



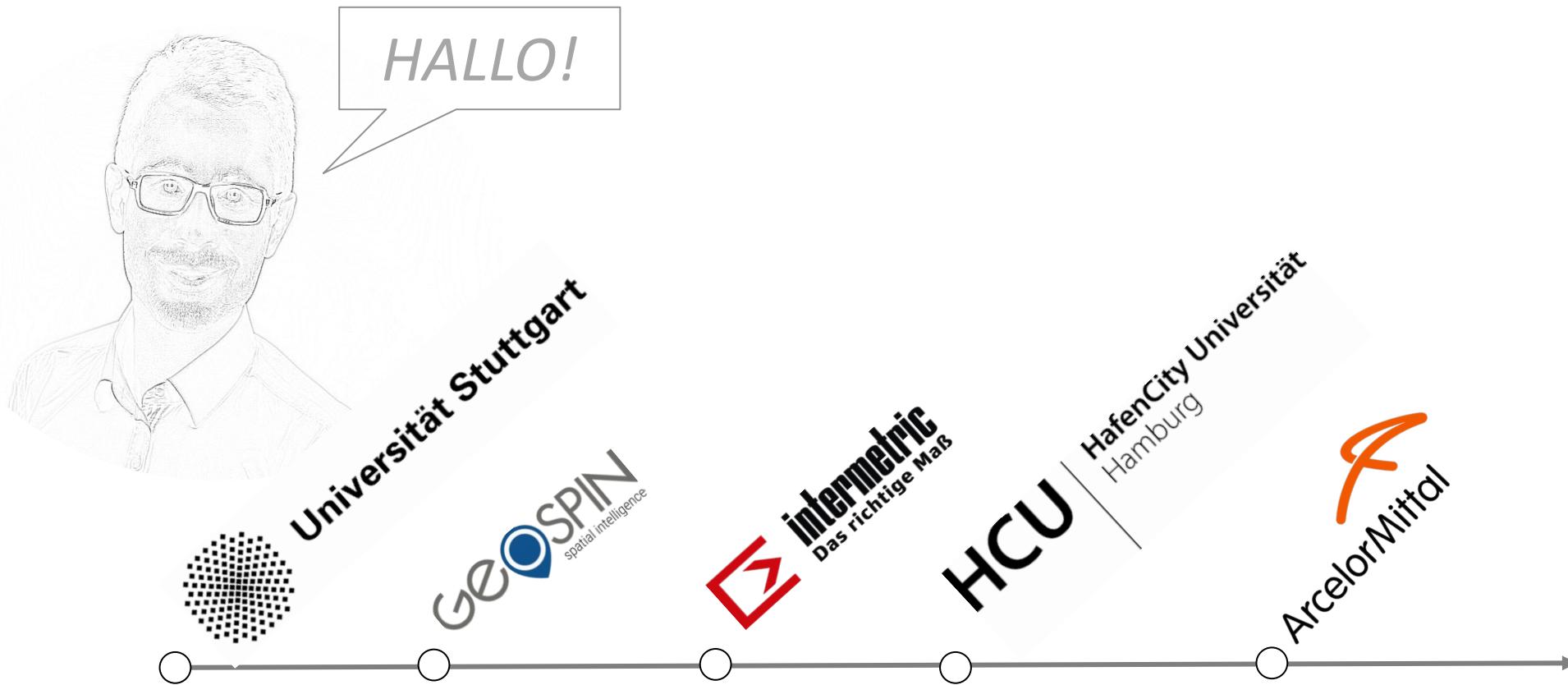
Hossein Shoushtari

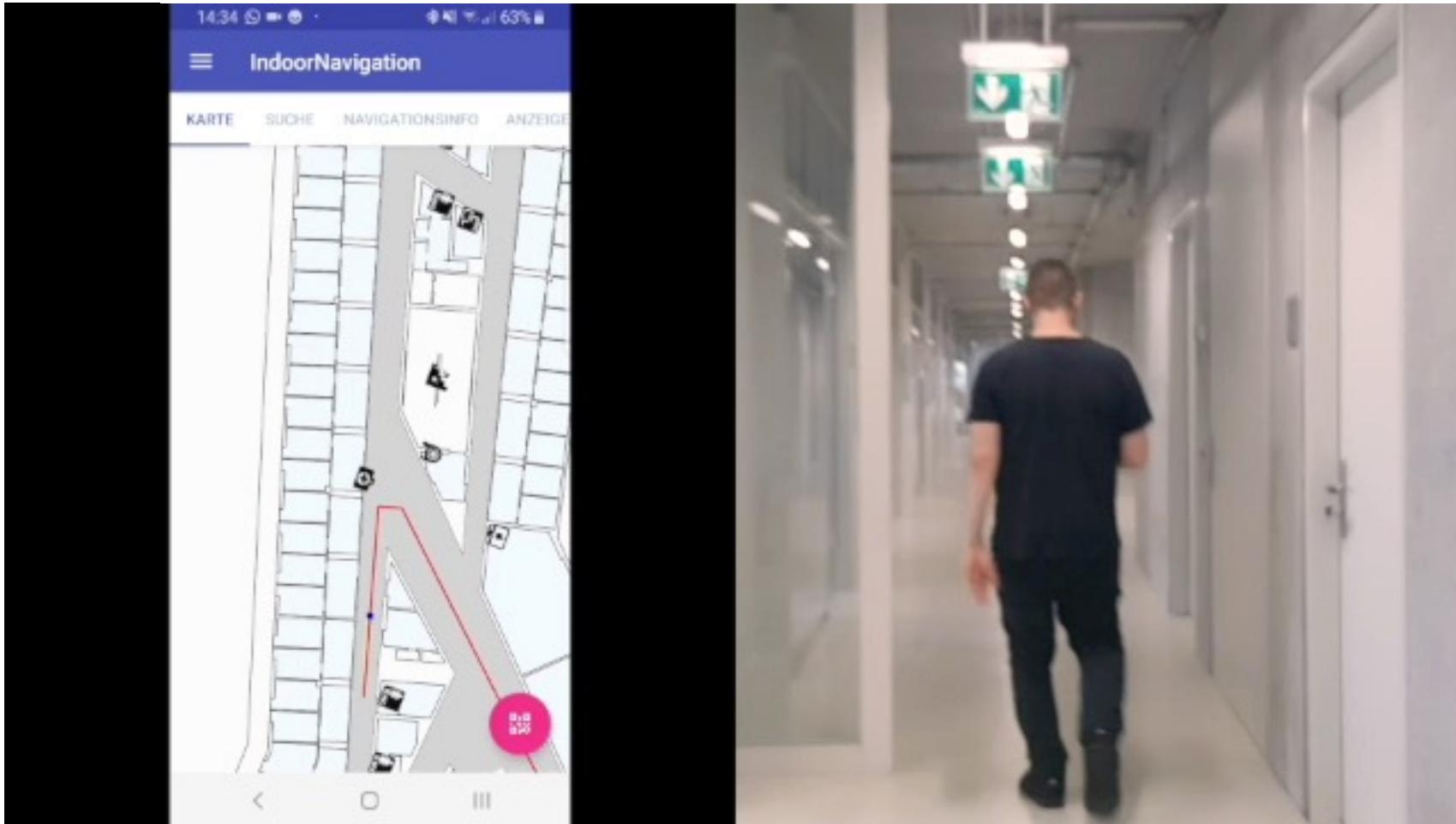
# Entwicklung einer Geodatenbank

zur Integration und Analyse von Indoor Positionierungsdaten

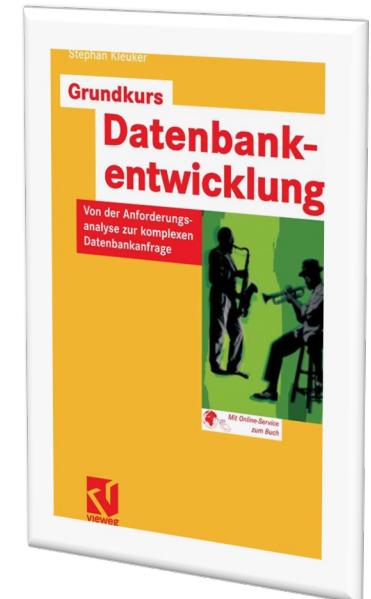
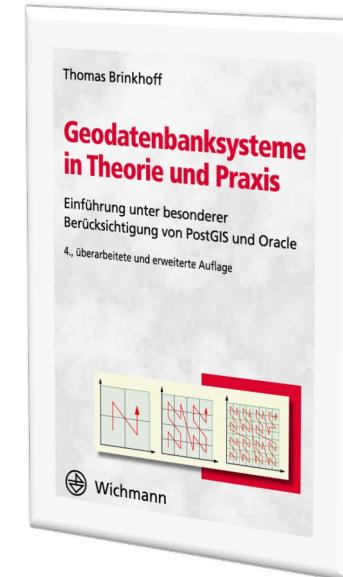


Sommersemester 2025 | Probelehrveranstaltung





Thomas Brinkhoff, Geodatenbanksysteme in Theorie und Praxis (2021), Wichmann Verlag, Berlin, ISBN: 9783879076956



Stephan Kleuker, Grundkurs Datenbankentwicklung (2006), Vieweg & Sohn Verlag, Wiesbaden, ISBN: 3-8348-0008-2

International Conference on  
Indoor Positioning and Indoor Navigation (IPIN)

Institute of Navigation (ION) International Technical Meeting (ITM) und GNSS+ > GNSS CHALLENGING ENVIRONMENTS

Vorlesungsskript:  
QR der Kurs-Websites:



