OPERATIONEN AUF EINER DATENBANK

Ein Benutzer stellt eine Anfrage:

Die Benutzer einer Datenbank können meist sowohl interaktiv als auch über Anwendungen Anfragen an eine Datenbank stellen:

```
PRINT Flugzeug, Standort
WHERE Status="defekt"
```

Der Anfrage-Prozessor interpretiert die Anfrage

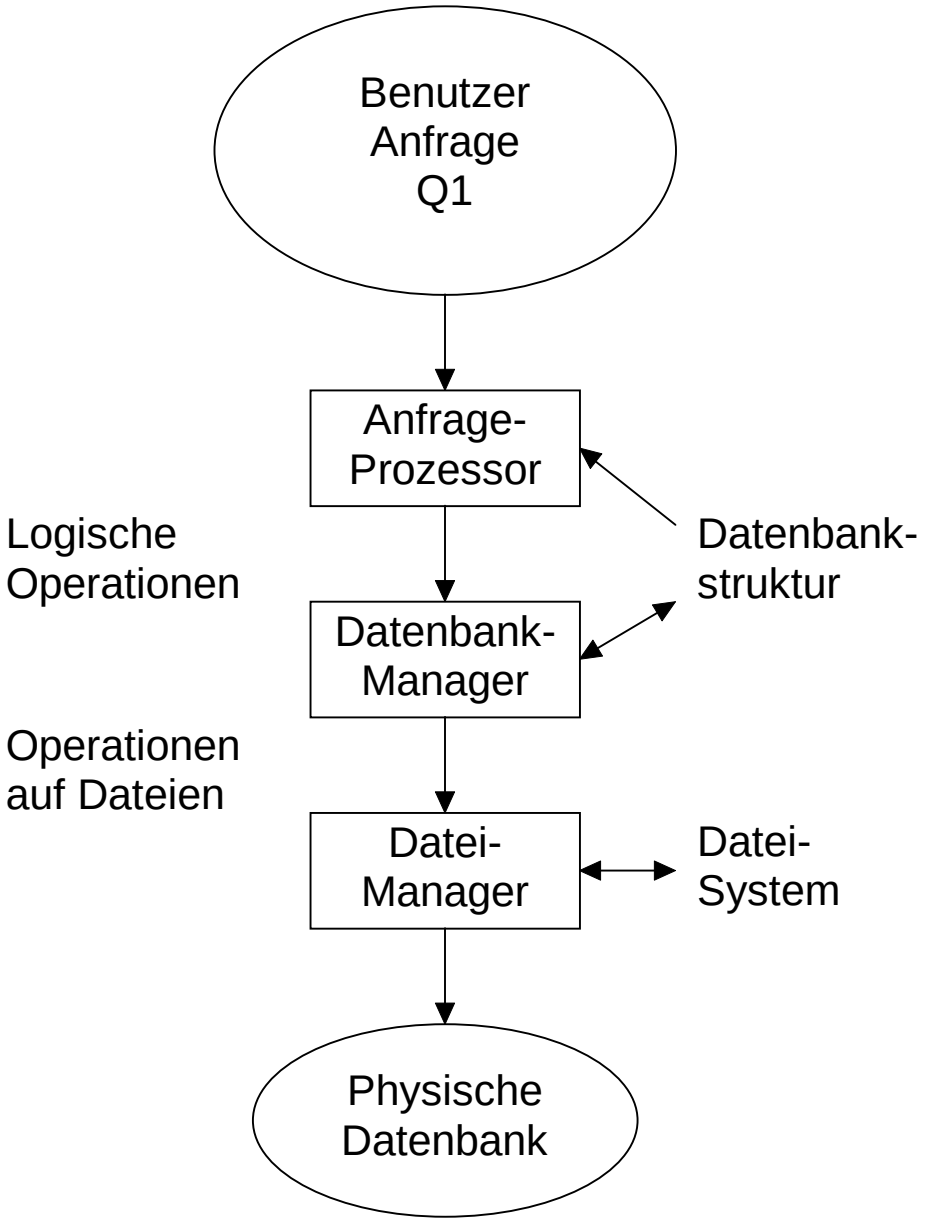
Eine solche Anfrage *Q1* wird an der Anfrage-Prozessor geleitet, der die Anfrage übersetzt (kompiliert), dazu benötigt er Informationen über die Struktur der Datenbank.

