## Wikidata-Datenmodell

Wie bereits im Kapitel 4 gesagt wurde, ist Wikidata eine multilinguale Graph-Datenbank, in der sowohl der Inhalt als auch Struktur des Graphen bearbeitet werden können. Es wird einem streng semantischen Prinzip durch Properties gefolgt. Klassen bzw. Instanzen können dabei frei erzeugt werden. Da die Daten in Wikidata als RDF-Tripeln gespeichert werden, lässt sich das Datenmodell von Wikidata auf einfacher Ebene gut mit RDF abbilden: mit Item (rdf:subject), Property (rdf:predicate) und Wert (rdf:object) (vgl. [RDF und Wikidata](https://www.wikidata.org/wiki/Wikidata:Relation_between_properties_in_RDF_and_in_Wikidata)). Im Rahmen dieser Arbeit wurde zwei Typen von Items verwendet: Person mit biographischer Datensätze und Weg zu Missionsorten.

Da sich die Informationen über die Personen nicht als homogen zeigten, wurde sämtliches verfügbares Material im Datenmodell aufgenommen. Die Tabelle 5.2 zeig die Modellierung aller denkbaren Relationen zu einer Person. Wikidata-Elementen wurden mithilfe der Wikidata-Eigenschaft P31 (Instanz bzw. Klasse) ein Typ Q5 (ein Mensch) zugewiesen. Es gibt die typische Information für eine Person wie Geschlecht, Verwandtschaften, Ehrpartner:in, die Ausbildung, wichtige Datenangaben. Ein ergänzendes Element des Datenmodell ist die Verwendung von Qualifikatoren. Sie ermöglichen es, Relationen mit zusätzlichen Kontextinformationen anzureichern: mit Datumsangaben (pq:580, pq:582 und pq:585) oder mit Quellenangaben zu Primärdigitalisat (pq:584). Um die in dieser Arbeit erzeugte Daten von anderen abzugrenzen, wurde ein Q-ID *WikiProjekt Digitalisat2Graph* erstellt und die Daten diesem zugeordnet.

|  |  |  |
| --- | --- | --- |
| Wikidata-Property | Beschreibung | Beispiel |
| rdfs:label (@de) | *Label* des Items (Deutsch, Literal) | Johann Christian Breutel (wd:Q21415674) |
| wdt:P31 | *Instanz* von (verlinkt) | Mensch (wd:Q5) |
| wdt:P19 | *Geburtsort* (verlinkt) | Weißenburg in Bayern (wd:Q44500) |
| wdt:P20 | *Sterbeort* (verlinkt) | Berthelsdorf (wd:Q502553) |
| wdt:P21 | *Geschlecht* (verlinkt) | männlich (wd:Q6581097) |
| wdt:P22 | *Vater* (verlinkt) | Johann Christian Breutel (wd:Q21415674) |
| wdt:P25 | *Mutter* (verlinkt) | Sophie Breutel (wd:Q125843987) |
| wdt:P26 | *Ehrpartner(in)* (verlinkt) | Sophie Breutel (wd:Q125843987) |
| wdt:P40 | *Kind(er)* (verlinkt) | Johann Christian Breutel (wd:Q21415674) |
| wdt:P69 | *besuchte Bildungseinrichtung* | Universität Leipzig (wd:Q154804) |
| wdt:P106 | *Tätigkeit* (verlinkt) | Botaniker (wd:Q2374149) |
| wdt:P463 | *Mitglied von* (verlinkt) | Herrnhuter Brüdergemeine (wd:Q159318) |
| wdt:P551 | *Wohnsitz* (verlinkt) | Berthelsdorf (wd:Q502553) |
| wdt:P569 | *Geburtsdatum* (Literal), ISO: YYYY-MM-DD/ YYYY-MM/ YYYY | 1788-01-21 |
| wdt:P570 | *Sterbedatum* (Literal), ISO: YYYY-MM-DD/ YYYY-MM/ YYYY | 1875-02-18 |
| wdt:P793 | *Schlüsselereignis* (verlinkt) | Weg zum Missionsort (Südafrika) (wd:Q135874728)/ Konfirmation (wd:Q214802) |
| wdt:P937 | *Wirkungsort* (verlinkt) | Südafrika (wd:Q258) |
| wdt:P1066 | *Schüler von* (verlinkt) | Johann Jürgen Stein (wd:Q123456933) |
| wdt:P1971 | *Anzahl der Kind(er),* (Literal) | 7 |
| wdt:P3373 | *Geschwister* (verlinkt) | Johann Christian Breutel (wd:Q21415674) |
| wdt:1344 | *Teilnahme an* (verlinkt) | Weg zum Missonsort (Südafrika): Johann G. und Johanna D. Bonatz (wd:Q135874728) |
| wdt:P5008 | *auf der Arbeitsliste des Wikimedia-Projektes* (verlinkt) | Digitalisat2Graph (wd:Q136079887) |
| pq:P580 | *Startzeitpunkt (Literal), ISO: YYYY-MM-DD/YYYY-MM/YYYY* | 1822-06-08 |
| pq:P582 | *Endzeitpunkt (Literal), ISO: YYYY-MM-DD/YYYY-MM/YYYY* | 1822-06-08 |
| pq:P585 | *Zeitpunkt/Stand (Literal), ISO: YYYY-MM-DD/YYYY-MM/YYYY* | 1822-06-08 |
| pq:P854 | *URL der Fundstelle (URL)* | URL |

