

TP1- US Covid Data Wharehouse

Pedro Pita, 19933

Tomás Ramos, 19934

INSTITUTO POLITÉCNICO DE BEJA
Escola Superior de Tecnologia e Gestão
Licenciatura em Engenharia Informática

TP1- US Covid Data Wharehouse

Elaborado por:

Pedro Pita, 19933

Tomás Ramos, 19934

Orientado por:

Isabel Brito

Relatório do trabalho prático 1 da unidade curricular Sistemas de Informação na
Escola Superior de Tecnologia e Gestão do Instituto Politécnico de Beja

Índice

1. Introdução.....	3
2. Caso de estudo.....	4
2.1. Tema.....	4
2.2. Fonte de dados.....	4
3. OLAP - Modelo Data Warehouse	5
3.1. Esquema escolhido.....	5
3.2. Tabelas de dimensão e factos	5
3.3. Modelo Multidimensional.....	7
4. Processo de ETL.....	8
4.1. Extract	8
4.2. Transform	8
4.3. Loading	10
5. OLAP – Cube e Analise de dados.....	11
5.1. Casos	12
5.2. Mortes	14
5.3. Hospital	16
5.4. Nível de risco	17
5.5. Testes	18
5.6. Vacinas	19
5.7. Nível de transmissão	20
6. Conclusões.....	21
Bibliografia	22

Lista de Figuras

Figura 1-Modelo Multidimensional.....	7
Figura 2- Representação da camada semântica	11
Figura 3- Comparação de número de casos em Outubro de 2020 e Outubro de 2021.....	12
Figura 4-Número de casos diários no país, com foco no dia 28 de janeiro de 2021	12
Figura 5- Número de casos diários, no país, clicando no mês que pretendemos.....	12
Figura 6- Número de casos totais por mês no país.....	13
Figura 7- Número de casos totais nos 15 estados mais afetados do país.....	13
Figura 8-Comparação dos casos no Alaska, Rhode Island e Puerto Rico no mês de abril de 2020 e abril de 2021.....	14
Figura 9-Número de mortes totais no país com foco no dia 15 de abril de 2021	14
Figura 10- Número de mortes por dia no país, com foco no dia 22 de junho de 2020	14
Figura 11- 20 Estados do país com mais mortes por Covid-19	15
Figura 12 - Quantidade de hospitalizados por dia no país.....	16
Figura 13- Comparação entre a capacidade hospitalar e o número de hospitalizados.....	16
Figura 14-Total dos níveis de risco dos estados do país	17
Figura 15- Número de testes negativos diários no país.....	18
Figura 16- Número total de testes positivos no país	18
Figura 17- Comparação do número total de testes negativos e positivos no país	18
Figura 18- Estados com mais vacinas administradas	19
Figura 19- Comparação de vacinas distribuídas e vacinação completa no país	19
Figura 20- Número de pessoas com vacinação completa no país	19
Figura 21- Estados com maior nível de transmissão	20

1. Introdução

Este trabalho tem como objetivo encontrar uma fonte de dados e fazer todo o processo de ETL (Extract – Transform – Loading), por forma a desenvolver e modelar um modelo multidimensional (Data Warehouse).

Desta forma, o relatório que se segue explica todo o processo de planeamento e realização do projeto. Assim sendo, apresenta-se dividido em quatro partes.

Primeiro, “Caso de Estudo”, onde é abordado o tema escolhido e a fonte de dados; o “Modelo data warehouse”, que expõe toda a estrutura da nossa data warehouse; depois, o “Processo ETL”, no qual explicamos todo o processo de ETL que realizamos, desde a extração dos dados até a inserção dos mesmos na nossa base de dados; e por fim, numa última e quarta parte, a “Cube e Analise de dados” onde apresentamos a ferramenta de análise de dados que utilizamos e os gráficos obtidos.

Todos os dados, scripts, prints, etc... utilizados ao longo do projeto, estão juntamente com os ficheiros do relatório e no [GitHub](#).

2. Caso de estudo

2.1. Tema

Para a realização deste trabalho decidimos utilizar o **Tema A- Covid 19** proposta pela docente da unidade curricular.

Inicialmente pensamos utilizar dados relacionados com a COVID-19 em Portugal, mas visto que é um tema muito utilizado, então decidimos “emigrar” para os Estados Unidos e procurar fontes de dados relacionados com o tema.

2.2. Fonte de dados

Durante a pesquisa por fontes de dados, encontramos uma API que nos permite obter dados diários nos vários estados desde o início da pandemia até ao momento, sendo ela a [“Covid Act Now”](#).

Esta API não só contém a contagem diária de casos e mortes, como também tem dados acerca da lotação das camas nos hospitais, níveis de transmissão, níveis de risco, doses de vacinação administradas, entre outros dados.

3. OLAP - Modelo Data Warehouse

3.1. Esquema escolhido

Ao nosso ver o esquema que mais se adequa a nossa Data Warehouse é o “star schema”, apesar de este ocupar mais espaço de armazenamento comparando a outros esquemas, como o snowflake, o star schema tem vantagens no que toca a consultar os dados, visto as relações entre as tabelas não serem tão complexas, ligando apenas as tabelas de dimensão com a tabela de factos.

3.2. Tabelas de dimensão e factos

Decidimos que a nossa data warehouse iria conter apenas 4 tabelas de dimensão e 1 de factos.

- **fact_informations:** A tabela “fact_informations” vai ser a nossa tabela de factos, esta tabela vai ser o centro de armazenamento de toda a informação, estando ligada as 4 tabelas de dimensão. A tabela vai ser constituída não só pelos dados ocorridos como também pelas chaves correspondentes as dimensões, a chave que vai identificar a nossa tabela de facto será uma chave composta pelas chaves estrangeiras das tabelas de dimensão.
 - **dateKeyFK, stateCodeFK, transmissionLevelFK, ruiskLevelFK:** Estas colunas correspondem a chave composta.
 - **currentCases:** Quantidade de casos totais;
 - **currentDeaths:** Quantidade de mortes totais;
 - **newCases:** Quantidade de novos casos;
 - **newDeaths:** Quantidade de novas mortes;
 - **contactTracers:** Quantidade de cadeias de transmissão;
 - **testsPositives:** Quantidade de testes positivos;
 - **testsNegatives:** Quantidade de testes negativos;
 - **vaccinesDistributed:** Quantidade de vacinas distribuídas
 - **vaccinesAdministered:** Quantidade de vacinas administradas;
 - **vaccinesCompleted:** Quantidade de vacinações completas;
 - **hospitalCapacity:** Capacidade de camas nos hospitais;
 - **hospitalCurrentUsageTotal:** Quantidade de camas ocupadas nos hospitais;
 - **hospitalCurrentUsageCovid:** Quantidade de camas ocupadas nos hospitais por doentes de covid.
- **dim_risk_level:** Este nível é atribuído conforme a quantidade de casos diários a cada 100 mil habitantes.
 - **level:** Esta coluna corresponde a chave primaria, sendo que o valor é o nível correspondente ao risco, numa escala de 0 a 5;
 - **description:** Descrição do nível, por exemplo, 0 = “Low”, 1 = “Medium”, ...