Der Datenbank-Manager

Der Datenbank-Manager transformiert die logischen oder algorithmischen Operationen in Operationen die der Datei-Manager verarbeiten kann. Er ist ebenfalls für die Ausgabe des Anfrage-Ergebnisses zuständig.

Der Datei-Manager

Der Datei-Manager holt die Daten aus der physischen Datenbank. Dies ist entweder der „Allzweck-“ Datei-Manager des Betriebssystems oder eine spezielle Ausführung, die besser für die Speicherung der Daten in der Datenbank eingerichtet ist.


Benutzer
Anfrage
Q1
Anfrage-
Prozessor
Logische
Operationen
Datenbank-
struktur
Datenbank-
Manager
Operationen
auf Dateien
Datei-
Manager
Datei-
System
Physische
Datenbank

PHYSISCHE DATENORGANISATION:

1. Überblick
2. Zugriff auf (Daten-)Sätze
3. Hashorganisation
4. Indexdateien
   - B*-Baum
5. Speicherstrukturen für Relationen
6. Join-Algorithmen

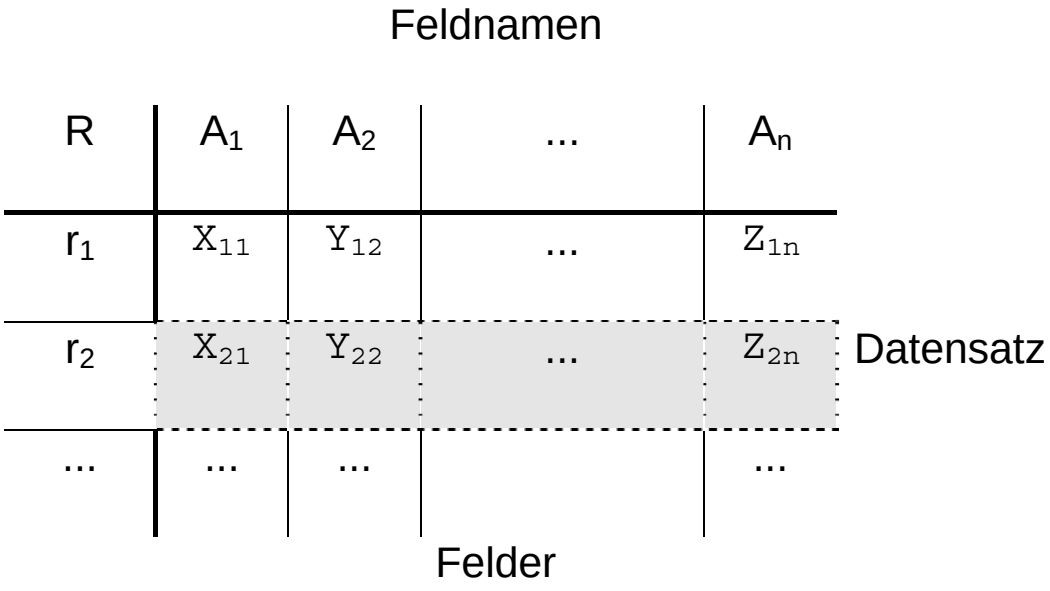
ÜBERBLICK

Problem: Speichern von Dateien, die aus einzelnen **Sätzen (records)** bestehen.

Ein Satz besteht aus einer festen Anzahl von **Feldern**, die zusammenhängend mit **fixem Platzbedarf** und in **festgelegter Reihenfolge** gespeichert sind.

Ein **Schlüssel** ist eine Kombination von Feldern, deren Werte einen bestimmten Satz eindeutig identifizieren.

Das **Format** eines Satzes ist eine Liste von Feldnamen.


