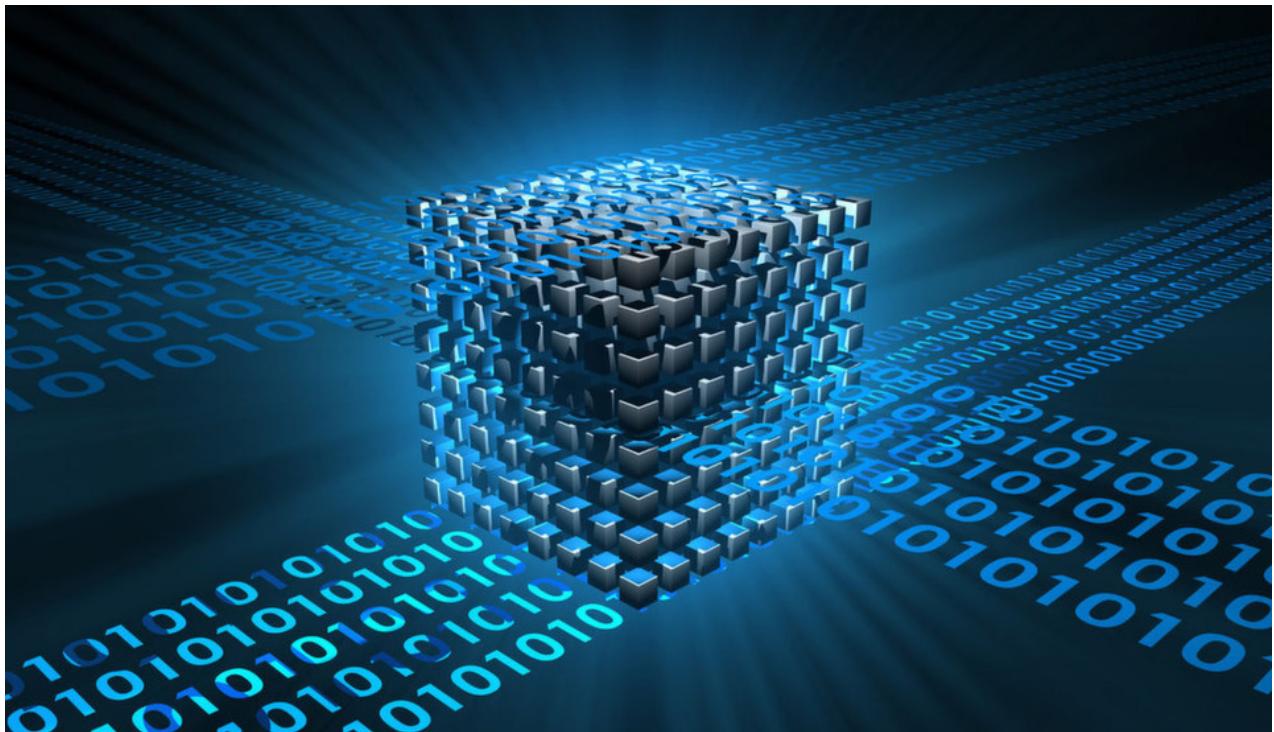

ZUSAMMENFASSUNG 2.KLASSE - INFORMATIK



INFORMATIK

Klasse: 2.Klasse

Erstellt von: GRUBER ELIAS

Erstellt am: 22. Juli 2017

KURZFASSUNG

Einleitung

Hier werden die wesentlichen Punkte des Informatik-Unterrichts der 2.Klasse erwähnt. Das Skript sollte als Lernunterlage für die Lehrabschlussprüfung dienen und das erlernen der einzelnen Themen erleichtern. Weiteres sind alle Angaben ohne Gewähr und die einzelnen Themen sind auf das wesentliche herunter gebrochen. Die Weitergabe an Personen, die nicht von GRUBER ELIAS genehmigt wurde, ist nicht gestattet.

