Aus- und Fortbildungszentrum

AFZ
wir bilden zukunft

Entwicklung von Datenbankanwendungen

Datenbankgrundlagen

Freie
Hansestadt
Bremen

Impressum

Herausgeber

Aus- und Fortbildungszentrum
für den bremischen öffentlichen Dienst
Doventorscontrescarpe 172C

28195 Bremen

Redaktion und Koordination

Lehreinheit für Informationstechnologien
Aus- und Fortbildungszentrum
Doventorscontrescarpe 172C

28195 Bremen

Tel.: +49 (0)421 361-16999
E-Mail: office@afz.bremen.de
E-Mail-Hotline: cc-egov@afz.bremen.de

1. Datenbanken - Begriffe

Datenbank

Sammlung strukturierter Informationen

Datenbank-Management-System (DMS)

Programm zur Speicherung und Organisation der Daten

Relationales Datenbank-Management-System

Die Daten werden in verschiedenen Tabellen (Relationen) gespeichert. Zwischen den Tabellen bestehen logische Zusammenhänge über die Inhalte von Datenfeldern.

MS-Access ist ein relationales Datenbank-Management-System.

Datenbankanwendung

Programm zur Verwaltung, Pflege, Anzeige, Ausgabe und Auswertung der Daten.

Kursverwaltung.accdb ist eine mit MS-Access erstellte Datenbankanwendung.

TnNr	Name	Vorname	Gesc	Geburtsdatum
102	Rausch	Mathilde	w	12.07.1957
103	Krehl	Axel	m	28.02.1950
104	Schmitthuber	Erich	m	09.01.1953
105	Krone	Hartmut	m	05.05.1961
106	Brand-Müller-W	Gerold	m	27.03.1960
107	Primus	Sophie	w	07.02.1954
108	Buchholz	Michael	m	25.05.1939

Tabelle (Relation)

Sammlung von Daten zu einem Thema in Zeilen und Spalten. Beispiel Tabelle TabTeilnehmer.

Name	Abendrot
Vorname	Hannelore
Geschlecht	w
Geburtsdatum	13.05.1962
Beurlaubt	☐
Funktion	Sachbearbeiter
Telefon	6005
BKZ	061

Datensatz (Tupel)

Alle Informationen zu einem Fall - entspricht einer Zeile der Tabelle. Beispiel: alle Daten zu Hannelore Abendrot.

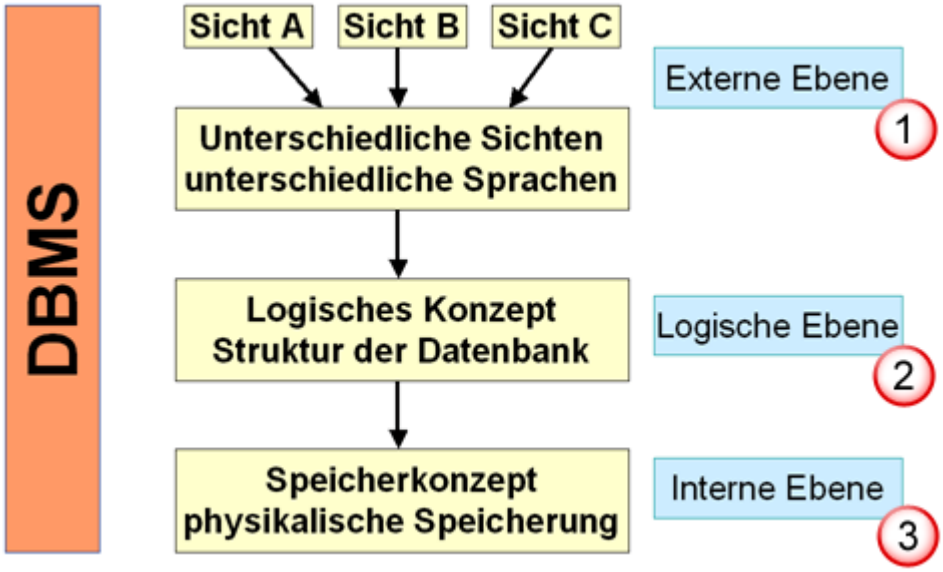
Vorname	Hannelore

Datenfeld (Attribut)

Jede einzelne Information innerhalb der Datensätze - entspricht einer Spalte der Tabelle. Beispiel: Hannelore ist Inhalt des Datenfeldes Vorname.

