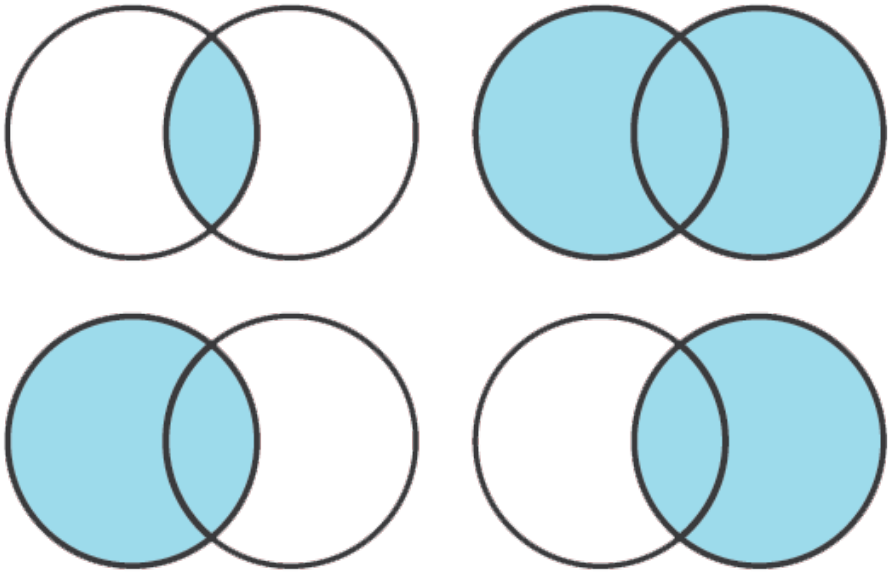
Verknüpfte Datenbanken

…das war noch nicht alles?!? 😱

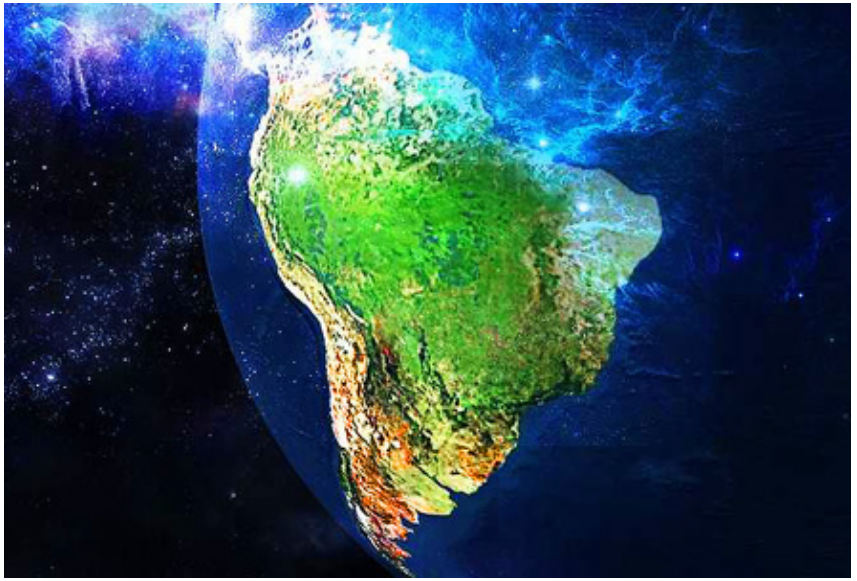


Quelle: https://images.app.goo.gl/AAD2Vade5SBnWscu5

TERRA-Datenbank

Quelle: https://goo.gl/images/pBfP5d

- Unterrichtsdatenbank für Geoinformationen am Sächsischen Bildungsserver (SBS)
- TERRA-Datenbank stammt ursprünglich aus: *Dürr, M., Radermacher, K.: "Einsatz von Datenbanksystemen - Ein Leitfaden für die Praxis". Berlin u.a.: Springer-Verlag 1990.*
- Ausschließlich für Informatik-Lehrzwecke → nicht für Geographie geeignet 😉
- Kein Anspruch auf Vollständigkeit und Korrektheit
- Datenbestand wird in unregelmäßigen Abständen von gröbsten Fehlern befreit



Quelle: https://goo.gl/images/cgYgSH

Den richtigen Datensatz finden… 🧐

- Daten in unterschiedlichen Tabellen hängen oft miteinander zusammen
- Tabellen alleine reichen nicht aus, um die Struktur von Daten zu erfassen

- Suche nach „Washington“
  - Mehrere nicht eindeutige Ergebnisse
  - SELECT Befehl kann nur sehr kompliziert und mit vielen Attributen gebaut werden

