M104 – Datenmodell implementieren

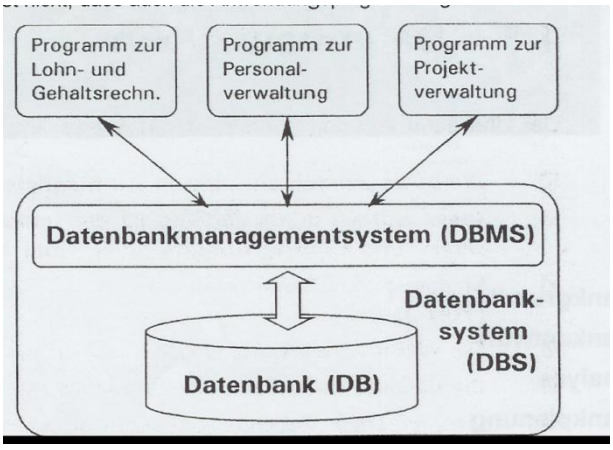
1 Datenbankgrundlagen

1.1 Datenbankgrundlagen

Datenbanken spielen beim Einsatz von Computern häufig eine zentrale Rolle. In vielen Fällen, wo Arbeitsabläufe computerunterstütz abgewickelt werden, ist die Speicherung grosser Datenmengen erforderlich. Typische Merkmale dafür wären:

- Redundanzen
- Inkonsistenzen
- Datenschutzprobleme
- Fehlende Datenunabhängigkeit

Durch die Entwicklung von Datenbanksystemen (DBS) wurden diese Probleme nach und nach gelöst. In einem DBS werden die Daten in einer Datenbank zusammengefasst, die ausschliesslich von dem Datenbankmanagementsystem (DBMS → Datenbanksoftware) verwaltet werden. Anwendungsprogramme greifen nun nicht mehr direkt auf die Daten zu, sondern stellen ihre Anforderungen nur noch an das DBMS. Die enge Verflechtung und Abhängigkeit von Daten und den damit arbeitenden Anwendungsprogrammen wird stark reduziert bzw. ganz aufgehoben.
Ein DBS besteht aus einer Anzahl von Datenbanken und dem DBMS.



Die Datenbank weist somit folgende Eigenschaften auf:

- In Datenbanken sind Daten entsprechend ihrer natürlichen Zusammenhänge gespeichert. Dabei ist es nicht entscheidend, in welcher Form die Daten in Anwendung benötigt werden. Die Daten der Datenbank bilden einen Ausschnitt aus der realen Welt ab.
- Auf die Daten einer Datenbank können viele Benutzer gleichzeitig zugreifen. Das DBMS verwaltet sowohl die Daten als auch die Zugriffe darauf und sorgt dafür, dass dieselben Daten nicht von mehreren Benutzern gleichzeitig bearbeitet werden können.

Das DBMS ist die Software, welche die Datenbanken verwaltet. Es ermöglicht dabei:

1. das Anlegen von Datenbanken
2. die Speicherung, Änderung und Löschung der Daten
3. das Abfragen der Datenbank
4. eine Verwaltung von Benutzern, Zugriffen und Zugriffsrechten

Verschiedene Datenbanktypen: (→ Glossar)

- Hierarchische Datenbanken
- Netzwerkdatenbanken
- Relationale Datenbanken
- Objektorientierte Datenbanken

Aufbau und Organisation einer Datenbank:

- 3-Ebenen-Modell mit externer, konzeptioneller und interner Ebene
- Grundlagen des DBMS
- Physische Datenbankarchitektur mit zentralisiertem, verteiltem, parallelem und Client-Server Datenbanksystem

1.2 Datenbankentwurf

1. **Daten analysieren (Analyse der Anforderungen)**
   Personengruppen, Gegenstände und ihre Eigenschaften sowie ihre Beziehungen untereinander definieren.
2. **DB-Plan zeichnen** (Konzeptioneller und logischer Entwurf)
   Dabei wird z.B das Entity-Relationship-Diagramm (ERD) erstellt.
3. **DB-Struktur nach Fehlern untersuchen**
   Fehler können dazu führen, dass Daten mehrfach gespeichert werden
4. **DB erstellen**
5. **Daten eingeben**
6. **DB-Lösung überprüfen**
   Kontrolle, ob gültige Daten eingegeben werden können und ob ungültige Daten vom System abgelehnt werden.
7. **Lösung dokumentieren**

Die Entwicklung einer Datenbank umfasst einen langen Weg. Für den konzeptionellen Entwurf der DB muss genügend Zeit investiert werden, da damit die Qualität der Datenbank beeinflusst wird.

