

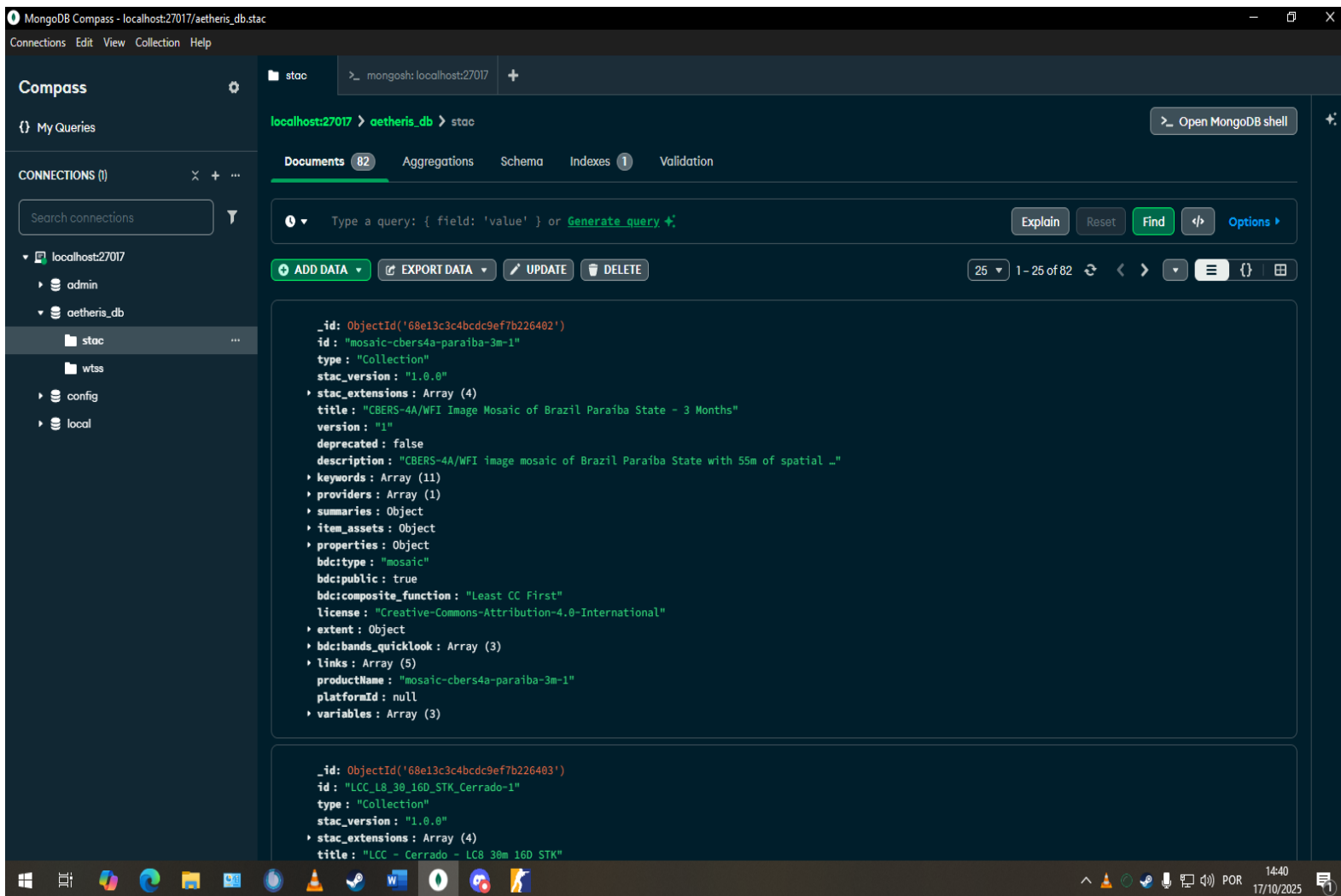
Requisito BDN.01 – Modelagem de Dados – Grupo Capydev – Projeto Aetheris

Objetivo

Demonstrar a capacidade de **modelar dados no MongoDB**, aplicando corretamente as estratégias de **embedding** e **referencing**, justificando as escolhas e validando a modelagem com inserções e consultas.

