_comercial'].unique())
```

7.3.10 Transformando o formato da data

```
2 df['dt_criacao_lista'] = pd.to_datetime(df['dt_criacao_lista'])
```

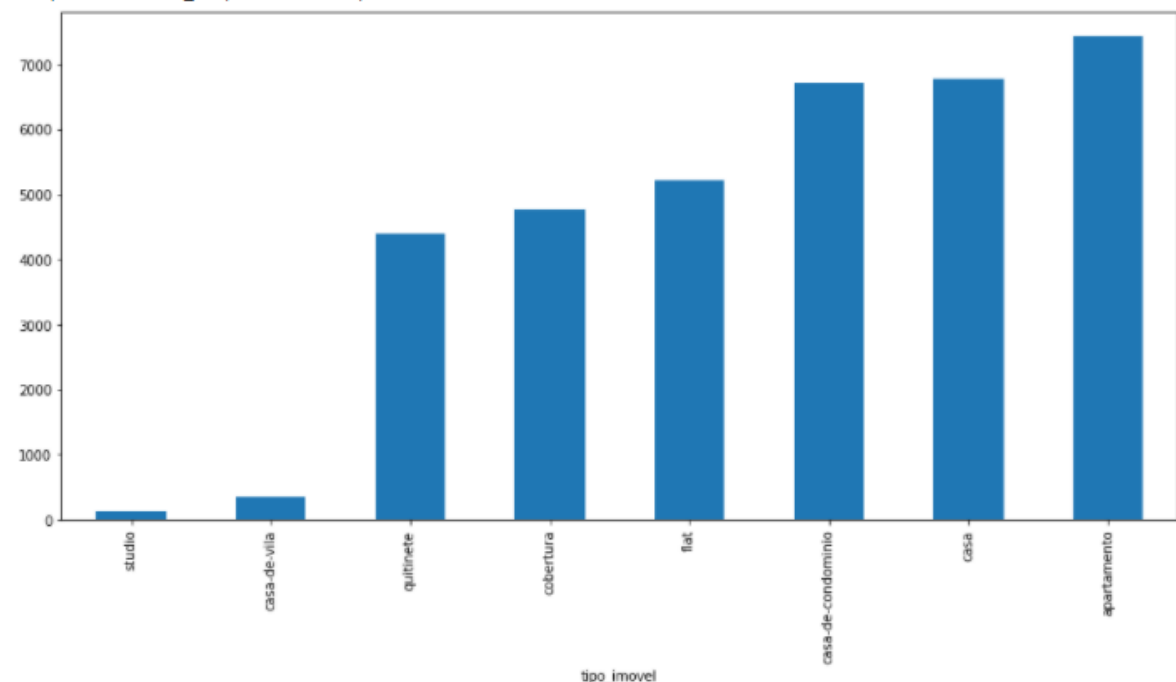
```
1 df.dt_criacao_lista.apply(lambda x: x.date())
```

```
1 df['data'] = df['dt_criacao_lista'].apply(lambda x: x.strftime('%Y-%m-%d'))
```

7.3.11 Plotagem (Comparando o tipo de imóvel mais anunciado no dataframe)

```
1 df.groupby(['tipo_imovel'])\
2 .tipo_imovel \
3 .count()\
4 .sort_values()\
5 .plot(kind='bar', figsize=(15,7),label='tipo_imovel ')
```

<matplotlib.axes._subplots.AxesSubplot at 0x7fa0d6f7cb90>



As manipulações foram realizadas para o melhor aproveitamento dos dados, filtrando e direcionando os nossos *insights*. Com a finalização do tratamento em Pandas os *dataframes* foram armazenados na pasta ‘Saída’ no *Bucket-GCP* do projeto.

7.3.12 *Dataframe* sendo salvo diretamente em pasta específica(saída) do *bucket*.