Tab. 5.2: Datenmodell für biographische Datensätzen in Wikidata

Der zweite Typ des Datenmodells umfasst die Wege zu Missionsorten. In den untersuchten Lebensläufen wurden neben biographischen Angaben auch die Reisen von Missionaren beschrieben: sowohl die Anreise zu einer Missionsstelle als auch Reisen zwischen verschiedenen Missionsorten (Elim, Silo usw.). Letztere waren jedoch nicht konsistent dokumentiert und die Beschreibungen dieser Reisen waren häufig unterbrochen. Dies machte die chronologische Rekonstruktion nicht möglich. Daher wurde entschieden, sich ausschließlich auf die Wege bis zur jeweiligen Missionsstelle zu konzentrieren. Dabei wurden drei solche Zielrichtungen identifiziert: Westindien, Labrador und Südafrika.

In den Lebensbeschreibungen wurden jeweils die Route sowie die Zusammensetzung der Mitreisenden festgehalten. Diese Informationen wurden zuerst in Tripeln-Form modelliert. Anschließend wurden die einzelnen Reisen in Wikidata je Reise nach Rote und Mitreisenden gruppiert und als eingeständige Einträge in Wikidata angelegt.

Das Modell erlaubt einige Eigenschaften zu beschreiben (Tabelle 5.3). Über wdt:P31 (Instanz von) wurde die Klasse *Route* definiert. Für jede Route wurden der geographische Start- (wdt:P1427) und Endpunkt (wdt:1444) erfasst und bei Bedarf lassen sich auch Zwischenstationen (wdt:P2825) optional angeben. Besonders wichtig ist hier der Einsatz von Qualifikatoren. Sie präzisieren die Aussagen zu Route, durch die Modellierung eines Datums (pq:P585). Die Quellennachweise wurden als Referenzen hinterlegt und über P854 verknüpft.

|  |  |  |
| --- | --- | --- |
| Wikidata-Property | Beschreibung | Beispiel |
| rdfs:label (@de) | *Label* des Items (Deutsch, Literal) | Weg zum Missonsort (Südafrika) (Q135874728) |
| wdt:P31 | *Instanz* von (verlinkt) | Route (Q1322323) |
| wdt:1427 | *geografischer Startpunkt* (verlinkt) | Herrnhut (wd:Q165140) |
| wdt:P1444 | *geografischer Endpunkt* (verlinkt) | Genadendal (wd:Q1501295) |
| wdt:P2825 | *Zwischenpunkt einer Reise* (verlinkt) | Herrnhut (wd:Q165140) |
| pq:P585 | *Zeitpunkt/Stand* (Literal), ISO: YYYY-MM-DD/YYYY-MM/YYYY | 1822-06-08 |
| pq:P854 | *URL der Fundstelle* (URL) | URL |
| wdt:P5008 | *auf der Arbeitsliste des Wikimedia-Projektes* (verlinkt) | Digitalisat2Graph (wd:Q136079887) |