1.3 Datenanalyse

Vorteile elektronischer DB's

- Flexibilität ist vorhanden
- Schnelligkeit ist gewährleistet
- Statistiken lassen sich schnell und mit wenig Aufwand erstellen
- Die Erfassung von Daten kann über Bildschirmmasken erfolgen
- Das System kontrolliert, dass keine falschen Daten eingetippt werden.

1.4 Datenbankplanung

Das Entity-Relationship-Modell (ERM) bzw. seine Visualisierung Entity-Relationship-Diagramm (ERD) ist ein weit verbreitetes Datenmodell

Entität: In einer Datenbank werden sog. Objekte und deren Eigenschaften gespeichert. Wenn wir über ein einzelnes Objekt reden, sprechen wir von einer Entität. (Als Entität (engl. Entity) wird in diesem Zusammenhang ein abgrenzbares Objekt der realen Welt oder der Vorstellungswelt bezeichnet, über das wir Informationen sammeln und auswerten können.)
Bsp: Hans Peter, 1.1.1900

Entitätsmenge Die Menge aller Objekte, die vom Typ her zusammengehören, wird Entitätsmenge genannt.
Bsp: Hans Peter, 1.1.1900
… (alle Karten der Kundendatei)

Entitätstyp Die Art der Objekte, die gespeichert bzw bearbeitet werden, nennt man Entitätstyp. Im Beispiel stellen Kunden, Filme und Ausleihen solche Entitätstypen dar.

Attribut und Attributswert: Attribut = Eigenschaft
Bsp: in Kunden „Anrede“, „Vornamen“

Wertebereich: Das ist die Menge der zulässigen Eigenschaftswerten für eine bestimmte Eigenschaft: Diese Mengen nennt man Domäne.
Bsp: für die Eigenschaft Anrede „Herr und Frau“

Attribut ohne Attributswert: **NULL-Wert.** Damit ist die Abwesenheit eines Eigenschaftswertes gemeint:
Achtung: nicht Texteintrag mit Leertaste oder 0

Primärschlüssel

Ein Primärschlüssel ist eine Eigenschaft (Kombination von Eigenschaften), deren Inhalt eine Entität eindeutig identifiziert.

Eigenschaften:
Eindeutigkeit: Jeder Kunde erhält eine eindeutige Nummer. Eine Nummer wird nur 1-mal vergeben.

Laufende Zuteilbarkeit: Beim Erfassen eines Kunden muss seine Nummer schnell und unkompliziert bestimmt werden können => fortlaufende Nummer

Kürze und Prägnanz: Wenn der PK lang und kompliziert ist, besteht die Gefahr von Tippfehlern.

Dauerhaftigkeit: Der Wert eines Primärschlüssels darf sich nicht ändern.

Fremdschlüssel
Ist ein Schlüssel, der die Verbindung mit einer anderen, d.h. fremden Tabelle ermöglicht.

Assoziation
Ist eine Zuordnung, welche festlegt, wie viele Entitäten einer Tabelle zu einer Entität einer anderen Tabelle gehören können.

1.5 Redundanzen

Begriff (ERD)	Synonym DE	Synonym EN
Entitätsmenge	Tabelle, Relation	Table, Relation
Entität	Zeile, Tupel,Datensatz	Row
Attribut, Eigenschaft	Spalte, Datenfeld	Column, Field
Beziehung	Beziehung	Relationship

Redundanz = mehrfach vorhandene Informationen

Konsequenz davon ist das Austauschen von Informationen, welche sich widersprechen können. Diese widersprüchlichen Informationen heissen Anomalien (Inkonsistenzen).
- Probleme beim Ändern von Daten (Mutations- oder Änderungsanomalie)
- Probleme beim Löschen von Daten (Lösch-Anomalie)
- Probleme beim Einfügen von Daten (Einfüge-Anomalie)

Normalisierungsprozess
Beim Normalisierungsprozess werden Tabellen systematisch nach vorgegebenen Regeln analysiert und nach Bedarf umstrukturiert.

