

2.2 Datenmodellierung und Datenbanksysteme

2.2.1 Vorüberlegungen

Didaktisch-methodische Vorüberlegungen

Um die immer weiter anwachsende Informationsflut in den Griff zu bekommen, werden in großem Umfang Datenbanken zur Verwaltung von Information eingesetzt. Kenntnisse darüber, mit welchen Verfahren Informationen gespeichert, verknüpft und ausgewertet werden können, tragen auf mehrere Arten zur Mündigkeit des Bürgers bei: Als Anwender kann man die zur Verfügung stehenden Daten effizienter für sich nutzen; als Betroffener kann man besser abschätzen, welche Aussagen aus der Zusammenführung von Daten gewonnen werden können, und als Entscheider muss man fähig sein, eine Datenbank zu konzipieren.

Statische Datenmodellierung hat die Aufgabe, vorhandene Datenmengen zu strukturieren, die gefundene Struktur in ein Tabellenschema zu übertragen und aus diesem Tabellenschema eine Datenbank anzulegen. Durch die Analyse der Objekte, deren Daten einbezogen werden sollen, kommt man zu den beteiligten Klassen und deren Beziehungen zueinander. Klassen sowie die Beziehungen werden mit Hilfe von Tabellen umgesetzt. Das Tabellenschema kann direkt im relationalen Datenbanksystem implementiert werden. Eine sorgfältige objektorientierte Modellierung bei der Konzeption der Datenbank erspart die Thematisierung der Normalformen.

Bei der Modellierung ist zu berücksichtigen, dass die Datenspeicherung stets im Hinblick auf eine vorgegebene Zielsetzung erfolgt. Daten werden einzig und allein zu dem Zweck erfasst, sie später wieder nach den unterschiedlichsten Gesichtspunkten auswerten zu können. Bei der Strukturierung der Daten muss bereits berücksichtigt werden, welche Auswertungen möglich sein sollen. So kann z. B. die Frage, ob bei einer Adresse Straße und Hausnummer in zwei getrennten Attributen oder in einem einzigen Attribut zu speichern sind, nur durch die geplante Art der Verwendung beantwortet werden. Wird die Adresse als Ganzes benötigt, so genügt ein Attribut. Sollen aber auch Fragen wie „Welche Personen wohnen in der gleichen Straße?“ beantwortet werden, müssen zwei Attribute vorgesehen sein. Die Schüler können also nur dann auftragsorientiert modellieren, wenn sie auch die Möglichkeiten und Grenzen der Datenbankabfragen kennen und das Ziel der Modellierung vollständig vor Augen haben; daher müssen Datenmodellierung und Abfragetechnik parallel behandelt werden. Außerdem bietet diese Vorgehensweise den Vorteil, dass sich dadurch von Anfang an ein hoher Anteil an Eigenaktivität der Schüler realisieren lässt. So wird der handlungsorientierte Unterrichtsansatz der Informatik unterstützt.

Die Schüler haben sich bis zu diesem Zeitpunkt noch nicht bewusst mit größeren Datenmengen und ihrer Verwaltung auseinandergesetzt. Der Begriff „Daten“ (singular „Datum“) ist den Schülern aus dem Kapitel „Funktionen und Datenflüsse“ bekannt. Während dort jedoch die Verarbeitung einzelner Daten im Vordergrund stand, geht es nun um die Strukturierung umfangreicher statischer Datenmengen. Zur Modellierung wird dabei auf den objektorientierten Ansatz aus der Unterstufe zurückgegriffen. Die Erarbeitung der Begriffe und Modelle geht stets Hand in Hand mit ihrer Umsetzung in relationalen Datenbanken.

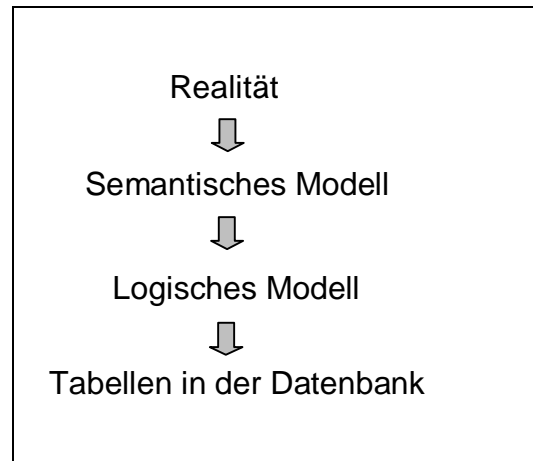
Im Hinblick auf eine altersgemäße Darstellung verzichtet man auf unnötige theoretische Details. Beispielsweise geht man zwar auf Redundanz und die damit verbundenen Probleme

ein, spricht aber Normalformen nicht explizit an, da ein gutes Klassenmodell bei der Umsetzung ohnehin die Einhaltung der ersten drei Normalformen sicherstellt.¹

Die Datenmodellierung

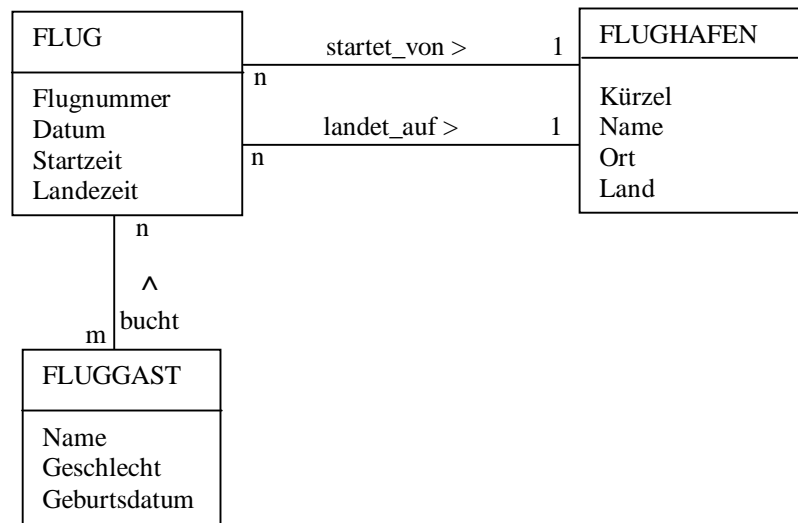
Wie beim Themenbereich „Funktionen und Datenflüsse“ steht auch beim Entwurf von Datenbanken der Modellierungsvorgang im Mittelpunkt.

Der Modellierungsvorgang besteht im Wesentlichen aus drei Schritten. Ausgehend von einer realen Situation wird ein semantisches Modell erstellt. Hier spielt das Denken in Klassen und Beziehungen die entscheidende Rolle; das semantische Modell stellt eine formale und strukturierte Beschreibung aller relevanten Objekte und deren Beziehungen untereinander dar. Im nächsten Modellierungsschritt wird es nach formalen Regeln in ein logisches Modell, hier das relationale Datenbankmodell, umgesetzt. Dieses wird im Datenbankprogramm direkt in die Tabellendefinition übergeführt.



Folgendes Beispiel verdeutlicht diese Schritte:

Ein Flugbuchungssystem erfordert das Speichern von umfangreichen Datenmengen. Die Klassen und ihre Beziehungen werden mit nebenstehendem Klassendiagramm in UML-Notation modelliert.



Die in vielen Büchern noch zu findende Darstellung als ER-Diagramm ist nicht mehr gebräuchlich. Sie wird im Unterricht deshalb nicht verwendet. Außerdem sind die Schüler mit dem Klassendiagramm bereits vertraut.

Dieses semantische Modell wird nach folgenden Regeln in das logische Modell übertragen:

- Jeder Klasse entspricht eine Tabelle.
- Jedem Attribut ist eine Spalte zugeordnet.
- Dem eindeutigen Objektbezeichner entspricht der Schlüssel der jeweiligen Tabelle, gegebenenfalls muss eine zusätzliche Spalte für einen künstlichen Schlüssel vereinbart werden.

¹ Literatur:

G. Schlageter, W. Stucky, Datenbanksysteme: Konzepte und Modelle, Teubner Studienbücher, 1983

A. Kemper, A. Eickler, Datenbanksysteme. Eine Einführung, Oldenbourg Wissenschaftsverlag GmbH, 2006

B. Matzke, Datenbanken im Unterricht, <http://www.schule.bayern.de/texte/datenbanktheorie.pdf>, 2000

- Zur Realisierung der 1:n-Beziehung erhält die Tabelle auf der „n-Seite“ eine weitere Spalte für den Fremdschlüssel.
- Für die m:n-Beziehung muss eine weitere Tabelle angelegt werden. Sie erhält als Spalten die Primärschlüssel der beiden miteinander in Beziehung stehenden Tabellen.

Das zugehörige logische Modell beschreibt die Struktur der Relationen. Durchgehend unterstrichen sind die Schlüssel, die Fremdschlüssel sind gestrichelt markiert. Dabei werden werkzeugabhängig die Datentypen festgelegt. Die Domänenangabe wird beim Relationenschema in der Regel weggelassen, weil die Datentypen werkzeugabhängig sind.

Das Tabellenschema lässt sich eins zu eins in die Tabellendefinition des Datenbankprogramms übertragen.

```
flug (Flugnummer: varchar(10), Datum: date, Startzeit: time,
      Landezeit: time, Startflughafen: varchar(5),
      Zielflughafen: varchar(5))
fluggast (PersonID: int(11), Name: varchar(50), Geschlecht:
          char(1), Geburtsdatum: date)
flug_zu_fluggast (Flugnummer: varchar(10), Datum: date,
                  PersonID: int(11))
flughafen (Kürzel: varchar(5), Name: varchar(50), Ort:
           varchar(50), Land: varchar(50))
```

flug

Feld	Typ
<u>Flugnummer</u>	varchar(10)
<u>Datum</u>	date
Startzeit	time
Landezeit	time
Startflughafen	varchar(5)
Zielflughafen	varchar(5)

flughafen

Feld	Typ
<u>Kürzel</u>	varchar(5)
Name	varchar(50)
Ort	varchar(50)
Land	varchar(50)

flug_zu_fluggast

Feld	Typ
<u>Flugnummer</u>	varchar(10)
<u>Datum</u>	date
<u>PersonID</u>	int(11)

fluggast

Feld	Typ
<u>PersonID</u>	int(11)
Name	varchar(50)
Geschlecht	char(1)
Geburtsdatum	date

Die Abgrenzung zwischen semantischem und logischem Modell wird durch konsequente Einhaltung von Schreibkonventionen unterstützt. Klassenbezeichner werden in Großbuchstaben geschrieben. Tabellen erhalten den gleichen Bezeichner in Kleinbuchstaben. Um zu verdeutlichen, auf welches Modell man sich gerade bezieht, werden im Unterrichtsgespräch die folgenden Fachbegriffe sorgfältig zugeordnet:

Semantisches Modell	Logisches Modell
Klasse	Tabellendefinition
Attribut	Spaltenname
Attributwert	Zelleninhalt
Datenobjekt	Datensatz
Beziehung	Fremdschlüssel, Beziehungstabelle

Den Schülern soll der Unterschied zwischen dem semantischen Modell und dem logischen Modell bewusst werden. Man kann dabei allerdings die explizite Darstellung des Relationenschemas im Unterricht auf einige Beispiele beschränken und das Klassendiagramm üblicherweise mit oben beschriebenen Regeln unmittelbar in Datenbanktabellen umsetzen.

Anmerkungen zur Kardinalität

Bereits beim Erstellen des UML-Diagramms macht man sich zur Kardinalität der Beziehungen Gedanken. Die Kardinalität gibt an, wie viele Objekte der einen Klasse mit wie vielen Objekten der anderen Klasse in Beziehung treten können. In der Unterstufe wurden 1:1- und 1:vielen-Beziehungen betrachtet; letztere wurden im Klassendiagramm mit Hilfe des „Knödels“ dargestellt. In der Mittelstufe wird eine differenziertere Schreibweise nach UML-Standard eingeführt; folgende Bezeichnungen werden in der Handreichung verwendet:

- Typisch sind 1:1-, 1:0,1-, 1:n- bzw. n:m-Beziehungen.
- n oder m steht für eines oder viele Objekte.
- „0,1“ steht für kein oder ein Objekt; alternativ schreibt man auch „c“ (conditional).
- „0..n“ steht für keines, eines oder viele Objekte.
- Bereichsangaben wie 1..5:2..m wären möglich, werden im Unterricht jedoch nur bei Bedarf eingesetzt.

Anmerkungen zu Schlüssel und Fremdschlüssel

Im Klassenmodell werden die Individualität und die eindeutige Ansprechbarkeit der Objekte durch den Objektbezeichner sichergestellt. Da dieser nicht in das logische Modell übernommen wird, führt man im Relationenmodell zur Sicherung der Eindeutigkeit der Datensätze den (Primär-)Schlüssel ein. Ein **Schlüssel** ist eine Menge von Spalten, durch deren Werte jeder Datensatz eindeutig bestimmt ist. Im Extremfall kann es vorkommen, dass zwei Datenobjekte in allen Attributwerten übereinstimmen. Die zugehörigen Datensätze wären dann identische Tupel, also nicht unterscheidbar. In diesem Fall muss zur Sicherstellung der Individualität eine weitere Spalte (z. B. laufende Nummer) als Schlüssel eingefügt werden. So einen Schlüssel nennt man „künstlich“, weil dieser Spalte kein Attribut der Klasse entspricht. Die Verwendung eines künstlichen Schlüssels ist auch dann sinnvoll, wenn ansonsten zu viele Spalten für den Schlüssel herangezogen werden müssten.

Zur Abbildung der 1:n-Beziehung aus dem Klassenmodell in das Tabellenmodell erhält die Tabelle auf der „n-Seite“ eine weitere Spalte für den **Fremdschlüssel**. In die Felder der Fremdschlüsselspalte wird der Primärschlüsselwert des zugehörigen Datensatzes der „1-Seite“ eingetragen. Besteht der Primärschlüssel der „1-Seite“ aus mehreren Spalten, so müssen entsprechend viele Fremdschlüsselspalten in die Tabelle auf der „n-Seite“ übernommen werden.

Bei der Übertragung einer n:m-Beziehung aus dem Klassenmodell muss in das Tabellenmodell eine weitere Tabelle aufgenommen werden. Ihre Spalten enthalten die Primärschlüssel der beiden zu verknüpfenden Tabellen als Fremdschlüssel. Der Primärschlüssel dieser Beziehungstabelle wird aus allen Spalten gebildet.

Abfragen

Bereits bei der Modellierung wurden mögliche Fragestellungen an die Datenbank berücksichtigt. Die Abfragen werden in der universellen Datenbankabfrage- und -manipulationssprache SQL (engl. „Structured Query Language“) formuliert. SQL ist eine Datenbanksprache, die auf dem relationalen Datenmodell basiert. Auch die von vielen Werkzeugen angebotenen Assistenten verwenden SQL im Hintergrund. Deren Einsatz im Unterricht würde den Blick auf die gewünschten Lerninhalte verstellen. Natürlich kann es nicht Ziel des Unterrichts sein, einen vollständigen Überblick über alle SQL-Abfrageklauseln zu erhalten.

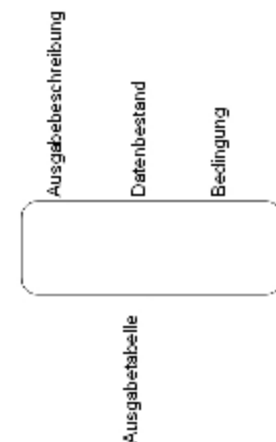
Als Anhaltspunkt für eine zweckmäßige Auswahl dient folgende Aufstellung, die allerdings in dieser allgemeinen Art nicht mit den Schülern behandelt werden sollte:

SELECT * Spalte1[, Spalte2, ...] Ausdruck1[, Ausdruck2, ...]	Auswahl aller (*) oder einiger Spalten und Ausdrücke wie beispielsweise SUM()
FROM Tabelle1[, Tabelle2, ...]	verwendete Tabellen
[WHERE Kriterium]	Kriterien enthalten Vergleichs- oder logische Operatoren wie <, >=, =, NOT, AND, OR
[GROUP BY SpalteX [, SpalteY, ...]]	Bei Verwendung von Aggregatfunktionen müssen unter GROUP BY mindestens die Spalten aufgeführt werden, die auch unter SELECT stehen.
[ORDER BY SpalteA AusdruckA [ASC DESC], SpalteB AusdruckB [ASC DESC], ...]	Sortieren der ausgegebenen Datensätze: Die erstgenannte Spalte gibt den ersten Sortierschlüssel an usw.

Für die Schüler ist es wichtig zu wissen, dass jede Abfrage auf zwei Grundoperationen aufbaut, die in der Regel miteinander kombiniert werden: Eine **Selektion** bedeutet eine Auswahl von Datensätzen (Zeilen) der Datentabelle mit bestimmten Eigenschaften, eine **Projektion** wählt bestimmte Spalten der Datentabelle aus.

So kann man in Anknüpfung an das vorhergehende Kapitel „Funktionen und Datenflüsse“ eine Abfrage als Funktion mit drei Parametern deuten. In den Prozess fließen die verwendeten Tabellen als Datenbestand, eine Beschreibung der auszugebenden Spalten und die Selektionsbedingung. Er liefert als Ausgabe eine neue Tabelle. Bedingungen als Funktionen sind den Schülern bereits bekannt. Genau genommen fließt hier nicht der Wahrheitswert in den Prozess, sondern die Beschreibung der Bedingung, die für jeden Datensatz ausgewertet wird.

Der Vorteil der funktionalen Sichtweise wird dann deutlich, wenn die Ausgabetable mit einer weiteren Funktion, beispielsweise über eine „ORDER BY“-Klausel oder eine Unterabfrage, weiter verarbeitet wird.



Als Hintergrundwissen für den Lehrer ist die Kenntnis der Auswertungsreihenfolge nützlich: Zunächst wird – bei Abfragen über mehrere Tabellen – das Kreuzprodukt der beteiligten Tabellen gebildet (FROM), anschließend erfolgt die Selektion der Datensätze anhand der Join- und anderen Bedingungen (WHERE). Bei Bedarf werden jetzt die Datensätze für die Aggregatfunktionen gruppiert (GROUP BY) und die Aggregatfunktionen in jeder Gruppe ausgewertet. Nach der Sortierung (ORDER BY) werden im letzten Schritt die Spalten und Ausdrücke ausgewählt (SELECT).

Werkzeuge zur unterrichtlichen Umsetzung

Bei der Entscheidung für ein Datenbankmanagementsystem spielt die leicht erlernbare Bedienbarkeit eine große Rolle. Wesentlich ist außerdem, dass die erarbeiteten Modelle anschaulich in Datenbanken umgesetzt werden können. Einplatzsysteme sind mit Einschränkungen verwendbar, weil damit die Mehrbenutzerproblematik zwar sichtbar gemacht, aber nur unzureichend gelöst werden kann.