2. Datenbank-Architektur

Das ANSI / SPARC* – Drei-Schichtenmodell (1975) gliedert eine Datenbank in drei unterschiedliche, voneinander getrennte Ebenen. Diese werden als extern, logisch und intern beschrieben.



① Jeder Ausschnitt / jede Benutzersicht muss aus der logischen Gesamtsicht aller Daten herzuleiten sein. Es müssen neue Sichten erstellt / vorhandene Sichten geändert werden können, ohne die Struktur der DB zu ändern.

② Die Struktur enthält alle relevanten Beschreibungen der Daten (Relationen, Attribute), die Konsistenzbedingungen usw.

③ Der Speicherplatz muss verändert werden können, ohne die logische Struktur der DB zu berühren (andere Speichermedien, Aufteilung auf verschiedene Platten, Spiegelung ...)

* ANSI American Nation Standard Institute
SPARC Standard Planing and Requirements Comite

3. Relationales Datenmodell

Eine relationale Datenbank *)speichert alle Daten (auch die Daten über die Daten) grundsätzlich in 'einfachen' Tabellen.

Das relationale Modell besteht aus Definitionen von Objekten, Operationen und Regeln.

3.1 Relationale Objekte

Relation	Tabellarische Sammlung der Daten Eigenschaften einer Relation: – keine doppelten Tupel – Tupelreihenfolge ist nicht definiert – Attributreihenfolge ist nicht definiert – Attributwerte sind atomar
Degree	Ausdehnungsgrad der Tabelle (Anzahl der Tupel)
Attribut	Spalte (Datenfeld) Anzahl ist definiert
Domäne	Wertebereich eines Attributs
Tupel	Zeile (Datensatz) Anzahl ist variabel
Candidate Key	Eindeutiger Schlüssel Alle Werte sind eindeutig; er kann aus einem oder mehreren Attributen bestehen; eine Relation kann mehrere Candidate Keys besitzen.
Primary Key	Haupt-(Primär-)schlüssel Ein Candidate Key wird zum Primary Key erklärt. Dieser identifiziert jeden Tupel eindeutig.
Foreign Key	Fremdschlüssel Dessen Werte sind in einer anderen Relation als Primary Key definiert.

3.2 Relationale Integritätsregeln

Entity-Integrität

Mit der Entity-Integrität wird sichergestellt, dass jedes Tupel (Entity) in einer Relation einen eindeutigen Schlüssel besitzt. Dieses Attribut darf zu keinem Zeitpunkt einen NULL-Wert enthalten.

Referentielle Integrität

Eine Relation R2 besitzt einen Foreign Key, der auf einen Primary Key in einer Relation R1 verweist. Dann muss jeder Wert des Foreign-Key in R2 gleich einem Wert des Primary Key in R1 sein oder der Wert des Foreign-Key ist ein NULL-Wert.

*) Der Begriff relationale Datenbank wurde 1970 von E.F. Codd eingeführt. Im Unterschied zu hierarchischen oder netzwerkartigen Datenbankmanagementsystemen basiert das relationale Modell auf den mathematischen Grundlagen der relationalen Algebra.

4. Datenbankentwicklung

4.1 Verfahren und Darstellungsmethoden

- Anforderungsdefinition
- Aufgaben-Analyse
- Kommunikationsanalyse
- Input / Output - Analyse
- Modultechnik
- Datenfluss-Analyse
- Programmablaufplan / Struktogramm
- Data-Dictionary
- Strukturierte Analysen (z. B. Entity Relationship)

4.2 Vorteile einer strukturierten Datenbankentwicklung

Analyse-Modelle ...

- zwingen Entwickler und Projektbeteiligte über das Problemfeld und die Aufgabenstellung nachzudenken
- zeigen, wie weit die Anforderungen und das Umfeld verstanden worden sind.

Visualisierte Modelle bieten eine gute Gesprächsgrundlage

- decken Schwachstellen und Missverständnisse auf
- verdeutlichen / visualisieren Schnittstellen
- ermöglichen eine bessere Planung, Durchführung und Aufteilung von Entwicklungsaufgaben
- sind der erste Schritt zu einer guten Dokumentation

5. Entity Relationship-Diagramm

Für die Entwicklung komplexer Strukturen, deren Zusammenhänge nicht vollständig bekannt sind, eignet sich die **Entity Relationship-Methode**:

- Für die zu modellierende ‚Mini-Welt' werden zunächst Hauptgruppen - **Entities** - gebildet.