Feldnamen
R
A1
A2
...
An
r1
X11
Y12
...
Z1n
r2
X21
Y22
...
Z2n
Datensatz
...
...
...
...
Felder

DATEI AUS DATENSÄTZEN
(FILE OF RECORDS)

Operationen:

1. **Einfügen** eines Satzes. **(insert)**
2. **Löschen** eines Satzes. **(delete)**
3. **Finden** eines Satzes mit einem bestimmten Wert in einem Feld oder einer Kombination von Feldwerten. **(lookup)**
4. **Verändern** eines Satzes. **(modify)**

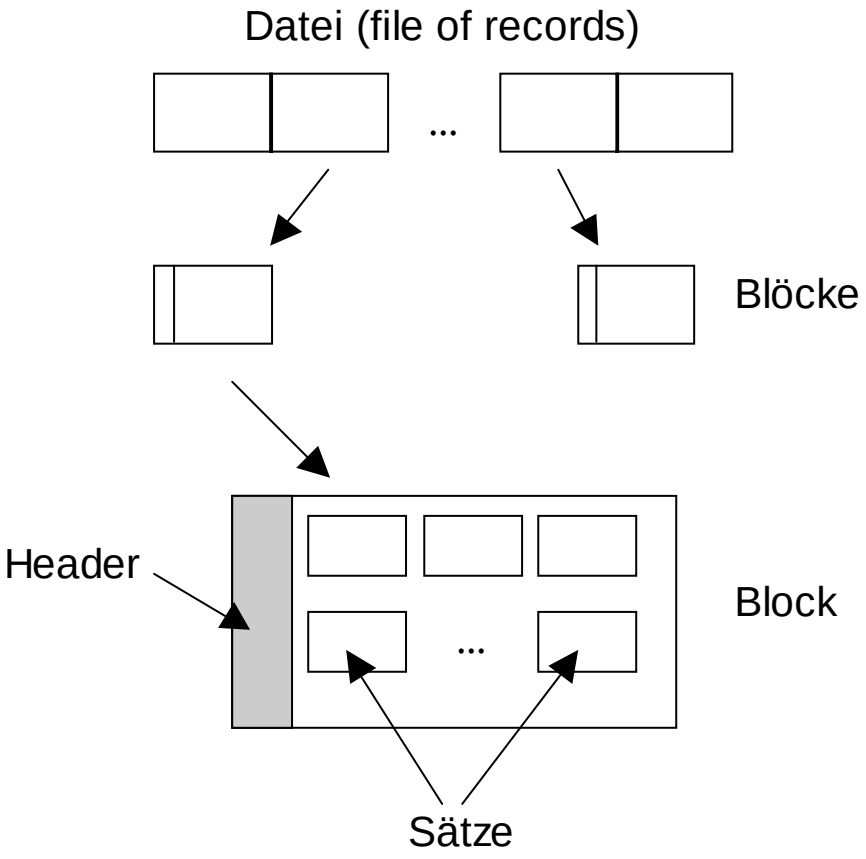
Vereinfachtes Modell für Speicherung auf Massenspeicher:

- Ein Datei-System bietet Zugriff auf eine Menge gleich großer Blöcke (blocks, pages), bestehend aus einer bestimmten Anzahl von Bytes. Typischerweise in einer Größenordnung von 2^9 (512) bis 2^{12} (4096) Bytes aber auch noch größer.

- Der Zugriff auf einen Block erfolgt direkt über dessen (feste) Blockadresse.

- Das Datei-System liefert unbenutzte Blöcke und übernimmt freigewordene Blöcke.

Aufgabe des Datei-Systems:



Blockeinteilung:

- Ein Teil der Bytes in einem Block, der sogenannte Header, dient zum Speichern von Kontrollinformationen (z.B. welche Bereiche des Blocks noch nicht mit Datensätzen belegt sind).

- Der restliche Bereich ist in Subblöcke unterteilt, die jeweils einen Satz aufnehmen können.

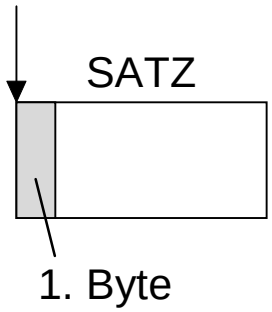
Blöcke werden immer als Ganzes gelesen oder geschrieben. Der Zeitaufwand für einen solchen Blockzugriff ist im allgemeinen wesentlich höher als für die auf einem Block im Hauptspeicher durchzuführenden Operationen.

→ Die Geschwindigkeit von Algorithmen ist bestimmt durch die Anzahl der Blockzugriffe auf dem Sekundärspeicher.