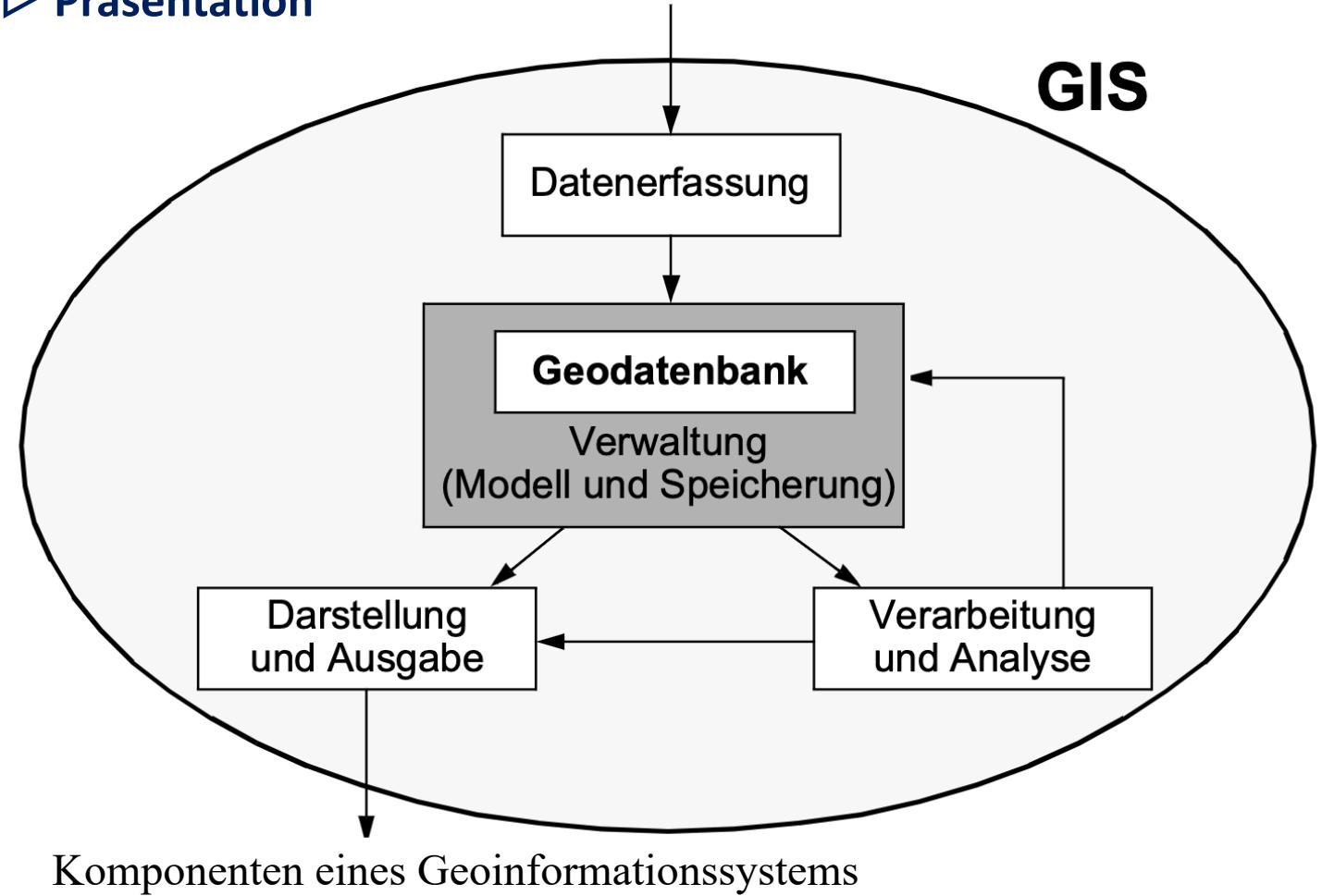
	Vorlesung	Übung	Beschreibung
Woche 1	Datenbanksysteme	Übung 1 Einweisung Daten laden & transformieren	Aktualisieren der Kurs-Websites Vorlesungsaufzeichnung
Woche 2	Know-Hows Video	Übung 1 Daten laden & transformieren	  Krankenhausanwendung
Woche 3	Konzeptionelle Modellierung mit UML	Übung 2 Einweisung Datenvorschau	Aktualisieren der Kurs-Websites Indoor Positionierung als Beispiele
Woche 4	Know-Hows Video	Übung 2 Datenvorschau	  Krankenhausanwendung
Woche 5	Von UML zum logischen Modell	Übung 3 Einweisung Datenvorschau	Aktualisieren der Kurs-Websites Indoor Positionierung als Beispiele
Woche 6	Know-Hows Video	Übung 3 Datenvorschau	Krankenhausanwendung
Woche 7	Modellierung von Geodaten in SQL	Übung 4 Einweisung Daten Speichern	  Aktualisieren der Kurs-Websites Indoor Positionierung als Beispiele
Woche 8	Know-Hows Video	Übung 4 Daten Speichern	Krankenhausanwendung
Woche 9	Abfrage topologischer Relationen	Übung 5 Einweisung Daten Speichern	Aktualisieren der Kurs-Websites BIM Model als Beispiele
Woche 10	Know-Hows Video	Übung 5 Daten Speichern	  Krankenhausanwendung
Woche 11	Know-Hows Video	Übung 6 Geometriesche Analyse	    Krankenhausanwendung

- 00 - Einleitung u. Wiederholung
- 01 - Konzeptionelle Modellierung mit UML
  - Anforderungsanalyse
  - Indoor Positionierung
  - Inertial Lokalization
  - Unified Modeling Language (UML)
- 02 - Von UML zum logischen Modell
  - Logisches Modell in Tabellenform
- 03 - Modellierung von Geodaten in SQL
  - Structed Query Language (SQL)
  - Anfragen auf Tabellen
- 04 - Ausblick und Fazit

## Erfassung ▷ Verwaltung ▷ Analyse und ▷ Präsentation

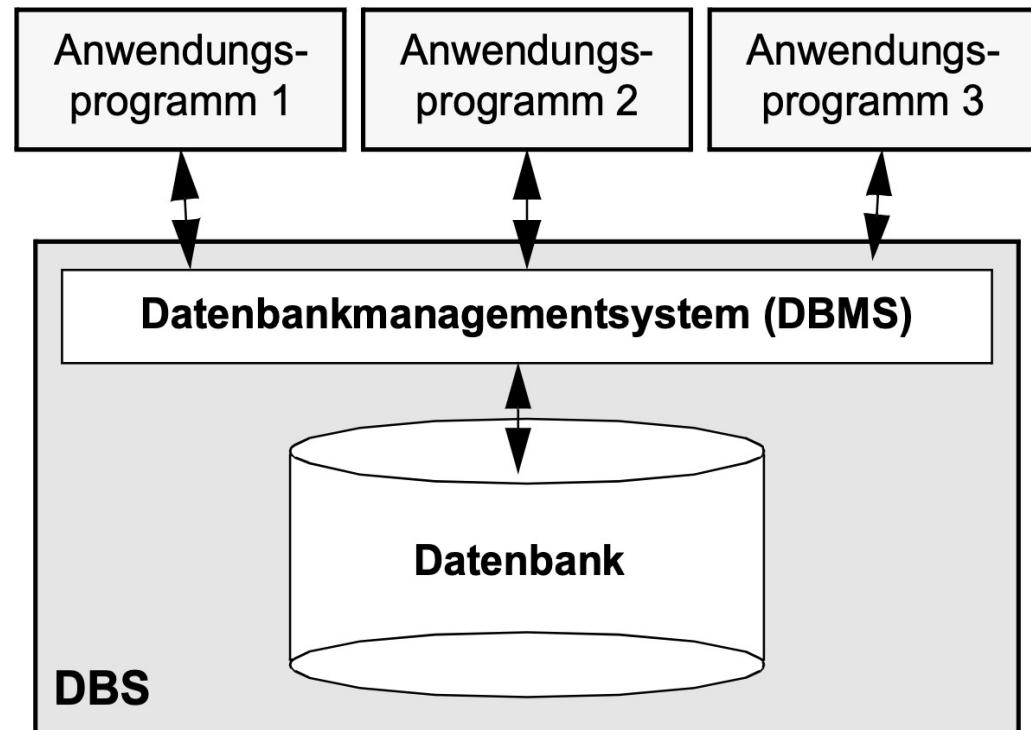
### Hauptgeometriertypen:

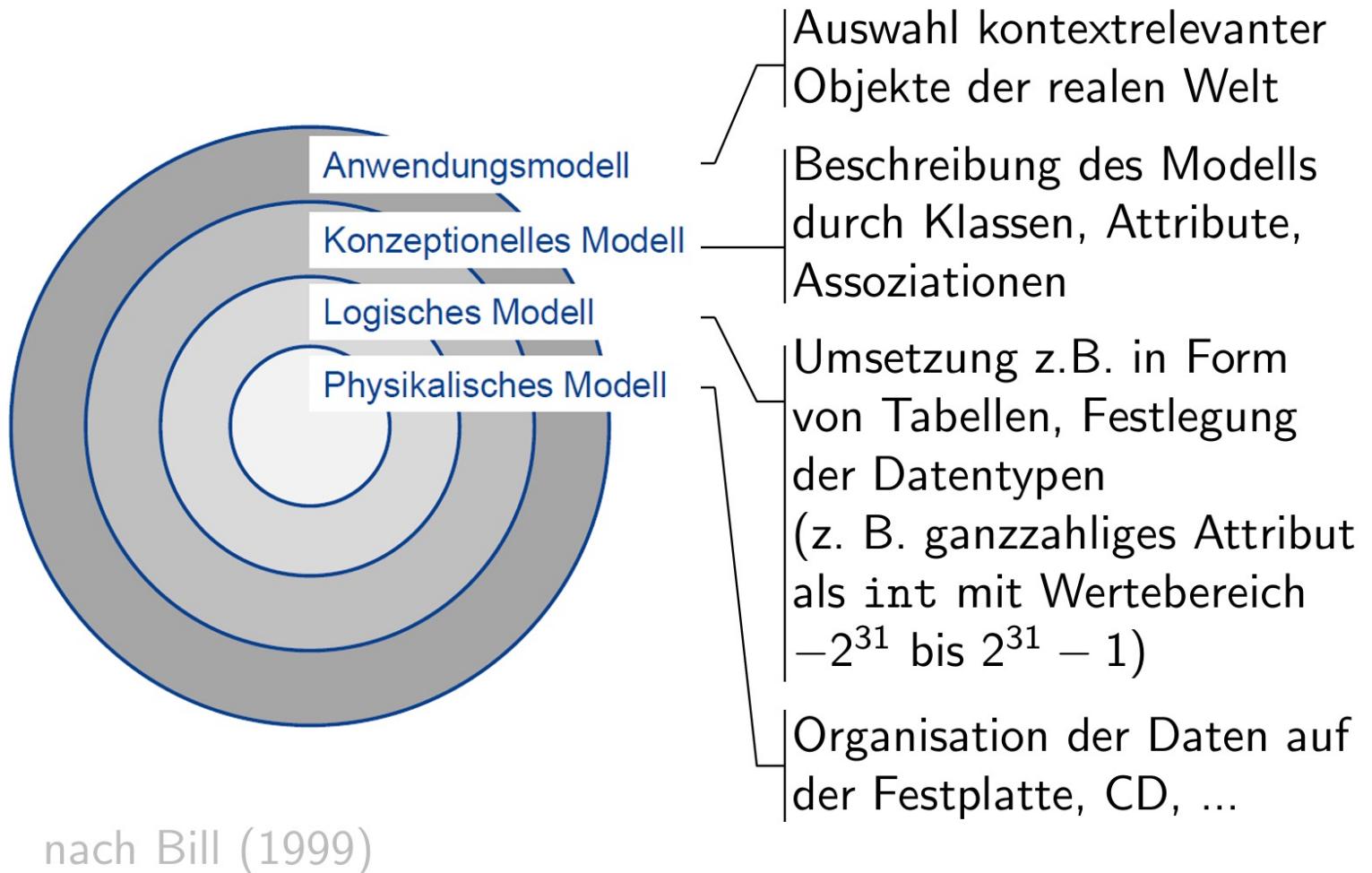
- Vektordaten
  - Punkt
  - Linie
  - Flächen (Polygon)
- Rasterdaten



## Was ist eigentlich eine Geodatenbank?

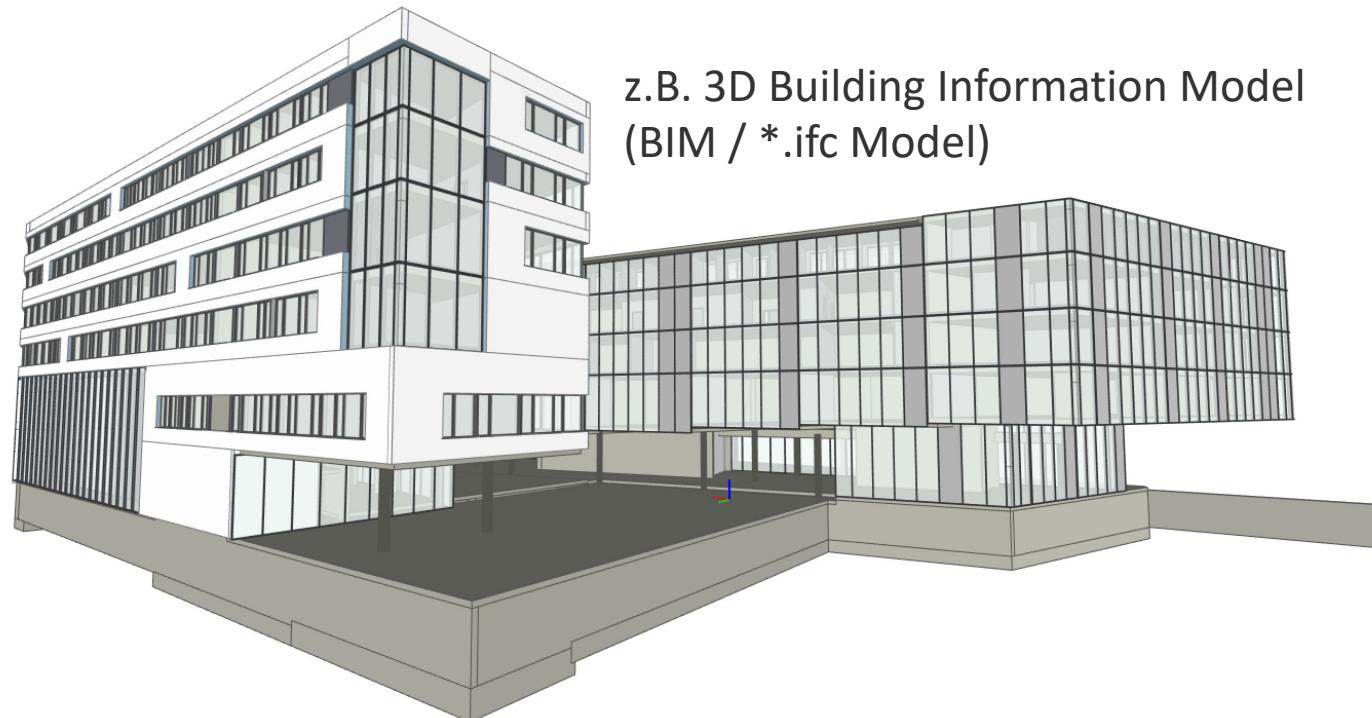
- .. ist eine **Datenbank**, welche für die Verarbeitung von räumliche Daten optimiert ist!
- Die Datenbank (DB) ist die Sammlung einheitlich beschriebener, persistent gespeicherter Daten.
- Das **Datenbankmanagementsystem (DBMS)** ist das Programm (packet), das die einheitliche Beschreibung, die sichere Verwaltung und die schnelle Abfrage der Datenbank ermöglicht.