- **dim_transmission_level:** Este nível é atribuído conforme a dificuldade de encontrar relação dos casos confirmados e suspeitos, ou seja, quantos mais casos confirmados sem relação a outros casos, maior é este nível.
 - **level:** Esta coluna corresponde a chave primaria, sendo que o valor é o nível correspondente ao risco de transmissão, numa escala de 0 a 4;
 - **description:** Descrição do nível, por exemplo, 0= “Low”, 1 = “Moderate”, ...
- **dim_date:** A tabela “dim_date” vai ser uma tabela de dimensão, onde cada registo identifica um dia do ano e todos os dados relacionados com aquele dia.
 - **dateKey:** Esta coluna corresponde a chave primaria, sendo que o valor é atribuído removendo os hifenes da data correspondente, por exemplo, a dataKey correspondente a data “2021-10-09” será “20211009”;
 - **date:** Data correspondente, por exemplo, “2021-10-09”;
 - **year:** Ano correspondente, por exemplo, “2021”;
 - **month:** Número do mês correspondente, por exemplo, “10”;
 - **monthName:** Nome do mês correspondente, por exemplo, “October”;
 - **dayOfMonth:** Número correspondente ao dia do mês, por exemplo, “9”;
 - **dayOfWeek:** Número do dia da semana de 0 a 6, por exemplo, Sábado corresponde ao número “5”;
 - **dayOfWeekName:** Nome do dia da semana, por exemplo, “Saturday”.
- **dim_state:** A tabela “dim_state” vai ser uma tabela de dimensão, onde cada registo identifica um estado do país e todos os dados correspondentes a esse estado.
 - **stateCode:** Esta coluna corresponde a chave primaria, sendo que o valor é a abreviatura do nome do estado, por exemplo, o estado “Alaska” tem o código “AK”;
 - **name:** Esta coluna corresponde ao nome por extenso do estado, por exemplo, “Alaska”;
 - **fips:** Esta coluna corresponde ao número federal de cada estado, por exemplo, número 2 corresponde ao “Alaska”;
 - **population:** Corresponde ao número total de habitantes no estado correspondente, por exemplo, “Alaska” tem 731545 habitantes;
 - **countryCode:** Corresponde ao código de cada país, ou seja, Estados Unidos corresponde a EUA;
 - **countryName:** Nome por extenso do país, por exemplo, United States.

3.3. Modelo Multidimensional

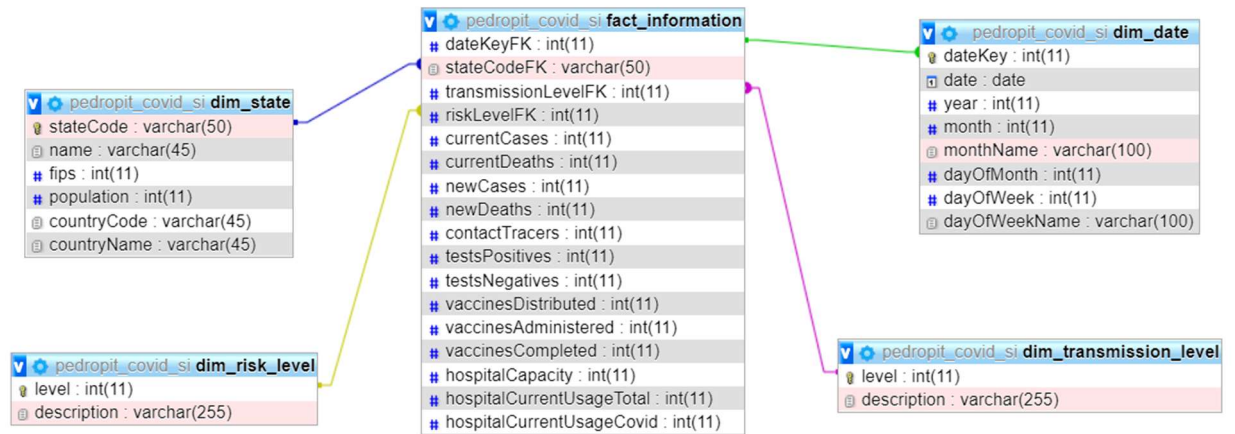


Figura 1-Modelo Multidimensional

4. Processo de ETL

4.1. Extract

No site da API temos várias formas de fazer pedido de dados, no caso, pretendemos obter todos os dados diários disponíveis desde o início da pandemia nos vários estados dos EUA.

Para obter esses dados, apenas precisasse fazer um GET request ao seguinte link:

<https://api.covidactnow.org/v2/states.timeseries.json?apiKey=8ffcb17422284eef928291e8043167e9>

Os dados vêm em formato JSON, com o seguinte formato:

Ligação ao ficheiro no nosso Github -> [clique aqui](#)

4.2. Transform

Agora que temos milhares de dados impossíveis de serem lidos sem tratamento, temos de os transformar numa estrutura que permita o loading na nossa data warehouse.

Para isso, decidimos fazer 3 scripts em JavaScript (clique no nome do ficheiro para redirecionar para o github):

- **datesTratment.js**
 - **Script:** [GitHub](#)
 - **Processo:** Este ficheiro é responsável por ler o ficheiro de dados “brutos”, verificar qual é a data de início e a data fim dos dados disponíveis, percorrer todas as datas entre a data início e a data fim e adicionar essas mesmas datas num ficheiro JSON com a estrutura da tabela “dim_date”.
 - **Estrutura:**

```
{
  "dates": [
    {
      "dateKey": int,
      "date": date,
      "year": int,
      "month": int,
      "monthName": string,
      "dayOfMonth": int,
      "dayOfWeek": int,
      "dayName": string
    },
    {
      "dateKey": int,
      "date": date,
      "year": int,
      "month": int,
      "monthName": string,
      "dayOfMonth": int,
      "dayOfWeek": int,
      "dayName": string
    }
  ]
},
{
  "dates": [
    {
      "dateKey": "20200121",
      "date": "2020-01-21",
      "year": 2020,
      "month": 1,
      "monthName": "January",
      "dayOfMonth": 21,
      "dayOfWeek": 2,
      "dayOfWeekName": "Tuesday"
    },
    {
      "dateKey": "20200122",
      "date": "2020-01-22",
      "year": 2020,
      "month": 1,
      "monthName": "January",
      "dayOfMonth": 22,
      "dayOfWeek": 3,
      "dayOfWeekName": "Wednesday"
    }
  ],
  {
```

- **Saída:** [GitHub](#)

- **statesTreatment.js**

- **Script:** [GitHub](#)

- **Processo:** Este Script é responsável por percorrer o ficheiro de dados “brutos” e recolher todos os dados relacionados com os estados.