Name	Land	Einwohner	Laenge	Breite
Washington	Großbritannien	51843	-1.5300	54.9100
Washington	Großbritannien	1930	-0.4063	50.9030
Washington	Vereinigte Staaten von Amerika	13660	-80.2506	40.1750
Washington	Vereinigte Staaten von Amerika	9740	-77.0519	35.5536
Washington	Vereinigte Staaten von Amerika	4295	-82.7414	33.7353
Washington	Vereinigte Staaten von Amerika	11509	-87.1750	38.6583
Washington	Vereinigte Staaten von Amerika	25139	-83.0322	42.7525
Washington	Vereinigte Staaten von Amerika	1903	-88.4606	44.8047
Washington	Vereinigte Staaten von Amerika	18760	-113.5033	37.1194
Washington	Vereinigte Staaten von Amerika	4740	-73.6810	41.7867
Washington	Vereinigte Staaten von Amerika	13160	-89.4206	40.7039
Washington	Vereinigte Staaten von Amerika	13240	-91.0133	38.5519
Washington	Vereinigte Staaten von Amerika	7000	-91.4208	44.7700
Washington	Vereinigte Staaten von Amerika	5582170	-77.0200	38.9100

Quelle: https://bit.ly/2VEwylJ

Here‘s my key… 🔑

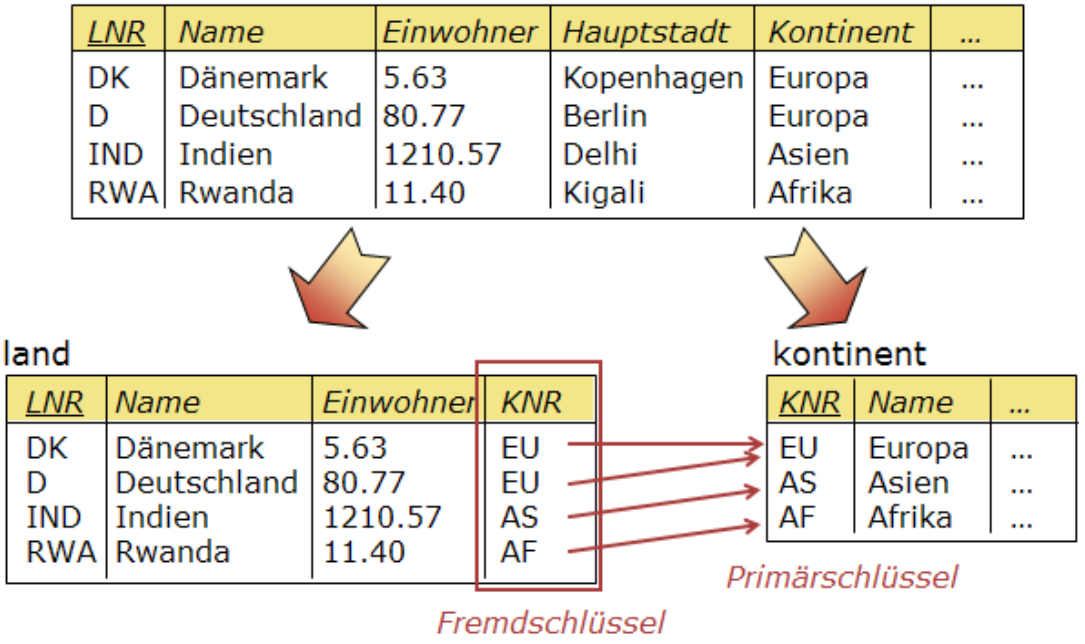
- Lösung → der **Primärschlüssel**

- Kombination von Attributen
  - Identifizierung eines Datensatzes wird dadurch eindeutig
- Schlüssel ist minimal
  - kein Attribut darf fehlen, um den Datensatz eindeutig zu finden
- Benötigt man mehrere Attribute (z.B. Name, Länge, Breite)
  - Erstellung eines **künstlichen Primärschlüssels** (z.B. Ländernummer in TERRA-Datenbank)

