Smart City IoT – Intelligentes Energiemanagementsystem

Teilnehmerinnen / Teilnehmer:
Mahrokh Javadi
Shima Maheronnaghsh

Kurs / Modul:
Data Engineering – Abschlussprojekt

Schulungsanbieter / Unternehmen:
alfatraining Bildungszentrum

Agenda

1. Einführung & Projektziele
2. Datenquellen und Datenstruktur
3. Data Engineering Pipeline
4. ER- & Relationales Modell
5. Normalisierung (1NF – 3NF)
6. Data Warehouse & DWH-Fragen
7. Slowly Changing Dimensions (SCD)
8. Tools & Technologien
9. Ergebnisse & Nutzen
10. Fazit & Ausblick

Einführung & Projektziele

Hintergrund

- In modernen Städten erzeugen IoT-Sensoren täglich riesige Mengen an Energiedaten.
- Diese Daten sind häufig unstrukturiert, redundant oder unvollständig.
- Eine saubere und konsistente Datenbasis ist erforderlich, um fundierte Analysen zu ermöglichen.

Projektziel

- Aufbau eines integrierten Energiemanagementsystems für Smart Cities.
- Entwicklung einer vollständigen Data Engineering Pipeline:

Datenerfassung → Bereinigung → Integration → Modellierung → Analyse.

- Bereitstellung strukturierter, qualitativ hochwertiger und vertrauenswürdiger Daten

als Grundlage für Business Intelligence und Data Science.

- Erkennung und Vorhersage zukünftiger Energie-Trends zur Unterstützung datengetriebener Entscheidungen.
- Beitrag zur Digitalisierung und Nachhaltigkeit im Energiesektor.

Wissenschaftliche Relevanz

- Nutzung datengetriebener Methoden zur Optimierung des Energieverbrauchs.
- Verbindung von IoT, Datenmodellierung und Analytik.
- Grundlage für Predictive Analytics und intelligente Entscheidungsprozesse in Smart Cities.

Einschränkungen / Randbedingungen

Strukturelle Einschränkungen

- Eine Lokation → mehrere Sensoren | Ein Sensor → nur eine Lokation
- Eine Lokation → mehrere Wettermessungen | Eine Wettermessung → nur eine Lokation
- Ein Sensor → mehrere EnergieAblesungen | Eine Ablesung → nur ein Sensor

Daten- und Integritätsbedingungen

- Jede EnergieAblesung gehört zu einem Zeitstempel (1 Zeit → n Ablesungen)
- Marktinformationen abhängig von Zeit und Standort (1 Zeit + 1 Lokation → 1 MarketInfo)
- Wetterdaten abhängig von Zeit und Lokation (1:N)

Beispielhafte Randbedingungen

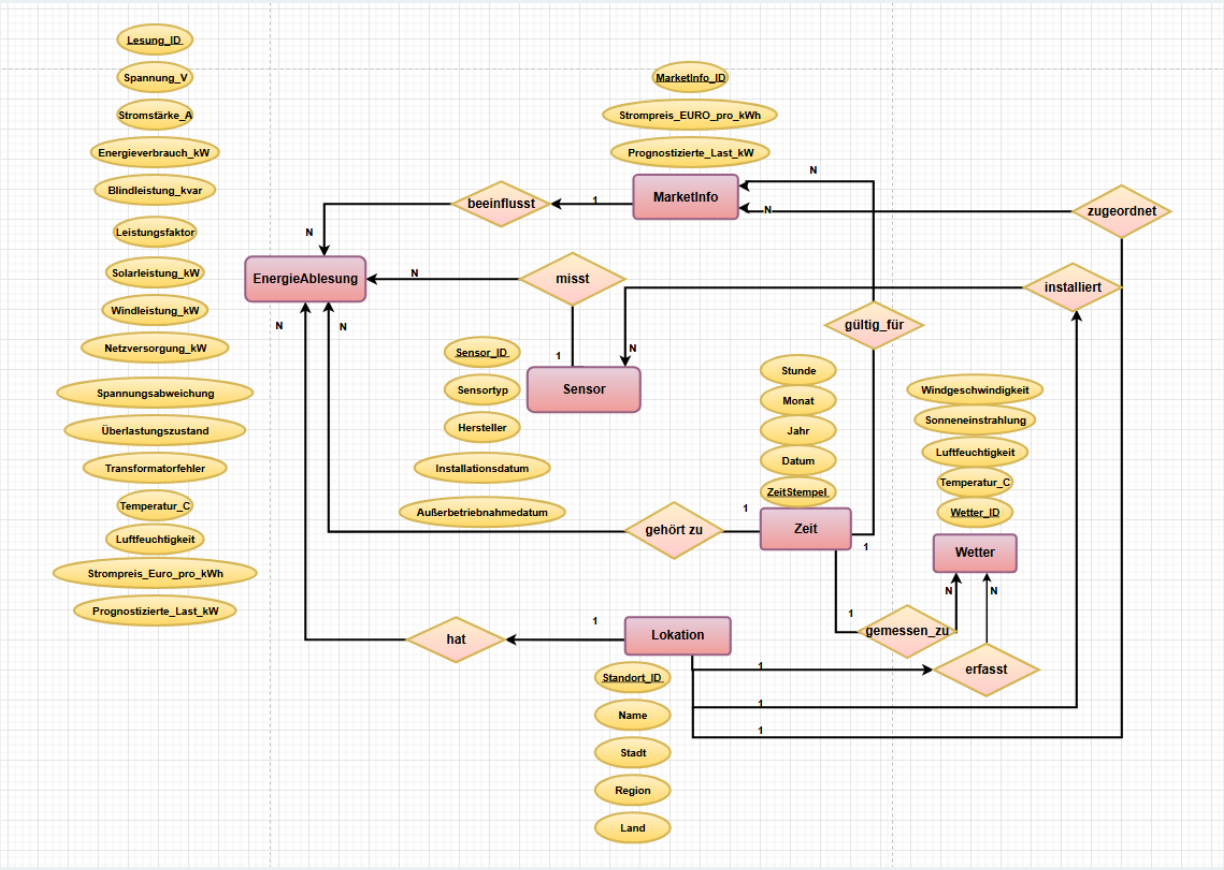
- Keine Wettermessung ohne zugehörige Lokation und Zeit
- Jeder Sensor → nur ein Hersteller
- Jede EnergieAblesung eindeutig durch Sensor + Zeitstempel