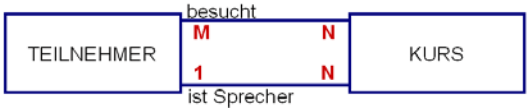
Entity: TEILNEHMER KURS

- Die Beziehung - **Relationship** - zwischen zwei Entities wird definiert.

Relationship: besucht / wird besucht von

Der Grad einer Beziehung - **Degree** - muss definiert werden:

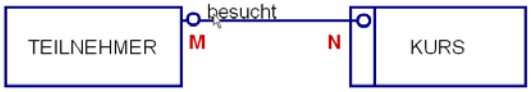
1 : 1 - Beziehung
1 : N - Beziehung
M : N - Beziehung



Beispiele für Beziehungen:

Teiln. müssen keinen Kurs besuchen, Kurse müssen keine Teiln. haben

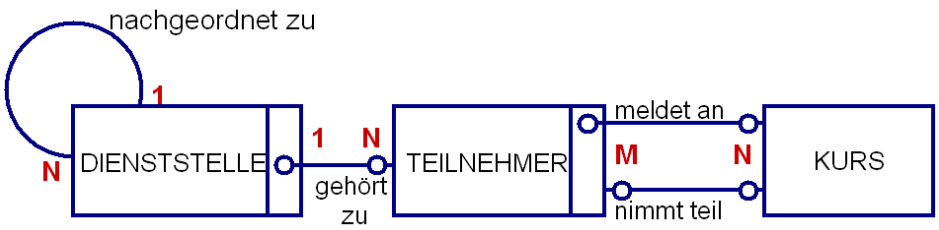
Teiln. müssen keinen Kurs besuchen, Kurse müssen mindestens einen Teiln. haben



Teiln. müssen mindestens einen Kurs besuchen, Kurse müssen mindestens einen Teiln. haben

Teiln. müssen mindestens einen Kurs besuchen, Kurse müssen keine Teiln. haben

Ein fertiges Diagramm:



6. Datenstrukturen entwickeln

Die verlässliche Nutzung von Daten, die in verschiedenen Tabellen gespeichert sind, setzt ein Datenmodell voraus, das bestimmte Kriterien erfüllt:

- keine unerwünschten Abhängigkeiten beim Ändern, Anfügen oder Löschen von Daten (Update-Anomalien)
- geringer Aufwand bei Ergänzungen oder Umstrukturierungen
- keine mehrfach gespeicherten Daten (Redundanzen)
- verständliches Datenmodell für Benutzer und Entwickler

Ein derartiges Modell wird durch die 'Normalisierung' der Daten, das heißt durch das Aufteilen der Daten in mehrere Relationen (Tabellen) erreicht.
‚Nebenbei' wird bewirkt, dass sich der Datenbankdesigner systematisch und intensiv mit den Daten und damit mit der Fachanwendung beschäftigen muss.

E.F. Codd unterscheidet drei Normalisierungsregeln. Später wurden diese um zwei weitere Regeln ergänzt, die aber in der Praxis keine Bedeutung gewonnen haben. Die Stufen der Normalisierung werden nacheinander ausgeführt. Das heißt, der Prozess der Normalisierung beginnt immer mit der 1. Normalform. Nachdem die Daten die 1. Normalform erfüllen, werden die Tabellen so zerlegt, dass sie die 2. Normalform erfüllen usw.

1. Normalform

Eine Relation ist in der ersten Normalform, wenn alle Attribute nur atomare Werte beinhalten, d. h. jede trennbare Einheit wird in einem eigenen Feld gespeichert.

Im folgenden Beispiel beinhalten die Felder Name und Anschrift mehrere Werte:

Name	Vorname	Anschrift
Dr. von Oberen	Heinrich	Südweststr. 123, 28195 Bremen

Jede Information wird eindeutig bezeichnet; die Reihenfolge ist beliebig.

2. Normalform

Eine Relation ist in der zweiten Normalform, wenn sie sich in der ersten Normalform befindet und jedes Nicht-Schlüssel-Feld voll funktional abhängig ist vom (gesamten) Schlüssel.

