GEFÖRDERT VOM
Bundesministerium für Bildung und Forschung

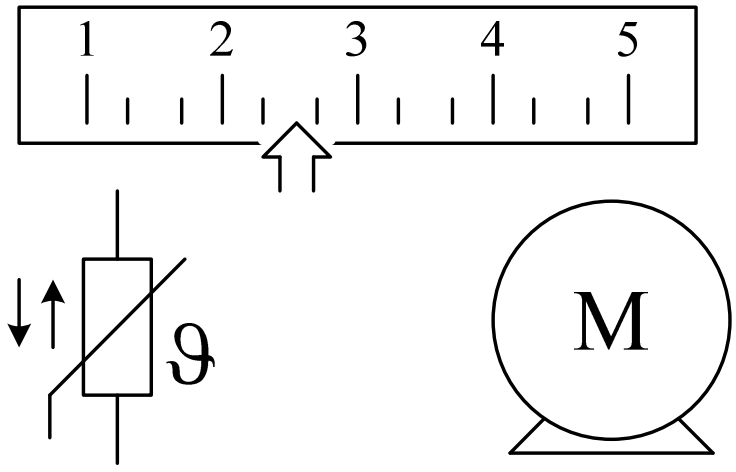
Prof. Dr.-Ing. Gerd W. Wöstenkühler
gwoestenkuehler@hs-harz.de
https://www.hs-harz.de/gwoestenkuehler/zur-person/

AUFSTIEG DURCH BILDUNG
OFFENE HOCHSCHULEN

Berufsbegleitender Bachelor Studiengang Wirtschaftsingenieurwesen

Fach

Messtechnik, Sensorik und Aktorik



Inhaltsverzeichnis

0.1 Überblick

Diese Veranstaltung findet laut Lehrplan im vierten Semester statt. Voraussetzung sind Grundlagen der Elektrotechnik, Physik und Mathematik. Aufgrund der sehr begrenzten Zeit kann nur eine Themenauswahl aus dem Bereich Messtechnik, Sensorik und Aktorik exemplarisch vertieft werden mit der Hoffnung, dass sich die Teilnehmenden später benötigte und nicht behandelte Themen selbständig erarbeiten können.

0.2 Inhalt

Im **ersten** Kapitel wird vorbereitend das Verhalten von LTI-Systemen behandelt. Das Verhalten ist eine Grundvoraussetzung für eine einfache Verhaltensbeschreibung technischer Systeme. Für diese Beschreibung sind die Kenntnisse der Grenzen dieser Gültigkeit notwendig. Als zweiter exemplarischer Schwerpunkt wird die mathematische Darstellung von Drehzeigern in der komplexen Ebene vertieft. Diese Themen sind eine Wiederholung aus Grundlagen der Elektrotechnik und der Mathematik, die für die weiteren Kapitel wichtig sind.

Das **zweite** Kapitel behandelt die Messtechnik oder genauer gesagt die Elektrische Messtechnik. Es besteht aufgrund der begrenzten Zeit die Randbedingung den Inhalt auf das Wesentliche zu begrenzen. Gleiches gilt für die Erstellung von Buchbeiträgen in fachlichen Taschenbüchern. Als theoretische Grundlage wird mit freundlicher Genehmigung des Verlages das von mir erstellte Kapitel Messtechnik aus *Taschenbuch der Technischen Formeln* [1] verwendet. Im Gegensatz zum Script sind die begleitenden Texte ebenfalls kompakt gefasst. Vorteil ist, dass die Qualität eines Buchbeitrags durch Rezessionen gesichert ist. Zur Erläuterung und Vertiefung wird das Kapitel mit angepassten Übungsaufgaben ergänzt.

Im **dritten** Kapitel kann ebenfalls ein von mir geschriebener Buchbeitrag verwendet werde. Es ist das Kapitel Sensoren aus *Taschenbuch der Mechatronik* [2], das ich als Autor mit freundlicher Genehmigung des Verlages für diese Vorlesung verwenden kann. Zur Erläuterung und Vertiefung wird das Kapitel wiederum mit angepassten Übungsaufgaben ergänzt.

Im **vierten** Kapitel wird die Randbedingung berücksichtigt, dass im Studiengang keine elektrischen Antriebe als separates Modul vorgesehen sind. Aus diesem Grund werden im Abschnitt Aktoren im Wesentlichen die Grundlagen der elektrischen Antriebe behandelt.

0.3 Präsenzveranstaltung

Während der Präsenzveranstaltung werden drei Versuche durchgeführt. Für den Themenbereich Messtechnik erfolgt dies an einem Digitalspeicheroszilloskop als universelles Messmittel. Neben einfachen Messungen sind einmalige schnelle und langsame Vorgänge zu erfassen und auszuwerten. Im Bereich Sensorik werden verschiedene industriell einsetzbare Sensoren zur einfachen Erkennung von Materialien als auch zur Füllstandsmessung angewendet. Neben den eher langsamen Vorgängen wird als Abschluss die exemplarische Bestimmung der Grenzfrequenz zweier schaltender Sensoren ermittelt. Im Versuch über Aktoren wird die Kraft in Abhängigkeit der Bewegungsstrecke an pneumatischen Zylindern gemessen. Auch dieser Versuch schließt mit einer Betrachtung dynamischer Vorgänge ab.

0.4 Prüfung

Als Prüfung stehen zwei Varianten zur Verfügung (Präsentation gemäß § 9 (5) der Prüfungsordnung oder Klausur (K60) gemäß § 10 (1) der Prüfungsordnung). Die jeweilige Prüfungsart wird zu Beginn der Veranstaltung bekanntgegeben.

Variante 1: Während der Selbstlernphase ist eine Präsentation über einen speziellen Sensor, Aktor oder einer Kombination aus Sensor und Aktor zu erstellen und in der Gruppe inklusive einer anschließenden Diskussion (Fragen zum Inhalt) zu präsentieren. Die Note setzt sich aus dem Bericht, der Präsentation sowie der anschließenden Diskussion (Beantwortung der Fragen) zusammen.

Variante 2: Im Anschluss an die Labore werden in der Art der klassischen Vorlesung/Übung ergänzende Sensoren und Aktoren vorgestellt und diskutiert. Als Abschluss erfolgt eine Klausur über die Inhalte der Unterlagen (Skript mit Übungsaufgaben), der Labore und der klassischen Vorlesungen/Übungen.

Literatur

[1] **Wöstenkühler, G.W.:** *Taschenbuch der Technischen Formeln*, Kapitel Messtechnik, Karl-Friedrich Fischer (Hrsg.), München: Hanser, 4. Auflage, 2010, Seite 379-411

[2] **Wöstenkühler, Gerd Walter:** *Taschenbuch der Mechatronik*, Kapitel 8: Sensoren, Ekbert Hering und Heinrich Steinhart (Hrsg.), München: Hanser, 2. Auflage, 2015, Seite 272-314

[3] **Fischer, Rolf:** *Elektrische Maschinen*. München: Hanser, 16. Auflage, 2013

[4] **Heimann, Bodo, Gerth, Wilfried, Popp, Karl:** *Mechatronik – Komponenten-Methoden-Beispiele*. München: Hanser, 4. Auflage, 2015

[5] **Hering, E., Bressler, K., Gutekunst, J.:** *Elektronik für Ingenieure und Naturwissenschaftler*. Berlin: Springer, 6. Auflage, 2014

[6] **Hering, E., Schönfelder, G.:** *Sensoren in Wissenschaft und Technik – Funktionsweise und Einsatzgebiete*. Wiesbaden: Vieweg+Teubner, 2012

[7] **Niebuhr, J., Lindner, G.:** *Physikalische Messtechnik mit Sensoren*. München: Oldenbourg, 6. Auflage, 2011

[8] **Profos, P., Pfeifer, T.:** *Handbuch der industriellen Meßtechnik*. München: Oldenbourg, 6. Auflage, 1994

[9] **Schröder, Dierk:** *Elektrische Antriebe – Grundlagen*. Berlin: Springer, 4. Auflage, 2009

[10] **Schrüfer, Elmar, Reindl, Leonhard, und Zagar, Bernhard:** *Elektrische Messtechnik – Messung elektrischer und nichtelektrischer Größen*. München: Hanser, 10. Auflage, 2012

[11] **Tränkler, H.-R., Reindel, L.:** *Sensortechnik – Handbuch für Praxis und Wissenschaft*. Berlin: Springer, 2. Auflage 2014

Inhalte Modul Digital- und Steuerungstechnik

Prof. Dr. René Simon

- Einführung Digitaltechnik
- Schaltsysteme
- Realisierungen
- Schaltalgebra
- Grunddefinitionen
- Grundgesetze
- Schaltfunktionen
- Normalformen
- Automatisierungssystem
- Startup SPS-Programmierung
- Struktur und Funktionalität industrieller Steuerungen
- Endliche Automaten
- Strukturierte Programmierung
- Mehrfachinstanziierung von Funktionsbausteinen
- Archivieren, Dokumentieren
- Weltmarkt für Speicherprogrammierbare Steuerungen

Berufsbegleitender Bachelor-Studiengang Wirtschaftsingenieurwesen

Datenbanksysteme

Arbeitsunterlagen für das Selbststudium

Prof. Dr. Kerstin Schneider

INHALT

0. ÜBERBLICK

Die Vorlesung „Datenbanksysteme“ gliedert sich in Präsenzveranstaltung, Selbstlernphase und Präsenzveranstaltung sowie eine Prüfungsphase: die selbstständige Erstellung bzw. Fertigstellung der abzugebenden Entwurfsarbeit (Hausarbeit)[1].

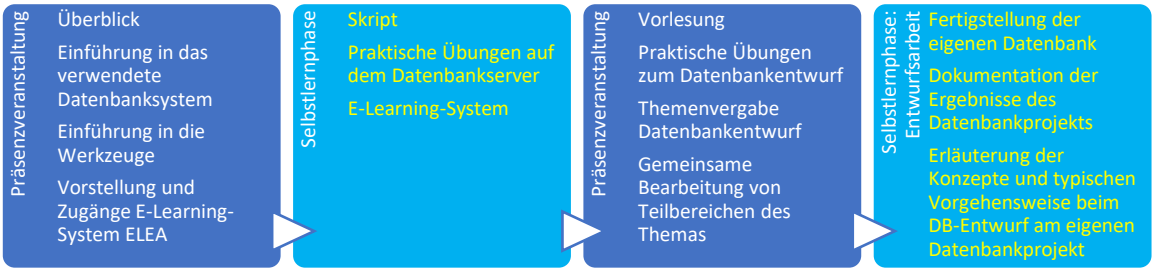


ABBILDUNG 1 GEPLANTE PHASEN DER LEHRVERANSTALTUNG

Da in der Selbstlernphase die Nutzung von Werkzeugen und der Zugriff auf den Datenbankserver für das effiziente Lernen im Bereich Datenbanken wichtig sind, muss eine Einführung in die Werkzeuge in einer ersten Präsenzphase erfolgen. Zudem wird bzw. wurde Ihnen in der ersten Präsenzphase eine Einführung in das DBMS den Datenbank-Entwurfsprozess gegeben um Sie in die Lage zu versetzen eine sinnvolle Selbstlernphase durchzuführen.

Das vorliegende Skript dient zum Selbststudium und enthält im wesentlichen diejenigen Anteile des Lehrstoffs für diese Phase.

Sie können mich jederzeit insbesondere in den Selbstlernphasen über Email sehr gut erreichen: kschneider@hs-harz.de.

In dringenden Fällen über: 0173 6659067

0.1. ÜBERBLICK ÜBER MATERIALIEN ZUM SELBSTSTUDIUM (SKRIPT) UND LEHRVERANSTALTUNG

Im ersten Teil der Vorlesung (1. Präsenzphase) wurde bereits in die zu verwendenden Werkzeuge eingeführt, so dass ein Selbststudium ausgewählter Inhalte möglich wird.

Besonders in Studiengängen ohne direkten Informatikbezug ist es sinnvoll ein praktisches Datenbankentwurfsprojekt durchzuführen. Die Inhalte können durch die praktischen Übungen effizienter erfasst werden, so dass der vorgegebene Zeitrahmen einer (berufsbegleitenden) Lehrveranstaltung optimal genutzt werden kann. Gleichzeitig können die Herausforderungen und Aspekte, die bei Datenbankprojekten in Unternehmen entstehen bzw. bedacht werden müssen, durch die eigenen Erfahrungen in der Lehrveranstaltung besser erkannt werden.

Es ist insbesondere wichtig die Datenbanksprache SQL zu einem gewissen Grad zu beherrschen. Für das Erlernen dieser Sprache und der dafür notwendigen Übungszeiten soll die Selbstlernphase genutzt werden.

[1] Die Art der Prüfung wird in der ersten Präsenzlehrveranstaltung bekanntgegeben und alle Details zur Abgabe werden besprochen. Sollte in einem Semester eine Abschlussklausur oder eine mündliche Prüfung durchgeführt werden, wird dies bereits in der ersten Veranstaltung festgelegt.

In den folgenden Materialien zum Selbststudium wird nach einem kurzen Überblick der Vorgehensweise beim Datenbankentwurf und der wichtigsten Konzepte der Schwerpunkt auf das Erlernen der Datenbanksprache SQL gelegt.

Bis zur zweiten Präsenzveranstaltung sind Sie dementsprechend angehalten SQL zu üben, so dass Sie in der Lage sind die Lehrveranstaltung erfolgreich zu absolvieren. Beim Erlernen von SQL werden gleichzeitig wichtige Konzepte einer relationalen Datenbank behandelt.

0.2. DATENBANKSERVER, EINGESETZTE WERKZEUGE

In der Lehrveranstaltung wird mindestens ein Datenbankserver (zurzeit Oracle 11g bzw. Oracle 12c) eingesetzt, welcher zum Zugriff aus dem Intranet der Hochschule zur Verfügung steht.
Die Verbindung mit dem Intranet der Hochschule kann bspw. über VPN oder bereitgestellte Virtuelle Maschinen erfolgen.

Es besteht zusätzlich die Möglichkeit, sich auf einem eigenen Rechner einen Oracle Database Express Server (frei zur nicht kommerziellen Nutzung unter www.oracle.com) zu installieren, um damit offline arbeiten zu können. Sollten über die Einführung dazu in der ersten Selbstlernphase hinaus weitere Fragen bestehen, können Sie mich gerne kontaktieren.

Um Verbindung mit dem Datenbankserver aufzunehmen und insbesondere mit SQL zu kommunizieren wird das Werkzeug SQL Developer von Oracle eingesetzt. Mit dem SQL Developer kann auf unterschiedliche Datenbankserver insbesondere mit SQL zugegriffen werden. Es ist auch möglich auf einem eigenen Rechner den SQL Developer zu nutzen. Für die Kommunikation mit dem Datenbankserver der Hochschule muss eine Intranet-Verbindung bestehen. Ist ein Oracle Express Server auf dem eigenen Rechner installiert kann offline gearbeitet werden.
Der SQL Developer ist frei verfügbar und kann über die Oracle-Website herunterladen werden. (http://www.oracle.com/technetwork/developer-tools/sql-developer/overview/index.html)

Als Datenmodellierungswerkzeug steht auf den Hochschulrechnern der SAP PowerDesigner zur Verfügung. Mit diesem Werkzeug lassen sich die einzelnen Phasen des Datenbankentwurfs werkzeuggestützt durchführen. Beispielsweise kann das Konzeptuelle Modell in ein Logisches Modell quasi auf Knopfdruck transformiert werden. SQL-Anweisungen zur Erstellung der Tabellen können aus dem Physischen Modell generiert werden. Dies ist notwendig da der vorgegebene Zeitrahmen optimal genutzt werden muss. Die intellektuelle Kontrolle über die einzelnen Phasen und Konzepte soll trotz der Werkzeugunterstützung erreicht werden.
Die Möglichkeit mit einer Testversion offline auf dem eigenen Rechner für 30 Tage zu arbeiten besteht (siehe https://www.sap.com/germany/products/powerdesigner-data-modeling-tools.html).

Die Rolle von Datenbanksystemen in Anwendungen wird zum besseren Verständnis, besonders für nicht informatikbezogene Studiengänge, praktisch demonstriert. In einer beispielhaften Webanwendung wird die eigene Datenbank genutzt, manipuliert und abgefragt. Es wird der Zugriff auf ein RDBMS mit SQL aus einer Programmiersprache (hier JDBC und Java) heraus durchgeführt. Dazu werden Java Server Pages (JSP) und HTML genutzt. Dies wird gemeinsam in der Präsenzphase durchgeführt.
Zum Einsatz kommen: Apache Tomcat (tomcat.apache.org), Java JDK (www.oracle.com), OJDBC (www.oracle.com)

0.3. ELEA: EIN E-LEARNING-SYSTEM FÜR DIE DB-LEHRE

Im Online-Übungssystem ELEA stehen für Sie aktuelle Labore bereit, die Ihnen nach dem Einloggen im ELEA-System zur Auswahl stehen. Die Labore besitzen jeweils unterschiedliche Schwierigkeitsstufen und greifen auf unterschiedliche Datenbanken bzw. Schemas zu.

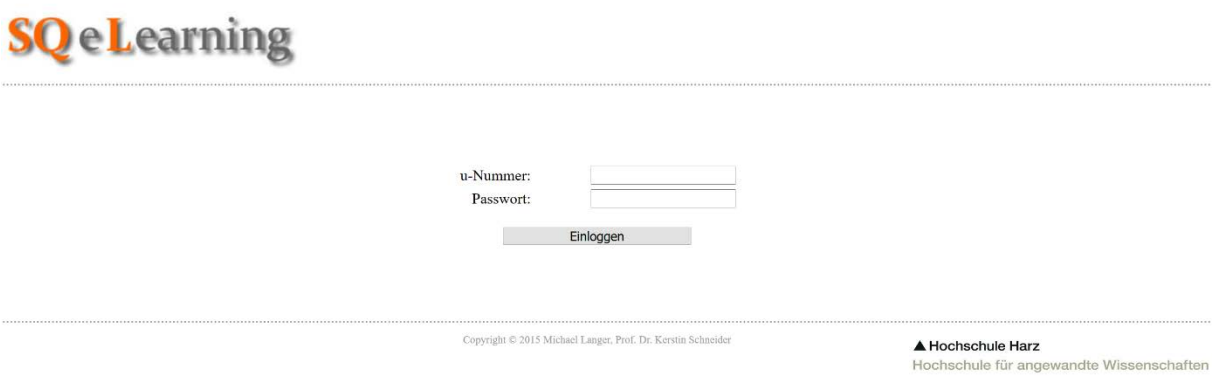


ABBILDUNG 2 ANMELDUNG ZUM ELEA-SYSTEM

Das ELEA-System ist eine Eigenentwicklung von meinem Bereich Datenbanken an der Hochschule Harz. Es ist über die URL http://datenbanken.hs-harz.de zu erreichen. Das ELEA-System steht zurzeit ausschließlich Studierenden der Hochschule Harz zur Verfügung. Für die Anmeldung am ELEA-System benötigen Sie Ihre regulären Zugangsdaten über Ihre u-nummer (siehe Abbildung 2).