Business DB – Design (ER)

ER-Diagram:

Entitäten

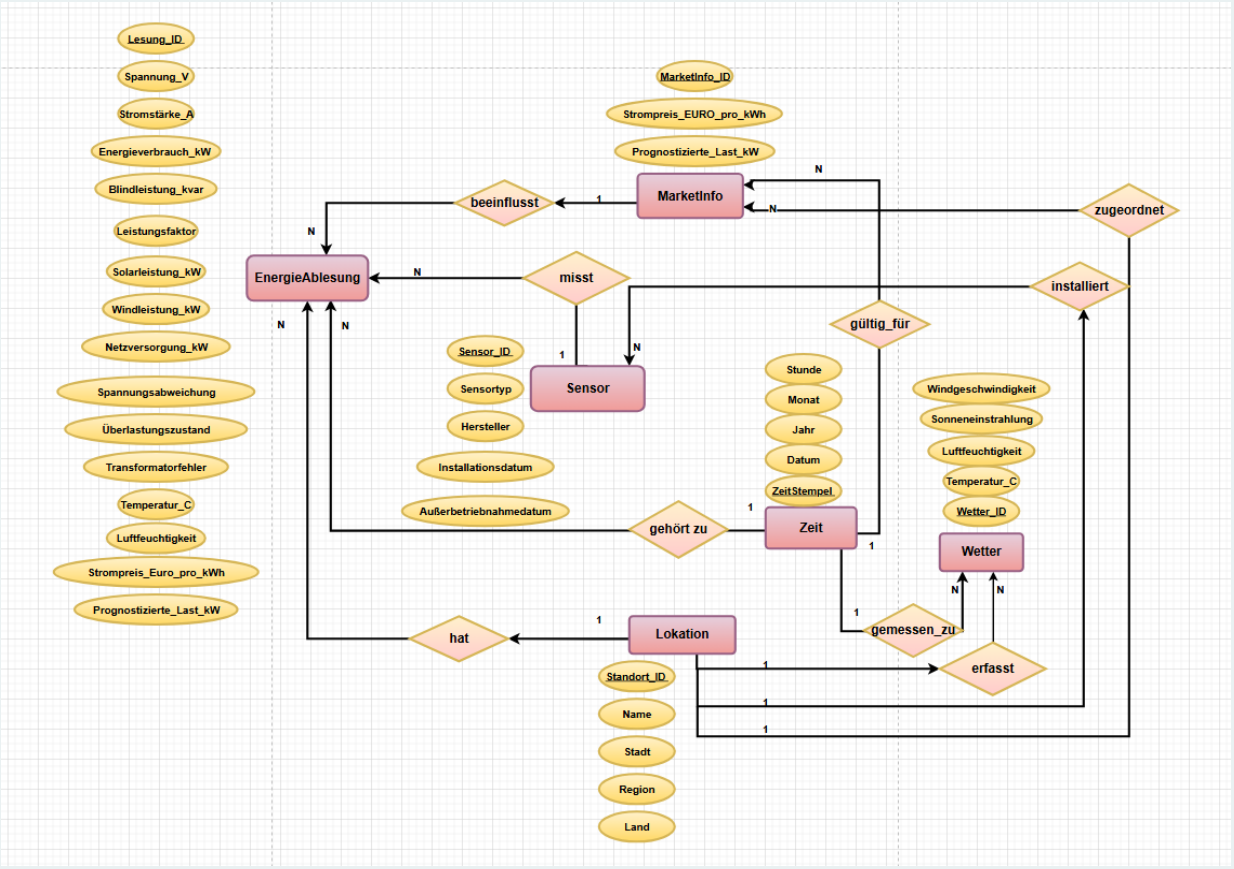
- EnergieAblesung:
- Sensor:.
- Lokation:
- Zeit:
- Wetter
- MarketInfo:



Business DB – Design (ER)

