M164_Datenbanken erstellen und Daten einfügen

Tag 2

Recap

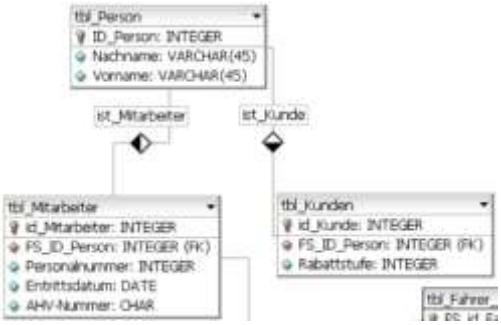
Was sind Entitäten? Beispiele?

Es sind die zentralen Objekte, über die Informationen gespeichert werden. zB Person, Kunde oder Mitarbeiter sind Objekte.

->Attribute sind dann zB. Name, Vorname, Personalnummer, ID_Person(FK)

Generalisierung/ Spezialisierung

Wenn mehrere Entitätstypen gemeinsame Attribute besitzen, aber diese Entitäten getrennt modelliert werden, entstehen Redundanzen. (zur Vorstellung: Prinzip Vererbung)



Generalisierung: Gemeinsame Attribute verschiedener Entitätstypen werden in einer **übergeordneten Entität** zusammengeführt. übergeordnete Entität (Objekt-> Parentobjekt). Dies **reduziert Datenredundanz** und **verbessert die Konsistenz**. *(Parenobjekt->tbl_Person)*

Spezialisierung: Sie werden zB mit einem **FK** auf den **Parentobjekt verwiesen**. Zudem mit eigenen **Attributen erweitert**. Über eine **«is_a» Beziehung** mit der Parentobjekt verbunden. *(untergeordnete Entitätstypen/Childobjekt ->tbl_Mitarbeiter und tbl_Kunden)*

Weiteres Bsp: Fahrzeug mit gemeinsamen Attributen: Hersteller, Modell, Baujahr (Generalisiert), Auto oder Motorrad mit zusätzlichen Attributen je anders: Türen, Kraftstoff (Spezialisiert)

Beziehungsarten: Indentifying / Non-Identifying Relationship

Child Tabelle (hat PK), Parent Tabelle (hat Foreign Key).

Identifying Relationship

Beispiel: Person und Ausweis (Ausweis kann nicht ohne Person existieren). Stellt sicher das die Abhängigkeit besteht. Gewährleistet Datenintegrität, vermeiden Redundanzen

- Der Fremdschlüssel (FK) ist Teil des Primärschlüssels (PK) der Kind-Tabelle.
- Der Datensatz in der Kind-Tabelle kann nicht ohne die Master-Tabelle existieren.
- Starke Abhängigkeit zwischen den Tabellen.
- wird oft mit einem durchgezogenen Linien-Diamanten dargestellt.

```
CREATE TABLE Gebäude (
  ID_Gebäude INT PRIMARY KEY AUTO_INCREMENT,
  Name VARCHAR(50) NOT NULL,
  Adresse VARCHAR(100) NOT NULL
);

CREATE TABLE Raum (
  ID_Raum INT NOT NULL,
  FK_ID_Gebäude INT NOT NULL,
  Bezeichnung VARCHAR(30) NOT NULL,
  PRIMARY KEY (ID_Raum, FK_ID_Gebäude),  -- PK besteht aus zwei Spalten (Identifying Relationship)
  FOREIGN KEY (FK_ID_Gebäude) REFERENCES Gebäude(ID_Gebäude) ON DELETE CASCADE
);
```

Non-Identifying Relationship

Beispiel: Person kann mehrere Kleidungsstücke besitzen. Kleidungsstück kann aber theoretisch auch einer anderen Person zugeordnet werden.

- Der Fremdschlüssel (FK) ist NICHT Teil des Primärschlüssels (PK) der Kind-Tabelle.
- Die Kind-Tabelle kann unabhängig von der Master-Tabelle existieren.
- Schwächere Abhängigkeit.
- wird oft mit einer gestrichelten Linien-Diamanten dargestellt.

DBMS (Datenbankmanagementsystem)

Folgendes muss erhalten werden (einige von vielen):

Integrierte Datenhaltung

- Alle Daten werden einheitlich verwaltet und an einer zentralen Stelle gespeichert.
- Beziehungen zwischen Daten können effizient verknüpft werden.
- Kontrollierte Redundanz kann zur Leistungssteigerung genutzt werden.

Datenbanksprache (Query Language)

- DQL (Data Query Language) → SELECT (Abfragen von Daten)
- DDL (Data Definition Language) → CREATE, ALTER, DROP (Tabellenstruktur verwalten)
- DML (Data Manipulation Language) → INSERT, UPDATE, DELETE (Daten ändern)
- DCL (Data Control Language) → GRANT, REVOKE (Zugriffsrechte verwalten)
- TCL (Transaction Control Language) → COMMIT, ROLLBACK (Transaktionen steuern)

Katalog (Data Dictionary)

- Speichert Metadaten (Daten über die Datenbankstruktur).
- Erlaubt die Verwaltung von Tabellen, Sichten, Indizes usw.

Benutzersichten

- Verschiedene Benutzer haben unterschiedliche Sichten (Views) auf die Daten.
- Beispiel: Ein Buchhalter sieht nur Finanzdaten, während ein Lagerarbeiter Bestandsdaten sieht.