- Ein klassischer Softwareentwicklungsprozess beginnt mit der **Anforderungsanalyse**, in der eine Analytiker:in versucht, mit einer Kund:in die Aufgaben des zu erstellenden Systems zu präzisieren.

- Welche Szenarien oder User Stories gibt es hier?
  - Fußgängernavigation mit dem Handy, Objektverfolgung, etc.
- Von welchen Daten sprechen wir hier?



- Um dies zu erreichen, muss man das Know-how und die Bedürfnisse der Kund:innen kennen.



- Ein **Indoor-Positionierungssystem** (IPS) ist ein Netzwerk von Geräten, die zur Ortung von Personen oder Objekten verwendet werden, wenn GNSS nicht präzise genug sind oder völlig versagen, beispielsweise in mehrstöckigen Gebäuden, Flughäfen, Gassen, Parkhäusern und unterirdischen Standorten.

Input:

- Autonome Ansätze
  - IMU (Nicht räumlich) > Inertial Localization
  - Kamera (Raster) > Visual Odometry
  - LiDAR (Punkte) > LiDAR Odometry
- Infrastrukturbasierte Ansätze
  - Beacons (Nicht räumlich) > Fingerprinting
  - Mobilnetzwerk (Nicht räumlich, Punkte)  
    > Triangulation, Trilateration



- Ein **Indoor-Positionierungssystem** (IPS) ist ein Netzwerk von Geräten, die zur Ortung von Personen oder Objekten verwendet werden, wenn GNSS nicht präzise genug sind oder völlig versagen, beispielsweise in mehrstöckigen Gebäuden, Flughäfen, Gassen, Parkhäusern und unterirdischen Standorten.

Input:

- Autonome Ansätze
  - IMU (Nicht räumlich) > Inertial Localization
  - Kamera (Raster) > Visual Odometry
  - LiDAR (Punkte) > LiDAR Odometry
- Infrastrukturbasierte Ansätze
  - Beacons (Nicht räumlich) > Fingerprinting
  - Mobilnetzwerk (Nicht räumlich, Punkte)  
> Triangulation, Trilateration

– Output:

- Position (Punkte)
- Trajektorie (Linie)

– Umgebung:

- 2D Karte (Fläche)
- 3D Model (ifc BIM Datei)



- Ein **Indoor-Positionierungssystem** (IPS) ist ein Netzwerk von Geräten, die zur Ortung von Personen oder Objekten verwendet werden, wenn GNSS nicht präzise genug sind oder völlig versagen, beispielsweise in mehrstöckigen Gebäuden, Flughäfen, Gassen, Parkhäusern und unterirdischen Standorten.

Input:

- Autonome Ansätze
  - IMU (Nicht räumlich) > Inertial Localization
  - Kamera (Raster) > Visual Odometry
  - LiDAR (Punkte) > LiDAR Odometry

– Output:

- Position (Punkte)
- Trajektorie (Linie)

• Infrastrukturbasierte Ansätze

- Beacons (Nicht räumlich) > Fingerprinting
- Mobilnetzwerk (Nicht räumlich, Punkte)  
    > Triangulation, Trilateration

– Umgebung:

- 2D Karte (Fläche)
- 3D Model (ifc BIM Datei)



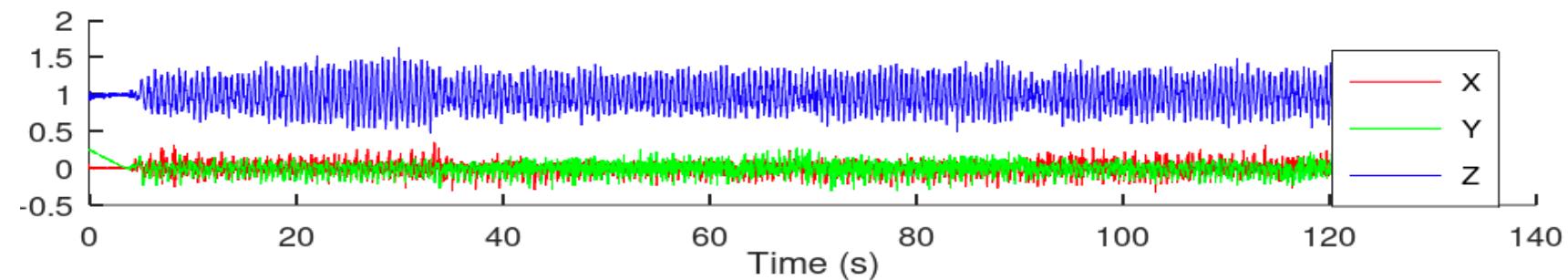
Input:

#Zeitstempel[ms]	BeschleunigungX[m/s <sup>2</sup> ]	BeschleunigungY[m/s <sup>2</sup> ]	BeschleunigungZ[m/s <sup>2</sup> ]	GyroskopX[rad/s]	GyroskopY[rad/s]	GyroskopZ[rad/s]
1655820538224	-1,051543	3,210835	10,203111	0,185284	0,081908	-0,334754
1655820538235	-1,051543	3,210835	10,203111	0,185284	0,081908	-0,334754
1655820538245	-1,051543	3,210835	10,203111	0,185284	0,081908	-0,334754
1655820538255	-0,874276	3,020314	9,244596	0,530737	-0,128037	-0,399611
1655820538265	-0,874276	3,020314	9,244596	0,530737	-0,128037	-0,399611

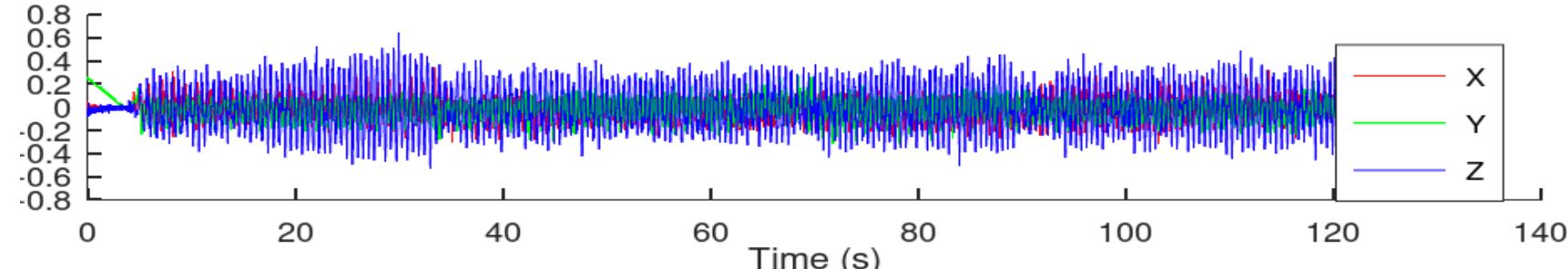
Zeitstempel

Sensoren X,Y,Z Achse

Beschleunigung [g]



Winkelgeschwindigkeit [rad/s]





Output:

Relativen Positionen (Trajektorie)

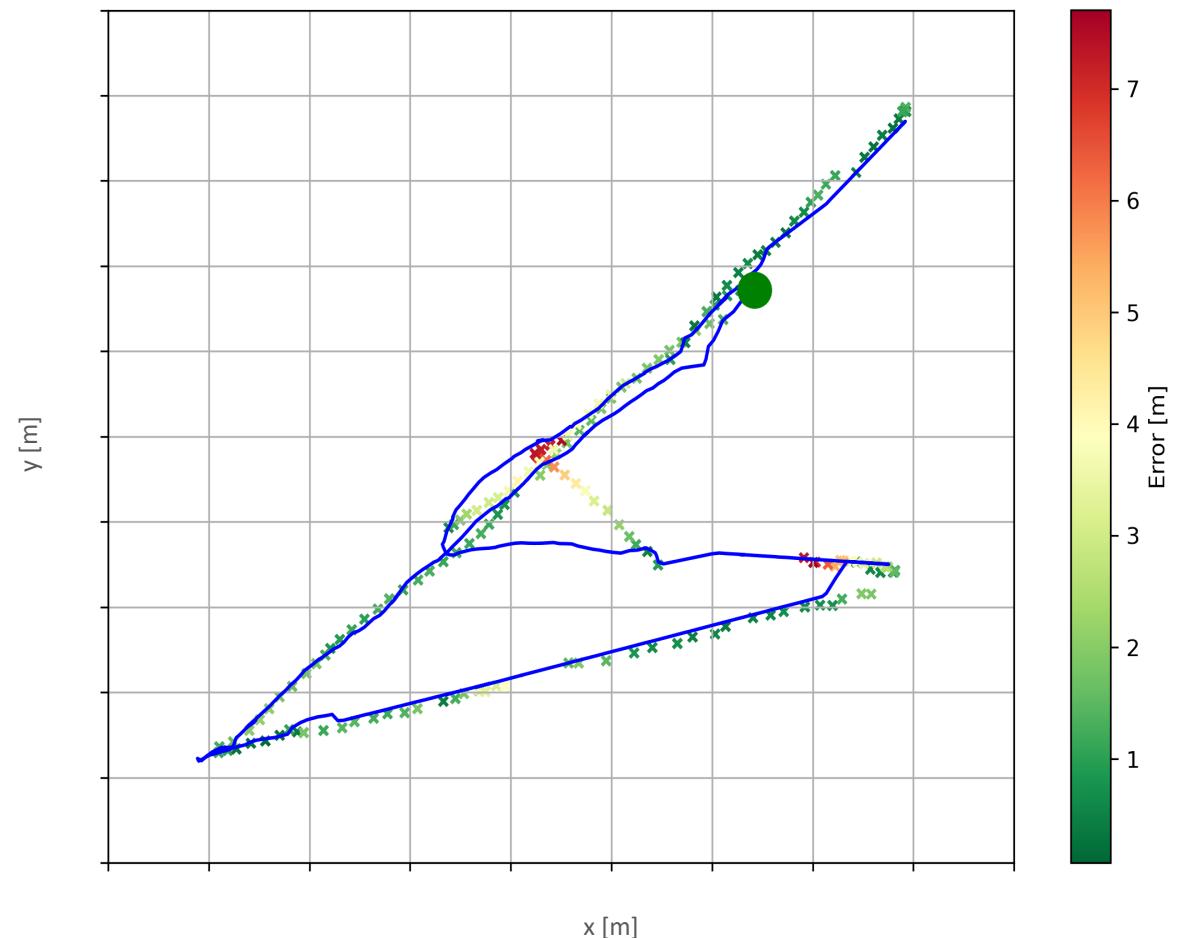
wenn

Anfangsposition und Anfangsrichtung

Bekannt ist....

