Inhaltsverzeichnis

1 Datenbanken **1**

1.1 Datenbankmodelle und -modellierung 4

1.1.1 Relationale und nicht-relationale Datenbanken 4

1.1.2 ERD (Entity-Relationship-Modell) 4

1.1.3 NoSQL 4

1.1.4 Welche Arten von NoSQL-Datenbanken gibt es? 5

1.2 Normalisierung 6

1.2.1 Anomalien 7

1.3 SQL 7

1.3.1 Projektion vs. Selektion 8

1.3.2 DDL, DML & DCL 9

1.3.3 CRUD 9

1.3.4 Subqueries 10

1.3.5 Aggregatfunktionen 10

1.3.6 Mengenoperationen (Schnitt-, Vereinigungs- und Differenzmenge) 10

1.3.7 SQL Injection 12

2 Qualitätssicherung **13**

2.1 PDCA-Zyklus 13

2.2 Incident Management 14

2.3 Service Level Agreement (SLA), Servicelevel 1-3 14

2.4 Testen 14

2.4.1 Klassifizierung von Testverfahren 14

2.5 Versionsverwaltung 15

3 IT-Sicherheit **15**

4 Datenschutz **16**

5 Netzwerktechnik **17**

6 Softwareentwicklung **18**

6.1 Algorithmen 18

6.1.1 Flussdiagramm 19

6.1.2 Struktogramm (Nassi-Shneiderman-Diagramm) 20

6.2 Schnittstellen, APIs, Datenaustausch 21

6.3 Objektorientierung 21

6.4 Programmiersprachen 22

6.5 UML Diagramme 23

6.5.1 Klassendiagramm 23

6.5.2 Use-Case-Diagramm (Anwendungsfalldiagramm) 24

6.5.3 Zustandsdiagramm 25

6.5.4 Aktivitätsdiagramm 26
6.6 Softwarearchitektur 27
6.7 Softwareergonomie 27
6.8 Software Engineering 27
6.9 Design Patterns 28
6.10 Softwarequalität 28
6.11 Webentwicklung 29

Quellen **30**

1 Datenbanken

IHK Belegsatz [8]

Syntax	Beschreibung
Tabelle	
CREATE TABLE Tabellenname(Spaltenname <DATENTYP>, Primärschlüssel, Fremdschlüssel)	Erzeugt eine neue leere Tabelle mit der beschriebenen Struktur
ALTER TABLE Tabellenname **ADD COLUMN** Spaltenname Datentyp **DROP COLUMN** Spaltenname Datentyp **ADD FOREIGN KEY**(Spaltenname) **REFERENCES** Tabellenname(Primärschlüssel)	Änderungen an einer Tabelle: Hinzufügen einer Spalte Entfernen einer Spalte Definiert eine Spalte als Fremdschlüssel
CHARACTER	Textdatentyp
DECIMAL	Numerischer Datentyp (Festkommazahl)
DOUBLE	Numerischer Datentyp (Doppelte Präzision)
INTEGER	Numerischer Datentyp (Ganzzahl)
DATE	Datum (Format DD.MM.YYYY)
PRIMARY KEY (Spaltenname)	Erstellung eines Primärschlüssels
FOREIGN KEY (Spaltenname) **REFERENCES** Tabellenname(Primärschlüsselspaltenname)	Erstellung einer Fremdschlüssel-Beziehung
DROP TABLE Tabellenname	Löscht eine Tabelle
Befehle, Klauseln, Attribute	
SELECT * | Spaltenname1 [, Spaltenname2, ...]	Wählt die Spalten einer oder mehrerer Tabellen, deren Inhalte in die Liste aufgenommen werden sollen; alle Spalten (*) oder die namentlich aufgeführten
FROM	Name der Tabelle oder Namen der Tabellen, aus denen die Daten der Ausgabe stammen sollen
SELECT ... **FROM ...** **(SELECT ...** **FROM ...** **WHERE ...) AS** tbl **WHERE ...**	Unterabfrage (subquery), die in eine äußere Abfrage eingebettet ist. Das Ergebnis der Unterabfrage wird wie eine Tabelle - hier mit Namen ttbl behandelt.

SELECT **DISTINCT**	Eliminiert Redundanzen, die in einer Tabelle auftreten können. Werte werden jeweils nur einmal angezeigt.
JOIN / **INNER JOIN**	Liefert nur die Datensätze zweier Tabellen, die gleiche Datenwerte enthalten
LEFT JOIN / **LEFT OUTER JOIN**	Liefert von der erstgenannten (linken) Tabelle alle Datensätze und von der zweiten Tabelle jene, deren Datenwerte mit denen der ersten Tabelle übereinstimmen
RIGHT JOIN / **RIGHT OUTER JOIN**	Liefert von der zweiten (rechten) Tabelle alle Datensätze und von der ersten Tabelle jene, deren Datenwerte mit denen der zweiten Tabelle übereinstimmen
WHERE	Bedingung, nach der Datensätze ausgewählt werden sollen
WHERE EXISTS (subquery) **WHERE NOT EXISTS** (subquery)	Die Bedingung EXISTS prüft, ob die Suchbedingung einer Unterabfrage mindestens eine Zeile zurückliefert. NO EXISTS negiert die Bedingung.
WHERE ... IN (subquery) **WHERE NOT ... IN** (subquery)	Der Wert des Datenfelds ist in der ausgewählten Menge vorhanden. Der Wert des Datenfelds ist in der ausgewählten Menge nicht vorhanden.
GROUP BY Spaltenname1 [, Spaltenname2, ...]	Gruppierung (Aggregation) nach Inhalt des genannten Feldes
ORDER BY Spaltenname1 [, Spaltenname2, ...] **ASC** | **DESC**	Sortierung nach Inhalt des genannten Feldes oder der genannten Felder ASC: aufsteigend; DESC: absteigend
Datenmanipulation	
DELETE FROM Tabellenname	Löschen von Datensätzen in der genannten Tabelle
UPDATE Tabellenname **SET**	Aktualisiert Daten in Feldern einer Tabelle
INSERT INTO Tabellenname [(spalte1, spalte2, ...)] **VALUES** (Wert Spalte 1 [, Wert Spalte 2, ...]) oder **SELECT ... FROM ... WHERE**	Fügt Datensätze in die genannte Tabelle ein, die entweder mit festen Werten belegt oder Ergebnis eines SELECT-Befehls sind
Berechtigungen kontrollieren	
CREATE Benutzer | Rolle **IDENTIFIED BY** 'Passwort'	Erzeugt einen neune Benutzer oder eine neue Rolle mit einem Passwort
GRANT Recht | Rolle **ON** *.* | Datenbank.* | Datenbank.Objekt **TO** Benutzer | Rolle [WITH GRANT OPTION]	Weist einem Benutzer oder einer Rolle ein Recht auf ein bestimmtes Datenbank-Objekt Weist einem Benutzer eine Rolle zu
REVOKE Recht | Rolle **ON** *.* | Datenbank.* | Datenbank.Objekt **FROM** Benutzer | Rolle	Entzieht einem Benutzer oder einer Rolle ein Recht auf ein bestimmtes Datenbank-Objekt Entzieht einem Benutzer eine Rolle