Um sich anmelden zu können, müssen Sie zuvor im ELEA-System eingetragen worden sein und Aufgaben und Labore zugeordnet bekommen haben. Im Rahmen dieser Lehrveranstaltung ist dies bereits geschehen. Die Ihnen zugeordneten oder individuelle bereitgestellten Labore werden Ihnen nach der Anmeldung in einer Liste zur Auswahl angezeigt.

Beachten Sie bitte: Erfolgreich bearbeitete Aufgaben werden nicht weiter angezeigt. Ein bearbeitetes Labor erscheint nicht mehr. Bei Ihrer nächsten Anmeldung zeigt das System die Aufgaben und Labore in dem aktuell von Ihnen noch zu bearbeitenden Zustand. Sollten Sie weitere Aufgaben oder Labore zum Üben wünschen oder Aufgaben nochmals wiederholen wollen, geben Sie mir eine kurze Info, dann stelle ich Ihnen dies zur Verfügung.

0.4. PRÜFUNGSLEISTUNG

Im Rahmen der Vorlesung wird ein Datenbankentwurfsprojekt durchgeführt. Die Themenvergabe erfolgt in der zweiten Präsenzphase. Während der zweiten Präsenzphase wird jeweils in Gruppen von zwei bis vier Studierenden an dem Entwurf einer Datenbank mit dem jeweiligen vorgegebenen Thema und den vorgegebenen Anforderungen gearbeitet, so dass die in der Vorlesung behandelten Themen jeweils am eigenen Projekt praktisch geübt und vertieft werden können. Vorlesung und praktische Übungen wechseln sich in der Präsenzphase häufig ab. Interaktionen sind erwünscht. Nutzen Sie die Möglichkeit zur Interaktion in dieser Phase. Eine gute Vorbereitung und ausreichendes Üben von SQL in der Selbstlernphase werden vorausgesetzt.

Jeder Studierende muss schließlich selbstständig eine Ergebnis-Dokumentation über das Projekt erstellen. In dieser individuellen Dokumentation soll von dem/der jeweiligen Studierenden in eigenen Worten und anhand des eigenen DB-projektes als durchgängiges Beispiel erläutert werden,

wie bei einem Datenbankentwurf vorzugehen ist. Die in der Vorlesung behandelten Konzepte sollen am eigenen Beispiel erläutert werden.

Nicht alle Details des Datenbankprojektes können während der Präsenzphase bearbeitet werden. Die während der Präsenzphase nicht fertiggestellten Teile müssen daher eigenständig fertiggestellt werden. Beispiele sind das Einfügen von ausreichend vielen Testdaten, oder die Erstellung einer vorgegebenen Anzahl und Art von Beispielanfragen. Beim konzeptuellen Datenmodell sollen alle Konzepte mindestens einmal Verwendung finden und zumindest erläutert werden.

Für die erfolgreiche Durchführung eines Datenbankentwurfsprojektes ist eine gute Vorbereitung in der Selbstlernphase sehr wichtig. Ein gutes praktisches Verständnis von SQL und eine gute Übersicht über den Entwurfsprozess sowie über die Werkzeuge und Konzepte, die zum Einsatz kommen, sind notwendig und Voraussetzung, um die DB-Projekte in der Präsenzphase und der Fertigstellungsphase erfolgreich durchführen zu können und eine Dokumentation im vorgegebenen Zeitrahmen (d.h. der Abgabetermin muss eingehalten werden) erstellen zu können.

Die Dokumentation ist spätestens zum vereinbarten Abgabetermin per E-Mail als Anhang(!)[2] an kschneider@hs-harz.de zu senden. Zusätzlich sind die entsprechenden Dateien (z.B. SQL-Skriptdateien (ddl, dml, drl) und Dateien mit den Datenmodellen (ER-Modell, Relationenmodell) des DB-Projektes als E-Mail-Anhang einzureichen. Die Dateien können gepackt werden (bspw. als zip-Datei) und ggfs. können Dokumentation und Projekt-Dateien auf mehrere E-Mails verteilt werden.

Achtung: Senden Sie sich unbedingt eine Kopie der E-Mail (mit BCC oder CC) an Ihre eigene Hochschul-E-Mailadresse (<unummer>@hs-harz.de). Dadurch können Sie selbst überprüfen, ob die E-Mail tatsächlich (an die Hochschule) versendet wurde und ob die Datei im Anhang lesbar und unbeschädigt ist. Eine Eingangsbestätigung wird nicht versandt.

0.5. HINWEISE ZUM ARBEITEN MIT DEM SKRIPT

Die Selbstlernmaterialien sollten konzentriert gelesen und gelernt werden. Für das Erlernen von SQL in Kapitel 2 gelten besondere Hinweise wie folgt:

Für Kapitel 2, welches die Datenbanksprache SQL behandelt, gilt: Es sind die wichtigsten(!) und unbedingt(!) auswendig zu Lernenden Begriffe/Schlüsselworte rot dargestellt.

Beispiele für SQL-Anweisungen sind grün dargestellt.

Faustregeln lila ☺

Beachten Sie bitte:

Lesen Sie sich das Skript konzentriert durch.
Sollten Sie einen Zugang zu einem Datenbanksystem wie besprochen haben oder erhalten, können Sie die SQL-Anweisungen mit dem SQL Developer ausprobieren. Dafür erhalten Sie zusätzlich eine Textdatei, welche alle in diesem Dokument erläuterten Anweisungen enthält. Aus dieser Textdatei lassen sich die Anweisungen kopieren und im SQL Developer ausführen.

[2] Die Dateien müssen explizit an die E-Mail angehängt sein. Ein Link zum Herunterladen der Dateien ist hier nicht zulässig.

Sollten Sie keinen Zugang haben, können Sie das Skript auch offline konzentriert durchsehen. Die Erläuterungen im Skript reichen zum Verständnis aus.

Wichtig ist, dass Sie das, was wir in der letzten Veranstaltung besprochen, ausprobiert und vorbereitet haben, nicht vergessen! Das Skript soll Ihnen helfen, relevante Teile zu vertiefen und ergänzen, so dass Sie für das Datenbankprojekt gut vorbereitet sind. Das vorliegende Skript befasst sich nach einer kurzen Wiederholung allgemeiner Aspekte zu Datenbanken im Wesentlichen mit der Sprache SQL, die, wie Sie bereits wissen, notwendig ist um mit dem (relationalen) Datenbanksystem zu kommunizieren.

Bitte notieren Sie sich alle Ihre Fragen, so dass wir diese besprechen und klären können! Fragen können Sie mir auch direkt per email senden.

Viel Erfolg!

1. GRUNDLAGEN VON DATENBANKEN

In diesem Kapitel wird eine kurze Darstellung wesentlicher Teile der in der ersten Präsenzphase bereits besprochenen Lehrinhalte mit den wichtigsten Stichpunkten gegeben.
Bitte prägen Sie sich diese ein!

- Ein DBMS ist eine Sammlung von Programmen, die es den Nutzenden ermöglichen Datenbanken zu erstellen, zu manipulieren und zu verwalten

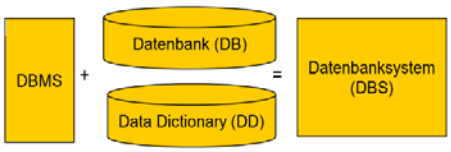


ABBILDUNG 3 DBMS, DB, DD, DBS

1.1. VORTEILE VON DATENBANKSYSTEMEN

Der Einsatz von Datenbanksystemen hat große Vorteile:

- Integration der Daten und unabhängige logisch zentralisierte Verwaltung der Daten
  - Herauslösen aller Aufgaben der Datenverwaltung und Konsistenzkontrolle aus den Anwendungsprogrammen
- Datenunabhängigkeit und Anwendungsneutralität beim logischen und physischen Datenbankentwurf
- Einfache und flexible Nutzung der Daten durch geeignete Anwendungsprogrammierschnittstellen
- Zentralisierung aller Maßnahmen zur Integritätskontrolle
- Transaktionsschutz für die gesamte Datenbankverarbeitung
- Effiziente und parallele Verarbeitung von großen Datenmengen – Mehrbenutzerbetrieb
- Hohe Verfügbarkeit und Fehlertoleranz
- Skalierbarkeit der Transaktionslast und der Datenvolumina

Wesentliche Eigenschaften von DBMS

- Große Datenmengen (Performanz)

  - Effiziente Speicherung und Verwaltung
  - Schnelle Zugriffsmöglichkeiten
  - Transparente Nutzung externer Speicher
  - Auch „kleine“ Datenmengen können sinnvoll verwaltet werden ☺
- Mehrbenutzerfähigkeit
- Dauerhaftigkeit der Daten (Persistenz)
- Fehlertoleranz
  - Wiederherstellbarkeit durch Recovery-Mechanismen,
  - Transaktionssicherheit, Zuverlässigkeit
- Einfache Abfragemöglichkeiten
  - ad-hoc Anfragen, auch eingebettet in Wirtssprachen
  - ohne Wissen über technische Aspekte / Implementierung
- Konsistenzüberwachung durch das Datenbanksystem
- Offenheit für neue Anwendungen
- Redundanzfreiheit der Daten aus Sicht der Anwendung

1.2. KLASSEN VON DATENBANKSYSTEMEN

Es existieren unterschiedliche Klassen von Datenbanksystemen (siehe Abbildung 4). Ein wesentlicher Unterschied ist die Art der verwendeten Datenstrukturen bzw. der logischen Datenmodellierung (Schema). Einige noSQL-DBMS sind sogar „schemafrei“. Dies geht jedoch über die Inhalte der Lehrveranstaltung hinaus.
Wir befassen uns in dieser Lehrveranstaltung im Wesentlichen mit Relationalen DBMS (RDBMS). Hier ist die Datenbanksprache grundsätzlich SQL und ist ein de-facto-Standard (→ Kapitel 2).

Für das allgemeine Verständnis insbesondere auch der anderen DBMS-Klassen ist es in jedem Fall wichtig die Konzepte von RDBMS zu kennen. Zudem sind RDBMS sehr verbreitet.

Hierarchische DBMS

Netzwerk-DBMS Deduktive DBMS

Relationale DBMS (RDBMS) u.v.m.

Objektorientierte DBMS (OODBMS)

XML-DBMS

Objekt-relationale DBMS (ORDBMS)

Spaltenbasierte DBMS

In-Memory DBMS

Weitere NoSQL-DBMS (“**not only SQL**” 😜):
Dokumentenbasierte DBMS, Key-Value Stores, Graph-DBMS, Multimodale DBMS, Grid &Cloud DB-Lösungen, MultiValue DBMS, Streaming DBMS, u.v.m.
(siehe http://nosql-database.org/index.html)

ABBILDUNG 4 KLASSEN VON DBMS

1.3. RELATIONALE DATENBANKEN

Tabelle (Relation) ist „einzige“ Datenstruktur (neben den atomaren Werten)

Tupel

Attributname 1	Attributname 2	Attributname 3	...	Attributname n

- Information wird nur durch die Datenwerte repräsentiert
- Primärschlüssel zur Identifikation des Tupels
- Integritätsbedingungen innerhalb/zwischen Tabellen: Relationale Invarianten
  - Fremdschlüssel als Verweis
- Eine Tabelle (Relation) ist eine „Menge“:
  - Die Reihenfolge der Zeilen (Tupel) einer Tabelle ist ohne Bedeutung
  - Die Eindeutigkeit der Zeilen (Tupel) wird garantiert
- Die Reihenfolge der Spalten ist ohne Bedeutung
  (aber die Zusammengehörigkeit von Attributnamen und Werten ist bedeutungsvoll)

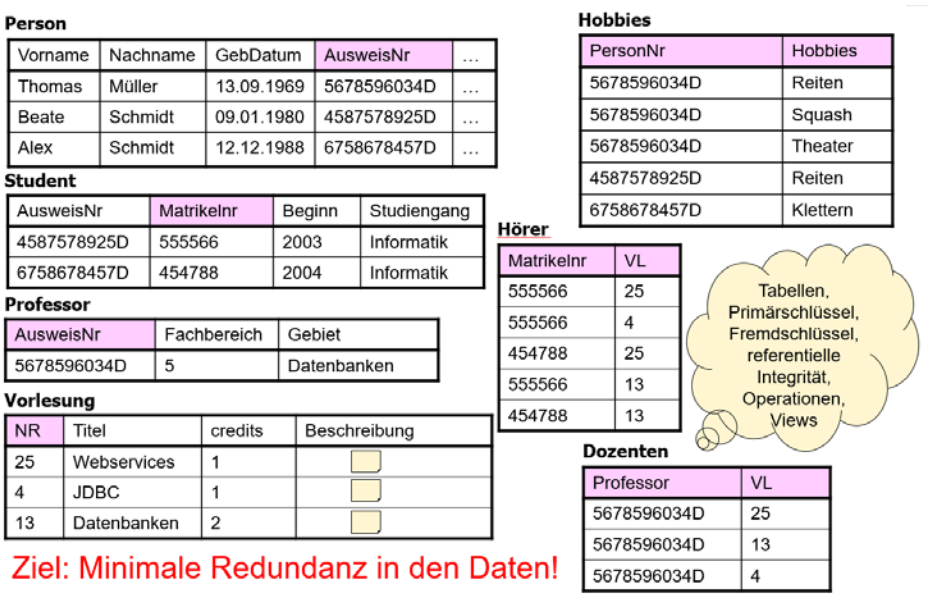


ABBILDUNG 5 BEISPIEL EINER RELATIONALEN DATENBANK

Der Primärschlüssel jeder Tabelle ist rot markiert.

Relationale Invarianten:

- Kein Primärschlüssel darf NULL sein
- Wenn ein Tupel in einer Tabelle über einen Fremdschlüssel ein anderes Tupel in der gleichen oder in einer anderen Tabelle über deren Primär- oder Kandidatenschlüssel referenziert, dann muss gelten:
  - Jeder Wert des Fremdschlüssels, der ungleich NULL ist, muss ein existierender Wert des korrespondierenden Primär- oder Kandidatenschlüssels sein.
  - Die Wertemengen (Domains) von Fremdschlüssel und korrespondierendem Primär- oder Kandidatenschlüssel müssen identisch sein.

1.4. ENTWURF EINER (RELATIONALEN) DATENBANK

Die grundsätzliche Vorgehensweise beim Datenbankentwurf soll die Abbildung 6 verdeutlichen.

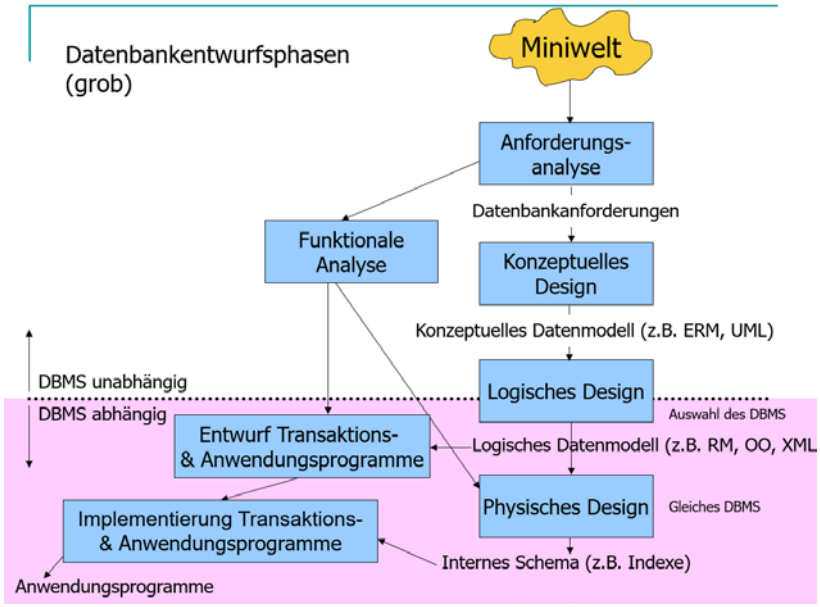


ABBILDUNG 6 PHASEN BEIM DATENBANKENTWURF (VGL. ELMASRINVATHE)

Das Ziel ist ein Modell eines anwendungsorientierten Teils der realen Welt (Miniwelt, Universe of Discourse). Es gilt

- Information wird als Daten gespeichert.
- Exakte, formale, eindeutige Abbildung (d.h. nicht mehrdeutig, nicht missverständlich)
- Aktuelle Information (hoher Grad an Aktualität)
- Einfache, verständliche, natürliche Repräsentation
- Sammlung des Informationsbedarfs während der Systemanalyse
  → Informationsmodell (allgemein Systemmodell)

Die Miniwelt ist so exakt wie möglich abzubilden, damit das System so viel wie möglich automatisch prüfen und unterstützen kann: system enforced integrity.

Mögliche Probleme und Herausforderungen bei der Informationserhebung

- Synonyme kontrollieren!
  - Wörter, die die gleiche Bedeutung haben und gegeneinander ausgetauscht werden können
    - Beispiel: MITGLIED und GENOSSE bedeutet im Steuerberatungsunternehmen das gleiche
- Homonyme beseitigen!
  - Wörter, die gleich geschrieben und gesprochen werden, aber eine deutlich andere Bedeutung haben
    - Beispiel STEUER als Lenkvorrichtung und als Abgabe an den Staat
- Äquipollenzen aufdecken!

