Tutorium 2

When you see a well designed database schema

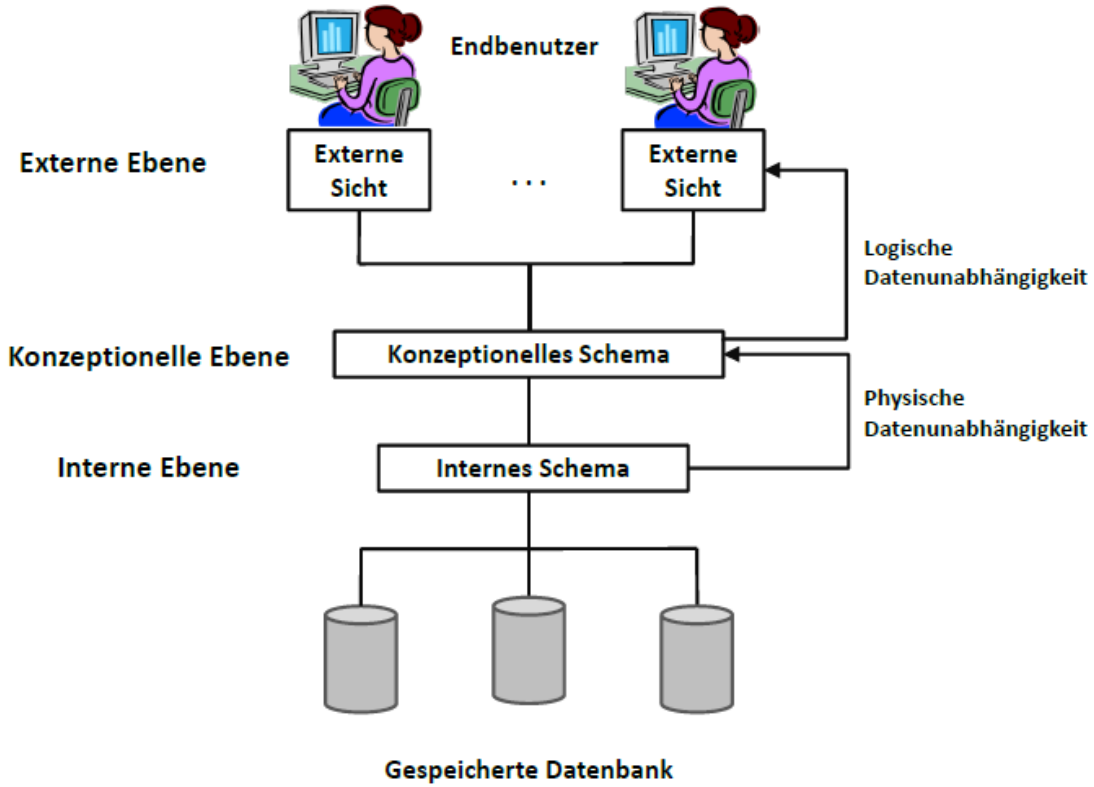
9.11.2022 – Finn Kapitza

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

1. Wiederholung - Datenbanksysteme

Datenbanksysteme

- Persistente Speicherung großer Mengen von Daten
- Gleichzeitiger Zugriff/Änderung des Datenbestandes von mehreren Personen



LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Aufgabe 2.1 – Datenbanksysteme vs. Dateinverwaltungssysteme