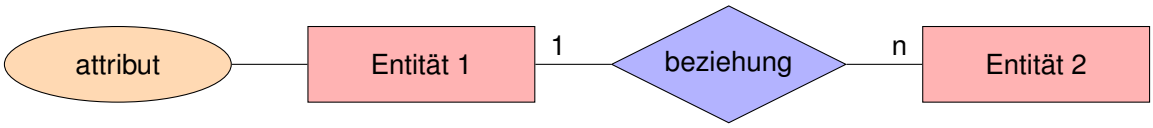
Aggregatfunktionen	
AVG (Spaltenname)	Ermittelt das arithmetische Mittel aller Werte im angegebenen Feld
COUNT (Spaltenname | *)	Ermittelt die Anzahl der Datensätze mit Nicht-NULL-Werten im angegebenen Feld oder alle Datensätze der Tabelle (dann mit Operator *)
SUM (Spaltenname | Formel)	Ermittelt die Summe aller Werte im angegebenen Feld oder der Formelergebnisse
MIN (Spaltenname | Formel)	Ermittelt den kleinsten aller Werte im angegebenen Feld
MAX (Spaltenname | Formel)	Ermittelt den größten aller Werte im angegebenen Feld
Funktionen	
LEFT (Zeichenkette, Anzahlzeichen)	Liefert *Anzahlzeichen* der Zeichenkette von links.
RIGHT (Zeichenkette, Anzahlzeichen)	Liefert *Anzahlzeichen* der Zeichenkette von rechts.
CURRENT	Liefert das aktuelle Datum mit der aktuellen Uhrzeit
CONVERT(time,[DatumZeit])	Liefert die Uhrzeit aus einer DatumZeit-Angabe
DATE (Wert)	Wandelt einen Wert in ein Datum um
DAY (Datum)	Liefert den Tag des Monats aus dem angegebenen Datum
MONTH (Datum)	Liefert den Monat aus dem angegebenen Datum
TODAY	Liefert das aktuelle Datum
WEEKDAY (Datum)	Liefert den Tag der Woche aus dem angegebenen Datum
YEAR (Datum)	Liefert das Jahr aus dem angegebenen Datum
DATEADD (Datumsteil, Intervall, Datum)	Fügt einem Datum ein Intervall (ausgefrückt in den unter Datumsteil angegebenen Einheiten) hinzu
DATEDIFF (Datumsteil, Anfangsdatum, Enddatum) Datumsteile: **DAY, MONTH, YEAR**	Liefert Enddatum-Startdatum (ausgedrückt in den unter Datumsteil angegebenen Einheiten)
Operatoren	
AND	Logisches UND
LIKE	Überprüfung von Text auf Gleichheit wenn Platzhalter ('regular expressions') eingesetzt werden.
NOT	Logische Negation
OR	Logisches ODER
IS NULL	Überprüfung auf NULL
=	Test auf Gleichheit
>, >=, <, <=, <>	Test auf Ungleichheit
*	Multiplikation
/	Division
+	Addition, positives Vorzeichen
-	Subtraktion, negatives Vorzeigen

1.1 Datenbankmodelle und -modellierung

1.1.1 Relationale und nicht-relationale Datenbanken

Eine nicht relationale Datenbank ist eine Datenbank, die nicht das tabellarische Schema mit Zeilen und Spalten verwendet, das in den meisten herkömmlichen Datenbanksystemen zum Einsatz kommt. Nicht relationale Daten verwenden stattdessen ein Speichermodell, das für die spezifischen Anforderungen des gespeicherten Datentyps optimiert ist. So können die Daten beispielsweise als einfache Schlüssel-Wert-Paare, als JSON-Dokumente oder als Diagramm mit Edges und Scheitelpunkten gespeichert werden. [12]

1.1.2 ERD (Entity-Relationship-Modell)



Kardinalitäten:

Entitäten besitzen unterschiedliche Kardinalitäten, also die Anzahl zuordenbarer Objekte einer anderen Entität. Es gibt die Ausprägungen 1:1 (eins zu eins), 1:n (eins zu mehreren) und n:m (mehrere zu mehreren). [4]

1.1.3 NoSQL

NoSQL-Datenbanken wurden speziell für bestimmte Datenmodelle entwickelt und speichern Daten in flexiblen Schemas, die sich leicht für moderne Anwendungen skalieren lassen. NoSQL-Datenbanken sind für ihre einfache Entwicklung, Funktionalität und Skalierbarkeit weithin bekannt. [1]

Flexibilität NoSQL-Datenbanken bieten in der Regel flexible Schemata, die eine schnellere und iterativere Entwicklung ermöglichen. Das flexible Datenmodell macht NoSQL-Datenbanken ideal für halbstrukturierte und unstrukturierte Daten.

Skalierbarkeit NoSQL-Datenbanken sind in der Regel so konzipiert, dass sie durch die Verwendung von verteilten Hardware-Clustern skaliert werden können, im Gegensatz zu einer Skalierung durch das Hinzufügen teurer und robuster Server. Einige Cloud-Anbieter übernehmen diese Vorgänge im Hintergrund als vollständig verwaltete Dienstleistung.

Hohe Leistung NoSQL-Datenbanken sind für bestimmte Datenmodelle und Zugriffsmuster optimiert. Diese ermöglichen eine höhere Leistung, als wenn Sie versuchen würden, ähnliche Funktionen mit relationalen Datenbanken zu erreichen.

Hochfunktionell NoSQL-Datenbanken bieten hochfunktionelle APIs und Datentypen, die speziell für ihre jeweiligen Datenmodelle entwickelt wurden.

1.1.4 Welche Arten von NoSQL-Datenbanken gibt es?

Schlüsselwertdatenbanken Eine Schlüsselwertdatenbank speichert Daten als eine Sammlung von Schlüsselwertpaaren, in denen ein Schlüssel als eindeutiger Identifikator dient. Schlüssel und Werte können alles sein, von einfachen Objekten bis hin zu komplexen zusammengesetzten Objekten. Anwendungsfälle wie Gaming, Werbung und IoT eignen sich besonders gut für das Schlüssel-Werte-Datenspeicherungsmodell.

Dokumentdatenbanken Dokumentdatenbanken verfügen über das gleiche Dokumentmodellformat, das Entwickler in ihrem Anwendungscode verwenden. Sie speichern Daten als JSON-Objekte, die flexibel, halbstrukturiert und hierarchisch aufgebaut sind. Aufgrund des flexiblen, semi-strukturierten und hierarchischen Aufbaus der Dokumente und Dokumentdatenbanken können diese entsprechend den Anforderungen der Anwendungen weiterentwickelt werden. Das Dokumentdatenbankmodell eignet sich gut für Kataloge, Benutzerprofile und Content-Management-Systeme, bei denen jedes Dokument einzigartig ist und sich im Laufe der Zeit weiterentwickelt.

Graphdatenbanken Der Zweck einer Graphdatenbank besteht darin, das Entwickeln und Ausführen von Anwendungen zu vereinfachen, die mit hochgradig verbundenen Datensätzen arbeiten. Sie verwenden Knoten zur Speicherung von Dateneinheiten und Edges zur Speicherung von Beziehungen zwischen Einheiten. Ein Edge hat immer einen Startknoten, einen Endknoten, einen Typ und eine Richtung. Er kann Eltern-Kind-Beziehungen, Aktionen, Besitzverhältnisse und Ähnliches beschreiben. Die Anzahl und Art der Beziehungen in einem Knoten ist nicht beschränkt. Sie können eine Graphdatenbank verwenden, um Anwendungen zu erstellen und auszuführen, die mit stark verbundenen Datensätzen arbeiten. Typische Anwendungsfälle für eine Graphdatenbank sind Social Networking, Empfehlungsmodule, Betrugserkennung und Wissensdiagramme.

In-Memory-Datenbanken Während andere nicht-relationale Datenbanken Daten auf Festplatten oder SSDs speichern, sind In-Memory-Datenspeicher so konzipiert, dass kein Zugriff auf Festplatten erforderlich ist. Sie eignen sich ideal für Anwendungen, die Reaktionszeiten im Mikrosekundenbereich erfordern oder große Verkehrsspitzen aufweisen. Sie können sie in Gaming- und Ad-Tech-Anwendungen für Features wie Bestenlisten, Sitzungsspeicher und Echtzeitanalysen verwenden.

Suchmaschinendatenbank Eine Suchmaschinendatenbank ist eine Art nichtrelationaler Datenbank, die sich der Suche nach Dateninhalten widmet, z. B. nach Anwendungsausgabeprotokollen, die von Entwicklern zur Problembehandlung verwendet werden. Sie verwendet Indizes, um ähnliche Merkmale unter den Daten zu kategorisieren und die Suchfunktion zu vereinfachen. Suchmaschinendatenbanken sind für die Sortierung unstrukturierter Daten wie Images und Videos optimiert.

1.2 Normalisierung

Quelle: DatabaseCamp [2]

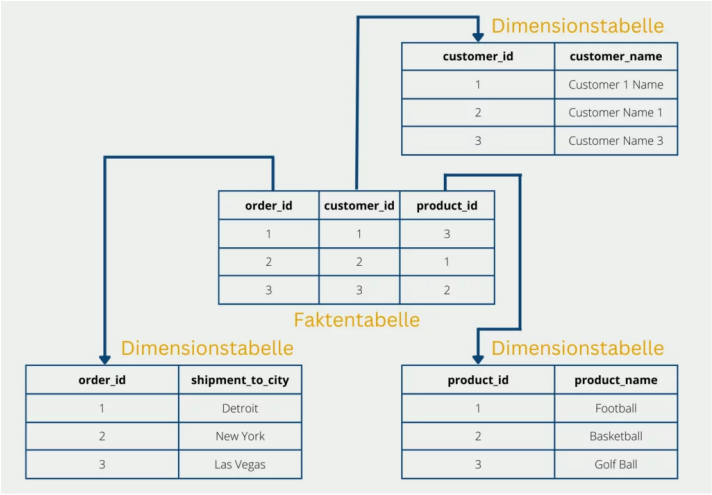
1. Normalform Eine Relation liegt in der ersten Normalform vor, wenn alle Attributwerte atomar vorliegen.

