



Informations- und Wissensmanagement

Kapitel 2: Datenbankdefinitionssprachen

Erik Buchmann



Datenbank-Technologie – Vielfalt

[Einleitung](#)

[create table](#)

[Integritäts-
bed.](#)

[alter/
drop table](#)

[Index](#)

[ODL](#)

- Thema dieser Vorlesung (im wesentlichen):
Relationale Datenbanken –
zugrundeliegende Struktur sind Relationen.
- Es gibt auch andere Arten von Datenbanken,
z. B. objektorientierte Datenbanken.
 - Objekte anstelle von Tupeln.





Erzeugung von Relationen

[Einleitung](#)

[create table](#)

[Integritäts-
bed.](#)

[alter/
drop table](#)

[Index](#)

[ODL](#)

- Vor dem Einfügen von Daten in Datenbank müssen wir Relationen erzeugen.
- Erzeugung einer Relation
 - Festlegung des Schemas
 - Bestimmung der Integritätsbedingungen





Aspekte von SQL

[Einleitung](#)

[create table](#)

[Integritäts-
bed.](#)

[alter/
drop table](#)

[Index](#)

[ODL](#)

- **SQL** – standardisierte Sprache für den Datenbank-Zugriff (relationale Datenbanken). Mehrere Aspekte:
 - Schemadefinition,
 - Datenmanipulation (Einfügen, Löschen, Ändern),
 - Anfragen.
- Gegenstand dieses Kapitels: **Schemadefinition.**
→ damit verknüpft: **Integritätssicherung**





Relevante SQL-Befehle

[Einleitung](#)

[create table](#)

[Integritäts-
bed.](#)

[alter/
drop table](#)

[Index](#)

[ODL](#)

- **(create | drop | alter) table**
 - primary key
 - foreign key ... references ...
 - not null
 - check
- **(create | drop | alter) domain**
 - check
- **(create | drop | alter) index**