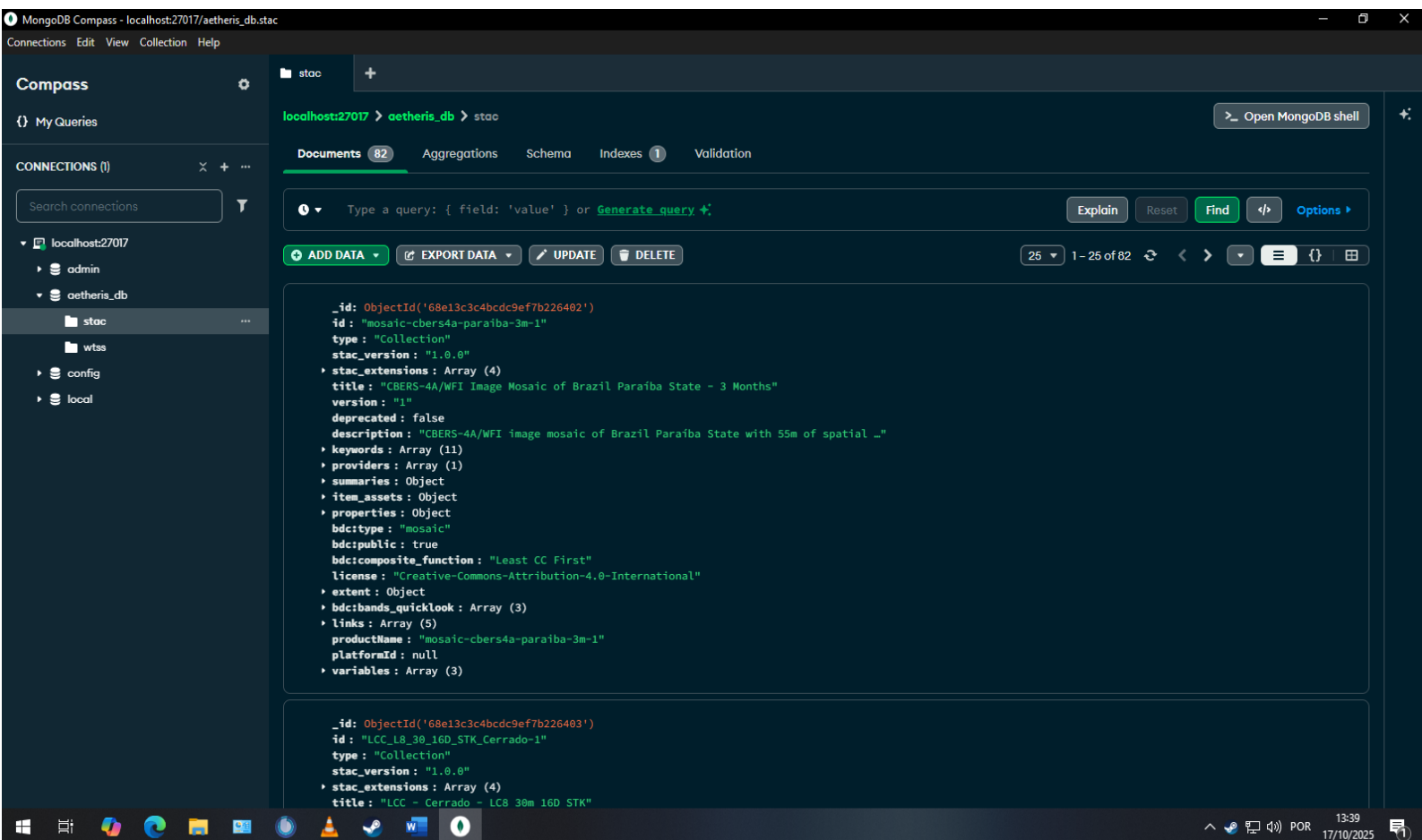
1. Criação do Banco de Dados

- Nome do banco escolhido pelo grupo (relacionado ao tema da ABP).
 - “aetheris_db”.
- Estrutura inicial das coleções (nomes e finalidade).
 - **STAC** – Armazena os dados das coleções da API STAC, servindo como catálogo de informações;
 - **WTSS** – Armazena os dados das coleções da API WTSS, providenciando valores de séries temporais.



2. Modelagem de Relacionamentos

- Exemplo de **embedding (documentos incorporados)**: pelo menos 1 coleção deve usar essa estratégia.



3. Scripts MongoDB

- Inserção de documentos nas coleções (insertOne, insertMany).