Im folgenden Beispiel für eine Kursanmeldung ist die Bezeichnung des Kurses lediglich vom Feld KursID aber nicht vom gesamten Schlüssel abhängig:

TnNr	KursID	KursBezeichnung
789	99/72.03f	Access für Quereinsteiger

(Der Primärschlüssel setzt sich aus der TnNr und der KursID zusammen.)

3. Normalform

Eine Relation ist in der dritten Normalform, wenn sie sich in der ersten und in der zweiten Normalform befindet. Es sind zusätzlich keine funktionalen Abhängigkeiten zwischen Feldern erlaubt, die nicht als Schlüssel definiert sind.

Im folgenden Beispiel ist die Bezeichnung der Dienststelle funktional nicht vom Schlüssel, sondern von der BKZ abhängig.

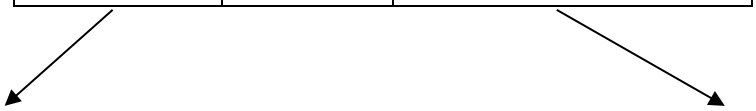
TnNr	Name	Vorname	BKZ	Dienststelle
789	Mustermann	Elfriede	050	Standesamt

(Primärschlüssel ist die TnNr.)

4. und 5. Normalform

Die vierte und fünfte Normalform behandeln paarweise auftretende mehrdeutige Abhängigkeiten wie im folgenden Beispiel:

Vorname	Geschlecht	Funktion
Elfriede	W	Sachbearbeitung
Fritz	M	Lehrpersonal
Gudrun	W	Sachbearbeitung
Fritz	M	Sachbearbeitung
Elfriede	W	Lehrpersonal
Heinrich	M	Lehrpersonal
Elfriede	W	Schreibtätigkeit



Vorname	Geschlecht
Elfriede	W
Fritz	M
Gudrun	W
Heinrich	M

Geschlecht	Funktion
W	Sachbearbeitung
M	Sachbearbeitung
W	Lehrpersonal
M	Lehrpersonal
W	Schreibtätigkeit

In der Praxis werden die 4. Und 5. Normalform selten erfüllt, da damit einerseits ein hoher Aufwand und Performance-Verluste verbunden sind, andererseits die Notwendigkeit der Normalisierung nicht zwingend erscheint.

7. Datenbank-Join

Ein Join ist die Verbindung zweier Relationen über Attribute.

Dabei gelten folgende Regeln:

- Die Attribute, über die der Join ausgeführt wird, müssen keine Schlüssel sein.
- Die Join-Attribute der beiden Relationen müssen nicht den gleichen Namen haben.
- Die den Join-Attributen zugrundeliegenden Domänen müssen gleich sein.
- Jede Relation kann mit jeder anderen gejoint werden (auch mit sich selbst).

 Das Ergebnis eines Join ist immer eine Relation!

Equi-Join Die Beziehung zwischen den Join-Attributen kann nur mit dem Vergleichsoperator = dargestellt werden.
(Verwendung in MS Access)

Inner Join Es werden nur Tupel in der Ergebnisrelation erzeugt, wenn der Attributwert der ersten Relation auch in der zweiten Relation vorkommt ('natürlicher Join').
(Standardeinstellung in MS Access)

Outer Join Es werden zumindest alle Tupel einer der beiden Relationen ausgegeben (Left Outer Join oder Right Outer Join).
(In MS Access einstellbar über Verknüpfungseigenschaften)

Theta-Join Die Beziehung zwischen den Join-Attributen kann mit einem der Vergleichsoperatoren =, >, >, <>, >=, <= dargestellt werden.
(In MS Access nur in SQL-Abfragen nutzbar)

Natural Join In der Ergebnisrelation sind die gleichen Attributwerte nur einmal enthalten
(In MS Access über Abfrage-Eigenschaft *Keine Duplikate* einstellbar)**Auto-Join, Self-Join**
Verknüpfung einer Relation mit sich selbst
(In MS Access Tabelle mehrfach zum Abfrageentwurf hinzufügen)

8. Relationale Operationen

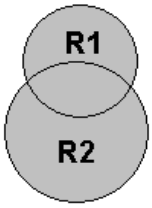
Restriktion

Restriktion wird auch als Selektion bezeichnet. Sie extrahiert auf Grund einer Bedingung (=, <, >, >=, >=, <>) **Tupel** aus einer Relation. Dies ist eine Auswahl der Tabellenzeilen.