Além dos dados do ficheiro, também foi necessário utilizar um array com todos os nomes dos estados por extenso, de forma a obter o nome do estado a partir da abreviatura, visto que o ficheiro de dados não continha o nome de cada estado. [Link do array.](#)

- **Estrutura:**



```

{
  "states": [
    {
      "stateCode": string,
      "name": string,
      "fips": int,
      "population": int,
      "countryCode": string,
      "countryName": string
    },
    {
      "stateCode": string,
      "name": string,
      "fips": int,
      "population": int,
      "countryCode": string,
      "countryName": string
    }
  ]
}

{
  "states": [
    {
      "stateCode": "AK",
      "name": "Alaska",
      "fips": "02",
      "population": 731545,
      "countryCode": "USA",
      "countryName": "United States"
    },
    {
      "stateCode": "AL",
      "name": "Alabama",
      "fips": "01",
      "population": 4903185,
      "countryCode": "USA",
      "countryName": "United States"
    }
  ]
}

```

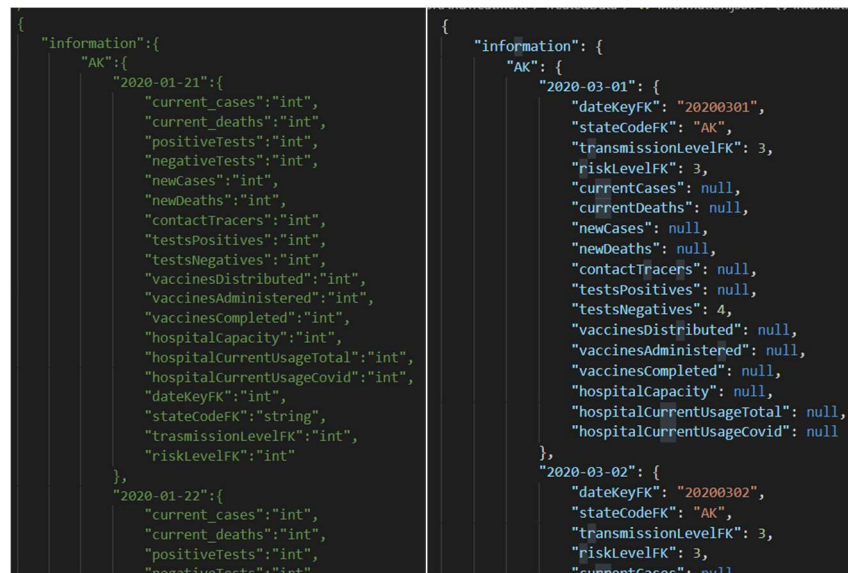
- **Saída:** [GitHub](#)

- **informationTreatment.js**

- **Script:** [GitHub](#)

- **Processo:** Este Script percorre todos os dados em “bruto” e recolhe todos os dados necessários para a nossa data warehouse, organizando os dados no ficheiro de saída por estado e depois por dia.

- **Estrutura:**



```

{
  "information": {
    "AK": {
      "2020-01-21": {
        "current_cases": "int",
        "current_deaths": "int",
        "positiveTests": "int",
        "negativeTests": "int",
        "newCases": "int",
        "newDeaths": "int",
        "contactTracers": "int",
        "testsPositives": "int",
        "testsNegatives": "int",
        "vaccinesDistributed": "int",
        "vaccinesAdministered": "int",
        "vaccinesCompleted": "int",
        "hospitalCapacity": "int",
        "hospitalCurrentUsageTotal": "int",
        "hospitalCurrentUsageCovid": "int",
        "dateKeyFK": "int",
        "stateCodeFK": "string",
        "transmissionLevelFK": "int",
        "riskLevelFK": "int"
      },
      "2020-01-22": {
        "current_cases": "int",
        "current_deaths": "int",
        "positiveTests": "int",
        "negativeTests": "int"
      }
    }
  }
}

{
  "information": {
    "AK": {
      "2020-03-01": {
        "dateKeyFK": "20200301",
        "stateCodeFK": "AK",
        "transmissionLevelFK": 3,
        "riskLevelFK": 3,
        "currentCases": null,
        "currentDeaths": null,
        "newCases": null,
        "newDeaths": null,
        "contactTracers": null,
        "testsPositives": null,
        "testsNegatives": 4,
        "vaccinesDistributed": null,
        "vaccinesAdministered": null,
        "vaccinesCompleted": null,
        "hospitalCapacity": null,
        "hospitalCurrentUsageTotal": null,
        "hospitalCurrentUsageCovid": null
      },
      "2020-03-02": {
        "dateKeyFK": "20200302",
        "stateCodeFK": "AK",
        "transmissionLevelFK": 3,
        "riskLevelFK": 3,
        "currentCases": null
      }
    }
  }
}

```

- **Saída:** [GitHub](#)

- **Níveis de transmissão e de risco:** Verificamos os níveis de transmissão e níveis de risco a partir do website da API e fizemos um JSON com os níveis e respetiva descrição, não necessitando de qualquer Script. ([Link do site oficial](#))

- **Saída:** [GitHub](#) “[Níveis de risco](#)” e “[Níveis de transmissão](#)”

4.3. Loading

De forma a fazer loading de todos os dados em JSON na nossa base de dados, criamos um projeto em PHP.

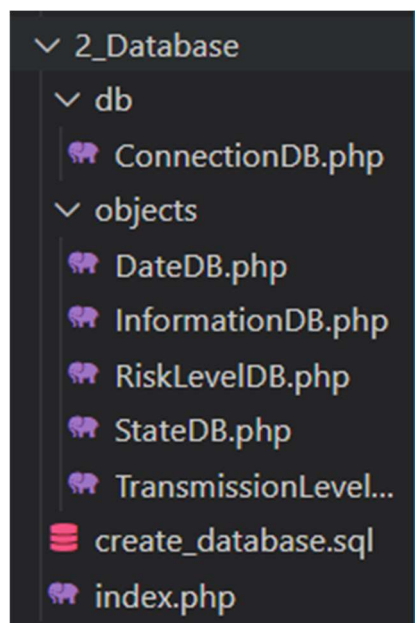
A hierarquia deste projeto é constituída por ([GitHub](#)):

index.php – neste ficheiro é instanciado todos os objetos e ficheiros necessários;

db/ConnectionDB.php – Esta Classe é responsável por fazer a conexão a base de dados;

objects/*.php – Cada classe deste diretório corresponde a 1 tabela na base de dados, contendo todas as funções correspondentes a essa tabela;

create_database.sql– Ficheiro SQL que contem toda a estrutura da base de dados.



Com esta estrutura se pretendermos inserir dados em alguma das tabelas, apenas temos de carregar o ficheiro JSON, instanciar o objeto correspondente a tabela e chamar a função “insert” passando os dados do ficheiro JSON como parâmetro.

```
$dates = json_decode(file_get_contents('../1_DataExportAndTreatment/TreatedData/dates.json', true));  
$dateDB = new DateDB($conn);  
$dateDB->insert($dates->dates);
```

Após os dados estarem totalmente carregados na base de dados, conseguimos observar que na tabela “dim_date” foram inseridos um total de 666 registos, na tabela “dim_state” foram inseridos 53 registos, na tabela “dim_risk_level” foram inseridos 6 registos, na tabela “dim_transmission_level” foram inseridos 5 registos e na tabela “fact_information” foram inseridos 33301 registos.

5. OLAP – Cube e Análise de dados

Inicialmente tivemos a ideia de desenvolver um site de análise de dados, onde pensamos em contruir o cube utilizando a plataforma [Cube.JS](#) e construir os gráficos em javascript, mas infelizmente a adaptação a plataforma não correu bem e fomos a procura de outros recursos.