Aufgabe 2-1 *Datenbanksysteme – Dateiverwaltungssysteme*

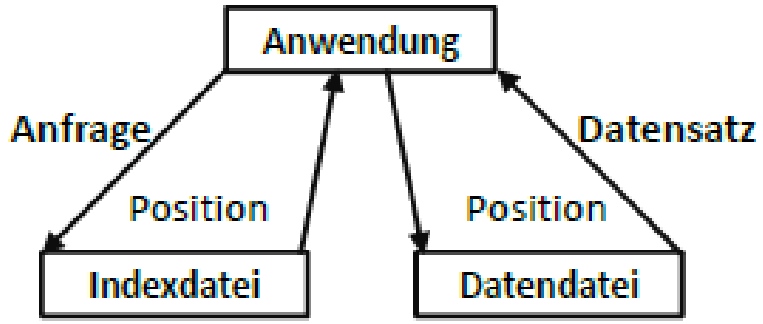
Erläutern Sie die Vorteile, die Datenbanksysteme gegenüber Dateiverwaltungssystemen durch die 3-Ebenen-Architektur (Externe, Interne, Konzeptionelle Ebene) besitzen. Insbesondere soll dabei auf folgende Punkte eingegangen werden:

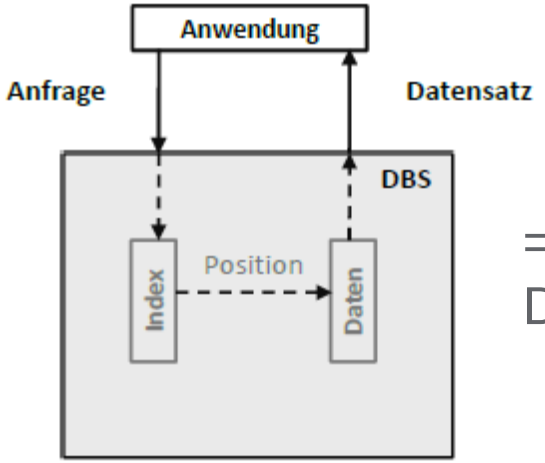
Aufgabe 2.1.a – Erweiterung der abgespeicherten Daten um ein Attribut

Dateiverwaltungssystem	Datenbanksystem
Änderung der Recordstrukturen der Dateien -> alle betroffenen Anwendungen (die auf die geänderten Dateien zugreifen) müssen geändert werden -> Zeitaufwendig -> Änderungen nur mit Ankündigung möglich	Änderung des konzeptionellen Schemas und der internen Ebene -> externe Sichten können meist unverändert bleiben (evtl. Müssen einige Benutzersichten verändert werden) -> Benutzersichten können im Laufe der Zeit einfach angepasst werden -> Änderungen können spontan und ohne Wissen der Anwender passieren

Aufgabe 2.1.b – Anlegen eines Index zum Schnelleren Zugriff auf die Datensätze

Dateiverwaltungssystem	Datenbanksystem
Zusätzliche Indexdatei muss erstellt und gewartet werden -> Änderung aller Anwendungsprogramme die diesen Index nutzen wollen	Zusätzliche Indexstruktur auf der Internen Ebene -> Anfragebearbeitung wird automatisch von Datenbankmanagementsystem gesteuert uns so fallen keine Änderungen für die Anwendungsprogramme an (aber dennoch schnellere Zugriffzeiten)





=> Physische Datenunabhängigkeit

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Aufgabe 2.1 – Weitere Vorteile

- Verminderte Redundanz
- Einhaltung von Datenintegrität
- Verbesserter Datenschutz
- Erleichterung von Standardisierungen

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

2. Wiederholung - Anomalien

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Anomalien in Datenbanken

- **Redundanz:**
  - Daten werden öfter gespeichert als notwendig
- **Änderungsanomalie:**
  - Bei Änderung eines Datensatzes müssen alle Zeilen geändert werden -> wenn eine Zeile vergessen wird ist die Integrität verletzt
- **Entfernungsanomalie:**
  - Beim Löschen einer Zeile können Informationen gelöscht werden die gar nicht gelöscht werden sollten
- **Einfügeanomalie:**
  - Beim Einfügen muss man immer eine ganze Zeile einfügen (keine partielle Einfügung möglich)