INHALTSVERZEICHNIS

1.	MODULARE PROGRAMMIERUNG	5
1.1	MODULARER ENTWURF	5
2.	OBJEKTORIENTIERTE PROGRAMMIERUNG	5
2.1	KLASSEN, UNTERKLASSEN, OBJEKT, INSTANZ	5
3.	UML (UNIFIED MODELING LANGUAGE)	6
4.	DATENKAPSELUNG	6
5.	KONSTRUKTOREN	7
6.	POLYMORPHISMUS	7
7.	ÜBERSCHREIBEN/ÜBERLADEN VON GEERBTEN DATEN	7
8.	VERERBUNG	8
9.	ABSTRAKTE BASISKLASSEN	8
10.	VERKETTETE LISTEN	9
10.1	EINFACH VERKETTETE LISTE	9
10.1	DOPPELT VERKETTETE LISTE	9
12.	BUBBLE SORT	10
13.	OPERATOREN - PRIORITÄTSTABELLE	10
13.1	STELLIGKEIT	10
13.2	PRIORITÄT	10
13.3	ASSOZIATIVITÄT	10
14.	DATENBANKEN	11
14.1	SCHICHTEN ARCHITEKTUR	11
14.2	DATENBANKMANAGEMENTSYSTEM	12
14.3	DATENBANKARCHITEKTUR	12
14.4	DATENBANKMODELL	13
14.5	ER DIAGRAMM	14
14.6	RELATIONALES DATENMODELLE	15
14.7	RELATIONALE ALGEBRA	15
15.	SQL (STRUCTURED QUERY LANGUAGE)	16
15.1	ALLGEMEINES	16
15.2	SELECT STATEMENTS	16
15.3	JOINS	17
15.4	AGGREGATFUNKTIONEN	17

15.5 INDIZES	17
15.6 STORED PROCEDURES	18
16. SOFTWAREQUALITÄT	18
17. IT-SICHERHEIT	19
17.1 IT-SICHERHEIT FÜR ARBEITNEHMER	19
17.1 IT-SICHERHEIT FÜR UNTERNEHMER	19

1. Modulare Programmierung

Definition: Die Idee der Modultechnik ist, eine Aufgabenstellung nicht als Ganzes zu lösen, sondern diese zunächst in kleine, überschaubare Teilaufgaben zu zerlegen. Die Lösungen der einzelnen Teilaufgaben ergeben zusammengefasst die Gesamtlösung. Programmtechnisch bedeutet dies, dass jede einzelne Teilaufgabe als eigenständiges Programmteil realisiert wird. Ein solches Programmteil nennt man dann ein Modul.

1.1 Modularer Entwurf

- ◆ Ist eine Mischung aus „bottom-up“ und „top-down“ Entwurf
- ◆ Grundlage ist die Prozedurale/Funktionell Programmierung
- ◆ Modularer Entwurf - Fragen werden gestellt um zu einem Konzept zu kommen

Welche Fragen müssen gestellt werden um zu einem Konzept zu kommen?

- Welche Aufgaben und Daten gehören zusammen?
- Wie gehören die einzelnen Aufgabenerstellungen logisch zusammen und welche Schnittstellen sind dabei nach außen erforderlich?
- Bei welchen Programmteilen sind in Zukunft Änderungen zu erwarten?
- Welche Aufgaben werden in ähnlicher oder gleicher Form wieder benötigt?

2. Objektorientierte Programmierung

Definition: Die objektorientierte Programmierung (OOP) ist eine Methode zur Modularisierung von Programmen, die sich stark von der klassischen prozeduralen Programmierung unterscheidet. Objekte können anderen Objekten Botschaften senden (indem sie deren Methoden aufrufen) und sie damit auffordern, ihren Zustand zu ändern.

2.1 Klassen, Unterklassen, Objekt, Instanz

Klassen: Klassen sind Datenstrukturen, die einerseits Variablen, zusätzlich aber auch gleich Funktionen enthalten, die diese Daten manipulieren können.

Unterklassen: Wie der Name es schon angibt, ist es einfach eine Klasse unter der Klasse. Die Unterklassen erben die Eigenschaften der darüber stehenden Klasse.