Foi aí que procuramos e testamos algumas outras soluções, tais o [Olap Cube Writter](#) e [ICCube](#). Também vimos outras soluções como o Excel ou até mesmo funções de exportar os dados em cube no T-SQL.

A solução que mais nos agradou e que achamos mais adequada para o problema em questão foi a do ICCube, este software permite nos conectar diretamente com a nossa data warehouse, criando automaticamente todas as dimensões e alguns measures de exemplo, sendo que somos completamente livres de editar a estrutura, ao contrário do que acontece no “Olap Cube Writter” que tem um limite de 3 measures na versão grátis.

Além disso este software também nos permite fazer toda a análise de dados sem sair do programa, tendo vários gráficos disponíveis.

Neste documento iremos deixar alguns exemplos de gráficos gerados através da plataforma utilizando os dados da data warehouse, também adicionamos outros gráficos, podendo ser consultados no [GitHub](#).

Ao clicar nos dados do gráfico, este apresenta o valor respectivo, e se clicarmos no mês, o gráfico apresenta os dias do mês.

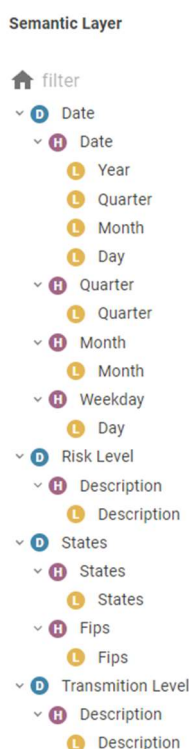


Figura 2- Representação da camada semântica

5.1. Casos

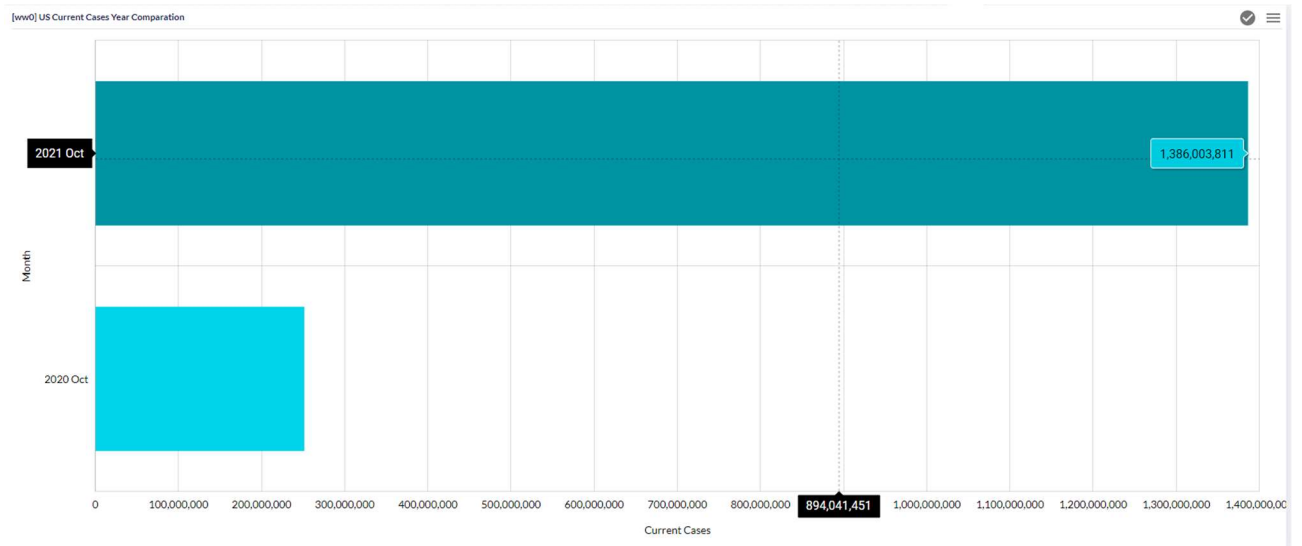


Figura 3- Comparação de número de casos em Outubro de 2020 e Outubro de 2021



Figura 4-Número de casos diários no país, com foco no dia 28 de janeiro de 2021

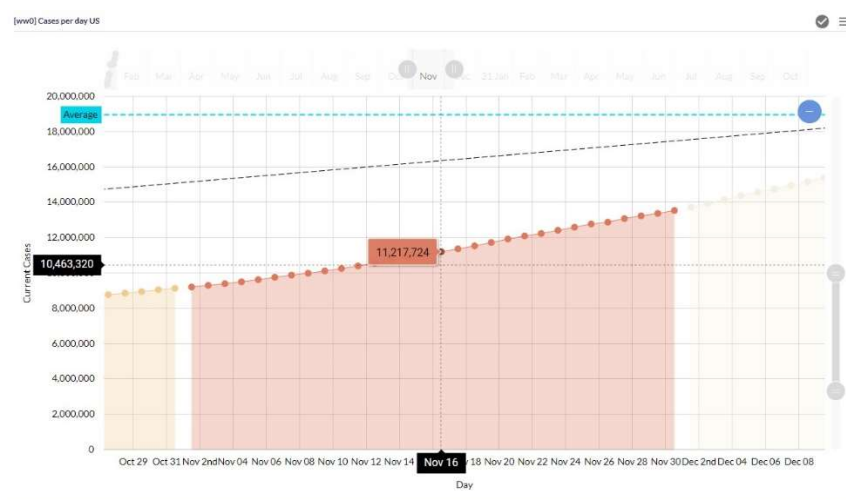


Figura 5- Número de casos diários, no país, clicando no mês que pretendemos

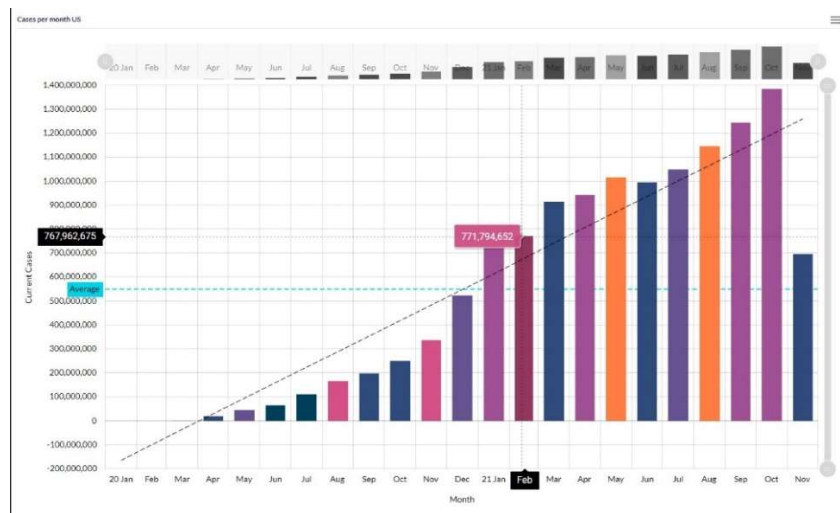


Figura 6- Número de casos totais por mês no país

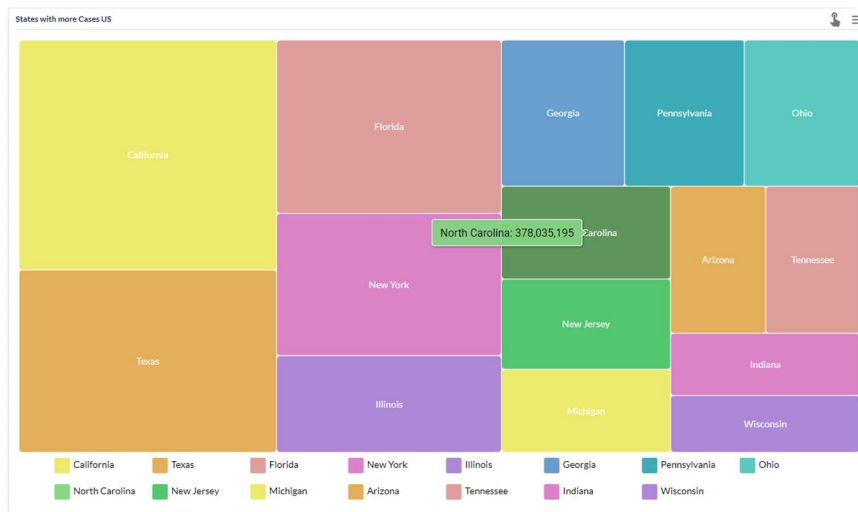


Figura 7- Número de casos totais nos 15 estados mais afetados do país

5.2. Mortes

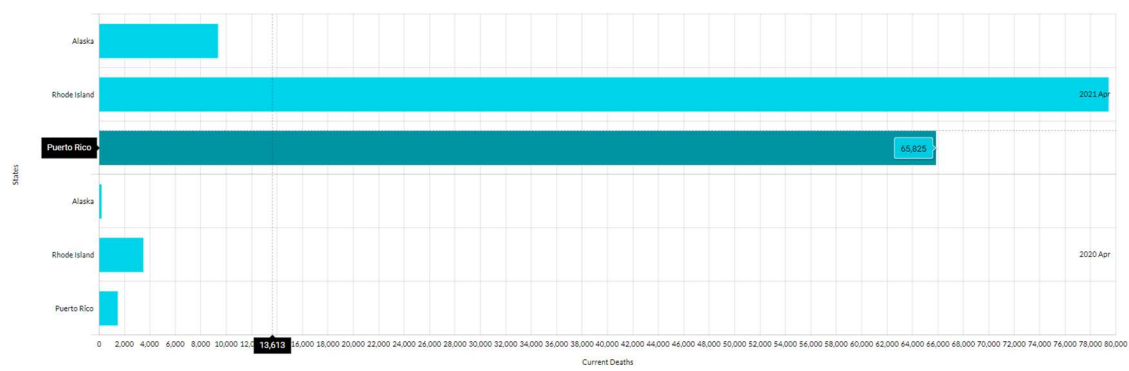


Figura 8-Comparação dos casos no Alaska, Rhode Island e Puerto Rico no mês de abril de 2020 e abril de 2021