Projektion

Die Projektion extrahiert **Attribute** aus einer Relation. Dies ist eine Auswahl der Tabellenzeilen einer Relation. Dadurch können Tupel der Relation zusammenfallen, da keine Duplikate zugelassen sind.

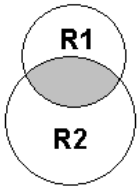
Union



Für eine Vereinigung von zwei Relationen müssen diese vereinigungskompatibel sein, d. h. sie müssen die gleiche Anzahl Attribute mit den entsprechenden Domänen besitzen.

Die Vereinigung der beiden Relationen R1 und R2 , ist die Menge aller Tupel, die entweder zu R1 oder zu R2 gehören. Anders ausgedrückt: die Vereinigung ist wieder eine Relation, die alle Tupel von R1 und R2 enthält, wobei Duplikate gelöscht werden.

Intersektion

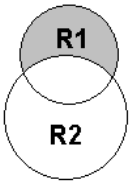


Der Durchschnitt zweier Relationen R1 und R2, ist die Menge aller Tupel, die sowohl in R1 als auch in R2 vorkommen. Die Ergebnisrelation enthält also alle Tupel, die in R1 und R2 gleichermaßen vorhanden sind.

Produkt

Das Kartesische Produkt zweier Relationen, R1 mal R2 ist die Menge aller verschiedenen Tupel, aller möglichen Kombinationen der einzelnen Attribute von R1 und R2.

Differenz



Die Differenz der beiden Relationen, R1 minus R2, ist die Menge aller Tupel, die zu R1 und nicht zu R2 gehören.

Division

Die Divison ist die zum (kartesischen) Produkt inverse, also umgekehrte Operation. Teile eine Relation R1 durch ein Attribut aus R2, wobei das Attribute aus R1 und R2 derselben Domäne entspringen müssen.

Diese Operationen dürfen nicht mit Befehlen – z. B. SQL-Befehlen – verwechselt werden. In der jeweiligen Befehlssyntax werden sie unterschiedlich formuliert oder sind unter Umständen gar nicht vorhanden.

9. SQL – Begriffe, Standards

SQL = Structured Query Language (strukturierte Abfragesprache)

SQL ist eine standardisierte Data Sublanguage zum Erstellen, Bearbeiten und Kontrollieren von relationalen Datenbanken.

SQL ist keine prozedurale Computersprache; d. h. SQL kann nicht alleinstehend, sondern nur in Verbindung mit einer anderen Sprache oder einer Anwendung genutzt werden.

SQL ...

- beruht auf dem relationalen Datenbankmodell
- ist eine high level Computersprache (mit normalen Englisch-Kenntnissen verständlich)
- ist interaktiv (ad hoc Datenbankabfragen)
- ist hersteller-unabhängig
- ist portabel zwischen verschiedenen Computersystemen
- ist im Standard von ANSI, ISO, X/OPEN ... enthalten
- wird von Microsoft ODBC unterstützt (Open Database Connectivity).

9.1 Befehlsgruppen in SQL

Definition von Daten (Data Definition Language - DDL)

- z.B. leere Datenbankstrukturen wie Tabellen,
- Indizes, Views anlegen, Strukturen und
- existierender Objekte ändern, Objekte löschen

Manipulation von Daten (Data Manipulation Language - DML)

- Daten anzeigen, auswerten
- Update-Anweisungen wie Einfügen, Ändern oder Löschen

Datenbankverwaltung (Data Control Language - DCL)

- Benutzerrechte, Datenbankgröße, Transaktionen

9.2 SQL-Standards

Es bestehen verschiedene SQL-Standards:

- SQL 86 (SQL1)
- SQL 89
- SQL 92 (SQL2)
- SQL3

Innerhalb dieser Standards gibt es jeweils drei Level:
(Entry – Intermediate – Full).

MS Access verwendet SQL 89 (Entry-Level) mit eigenem Dialekt.

MS Access-Projekte (ab Office 2000) verwenden SQL 92.

