ORACLE

MySQL

Gaining Skills by Gauging Gas: My Gasometer

Mario Beck

Oracle MySQL Solution Consulting

Agenda

- Why? – Idea, Goal, Usecase
- How? – What Technologies/MySQL Features are used
- What? – Any useful results?
- Wow! – What did I learn

Why? Idea, Goal, Usecase

Motivation, Intention, Idea

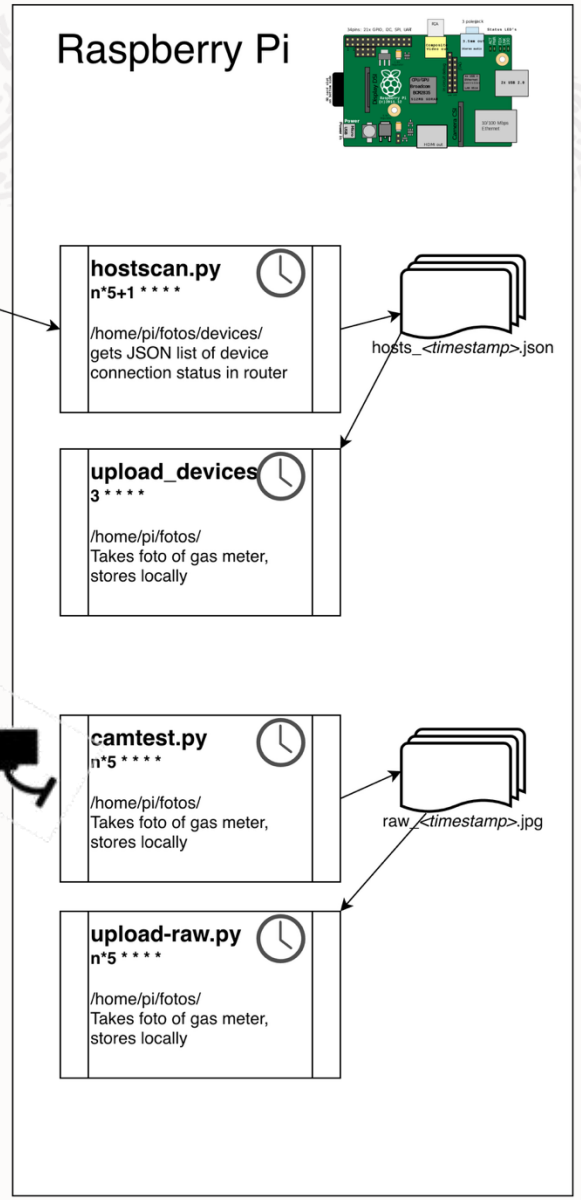
- Do something with Machine Learning
- Use real world and new data
- Do it with MySQL
- Use as many features as possible
- *„Detours increase knowledge of the area“* (Kurt Tucholsky, 1890 – 1935): No Optimizations
- Use whatever is available. Maintainability is for cowards: No Simplicity
- Something useful?
- Learn… even if there is no use in this case
- So…

Use Case

- Predict Gas Consumption based on
  - Outside temperature
  - People [not] at home
  - Fireplace [not] burning
- Take a photo every 5 minutes
  - Raspberry Pi
  - OCR is a nice ML use case

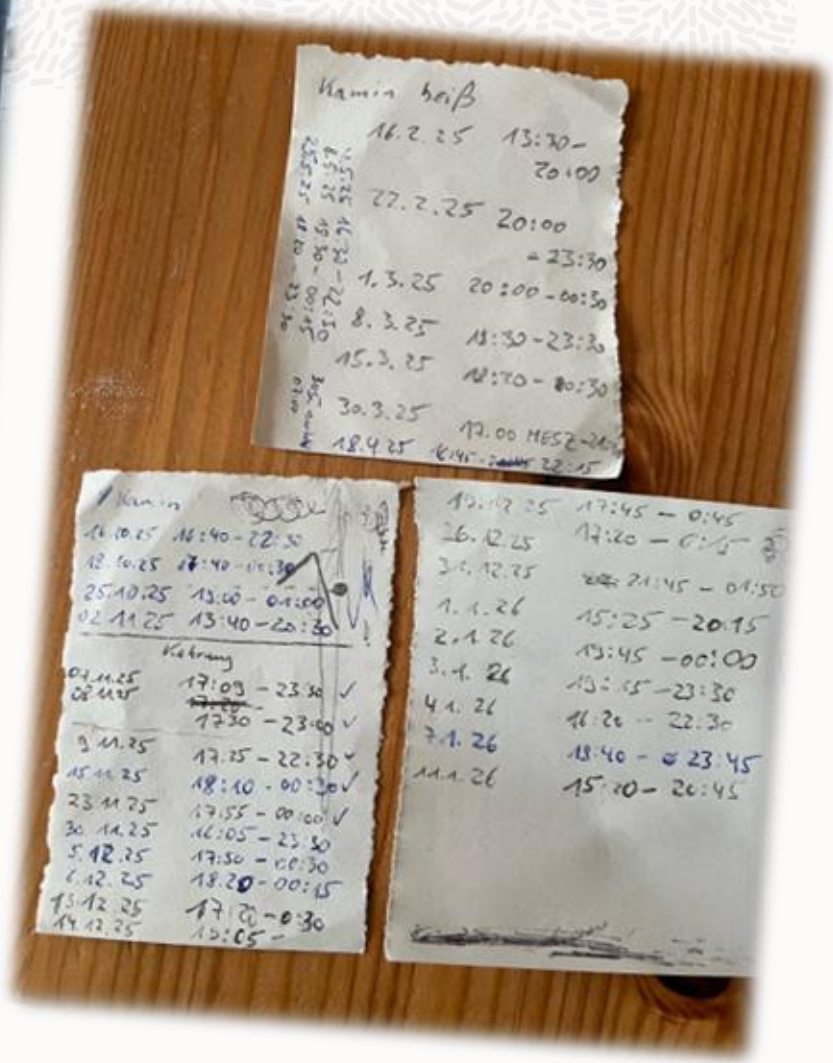
Use Case

- Predict Gas Consumption based on
  - Outside temperature
  - People [not] at home
  - Fireplace [not] burning
- Take a photo every 5 minutes
  - Raspberry Pi
  - OCR is a nice ML use case
- Mobile phones in WLAN
  - Query WLAN Router for Devices
  - JSON data. Great!
- Weather Data from DWD
  - Freely available, hourly temp and humidity



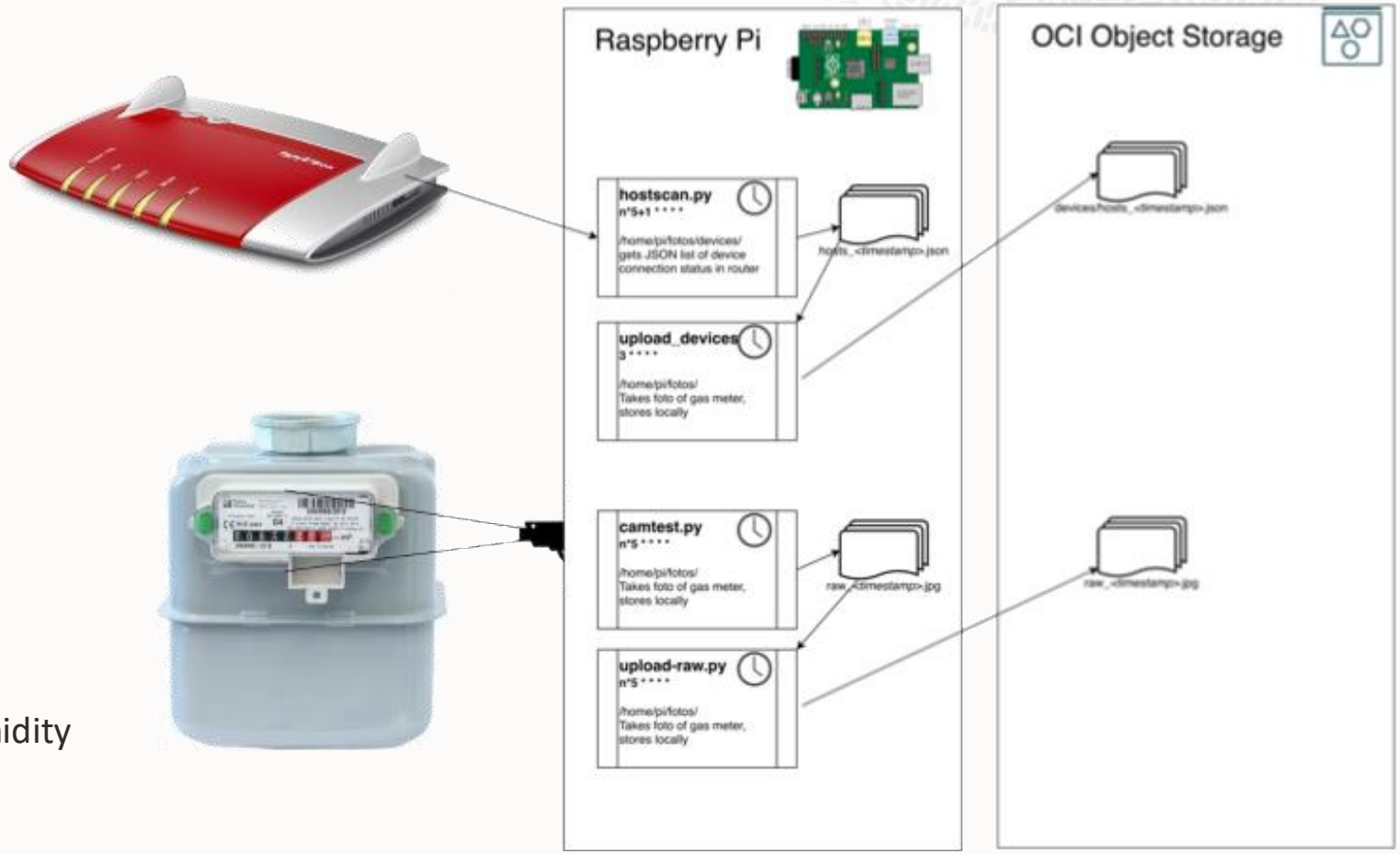
Use Case

- Predict Gas Consumption based on
  - Outside temperature
  - People [not] at home
  - Fireplace [not] burning
- Take a photo every 5 minutes
  - Raspberry Pi
  - OCR is a nice ML use case
- Mobile phones in WLAN
  - Query WLAN Router for Devices
  - JSON data. Great!
- Weather Data from DWD
  - Freely available, hourly temp and humidity
- Fireplace… manually for now