Das bedeutet, dass jedes Datenfeld lediglich einen Wert enthalten darf. Außerdem sollte sichergestellt sein, dass jede Spalte nur Werte desselben Datentyps (Numerisch, Text, etc.) enthält. Folgende Beispiele müssten entsprechend verändert werden, damit eine Datenbank in der 1. Normalform vorhanden ist:

- Adresse: “Hauptstraße 1, 12345 Berlin”
  - Straße: “Hauptstraße”
  - Hausnummer: “1”
  - PLZ: “12345”
  - Ort: “Berlin”
- Rechnungsbetrag: “128,45 €”
  - Betrag: “128,45”
  - Währung: “€”

2. Normalform Eine Relation liegt in der zweiten Normalform vor, wenn sie in der ersten Normalform vorliegt und alle Nichtschlüsselattribute voll funktional vom gesamten Primärschlüssel abhängig sind.

Der Primärschlüssel bezeichnet ein Attribut, das zur eindeutigen Identifikation einer Datenbankzeile verwendet werden kann. Dazu zählen beispielsweise die Rechnungsnummer zur Identifikation einer Rechnung oder die Ausweisnummer zur Identifikation einer Person.



Konkret bedeutet dies in der Anwendung, dass alle Merkmale ausgelagert werden müssen, die nicht ausschließlich vom Primärschlüssel abhängig sind. In der Praxis führt dies dann oft zu einem sogenannten Sternschema.

3. Normalform Eine Relation liegt in der dritten Normalform vor, wenn sie in der ersten und zweiten Normalform vorliegt und keine transitiven Abhängigkeiten bestehen.

Eine transitive Abhängigkeit liegt vor, wenn ein Attribut, welches kein Primärschlüssel ist, nicht nur von diesem abhängt, sondern auch von anderen Attributen.

Wenn wir in unserem Beispiel eine Tabelle haben, in der die Rechnungsnummer, die Produktnummer und der Preis gegeben ist, haben wir höchstwahrscheinlich eine transitive Abhängigkeit. Der Preis des Produktes hängt nämlich nicht wirklich von der Rechnungsnummer ab, sondern vielmehr von der Produktnummer, da für jedes Produkt ein fester Preis definiert ist.

Diese Abhängigkeit kann man auflösen, indem man die Produkte in eine neue Tabelle auslagert und somit das Attribut Preis aus der ursprünglichen Tabelle rausfällt.

1.2.1 Anomalien

Quelle: DatabaseCamp [3]

Anomalien in Datenbanken treten bei einer nicht existierenden oder fehlerhaften Normalisierung auf. Es existieren drei Arten von Datenbank-Anomalien, die Einfüge-Anomalie, die Änderungs-Anomalie und die Lösch-Anomalie.

In der Datenbanktentwicklung ist die Dritte Normalform oft ausreichend, um die perfekte Balance aus Redundanz, Performance und Flexibilität für eine Datenbank zu gewährleisten. Sie eliminiert auch die meisten Anomalien in einer Datenbank, aber nicht alle.

Einfügeanomalie Bei einem fehlerhaften oder inkorrekten Datenbankdesign kann es bei der Einfüge-Anomalie passieren, dass Daten gar nicht in die Datenbank übernommen werden, wenn zum Beispiel der Primärschlüssel keinen Wert erhalten hat, oder eine unvollständigen Eingabe von Daten zu Inkonsistenzen führt.

Änderungsanomalie Bei der Änderungs-Anomalie, auch Update-Anomalie genannt, werden gleiche Attribute eines Datensatzes in einer Transaktion nicht automatisch geändert. So entsteht eine Inkonsistenz der Daten.

Löschanomalie Bei einer Löschanomalie kann es passieren, dass ein Benutzer einer Datenbank aktiv Informationen löschen will und damit indirekt, aufgrund des fehlerhaften Datenbankdesigns, andere zusammenhängende Informationen parallel mitlöscht.

1.3 SQL

Alle SQL-Komponenten für Abfragen siehe *1 Datenbanken auf Seite 1*.

1.3.1 Projektion vs. Selektion

Quelle: Tino Hempel [16]

Selektion Bei der Selektion werden Zeilen aus einer Tabelle ausgewählt, die bestimmten Eigenschaften genügen.

Aus der Tabelle Schüler sollen alle Zeilen selektiert werden, in denen der Name MMüllerßteht. Die Selektion hat also die Form: $S_{Name} =_{'Mueller'} (Schueler)$

Schüler

SNr	Vorname	Name
4711	Paul	Müller
0815	Erich	Schmidt
7472	Sven	Lehmann
1234	Olaf	Müller
2313	Jürgen	Paulsen

$S_{Name} =_{'Mueller'} (Schueler)$

SNr	Vorname	Name
12	Paul	Müller
308	Olaf	Müller

Projektion Bei der Projektion werden Spalten aus einer Tabelle ausgewählt, die bestimmten Eigenschaften genügen.

Aus der Tabelle Schüler sollen alle Spalten mit dem Attribut 'Name' projiziert werden. Die Projektion hat also die Form: $P_{Name}(Schueler)$

Schüler

SNr	Vorname	Name
4711	Paul	Müller
0815	Erich	Schmidt
7472	Sven	Lehmann
1234	Olaf	Müller
2313	Jürgen	Paulsen

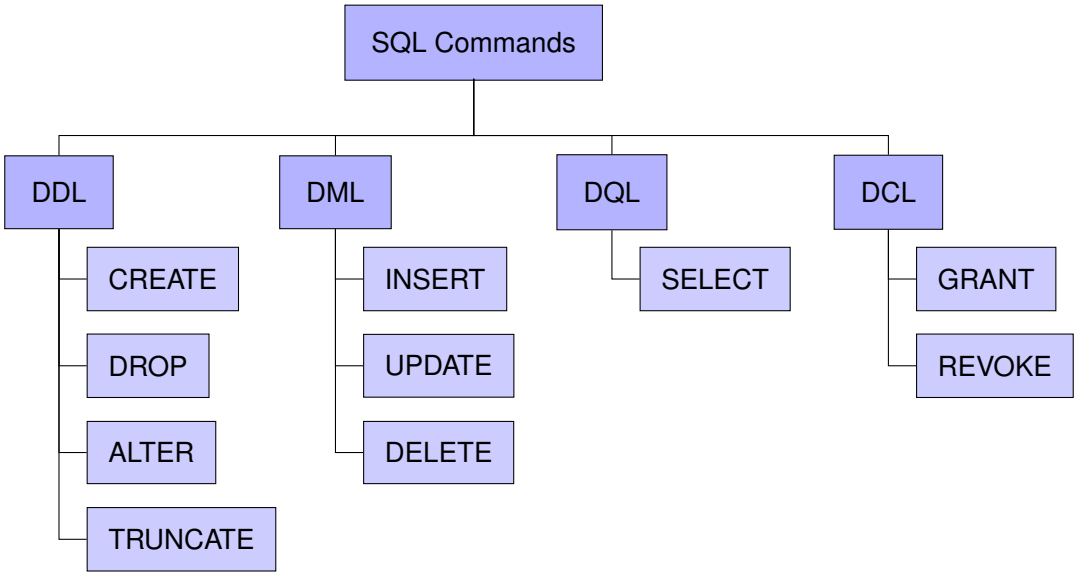
$P_{Name}(Schueler)$

Name
Müller
Schmidt
Lehmann
Müller
Paulsen

1.3.2 DDL, DML & DCL

Quelle: GeeksforGeeks [6]

1. DDL - Data Definition Language
2. DML - Data Manipulation Language
3. DQL - Data Query Language
4. DCL - Data Control Language



1.3.3 CRUD

Quelle: sqlshack.com [14]

C refers to CREATE aka add, insert. In this operation, it is expected to insert a new record using the SQL insert statement. SQL uses INSERT INTO statement to create new records within the table.

```
INSERT INTO <tablename> (column1,column2,...)
VALUES(value1,value2,...),( value1,value2,...), (value1,value2,...) ...
```

```
INSERT INTO dbo.Demo
(id, name)
VALUES
(2, 'Jayaram'),
(3, 'Pravitha');
```

R refers to SELECT (data retrieval) operation. The word 'read' retrieves data or record-set from a listed table(s). SQL uses the SELECT command to retrieve the data.

```
SELECT * FROM <TableName>;
```

U refers to Update operation. Using the Update keyword, SQL brings a change to an existing record(s) of the table. When performing an update, you'll need to define the target table and the columns that need to update along with the associated values, and you may also need to know which rows need to be updated. In general, you want to limit the number of rows in order to avoid lock escalation and concurrency issues.

```
UPDATE <TableName>
SET Column1=Value1, Column2=Value2,...
WHERE <Expression>
```

D refers to removing a record from a table. SQL uses the SQL DELETE command to delete the record(s) from the table.

```
DELETE FROM <TableName>
WHERE <Expression>
```

1.3.4 Subqueries

Fehlt

1.3.5 Aggregatfunktionen

Fehlt

1.3.6 Mengenoperationen (Schnitt-, Vereinigungs- und Differenzmenge)

Quelle: Glossar hs-augsburg [7]

Mengenoperatoren verbinden zwei Abfragen zu einem Resultat.