Beziehungen

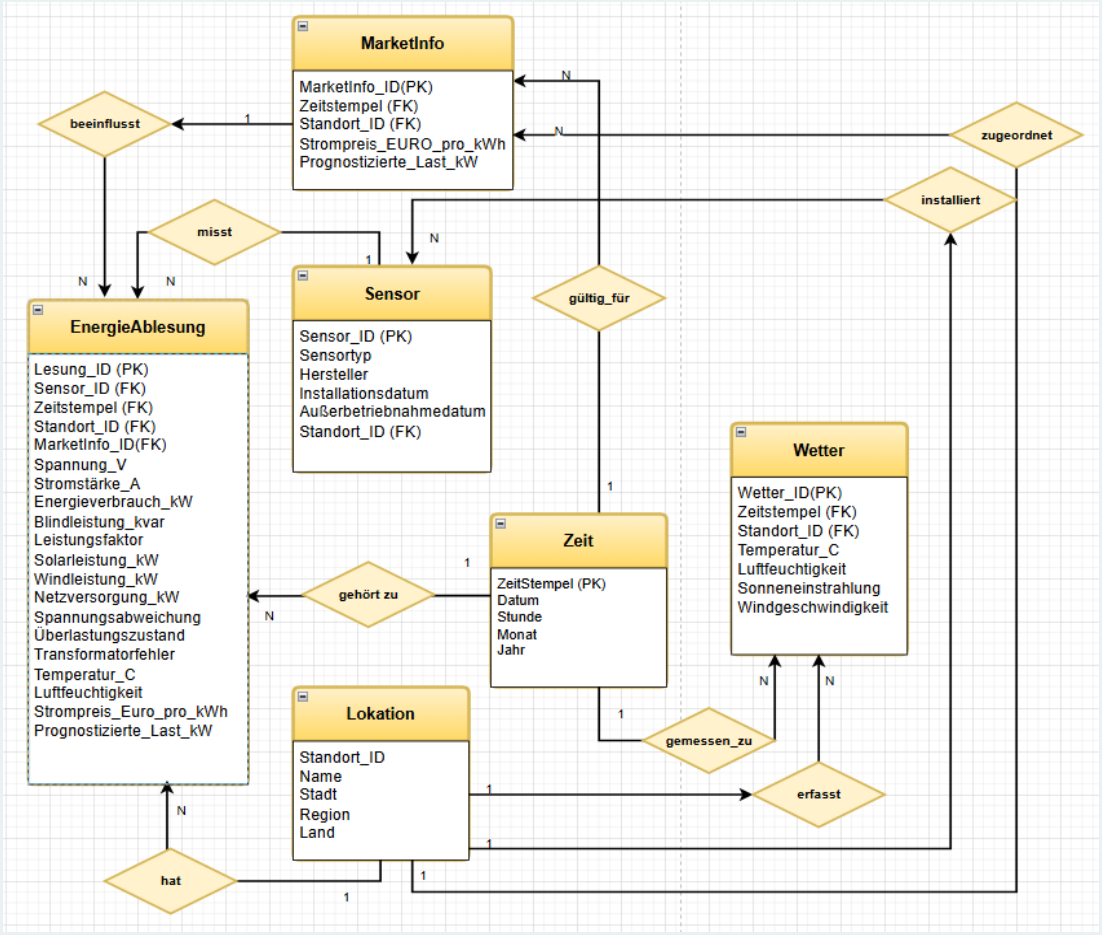
- Eine Lokation kann mehrere Sensoren besitzen, aber jeder Sensor gehört nur zu einer Lokation (1:N).
- Eine Lokation kann mehrere Wetterdatensätze haben, aber jeder Wetterdatensatz gilt nur für eine Lokation (1:N).
- Ein Sensor kann viele EnergieAblesungen erzeugen, aber jede Ablesung wird nur von einem Sensor gemessen (1:N).
- Eine Zeit kann für viele Datensätze (Ablesungen, Wetter, Marktinformationen) gelten (1:N).
- Eine MarketInfo bezieht sich auf eine Kombination von Zeit und Lokation (1:1 pro Zeit + Standort).



Business DB – Design (ER)

Objektbeschreibung

- EnergieAblesung ist das zentrale Objekt ,das alle Messwerte enthält.
- Sensor, Wetter, MarketInfo, Zeit und Lokation fungieren als Entitäten.
- Die Beziehungen werden über Primär- und Fremdschlüssel realisiert (1:N zwischen Dimension und Fakt).



Business DB – Design (ER)

Relationen aus Entitäten

Lokation : { [Standort_ID : integer, Name : string, Stadt : string, Region : string, Land : string] }

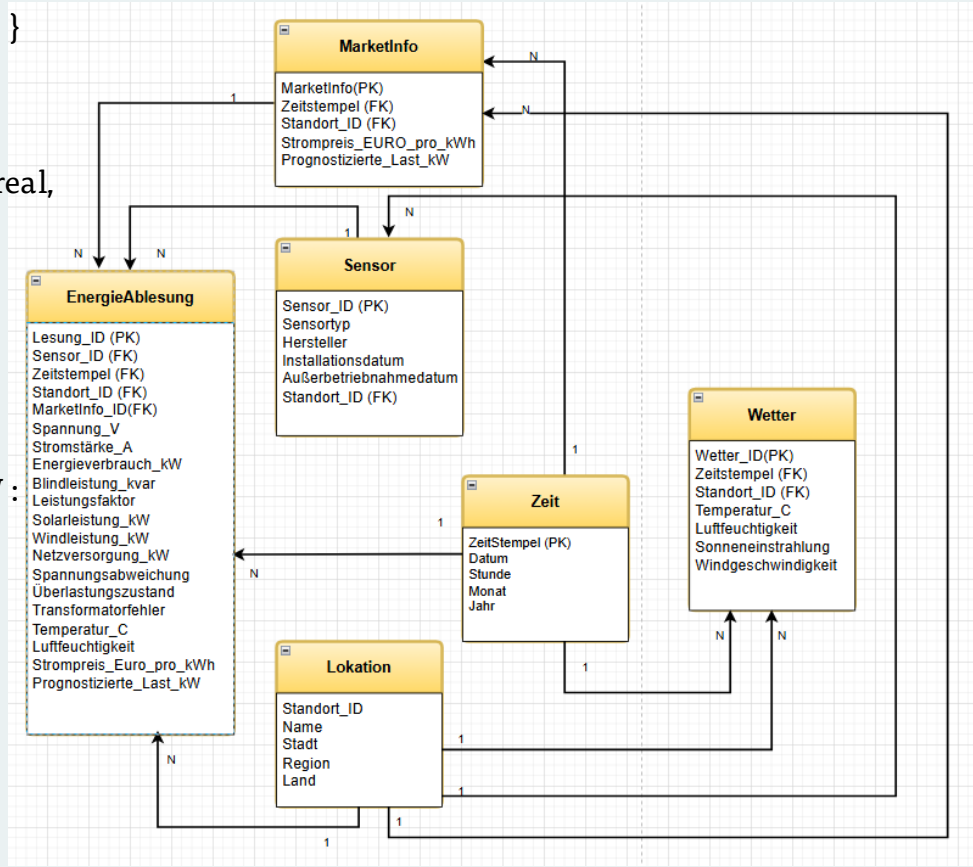
Zeit : { [Zeitstempel : timestamp, Datum : date, Stunde : integer, Monat : integer, Jahr : integer] }