In den Unterrichtsstunden, auf denen die folgenden Unterrichtsskizzen beruhen, wurde das kommerzielle MSAccess, das vielfach im Rahmen des Officepakets bereits auf den Rechnern installiert ist, und das frei verfügbare MySQL-Datenbankmanagementsystem eingesetzt. Den MySQL-Datenbankserver, das Administrationsprogramm MySQL Administrator und den Datenbankclient MySQL Query Browser findet man unter <http://dev.mysql.com/downloads/>. Relativ einfach zu installieren ist das XAMPP-Paket (<http://www.apachefriends.de>), welches den MySQL-Datenbankserver, den Webserver Apache und den Webclient phpMyAdmin enthält und in der „installer-Version“ sehr komfortabel die Komponenten verknüpft (Hinweise im Begleitmaterial XAMPP_und_mehr.doc). Damit lässt sich jeder Browser als Client verwenden, so dass auf den Arbeitsplatzrechnern keine Software eingerichtet werden muss.

Die im folgenden Unterrichtskonzept genannten Datenbanken liegen im Begleitmaterial sowohl in einer MSAccess-Version (*.mdb) als auch in einer MySQL-Version (*.sql) vor. Die *.mdb-Datei enthält eine MSAccess-Datenbank, die z. B. durch Doppelklick geöffnet wird. Die MySQL-Version ist eine Textdatei mit einer Folge von SQL-Anweisungen, welche die gewünschten Tabellen erzeugen und mit Daten füllen. Sie können mit Hilfe von phpMyAdmin oder MySQL Query Browser in eine bestehende (auch leere) Datenbank importiert werden.

2.2.2 Unterrichtskonzept

Die Inhalte des Lehrplans lassen sich – aufgrund der Verzahnung von Modellierung und Abfrage – nicht streng sequentiell abarbeiten, sondern werden nach dem Spiralprinzip auf unterschiedlichem Niveau an verschiedenen Stellen im Unterrichtsfortschritt immer wieder thematisiert.

Für die Behandlung des Lehrplankapitels „Inf 9.2 Datenmodellierung und Datenbanksysteme“ wird folgender Aufbau einer Unterrichtssequenz vorgeschlagen:

- Einblick: Große Datenmengen und ihre Verwaltung (2 Stunden)
- Datenbanken mit einer Tabelle (8 Stunden)
 - Klassendiagramm und relationale Datenbank (3 Stunden)
 - Einfache Abfragen (2 Stunden)
 - Abfragen mit Aggregatfunktionen (3 Stunden)
- Datenbanken mit mehreren Tabellen (13 Stunden)
 - Redundante Daten (1 Stunde)
 - Konzeption einer Datenbank, Klassendiagramm (2 Stunden)
 - Umsetzung der 1:n-Beziehung – Fremdschlüssel (2 Stunden)
 - Abfragen (3 Stunden)
 - Realisierung der n:m-Beziehung (4 Stunden)
 - Datenintegrität (1 Stunde)
- Einschränkung der Sicht auf Daten (1 Stunde)
- Datenpflege (2 Stunden)
- Datenschutz und Datensicherheit (1 Stunde)
- Komplexeres Anwendungsbeispiel (11 Stunden)

Alle im folgenden Text genannten Dateien stehen auf der Begleit-CD zu Verfügung. Sie sind sowohl mit MSAccess als auch mit MySQL ausgeführt. Auf die Angabe des Dateisuffix wurde deshalb verzichtet.

Einblick: Große Datenmengen und ihre Verwaltung (1.–2. Stunde)

In diesen beiden Stunden lernen die Schüler Beispiele für Sammlungen großer Datenmengen kennen und erhalten einen Einblick in den Verwendungszweck derartiger Datensammlungen; die Jugendlichen beschäftigen sich zum ersten Mal mit der Tabelle als Grundelement relationaler Datenbanken.

Impuls und Arbeit am Computer:

Zur Einführung in das Thema werden im Unterrichtsgespräch anhand einer Internetrecherche folgende Fragestellungen diskutiert und auf Beispielseiten werden Einblicke in bestehende Datensammlungen gewonnen:

- Welche Institutionen/Firmen sammeln (größere) Mengen an Daten (Bank, Einwohnermeldeamt, Verkehrssünderdatei, Bibliothek, ...)?
- Beispiele für Informationsquellen, die auf Datenbanken basieren, sind
 - Bayerische Staatsbibliothek: <http://www.bsb-muenchen.de/digital.htm>,
 - Kraftfahrtbundesamt: <http://www.kba.de/>,
 - Datenbank „Neue Medien im Unterricht“: <http://www.sodis.de/>,
 - Datenbank für Gruppenspiele: <http://www.spieledatenbank.de/>,
 - Digitales Lexikon: <http://de.wikipedia.org/wiki/Hauptseite>,
 - CD-Datenbank: http://www.freedb.org/freedb_search.php,
 - Flugbuchungssysteme, Systeme für Urlaubsbuchungen.
- Über welche Gegenstände wird Information gesammelt?
(Menschen, Bücher, Konten, Kraftfahrzeuge, ...)
- Werden die gesammelten Daten ausgewertet? Wenn ja, nach welchen Gesichtspunkten erfolgt die Auswertung?
- Wer nutzt die Information?
- Wie lässt sich die Menge an Daten strukturieren/organisieren?
- Wie hat man früher ohne Computer (kleinere) Datenmengen verwaltet?
(Karteikarten, Liste, Tabelle)

Bereits in dieser einführenden Sequenz kann man auf Fragen des Datenschutzes eingehen, wenn die von den Schülern gefundenen Beispiele dies nahelegen.

Die Schüler sind aus dem Kapitel „Funktionen und Datenflüsse“ bereits mit einem Tabellenkalkulationsprogramm vertraut. Deshalb ist es sinnvoll, sie zu Beginn mit einem Beispiel zu konfrontieren, das eine größere Ansammlung von Daten in einer Texttabelle oder einem Rechenblatt eines Tabellenkalkulationssystems abspeichert (Datei Einkauf.xls).

Die Schüler können eine derartige Tabelle auch selbst erstellen, wobei der Lehrer das Tabellengerüst vorgibt (Datei Einkauf0.xls).

Vorbereitende Hausaufgabe:

Sammle bis zur nächsten Stunde Kaufbelege und trage die Daten der Artikel in eine Tabelle (Datei Einkauf0.xls) ein. Speichere das Dokument unter dem Namen Einkauf_xx (xx steht für deinen Namen).

Aus datenschutzrechtlichen Gründen wird darauf hingewiesen, dass die Namen der Kunden keinesfalls Echtdaten sein dürfen.

Die Lehrkraft oder Schüler führen die verschiedenen Dokumente anschließend zu einer einzigen Tabelle zusammen:

	A	B	C	D	E	F	G	H
1	Kunde	Geschlecht	Sparte	Warenbezeichnung	Preis	Zahlungsart	Geschäft	Kaufdatum
2	Klaus Kreplin	m	Lebensmittel	Krustenbrot	2,40 €	bar	Bäckerei Hold	05.03.2007
3	Nicole Gütlings	w	Unterhaltung	Kinokarte	11,00 €	bar	Cinestadt	20.02.2007
4	Chantal Amberg	w	Übriges	Möbel	270,00 €	bar	AKEL Fürth	14.08.2007
5	Nina Hofer	w	Kleider	Hochzeitskleid	649,90 €	Karte	Bestadt	04.09.2007
6	Nicole Gütlings	w	Unterhaltung	Kinokarte	11,00 €	bar	Cinestadt	20.02.2007
7	Günther Gauss	m	Lebensmittel	Braune Limonade	2,00 €	Karte	Comarkt Schweinau	18.03.2006
8	Claudio Haas	m	Lebensmittel	Brownie	7,90 €	Karte	Euro-Markt Langwasser	30.08.2007
9	Michael Haas	m	Lebensmittel	Coop Citrus	1,50 €	bar	Comarkt Langwasser	31.12.2007

Derartige Daten über den Einkauf werden heutzutage bei der Kasseneingabe gespeichert. Auf diesem Weg lassen sich Datensammlungen zur Untersuchung des Kaufverhaltens aufbauen; Kundenkarten unterstützen dabei die Zuordnung Ware-Kunde.

Anhand der vorliegenden Tabelle diskutiert man mit den Schülern, welche Anforderungen an eine sinnvolle Datenverwaltung gestellt werden. Insbesondere sind nur die wichtigen Daten zu erfassen, unnötige müssen weggelassen werden. Anfragen nach unterschiedlichen Gesichtspunkten sollten beantwortet werden können.

Arbeit am Computer:

Die folgenden Beispiele beziehen sich auf die beiliegende Datei Einkauf.xls und müssen bei Verwendung eigener Daten angepasst werden. Die Schüler versuchen, folgende Fragen zu beantworten:

1. Welche Produkte hat Martina Groß eingekauft?
2. Welches Produkt war am teuersten?
3. Wie viele Artikel wurden bei Karma gekauft?
4. Wer hat insgesamt am meisten Geld ausgegeben?
5. Wie viele Artikel wurden bei Oldi am 26.8.2007 gekauft?
6. Haben am 02.09.2007 die männlichen oder die weiblichen Kunden im Durchschnitt mehr ausgegeben?

Die ersten drei Fragestellungen lassen sich in Tabellenkalkulationsprogrammen mit Hilfe von Such-, Sortier- und Filterfunktionen (Menü „Daten“) relativ komfortabel beantworten. Bei der vierten Frage müsste der Schüler kundenbezogen die Preise aufsummieren und anschließend das Maximum suchen. Der Aufwand ist dann kaum vertretbar, wenn das Programm keine automatische Auswertung innerhalb von Teilgruppierungen ermöglicht. Die fünfte Aufgabe erfordert ein Sortieren nach zwei Kriterien, Datum und Geschäft. Die Anzahl wird beispielsweise von Hand ausgewertet. Zur Lösung der sechsten Aufgabe müssten alle Datensätze zum Datum 02.09.2007 herausgefiltert werden. Danach könnte für jeden Kunden die Summe der Einkaufspreise errechnet werden. Ein möglicher Zwischenschritt wäre auch, die Preise – jeweils nach Herausfiltern eines bestimmten Geschlechts – zu summieren.

Schließlich wird über diese Summen geschlechtsspezifisch gemittelt. Diese Auswertung ist selbst beim geringen Datenbestand der Beispieltabelle nur als Fleißaufgabe anzubieten.

Die Beispiele zeigen, dass sich je nach Funktionsumfang des Tabellenkalkulationsprogramms komplexere Fragestellungen z. T. nur mit unvertretbarem Aufwand beantworten lassen, obwohl alle Informationen vorhanden sind. Die Schwierigkeiten wachsen mit zunehmendem Umfang der Datensammlung. Nachteilig wirkt sich auch aus, dass viele Daten mehrfach eingegeben wurden (Geschäft, Kunde, ...). Abgesehen vom Arbeitsaufwand führen dabei Tippfehler dazu, dass die betreffenden Datensätze anschließend kaum wiedergefunden werden können. Ein weiterer Nachteil macht sich bemerkbar, sobald die Daten auf mehr als eine Tabelle verteilt sind. Dann ist das Auffinden tabellenübergreifender Information mit Tabellenkalkulationsprogrammen nicht mehr leistbar.

Für umfangreichere Datenverwaltung sind Tabellenkalkulationsprogramme nur beschränkt geeignet; eigene Werkzeuge sind nötig: Datenbankverwaltungssysteme (DBMS, engl. „Data Base Management System“).

Hefteintrag:

Große Datenmengen und ihre Verwaltung

Große Datenmengen werden so gespeichert, dass gezielt ausgewählte Teile der Daten wiedergewonnen und je nach Fragestellung in geeigneter Zusammenstellung ausgegeben werden können.

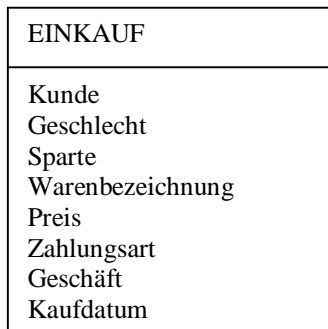
Modellierung einer Datenbank mit einer einzelnen Tabelle (3.–10. Stunde)

In der ersten größeren Unterrichtsequenz erkennen die Schüler, dass man gleichartige Daten als Objekte der gleichen Klasse auffassen kann. Sie erstellen ein Klassendiagramm, übertragen es in eine relationale Datenbank und lernen die dazu gehörenden Grundbegriffe kennen. Außerdem erwerben sie Kenntnisse zur Auswertung der Datenbank mit Hilfe von SQL-Abfragen. Die Beschränkung auf eine Tabelle hält die Komplexität der Datenbank zunächst in gewünschten Grenzen.

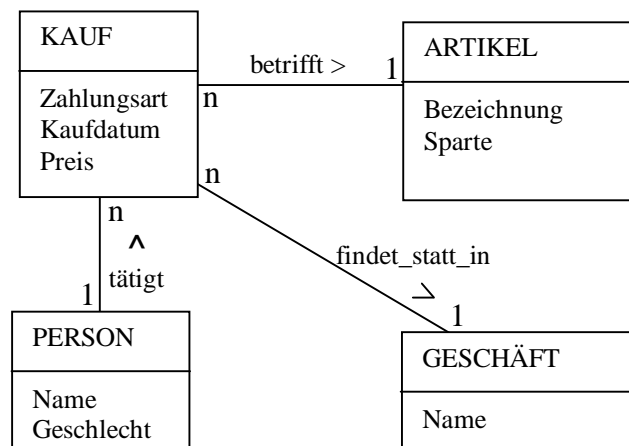
Klassendiagramm und relationale Datenbank (3 Stunden)

In diesem Unterrichtsblock wird der Klassenbegriff aufgegriffen und im Kontext der Datenbankmodellierung verwendet. Das semantische Modell wird anschließend in eine relationale Datenbank umgesetzt. Dazu arbeiten die Schüler erstmals mit dem Datenbankprogramm. Dabei wird auch der Begriff „Schlüssel“ angesprochen. Die Umsetzung wird durch die Betrachtung der Datentypen, ähnlich wie sie aus der funktionalen Modellierung bekannt sind, abgerundet.

Die bisherige Datensammlung mit dem Tabellenkalkulationsprogramm kann nicht alle Anforderungen erfüllen. Zur Einführung der Arbeit mit Datenbanken wird deshalb die Situation „Einkauf“ objektorientiert analysiert. Hier beschränkt sich die Analyse auf das Identifizieren der relevanten Attribute, die in das Diagramm einer Klasse EINKAUF eingetragen werden. Sie sind aus den Spaltenüberschriften der obigen Rechenblatttabelle ersichtlich. Im folgenden Modell gehört zu einem bestimmten Einkauf nur eine bestimmte Ware, die von einer bestimmten Person in einem bestimmten Geschäft gekauft wird.

Hefteintrag:**Klassendiagramm des Einkaufs****Hinweis:**

Der Einbau aller Attribute in eine einzige Klasse ist nicht optimal. In einer besseren Modellierung würde man die einzelnen Klassen GESCHÄFT, PERSON, ARTIKEL und KAUF herausarbeiten sowie zueinander in Beziehung setzen. Erst in späteren Beispielen wird der Schüler anhand von Redundanzproblemen die Notwendigkeit mehrerer Klassen bei der Datenmodellierung einsehen. Auf diese sachgerechtere Modellierung sollte man hinweisen. Sie wird an späterer Stelle im Rahmen einer Übungsaufgabe zur Konzeption von Datenbanken mit mehreren Klassen erfolgen.



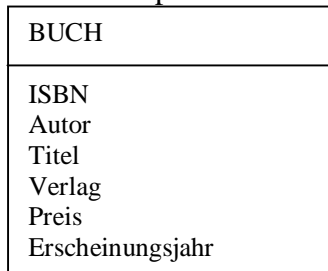
Da das Einführungsbeispiel vorrangig zur Klärung der Begriffe dienen soll und den Einstieg in den Umgang mit dem Datenbankprogramm bildet, ist die Beschränkung auf nur eine Klasse EINKAUF gerechtfertigt; sie ermöglicht in dieser Form eine hinreichende Vielfalt hinsichtlich Datentypen und Abfragen.

Übungsaufgabe:

Du willst die Daten der Bücher in deinem Bücherregal erfassen. Entwirf ein geeignetes Klassendiagramm der Klasse BUCH.

Mögliche Lösung:

Die Lösung hängt vom Kontext ab, beispielsweise davon, ob ein Buch in mehreren Exemplaren vorhanden ist.

**Vom semantischen Modell zur Datenbanktabelle**

Mit dem Klassendiagramm wurde das semantische Schema entwickelt, das sich nun in das logische Schema, hier das Relationenmodell, übertragen lässt. In der Jahrgangsstufe 9 wird man diesen Schritt nicht theoretisch vollziehen, sondern sich gleich mit der Umsetzung in eine relationale Datenbank beschäftigen.

Der mathematische Begriff „Relation“ als Teilmenge der Menge aller möglichen Tupel, welche sich aus den jeweiligen Wertebereichen der Komponenten bilden lassen, ist für die Schüler zu diesem Zeitpunkt unverständlich und wird deshalb nicht thematisiert. Sie sollen nur wissen, dass in relationalen Datenbanken die Daten in Tabellen abgelegt werden. In jedem Datensatz manifestiert sich die „Beziehung der Zusammengehörigkeit“ (Relation) der Werte.

Für den Unterrichtsverlauf ist es hilfreich, die bereits im Tabellenkalkulationssystem erfassten Daten in eine Datenbanktabelle zu übernehmen (Datei Kaufdaten). Bei Verwendung eigener Daten helfen folgende Hinweise zum Import der Exceltabelle:

- Die Spaltenüberschriften müssen in der ersten Zeile stehen.
- Die Spalten, die nicht Attribute von EINKAUF sind, werden gelöscht.
- Datenimport bei MSAccess: Datei – Externe Daten – Importieren – Fortsetzung dialoggesteuert.
- Datenexport von MSAccess nach MySQL: Hinweise hierzu finden sich im Begleitmaterial „XAMPP_und_mehr.doc“.

Arbeit am Computer:

Die Schüler sammeln erste Erfahrungen mit dem DBMS. Sie öffnen die vorbereitete Datenbank Kaufdaten. Im Unterrichtsgespräch wird die Tabelle „einkauf“ der Klasse EINKAUF gegenübergestellt. Datensätze können ergänzt werden.

EINKAUF
Kunde
Geschlecht
Sparte
Warenbezeichnung
Preis
Zahlungsart
Geschäft
Kaufdatum

einkauf : Tabelle							
Kunde	Geschlecht	Sparte	Warenbezeichnung	Preis	Zahlungsart	Geschäft	Kaufdatum
Klaus Kraplin	m	Lebensmittel	Krustenbrot	2,40 €	bar	Backerei Hold	05.03.2007
Nicole Güting	w	Unterhaltung	Kinokarte	11,00 €	bar	Cinestadt	20.02.2007
Chantal Ambeng	w	Übriges	Möbel	270,00 €	bar	AJCEI Fürth	14.08.2007
Nina Höller	w	Kleider	Hochzeitskleid	649,90 €	Karte	Elestadt	04.09.2007
Nicole Güting	w	Unterhaltung	Kinokarte	11,00 €	bar	Cinestadt	20.02.2007
Günther Gauss	m	Lebensmittel	Braune Limonade	2,00 €	Karte	Comarkt Schweinau	18.03.2006
Claudio Haas	m	Lebensmittel	Brownie	7,80 €	Karte	Euro-Markt Langwasser	30.08.2007
Michael Blue	m	Lebensmittel	Crème Patis	6,00 €	Karte	Supermarkt Langwasser	24.12.2007

Klassenbezeichner schreibt man wie in der Unterstufe in Großbuchstaben. Für die zugehörige Tabelle verwendet man aus didaktischen Gründen den gleichen Namen, schreibt ihn aber in Kleinbuchstaben. Dadurch wird einerseits der Zusammenhang zwischen Klasse und Tabelle sichtbar, andererseits wird deutlich, dass es sich um verschiedene Dinge handelt. Die Bezeichner von Tabellen und Spalten sollten möglichst keine Sonderzeichen (– , / , + usw.) enthalten, weil diese in Abfragen je nach Datenbankprogramm eine unterschiedliche Behandlung erfordern. Auf Sonderzeichen sollte man deshalb bereits im Klassendiagramm verzichten.

