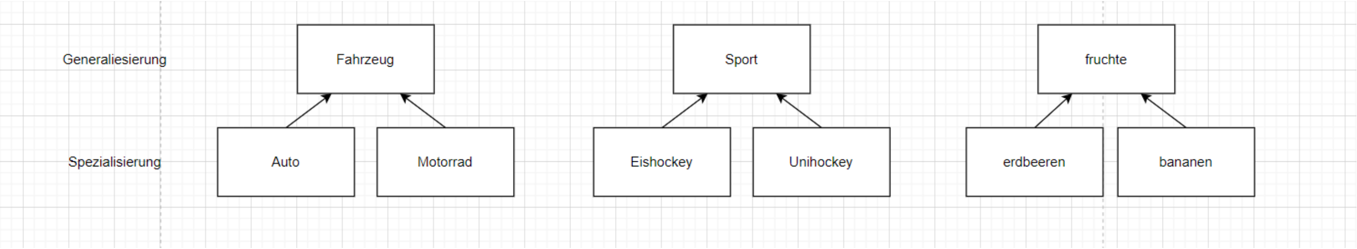
Module 164

PREVIOUS DAYS (need to complete)

Generalization / specialization (person with the role of driver or dispatcher).

The database modeling approach discussed here is based on the attribute concept, where attributes are defined with specific characteristics and assigned to entity types. A problem arises when multiple entity types share many attributes, leading to redundancy if a real-world object is described by several entity types. For instance, employees who also act as customers or drivers who also work as dispatchers illustrate this issue. According to Zehnder (1989), "local attributes" should only appear once in a database to avoid redundancy. The solution is to consolidate common attributes into a general entity type (generalization) while keeping non-common attributes within their respective specialized entity types (specialization). To prevent information loss, specialized tables should reference generalized tables through foreign keys, establishing an "is-a" relationship, similar to inheritance in object-oriented modeling.



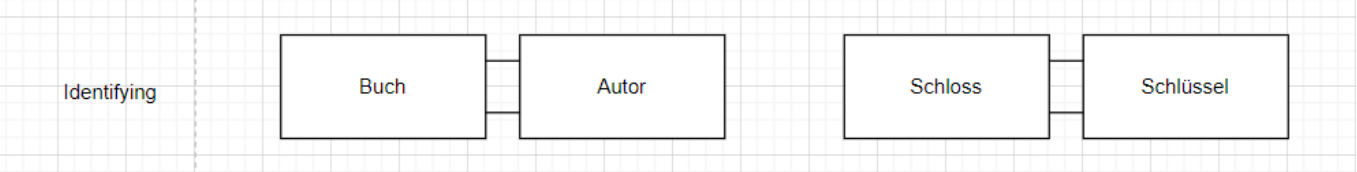
Relationship Types: Identifying/ Non-Identifiying relationship

In databases, relationships between tables can be categorized into identifying and non-identifying relationships. Here's a brief explanation of each:

1. **Identifying Relationship**:
    - The foreign key in the child table is part of its primary key.
    - This forms a key combination of multiple attributes, with the child table's record partially identified by the parent table's record.
    - Example: A room in a building where the room's ID includes the building's foreign key, making it crucial for identification.
2. **Non-Identifying Relationship**:

```
- The foreign key in the child table is not part of its primary key.
- This allows for the foreign key value to change without affecting the
child table's identity.
- Example: An employee and department relationship where the department's
foreign key is not part of the employee's primary key, allowing for
changes in the department without altering the employee's identity.
```

own example
classmates examples



use cases made by chatGPT for identifiying Relationship.