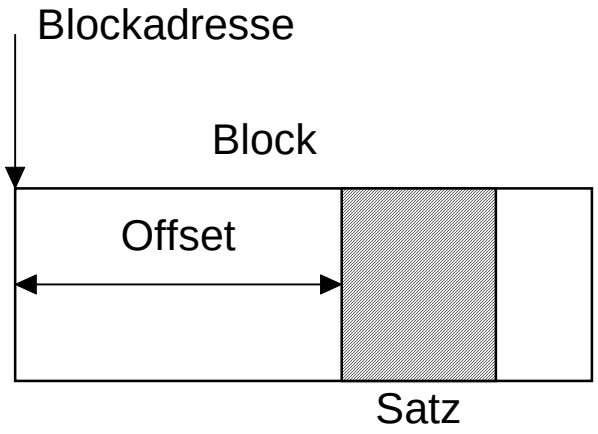
Adresse eines Satzes:

Ein Satz kann auf zwei verschiedene Arten adressiert sein.

1. Er kann direkt durch die Adresse seines ersten Bytes adressiert sein.



2. Er kann indirekt durch die Adresse seines Blocks und einen Offset adressiert sein.

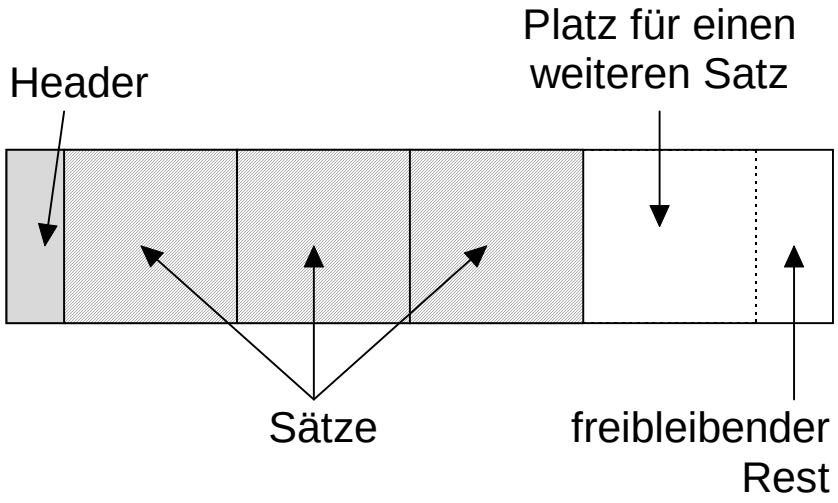


Interpretation von Rohdaten (raw data)

Datensätze im Rohformat (keine weiterführende Speicher-strategien) unterliegen folgenden Bedingungen:

- Festes Format.
- Felder in derselben Reihenfolge.
- Jedes Feld hat eine feste Anzahl von Bytes.
- Jedes Feld hat einen festen Datentyp.

Sätze dieser Art können zu mehreren in einem Block gehalten werden:



Zugriff auf Sätze

Grundlegende Zugriffsarten:

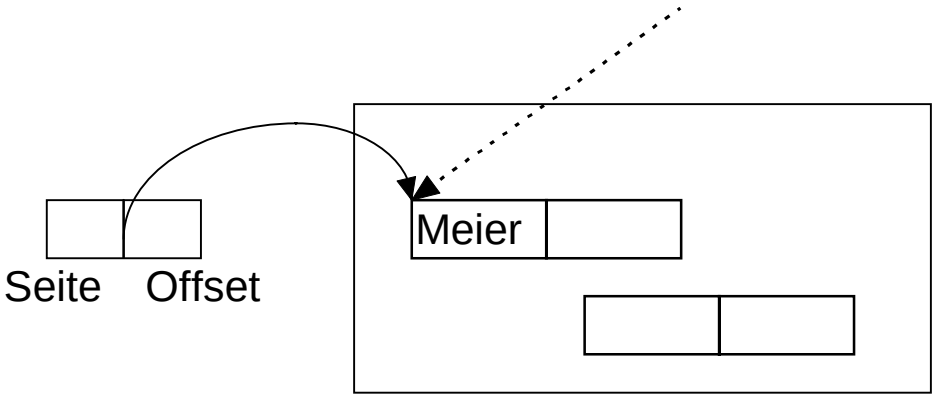
- Finden eines Satzes
- Einfügen eines Satzes
- Löschen eines Satzes

Der Zugriff auf einen Satz kann auf zwei Arten erfolgen:

- über den Schlüssel.
  Bei jedem Zugriff muß die Position des Satzes im Speicher dynamisch neu bestimmt werden
- über seine Adresse (Verwendung von Zeigern).
  Die Adresse besteht im einfachen Fall aus der Adresse des Blocks und der Position des Satzes innerhalb des Blocks.