Referenztrajektorie

Zur Bestimmung der Richtigkeit...

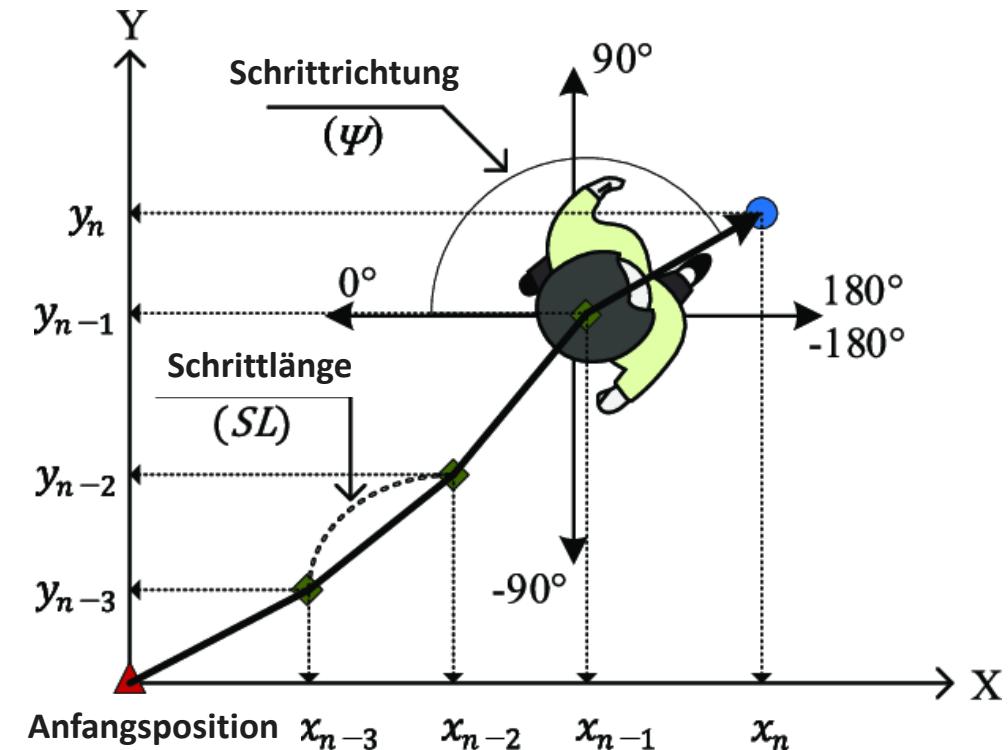




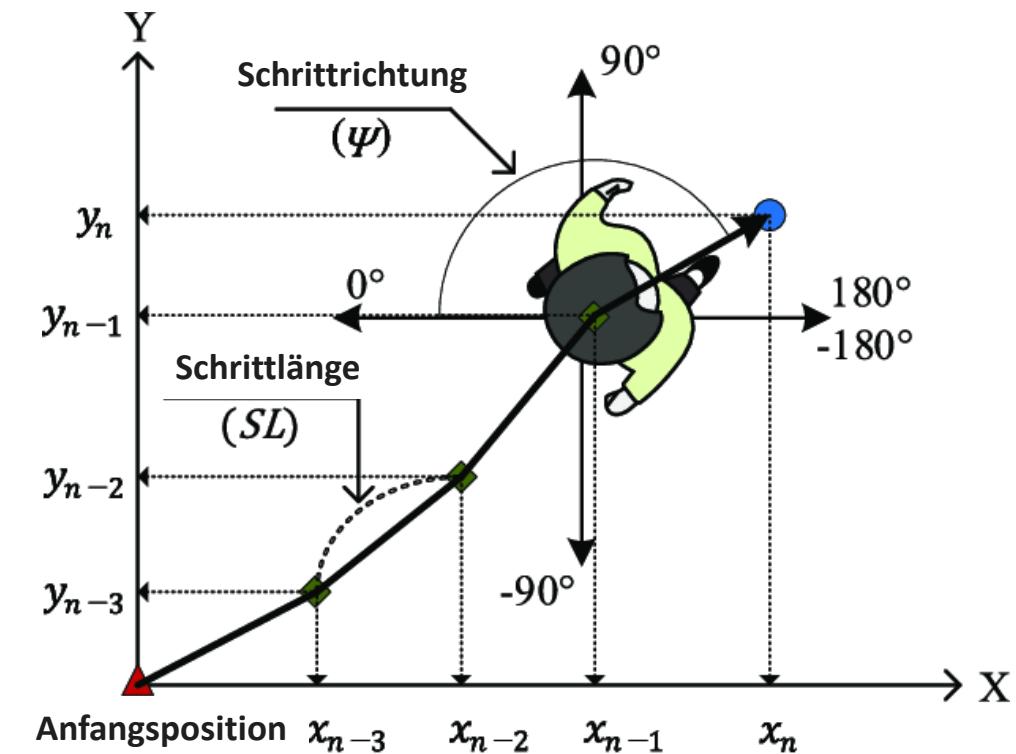
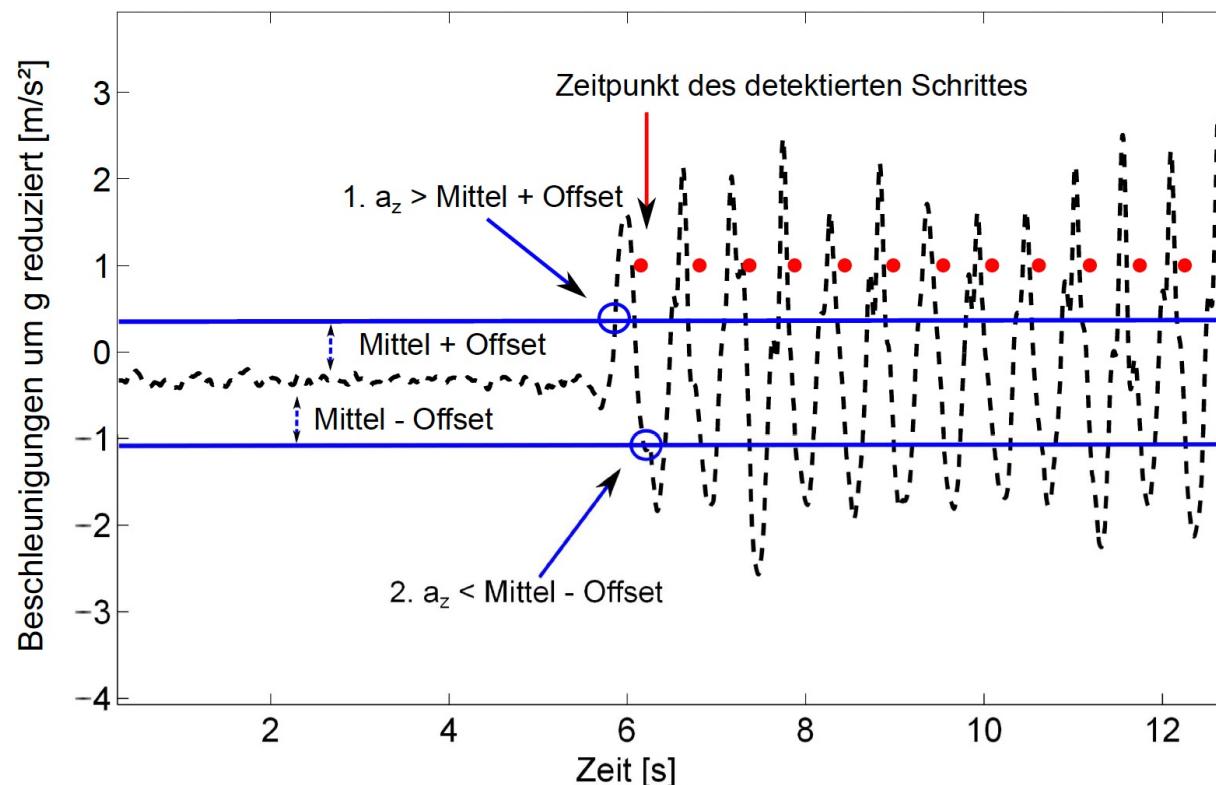
- **Inertial Lokalization** ist ein Verfahren zur relativen Positionsbestimmung, das aus einer Folge von Inertialsensormessungen (IMU-Daten) geschätzt wird.
- **relativen Positionsbestimmung:** Positionsbestimmung in Bezug auf den letzten Schritt oder die Anfangsposition.  
...Berechnung von  $\Delta x$ ,  $\Delta y$  und nicht  $x, y$ .

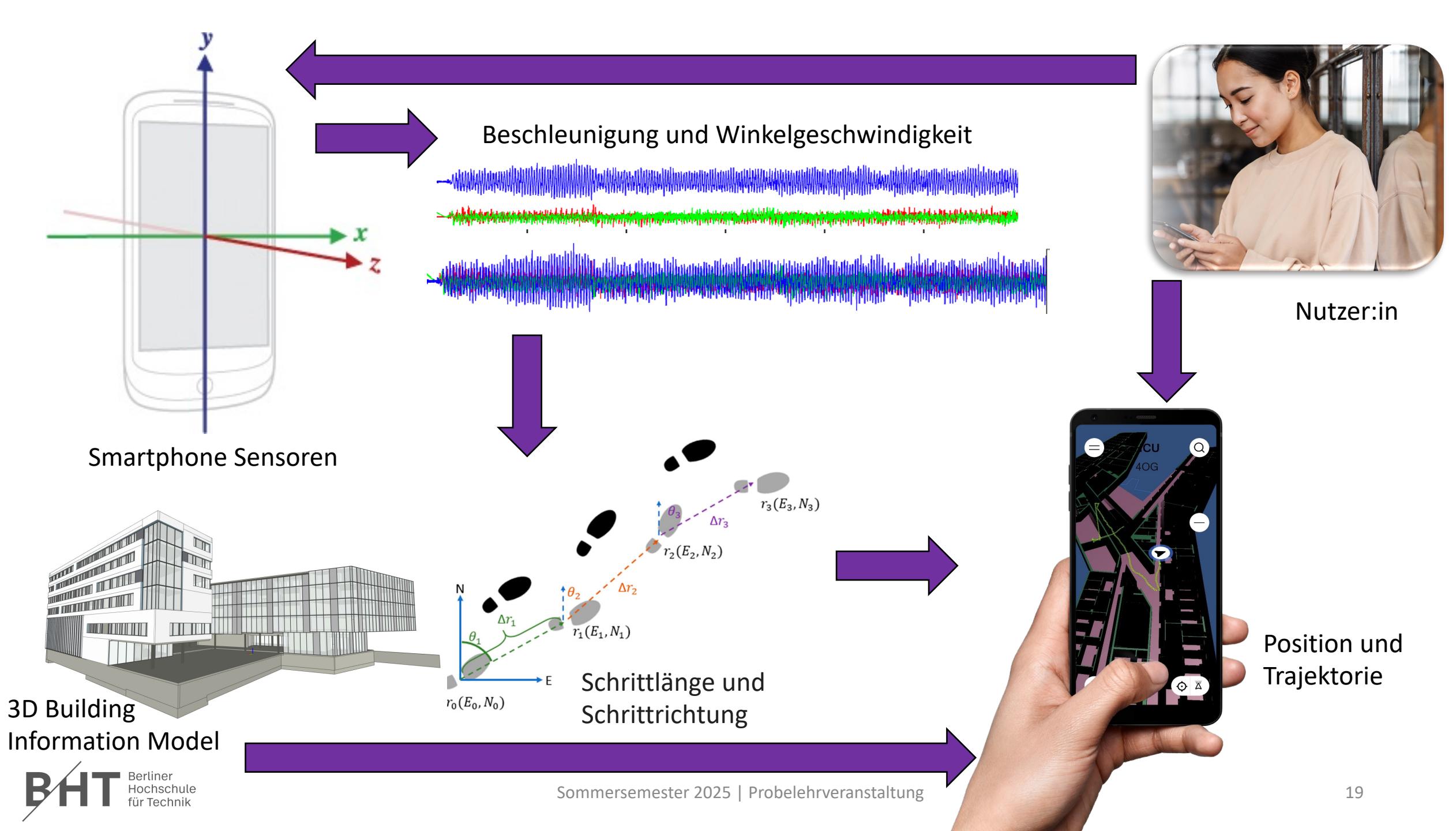


- **Inertial Localization** ist ein Verfahren zur relativen Positionsbestimmung, das aus einer Folge von Inertialsensormessungen (IMU-Daten) geschätzt wird.
- **relativen Positionsbestimmung:** Positionsbestimmung in Bezug auf den letzten Schritt oder die Anfangsposition.  
...Berechnung von  $\Delta x$ ,  $\Delta y$  und nicht  $x, y$ .
- **Pedesterian Dead Reckoning (PDR)** relative Positionsbestimmung einer sich bewegenden Person auf der Basis von Schritterkennung, Schrittänge und Schrittrichtung.