Tab. 5.3 Datenmodell für Routen in Wikidata

Die Verknüpfung zwischen Personen und Routen wurde auf Ebene der Personeninstanz modelliert. Die Zuordnung erfolgt mittels Property P1344 (*Teilnahme an*). Als P-Wert wurde die Q-ID des jeweiligen Roteobjekts hinterlegt. Die Modellierung der Missionswege wurde als besonders relevant erachtet, da sie ermöglichen, den Verlauf der Reise zu nachvollziehen und dabei die zentrale Umsteigpunkte zu identifizieren. Dadurch lassen sich die Knotenpunkte rekonstruieren, die für die Missionsarbeit eine wichtige Rolle spielen.

Mit dem Tool [QuickStatements](https://quickstatements.toolforge.org/) wurden die tabellarisch aufbereiten Daten in Wikidata hochgeladen. Jede Zeile kodiert eine Aussage zu einem Wikidata-Item und enthält mindestens Subjekt (Q-ID), Prädikat (P-ID) und Objekt (Wert). Optional können Qualifikatoren und Referenzen mitgegeben werden (Abbildung 5.8).

Ein Bild, das Text, Screenshot enthält.

KI-generierte Inhalte können fehlerhaft sein.

Abb. 5.8 Screenshot: QuickStatements

Nach dem Upload liegen die Informationen als Wikibase-Statement in Wikidata vor. Die RDF-Repräsintation wird von Wikidata bereitgestellt und kann u.a. im Turtle-Format aufgerufen werden. Eine Turtle-Datei beinhaltet [Präfix-Deklarationen](https://www.wikidata.org/wiki/EntitySchema:E49), Metadaten zum Datensatz und die direkten Aussagen eines Wikidata-Objekts mit Qualifikatoren und Referenzen. Das letzteres stellt die Abbildung 5.9 dar.

Ein Bild, das Text, Screenshot, Schrift enthält.

KI-generierte Inhalte können fehlerhaft sein.

Abb. 5.9 Ausschnitt aus der Turtle-Datei zum Datensatz Marianne Bonatz (Q135317408)

Die dargestellten Tripel folgen dem RDF-Datenmodell und spiegeln die semantische Struktur von Wikidata wider. Jede einzelne Aussage über ein Objekt wird durch eine eindeutige URI mit dem wds:-Präfix (Abbildung 5.8, rot markiert) gekennzeichnet. Dadurch wird die Aussage selbst als einstündige Ressource behandelt, sodass zusätzliche Informationen wie Qualifikatoren und Referenzen (Abbildung 5.8, lila markiert) direkt an die Aussage angehängt werden können.

Eine große Herausforderung bei der Datenmodellierung bestand in der Zuordnung von Relationen zwischen Hilfstabelle und Wikidata. Besonders anspruchsvoll war die Abgrenzung von Ortsaufenthalte (Wohnsitz, Wirkungsort) und Reisetätigkeit. Aufenthalte von Personen in Städten, die etwa dem Erlenen einer Fremdsprache oder dem langzeitlichen Besuch dienten, wurden als Wohnsitz modelliert. Als Wirkungsorte wurden alle Städte markiert, in denen die jeweilige Person länger- oder kurztätig war bspw. zu der Teilnahme an einer Konferenz oder zum Predigen. Allerdingst wurden kein projektspezifisches Kriterium zur Einstufung eines Wirkungsort und Wohnsitzes definiert (z.B. Mindestdauer oder Mindestzahl von Ereignissen). Als weitere soll der Kontext bzw. das Ziel von Reise, Besuch und Aufenthalt (Fremdsprache lernen, eine Kirche aufbauen, Kinder aus Missionsstellen nach Europa mitnehmen usw.) für die jeweiligen Personen präzisiert werden. Diese Beispiele verdeutlichen nur einen Teil der Herausforderungen bei der Datenmodellierung.