Das Resultat dieser Normalisierungsprozesse ist eine Datenbank mit Tabellen frei von Redundanzen, die ansonsten die Konsistenz der Datenbank gefährden würden.
Bei er Normalisierung geht es darum zu kontrollieren, dass sich alle Fehler in den richtigen Tabellen befinden. Wenn dies der Fall ist, können Daten nicht mehr doppelt gespeichert werden.
Der Normalisierungsprozess erfolgt in mehreren Schritten. Bei jedem Normalisierungsschritt wird die Form (Struktur) der Tabelle verändert. Die Tabelle befindet sich jeweils in einer der folgenden Formen:

- Erste Normalform (1NF)
- Zweite Normalform (2NF)
- Dritte Normalform (3NF)
- Boyce-Codde-Normalfomr (BCNF)
- Vierte Normalform (4NF)
- Fünfte Normalform (5NF)

2 Einführung SQL

2.1 Einführung in Datenbankabfragesprache SQL

2.1.1 Was ist SQL?

SQL ist eine Spezialsprache, die für den Entwurf und die Verwaltung von relationalen Datenbanken sowie die Manipulation der darin enthaltenen Daten entwickelt wurde. SQL wird durch einen ANSI-Standard definiert. Der Sprachumfang von SQL ist einer permanenten Weiterentwicklung und Standardisierung unterworfen. Derzeit relevant sind SQL-92, SQL-1999 sowie SQL-2003.
Alle Anbieter von relationalen DBMS haben ihre eigene Implementation von SQL, die sich mehr oder weniger vom Standard SQL-92 unterscheiden.
SQL-Server nennt sein erweitertes SQL „Transact SQL“, bei Oracle spricht man von „PL/SQL“.
Wenn man die Datenbank auf unterschiedlichen Datenbanksystemen verwenden will, sollte man auf die Verwendung von erweiterten Komponenten „z. B. Transact-SQL) verzichten. Darum ist eine enge Anlehnung an den ANSI SQL Standard besonders wichtig.
Obwohl SQL keine Allzweckprogrammiersprache ist, enthält sie alles, was man benötigt, um relationale Datenbanken zu erstellen, zu verwalten, zu sichern und zu schützen

SQL (Structured Query Language) besteht aus folgenden Hauptbestandteilen:

- **DDL (Data Definition Language)**
  DDL stellt alles zur Verfügung, was benötigt wird, um eine Datenbank und deren Elemente wie Relationen und Beziehungen zu definieren, zu ändern und zu löschen.
- **DML (Data Manipulation Language)**
  DML verfügt über Befehle, mit welchen man Daten (Informationen) in eine Relation einfügen und löschen kann.
- **DQL (Data Query Language)**
  DQL verfügt über Befehle, mit welchen man Informationen nach den verschiedensten Kriterien aus einer Datenbank abrufen (betrachten) kann.
- **DCL (Data Control Language)**
  DCL verfügt über Befehle, mit denen man die Datenbank vor unerwünschten Einflüssen schützen kann (Berechtigungen)

2.1.2 Entwicklungsgeschichte von SQL

- **ca. 1975**
  SEQUEL = Structured English Query Language, Vorläufer von SQL wird für das Projekt System R von IBM entwickelt.
- **1981**
  SQL gelangt mit SQL/Data Systems erstmals durch IBM auf den Markt.
- **1986**
  SQL1 wird von ANSI als Standard verabschiedet.
- **1987**
  SQL1 wird jetzt auch von ISO als Standard verabschiedet und 1989 nochmals überarbeitet.
- **1992**
  Der Standard SQL2 bzw. SQL-92 wird von der ISO verabschiedet.
- **1999**
  SQL3 bzw. SQL-1999 wird verabschiedet.
- **2003**
  SQL-2003 wird von der ISO als Nachfolger des SQL-1999 Standards verabschiedet.

2.3 Entwicklungsschritte der Datenbankerstellung

2.3.1 Datenbank erstellen

```
CREATE DATABASE DbName;
```

2.3.2 Datenbank aktivieren

```
USE DbName;
```

2.3.3 Tabellen erstellen

CREATE Table TKunde(
KNr INTEGER NOT NULL AUTO_INCREMENT,
Nachname VARCHAR(30),
u_KNr(KNr),
PRIMARY KEY(KNr)
);

Durch AUTO_INCREMENT wird veranlasst, dass die KNr bei jedem neuen Kunden automatisch um eins erhöht wird. Begonnen wird mit der Nummer 1. Spalten mit dem Attribut AUTO_INCREMENT müssen indexiert werden. Deshalb wird weiter die Zeile UNIQUE INDEX u_KNr(KNr) eingefügt.