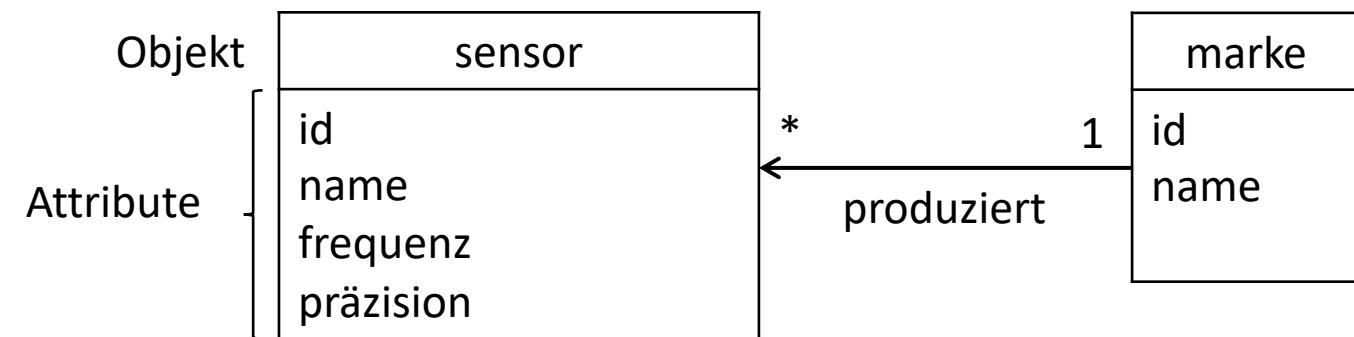


- **Pedesterian Dead Reckoning (PDR)** relative Positionsbestimmung einer sich bewegenden Person auf der Basis von Schritterkennung, Schrittänge und Schrittrichtung.





- Unified Modeling Language (UML): Eine Sprache, die auf Grafik- bzw. Diagrammen beruht.

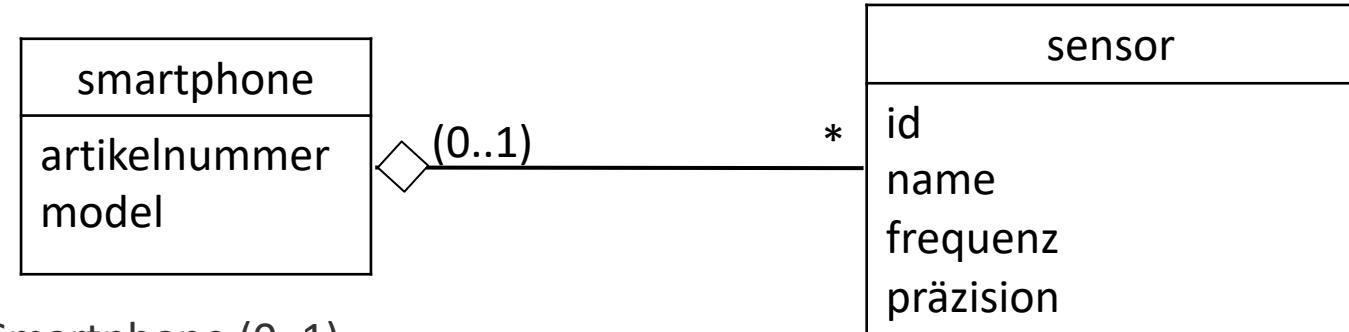


- **Assoziationen** sind Referenzen, die in einen Objekt gespeichert sind und auf ein Objekt verweisen.
- Beispiel: Jedes Objekt der Klasse „sensor“ speichert eine Referenz auf ein Objekt der Klasse „marke“.



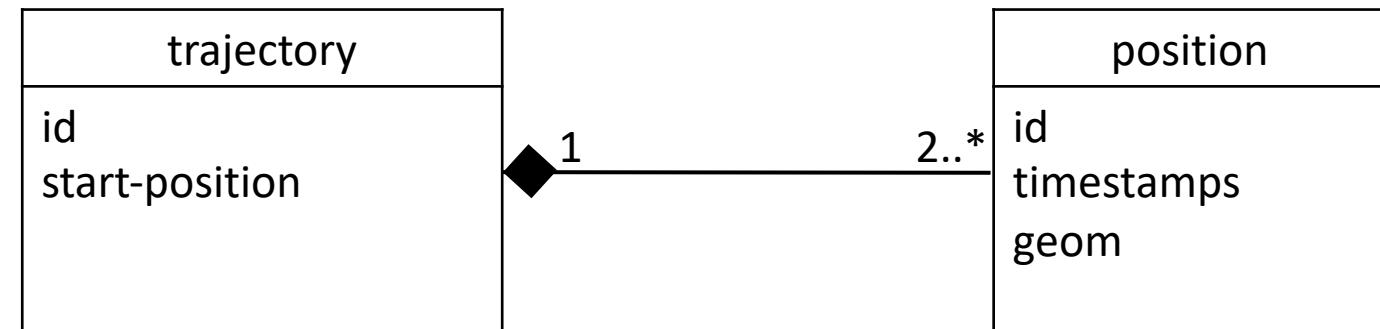
- Eine **Aggregation** ist eine besondere Assoziation, die eine Ist-Teil-von-Beziehung ausdrückt:

- Ein Sensor ist Teil von keinem oder genau einem Smartphone (0..1)
- Ein Smartphone hat viele Sensoren (\*)



- Eine **Komposition** ist eine Ist-Teil-von-Beziehung, bei der eine Abhängigkeit des Teils vom Ganzen besteht:

- Eine Trajektorie kann nicht ohne eine Position existieren.

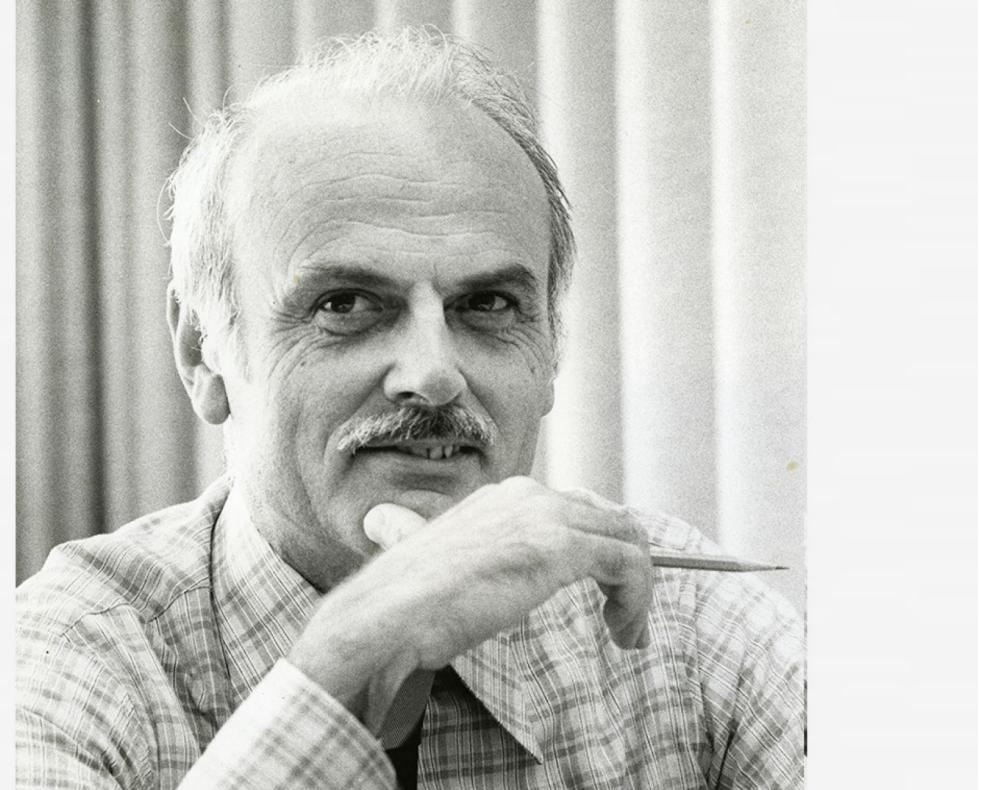
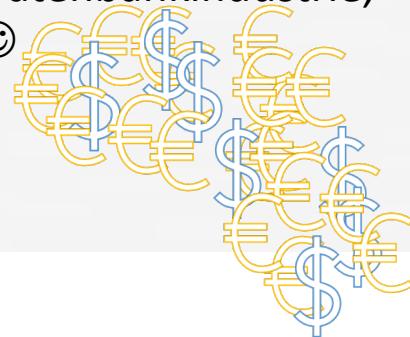


*Edgar F. Codd* (brit. Mathematiker bei IBM)

Der Erfinder hat relationale Datenbanken möglich gemacht...

Wie? ..das relationale Datenmodell wurde erstmals 1970 von ihm beschrieben.