In der Datenbank wird jedes Objekt des Einkaufs als Datensatz (Zellwerte einer Zeile), jeder Attributbezeichner als Spaltenüberschrift und der Klassenbezeichner als Tabellennamen abgebildet.

Hefteintrag:

Vom Klassendiagramm zur relationalen Datenbank

In unserem Beispiel entspricht jeder auf dem Kassensbon aufgeführte Kaufvorgang einer Zeile der Datentabelle.

Jedes **Daten-Objekt (Entität)** wird als Datensatz in einer Zeile wiedergegeben. Die Spaltenbezeichnungen nennen die **Attribute**. Die **Attributwerte** des Objekts finden sich in den Zellen der zugehörigen Zeile. Als Tabellennamen wählt man üblicherweise den **Klassenbezeichner**.

Datentypen

In einer Tabelle eines Tabellenkalkulationsprogramms können im Gegensatz zu den später verwendeten Datentabellen Einträge mit unterschiedlichen Datentypen in derselben Spalte stehen. Beim Einfügen weiterer Datensätze in die Datenbanktabelle erkennen die Schüler, dass hier in die Zellen einer Spalte nur bestimmte Werte eingetragen werden können. Man stößt wieder auf ähnliche elementare Datentypen, wie sie aus der funktionalen Modellierung bekannt sind. Anhand des vorliegenden Beispiels lernen die Schüler die wichtigsten im Werkzeug verfügbaren Datentypen mit den dort jeweils verwendeten Bezeichnungen kennen. Zur Veranschaulichung ändert man beispielsweise den Datentyp „Währung“ in „Integer“. Schaut man sich die Tabelle erneut an, sieht man den dadurch bewirkten Informationsverlust. Dieser Vorgang ist nicht rückgängig zu machen.

Hefteintrag:**Wertebereiche**

Jede Spalte hat einen bestimmten **Datentyp** (Domäne), z. B. Zahlenwerte, Text, Datumswerte.

EINKAUF
Kunde
Geschlecht
Sparte
Warenbezeichnung
Preis
Zahlungsart
Geschäft
Kaufdatum

einkauf : Tabelle		
	Feldname	Felddatentyp
	Kunde	Text
	Geschlecht	Text
	Sparte	Text
	Warenbezeichnung	Text
	Preis	Währung
	Zahlungsart	Text
	Geschäft	Text
	Kaufdatum	Datum/Uhrzeit

Der Datentyp der Spalte soll dem Wertebereich des entsprechenden Attributs im Klassendiagramm möglichst genau entsprechen.

Der Aufbau einer Tabelle lässt sich in einfacher Form als **Tabellenschema** wiedergeben:

einkauf (Kunde: Text, Geschlecht: Text, Sparte: Text, Warenbezeichnung: Text, Preis: Währung, Zahlungsart: Text, Geschäft: Text, Kaufdatum: Datum)

Für die Spalte Zahlungsart wäre eigentlich der Datentyp „Aufzählung“ sinnvoll, da nur die Werte „Karte“ und „bar“ (Wertebereich) vorkommen. Die Umsetzung in einer Datenbank ist an dieser Stelle jedoch aufwändig, kann jedoch, falls sie von den Schülern thematisiert wird, gezeigt werden.

Aufgabe und Arbeit am Computer:

Setze die von dir entworfene Klasse BUCH in eine Datenbanktabelle um und ergänze einige Datensätze. Überlege dir dabei, welche Datentypen die Spalten erhalten sollen.

Einführung des Schlüssels

Im Klassenmodell ist jedes Einkaufsobjekt individuell ansprechbar durch seinen Objektbezeichner. Die Bezeichner der Einkaufsobjekte könnten beispielsweise „Einkauf1“, „Einkauf2“ usw. lauten. Die Eindeutigkeit der Objekte muss sich auch in der Datenbank widerspiegeln. Sie ist dann erreicht, wenn sich jeder Datensatz von jedem anderen durch mindestens einen Wert unterscheidet. Allerdings ist die Berücksichtigung aller Spalten unhandlich und oft auch unnötig. Man beschränkt sich auf möglichst wenige Spalten. Die Menge der Spalten, deren Werte jeden Datensatz eindeutig festlegen, nennt man Schlüssel.

Den Schülern muss bewusst werden, dass sich Objekte, die in allen Attributwerten übereinstimmen, nicht ohne Weiteres auf das Tabellenmodell übertragen lassen. Gleiche Zeilen könnten leicht beim mehrfachen Kauf gleicher Waren durch die gleiche Person entstehen. Um die Individualität wieder herzustellen, muss eine weitere Spalte (z. B. laufende Nummer) hinzugefügt werden, die als Schlüssel verwendet wird. Einen derartigen Schlüssel nennt man „künstlich“, weil dieser Spalte kein Attribut der Klasse entspricht.

Hefteintrag (Fortsetzung):

Wir arbeiten mit relationalen Datenbanken, die Daten in Tabellen verwalten. Die Reihenfolge der Spalten und der Datensätze spielt keine Rolle. Je zwei Datensätze müssen sich in mindestens einem Feldwert unterscheiden.

Die Spalte bzw. Spaltenkombinationen, deren Werte jeden Datensatz eindeutig festlegen, nennt man **Schlüssel**.

Mit den Schülern wird diskutiert, welche Spalte oder Spaltenkombination für die Tabelle „einkauf“ als Schlüssel möglich ist.

Aufgabe 1 und Arbeit am Computer:

Versuche, einige der zuvor diskutierten Spaltenkombinationen als Schlüssel in die Tabelle „einkauf“ nachträglich einzuführen. Welche Antwort gibt das Datenbankprogramm auf ungeeignete Kombinationen?

In der beiliegenden Beispieldatenbank kommt letztendlich nur eine zusätzliche Spalte, die beispielsweise eine laufende Nummer enthält, als „künstlicher“ Schlüssel in Frage. Auch wenn die Schüler mit anderen Daten gearbeitet haben, sollen sie die Umsetzung eines solchen Schlüssels im Datenbankprogramm kennenlernen.

Hefteintrag (Fortsetzung):

Wenn sich keine Spaltenkombination als Schlüssel eignet, muss eine weitere Spalte als **künstlicher Schlüssel** eingefügt werden.

Im Tabellenschema wird der Schlüssel unterstrichen:

einkauf (Nr. Zahl, Kunde: Text, Geschlecht: Text, Sparte: Text,
Warenbezeichnung: Text, Preis: Währung, Zahlungsart: Text,
Geschäft: Text, Kaufdatum: Datum)

Anhand der folgenden Beispiele vertiefen die Schüler ihr Verständnis des Schlüsselbegriffs und üben das Auffinden eines geeigneten Schlüssels ein.

Aufgabe 2:

Du hast bereits die von dir entworfene Klasse BUCH in eine Datenbanktabelle übertragen. Suche einen geeigneten Schlüssel und setze ihn in der Datenbank um. Notiere auch das Tabellenschema.

Aufgabe 3:

Du willst eine Datenbank erstellen, die Nachnamen, Vornamen, Adressen (Wohnort, Straße, Hausnummer) und Telefonnummern deiner Freunde enthält. Was wären geeignete Schlüssel?

<input type="checkbox"/> Nachname	<input type="checkbox"/> Telefonnummer	<input type="checkbox"/> Nach- und Vorname
<input type="checkbox"/> Wohnort und Straße	<input type="checkbox"/> Nach-, Vorname, Telefonnummer	<input type="checkbox"/> Vorname, Telefonnummer

Hinweis: Die Beantwortung dieser Aufgabe bietet Anlass zur Diskussion, da sie vom zugrunde liegenden Sachverhalt abhängt. Beispielsweise kann eine bestimmte Handy- bzw. Telefonnummer durchaus mehreren Personen zugeordnet sein.

Aufgabe 4:

Die ausleihbaren Bücher der Unterstufenbibliothek sind in einer Tabelle erfasst. Sie enthält mindestens Autor, Titel, Verlag, Erscheinungsjahr, ISBN-Angabe und Regalnummer.

- a) Überlege dir geeignete Schlüssel.
- b) Entwirf ein Tabellenschema.
- c) In der Bibliothek gibt es drei Exemplare des Buchs „Die unendliche Geschichte“ von Michael Ende. Ist dein unter a) gefundener Schlüssel auch dafür geeignet?

Hinweis: In einer kleinen Bibliothek wird jeder Buchtitel nur in einem Exemplar vorhanden sein. Hier reicht die ISBN-Angabe als Schlüssel. Für Teilaufgabe c) muss zusätzliche Information in den Schlüssel aufgenommen werden, beispielsweise eine Exemplar- oder Inventarnummer.

Einfache Abfragen (2 Stunden)

In diesem Abschnitt erhalten die Schüler einen ersten Einblick in die Informationsgewinnung aus vorhandenen Tabellen unter Verwendung der universellen Datenbanksprache SQL (engl. „Structured Query Language“).

Praxisnahe Fragestellungen zeigen den Schülern, dass die Erfassung umfangreicher Datenmengen nur dann sinnvoll ist, wenn man daraus neue Information gewinnen kann. Dazu formuliert man sogenannte Abfragen, die vom Datenbankmanagementsystem beantwortet werden. Bei einer Abfrage wird aus den vorhandenen Datensätzen eine Teilmenge ausgewählt, die bestimmte Bedingungen erfüllt; von diesen Datensätzen werden ggf. nur bestimmte Spalten betrachtet. Das Ergebnis wird in einer Tabelle, der „Ergebnistabelle“, ausgegeben.

Die Schüler überlegen sich, welche neuen Informationen bestimmte Personengruppen, z. B. Abteilungsleitung oder Kunden, aus der Datenbank Einkauf gewinnen wollen und formulieren dazu sinnvolle Fragen (vgl. Arbeitsblatt_AbfragenGrundbausteine.doc), wie zum Beispiel:

- Welche Einkäufe wurden am 18.3.2006 getätigt?
- Welche Waren wurden überhaupt gekauft?
- Wie hoch ist der Preis jedes einzelnen gekauften Artikels?
- Welche Kosmetikartikel gibt es?
- Welche Produkte hat Nicole Gütling eingekauft?

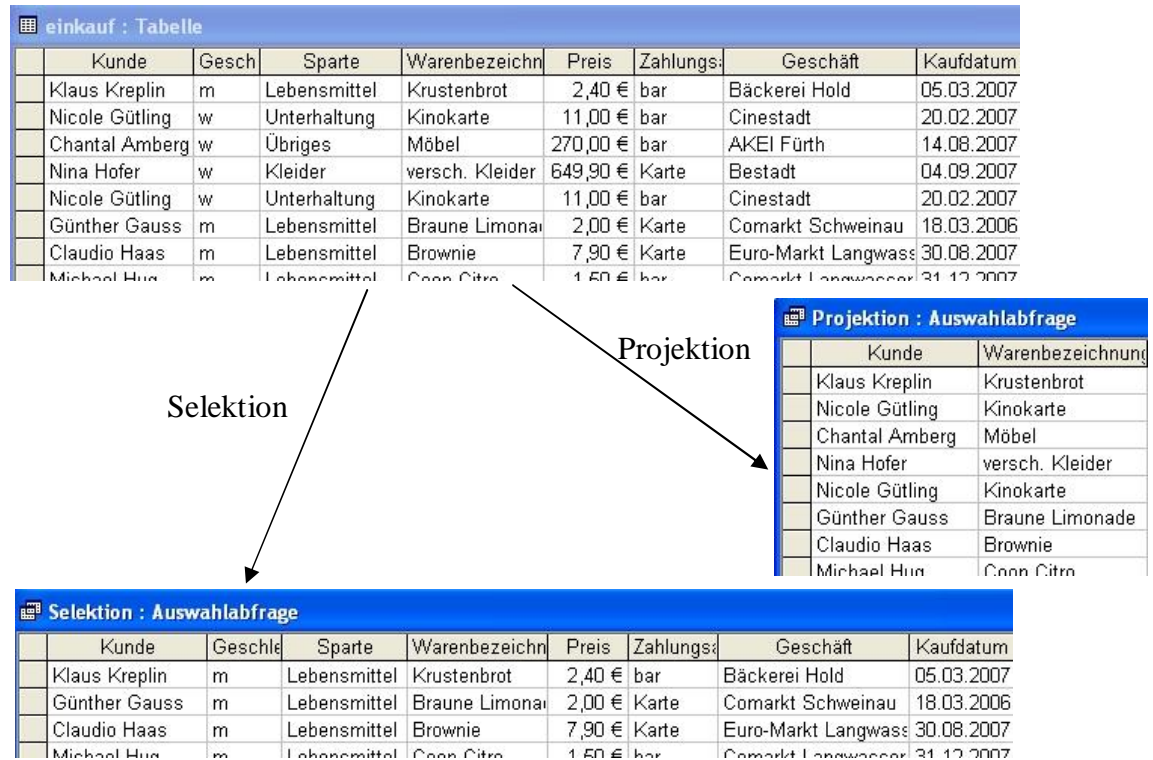
Sobald derartige Fragen formuliert sind, erarbeitet man mit den Schülern, in welcher Form eine Antwort ausgegeben werden sollte. Für die meisten Antworten liegt es nahe, diese als Tabellen auszugeben.

Um die oben entworfenen Fragen anschließend an die reale Datenbank stellen zu können, müssen sie in eine geeignete Sprache umformuliert werden. Auch wenn das verwendete Softwarewerkzeug die Erstellung von Abfragen mittels eines Assistenten ermöglicht und die Schüler damit schnell zu Ergebnissen kommen, sollen sie sich bereits an dieser Stelle konkret mit der Syntax einer Abfrage in SQL befassen; die Formulierung in SQL vermittelt ein tieferes Verständnis der Grundoperationen „Selektion“ und „Projektion“; zudem lassen die Assistenten nur einen eingeschränkten Handlungsrahmen zu.

Hefteintrag oder Arbeitsblatt_AbfragenGrundbausteine.doc:**Abfragen**

Abfragen sollen gezielt Teile der Information wiedergeben. Das Ergebnis einer Abfrage ist eine neue Tabelle (Ergebnistabelle).

Der Aufbau der Ergebnistabelle wird durch zwei Grundverfahren festgelegt:



Eine **Selektion** bedeutet eine Auswahl von Datensätzen (Zeilen) in der Datentabelle mit bestimmten Eigenschaften (z. B. Sparte = 'Lebensmittel').

Eine **Projektion** wählt bestimmte Spalten in der Datentabelle aus (z. B. Kunde und Warenbezeichnung).

In der Regel werden beide Grundverfahren miteinander kombiniert.

Bei der Selektion werden Datensätze ausgewählt, die eine bestimmte Bedingung erfüllen. Eine Bedingung ist ein Ausdruck, dessen Wert „wahr“ oder „falsch“ ist. Sie wird in der WHERE-Klausel von SQL formuliert. Bedingungen können mit den logischen Operatoren AND, OR und NOT verknüpft werden. Spaltenbezeichner, die in Bedingungen im WHERE-Teil der Abfrage auftauchen, müssen nicht unbedingt unter SELECT genannt werden.

Hefteintrag oder Arbeitsblatt_AbfragenGrundbausteine.doc:

Alle Abfragen werden in der werkzeugunabhängigen Datenbankabfrage- und Datenmanipulationssprache SQL (engl. „Structured Query Language“) formuliert.

SQL-Beispiel:

SELECT Warenbezeichnung, Preis

Auswahl bestimmter
Spalten (Projektion)

FROM einkauf

WHERE Preis > 5;

Auswahl der Datensätze
(Selektion) über eine Bedingung

Abfrage_Preis : Auswahlabfrage

	Warenbezeichnung	Preis
	Kinokarte	11,00 €
	Möbel	270,00 €
	versch. Kleider	649,90 €
	Kinokarte	11,00 €
	Brownie	7,90 €
	Honig	7,00 €

Weitere Beispiele für Bedingungen:

Warenbezeichnung = 'Kinokarte'

NOT(Sparte = 'Unterhaltung')

(Kaufdatum = #8/26/2007#) AND (Zahlung <> 'bar')

(Sparte = 'Kleider') OR (Sparte = 'Sport')

Die Darstellung von Datumswerten ist produktspezifisch. Beispielsweise wird der 26.8.2007 in MSAccess als Konstante #8/26/2007# eingegeben, in SQL als '2007-08-26'. Der Datentyp Währung entspricht eigentlich einer Festkommazahl. Das Währungssymbol darf deshalb in den SQL-Abfragen nicht genannt werden.

Anhand der anfangs genannten Fragestellungen üben die Schüler die Umsetzung der logischen Operationen.

Mit den bisherigen Abfragen konnte Information gezielt ausgewählt werden. Sinnvoll ist oft eine besondere Ordnung der Ergebnistabelle.

Hefteintrag oder Arbeitsblatt_AbfragenGrundbausteine.doc:

Mit dem Zusatz

ORDER BY Preis

wird die Ergebnistabelle nach der genannten Spalte aufsteigend geordnet. Der Zusatz

ORDER BY Preis **DESC**

ordnet in absteigender Reihenfolge.

Arbeit am Computer:

Im Arbeitsblatt_AbfragenGrundbausteine.doc sind weitere Aufgabenbeispiele auch zur ORDER-BY-Klausel aufgeführt (Datenbank Kaufdaten):

- Welche Lebensmittel wurden vor dem 1.1.2007 gekauft?
- Welche Waren, die billiger als 20 € waren, wurden nicht bar bezahlt?
- Liste alle Waren auf, die in der Zeit vom 13.9.07 bis 20.9.07 gekauft und mit Karte bezahlt wurden.
- Welches Produkt war am teuersten?
- Welche Waren wurden bei „Brutto Nürnberg“ gekauft? Liste sie in alphabetischer Reihenfolge auf.
- Welche Kunden haben am 5.9.2007 bei „Brutto Nürnberg“ gekauft?
- Martina Groß und Oliver Gross wohnen im gleichen Haus. Liste alle Waren auf, die sie gekauft haben.
- Liste alle Waren jeweils mit dem zugehörigen Preis und Geschäft auf, die von männlichen Kunden für mehr als 20 € gekauft wurden. Die Liste soll nach den Preisen sortiert sein. Lebensmittel sollen ausgeschlossen werden.
- Wer hat am meisten Geld ausgegeben?