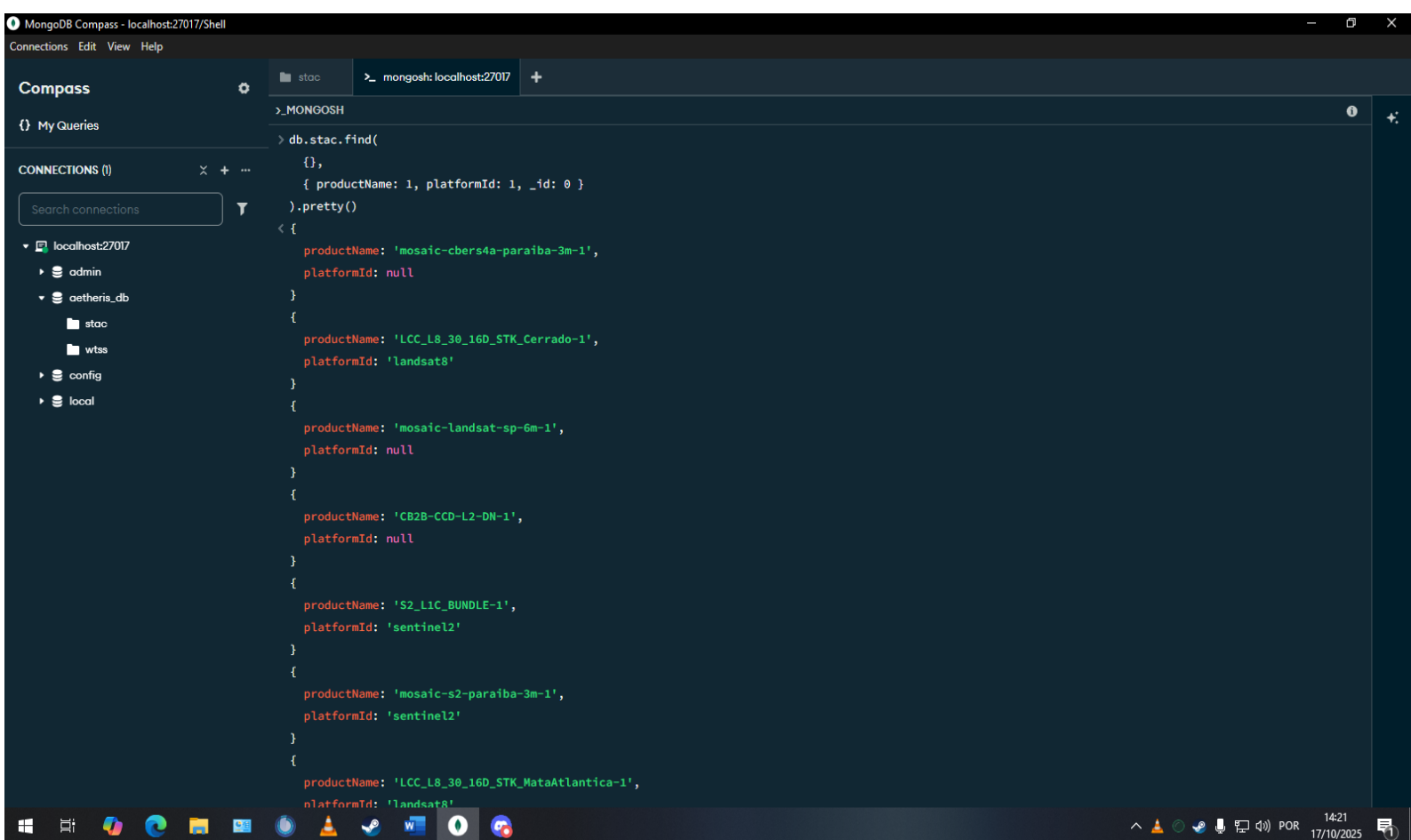
The screenshot shows the MongoDB Compass application window. The left sidebar displays the 'CONNECTIONS' panel with a search bar and a tree view showing the 'localhost:27017' connection. Under 'aetheris_db', the 'stac' collection is selected. The main panel shows the MongoDB Shell with the following commands and output:

```
> use aetheris_db
switched to db aetheris_db
> db.wtss.insertOne({
  "name": "MYD13Q1-6",
  "product": "MOD13Q1-6",
  "description": "Coverage description...",
  "attributes": [ "ndvi", "evi", "blue", "red" ],
  "timeline": [ "2000-02-18", "2000-03-05" ],
  "spatial_extent": { "u1": { "x": -76.5, "y": 15.2 }, "lr": { "x": -32.1, "y": -55.9 } },
  "dimensions": { "X": { "step": 0.002 }, "Y": { "step": 0.002 } }
})
{
  acknowledged: true,
  insertedId: ObjectId('68f27976043efeb4a502abd')
}
aetheris_db>
```