- Die selben Objekte werden aus verschiedenen Blickwinkeln betrachtet
    - Beispiel: LAGERBESTAND als mengenmäßige und WARENBESTAND als wertmäßige Rechnung über den Artikelbestand einer Firma

- Vagheiten klären!
    - Da inhaltlich keine klare Abgrenzung/Definition der Begriffe erfolgt, treten hinsichtlich der Objekte, die unter den Begriff fallen, Unklarheiten und Unsicherheiten auf
        - Beispiel: Gehört der WOHNSITZ als Ort der Berufsausübung eines Beraters noch zum Begriff UNTERNEHMEN

- Falsche Bezeichner ersetzen!
    - Abweichung der tatsächlichen von der zunächst suggerierten Wortbedeutung des Begriffs
        - Beispiel: Bei DATEV hat die BERATERNUMMER nicht die Funktion SteuerBERATER zu identifizieren, sondern sie identifiziert eine von mehreren NUTZUNGSBERECHTIGUNGEN, die ein SteuerBERATER hat

1.4.1. KONZEPTUELLE DATENMODELLIERUNG

Die konzeptuelle Datenmodellierung adressiert Menschen, daher ist eine grafische Darstellung sinnvoll. Alle Beteiligten an einem Datenbankprojekt sollen zu einem gemeinsamen Verständnis über die abzubildende Miniwelt gelangen. Ein Konzeptuelles Datenmodell ist dann von hoher Qualität, wenn es die Miniwelt so genau wie möglich abbildet und wenn Personen, die das Modell sehen diese Information (d.h. die abgebildete Miniwelt) daraus eindeutig und unmissverständlich erfassen können.

Wir setzen hier das Entity-Relationship-Modell (ERM) ein. Wesentliche Konzepte im ERM sind:

- Reguläre und schwache Entitätsmengen
- Attribute, mehrwertige Attribute, abgeleitete Attribute und zusammengesetzte Attribute
- Primär- und Kandidatenschlüssel
- Beziehungstypen: binär bzw. mehrstellig; rekursiv;
  Rollen
- Kardinalitäten
  Bei binären Beziehungstypen treten die folgenden Kardinalitäten auf:
    - one-to-one, 1:1
    - one-to-many, 1:n
    - many-to-one, n:1
    - many-to-many, n:m
- Kardinalitätsrestriktionen, bspw. min-max-Notation

Es gibt unterschiedliche ERM-Notationen zur Darstellung. Wichtig sind die grundsätzlichen Konzepte.

1.4.2. LOGISCHE DATENMODELLIERUNG

Hier wird das von der DBMS-Klasse unabhängige konzeptuelle Datenmodell in ein zu dem entsprechenden DBMS passendes Datenmodell abgebildet.

Für ein RDBMS muss auf ein Relationenmodell abgebildet werden, d.h. es werden bspw. die entsprechenden Tabellen (=Relationen) festgelegt.

Es wird angestrebt

- die Anzahl der Tabellen zu minimieren,
- NULL-Werte, wenn möglich zu vermeiden
- sowie Redundanzen möglichst zu vermeiden.

Der Grad der Minimalität von Redundanz lässt sich über die Erreichung von Normalformen (NF2 = Non First Normal Form, 1NF, 2NF, 3NF, BCNF, 4NF, 5NF) überprüfen. Es gilt: Für eine höhere NF muss grundsätzlich die nächstniedrigere bereits erreicht sein, dazu kommen weitere Bedingungen. Es gilt also. Ist ein RM in 3NF, dann ist es auch in 2NF und dann ist es auch in 1NF. Sobald die 1NF für eine Tabelle erreicht ist, spricht man von normalisierten Tabellen. Sind alle Tabellen einer DB in 1NF, so ist die DB in 1NF. Die DB wird dann als normalisiert bezeichnet.

Diese Normalformen sagen dementsprechend etwas über die „Qualität“ des erstellten Datenmodells aus. Die Prüfung auf Normalformen und die Umwandlung des Datenmodells mit dem Ziel einer höheren Normalform kann sowohl am ER-Modell als auch im RM durchgeführt werden. Wir werden das RM prüfen. Da zur Erreichung von höheren Normalformen ggfs. mehr Tabellen notwendig sind, muss hier ein Kompromiss zwischen (weniger Redundanz oder weniger Tabellen) gefunden werden, so dass in der Regel nur die 3NF angestrebt wird.

Je höher die Normalform desto geringer kann die Redundanz sein. Der Vorteil von geringerer Redundanz ist das Vermeiden von möglichen Änderungsanomalien, die zu inkonsistenten Daten führen. Die Normalformen und die Vorgehensweise bei einer Normalisierung werden in der Präsenzphase besprochen. Wird eine sehr gute konzeptuelle Datenmodellierung durchgeführt, wird das erstellte Datenmodell bereits in 3NF sein.

Die Abbildung von einem ERM in ein RM wird typischerweise in sieben Schritten durchgeführt:

1. Reguläre Entitätstypen
2. Schwache Entitätstypen
3. 1:1-Relationshiptypen
4. 1:n-Relationshiptypen
5. n:m-Relationshiptypen
6. n:m:p-Relationshiptypen
7. Mehrwertige Attribute

1.4.3. PHYSISCHE DATENMODELLIERUNG

Im Rahmen der physischen Datenmodellierung werden administrative Aufgaben durchgeführt. Darunter fallen Aufgaben wie das Erzeugen von Zugriffspfaden für Optimierungen oder die Verteilung auf verschiedene DB-Server. Diese Aspekte werden in der Lehrveranstaltung nicht vertieft behandelt.

2. DIE DATENBANKSPRACHE FÜR RELATIONALE DBMS: SQL

SQL: Structured Query Language

Ein Relationales Datenbanksystem wird mit SQL-Anweisungen gesteuert. Es bearbeitet SQL-Anweisungen und antwortet in der Regel auf jede einzelne dieser Anweisungen.

Mögliche Antwortarten sind sinngemäß:
- Das Gewünschte wurde erfolgreich ausgeführt
- Das Gewünschte wurde aus folgendem Grund nicht ausgeführt: …
- Hier ist die gewünschte Information

SQL-Anweisungen werden meist in die folgenden Bereiche eingeteilt:

DDL: Data Definition Language
DML: Data Manipulation Language
DRL: Data Retrieval Language / Jedoch häufiger wird hier von Query bzw. Queries oder SQL-Anfragen bzw. Anfragen gesprochen. Eine SQL-Anweisung, die Information aus der Datenbank liest und für das gewünschte Ergebnis bearbeitet und aufbereitet wird als Query bezeichnet. Die Daten in der Datenbank werden dabei nicht verändert.
DCL: Data Control Language / Hier handelt es sich um die vielfältigen administrativen Aufgaben: Datenbanken anlegen, Benutzer anlegen und verwalten, Rechtevergabe, Speicherverwaltung, Definition von Wiederherstellungsmaßnahmen, und vieles mehr. Dieser Bereich ist nicht Teil der Vorlesung.

Im Folgenden werden die grundlegenden Prinzipien und Sprachkonzepte an einem durchgängigen Beispiel vorgestellt und somit die bereits durchgeführte Einführung vertieft. Der Schwerpunkt liegt hierbei auf dem Nutzen für das Üben von SQL- Anweisungen und dem „raschen“ Erlernen der Sprachkonzepte sowie dem „einfachen“ Erkennen der unterliegenden Prinzipien. Die Sinnhaftigkeit mancher Anweisungen und Anweisungsfolgen tritt hierbei in den Hintergrund.
Es dient insgesamt als Vorbereitung für das Datenbank-Projekt, welches im zweiten Teil der Lehrveranstaltung durchgeführt wird.
Obwohl alle relationalen Datenbanksysteme SQL verwenden (de facto Standard), gibt es kleine Unterschiede (z.B. varchar2 oder vchar). Alle folgenden SQL-Beispiele sind für das in der Vorlesung verwendete Datenbanksystem Oracle 11g bzw. Oracle 12c formuliert.

2.1. DDL

Data Definition Language

Unter dem Akronym DDL werden die Sprachanteile von SQL bzw. die Teilmenge der SQL-Anweisungen zusammengefasst, welche zur Definition der Datenstrukturen (Metadaten) genutzt werden.

Wichtigste Schlüsselworte: CREATE – DROP – ALTER

Als Faustregel gilt:
Was mit create erzeugt werden kann, kann mit drop entfernt und mit alter geändert werden.
Beispiele (DDL und DCL):
create database …, drop database …, alter database …
create user …, drop user …, alter user …
create type … , drop type …, alter type …

create view … , drop view …, alter view …
create table … , drop table …, alter table …
und viele mehr.

Die Daten werden in einem relationalen Datenbanksystem grundsätzlich in Tabellen (Relationen) verwaltet. Eine Tabelle ist eine Menge von Zeilen (Tupel).

Relevante Datenstrukturen **TABLE** und VIEW.
Views (Sichten auf Tabellen in Tabellenform) werden wir erst später behandeln.

Die Datentypen unterscheiden sich von DBS zu DBS. In den Beispielen wird Oracle 11g zugrunde gelegt, da es in der Vorlesung verwendet wird.

Beispiele zur Definition von Tabellen

Mindestens zu definieren: Tabellenname, eine Spalte (Spaltenname und Datentyp, d.h. Definition der erlaubten Werte für die Spalte)

Wichtige Datentypen:

Zeichenketten

- mit fester Länge: char(6)
- mit variabler Länge und Maximalwert: varchar2(15).
Wird kein Maximalwert für die Zeichenanzahl angegeben, gilt ein Defaultwert z.B. von 4000

Zahlen

- Natürliche Zahl (ohne Komma): number oder number(4)
Wird kein Maximalwert für die Stellenanzahl angegeben, gilt ein Defaultwert z.B. von 38
- Zahl mit Nachkommastellen: number(6,2) → insgesamt 6 Stellen davon sind 2 Nachkommastellen 1000.00

Datumswerte

- to_date(‘10.11.12‘, ‚DD-MM-YY‘)
oder to_date(‘11.10.12‘, ‚MM-DD-YY‘)
oder to_date(‘10.11.2012‘, ‚DD-MM-YYYY‘)
oder to_date(‘2012‘,‘YYYY‘) hier wird bspw. nur das Jahr verwendet. Weitere Infos → SQL-Reference

Wahrheitswerte

Es ist sinnvoll auf die explizite Angabe eines Boolean-Datentyps zu verzichten und stattdessen eine number(1) zu definieren, welche mit einer Bedingung auf die Werte 1 und 0 eingeschränkt wird.

Beispiel: Erzeugen einer einfachen Tabelle mit einer einzigen Spalte

```
CREATE TABLE Studierende (uNummer char(6))
```

Übrigens ist in Schlüsselworten die Groß- und Kleinschreibung vollkommen egal.
creAte, CREATE, Create, CrEaTe, creatE sind bspw. erlaubt.
Letztendlich ist die Lesbarkeit entscheidend.

Um eine Tabelle mit mehr als einer Spalte zu erzeugen, müssen für alle Spalten der Spaltenname und der Datentyp angegeben werden und diese Spaltendefinitionen müssen untereinander durch ein Komma getrennt werden.

```
CREATE TABLE Studierende (uNummer char(6), Vorname varchar2(15), Nachname
varchar2(15), Geburtstag date)
```

Dieser Befehl wird allerdings vom Datenbanksystem nicht ausgeführt werden, da eine Tabelle mit demselben Namen bereits vorhanden ist. Die Tabelle muss zunächst entfernt werden und neu erzeugt werden oder die Tabelle muss geändert werden.

```
DROP TABLE Studierende

CREATE TABLE Studierende (uNummer char(6),
Vorname varchar2(15), Nachname varchar2(15),
Geburtstag date)
```

```
ALTER Table Studierende add
(Vorname varchar2(15), Nachname
varchar2(15), Geburtstag date)
```

Der Befehl ALTER … zum Ändern einer Tabelle hätte ebenso verwendet werden können. Die Tabelle muss dazu vorhanden sein. Also wird kein „drop table“-Befehl benötigt.

Es sollen nun noch weitere Spalten in der Tabelle existieren:

```
DROP TABLE Studierende

CREATE TABLE Studierende (uNummer char(6), Vorname varchar2(15), Nachname
varchar2(15), Geburtstag date, Studiengang varchar2(5), Studienbeginn number(5), Gebuehr
number(5,2))

DROP TABLE Studierende
```

Formatierung der Anweisung für menschliche Bearbeitende optimiert:

```
CREATE TABLE Studierende (
uNummer char(6)
,Vorname varchar2(15)
,Nachname varchar2(15)
,Geburtstag date
,Studiengang varchar2(5)
,Studienbeginn number(5)
,Gebuehr number(5,2)
)

DROP TABLE Studierende
```

Für das Datenbanksystem existieren keine Zeilenumbrüche innerhalb von SQL-Anweisungen. Es erwartet eine Zeichenkette ohne diese unsichtbaren Sonder- bzw. Formatierungszeichen (z.B. Schriftfarbe und -größe). Daher ist es auch nicht sinnvoll SQL-Befehle bspw. aus doc-Formaten heraus zu kopieren, da dann die unsichtbaren Sonderzeichen z.B. Zeilenumbruch in der Zeichenkette bleiben und Fehler auslösen. In Textdateien sind diese Sonderzeichen nicht enthalten. Sie können die SQL-Anweisungen in Dateien kopieren und diese als Textdatei abspeichern, dabei werden die unerwünschten Zeichen entfernt …

CONSTRAINTS sind Integritätsbedingungen, die vom Datenbanksystem geprüft und gewährleitet werden. Die Werte (d.h. die Daten) in der Datenbank sollen mit der Miniwelt übereinstimmen (→ Integrität der Datenbank). Je genauer dazu Constraints (→ Integritätsbedingungen) definiert werden, umso besser wird dies automatisch vom Datenbanksystem gewährleistet werden können (→ system enforced integrity). Es kann bspw. das Einfügen von unpassenden Werten vom Datenbanksystem automatisch verweigert werden. Denken Sie daran, dass eine Datenbank in der Regel so viele Werte und Daten enthält, dass diese kaum oder gar nicht mehr nachträglich in sinnvoller Zeit geprüft und korrigiert werden können. Der Arbeits- und Forschungsbereich der sich mit dem Säubern von inkonsistenten Daten befasst heißt **Data Curation**. Am besten ist, man lässt dieses Problem gar nicht erst aufkommen.

Relevante Contraints: PRIMARY KEY, NOT NULL, UNIQUE, CHECK, FOREIGN KEY

Alle minimalen Mengen von Spalten, deren Werte einzelne Zeilen eindeutig identifizieren, werden als Kandidatenschlüssel bezeichnet. D.h. keine Wertkombination darf mehr als einmal in diesen Spalten der Tabelle vorkommen (UNIQUE), in jeder Zeile müssen diese Spalten zudem mit Werten belegt sein (NOT NULL) und es darf keine echte Teilmenge dieser Spalten existieren für die diese Definition erfüllt ist (MINIMALITÄT). Einer der Kandidatenschlüssel einer Tabelle wird als Primarschlüssel (PRIMARY KEY) ausgewählt und definiert. Die anderen Kandidatenschlüssel sollten natürlich als NOT NULL und UNIQUE definiert werden (siehe unten). Beachten Sie, ob eine Menge von Spalten ein Kandidatenschlüssel ist, wird immer durch die jeweilige Miniwelt bestimmt.
Tabellen können auch ohne die Angabe eines Primärschlüssels definiert werden. Dies ist in der Regel aber nicht sinnvoll (siehe oben: system enforced inegrity)

```
CREATE TABLE Studierende (
uNummer char(6) PRIMARY KEY
,Vorname varchar2(15)
,Nachname varchar2(15)
,Geburtstag date
,Studiengang varchar2(5)
,Studienbeginn number(5)
,Gebuehr number(5,2)
)
```

Inline-Definition.
Gelten mehrere Constraints für eine Spalte, können diese in beliebiger Reihenfolge durch ein Leerzeichen getrennt aufgezählt werden.
Beispiele weiter unten.

```
DROP TABLE Studierende
```

```
CREATE TABLE Studierende (
uNummer char(6)
,Vorname varchar2(15)
,Nachname varchar2(15)
,Geburtstag date
,Studiengang varchar2(5)
,Studienbeginn number(5)
,Gebuehr number(5,2)
,PRIMARY KEY(uNummer)
)
```

out-of-line-Definition.
Achtung: Immer erst alle Spalten definieren, danach die Constraints, welche out-of-line definiert werden. Mehrere Constraints werden durch ein Komma getrennt.

Die Reihenfolge der Constraints hat keine Bedeutung.

```
DROP TABLE Studierende

CREATE TABLE Studierende (
uNummer char(6)
,Vorname varchar2(15)  not null
,Nachname varchar2(15)  not null
,Geburtstag date not null
,Studiengang varchar2(5)
,Studienbeginn number(5)
,Gebuehr number(5,2)
,PRIMARY KEY(uNummer)
,UNIQUE(Vorname, Nachname, Geburtstag)
)
```

Bezieht sich ein Constraint auf mehrere Spalten, so muss dieser Constraint out-of-line definiert werden. Im Beispiel soll die Kombination von Vorname, Nachname, Geburtstag eindeutig und not null sein.

Eine Ausnahme bildet NOT NULL. Dieser Constraint muss immer inline definiert werden. Wenn mehrere Spalten not null sein müssen, dann gilt dies immer auch für jede einzelne Spalte. Daher wird dies immer in die Spaltendefinition (inline) aufgenommen.

Da es in diesem Beispiel zwei Kandidatenschlüssel gibt: (unummer) oder (Vorname, Nachname, Geburtstag) könnte auch (Vorname, Nachname, Geburtstag) als Primary Key definiert werden. Wir gehen davon aus, dass (Vorname, Nachname, Geburtstag) in unserer Miniwelt minimal sind, d.h. es mehrere Personen geben kann, mit demselben Vornamen, Nachnamen oder Geburtstag sowie auch mit gleichem Vor- und Nachnamen oder gleichem Vornamen und Geburtstag oder auch gleichem Nachnamen und Geburtstag. Und natürlich kann ein Vorname mehrmals vorkommen, ein Nachname sowie ein Geburtstag ebenso. Die Definition eines zusammengesetzten Primärschlüssel muss immer Out-of-Line geschehen. Als zusammengesetzter Primärschlüssel wird ein Primärschlüssel bezeichnet, welcher aus mehr als einer Spalte besteht.

```
DROP TABLE Studierende

CREATE TABLE Studierende (
uNummer char(6) not null uniqiue
,Vorname varchar2(15)
,Nachname varchar2(15)
,Geburtstag date
,Studiengang varchar2(5)
,Studienbeginn number(5)
,Gebuehr number(5,2)
, PRIMARY KEY(Vorname, Nachname, Geburtstag)
)
```

Besteht ein Primary Key aus mehr als einer Spalte, muss dieser Constraint Out-of-Line definiert werden, da das Schlüsselwort nur einmal vorkommen darf und sich die inline-Definition immer nur auf die entsprechende Zeile bezieht. Dies gilt ebenso für andere Constraints, die sich auf mehrere Spalten beziehen (siehe oben).