```
1 client = storage.Client()
2 bucket = client.get_bucket('projeto-final-grupo09')
3
4 bucket.blob('saida_dados/Projeto_Final_ZAP_Pandas').upload_from_string(df.to_csv(index=False), 'text/csv')
```

7.4 PySpark



PySpark é a interface de alto nível que permite que você consiga acessar e usar o *Spark* por meio da linguagem *Python*. Usando-o, é possível escrever o código utilizando apenas a própria linguagem.

No projeto, o *PySpark* foi utilizado para manipular o dataframe, *properati-data-public*, no *Colab* do seguinte modo:

7.4.1 Instalação do PySpark e da biblioteca necessária para conexão com a Google Cloud Platform

```
1 pip install gcsfs
```

```
1 pip install pyspark
```

7.4.2 Importação das bibliotecas e funções necessárias para análise

(Obs.: Foi importada a biblioteca *Pandas* para fazer a atribuição posterior dos dados ao dataframe.)

```
1 import pandas as pd
2 from pyspark.sql import SparkSession
3 import pyspark.sql.functions as F
4 from pyspark.sql.window import Window
5 from pyspark.sql.functions import monotonically_increasing_id
6 import os
7 from google.cloud import storage
```

7.4.3 Início da conexão com a GCP

```
1 serviceaccount = '/content/chave_acesso_edlaine'
2 os.environ["GOOGLE_APPLICATION_CREDENTIALS"] = serviceaccount
```

7.4.4 Iniciar sessão *Spark*

```
1 spark = SparkSession.builder\
2     .master('local')\
3     .appName('ProjetoEngDaods')\
4     .config('spark.ui.port', '4050')\
5     .getOrCreate()
6 spark
```

7.4.5 Lendo/abrindo o *dataframe* que será manipulado

```
1 dfspark = pd.read_csv('gs://projeto-final-grupo09/saida_dados/ProjetoFinalPandas02(csv)')
2 df = spark.createDataFrame(dfspark)
```

7.4.6 Visualização do *dataframe*

```
1 df.show(10)
```

Unnamed: 0	identificacao	criado_em	tipo_propriedade	nome_do_local	estado	preco
0	47c47b29f4b5d901e...	2018-01-24	casa	Bahia	Bahia	577083.67
1	0916e4dea826443b2...	2018-01-22	casa	Bahia	Bahia	248742.97
2	e206166a672764f56...	2018-01-09	loja	Pará	Pará	NaN
3	6986e2ddcb8c24d9a...	2018-01-09	apartamento	Pará	Pará	650960.32
4	a7de4dc532374b2c3...	2018-01-09	apartamento	Pará	Pará	220325.02
5	e37003e790accd231...	2017-09-05	casa	Amapá	Amapá	125882.81
6	78664e248ec3e5038...	2017-08-30	casa	Ceará	Ceará	858324.85
7	88cf7471558753502...	2017-09-27	casa	Paraná	Paraná	290428.43
8	763d3fc12045ec194...	2017-09-27	casa	Paraná	Paraná	400590.96
9	1a7b284bbcd0da8b5...	2017-09-27	casa	Paraná	Paraná	450664.83

7.4.7 Verifica se há diferença entre as colunas em questão e quantos são diferentes

```
1 df.select('estado').filter(df.nome_do_local != df.estado).count()
```

```
413290
```

7.4.8 Verifica os valores iguais entre as colunas em questão

```
1 df.select('estado').filter(df.nome_do_local == df.estado).count()
```

```
4394
```

7.4.9 Visualização das ocorrências dos tipos de propriedades e seu número de ocorrências presentes no *dataframe*

```
1 df.groupBy('tipo_propriedade').count().show(10)
```

```
+-----+-----+
|tipo_propriedade| count|
+-----+-----+
|      apartamento|273577|
|             casa|135213|
|             loja|  8894|
+-----+-----+
```

7.4.10 Visualização do esquema do *dataframe*

```
1 df.printSchema()
```

```
root
|-- tipo_propriedade: string (nullable = true)
|-- estado: string (nullable = true)
|-- preco: double (nullable = true)
|-- area_construcao_em_m2: double (nullable = true)
|-- quartos: double (nullable = true)
|-- area_total_por_m2: double (nullable = true)
|-- data_criacao: string (nullable = true)
|-- local_imovel: string (nullable = true)
|-- Index: long (nullable = true)
|-- ID: string (nullable = true)
|-- preco_m2: double (nullable = true)
```

7.4.11 Salva o *dataframe* tratado em formato CSV direto no *bucket - GCP*

