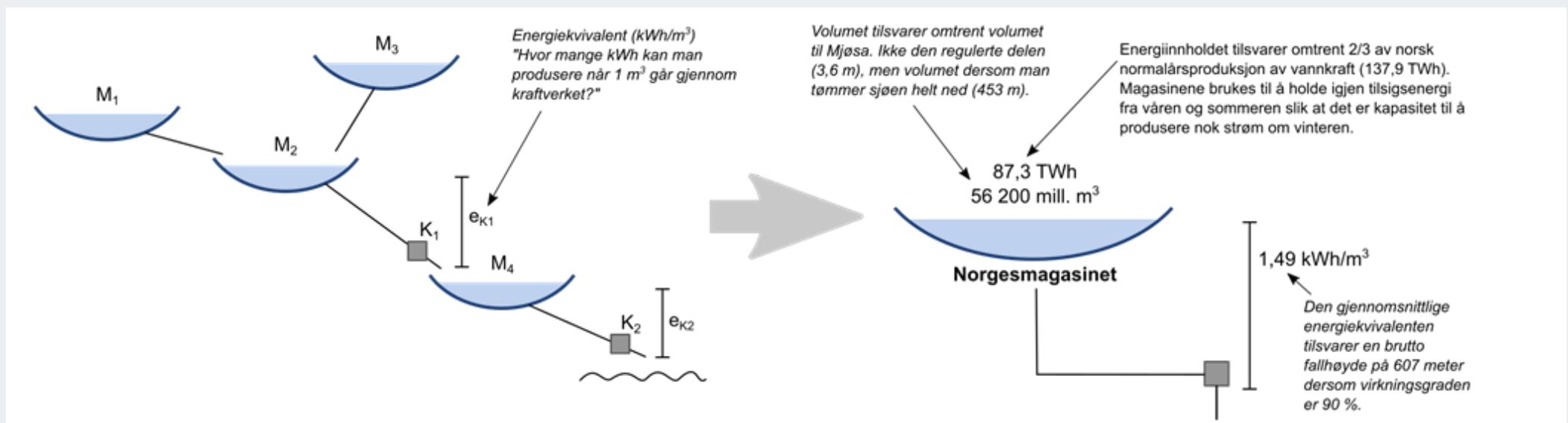


Case: Datainnsamling, Modellering og Visualisering av Magasininstatistikk

17.02.2025
Ivar Haugland



Presentasjonen:

1. Gjennomgang av case
2. Visualisering og diskusjon om data

Oppgave:

"Du er en dataanalytiker som har fått i oppdrag å hente data fra et API-endepunkt som gir tilgang til magasinstatistikk fra Norges vassdrags- og energidirektorat (NVE). Målet er å transformere denne dataen, lagre den i en database etter stjernemodellen, og lage visualiseringer i Power BI Desktop som gir innsikt i dataene.

API-endepunkt: Magasinstatistikk API"

Tilnærmingen min:

1. Lese litt om NVE og hva magasinstatistikk brukes til, ta notater om det
2. Sjekke ut API-dokumentasjonen i swagger og forstå de ulike endepunktene
3. Lære om og friske opp kunnskap til verktøy som kan brukes (Stjernemodell, Power BI, DBT)
4. Designe arkitektur i applikasjonen og datamodeller
5. Opprette kodebase med egnet stack
6. Implementer
7. Visualisere resultatene

30 sekunder om magasinstatistikk (kilde: NVE - "Om magasinstatistikken")

Magasin: Oppsamling av vann som brukes til å skape vannkraft når det trengs

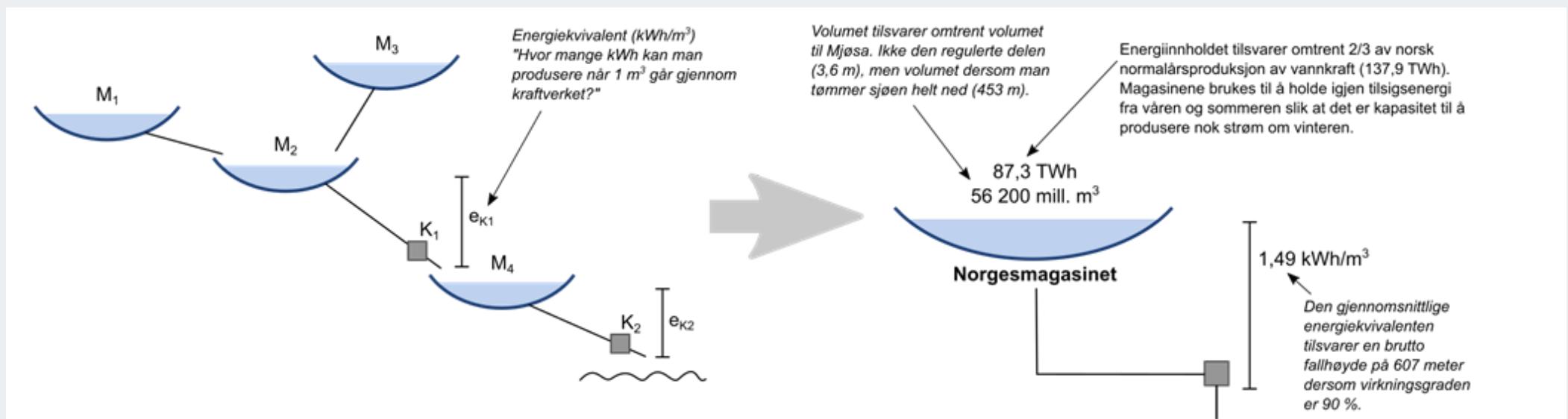
Fyllingsgrad:

- Prosentandel av vann i forhold til kapasiteten
- "Hvor mye energi er tilgjengelig?"

$$\text{Fyllingsgrad} = \frac{\sum E_{M\ i, \text{uke } j}}{\sum E_{M\ i, \text{maks}}}$$

Magasinstatistikk er viktig for:

- Markedsinfo
- Risiko for energiknapphet og avdekking
- Tiltak for å opprettholde forsyningen av strøm ved ekstraord. hendelser



Bemerkelser i eksternt API dokumentasjon

Area

```
  navn          string  
  nullable: true  
  
  navn_langt   string  
  nullable: true  
  
  beskrivelse  string  
  nullable: true  
  
  omrType      string  
  nullable: true  
  
  omrnrr       integer($int32)  
}
```

CurrentAreas

```
  description: Hent områder  
  
  land          
      
    Area > {...}]  
      
    Elspot  
        
      Area > {...}]  
        
      Vassdrag  
          
        Area > {...}]  
      
```

MagasinstatistikkOffentligMinMaxMedianModel

```
  omrType      string  
  nullable: true  
  
  Område type  
  
  omrnrr       integer($int32)  
  iso_uke     integer($int32)  
  minFyllingsgrad number($float)  
  minFyllingTWH  number($float)  
  medianFyllingsGrad number($float)  
  medianFylling_TWH number($float)  
  maxFyllingsgrad number($float)  
  maxFyllingTWH  number($float)
```

MagasinstatistikkModel

```
  description: Magasinstatistikk  
  
  dato_Id      string  
  nullable: true  
  
  Dato Id  
  
  omrType      string  
  nullable: true  
  
  Område type  
  
  omrnrr       integer($int32)  
  
  Område nummer  
  
  iso_aar      integer($int32)  
  
  ISO år  
  
  iso_uke     integer($int32)  
  
  ISO uke  
  
  fyllingsgrad number($float)  
  
  Fyllingsgrad  
  
  kapasitet_TWh number($float)  
  
  Kapasitet TWh  
  
  fylling_TWh  number($float)  
  
  Fylling TWh  
  
  neste_Publiseringdato string($date-time)  
  
  Neste publiseringdato  
  
  fyllingsgrad_forrige_uke number($float)  
  
  Fyllingsgrad forrige uke  
  
  endring_fyllingsgrad number($float)  
  
  Endring fyllingsgrad  
}
```