1. **Public Policy**
    - **Social Program Evaluation:** Identifying relationships between program interventions, target populations, and outcomes helps in assessing program effectiveness.
    - **Crime Analysis:** Understanding relationships between socio-economic factors, crime incidents, and law enforcement practices aids in developing effective crime prevention strategies.
    - **Urban Planning:** Mapping relationships between urban infrastructure, population dynamics, and land use patterns supports sustainable urban development.
2. **Environmental Studies**
    - **Ecosystem Analysis:** Identifying relationships between species, habitats, and environmental factors aids in ecosystem conservation and management.
    - **Climate Change Research:** Understanding the relationships between human activities, climate variables, and environmental impacts helps in developing mitigation and adaptation strategies.
    - **Resource Management:** Analyzing relationships between natural resources, consumption patterns, and sustainability practices supports effective resource management.
3. **Research and Development**
    - **Scientific Research:** Identifying relationships between variables in experiments and studies helps in hypothesis testing and theory development.
    - **Innovation Networks:** Mapping relationships between researchers, institutions, and innovations fosters collaboration and knowledge transfer.

- **Patent Analysis:** Understanding relationships between patents, inventors, and technological domains supports strategic intellectual property management.

4. **Technology and Telecommunications**
    - **Network Analysis:** Identifying relationships between network components (e.g., servers, routers, devices) aids in network optimization and troubleshooting.
    - **User Behavior Analysis:** Understanding relationships between user activities, service usage, and device interactions helps in improving user experience and service delivery.
    - **Cybersecurity:** Recognizing relationships between different network activities and potential threats enhances threat detection and response strategies.
5. **Social Networks**
    - **Friendship and Interaction Analysis:** Identifying relationships in social networks helps understand community structure, influence patterns, and information dissemination.
    - **Influencer Identification:** Recognizing relationships between users and their content interactions helps in identifying key influencers within a network.
    - **Behavioral Analysis:** Understanding relationships between user behaviors and network dynamics can improve content recommendations and user engagement.

DBMS (Database Management System)

A database system (DBS) is designed for efficient, consistent, and permanent electronic data management, providing data subsets in various formats for users and applications. A DBS comprises the database management system (DBMS) and the database (DB). The DBMS organizes data storage and controls access, offering a database language for querying and managing data. The most common database system is relational.

Features of a DBMS

A DBMS must provide:

- **Integrated data storage:** Unified management of all application data, allowing complex relationships and efficient data linking. Controlled redundancy can enhance processing efficiency.
- **Database language:** Includes Data Request (retrieval), Data Manipulation Language (DML), Data Definition Language (DDL), and Data Control Language (DCL).
- **User interfaces:** Various interfaces, such as query languages, application programming interfaces, graphical user interfaces (GUI), and web access.

- **Catalogue:** Access to metadata via a data dictionary.
- **User views:** Different views for different user classes, defined in the database's external schema.
- **Consistency control:** Ensures database correctness through integrity assurance, defined by user constraints, and physical integrity.
- **Data access control:** Prevents unauthorized data access through rules and defined rights.
- **Transactions:** Combines multiple changes into atomic transactions, ensuring durability if successful.
- **Multi-user capability:** Synchronizes competing transactions to avoid conflicts, maintaining data isolation for users.
- **Backup:** Restores the database to a correct state after errors.

Advantages of Using a Database

- **Standards:** Facilitates central data organization standards.
- **Efficient data access:** Uses advanced techniques for storing and retrieving large data volumes.
- **Shorter development times:** Offers common functions for faster application development.
- **Flexibility:** Allows database structure modifications without significant impact on existing data and applications.
- **High availability:** Supports high-availability applications through synchronization.
- **Cost-effectiveness:** Centralized investment in powerful hardware reduces overall costs.

Disadvantages of Database Systems

- **High initial investment:** Requires significant expenditure on hardware and software.
- **General-purpose software:** Less efficient for specialized applications.
- **Optimization limits:** Can only be optimized for some applications.
- **Additional costs:** Involves expenses for data security, synchronization, and consistency control.
- **Skilled personnel:** Needs experts like database designers and administrators.
- **Centralization vulnerability:** Risks associated with centralization.