9.3 SQL – Syntax

Auswahlabfrage

Eine Abfrage besteht immer mindestens aus den Abschnitten **SELECT** und **FROM**; SELECT definiert die Spalten (Felder) und FROM gibt die Herkunftstabelle(n) an.

Beispiel: SELECT tabTeilnehmer.Nachname, tabTeilnehmer.Vorname FROM tabTeilnehmer;

Im SELECT-Abschnitt können die Optionen ALL, DISTINCT, DISTINCTROW oder TOP eingesetzt werden. Mit AS können Spaltenüberschriften definiert werden. Wenn im FROM-Abschnitt mehrere Tabellen angegeben werden können sie mit INNER JOIN … ON, RIGHT JOIN … ON oder LEFT JOIN … ON verknüpft werden.

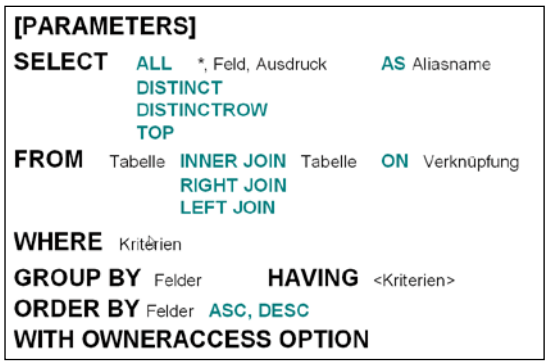
Um die Daten einzuschränken, wird ein **WHERE**-Abschnitt angefügt, in dem die Felder, Operatoren und Vergleichsausdrücke angegeben werden.

Beispiel: SELECT tabTeilnehmer.Nachname, tabTeilnehmer.Vorname FROM tabTeilnehmer WHERE tabTeilnehmer.Nachname = "Meyer";

Für eine gruppierte Abfrage wird der Abschnitt **GROUP BY** mit den Feldern für die Gruppierung hinzugefügt. Er kann für Eingrenzungen auf bestimmte Gruppierungsergebnisse durch **HAVING** ergänzt werden.

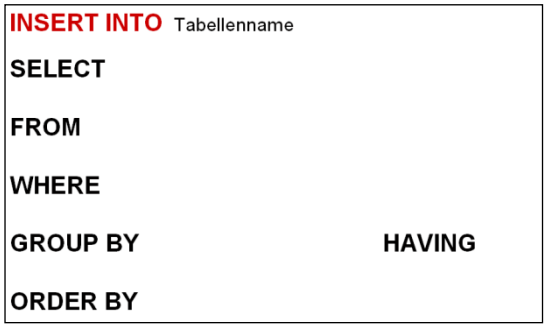
Abschließend kann die Sortierung im Abschnitt **ORDER BY** angegeben werden. Die Sortierfolge wird mit ASC oder DESC festgelegt.

Für Parameterabfragen wird dem SQL-Statement der Abschnitt **PARAMETERS** vorangestellt.



Anfügeabfrage

Eine Anfügeabfrage besteht aus den Abschnitten **INSERT INTO**, **SELECT**, **FROM**, **WHERE**, **GROUP BY**, **HAVING** und **ORDER BY**. Im Abschnitt INSERT INTO wird die Tabelle angegeben, der die Datensätze aus dem Abfrageergebnis angefügt werden.



Tabellenerstellungsabfrage

Eine Tabellenerstellungsabfrage besteht aus den Abschnitten **SELECT, INTO**, **FROM**, **WHERE**, **GROUP BY**, **HAVING** und **ORDER BY**. Im Abschnitt INTO wird die Tabelle benannt, die mit den Datensätzen aus dem Abfrageergebnis gefüllt wird.

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

Aktualisierungsabfrage

Eine Aktualisierungsabfrage besteht aus den Abschnitten **UPDATE**, **SET** und **WHERE**. Update benennt die Tabelle und SET enthält die Aktualisierungsanweisung.