Beispiel UNIQUE(Vorname, Nachname, Geburtstag)

```
DROP TABLE Studierende
```

Jedem Constraint wird grundsätzlich im Datenbanksystem ein eigener eindeutiger Name gegeben. Dies geschieht entweder automatisch durch das Datenbanksystem oder explizit durch den Benutzer. Dazu ist das Schlüsselwort CONSTRAINT mit dem gewünschten Namen in der Constraint-Definition

jeweils zu Beginn anzugeben. In einem CREATE TABLE Befehl können explizite und automatische Namensgebungen gemischt vorkommen.

```
CREATE TABLE Studierende (
uNummer char(6) CONSTRAINT Stud_UQ unique CONSTRAINT Stud_NN not null
,Vorname varchar2(15)
,Nachname varchar2(15)
,Geburtstag date
,Studiengang varchar2(5)
,Studienbeginn number(5)
,Gebuehr number(5,2)
, CONSTRAINT Stud_PK PRIMARY KEY(Vorname, Nachname, Geburtstag)
)

DROP TABLE Studierende

CREATE TABLE Studierende (
uNummer char(6) CONSTRAINT Stud_NN not null
,Vorname varchar2(15)
,Nachname varchar2(15)
,Geburtstag date
,Studiengang varchar2(5)
,Studienbeginn number(5)
,Gebuehr number(5,2)
, CONSTRAINT Stud_PK PRIMARY KEY(Vorname, Nachname, Geburtstag)
, CONSTRAINT Stud_UQ unique (unummer)
)

DROP TABLE Studierende
```

In einem Befehl können explizite und automatische Namensgebungen gemischt vorkommen.

```
CREATE TABLE Studierende (
uNummer char(6) CONSTRAINT Stud_NN not null
,Vorname varchar2(15)
,Nachname varchar2(15)
,Geburtstag date
,Studiengang varchar2(5)
,Studienbeginn number(5)
,Gebuehr number(5,2)
, CONSTRAINT Stud_PK PRIMARY KEY(Vorname, Nachname, Geburtstag)
, unique (unummer)
)
```

Constraints können aus Tabellen mit dem ALTER TABLE Befehl entfernt werden. Es ist auch möglich Constraints kurzfristig an- und auszuschalten (ENABLE bzw. DISABLE).
In jedem Fall ist es dazu notwendig den Namen des Constraints zu kennen. Die Namen der Constraints einer Tabelle lassen sich aus dem Data Dictionary (→ Metadaten) herauslesen. Einfacher ist es natürlich, wenn der Name selbst explizit vergeben wurde und somit bereits bekannt ist.

Vereinfachen wir die Tabelle kurzfristig:

```
DROP TABLE Studierende

CREATE TABLE Studierende (
uNummer char(6) CONSTRAINT Stud_PK PRIMARY KEY
,Vorname varchar2(15)
,Nachname varchar2(15)
,Geburtstag date
,Studiengang varchar2(5)
,Studienbeginn number(5)
,Gebuehr number(5,2)
)
```

Beim Befehl ALTER TABLE wird der Constraint-Name verwendet:

```
ALTER TABLE Studierende DROP CONSTRAINT Stud_PK

ALTER TABLE Studierende ADD CONSTRAINT Stud_PK PRIMARY KEY(Vorname, Nachname, Geburtstag)

ALTER TABLE Studierende DROP CONSTRAINT Stud_PK

ALTER TABLE Studierende ADD CONSTRAINT Stud_PK PRIMARY KEY(UNUMMER)

ALTER TABLE Studierende ADD CONSTRAINT Stud_UQ UNIQUE (Vorname, Nachname, Geburtstag)
```

Was ist passiert? ☺

Not Null kann nicht explizit benannt werden, da kein out-of-line constraint möglich ist (siehe oben). Der folgende Befehl ist demnach nicht möglich:

```
ALTER TABLE Studierende ADD CONSTRAINT Stud_NachnameNN Not Null (Nachname)
```

Bisher wurden die Constraints not null, unique, primary key betrachtet. Es fehlen noch die Constraints check und foreign key.

Mit einer check-Bedingung (check-Constraint) können die Werte für eine oder mehrere Spalten eingeschränkt werden. Das ist sinnvoll, da es vorteilhaft ist, wenn das Datenbanksystem Eingaben oder Änderungswünsche genauer prüfen kann und ggfs. falsche Werte öder Änderungen abweisen kann, welche die Konsistenz oder die Integrität der Datenbank gefährden.

```
DROP TABLE Studierende
```

```
CREATE TABLE Studierende (
uNummer char(6) CONSTRAINT Stud_NN not null
,Vorname varchar2(15)  not null
,Nachname varchar2(15)  not null
,Geburtstag date
,Studiengang varchar2(5) CONSTRAINT STUD_CHK check in ('BWING','WING', 'INF', 'WINF',
'MINF','IINF') not null
,Studienbeginn number(5)
,Gebuehr number(5,2))
, CONSTRAINT Stud_PK PRIMARY KEY(Vorname, Nachname, Geburtstag)
, CONSTRAINT Stud_UQ unique(UNUMMER))

ALTER TABLE Studierende ModiFY CONSTRAINT STUD_CHK check in (,BWING','WING',
,INF', ,WINF', ,MINF','IINF', ,EADM')
```

Das viele Löschen und Erzeugen einer Tabelle sollte man in der Praxis durch vorheriges gutes Planen der Datenbank auf ein Minimum beschränken (→ Dies werden wir in unserem Datenbankprojekt genauer beleuchten). Und im späteren laufenden Betrieb der Datenbank, wenn die Daten bereits mit aktuellen Daten gefüllt sind, ist dies sowieso nur noch eingeschränkt möglich ;-) Aber beim Üben ist natürlich alles erlaubt – und Übung macht den Meister.

```
CREATE TABLE STUDIENGANG (
ID varchar2(5) PRIMARY KEY
,Name varchar2(30)
,Abschluss varchar2(6) check Abschluss in (,B.Eng.', ,B.Sc.')
)

ALTER TABLE Studierende drop CONSTRAINT STUD_CHK

ALTER TABLE Studierende ADD CONSTRAINT Stud_FK FOREIGN KEY(Studiengang)
REFERENCES STUDIENGANG(ID)
```

Mit einem Fremdschlüssel kann lediglich referenziert werden auf einen Kandidatenschlüssel einer anderen oder der gleichen Tabelle, d.h. die entsprechenden Spalten müssen unique und not null sein bzw. als primary key definiert sein. Beachten Sie, der Primärschlüssel ist natürlich auch ein Kandidatenschlüssel

Die Datentypen der jeweiligen referenzierenden und referenzierten Spalten müssen exakt übereinstimmen.

Als Eingabe in der Tabelle Studierende für die Spalte Studiengang sind nur Werte erlaubt, die bereits in der Tabelle Studiengang als ID enthalten sind (→ Referentielle Integrität).

Das bedeutet in diesem Beispiel zudem, dass die check-Bedingung, die die erlaubten Werte des Studiengangs definiert hatte, nicht mehr sinnvoll und notwendig ist und daher nicht mehr definiert werden sollte. Die check-Bedingung STUD_CHK wurde daher zuerst gelöscht.

In einer Tabelle können mehrere Fremdschlüssel definiert sein.

Als Beispiel ergänzen wir zusätzlich einen Fremdschlüssel auf einen Studiengangssprecher zu den Studiengängen, welcher auf den Primarschlüssel der Tabelle Studierende verweisen soll. Man beachte, dass sowohl Fremdschlüssel als auch Primärschlüssel aus drei Attributen/Spalten bestehen.

Ergänzen von Spalten in einer Tabelle ist mit dem Alter Befehl möglich:

```
ALTER TABLE Studiengang add Studiengangssprechervorname varchar2(15)

ALTER TABLE Studiengang add Studiengangssprechernachname varchar2(15)

ALTER TABLE Studiengang add Studiengangssprechergeb date

ALTER TABLE Studiengang ADD CONSTRAINT Stud_FK FOREIGN
KEY(Studiengangssprechervorname, Studiengangssprechernachname,
Studiengangssprechergeb) REFERENCES Studierende (vorname, nachname, geburtsdatum)
```

Wir könnten aber ebenso aus der Tabelle Studiengang lediglich auf den Kandidatenschlüssel unummer der Tabelle Studierender verweisen

```
ALTER TABLE Studiengang drop CONSTRAINT Stud_FK

ALTER TABLE Studiengang drop Studiengangssprechervorname

ALTER TABLE Studiengang drop Studiengangssprechernachname

ALTER TABLE Studiengang drop Studiengangssprechergeb

ALTER TABLE Studiengang add Studiengangssprecherunummer char(6)

ALTER TABLE Studiengang ADD CONSTRAINT Stud_FK FOREIGN
KEY(Studiengangssprecherunummer) REFERENCES Studierende (unummer)

ALTER TABLE Studiengang RENAME Studiengaenge
```

2.2. DML

Wichtigste Schlüsselworte: INSERT INTO, DELETE FROM, UPDATE

INSERT INTO: Mit dem Insert-Befehl werden ganze Zeilen in Tabellen eingefügt.
DELETE FROM: Mit dem Delete-Befehl werden ganze Zeilen aus einer Tabelle gelöscht.
UPDATE: Und mit dem Update-Befehl werden in Tabellen vorhandene Zeilen geändert bzw. Werte in diesen Zeilen werden aktualisiert.

Beim Einfügen von Zeilen in eine Tabelle muss der Tabellenname immer genannt werden und:
- Es werden entweder die Werte explizit aufgelistet, die in der einzufügenden Zeile enthalten sind oder
- die Zeilen die eingefügt werden sollen, werden als Ergebnis einer SELECT-Anfrage ermittelt.

Da wie bereits besprochen eine SELECT-Anfrage immer eine Tabelle als Ergebnis liefert und eine Tabelle eine Menge von Zeilen darstellt, muss in diesem Fall lediglich sichergestellt sein, dass die

Zeilen aus dem Anfrageergebnis zu der Tabelle, in welche diese Zeilen eingefügt werden sollen, passen, z.B. Werteanzahl und Datentypen. Das Einfügen unter Verwendung einer SELECT-Anfrage wird erst später in dem Abschnitt DRL beschrieben.

Beim Einfügen von einer Zeile mit Auflistung der einzelnen Werte gilt:

- Zeichenketten werden in einfachen Hochkommas eingeschlossen:
'Dies ist eine Zeichenkette'

- Zahlen werden ohne Hochkomma angegeben: 13
Ein Komma wird innerhalb von Zahlen als Punkt geschrieben: 13.50
Durch die Verwendung der amerikanischen Schreibweise wird eine Verwechslung mit den trennenden Kommas zwischen den aufgezählten Werten vermieden.

- Datumswerte werden mit einer Funktion und der entsprechenden Formatangabe eingefügt:
to_date (Datum, Format)
Es muss klar sein, welche Zahlen jeweils Tag, Monat, Jahr oder sogar Stunde, Minute oder Sekunde darstellen. Der 10.11.2012 wird beispielsweise in amerikanischer Schreibweise als 11/10/2012 dargestellt.
Entsprechend kann beim Lesen eines Datumwertes aus der Datenbank (siehe Abschnitt DRL) mit einer Funktion das gewünschte Ausfgabeformat angegeben werden:
to_char(Datum, Format).

```
Insert into Studierende values ('u13131','Tom','Schlau',to_date('13.11.1991','DD-MM-YYYY'),'BWING',20152,10.50)
```

Dieser Befehl wird allerdings nicht vom Datenbanksystem durchgeführt werden, da im Fremdschlüssel (Studiengang) ein Wert steht ('BWING'), welcher im referenzierten Primarschlüssel (Studiengang(ID)) noch nicht vorhanden ist. Zunächst muss dieser Studiengang eingetragen werden:

```
Insert into Studiengang values ('BWING', 'Wirtschaftsingenieurwesen Bachelor berufsbgleitend', 'B.Eng.', null)
```

Es kann für diesen Studiengang noch kein Studiengangssprecher eingetragen werden, da es noch keine Studierenden gibt!
Mit **null** wird ein fehlender Wert bezeichnet.

Allerdings ist die Zeichenkette für den Studiengang zu lang. Um den Studiengang eintragen zu können, muss für die definierte variable Zeichenkette ein höherer Maximalwert bestimmt werden:

```
alter table Studiengang Modify (Name varchar2(60))

Insert into Studiengang values ('BWING','Wirtschaftsingenieurwesen Bachelor berufsbgleitend','B.Eng.', null)

Insert into Studierende values ('u13131','Tom','Schlau',to_date('13.11.1991','DD-MM-YYYY'),'BWING',20152,10.50)
```

Anzeige vom gesamten Inhalt einer Tabelle mit der folgenden Anfrage (→ weitere Anfragen werden im Abschnitt DRL behandelt)

Select * from Studiengang

bzw.

Select * from Studierende

Die bisher mit den Insert-Befehlen eingefügten Inhalte sieht man jedoch lediglich selbst und sie sind auch noch nicht dauerhaft in der Datenbank eingefügt. Damit das bisher Eingefügte wirklich für alle sichtbar wird und die Änderungen dauerhaft auf der Datenbank durchgeführt werden, muss die aktuelle Transaktion erfolgreich mit einem commit-Befehl beendet werden (→ siehe ACID-Transaktionen. Dies wird später noch detailliert behandelt werden):

Commit

Bei den DDL-Anweisungen ist der sql developer so eingestellt, dass nach jeder Anweisung automatisch ein commit-Befehl durchgeführt wird. Daher wurden diese Definitionen und Änderungen sofort für alle sichtbar und dauerhaft durchgeführt.

Der Abbruch einer Transaktion wird mit dem Befehl rollback angefordert. Eine Transaktion wird ganz oder gar nicht durchgeführt. Alle noch nicht mit einem erfolgreichen commit auf der Datenbank durchgeführten Änderungen, werden entweder beim Schließen des sql developers oder mit einem explizit angewiesenen rollback, wieder zurückgenommen.

Rollback

Da als letzter Befehl vor dem rollback-Befehl ein commit durchgeführt wurde, hat der rollback-Befehl keine Auswirkung, da keine Änderungen zurück genommen werden können.

Wichtige Schlüsselworte: commit, rollback

Insert into Studiengang values (‘WING‘, ‘Wirtschaftsingenieurwesen Bachelor‘, ‘B.Eng.‘, null)

Select * from Studiengang

Durch dieses Insert wird eine neue Transaktion begonnen. Will man eine dauerhafte und für alle sichtbare Änderung durchführen:

commit

Ein commit-Befehl kann auch nach mehreren Anweisungen durchgeführt werden. Je nachdem ob das commit erfolgreich ist, werden dann alle betroffenen Anweisungen gemeinsam durchgeführt oder keine wird durchgeführt, d.h. alle werden zurückgenommen.

Für den Studiengang BWING kann der Studierende mit der unummer u13131 eingetragen werden. Dazu muss die Zeile aktualisiert werden:

Update Studiengang set Studiengangssprecherunummer = ‘u13131‘

Select * from Studiengang

Da keine where-Bedingung existiert, werden alle Zeilen geändert. Das ist nicht korrekt. Es gibt nun mehrere Möglichkeiten. Es kann ein rollback durchgeführt werden.

Rollback

Oder alle Zeilen werden aktualisiert, indem die Zellen der Spalte Studiengangssprecher in allen Zeilen geleert werden, d.h. der Studiengangssprecher auf „keinen Wert“ gesetzt wird.

```
Update Studiengang set Studiengangssprecherunummer = null
```

Nachsehen:

```
Select * from Studiengang
```

Schließlich wird die Aktualisierung angewiesen für alle Zeilen, deren ID den Wert BWING hat. Dies trifft nur auf eine Zeile zu. Da die ID als Primärschlüssel definiert ist, kann dieser Wert auch nur in einer Zeile vorkommen.

```
Update Studiengang set Studiengangssprecherunummer = ‘u13131‘ where ID = ‘BWING‘

Commit

Insert into Studierende values (‘u13131‘,‘Tom‘,‘Schlau‘,to_date(‘13.11.1991‘,‘DD-MM-YYYY‘),‘BWING‘,20152,10.50)
```

Ist die Reihenfolge der Spalten nicht bekannt, können die Spalten namentlich aufgezählt werden:

```
Insert into Studierende (unummer, nachname, geburtstag, vorname, studiengang, studienbeginn, gebuehr) values (‘u11111‘,‘Superschlau‘,to_date(‘13.11.1991‘,‘DD-MM-YYYY‘),‘Tom‘, ‘BWING‘,20152,10.50)
```

Man beachte hier, dass die Reihenfolge der Spalten egal ist. Allerdings müssen die Werte zu den Spalten passen.

Spalten, die nicht aufgezählt werden, werden implizit mit null belegt:

```
Insert into Studierende (unummer, nachname, geburtstag, vorname, studiengang) values (‘u11113‘,‘Tim‘,‘Schlau‘,to_date(‘13.11.1991‘,‘DD-MM-YYYY‘),‘BWING‘)
```

Für Spalten, die nicht aufgezählt werden und die mit null belegt werden sollen, muss natürlich die Eingabe von null d.h. die Eingabe von keinem Wert erlaubt sein.

```
Insert into Studierende (unummer, studiengang, studienbeginn, gebuehr) values (‘u11112‘,‘BWING‘,20152,10.50)
```

Dieser Befehl wurde nicht ausgeführt. Aber der folgende Befehl wird ausgeführt:

```
Insert into Studierende (unummer, vorname, nachname, geburtstag) values (‘u11112‘,‘Tim‘,‘Schlauer‘,to_date(‘13.11.1991‘,‘DD-MM-YYYY’))

delete from Studierende where unummer like 'u1111_'
```

In der beiliegenden Datei sind weitere Insert-Befehle enthalten. Sodass die Anfragen im nächsten Abschnitt auf Tabellen ausgeführt werden können, die mehrere Zeilen enthalten.

Bisher wurden die Befehle immer einzeln ausgeführt. Dies war erwünscht, da jeder einzelne Befehl und seine Auswirkung genau betrachtet werden sollten. Nun können mehrere Befehle zusammen an das Datenbanksystem übermittelt werden, so dass mehrere Zeilen in die Tabellen eingefügt werden.