Folgende zusätzliche Fragestellungen können von den Schülern untersucht werden:

- Muss die Spalte, nach der geordnet wird, in der SELECT-Klausel vorkommen?
- Kann auch nach mehreren Spalten geordnet werden?
- Welche Auswirkung hat die Reihenfolge der angegebenen Sortierspalten?

Abfragen mit Aggregatfunktionen (3 Stunden)

In diesen drei Stunden erweitern die Schüler ihre Möglichkeiten, aus Datenbanken Informationen zu gewinnen. Sie lernen die wichtigsten Aggregatfunktionen kennen und anwenden. Damit sind sie in der Lage, Daten nicht nur auszuwählen, sondern auch auszuwerten.

Bereits in den Einführungsbeispielen des ersten Kapitels wurde deutlich, dass in vielen Fällen die bloße Auswahl an Information nicht genügt, die erfassten Daten sollen auch ausgewertet werden. Naheliegend sind hier Fragestellungen, wie:

- Wie viel kostet die teuerste Ware?
- Wie viele Waren wurden mit Karte bezahlt?
- Wie viel hat Chantal Amberg insgesamt ausgegeben?

Hierzu werden Aggregatfunktionen benötigt.

Hinweis: Die Aggregatfunktionen werden anhand einiger vorgegebener SQL-Abfragen eingeführt; dabei erarbeiten die Schüler auch die zugrunde liegenden (umgangssprachlich formulierten) Fragestellungen. Eine entsprechende Aufgabe ist im entsprechenden Arbeitsblatt (vgl. Arbeitsblatt_Aggregatfunktionen.doc) formuliert.

Die Einführungsbeispiele können den Schülern auch mit Hilfe der beiliegenden Präsentation „Aggregatfunktionen.ppt“ erläutert werden.

Hefteintrag oder Arbeitsblatt_Aggregatfunktionen.doc:**Aggregatfunktionen**

Aggregatfunktionen berechnen Summe, Durchschnitt, Maximalwert bzw. Minimalwert von Spaltenwerten oder die Anzahl von Datensätzen.

<i>SQL</i>	<i>Bedeutung</i>
SUM(Spalte)	Summe der Spaltenwerte
AVG(Spalte)	Durchschnitt der Spaltenwerte
MAX(Spalte)	Maximalwert der Spaltenwerte
MIN(Spalte)	Minimalwert der Spaltenwerte
COUNT(*)	Anzahl der Datensätze

SQL-Beispiel 1:

```
SELECT MAX(Preis)
```

```
FROM einkauf;
```

einkauf : Tabelle							
Kunde	Gesch.	Sparte	Warenbezeichn.	Preis	Zahlungs-	Geschäft	Kaufdatum
Klaus Knaplin	m	Lebensmittel	Krustenbrot	2,40 €	bar	Bäckerei Hold	05.03.2007
Nicole Götting	w	Unterhaltung	Kinokarte	11,00 €	bar	Cinestadt	20.02.2007
Chantal Amberg	w	Übriges	Möbel	270,00 €	bar	AKEL Fürth	14.08.2007
Nina Hefer	w	Kleider	versch. Kleider	649,90 €	Karte	Bestadt	04.09.2007
Nicole Götting	w	Unterhaltung	Kinokarte	11,00 €	bar	Cinestadt	20.02.2007
Günther Gauss	m	Lebensmittel	Braune Limonade	2,00 €	Karte	Comarkt Schweinsau	18.03.2006
Claudio Haas	m	Lebensmittel	Brownie	7,50 €	Karte	Euro-Markt Langwies	30.08.2007
Michael Maas	w	Lebensmittel	Cheese Pites	4,40 €	bar	Postmarkt Langwies	26.12.2007

Max_Preis : Auswahlabfrage	
Expr1000	
	649,90 €

Die Abfrage wertet die Spalte „Preis“ aus und erzeugt eine Tabelle mit einer neuen Spaltenüberschrift, die in diesem Beispiel vom Datenbankprogramm gewählt wird.

SQL-Beispiel 2:

```
SELECT COUNT(*) AS Anzahl
```

FROM einkauf

WHERE Zahlungsart = 'Karte';

einkauf : Tabelle								
	Kunde	Gesch.	Sparte	Warenbezeichn.	Preis	Zahlungs.	Geschäft	Kaufdatum
	Klaus Krepfin	m	Lebensmittel	Krustenbrot	2,40 €	bar	Bäckerei Hold	05.03.2007
	Nicole Gülling	w	Unterhaltung	KinoKarte	11,00 €	bar	CineStadt	20.02.2007
	Chantal Amberg	w	Übriges	Möbel	270,00 €	bar	AKEI Fürth	14.08.2007
	Nina Heller	w	Kleider	versch. Kleider	649,90 €	Karte	Bestadt	04.09.2007
	Nicole Gülling	w	Unterhaltung	KinoKarte	11,00 €	bar	CineStadt	20.02.2007
	Günther Gauss	m	Lebensmittel	Braune Limonade	2,00 €	Karte	Comarkt Schweinau	18.03.2006
	Claudio Haas	m	Lebensmittel	Brownie	7,90 €	Karte	Euro-Markt Langwass	30.08.2007
	Michael Hug	m	Kunde	Gesch.	Sparte	Warenbezeichn.	Preis	Zahlungs.
	Judith Meyer	w	Nina Heller	w	Übriges	versch. Kleider	649,90 €	Karte
	Annadee Schütz	w	Comarkt-Cards	m	Lebensmittel	Braune Limonade	2,00 €	Karte
	Christina Schmei	w	Audio-Hees	m	Lebensmittel	Brownie	7,90 €	Karte
	Philipp Schmei	w	Arvidas Schütz	w	Lebensmittel	10-Minuten-Brot	2,80 €	Karte
	Philipp Mischler	w	Comarkt Schmei	m	Lebensmittel	Milchbrötchen	1,20 €	Karte
	Monika Hoffman	w	Einige Schmei	m	Unterhaltung	KinoKarte	11,00 €	Karte
	Andreas Brande	m	Einige Schmei	m	Lebensmittel	Kaffeeersatz	17,95 €	Karte
	Christian Schmei	m	Comarkt Schmei	m	Kleider	St. Jakobs	24,90 €	Karte
	Andreas Brande	w	Einige Schmei	m	Lebensmittel	Brot	14,95 €	Karte
	Trina Thöbchen	m	Lebensmittel	Brot	1,40 €	Karte	Comarkt	04.09.2007
	Florian Thöbchen	m	Lebensmittel	Brot	1,40 €	Karte	Comarkt	04.09.2007

Kartenzahlung : Auswahlabfrage

Anzahl

63

Zunächst wird die Bedingung Zahlungsart = 'Karte' ausgewertet. Anschließend wird die Zahl der Datensätze bestimmt und in einer Tabelle mit der gewünschten Spaltenüberschrift „Anzahl“ ausgegeben.

Das Zeichen „*“ in der Funktion COUNT() steht für die Liste aller Spaltennamen. Hier könnte auch COUNT(Kunde) oder COUNT(Warenbezeichnung) stehen; COUNT zählt nur die Anzahl der Einträge.

Für die Auswertung des Kaufverhaltens ist es interessant, wie hoch der Anteil der Bar- bzw. Kartenzahlungen bei allen Einkäufen eines Kunden ist. Eine Vorstufe dazu ist die Information, wie oft jeder Kunde bar bezahlt hat. Dafür müssen die Datensätze – beschränkt auf Barzahlungen – hinsichtlich der Kunden zusammengefasst und dann für jeden Kunden einzeln ausgezählt werden. Kunden, die nie bar gezahlt haben, werden dabei allerdings nicht erfasst.

Hefteintrag oder Arbeitsblatt_Aggregatfunktionen.doc:**SQL-Beispiel 3:**

```

SELECT Kunde, COUNT(*) AS Anzahl
FROM einkauf
WHERE Zahlungsart = 'bar'
GROUP BY Kunde;

```

einkauf : Tabelle

Kunde	Gesch	Sparte	Warenbezeichn	Preis	Zahlungs-	Geschäft	Kaufdatum
Klaus Kreplin	m	Lebensmittel	Krustenbrot	2,40 €	bar	Bäckerei Hold	05.03.2007
Nicole Gütling	w	Unterhaltung	Kinokarte	11,00 €	bar	Cinestadt	20.02.2007
Chantal Amberg	w	Übriges	Möbel	270,00 €	bar	A&E Fürth	14.08.2007
Nina Holler	w	Kleider	versch. Kleider	649,90 €	Karte		
Nicole Gütling	w	Unterhaltung	Kinokarte	11,00 €	bar		
Günther Gauss	m	Lebensmittel	Braune Limonade	2,00 €	Karte		
Claudio Haas	m	Lebensmittel	Brownie	7,90 €	Karte		
Michael Hug	m	Lebensmittel	Coop Citrus	1,50 €	bar		
Judith Meyer	w	Lebensmittel	Honig	7,00 €	bar		
Amadeus Schütz	w	Lebensmittel	10-Minuten-Reiz	3,60 €	Karte		
Christina Schmei	w	Lebensmittel	Milchdrink Bio	1,70 €	Karte		
Philipp Schmei	m	Unterhaltung	Kinokarte	8,00 €	Karte		
Philipp Mächler	m	Lebensmittel	Kalbmedaillon	15,25 €	Karte		
Monika Hoffman	w	Übriges	Ajax Bad	4,10 €	bar		
Andreas Brande	m	Lebensmittel	Schokolade	1,75 €	bar		
Christian Schmei	m	Lebensmittel	Bio Ei	3,30 €	bar		
Andreas Brande	m	Lebensmittel	Milchshake	1,90 €	bar		

Zwischenschritt (grau hinterlegt):

Kunde	Geschlecht	Sp
Alicia Solis	w	Kleide
Alicia Solis	w	Leben
Alicia Solis	w	Übrige

Barzahler : Auswahlabfrage

Kunde	Anzahl
Alicia Solis	3
Aline Hauenstein	2
Amanda Guyot	4
Andreas Muetar	3

Nach der Auswertung der Bedingung Zahlungsart = 'bar' werden die Datensätze nach Kunden zu Gruppen zusammengefasst. Für jede Gruppe wird die Zahl der Datensätze bestimmt und zusammen mit dem Namen des Kunden in einer Tabelle ausgegeben.

Unter SELECT dürfen außer den Aggregatfunktionen nur Spalten erscheinen, die unter GROUP BY aufgeführt sind.

Hinweis: Die Darstellung des Zwischenschritts im grau hinterlegten Kasten zeigt nur eine verkürzte Tabelle. Tatsächlich werden intern hier noch alle Spalten berücksichtigt.

Im folgenden Beispiel werden die Gedankengänge anhand einer weiteren Aggregatfunktion nochmals dargestellt. Die Abfrage wird durch die ORDER BY-Klausel ergänzt, mit der die Ausgabetabelle sortiert wird.

Hefteintrag oder Arbeitsblatt_Aggregatfunktionen.doc:**SQL-Beispiel 4:**

```
SELECT Sparte, SUM(Preis) AS Summe
```

```
FROM einkauf
```

```
WHERE Kaufdatum = #8/26/2007#
```

```
GROUP BY Sparte
```

```
ORDER BY SUM(Preis) DESC;
```

einkauf : Tabelle

Kunde	Gesch	Sparte	Warenbezeichn	Preis	Zahlungs-	Geschäft	Kaufdatum
Klaus Kreppl	m	Lebensmittel	Krustenbrot	2,40 €	bar	Bäckerei Hold	05.03.2007
Nicole Gülling	w	Unterhaltung	Kinokarte	11,00 €	bar	CineStadt	20.02.2007
Chantal Amberg	w	Übriges	Möbel	270,00 €	bar	AKEI Fürth	14.08.2007
Nina Hoffer	w	Kleider	versch. Kleider	649,90 €	Karte	Bestadt	04.09.2007
Nicole Gülling	w	Unterhaltung	Kinokarte	11,00 €	bar	CineStadt	20.02.2007
Günther Gauss	m	Lebensmittel	Braune Limonade	2,00 €	Karte	Comarkt Schweinau	18.03.2006
Claudio Haas	m	Lebensmittel	Brownie	7,90 €	Karte	Euro-Markt Langerach	30.06.2007
Michael Hug	m	Lebensmittel	Coop Citro	1,50 €	bar	Comarkt	
Judith Meyer	w	Lebensmittel	Hong	7,00 €	bar	Brutto N...	
Annadere Schütz	w	Lebensmittel	10-Minuten-Rei	3,60 €	Karte	Imagi Sch...	
Christina Schmei	w	Lebensmittel	Milchdrink Bio	1,70 €	Karte	Loisl	
Philipp Schmei	m	Unterhaltung	Kinokarte	8,00 €	Karte	CineStadt	
Philipp Mächler	m	Lebensmittel	Kollbarnedailon	15,25 €	Karte	Brutto N...	28.03.2007
Monika Hoffm							
Andreas Bra							
Christian Sc							
Karlheinz Dost							

Auswahl der Datensätze mit Kaufdatum 26.8.2007

Bildung von Gruppen gleicher Sparte, Anwendung der SUM-Funktion auf diese Gruppen, Auswahl der Spalten

Sparte	Summe
Büroartikel	2,10 €
Lebensmittel	10,05 €
► Übriges	1,90 €

Absteigendes (DESC) Sortieren nach den entstandenen Summenwerten

Beispiel 4 : Auswahlabfrage

Sparte	Summe
Lebensmittel	10,05 €
Büroartikel	2,10 €
► Übriges	1,90 €

Die Abfrage gibt eine Liste aller Sparten aus, in denen am 26.8.2007 Waren gekauft wurden, unter Angabe der jeweiligen Preissumme. Die Liste ist nach der Höhe der Preissummen absteigend sortiert.

Arbeit am Computer (Arbeitsblatt_Aggregatfunktionen.doc):

Die Erstellung von Abfragen wird anhand vieler Aufgabenbeispiele zur Datenbank Kaufdaten vertieft:

- Wie viel kostet das teuerste Produkt?
- Wie viele Waren der Sparte „Lebensmittel“ wurden gekauft?
- Wie viele Waren wurden bei gekauft?
- Wie viel Geld wurde für Sportartikel ausgegeben?
- Wie viele Waren, die mehr als 10 €kosten, wurden am 2.9.2007 gekauft?
- Wie viele Waren wurden bei am gekauft?

- Vergleiche den Gesamtumsatz bei Kartenzahlung mit dem Gesamtumsatz bei Barzahlung.
- Wie viel kostete ein Kosmetikartikel durchschnittlich?
- Wie viel Geld haben am die männlichen Kunden ausgegeben?
- Liste in alphabetischer Reihenfolge alle Kunden mit der Summe ihrer Ausgaben auf.
- Wie viel Geld haben insgesamt die männlichen Kunden und wie viel insgesamt die weiblichen Kunden ausgegeben?
- Liste alle Geschäfte sortiert nach mittlerem Preisniveau auf.
- Liste alle Sparten auf. Es soll für jede Sparte der gesamte Umsatz am ausgegeben werden. Die Liste soll nach dem Gesamtumsatz sortiert sein, beginnend mit der umsatzstärksten Sparte.
- Liste für jeden Tag die Summe der Preise aller gekauften Waren auf.
- Wer hat das teuerste Produkt gekauft?
- An welchem Tag waren die Einnahmen insgesamt am größten/kleinsten?

Sinnvoll sind auch Rückübersetzungsaufgaben der folgenden Art:

Beschreibe möglichst genau, welche Ausgabe durch folgende SQL-Abfrage erzeugt wird:

```
SELECT Geschäft, AVG(Preis) AS Preisniveau
FROM einkauf
WHERE Kaufdatum > #05/07/2006# AND Kaufdatum < #05/07/2007#
GROUP BY Geschäft
ORDER BY AVG(Preis) DESC;
```

Der in dieser Unterrichtssequenz gegebene Überblick über SQL-Abfragen ist keineswegs vollständig, bietet aber ein ausreichendes Fundament, um typische Fragestellungen an einen Datenbestand beantworten zu können. Problemstellungen, die – wie die beiden letzten Aufgaben – verschachtelte Abfragen erfordern, können auch in zwei Schritten unter Verwendung des Zwischenergebnisses gelöst werden. Die HAVING-Klausel wird nur dann verwendet, wenn die Schüler eine Fragestellung aufwerfen, die ohne diese Klausel nicht beantwortet werden kann.

Nur bei Bedarf wird man den Schülern tiefergehende Hinweise geben, z. B.:

- SELECT DISTINCT ... unterbindet gleiche Zeilen in der Ergebnistabelle.
- Spaltenbezeichner, die Sonderzeichen enthalten, müssen gesondert behandelt werden. In MSAccess werden sie in eckige Klammern geschrieben, in MySQL mit Akzenten eingeschlossen. Deshalb sollten Sonderzeichen hier nicht verwendet werden.

Auf Abfrageassistenten sollte man verzichten. Sie erzeugen oft komplizierte SQL-Ausdrücke durch unnötig viele Klammern und viele HAVING-Bedingungen. Kompliziertere Aufgabenstellungen sind damit nicht immer lösbar.

Datenbanken mit mehreren Tabellen (11.–23. Stunde)

Zur Einführung der Begriffe und zum ersten Kennenlernen der SQL-Abfragen war die Beschränkung auf eine einzige Tabelle sinnvoll. Reale Datenbanken bestehen jedoch in der Regel aus sehr vielen Tabellen. Für die Bearbeitung komplexerer statischer Datenmengen ist die Speicherung in einer einzigen Tabelle ineffizient und fehleranfällig. Bei der Konzeption einer Datenbank wird der Lehrer zwar die Normalformen berücksichtigen, im Unterricht werden diese aber nicht explizit thematisiert, weil bei einem sorgfältigen Entwurf des Klassendiagramms die ersten drei Normalformen in der Regel ohnehin erfüllt sind.

Redundante Informationen (1 Stunde)

In dieser Stunde setzen sich die Schüler mit der Problematik redundanter Daten auseinander; sie erkennen eine ungeeignete Modellierung als Ursache. Da mit einem neuen Beispiel gearbeitet wird, benötigen die Schüler Einlesezeit.

Um den Schülern die Nachteile redundanter Datenhaltung vor Augen zu führen, weicht man hier von der eigentlich sinnvollen Vorgehensweise ab und arbeitet mit einer Datentabelle, ohne vorher ein Klassendiagramm erstellt zu haben. Das Entstehen redundanter Datenhaltung ist durchaus typisch, wenn vorher nicht sorgfältig modelliert wurde. Bewusst wird ein neues Beispiel gewählt, mit dem sich die Problematik augenfälliger zeigen lässt als mit der Einkauf-Datenbank.