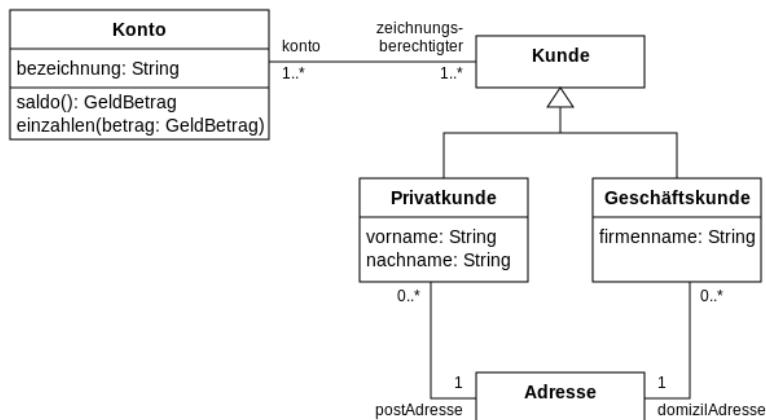
Objekt: Ist die Instanz einer Klasse

Instanz: Wenn z.B. ein Programm geöffnet wird, ist es eine neue Instanz

3. UML (Unified Modeling Language)

Definition: UML ist eine Sprache und Notation, um objektorientierte Systeme zu beschreiben. So legt die UML z.B. Symbole für Klassen, Objekte und drei Beziehungen fest.

Beispiel:



4. Datenkapselung

Definition: Bietet die Möglichkeit, die Datenveränderung einzuschränken und ungewollten Zugriff für bestimmte Daten zu schützen.

Es gibt 3 verschiedene Datenkapselung Möglichkeiten:

private(-): Zugriff nur von innerhalb der Klasse möglich (-=private)

public(+): Zugriff von überall möglich (+=public)

protected(#): Zugriff von eigener Klasse in eine vererbte Klasse

5. Konstruktoren

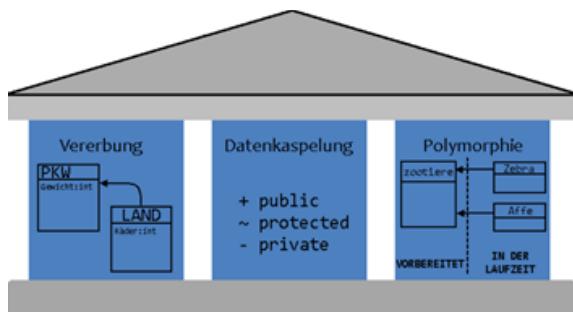
Definition: Die Aufgabe von Konstruktoren ist, Objekte in einen definierten Anfangszustand zu bringen und so benötigte Ressourcen zu reservieren, insofern diese zum Zeitpunkt der Objekterstellung bereits bekannt sind.

Beispielcode in C++:

```
class Foobar
{
public:
    Foobar() // Konstruktor
    {
    }
};
```

6. Polymorphismus

Definition: Hat man gleichnamige Methode in der Basisklasse und in den von dieser Basisklasse abgeleiteten Klassen, so ist die so genannte späte Bindung (Bindung zur Laufzeit) möglich, was man auch als Polymorphismus bezeichnet. Man speicher hierbei verschiedene Objekte der Basisklasse oder der abgeleiteten Klassen in einem Array, dessen Elementtyp die Basisklasse ist. Bei Zugriff auf die einzelnen Elemente dieses Arrays wird dann abhängig von der Klasse, zu dem das jeweilige Objekt gehört, automatisch die zu dieser Klasse gehörige richtige Methode aufgerufen.

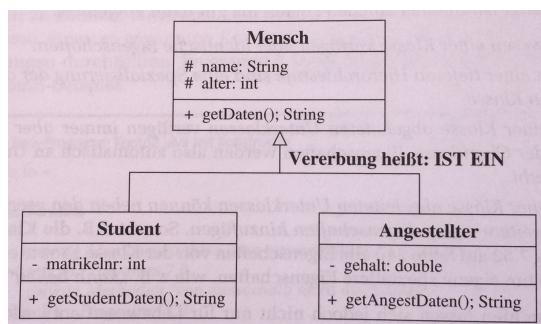


7. Überschreiben/Überladen von geerbten Daten

Definition: Es ist möglich, von der Basisklasse geerbte Methoden in einer abgeleiteten Klasse zu überschreiben. Dazu muss man in der abgeleiteten Klasse diese geerbte Methode nur überschreiben, indem man sie dort einfach neu definiert.

8. Vererbung

Definition: In der unten stehenden Abbildung zeigt eine Basisklasse Mensch, von der zwei Unterklassen Student und Angestellter abgeleitet sind, die beide sowohl die Attribute name und Alter als auch die Methode `getDaten()` von der Klasse Mensch erben, Ihrerseits erweitern sie aber diese geerbten Eigenschaften noch um `matrikelnr` und `getStudentDaten()` bzw. um `gehalt` und `getAngestDaten()`. Beim Vererben von Datenelementen werden diese im jedem Objekt der erbenden Klasse jeweils in einer eigenen Kopie neu angelegt. Da allerdings private Elemente einer Klasse grundsätzlich nicht weitervererbt werden, müsste man ohne dieses Schlüsselwort `protected` alle Elemente einer Basisklasse, die weitervererbt werden sollten, als `public` einstufen.