```

1 client = storage.Client()
2 bucket = client.get_bucket('projeto-final-grupo09')
3
4 bucket.blob('saida_dados/ProjetoFinalPySpark02.csv').upload_from_string(df.toPandas().to_csv(index=False), 'text/csv')

```

8. CARREGAMENTO FINAL

No final do tratamento com as tecnologias citadas anteriormente, foi feito o armazenamento dos arquivos em nuvem, como solicitado na descrição do projeto. Este carregamento foi feito com o próprio PySpark, utilizando suas funções.

← Bucket details

projeto-final-grupo09

Location: southamerica-east1 (São Paulo) | Storage class: Standard | Public access: ⚠ Subject to object ACLs | Protection: None

OBJECTS | CONFIGURATION | PERMISSIONS | PROTECTION | LIFECYCLE

Buckets > projeto-final-grupo09 > saida_dados

UPLOAD FILES | UPLOAD FOLDER | CREATE FOLDER | MANAGE HOLDS | DOWNLOAD | DELETE

Filter by name prefix only ▾ | Filter Filter objects and folders | Show deleted data

<input type="checkbox"/>	Name	Size	Type	Created ?	Storage class	Last modified	Public access ?	Version	
<input type="checkbox"/>	ProjetoFinalPandas02(csv)	50.2 MB	application/octet-stream	Nov 19, 2...	Standard	Nov 19, 20...	Not public	—	↓ ⋮
<input type="checkbox"/>	ProjetoFinalPySpark02.csv	50.2 MB	text/csv	Nov 20, 2...	Standard	Nov 20, 20...	Not public	—	↓ ⋮
<input type="checkbox"/>	Projeto_Final_ZAP_Pandas	5.7 MB	text/csv	Nov 23, 2...	Standard	Nov 23, 20...	Not public	—	↓ ⋮
<input type="checkbox"/>	Projeto_Final_ZAP_Pyspark.csv	5.9 MB	text/csv	Nov 23, 2...	Standard	Nov 23, 20...	Not public	—	↓ ⋮

9. SPARK SQL E BIGQUERY



As manipulações em *Spark SQL* foram realizadas em ambiente *BigQuery*. As pesquisas desenvolvidas neste ponto determinaram os **insights** para a conclusão do projeto.

9.1 Pré - Pandemia

Foram pontuados:

- Média de Preços de compra por estado entre os períodos de 2017 a 2018;
- Média de preço do m² por estado entre os períodos de 2017 a 2018;
- Número de anúncios por estado entre os anos de 2017 e 2018.

The screenshot shows a SQL editor with three tabs: 'EDITOR', 'CONSUL...', and 'CONSUL...'. The active tab is 'CONSUL...', displaying three SQL queries. The first query calculates the average purchase price by state for 2017-2018. The second query calculates the average price per square meter (m²) by state for 2017-2018, filtering for prices greater than 1. The third query counts the number of occurrences by state for 2017-2018, ordered by descending count. The interface includes a sidebar with icons for search, view, and other functions, and a top bar with options like 'FEATURES & INFO', 'SHORTCUT', and 'DISABLE EDITOR TABS'.

```

1  -- Média de Preços de compra por estado no período de 2017 - 2018:
2  SELECT estado,
3  AVG(preco) AS Media_Precio_Compra
4  FROM setor_imobiliario_insights.dados_bq
5  GROUP BY (estado)
6  ORDER BY (estado);
7
8  -- Média de preço do M² por estado no período entre 2017 - 2018:
9  SELECT estado,
10 AVG(preco_m2) AS Media_Precio_M2
11 FROM setor_imobiliario_insights.dados_bq
12 WHERE preco_m2 > 1
13 GROUP BY (estado)
14 ORDER BY (estado) ASC ;
15
16 -- Número de ocorrencias por estado entre 2017 - 2018:
17 SELECT estado,
18 COUNT(estado) AS Ocorrencias_Estados
19 FROM setor_imobiliario_insights.dados_bq
20 GROUP BY (estado)
21 ORDER BY (Ocorrencias_Estados) DESC;

```

9.2 Pandemia

Na manipulação do *dataframe* referente ao ano de 2020, foram necessárias algumas manipulações de separações nos dados, com intuito de obtermos um melhor aproveitamento e leitura das informações.

Foram trabalhadas na manipulação:

- Números nulos na coluna 'valor_vendas' (Alguns imóveis não possuíam os seus preços de aluguel e venda, por isso a distinção foi necessária);
- Números de não nulos na coluna 'valor_vendas';
- Criação de *dataframe* (*setor_imobiliario_insights.nao_nulos*) para comparação de valores com *strings* formatadas;
- Transformação da coluna 'valor_venda' para o formato *float*;
- Separação entre anos.