Beispieltabelle:

Tabelle student

matrikel_nr	name	vorlesung
911574	Meier	Java
676676	Schulz	Datenbanken

Tabelle lehrender

matrikel_nr	name	vorlesung
878999	Kowa	Datenbanken
665544	Müller	XML

UNION bildet die Vereinigung zweier Relationen indem Zeilen der ersten Menge oder des ersten Operanden mit allen Zeilen der zweiten Menge zusammengefasst werden. Zeilen, die in der Ergebnismenge zweimal vorkommen, werden zu einer einzigen Zeile zusammengefasst. Die Datentypen der Spalten müssen kompatibel sein, d.h. es muss entweder ein impliziter Cast (z.B. int auf double) möglich sein, oder wenn dies nicht möglich ist, muß ein expliziter Cast erfolgen. - dies bezieht sich auch auf die Anordnung der Spalten in der Abfrage.

```
1  SELECT name  FROM student
2  UNION
3  SELECT name FROM lehrender
```

Ergebnis:

name
Meier
Schulz
Kowa
Müller

UNION ALL vereinigt alle Zeilen der ersten Menge oder des ersten Operanden mit allen Zeilen der zweiten Menge. Im Unterschied zu UNION werden auch die Duplikate ausgegeben.

INTERSECT überprüft die Zeilen der beiden Eingangsmengen und gibt nur jene Zeilen aus, die in beiden Eingangsmengen vorkommen. Die Durchschnittsmenge wird aus den zwei Relationen gebildete. Auch hier werden vor dem Erstellen der Ergebnismenge die redundanten Zeilen ausgeschaltet.

```
1  SELECT vorlesung FROM student
2  INTERSECT
3  SELECT vorlesung FROM lehrender
```

Ergebnis:

vorlesung
Datenbanken

MINUS gibt die Zeilen aus, die in der ersten Menge, NICHT aber in der zweiten Menge enthalten sind. Zeilen, die in der ersten Menge zweimal vorkommen, werden auf Redundanz überprüft und komprimiert, bevor der Vergleich mit der zweiten Menge beginnt.

```
SELECT vorlesung FROM student
MINUS
SELECT vorlesung FROM lehrender
```

Ergebnis:

vorlesung
Java

1.3.7 SQL Injection

Quelle: kaspersky [10]

Eine SQL-Injection, manchmal abgekürzt als SQLi, ist eine Art von Sicherheitslücke, bei der ein Angreifer einen Teil des SQL-Codes verwendet, um eine Datenbank zu manipulieren und Zugriff auf potenziell wertvolle Informationen zu erhalten. Dies ist eine der häufigsten und bedrohlichsten Angriffsarten, da sie potenziell gegen jede Webanwendung oder Webseite eingesetzt werden kann, die eine SQL-basierte Datenbank verwendet (was bei den meisten der Fall ist).

Arten der SQL-Injection

In-band SQLi Diese Art von SQLi-Angriff ist für Angreifer sehr einfach, da sie denselben Kommunikationskanal verwenden, um Angriffe zu starten und Ergebnisse zu sammeln. Diese Art von SQLi-Angriff hat zwei Untervarianten:

- **Fehlerbasierte SQLi:** Die Datenbank erzeugt aufgrund der Aktionen des Angreifers eine Fehlermeldung. Anhand der von diesen Fehlermeldungen generierten Daten sammelt der Angreifer Informationen über die Datenbankinfrastruktur.
- **Union-basierte SQLi:** Der Angreifer verwendet den UNION-SQL-Operator, um die gewünschten Daten zu erhalten, indem er mehrere Select-Anweisungen in einer einzigen HTTP-Antwort zusammenfasst.

Inferentielle SQLi (auch bekannt als Blind SQL Injection) Bei dieser Art von SQLi nutzen Angreifer die Antwort- und Verhaltensmuster des Servers nach dem Senden von Daten-Nutzdaten, um mehr über seine Struktur zu erfahren. Die Daten werden nicht von der Datenbank der Webseite an den Angreifer übertragen, so dass der Angreifer die Informationen über den In-Band-Angriff nicht sieht (daher der Begriff 'blinde SQLi'). Inferentielle SQLi kann in zwei Untertypen unterteilt werden:

- **Zeitbasierte SQLi:** Angreifer senden eine SQL-Abfrage an die Datenbank und lassen die Datenbank einige Sekunden warten, bevor sie die Abfrage als wahr oder falsch beantwortet.

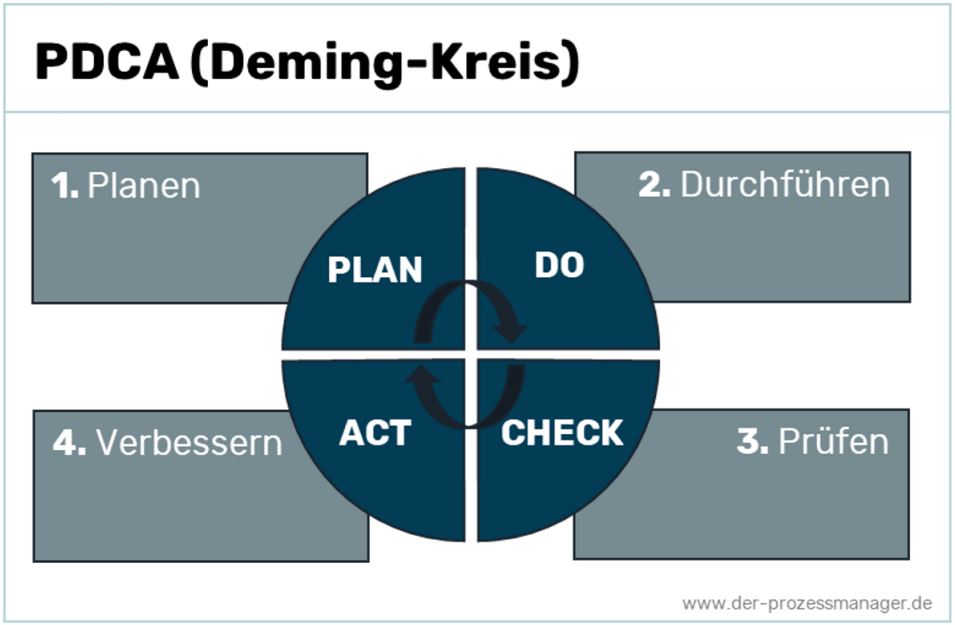
- **Boolean SQLi:** Angreifer senden eine SQL-Abfrage an die Datenbank und lassen die Anwendung daraufhin entweder ein wahres oder ein falsches Ergebnis erzeugen.

2 Qualitätssicherung

2.1 PDCA-Zyklus

Quelle: der-prozessmanager.de [5]

Der PDCA-Zyklus (auch Deming-Kreis, Deming-Zyklus oder PDCA Kreislauf) bezeichnet ein grundlegendes Konzept im kontinuierlichen Verbesserungsprozess. Es dient der Weiterentwicklung von Produkten und Dienstleistungen sowie bei der Fehler-Ursache-Analyse. Der PDCA-Kreis besteht aus den vier sich wiederholenden Phasen: Plan-Do-Check-Act (dt. Planen – Umsetzen – Überprüfen – Handeln).



Vorteile des PDCA-Kreises

Der wesentliche Vorteil der PDCA-Methode ist wohl die einfache Anwendbarkeit. Das Vorgehen hinsichtlich der spezifischen Aufgaben und Problemstellungen kann nahezu uneingeschränkt angepasst werden. Mit den Schritten Plan, Do, Check und Act bleibt dennoch ein solides Gerüst bestehen.