Quelle: https://bit.ly/2VEwylJ

Fremdschlüssel 🔑

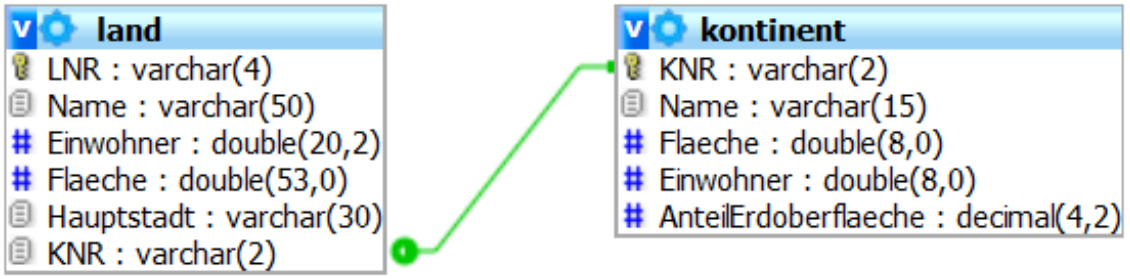
- Alles in einer Tabelle speichern → **nicht sinnvoll!**
- Mögliche Probleme:
  - Falsche Daten → **inkonsistente Daten**
  - Mehrfach gespeicherte Daten → **redundante Daten**
- Lösung:
  - Daten auf mehrere Tabellen aufteilen
  - Verweise zeigen auf "verwandte" Daten → **Fremdschlüssel**
    - Attribut einer Tabelle, dass auf den Primärschlüssel einer anderen Tabelle verweist



Quelle: https://bit.ly/2qdK8AR

Beziehungen unter Tabellen ❤️

- Fremdschlüssel verbindet zwei Tabellen miteinander → **Beziehung beider Tab.**
- Beziehungen sind zweit wichtigstes Konzept relationaler Datenbanken

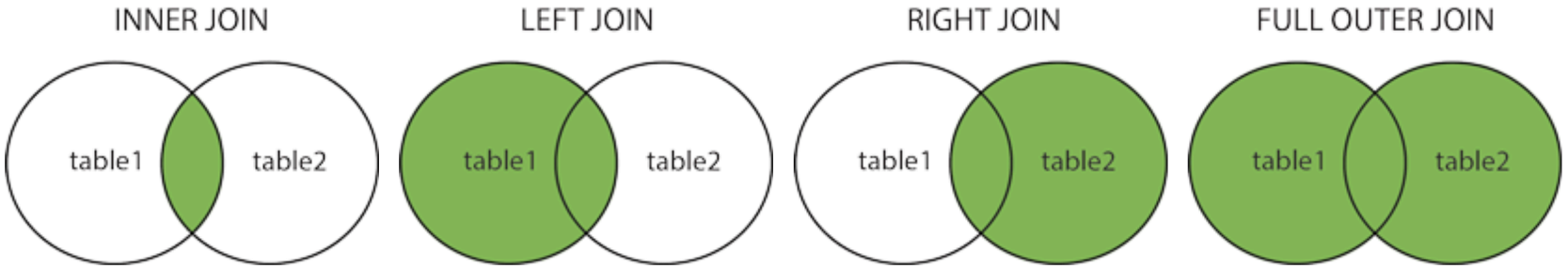


Quelle: https://bit.ly/2qdK8AR

- Relationales Datenbanksystem garantiert solche Beziehungen 😍
  - Werte nur dann in den Fremdschlüsseln erlaubt, wenn auch vorhanden in der Haupttabelle
  - **Beispiel:** kein Land mit der KNR "xy" speicherbar - Schlüsselwert existiert in kontinent-Tabelle nicht

SQL Joins

- Abfrage bzw. Ausgabe aus **mehreren Tabellen**



Quelle: https://bit.ly/2mYxnr4

- **(INNER) JOIN** : Liefert Datensätze, die in beiden Tabellen passende Werte haben
- **LEFT (OUTER) JOIN** : Liefert alle Datensätze aus der linken Tabelle, und passende Datensätze aus der rechten Tabelle
- **RIGHT (OUTER) JOIN** : Liefert alle Datensätze aus der rechten Tabelle, und passende Datensätze aus der linken Tabelle
- **FULL (OUTER) JOIN** : Liefert alle Datensätze, die eine Übereinstimmung in der linken oder rechten Tabelle haben

Wir basteln einen JOIN-Befehl… 🤓

- Bereits bekannter SQL-SELECT Syntax wird im WHERE Teil um ein JOIN erweitert

- **INNER JOIN:**

SELECT Bestellungen.BestellungID, Kunden.KundeName **FROM** Bestellungen
INNER JOIN Kunden **ON** Bestellungen.KundenID = Kunden.KundenID;

- **LEFT JOIN:**

SELECT Kunden.KundenName, Bestellungen.BestellungID **FROM** Kunden
LEFT JOIN Bestellungen **ON** Kunden.KundenID = Bestellungen.KundenID;