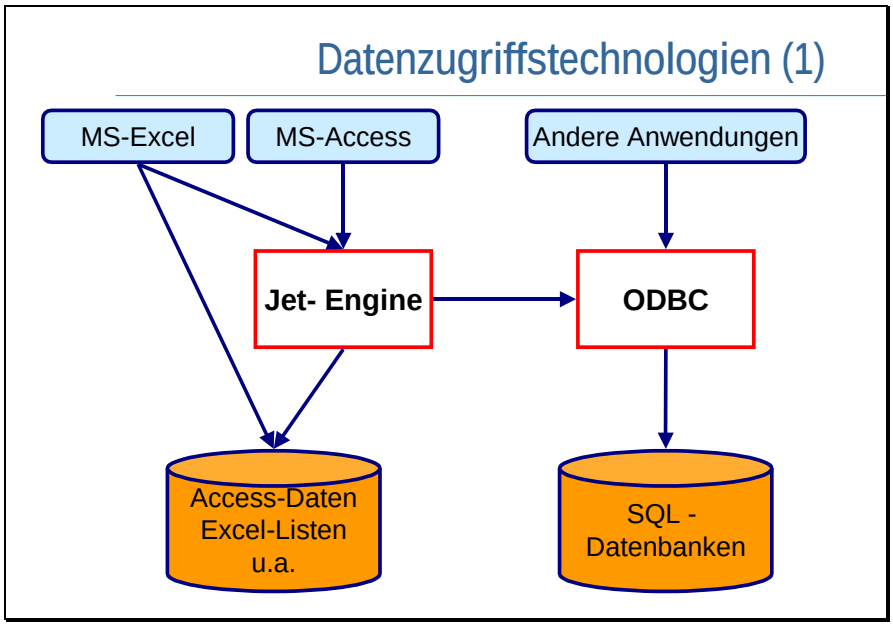
```
UPDATE Tabellenname SET Feld = Ausdruck
WHERE
```

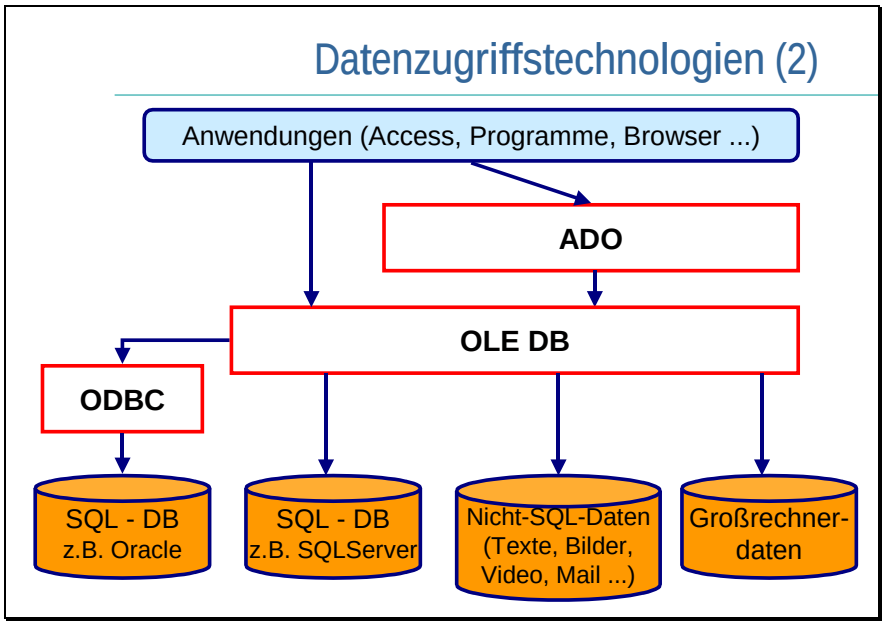
Löschabfrage

Eine Löschabfrage besteht aus den Abschnitten **DELETE**, **FROM** und **WHERE**. Da eine Löschabfrage immer Datensätze vollständig löscht, wird im Abschnitt DELETE immer das * angegeben.

```
DELETE *
FROM
WHERE
```

10. Datenzugriffstechnologien





11. Data Warehouse

Data Warehousing Vorgang, Daten aus einer Vielzahl von Systemen zu sammeln, zu bereinigen, zu sichten und zur Analyse und Berichterstattung zu Verfügung zu stellen.

Data Warehouse sammelt Daten in einer Datenbank und stellt die Informationen unterschiedlichen, auf Planung und Entscheidung ausgerichteten Auswertungssystemen zur Verfügung.
Dazu wird eine grobe Zweiteilung der Daten vorgenommen:
Daten für operative Zwecke / Daten für dispositive Zwecke.
Die Daten für dispositive Zwecke bilden das Data Warehouse.
Es stellt alle für das Gesamtunternehmen relevanten Daten zur Verfügung. Dies setzt die umfangreiche Planung und Entwicklung eines unternehmensweiten Datenmodells und die entsprechende Datenaufbereitung voraus (Top-Down-Ansatz).