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

3. Wiederholung – Relationales Modell

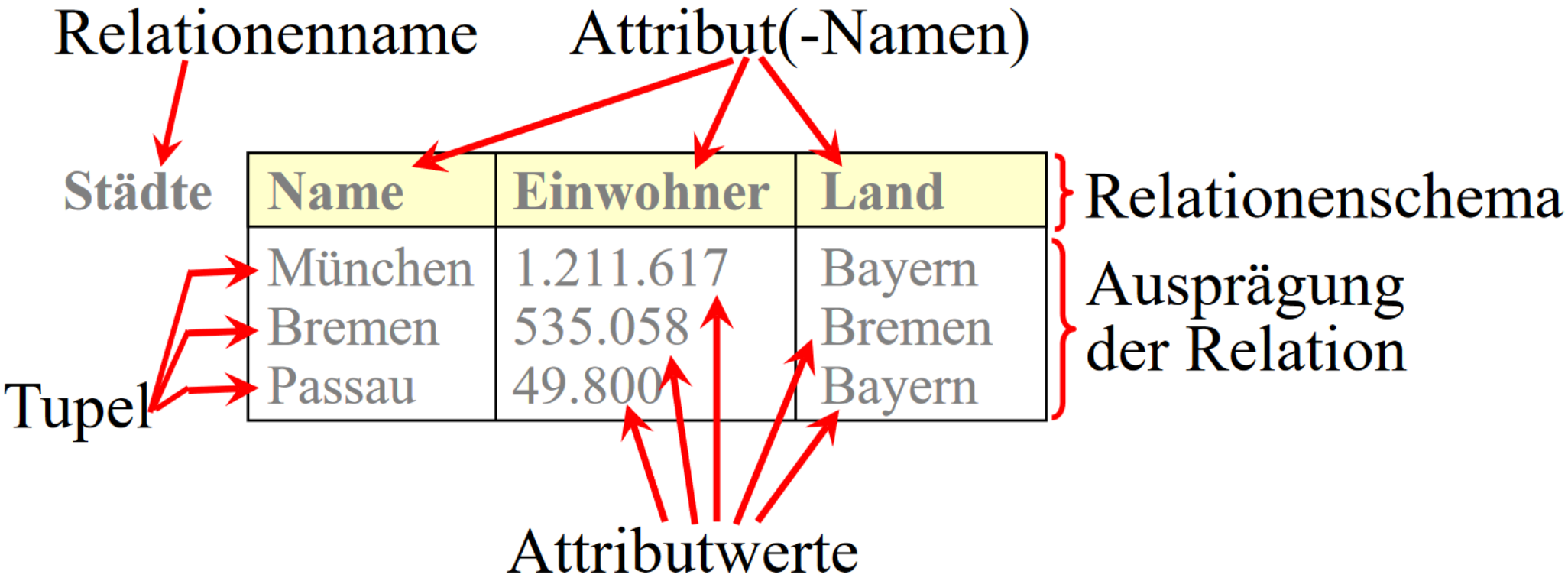
LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Das Relationale Modell

1. Geordnetes Relationenschema:
   - k-Tupel aus Domains (Attribute)
   - Attribute werden anhand ihrer Position im Tupel referenziert
   - Attribute können auch zusätzlich einen Attributnamen haben
   - $R = (A_1 : D_1, \ldots A_k : D_k)$ -> Relationenschema
   - **Relation:** Ausprägung eines Relationenschemas
   - **Datenbankschema:** Menge von Relationschemata
   - **Datenbank:** Menge von Relationen (Ausprägungen)

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Begriffe im Relationalen Modell



LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Definition Schlüssel

Teilmenge **S** der Attribute eines Relationenschemas **R** heißt Schlüssel, falls gilt

1. Eindeutigkeit: keine zwei Tupel dürfen sich in allen Attributen von **S** gleichen

2. Minimalität: keine echte Teilmenge von **S** erfüllt die Eindeutigkeit

Trivial: Schlüssel mit nur einem Attribut sind immer minimal

Wenn zusammengesetzt: Prüfe jede Teilmenge von **S** auf Eindeutigkeit

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Aufgabe 2.2 – Relationales Datenmodell