ANSI-SPARC-Architektur

Einleitung

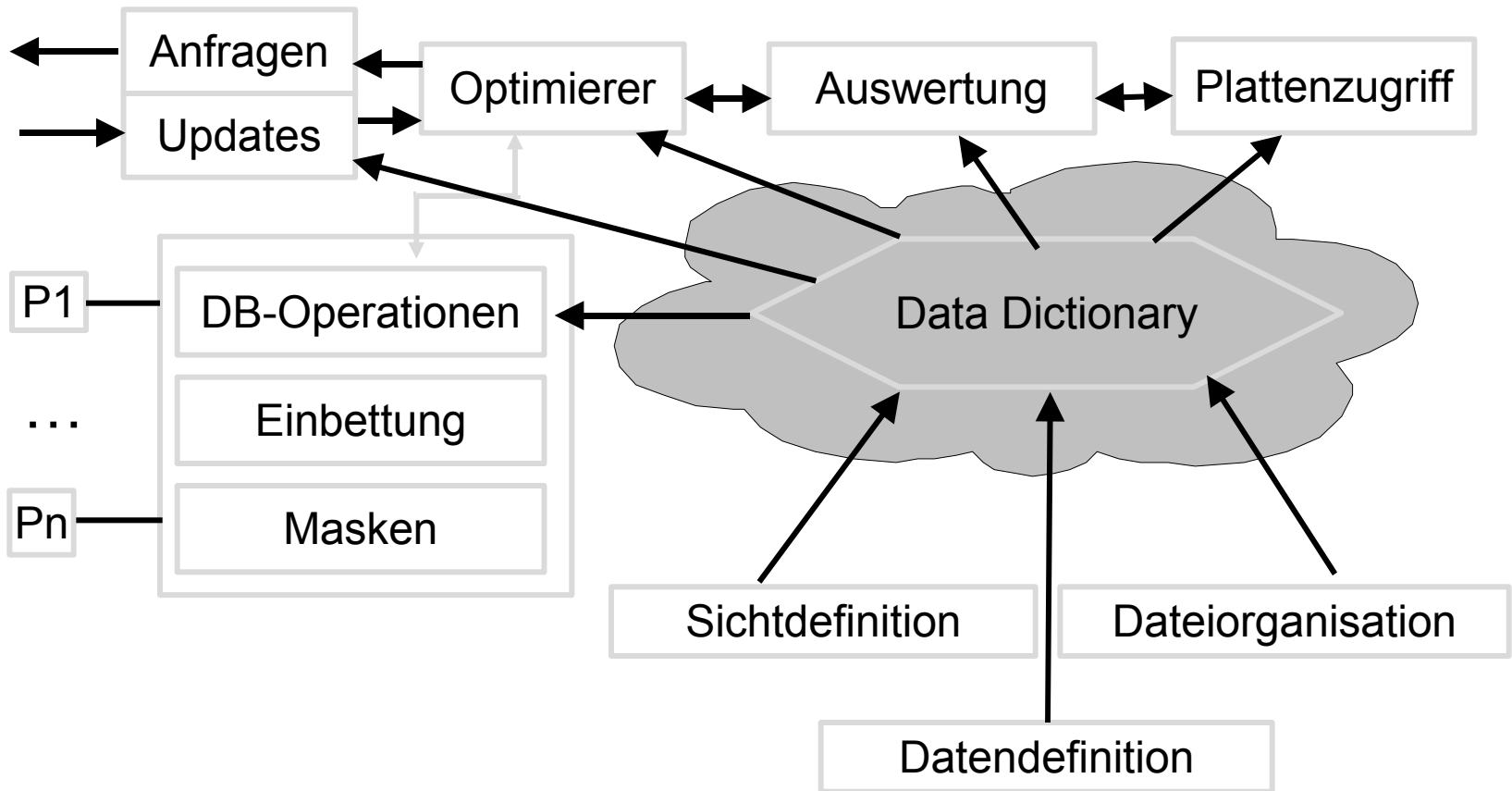
create table

Integritäts-
bed.

alter/
drop table

Index

ODL



Data Dictionary – Illustration

AUSLEIH	INV.NR	NAME

BUCH	INV.NR	NAME	TITEL	ISBN	AUTHOR

REL	RNR	NAME
	1	AUSLEIH
	2	BUCH

ATTR	RNR	NAME	TYP	SCHLÜSSEL
	1	INV.NR	int	true
	1	NAME	String	false
	2	INV.NR	int	true
	2	TITEL	String	false

[Einleitung](#)
[create table](#)
[Integritäts-
bed.](#)
[alter/
drop table](#)
[Index](#)
[ODL](#)



Datenbankdefinitionssprachen

[Einleitung](#)

[create table](#)

[Integritäts-
bed.](#)

[alter/
drop table](#)

[Index](#)

[ODL](#)

- **SQL-DDL**
Teil der Standardsprache
für relationale Datenbanksysteme: **SQL**.
(DDL = Data Definition Language)
- **ODL** (Object Definition Language)
für objektorientierte Datenbanksysteme
nach dem ODMG-Standard.





Relationales Modell: SQL-DDL

[Einleitung](#)

[create table](#)

[Integritäts-
bed.](#)

[alter/
drop table](#)

[Index](#)

[ODL](#)

- SQL-DDL umfaßt alle Klauseln von SQL zur Manipulation der Datendefinitionen
 - Typen
 - z.B. integer, decimal, varchar, text
 - Wertebereichen
 - Einschränkung der Wertebereiche existierender Typen
 - Typ + Wertebereich = Domain
 - Relationenschemata
 - create / drop / alter table
 - Integritätsbedingungen
 - check-Constraints, Fremdschlüsselbeziehungen, not null, etc.



