LU01c - DBs Einsetzen & Relationale Datenbanken

Datenbanken einsetzen

Datenbanken haben sich als fester Bestandteil von IT-Landschaften etabliert. Sie spielen eine wesentliche Rolle, wo immer Software-Applikationen zum Einsatz kommen und auf eine gemeinsame, integrierte Datenbasis zugreifen.

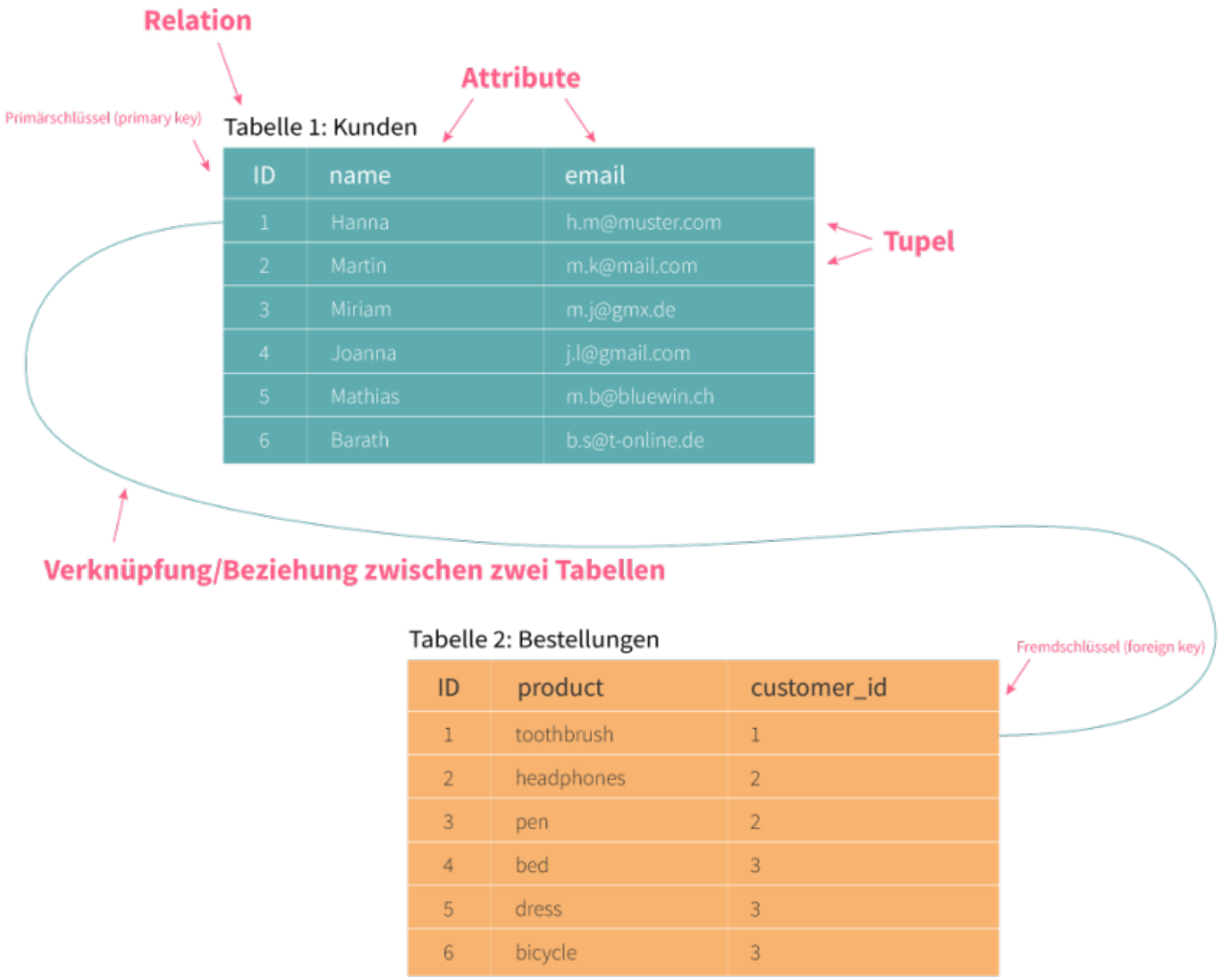
Vorteile beim Einsatz von Datenbanken

- Eine Datenbank ermöglicht die **dauerhafte, zentrale Speicherung von Daten** und kann diese eigenständig verwalten.
- Der Einsatz von Datenbanken **verhindert Redundanzen** [1)] und Inkonsistenzen [2)]. Zudem wird die **Programm-Daten-Abhängigkeit** [3)] aufgelöst, da alle Programme eine zentrale Datenbasis verwenden.
- DBMS bieten eine Vielzahl von **Datenanalyse-Methoden** zur Auswertung von Informationen. Die SQL-Skriptsprache stellt Funktionen zur Berechnung, Aggregation und Sortierung der Daten zur Verfügung.
- Datenbankmanagementsysteme (DBMS) sind **abfrageoptimiert** und können Daten in einem Bruchteil der Zeit auslesen, die andere Dateiformate (z.B. Excel, CSV) benötigen.

Relationale Datenbanken (Relational Databases)

Relationale Datenbanksysteme (RDBMS, *Relational Database Management Systems*) gibt es seit den 1980er Jahren. Sie haben sich zum De-facto-Standard für die Verwaltung strukturierter Daten entwickelt und sind heute die am weitesten verbreitete Form von Datenbanksystemen.

Relationale Datenbanken speichern Informationen in **Tabellen** (tables). Tabellen bestehen aus **Zeilen** (rows, auch **Tupel/tuples** genannt) und **Spalten** (columns, auch **Attribute/attributes** genannt). - Eine **Zeile** (Tupel) steht für einen vollständigen Datensatz, zum Beispiel einen Kunden oder eine Bestellung. - Eine **Spalte** (Attribut) beschreibt eine bestimmte Eigenschaft dieses Datensatzes, zum Beispiel den Namen oder die E-Mail-Adresse eines Kunden. - Ein einzelner Eintrag in einer Zelle ist ein **Attributwert** (attribute value).