Sollen mehrere Befehle ausgeführt werden, dann müssen diese durch ein Semikolon voneinander getrennt werden.

Beispiele zum Einfügen von Studierenden und Studiengängen:

```
Insert into Studierende values ('u13121','Max','Dunkel',to_date('13.12.1991','DD-MM-YYYY'),'BWING',20152,10.50);

Insert into Studierende values ('u13122','Lu','Lustig',to_date('11.01.1992','DD-MM-YYYY'),'BWING',20152,10.50);

Insert into Studierende values ('u13123','Mo','Helle',to_date('13.02.1991','DD-MM-YYYY'),'BWING',20152,10.50);

Insert into Studierende values ('u13124','Uli','Exakt',to_date('11.04.1992','DD-MM-YYYY'),'BWING',20152,10.50);

Insert into Studierende values ('u13125','Ali','Genau',to_date('12.05.1989','DD-MM-YYYY'),'BWING',20152,10.50);

Insert into Studierende values ('u13126','Olli','Wild',to_date('01.06.1995','DD-MM-YYYY'),'BWING',20152,10.50);

Insert into Studierende values ('u13127','Oli','Kräftig',to_date('31.08.1993','DD-MM-YYYY'),'BWING',20152,10.50);

Insert into Studierende values ('u13128','Alex','Mutig',to_date('08.08.1990','DD-MM-YYYY'),'BWING',20152,10.50);

Insert into Studierende values ('u13129','Sara','Rasch',to_date('30.06.1991','DD-MM-YYYY'),'BWING',20152,10.50);

Insert into Studierende values ('u13111','Till','Golden',to_date('13.02.1991','DD-MM-YYYY'),'BWING',20152,10.50);

Insert into Studierende values ('u13112','Mia','Silber',to_date('11.05.1992','DD-MM-YYYY'),'BWING',20152,10.50);

Insert into Studierende values ('u13113','Ker','Braun',to_date('23.02.1991','DD-MM-YYYY'),'BWING',20152,10.50);

Insert into Studierende values ('u13114','Kia','Schwarz',to_date('21.04.1992','DD-MM-YYYY'),'BWING',20152,10.50);

Insert into Studierende values ('u13115','Ke','Rot',to_date('22.05.1989','DD-MM-YYYY'),'BWING',20152,10.50);

Insert into Studierende values ('u13116','Lu','Weiss',to_date('21.06.1995','DD-MM-YYYY'),'BWING',20152,10.50);

Insert into Studierende values ('u13117','Pete','Stark',to_date('21.08.1993','DD-MM-YYYY'),'BWING',20152,10.50);
```

```
Insert into Studierende values ('u13118','Lou','Rasend',to_date('09.08.1990','DD-MM-YYYY'),'BWING',20152,10.50);

Insert into Studierende values ('u13119','Jan','Neugierig',to_date('10.06.1991','DD-MM-YYYY'),'BWING',20152,10.50);
```

Und nun betrachten wir das Ergebnis der Anweisungen, indem wir die beiden Tabellen ausgeben lassen:

```
Select * from Studiengang;

Select * from Studierende;
```

2.3. DRL

Wichtigste Schlüsselworte (Reihenfolge bitte unbedingt einprägen):

```
SELECT
FROM
WHERE
GROUP BY
HAVING
ORDER BY
```

```
select * from Studierende
```

Ausgabe der gesamten Tabelle Studierende. Eine Tabelle besteht aus einer Menge von Zeilen. Diese haben keine Ordnung. Die Ausgabe der Zeilen einer Tabelle geschieht in der Reihenfolge, die gerade am schnellsten vom Datenbanksystem berechnet werden kann. Nur wenn eine bestimmte Ausgabereihenfolge notwendig ist, kann diese durch order by bestimmt werden. Das Sortieren ist grundsätzlich zeitaufwendig, insbesondere, wenn große Mengen sortiert werden müssen (→ Komplexität von Sortieralgorithmen).

➔ Nur Sortieren, wenn notwendig.

```
select *
from Studierende
order by unummer
```

unummer	Vorname	Nachname	Geburtstag	Studiengang	Studienbeginn	Gebuehr
u13111	Till	Golden	13.02.91	BWING	20152	10.5
u13112	Mia	Silber	11.05.92	BWING	20152	10.5
u13113	Ker	Braun	23.02.91	BWING	20152	10.5
u13114	Kia	Schwarz	21.04.92	BWING	20152	10.5
u13115	Ke	Rot	22.05.89	BWING	20152	10.5
u13116	Lu	Weiss	21.06.95	BWING	20152	10.5
u13117	Pete	Stark	21.08.93	BWING	20152	10.5
u13118	Lou	Rasend	09.08.90	BWING	20152	10.5
u13119	Jan	Neugierig	10.06.91	BWING	20152	10.5
u13121	Max	Dunkel	13.12.91	BWING	20152	10.5
u13122	Lu	Lustig	11.01.92	BWING	20152	10.5
u13123	Mo	Helle	13.02.91	BWING	20152	10.5
u13124	Uli	Exakt	11.04.92	BWING	20152	10.5

u13125	Ali	Genau	12.05.89	BWING	20152	10.5
u13126	Olli	Wild	01.06.95	BWING	20152	10.5
u13127	Oli	Kräftig	31.08.93	BWING	20152	10.5
u13128	Alex	Mutig	08.08.90	BWING	20152	10.5
u13129	Sara	Rasch	30.06.91	BWING	20152	10.5

→ 18 Zeilen gewählt

```
select *
from Studierende
order by Nachname, Vorname
```

```
select *
from Studierende
order by Nachname asc, Vorname desc
```

```
select *
from Studierende
order by Nachname desc, Vorname desc
```

asc → aufsteigend
desc → absteigend
nulls first → Null-Werte zuerst
nulls last→ Null-Werte zuletzt

Es können Spaltennamen aus den Eingabetabellen als auch der Ausgabetabelle angegeben werden.
Nach der zweiten Spalte wird erst sortiert, wenn die erste Spalte gleiche Werte hat, usw.: Schneider Alex, Schneider Max

```
select unummer, Nachname
from Studierende
```

Projektion der Spalten unummer und Nachname

unummer	Nachname
u13121	Dunkel
u13122	Lustig
u13123	Helle
u13124	Exakt
u13125	Genau
u13126	Wild
u13127	Kräftig
u13128	Mutig
u13129	Rasch
u13111	Golden
u13112	Silber
u13113	Braun
u13114	Schwarz
u13115	Rot
u13116	Weiss
u13117	Stark
u13118	Rasend
u13119	Neugierig

```
select unummer NUMMER, nachname NAME
from Studierende
```

```
select unummer nummer, nachname Name
from Studierende
```

Projektion der Spalten unummer und nachname. Die Ergebnisspalten werden in NUMMER und NAME umbenannt. Die Einstellung des Datenbanksystems führt zu Darstellung der Ergebnisspalten in

Großbuchstaben. Soll die Schreibweise der Ergebnisspalten beibehalten werden schließt man den neuen Spaltennamen in doppelte Hochkomma ein (→ siehe nächstes Beispiel):

NUMMER	NAME
u13121	Dunkel
u13122	Lustig
u13123	Helle
u13124	Exakt
u13125	Genau
u13126	Wild
u13127	Kräftig
u13128	Mutig
u13129	Rasch
u13111	Golden
u13112	Silber
u13113	Braun
u13114	Schwarz
u13115	Rot
u13116	Weiss
u13117	Stark
u13118	Rasend
u13119	Neugierig

select unummer nummer, nachname “Name“ from Studierende

Projektion der Spalten unummer und Nachname. Die Ergebnisspalten werden in NUMMER und Name umbenannt. Damit die Groß- und Kleinschreibung der gewünschten Bezeichnungen der Ergebnisspalten erhalten bleibt, schließt man die Spaltennamen in doppelte Hochkomma ein:

NUMMER	Name
u13121	Dunkel
u13122	Lustig
u13123	Helle
u13128	Mutig
u13125	Genau
u13126	Wild
u13127	Kräftig
u13129	Rasch
u13111	Golden
u13112	Silber
u13113	Braun
u13114	Schwarz
u13115	Rot
u13116	Weiss
u13117	Stark
u13118	Rasend
u13119	Neugierig
u13124	Exakt

select unummer nummer, nachname „Der Name des/r Studierenden“ from Studierende

Projektion der Spalten unummer und Nachname. Die Ergebnisspalten werden in NUMMER und „Der Name des/r Studierenden“ umbenannt:

NUMMER	Der Name des/r Studierenden
u13121	Dunkel
u13122	Lustig
u13123	Helle
u13128	Mutig

u13125	Genau
u13126	Wild
u13127	Kräftig
u13129	Rasch
u13111	Golden
u13112	Silber
u13113	Braun
u13114	Schwarz
u13115	Rot
u13116	Weiss
u13117	Stark
u13118	Rasend
u13119	Neugierig
u13124	Exakt

Informationen können aus mehreren Spalten ausgelesen und zusammengefügt oder miteinander verrechnet werden. Im folgenden Beispiel werden die Zeichenketten aus den beiden Spalten zusammengefügt und in einer Ergebnisspalte ausgegeben. Es wird vergessen ein Leerzeichen dazwischen zu setzen.

```
select unummer, vorname ||  nachname from Studierende
```

unummer	vorname||nachname
u13111	TillGolden
u13112	MiaSilber
u13113	KerBraun
u13114	KiaSchwarz
u13115	KeRot
u13116	LuWeiss
u13117	PeteStark
u13118	LouRasend
u13119	JanNeugierig
u13121	MaxDunkel
u13122	LuLustig
u13123	MoHelle
u13124	UliExakt
u13125	AliGenau
u13126	OlliWild
u13127	OliKräftig
u13128	AlexMutig
u13129	SaraRasch

Mit einem Leerzeichen:

```
select unummer, vorname || ' ' || nachname from Studierende
```

unummer	vorname|| ‘ ‘ ||nachname
u13111	Till Golden
u13112	Mia Silber
u13113	Ker Braun
u13114	Kia Schwarz
u13115	Ke Rot
u13116	Lu Weiss
u13117	Pete Stark
u13118	Lou Rasend
u13119	Jan Neugierig
u13121	Max Dunkel
u13122	Lu Lustig
u13123	Mo Helle
u13124	Uli Exakt
u13125	Ali Genau
u13126	Olli Wild
u13127	Oli Kräftig
u13128	Alex Mutig

u13129	Sara Rasch

Umbenennen der Ergebnisspalte in Name:

```
select unummer, vorname || ' ' || nachname “Name“ from Studierende
```

unummer	Name
u13111	Till Golden
u13112	Mia Silber
u13113	Ker Braun
u13114	Kia Schwarz
u13115	Ke Rot
u13116	Lu Weiss
u13117	Pete Stark
u13118	Lou Rasend
u13119	Jan Neugierig
u13121	Max Dunkel
u13122	Lu Lustig
u13123	Mo Helle
u13124	Uli Exakt
u13125	Ali Genau
u13126	Olli Wild
u13127	Oli Kräftig
u13128	Alex Mutig
u13129	Sara Rasch

Weitere Beispiele. Wie sieht wohl das Ergebnis aus?

```
Select ‘Der/die Studierende ‘ || vorname || '  ' || nachname || ‘ hat die unummer ‘ || unummer || ‘ und ist im Studiengang ‘ || Studiengang || ‘ eingeschrieben‘ “Information über Studierende“ from Studierende
```

```
select ‘Der Nachname des Studierenden mit der unummer ‘|| unummer || ‘ hat ‘ || length(nachname) || ‘ Buchstaben ‘ “Information über Studierende“ from Studierende
```

```
select ‘Der Nachname des Studierenden mit der unummer ‘|| unummer || ‘ hat ‘ || length(nachname)-length(Vorname) || ‘ Buchstaben mehr als der Vorname‘ “Information über Studierende“ from Studierende
```

Falls das wichtig wäre zu wissen ☺

Die nächsten 3 Beispiele dienen zur Erinnerung:

```
select unummer, vorname || ' ' || nachname Name, 7, 'Lalala' from Studierende
```

unummer	Name	7	LALALA
u13111	Till Golden	7	Lalala
u13112	Mia Silber	7	Lalala
u13113	Ker Braun	7	Lalala
u13114	Kia Schwarz	7	Lalala
u13115	Ke Rot	7	Lalala
u13116	Lu Weiss	7	Lalala
u13117	Pete Stark	7	Lalala
u13118	Lou Rasend	7	Lalala
u13119	Jan Neugierig	7	Lalala
u13121	Max Dunkel	7	Lalala
u13122	Lu Lustig	7	Lalala
u13123	Mo Helle	7	Lalala

u13124	Uli Exakt	7	Lalala
u13125	Ali Genau	7	Lalala
u13126	Olli Wild	7	Lalala
u13127	Oli Kräftig	7	Lalala
u13128	Alex Mutig	7	Lalala
u13129	Sara Rasch	7	Lalala

```
select unummer, vorname || ' ' || nachname Name, 7 Sieben , 'Lalala' from Studierende
```

unummer	Name	Sieben	LALALA
u13111	Till Golden	7	Lalala
u13112	Mia Silber	7	Lalala
u13113	Ker Braun	7	Lalala
u13114	Kia Schwarz	7	Lalala
u13115	Ke Rot	7	Lalala
u13116	Lu Weiss	7	Lalala
u13117	Pete Stark	7	Lalala
u13118	Lou Rasend	7	Lalala
u13119	Jan Neugierig	7	Lalala
u13121	Max Dunkel	7	Lalala
u13122	Lu Lustig	7	Lalala
u13123	Mo Helle	7	Lalala
u13124	Uli Exakt	7	Lalala
u13125	Ali Genau	7	Lalala
u13126	Olli Wild	7	Lalala
u13127	Oli Kräftig	7	Lalala
u13128	Alex Mutig	7	Lalala
u13129	Sara Rasch	7	Lalala

```
select unummer, vorname || ' ' || nachname name, 6+1, 8*7 „Name“ from Studierende
```

unummer	Name	6+1	Name
u13111	Till Golden	7	56
u13112	Mia Silber	7	56
u13113	Ker Braun	7	56
u13114	Kia Schwarz	7	56
u13115	Ke Rot	7	56
u13116	Lu Weiss	7	56
u13117	Pete Stark	7	56
u13118	Lou Rasend	7	56
u13119	Jan Neugierig	7	56
u13121	Max Dunkel	7	56
u13122	Lu Lustig	7	56
u13123	Mo Helle	7	56
u13124	Uli Exakt	7	56
u13125	Ali Genau	7	56
u13126	Olli Wild	7	56
u13127	Oli Kräftig	7	56
u13128	Alex Mutig	7	56
u13129	Sara Rasch	7	56

Kommen wir nun zur where-Bedingung. Die Where-Bedingung für jede Zeile der Eingabemenge ausgewertet. Das Ergebnis (Wahr oder Falsch) dieser Auswertung entscheidet, ob die jeweilige Zeile für die Berechnung des Ergebnisses berücksichtigt wird oder ob die Zeile ignoriert wird.

```
select unummer, vorname || ' ' || nachname, 7+1, 8*7 Name
from Studierende
where nachname like 'S%'
```

unummer	Name	6+1	Name

u13112	Mia Silber	7	56
u13114	Kia Schwarz	7	56
u13117	Pete Stark	7	56

```
select unummer, vorname || ' ' || nachname, 7+1, 8*7 Name
from Studierende
where nachname like 'S%' and vorname like 'M%'
```

unummer	Name	6+1	Name
u13112	Mia Silber	7	56

```
select unummer, vorname || ' ' || nachname, 7+1, 8*7 Name
from Studierende
where nachname like 'S%' and vorname like 'M%' or vorname like 'A%'
```

unummer	Name	6+1	Name
u13112	Mia Silber	7	56
u13125	Ali Genau	7	56
u13128	Alex Mutig	7	56

Typische Aggregatfunktionen:
COUNT(*), MIN(SpalteX), MAX(SpalteX), SUM((SpalteX), AVG((SpalteX), MEDIAN((SpalteX)

Aggregatfunktionen können Spaltenwerte mehrere Zeilen gleichzeitig in Berechnungen einbeziehen. Wenn keine GROUP BY Klausel angegeben wird, dann werden alle Zeilen der Tabelle in die Berechnung einbezogen.

```
select count(*)  from Studierende
```

COUNT(*)
18

```
select count(*) “Anzahl” from Studierende
```

Anzahl
18

```
select count(*) “Anzahl Studierende” from Studierende
```

Anzahl Studierende
18

```
select count(*) Anzahl, min (Geburtstag) Name from Studierende
```

Anzahl	min(Geburtstag)
18	12.05.89

Mit GROUP BY werden Gruppen (disjunkte Teilmengen) von Zeilen gebildet, für die jeweils eine Ergebniszeile berechnet wird und dabei eine oder mehrere Aggregatfunktionen ausgewertet werden.