Dann? ..Entstehung der Datenbankindustrie, die Milliarden einbringt ☺



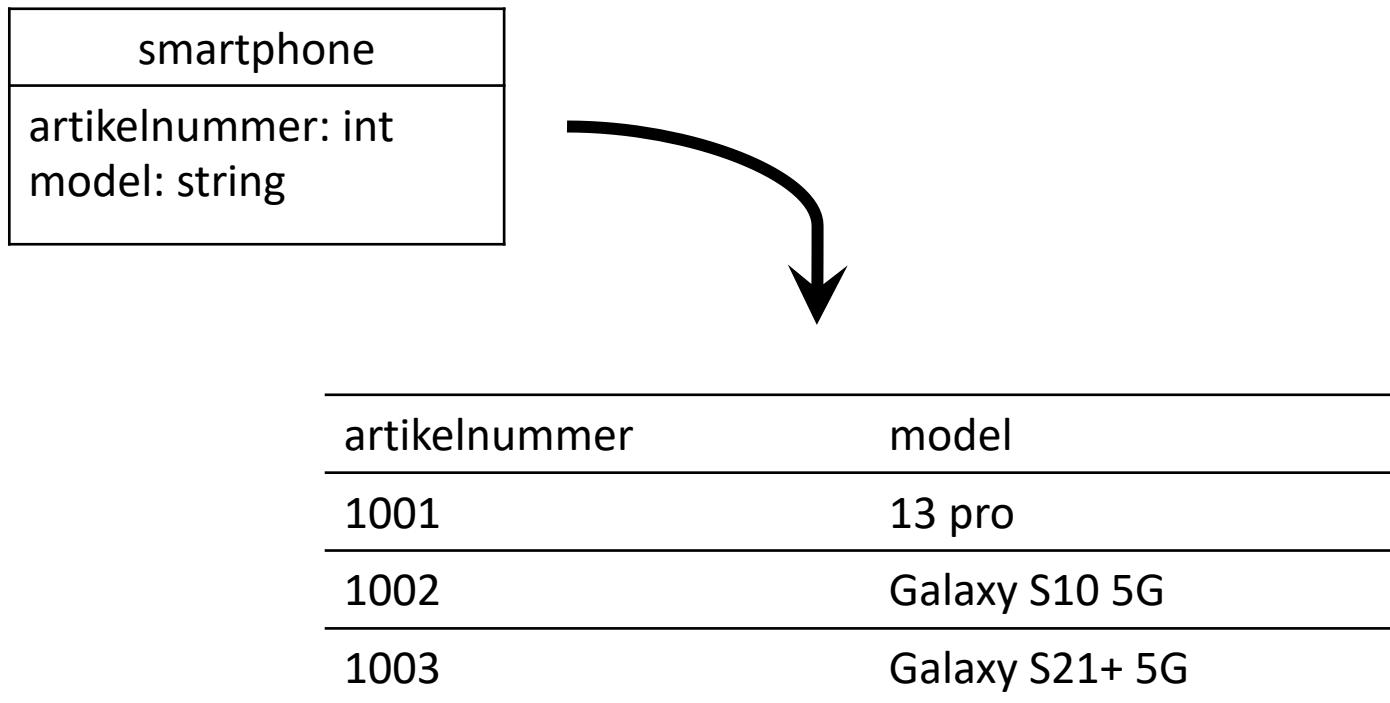
- Jede Klasse wird in eine Tabelle bzw. **Relation** übersetzt.
- Attribute werden als Spalten umgesetzt.
- Eine Zeile heißt **Tupel**

sensor
id: serial
name: string
frequenz: int
präzision: float



id	name	frequenz	präzision
1	Beschleunigung	200	0.01
2	Gyroskop	200	0.01
3	Gyroskop	100	0.001

- Jede Klasse wird in eine Tabelle bzw. **Relation** übersetzt.
- Attribute werden als Spalten umgesetzt.



- Assoziationen werden durch **Schlüsselverweise** realisiert.
- Jede Tabelle hat einen **Schlüssel** bzw. eine Spalte, mit deren Werten die Zeile identifiziert werden können.

smartphone	artikelnummer	model
	1001	13 pro
	1002	Galaxy S10

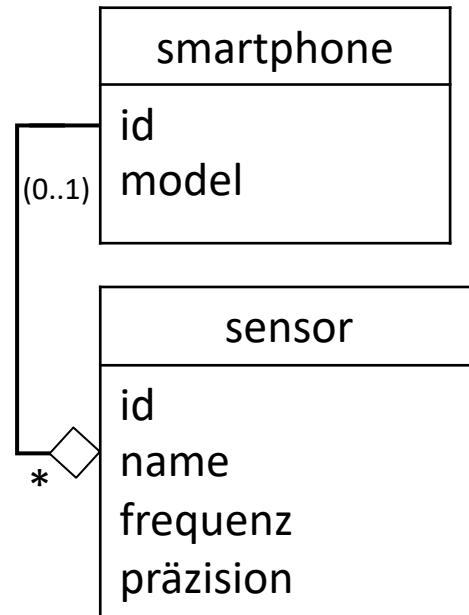
sensor	id	name	frequenz	praezision
	1	IMU	200	0.01
	2	IMU	100	0.001
	3	Kamera	30	0. 1

- Ein Schlüssel kann auch aus mehreren Spalten bestehen. z.B. Straße + Hausnummer als Schlüssel für Gebäude

- Ausgangstabelle der Assoziation erhält zusätzliche Spalte mit Schlüssel der Zieltabelle (**Fremdschlüssel**)

	smartphone	artikelnummer	model
		1001	13 pro
		1002	Galaxy S10

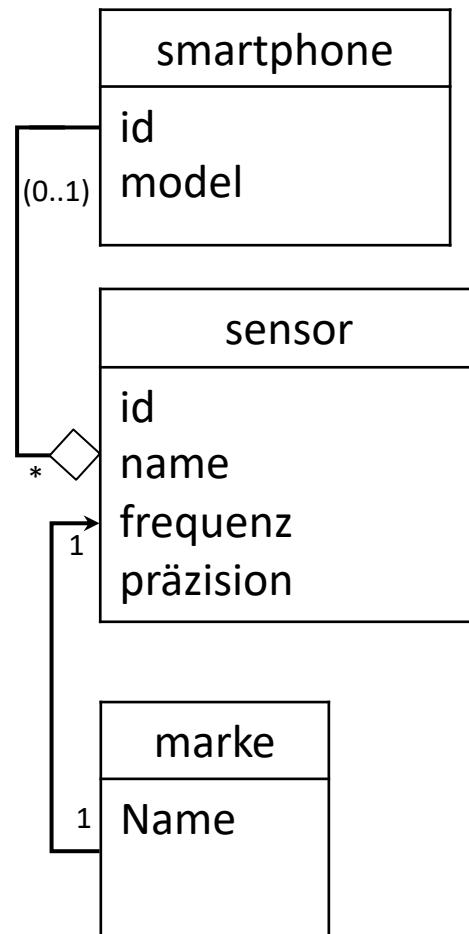
	sensor	id	name	frequenz	praezision	smartphone
		1	IMU	200	0.01	1001
		2	IMU	100	0.001	1002
		3	Kamera	30	0.1	1001



- Ausgangstabelle der Assoziation erhält zusätzliche Spalte mit Schlüssel der Zieltabelle (**Fremdschlüssel**)

	smartphone	artikelnummer	model
		1001	13 pro
		1002	Galaxy S10

	sensor	id	name	frequenz	praezision	smartphone	marke
		1	IMU	200	0.01	1001	Bosch
		2	IMU	100	0.001	1002	Null
		3	Kamera	30	0.1	1001	Sony



smartphone	artikelnummer	model
	1001	13 pro
	1002	Galaxy S10

sensor	id	name	frequenz	praezision	smartphone	marke
	1	IMU	200	0.01	1001	Bosch
	2	IMU	100	0.001	1002	Null
	3	Kamera	30	0. 1	1001	Sony

app-user	personalnummer	name	smartphone
	68548	Müller	1001
	68459	Mayer	1002

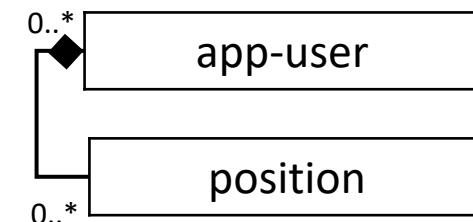
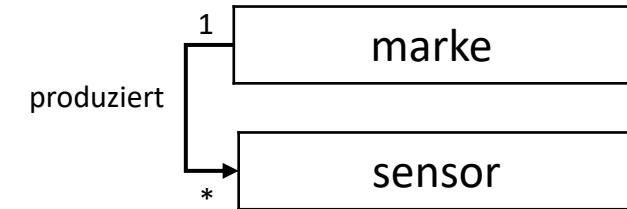
smartphone	artikelnummer	model	position
	1001	13 pro	POINT(30.3,10.1)
	1002	Galaxy S10	POINT(42.7,12.3)

sensor	id	name	frequenz	praezision	smartphone	marke
	1	IMU	200	0.01	1001	Bosch
	2	IMU	100	0.001	1002	Null
	3	Kamera	30	0. 1	1001	Sony

app-user	personalnummer	name	smartphone
	68548	Müller	1001
	68459	Mayer	1002

?

- Ausgangstabelle der Assoziation erhält zusätzliche Spalte mit Schlüssel der Zieltabelle (**Fremdschlüssel**)
- Dies funktioniert nur, wenn eine der Klassen die Multiplizität 1 hat.
- ungeklärtes Problem:  
Multiplizität > 1 auf beiden Seiten
- Denkbare Ansatz: Spalte mit Liste von Fremdschlüsseln



Dies widerspricht der Idee, Daten auf Spalten zu verteilen!  
Algorithmen zur Suche in relationalen Daten sind nicht auf diese Speicherung ausgelegt.

personalnummer	name	position
68548	Müller	101,102
68459	Mayer	105,106,107

- ungeklärtes Problem:

Multiplizität > 1 auf beiden Seiten

app-user

	personalnummer	name
	68548	Müller
	68459	Mayer

position

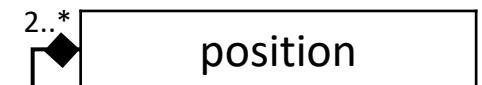
	id	timestamps	geom
	101	1742386937742	POINT(30.3,10.1)
	102	1742386937867	POINT(42.7,12.3)

- Lösung: Assoziation wird als Tabelle aufgelöst:

trajectory

	id	geom
	1	POINT(30.3,10.1) Line(ARRAY (position[101,102]))
	2	POINT(32.5,14.6) Line(ARRAY (position[105,106,107]))

personalnummer	name	position
68548	Müller	101, 102
68459	Mayer	105,106,107





- SQL (Structured Query Language): Standardisierte Sprache zur Definition und Manipulation relationaler Datenbanken



- Structured Query Language (SQL): Standardisierte Sprache zur Definition und Manipulation relationaler Datenbanken

sensor	id	name	frequenz	präzision	smartphone

```
CREATE TABLE sensor (
    id INT PRIMARY KEY NOT NULL,
    artikelnummer INT NOT NULL,
    name VARCHAR,
    frequenz INT,
    praezision FLOAT,
    FOREIGN KEY (smartphone) REFERENCES smartphone (artikelnummer)
);
```

- Structured Query Language (SQL): Standardisierte Sprache zur Definition und Manipulation relationaler Datenbanken

sensor	id	name	frequenz	präzision	smartphone
	1	IMU	200	0.01	1001
	2	IMU	100	0.001	1002
	3	Kamera	30	0. 1	1001

```
CREATE TABLE sensor (
    id INT PRIMARY KEY NOT NULL,
    artikelnummer INT NOT NULL,
    name VARCHAR,
    frequenz INT,
    praezision FLOAT,
    FOREIGN KEY (smartphone) REFERENCES smartphone (artikelnummer)
);
```

```
INSERT INTO sensor VALUES (1, 'IMU', 200, 0.01, 1001),
(2, 'IMU', 100, 0.001, 1002), (3, 'Kamera', 30, 0. 1, 1001);
```



position

	id	timestamps	geom
	101	1742386937742	POINT(30.3,10.1)
	102	1742386937867	POINT(42.7,12.3)

```
CREATE TABLE position (
    id INT PRIMARY KEY NOT NULL,
    timestamps INT,
    geom geometry(POINT, 25832)
);
```

## Koordinatenbezugssysteme (KBS)

- KBS in PostGIS werden durch SRIDs identifiziert
  - Spatial Reference System Identifier (SRID)
- EPSG Geodetic Parameter Datenbank (EPSG codes)
  - European Petroleum Survey Group Geodesy (EPSG)

Code	Name
4326	Geografische Koordinaten WGS84 Bezugssystem
25832	UTM Zone 32N, ETRS89 (European Terrestrial Reference System 1989)
3857	WGS 84 / Pseudo-Mercator
...	...



sensor	id	name	frequenz	präzision	smartphone
	1	IMU	200	0.01	1001
	2	IMU	100	0.001	1002
	3	Kamera	30	0. 1	1001

Welche Sensoren hat das Smartphone 1001?