Musikgruppen_redundant ist eine schlecht konzipierte Datenbank, die zu redundanten Daten führt (vgl. auch Musikgruppen_redundant.doc).

musik : Tabelle									
	Vorname	Nachname	Geburtsjahr	Instrument	Bandname	Stil	CDTitel	LiedTitel	Länge
	Ge	Waida	1968	Stimme	Doktoren	Punk	Der Blinddarm	Schrei nach Liebe	00:04:20
	Gemma	Hamm	1963	Schlagzeug	Doktoren	Punk	Der Blinddarm	Schrei nach Liebe	00:04:20
	Host	Mi	1962	Gitarre	Doktoren	Punk	Der Blinddarm	Schrei nach Liebe	00:04:20
	Jill	Hutu		Stimme	Katzen	Pop	Sounds Bad	Baby don't you hurt me	00:02:42
	Liza	McKilinan	1981	Stimme	Katzen	Pop	Sounds Bad	Baby <u>don't</u> you hurt me	00:02:42
	Nadja	Kuntze	1983	Stimme	Katzen	Pop	Sounds Bad	Baby don't you hurt me	00:02:42
	Jill	Hutu		Stimme	Katzen	Pop	Sounds Bad	Feels so good	00:03:30
	Liza	McKilinan	1981	Stimme	Katzen	Pop	Sounds Bad	Feels so good	00:03:30
	Nadja	Kuntze	1983	Stimme	Katzen	Pop	Sounds Bad	Feels so good	00:03:30
	Jill	Hutu		Stimme	Katzen	Pop	Sounds Bad	It's OK	00:03:15
	Liza	McKilinan	1981	Stimme	Katzen	Pop	Sounds Bad	It's OK	00:03:15

Mit dieser Datenbank erarbeiten die Schüler, welche Nachteile und Probleme die Mehrfachspeicherung der gleichen Daten mit sich bringt. Abgesehen vom erhöhten Arbeitsaufwand bei der Eingabe und der Größe sowie Unübersichtlichkeit der Tabelle bringen Tippfehler Inkonsistenzen mit sich. Beispielsweise steht in der Datentabelle ein falsch geschriebener Liedtitel der Gruppe „Katzen“. Bei Abfragen können diese Tippfehler zu Mehrdeutigkeiten führen. Sucht man z. B. nach den Liedtiteln der CD „Sounds Bad“, so erhält man zusätzlich einen falschen Titel. Dies wirkt sich insbesondere dann nachteilig aus, wenn der Fehler nicht offensichtlich ist.

Die Ursache dieser Redundanz wird diskutiert. Dabei arbeiten die Schüler heraus, dass hier in einer einzigen Datentabelle Information über Objekte verschiedener Klassen miteinander verknüpft werden. Zur Analyse der gespeicherten Information lässt man die Schüler die Objekte und ihre Klassen herausfinden. Alle Spaltenüberschriften werden daraufhin überprüft, zu welchen Klassen sie als Attribute passen. In obigem Beispiel finden sich vier Klassen (PERSON, GRUPPE, CD, LIED). Diese stehen ihrerseits in Beziehung zueinander.

Hefteintrag:**Redundante Information**

In der Beispieltabelle ist die gleiche Information an vielen Stellen eingetragen. Die Mehrfachspeicherung gleicher Information nennt man **Redundanz**. Beispielsweise werden Liedtitel bei jedem Bandmitglied wiederholt angegeben.

Abgesehen vom erhöhten Eingabeaufwand kann diese Redundanz zu **Inkonsistenzen** (Unstimmigkeiten) führen.

Beispiel: Der Liedtitel ist bei einer Person falsch getippt, bei den anderen Mitgliedern der Band nicht.

Redundanz tritt dann auf, wenn Informationen über mehrere Klassen in einer einzigen Tabelle wiedergegeben werden.

Konzeption einer Datenbank, Klassendiagramm (2 Stunden)

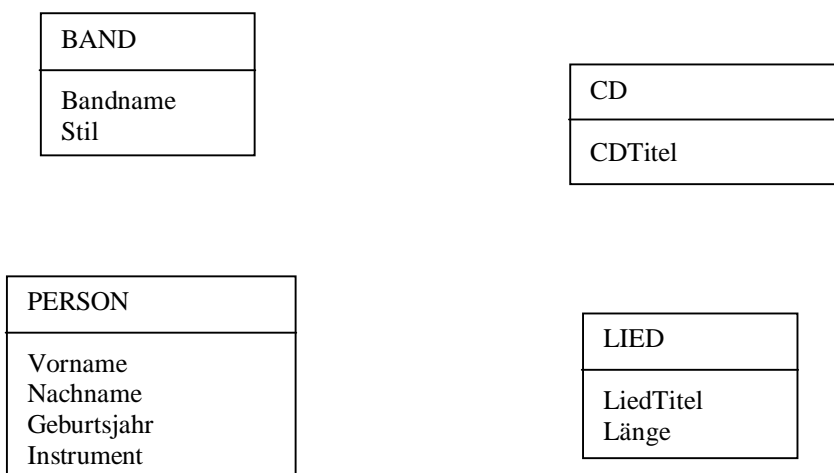
Das Ziel dieses Teilkapitels ist die Modellierung einer Datenbank mit mehreren Klassen und deren Beziehungen zueinander sowie die Umsetzung dieses Modells in Tabellen einer relationalen Datenbank. Zunächst treten nur 1:n-Beziehungen auf.

Nachdem die Klassen für die Musikdatenbank gefunden sind, empfiehlt es sich, das Klassendiagramm Schritt für Schritt (vgl. Arbeitsblatt_Entwurf_Klassendiagramm.doc) zu entwickeln. Im ersten Schritt werden nur die Klassen mit ihren Attributen dargestellt. Es folgen die Beziehungen, die im letzten Schritt durch ihre Kardinalitäten ergänzt werden. Abweichend von der Reihenfolge im Arbeitsblatt kann auch schon nach dem ersten Schritt das Klassenmodell in Tabellen einer Datenbank umgesetzt werden. Die zu jeder Klasse angelegte Tabelle erhält zunächst nur die Attributbezeichner als Spaltenüberschriften. Bei der Wahl geeigneter Schlüssel entsteht in der Tabelle „person“ eventuell eine weitere Spalte mit einem künstlichen Primärschlüssel (Datenbank Musikgruppen_4Tabellen). Erst nach Einführung von Fremdschlüsseln wird die Datenbank durch die Realisierung der Beziehungen ergänzt.

Hefteintrag oder Arbeitsblatt_Entwurf_Klassendiagramm.doc:**Datenbanken mit mehreren Tabellen, Klassendiagramm**

Die redundante Datenbank muss neu geplant werden.

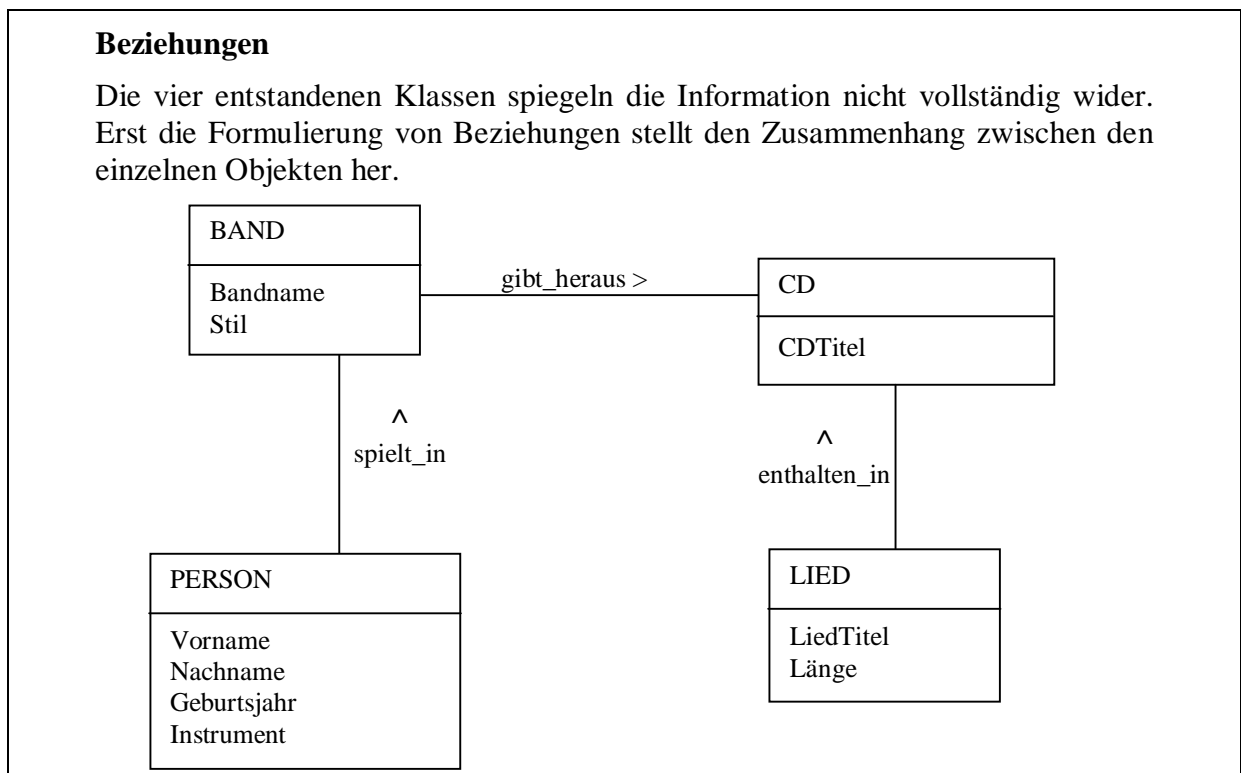
Klassenmodell der Musikgruppen-Datenbank:



Anhand eines einfachen Beispiels wird den Schülern bewusst, dass durch eine Übertragung der Objekte in die Tabellen die ursprüngliche Information noch nicht vollständig erfasst ist. So kann der Datenbank etwa nicht entnommen werden, in welcher Band „Jill Hutu“ spielt. Erst die Formulierung der Beziehungen zwischen den Klassen und die nachfolgende Realisierung in der Datenbank beseitigen diesen Mangel.

Die Modellierung soll Vereinfachungen wie „Jeder Musiker spielt nur in einer Band“, „Jedes Lied ist nur auf einer CD“, „Jede Band hat mindestens ein Mitglied“ und „Auf jeder CD spielt nur eine Band“ enthalten. Derartige Rahmenbedingungen müssen vor jeder Modellierung diskutiert und vereinbart werden. An dieser Stelle bietet es sich an, erneut Grundgedanken der Modellierung (Idealisierung, Abstraktion; vgl. Kapitel 1. 1) anzusprechen.

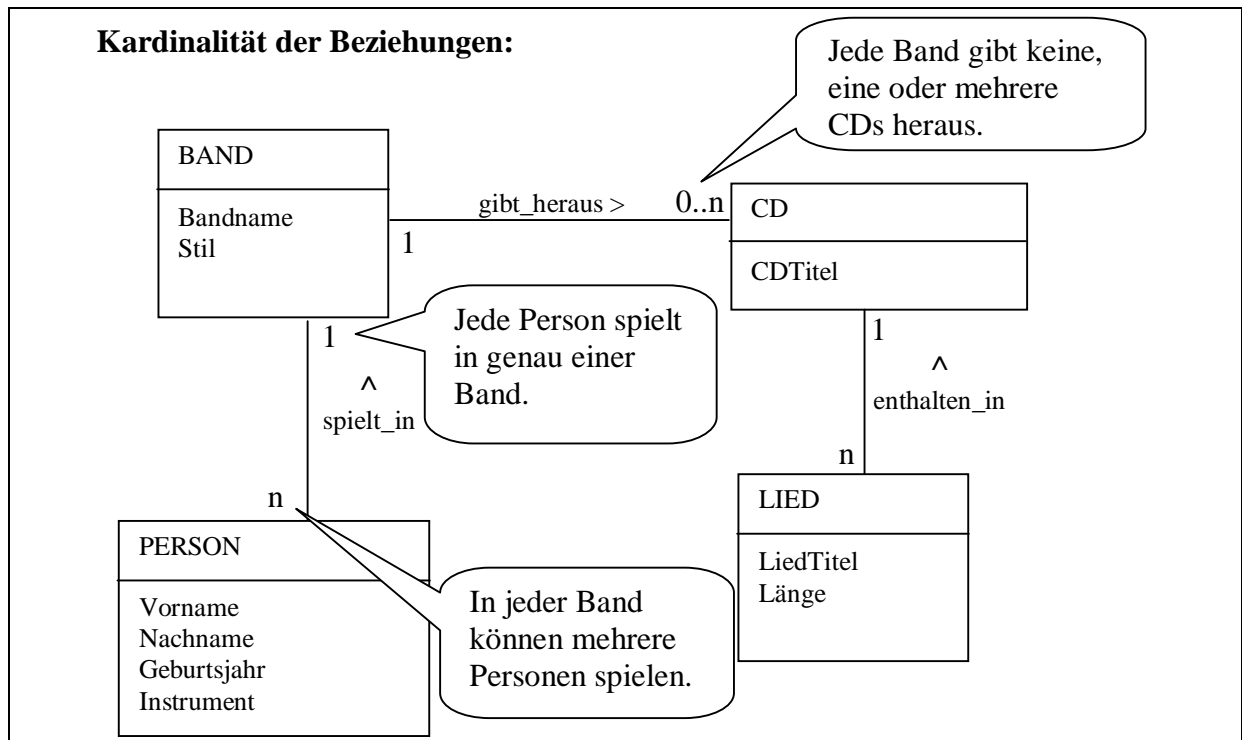
Hefteintrag oder Arbeitsblatt Entwurf_Klassendiagramm.doc:



Vor der Umsetzung in eine relationale Datenbank muss man Überlegungen zu den Kardinalitäten der Beziehungen anstellen. Eine vereinfachte Darstellung einer 1-zu-viele-Beziehung ist bereits aus der Unterstufe bekannt. Sie wird hier aufgegriffen und in eine fachlich detailliertere Nomenklatur übergeführt. Folgende Bezeichnungen für die Kardinalität werden verwendet:

- 1 steht für genau ein Objekt,
- n oder m steht für eines oder viele Objekte,
- „0,1“ steht für kein oder ein Objekt,
- „0..n“ steht für keines, eines oder viele Objekte.

Die Kardinalitäten werden im Arbeitsblatt durch Legenden an den entsprechenden Beispielen eingeführt.

Hefteintrag oder Arbeitsblatt_Entwurf_Klassendiagramm.doc:**Arbeitsauftrag:**

Die Schüler setzen die Klassen zunächst in ein Tabellenschema um. Dabei müssen sie zu jeder Spalte einen geeigneten Datentyp wählen. Außerdem suchen sie zu jeder Tabelle einen geeigneten Primärschlüssel. Für die Tabelle „person“ empfiehlt sich ein künstlicher Schlüssel.

Mögliche Lösung: band (Bandname: Text, Stil: Text)
 cd (CDTitel: Text)
 lied (LiedTitel: Text, Länge: Zeit)
 person (Nr: Zahl, Vorname: Text, Nachname: Text, Geburtsjahr: Zahl, Instrument: Text)

Arbeit am Computer:

Das Tabellenschema wird anschließend in eine Datenbank übertragen. Die Schüler ergänzen geeignete Datensätze (Datenbank Musikgruppen_4Tabellen).

band : Tabelle		
	Feldname	Felddatentyp
	Bandname	Text
	Stil	Text

person : Tabelle		
	Feldname	Felddatentyp
	Nr	AutoWert
	Vorname	Text
	Nachname	Text
	Geburtsjahr	Zahl
	Instrument	Text

cd : Tabelle		
	Feldname	Felddatentyp
	CDTitel	Text

lied : Tabelle		
	Feldname	Felddatentyp
	LiedTitel	Text
	Länge	Datum/Uhrzeit

Aufgabe:

Auch im Einführungsbeispiel zum Thema „Einkauf“ wurde Information über Objekte, die man verschiedenen Klassen zuordnen müsste, in einer einzigen Tabelle zusammengefasst. Finde diese verschiedenen Klassen und entwirf das zugehörige Klassendiagramm einschließlich Beziehungen und Kardinalitäten.

Umsetzung der 1:n-Beziehung – Fremdschlüssel (2 Stunden)

In diesem Abschnitt lernen die Schüler, wie eine 1:n-Beziehung mit Hilfe einer Fremdschlüsselspalte in eine relationale Datenbank umgesetzt wird.

Die Schüler haben bereits für die vier erzeugten Tabellen geeignete Primärschlüssel definiert. Um beispielsweise „Jill Hutu“ wieder mit seiner Band „Katzen“ in Verbindung zu bringen, muss diese Information in den Datensatz der Person „Jill Hutu“ aufgenommen werden. Man benötigt deshalb in der Tabelle „person“ eine zusätzliche Spalte, in der der Primärschlüssel der zugehörigen Band „Katzen“ eingetragen wird. Diese zusätzliche Spalte nennt man „Fremdschlüssel“. Die Überlegung, alternativ in der Tabelle „band“ eine Spalte für die Mitglieder aufzunehmen, scheitert, da jede Band mehrere Mitglieder haben kann. Somit müssten nichtatomare Werte eingetragen werden, auf die dann allerdings nicht einzeln zugegriffen werden könnte. Anhand dieser Überlegung erkennen die Schüler, dass zur Umsetzung einer 1:n-Beziehung der Fremdschlüssel immer in die Tabelle auf der „n-Seite“, hier „person“, eingefügt werden muss.¹

Hefteintrag:

Realisierung der 1:n-Beziehung

Zur Realisierung der 1:n-Beziehung erhält die Tabelle auf der „n-Seite“ eine weitere Spalte. Diese ermöglicht eine eindeutige Zuordnung der Datensätze der verschiedenen Tabellen. Die neue Spalte heißt **Fremdschlüsselspalte**. In ihre Zellen wird jeweils der Primärschlüsselwert des zugehörigen Datensatzes der „1-Seite“ eingetragen.

Beispielsweise wird zur Realisierung der Beziehung „enthalten_in“ in der Tabelle „lied“ als weitere Spalte „CDTitel“ eingefügt. In jedem Datensatz steht dort der Titel der zugehörigen CD.

lied		
LiedTitel	Länge	CDTitel
Emotion	00:05:10	Best of
Everyday	00:03:33	Bang

Fremdschlüssel

cd	
CDTitel	Bandname
Best of	DC
Bang	Good John

Primärschlüssel

Im Tabellenschema wird der Fremdschlüssel unterbrochen unterstrichen:

cd (CDTitel: Text, Bandname: Text)

lied (LiedTitel: Text, Länge: Zeit, CDTitel: Text)

Arbeitsauftrag und Arbeit am Computer:

Die Schüler setzen die Beziehungen im Tabellenschema um und ergänzen die jeweiligen Fremdschlüssel. Anschließend realisieren sie die Beziehungen in der Datenbank Musikgruppen_4Tabellen.

(vgl. auch Datenbank Musikgruppen_mitFremdschlüssel)

¹ Bei der 1:n-Beziehung wird ein Fremdschlüssel auf der Seite angelegt, die zur „1“ verweist, also auf der n-Seite. Bei einer 1:1-Beziehung kann dieser Fremdschlüssel auf einer der beiden Seiten angelegt werden. Auch bei einer 1:0..1-Beziehung kann der Fremdschlüssel auf einer der beiden Seiten angelegt werden. Es ist aber sinnvoll, ihn auf der Seite abzulegen, die zur „1“ verweist, da ansonsten viele NULL-Werte auftreten (vgl. Projektvorschlag „Verwaltung der Schuldaten“ im Abschnitt „Komplexeres Anwendungsbeispiel“).