Weitere Vorteile:

- Benötigt wenig Anleitung auf Grund des einfachen Aufbaus
- Die kreisförmige Konzeption ermöglicht ständige Verbesserung
- Durch den iterativen Ansatz lässt der PDCA-Zyklus Kontrolle und Analyse zu

Nachteile des PDCA-Kreises

Der große Vorteil des Demingkreises ist gleichzeitig auch ein wesentlicher Nachteil: Schnelle Problemlösungen lassen sich mit Hilfe des PDCA-Zyklus nicht umsetzen.

Nachteile auf einen Blick:

- Unklare Definition der einzelnen Schritte kann zu falschem Einsatz führen
- Verbesserungen im Unternehmen müssen langfristig gedacht sein
- Eher ein reaktiver Ansatz, statt proaktiv

2.2 Incident Management

Fehlt

Quelle: it-processmaps.com [15]

2.3 Service Level Agreement (SLA), Servicelevel 1-3

2.4 Testen

2.4.1 Klassifizierung von Testverfahren

- Wer testet?
  - Mensch (manuell) vs. Maschine (automatisch)
  - Entwickler vs. Benutzer
- Was wird getestet?
  - Komponente (Unit-Test/Funktionstest/Klassentest) vs. Integration vs. System (End-to-End)
  - Testpyramide
- Wie wird getestet?
  - Bottom-Up vs. Top-Down
  - statisch (Kompilierzeit) vs. dynamisch (Laufzeit)
  - ohne Kenntnis des Codes (Blackbox) vs. mit Kenntnis des Codes (Whitebox)
  - explorativ
  - Schreibtischtest/Review
- Wann wird getestet?

  - Vor vs. nach der Entwicklung
  - Abnahmetest
- Warum wird getestet?
  - Regressionstest
  - Lasttest/Belastungstest
  - Smoketest

2.5 Versionsverwaltung

- Werkzeuge zur Versionsverwaltung einsetzen
- Eigenschaften eines Versionsverwaltungssystems beschreiben
  - SVN, CVS, TFS mit Source Safe, Git
  - VCS vs. DVCS
- Nutzen und Anwenden einschlägiger Systeme, z.B. Git
- Funktionen, z.B. Commit, Revert, Branch, Merge, Cherry-Pick,
- Pull/Push, Rebase
- Übliche Workflows im Team, z.B. Pull/Merge Requests

3 IT-Sicherheit

Fehlt

- Datensicherheit (Authentifizierung, Autorisierung, Verschlüsselung)
- Bedrohungsszenarien erkennen und Schadenspotenziale unter Berücksichtigung wirtschaftlicher und technischer Kriterien einschätzen
- Für jede Anwendung, die verwendeten IT-Systeme und die verarbeiteten Informationen gilt: Betrachtung zu erwartender Schäden, die bei einer Beeinträchtigung von Vertraulichkeit, Integrität oder Verfügbarkeit entstehen könnten
- geeignete Gegenmaßnahmen, z.B. USV-Anlagen, Klimageräte, Firewalls
- Einteilung in die drei Schutzbedarfskategorien „normal“, „hoch“ und „sehr hoch“ (analog IT-Grundschutz des BSI)
- Begriffe kennen/erläutern

  - Hacker (White Hat, Black Hat), Cracker, Script-Kiddies
  - Spam, Phishing, Sniffing, Spoofing, Man-in-the-Middle
  - SQL-Injection, XSS, CSRF, Session Hijacking, DoS, DDoS
  - Viren, Würmer, Trojaner, Hoax, Dialer (veraltet), Keylogger, Botnetze, Spyware, Adware, Ransomware, Scareware
  - Backdoor, Exploit, 0-Day-Exploit, Rootkit
  - Verbreitung von Viren/Würmer/Trojaner erläutern

4 Datenschutz

Fehlt

- Datenschutzgesetze – national und auf EU-Ebene, z.B. Datenschutzgrundverordnung (DSGVO), BDSG
- Grundsätze des Datenschutzes (Art. 5)
  - Rechtmäßigkeit/Gesetzmässigkeit (Erfordernis der gesetzlichen Grundlage)
  - Transparenz gegenüber den betroffenen Personen Zweckbindung
  - Datenminimierung/Verhältnismässigkeit (Datensparsamkeit und Datenvermeidung)
  - Richtigkeit
  - Speicherbegrenzung
  - Integrität und Vertraulichkeit
  - Rechenschaftspflicht
  - Informationssicherheit
- Betroffenenrechte
  - Recht auf Information
  - Recht auf Auskunft
  - Recht auf Berichtigung
  - Recht auf Löschung
  - Recht auf Einschränkung der Bearbeitung
  - Recht auf Widerspruch

  - Recht auf Datenübertragbarkeit
- Persönlichkeitsrechte
  - Recht auf informationelle Selbstbestimmung
  - Recht am eigenen Bild
  - Recht am geschriebenen/gesprochenen Wort
  - Recht auf Schutz vor Imitation der Persönlichkeit
  - Recht auf Schutz der Intim-, Privat- und Geheimsphäre

5 Netzwerktechnik

Fehlt

- Adressierung
  - IPv4/IPv6, MAC, ARP
- Routing, Switching
- DNS, DHCP
- TCP/UDP
- HTTPS, TLS/SSL, IPsec
  - Hash, Signatur, Zertifikat, Certificate Authority
- Verschlüsselung (pre-shared key, RADIUS . . .)
- LAN/WAN/MAN/GAN
- Strukturierte Verkabelung
  - primäre/sekundäre/tertiäre Verkabelung
  - Kabeltypen (Twisted Pair, LWL)
- VLAN
- Sicherheitskonzepte und -risiken: WEP, WPA
- Netzwerktopologien
- Netzwerkplan
- VPN
  - Funktionsweise und Vorteile von VPN beschreiben

  - VPN-Modelle
  - Tunneling
- Serverarten: Mailserver, Webserver, Groupware, Datenbanken, Proxy
- Sicherstellung des Betriebs
  - Elektrotechnisch (USV)
  - Hardwaretechnisch (Redundanzen), RAID
  - Softwaretechnisch (Back-ups. . .)
- Firewall
- Portsecurity, Port-Forwarding

6 Softwareentwicklung

Fehlt

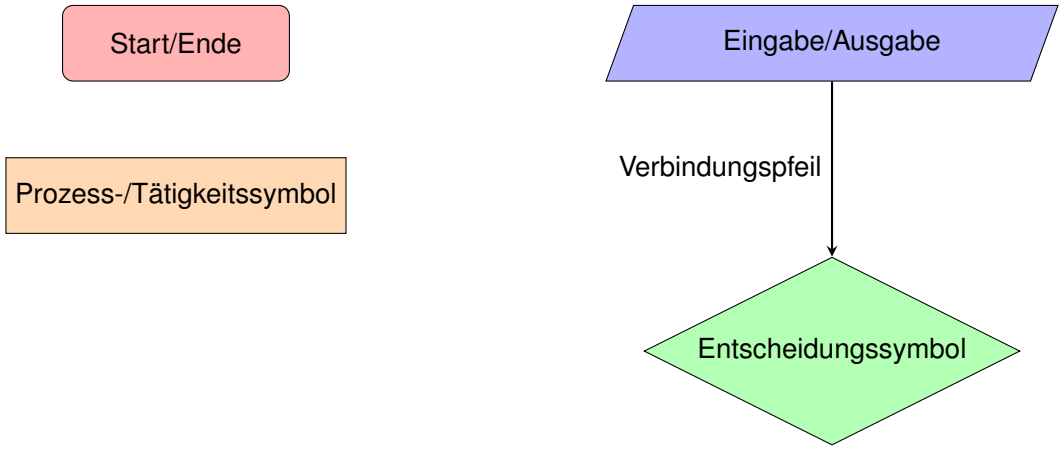
- Allgemeines Fehlerhandling in Programmen
  - Exceptions, Return/Exit Codes
  - Unterschied syntaktische/semantische Fehler
- Rechnerarchitektur: CPU, BUS, Speicher und deren Adressierung
- Lizenzen unterscheiden
  - Open Source, proprietär
- Informationspflichten zu Produkten, Namens- und Markenrecht, Urheber- und Nutzungsrecht, Persönlichkeitsrecht, unlauterer Wettbewerb

6.1 Algorithmen

- Abbildung der Kontrollstrukturen mittels Struktogramm, PAP oder Pseudocode als didaktisches Hilfsmittel
- grundlegende Algorithmen kennen, eigene Algorithmen auch programmiersprachenfrei formulieren und zur Lösung von Problemen, z.B. in einem IT-System bzw. einer Softwareanwendung einsetzen
- Entwickeln und Darstellen von Programmlogiken unabhängig von der Programmiersprache, z.B. mithilfe von Struktogrammen nach Nassi-Shneidermann sowie Strukturdiagrammen und Verhaltensdiagrammen aus der UML