Sensor : { [Sensor_ID : integer, Sensortyp : string, Hersteller : string, Installationsdatum : date, Außerbetriebnahmedatum : date, Standort_ID : integer] }

Wetter : { [Wetter_ID : integer, Zeitstempel : timestamp, Standort_ID :integer,Temperatur_C : real, Luftfeuchtigkeit : real,Sonneneinstrahlung : real, Windgeschwindigkeit : real] }

MarktInformation : { [MarketInfo_ID : integer, Zeitstempel : timestamp, Standort_ID : integer,Strompreis_Euro_pro_kWh : real, Prognostizierte_Last_kW : real] }

EnergieAblesung : { [Lesung_ID : integer, Sensor_ID : integer, Zeitstempel : timestamp, Standort_ID : integer,MarketInfo_ID : integer, Spannung_V : real, Stromstärke_A : real, Energieverbrauch_kW : real, Blindleistung_kvar : real,Leistungsfaktor : real, Solarleistung_kW : real, Windleistung_kW :real,Netzversorgung_kW : real, Spannungsabweichung : real,Überlastungszustand : boolean, Transformatorfehler : boolean,Temperatur_C : real, Luftfeuchtigkeit : real , Strompreis_Euro_pro_kWh : real,Prognostizierte_Last_kW: : real] }



Business DB – Design (ER)

Relationen aus Beziehungstypen:

installiert : { [Standort_ID : integer, Sensor_ID : integer] }

misst : { [Sensor_ID : integer, Lesung_ID : integer] }

gehört_zu : { [Zeitstempel : timestamp, Lesung_ID : integer] }

hat : { [Standort_ID : integer, Lesung_ID : integer] }

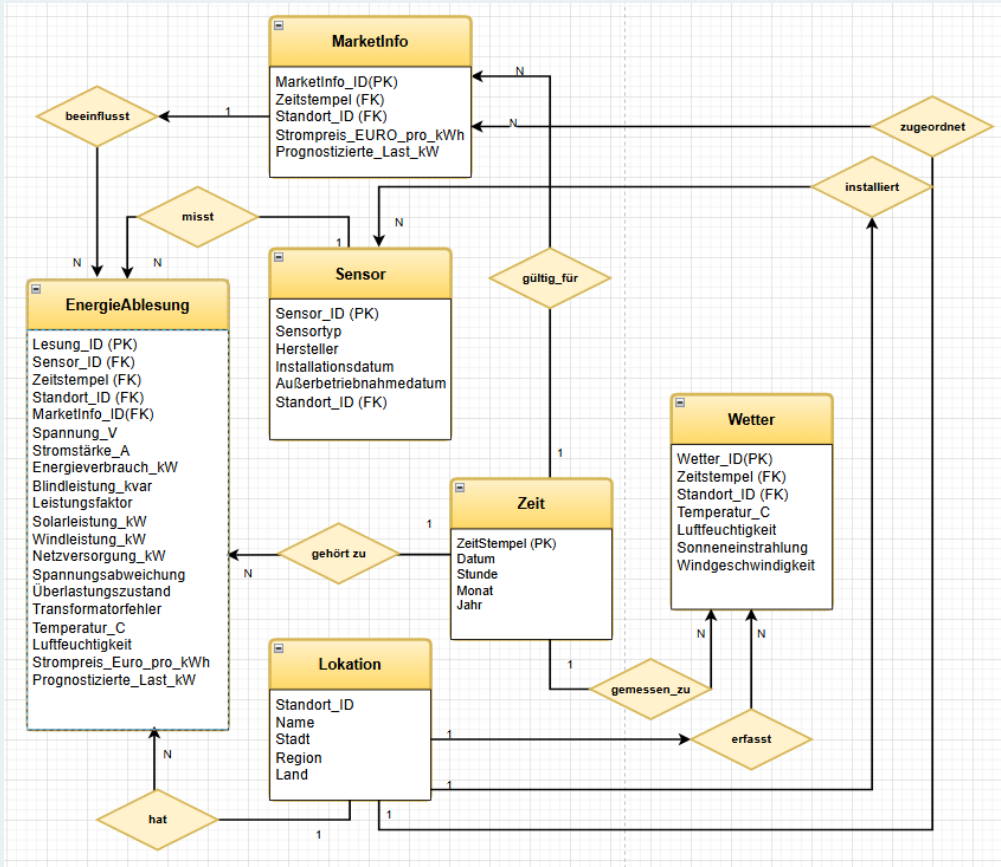
beeinflusst : { [MarketInfo_ID : integer, Lesung_ID : integer] }

erfasst : { [Standort_ID : integer, Wetter_ID : integer] }

gemessen_zu : { [Zeitstempel : timestamp, Wetter_ID : integer] }

gültig_für : { [Zeitstempel : timestamp, MarketInfo_ID : integer] }

zugeordnet : { [Standort_ID : integer, MarketInfo_ID : integer] }



Business DB – Relationenmodell

- Begründung der Zusammenfassung
  - Alle Beziehungen im Modell waren vom Typ 1:N und besaßen keine eigenen Attribute.
  - Daher konnten sie in die Tabellen der N-Seite integriert werden (z. B. Sensor enthält Standort_ID).
  - Es gab keine N:M- und keine 1:1-Beziehungen, somit war keine separate Beziehungstabelle erforderlich.
  - Nur wenn eine Beziehung eigene Attribute hat, wird sie als eigene Relation geführt.
  - Da dies hier nicht der Fall war, konnten die Beziehungstabellen gelöscht und mit den Entitätstabellen zusammengeführt werde

Business DB –Normalisierung (1NF – 3NF)

Ziel der Normalisierung
- Vermeidung von Redundanzen und Sicherstellung der Datenintegrität
- Jede Tabelle enthält nur atomare Werte und eindeutige Primärschlüssel