2.3.4 Datentyp festlegen

MySQL unterstützt dabei folgende Datentypen:

	Datentyp	Erläuterung
Numerisch	DECIMAL oder NUMERIC	Speichert Zahlen von -1.7976931348623157E+308 bis -2.2250738585072014E-308 und 2.2250738585072014E-308 bis 1.7976931348623157E+308.
	INTEGER oder INT	Speichert ganze Zahlen von -2'147'483'648 bis 2'147'483'647. Es sind keine Buchstaben erlaubt.
Datum und Zeit	DATE	Speichert Daten von 1000-01-01 bis 9999-12-31. Bei MySQL wird das Datum immer im Format Jahr-Monat-Tag gespeichert und bearbeitet. Jeder Monat hat 31 Tage.
	DATETIME	Speichert Daten und Zeilen von 1000-01-01 00:00:00 bis 9999-12-31 23:59:59.
	TIME	Speichert Zeiten von -838:59:59 bis 838:50:59.
	YEAR	Speichert eine Zahl zwischen 1901 bis 2155.
Text	CHAR	Speichert alphanumerische Zeichen. Die maximale Anzahl Zeichen beträgt 255 Zeichen. Die zur Zeit der Tabellenerstellung definiert Länge ist fix.
	ENUM ('Herr','Frau')	Speichert alphanumerische Zeichen. Der Benutzer kann nur Begriffe eingeben, die im ENUM aufgelistet sind.
	TEXT	Speichert bis zu 65535 alphanumerische Zeichen.
	VARCHAR	Speichert alphanumerische Zeichen. Die maximale Anzahl Zeichen beträgt 255 Zeichen. Die Länge ist variabel und ist vom Inhalt abhängig.

2.3.6 Tabellen einer Datenbank auflisten

SHOW TABLES;

2.3.7 Tabellenstruktur anzeigen

SHOW COLUMNS FROM Tabellenname;
oder
SHOW FIELDS FROM Tabellenname;

2.3.8 Tabellenstruktur bearbeiten

Spalten hinzufügen
ALTER TABLE Kunden ADD COLUMN Titel VARCHAR(10);

Spalten löschen
ALTER TABLE Kunden DROP COLUMN Titel;

2.3.9 Daten erfassen

INSERT INTO Kunde(KNr, Anrede, Nachname)
VALUES (NULL,'Herr','Kebap');

2.3.10 Tabellen löschen

DROP TABLE Kunde;

2.3.11 Datenbank löschen

DROP DATABASE DbName;

2.4 Daten selektieren und auswerten

SELECT * FROM Kunde [WHERE Name = ‚Me']

2.5 Daten und Verknüpfungen testen

2.5.1 Bildschirmmaske

Gestaltung der Bildschirmmasken:

- **Elemente gruppieren**
  Bei der Platzierung darauf achten, dass Felder, die inhaltlich zusammengehören. auch zusammen erscheinen
- **Abstände und Ausrichtung**
  Abstände tragen zur besserer Lesbarkeit bei
- **Schrift**
  Serifenlose Schriften sind auf dem Bildschirm besser lesbar.
  GROSSBUCHSTABEN und *kursiv* eignet sich auch nicht für das menschliche Auge
- **Farben**
  zu viele Farben wirken für unser Auge ermüdend und für unser Gehirn verwirrend.
  Rot = Gefahr, Stopp, Verbot, Fehler
  Orange = Vorsicht, Achtung
  Grün = Keine Gefahr, Fluchtweg Weg frei, Sicherheit
  Menschen assoziieren Farben mit Gefühlen und Ereignissen
  ➔ grüne Fehlermeldung würde nicht bemerkt werden!!!
- **Weg des Tabulators**
  sollte nachvollziehbar sein (nicht suchen)
- **Eingabehilfen**
  Checkboxen, Optionsfelder, Kombinationsfelder

2.5.2 Datenintegrität

Datensicherheit umfasst wirksame Massnahmen gegen den Datenverlust.
Datenschutz beinhaltet Massnahmen gegen den Missbrauch von Daten.
Datenkonsistenz setzt voraus, dass sich die Daten nicht widersprechen.

Datenkonsistenz kann vermieden werden durch:

- Normalisierung des Datenbestandes
- Wahl des richtigen Datentyps bei der Definition der Tabellen
- Einbau von Kontrollmechanismen in der Bildschirmmaske
- Einsatz von grafischen Elementen in der Bildschirmmaske, die nur bestimmte Eingaben erlauben.

2.5.3 Test

Bei reinen Textfeldern ist zu überprüfen, ob das Feld für den längsten möglichen Eintrag lang genug ist.