Damit die Schüler diesen Arbeitsauftrag durchführen können, sollten sie einen Ausdruck der anfänglichen redundanten Tabelle (Musikgruppen_redundant.doc) sowie des vollständigen Klassendiagramms vor Augen haben. So können sie die ursprünglichen Beziehungen wieder nachvollziehen.

Aufgabe:

Übertrage das Klassendiagramm zum Thema „Einkauf“ in eine Datenbank.
Erarbeite vorher das den Klassen jeweils zugrunde liegende Tabellenschema und überlege dir geeignete (Fremd-)Schlüssel.

Abfragen (3 Stunden)

In diesen drei Stunden lernen die Schüler, Information aus mehreren Tabellen zu gewinnen und in einer Ergebnistabelle auszugeben. Sie erkennen das Kreuzprodukt als gedanklichen Zwischenschritt und den „Join“ als Verknüpfung zusammengehöriger Datensätze.

Arbeit am Computer:

Zur kurzen Wiederholung kann man mit einfachen Abfragen innerhalb einer Tabelle beginnen (ergänzte Datenbank Musikgruppen_4Tabellen; vgl. auch Datenbank Musikgruppen_mitFremdschlüssel).

Beispiele (siehe Arbeitsblatt_AbfragenMehrereTabellen.doc):

- Liste alle Gitarrenspieler auf.
- Nenne alle Mitglieder der Band „Doktoren“.
- Nenne alle Stücke der CD „Spain“.
- Welche Lieder dauern länger als 4 Minuten?
- Liste alle CD-Titel jeweils mit dem Stil der Band auf.

Beim ersten Versuch einer Abfrage, die Informationen aus mehreren Tabellen erfordert, z. B. „Liste alle CD-Titel mit dem jeweiligen Stil der Band auf“, werden die Schüler vermutlich eine Unmenge von – größtenteils unerwünschten – Resultaten erhalten.

Die Abfrage

```
SELECT CDTitel, Stil
FROM cd, band;
```

liefert alle Datensätze, die sich aus beliebigen CD-Titeln und Stilrichtungen aller Bands kombinieren lassen (Kreuzprodukt)¹, ohne dass die Beziehung erfüllt ist. Erst die Verknüpfung des Fremdschlüssels in der Tabelle „cd“ mit dem zugehörigen Primärschlüssel der Tabelle „band“ (Join) liefert das gewünschte Ergebnis. Die kurze Präsentation „Kreuzprodukt und Join.ppt“ veranschaulicht das Problem und seine Lösung. Wie auch nachstehender Hefteintrag beruht sie auf gekürzten Tabellen.

¹ Beim Kreuzprodukt zweier Tabellen mit n bzw. m Datensätzen entsteht eine neue Tabelle mit n·m Datensätzen,

z. B. liefert

A1	A2	A3
B1	B2	B3

x

C1	C2
D1	D2
E1	E2

die Tabelle

A1	A2	A3	C1	C2
A1	A2	A3	D1	D2
A1	A2	A3	E1	E2
B1	B2	B3	C1	C2
B1	B2	B3	D1	D2
B1	B2	B3	E1	E2

Hefteintrag oder Arbeitsblatt_AbfragenMehrereTabellen.doc:**Kreuzprodukt und Join**

Beispiel: Liste alle CD-Titel mit dem jeweiligen Stil der Band auf.

Lösungsversuch:

```
SELECT CDTitel, Stil
FROM cd, band;
```

cd	
CDTitel	Bandname
Der Blinddarm	Doktoren
Hell out of it	Devils
Mountains	Tears
When the Devils ride	Devils



band	
Bandname	Stil
Devils	Pop
Doktoren	Punk
Tears	Gothic

Kreuzprodukt	
CDTitel	Stil
Der Blinddarm	Pop
Der Blinddarm	Punk
Der Blinddarm	Gothic
Hell out of it	Pop
Hell out of it	Punk
Hell out of it	Gothic
Mountains	Pop
Mountains	Punk
Mountains	Gothic
When the Devils ride	Pop
When the Devils ride	Punk
When the Devils ride	Gothic

Bei der Verwendung mehrerer Tabellen in einer Abfrage erhält man zunächst alle möglichen (auch unbeabsichtigten) Kombinationen von Datensätzen aller beteiligten Tabellen (**Kreuzprodukt**).

Richtige Lösung:

```
SELECT CDTitel, Stil
FROM cd, band
WHERE cd.Bandname = band.Bandname;
```

Joinbedingung (Abgleich von Fremdschlüssel mit zugehörigem Primärschlüssel)

Kreuzprodukt mit Join	
CDTitel	Stil
Der Blinddarm	Punk
Hell out of it	Pop
Mountains	Gothic
When the devils ride	Pop

Erst der Vergleich des Fremdschlüssels mit dem zugehörigen Primärschlüssel schränkt die Abfrage auf die gewünschten Ergebnisse ein. Allgemein bezeichnet man die Verknüpfung zweier Tabellen als **Join**. Die Joinbedingung verknüpft mit Hilfe des Fremdschlüssels die zusammengehörigen Datensätze der beiden Tabellen. Auf diese Weise wurde auch die 1:n-Beziehung aus dem Klassendiagramm in die Datenbank umgesetzt.

Hinweis: Da in beiden Tabellen eine Spaltenüberschrift „Bandname“ vorkommt, muss der Tabellename vorangestellt werden.

Eine weitere, aber kompliziertere Möglichkeit zur Realisierung der Verknüpfung würde über die Klausel `INNER JOIN` formuliert. Da diese Variante aber eine komplexere Syntax aufweist, empfiehlt sich die Beschränkung auf die oben beschriebene Form.

Arbeit am Computer (Arbeitsblatt_AbfragenMehrereTabellen.doc):

Abfragen auf mehreren Tabellen unter Verwendung von Joinbedingungen werden an zahlreichen Beispielen vertieft.

- Liste alle Musiker aus Pop-Bands auf.
- In welchen Bands spielen Musiker, die vor 1970 geboren wurden?
- Welchen Stil (keine Mehrfachnennungen) haben die Bands, in denen Musiker spielen, die vor 1970 geboren wurden?
- Liste alle Lieder auf, die von Rock-Bands gespielt werden.
- Erzeuge eine Liste aller Daten, so wie du sie als Papierkopie der redundanten Tabelle erhalten hast. Sortiere sie alphabetisch nach Bandname und CDTitel.
- Auf welchen CDs wirkt Katja Biller mit?
- Welche Musiker (Vor- und Nachname) spielen auf der CD „Spain“?
- Wer singt in dem Lied mit dem Titel „Hot Temptation“?
- Welche Lieder, bei denen Jill Hutu mitwirkt, dauern länger als 3:30 Minuten?
- Berechne die Mitgliederzahlen aller Bands mit Hilfe der `COUNT`-Funktion.
- Berechne die Spieldauer jeder erfassten CD.
- Wie lange dauert das längste Lied der Band „Katzen“?
- Wie heißt das längste Lied der Band „Katzen“?

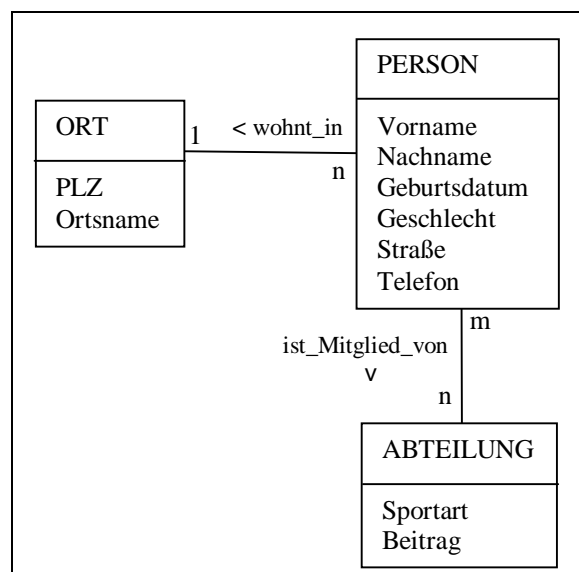
Realisierung der n:m-Beziehung (4 Stunden)

In diesen vier Unterrichtsstunden lernen die Schüler, auch die n:m-Beziehung in einer relationalen Datenbank umzusetzen. Sie erkennen, dass das für die 1:n-Beziehung verwendete Verfahren nicht ausreicht, sondern zusätzlich eine Beziehungstabelle benötigt wird.

Die bisherigen Beispiele enthielten nur 1:n-Beziehungen.

Bei der Modellierung eines weiteren Themas „Sportverein“ tritt eine n:m-Beziehung zwischen den Klassen `ABTEILUNG` und `PERSON` auf: Eine Person kann Mitglied mehrerer Abteilungen sein, umgekehrt sind wohl in jeder Abteilung mehrere Personen Mitglied.

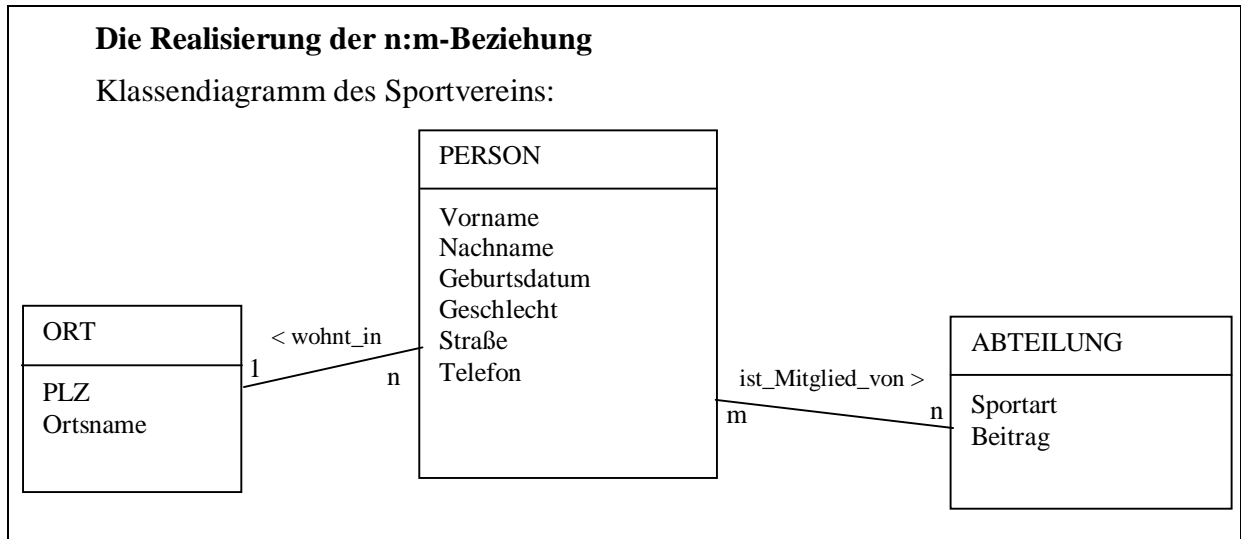
Während im Klassendiagramm n:m-Beziehungen keine wesentliche Neuerung bedeuten, erfordert die Umsetzung in eine relationale Datenbank eine Erweiterung des bisherigen Vorgehens.



Dieses Beispiel wird anhand von `Arbeitsblatt_nm_Bezeichnung.doc` schrittweise durchgeführt.

Arbeitsauftrag (Arbeitsblatt_nm_Beziehung.doc):

Die Schüler erarbeiten ein Konzept zur Verwaltung der Daten eines Sportvereins. Als Klassen sollten mindestens die Mitglieder, ihre Wohnorte und die Abteilungen des Vereins vorkommen. Die Jugendlichen überlegen sich geeignete Attribute und erstellen ein Klassendiagramm mit Kardinalitäten.

Hefteintrag oder Arbeitsblatt_nm_Beziehung.doc:

An dieser Stelle bietet es sich an, auf die Problematik verschiedener Schreibweisen für Kardinalitäten sowie auf die jeweils berücksichtigten Rahmenbedingungen bei der Modellierung hinzuweisen. Sind im Verein auch Mitglieder, die keiner Abteilung angehören, so müsste nach der in der Handreichung verwendeten Schreibweise bei der Beziehung „ist_Mitglied_von“ die Angabe „n“ durch „0..n“ ersetzt werden; die Umsetzung der Beziehung wird dadurch nicht beeinflusst.

Problematik der Umsetzung des Klassendiagramms

Anhand der Präsentation „nm_Beziehung.ppt“ wird den Schülern deutlich, dass die Lösung mit der Fremdschlüsselspalte, wie sie aus der 1:n-Beziehung bekannt ist, hier zu nichtatomaren Einträgen in Tabellenfeldern führen kann:

Würde man z. B. versuchen, die Beziehung über eine Fremdschlüsselspalte „Mitglied“ in der Tabelle „abteilung“ herzustellen, so müssten in dieser Spalte die Primärschlüsselwerte aller Mitglieder dieser Abteilung eingetragen werden. Da es den Datentyp „Reihung von Integer“ nicht gibt, müsste diese Spalte den Datentyp „Text“ erhalten. Die enthaltene Information ist durch Abfragen nicht wiederzugewinnen, da diese Fremdschlüsselspalte „Mitglieder“ mit der Primärschlüsselspalte „M_ID“ nicht über eine Join-Bedingung verknüpft werden kann.

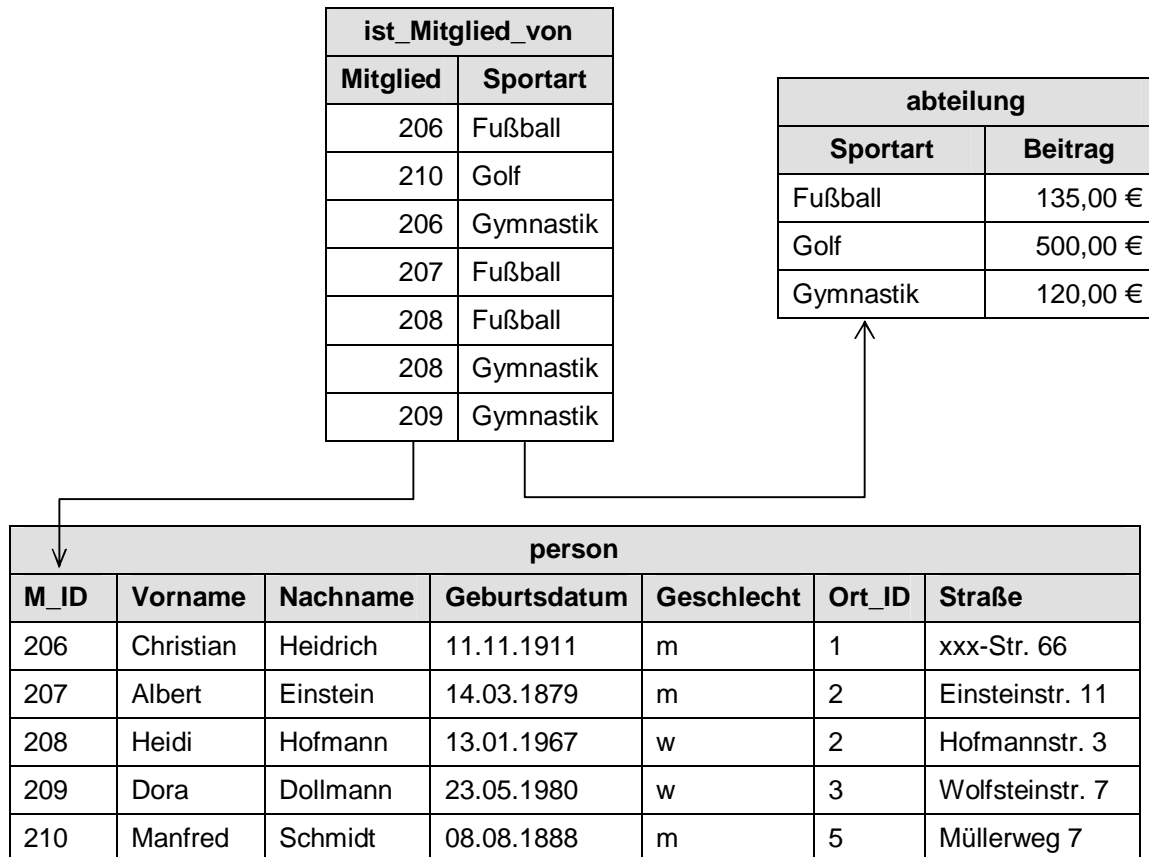
abteilung		
Sportart	Beitrag	Mitglied
Fußball	135,00 €	206 207 208
Golf	500,00 €	210
Gymnastik	120,00 €	208 209 206

person						
M_ID	Vorname	Nachname	Geburtsdatum	Geschlecht	Ort_ID	Straße
206	Christian	Heidrich	11.11.1911	m	1	xxx-Str. 66
207	Albert	Einstein	14.03.1879	m	2	Einsteinstr. 11
208	Heidi	Hofmann	13.01.1967	w	2	Hofmannstr. 3
209	Dora	Dollmann	23.05.1980	w	3	Wolfsteinstr. 7
210	Manfred	Schmidt	08.08.1888	m	5	Müllerweg 7

Entsprechende Schwierigkeiten würden sich auch ergeben, wenn eine Fremdschlüsselspalte „Sportart“ in die Tabelle „person“ eingefügt würde.

Die Grundfrage zur Lösung des Problems „Welche Datensätze in den Tabellen ‚abteilung‘ und ‚person‘ muss man aufgrund der realen Beziehung zusammenklammern?“ führt zu einer Tabelle von „Klammern“, wobei jede „Klammer“ durch das Paar der Primärschlüsselwerte der beiden Datensätze dargestellt wird.

Die Realisierung einer n:m-Beziehung durch eine weitere Tabelle wird den Schülern mit der oben genannten Präsentation veranschaulicht. Die Spalten dieser Beziehungstabelle enthalten die Primärschlüssel der beiden Klassen als Fremdschlüssel. Anhand des nachfolgend aufgeführten Beispiels sollte auch diskutiert werden, welchen Primärschlüssel die neue Beziehungstabelle erhält. Er setzt sich aus den beiden Fremdschlüsselspalten zusammen.



Arbeit am Computer:

Für den Unterrichtsablauf kommen, je nach verfügbarer Zeit, mehrere Varianten in Frage:

- Die Schüler können die ganze Datenbank neu anlegen.
- Die Schüler können eine bereits angelegte Datenbank Sportverein0 mit Daten füllen. Die Tabellen „person“ und „ort“ enthalten eine weitere Spalte als Primärschlüssel (z. B. Ort_ID). Die 1:n-Beziehung zwischen den Klassen PERSON und ORT ist in der Datenbank noch nicht umgesetzt, sondern soll von den Schülern als Übung des bereits Erlernten umgesetzt werden.
- Die Schüler können die bereits mit Daten gefüllte Datenbank Sportverein1 verwenden. Die Spalte Ort_ID ist in der Tabelle „person“ Fremdschlüssel. Die 1:n-Beziehung zwischen PERSON und ORT ist hier bereits realisiert.