```

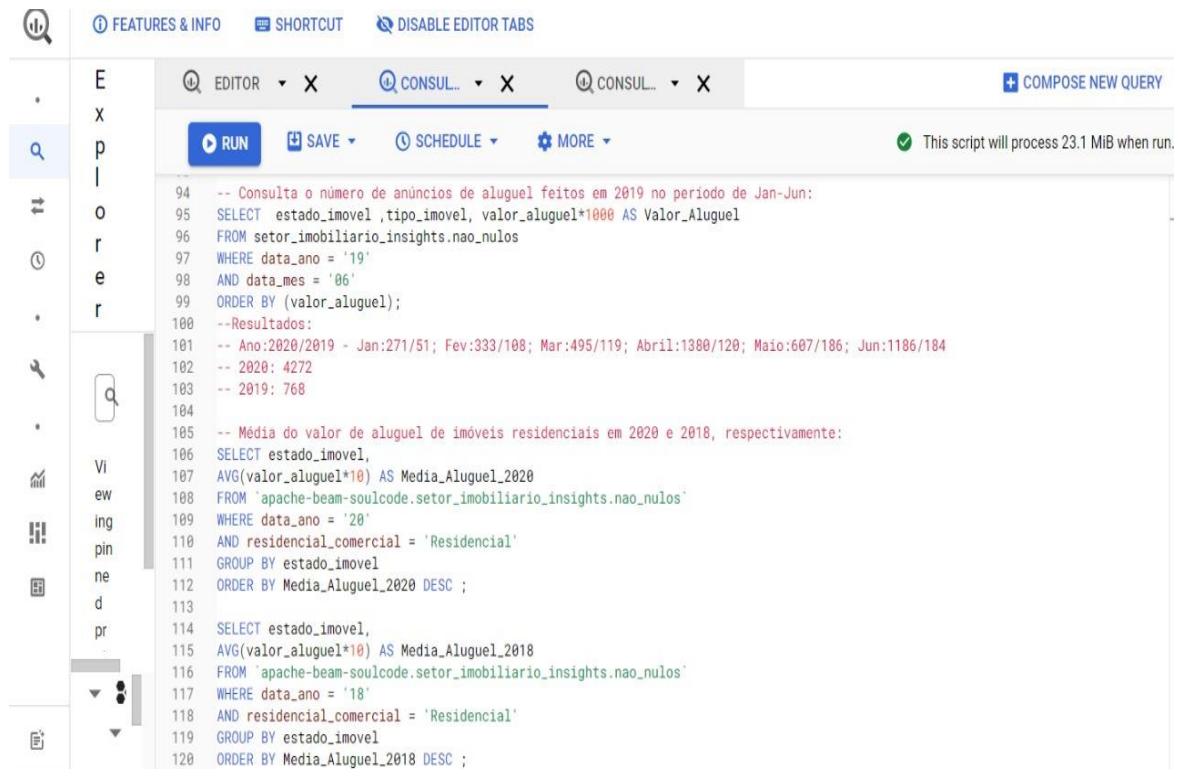
1  -- Número de nulos em valor_vendas: 26129
2  SELECT *
3  FROM setor_imobiliario_insights.data_zap
4  WHERE valor_venda IS NULL;
5
6  -- Número de NÃO nulos em valor_vendas: 9553
7  SELECT *
8  FROM setor_imobiliario_insights.data_zap
9  WHERE valor_venda IS NOT NULL;
10
11 -- Criação dataset para comparação de valores com strings formatadas:
12 CREATE OR REPLACE TABLE setor_imobiliario_insights.nao_nulos AS (
13 SELECT * EXCEPT (valor_venda), SUBSTRING(valor_venda, 0, 5) AS valor_venda
14 FROM setor_imobiliario_insights.data_zap
15 WHERE valor_venda is not null
16 );
17
18 -- Transformação da coluna 'valor_venda' para float:
19 CREATE OR REPLACE TABLE setor_imobiliario_insights.nao_nulos AS (
20 SELECT * EXCEPT(valor_venda), CAST(valor_venda AS FLOAT64) AS valor_venda
21 FROM setor_imobiliario_insights.nao_nulos
22 );
23
24 -- Separação entre ano:
25 CREATE OR REPLACE TABLE setor_imobiliario_insights.nao_nulos AS (
26 SELECT *, FORMAT_DATE("%y", data) AS data_ano,
27 FROM setor_imobiliario_insights.nao_nulos
28 );

```

9.3 Pandemia (não_nulos)

Com o *dataframe* (*setor_imobiliario_insights.nao.nulos*) e sua formatação mais específica para foco do projeto as seguintes pesquisas foram realizadas:

- Número de anúncios de aluguel feitos no ano de 2019 entre os meses de Janeiro e Junho;
- Média do valor de aluguel de imóveis residenciais nos anos 2020 e 2018.



The screenshot shows the Google Data Studio SQL Editor interface. The left sidebar contains navigation icons for Explorer, Search, Recent, and Views. The main editor area displays two SQL queries. The first query (lines 94-104) filters data for the year 2019 and month 06, ordering by rental value. The second query (lines 106-120) calculates the average rental value for residential properties in 2020 and 2018, grouped by state. The results of the first query are displayed below the code, showing monthly rental values for 2019 and the total for 2020.