Anforderungen an eine relationale DDL

[Einleitung](#)

[create table](#)

[Integritäts-
bed.](#)

[alter/
drop table](#)

[Index](#)

[ODL](#)

- Nach Codd 1982 Sprachmittel zur Definition von
 - Attributen, Wertebereichen,
 - Relationenschemata,
 - Primärschlüsseln, Fremdschlüsseln.
- Praxis SQL-89
 - Rel. Schemata mit Attributen und Wertebereichen.
- Ab SQL-92 vollständig
 - Schlüssel- und Fremdschlüsseldefinitionen möglich.
- Aktuelle Praxis
 - DDL von SQL-92 in allen größeren DBMS (Oracle, DB2, MS SQL Server) implementiert
 - kleinere DBMS (mysql, postgres) und Spezialsysteme (SQLite) kennen Teilmenge



Unterschiede zur relationalen Algebra

[Einleitung](#)

[create table](#)

[Integritäts-
bed.](#)

[alter/
drop table](#)

[Index](#)

[ODL](#)

- in SQL
 - NULL als Attributwert grundsätzlich zulässig.
 - Ebenso Duplikate, solange kein Konflikt mit Schlüsseldefinition.
→ Relation in SQL ist eine **Multimenge**.



SQL als Definitionssprache (1)

[Einleitung](#)

[create table](#)

[Integritäts-
bed.](#)

[alter/
drop table](#)

[Index](#)

[ODL](#)

- Externe Ebene
 - create view,
 - drop view;
- Konzeptuelle Ebene
 - create table,
 - alter table,
 - drop table;





SQL als Definitionssprache (2)

[Einleitung](#)

[create table](#)

[Integritäts-
bed.](#)

[alter/
drop table](#)

[Index](#)

[ODL](#)

- Konzeptuelle Ebene (erst in SQL-92)
 - create domain,
 - alter domain,
 - drop domain;
- Interne Ebene
 - create index,
 - alter index,
 - drop index.





Die Anweisung create table (1)

Einleitung

[create table](#)

Integritäts-
bed.

alter/
drop table

Index

ODL

- Syntax:
create table basisrelationenname (
spaltenname_1 wertebereich_1 [**not null**],
...
spaltenname_k wertebereich_k [**not null**])
- Führt zu
 - Eintrag in Data Dictionary,
 - Vorbereitung der „leeren Basisrelation“ in der Datenbank.
- Data Dictionary Zugriff mit Oracle beispielsweise:
 - **select * from** user_tables;
 - **select * from** user_tab_columns

Beispiel

X aterm

```
mysql> create table buecher (  
  -> ISBN char(10) not null,  
  -> Titel varchar(200),  
  -> Verlagsname varchar(30));  
Query OK, 0 rows affected (0.07 sec)
```

```
mysql> desc buecher;
```

Field	Type	Null	Key	Default	Extra
ISBN	char(10)	NO			
Titel	varchar(200)	YES		NULL	
Verlagsname	varchar(30)	YES		NULL	

```
3 rows in set (0.00 sec)
```

```
mysql> █
```



Die Anweisung create table (2)

Einleitung
[create table](#)
Integritäts-
bed.
alter/
drop table
Index
ODL

- **not null** schließt *Nullwerte* als Attributwerte in bestimmten Spalten aus.
 - zur Wiederholung: im DB-Kontext
 - *null* $\hat{=}$ keine Information,
 - *null* \neq 0
- **create table** Bücher (
ISBN *char(10)* **not null**,
Titel *varchar(200)*,
Verlagsname *varchar(30)*)



„not null“ - Constraint

```
X aterm
mysql> create table buecher (
  -> ISBN char(10) not null,
  -> Titel varchar(200),
  -> Verlagsname varchar(30));
Query OK, 0 rows affected (0.07 sec)

mysql> desc buecher;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| ISBN       | char(10)      | NO   |     |         |       |
| Titel      | varchar(200)  | YES  |     | NULL    |       |
| Verlagsname | varchar(30)   | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)

mysql> insert into buecher values (NULL, 'Datenbanksysteme', 'mitp');
ERROR 1048 (23000): Column 'ISBN' cannot be null
mysql>
```



„Philosophische“ Überlegungen dazu

- Warum solche Constraints in der Datenbank?
„Anwendung kann es doch auch sicherstellen.“
 - Datenintegrität!
 - mehrere Applikationen können auf einer DB arbeiten
 - DBMS soll Redundanzfreiheit nicht nur auf Daten- sondern auch auf Code-Ebene sicherstellen

Einleitung

[create table](#)