Die n:m-Beziehung zwischen den Klassen PERSON und ABTEILUNG ist in jedem Fall noch von den Schülern umzusetzen.

Arbeitsauftrag und Arbeit am Computer (Arbeitsblatt_nm_Bezeichnung.doc):

Die Schüler entwerfen das Tabellenschema zu den obigen Klassen und legen in einer neuen Datenbank die zugehörigen Tabellen an. Sie setzen die 1:n-Beziehung aus dem Klassendiagramm in die Datenbank um. In die Tabellen werden geeignete Daten eingegeben (Datenbank Sportverein0; vgl. auch Datenbank Sportverein1).

Zur Vereinfachung der Eingabe könnte in der Tabelle „ort“ der künstliche Primärschlüssel Ort_ID weggelassen und PLZ als Primärschlüssel verwendet werden. Es gibt allerdings Gegenden, in denen mehrere kleine Orte die gleiche PLZ haben. Ähnliches gilt für die Tabelle

„abteilung“, wo auch „Sportart“ der Primärschlüssel sein könnte. Die Eingabe der Beziehungstabelle wird aber durch den kurzen künstlichen Schlüssel beschleunigt. Außerdem erleichtert der künstliche Schlüssel eine eventuelle spätere Änderung des Sportartbezeichners, da eine derartige Änderung dann nur an einer Stelle vorgenommen werden muss.

Hefteintrag (Fortsetzung):

Zur Realisierung einer n:m-Beziehung muss eine weitere Tabelle angelegt werden, deren Spalten die Primärschlüssel der beiden zu verknüpfenden Tabellen als Fremdschlüssel enthalten. Der Primärschlüssel der Beziehungstabelle ist die Kombination der Fremdschlüsselspalten.

Beispiel:

Die n:m-Beziehung „ist_Mitglied_von“ erfordert in Sportverein0 eine neue Tabelle mit den Spalten „M_ID“ und „Sport_ID“. Diese werden mit den entsprechenden Spalten der Tabellen „person“ und „abteilung“ verknüpft.

Arbeitsauftrag und Arbeit am Computer (Arbeitsblatt_nm_Beziehung.doc):

Die Schüler erstellen in ihrer Datenbank Sportverein die Beziehungstabelle „ist_Mitglied_von“ und überlegen sich den zugehörigen Primärschlüssel.

Sie ordnen mit Hilfe der Beziehungstabelle **einige** Mitglieder **einigen** Abteilungen zu. Es sollte Mitglieder geben, die mehreren Abteilungen angehören. Ebenso sollten die Sportabteilungen jeweils mehr als ein Mitglied haben (Datenbank Sportverein1; vgl. auch Datenbank Sportverein1_nm).

Arbeit am Computer: Die Schüler führen Anfragen an die ergänzte Datenbank Sportverein1 durch (Arbeitsblatt_nm_Beziehung.doc; vgl. auch Datenbank Sportverein1_nm):

- In welchen Orten wohnen die Mitglieder der Fußball-Abteilung?
- Liste in alphabetischer Reihenfolge alle Mitglieder der Handball-Abteilung auf, die vor dem 1.1.1982 geboren wurden.
- In welchen Abteilungen sind Mitglieder aus Augsburg?
- Ist eine Person Mitglied mehrerer Abteilungen, so muss sie die Summe der Beiträge entrichten. Erstelle eine Liste mit den Beiträgen aller Mitglieder.
- Liste alle Orte auf, in denen Vereinsmitglieder wohnen. Hinter jedem Ort soll die Anzahl der Vereinsmitglieder aus diesem Ort angegeben werden.
- Jeder Abteilungsleiter möchte wissen, wie viele männliche und wie viele weibliche Mitglieder seine Abteilung besitzt. Erstelle eine entsprechende Liste.

Manchmal ist es sinnvoll, Datensätze ausfindig zu machen, auf die in den Beziehungstabellen gerade nicht verwiesen wird. Beispielsweise werden Personen, die in keiner Abteilung Mitglied sind, als mögliche Karteileichen gesucht. Dies erfordert die Kenntnis zweier neuer Möglichkeiten, die in leistungsstarken Klassen durchaus angesprochen werden können:

- Geschachtelte Abfragen wurden bereits in den Lösungen zu einigen Aufgaben erwähnt. Eine Ergebnistabelle mit nur einer Zelle konnte dort wie ein konkreter Wert verwendet werden.
- Umfasst die Ergebnistabelle eine Spalte mit mehreren Zeilen, so kann diese als Menge aufgefasst werden. Mit dem Operator IN kann überprüft werden, ob ein Wert in dieser Menge enthalten ist.

Die oben angesprochenen Karteileichen würden damit folgendermaßen gefunden:

```
SELECT *  
FROM person  
WHERE NOT(M_ID IN (SELECT M_ID FROM ist_Mitglied_von));
```

Dieses Beispiel findet sich nicht auf dem Arbeitsblatt, da es das Anforderungsniveau des Lehrplans übersteigt.

Datenintegrität (1 Stunde)

Daten in einer Datenbank können unvollständig, nicht korrekt oder widersprüchlich sein. Aufgabe eines Datenbankprogramms ist es, derartige Fehler – soweit möglich – zu erkennen oder sogar zu verhindern.

Das Herstellen von referentieller Integrität und die Eingabe von Integritätsbedingungen sind in den meisten DBMS einfach möglich. In diesem Fall können die Schüler selbst damit Erfahrungen sammeln. In MSAccess lässt sich referentielle Integrität als Eigenschaft der Fremdschlüsselbeziehung, die mit Hilfe des Beziehungsassistenten graphisch eingegeben wurde, auswählen. Auch Löschweitergabe (ON DELETE CASCADE) und Aktualisierungsweitergabe (ON UPDATE CASCADE) sind einstellbar. In MySQL wird die referentielle Integrität einschließlich des Weitergabeverhaltens über Popup-Menüs der Client-Programme definiert, vorausgesetzt wird der Tabellentyp InnoDB bei beiden Tabellen sowie ein Index für den Fremdschlüssel.

Professionelle Datenbanksysteme verfügen über weitere Möglichkeiten für Integritätsbedingungen. So können beispielsweise für einzelne Attribute spezielle Wertebereiche oder sonstige Bedingungen angegeben werden. Eine Eingabe oder Änderung eines Datensatzes wird verboten, wenn sie den Bedingungen widerspricht.

Hefteintrag:**Datenintegrität**

Ein gutes Datenbankprogramm hilft dabei, dass die eingegebenen Daten vollständig, korrekt und widerspruchsfrei (= integer) sind und bleiben. Dies bezeichnet man als Datenintegrität.

Das Datenbankprogramm muss unabsichtliche Fehlbedienungen oder auch beabsichtigte Fehleingaben erkennen und Kontrollmöglichkeiten besitzen, um die Zulässigkeit der Eingaben zu prüfen.

Bei der Definition einer Datenbank kann man bestimmte Integritätsregeln für die Felder der Datenbank festlegen.

Beispiele:

- Das Geburtsdatum eines Kunden darf nicht vor dem 1.1.1900 liegen.
- Das Geschlecht einer Person muss männlich oder weiblich sein.
- In der als Primärschlüssel vorgesehenen Spalte oder in einer anderen für die Benutzung wesentlichen Spalte darf kein leerer Wert stehen, weil sonst ein eindeutiger Bezug nicht mehr möglich ist.
- In der Fremdschlüsselspalte Ort_ID der Tabelle „person“ dürfen nur Werte stehen, die auch in der Primärschlüsselspalte der Tabelle „ort“ vorkommen (referentielle Integrität). Stellt man z. B. die referentielle Integrität der Beziehung zwischen den Spalten Ort_ID der beiden Tabellen „ort“ und „person“ her und möchte dann in der Tabelle „person“ in der Spalte Ort_ID den Wert „200“ eintragen, so würde dieser Datensatz abgewiesen, falls in der Tabelle „ort“ kein Datensatz mit dem Primärschlüsselwert „200“ existiert.

Wenn es das Datenbankprogramm zulässt, sollen die Schüler selbst die Auswirkungen der referentiellen Integrität anhand einiger Beispiele erfahren. Das auf der Begleit-CD verfügbare Arbeitsblatt_Integrität.doc orientiert sich am Datenbankprogramm MSAccess, für andere Programme sind ähnliche Fragestellungen empfehlenswert:

Arbeit am Computer (Arbeitsblatt_Integrität.doc):

Die Schüler fügen in der Datenbank Musikgruppen_mitFremdschlüssel alle Fremdschlüssel-Beziehungen graphisch ein. Sie stellen die referentielle Integrität als Eigenschaft der Beziehung her.

- Was passiert, wenn du in der Tabelle „lied“ einen weiteren Datensatz einfügst und in der Spalte CDTitel den Wert „Greatest Hits“ einträgst?
- Was musst du unternehmen, damit der oben genannte Datensatz nicht abgewiesen wird?
- Versuche, in der Tabelle „band“ den Datensatz von „Doktoren“ zu löschen. Ist das möglich? Wie wirkt sich das gegebenenfalls auf die anderen Tabellen aus?
- Korrigiere Schreibfehler bei den Bandnamen. Schreibe z. B. „For“ statt „Four“. In welcher Tabelle sollte man die Änderung überhaupt vornehmen? Ist sie möglich? Wie wirkt sich das gegebenenfalls auf die anderen Tabellen aus?

- Welche Eigenschaften hinsichtlich referentieller Integrität, Aktualisierungsweitergabe und Löschweitergabe sollten in der ergänzten Datenbank Sportverein1 die Beziehungen zu den Tabellen „person“ und „abteilung“ haben? Füge die gewünschten Einstellungen in die Datenbank ein.

Einschränkung der Sicht auf Daten (24. Stunde)

Die Schüler sollen in dieser Stunde einen Einblick in die Möglichkeiten erhalten, die Sicht auf Daten einzuschränken.

Oft möchte man bestimmten Personen nicht das gesamte Datenmaterial zugänglich machen, sei es aus Gründen des Datenschutzes oder um für die jeweilige Person unwesentliche Daten auszublenden. Sogenannte Views („Sichten“) erlauben es, den vorhandenen Datenbestand unter bestimmten Gesichtspunkten geordnet darzustellen. Eine View wird durch eine Select-Anweisung festgelegt.

In MSAccess wird jede gespeicherte Abfrage automatisch als View behandelt. Die entsprechende Ausgabetabelle kann wie eine Datentabelle in Abfragen verwendet werden. Die SQL-Anweisung CREATE VIEW ist deshalb explizit nicht vorgesehen.

In MySQL wird eine View explizit mit der SQL-Anweisung CREATE VIEW erzeugt. Sie wird wie eine Tabelle in die Datenbank eingetragen.

Im Beispiel der von den Schülern ergänzten Musikdatenbank lässt sich eine View, die zu jeder CD den Namen der Band und die Anzahl der Lieder auf der CD angibt, wie folgt erstellen:

```
CREATE VIEW CD_Übersicht AS
SELECT cd.CDTitel, Bandname, COUNT(*) AS Anzahl
FROM cd, lied
WHERE cd.CDTitel = lied.CDTitel
GROUP BY Bandname, cd.CDTitel;
```

Die Ausgabetabelle der Select-Anweisung wird wie eine eigenständige Tabelle behandelt. Im Gegensatz zu einer „echten“ Tabelle entstehen hier aber keine redundanten Daten. Werden die zugrunde liegenden Tabellen geändert, indem z. B. eine neue CD eingefügt wird, ändert sich automatisch auch das Ergebnis der View CD_Übersicht.

Views sind aber keine ausschließlichen Leseansichten. Views können auch zur Dateneingabe und -änderung verwendet werden, wenn bei der Festlegung der View

- GROUP BY, DISTINCT, HAVING nicht vorkommen,
- im SELECT-Teil nur Spalten einer Tabelle vorkommen,
- keine Unterabfragen in der SELECT-Klausel vorkommen oder
- keine Aggregatfunktionen vorkommen.

Wenn eine dieser Bedingungen verletzt ist, wird die View automatisch zur Read-Only-View, d. h., man kann Daten der View nicht ändern, löschen oder erweitern. Mit den Schülern diskutiert man, warum diese Einschränkungen erforderlich sind. (Hinweis: Nur mit diesen Einschränkungen ist das Auffinden des Originaldatensatzes möglich, der z. B. gelöscht werden soll.)

Im ersten Beispiel können deshalb keine Datenänderungen in der View-Tabelle vorgenommen werden. Im nachfolgenden Beispiel, das sich auf die Datenbank

Sportverein1_nm bezieht, ist es jedoch möglich, die Ergebnistabelle der View zu bearbeiten und beispielsweise die Adresse einer Person der Abteilung Fußball zu ändern.

```
CREATE VIEW Fussballer AS
SELECT Vorname, Nachname, Geburtsdatum, Geschlecht, Strasse, Telefon, Ort_ID
FROM person, abteilung, ist_Mitglied_von
WHERE person.M_ID = ist_Mitglied_von.M_ID
AND ist_Mitglied_von.Sport_ID = abteilung.Sport_ID
AND Sportart = 'Fußball';
```

Arbeit am Computer:

Die Schüler sollen diese oder ähnliche Beispiele aus bereits in den Vorstunden durchgeführten Abfragen bearbeiten. Sie untersuchen, wie sich die erzeugten Views (Tabellen) bei Änderung des Datenbestandes ändern bzw. wie die ursprünglichen Daten durch Einfügen in die oder Ändern der View-Tabelle beeinflusst werden.

Datenpflege (25.–26. Stunde)

Ziel dieser Unterrichtseinheit ist es, den Schülern einen Einblick zu geben, wie in einer Datenbank nicht nur einzelne Daten eingefügt, gelöscht oder geändert werden können, sondern ganze Datenbestände mit einer einzigen Anweisung aktualisiert werden. Dabei lernen die Jugendlichen die Sprache SQL nicht nur als Datenbankabfrage-, sondern auch als Datenbankmanipulationssprache kennen. Die im Arbeitsblatt_Datenpflege.doc genannten Aufgabenbeispiele beziehen sich auf verschiedene Datenbanken. Die Fragestellungen können jedoch unschwer – je nach Unterrichtssituation vor Ort – auch auf eine einzige Datenbank aus dem Angebot des Begleitmaterials zugeschnitten werden.

Die Schüler haben bereits einzelne Datensätze mit Hilfe der graphischen Oberfläche verändert, eingefügt oder gelöscht. Bisweilen ergibt sich die Notwendigkeit, mehrere Datensätze mit einer einzigen Anweisung zu manipulieren; dies ist mit der graphischen Oberfläche nur mit sehr großem Aufwand möglich. Zudem kann der Zugriff auf ein Datenbanksystem aus anderen Programmen heraus nur über SQL-Anweisungen erfolgen.

Änderungsanweisung (UPDATE)

Zur Motivation dient folgendes Beispiel (Arbeitsblatt_Datenpflege.doc):

Beispiel 1:

Das Geschäft „Oldi“ wurde von der Handelskette „Brutto“ übernommen. In der Datenbank Kaufdaten sollen deshalb alle betreffenden Datensätze geändert werden.

Die Änderung jedes einzelnen Datensatzes der Tabelle „einkauf“ in der Datenbank Kaufdaten wäre mühsam und fehleranfällig. SQL stellt folgende Lösung zur gleichzeitigen Änderung aller Datensätze zur Verfügung.

```
UPDATE einkauf
SET Geschäft = 'Brutto'
WHERE Geschäft = 'Oldi';
```

Anhand dieses Beispiels wird allgemein die Syntax der Änderungsanweisung vorgestellt:

```
UPDATE Tabelle
SET Spalte1 = NeuerWert1, Spalte2 = NeuerWert2, ...
WHERE ... ;
```

Bedingung zur Auswahl der
zu ändernden Datensätze

(Hinweis: Die TermAuswertungen für NeuerWert1, NeuerWert2, ... werden für jeden gefundenen Datensatz ausgeführt; erst anschließend werden die Ergebnisse in die Zielfelder Spalte1, Spalte2, ... eingetragen.)

Arbeit am Computer (Arbeitsblatt_Datenpflege.doc):

Die Schüler testen obiges Beispiel und lösen weitere Aufgaben:

- Aufgrund der Erhöhung der Mehrwertsteuer beschließt der Vorstand des Sportvereins, alle Beiträge um 3 % anzuheben. Schreibe in der Datenbank Sportverein2 eine passende Änderungsanweisung.
- Für die Zukunft ist geplant, dass die Datenverwaltung mehrerer Sportvereine durch die gleiche Datenbank übernommen wird. Dazu wird in der Tabelle „abteilung“ der Datenbank Sportverein2 zunächst eine weitere Spalte „Verein“ eingefügt. In alle Felder dieser Spalte soll dann „TSV06“ eingetragen werden.
- Die Kampfsportarten Boxen, Ringen und Judo werden in einen neuen Verein mit Namen „SC07“ ausgegliedert. Gleichzeitig sinken die Beiträge in diesen Abteilungen um 20 € Schreibe eine passende Änderungsanweisung.

Löschanweisung (DELETE)

Die Notwendigkeit der Löschung mehrerer Datensätze mit einem gemeinsamen Kriterium verdeutlicht folgende Aufgabe.

Beispiel 2:

Das Geschäft „Gutkauf“ muss schließen. Die zugehörigen Daten in der Datenbank Kaufdaten sollen gelöscht werden.

Auch hier wäre in einer großen Datenbank das Heraussuchen der zugehörigen Datensätze sehr aufwändig. SQL ermöglicht die Sammel Löschung:

```
DELETE FROM einkauf  
WHERE Geschäft = 'Gutkauf';
```

Anhand dieses Beispiels wird allgemein die Syntax der Löschanweisung vorgestellt:

```
DELETE FROM Tabelle  
WHERE ... ;
```

Bedingung zur Auswahl der
zu löschenden Datensätze

Falls die Bedingung Informationen aus weiteren Tabellen benötigt, so ist dies mit Hilfe geschachtelter Abfragen möglich, die als Vertiefung der Lehrplanvorgaben behandelt werden können. Eine verpflichtende Behandlung ist nicht vorgesehen.

Arbeit am Computer (Arbeitsblatt_Datenpflege.doc):

Die Schüler führen das Beispiel aus und lösen weitere Aufgaben:

- Welche Auswirkung hat die folgende Abfrage auf die Datenbank Sportverein?
DELETE FROM ist_Mitglied_von
WHERE M_ID = (SELECT M_ID FROM person WHERE Vorname = 'Karl'
AND Nachname = 'Seifert');
- Alle in Daberg wohnenden Mitglieder wollen den Sportverein verlassen. Ermittle in der Datenbank Sportverein2 zunächst die zugehörige(n) Ort_ID und schreibe dann eine passende Löschanweisung.

Einfügeanweisung (INSERT)

Auch wenn die Schüler das Einfügen einzelner Datensätze bereits mit der graphischen Oberfläche mehrfach durchgeführt haben, ist es sinnvoll, dass sie die Möglichkeit kennenlernen, neue Datensätze mit der Sprache SQL einzufügen. Einerseits wird diese Vorgehensweise später verwendet, wenn Datenbanken auf Programmiersprachenebene manipuliert werden sollen, andererseits bereitet das Einfügen einzelner Datensätze auf das nachfolgende Einfügen zahlreicher Datensätze aus Ergebnistabellen vor.