Alle Zeilen, die in der Spalte Studiengang den gleichen Werte enthalten bilden jeweils eine Gruppe:

```
select count(*) “Anzahl” from Studierende group by Studiengang
```

Anzahl
18

Wird GROUP BY verwendet, dürfen im select nur Aggregatfunktionen oder die Spalten aus dem GROUP BY verwendet werden. Dann ist garantiert, dass für die jede Ergebniszeile jeweils nur ein Wert pro Spalte berechnet wird.

```
select count(*) “Anzahl”, Studiengang from Studierende group by Studiengang
```

Anzahl	STUDIENGANG
18	BWING

Jetzt werden zunächst die restlichen der 79 Studierenden eingefügt, bevor die nächsten Beispiele dargestellt werden. Die Ergebnismengen der Anfrage sind dadurch besser zum Verständnis geeignet.

```
Insert into Studierende values ('u13133','Tim','Fleissig',to_date('13.12.1991','DD-MM-YYYY'),'BWING',20152,10.50);

Insert into Studierende values ('u13134','Tom','Schlau',to_date('11.11.1992','DD-MM-YYYY'),'BWING',20152,10.50);

Insert into Studierende values ('u13135','Tom','Begabt',to_date('12.11.1991','DD-MM-YYYY'),'BWING',20152,10.50);

Insert into Studierende values ('u13136','Tom','Tolerant',to_date('01.11.1992','DD-MM-YYYY'),'BWING',20152,10.50);

Insert into Studierende values ('u13137','Tom','Lustig',to_date('13.08.1993','DD-MM-YYYY'),'BWING',20152,10.50);

Insert into Studierende values ('u13138','Tom','Mutig',to_date('08.08.1991','DD-MM-YYYY'),'BWING',20152,10.50);

Insert into Studierende values ('u13139','Tom','Schnell',to_date('04.06.1990','DD-MM-YYYY'),'BWING',20152,10.50);

Insert into Studierende values ('u13141','Jon','Fleissig',to_date('03.02.1991','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13142','Jim','Schlau',to_date('19.10.1992','DD-MM-YYYY'),'BWING',20152,10.50);

Insert into Studierende values ('u13143','Jon','Fleissig',to_date('18.12.1991','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13144','Jim','Schlau',to_date('11.01.1992','DD-MM-YYYY'),'WING',20152,10.50);
```

```
Insert into Studierende values ('u13145','Pat','Begabt',to_date('16.01.1991','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13146','Pit','Tolerant',to_date('01.03.1992','DD-MM-YYYY'),'BWING',20152,10.50);

Insert into Studierende values ('u13147','Pia','Lustig',to_date('13.05.1993','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13148','Pius','Mutig',to_date('08.05.1991','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13149','Maja','Schnell',to_date('04.05.1990','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13151','Eva','Gut',to_date('13.10.1991','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13152','Ben','Besser',to_date('11.12.1992','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13153','Emma','Erlesen',to_date('13.07.1991','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13154','Uma','Belesen',to_date('11.07.1992','DD-MM-YYYY'),'BWING',20152,10.50);

Insert into Studierende values ('u13155','Umma','Wissend',to_date('12.06.1991','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13156','Lea','Freudig',to_date('01.02.1992','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13157','elea','Selig',to_date('13.08.1993','DD-MM-YYYY'),'BWING',20152,10.50);

Insert into Studierende values ('u13158','Theo','Forsch',to_date('08.01.1991','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13159','Thea','Rasant',to_date('04.01.1990','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13161','Paul','Top',to_date('13.10.1991','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13162','Pius','Frisch',to_date('25.12.1992','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13163','Pia','Gelb',to_date('24.07.1991','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13164','Fynn','Grün',to_date('23.07.1992','DD-MM-YYYY'),'BWING',20152,10.50);
```

```
Insert into Studierende values ('u13165','Mats','Treffend',to_date('04.06.1991','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13166','Sam','Fertig',to_date('29.02.1992','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13167','Jay','Fix',to_date('27.08.1993','DD-MM-YYYY'),'BWING',20152,10.50);

Insert into Studierende values ('u13168','Chai','Super',to_date('16.01.1991','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13169','Phil','Fit',to_date('04.06.1990','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13171','Suse','Kernig',to_date('30.10.1991','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13172','Linn','Best',to_date('25.02.1992','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13173','Liz','Bestanden',to_date('07.07.1991','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13174','Kim','Hell',to_date('09.07.1992','DD-MM-YYYY'),'BWING',20152,10.50);

Insert into Studierende values ('u13175','Jaz','Bunt',to_date('04.01.1991','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13176','Fox','Nett',to_date('02.02.1992','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13177','Jack','Schön',to_date('27.12.1993','DD-MM-YYYY'),'BWING',20152,10.50);

Insert into Studierende values ('u13178','Ida','Eilig',to_date('16.06.1991','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13179','Mia','Fix',to_date('04.08.1990','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13181','Maik','Lieblich',to_date('10.10.1991','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13182','Mike','Liebig',to_date('05.02.1992','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13183','Mick','Strebsam',to_date('07.08.1991','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13184','Lou','Toll',to_date('09.06.1992','DD-MM-YYYY'),'BWING',20152,10.50);
```

```
Insert into Studierende values ('u13185','Liv','Besonders',to_date('04.03.1991','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13186','Susi','Anders',to_date('02.09.1992','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13187','Susa','Groß',to_date('13.12.1993','DD-MM-YYYY'),'BWING',20152,10.50);

Insert into Studierende values ('u13188','Tommi','Klein',to_date('18.06.1991','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13189','Tami','Riesig',to_date('19.08.1990','DD-MM-YYYY'),'WING',20152,10.50);

Commit;

Insert into Studiengang values ('INF','Informatik Bachelor','B.Sc.',null);

Commit;

Insert into Studierende values ('u13191','Ole','Superb',to_date('10.11.1991','DD-MM-YYYY'),'INF',20152,13.50);

Insert into Studierende values ('u13192','Peg','Herrlich',to_date('15.02.1992','DD-MM-YYYY'),'INF',20152,10.50);

Insert into Studierende values ('u13193','Pam','Wunderlich',to_date('17.08.1991','DD-MM-YYYY'),'INF',20152,10.50);

Insert into Studierende values ('u13194','Tin','Nachsichtig',to_date('19.06.1992','DD-MM-YYYY'),'BWING',20152,10.50);

Insert into Studierende values ('u13195','Pin','Einsichtig',to_date('14.03.1991','DD-MM-YYYY'),'WING',20152,10.50);

Insert into Studierende values ('u13196','Ali','Glücklich',to_date('22.09.1992','DD-MM-YYYY'),'INF',20152,13.50);

Insert into Studierende values ('u13197','Oleg','Zufrieden',to_date('23.12.1993','DD-MM-YYYY'),'BWING',20152,10.50);

Insert into Studierende values ('u13198','Tjis','Massig',to_date('28.08.1991','DD-MM-YYYY'),'INF',20152,13.50);

Insert into Studierende values ('u13199','Kurt','Rundlich',to_date('29.10.1990','DD-MM-YYYY'),'INF',20152,10.50);

Commit;
```

Jetzt noch einige Anfragen:

```
select *
from Studierende
order by unummer
```

unummer	Vorname	Nachname	Geburtstag	Studiengang	Studienbeginn	Gebuehr
u13111	Till	Golden	13.02.91	BWING	20152	10.5
u13112	Mia	Silber	11.05.92	BWING	20152	10.5
u13113	Ker	Braun	23.02.91	BWING	20152	10.5
u13114	Kia	Schwarz	21.04.92	BWING	20152	10.5
u13115	Ke	Rot	22.05.89	BWING	20152	10.5
u13116	Lu	Weiss	21.06.95	BWING	20152	10.5
u13117	Pete	Stark	21.08.93	BWING	20152	10.5
u13118	Lou	Rasend	09.08.90	BWING	20152	10.5
u13119	Jan	Neugierig	10.06.91	BWING	20152	10.5
u13121	Max	Dunkel	13.12.91	BWING	20152	10.5
u13122	Lu	Lustig	11.01.92	BWING	20152	10.5
u13123	Mo	Helle	13.02.91	BWING	20152	10.5
u13124	Uli	Exakt	11.04.92	BWING	20152	10.5
u13125	Ali	Genau	12.05.89	BWING	20152	10.5
u13126	Olli	Wild	01.06.95	BWING	20152	10.5
u13127	Oli	Kräftig	31.08.93	BWING	20152	10.5
u13128	Alex	Mutig	08.08.90	BWING	20152	10.5
u13129	Sara	Rasch	30.06.91	BWING	20152	10.5
u13133	Tim	Fleissig	13.12.91	BWING	20152	10.5
u13134	Tom	Schlau	11.11.92	BWING	20152	10.5
u13135	Tom	Begabt	12.11.91	BWING	20152	10.5
u13136	Tom	Tolerant	01.11.92	BWING	20152	10.5
u13137	Tom	Lustig	13.08.93	BWING	20152	10.5
u13138	Tom	Mutig	08.08.91	BWING	20152	10.5
u13139	Tom	Schnell	04.06.90	BWING	20152	10.5
u13141	Jon	Fleissig	03.02.91	WING	20152	10.5
u13142	Jim	Schlau	19.10.92	BWING	20152	10.5
u13143	Jon	Fleissig	18.12.91	WING	20152	10.5
u13144	Jim	Schlau	11.01.92	WING	20152	10.5
u13145	Pat	Begabt	16.01.91	WING	20152	10.5
u13146	Pit	Tolerant	01.03.92	BWING	20152	10.5
u13147	Pia	Lustig	13.05.93	WING	20152	10.5
u13148	Pius	Mutig	08.05.91	WING	20152	10.5
u13149	Maja	Schnell	04.05.90	WING	20152	10.5
u13151	Eva	Gut	13.10.91	WING	20152	10.5
u13152	Ben	Besser	11.12.92	WING	20152	10.5
u13153	Emma	Erlesen	13.07.91	WING	20152	10.5
u13154	Uma	Belesen	11.07.92	BWING	20152	10.5
u13155	Umma	Wissend	12.06.91	WING	20152	10.5
u13156	Lea	Freudig	01.02.92	WING	20152	10.5
u13157	elea	Selig	13.08.93	BWING	20152	10.5
u13158	Theo	Forsch	08.01.91	WING	20152	10.5
u13159	Thea	Rasant	04.01.90	WING	20152	10.5
u13161	Paul	Top	13.10.91	WING	20152	10.5
u13162	Pius	Frisch	25.12.92	WING	20152	10.5
u13163	Pia	Gelb	24.07.91	WING	20152	10.5
u13164	Fynn	Grün	23.07.92	BWING	20152	10.5
u13165	Mats	Treffend	04.06.91	WING	20152	10.5
u13166	Sam	Fertig	29.02.92	WING	20152	10.5
u13167	Jay	Fix	27.08.93	BWING	20152	10.5
u13168	Chai	Super	16.01.91	WING	20152	10.5
u13169	Phil	Fit	04.06.90	WING	20152	10.5
u13171	Suse	Kernig	30.10.91	WING	20152	10.5
u13172	Linn	Best	25.02.92	WING	20152	10.5
u13173	Liz	Bestanden	07.07.91	WING	20152	10.5
u13174	Kim	Hell	09.07.92	BWING	20152	10.5
u13175	Jaz	Bunt	04.01.91	WING	20152	10.5
u13176	Fox	Nett	02.02.92	WING	20152	10.5
u13177	Jack	Schön	27.12.93	BWING	20152	10.5
u13178	Ida	Eilig	16.06.91	WING	20152	10.5
u13179	Mia	Fix	04.08.90	WING	20152	10.5
u13181	Maik	Lieblich	10.10.91	WING	20152	10.5

u13182	Mike	Liebig	05.02.92	WING	20152	10.5
u13183	Mick	Strebsam	07.08.91	WING	20152	10.5
u13184	Lou	Toll	09.06.92	BWING	20152	10.5
u13185	Liv	Besonders	04.03.91	WING	20152	10.5
u13186	Susi	Anders	02.09.92	WING	20152	10.5
u13187	Susa	Groß	13.12.93	BWING	20152	10.5
u13188	Tommi	Klein	18.06.91	WING	20152	10.5
u13189	Tami	Riesig	19.08.90	WING	20152	10.5
u13191	Ole	Superb	10.11.91	INF	20152	13.5
u13192	Peg	Herrlich	15.02.92	INF	20152	10.5
u13193	Pam	Wunderlich	17.08.91	INF	20152	10.5
u13194	Tin	Nachsichtig	19.06.92	BWING	20152	10.5
u13195	Pin	Einsichtig	14.03.91	WING	20152	10.5
u13196	Ali	Glücklich	22.09.92	INF	20152	13.5
u13197	Oleg	Zufrieden	23.12.93	BWING	20152	10.5
u13198	Tjis	Massig	28.08.91	INF	20152	13.5
u13199	Kurt	Rundlich	29.10.90	INF	20152	10.5

➔ 79 Zeilen gewählt

```
select count(*) “Anzahl” from Studierende group by Studiengang
```

Anzahl
37
36
6

```
select count(*) “Anzahl” , Studiengang from Studierende group by Studiengang
```

Anzahl	STUDIENGANG
37	BWING
36	WING
6	INF

```
select count(*) "Anzahl Studierende", length(Vorname) BuchstabenimVornamen
from Studierende
group by length(Vorname)
```

Anzahl Studierende	BUCHSTABENIMVORNAMEN
4	2
30	4
1	5
44	3

```
select count(*) "Anzahl Studierende", length(Vorname) BuchstabenimVornamen
from Studierende
group by length(Vorname)
having count(*) > 10
```

Anzahl Studierende	BUCHSTABENIMVORNAMEN
30	4
44	3

Bisher wurde lediglich eine Eingabetabelle verwendet. Werden zwei (oder mehr) Tabellen in der FROM-Klausel aufgelistet, so wird jede Kombination der Zeilen (→ Kartesisches Produkt) für die weitere Berechnung in Betracht gezogen. Dies kann schnell zu sehr großen Ausgabemengen und Lasten für das Datenbanksystem führen.

Mit der Where-Bedingung (→ Verbundbedingung) lassen sich die Kombinationen auf die gewünschten und sinnvollen Kombinationen einschränken (→ Verbundoperation / JOIN).

Drei Beispiele zur Erinnerung.

```
select unummer, Studiengang
from Studierende, Studiengang
where unummer like 'u13133' and Studiengang = ID
```

```
select unummer, Studiengang
from Studierende, Studiengang
where unummer like 'u13133' and Studierende.Studiengang = Studiengang.ID
```

```
select unummer, Studiengang
from Studierende a, Studiengang b
where unummer like 'u13133' and a.Studiengang = b.ID
```

Geschachtelte Anfragen in nahezu beliebiger Tiefe. Um den Überblick zu behalten ist die Reihenfolge der Schlüsselworte wichtig: SELECT FROM WHERE GROUP BY HAVING ORDER BY!

SELECT-Anfragen können geschachtelt werden. Die Reihenfolge der Schlüsselworte bleibt auf jeder Ebene wie gehabt und wie hoffentlich bereits eingeprägt ☺
Abstrakte Beispiele:

```
SELECT (SELECT ... FROM ...)
FROM (SELECT ... FROM ...), (SELECT ... FROM ...)
WHERE ... (SELECT ... FROM ...)
GROUP BY ... (SELECT ... FROM ...)
HAVING ... (SELECT ... FROM ...)
ORDER BY
```

```
SELECT (SELECT ... FROM ...) FROM (SELECT ... FROM (SELECT (SELECT ... FROM ...)
FROM (SELECT ... FROM ...), (SELECT ... FROM ...) WHERE ... (SELECT ... FROM ...)
GROUP BY ... (SELECT ... FROM ...) HAVING ... (SELECT ... FROM ...)
ORDER BY ...) , (SELECT ... FROM ...) WHERE ... (SELECT ... FROM ...)
GROUP BY ... (SELECT ... FROM ...) HAVING ... (SELECT ... FROM ...) ORDER BY ...
```

Alles klar? Bis zum nächsten Teil ☺

Literaturhinweise:

Machen Sie sich unbedingt mit der SQL-Reference von Oracle vertraut, wie bereits besprochen. Zu finden unter www.oracle.com

---

Das Datenbanken Online Lexikon der FH Köln kann sicher die ein oder andere Frage beantworten:
http://lwibs01.gm.fh-koeln.de/wikis/wiki_db/index.php

http://lwibs01.gm.fh-koeln.de/wikis/wiki_db/index.php?n=Category.SQL

Das Buch „Elmasri, Navathe: Grundlagen von Datenbanksystemen, Pearson Studium“ ist in der Bibliothek in ausreichender Anzahl verfügbar.

3. ELEA-SYSTEM: BEISPIEL-LABOR „SEHENSWÜRDIGKEITEN“

Im Online-Übungssystem ELEA stehen für Sie aktuelle Labore bereit, die Ihnen nach dem Einloggen im ELEA-System zur Auswahl stehen. Die Labore besitzen jeweils unterschiedliche Schwierigkeitsstufen und greifen auf unterschiedliche Datenbanken bzw. Schemas zu.

Wählen Sie zunächst das Labor „Sehenswürdigkeiten“ aus. Es ist ein grundlegendes SQL-Labor.

Hier geht es um das Üben von SQL-Anfragen auf einer Beispieldatenbank, welche Information zu Sehenswürdigkeiten enthält.

Sie können die SQL-Anfragen die Ihnen über dieses Labor zugeordnet sind beliebig oft „probieren“.

Das ELEA-System wird in diesem Labor jeweils eine von drei möglichen Antworten-Arten geben:

- Ihre Anfrage konnte nicht ausgeführt werden zusammen mit der zugehörigen Fehlerausgabe des Datenbanksystems.
- Ihre Anfrage liefert nicht das richtige Ergebnis.
  Gleichzeitig werden Ihnen die beiden Ergebnistabellen zum Vergleich angezeigt.
- Ihre Anfrage liefert das richtige Ergebnis.

Da Sie nach dem Ausführen Ihrer Lösungsanfrage, das Ergebnis mit dem gewünschten Ergebnis vergleichen können, können Sie sich Schritt für Schritt dem gewünschten Ergebnis nähern.
Sobald eine Anfrage ein richtiges Ergebnis liefert, wird sie aus Ihrem Labor entfernt.

Sollten Sie eine Aufgabe nicht lösen können, können Sie diese überspringen. Die Aufgabe wird nach hinten verschoben und zu einem späteren Zeitpunkt wieder erscheinen.

Eine Übersicht über die jeweils für das jeweilige Labor aktuellen Tabellen finden Sie auch unter „Hilfe“ rechts oben im System. Zur besseren Übersicht sind in diesem Dokument einige Aufgaben dargestellt. So können Sie kurzfristig auch offline üben.

Zunächst die Beschreibung der für das Labor „Sehenswürdigkeiten“ relevanten Tabellen.

3.1. BEISPIEL-DATENBANK „SEHENSWÜRDIGKEITEN“

Folgende Tabellen stehen zur Verfügung!