Ein Computerspielegeschäft spezialisiert auf Klassiker bietet Spiele verschiedener Studios zu bestimmten Preisen an. Zu jedem Studio wird dabei die Mitarbeiterzahl gespeichert (wieso weiß nur der Geschäftsinhaber). Jedes angebotene Spiel hat ein Erscheinungsdatum und wurde von genau einem Studio herausgegeben. Die Informationen über die vorhandenen Spiele werden in einer Tabelle mit den Attributen **Studio**, **Mitarbeiteranzahl**, **Spiel**,**Erscheinungsdatum** und **Preis** gespeichert. Die Tabelle habe folgenden Inhalt:

Studio	Mitarbeiteranzahl	Spiel	Erscheinungsdatum	Preis
LucasArts	45	The Secret of Monkey Island	1990	49,99
Atari	3	E.T. the Extra-Terrestrial	1982	179,99
LucasArts	45	Sam& Max Hit the Road	1993	39,95
Nintendo	72	Super Mario Bros.	1985	45,99
LucasArts	45	Day of the Tentacle	1993	29,99
Nintendo	72	Super Metroid	1985	45,99

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Aufgabe 2.2.a – Welcher Nachteil ergibt sich, wenn die Tabelle nach obigen Schema gespeichert werden?

- Für jedes Spiel ist eigentlich nur das Studio, der Titel, das Erscheinungsjahr und der Preis wichtig
- ABER: es wird zusätzlich noch die Mitarbeiterzahl des Studios gespeichert
- In einigen Fällen (z.B. LucasArts) ist die Mitarbeiterzahl 3 mal gespeichert

-> Redundanz

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Aufgabe 2.2.b – LucasArts stellt einen neuen Mitarbeiter ein. Der Geschäftsführer besteht darauf das diese wichtige Information aktualisiert wird. Was ist zu beachten? Welches Problem tritt auf?

- Das Geschäft bietet 3 Spiele von LucasArts an

-> in 3 Tupeln muss die Mitarbeiterzahl erhöht werden

- Wenn ein Tupel vergessen wurde, erhält man einen inkonsistenten Datenbankzustand

-> Änderungsanomalie

Aufgabe 2.2.b – Niemand kauft das Spiel E.T. und der Laden nimmt es daher nach einer Zeit aus seinem Sortiment. Die entsprechende Zeile wird daher aus der Tabelle entfernt. Welcher Nachteil entsteht?

- Das Spiel *E.T. the Extra-Terrestial* ist das einzige Spiel von Atari im Geschäft
- Beim Löschen der Zeile geht auch die Mitarbeiterzahl von Atari verloren
- Wenn ein neues Spiel von Atari angeboten wird, muss die Mitarbeiterzahl wieder eingefügt werden

-> Entfernungsanomalie

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Aufgabe 2.2.c – Welches Problem ergibt sich, wenn ein neues Studio inkl. Mitarbeiterzahl in die Tabelle aufgenommen werden soll, für das aber noch kein Spiel verkauft wird?

- Zum Einfügen eines neuen Studios (Name, Mitarbeiterzahl) wird in diesem Schema auch ein Spiel benötigt
- Man kann kein neues Studio ohne ein Spiel einfügen

-> Einfügeanomalie

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

2.2.d – Spalten sie die Tabelle in mindestens 2 Tabellen auf sodass die Probleme und Nachteile aus (a)-(d) vermieden werden. Kennzeichne die Schlüssel. Alle Namen sind eindeutig. Keine neuen Attribute.