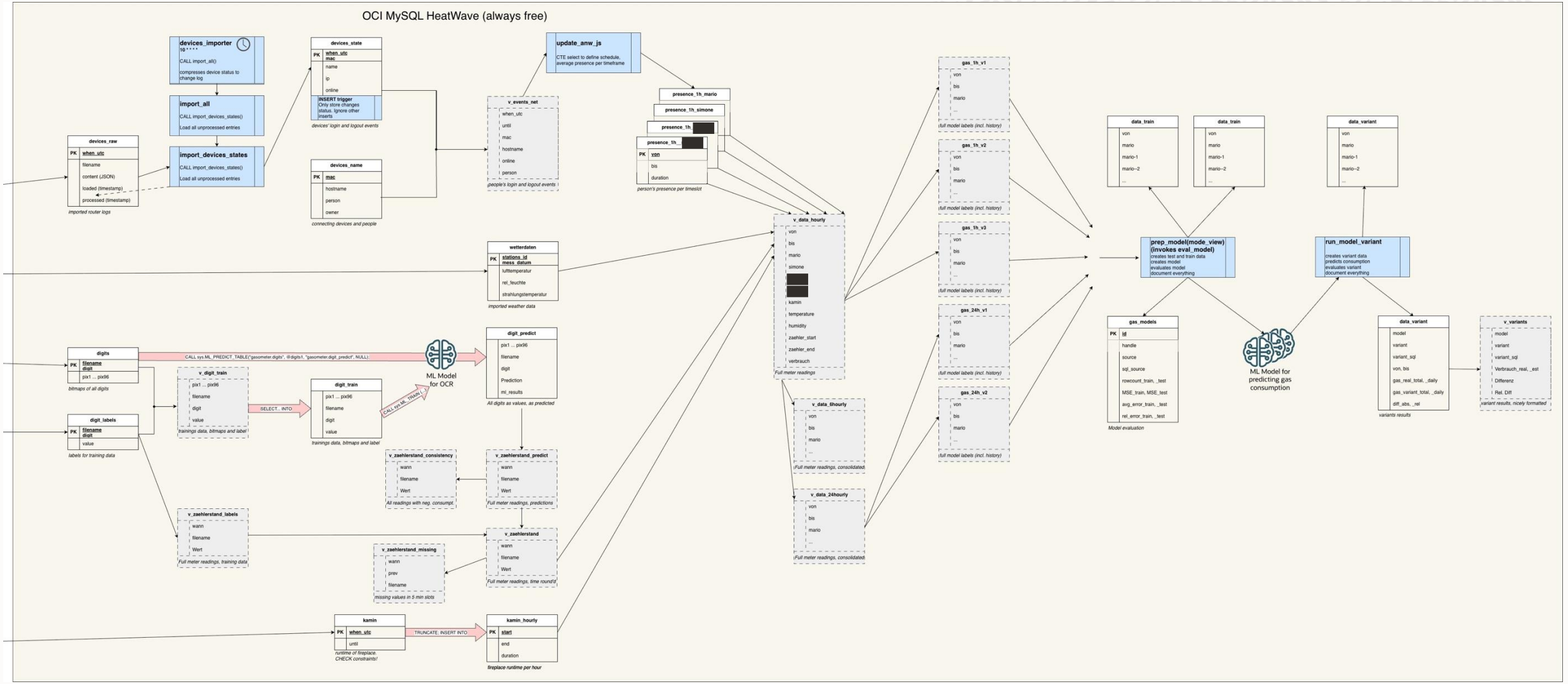


Use Case

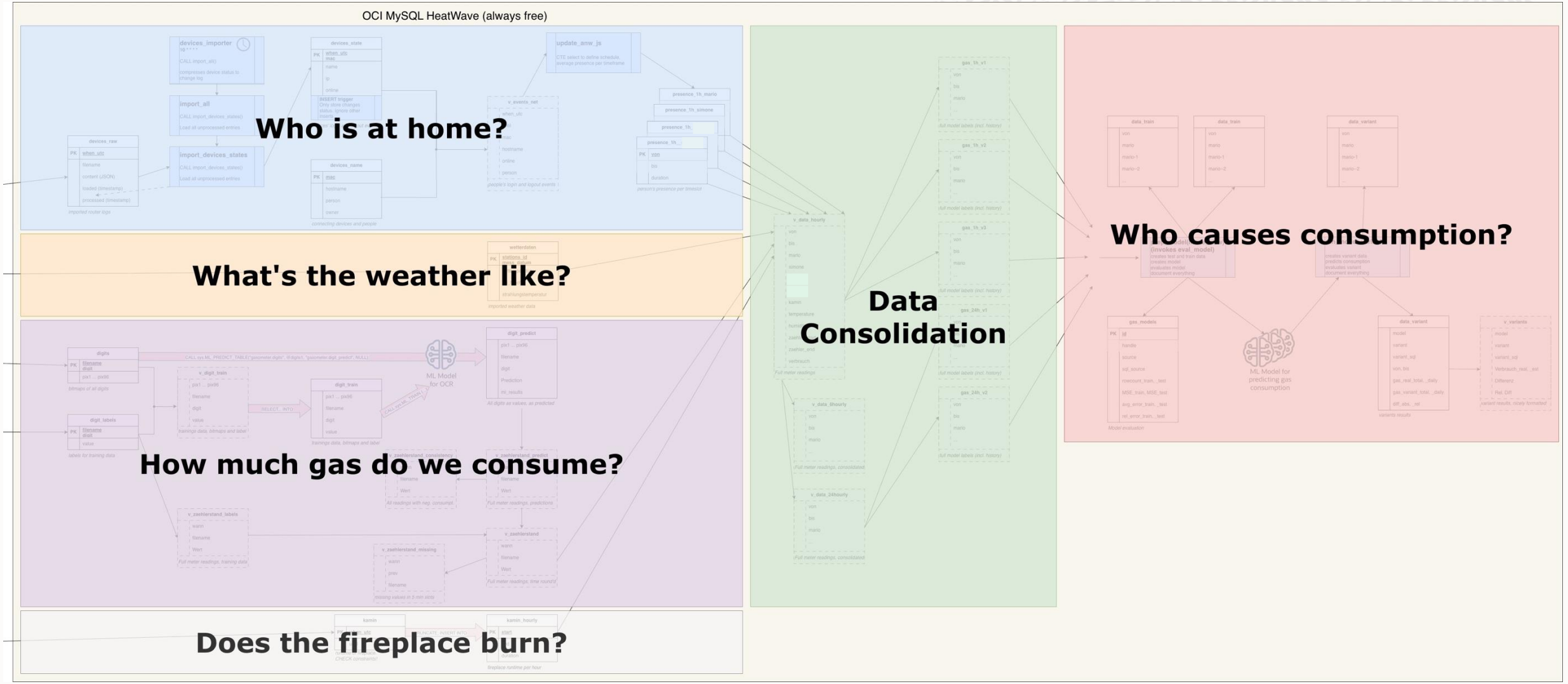
- Predict Gas Consumption based on
  - Outside temperature
  - People [not] at home
  - Fireplace [not] burning
- Take a photo every 5 minutes
  - Raspberry Pi
  - OCR is a nice ML use case
- Mobile phones in WLAN
  - Query WLAN Router for Devices
  - JSON data. Great!
- Weather Data from DWD
  - Freely available, hourly temp and humidity
- Fireplace… manually for now
- Ship data to OCI MySQL always free