9. Abstrakte Basisklassen

Definition: Abstrakte Basisklassen dienen dazu, Klassen mit gemeinsamen Eigenschaften zusammenzufassen, wobei diese Eigenschaften also so genannte abstrakte Methoden lediglich definiert, aber erst in den Unterklassen implementiert werden müssen.

Es wird eine abstrakte Methode angelegt (z.B. Motor starten) und danach werden die einzelnen Methoden (Benzin, Diesel, Turbinenmotor) ausprogrammiert.

Abstrakte Klassen sind eine Zusammenfassung aller Objekte, wo schon viele Informationen vorhanden sind.

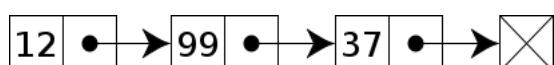
Erst später werden die einzelnen Instanzen aufgerufen.

10. Verkettete Listen

10.1 Einfach verkettete Liste

Definition: Die verkettete Liste ist eine dynamische Datenstruktur, die angeordnete Speicherung von Datenelementen erlaubt. Die Anzahl der Objekte muss dabei im Vorhinein nicht festgelegt werden und bleibt für die gesamte Lebenszeit der Liste offen.

Beispiel:



10.1 Doppelt verkettete Liste

Definition: Im Unterschied zu einer einfach verketteten Liste kann man sich in einer doppelt verketteten Liste über den Nachfolger-Zeiger nicht nur vorwärts, sondern über den Vorgänger-Zeiger auch rückwärts in der Liste bewegen. Dies vereinfacht spezifische Operationen mit der Liste wie das Einfügen neuer oder das Entfernen alter Listenelemente, da man die Liste nicht von Anfang an durchlaufen muss.

Beispiel:



head = Erste Element der Liste

cursor = Aktuelle Element der Liste

tail = Letzte Element der Liste

null = Zeiger zeigt auf nichts - ACHTUNG bedeutet NICHTS, NICHT null

12. Bubble Sort

Vorgehensweise: Vergleicht immer ob der linke Wert größer ist als der Rechte, wenn ja wird es vertauscht.
Je Durchgang ist mindestens das letzte Element korrekt.

Optimierung:

- durch weglassen des letzten Elements bei weiteren Durchgängen
- wenn garnicht getauscht wird, ist die Liste fertig

Beispielcode:

```
bubbleSort3(Array A)
    n = A.size
    do{
        newn = 1
        for (i=0; i<n-1; ++i){
            if (A[i] > A[i+1]){
                A.swap(i, i+1)
                newn = i+1
            } // ende if
        } // ende for
        n = newn
    } while (n > 1)
```

13. Operatoren - Prioritätstabelle

13.1 Stelligkeit

Bedeutung: Wie viele Elemente ein Operator unterstützt. Es gibt ein, zwei und mehrstellige Operatoren

- Einstellige (++ oder --)
- Zweistellige (+,-,*,/)
- Mehrstellige

13.2 Priorität

Bedeutung: Ist die Arbeitsreihenfolge wie eine Rechnung ausgeführt wird.

13.3 Assoziativität

Bedeutung: Bei mehreren gleichartigen Operatoren, welcher Operator zuerst ausgeführt wird.

14. Datenbanken

Definition: Die Aufgabe von Datenbanken ist es, Daten bzw. Informationen in einer vordefinierten Struktur speichern/bearbeiten/löschen.