## SPARQL

Laube-Artikel

## Visualisierung

Dies wurde mithilfe spezieller Web-Frameworks erreicht, die es ermöglichen, Python effektiv für die Webentwicklung zu nutzen.

Diese Frameworks lassen sich in zwei Typen unterteilen: Full-Stack und Non-Full-Stack. Erstere bieten eine umfassende Lösung für alle technischen Anforderungen für die Webentwicklung (z. B. Django, Pyromid), während letztere nur Grundfunktionen wie das Weiterleiten von HTTP-Anfragen an die relevanten Controller ermöglichen. Non-Full-Stack benötigen in der Regel Unterstützung durch weitere Application Programming Interfaces (APIs) und Tools. Da die älteren Frameworks wie *Flask* und *Django* die Kenntnisse in HTML und CSS erfordern (vgl. Khorasani 2022: [Getting Started with Streamlit](https://learning.oreilly.com/library/view/web-application-development/9781484281116/html/523242_1_En_1_Chapter.xhtml)), entschieden wir uns gegen deren Verwendung und wählten stattdessen das Framework *Streamlit*.

*Streamlit* ist ein Open-Source-Python-Framework, das im Oktober 2019 veröffentlicht wurde. Es wurde entwickelt, um Datenwissenschaftler:innen und Ingenieur:innen im Bereich maschinelles Lernen zu ermöglichen, Daten ohne Vorkenntnisse in der Webentwicklung direkt in einer Benutzeroberfläche zu präsentieren. Benutzer:innen können mit der Anwendung interagieren, indem sie Daten ändern oder hochladen, um sie zu visualisieren oder unterschiedliche Modelle anzuwenden (vgl. Raghavendra 2023: [Introduction to Streamlit](https://learning.oreilly.com/library/view/beginners-guide-to/9781484289839/html/532543_1_En_1_Chapter.xhtml)). Die Entwicklung erfolgt ausschließlich in Python und die Entwicklungszeit wird deutlich verkürzt, da die Änderungen am Webdokument direkt am Document Object Model (DOM)[[1]](#footnote-1) vorgenommen werden. Dadurch werden Seitenaktualisierungen für die Nutzer:innen in wenigen Sekunden sichtbar (vgl. Khorasani 2022: [Getting Started with Streamlit](https://learning.oreilly.com/library/view/web-application-development/9781484281116/html/523242_1_En_1_Chapter.xhtml)).

Mit *Streamlit* können sich Entwickler:innen vollständig auf die Implementierung der Backend-Logik konzentrieren, während die Frontend-Umsetzung weitgehend vom Framework übernommen wird. Die persönliche Gestaltung einer Webseite wird mithilfe von Sidebar, Spalten, Container usw. erfüllt. Zudem können dynamische Inhalte mithilfe von Platzhaltern angezeigt werden. Darüber hinaus können mit *Streamlit* beliebig viele Seiten und verschachtelte Unterseiten erstellt werden (vgl. Khorasani 2022: [Archietecting the User Interface](https://learning.oreilly.com/library/view/web-application-development/9781484281116/html/523242_1_En_3_Chapter.xhtml)).

Der erste Schritt zu der Konfiguration der Webseite erfolgte mithilfe der Python-Bibliothek *Streamlit*. Mit dem Befehl st.set\_page\_config() wurden der Titel und das Layout definiert. Bei der Erstellung der Webseite war es wichtig, eine durchdachte Struktur zu entwickeln. Um die interaktive digitale Darstellung des Friedhofs von der Analyse zu trennen, wurden zwei separate Seiten konzipiert: *Digitaler Gottesacker* und *Analyse*. Dies wurde durch Konfiguration einer Sidebar (st.sidebar) im linken Bereich der Webseite ermöglicht, was eine intuitive Navigation zwischen den beiden Seiten erlaubt.