"Data exploration" av i jupyter

1. df.head()

	dato_Id	omrType	omrnر	iso_aar	iso_uke	fyllingsgrad	kapasitet_TWh	fylling_TWh	neste_Publiseringsdato	fyllingsgrad_forrige_uke	endring_fyllingsgrad
0	2007-03-04	EL	5	2007	9	0.416593	17.425789	7.259468	0001-01-01T00:00:00	0.450818	-0.034225
1	2015-02-08	EL	5	2015	6	0.485430	17.425789	8.458999	0001-01-01T00:00:00	0.527302	-0.041872
2	2005-09-11	EL	5	2005	36	0.875464	17.425789	15.255660	0001-01-01T00:00:00	0.858146	0.017319
3	2009-02-22	EL	5	2009	8	0.391005	17.425789	6.813575	0001-01-01T00:00:00	0.425169	-0.034164
4	1995-05-07	EL	5	1995	18	0.172278	17.425789	3.002083	0001-01-01T00:00:00	0.175498	-0.003219
5	2015-02-15	EL	5	2015	7	0.448829	17.425789	7.821195	0001-01-01T00:00:00	0.485430	-0.036601

2. df.shape -> (14139, 11)

3. Navn på kolonner

4. Datatyper

dato_Id	object
omrType	object
omrnر	int64
iso_aar	int64
iso_uke	int64
fyllingsgrad	float64
kapasitet_TWh	float64
fylling_TWh	float64
neste_Publiseringsdato	object
fyllingsgrad_forrige_uke	float64
endring_fyllingsgrad	float64
dtype:	object

5. df['iso_aar'].sort_values().unique()

```
array([1995, 1996, 1997, 1998, 1999, 2000, 2001, 2002, 2003, 2004, 2005,
       2006, 2007, 2008, 2009, 2010, 2011, 2012, 2013, 2014, 2015, 2016,
       2017, 2018, 2019, 2020, 2021, 2022, 2023, 2024, 2025])
```

6. df.groupby(['omrType'])['fyllingsgrad'].mean()

omrType	fyllingsgrad
EL	0.615916
NO	0.623491
VASS	0.625474
Name: fyllingsgrad, dtype: float64	

Bla bla bla...

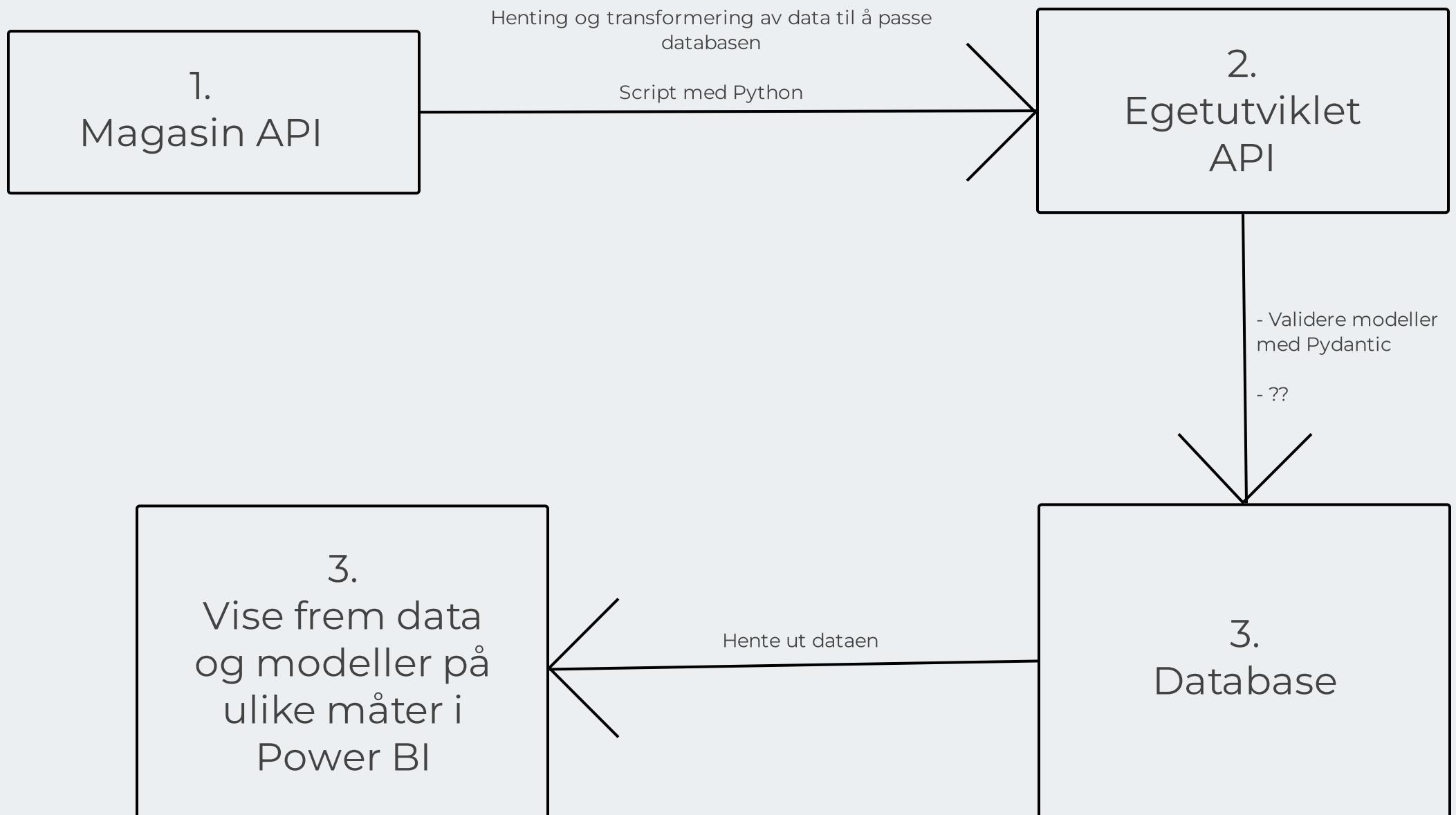
STARTE UKER PÅ SØNDAG! (som er målingsdagen)

```
MagasinstatistikkModel {  
    description: Magasinstatistikk  
    dato_Id: string nullable: true  
    omrnr: integer($int32)  
    iso_aar: integer($int32)  
    iso_uke: integer($int32)  
    fyllingsgrad: number($float)  
    kapasitet_TWh: number($float)  
    fylling_TWh: number($float)  
    neste_Publiseringsdato: string($date-time)  
    fyllingsgrad_forrige_uke: number($float)  
    endring_fyllingsgrad: number($float)  
}
```