SELECT \* FROM sensor WHERE smartphone = 1001;

alle Spalten      Tabelle

Auswahl der Zeilen

Ergebnis:

sensor	id	name	frequenz	präzision	smartphone
	1	IMU	200	0.01	1001
	3	Kamera	30	0. 1	1001



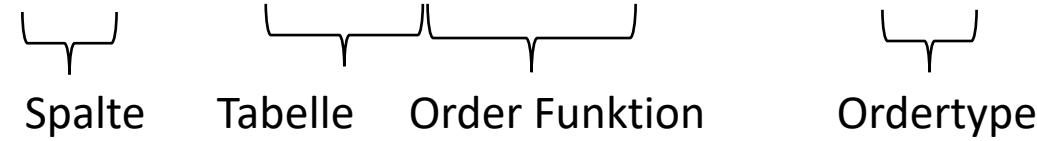
position

	id	timestamps	geom
101	1742386937742	POINT(30.3,10.1)	
102	1742386937867	POINT(42.7,12.3)	
103	1742386937875	POINT(46.1,13.5)	

Wo die Trajektorie beginnt und endet:

(SELECT geom FROM position ORDER BY timestamps ASC LIMIT 1)  
UNION ALL

(SELECT geom FROM position ORDER BY timestamps DESC LIMIT 1)



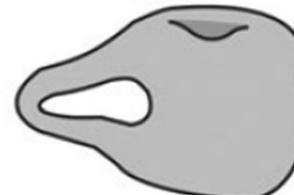
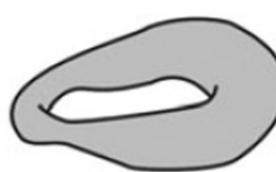
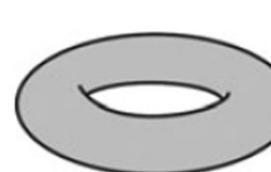
Ergebnis:

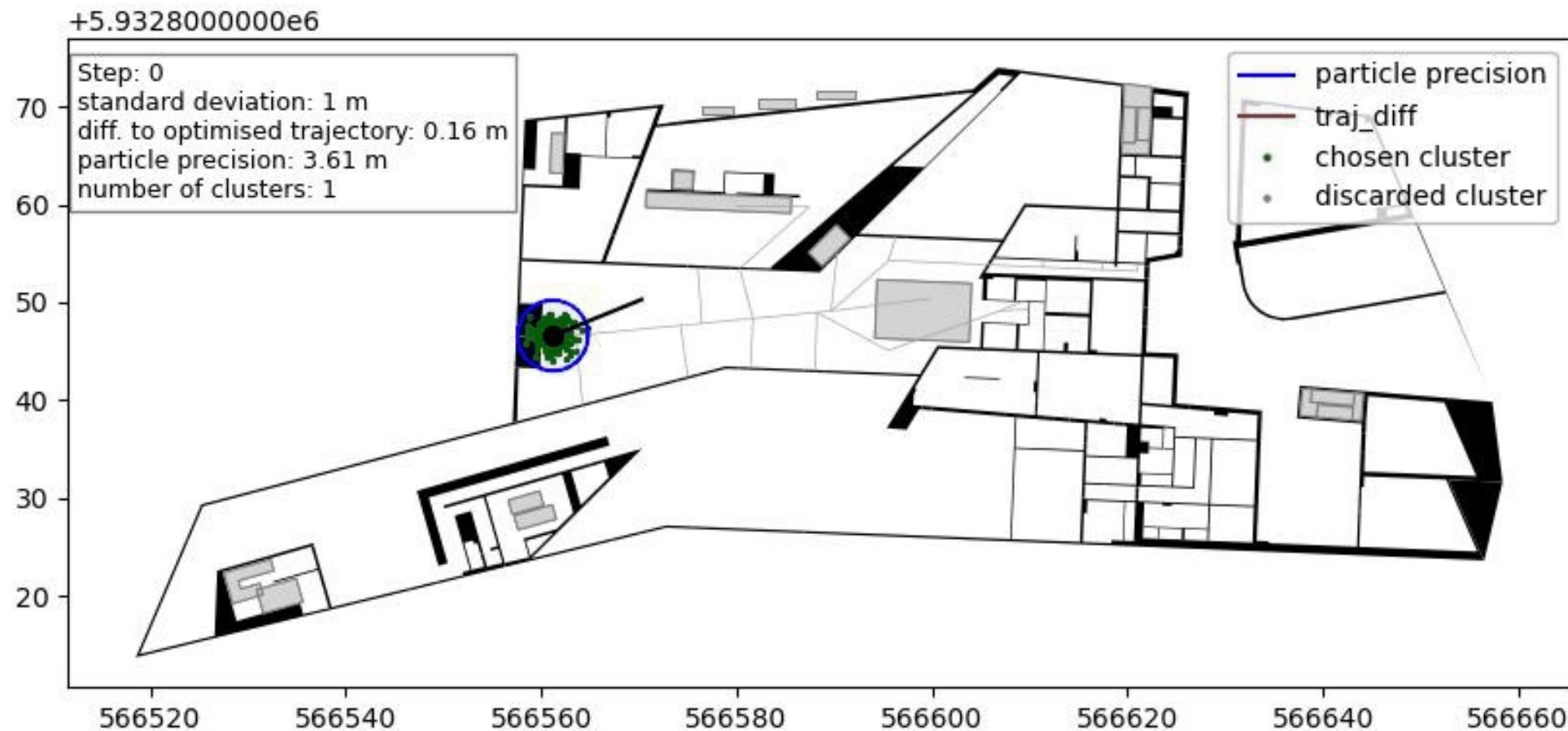
	id	timestamps	geom
101	1742386937742	POINT(30.3,10.1)	
103	1742386937875	POINT(46.1,13.5)	



- **Frage:** In welchem Raum befinden sich Frau Müller und Herr Mayer ☺ ? > Kartendaten speichern
- Dafür braucht man topologische Relationen:

Zwei Objekte A, B heißen **topologisch äquivalent**, wenn es eine topologische Transformation gibt, die A in B abbildet.







- Relationale Datenmodell
- Aufbau Datenbank in Tabellenform
- Grundlagen der SQL-Sprache (z. B. SELECT, INSERT, WHERE etc.)
- Daten in einer Datenbank speichern und abfragen
- Indoor-Positionierungsdaten Analyse
- gezielte Fragen zu den Daten, auch in Geodaten

# Übung

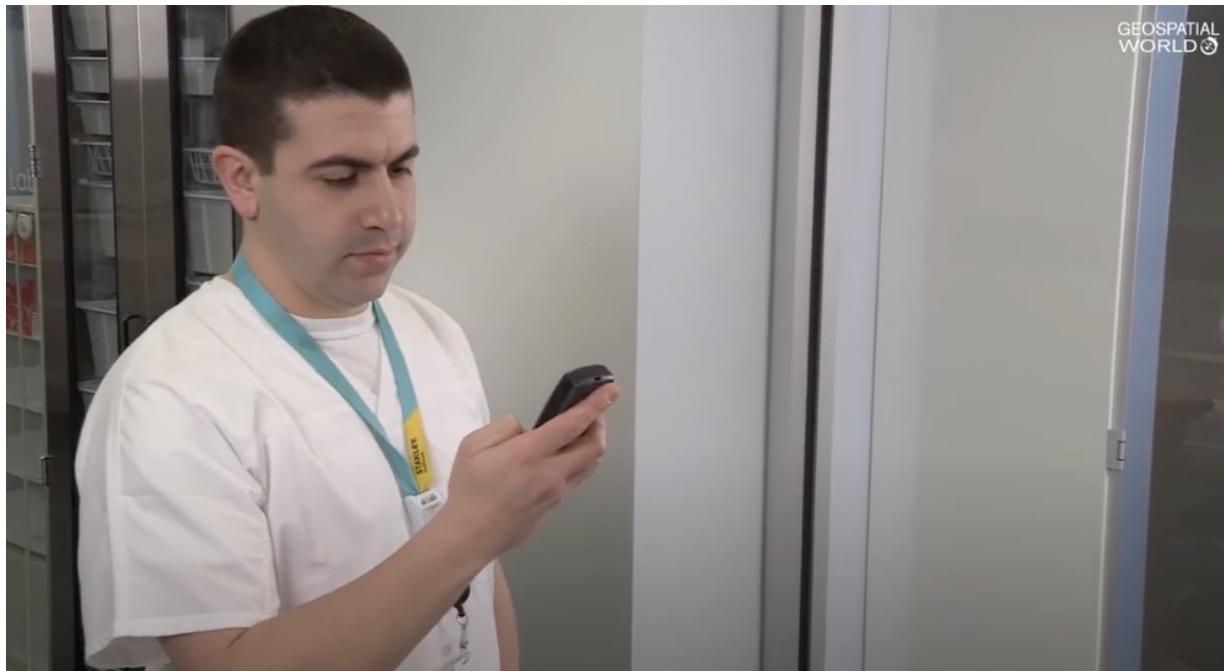
- Einweisung in eine praktische Übung



05-1

## Szenario

Die Leitung eines Krankenhauses soll durch ein mobiles Geoinformationssystem (GIS) unterstützt werden, um effizientere Dienste trotz Personalmangels sicherzustellen. Das mobile GIS bietet eine Navigationsfunktion sowohl für Besucher:innen als auch für das Krankenhauspersonal (z. B. Ärzt:innen, Pflegekräfte).



Wenn sich Besucher:innen im Krankenhaus orientieren möchten, können sie sich selber über das System navigieren, wodurch auch Kolleg:innen entlastet werden und Zeit eingespart wird. Sollte eine erkrankte Person Hilfe benötigen, kann diese über die App, das Telefon oder über Notrufknöpfe Unterstützung anfordern. Je nach Bedarf wird das nächstgelegene verfügbare Personal automatisch benachrichtigt. Dabei ist die Einhaltung des Datenschutzes für das Personal selbstverständlich gewährleistet.



05-2

## PostgreSQL und PostGIS

- PostgreSQL ist ein Open Source Objekt-relationales Datenbanksystem, das die Sprache SQL verwendet und erweitert (<https://www.pgadmin.org/>)



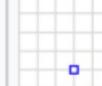
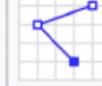
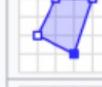
The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure under the 'Schemas' section, specifically the 'public' schema, which contains the 'clubkataster' table. The main pane shows the following SQL code:

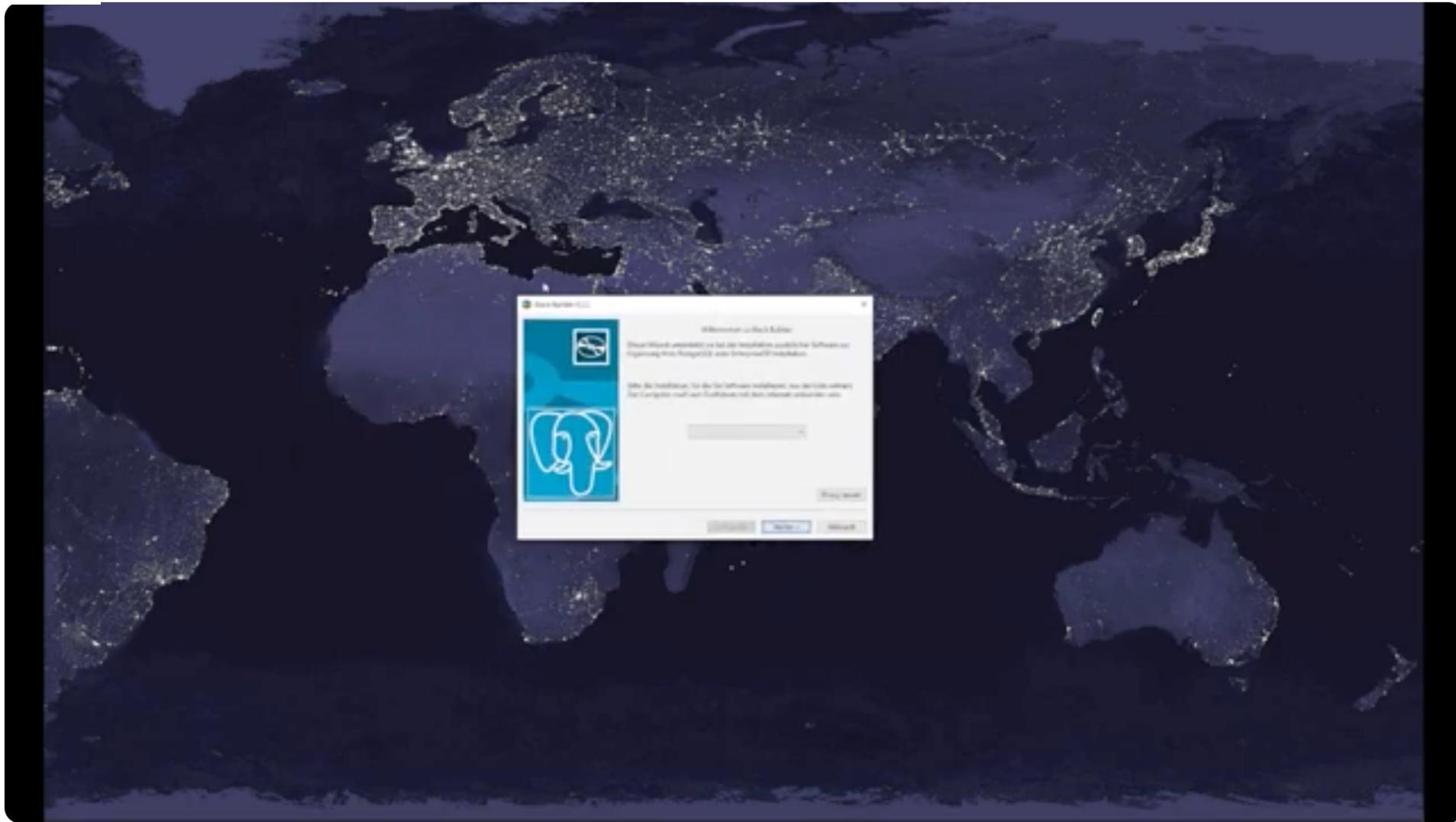
```
1 CREATE TABLE clubkataster (
2     id SERIAL PRIMARY KEY,
3     x DECIMAL,
4     y DECIMAL,
5     name TEXT,
6     club_adresse TEXT,
7     ort TEXT,
8     nummer INT,
9     info_kataster TEXT,
10    neu BOOLEAN,
11    geschlossen INT
12 );
```

The 'Messages' tab at the bottom indicates that the 'CREATE TABLE' command was successful.

- PostGIS als Erweiterung zu PostgreSQL (<https://www.postgis.net>)
- Unterschiedliche Ansätze zur Speicherung und Analyse von Geodaten
- PostGIS bietet uns WKT (Well Known Text) Format an
- Menschenlesbares Format (im Gegensatz zu Well Known Bytes (WKB))



Type	Examples
Point	 POINT (30 10)
LineString	 LINESTRING (30 10, 10 30, 40 40)
Polygon	 POLYGON ((30 10, 40 40, 20 40, 10 20, 30 10))
	 POLYGON ((35 10, 45 45, 15 40, 10 20, 35 10), (20 30, 35 35, 30 20, 20 30))



## PostgreSQL und PostGIS einfach und schnell installieren - Geodatenbanken



eine Linie aus zwei Punkten SQL



Snowflake Documentation

<https://docs.snowflake.com> › sql-reference › functions :

## ST\_MAKELINE

Im ersten Beispiel wird eine Linie zwischen zwei Punkten konstruiert: `SELECT ST_MAKELINE(TO_GEOGRAPHY('POINT(1.0 2.0)'), TO_GEOGRAPHY('POINT(3.5 4.5)')) AS ...`



create a line of two points sql



PostGIS

<https://postgis.net> › docs › ST\_... · Diese Seite übersetzen :

## ST\_MakeLine

Create a line composed of two points. `SELECT ST_AsText(ST_MakeLine(ST_Point(1,2), ST_Point(3,4)))`; st\_astext ----- ...

Konvertierung, Ausgabe,  
Formatierung  
  
ST\_ASGEOSTRING  
ST\_ASWKB  
ST\_ASBINARY  
ST\_ASEWKB  
ST\_ASWKT  
ST\_ASTEXT  
ST\_ASEWKT  
ST\_GEOHASH

#### Konstruktor

ST\_MAKELINE  
  
ST\_MAKEGEOMPOINT  
ST\_GEOPPOINT  
ST\_MAKEPOINT  
ST\_POINT  
ST\_MAKEPOLYGON  
ST\_POLYGON  
ST\_MAKEPOLYGONORIENTED

Accessor (Zugriffsmethode)  
  
ST\_DIMENSION  
ST\_ENDPOINT  
ST\_POINTN  
ST\_SRID  
ST\_STARTPOINT  
ST\_X  
ST\_XMAX

#### Kategorien:

Geodatenfunktionen

# ST\_MAKELINE

Konstruiert ein [GEOGRAPHY](#)- oder ein [GEOMETRY](#)-Objekt, das eine Linie repräsentiert, die die Punkte in den Eingabeobjekten verbindet.

#### Siehe auch:

[TO\\_GEOGRAPHY](#), [TO\\_GEOMETRY](#)

#### Auf dieser Seite

Syntax  
Argumente  
Rückgabewerte  
Nutzungshinweise  
Beispiele  
  
Verwandte Inhalte  
Datentypen für Geodaten

## Syntax

```
ST_MAKELINE( <geography_expression_1> , <geography_expression_2> )  
ST_MAKELINE( <geometry_expression_1> , <geometry_expression_2> )
```

## Argumente

<*geography\_expression\_1*>

Ein GEOGRAPHY-Objekt, das die zu verbindenden Punkte enthält. Dieses Objekt muss ein Punkt, MultiPoint oder LineString sein.

<*geography\_expression\_2*>

Ein GEOGRAPHY-Objekt, das die zu verbindenden Punkte enthält. Dieses Objekt muss ein Punkt, MultiPoint oder LineString sein.

## Rückgabewerte

Die Funktion gibt einen Wert vom Typ GEOGRAPHY oder GEOMETRY zurück. Der Wert ist ein LineString, der alle durch die GEOGRAPHY- oder GEOMETRY-Eingabeobjekte angegebenen Punkte verbindet.

## Beispiele



05-2

- LINestring von Zwei Punkten?

```
SELECT ST_MAKELINE(  
    TO_GEOGRAPHY('POINT(37.0 45.0)'),  
    TO_GEOGRAPHY('POINT(38.5 46.5)')  
) AS line_between_two_points;  
+-----+  
| LINE_BETWEEN_TWO_POINTS |  
+-----+  
| LINESTRING(37 45,38.5 46.5) |  
+-----+
```

- Wie kann man Trajektorie als LINestring definieren?

```
SELECT gps.track_id, ST_MakeLine(gps.geom ORDER BY gps_time) As geom  
FROM gps_points As gps  
GROUP BY track_id;
```

# **Geodatenbank, SoSe 2025**

## **Hausübungsblatt #1**

### **Aufgabe 1**

**6 Punkte**

**Abgabe: Dienstag, 06.05.2025, 23:55 Uhr.**

Die Leitung eines Krankenhauses soll durch ein mobiles Geoinformationssystem (GIS) unterstützt werden, um effizientere Dienste trotz Personalmangels sicherzustellen. Das mobile GIS bietet eine Navigationsfunktion sowohl für Besucher:innen als auch für das Krankenhauspersonal (z. B. Ärzt:innen, Pflegekräfte).

Wenn sich Besucher:innen im Krankenhaus orientieren möchten, können sie sich selber über das System navigieren, wodurch auch Kolleg:innen entlastet werden und Zeit eingespart wird. Sollte eine erkrankte Person Hilfe benötigen, kann diese über die App, das Telefon oder über Notrufknöpfe Unterstützung anfordern. Je nach Bedarf wird das nächstgelegene verfügbare Personal automatisch benachrichtigt. Dabei ist die Einhaltung des Datenschutzes für das Personal selbstverständlich gewährleistet.

Die Arbeitsgeräte (Smartphones oder Tablets) sind eindeutig identifizierbar und mit verschiedenen Sensoren (z. B. Beschleunigungssensor, Gyroskop mit Messung in drei Achsen) ausgestattet. Die Sensordaten werden in einer Datei `IMU.csv` gespeichert. Ein in der Forschung entwickelter Inertialalgorithmus kann anhand dieser IMU-Daten sowie einer Startposition und -richtung eine relative Position berechnen. Die berechneten Positionsdaten liegen in der Datei `Positions.csv` vor.

Für die automatische Integration in ein GIS-System soll eine **PostgreSQL-Datenbank** verwendet werden – idealerweise mit der Erweiterung **PostGIS** zur Durchführung geobasierter Analysen.

### *Aufgabenstellung:*

Erstellen Sie auf Basis des beschriebenen Szenarios eine Geodatenbank. Gehen Sie dabei wie folgt vor:

- Erstellen Sie eine Tabelle in Ihrer PostgreSQL-Datenbank, welche mindestens alle Spalten der CSV-Datei (`IMU.csv` oder `Positions.csv`) abbildet.  
*Hinweis:* Legen Sie eine ID-Spalte mit `SERIAL PRIMARY KEY` an – das erleichtert spätere Abfragen.
- Importieren Sie den Inhalt der CSV-Dateien in die entsprechenden Tabellen.  
*Hinweis:* Eine Anleitung zum CSV-Import finden Sie auf der Kurswebseite bzw. in den Übungsunterlagen.

### **Individuelle Abgabe:**

Bitte melden Sie sich bis zum Abgabetermin per E-Mail zurück und beschreiben Sie kurz:

- Wie Sie die Aufgaben umgesetzt haben
- Welche Herausforderungen oder Fragen dabei aufgetreten sind

1

## Aufgabe 2

14 Punkte

**Abgabe: Dienstag, 13.05.2025, 23:55 Uhr.**

Basierend auf den importierten und gespeicherten Daten (z. B. Positionsdaten) sollen Sie nun gezielte GIS-Abfragen durchführen.

*Bearbeiten Sie bitte folgende Teilaufgaben:*

- Erstellen Sie eine zusätzliche Spalte vom Typ `geometry(Point, 25832)` zur Speicherung von Punkt-Geometrien.
- Schreiben Sie eine `UPDATE`-Abfrage, mit der Sie die Geometriespalte auf Basis der X- und Y-Koordinaten mit Punkten füllen.
- Berechnen Sie den Abstand (Luftlinie) zwischen dem Beginn und dem Ende der Trajektorie und geben Sie das Ergebnis in Metern aus.
- Erstellen Sie eine zusätzliche Geometriespalte für die vollständige Trajektorie und berechnen Sie die zurückgelegte Wegstrecke (z. B. mit `ST_Length` und `ST_MakeLine`).
- Erstellen Sie eine weitere Geometriespalte mit der SRID 4326 und füllen Sie diese durch Transformation der bisherigen Punktgeometrien (z. B. mit `ST_Transform`).

### Gruppenarbeit:

Bitte bilden Sie Zweiergruppen und reichen gemeinsam einen kurzen Bericht ein, der folgende Inhalte enthält:

- Detaillierte Beschreibung der Schritte (inkl. SQL-Abfragen)
- Know-how über das Vorgehen mit PostGIS
- Ergebnisse der einzelnen Abfragen



Vorlesungsskript:  
QR der Kurs-Websites:

Hossein Shoushtari  
Wissenschaftlicher Mitarbeiter  
HafenCity Universität & Arcelormittal Bremen  
[hossein.shoushtari@hcu-hamburg.de](mailto:hossein.shoushtari@hcu-hamburg.de)  
[hossein.shoushtari@arcelormittal.com](mailto:hossein.shoushtari@arcelormittal.com)