1. Normalform (1NF)
- Alle Attribute sind atomar (nur ein einzelner Wert)
- Keine wiederholenden Spalten (z. B. keine Temperatur_1, Temperatur_2 ...)
- Jede Tabelle besitzt einen klar definierten Primärschlüssel

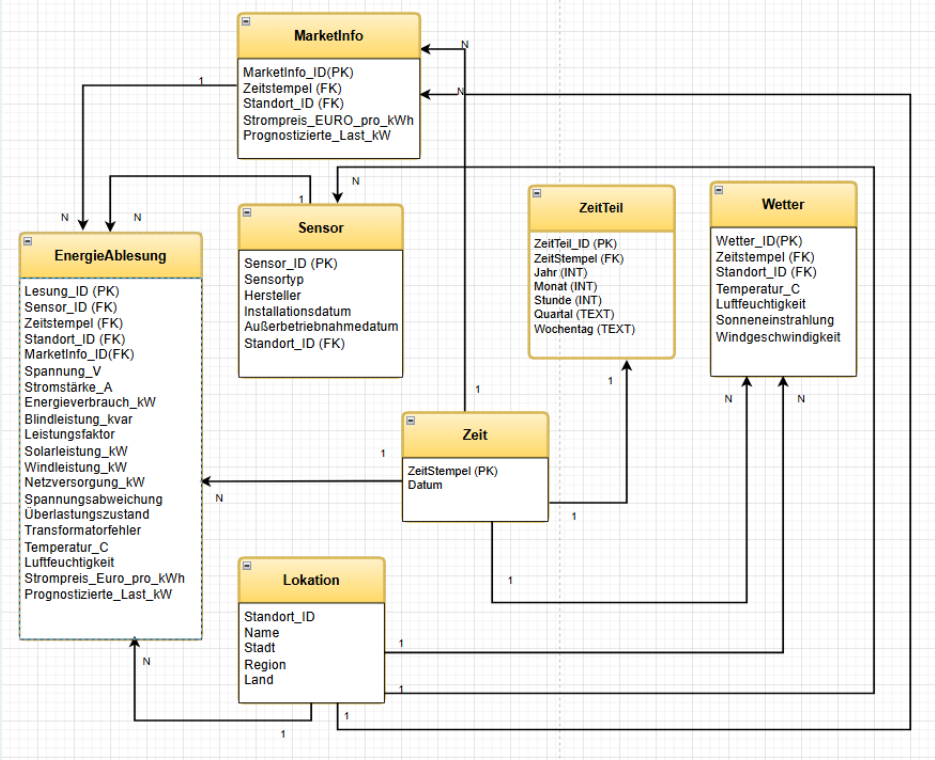
→ Das gesamte Modell erfüllt bereits die 1NF

2. Normalform (2NF)
- Alle Tabellen sind bereits in 1NF
- Keine partiellen funktionalen Abhängigkeiten, da keine zusammengesetzten Primärschlüssel existieren

→ Alle Nicht-Schlüsselattribute hängen vollständig vom Primärschlüssel ab
→ Modell erfüllt die 2NF

3. Normalform (3NF)
- Keine transitiven Abhängigkeiten zwischen Nicht-Schlüsselattributen
- In der Tabelle Zeit bestand: Datum → (Jahr, Monat, Stunde)
- Lösung: Aufteilung in Zeit und ZeitTeil (1:1-Beziehung)

→ Alle Attribute hängen direkt vom Primary Key ab
→ Modell vollständig 3NF-konform



Business DB –Speicherberechnung

Fachliche Bewertung des Speicherverbrauchs

• Der Speicherbedarf der einzelnen Tabellen ergibt sich aus der Summe der Spaltengrößen pro Datensatz sowie der Gesamtzahl der erfassten Zeilen.

• Die Faktentabelle EnergieAblesung weist aufgrund ihrer hohen Anzahl an Messattributen und der großen Datenmenge den mit Abstand größten Speicherbedarf auf (≈ 122 MB).

• Dimensionstabellen wie Wetter, Lokation, Zeit und Sensor besitzen deutlich geringere Attributanzahlen und sind daher speichereffizient strukturiert.

• Die Tabelle Wetter benötigt lediglich 32 Byte pro Zeile, wodurch ihr Gesamtspeicherbedarf im niedrigen Kilobyte-Bereich bleibt.

• Die aggregierte Berechnung ergibt einen geschätzten Gesamtspeicherbedarf von ca. 122,6 MB für das gesamte Datenbankmodell – eine effiziente Größe für analytische Workloads im DWH-Kontext.

Tabelle 4: Wetter – Spaltengrößen

Spalte	Datentyp	Größe (Byte)	Beschreibung
Wetter_ID (PK)	INTEGER	4	Primärschlüssel
Zeitstempel (FK)	TIMESTAMP	8	Zeitreferenz
Standort_ID (FK)	INTEGER	4	Verweis auf Lokation
Temperatur_C	REAL	4	Temperatur
Luftfeuchtigkeit	REAL	4	Feuchtigkeit
Sonneneinstrahlung	REAL	4	Sonneneinstrahlung
Windgeschwindigkeit	REAL	4	Windgeschwindigkeit
Summe pro Zeile		**32**	

Annahme: 8 760 Datensätze
Gesamtspeicher: 32 × 8 760 = ≈ 280 320 B (≈ 274 KB)

Tabelle 7: Gesamtspeicher (Schätzung)

Tabelle	Speicherbedarf
Lokation	≈ 20 KB
Zeit	≈ 197 KB
Sensor	≈ 92 KB
Wetter	≈ 274 KB
MarktInformation	≈ 23 KB
EnergieAblesung	≈ 122 MB
Gesamt	**≈ 122,6 MB**

DWH Fragen

Zentrale analytische Fragestellungen

- Energieverbrauch pro Standort und Tag

Welche Standorte verbrauchen täglich am meisten Energie?

- Peak Load Analyse (höchste Stunde)

Wann treten Lastspitzen auf und wo entstehen Engpässe?