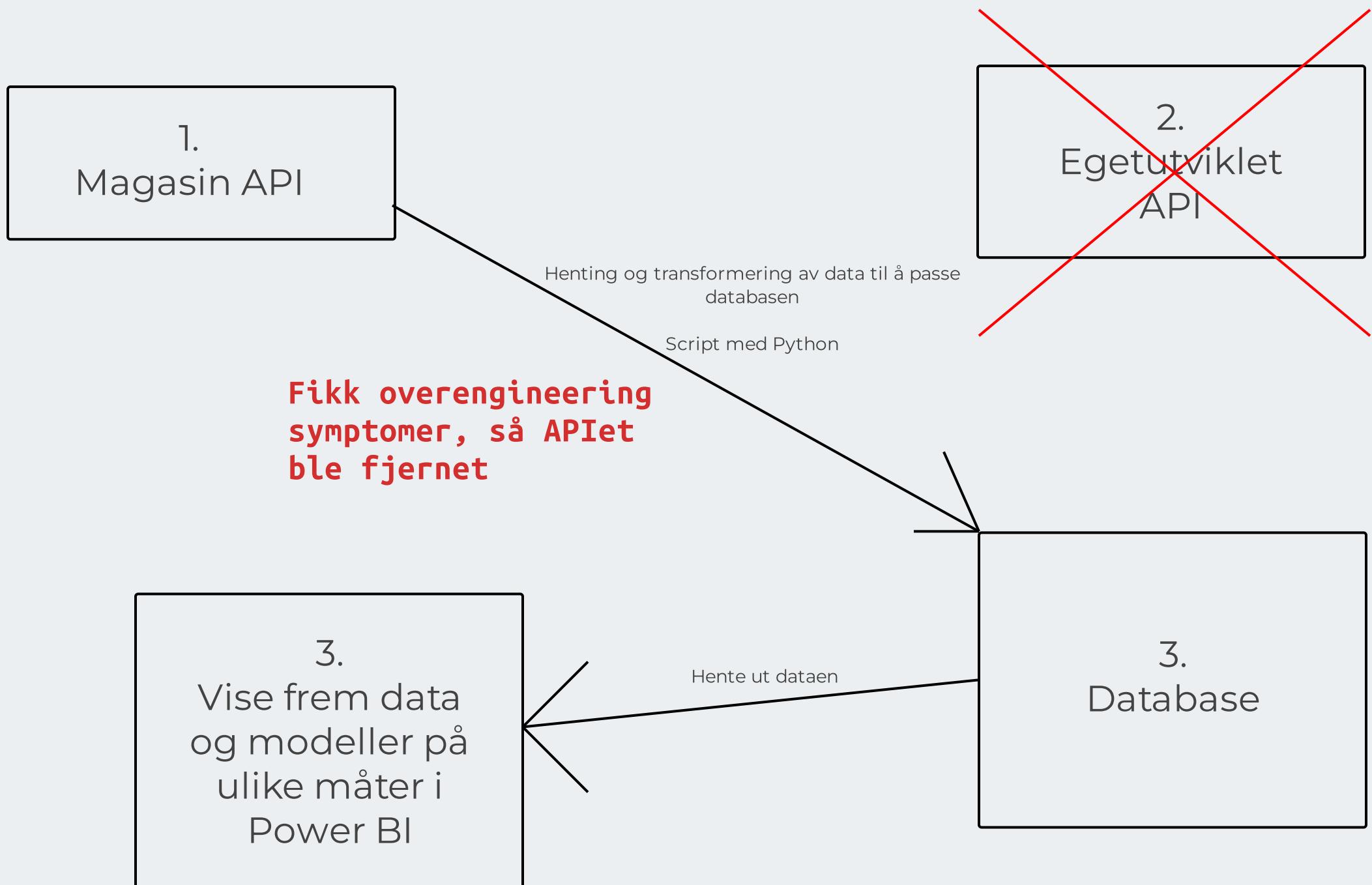
Kunnskap om modeller og verktøy

- Skjønte stjernemodell litt fra før
 - Faktatabell, dimensjoner, koble tabeller osv.
- Usikker på Power BI og hvilke verktøy det tilbyr
- Skjønte at man kan transformere data på et vis med DBT, men ikke detaljer rundt det
 - Hadde ikke planer om å bruke det i starten av prosjektet, men det kom etterhvert
- Hadde brukt SQL og Python før heldigvis

Pipeline arkitektur: Design plan 1



Pipeline arkitektur: Design plan 2



Framgangsmåte "Extract and load raw source data" - modulen

1. Designe database modellen for den rå dataen
 - Vil den skal etterligne datamodellene i APIet
 - Stjernemodell
2. Skrive "pseudokode" i jupyter
 - Hvilke funksjoner har vi
 - Skape intuitiv rekkefølge på operasjoner
 - Teste og bli kjent med dataen
 - Nødvendige Python bibliotek som bør installeres
3. Implementere i applikasjonen
 - Hvilke mapper og struktur gir mening?
 - Fin lesbar kode
 - ^
 - | Balansere disse til beste evne
 - | Hjelper å teste/leke først i jupyter
 - v
 - Optimalisering
 - Stacken

Oversikt "Extract and load raw source data" - modulen

```
| source/ - Navn på modul (generelt navn for nå, for å ikke overkomplisere  
i starten)  
|   | db/  
|   |   | nve_db.sql - Database modellen  
|   | load/  
|   |   | __pycache__/  
|   |   | jnotebook/ - "Data exploration" og "Pseudukode"  
|   |   | logs/  
|   |   | utils/  
|   |   |   | prepare_models.py - Forberede data rett før databaseinjeksjon  
|   |   |   | utils.py - Hjelpefunksjoner til scriptet  
|   |   |   | database.py - Engine til å kontakte database  
|   |   |   | load_config.py - Miljøvariabler som man kan spesifisere  
|   |   |   | main.py - Hovedfil til å kjøre python scriptet  
|   | prestart.sh* - Kjører scriptet for Docker containeren  
|   | Dockerfile  
|   | pyproject.toml - Dependencies og modul informasjon
```

Database modell for rå data: nve_db.sql

area	
id	int
navn	text?
navn_langt	text?
beskrivelse	text?
omr_type	text?
omrnr	int
current_area	text
Indexes	
* omr_type,omrnr	

dates	
id	bigint
iso_dato	date
iso_aar	int
iso_uke	int
iso_maaned	int
iso_dag	int

```
CREATE TABLE IF NOT EXISTS area (
    id SERIAL PRIMARY KEY,
    navn TEXT,
    navn_langt TEXT,
    beskrivelse TEXT,
    omr_type TEXT,
    omrnr INTEGER NOT NULL,
    current_area TEXT NOT NULL,
    UNIQUE(omr_type, omrnr)
);
```

magasinstatistikk_min_max_median_model

id	bigint
area_id	int
iso_uke	int
min_fyllingsgrad	decimal(10,6)
min_fylling_twh	decimal(10,6)
max_fyllingsgrad	decimal(10,6)
max_fylling_twh	decimal(10,6)
median_fyllingsgrad	decimal(10,6)
medie_fylling_twh	decimal(10,6)