The screenshot shows the MongoDB Compass application window. The left sidebar displays the 'CONNECTIONS' panel with a search bar and a tree view showing the 'localhost:27017' connection. Under 'aetheris_db', the 'stac' collection is selected. The main panel shows the MongoDB Shell with the following commands and output:

```
> db.stac.insertMany([
  {
    "id": "S2-MSI-L2A-16D-1",
    "productName": "S2-MSI-L2A-16D-1",
    "platformId": "Sentinel2",
    "title": "Sentinel-2 MSI Level-2A 16-day",
    "description": "Surface reflectance composites.",
    "variables": ["B01", "B02", "B03", "B04", "B05", "B06"]
  },
  {
    "id": "MOD13Q1-6",
    "productName": "MOD13Q1-6",
    "platformId": "modis",
    "title": "MODIS Vegetation Index 16-Day L3 Global 250m",
    "description": "MODIS Terra Vegetation Index.",
    "variables": ["ndvi", "evi", "red", "blue", "nir", "mir"]
  },
  {
    "id": "LANDSAT-2-16D-1",
    "productName": "LANDSAT-2-16D-1",
    "platformId": "landsat-2",
    "title": "Landsat-2 MSI Level-2A 16-day",
    "description": "Surface reflectance composites.",
    "variables": ["red", "green", "blue", "nir", "swir1", "swir2"]
  }
])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('68f279d4043efeb4a502abe'),
    '1': ObjectId('68f279d4043efeb4a502abf')
  }
}
```

- Consultas básicas para validar os dados (find com projeção de campos).



- Consultas com operadores lógicos (\$and, \$or).

The screenshot shows the MongoDB Compass application running on localhost:27017. The left sidebar displays the database structure with 'aetheris_db' expanded, showing collections 'stac', 'wtas', 'config', and 'local'. The main panel shows a query in the 'MongoShell' tab:

```
>_MONGOSH
> db.stac.find({
  $and: [
    { "platformId": "modis" },
    { "variables": "nir" }
  ]
}).pretty()
< {
  _id: ObjectId('68f279d4043efeb4a502abf'),
  id: 'MOD13Q1-6',
  productName: 'MOD13Q1-6',
  platformId: 'modis',
  title: 'MODIS Vegetation Index 16-Day L3 Global 250m',
  description: 'MODIS Terra Vegetation Index.',
  variables: [
    'ndvi',
    'evi',
    'red',
    'blue',
    'nir',
    'mir'
  ]
}
```

The screenshot shows the MongoDB Compass application running on localhost:27017. The left sidebar displays the database structure with 'aetheris_db' expanded, showing collections 'stac', 'wtas', 'config', and 'local'. The main panel shows a query in the 'MongoShell' tab:

```
>_MONGOSH
> db.stac.find({
  $or: [
    { "platformId": "sentinel2" },
    { "platformId": "landsat-2" }
  ]
}).pretty()
< {
  _id: ObjectId('68e13c3c4bcd9ef7b226406'),
  id: 'S2_L1C_BUNDLE-1',
  type: 'Collection',
  stac_version: '1.0.0',
  stac_extensions: [
    'https://stac-extensions.github.io/version/v1.0.0/schema.json',
    'https://stac-extensions.github.io/processing/v1.0.0/schema.json',
    'https://stac-extensions.github.io/item-assets/v1.0.0/schema.json',
    'https://stac-extensions.github.io/eo/v1.0.0/schema.json'
  ],
  title: 'Sentinel-2 - Level-1C',
  version: '1',
  deprecated: false,
  description: 'Copernicus Sentinel-2/MSI Level-1C product over Brazil. Level-1C product provides orthorectified Top-Of-Atmosphere (TOA) reflectance images',
  keywords: [
    'sentinel',
    'sentinel-2',
    'msi',
    'level-1c',
    'top of atmosphere reflectance',
    'earth observation',
    'brazil'
  ],
  providers: [

```

4. Comparação com SQL (PostgreSQL)

FIND:

PostgreSQL:

```
SELECT product_name, platform_id FROM stac_products;
```

MongoDB:

```
db.stac.find({ platformId: 'modis', variables: 'nir' })
```

AND:

PostgreSQL:

```
SELECT
    p.*
FROM
    stac_products AS p
JOIN
    stac_product_variables AS pv ON p.id = pv.product_id
JOIN
    variables AS v ON pv.variable_id = v.id
WHERE
    p.platform_id = 'modis' AND v.name = 'nir';
```

MongoDB:

```
db.stac.find({ $or: [ { platformId: 'sentinel2' }, { platformId: 'landsat-2' } ] })
```