- Einfluss des Wetters auf den Verbrauch
- Wie stark beeinflussen Temperatur, Wind oder Sonne den Energieverbrauch?
- Genauigkeit der Marktprognosen

Wie gut stimmen prognostizierte Lasten mit den realen Messungen überein?

- Erneuerbare Energie – Anteil Solar/Wind
- Wie hoch ist der Anteil erneuerbarer Energie am Gesamtverbrauch?

DWH Design- mER-Diagramm

Was zeigt das mER-Diagramm?

• Das mER-Modell bildet die multidimensionale Struktur des Data Warehouses ab.

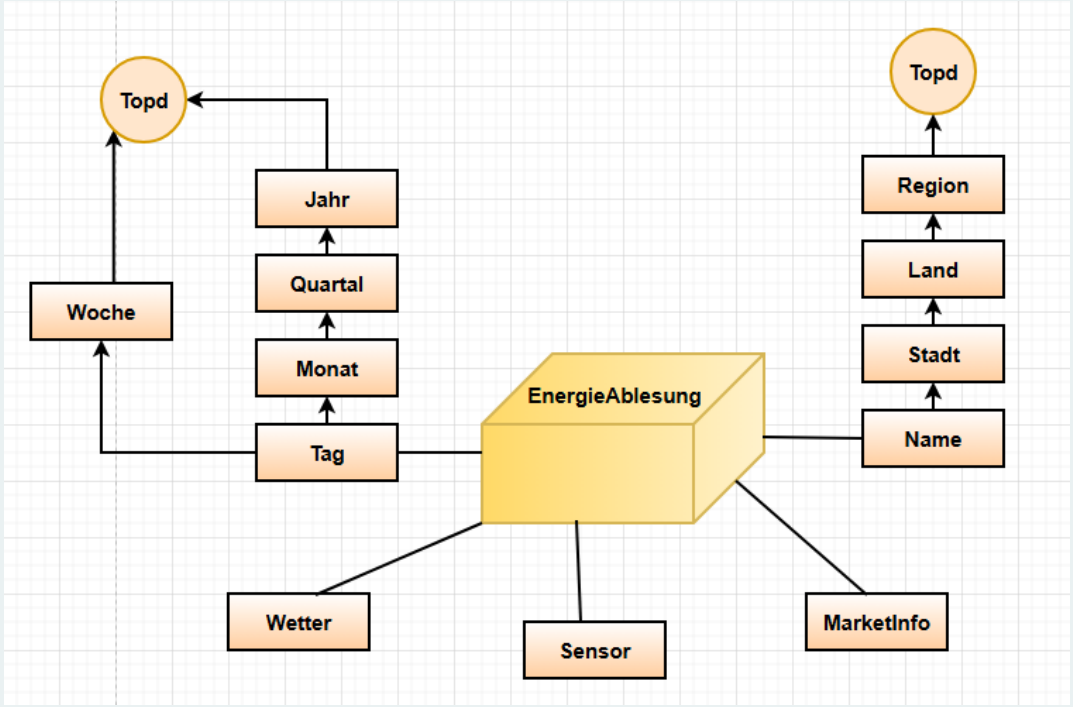
- Die Faktentabelle (EnergieAblesung) steht im Zentrum der Analyse.
- Alle Dimensionen (Zeit, Standort, Sensor, Wetter, MarketInfo) sind analytisch angebunden.

Hierarchien der Dimensionen

- Zeitdimension: Tag → Monat → Quartal → Jahr → Woche → TopD
- Standortdimension: Name → Stadt → Land → Region → TopD
- Die Hierarchien erlauben Drill-Down und Roll-Up Analysen.

Zweck des mER-Modells

- Zerlegt die Datenbank in Fakt + Dimensionen für OLAP-Analysen.
- Beschreibt, wie Daten entlang verschiedener Achsen (Zeit, Standort, Wetter, Markt) aggregiert werden können.
- Grundlage für das Star Schema und spätere Visualisierungen (z. B. KPI-Dashboards).



Data Warehouse Structure: Star Schema

Warum Star Schema? (OLAP-Struktur)

- Das Star Schema ist eine typische OLAP-Struktur (Online Analytical Processing).
- OLAP = Analytische, aggregierte, mehrdimensionale Auswertung großer Datenmengen.
- Im Gegensatz zu OLTP (transaktionsorientiert) ist OLAP für Analyse, nicht für tägliche operativen Prozesse.
- Die zentrale Faktentabelle + Dimensionen ermöglichen schnelle Abfragen, Summen, Trends und Zeitreihenanalysen.