magasinstatistikk_model

id	bigint
dato_id	bigint?
area_id	bigint
fyllingsgrad	decimal(10,6)
kapasitet_twh	decimal(10,6)
fylling_twh	decimal(10,6)
neste_publiseringsdato	timestamp?
fyllingsgrad_forrige_uke	decimal(10,6)
endring_fyllingsgrad	decimal(10,6)

```
CREATE TABLE magasinstatistikk_min_max_median_model (
    id BIGSERIAL PRIMARY KEY,
    area_id INTEGER NOT NULL,
    iso_uke INTEGER NOT NULL,
    min_fyllingsgrad NUMERIC(10, 6) NOT NULL,
    min_fylling_twh NUMERIC(10, 6) NOT NULL,
    max_fyllingsgrad NUMERIC(10, 6) NOT NULL,
    max_fylling_twh NUMERIC(10, 6) NOT NULL,
    median_fyllingsgrad NUMERIC(10, 6) NOT NULL,
    median_fylling_twh NUMERIC(10, 6) NOT NULL,
    FOREIGN KEY (area_id) REFERENCES area (id)
);
```

```
CREATE TABLE dates (
    id BIGSERIAL PRIMARY KEY,
    iso_dato DATE NOT NULL UNIQUE,
    iso_aar INTEGER NOT NULL,
    maaling_uke INTEGER NOT NULL, -- Week of measuring
    iso_maaned INTEGER NOT NULL,
    iso_dag INTEGER NOT NULL
);
```

```
CREATE TABLE IF NOT EXISTS magasinstatistikk_model (
    id BIGSERIAL PRIMARY KEY,
    dato_id BIGINT,
    area_id BIGINT NOT NULL,
    fyllingsgrad NUMERIC(10, 6) NOT NULL,
    kapasitet_twh NUMERIC(10, 6) NOT NULL,
    fylling_twh NUMERIC(10, 6) NOT NULL,
    neste_publiseringsdato TIMESTAMP,
    fyllingsgrad_forrige_uke NUMERIC(10, 6) NOT NULL,
    endring_fyllingsgrad NUMERIC(10, 6) NOT NULL,
    FOREIGN KEY (area_id) REFERENCES area (id),
    FOREIGN KEY (dato_id) REFERENCES dates (id)
);
```

Databasemodell -- mapping --> API modell

```
CREATE TABLE IF NOT EXISTS area (
    id SERIAL PRIMARY KEY,
    navn TEXT,
    navn_langt TEXT,
    beskrivelse TEXT,
    omr_type TEXT,
    omrnrs INTEGER NOT NULL,
    current_area TEXT NOT NULL,
    UNIQUE(omr_type, omrnrs)
);
```

```
CREATE TABLE IF NOT EXISTS magasinstatistikk_model (
    id BIGSERIAL PRIMARY KEY,
    dato_id BIGINT,
    area_id BIGINT NOT NULL,
    fyllingsgrad NUMERIC(10, 6) NOT NULL,
    kapasitet_twh NUMERIC(10, 6) NOT NULL,
    fylling_twh NUMERIC(10, 6) NOT NULL,
    neste_publiseringsdato TIMESTAMP,
    fyllingsgrad_forrige_uke NUMERIC(10, 6) NOT NULL,
    endring_fyllingsgrad NUMERIC(10, 6) NOT NULL,
    FOREIGN KEY (area_id) REFERENCES area (id),
    FOREIGN KEY (dato_id) REFERENCES dates (id)
);
```

```
CREATE TABLE magasinstatistikk_min_max_median_model (
    id BIGSERIAL PRIMARY KEY,
    area_id INTEGER NOT NULL,
    iso_uke INTEGER NOT NULL,
    min_fyllingsgrad NUMERIC(10, 6) NOT NULL,
    min_fylling_twh NUMERIC(10, 6) NOT NULL,
    max_fyllingsgrad NUMERIC(10, 6) NOT NULL,
    max_fylling_twh NUMERIC(10, 6) NOT NULL,
    median_fyllingsgrad NUMERIC(10, 6) NOT NULL,
    median_fylling_twh NUMERIC(10, 6) NOT NULL,
    FOREIGN KEY (area_id) REFERENCES area (id)
);
```

```
Area < {
    navn
    navn_langt
    beskrivelse
    omrType
    omrnrs
}
```

```
CurrentAreas < {
    description: "Hent områder"
    land
    elspot
    vassdrag
}
```

```
CREATE TABLE dates (
    id BIGSERIAL PRIMARY KEY,
    iso_dato DATE NOT NULL UNIQUE,
    iso_aar INTEGER NOT NULL,
    iso_uke INTEGER NOT NULL,
    iso_maaned INTEGER NOT NULL,
    iso_dag INTEGER NOT NULL
);
```

```
MagasinstatistikkModel < {
    description: "Magasinstatistikk"
    dato_Id
    omrType
    omrnrs
    iso_aar
    iso_uke
    fyllingsgrad
    kapasitet_TWh
    fylling_TWh
    neste_Publiseringsdato
    fyllingsgrad_forrige_uke
    endring_fyllingsgrad
}
```

```
MagasinstatistikkOffentligMinMaxMedianModel < {
    omrType
    omrnrs
    iso_uke
    minFyllingsgrad
    minFyllingTWH
    medianFyllingsGrad
    medianFylling_TWH
    maxFyllingsgrad
    maxFyllingTWH
}
```

Flyt i "Extract and load raw source data"-Python script

Kan beskrives gjennom logs:

1. Starting extract and load script with NVE magasin statistics for source database...

Extract data from Magasin API:

2. Loaded and prepared area df from '\$BIAPI_STR_PREFIX/HentOmråder'
3. Loaded magasin statistics from '\$BIAPI_STR_PREFIX/HentOffentligData'
4. Loaded and prepared 'dates'

Transformations:

5. Dropped columns ['iso_aar', 'iso_uke'] and renamed in magasin_df

Load prepared data:

6. Loaded dato data into db
7. Loaded area data into db

More transformations and load prepared data:

8. Prepared and loaded magasin statistics data into db
9. Prepared and loaded magasin min max median data into db

Finish:

10. Finished loading NVE magasin API data into source db

Men...

Noe mer som kan gjøres før visualisering?

...

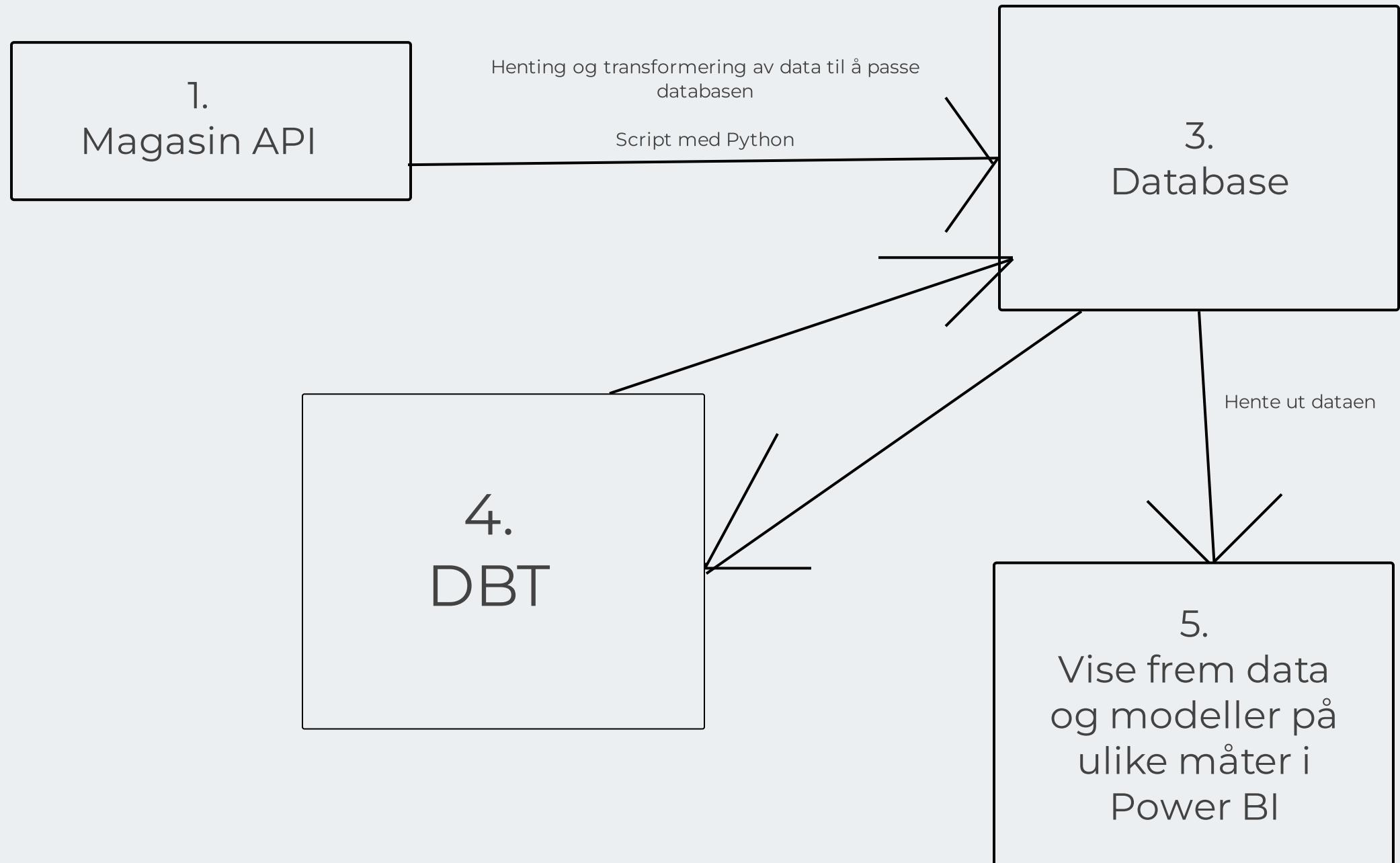
Hva med:

- *Skape mindre sannsynlighet for misforståelse i dataen
(spesielt hvis flere jobber med den)*
 - *Danne faste konvensjoner*
 - *God dokumentasjon om datamodeller*
 - *Gjøre dataen "ryddig" i struktur og navngiving*
- *Flere modeller som er forhåndsprosessert (gir less cost)*

DBT

*Litt overkill for å løse casen, men viktig hvis prosjektet var
ekte*

Pipeline arkitektur: Design plan (nåværende)



Oversikt "Transform with DBT" - modulen

Generelt et helt standard DBT prosjekt generert

```
| | source/
| | transform/
| | | nve_statistics_transform/
| | | | analyses/
| | | | dbt_packages/
| | | | logs/
|
| | | | macros/
| | | | | aggregation_macros.sql
|
| | | | models/
| | | | | nve_reservoir/
|
| | | | | staging/ - Atomer/Byggeblokkene våre (nærmest kilde databasen)
| | | | | | _nve_reservoir__models.yml
| | | | | | _nve_reservoir__sources.yml
| | | | | | stg_nve_reservoir__areas.sql
| | | | | | stg_nve_reservoir__dates.sql
| | | | | | stg_nve_reservoir__reservoir_measures.sql
| | | | | | stg_nve_reservoir__reservoir_measures_metrics.sql
|
| | | | | intermediate/ - "Molekylene"
| | | | | | _int_nve_reservoir__models.yml
| | | | | | int_res_measures_pivot_fill_capacity_to_order_areas.sql
| | | | | | int_res_measures_pivot_fill_deg_to_order_areas.sql
| | | | | | int_res_measures_pivot_measures_to_order_areas.sql
|
| | | | | marts/ - "DNA / kompliserte kjemiske forbindelser" nivået
| | | | | | average_metrics_reservoir_measures.sql
| | | | | | reservoir_measures.sql
| | | | | | reservoir_measures_metrics_for_areas.sql
|
| | | | README.md
| | | | dbt_project.yml
| | | | dependencies.yml
```

1. DBT transformasjoner utført på dataen:

- Kolonnenavn med fast språk (valgte engelsk oversettelse)
- Produsere forskjellige metrics (men dette er litt dårlig praksis uten å bruke MetricFlow)
- Denormaliserte tabeller for raskere og mindre prosessering i queries til visualisering
- Skille mellom views, ephemerals og materialized tabeller for å lettere bygge opp interessante datamodeller

2. Dokumentasjon om datamodellene

iso_week	<input type="text"/>	=	<input type="text"/>	1	iso_week	<input type="checkbox"/> descending	50	SELECT	!
(anywhere)	<input type="text"/>	=	<input type="text"/>			<input type="checkbox"/> descending			

Min, max, median model table originally

SELECT * FROM "stg_nve_reservoir__reservoir_measures_metrics" WHERE "iso_week" = '1' ORDER BY "iso_week" LIMIT 50

(0.001s) Edit

<input type="checkbox"/> Modify	measure_metric_id	area_id	iso_week	min_filling_degree	min_filling_twh	max_filling_degree	max_filling_twh	median_filling_degree	median_filling_twh
<input type="checkbox"/> edit	1	2	1	0.456893	2.742963	0.759600	4.560268	0.616703	3.702380
<input type="checkbox"/> edit	54	3	1	0.404051	13.734142	0.859533	29.216505	0.717095	24.374863
<input type="checkbox"/> edit	107	4	1	0.395877	3.611468	0.732595	6.683239	0.610952	5.573526
<input type="checkbox"/> edit	160	5	1	0.463787	9.678631	0.822122	17.156641	0.645429	13.469269
<input type="checkbox"/> edit	213	6	1	0.433804	7.559378	0.762482	13.286841	0.665023	11.588549
<input type="checkbox"/> edit	266	1	1	0.426906	37.316658	0.787652	68.850120	0.666378	58.249350
<input type="checkbox"/> edit	319	7	1	0.422694	15.219154	0.800466	28.820896	0.670393	24.137615
<input type="checkbox"/> edit	372	8	1	0.415112	9.655321	0.823836	19.162083	0.723030	16.817356
<input type="checkbox"/> edit	425	9	1	0.442647	12.459222	0.785383	22.106274	0.626173	17.624962

New table denormalized with year, month and for each area granularity

<input type="checkbox"/> Modify	area_id	iso_week	iso_month	iso_year	area_type	area_number	name_long	name	area_filling_twh	area_filling_degree	min_week_filling_degree	min_week_filling_twh	max_week_filling_degree	max_week_filling_twh
<input checked="" type="checkbox"/> edit	1	1	1	2016	NO	0	Norge	Norge	68.850120	0.787652	0.605868	3.637335	0.859533	68.850120
<input type="checkbox"/> edit	7	1	1	2016	VASS	1	Vassdragsområde 1	VASS1	27.654310	0.768065	0.605868	3.637335	0.859533	68.850120
<input type="checkbox"/> edit	2	1	1	2016	EL	1	Eisspotområde 1	NO 1	3.637335	0.605868	0.605868	3.637335	0.859533	68.850120
<input type="checkbox"/> edit	8	1	1	2016	VASS	2	Vassdragsområde 2	VASS2	19.162083	0.823836	0.605868	3.637335	0.859533	68.850120
<input type="checkbox"/> edit	3	1	1	2016	EL	2	Eisspotområde 2	NO 2	29.216505	0.859533	0.605868	3.637335	0.859533	68.850120
<input type="checkbox"/> edit	9	1	1	2016	VASS	3	Vassdragsområde 3	VASS3	22.044813	0.783200	0.605868	3.637335	0.859533	68.850120
<input type="checkbox"/> edit	4	1	1	2016	EL	3	Eisspotområde 3	NO 3	6.287615	0.689228	0.605868	3.637335	0.859533	68.850120
<input type="checkbox"/> edit	5	1	1	2016	EL	4	Eisspotområde 4	NO 4	17.156641	0.822122	0.605868	3.637335	0.859533	68.850120
<input type="checkbox"/> edit	6	1	1	2016	EL	5	Eisspotområde 5	NO 5	12.519998	0.718475	0.605868	3.637335	0.859533	68.850120