Bei Kombinationsfeldern ist sicherzustellen, dass nur aufgelistete Werte eingegeben werden dürfen,

Das Testprotokoll dient als Basis für allfällige Korrekturen und als Beweismittel.

2.5.4 Datenbank dokumentieren

Zur Dokumentation gehört:

- Das Entity-Relationship-Diagramm (ERD)
- Die Beschreibung jeder Tabelle (mit dem Befehl SHOW FIELDS FROM…)
- Abbildung aller Bildschirmmasken
- Abbildung der Berichte
- Testprotokoll

2.6 Anwendung und Übungen

2.6.1 ODBC

ODBC (Open DataBase Connectivity – „Offene Datenbank-Verbindungsfähigkeit“) ist eine standardisierte Datenbankschnittstelle, die SQL als Datenbanksprache verwendet. ODBC bietet also eine Programmierschnittstelle (API), die es einem Programmierer erlaubt, seine Anwendung relativ unabhängig vom verwendeten Datenbankmanagementsystem (DBMS) zu entwickeln, wenn dafür ein ODBC-Treiber existiert.

ODBC wurde ursprünglich von Microsoft auf Basis des Call Level Interface von X/Open und ISO/IEC entwickelt, ist aber inzwischen auch von anderen Softwareherstellern übernommen worden. In vielen Bereichen ist ODBC mittlerweile als Standard etabliert.

ODBC-Treiber haben einen unterschiedlichen Funktionsumfang:

- Core (nur Basisfunktionalität)
- Level 1
- Level 2

Moderne Programmierumgebungen erlauben dadurch den unkomplizierten Zugriff auf sehr viele unterschiedliche Datenbank-Managementsysteme (über vorgefertigte datensensitive Steuerelemente). Der Datenzugriff erfolgt nie unmittelbar auf eine Tabelle oder eine Datenbank, sondern immer über die entsprechende (ODBC-)Komponente. Mit ODBC kann auf jede lokale oder ferne Datenquelle zugegriffen werden.

Für objektorientierte Programmiersprachen (z. B. C++, Java) sind Klassen verfügbar, die Methoden für den Umgang mit dem Datenmaterial der unterschiedlichen Datenbank-Systeme definieren. Der Programmierer braucht sich um datenbankspezifische Details nicht mehr zu kümmern.

Hinweis: Auch das für das Microsoft-Jet-Datenbankmodul optimierte DAO (Data Access Objects) ermöglicht den mittelbaren Zugriff auf ODBC.

Ab Windows 2000 ist ODBC als Bestandteil von MDAC integraler Bestandteil des Betriebssystems. Für frühere Windowsversionen kann es kostenfrei nachinstalliert werden.

3 SQL – DDL

3.1 Datenbanken erstellen, betrachten und löschen

DDL ist der Teil der Sprache SQL, welcher für die Erstellung und Zerstörung von Objekten zuständig ist.

Datenbanken erstellen

```
CREATE DATABASE DbName;
USE DbName;
```

Datenbanken löschen

```
DROP DATABASE DbName;
```

3.2 Relationen erstellen, bearbeiten und löschen

3.2.1 SQL-Datentypen

Numerische Datentypen (Ganzzahlen)
TINYINT < SMALLINT < INTEGER

Numerische Datentypen (Fliesskommazahlen)
FLOAT < DOUBLE PRECISION

Numerische Datentypen (Festkommazahlen)
NUMERIC / DECIMAL (Präzision, Skalierung)

Datumswerte

- DATE (MySQL)
- DATETIME
- SMALLDATETIME (SQLServer)

Zeichen
CHAR(Länge), VARCHAR(Länge), TEXT, BLOB

3.2.2 CREATE TABLE Anweisung

Die Anweisung **CREATE TABLE** dient der Erstellung von Relationen (Tabellen) in einer Datenbank.

Nach SQL-92 können folgende Konsistenzbedingungen für eine Tabelle festgelegt werden:

- PRIMARY KEY
- UNIQUE (Kandidatenschlüssel)
- FOREIGN KEY
- CHECK (Einschränkungen des Wertebereichs)
- NULL/NOT NULL (Verbot von Nullmarken in Spalten)
- CHECK (Spaltenübergreifende Integritätsbedingungen)
- ASSERTION (Tabellenübergreifende Integritätsbedingungen)

Bestehende Tabellen ändern
ALTER TABLE Tabellenname ADD Test VARCHAR(9) DEFAULT('Unbekannt')