```

94 -- Consulta o número de anúncios de aluguel feitos em 2019 no período de Jan-Jun:
95 SELECT estado_imovel, tipo_imovel, valor_aluguel*1000 AS Valor_Aluguel
96 FROM setor_imobiliario_insights. nao_nulos
97 WHERE data_ano = '19'
98 AND data_mes = '06'
99 ORDER BY (valor_aluguel);
100 --Resultados:
101 -- Ano:2020/2019 - Jan:271/51; Fev:333/108; Mar:495/119; Abril:1380/120; Maio:607/186; Jun:1186/184
102 -- 2020: 4272
103 -- 2019: 768
104
105 -- Média do valor de aluguel de imóveis residenciais em 2020 e 2018, respectivamente:
106 SELECT estado_imovel,
107 AVG(valor_aluguel*10) AS Media_Aluguel_2020
108 FROM `apache-beam-soulcode.setor_imobiliario_insights.nao_nulos`
109 WHERE data_ano = '20'
110 AND residencial_comercial = 'Residencial'
111 GROUP BY estado_imovel
112 ORDER BY Media_Aluguel_2020 DESC ;
113
114 SELECT estado_imovel,
115 AVG(valor_aluguel*10) AS Media_Aluguel_2018
116 FROM `apache-beam-soulcode.setor_imobiliario_insights.nao_nulos`
117 WHERE data_ano = '18'
118 AND residencial_comercial = 'Residencial'
119 GROUP BY estado_imovel