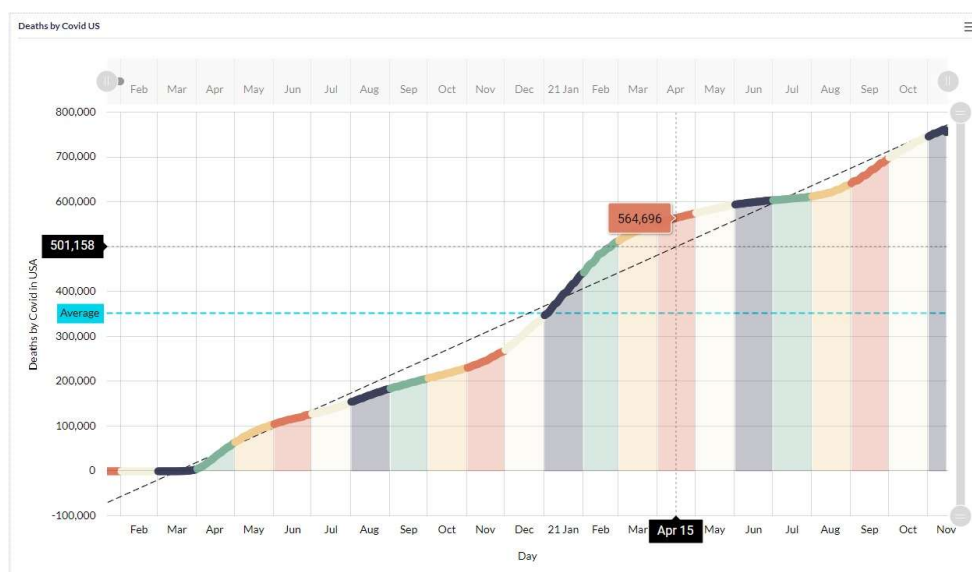


Figura 9-Número de mortes totais no país com foco no dia 15 de abril de 2021

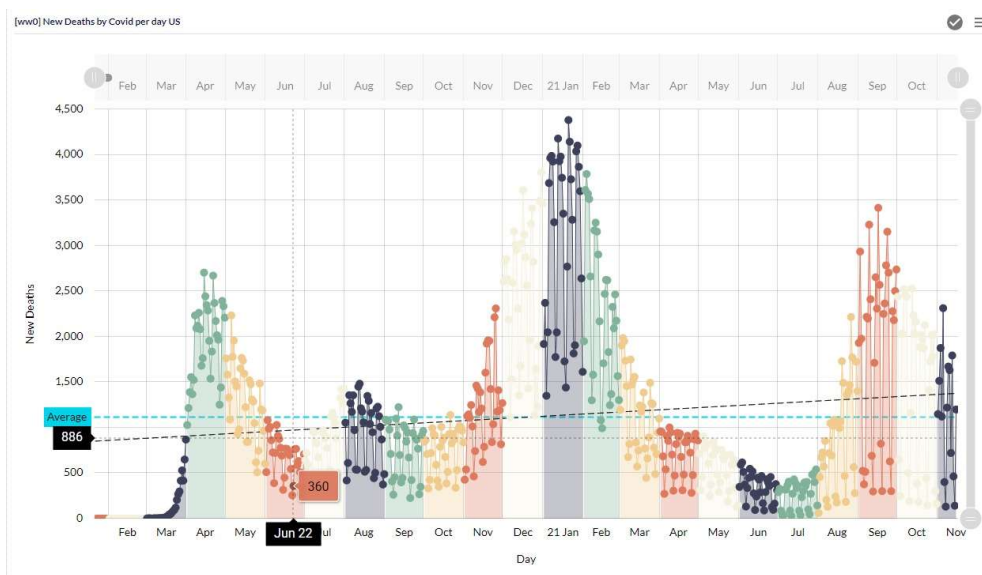


Figura 10- Número de mortes por dia no país, com foco no dia 22 de junho de 2020

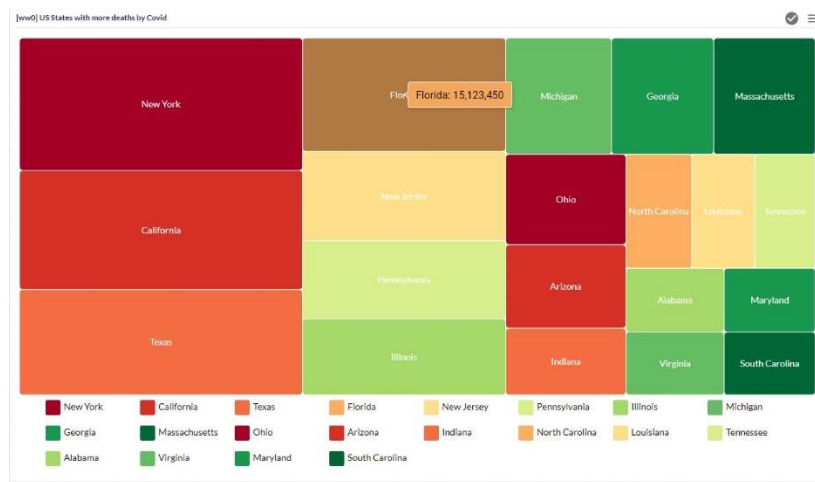


Figura 11- 20 Estados do país com mais mortes por Covid-19

5.3. Hospital

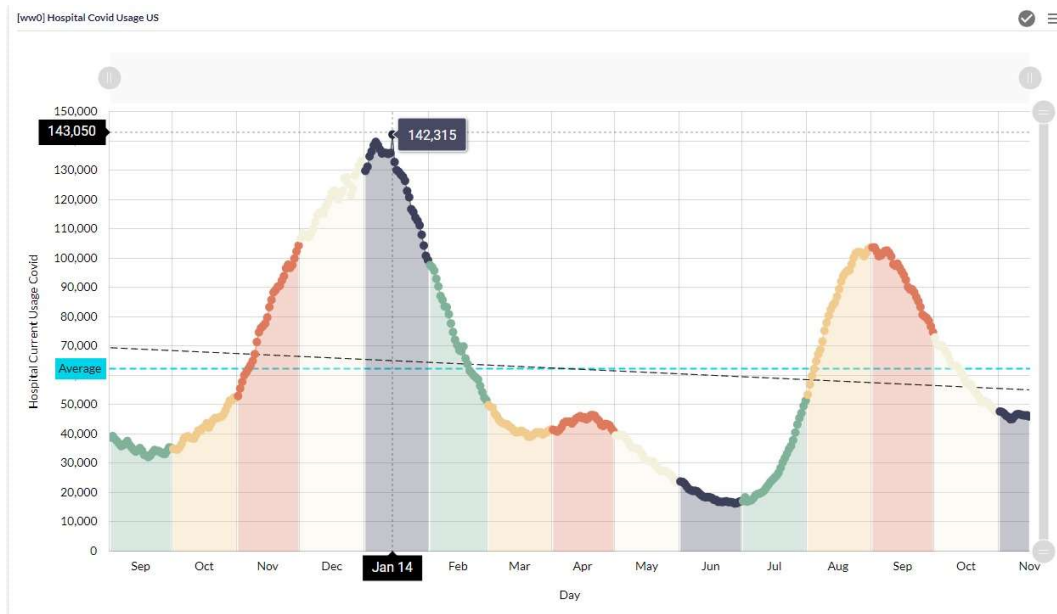


Figura 12 - Quantidade de hospitalizados por dia no país

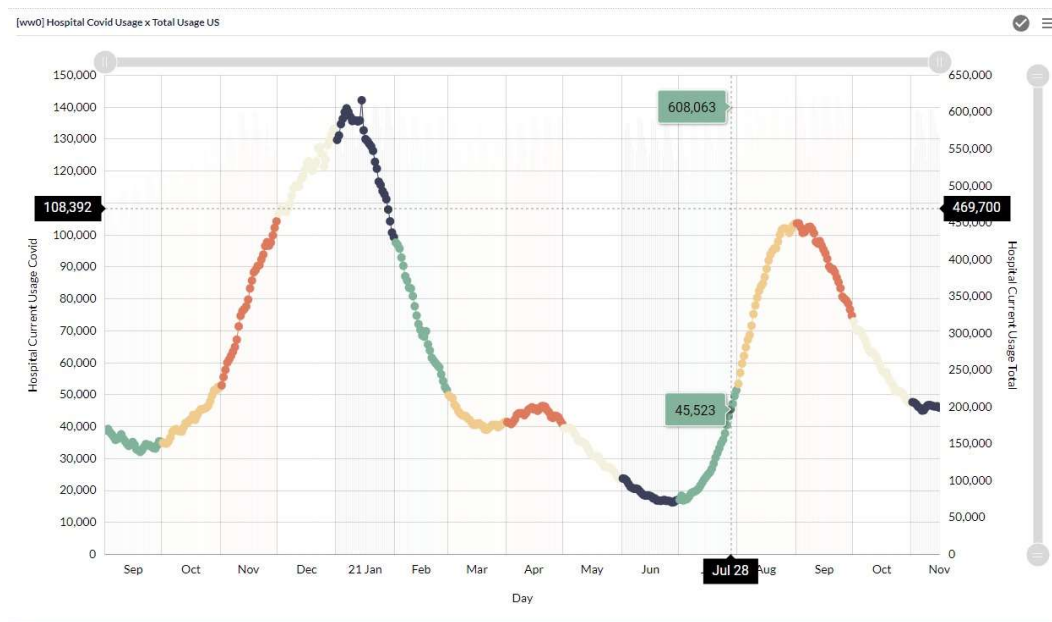


Figura 13- Comparação entre a capacidade hospitalar e o número de hospitalizados

5.4. Nível de risco

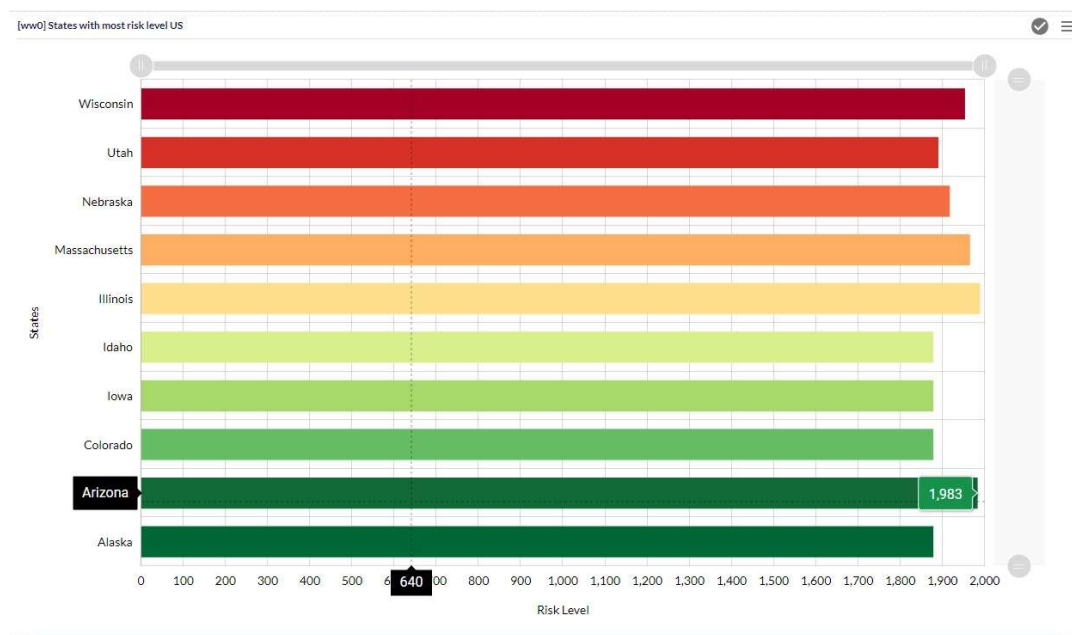


Figura 14-Total dos níveis de risco dos estados do país