ADRESSEN
ADRESSEN_ID (NUMBER(22)), HAUSNR (VARCHAR2(5)), ORT (VARCHAR2(30)), PLZ (NUMBER(22)), STRASSE (VARCHAR2(30))

ALIASNAMEN
ALIASNAME (VARCHAR2(50)), NAME (VARCHAR2(50))

OEFFNUNGSZEITEN
BEGINNMONAT (VARCHAR2(9)), BEGINNUHRZEIT (NUMBER(22)), ENDEMONAT (VARCHAR2(9)), ENDEUHRZEIT (NUMBER(22)), NAME (VARCHAR2(50)), WOCHENTAG (VARCHAR2(10))

SEHENSWUERDIGKEITEN
ADRESSEN_ID (NUMBER(22)), BAUJAHR (NUMBER(22)), BAUJAHRTEXT (VARCHAR2(50)), BESCHREIBUNG_KURZ (VARCHAR2(2000)), NAME (VARCHAR2(50)), OEFFNUNGSZEITEN_TEXT (VARCHAR2(2000)), TELNR (NUMBER(22))

SEHENSWUERDIGKEITEN_TYP
NAME (VARCHAR2(50)), TYP_ID (NUMBER(22))

TYP
ID (NUMBER(22)), NAME (VARCHAR2(30))

UEBERNACHTUNGSMOEGLICHKEITEN
ADRESSEN_ID (NUMBER(22)), NAME (VARCHAR2(50)), PREIS (NUMBER(22)), TELNR (NUMBER(22)), TYP (VARCHAR2(30))

3.1.1. FREMDSCHLÜSSELBEZIEHUNGEN

- Die Spalte Adressen_ID in der Tabelle Sehenswürdigkeiten ist ein Fremdschlüssel und verweist auf Spalte Adressen_id in der Tabelle Adressen
- Die Spalte Adressen_ID in der Tabelle Übernachtungsmoeglichkeiten ist ein Fremdschlüssel und verweist auf Spalte Adressen_id in der Tabelle Adressen
- Die Spalte Name in der Tabelle Aliasnamen ist ein Fremdschlüssel und verweist auf Spalte Name in der Tabelle Sehenswuerdigkeiten
- Die Spalte Typ_ID in der Tabelle Sehenswuerdigkeiten_Typ ist ein Fremdschlüssel und verweist auf Spalte ID in der Tabelle Typ
- Die Spalte Name in der Tabelle Sehenswuerdigkeiten_Typ ist ein Fremdschlüssel und verweist auf Spalte Name in der Tabelle Sehenswuerdigkeiten
- Die Spalte Name in der Tabelle Oeffnungszeiten ist ein Fremdschlüssel und verweist auf Spalte Name in der Tabelle Sehenswuerdigkeiten

3.1.2. BEISPIELANFRAGEN

Welche Sehenswürdigkeiten sind in der Datenbank enthalten?

NAME	ADRESSEN_ID	BESCHREIBUNG_KURZ	BAUJAHR	BAUJAHRTEXT	TELNR	OEFFNUNGSZEITEN_TEXT
Schloß Wernigerode	1	Das Schloss in Wernigerode, ursprünglich eine mittelalterliche Burg, ist gleichzeitig ein Museum und kann das ganze Jahr über besichtigt werden.		im ersten Viertel des 12. Jahrhunderts	3943553030	Mai bis Oktober: Täglich 10.00 bis 18.00 Uhr (letzter Einlass 17.30 Uhr) November bis April: Dienstag bis Freitag 10.00 bis 16.00 Uhr Sonnabend, Sonntag und an Feiertagen 10.00 bis 18.00 Uhr
Das kleinste Haus	3	Das kleinste Haus zählt zu den berühmtesten Gebäuden in Wernigerode und war früher ein Wohnhaus für Handwerkerfamilien.	1792	es wurde 1792 erbaut	3943606016	täglich von 10.00 bis 16.00 Uhr
Harzmuseum	4	Das Harzmusem bietet Einblicke in die Geschichte der Stadt und in die faszinierende Naturwelt des Harzes.	1821	es wurde 1821 als Wohnhaus errichtet	3943654450	Montag bis Samstag 10.00 Uhr bis 17.00 Uhr (auch Feiertags)
Feuerwehrmuseum	5	Das Feuerwehrmuseum beinhaltet eine Sammlung von historischer Feuerwehrtechnik und bietet Einblick in die Chronik der Feuerwehr von Wernigerode			3943601131	Do.: 14:30 Uhr bis 16:30 Uhr Sa.: 14:30 Uhr bis 17:00 oder nach Vereinbarung
Museum für Luftfahrt und Technik	6	In diesem Museum können Flugzeuge und Modelle besichtigt werden.			3943632790	Mo. und Di.: 10:00 Uhr bis 15:00 Uhr Mi. bis So.: 10:00 Uhr bis 17:00 Uhr
Schmiedemuseum	7	Im Schmiedemuseum kann eine originale Arbeitsstätte aus dem 17. Jahrhundert besichtigt werden.	1678	wurde 1678 erbaut	3943601772	Samstag 10.00 – 14.00 und auf Anfrage
Mahn- und Gedenkstätte	8	Mahn- und Gedenkstätte			3943632109	Montag bis Freitag: 08.00 bis 15.00 Uhr
Mühlenmuseum Wernigerode	9	Mühlenmuseum Wernigerode und Galerie im Kornboden			3943249604	Mai bis Oktober, Samstag und Sonntag von 13.00 bis 17.00 Uhr oder nach Vereinbarung
Galerie im Antiquariat Wernigerode	10	In dieser Galerie wird moderne Kunst präsentiert.			3943604232	Montag bis Mittwoch 10.00 - 18.30 Uhr, Donnerstag 10.00 - 20.00 Uhr, Freitag 10.00 - 18.30 Uhr, Samstag 10.00 - 16.00 Uhr
Galerie im Zentrum Harzkultur	7	Hier findet man ein umfangreiches Literaturarchiv zu den Themen der Harzregion.			3943905977	Mo. bis Fr.: 10.00 Uhr bis 12.00 Uhr und 14:00 Uhr bis 15:00 Uhr oder nach Vereinbarung
Baumannshöhle	20	Die Baumannshöhle ist eine von zwei Höhlen, die in Rübeland besichtigt werden können.			3945449132	Saison (Juli/Aug.) 9.00 - 17.30 Uhr,Nov. - Jan. 9.00 - 15.30 Uhr,Febr. - Juni 9.00 - 16.30 Uhr,Sept./Okt. 9.00 - 16.30 Uhr
Hermannshöhle	20	Die Hermannshöhle ist eine von zwei Höhlen, die in Rübeland besichtigt werden können.			3945449132	Saison (Juli/Aug.) 9.00 - 17.30 Uhr,Nov. - Jan. 9.00 - 15.30 Uhr,Febr. - Juni 9.00 - 16.30 Uhr,Sept./Okt. 9.00 - 16.30 Uhr
Brockenhaus	21	Das Brockenhaus auf dem Brocken ist Museum und Ausblick in die Landschaft zugleich.			394550005	365 Tage im Jahr von 9.30 bis 17.00 Uhr
Klosterkirche Drübeck	22	Das Kloster stellt ein Baudenkmal an der Straße der Romanik dar.			3945294300	Besichtigung: Montag bis Freitag von 11.00-16.00 Uhr, Samstag und Sonntag von 10.00-16.00 Uhr
Pullman City II	23	Pullman City II ist eine Westernstadt, in der man viel erleben kann.		in der Mitte des 19. Jahrhunderts	394597310	Während der Saison: Täglich von 10:00 Uhr bis 01.00 Uhr
Klosterkirche Ilsenburg	24	Die Klosterkirche in Ilsenburg ist eines der ersten Bauwerke der HirsauerBauschule in Deutschland.		im 11. Jahrhundert	3945219433	Mai bis Oktober: Täglich von 10.00 Uhr - 16.00 Uhr, November bis April: Montag - Freitag 13:00 - 14:00 Uhr, Samstag, Sonntag, Feiertag 11:00 - 12:00 Uhr

Geben Sie den Namen, die Kurzbeschreibung und das Baujahr aller Sehenswürdigkeiten aus.

NAME	BESCHREIBUNG_KURZ	BAUJAHR
Schloß Wernigerode	Das Schloss in Wernigerode, ursprünglich eine mittelalterliche Burg, ist gleichzeitig ein Museum und kann das ganze Jahr über besichtigt werden.	
Das kleinste Haus	Das kleinste Haus zählt zu den berühmtesten Gebäuden in Wernigerode und war früher ein Wohnhaus für Handwerkerfamilien.	1792
Harzmuseum	Das Harzmusem bietet Einblicke in die Geschichte der Stadt und in die faszinierende Naturwelt des Harzes.	1821
Feuerwehrmuseum	Das Feuerwehrmuseum beinhaltet eine Sammlung von historischer Feuerwehrtechnik und bietet Einblick in die Chronik der Feuerwehr von Wernigerode	
Museum für Luftfahrt und Technik	In diesem Museum können Flugzeuge und Modelle besichtigt werden.	
Schmiedemuseum	Im Schmiedemuseum kann eine originale Arbeitsstätte aus dem 17. Jahrhundert besichtigt werden.	1678
Mahn- und Gedenkstätte	Mahn- und Gedenkstätte	
Mühlenmuseum Wernigerode	Mühlenmuseum Wernigerode und Galerie im Kornboden	
Galerie im Antiquariat Wernigerode	In dieser Galerie wird moderne Kunst präsentiert.	
Galerie im Zentrum Harzkultur	Hier findet man ein umfangreiches Literaturarchiv zu den Themen der Harzregion.	
Baumannshöhle	Die Baumannshöhle ist eine von zwei Höhlen, die in Rübeland besichtigt werden können.	
Hermannshöhle	Die Hermannshöhle ist eine von zwei Höhlen, die in Rübeland besichtigt werden können.	
Brockenhaus	Das Brockenhaus auf dem Brocken ist Museum und Ausblick in die Landschaft zugleich.	
Klosterkirche Drübeck	Das Kloster stellt ein Baudenkmal an der Straße der Romanik dar.	
Pullman City II	Pullman City II ist eine Westernstadt, in der man viel erleben kann.	
Klosterkirche Ilsenburg	Die Klosterkirche in Ilsenburg ist eines der ersten Bauwerke der HirsauerBauschule in Deutschland.	

Anzeige des Datensatzes der Sehenswürdigkeit "Schmiedemuseum" in der Tabelle Sehenswürdigkeiten:

NAME	ADRESSEN_ID	BESCHREIBUNG_KURZ	BAUJAHR	BAUJAHRTEXT	TELNR	OEFFNUNGSZEITEN_TEXT
Schmiedemuseum	7	Im Schmiedemuseum kann eine originale Arbeitsstätte aus dem 17. Jahrhundert besichtigt werden.	1678	wurde 1678 erbaut	3943601772	Samstag 10.00 – 14.00 und auf Anfrage

Finden Sie alle Sehenswürdigkeiten, die mit einem M beginnen.

NAME	ADRESSEN_ID	BESCHREIBUNG_KURZ	BAUJAHR	BAUJAHRTEXT	TELNR	OEFFNUNGSZEITEN_TEXT
Mahn- und Gedenkstätte	8	Mahn- und Gedenkstätte			3943632109	Montag bis Freitag: 08.00 bis 15.00 Uhr
Museum für Luftfahrt und Technik	6	In diesem Museum können Flugzeuge und Modelle besichtigt werden.			3943632790	Mo. und Di.: 10:00 Uhr bis 15:00 Uhr Mi. bis So.: 10:00 Uhr bis 17:00 Uhr
Mühlenmuseum Wernigerode	9	Mühlenmuseum Wernigerode und Galerie im Kornboden			3943249604	Mai bis Oktober, Samstag und Sonntag von 13.00 bis 17.00 Uhr oder nach Vereinbarung

Welche Sehenswürdigkeiten beginnen mit einem M oder mit S?

NAME	ADRESSEN_ID	BESCHREIBUNG_KURZ	BAUJAHR	BAUJAHRTEXT	TELNR	OEFFNUNGSZEITEN_TEXT
Schloß Wernigerode	1	Das Schloss in Wernigerode, ursprünglich eine mittelalterliche Burg, ist gleichzeitig ein Museum und kann das ganze Jahr über besichtigt werden.		im ersten Viertel des 12. Jahrhunderts	3943553030	Mai bis Oktober: Täglich 10.00 bis 18.00 Uhr (letzter Einlass 17.30 Uhr) November bis April: Dienstag bis Freitag 10.00 bis 16.00 Uhr Sonnabend, Sonntag und an Feiertagen 10.00 bis 18.00 Uhr
Museum für Luftfahrt und Technik	6	In diesem Museum können Flugzeuge und Modelle besichtigt werden.			3943632790	Mo. und Di.: 10:00 Uhr bis 15:00 Uhr Mi. bis So.: 10:00 Uhr bis 17:00 Uhr
Schmiedemuseum	7	Im Schmiedemuseum kann eine originale Arbeitsstätte aus dem 17. Jahrhundert besichtigt werden.	1678	wurde 1678 erbaut	3943601772	Samstag 10.00 – 14.00 und auf Anfrage
Mahn- und Gedenkstätte	8	Mahn- und Gedenkstätte			3943632109	Montag bis Freitag: 08.00 bis 15.00 Uhr
Mühlenmuseum Wernigerode	9	Mühlenmuseum Wernigerode und Galerie im Kornboden			3943249604	Mai bis Oktober, Samstag und Sonntag von 13.00 bis 17.00 Uhr oder nach Vereinbarung

Welche Sehenswürdigkeiten beginnen mit einem M oder enden mit e?

NAME	ADRESSEN_ID	BESCHREIBUNG_KURZ	BAUJAHR	BAUJAHRTEXT	TELNR	OEFFNUNGSZEITEN_TEXT
Schloß Wernigerode	1	Das Schloss in Wernigerode, ursprünglich eine mittelalterliche Burg, ist gleichzeitig ein Museum und kann das ganze Jahr über besichtigt werden.		im ersten Viertel des 12. Jahrhunderts	3943553030	Mai bis Oktober: Täglich 10.00 bis 18.00 Uhr (letzter Einlass 17.30 Uhr) November bis April: Dienstag bis Freitag 10.00 bis 16.00 Uhr Sonnabend, Sonntag und an Feiertagen 10.00 bis 18.00 Uhr
Museum für Luftfahrt und Technik	6	In diesem Museum können Flugzeuge und Modelle besichtigt werden.			3943632790	Mo. und Di.: 10:00 Uhr bis 15:00 Uhr Mi. bis So.: 10:00 Uhr bis 17:00 Uhr
Mahn- und Gedenkstätte	8	Mahn- und Gedenkstätte			3943632109	Montag bis Freitag: 08.00 bis 15.00 Uhr
Mühlenmuseum Wernigerode	9	Mühlenmuseum Wernigerode und Galerie im Kornboden			3943249604	Mai bis Oktober, Samstag und Sonntag von 13.00 bis 17.00 Uhr oder nach Vereinbarung
Galerie im Antiquariat Wernigerode	10	In dieser Galerie wird moderne Kunst präsentiert.			3943604232	Montag bis Mittwoch 10.00 - 18.30 Uhr, Donnerstag 10.00 - 20.00 Uhr, Freitag 10.00 - 18.30 Uhr, Samstag 10.00 - 16.00 Uhr
Baumannshöhle	20	Die Baumannshöhle ist eine von zwei Höhlen, die in Rübeland besichtigt werden können.			3945449132	Saison (Juli/Aug.) 9.00 - 17.30 Uhr,Nov. - Jan. 9.00 - 15.30 Uhr,Febr. - Juni 9.00 - 16.30 Uhr,Sept./Okt. 9.00 - 16.30 Uhr
Hermannshöhle	20	Die Hermannshöhle ist eine von zwei Höhlen, die in Rübeland besichtigt werden können.			3945449132	Saison (Juli/Aug.) 9.00 - 17.30 Uhr,Nov. - Jan. 9.00 - 15.30 Uhr,Febr. - Juni 9.00 - 16.30 Uhr,Sept./Okt. 9.00 - 16.30 Uhr

Wie lautet die Adresse vom Schloß Wernigerode?

ADRESSEN_ID	STRASSE	HAUSNR	PLZ	ORT
1	Am Schloss	1	38855	Wernigerode

Wie wird Pullman City II auch genannt?

NAME	ALIASNAME
Pullman City II	Westernstadt

Wie viele Sehenswürdigkeiten gibt es von jedem Typ?
Geben Sie zusätzlich den Namen des Typs aus.

Anzahl	Name
1	Schloss
8	Museum
1	Gedenkstätte
3	Galerie
2	Höhle
2	Kirche
1	Sehenswürdigkeit

Wie viele Museen gibt es in der Datenbank?

COUNT(*)
8

Die aktuelle Anzahl der Museen in der Datenbank:

Anzahl Museen
8

Zu welchen Typen wird das Schloss in Wernigerode zugeordnet?

TYP_ID	Name
1	Schloss
2	Museum

Von Welchen Typen gibt es mehr als 2 Sehenswürdigkeiten? Geben Sie jeweils Anzahl und Namen des Typs aus.

COUNT(*)	Name
8	Museum
3	Galerie

Sortieren Sie die Namen der Sehenswürdigkeiten aufsteigend!

Name Aufsteigend Sortiert
Baumannshöhle
Brockenhaus
Das kleinste Haus
Feuerwehrmuseum
Galerie im Antiquariat Wernigerode
Galerie im Zentrum Harzkultur
Harzmuseum
Hermannshöhle
Klosterkirche Drübeck
Klosterkirche Ilsenburg
Mahn- und Gedenkstätte
Mühlenmuseum Wernigerode
Museum für Luftfahrt und Technik
Pullman City II
Schloß Wernigerode
Schmiedemuseum

Welche Sehenswürdigkeiten haben keinen Baujahrtext?

NAME	ADRESSEN_ID	BESCHREIBUNG_KURZ	BAUJAHR	BAUJAHRTEXT	TELNR	OEFFNUNGSZEITEN_TEXT
Feuerwehrmuseum	5	Das Feuerwehrmuseum beinhaltet eine Sammlung von historischer Feuerwehrtechnik und bietet Einblick in die Chronik der Feuerwehr von Wernigerode			3943601131	Do.: 14:30 Uhr bis 16:30 Uhr Sa.: 14:30 Uhr bis 17:00 oder nach Vereinbarung
Museum für Luftfahrt und Technik	6	In diesem Museum können Flugzeuge und Modelle besichtigt werden.			3943632790	Mo. und Di.: 10:00 Uhr bis 15:00 Uhr Mi. bis So.: 10:00 Uhr bis 17:00 Uhr
Mahn- und Gedenkstätte	8	Mahn- und Gedenkstätte			3943632109	Montag bis Freitag: 08.00 bis 15.00 Uhr
Mühlenmuseum Wernigerode	9	Mühlenmuseum Wernigerode und Galerie im Kornboden			3943249604	Mai bis Oktober, Samstag und Sonntag von 13.00 bis 17.00 Uhr oder nach Vereinbarung
Galerie im Antiquariat Wernigerode	10	In dieser Galerie wird moderne Kunst präsentiert.			3943604232	Montag bis Mittwoch 10.00 - 18.30 Uhr, Donnerstag 10.00 - 20.00 Uhr, Freitag 10.00 - 18.30 Uhr, Samstag 10.00 - 16.00 Uhr
Galerie im Zentrum Harzkultur	7	Hier findet man ein umfangreiches Literaturarchiv zu den Themen der Harzregion.			3943905977	Mo. bis Fr.: 10.00 Uhr bis 12.00 Uhr und 14:00 Uhr bis 15:00 Uhr oder nach Vereinbarung
Baumannshöhle	20	Die Baumannshöhle ist eine von zwei Höhlen, die in Rübeland besichtigt werden können.			3945449132	Saison (Juli/Aug.) 9.00 - 17.30 Uhr,Nov. - Jan. 9.00 - 15.30 Uhr,Febr. - Juni 9.00 - 16.30 Uhr,Sept./Okt. 9.00 - 16.30 Uhr
Hermannshöhle	20	Die Hermannshöhle ist eine von zwei Höhlen, die in Rübeland besichtigt werden können.			3945449132	Saison (Juli/Aug.) 9.00 - 17.30 Uhr,Nov. - Jan. 9.00 - 15.30 Uhr,Febr. - Juni 9.00 - 16.30 Uhr,Sept./Okt. 9.00 - 16.30 Uhr
Brockenhaus	21	Das Brockenhaus auf dem Brocken ist Museum und Ausblick in die Landschaft zugleich.			394550005	365 Tage im Jahr von 9.30 bis 17.00 Uhr
Klosterkirche Drübeck	22	Das Kloster stellt ein Baudenkmal an der Straße der Romanik dar.			3945294300	Besichtigung: Montag bis Freitag von 11.00-16.00 Uhr, Samstag und Sonntag von 10.00-16.00 Uhr

Für welche Sehenswürdigkeiten ist ein Baujahrtext vorhanden?