120 ORDER BY Media_Aluguel_2018 DESC ;

```

10. DATA STUDIO



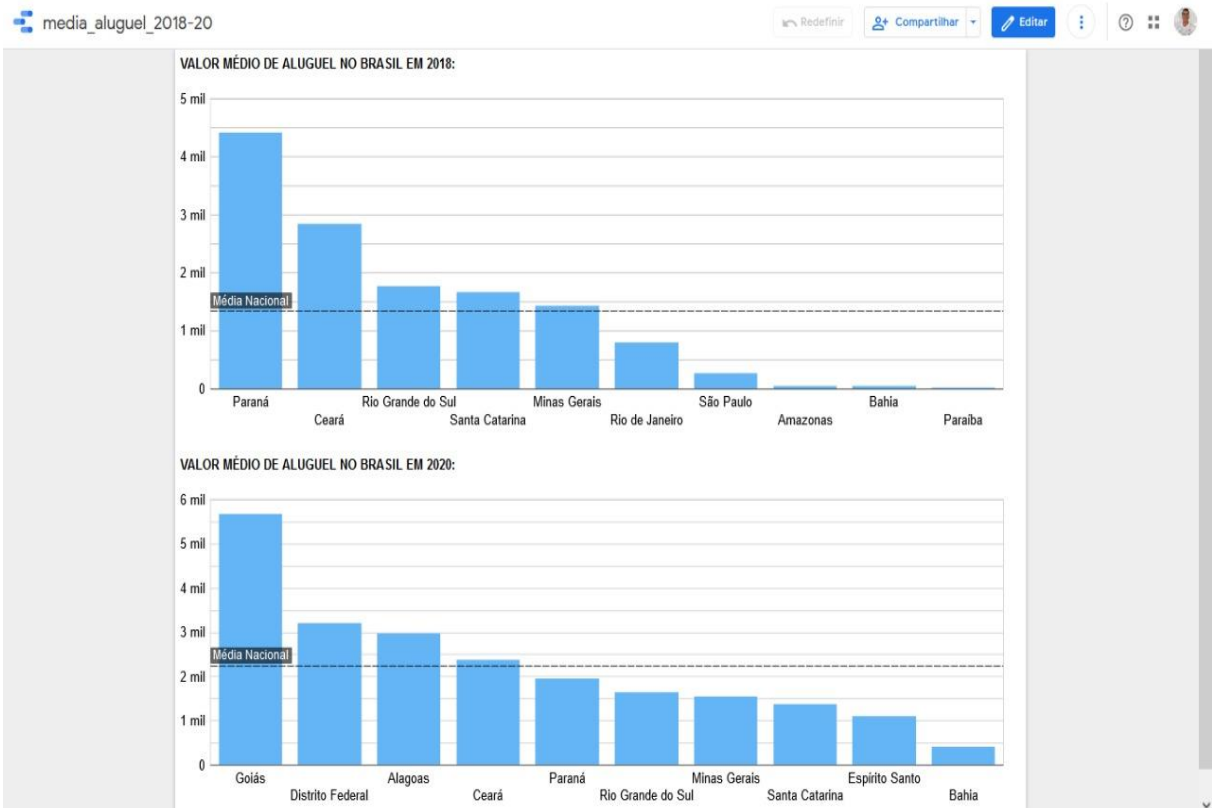
O Google Data Studio é uma ferramenta do Google que permite criar um dashboard personalizado.

Após conclusão dos *insights* que fundamentam o projeto através do *BigQuery*, segue como parte obrigatória a exibição dos dados no *Data Studio*.

Elencados abaixo, seguem de forma gráfica os insights que nortearam o projeto:

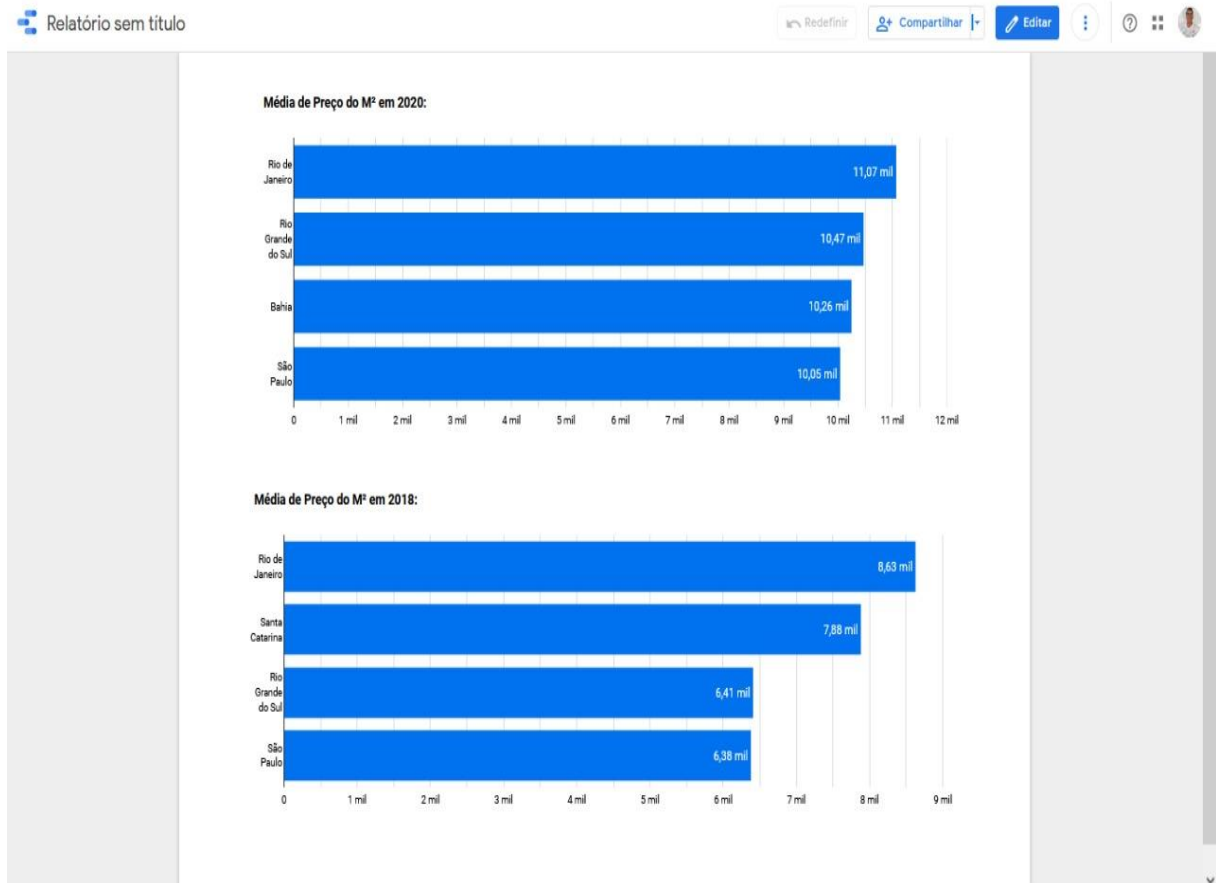
10.1 Valor médio de aluguel no brasil 2018 / 2020

Ranking dos 10 estados com maior valor médio:

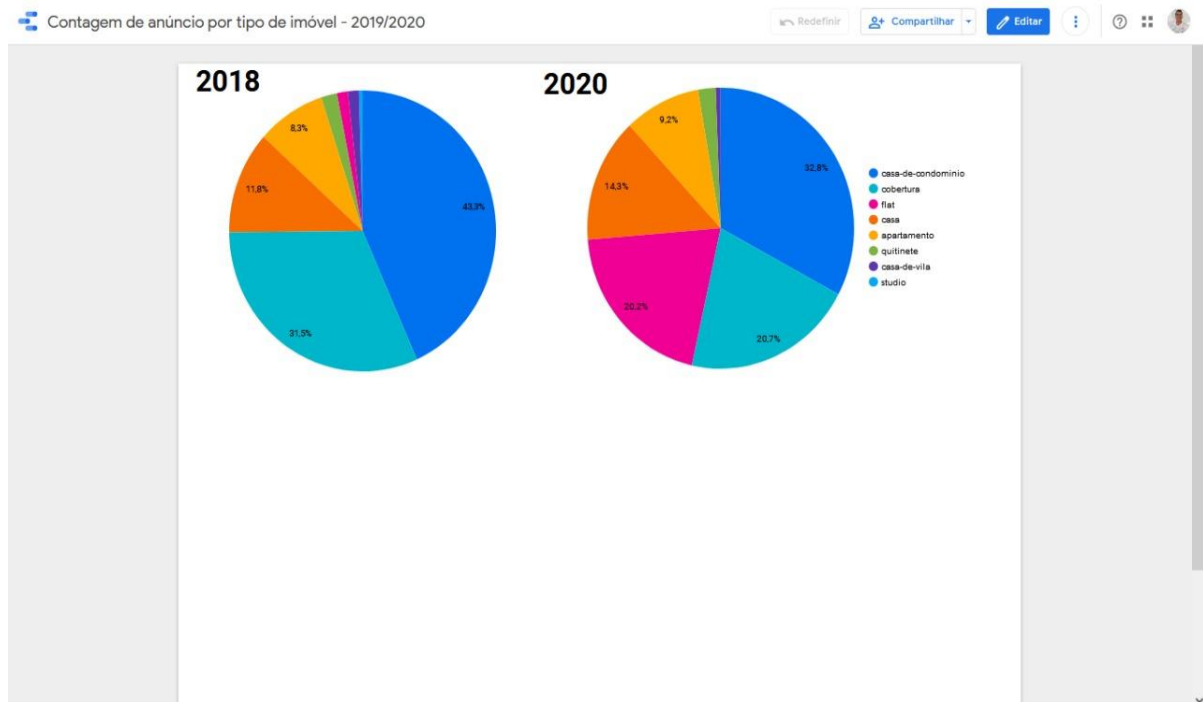


10.2 Valor médio do m² por anúncios de vendas de imóveis 2018 / 2020

Ranking dos 4 estados com maior valor por m²:



10.3 Quantidade de anúncios por tipo de imóvel 2018/2020



11. CONCLUSÃO

A lapidação dos dataframes, através do processo de ETL, forneceu uma amostragem de dados comparativos que facilitaram o norteamento das pesquisas, como também a fundamentação do projeto.