Data Architecture

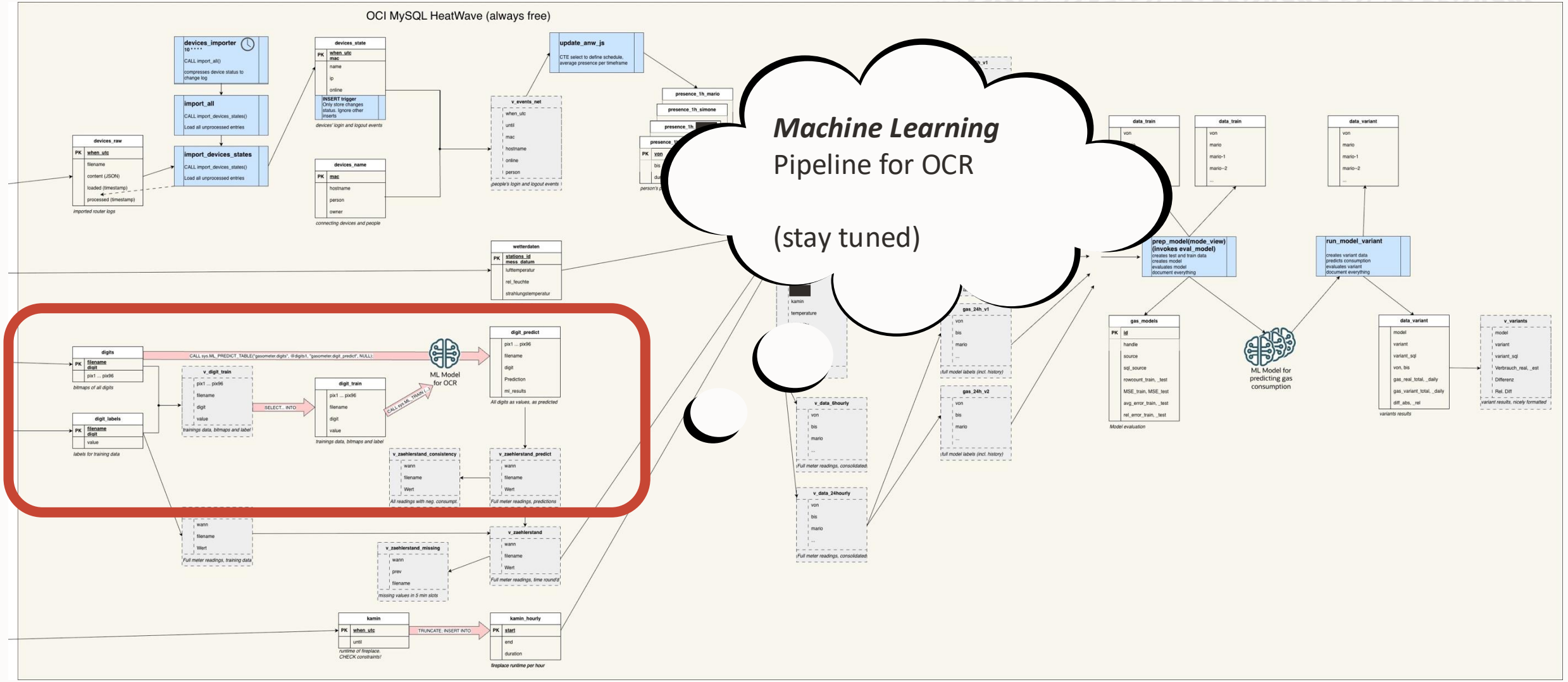


Data Architecture

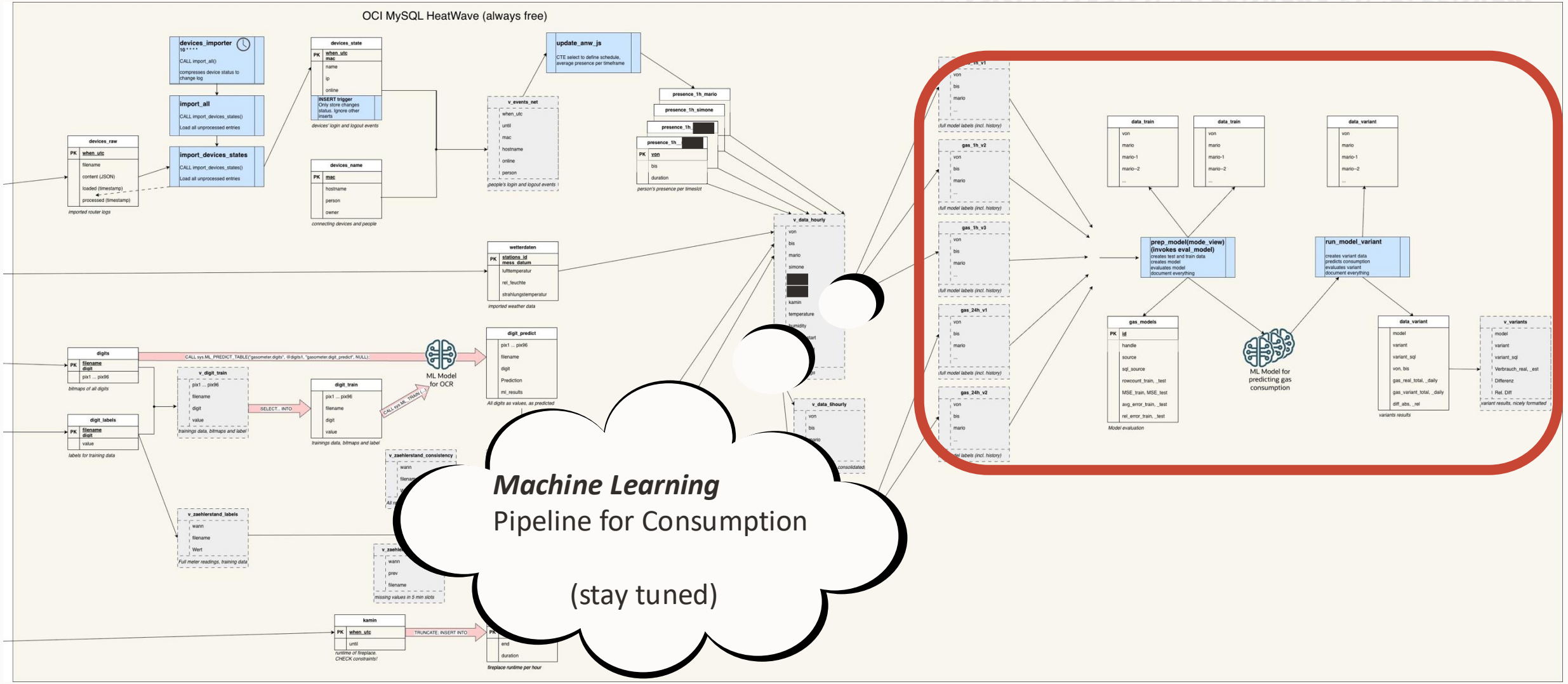


- How? – What Technologies / MySQL Features are used

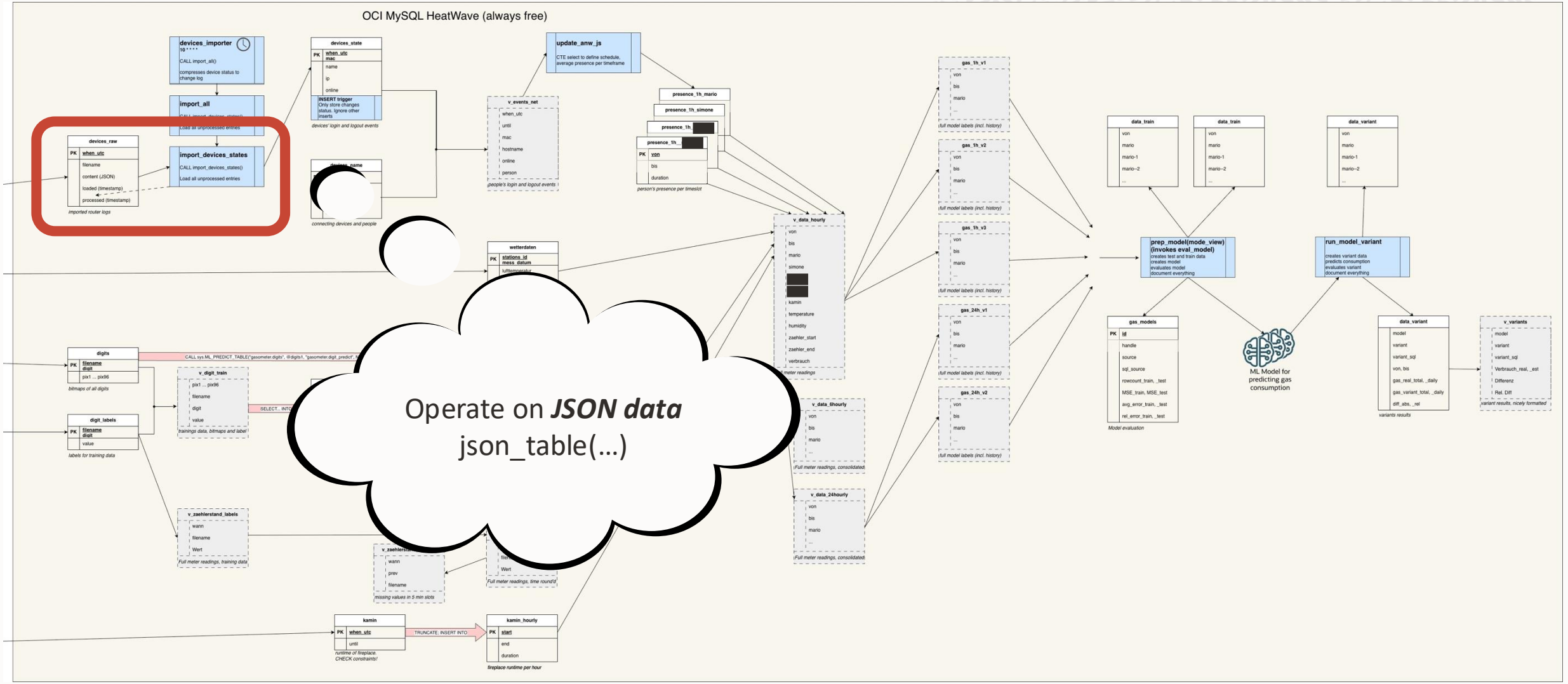
Different Technologies



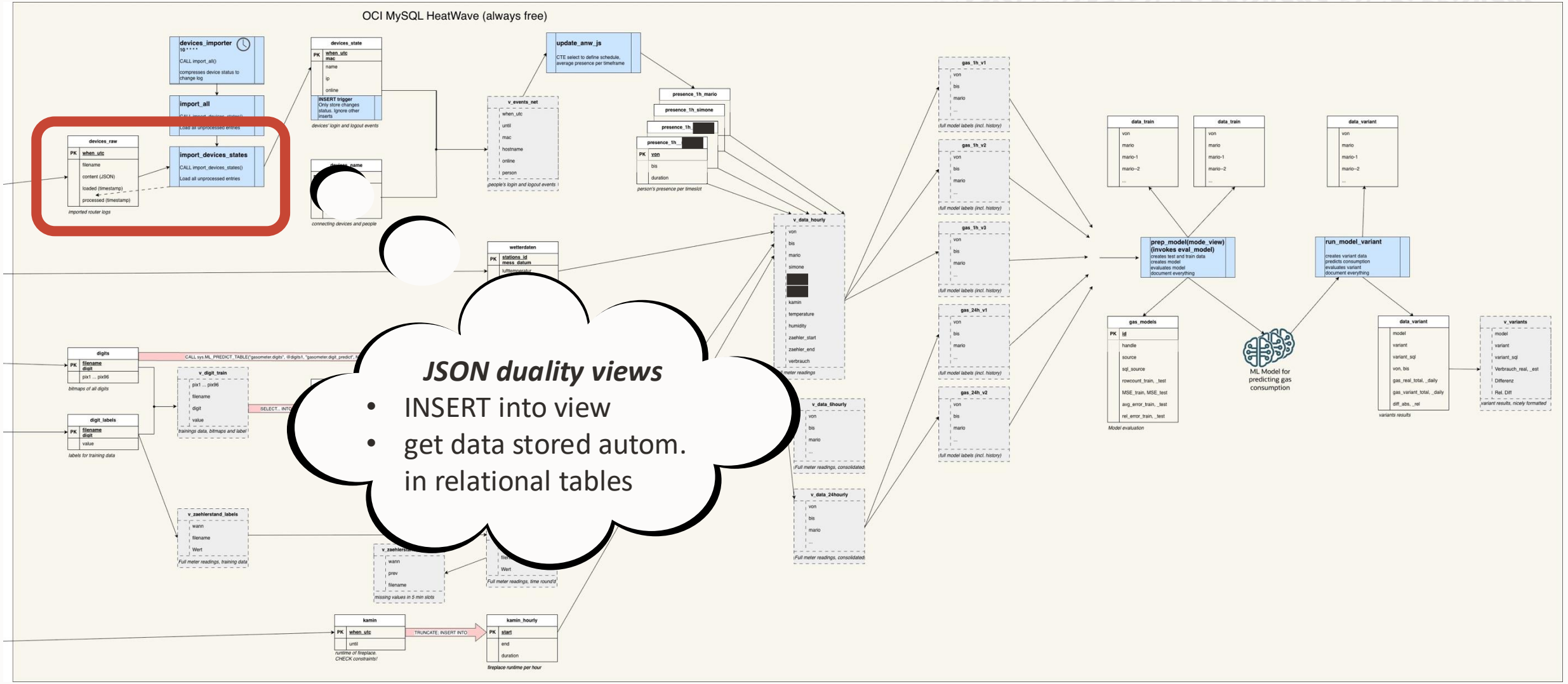
Different Technologies



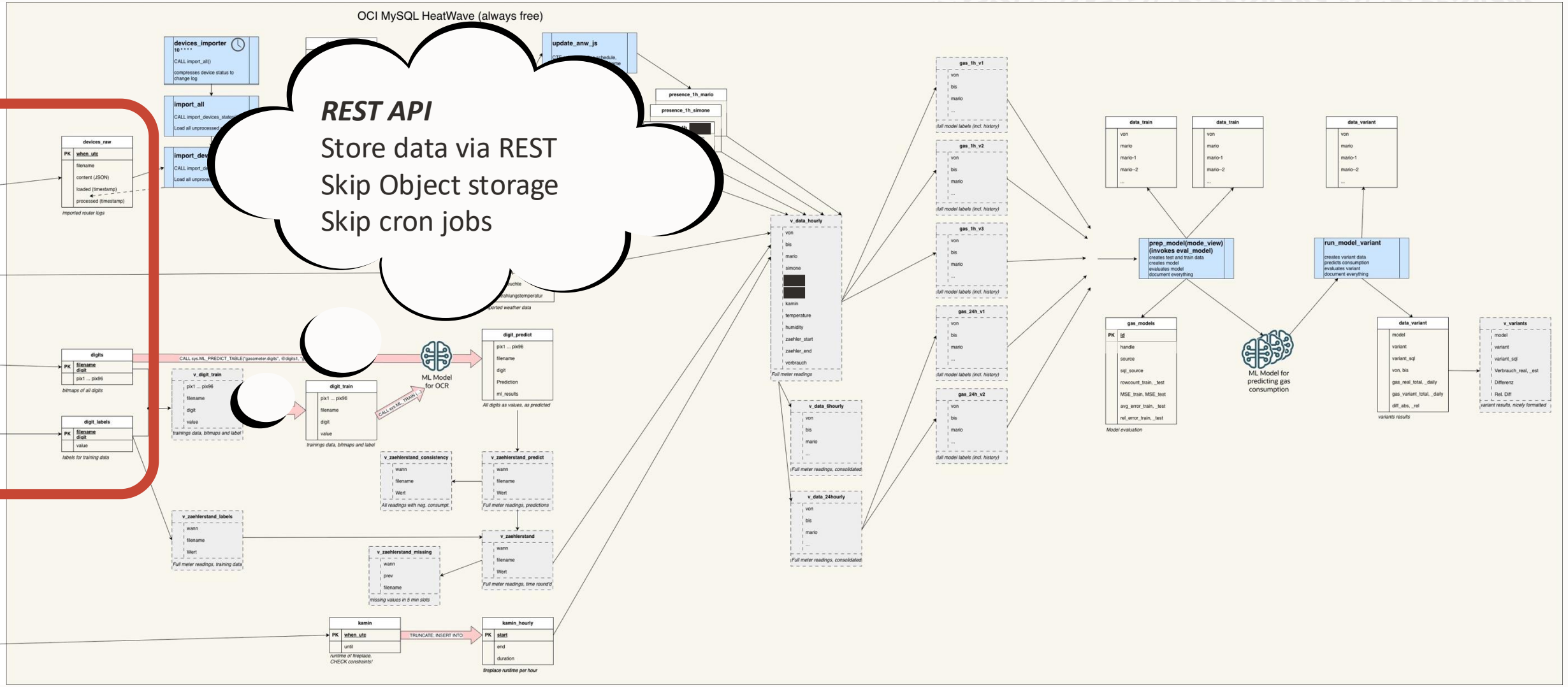
Different Technologies



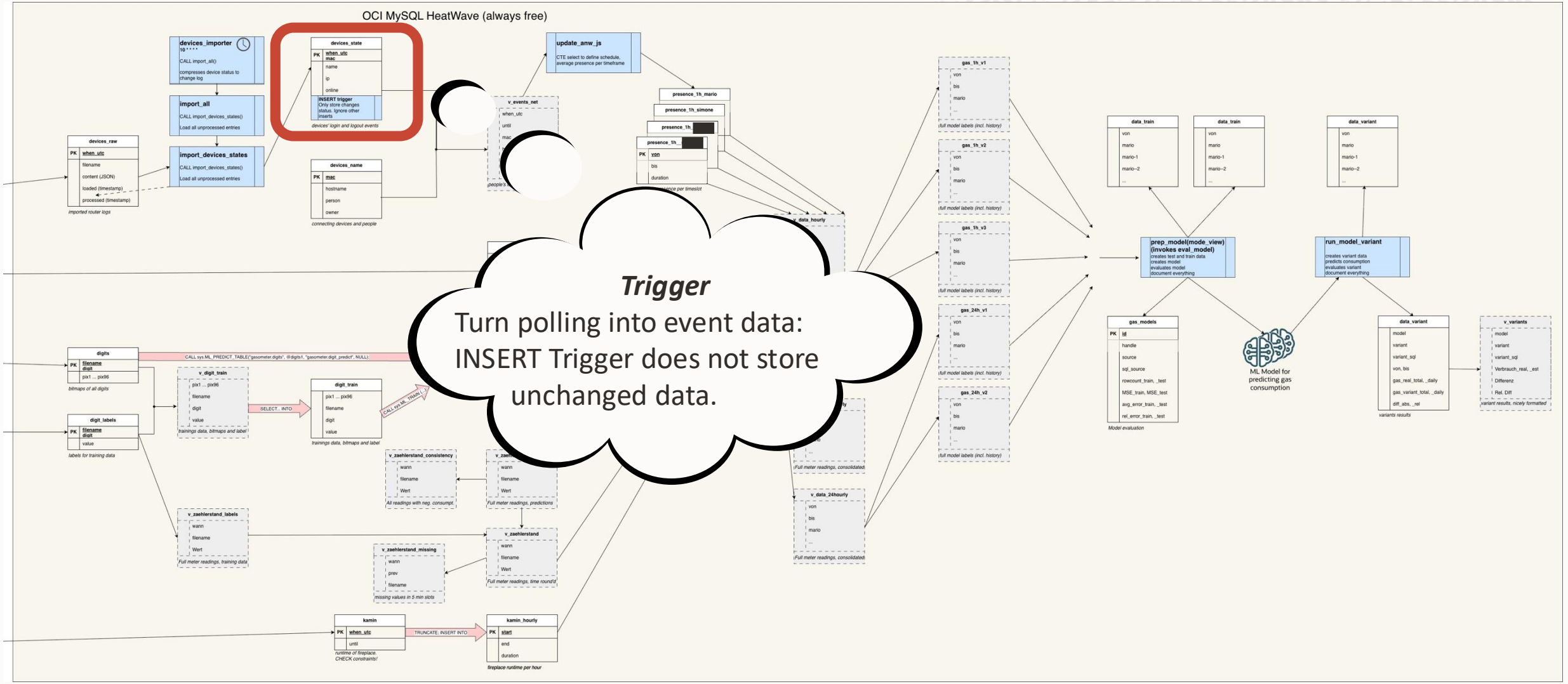
Different Technologies



Different Technologies



Different Technologies



Trigger

Turn polling into event data:
INSERT Trigger does not store unchanged data.

Trigger to store only changed data

- Trigger is executed BEFORE the insert.
- If the last state is also the new state, raise my personal SQL error -> Skip the INSERT
- In calling procedure, catch this SQL exception to continue execution.

```
CREATE TRIGGER `only_state_change` BEFORE INSERT ON `devices_state` FOR EACH ROW BEGIN

    DECLARE last_state BOOLEAN;

    SELECT online INTO last_state FROM devices_state
        WHERE devices_state.mac=NEW.mac ORDER BY devices_state.when_utc DESC LIMIT 1;