Diverse Abkürzungen für Datenbanken:

DBS = Datenbanksystem

DBMS = Datenbankmanagementsystem

DB = Datenbank (Daten sammeln / Speicherung)

	MySQL	Access	MongoDB
Erscheinungsjahr	1995	1992	2009
Entwickler	Oracle	Microsoft	Mongo DB, INC
Aktuelle Version	5.7.17 - Dezember 2016	16.0.4229.1024 - 2016	3.4.2 - Februar 2017
Sprache	C und C++	C++	C++
Server Betriebssysteme	FreeBSD, Linux, OSX, Windows, Solaris	Windows	Linux, OSX, Windows, Solaris
XML Unterstützung	ja		
SQL Unterstützung	ja	ja	nein
Fremdschlüssel	ja	ja	nein
Programmiersprachen	C, C#, C++, Java, Javascript, PHP, Perl, Delphi, Eiffel	C,C#,C++,Delphi,Java,VBA,Visual basic.NET	C, C#, C++, Java, Javascript, PHP, PowerShell

14.1 Schichten Architektur

Die 3-Schichten Architektur

- ◆ internes Schema (physische Schicht)
- ◆ konzeptuelles Schema (logische Gesamtsicht)
- ◆ externe Schemata (Benutzersichten)

14.2 Datenbankmanagementsystem

Funktionen:

- ◆ Datenhaltung (speichern, löschen, ändern)
- ◆ Datenschutz (Benutzerverwaltung, Zugriff)
- ◆ Datensicherheit (Raid, Backup)
- ◆ Datenintegrität
- ◆ Multiuserfähigkeit
- ◆ Trigger-& Stored Procedures

14.3 Datenbankarchitektur

Architektur moderner Datenbanksysteme

- ◆ Datenbank-Architektur
- ◆ Software-Architektur
- ◆ Datenbank-Prozesse
- ◆ Data Dictionary

Physische Struktur

- ◆ beliebig viel DB-Files auf beliebige Plattenlaufwerke verteilt; enthalten alle DB-Objekte wie Datenstrukturen, prozedurale Objekte, Zugriffsstrukturen, Daten
- ◆ mindesten zwei REDO-Log-Files; protokollieren und speichern alle Datenänderungen; dienen zur Wiederherstellung der Datenbank bei Systemausfällen
- ◆ mindestens zwei Control-Files; enthalten die Grundstrukturen und Grundinformationen über eine Oracle-Datenbank (Zeitpunkt der Erstellung, Namen aller DB-Files, Namen aller REDO-Files und deren Sequenznummern, Zeitpunkt des letzten Checkpoints und zugehörige REDO-Files,...)

Logische Struktur der DB-Files

- ◆ DB-Files werden Tablespace zugeordnet (Tablespace: Speicherbereich wo logische Datenbanken angelegt werden. Dient dazu, die Datenfiles zu unterscheiden)
- ◆ Je Datenbank existiert mind. ein Tablespace (Tablespace SYSTEM), der beim Installieren mit der Datenbank eingerichtet wird, weitere können angelegt werden
- ◆ alle DB-Objekte werden einem Tablespace zugeordnet
- ◆ Einrichtung nur mit DBA-Rechten möglich

Zentralisierte Datenbanksystem

Es gibt einen zentralen Zugriffspunkt und alle Daten befinden sich zentral in der Datenbank.

Vorteil: kostengünstiger, leichtere Verwaltung

Nachteil: Bei Ausfall und Fehlern nicht mehr verwendbar.

Verteiltes Datenbanksystem

Ist physisch getrennt und logisch aber wieder miteinander verbunden.

Vorteile: Ausfallsicherheit gegeben

Nachteile: teurer, mehrere Sachen einzurichten

14.4 Datenbankmodell

In der Informatik wird zwischen grundsätzlich zwischen 5 Datenbankmodelle unterschieden:

❖ **hierarchische**

- Linux, Planungsphase, Baumartig

❖ **netzwerkartige**

- ähnlich wie hierarchische, Kreisverbindungen möglich, keine Hierarchie

❖ **relationale**

- Beziehung, Tabellenform, Schlüssel

❖ **objektorientierte**

- nicht strukturiert, 1 Objekt = 1 Datensatz

❖ **dokumentorientiert**

- ähnlich wie objektorientiert, 1 Dokument = als Dokument gespeichert + alle Daten

14.5 ER Diagramm

Definition/Bedeutung: E/R-Modell (Entity-Relationship-Modell)

Entität: Ist ein individuell identifizierbares Objekt der Wirklichkeit
z. B. der Angestellte Müller, das Projekt 3232

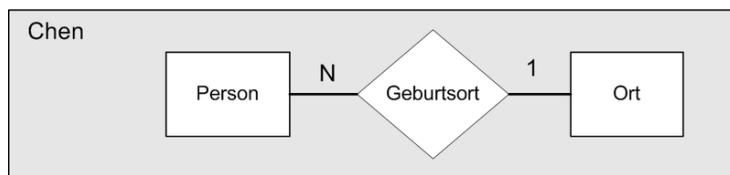
Beziehungen: Verknüpfung / Zusammenhang zwischen zwei oder mehreren Entitäten
z. B. Angestellter Müller leitet Projekt 3232.