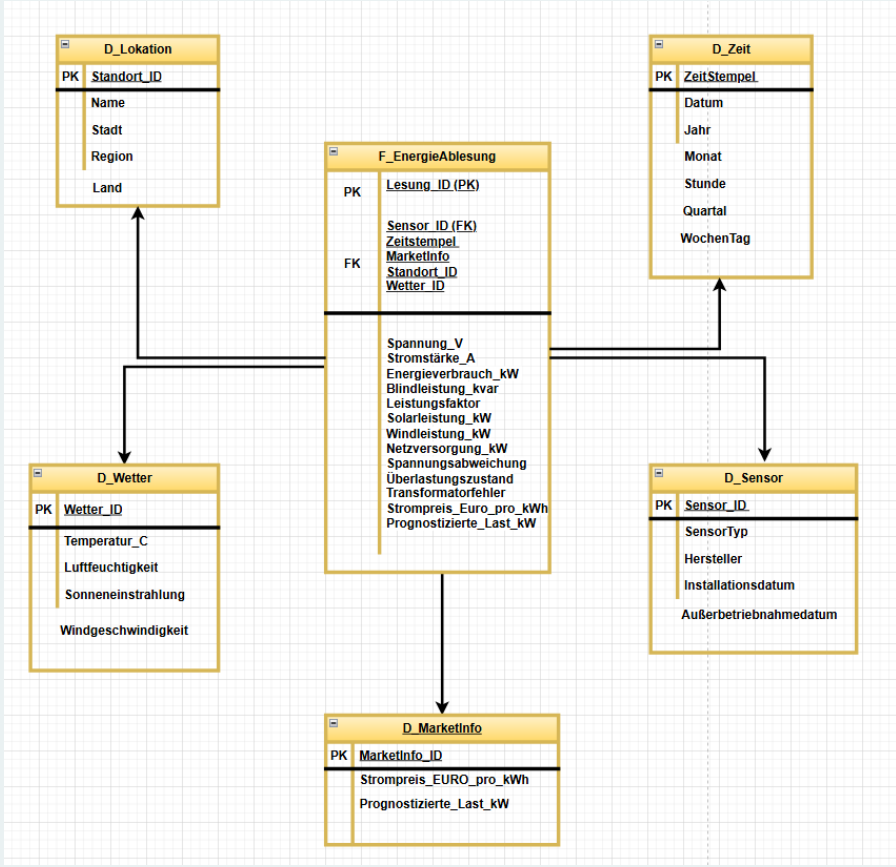
Data Warehouse Structure: Star Schema

Faktentabelle (Fact Table)

- Die Faktentabelle enthält alle gemessenen Werte (Spannung, Strom, Verbrauch, Wind, Sonne, Temperatur usw.).
- Diese Messwerte sind numerisch, aggregierbar und analyseorientiert → ideal für OLAP.
- Jeder Fakt verweist über Fremdschlüssel auf die zugehörigen Dimensionen (Zeit, Standort, Sensor, Wetter, MarketInfo).

Dimensionstabellen

- Dimensionen liefern Kontext für die Messwerte:
- Zeit (Jahr, Monat, Stunde ...)
- Lokation (Name, Stadt, Region ...)
- Sensor (Typ, Hersteller ...)
- Wetter (Temperatur, Wind ...)
- MarketInfo (Preis, Last)
- Dimensionen ermöglichen Drill-Down / Roll-Up und flexible Analysen.



Welche SCD-Varianten haben Sie gwählt

Verwendete SCD-Typen im Projekt

- Im Smart-City-IoT-Projekt wurde SCD Typ 2 angewendet.
- SCD Typ 2 erzeugt für jede Änderung einen neuen Datensatz mit Start- und Enddatum.
- Historische und aktuelle Zustände bleiben vollständig nachvollziehbar.

Warum SCD Typ 2?

- Vollständige Historisierung aller relevanten Dimensionen
- Erforderlich für Trendanalysen, Zeitreihen, Prognosen und KI-Modelle
- Exakte Nachverfolgung von Änderungen in:
- D_Sensor (Typ, Hersteller, Installationsdatum)
- D_Lokation (Namensänderungen, organisatorische Änderungen)
- D_MarketInfo (Preis und Last)
- D_Wetter (Kalibrierungen, neue Messwerte)

Tabelle vor der Änderung:

Tabelle 2: Sensor – Zustand vor der Änderung

Sensor_ID	Standort_ID	SensorTyp	Hersteller	Installdatum	StartDate	EndDate	IsCurrent
S-9F42EE91	AST-001	IndustrialMeter	ABB	2024-01-01	2024-01-01	NULL	TRUE

Änderung: Am 30.04.2024 wurde der Sensortyp auf *SmartMeter* aktualisiert.

Tabelle nach Anwendung von SCD Typ 2:

Tabelle 3: Sensor – nach Anwendung von SCD Typ 2

Sensor_ID	Standort_ID	SensorTyp	Hersteller	Installdatum	StartDate	EndDate	IsCurrent
S-9F42EE91	AST-001	IndustrialMeter	ABB	2024-01-01	2024-01-01	2024-04-30	FALSE
S-9F42EE91	AST-001	SmartMeter	ABB	2024-04-30	2024-04-30	NULL	TRUE

Ergebnis: Der alte Datensatz wird geschlossen, ein neuer dokumentiert die Geräteänderung.

Technischer Hinweis

- SCD Typ 2 führt zu mehr Datensätzen und größeren Dimensionstabellen.
- Daher werden umfangreiche Änderungsdaten in separaten Tabellen/CSV-Dateien ausgelagert,

um das Haupt-DWH performant und übersichtlich zu halten.

ETL-Prozess: Gesamtüberblick

ETL-Prozess im Smart City IoT Projekt

- Die Rohdaten lagen zunächst als große CSV-Datei mit umfangreichen IoT-Messwerten (Sensor- und Energiedaten) vor.
- Diese CSV-Daten wurden vollständig in eine stagingartige MySQL-Tabelle importiert, um eine zentrale Ausgangsbasis zu schaffen.
- Auf Basis dieser importierten Tabelle wurden anschließend alle Dimensionstabellen des Star-Schemas abgeleitet und befüllt.
- Ziel des ETL-Prozesses war die Erstellung einer sauberen, strukturierten und analytisch nutzbaren Datenbasis.

ETL-Prozess –Transformation & Bereinigung

Bereinigung & Erstellung der Dimensionstabellen

- Aus der großen Import-Tabelle wurden alle relevanten Attribute extrahiert, um die Dimensionen Sensor, Wetter, Lokation, MarketInfo und Zeit aufzubauen.

- Für jede Dimension wurden Duplikate entfernt (z. B. gleiche Sensornummern, identische Wetterwerte, doppelte Standorteinträge).

- Nach der Bereinigung erhielt jede Dimension einen eindeutigen Primärschlüssel, der später als Fremdschlüssel im DWH genutzt wurde.

- Anschließend wurden die neuen IDs wieder in die große Ursprungstabelle zurückgeführt, um ein konsistentes Datenmodell aufzubauen.

ETL-Prozess –Laden der Faktentabelle & Finalisierung des Modells

Aufbau der Faktentabelle & Finalisierung

• Nachdem die Bereinigung sämtlicher Dimensionstabellen abgeschlossen war, wurde die bereinigte Haupttabelle zur Erstellung der Faktentabelle EnergieAblesung verwendet.