Beispiel 3:

In die vorhandene Tabelle

person (M_ID, Vorname, Nachname, Geburtsdatum, Geschlecht, Straße, Telefon, Ort_ID)

der Datenbank Sportverein2 soll Herr Norbert Nadel, Försterweg 6, 85368 Wang, geboren am 17.7.1981, aufgenommen werden. Seine Telefonnummer ist nicht bekannt.

Da die Syntax der INSERT-Anweisung den Schülern noch nicht bekannt ist, muss die Lösung vorgestellt werden. Zuvor wird die Ort_ID seines Wohnorts durch eine Abfrage ermittelt:

```
SELECT Ort_ID FROM ort WHERE PLZ = '85368' AND Ortsname = 'Wang';  
INSERT INTO person(Vorname, Nachname, Geburtsdatum, Geschlecht, Straße,  
Ort_ID)  
VALUES ('Norbert', 'Nadel', #07/17/1981#, 'm', 'Försterweg 6', 14);
```

Das Beispiel zeigt, dass nicht alle Spalten der Tabelle angegeben werden müssen; M_ID und Telefon fehlen. Die Reihenfolge der Spaltenbezeichner muss natürlich in der Reihenfolge der Daten eingehalten werden. Die Angabe der Spalten ist nicht unbedingt notwendig, wenn vollständige Datensätze eingefügt werden, sichert aber gegen spätere Änderungen der Spaltenreihenfolge.

Die Schüler sollen untersuchen, welche Einträge die Datenbank in den nicht aufgeführten Feldern des neuen Datensatzes vornimmt.

Die Syntax erlaubt auch das Einfügen mehrerer Datensätze in einer einzelnen Anweisung. In MSAccess2000 ist diese Option im Gegensatz zu MySQL nicht vorgesehen.

INSERT INTO Tabelle (Spalte1, Spalte2, ...)

VALUES (WertA1, WertA2, ...), ————— Datensatz A
(WertB1, WertB2, ...), ————— Datensatz B
... ;

Arbeit am Computer (Arbeitsblatt_Datenpflege.doc):

Die Schüler testen das obige Beispiel und lösen eine weitere Aufgabe:

Ermittle die M_ID des neuen Mitglieds „Norbert Nadel“ und lasse ihn mit Hilfe einer INSERT-Anweisung auf der Tabelle „ist_Mitglied_von“ in die Abteilungen „Basketball“ und „Badminton“ eintreten.

Typisches Beispiel für das Einfügen mehrerer Datensätze ist die Zusammenführung mehrerer Datentabellen mit ähnlicher inhaltlicher Struktur. Das folgende Einführungsbeispiel erfordert zwei neue Tabellen (Datenbank Kaufdaten2).

Beispiel 4:

In die vorhandene Tabelle

einkauf (Kunde, Geschlecht, Sparte, Warenbezeichnung, Preis, Zahlungsart, Geschäft, Kaufdatum)

der Datenbank Kaufdaten2 sollen Daten eingefügt werden, die aus anders strukturierten Quelltabellen stammen:

person (P_ID, Name, Geschlecht)

kauf (ID, Bezeichnung, Preis, Geschäft, Datum, Zahlungsart, Sparte, P_ID)

Beim Einfügen der neuen Daten muss die Information aus den Tabellen „person“ und „kauf“ erst zusammengeführt werden, um dann in die Tabelle „einkauf“ einzufließen:

```
INSERT INTO einkauf(Kunde, Geschlecht, Sparte, Warenbezeichnung, Preis,
Zahlungsart, Geschäft, Kaufdatum)
SELECT Name, Geschlecht, Sparte, Bezeichnung, Preis, Zahlungsart, Geschäft,
Datum
FROM person, kauf
WHERE person.P_ID = kauf.P_ID;
```

Das Beispiel zeigt, dass die Einfügeanweisung im Wesentlichen aus einer normalen Abfrage besteht. Ihr vorangestellt ist die „Tabelle“, in welche die Ergebnistabelle der Abfrage eingefügt werden soll.

INSERT INTO Tabelle (Spalte1, Spalte2, ...)

SELECT ...
FROM ...
WHERE ... ;

Abfrage, deren Ergebnistabelle die Tabellenstruktur (Spalte1, Spalte2, ...) hat.

Arbeit am Computer(Arbeitsblatt_Datenpflege.doc):

Die Schüler testen das vorstehende Beispiel und lösen weitere Aufgaben:

- In den Sportverein werden weitere Mitglieder aufgenommen, die bereits in einer Tabelle „personneu“ erfasst sind (Datenbank Sportverein3).

personneu (Nachname, Vorname, Geburtsdatum, Strasse, Wohnort, Telefon)

Füge eine Spalte Ort2 in die Tabelle „person“ ein, die nach dem Einfügen der neuen Mitglieder vorübergehend den jeweils in „personneu“ erfassten Ortsnamen enthält. Schreibe die zugehörige Einfügeanweisung.

Ergänze die Tabelle „ort“ um die noch fehlenden Wohnorte der neuen Mitglieder und trage zugehörige Postleitzahlen ein. Leider hat der Vorstand vergessen, diese zu erfragen.

Fülle die Spalte Ort_ID von „person“ aus.

Lösche zuletzt die Hilfsspalte Ort2.

- Die Fußballspieler sollen ab sofort in einer eigenen Tabelle erfasst werden. Erzeuge in der Datenbank Sportverein2 zunächst die leere Tabelle „fußballer“ mit der gleichen Struktur wie „person“ und schreibe dann eine passende Einfügeanweisung.

Zusammenfassender Hefteintrag:**Datenpflege**

SQL ist nicht nur eine Datenbankabfragesprache, sondern auch eine Datenbankmanipulationssprache. Damit können mehrere Datensätze gleichzeitig geändert (UPDATE), gelöscht (DELETE) oder eingefügt (INSERT) werden.

Die folgende Zusatzaufgabe eignet sich für Fortgeschrittene:

Die Mitglieder der Judo-Abteilung haben den Vorstand „aufs Kreuz“ gelegt. Deshalb wird diese Abteilung aufgelöst und alle Mitglieder dieser Abteilung werden aus dem Verein ausgeschlossen.

Datenschutz und Datensicherheit (27. Stunde)

Auf das Thema Datenschutz wird man wohl immer dann eingehen, wenn es in den genannten Beispielen um personenbezogene Daten geht. Bei der Arbeit mit den Einkaufsdaten können die Schüler gegebenenfalls mit den Begriffen „Kundenkarte“ und „Datenschutz“ in einer Suchmaschine recherchieren. Einen Ausgangspunkt zur umfassenden Information über das Thema Datenschutz bietet die Seite des „Berliner Beauftragten für Datenschutz und Informationsfreiheit“ (<http://www.datenschutz-berlin.de/>) oder auch die Seite des bayerischen Landesbeauftragten (<http://www.datenschutz-bayern.de/>), einfachere Begriffsklärungen bietet Wikipedia ([http://de.wikipedia.org/wiki/Personenbezogene Daten](http://de.wikipedia.org/wiki/Personenbezogene_Daten)). Aktuelle Themen sind auch die Erfassung biometrischer Daten (<http://www.bromba.com/faq/bprifaqd.htm>) und der unauffällige Einsatz von RFIDs ([http://de.wikipedia.org/wiki/RFID#Bedenken und Kritik](http://de.wikipedia.org/wiki/RFID#Bedenken_und_Kritik)).

In einem Client-Server-System wie MySQL kann den Schülern die Einschränkung der Schreib-, Lese- oder Änderungsrechte sowohl auf Daten- als auch auf Tabellenentwurfsebene demonstriert werden. In Einzelplatzsystemen ist dies weniger gut möglich, da der Anwender voreingestellt Administratorrechte auf der Datenbank hat.

Hefteintrag:**Datenschutz, Datensicherheit**

Ein professionelles Datenbankprogramm muss in der Lage sein, verschiedenen Benutzern unterschiedliche Zugriffsrechte einzuräumen. Ein bestimmter Benutzer darf bestimmte Daten

- nicht einsehen,
- nur abfragen, aber nicht ändern, oder
- uneingeschränkt bearbeiten.

In der Regel werden die Benutzer in verschiedene Gruppen eingeordnet, denen passwortgeschützt unterschiedliche Rechte erteilt werden.

Da ein Datenverlust erheblichen finanziellen Schaden verursacht, muss die Datenbank gegen Programmfehler und Hardwareausfälle gesichert sein.

In der Sportvereindatenbank wären beispielsweise folgende Rechte sinnvoll:

- Der **Administrator** hat uneingeschränkte Rechte: Er darf neue Tabellen anlegen, Tabellen und Beziehungen ändern, Daten eingeben und ändern sowie Rechte vergeben.
- Der **Vorstand** darf keine Tabellen anlegen oder löschen; er darf auch die Attribute der Tabellen und die Beziehungen nicht ändern. Der Vorstand darf jedoch Mitglieder

löschen bzw. einfügen oder deren Daten ändern; er darf auch Abteilungen löschen bzw. einfügen oder deren Daten ändern und er darf die Beziehungstabelle bearbeiten.

- Ein **Abteilungsleiter** darf nur auf die Daten der Mitglieder seiner Abteilung lesend und schreibend zugreifen.
- Die **Mitglieder** dürfen alle Daten einsehen (eventuell nicht alle Personendaten), aber nichts ändern.
- **Andere** Personen dürfen nur die Daten der Abteilungen einsehen.

Aufgabe:

Zusammen mit deinen Freunden hast du eine Musikdatenbank (ähnlich wie im Unterricht) mit Daten aus euren CDs gefüllt.

Überlege dir, welche Rechte jeder deiner Mitschüler erhalten soll.

Arbeit am Computer:

Falls das Datenbankprogramm dies ermöglicht, sollte jeder Schüler für seine Musikdatenbank aus den vorherigen Stunden geeignete Rechte vergeben und dann die Auswirkungen von Mitschülern testen lassen.

Datenschutz und Views

Auch Views lassen sich sehr gut zur Absicherung des Datenschutzes einsetzen. Die Vergabe von Zugriffsrechten auf Tabellen bzw. Tabellenspalten erlaubt eine „senkrechte“ Einschränkung des Zugriffs. Dem Benutzer sind nur einige Tabellen bzw. Tabellenspalten zugänglich. Dort kann er aber je nach den eingeräumten Rechten alle Werte sehen bzw. bearbeiten. Mit Hilfe von Views sind auch „waagrechte“ Einschränkungen möglich, d. h., der Benutzer kann nur bestimmte Datensätze oder Teile davon sehen. Im oben angeführten Beispiel der Sportvereindatenbank wurde folgende View eingerichtet:

```
CREATE VIEW Fussballer AS
SELECT Vorname, Nachname, Geburtsdatum, Geschlecht, Strasse, Telefon, Ort_ID
FROM person, abteilung, ist_Mitglied_von
WHERE person.M_ID = ist_Mitglied_von.M_ID
AND ist_Mitglied_von.Sport_ID = abteilung.Sport_ID
AND Sportart = 'Fußball';
```

Der Abteilungsleiter der Abteilung Fußball bekommt Lese- und Änderungsrechte für diese View. Dann kann er die Daten aller Fußballer einsehen und gegebenenfalls ändern, die Daten der anderen Vereinsmitglieder sind für ihn jedoch nicht zugänglich.

Komplexeres Anwendungsbeispiel (28.–38. Stunde)

Am Ende der Unterrichtssequenz sollen die Schüler das Erlernte auf ein umfangreicheres Beispiel anwenden. Innerhalb der Klasse werden dabei von verschiedenen Gruppen unterschiedliche Themen bearbeitet. Alternativ kann auch das gleiche Thema mehrfach vergeben werden. Wichtig ist, dass alle Projekte nach Fertigstellung vor der Klasse präsentiert und diskutiert werden.

Die Bearbeitung der von den Schülern gewählten Themen sollte Konzeption sowie Umsetzung einer Datenbank sowie geeignete Abfragen umfassen. Bei der Modellierung muss bereits darauf geachtet werden, dass Redundanzen vermieden werden. Je nach verfügbarem Zeitumfang können auch einfache Berichte und Formulare erstellt werden, so dass innerhalb jeder Gruppe nach der Leistungsfähigkeit der einzelnen Schüler differenziert werden kann.

Neben den im Lehrplan vorgeschlagenen Themen eignen sich z. B.:

- Verwaltung von Flugbuchungen
- Verwaltung der Schuldaten (siehe unten)
- Verkehrssünderkartei
- Verwaltung eines Fahrradverleihs
- Organisation geographischer Daten
- Datenverwaltung eines Kaufhauses, Bekleidungsgeschäfts, ...
- Datenverwaltung einer Versicherung
- Datenverwaltung eines Krankenhauses
- Datenverwaltung eines Zoos
- Filmarchiv

Die Themen sollten sich nach den Interessen der Schüler richten. Aus datenschutzrechtlichen Gründen wird darauf hingewiesen, dass bei allen Beispielen mit personenbezogenen Daten keinesfalls mit Echtdaten gearbeitet werden darf.

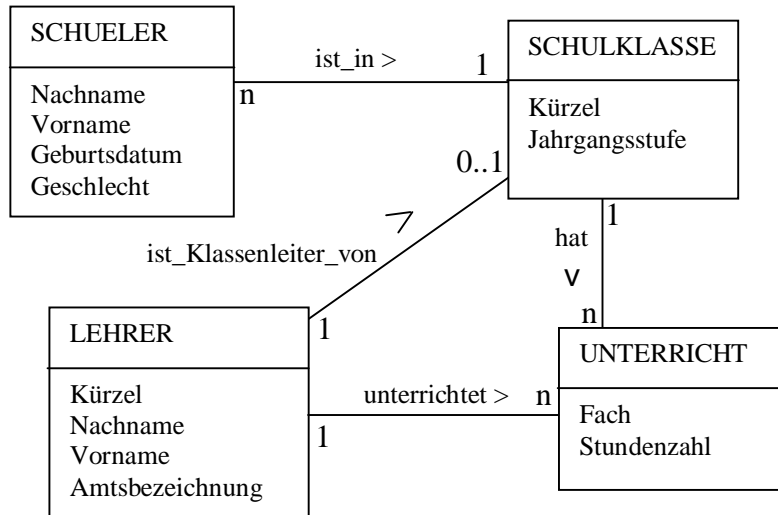
Für jedes Thema bietet sich folgende Vorgehensweise an:

- a) Erstellung eines Klassendiagramms mit den jeweiligen Attributen und den Beziehungen zwischen den Klassen.
- b) Erstellung der jeweiligen Tabellen in einer Datenbank, Realisierung der Beziehungen. Dabei ist zu überlegen, ob referentielle Integrität, Löschweitergabe bzw. Aktualisierungsweitergabe sinnvoll sind. Es müssen genügend viele Daten in die Tabelle eingegeben werden, um sinnvolle Ergebnistabellen bei Abfragen zu erhalten.
- c) Überlegung, welche Benutzergruppen welche Rechte auf der Datenbank haben sollten.
- d) Erstellung von Abfragen für eine Auswahl sinnvoller Fragestellungen an die Datenbank.

Es hat sich bewährt, für den Arbeitsfortschritt einen festen zeitlichen Rahmen zu vereinbaren (Meilensteine). Spätestens nach zwei Stunden sollte das Klassendiagramm vorgelegt werden, das von der Lehrkraft (eventuell mit der ganzen Klasse) hinsichtlich Korrektheit und Umfang überprüft wird.

Die folgende Skizze für ein Projekt **Verwaltung der Schuldaten** verdeutlicht den Umfang eines solchen Themas.

Klassendiagramm:



Aus dem Klassendiagramm¹ ergeben sich folgende Tabellen:

schueler : Tabelle		
	Feldname	Felddatentyp
🔑	S_ID	AutoWert
	Nachname	Text
	Vorname	Text
	Geburtsdatum	Datum/Uhrzeit
	Geschlecht	Text
	K_Kürzel	Text

schulklasse : Tabelle		
	Feldname	Felddatentyp
🔑	Kürzel	Text
	Jahrgangsstufe	Zahl
	Klassenleiter	Text

lehrer : Tabelle		
	Feldname	Felddatentyp
🔑	Kürzel	Text
	Nachname	Text
	Vorname	Text
	Amtsbezeichnung	Text

unterricht : Tabelle		
	Feldname	Felddatentyp
🔑	L_Kürzel	Text
🔑	K_Kürzel	Text
🔑	Fach	Text
	Stundenzahl	Zahl

Auf der Datenbank sollen Schüler die Namen ihrer Mitschüler lesen dürfen. Lehrer haben auf alle Tabellen lesenden Zugriff. Das Sekretariat hat außerdem auf die Tabellen „schueler“ und „lehrer“ schreibenden Zugriff, das Direktorat hat zusätzlich auf der Tabelle „unterricht“ Schreibrechte. Nur der Datenbankadministrator darf auch die Tabellenstruktur ändern.

Beispiele für geeignete Abfragen:

1. Welche Lehrkräfte (Name, Vorname) unterrichten Mathematik?
2. Welche Klassen haben Informatik?
3. Liste alle Schüler (Name, Vorname) auf, die Informatik haben.

¹ Die Klasse UNTERRICHT übernimmt die n:m-Kopplung zwischen LEHRER und SCHULKLASSE. Da diese Beziehung die Attribute „Fach“ und „Stundenzahl“ hat, wird eine eigene Klasse formuliert. Die Tabelle „unterricht“ hat als Fremdschlüssel die Schlüssel von „lehrer“ und „schulklasse“; diese bilden zusammen mit „Fach“ den Schlüssel von „unterricht“.

4. Welche Fächer werden in der Klasse 6 a unterrichtet?
5. Welche Lehrer (Name, Vorname) sind Klassenleiter in der Jahrgangsstufe 6?
6. Erstelle eine Liste aller Schüler (Name, Vorname), die vor dem 1.1.1993 geboren wurden.
7. Liste alle Fächer auf, die der Lehrer XY unterrichtet.
8. Welche Lehrer unterrichten Schüler, die vor dem 1.1.1993 geboren wurden?
9. Erstelle eine Liste aller Fächer, in denen der Schüler XY unterrichtet wird.
10. Welche Lehrer unterrichten Mathematik oder Informatik?
11. Welche Lehrer unterrichten Mathematik und Informatik?
12. Welche Fächer werden zweistündig unterrichtet?
13. Welche Fächer werden in der Jahrgangsstufe 7 unterrichtet?

Bei der Vorstellung der Ergebnisse können neben dem Datenbankprogramm durchaus auch Präsentationsprogramme zum Einsatz kommen.

2.2.3 Materialien

Die genannten Arbeitsblätter, Präsentationen und Datenbanken sind auf der Begleit-CD oder über die Homepage des ISB (www.isb.bayern.de à Gymnasium à Fach Informatik à Materialien) verfügbar. Sie sind auf das obige Unterrichtskonzept zugeschnitten, lassen sich jedoch unschwer an die eigene Unterrichtssituation anpassen. Auf der Begleit-CD finden sich auch die Lösungen zu den Arbeitsblättern und Aufgaben.