NAME	ADRESSEN_ID	BESCHREIBUNG_KURZ	BAUJAHR	BAUJAHRTEXT	TELNR	OEFFNUNGSZEITEN_TEXT
Schloß Wernigerode	1	Das Schloss in Wernigerode, ursprünglich eine mittelalterliche Burg, ist gleichzeitig ein Museum und kann das ganze Jahr über besichtigt werden.		im ersten Viertel des 12. Jahrhunderts	3943553030	Mai bis Oktober: Täglich 10.00 bis 18.00 Uhr (letzter Einlass 17.30 Uhr) November bis April: Dienstag bis Freitag 10.00 bis 16.00 Uhr Sonnabend, Sonntag und an Feiertagen 10.00 bis 18.00 Uhr
Das kleinste Haus	3	Das kleinste Haus zählt zu den berühmtesten Gebäuden in Wernigerode und war früher ein Wohnhaus für Handwerkerfamilien.	1792	es wurde 1792 erbaut	3943606016	täglich von 10.00 bis 16.00 Uhr
Harzmuseum	4	Das Harzmusem bietet Einblicke in die Geschichte der Stadt und in die faszinierende Naturwelt des Harzes.	1821	es wurde 1821 als Wohnhaus errichtet	3943654450	Montag bis Samstag 10.00 Uhr bis 17.00 Uhr (auch Feiertags)
Schmiedemuseum	7	Im Schmiedemuseum kann eine originale Arbeitsstätte aus dem 17. Jahrhundert besichtigt werden.	1678	wurde 1678 erbaut	3943601772	Samstag 10.00 – 14.00 und auf Anfrage
Pullman City II	23	Pullman City II ist eine Westernstadt, in der man viel erleben kann.		in der Mitte des 19. Jahrhunderts	394597310	Während der Saison: Täglich von 10:00 Uhr bis 01.00 Uhr
Klosterkirche Ilsenburg	24	Die Klosterkirche in Ilsenburg ist eines der ersten Bauwerke der HirsauerBauschule in Deutschland.		im 11. Jahrhundert	3945219433	Mai bis Oktober: Täglich von 10.00 Uhr - 16.00 Uhr, November bis April: Montag - Freitag 13:00 - 14:00 Uhr, Samstag, Sonntag, Feiertag 11:00 - 12:00 Uhr

Geben Sie Information zu den Sehenswürdigkeiten in der folgenden Art und Weise aus:

Name	Kurzbeschreibung	Weitere Information
Schloß Wernigerode	Das Schloss in Wernigerode, ursprünglich eine mittelalterliche Burg, ist gleichzeitig ein Museum und kann das ganze Jahr über besichtigt werden.	Weitere Information ist unter der Telefonnummer 03943553030zu erhalten!
Das kleinste Haus	Das kleinste Haus zählt zu den berühmtesten Gebäuden in Wernigerode und war früher ein Wohnhaus für Handwerkerfamilien.	Weitere Information ist unter der Telefonnummer 03943606016zu erhalten!
Harzmuseum	Das Harzmusem bietet Einblicke in die Geschichte der Stadt und in die faszinierende Naturwelt des Harzes.	Weitere Information ist unter der Telefonnummer 03943654450zu erhalten!
Feuerwehrmuseum	Das Feuerwehrmuseum beinhaltet eine Sammlung von historischer Feuerwehrtechnik und bietet Einblick in die Chronik der Feuerwehr von Wernigerode	Weitere Information ist unter der Telefonnummer 03943601131zu erhalten!
Museum für Luftfahrt und Technik	In diesem Museum können Flugzeuge und Modelle besichtigt werden.	Weitere Information ist unter der Telefonnummer 03943632790zu erhalten!
Schmiedemuseum	Im Schmiedemuseum kann eine originale Arbeitsstätte aus dem 17. Jahrhundert besichtigt werden.	Weitere Information ist unter der Telefonnummer 03943601772zu erhalten!
Mahn- und Gedenkstätte	Mahn- und Gedenkstätte	Weitere Information ist unter der Telefonnummer 03943632109zu erhalten!
Mühlenmuseum Wernigerode	Mühlenmuseum Wernigerode und Galerie im Kornboden	Weitere Information ist unter der Telefonnummer 03943249604zu erhalten!
Galerie im Antiquariat Wernigerode	In dieser Galerie wird moderne Kunst präsentiert.	Weitere Information ist unter der Telefonnummer 03943604232zu erhalten!
Galerie im Zentrum Harzkultur	Hier findet man ein umfangreiches Literaturarchiv zu den Themen der Harzregion.	Weitere Information ist unter der Telefonnummer 03943905977zu erhalten!
Baumannshöhle	Die Baumannshöhle ist eine von zwei Höhlen, die in Rübeland besichtigt werden können.	Weitere Information ist unter der Telefonnummer 03945449132zu erhalten!
Hermannshöhle	Die Hermannshöhle ist eine von zwei Höhlen, die in Rübeland besichtigt werden können.	Weitere Information ist unter der Telefonnummer 03945449132zu erhalten!
Brockenhaus	Das Brockenhaus auf dem Brocken ist Museum und Ausblick in die Landschaft zugleich.	Weitere Information ist unter der Telefonnummer 0394550005zu erhalten!
Klosterkirche Drübeck	Das Kloster stellt ein Baudenkmal an der Straße der Romanik dar.	Weitere Information ist unter der Telefonnummer 03945294300zu erhalten!
Pullman City II	Pullman City II ist eine Westernstadt, in der man viel erleben kann.	Weitere Information ist unter der Telefonnummer 0394597310zu erhalten!
Klosterkirche Ilsenburg	Die Klosterkirche in Ilsenburg ist eines der ersten Bauwerke der HirsauerBauschule in Deutschland.	Weitere Information ist unter der Telefonnummer 03945219433zu erhalten!

4. KONTROLLFRAGEN

In diesem Kapitel befinden sich einige Kontrollfragen und Übungsaufgaben.

4.1. SQL-DRL

Gegeben seien die folgenden Tabellen.

Alle Primärschlüssel sind unterstrichen.

Tabelle Mitarbeiter

MNR	Vorname	Nachname	GebDatum	Abteilung	Einstellung	Funktion
1333	Alex	Müller	13.11.1984	Entwicklung	15.10.2000	Projektleiter
1533	Marcello	Mariucci	22.02.1992	Qualität	01.04.2010	Qualitätsingenieur
1222	Jasmin	Salic	25.06.1984	Entwicklung	15.08.2012	Entwickler
2345	Pete	Miller	13.01.1966	Vorstand	01.05.2013	Assistent
3333	Rudi	Meyer	14.02.1990	Marketing	15.08.2013	Marketingleiter
4545	Pat	Meyer	24.09.1977	Vorstand	01.10.2010	Geschäftsführer
2332	Peter	Schneider	13.12.1984	Entwicklung	10.10.2001	Entwickler

Tabelle Schulungen

SchulungsID	Schulung	Datum	Schulungstage	ReguläreTeilnahmegebühr
O10	Oracle 10g	10.10.2013	3	10.000
OBI	Oracle BI	13.11.2013	5	20.000
ODev1	Oracle Developer	17.12.2013	2	8.000
O12	Oracle 12c	14.01.2014	4	15.000
CRM	CRM	20.01.2014	3	15.000
MF	Mitarbeiterführung	10.01.2014	5	20.000
SAPH	SAP HANA	13.11.2013	4	15.000
ODev2	Oracle Developer	10.01.2014	2	8.500

Tabelle Mitarbeiterschulungen

MitarbeiterNR	SchulungsID	BezahlteTeilnahmegebühr
1333	O10	9.500
1333	OBI	15.000
1333	O12	10.000
2345	MF	15.000
4545	CRM	15.000
4545	MF	15.000
2332	OBI	15.000
2332	O12	13.000
1333	ODev1	7.000
2332	ODev2	7.000
3333	OBI	10.000
3333	CRM	10.000

Fremdschlüsseleziehungen

- Die Spalte MitarbeiterNR ist Fremdschlüssel und verweist auf MNR in Tabelle Mitarbeiter
- Die Spalte SchulungsID ist Fremdschlüssel und verweist auf SchulungsID in Tabelle Schulungen

Formulieren Sie die folgenden Anfragen in SQL!

a) Wie lauten Vorname, Nachame, GebDatum aller Mitarbeiter, deren Nachname mit M beginnt?

Vorname	Nachname	GebDatum
Alex	Müller	13.11.1984
Marcello	Mariucci	22.02.1992
Pete	Miller	13.01.1966
Pat	Meyer	24.09.1977
Rudi	Meyer	14.02.1990

b) Wie lauten Vorname, Nachname, Geburtsdatum (Achtung auf die Spaltennamen achten!) aller Mitarbeiter, deren Nachname mit S beginnt oder mit M beginnt, aber nicht mit i endet?

Vorname	Nachname	Geburtsdatum
Alex	Müller	13.11.1984
Jasmin	Salic	25.06.1984
Pete	Miller	13.01.1966
Rudi	Meyer	14.02.1990
Pat	Meyer	24.09.1977
Peter	Schneider	13.12.1984

c) Geben Sie alle Vorname und Nachnamen der Mitarbeiter, die an Schulungen mit der SchulungsID CRM teilgenommen haben, sowie deren jeweils dafür BezahlteTeilnahmegebühr nach Nachnamen und Vornamen aufsteigend sortiert aus.

Vorname	Nachname	Teilnahmegebühr für CRM
Pat	Meyer	15.000
Rudi	Meyer	10.000

d) Welche Entwickler haben an Oracle 12c oder Oracle BI Schulungen teilgenommen?

MNR	Vorname	Nachname	Schulung	Funktion
1222	Jasmin	Salic	Oracle 12c	Entwickler
2332	Peter	Schneider	Oracle BI	Entwickler
2332	Peter	Schneider	Oracle 12c	Entwickler

e) Geben Sie die Mitarbeiter mit Vorname, Nachname, Funktion und Abteilung nach Alter absteigend sortiert wie folgt aus:

Mitarbeiterbeschreibungen
Pete Miller ist Assistent und arbeitet in der Abteilung Vorstand
Pat Meyer ist Geschäftsführer und arbeitet in der Abteilung Vorstand
Jasmin Salic ist Entwickler und arbeitet in der Abteilung Entwicklung
Alex Müller ist Projektleiter und arbeitet in der Abteilung Entwicklung
Pete Schneider ist Entwickler und arbeitet in der Abteilung Entwicklung
Rudi Meyer ist Marketingsleiter und arbeitet in der Abteilung Marketing
Marcello Mariucci ist Qualitätsingenieur und arbeitet in der Abteilung Qualität

f) Wie groß ist die Gesamtsumme der bezahlten Teilnahmegebühren für Mitarbeiter, die an Schulungen teilgenommen haben?

MitarbeiterNR	Teilnahmegebühren gesamt
1333	41.500
2345	15.000
4545	30.000
2332	35.000
3333	20.000

g) Geben Sie die Anzahl der Mitarbeiter pro Abteilung aus, jedoch nur für Abteilungen mit mehr als einem Mitarbeiter.

Abteilung	Mitarbeiteranzahl
Entwicklung	3
Vorstand	2

4.2. KONZEPTIONELLE DATENMODELLIERUNG (ENTITY-RELATIONSHIP-MODELL)

Sie haben die Aufgabe ein konzeptuelles Datenmodell für die Datenbank der HarzerStudentenBank (Abschnitt 5.1) sowie für die Brockenflüge (Abschnitt 5.2) zu erstellen. Dazu erhalten Sie im Folgenden die Ergebnisse einer Analyse der Miniwelt.

Definieren sie für jeden Entitätstyp einen Primärschlüssel (Schlüsselattribute). Falls notwendig kann ein künstlicher Schlüssel erzeugt werden. Begründen Sie in jedem Fall Ihre Entscheidung.

Leider haben die Business Analysten, welche die Analyse der Miniwelt durchgeführt haben, keine Datentypen bzw. Wertemengen für die Attribute angegeben. Bestimmen Sie sinnvolle Domänen.

Welche Informationen vermissen Sie darüber hinaus in der Beschreibung der Miniwelt? Was müssten Sie bei dem Business Analysten nachfragen?

Wie sieht das Entity-Relationship-Modell jeweils aus?

4.2.1. MINIWELT: HARZER-STUDIERENDEN-BANK

Beschreibung der Miniwelt:

Die HarzStudentenBank vergibt Kredite, die aus mindestens einer Rate bestehen. Für die einzelnen Raten eines Kredits können unterschiedliche Beträge und Fälligkeiten definiert werden. Die Raten eines Kredits werden fortlaufend mit 1 beginnend nummeriert. Es soll zu jeder Rate gespeichert werden, ob die Rate bereits bezahlt wurde. Für einen Kredit wird ein Zins und eine Kredithöhe sowie die sonstigen Konditionen festgelegt. Jeder Kredit besitzt eine eindeutige Nummer. Ein Kredit hat eine Laufzeit. Die Vergabe der Kredite erfolgt an die Kreditnehmer nur über die Zweigstellen. Ein Kreditnehmer kann nur einen Kredit pro Zweigstelle erhalten. Zu jedem Kreditnehmer werden Name, bestehend aus Vorname und Nachname, Telefon und Adresse erfasst. Das Datum der Kreditvergabe muss immer erfasst werden. Die Zweigstellen besitzen eine Identifikationsnummer, eine Adresse, und beliebig viele Telefonnummern sowie eine Faxnummer. Die Adressinformation besteht aus Straße, Hausnummer, Postleitzahl und Ort. Jede Zweigstelle wird von einem Mitarbeiter geleitet. In einer Zweigstelle arbeiten mindestens zwei Mitarbeiter. Es ist wichtig zu wissen seit wann die Mitarbeiter jeweils in der Zweigstelle arbeiten. Einige Mitarbeiter haben Personalverantwortung über andere Mitarbeiter. Diese Mitarbeiter müssen nicht in der gleichen Zweigstelle arbeiten. Zu jedem Mitarbeiter werden Personalnummer, Name, Adresse, eine Bankverbindung und das Gehalt verwaltet. Es ist auch wichtig zu wissen seit wann ein Mitarbeiter insgesamt für die Bank arbeitet.

4.2.2. MINIWELT: BROCKENFLÜGE

Beschreibung der Miniwelt

Der Flughafen „Brockenflüge“ hat zu tun mit Flugzeugen, Flugrouten, Landungen, Abflügen, Piloten, Passagieren, usw. Jedes eingesetzte Flugzeug hat eine Seriennummer, einen Namen sowie den Namen der Fluggesellschaft. Jedes Flugzeug korrespondiert einem Flugzeugtyp mit den folgenden Eigenschaften: Herstellername, Modellname, Anzahl der Sitzplätze, Kapazität,

Startbeschleunigung, Maximale Fluggeschwindigkeit, Landegeschwindigkeit. Die Piloten haben die Berechtigung bestimmte Flugzeugtypen fliegen zu dürfen. Piloten besitzen einen Namen und ein Geburtsdatum. Ein Pilot gehört zu dem Flugpersonal des Flughafens, für jedes Mitglied des Flugpersonals gibt es genau eine Personalinformation bestehend aus Personalnummer, Einstellungsdatum, Vertragskonditionen, Adresse, Bankverbindung und Gehalt.

Passagiere buchen Abflüge zu bestimmten Abflugzeiten. Zu Passagieren werden Name, Adresse, und Personalausweisnummer gespeichert. Das Buchungsdatum wird gespeichert. Das Flugpersonal wird jeweils den Abflügen zugeordnet, ebenso wie bei Abflügen eingesetzte Flugzeuge. Die Abflüge sind jeweils einer Route zugeordnet. Jede Route hat einen Zielflughafen und kann mehrere Zwischenstopps auf Flughäfen enthalten, die Nummer des Zwischenstopps bezogen auf eine Route wird gespeichert.

5. LITERATURHINWEISE

- Database SQL Language Reference, Oracle, www.oracle.com bzw. (https://docs.oracle.com/database/122/SQLRF/Introduction-to-Oracle-SQL.htm#SQLRF001)

- Elmasri, Navathe: Grundlagen von Datenbanksystemen, 3. Aktualisierte Auflage, Bachelorausgabe, Pearson Studium, 2009 bzw. Database Systems, Prentice Hall; 6th edition, 2013

- Kemper; Eickler; Datenbanksysteme: Eine Einführung. 9. erw. und akt. Auflage, De Gruyter Oldenbourg, 2013
- Kudraß (Hrsg.): Taschenbuch Datenbanken, 2. Auflage Hanser Verlag,2013.
- Vossen: Datenmodelle, Datenbanksprachen und Datenbankmanagementsysteme, 5. Auflage, Oldenbourg Verlag, 2008.
- Faeskorn-Woyke, Bertelsmeier, Riemer, Bauer: Datenbanksysteme, Theorie und Praxis mit SQL2003, Oracle und MySQL, Pearson Studium Verlag, e-books, 2007