    IF last_state = NEW.online THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT="No state change. Row ignored"
    END IF;

END
```

CTE to reshape event-based data into fixed sample interval data

```
var stmt =
  "INSERT INTO " + tblname + " WITH RECURSIVE date_cte AS ( " +
    "SELECT CAST('" + von + "' AS DATETIME) AS von, " +
           "CAST('" + von + "' AS DATETIME) + " + interval + " AS bis " +
    "UNION ALL " +
    "SELECT von + " + interval + ", bis + " + interval + " " +
    "FROM date_cte " +
    "WHERE bis + " + interval + " < '" + bis + "' " +
  ") " +
  "SELECT /*+ SET_VAR(cte_max_recursion_depth=2000) */ t.*, " +
    "SUM(e.online * TIME_TO_SEC(TIMEDIFF(LEAST(t.bis, e.until), GREATEST(t.von, e.when_utc)))) / " +
      "TIME_TO_SEC(TIMEDIFF(t.bis, t.von)) AS duration " +
  "FROM v_events_net e " +
  "JOIN date_cte t " +
  "WHERE e.person='" + person + "' " +
    "AND e.when_utc <= t.bis " +
    "AND e.until >= t.von " +
  "GROUP BY t.von, t.bis, e.person " +
  "ORDER BY t.von;";

try {
rs = session.runSql(stmt);
```