Foreign Key (Fremdschlüssel)
- Die Veränderung kann ganz verboten werden → NO ACTION
- Die Veränderung kann an den Fremdschlüssel weitergegeben werden → CASCADE
- Die Veränderung kann den Fremdschlüsselwert auf NULL setzen → SET NULL
- Die Veränderung kann den Fremdschlüsselwert auf einen Defaultwert setzen → SET DEFAULT

Referenzielle Datenintegrität
Unter Datenintegrität versteht man die **Fehlerfreiheit, Genauigkeit und Zuverlässigkeit,** d.h. die **Qualität** von Daten. Insbesondere erkennt man die Genauigkeit bei Änderung von Daten.
Die referenzielle Datenintegrität definiert nun die Beziehungen zwischen Datenzeilen über mehrere Tabellen. In SQL-Server basiert diese Durchsetzung der referenziellen Integrität auf Beziehungen zischen PK und FK bzw zwischen FK und KK.
→ FOREIGN KEY-Einschränkungen

3.3 Normalisierung

3.3.1 Erste und zweite Normalform

Erste Normalform. Die Universal Relation wird in der sogenannten ersten Normalform vorausgesetzt. Dazu wird gefordert, dass alle Attributwerte atomaren Charakter haben, also aus Sicht des Datenbank keine interne Struktur besitzen: wie z.B. ein Nachname, eine Kursnummer oder ein Kursgebühr. Tabellenzeilen, die sogenannte Mehrwertattribute enthalten, müssen so vervielfacht werden, dass jede Zeile in jedem Attribut nur einen einzigen Attributwert speichert (siehe Abb. 4.3). Relationen in der ersten Normalform sind also "flache Tabellen" ohne jede "Schachtelung".
Zweite Normalform. Die zweite Normalform bricht nun die Universal Relation in sinnvolle Teiltabellen auf. Um diesen Prozess formulieren zu können, benötigen wir ein weiteres zentrales Konzept der relationalen Datenbank-Theorie: Die Schlüsseleigenschaft von Attributen für eine Relation.

3.3.2 Dritte Normalform

Bei genauer Analyse fällt auf, dass die zweite Normalform noch nicht gegen Anomalien gefeit ist. Der Grund liegt darin, dass manche Attribute nur indirekt (transitiv) vom Schlüssel abhängig sind. Diese Fälle soll die dritte Normalform beseitigen.

3.3.3 Weitere Normalformen

Die vierte und fünfte Normalform sind wohl deswegen nicht so populär, weil die Situationen unwahrscheinlicher sind, in denen eine Verletzung der Normalformenbedingungen zu erwarten ist. Dies vor allen Dingen deshalb, weil man bereits intuitiv solche Verletzungen vermeidet. Insbesondere wenn man den Weg wählt, zunächst ein ER-Modell, das dann in einen relationalen Entwurf umgesetzt wird, reicht die Prüfung auf dritte Normalform. Verletzungen der zweiten oder dritten Normalform sind zwar nicht zu erwarten, aber immerhin denkbar und sollten durch Nachprüfen ausgeschlossen werden. In allen praktisch relevanten Fällen kann davon ausgegangen werden, dass dann auch die Bedingungen der vierten und fünften Normalform nicht verletzt sind.

3.3.4 Integritätsbedingungen

Die Forderung nach Integrität einer Datenbank zielt darauf, daß das gespeicherte Datenabbild tatsächlich auch der Wirklichkeit entsprechen muß. Fehlerhafte Eingaben, unterlassene Pflege (Updates) stellen schnell den Nutzen einer Datenbank in Frage. Datenintegrität läßt sich beim Datenbankentwurf nicht erzwingen, jedoch in speziellen wichtigen Fällen unterstützen.

- Datentypen und Wertebereiche sind auf der Ebene einzelner Attribute ein wichtiges Instrument, um Datenfehler zu entdecken und auszuschließen.
- Eine weitere Klasse möglicher Fehler wird durch strukturelle Eigenschaften der Datenbank in Verbindung mit der Überprüfung von Schlüsseleigenschaft und von Fremdschlüsseln (Referenzielle

Integrität; siehe Def. 8)
sichergestellt.

4 DML, DCL und DQL von SQL

4.2 SQL-Data Manipulation Language (DML)

4.2.1 Daten einfügen (INSERT- Anweisung)

INSERT INTO Tabelle VALUES (Spalte1, Spalte2,…Spalte n);

INSERT INTO Tabelle (welche Spalten) VALUES (Spalte1, Spalte2,…Spalte n);

INSERT INTO Tabelle (welche Spalten) VALUES SELECT ….