Beispiel einer Kundentabelle mit verknüpfter Bestelltabelle: - Jede **Zeile** repräsentiert einen Kunden. - Die **Spalten** enthalten Merkmale wie Name und E-Mail-Adresse.

In der zweiten Tabelle **Bestellungen**, repräsentiert jede Zeile eine Bestellung mit Attributen wie das bestellte Produkt oder wer die Bestellung getätigt hat (via Fremdschlüssel → Erklärung dazu, weiter unten).

Was bedeutet „Relation“?

Der Begriff **Relation** stammt aus der Mathematik und ist die Grundlage relationaler Datenbanken. Eine Relation ist nichts anderes als eine Tabelle mit Daten einer bestimmten Art (z.B. Kunden, Bestellungen). Der Name „relational“ ergibt sich daraus, dass Tabellen miteinander in **Beziehung** (relation) gesetzt werden können.

Wie sind Tabellen miteinander verbunden?

Tabellen können über **Schlüssel** verknüpft werden: - Ein **Primärschlüssel** (primary key) identifiziert

jede Zeile in einer Tabelle eindeutig. - Ein **Fremdschlüssel** (foreign key) verweist auf den Primärschlüssel einer anderen Tabelle.

Beispiel: - In der **Kundentabelle** gibt es für jeden Kunden eine eindeutige **Kundennummer (ID)** (Primärschlüssel). - In der **Bestellungstabelle** wird dieselbe Kundennummer als **Fremdschlüssel (customer_id)** gespeichert. So lässt sich jede Bestellung eindeutig einem Kunden zuordnen.

Bekannte relationale Datenbanksysteme (RDBMS)

- **MySQL**
- **MariaDB**
- **PostgreSQL**
- **SQLite**
- IBM DB2
- Oracle Database
- Microsoft SQL Server

MySQL, MariaDB, PostgreSQL und SQLite sind frei nutzbare, Open-Source-Datenbanksysteme. Besonders MySQL (und MariaDB, das ein Fork von MySQL ist) sind weit verbreitet und kommen häufig in Webanwendungen zum Einsatz. Beispielsweise nutzt WordPress MySQL, was bedeutet, dass über 40% aller Websites weltweit ihre Daten über MySQL oder MariaDB verwalten.