Recursive CTE

Joined rows within the time window

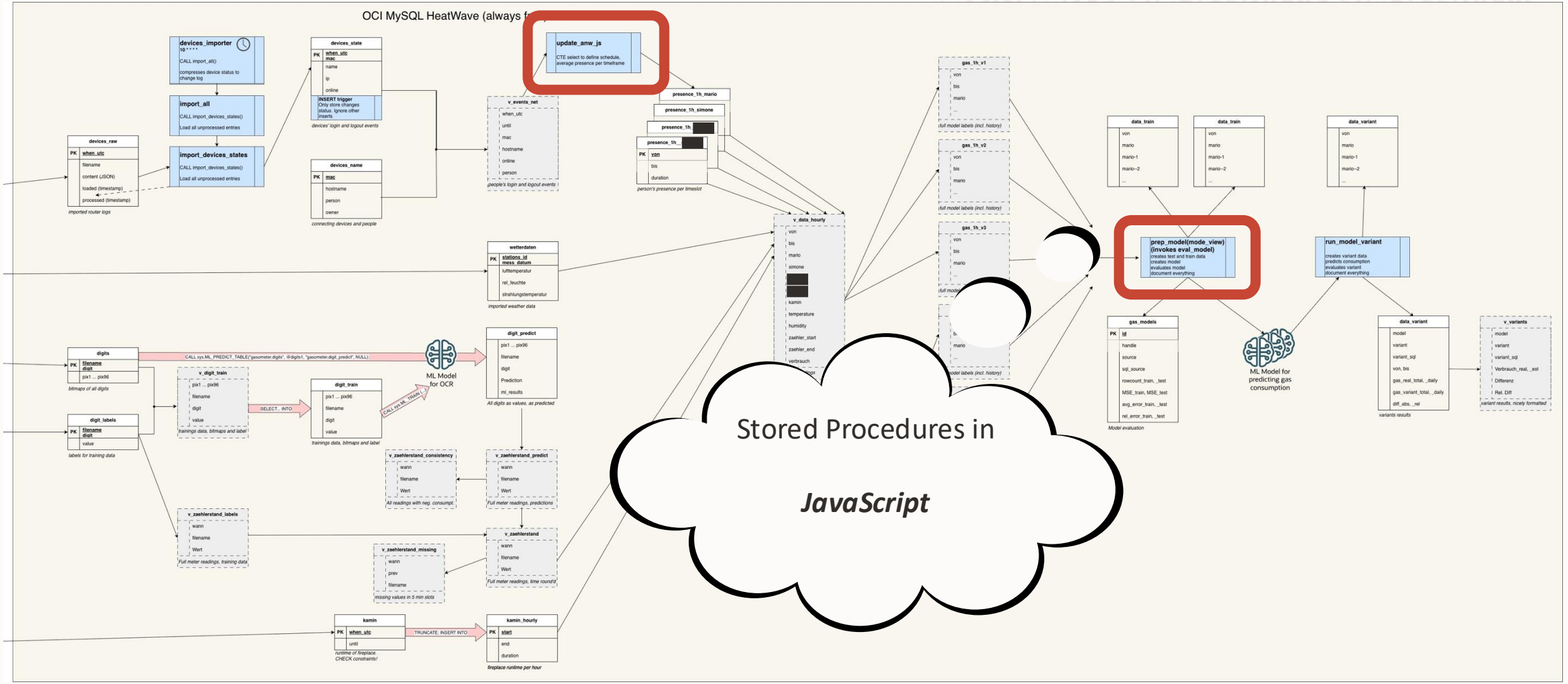
CTE to reshape event-based data into fixed sample interval data

```
var stmt =
  "INSERT INTO " + tblname + " WITH RECURSIVE date_cte AS ( " +
    "SELECT CAST('" + von + "' AS DATETIME) AS von, " +
           "CAST('" + von + "' AS DATETIME) + " + interval + " AS bis " +
    "UNION ALL " +
    "SELECT von + " + interval + ", bis + " + interval + " " " +
    "FROM date_cte " +
    "WHERE bis + " + interval + " < '" + bis + "' " " +
  ") " +
  "SELECT /*+ SET_VAR(cte_max_recursion_depth=2000) */ t.*, " +
    "SUM(e.online * TIME_TO_SEC(TIMEDIFF(LEAST(t.bis, e.until), GREATEST(t.von, e.when_utc)))) / " +
      "TIME_TO_SEC(TIMEDIFF(t.bis, t.von)) AS duration " +
  "FROM v_events_net e " +
  "JOIN date_cte t " +
  "WHERE e.person='" + person + "' " +
    "AND e.when_utc <= t.bis " +
    "AND e.until >= t.von " +
  "GROUP BY t.von, t.bis, e.person " +
  "ORDER BY t.von;";

try {
rs = session.runSql(stmt);
```

Optimizer hint

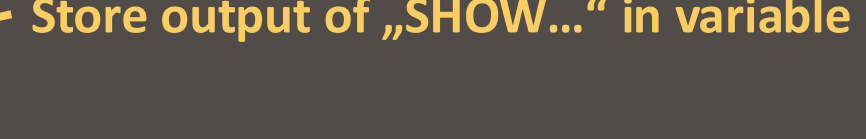
JavaScript

Different Technologies



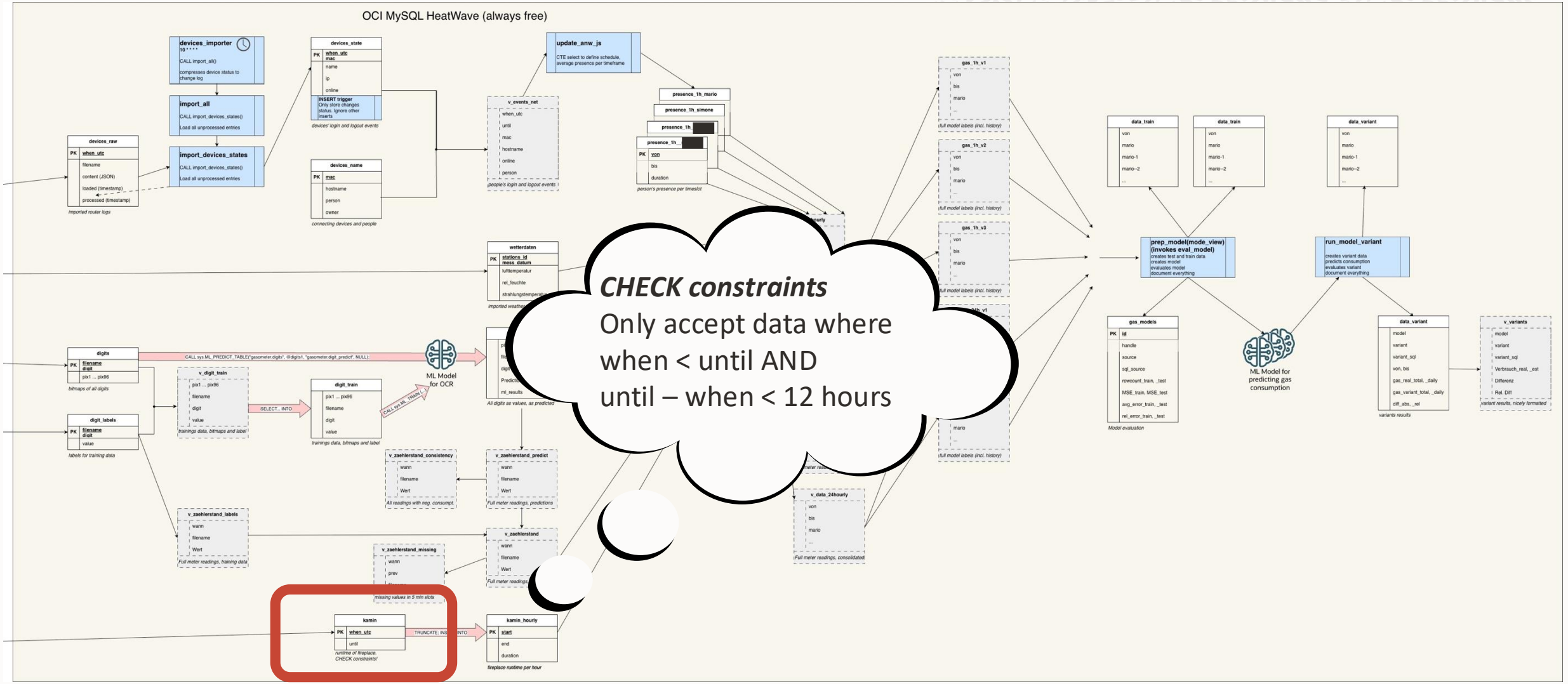
JavaScript can do more than SQL

```
CREATE PROCEDURE prep_model(model_view VARCHAR(100))
LANGUAGE JAVASCRIPT
AS
'
    var rs = session.runSql("SHOW CREATE VIEW "+model_view);
    var row = rs.fetchOne();
    // The CREATE TABLE output is typically in the second column
    var create_sql = row[1];
    // Update the gas_models table
    var ps = session.prepare("INSERT INTO gas_models (id,source,sql_source)
VALUES(NULL,?,?)").bind(model_view, create_sql);
    ps.execute();
    ps.deallocate();
```



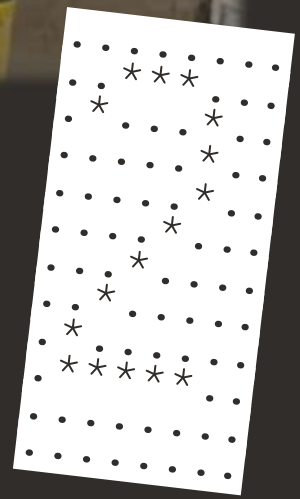
- Handle data-intensive app functionality in stored programs
  - Minimize data movement
  - Reduce cost
  - Improve Security
  - Simplify complex ETL → ELT

Different Technologies



Machine Learning for OCR

- ML OCR / ODR: Classification Task
- Digits cut from photo
- Transformed to 8x12 pixels
- 96 features get classified to class 0…9
- Know your data:
  - Readings are monotonically increasing
  - Rightmost digits are not important



Machine Learning for OCR

- Prep notebook
- connect to database

```
#!pip3 install numpy
#!pip3 install tensorflow
#!pip3 install mysql-connector-python
#!pip3 install scikit-learn
#!pip3 install matplotlib

import math
import matplotlib.pyplot as plt
import mysql.connector
import numpy as np
import tensorflow as tf
from tensorflow.keras import layers, models
from tensorflow.keras.utils import to_categorical
from sklearn.model_selection import train_test_split
from IPython.display import Image, display

config = {
    'user': 'admin',
    'password': 'xxxxxxxxxx',
    'host': '10.0.1.205',
    'database': 'gasometer',
    'raise_on_warnings': True
}
```