5.5. Testes

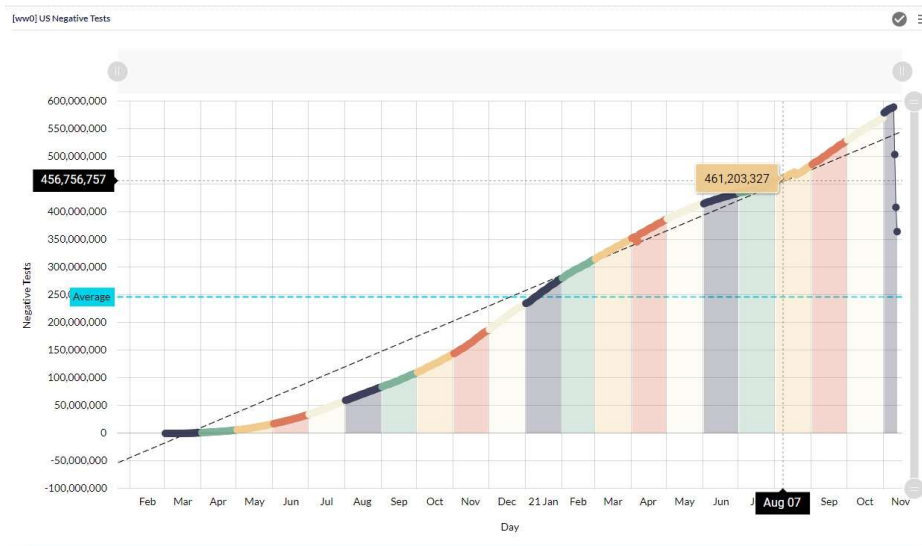


Figura 15- Número de testes negativos diários no país



Figura 16- Número total de testes positivos no país

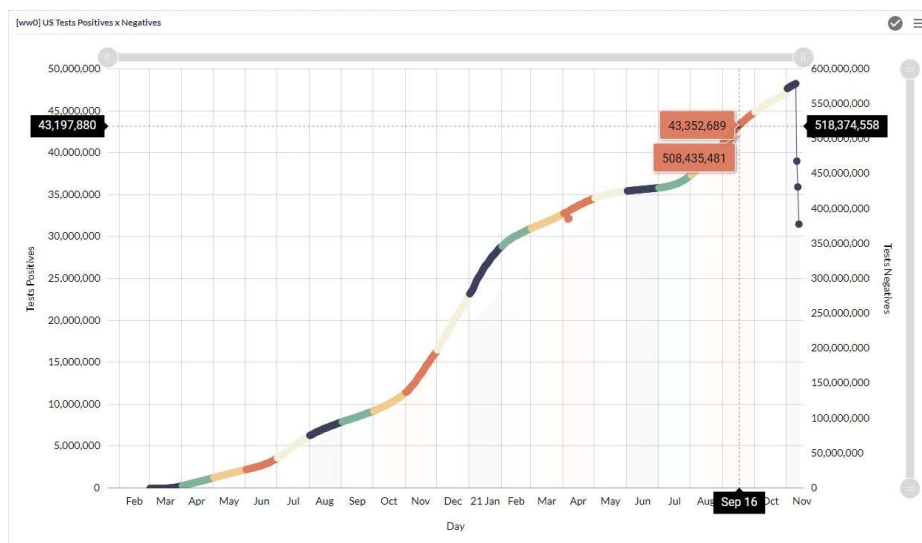


Figura 17- Comparação do número total de testes negativos e positivos no país

5.6. Vacinas

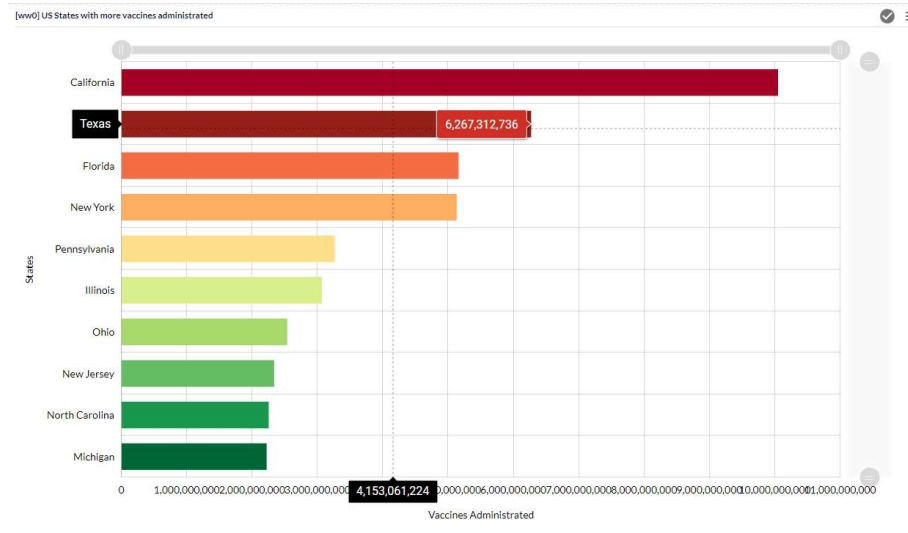


Figura 18- Estados com mais vacinas administradas

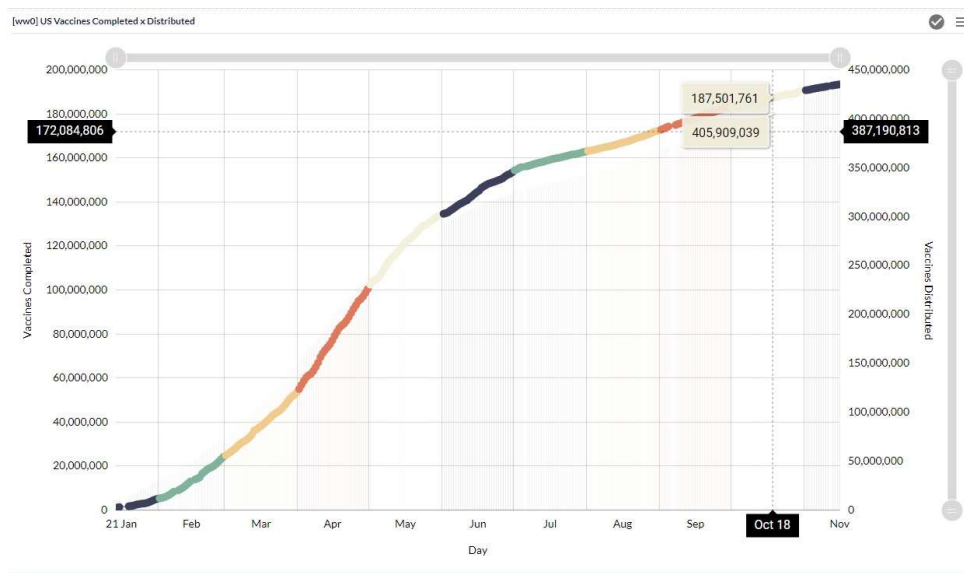


Figura 19- Comparação de vacinas distribuídas e vacinação completa no país

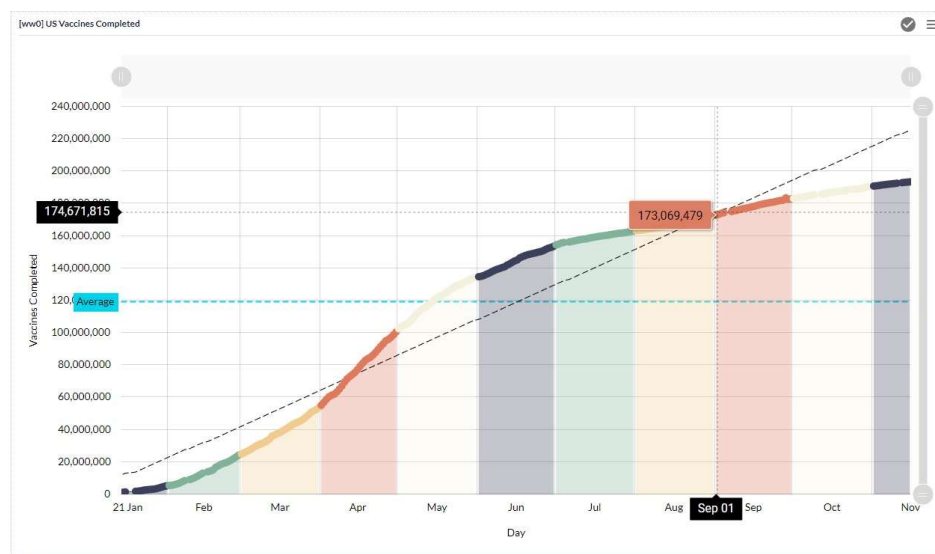


Figura 20- Número de pessoas com vacinação completa no país

5.7. Nível de transmissão

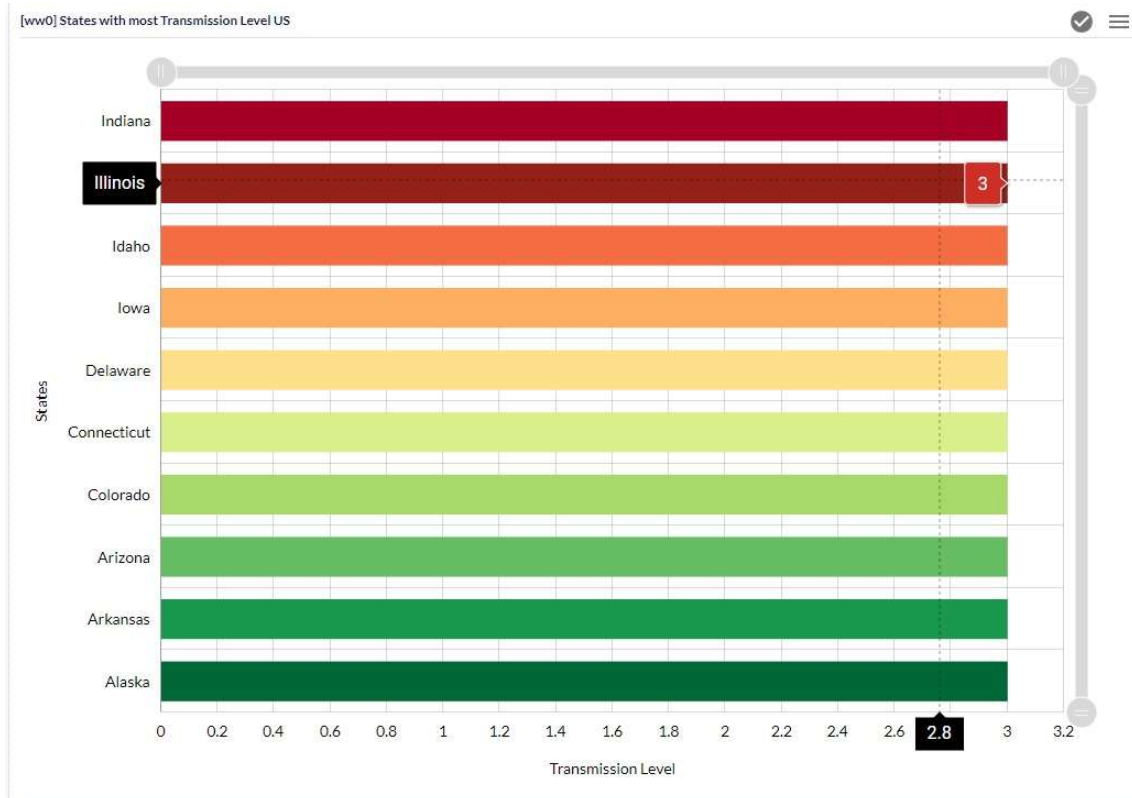


Figura 21- Estados com maior nível de transmissão