4.2.2 Daten ändern (UPDATE-Anweisung)

UPDATE Tabelle SET Spalte = Neuer Wert WHERE Bedingung

4.2.3 Daten löschen (DELETE-Anweisung)

DELETE FROM Tabelle WHERE Bedingung

4.3 SQL-Data Control Language (DCL)

Data Control Language = Datenkontrollsprache dient dazu, die DB vor unautorisierten Zugriffen zu schützen.

4.3.1 Berechtigungen vergeben (GRANT)

GRANT [welche Aktionen sind erlaubt] ON [bei welchen Tabellen]] TO [welche Benutzer und Gruppen]

4.3.2 Berechtigungen entziehen (REVOKE)

REVOKE [welche Aktionen werden entzogen] ON [bei welchen Tabellen] TO [welche Benutzer und Gruppen]

Attribut CHECK(Attribut IN(‚X’,’Y’,’Z’);

5 Theorie-Fragen

Was ist der Unterschied zwischen Information und Daten?

Daten enthalten neben Informationen auch Redundanzen. Informationen sind das, was für den Benutzer wichtig ist. Daten alleine stellen keine Information dar, da der Bezug zu Zweck und Nutzen bzw. der Zusammenhang fehlt.

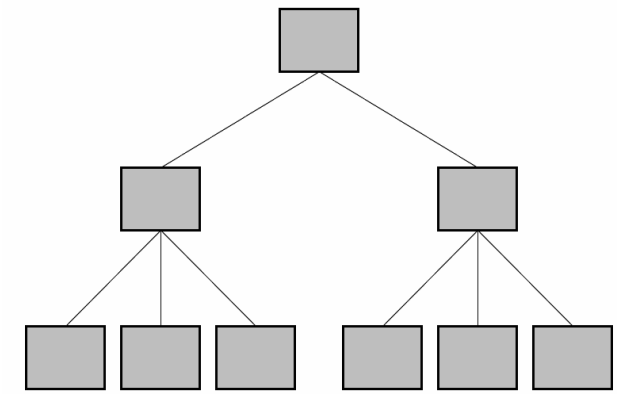
Wie nennt man die Überschrift einer Tabelle?

Metainformation

6 Glossar

6.1 Hierarchisches Datenbankmodell

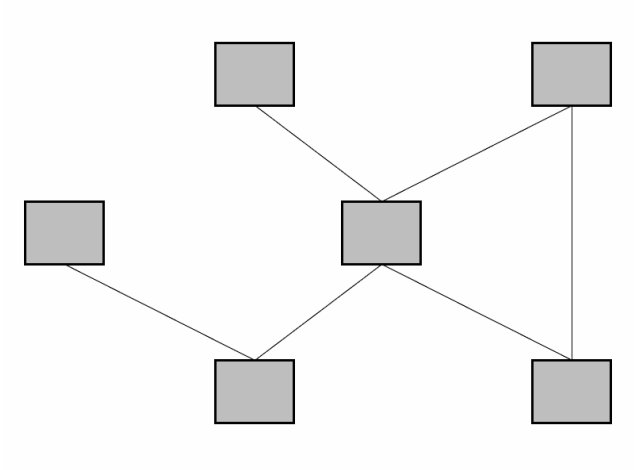
Ein Hierarchisches Datenbankmodell ist das älteste Datenbankmodell, es bildet die Realwelt durch eine hierarchische Baumstruktur ab. Jeder Satz (Record) hat also genau einen Vorgänger, mit Ausnahme genau eines Satzes, nämlich der Wurzel der so entstehenden Baumstruktur.
Die Daten werden in einer Reihe von Datensätzen gespeichert, mit denen verschiedene Felder verknüpft sind. Die Instanzen eines bestimmten Datensatzes werden als Datensatzabbild zusammengefasst. Diese Datensatzabbilder sind vergleichbar mit den Tabellen einer relationalen Datenbank.
Verknüpfungen zwischen den Datensatzabbildern werden in hierarchischen Datenbanken als Eltern-Kind-Beziehungen (Parent-Child Relationships, PCR) realisiert, die in einer Baumstruktur abgebildet werden. Der Nachteil von hierarchischen Datenbanken ist, dass sie nur mit einem solchen Baum umgehen können. Verknüpfungen zwischen verschiedenen Bäumen oder über mehrere Ebenen innerhalb eines Baumes sind nicht möglich.