Data Marts sind kleine Data Warehouses, die speziell auf Abteilungen / Teilbereiche zugeschnitten sind. Im Gegensatz zum Data Warehouse können die Daten in kleinen Netzwerken zur Verfügung gestellt werden. (Buttom-Up-Ansatz).

Knowledge Discovery
Prozess der Entdeckung – von der Formulierung einer Frage bis zur Auswertung der Antworten; umfasst Datenvorbereitung, Datenauswahl, - bereinigung und das Data Mining.

Data Mining Aufspüren von Geschäftsinformationen in umfangreichen Datenbanken; eine spezielle Form der Datenanalyse, die versteckte Trends aufzeigt.

OLAP (Online Analytical Processing)
Fähigkeit, Daten hierarchisch zu analysieren und intuitiv aus allen Blickwinkeln bewerten zu können. Grundlage ist immer ein mehrdimensionales Datenmodell.
Die Daten werden in Cubes gehalten. Dies sind Datenräume, die nach beliebigen Kriterien intuitiv erforscht werden können. Jeder Cube besteht aus Dimensionen und den Measures (Elementen) mit den quantitativen Inhalten. Dimensionen können Hierarchien aufweisen.

OLAP setzt voraus, dass die Einzeldatensätze für jede Dimension im Voraus zusammengefasst (denormalisiert) und berechnet (aggregiert) werden. (Die Größe der OLAP-Datenbanken ist schwer vorhersehbar, da sie exponentiell anwachsen. Die Datenexplosion kann bei 1000 : 1 liegen.)
OLAP Architekturen:

ROLAP	relationales OLAP
MOLAP	multidimensionales OLAP
HOLAP	hybrides OLAP (Detaildaten relational, Aggreationen in separaten MOLAP-Containern)

Lernmaterial

Lernmaterial – Schulungsunterlagen, Übungsdateien, Lernprogramme und Tipps & Tricks – finden Sie im Internet unter der Adresse:

https://www.afz.bremen.de/lernen

Wählen Sie das gewünschte Thema über die Menüstruktur am oberen Rand der Seite oder aus der Liste aus, die Sie im rechten Bereich über die Infobox **Gesamtliste der Schulungsunterlagen** in den einzelnen Untermenüpunkten erreichen können.

Hier können Sie Themen nachschlagen, Ihre Kenntnisse aktualisieren (z. B. bei neuer Programmversion) oder sich zusätzliche Themen erarbeiten. Sie können das Lernmaterial als **PDF-Dokumente** am Bildschirm lesen, auf Ihrem Computer speichern oder ausdrucken. Zum Teil stellen wir zusätzlich **Übungsdateien** in gepackter Form (Zip-Archiv) zur Verfügung.

Tipps & Tricks

Oft sind es die kleinen Dinge, die die Arbeit am PC erleichtern. Dazu haben wir Tipps und Tricks zusammengestellt. Diese finden Sie sowohl bei den einzelnen Programmen als auch in einer Gesamtliste, die Sie über die Infobox **Tipps und Tricks** im rechten Bereich bei den einzelnen Untermenüpunkten erreichen können. Vielleicht entdecken Sie hier etwas, um Ihre Arbeit effektiver zu gestalten.

Kompetenzzentrum E-Government (CC-EGov)

Sollten Sie als Beschäftigte der Freien Hansestadt Bremen bei Ihrer Arbeit auf Probleme stoßen, die beim Einsatz Ihrer Softwareausstattung auftreten (Probleme mit Word-Dokumenten, Excel-Tabellen etc.), können Sie sich mit Ihren Fragen, Problemstellungen oder Fehlermeldungen telefonisch oder per E-Mail an uns wenden:

cc-egov@afz.bremen.de Tel. 16 999

Beschreiben Sie Ihre Frage bzw. die Fehlersituation und Ihre bisherige Vorgehensweise und fügen Sie die Dateien im Original-Dateiformat als Anlage bei. Wir beantworten Ihre Fragen so schnell wie möglich, in jedem Fall melden wir uns innerhalb weniger Tage bei Ihnen.