- Rekursion: Funktionsweise, Vor-/Nachteile
- Algorithmen implementieren/durchspielen
  - Mittelwert
  - doppelte Einträge in einem Array finden/löschen
  - Dateibäume rekursiv kopieren
  - (Zinses-)Zinsberechnung
  - Planen eines regelmäßigen Backups
  - Ablauf einer Benutzerauthentifizierung an einer Website
  - Abbuchen von einem Konto
  - Lineare Suche
  - Binäre Suche
  - Bubble Sort

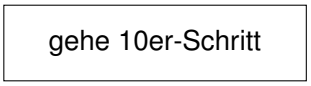
6.1.1 Flussdiagramm



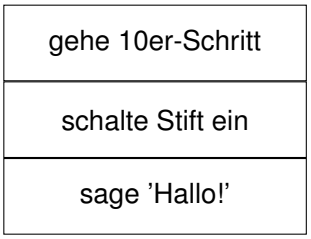
6.1.2 Struktogramm (Nassi-Shneiderman-Diagramm)

Quelle: Lehrerfortbildung-bw.de [11]

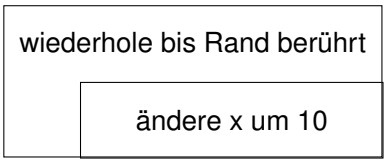
Anweisung



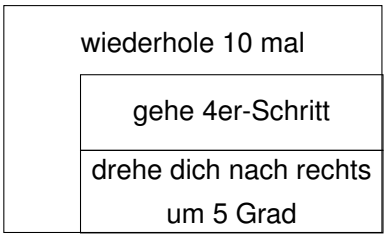
Sequenz



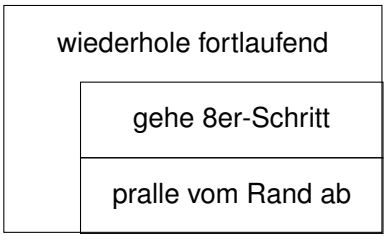
Schleife mit Bedingung



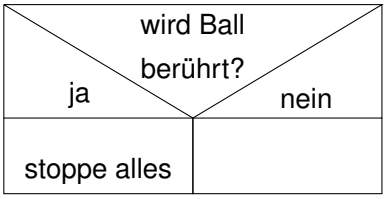
Schleife mit Zähler



Endlosschleife



Verzweigung mit Alternative



6.2 Schnittstellen, APIs, Datenaustausch

- Datenaustauschformate: CSV, XML, JSON
- XML
  - Wohlgeformtheit, Validität
  - DTD, Schema, RelaxNG, Schematron
  - XSLT, XSL-FO
- JSON
  - Syntax, Vor-/Nachteile, Einsatzgebiete
- REST
  - Adressierbarkeit, Zustandslosigkeit, einheitliche Schnittstelle (uniform interface), Ressource vs. Repräsentation
- Webservices
  - SOAP

6.3 Objektorientierung

- Prinzipien der OOP
  - Begriffe der OOP erläutern: Attribut, Nachricht/Methodenaufruf, Persistenz, Schnittstelle/API/Interface, Polymorphie, Vererbung
  - Bestandteile von Klassen
  - Unterschied Klasse/Objekt
  - Unterschied Klasse/Interface
  - Erklärung Klassenbibliothek vs. Framework
  - Klassenbeziehungen: Assoziation, Aggregation, Komposition, Spezialisierung, Generalisierung
- Unterschied statische/nicht-statische Methoden und Attribute
- Datenstrukturen (Baum, Array)
- funktionale Aspekte in modernen Sprachen: Lambda-Ausdrücke, Functional Interfaces, Map/Filter/Reduce, deklarativ vs. imperativ

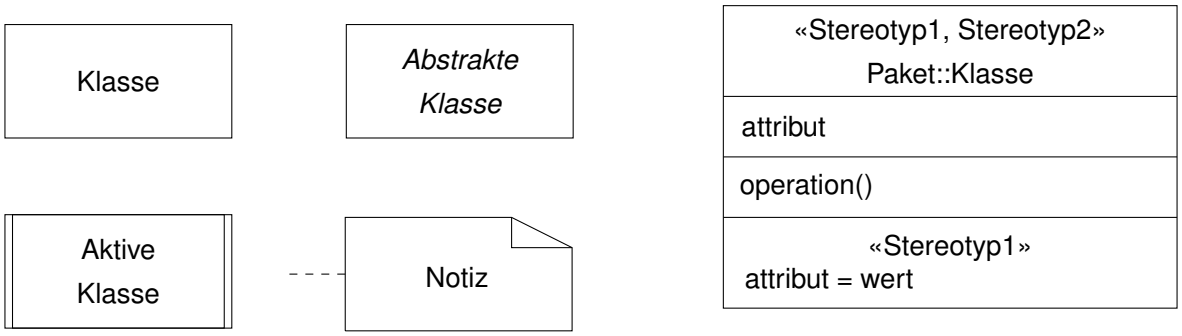
6.4 Programmiersprachen

- Programmierparadigmen: unstrukturiert, strukturiert, prozedural, funktional, objektorientiert, logisch
- imperativ vs. deklarativ
- synchrone vs. asynchrone Programmierung
- Herausforderungen paralleler Programmierung
- Eigenschaften funktionaler Programmierung
  - Pattern Matching

6.5 UML Diagramme

Quelle: Oose.de - Notationsübersicht UML [13] & IHK Belegsatz

6.5.1 Klassendiagramm



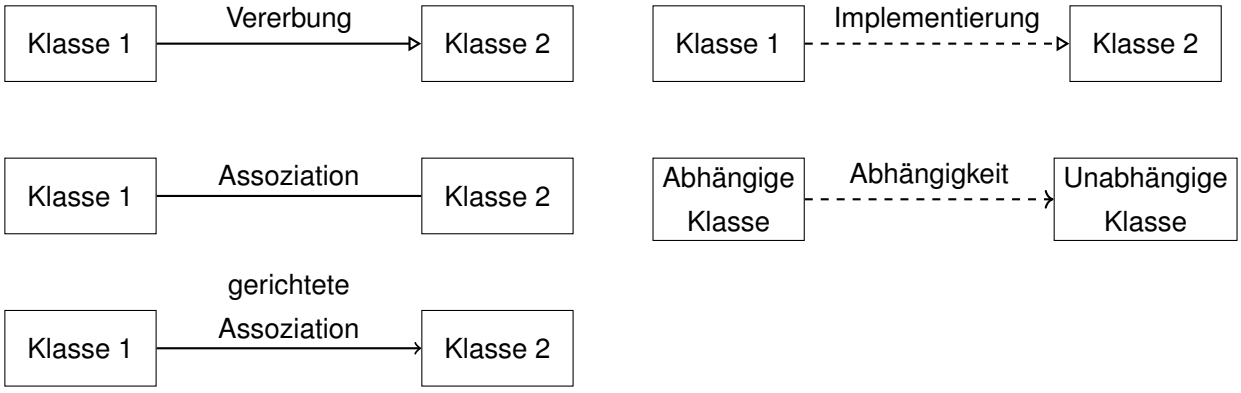
Sichtbarkeit:

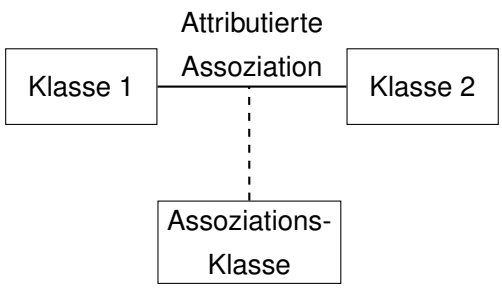
- Öffentlich (+)
- Privat (-)
- Geschützt (#)
- Paket (~)
- Abgeleitet (/)
- Statisch (unterstrichen)

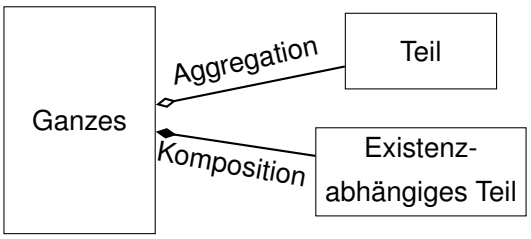
Syntax für Attribute:
Sichtbarkeit Attributname: Typ {Eigenschaften}