DAG for modellen i forrige slide



Energiinnhold (tWh) i Norge 2025 med min, max for måling pr uke

1. jan 2025 -> nå

Fyllingsgrad forrige uke

area_filling_degree

63.65%

Endring FG forrige uke

change_filling_degree

-3.23%

Fylling tWh

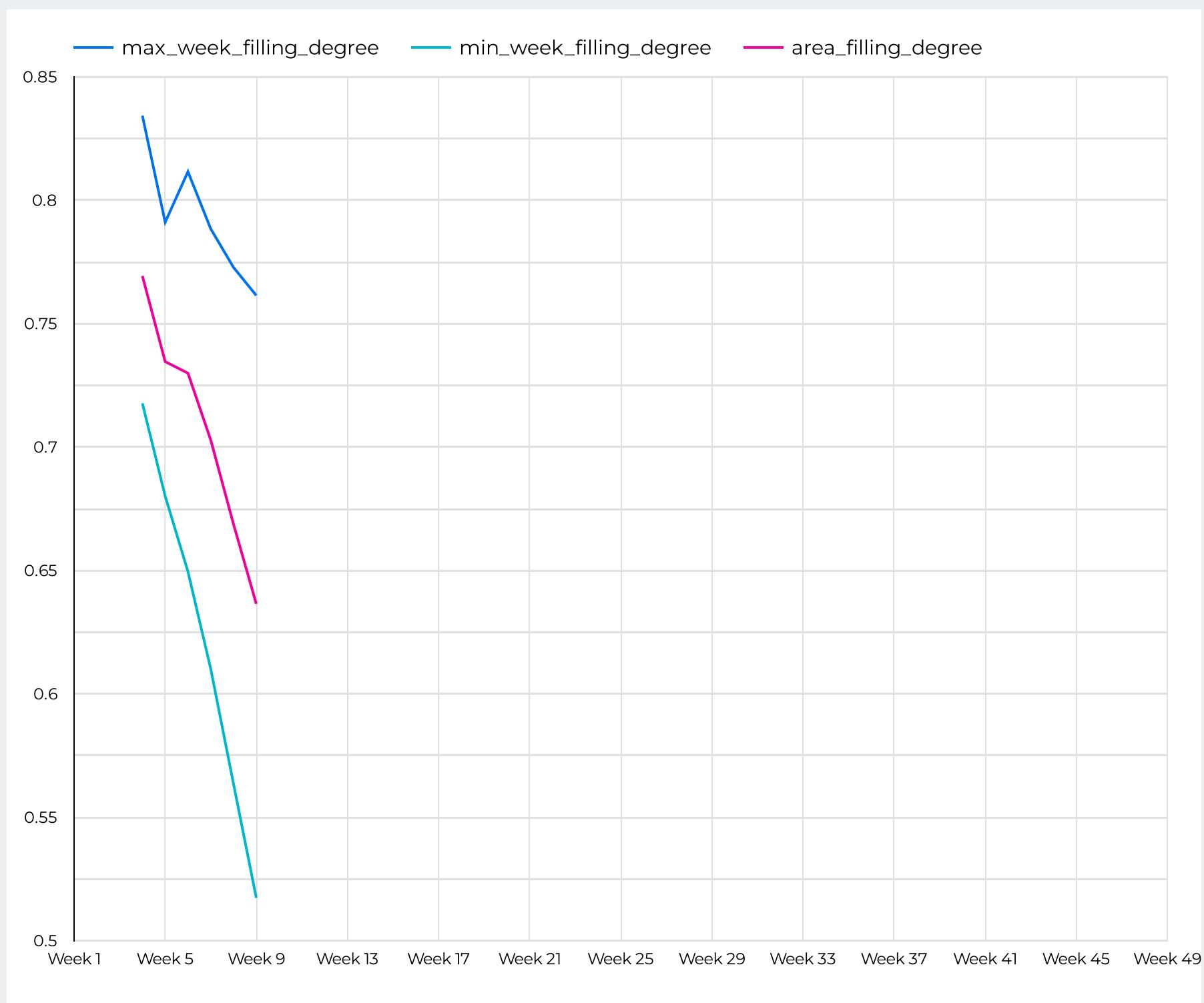