Transaktionsmanagement

- **A**tomarität: Ganz oder gar nicht.
- **C**onsistency: Daten bleiben konsistent.
- **I**solation: Keine gegenseitige Beeinflussung.
- **D**auerhaftigkeit: Änderungen sind permanent.

Nachteile von Datenbanksystemen

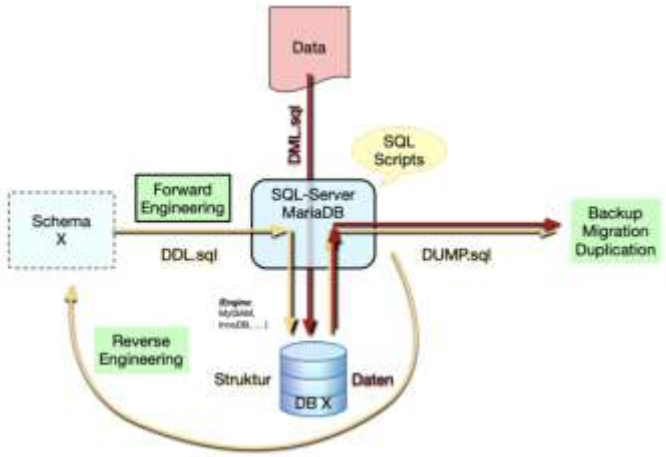
Trotz der Vorteile eines DBMS gibt es auch Situationen, in denen ein solches System unnötig hohe Zusatzkosten im Vergleich zur traditionellen Dateiverarbeitung mit sich bringen würde:

- Hohe Anfangsinvestitionen für Hardware und Datenbanksoftware.
- Ein DBMS ist Allzweck-Software, somit weniger effizient für spezialisierte Anwendungen.
- Bei konkurrierenden Anforderungen kann das Datenbanksystem nur für einen Teil der Anwendungsprogramme optimiert werden.
- Mehrkosten für die Bereitstellung von Datensicherheit, Mehrbenutzer- Synchronisation und Konsistenzkontrolle.
- Hochqualifiziertes Personal erforderlich, z. B. Datenbankdesigner, Datenbankadministrator.
- Verwundbarkeit durch Zentralisierung (Ausweg: Verteilung).

Die wichtigsten 10 DB-Engines:

Rank			DBMS	Database Model	Score		
Mar 2025	Feb 2025	Mar 2024			Mar 2025	Feb 2025	Mar 2024
1.	1.	1.	Oracle	Relational, Multi-model	1253.08	-1.74	+32.02
2.	2.	2.	MySQL	Relational, Multi-model	988.13	-11.86	-113.37
3.	3.	3.	Microsoft SQL Server	Relational, Multi-model	788.14	+1.27	-57.67
4.	4.	4.	PostgreSQL	Relational, Multi-model	663.42	+3.81	+28.52
5.	5.	5.	MongoDB	Document, Multi-model	396.42	-0.21	-28.11
6.	7.	9.	Snowflake	Relational	161.78	+6.20	+36.40
7.	6.	6.	Redis	Key-value, Multi-model	155.36	-2.55	-1.64
8.	8.	7.	Elasticsearch	Multi-model	131.38	-3.25	-3.41
9.	9.	8.	IBM Db2	Relational, Multi-model	126.57	+1.14	-1.18
10.	10.	10.	SQLite	Relational	113.08	-0.74	-5.08

Forward Engineering



Der Prozess, bei dem man aus einem vorher erstellten Datenmodell automatisch eine funktionierende Datenbank erstellt. Das Datenmodell zeigt, wie die Datenbank aussehen soll, mit Tabellen, Spalten und Verbindungen zwischen den Daten.

Statt die Datenbank manuell zu bauen, nimmt man also das Modell und lässt ein Computerprogramm daraus die eigentliche Datenbank erzeugen. Das hat den Vorteil, dass der Aufbau schneller und fehlerfreier ist, weil Probleme schon im Modell erkannt werden können, bevor die Datenbank tatsächlich verwendet wird.

DDL-Intro (Data Defintion Language)

- CREATE DATABASE / SCHEMA
- CREATE TABLE
- DROP TABLE
- ALTER TABLE

CREATE SCHEMA erstellt also eine leere Datenbank, in der Tabellen erstellt und Daten gespeichert werden können. CREATE SCHEMA in SQL synonym zu CREATE DATABASE

```
CREATE SCHEMA db_name;
```

Folgend noch mit Standard Charset und Kollation. (Hinweis: Kollation = Art wie sortiert wird. _ci = case insensitive, _cs = case sensitive)

```
CREATE SCHEMA schema_name
 DEFAULT CHARACTER SET utf8mb4
 DEFAULT COLLATE utf8mb4_general_ci;
```

CREATE TABLE

```
CREATE TABLE customers (
  id INT NOT NULL AUTO_INCREMENT,
  name VARCHAR(50) NOT NULL,
  email VARCHAR(100) NOT NULL UNIQUE,
  age INT,
  PRIMARY KEY (id)
);
```

DROP TABLE

```
DROP TABLE table_name;
```

um sicherzustellen, dass SQL keine Fehlermeldung zurückgibt, wenn die Tabelle, die Sie löschen möchten, nicht existiert:

```
DROP TABLE IF EXISTS table_name;
```

ALTER TABLE

```
ALTER TABLE table_name ADD column_name column_definition;
```

```
ALTER TABLE table_name RENAME [TO] new_table_name;
```

```
ALTER TABLE table_name RENAME COLUMN old_column_name TO new_column_name;
```

```
ALTER TABLE table_name DROP column_name;
```