Eigenschaft: Was über eine Entität (im Kontext) von Interesse ist
z. B. das Eintrittsdatum des Angestellten Müller.

Im Rahmen der Modellierung werden aus den vorgenannten Sachverhalten gleichartige Typen gebildet. Alle identifizierten Typ-Ausprägungen werden im Modell exakt definiert und beschrieben. Diese Typen unterscheiden sich nach:

- ◆ **Entitätstyp:** Typisierung gleichartiger Entitäten
z. B. Angestellter, Projekt, Buch, Autor, Verlag
- ◆ **Beziehungstyp:** Typisierung gleichartiger Beziehungen
z. B. Angestellter **leitet** Projekt
- ◆ **Attribut:** Typisierung gleichartiger Eigenschaften
z. B. Nachname, Vorname und Eintrittsdatum für den Entitätstyp Angestellter.

Beziehungen (Kardinalitäten)



- ◆ 1:1
 - Ein zugelassenes Fahrzeug hat genau ein KFZ-Kennzeichen und jedes KFZ-Kennzeichen gehört zu genau einem Fahrzeug
- ◆ n:1
 - Eine Person ist in einem (1) Ort geboren. Ein Ort ist Geburtsort von beliebig vielen (N) Personen.
- ◆ 1:n
 - Umgekehrt wie bei n:1

◆ n:m

- Ein Professor unterrichtet üblicherweise mehrere Studenten. Ein Student hört Vorlesungen von mehreren Professoren. → Entsteht eine neue Tabelle mit den Primärschlüsseln von links und rechts

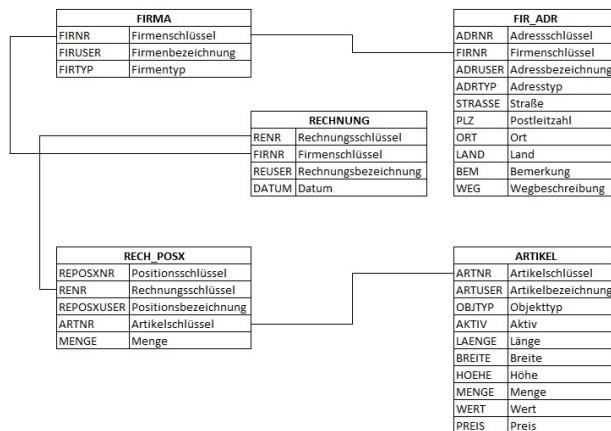
◆ m:n

- Umgekehrt von n:m → Entsteht eine neue Tabelle mit den Primärschlüsseln von links und rechts

14.6 Relationales Datenmodelle

Definition: Das relationale Datenmodell beschreibt die Darstellung von Daten in Tabellenform.

Beispiel:



14.7 Relationale Algebra

Definition: In der Theorie der Datenbanken versteht man unter einer Relationenalgebra oder einer relationalen Algebra eine formale Sprache, mit der sich Abfragen über einem relationalen Schema formulieren lassen. Sie erlaubt es, Relationen miteinander zu verknüpfen oder zu reduzieren und komplexere Informationen daraus herzuleiten.

Selektion: Mit der Selektion werden bestimmte Zeilen aus einer bestehenden Relation ausgewählt. Das Ergebnis der Selektion ist wieder eine Relation nämlich die Ergebnismenge von Zeilen die die Selektionsbedingung erfüllen.

◆ Alle Datensätze wo der Name mit „H“ beginnt.

Projektion: Mit der Projektion werden **Attribute** (Spalten) aus einer bestehenden **Relation** ausgewählt und eventuell deren Reihenfolge in der Ergebnismenge vertauscht.

- ◆ Von allen Datensätzen nur der Name und Geburtsdatum

Produkt: Ein Produkt erzeugt aus zwei Tabellen eine neue Tabelle aus der Kombination aller Zeilen. Jede Zeile der einen Tabelle wird mit jeder Zeile der anderen Tabelle kombiniert.