Machine Learning for OCR

- Prep notebook
- connect to database
- Load data to Numpy

```
#!pip3 install numpy
#!pip3 install tensorflow
#!pip3 install mysql-connector-python
#!pip3 install scikit-learn
```

```
def fetch_table_as_numpy_array(sql_statement):
    try:
        connection = mysql.connector.connect(**config)
        if connection.is_connected():
            cursor = connection.cursor()
            cursor.execute(sql_statement)
            rows = cursor.fetchall()
            column_names = [i[0] for i in cursor.description]
            numpy_array = np.array(rows)
            print("Spaltennamen:", column_names)
            return numpy_array

    except mysql.connector.Error as err:
        print(f"Fehler: {err}")
    finally:
        # Verbindung schließen
        if 'connection' in locals() and connection.is_connected():
            cursor.close()
            connection.close()
            print("MySQL-Verbindung ist geschlossen")

# Tabelle "train" als NumPy-Array abrufen
# v_digit_train enthaelt alle Digits. v_train enthält alle außer digit=7. Die letzte Ziffer ist oft entstellt und ver
full_array = fetch_table_as_numpy_array("SELECT filename, digit, pix1, pix2, pix3, pix4, pix5, pix6, pix7, pix8, pix9
meta_array = full_array[:, [0,1,98]]
train_array = full_array[:,2:].astype(int)

# Ausgabe des NumPy-Arrays
#print(train_array[:3])
```

Machine Learning for OCR

- Prep notebook
- connect to database
- Load data to Numpy
- Split Train/Test Data

```
#!pip3 install numpy
```

```
X = train_array[:, :96]  # All rows, first 96 columns
X = X / 255.0 - .5       # normalize to -.5 to +.5
num_samples, num_pixels = X.shape       # reshape because we need a 12x8 matrix again
X = np.reshape(X, shape=(num_samples,12,8,1))

y = train_array[:, -1]   # All rows, only last column
y = to_categorical(y, 10)   # change to array of 10 binaries for classification

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.1)
```

```
def
```

```
            return numpy_array

    except mysql.connector.Error as err:
        print(f"Fehler: {err}")
    finally:
        # Verbindung schließen
        if 'connection' in locals() and connection.is_connected():
            cursor.close()
            connection.close()
            print("MySQL-Verbindung ist geschlossen")

# Tabelle "train" als NumPy-Array abrufen
# v_digit_train enthaelt alle Digits. v_train enthält alle außer digit=7. Die letzte Ziffer ist oft entstellt und ver
full_array = fetch_table_as_numpy_array("SELECT filename, digit, pix1, pix2, pix3, pix4, pix5, pix6, pix7, pix8, pix9
meta_array = full_array[:, [0,1,98]]
train_array = full_array[:,2:].astype(int)

# Ausgabe des NumPy-Arrays
#print(train_array[:3])
```

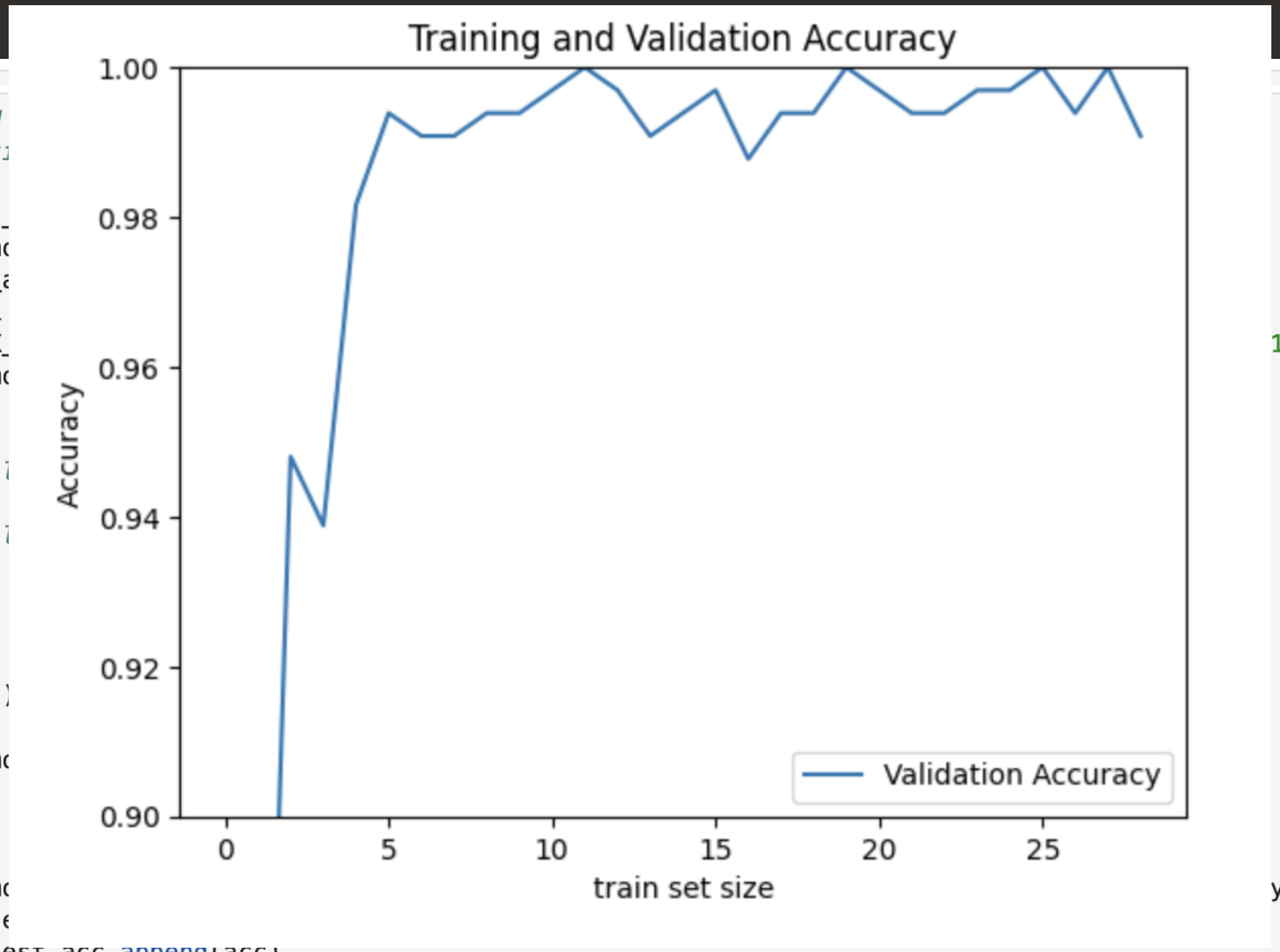
Machine Learning for OCR

- Prep notebook
- connect to database
- Load data to Numpy
- Split Train/Test Data
- Pick algorithm (CNN)
- Setup model
- Guess Hyperparams

```
# CNN Model setup and compile and train
# Optimierung der Trainings-Größe

batch_size=128
num_models = math.ceil(num_samples*0.9/100)
test_acc = []
for i in range(0,num_models-1):
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.1, train_size=(i+1)*100)
    model = models.Sequential([
        layers.Input(shape=(12, 8, 1)),  # Eingabeschicht mit der Form (12, 8, 1)
        layers.Conv2D(32, (3, 3), activation='relu'),
#     layers.MaxPooling2D((2, 2)),
        layers.Conv2D(64, (3, 3), activation='relu'),
#     layers.Conv2D(10, (3, 3), activation='relu'),
        layers.MaxPooling2D((2, 2)),
        layers.Flatten(),
        layers.Dense(64, activation='relu'),
        layers.Dense(10, activation='softmax')
    ])

    model.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])

    model.fit(X_train, y_train, epochs=20, batch_size=batch_size, validation_data=(X_test, y_test), ve
    test_loss, acc = model.evaluate(X_test, y_test, verbose=0)
    test_acc.append(acc)
    print("Lauf #"+str(i)+"/"+str(num_models-1)+" trainset=" + str((i+1)*100)+" Accuracy="+str(acc))
```

Machine Learning for OCR

- Prep notebook
- connect to database
- Load data to Numpy
- Split Train/Test Data
- Pick algorithm (CNN)
- Setup model
- Guess Hyperparams
- Tune 1 parameter



Machine Learning for OCR

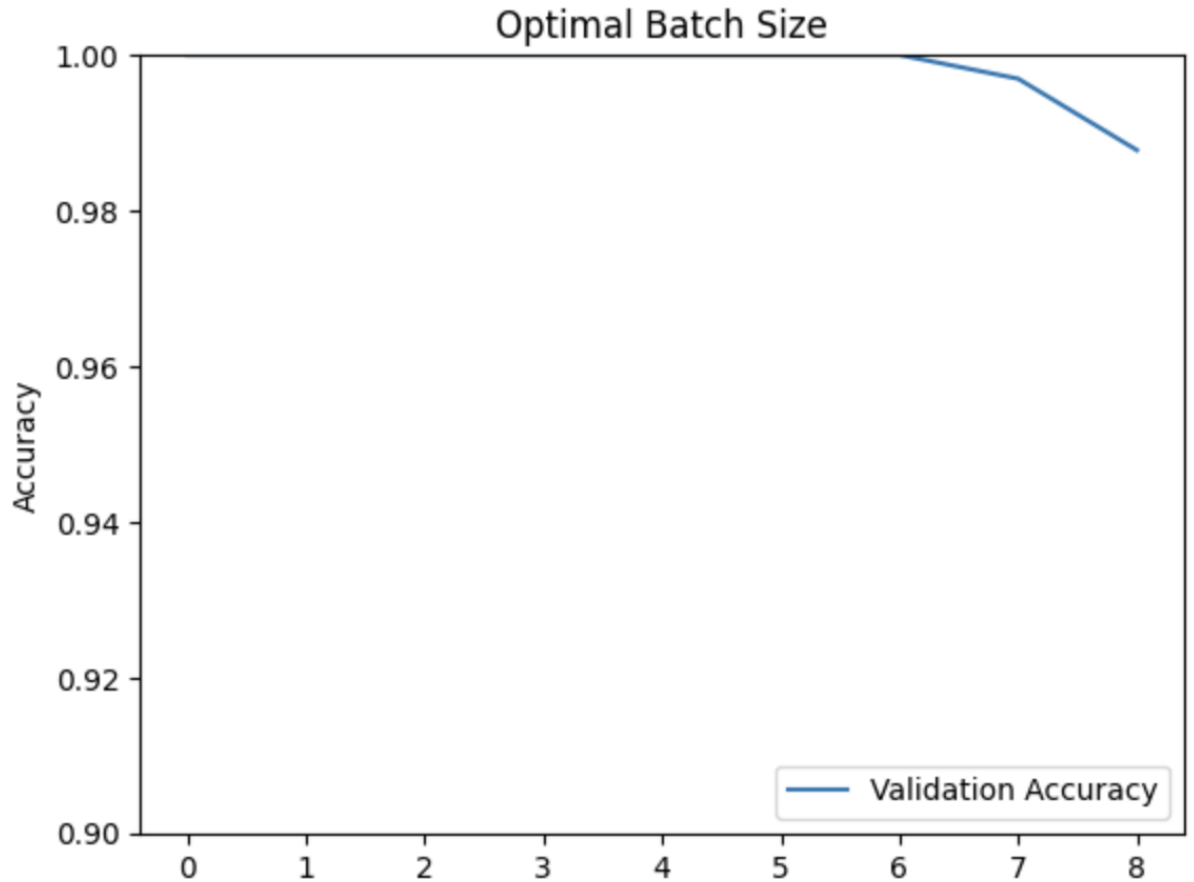
- Prep notebook
- connect to database
- Load data to Numpy
- Split Train/Test Data
- Pick algorithm (CNN)
- Setup model
- Guess Hyperparams
- Tune 1 parameter
- Optimize 2nd HP

```
# CNN Model setup and compile and train
# Optimierung der Trainings-Größe

training_size = 2700
num_models = [4,8,16,32,64,128,256,512,1024]

test_acc = []
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.1, train_size=training_s

for i in num_models:
    model = models.Sequential([
        layers.Input(shape=(12, 8, 1)),  # Eingabeschicht mit der Form (12, 8, 1)
        layers.Conv2D(32, (3, 3), activation='relu'),
#     layers.MaxPooling2D((2, 2)),
        layers.Conv2D(64, (3, 3), activation='relu'),
#     layers.Conv2D(10, (3, 3), activation='relu'),
        layers.MaxPooling2D((2, 2)),
        layers.Flatten(),
        layers.Dense(64, activation='relu'),
        layers.Dense(10, activation='softmax')
    ])

    model.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])

    model.fit(X_train, y_train, epochs=20, batch_size=i, validation_data=(X_test, y_test), ve
    test_loss, acc = model.evaluate(X_test, y_test, verbose=0)
    test_acc.append(acc)
    print("Batch Size: "+str(i)+" Accuracy="+str(acc))

# plot the training history
```