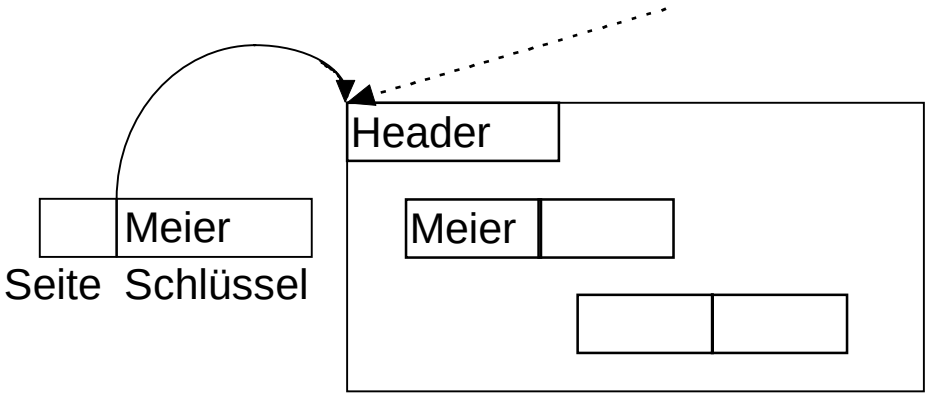
Pinned Records

Wird auf einen Satz über seine Adresse zugegriffen, so ist der Satz, sobald seine Adresse an einer anderen Stelle verwendet wird, auf seiner Position „pinned down“ (festgenagelt). D.h. bei Verlegen oder Löschen des Satzes würden „dangling“ (hängende) Zeiger entstehen.



Teilweise Abhilfe:

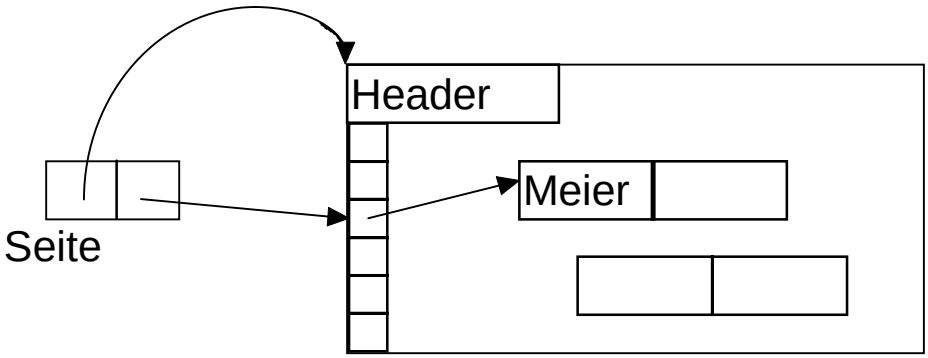
- Die Adresse besteht nur aus der Blockadresse, der Block muss also immer sequentiell nach dem Satz durchsucht werden; dies ermöglicht wenigstens die Verlagerung innerhalb des Blocks.



- wird ein Satz gelöscht, so wird dieser Subblock nicht wieder verwendet. Dies wird über ein gesetztes Bit im Header des Blocks erreicht.

Tuple Identifier

- Eine andere Lösung ist die Verwendung von tuple identifiers (TIDs: System R): der zweite Teil der Adresse zeigt nicht auf die eigentliche Position im Block. Sondern auf eine Position in einer Tabelle im Header, in der die Postion angegeben wird. Innerhalb des Blockes muß so nicht nach dem Satz gesucht werden.



- Verwendung eines Dense Index (Teil 4 der VL).
  Die Adresse beziehen sich dann auf Indexeinträge, während die Sätze selbst frei beweglich bleiben.

→ alternative Dateiorganisationen, die es erlauben:

- Anfragen an große Datenmengen zu stellen,
  ohne dabei mehr als einen kleinen Teil der Datei zu durchsuchen.

- nicht zuviel Speicherplatz für die Organisation zu verwenden.

- Einfüge- und Löschoperationen nicht zu kompliziert werden zu lassen.