- ◆ Erzeugt aus zwei Tabellen eine neue Tabelle aus der Kombination aller Zeilen

Differenz: Alle Zeilen einer Relation die NICHT in einer zweiten Relation vorhanden sind.

- ◆ Alle Mitarbeiter die NICHT in der Abteilung 1 sind

15. SQL (*Structured Query Language*)

15.1 Allgemeines

Definition: SQL ist eine Datenbanksprache zur Definition von Datenstrukturen in relationalen Datenbanken sowie zum Bearbeiten (Einfügen, Verändern, Löschen) und Abfragen von darauf basierenden Datenbeständen.

- ◆ DDL (Data Definition Language)
 - ✓ CREATE, DROP, ...
- ◆ DCL (Data Control Language)
 - ✓ Benutzerrechte (GRANT, REVOKE)
- ◆ DML (Data Manipulation Language)
 - ✓ UPDATE, INERST INTO
- ◆ DQL (Data Query Language)
 - ✓ SELECT - (Selection, Projektion, Vereinigungsmengen)

15.2 Select Statements

Definition: Mit dem SELECT Statement können in der SQL Datenbank gespeicherte Daten gezielt gesucht und ausgegeben werden.

Die SELECT Anweisung ähnelt einer Frage bzw. Aufforderung an das Datenbanksystem, die gewünschten Daten zu liefern.

15.3 Joins

Definition: Ein Join (zu Deutsch: Verbund oder Verbindung) dient der Zusammenführung von zwei Tabellen unter bestimmten Kriterien. Durch verschiedene Arten von Joins werden dabei zusätzlich zu den eigentlichen Kriterien noch Grundregeln für die Ergebnismenge festgelegt.

◆ INNER JOIN

```
SELECT e.ename,d.dname FROM emp e JOIN dept d ON e.deptno = d.deptno
```

◆ OUTER JOIN (LEFT, RIGTH JOIN)

```
SELECT e.ename,d.dname FROM emp e, RIGHT JOIN dept d ON e.deptno = d.deptno
```

15.4 Aggregatfunktionen

Definition: Aggregatfunktionen bilden genau einen Ergebniswert aus einer Vielzahl von Eingabewerten.

Die fünf wichtigsten SQL-Aggregatfunktionen sind:

- ◆ COUNT - Anzahl
- ◆ SUM - Summe
- ◆ AVG - Durchschnittswert (arithm. Mittel)
- ◆ MAX - Maximalwert
- ◆ MIN - Minimalwert

Beispiel:

```
SELECT avg(sal) FROM emp
```

15.5 Indizes

Definition: Die Suche in großen Datenbeständen kann mit Indizes erheblich beschleunigt werden. Ein Index ähnelt dem Inhaltsverzeichnis eines Buches, in dem Suchbegriffe geordnet vorliegen und auf die entsprechenden Seiten des Buches verwiese wird.

Beispiel:

```
CREATE INDEX indexname ON tabellenname;
```

15.6 Stored Procedures

Definition: Sind auf dem Server gespeicherte Prozeduren (Unterprogramme), in denen neben den üblichen SQL-Anweisungen auch zusätzliche Befehle zur Ablaufsteuerung und Auswertung von Bedingungen zur Verfügung stehen.

Dient zum Geschwindigkeitsgewinn bei der Ausführung und für besser Sicherheit

16. Softwarequalität

Qualität für Software aus Kundensicht

Usability	Sicherheit	Funktion
Benutzeroberfläche Design Shortcut's	SSL Übertragung Sichere Bezahlungsmethoden vertrauenswürdige Website Bugs mit großen Auswirkungen verhindern	fehlerfrei Funktionen / keine Bugs Dropdown Menüs

Weitere essenzielle Punkte für die Entwicklung einer Software:

- ◆ Wahl der Oberfläche
- ◆ Design / Oberfläche
- ◆ Abläufe definieren (Funktionen)
- ◆ Datenanalyse
- ◆ Sicherheit der Anwendung
- ◆ Detailfunktionen

17. IT-Sicherheit

17.1 IT-Sicherheit für Arbeitnehmer

Im Anhang vorhanden - Dokument: IT Sicherheit_Zusammenfassung_Mitarbeiter

17.1 IT-Sicherheit für Unternehmer

Im Anhang vorhanden - Dokument: IT Sicherheit_Zusammenfassung_Unternehmen