SQL

Relationale Datenbanken nutzen die **Structured Query Language (SQL)**, um Daten abzufragen (query), zu modifizieren (manipulate), zu löschen (delete) und zu verwalten. SQL ist eine standardisierte Programmiersprache, die speziell für die Verwaltung relationaler Daten entwickelt wurde. Dabei haben verschiedene Datenbankanbieter oft eigene Erweiterungen oder Dialekte von SQL entwickelt, wie z.B. T-SQL (Transact-SQL) von Microsoft oder PL/SQL (Procedural Language/SQL) von Oracle.

Video: Relationale und NoSQL-Datenbanken

Dieses Video erklärt den Aufbau von relationalen Datenbanken. [4)]
Nutzen Sie die automatische Untertitel-Übersetzung von YouTube, wenn die englische Sprache Verständnisschwierigkeiten bereitet.

NoSQL-Datenbanken zur Speicherung grosser/hochfrequenter Datenmengen

NoSQL-Datenbanken sind eine jüngere Technologie und werden vor allem verwendet, wenn Daten in einer **unstrukturierten oder wenig strukturierten Form** abgelegt werden müssen. NoSQL-Datenbanken wurden besonders im Zusammenhang mit dem Big Data Hype populär. Sie ermöglichen

es, **grosse Datenmengen schnell zu speichern und auszuwerten**, wobei Konsistenz und Aktualität der Daten weniger im Fokus stehen. Die Hauptaufgabe dieser Datenbanken ist es, grosse und hochfrequente Daten effizient zu verarbeiten.

In diesem Video wird erklärt, was mit Big Data gemeint ist.

Bekannte NoSQL-Datenbanksysteme

- MongoDB
- Redis
- Apache Cassandra

Vergleich von Relationalen und NoSQL-Datenbanken

- Relationale Datenbanken wurden ursprünglich aufgrund von **Speicherknappheit** und teurer Speichermedien zum Standard. Mit der Entwicklung effizienterer Speichertechnologien sind diese Probleme heute nicht mehr so gravierend und Speicherplatz ist relativ günstig.
- Relationale Datenbanken sind aufgrund von Integritätsprüfungen und Normalisierungen **nicht für die schnelle Speicherung grosser Datenmengen** geeignet. Für diese Aufgaben bieten **NoSQL-Datenbanken** eine bessere Lösung, da sie auf eine schnelle Verarbeitung ohne hohe Integritätsanforderungen optimiert sind.
- Relationale Datenbanken sind nach wie vor **unverzichtbar**, wenn es auf **Genauigkeit, Integrität, Aktualität** und **Nachvollziehbarkeit** ankommt. Typische Anwendungsfälle sind **Transaktionssysteme** wie Webshops, Online-Banking und Buchungssysteme.

Quellen:

- Datenbanken verstehen
- Datenbanken Kompaktkurs

1)
Ohne Datenbank kommt es schnell zu mehrfachen Datensätzen, z. B. ein Kunde in mehreren Excel-Listen unterschiedlicher Abteilungen. Mit einer zentralen Datenbank wird jeder Kunde nur einmal gespeichert – alle greifen auf denselben Datensatz zu.
2)
Wenn dieselbe Information mehrfach gespeichert wird, können Abweichungen entstehen – z. B. wird die Adresse eines Kunden nur in einer Datei geändert, aber nicht in der anderen. Mit einer zentralen Datenbank gibt es nur eine Datenquelle und damit keine widersprüchlichen Angaben.
3)
Früher hatten Programme eigene Datenbestände, die gepflegt werden mussten. Änderungen in der Datenstruktur erforderten oft Programmänderungen. Mit einem Datenbankmanagementsystem (DBMS) werden die Daten getrennt von der Logik verwaltet. Mehrere Programme können so dieselben Daten nutzen, ohne sie selbst speichern zu müssen.
4)
IBM Technology: Relational vs. Non-Relational Databases

From:
https://wiki.bzz.ch/ - **BZZ - Modulwiki**

Permanent link:
https://wiki.bzz.ch/modul/m290_guko/learningunits/lu01/theorie/c_relational_dbs

Last update: **2025/08/17 20:30**