As ferramentas, bibliotecas e linguagens que foram utilizadas para trabalhar os bancos de dados exibiram de forma clara as mudanças nos perfis de anúncios e valores que o setor imobiliário brasileiro enfrentou com a pandemia da Covid -19.

Conseguimos pontuar que:

- O valor médio do preço nos aluguéis de imóveis no Brasil em 2020 sofreu alta superior a 40% se comparado ao ano de 2018.
- O preço médio de vendas por m² de imóveis também obteve uma alta em seus valores em 2020, com variações entre 28 e 63%.
- No tipo de imóvel ofertado observou-se o aumento nos tipos flat, casa de condomínio e studio no ano de 2020. Tipos que geralmente são utilizados como investimentos na área de hospedagem.

- Após toda a análise realizada, chegamos à conclusão que o setor imobiliário brasileiro sofreu aumentos significativos na áreas de venda e locação de imóveis, além da criação de uma nova vertente de oferta em propriedades com altos custos de manutenção no ano de 2020, essas mudanças ocorreram por desdobramentos da pandemia de Covid-19.

12. BIBLIOGRAFIA

HARVE (Brasil). Python para que serve: top 5 utilidades: Python para que serve? Python é uma linguagem Open-Source de propósito geral usado bastante em data science, machine learning, desenvolvimento de web, desenvolvimento de aplicativos, automação de scripts, fintechs e mais.. [S. l.], 2021. Disponível em: <https://harve.com.br/blog/programacao-python-blog/python-para-que-serve-top-5-utilidades/>. Acesso em: 24 nov. 2021.

COLABORATORY (org.). Perguntas frequentes: Noções básicas. <https://research.google.com/colaboratory/intl/pt-BR/faq.html>. [S. l.]: Google, 2020. Disponível em: <https://referenciabibliografica.net/a/pt-br/ref/abnt>. Acesso em: 26 nov. 2021.

GOOGLE (org.). Colaboratory: Noções básicas. [S. l.], 2020. Disponível em: <https://referenciabibliografica.net/a/pt-br/ref/abnt>. Acesso em: 27 nov. 2021.

COM SCHOOL (org.). O que é Google Data Studio? ComSchool Explica: Google Data Studio: Entenda a importância desta ferramenta. [S. l.], 29 nov. 2021. Disponível em: <https://news.comschool.com.br/o-que-e-google-data-studio-comschool-explica/>. Acesso em: 26 nov. 2021.