Machine Learning for OCR

- Prep notebook
- connect to database
- Load data to Numpy
- Split Train/Test Data
- Pick algorithm (CNN)
- Setup model
- Guess Hyperparams
- Tune 1 parameter
- Optimize 2nd HP
- Give up on optimizing

```
Batch Size: 4 Accuracy=1.0
Batch Size: 8 Accuracy=1.0
Batch Size: 16 Accuracy=1.0
Batch Size: 32 Accuracy=1.0
Batch Size: 64 Accuracy=1.0
Batch Size: 128 Accuracy=1.0
Batch Size: 256 Accuracy=1.0
Batch Size: 512 Accuracy=0.9969419240951538
Batch Size: 1024 Accuracy=0.9877675771713257
```



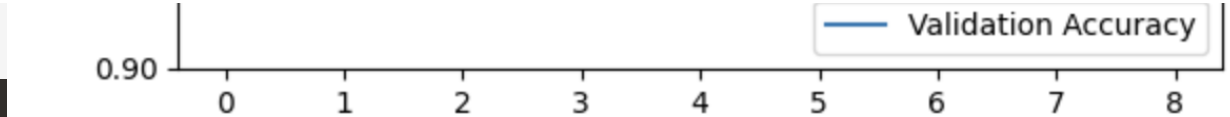
Machine Learning for OCR

- Prep notebook
- connect to database
- Load data to Numpy
- Split Train/Test Data
- Pick algorithm (CNN)
- Setup model
- Guess Hyperparams
- Tune 1 parameter
- Optimize 2nd HP
- Give up on optimizing
- Build final model
- Accuracy 99,1%

Batch Size: 4 Accuracy=1.0

```
# building the one model with optimal hyperparameters

batch_size=128
train_size=2700

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.1, train_size=train_size)
model = models.Sequential([
        layers.Input(shape=(12, 8, 1)),  # Eingabeschicht mit der Form (12, 8, 1)
        layers.Conv2D(32, (3, 3), activation='relu'),
#    layers.MaxPooling2D((2, 2)),
        layers.Conv2D(64, (3, 3), activation='relu'),
#    layers.Conv2D(10, (3, 3), activation='relu'),
        layers.MaxPooling2D((2, 2)),
        layers.Flatten(),
        layers.Dense(64, activation='relu'),
        layers.Dense(10, activation='softmax')
    ])

model.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])

model.fit(X_train, y_train, epochs=20, batch_size=batch_size, validation_data=(X_test, y_test), verb
test_loss, acc = model.evaluate(X_test, y_test, verbose=0)
print("Final model:"+" Accuracy="+str(acc))
```

```
Final model: Accuracy=0.9908257126808167
```



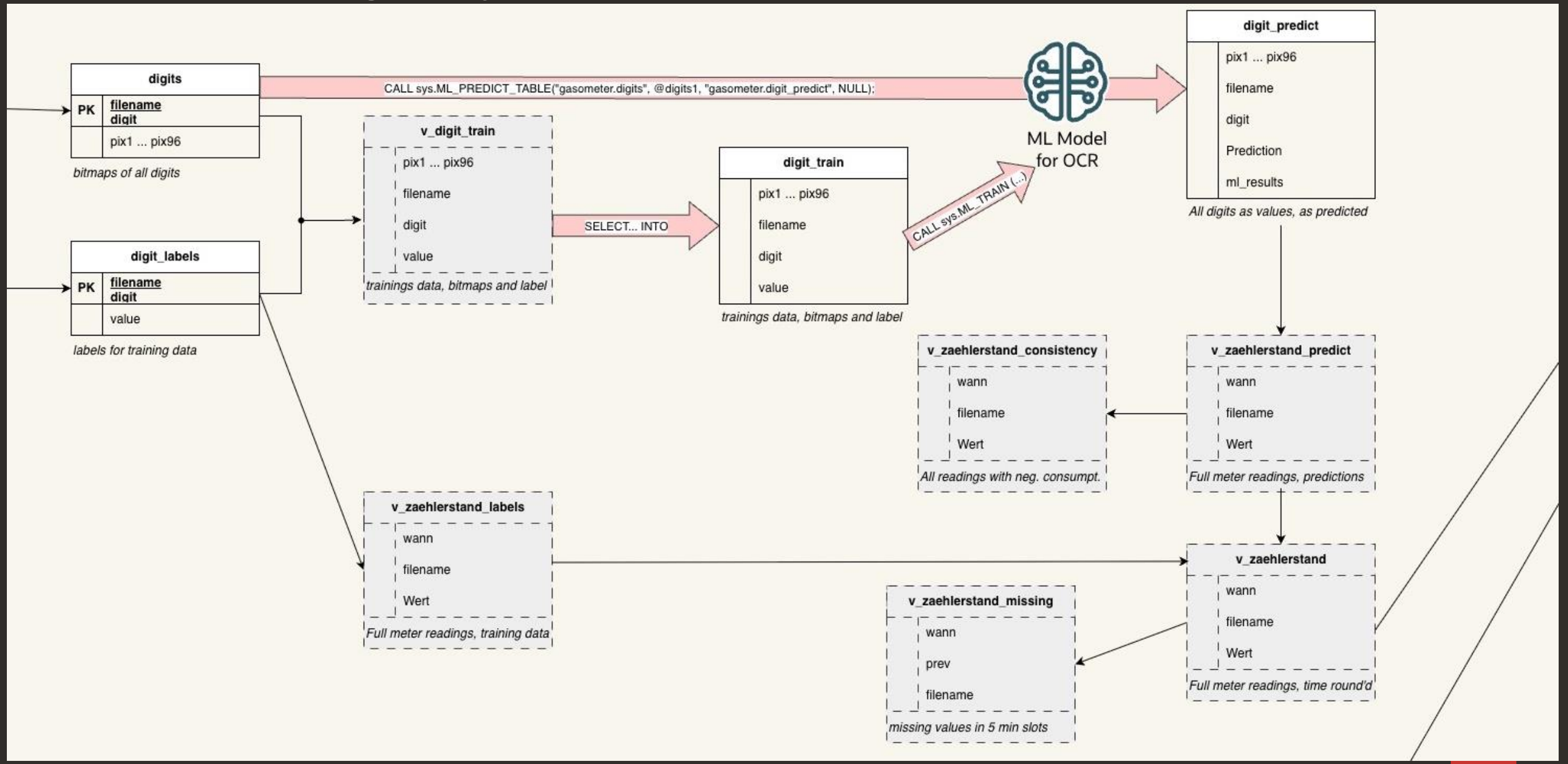
Machine Learning for OCR

- Prep notebook
- Give up c
- Build fina
- Accuracy 99,1%

```
# SQL statements to achieve the same in HeatWave
"""
CREATE TABLE digit_train SELECT * FROM v_digit_train;

# learning took 30 minutes on 1 core
CALL sys.ML_TRAIN ('gasometer.digit_train', 'value',
     JSON_OBJECT('task', 'classification',
                 'optimization_metric', 'accuracy',
                 'exclude_column_list', JSON_ARRAY('digit','filename')
                ), @digits1);
```