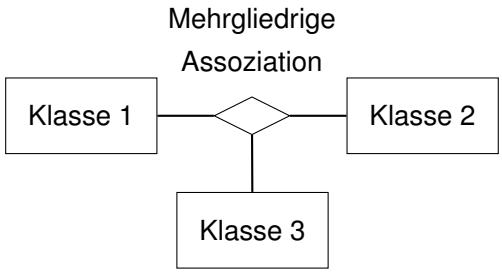
Syntax für Methoden:
Sichtbarkeit Methodenname(parpameter1: Typ, ...): Rückgabewert {Eigenschaften}

Eigenschaften:
{static,final, ...}



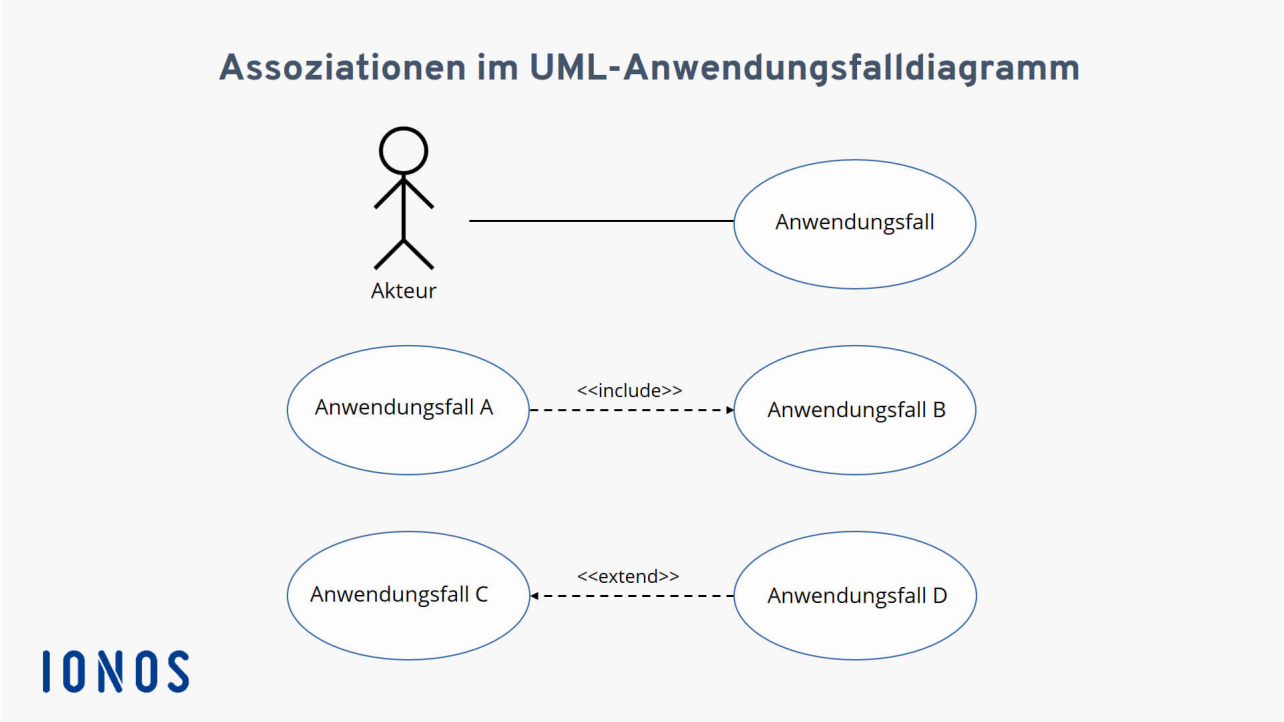


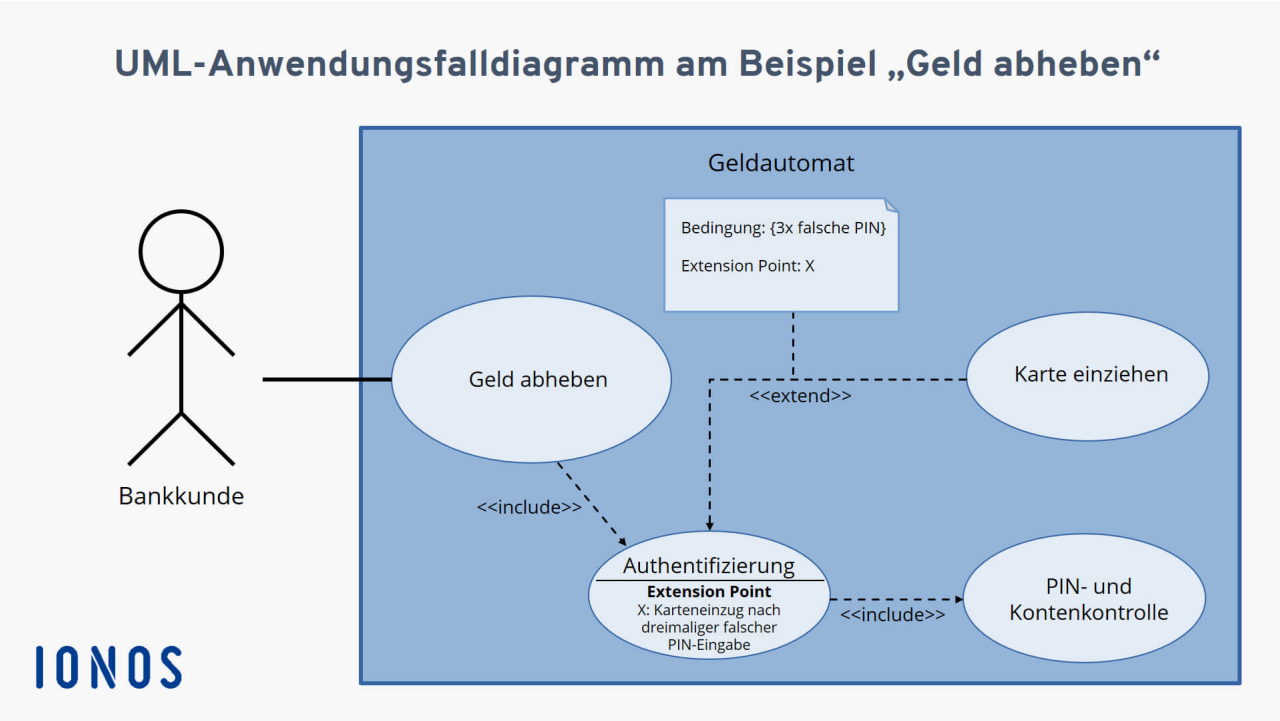




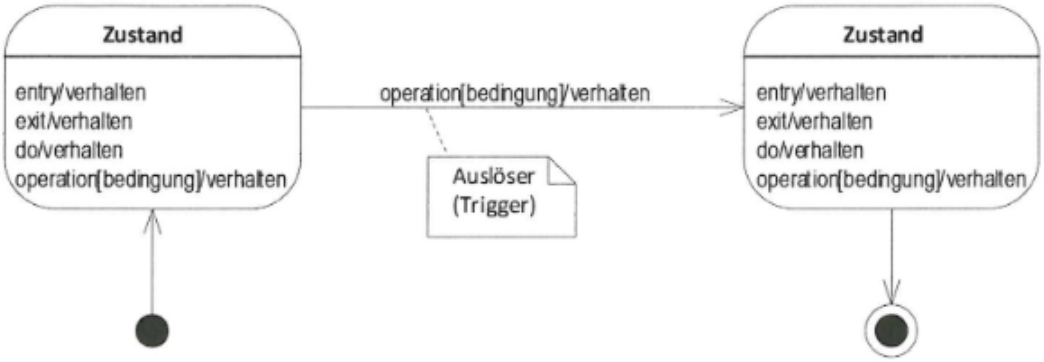
6.5.2 Use-Case-Diagramm (Anwendungsfalldiagramm)

Quelle: Ionos [9]