filling_twh

168.2

Kapasitet tWh

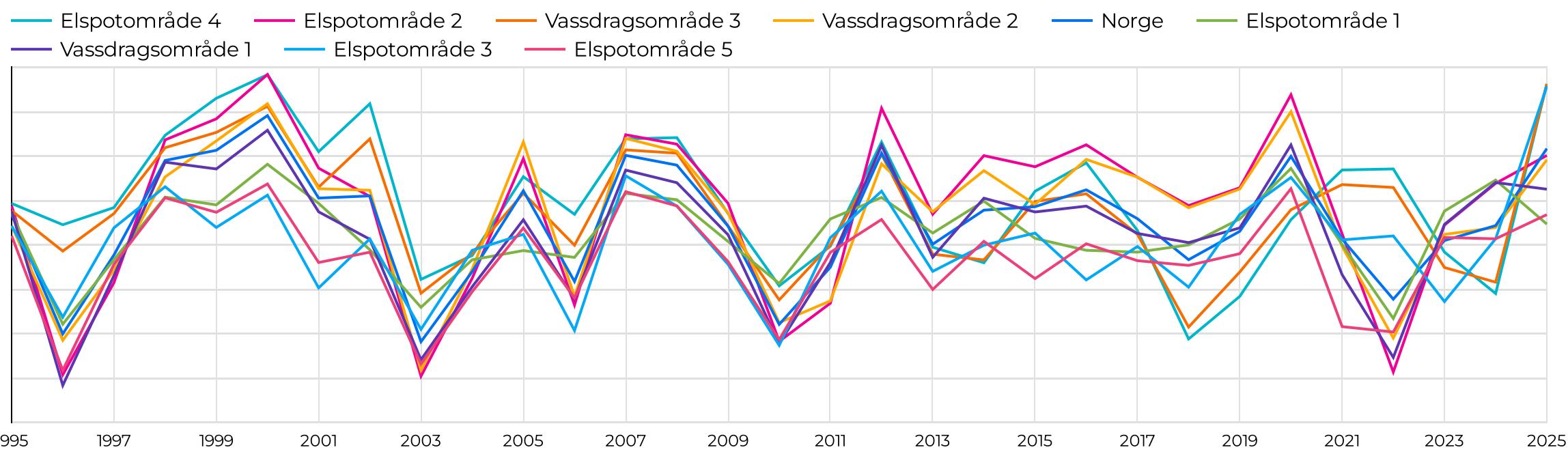
capacity_twh

262.24

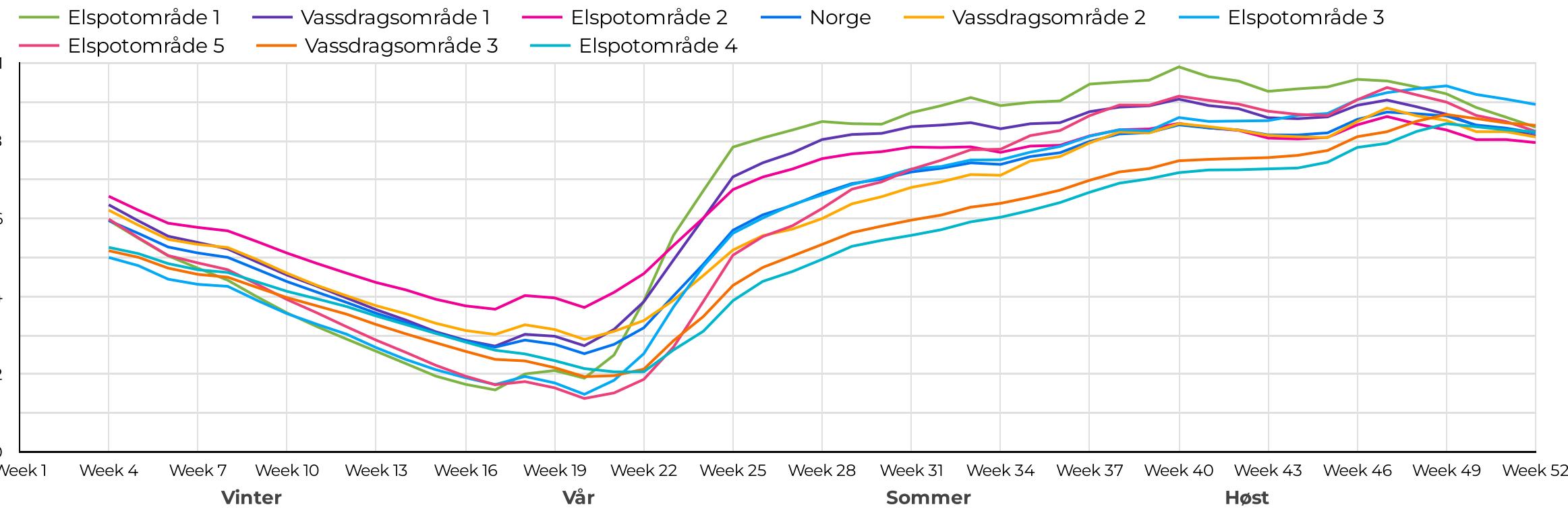


Fyllingsgrad (forskjellig grain) i hvert område over tid

1995-2025

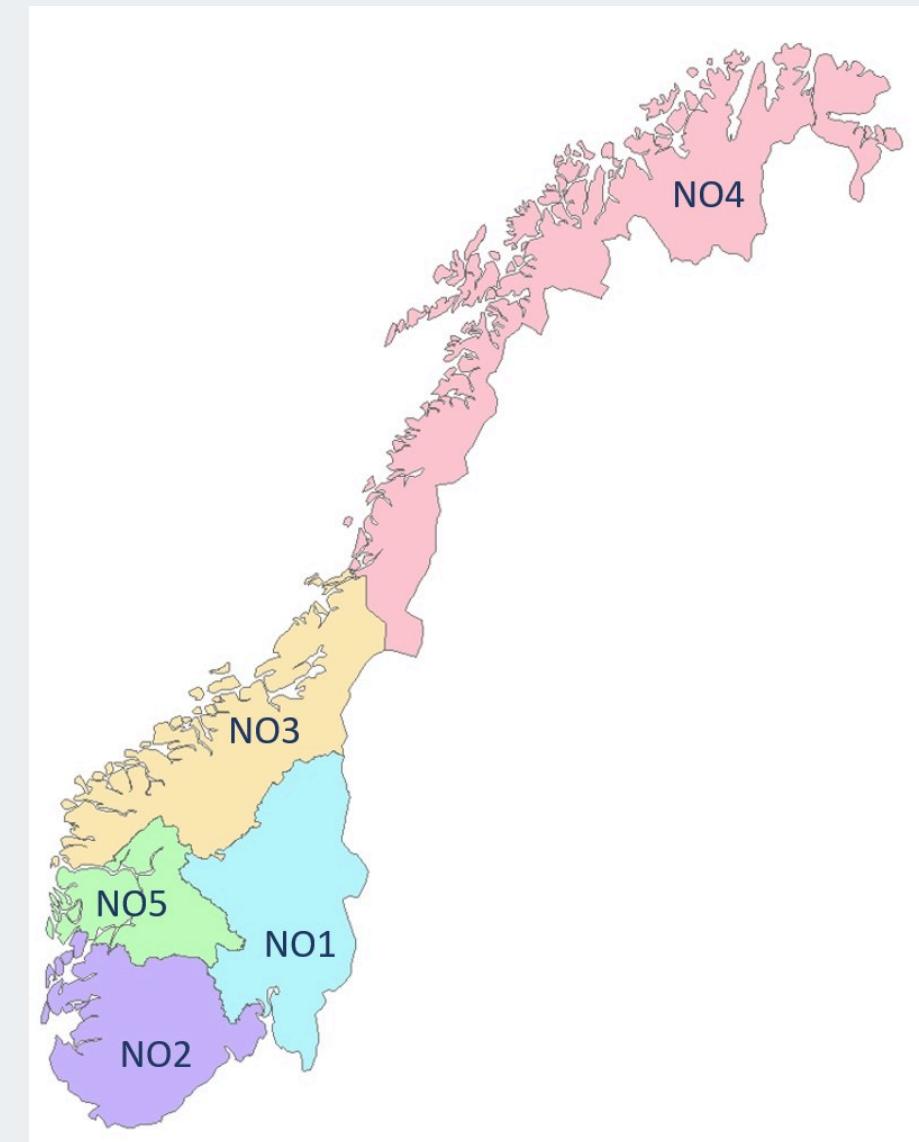
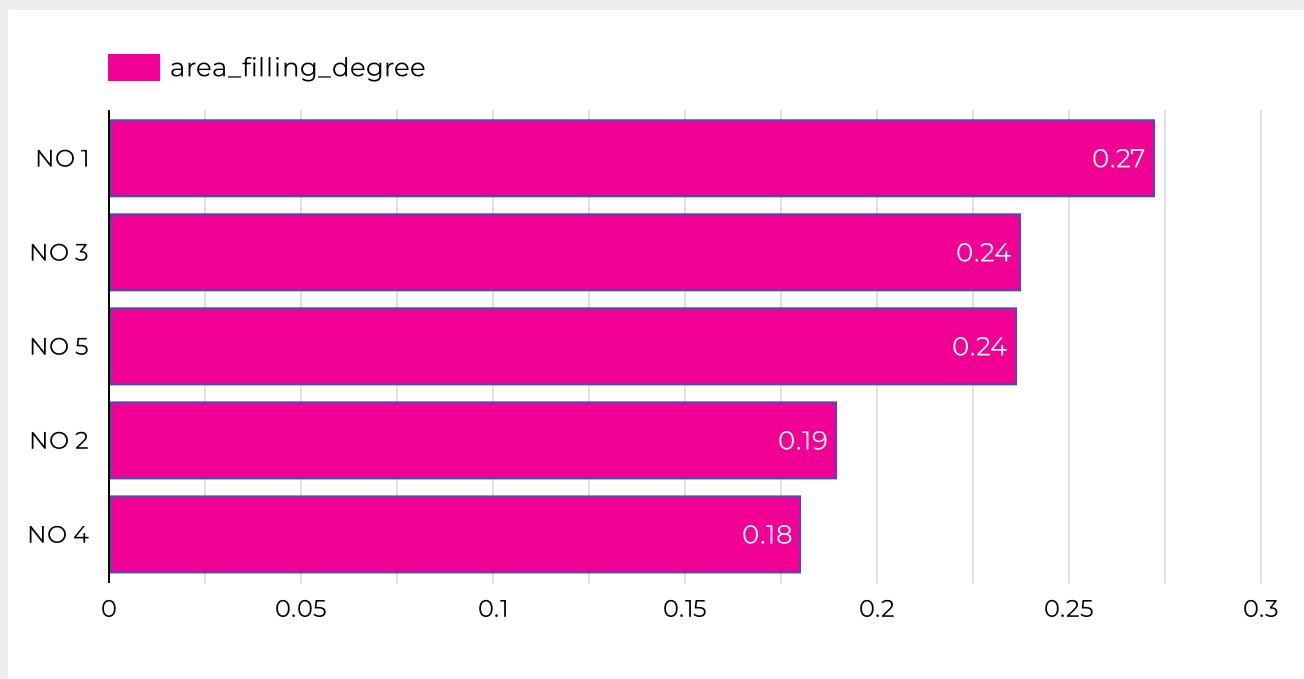