6.2 Netzwerkdatenbankmodell

Das Netzwerkdatenbankmodell wurde von der Data Base Task Group (DBTG) des Programming Language Committee (später COBOL Committee) der Conference on Data Systems Language (CODASYL) vorgeschlagen, der Organisation die auch für die Definition der Programmiersprache COBOL verantwortlich war. Es ist auch unter den Namen "CODASYL Datenbankmodell" oder "DBTG Datenbankmodell" bekannt und entsprechend stark von Cobol beeinflusst. Der fertige DBTG-Bericht wurde 1971, etwa zur gleichen Zeit wie die ersten Veröffentlichungen über das relationale Datenbankmodell, vorgestellt. Er enthielt Vorschläge für drei verschiedene Datenbanksprachen: Eine Schema Data Description Language oder Schema-Datenbeschreibungssprache, eine Subschema Data Description Language oder Subschema-Datenbeschreibungssprache und eine Data Manipulation Language oder Datenmanipulationssprache.

Das Netzwerk-Modell fordert keine strenge Hierarchie sondern kann auch m:n-Beziehungen abbilden, d. h. es kann ein Datensatz mehrere Vorgänger haben. Auch können mehrere Datensätze an oberster Stelle stehen. Es existieren meist unterschiedliche Suchwege, um zu einem bestimmten Datensatz zu kommen. Man kann es als eine Verallgemeinerung des hierarchischen Datenbankmodells sehen.



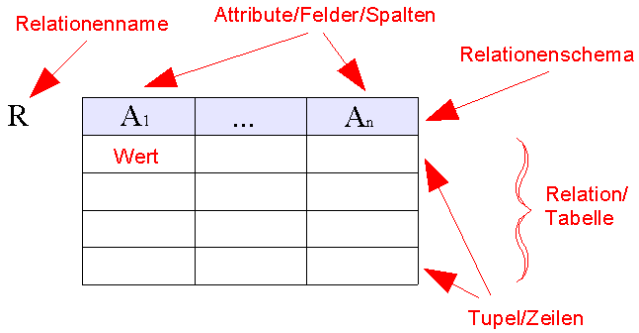
6.3 Relationales Datenbankmodell

Eine relationale Datenbank ist eine Datenbank, die auf dem relationalen Datenbankmodell basiert. Das Datenbankmodell wurde von Edgar F. Codd 1970 erstmals vorgeschlagen und ist heute, trotz einiger Kritikpunkte, ein etablierter Standard zum Speichern von Daten. Das zugehörige Datenbankmanagementsystem wird als das relationale Datenbankmanagementsystem (RDBMS) bezeichnet. Bekannt im Zusammenhang mit relationalen Datenbanken ist die Datenbanksprache SQL, zum Abfragen und Manipulieren der Daten in der Datenbank.

Grundlage des Konzeptes relationaler Datenbanken ist die Relation, ein im mathematischen Sinn wohldefinierter Begriff. Dabei handelt es sich im Wesentlichen um eine mathematische Beschreibung für eine Tabelle (siehe dazu Datenbank Relation). Operationen auf diesen Relationen werden durch die Relationale Algebra bestimmt, diese Operationen finden im Sprachumfang von SQL Berücksichtigung.

Trotz der mathematischen, abstrakten Definition des Datenbankmodells, sind Relationale Datenbanken

vergleichsweise einfach und flexibel zu handhaben. Dies hatte großen Einfluss auf den Erfolg dieser Datenbanktechnik.



6.4 Objektorientiertes Datenbankmodell

Eine objektorientierte Datenbank ist eine Datenbank, deren Inhalt Objekte im Sinn der Objektorientierung sind. Als ein Objekt wird die Zusammenfassung von zugehörigen Attributen bezeichnet, also gehört zum Beispiel die Farbe und das Gewicht eines Autos zu dem Objekt Auto. Attribute beschreiben ein Objekt näher. Daten und Methoden werden nicht getrennt gespeichert.
Der Vorteil einer objektorientierten Datenbank liegt in der Möglichkeit, Objekte ineinander zu schachteln, um Strukturen abbilden zu können, wie zum Beispiel Firma(Abteilung(Mitarbeiter)).
Im englischen und auch im deutschen Sprachgebrauch ist anstelle der Bezeichnung objektorientierte Datenbank auch die Bezeichnung Objektdatenbank (engl. object database) gebräuchlich. Diese Bezeichnung ist kürzer und genauer, denn die Datenbank selbst ist nicht objektorientiert, sondern speichert nur Objekte.