Netzwerkanalyse biete neue Möglichkeit zur Visualisierung und Analyse von Sammlungswissen. Ihre zentralen Ziele sind die Identifikation wichtiger Verbindungen und Strukturen. (vgl. Werner 2020: 248–249).

Für geistes- und sozialwissenschaftliche Forschung ist die Verbindung zwischen Netzwerken und Visualisierung von großer Bedeutung. Die graphische Visualisierung von abstrakten oder komplexen Strukturen macht die Zusammenhänge, die sich auf rein sprachlichem Wege nur schwer oder unzureichend vermitteln lassen, greifbar. Digitale Werkzeuge bieten hier neue Möglichkeiten, große Datenbestände nicht nur zu verarbeiten, sondern auch visuell aussagekräftige Formen überführen können. Daraus ergibt sich einen Anknüpfungspunk für die Verwendung netzwerkbasierter Modelle. Für die biographische Forschung eröffnet der Netzwerkansatz die Perspektive, individuelle Lebensläufe nicht isoliert zu betrachten, sondern als Teil größer Beziehungsgeflechte (vgl. Dörpinghaus 2022).

## Auswertung / Perspektive

**OCR**: Zur Verbesserung der Ergebnisse könnte mehrstufige Vorgehensweise: Mehrere OCR-Durchläufe mit variierenden Parametern von Schwellenwerten bspw. und ggf. deren spätere Konsolidierung. Auf diese Weise lassen sich Fehlertypen gegenseitig kompensieren. Darüber hinaus konnte auch adaptive Vorverarbeitung helfen. Man könnte die Bilder sauber/ verschmutz, hell/dunkel annotieren und pro Klasse geeignete Parameter wählen. Als Nachkorrektur kann ein LLM eingesetzt werden.

**XML**: mehr Attributen zu definieren, Versionierung zeigen, Personen und Körperschaft gleich mit Wikidata Normdaten kodieren, Datumsangaben zu normieren,

**Relation**: viele Information ist verloren gegangen, z.B. Zweck der Reise (Fremdsprache lernen, eine neue Kirche bauen oder einen passenden Ort für weitere Mission zu finden)

**Programmieren mit ChatGPT**:

Im Rahmen dieser Arbeit wurde das Programmieren mit der Unterstützung von ChatGTP eingesetzt. Insgesamt erwiest sich dieser Ansatz als gut geeignet. Besonders hilfreich war die Möglichkeit, den eigenen Code überprüfen zu lassen. Außerdem zeigte sich das Modell nützlich bei der Beschreibung der benötigten Datenformat.

Herausfordernd war hingegen die Arbeit mir komplexeren Problemen. ChatGPT lieferte selten sofort eine vollständige und funktionierende Lösung für größere Aufgaben. Problematisch war zudem, dass der generierte Code bei Nachfragen oder Fehlerbehebungen teilweise vollständig umgeschrieben wurde. Dabei kam es vor, dass alternative Python-Bibliotheken verwendet oder Variablennamen verändert wurden. Dies erschwerte die Nachvollziehbarkeit und Konsistenz des Codes. Dieses Problem wurde aber durch eine Anpassung des Prompts gelöst, indem bspw. bestimmte Bibliotheken festgelegt wurden.

Insgesamt zeigt die Bewertung, dass ChatGPT als nützliches Werkzeug zur Code-Erstellung und -Optimierung eingesetzt werden kann, gleichzeitig aber ein kritisches Nacharbeiten und eigenständiges Verständnis des Codes unverzichtbar bleibt.

## Zusammenfassung

## Literaturverzeichnis

## Anhang

1. DOM ist eine Programmier-Schnittstelle für HTML- und XML-Dokumente, die die logische Struktur und den Zugriff auf Dokumente definiert. Damit kann mit einigen Ausnahmen auf alle Elemente eines HTML- oder XML-Dokuments zugegriffen und diese verändert, entfernt oder ergänzt werden (vgl. World Wide Web Consortium (W3C): [What ist he Document Object Model?](https://www.w3.org/TR/WD-DOM/introduction.html)). [↑](#footnote-ref-1)