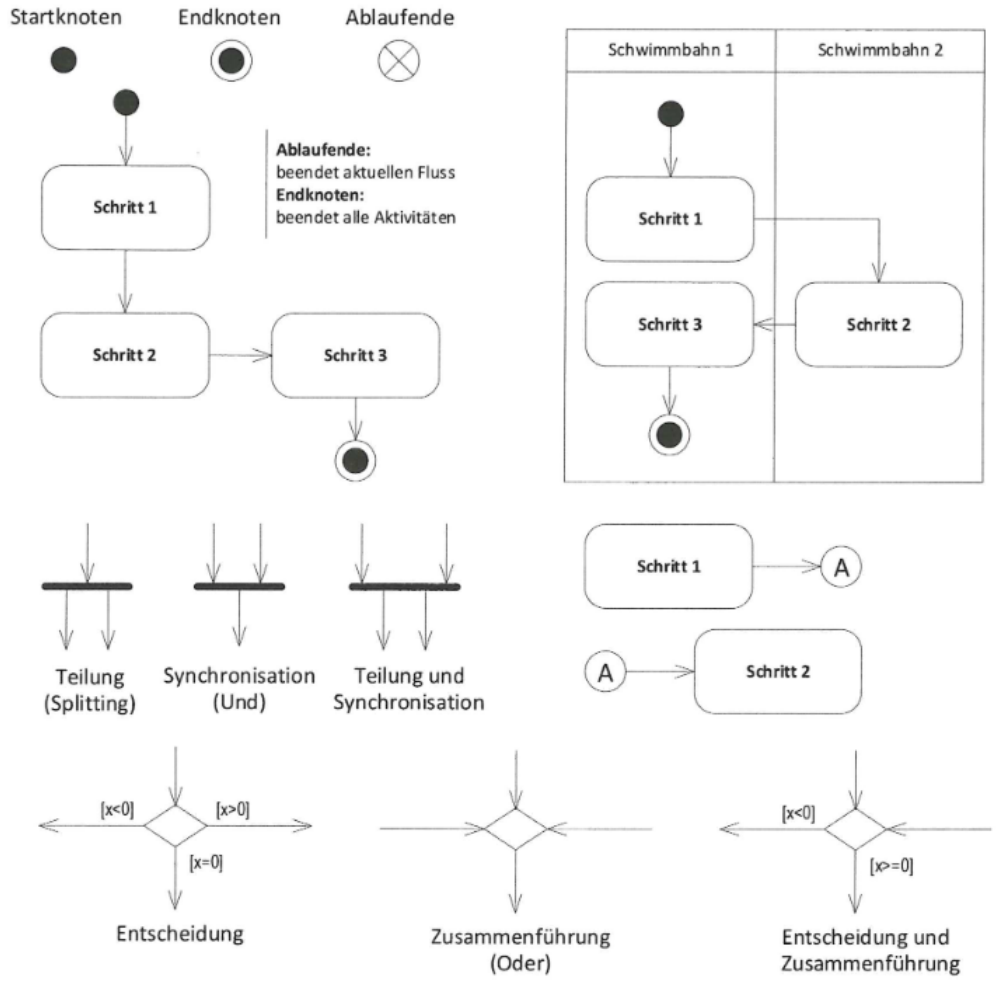




6.5.3 Zustandsdiagramm



6.5.4 Aktivitätsdiagramm



Fehlt

6.6 Softwarearchitektur

- Bottom-Up- und Top-Down-Verfahren bei der Modellierung erläutern
- Funktion/Vorteile der Modularisierung von Programmen
- Softwarearchitektur
  - 3-Schichten-Modell
  - 7-Schichten-Modell
  - Model View Controller (MVC)
  - Model View Presenter (MVP)
  - Model-View-ViewModel (MVVM)
  - REST

6.7 Softwareergonomie

- Mock-up
- Usability vs. User-Experience
- Entwurf der Bildschirmausgabemasken (Softwareergonomie, Barrierefreiheit)
- Barrierefreiheit bzw. Inklusives Design

6.8 Software Engineering

- Entwicklungsprozesse wie das Wasserfallmodell
- Iterative Modelle, z.B. Spiralmodell, V-Modell (XT)
- Agile Modelle: Scrum, Extreme Programming, Kanban
- Top-Down-Entwurf vs. Bottom-Up-Entwurf

6.9 Design Patterns

- Design Patterns kennen/erklären/implementieren
  - Singleton
  - Observer
  - Factory
  - Strategy
  - Decorator
  - MVC

6.10 Softwarequalität

- Software-Qualitätsmerkmale nach ISO 9126 nennen und erläutern
  - Funktionalität: Angemessenheit, Interoperabilität, Ordnungsmäßigkeit, Richtigkeit, Sicherheit
  - Änderbarkeit: Analysierbarkeit, Modifizierbarkeit, Testbarkeit, Stabilität
  - Übertragbarkeit: Anpassbarkeit, Austauschbarkeit, Installierbarkeit, Koexistenz
  - Effizienz: Verbrauchsverhalten, Zeitverhalten
  - Zuverlässigkeit: Fehlertoleranz, Reife, Wiederherstellbarkeit
  - Benutzbarkeit: Attraktivität, Bedienbarkeit, Erlernbarkeit, Verständlichkeit
- Software-Qualitätsmerkmale nach ISO 25010 nennen und erläutern
  - Functional Suitability: Functional Completeness, Functional Correctness, Functional Appropriateness
  - Performance Efficiency: Time Behaviour, Resource Utilization, Capacity
  - Compatibility: Co-existence, Interoperability
  - Usability: Appropriateness Recognizability, Learability, Operability, User Error Protection, User Interface Aesthetics, Accessibility
  - Reliability: Maturity, Availability, Fault Tolerance, Recoverability
  - Security: Confidentiality, Integrity, Non-repudiation, Authenticity, Accountability
  - Maintainability: Modularity, Reusability, Analysability, Modifiability, Testability
  - Portability: Adaptability, Installability, Replaceability
- Maßnahmen zur Qualitätssicherung

  - Continuous Integration/Delivery/Deployment

6.11 Webentwicklung

- Web 2.0
  - Social Networks, Wikis, Blogs, Twitter, Forum, Podcast
- Web 3.0
- Angriffsmöglichkeiten gegen Anwendungen abgrenzen
  - SQL-Injection, Session Hijacking, DoS, DDoS

Quellen

[1] Amazon. Was ist NoSQL? https://aws.amazon.com/de/nosql/, 2024.

[2] Database camp. Normalisierung. https://databasecamp.de/daten/normalisierung, May 2023.

[3] Database camp. Anomalien in Datenbanken. https://www.datenbanken-verstehen.de/datenmodellierung/normalisierung/anomalien-datenbank/, 2024.

[4] Datenbanken-verstehen.de. Entity Relationship Diagram. https://www.datenbanken-verstehen.de/lexikon/entity-relationship-diagram/, 2024.

[5] der-prozessmanager, Michael Durst, Sascha Hertkorn, Christopher Eischer, Nico Schweisser. Was ist ein PDCA-Zyklus? Plan-Do-Check-Act einfach erklärt. https://der-prozessmanager.de/aktuell/wissensdatenbank/pdca-zyklus, 2021.

[6] GeeksforGeeks. SQL | DDL, DQL, DML, DCL and TCL Commands. https://www.geeksforgeeks.org/sql-ddl-dql-dml-dcl-tcl-commands/, October 2023.

[7] hs-augsburg, Kowa. Mengenoperatoren in SQL. https://glossar.hs-augsburg.de/Mengenoperatoren_in_SQL, July 2012.

[8] Industrie und Handelskammer. Belegsatz. https://cloud.agb.schule/apps/files/?dir=/Klassen/FIA101/Tauschen/3.%20Jahr/02%20-%20Pr%C3%BCfungsvorbereitung/03%20-%20Sommer%202023&fileid=10240654, February 2024.

[9] Ionos. Das Use-Case-Diagramm (Anwendungsfalldiagramm) in UML. https://www.ionos.de/digitalguide/websites/web-entwicklung/anwendungsfalldiagramm/, March 2020.

[10] kaspersky. Was ist SQL-Injection? Definition und Erläuterung. https://www.kaspersky.de/resource-center/definitions/sql-injection, 2024.

[11] Lehrerfortbildung-bw.de. Struktogramme. https://lehrerfortbildung-bw.de/u_matnatech/informatik/gym/bp2016/fb1/2_algorithmen/1_hintergrund/2_hintergrund/6_struktogramm/, April 2024.

[12] Microsoft, Zoiner Tejada. Nicht relationale Daten und NoSQL. https://learn.microsoft.com/de-de/azure/architecture/data-guide/big-data/non-relational-data, 2024.

[13] oose.de. Notationsübersicht UML 2.5. https://www.oose.de/wp-content/uploads/2012/05/UML-Notations%C3%BCbersicht-2.5.pdf, 2013.

[14] Prashanth Jayaram, sqlshack.com. CRUD operations in SQL Server. https://www.sqlshack.com/crud-operations-in-sql-server/, December 2018.

[15] processmaps.com, Stefan Kempter. Incident Management. https://wiki.de.it-processmaps.com/index.php/Incident_Management, 2024.

[16] Tino Hempel. Selektion, Projektion und Join in PROLOG. https://www.tinohempel.de/info/info/datenbanken_prolog/abfragen_II.htm, 2006.