6. Conclusões

Este trabalho, inserido na UC de Sistemas de Informação, apresentou-se bastante desafiante. Desta forma, exigiu um grande sentido de organização entre nós. Mesmo assim, ao nosso ver, o resultado foi bem conseguido. Foram cumpridas todas as etapas estipuladas no enunciado, tal como presumido.

Os gráficos obtidos parecem ir de encontro ao pretendido, apesar de que o “transmission level” e o “Risk level” não nos parece muito corretos, visto que os dados não variam muito ao longo do tempo.

Bibliografia

API DOCS. (02 de 12 de 2021). Obtido de <https://apidocs.covidactnow.org>

Array dos estados. (02 de Dezembro de 2021). Obtido de <https://gist.github.com/iamjason/8f8f4bc00c13de86bcad>

GitHub. (02 de Dezembro de 2021). Obtido de <https://www.github.com>

ICCube – Software usado para analise de dados . (02 de Dezembro de 2021). Obtido de <https://www.iccube.com>

Implementing a Data Warehouse with SQL Server, 01, Design and Implement Dimensions and Fact Tables. (02 de Dezembro de 2021). Obtido de <https://www.youtube.com/watch?v=StoWu2A8Ufs>

PHP Data Objects. (02 de Dezembro de 2021). Obtido de <https://www.php.net/manual/en/book.pdo.php>

Top 10 Best OLAP Tools. (02 de Dezembro de 2021). Obtido de <https://www.softwaretestinghelp.com/best-olap-tools/>

Understand OLAP, MDX and Business. (02 de Dezembro de 2021). Obtido de <https://www.youtube.com/watch?v=yoE6bgJv08E>