Studios(Studio, Mitarbeiterzahl)

Spiele(Studio, Spiel, Erscheinungsjahr, Preis)

Studio	#Mitarbeiter
LucasArts	45
Atari	3
Nintendo	72

Studio	Spiel	E-Jahr	Preis
LucasArts	The Secret of Monkey Island	1990	49,99
Atari	E.T. the Extra-Terrestrial	1982	179,99
LucasArts	Sam& Max Hit the Road	1993	39,95
Nintendo	Super Mario Bros.	1985	45,99
LucasArts	Day of the Tentacle	1993	29,99
Nintendo	Super Metroid	1985	45,99

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Data Definition Language (DDL)

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Data Definition Language -> Anlegen von Tabellen

```
CREATE TABLE tabellenname (
    attribut1 datentyp1 [constraint11] [,...],
    attribut2 datentyp2 [constraint21] [,...],
        ...,
    attributk datentypk [constraintk1] [,...],
    [tabellenconstraint1, ...,tabellenconstraintm]
    );
```

- $attribut_i$ – Name des i-ten Attributs

- $datentyp_i$ – Datentyp des i-ten Attributs
  - CHAR(*n*) – String der festen Länge *n*
  - VARCHAR(*n*) – String variabler Länge (maximal *n*)
  - INT – ganze Zahl (positive oder negative natürliche Zahl)
  - DECIMAL(*n*, *m*) – Festkommazahl mit *n* Stellen insgesamt, *m* davon hinter dem Komma
  - FLOAT – Gleitkommazahl, Kommazahl aber egal wie viele Stellen vor oder hinter dem Komma

- $constraint_{ik}$ – k-ter Constraint des i-ten Attributs -> mehrere möglich (auch keiner)
  - NOT NULL – Attribut muss gefüllt werden
  - UNIQUE – Attribut darf nicht doppelt vorkommen
  - PRIMARY KEY – Attribut ist alleiniger Primärer Schlüssel
  - CHECK(*b*) – Attribut muss Bedingung b erfüllen (z.B. CHECK $attribut_i > 0$)
  - DEFAULT *x* – Wenn nicht gefüllt, dann Default wert *x*
  - REFERENCES **t**(*a*) – Fremdschlüssel der auf Attribut *a* in Tabelle **t** verweist

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Data Definition Language -> Anlegen von Tabellen

- tabellenconstraint$_i$ – gilt meist für mehrere Attribute
  - PRIMARY KEY($a_1, \ldots a_k$) – Zusammengesetzter Primärere Schlüssel
  - *FOREIGN KEY*($a_1, \ldots a_k$) REFERENCES ***t***($b_1, \ldots b_k$) – Wenn mehrere Fremdschlüssel auf eine Tabelle verweisen

```
CREATE TABLE tabellenname (
   attribut1 datentyp1 [constraint11] [,...],
   attribut2 datentyp2 [constraint21] [,...],
        ...,
   attributk datentypk [constraintk1] [,...],
   [tabellenconstraint1, ...,tabellenconstraintm]
   );
```

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Data Definition Language -> Verändern von Tabellen

```
ALTER TABLE tabellenname
    ADD (attribut datentyp); |
    MODIFY (attribut datentyp); |
    DROP (attribut);
```

- ADD (*attribut datentyp*) – Hinzufügen eines Attributs
- MODIFY (*attribut neuer_datentyp*) – Ändern eines Attributs
- DROP (*attribut*) – Löschen eines Attributs
- ADD CONSTRAINT (*constraint_name constraint)* – Hinzufügen eines *constraint* mit Name = *constraint_name*

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Data Definition Language -> Löschen von Tabellen

```
DROP TABLE tabellenname;
```

- Tabelle mit Name *tabellenname* wird gelöscht

-> Auf referenzielle Integrität aufpassen

Wenn Tabelle **ABC** auf Tabelle **A** verweist, darf Tabelle **A** nicht zuerst gelöscht werden, da die Verweise dann in der Luft hängen (dangling references)

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Finn Kapitza
FInn.Kapitza@campus.lmu.de