LIST GIVEN BY THE EXERCISE

DBMS	$?	Manufacturer	Model/Characteristics
Adabas	$	Software AG	NF2 model (non-normalized)
Cache	$	InterSystems	hierarchical, "postrelational"
DB2	$	IBM	Object-relational
Firebird		-	relational, based on InterBase
IMS	$	IBM	hierarchical, mainframe-DBMS
Informix	$	IBM	Object-relational
InterBase	$	Borland	relational
MS Access	$	Microsoft	relational, desktop system
MS SQL Server	$	Microsoft	Object-relational
MySQL		MySQL AB	relational
Oracle	$	ORACLE	Object-relational
PostgreSQL		-	object-relational, emerged from Ingres and Postgres
Sybase ASE	$	Sybase	relational
Versant	$	Versant	Object-oriented
Visual FoxPro	$	Microsoft	relational, desktop system
Teradata	$	NCR Teradata	High-performance relational DBMS, especially for data warehouses

LIST FROM DB ENGINE RANKING

DBMS	$?	Manufacturer	Model/Characteristics
Oracle	$	ORACLE	Relational, Multi-model
MySQL		ORACLE	Relational, Multi-model
Microsoft SQL Server	$	Microsoft	Relational, Multi-model
PostgreSQL		PostgresSQL Global Development Group	Relational, Multi-model
MongoDB		MongoDB, Inc	Relational, Multi-model
Redis		Redis project core team, inspired by Salvatore Sanfilippo	Relational, Multi-model
Elasticsearch		Elastic	Relational, Multi-model
IBM Db2	$	IBM	Relational, Multi-model

DBMS	$?	Manufacturer	Model/Characteristics
Snowflake	$	Snowflake Computing Inc.	Relational
SQLite		Dwayne Richard Hipp	Relational
Microsoft Access	$	Microdsoft	Relational
Cassandra		Apache Software Foundation	Wide Column, Multi-model
MariaDB		MariaDB Corporation ab (MariaDB Enterprise), Maria DB Founsation(community MariaDB Server)	Relational, Multi-model
Splunk	$	Spluk Inc.	
Databricks	$	Databricks	Multi-model
Microsoft Azure SQL Database	$	Microsoft	Relational, Multi-model

COMPARISON OF THE TOP 3

Adabas VS Oracle

This is a rough comparison between both of them

Name	Adabas	Oracle
Primary database model	Multivalue DBMS	Relational DBMS
DB-Engines Ranking	Score 3.17 Rank 94 Overall Rank 1 Multivalue DBMS	Score 1236.29 Rank 1 Overall Rank 1 Relational
Developer	Software AG	Oracle
Initial release	1971	1980
License	Commercial	Commercial

MySQL Vs Cache

Sadly Cache isn't on the list of DB-Engines.com so I can't compare both of them.

Microsoft SQL Server Vs DB2

Sadly DB2 isn't on the list of DB-Engines.com so I can't compare both of them.

LB1- Teil 1 Ecolm 5Seiten 1/2h (1Tag-3Tag)
Teil2 Openbook Praxis 40min

Data type	MariaDB	Example	Remark/Setting
Integers	INT	INT 1254	from -32768 to 32767
Natural Numbers	doesn't exist		
Fixed-Point numbers (decimals)	Decimal (M[,D])	Decimal (6,2) 1234.56	M= Total number of digits D=Decimal places
Bulleted Types			
Boolean (logical values)	Boolean(TINYINT(1))		
Character (single character)			
Floats	FLOAT		
Fixed-length string	CHAR		
Variable length string	VARCHAR		
Date and/or time	Date (YYYY-MM-DD)	Date (2024.06.03)	
Timestamp	TIMESTAMP (YYYY-MM-DD HH:MM:SS)	TIMESTAMP (2024-06-03 17:14:42)	
Binary data objects of variable length (e.g. image)			
Compound	doesn't exist		
JSON	JSON Data Type		

Data type	mySQL	Example	Remark/Setting
Integers			
Natural Numbers			
Fixed-Point numbers (decimals)			
Bulleted Types			
Boolean (logical values)			
Character (single character)			
Floats			
Fixed-length string			
Variable length string			
Date and/or time	Date (YYYY-	Date	

Data type	mySQL	Example	Remark/Setting
	MM-DD)	(2024.06.03)	
Timestamp			
Binary data objects of variable length (e.g. image)			
Compound			
JSON			