1 år (uker) - 1. jan 2024 - 1. jan 2025



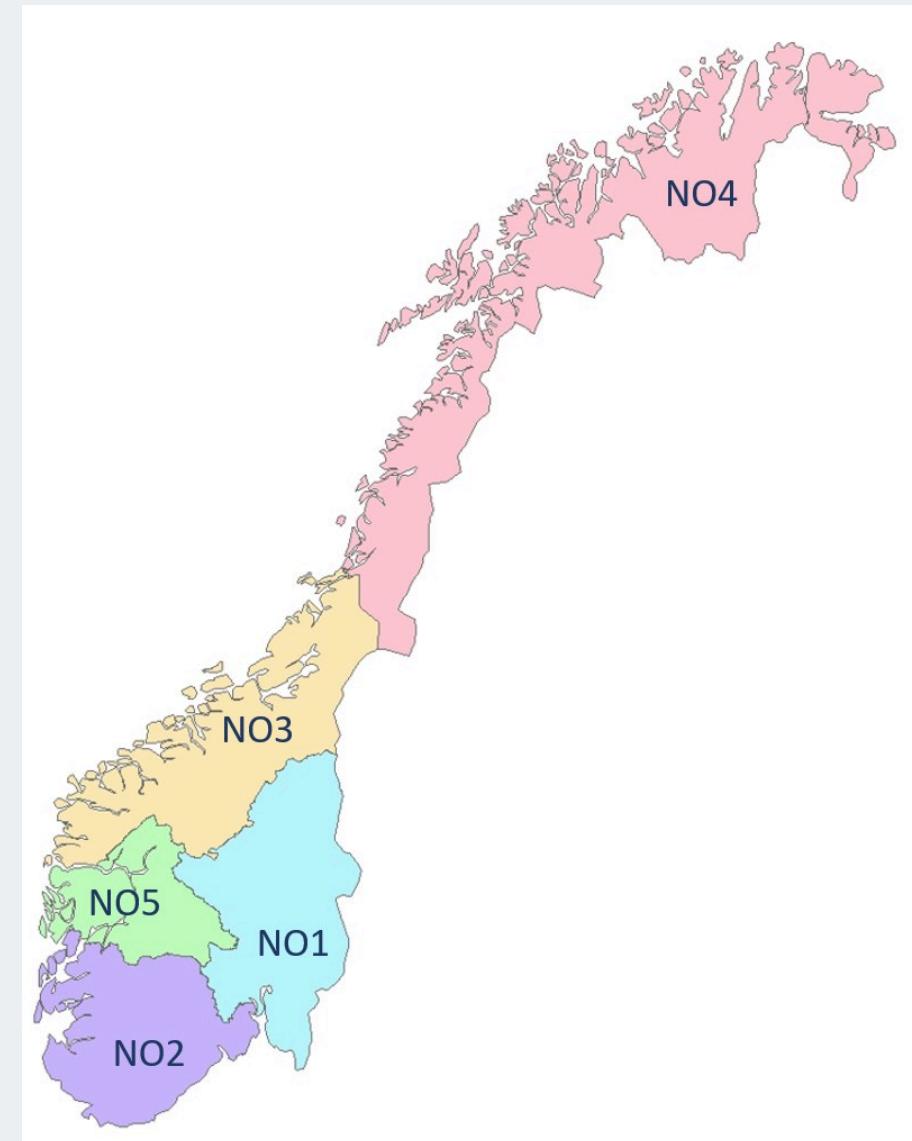
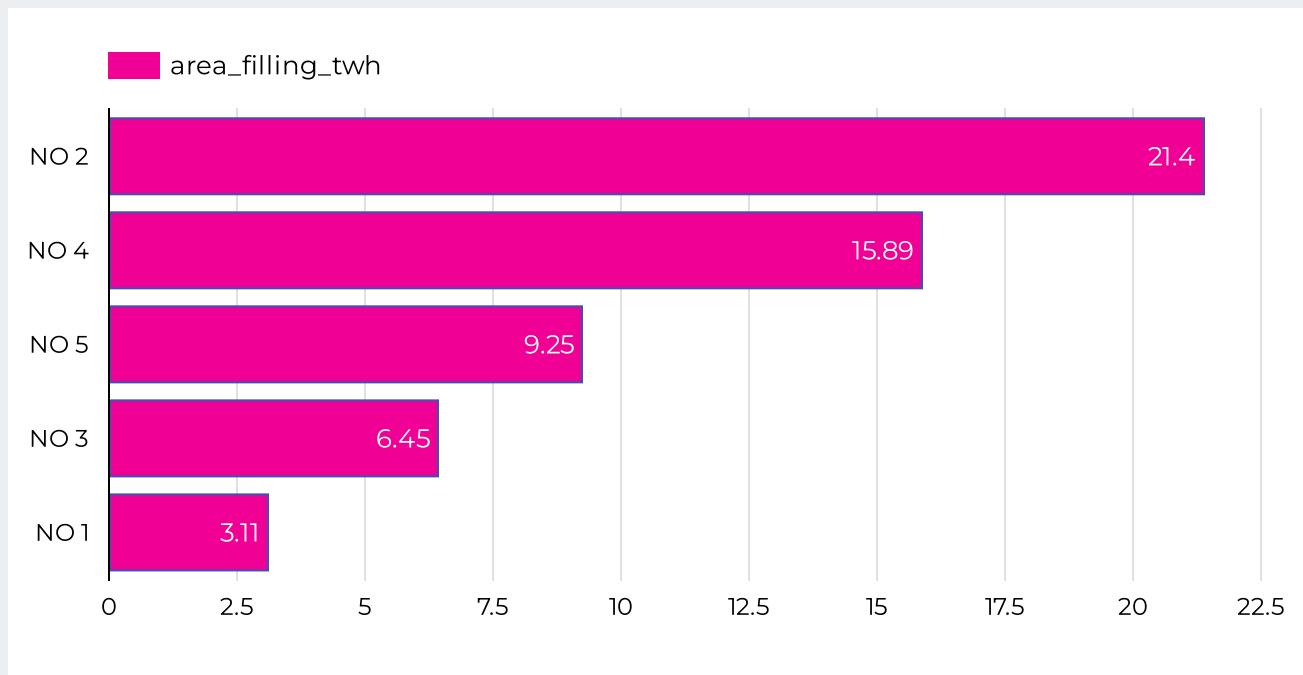
Mest varierende fyllingsgrad fra gjennomsnittet i området

Standardavvik for fyllingsgrad i område



Hvilke fylling (tWh) har områdene?

Fylling (tWh) per område i forrige uke



Stacken

- Python (pandas, sqlalchemy, requests, logging)
- SQL og PostgreSQL
- DBT
- Docker
- Adminer
- Bash script

Referanser

"Om magasinstatistikk" - (NVE) <https://www.nve.no/energi/analyser-og-statistikk/om-magasinstatistikken/>

DBT docs: <https://docs.getdbt.com/docs/introduction>