• Alle Messwerte (z. B. Spannung, Strom, Energieverbrauch, Wetterparameter, Marktparameter) wurden direkt aus der bereinigten Haupttabelle übernommen.

• Die zuvor separate Zeit-Tabelle wurde entfernt, da der Zeitstempel selbst als eindeutige Zeitreferenz diente und in der Faktentabelle gespeichert werden konnte.

• Damit entstand ein vollständig bereinigtes, konsistentes und analysierbares Star-Schema, das für OLAP-Berichte und IoT-Analysen geeignet ist.

SCD Typ 2 – Umsetzung in MySQL (Stored Procedure)

Zweck der Prozedur:

- Automatisches Erkennen von Änderungen in der *Location*-Dimension
- Historisierung des alten Datensatzes (Ende setzen)
- Anlage eines neuen, aktuellen Datensatzes (*Surrogate Key*, Startzeit)

Vorgehensweise (Kurzüberblick):

- Prüfung, ob der aktuelle Eintrag bereits existiert
- Bei Änderung:
  - Alter Eintrag wird geschlossen (*is_current = 0*, *end_date = NOW()*)
  - Neuer Eintrag wird mit den aktualisierten Werten angelegt
- Einträge werden somit vollständig versioniert

```sql
DROP PROCEDURE IF EXISTS scd_upsert_location;

DELIMITER $$

CREATE PROCEDURE scd_upsert_location(
    IN p_name   VARCHAR(200),
    IN p_stadt  VARCHAR(200),
    IN p_region VARCHAR(200),
    IN p_land   VARCHAR(200)
)
BEGIN
    DECLARE v_name   VARCHAR(200);
    DECLARE v_stadt  VARCHAR(200);
    DECLARE v_region VARCHAR(200);
    DECLARE v_land   VARCHAR(200);

    SELECT name, stadt, region, land
    INTO v_name, v_stadt, v_region, v_land
    FROM locations
    WHERE name = p_name
      AND is_current = 1
    ORDER BY start_date DESC
    LIMIT 1;

    IF v_name IS NULL THEN
        INSERT INTO locations (name, stadt, region, land, start_date, end_date, is_current)
        VALUES (p_name, p_stadt, p_region, p_land, NOW(), NULL, 1);

    ELSEIF v_stadt <> p_stadt
        OR   v_region <> p_region
        OR   v_land   <> p_land THEN

        UPDATE locations
        SET end_date  = NOW(),
            is_current = 0
        WHERE name = p_name
          AND is_current = 1;

        INSERT INTO locations (name, stadt, region, land, start_date, end_date, is_current)
        VALUES (p_name, p_stadt, p_region, p_land, NOW(), NULL, 1);

    END IF;
END$$

DELIMITER ;

-- ---------------- test ----------------
CALL scd_upsert_location('Shanghai_LoadStation', 'Beijing', 'Beijing', 'China');

SELECT id, name, stadt, region, land, start_date, end_date, is_current
FROM locations
WHERE name = 'Shanghai_LoadStation'
ORDER BY start_date;
```

id	name	stadt	region	land	start_date ▲ 1	end_date	is_current
13	Shanghai_LoadStation	Shanghai	Asia	LoadHub	2025-11-13 12:45:46	2025-11-13 14:41:45	0
18	Shanghai_LoadStation	Tehran	Tehran	Iran	2025-11-13 14:41:45	2025-11-13 14:44:23	0
19	Shanghai_LoadStation	Beijing	Beijing	China	2025-11-13 14:44:23	*NULL*	1

Was haben wir gelernt

Wir haben gelernt:

- große CSV-Datenmengen strukturiert aufzubereiten und sie Schritt für Schritt in ein professionelles Data-Warehouse-Modell zu überführen.
- wie wichtig Datenqualität, saubere Dimensionen und eindeutige Schlüssel für ein zuverlässiges Analysemodell sind.
- komplexe Aufgaben in klare ETL-Phasen zu zerlegen und systematisch umzusetzen: Extraktion, Bereinigung, Transformation, Laden.

Ich nehme mit,wie konzeptionelle Modelle (ER), logische Modelle (3NF) und analytische Modelle (Star Schema) zusammenhängen und sich gegenseitig ergänzen.

Highlights

Erfolgreiche Umsetzung eines vollständigen ETL-Prozesses:

von großen CSV-Rohdaten bis zum analysierbaren Star-Schema

Mit sauberen Dimensionen und einer konsistenten Faktentabelle.

Aufbau eines professionellen Data-Warehouse-Modells inklusive Normalisierung, SCD-Konzept, Schlüsselgenerierung und datengetriebener Analysen im Smart-City-IoT-Kontext.

Eventuelle Probleme & Lösungen

Typische Probleme

- Unterschiedliche Werteformate und fehlende Konsistenz in den CSV-Daten
- Doppelte Einträge in Dimensionstabellen (Wetter, Sensor, Lokation, MarketInfo)
- Schwierigkeiten bei der Zuordnung der Fremdschlüssel aufgrund ähnlicher Messwerte
- Hohe Datenmenge → lange Ladezeiten und komplexe Bereinigung

Lösungen

- Standardisierung und Rundung aller Messwerte für eindeutige Zuordnung
- Einsatz von systematischen Bereinigungsprozessen (Duplikaterkennung & -entfernung)
- Erstellung eindeutiger IDs und saubere Füllung aller Fremdschlüssel
- Aufbau eines klaren ETL-Prozesses zur Strukturierung des Datenflusses

Was sollten meine Kollegen wissen

Klare und strukturierte Arbeitsweise hilft, komplexe Aufgaben leichter zu bewältigen.

Bei großen Projekten frühzeitig planen und Aufgaben in sinnvolle Schritte aufteilen.

Die Teamgröße sollte dem Umfang und der Komplexität des Projekts entsprechen.

Vielen Dank