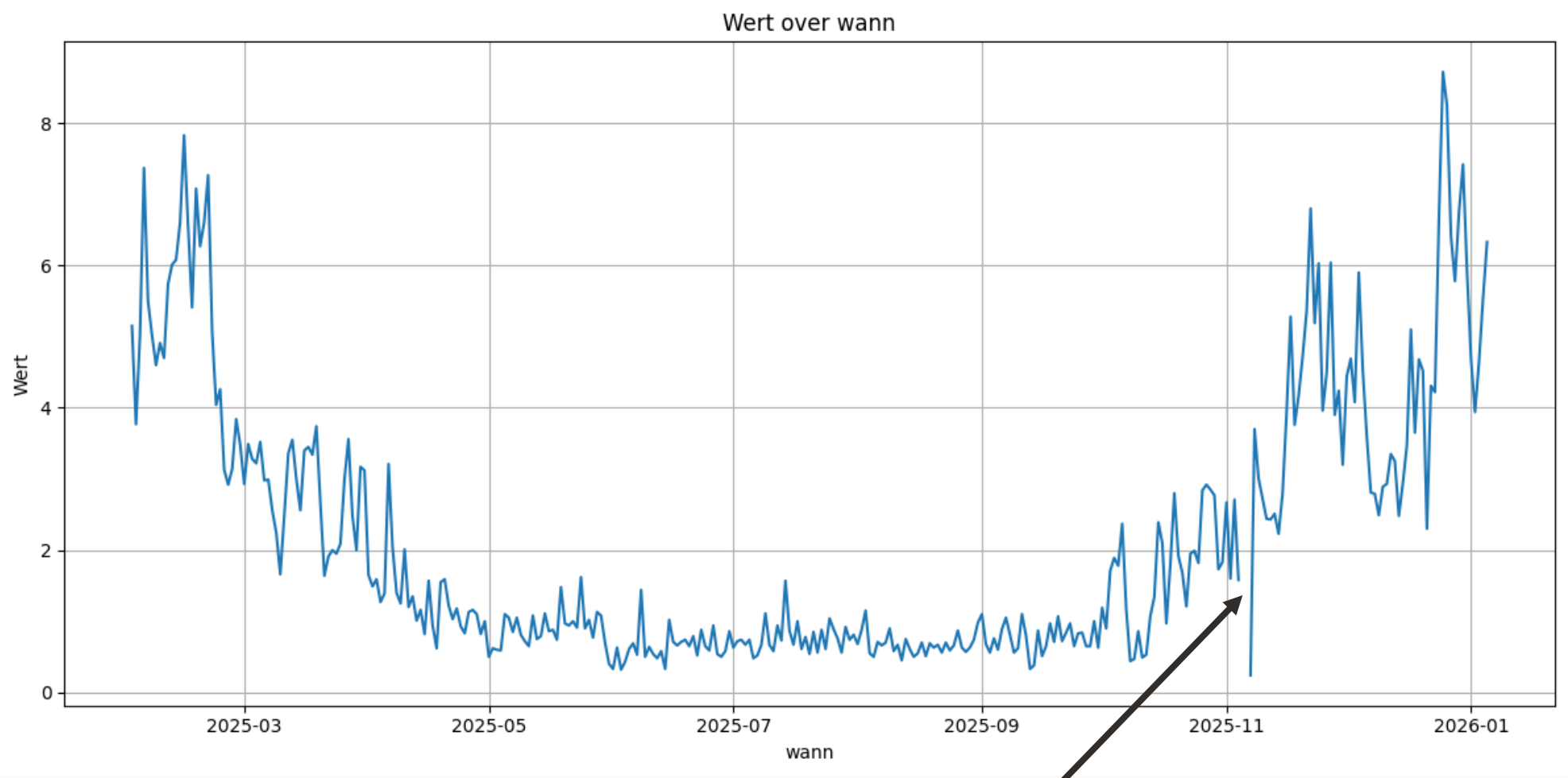
- One statement – 30 min on 1 core or 2 min on 16 cores
- Accuracy: 99.1%

Machine Learning Example: Gasometer



- What? – Any useful results?

Gas Consumption per Day



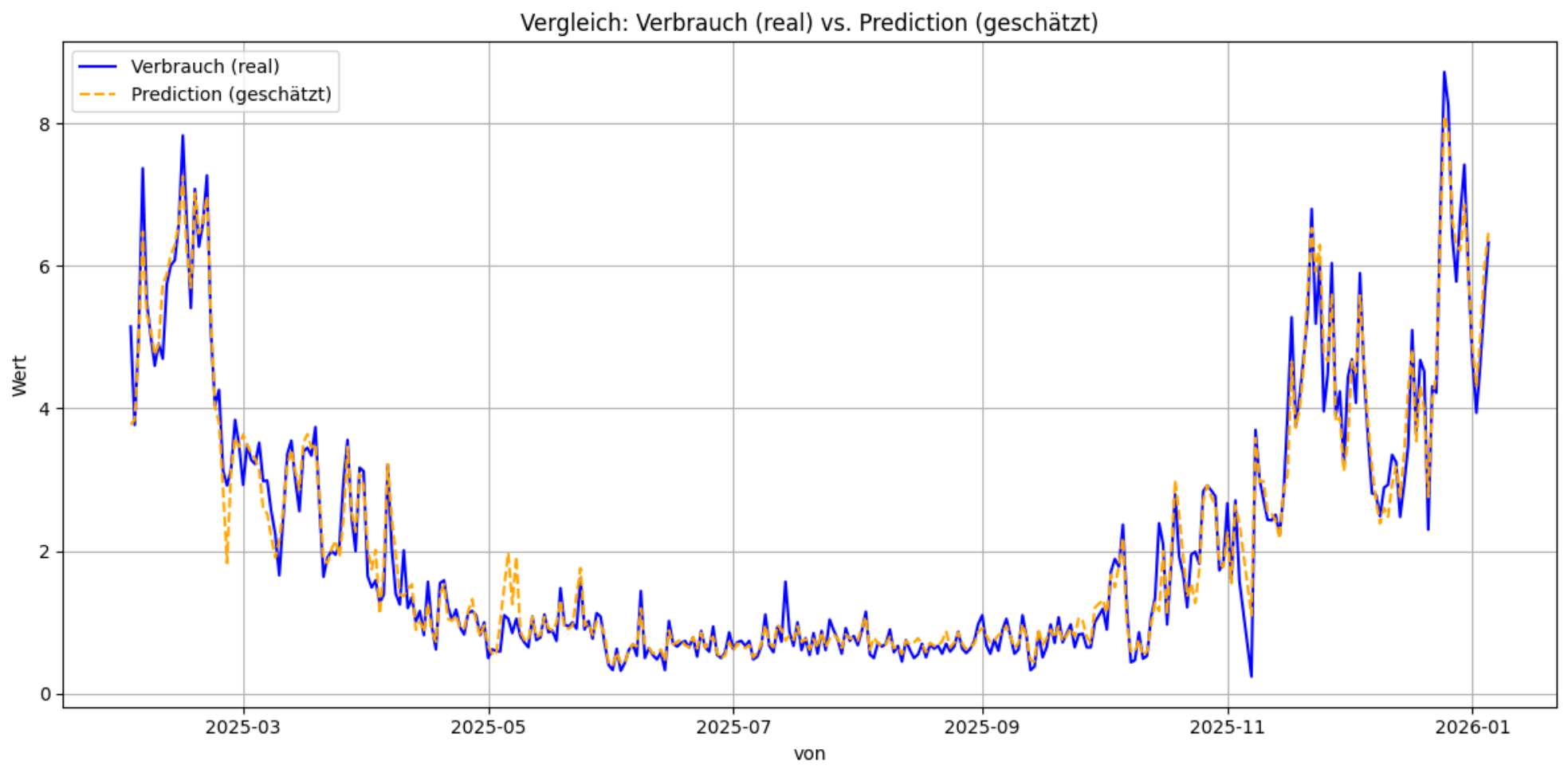
Gap in data (Learning: Always monitor!)

Gas Prediction Models

- Different models have different feature sets
- 1h is too fine-grained for proper accuracy
- 24h with 5 days of temperature history is best
- MSE_train==0…oh, oh… did it memorize?
- Unit of MSE is m^6. Can't compare 1h to 24h
- Best daily prediction model is within +/- 16.7%

handle	source	# train	# test	MSE_train	MSE_test	rel_err.train	rel_err.test
BASE_1h	v_data_hourly	8414	8414	0.0129	0.0129	98.5 %	98.5 %
gas_1	gas_6h_v2	1083	258	0.0241	0.0422	21.4 %	28.5 %
gas_8	gas_24h_v2	267	69	0.0000	0.3561	0.0 %	18.8 %
gas_11	gas_24h_v2	274	62	0.0000	0.3147	0.0 %	19.6 %
gas_12	gas_24h_v2	261	75	0.0000	0.3460	0.0 %	16.7 %
gas_3	gas_24h_v1	273	63	0.1333	0.3996	11.9 %	19.1 %
gas_9	gas_24h_v1	278	58	0.0000	0.4295	0.0 %	22.0 %
gas_10	gas_24h_v1	268	68	0.0433	0.2430	6.7 %	18.3 %
gas_6	gas_1h_v3	6495	1539	0.0040	0.0045	44.1 %	50.4 %
gas_13	gas_1h_v3	6430	1604	0.0029	0.0052	38.6 %	50.9 %
gas_5	gas_1h_v2	6457	1577	0.0038	0.0052	42.6 %	49.4 %
gas_4	gas_1h_v1	6384	1650	0.0041	0.0051	45.3 %	48.2 %

Gas Consumption vs. Prediction



What-if Scenarios for model “gas_12”

- People have little influence
- Best if Mario stays away
- My son can keep his GF
- The fireplace is useless
- Climate change is risky

variant	Verbrauch real	Verbrauch est.	Differenz	rel. Diff.
base	708.3 qm	711.1 qm	2.8	0.4 %
mario%=0	708.3 qm	688.0 qm	-20.3	-2.9 %
mario%=1	708.3 qm	720.2 qm	11.9	1.7 %
simone%=0	708.3 qm	688.1 qm	-20.2	-2.8 %
simone%=1	708.3 qm	728.5 qm	20.2	2.9 %
son%=0	708.3 qm	691.1 qm	-17.2	-2.4 %
son%=1	708.3 qm	746.6 qm	38.4	5.4 %
girlfr%=0	708.3 qm	711.1 qm	2.8	0.4 %
girlfr%=1	708.3 qm	711.1 qm	2.8	0.4 %
kamin%=0	708.3 qm	711.1 qm	2.8	0.4 %
kamin%=1	708.3 qm	711.1 qm	2.8	0.4 %
temp +1	708.3 qm	648.2 qm	-60.1	-8.5 %
temp -1	708.3 qm	780.9 qm	72.6	10.3 %
temp +2	708.3 qm	579.2 qm	-129.1	-18.2 %
temp -2	708.3 qm	860.7 qm	152.4	21.5 %

- Wow! – What did I learn

Some Learnings

- WLAN presence detection is weak
  - Weak WLAN -> Switch back to mobile signal -> “not at home”
  - My mother committed burglary! …. Or passed by our house and logged into our WLAN
  - **Easy to collect, easy to draw wrong conclusions**
- Machine Learning
  - Know your data
  - **Easy to misinterpret results – Know what you are doing**
  - OCR: Work on image quality!
  - Accuracy is too low in the gasometer models to identify small effects
- Project in General
  - Document everything – You will have forgotten everything after one month
  - MONITOR!!! Logs are useful, alarms are useful. (Hopefully backups are not…)
  - MySQL might not be dedicated to certain use cases, but is generically powerful and versatile
  - **Playing is the best way to learn!**

Q&A

Thank you!

Mario.beck@oracle.com


ORACLE