Integritäts-
bed.

alter/
drop table

Index

ODL



Erlaubte Wertebereiche in create table

Einleitung

[create table](#)

Integritäts-
bed.

alter/
drop table

Index

ODL

- **integer** (oder auch integer4, int) und **smallint** (oder auch integer2),
- **float(p)** (oder auch kurz float),
- **decimal(p,q)** und **numeric(p,q)**
- **character(n)** (oder kurz char(n), Strings fester Länge n,
- **varchar(n)** für Strings variabler Länge, maximal n Zeichen
- **bit(n)** für Bitfolgen,
- **date**, **time** bzw. **timestamp** für Datums-, Zeit- und kombinierte Datums-Zeit-Angaben.

jeweils p Stellen insgesamt, davon q nach dem Komma



Integritätsbedingungen (1)

Einleitung
create table
Integritäts-
bed.
alter/
drop table
Index
ODL

- Schlüssel kann aus beliebig vielen Attributen bestehen. „**Primattribute**“
- Entscheidung, welche Attribute Schlüssel bilden, ist anwendungsspezifisch.
- I.a. gibt es mehrere Schlüssel.
- **Primattribut**: ein Element eines Schlüssels.

Name	Vorname	Adresse
Erik	Buchmann	Im Nirgendwo 7
Mirco	Stern	Grube 3
Klemens	Böhm	Auf dem Holzweg 5
Markus	Bestehorn	Umgehungsstraße 42





Integritätsbedingungen (2)

Einleitung
create table
Integritäts-
bed.
alter/
drop table
Index
ODL

- INV.NR in AUSLEIH ist **Fremdschlüssel**.

AUSLEIH	INV.NR	NAME
	4711	Meyer
	1201	Schulz
	0007	Müller
	4712	Meyer

BUCH	INV.NR	TITEL	ISBN	AUTOR
	0007	Dr. No	3-324	Fleming
	1201	Objektbanken	3-111	Heuer
	4711	Datenbanken	3-345	Vossen
	4712	Datenbanken	3-345	Ullman
	4717	PASCAL	3-989	Wirth





SQL-89 Level 2 mit IEF

Einleitung
create table
Integritäts-
bed.
alter/
drop table
Index
ODL

- Zweite Stufe der SQL-89-Norm sieht Zusatz *IEF (Integrity Enhancement Feature)* vor.
- Definition von Schlüsseln und Fremdschlüsseln.



Beispiel Tabellendefinition mit IEF

Einleitung
create table
Integritäts-
bed.
alter/
drop table
Index
ODL

```
create table Bücher (  
    ISBN char(10) not null,  
    Titel varchar(200),  
    Verlagsname varchar(30),  
    primary key (ISBN),  
    foreign key (Verlagsname)  
        references Verlage (Verlagsname)  
)
```

ein oder
mehrere
Attribute

„Verlagsname“ in
Relation „Bücher“
verweist auf
„Verlagsname“ in
Relation „Verlage“

Attributname hier in
beiden Relationen
identisch, das muss
aber nicht so sein!



create table in SQL-92

Einleitung

create table

Integritäts-
bed.

alter/
drop table

Index

ODL

- **create table** Bücher (
 ISBN *char*(10),
 Titel *varchar*(200),
 Verlagsname *varchar*(30),
 primary key (ISBN),
 foreign key (Verlagsname)
 references Verlage (Verlagsname)
)
• **not null** implizit durch die **primary key**-Klausel.

Duplikate im Primärschlüssel

X aterm

```
mysql> create table buecher (  
  -> ISBN char(10) primary key,  
  -> Titel varchar(200),  
  -> Verlagsname varchar(30));  
Query OK, 0 rows affected (0.08 sec)  
  
mysql>  
mysql>  
mysql> insert into buecher values (1, 'Datenbanksysteme', 'mitp');  
Query OK, 1 row affected (0.00 sec)  
  
mysql> insert into buecher values (1, 'Database Systeme', 'AddisonWesley');  
ERROR 1062 (23000): Duplicate entry '1' for key 1  
mysql> █
```

Primärschlüssel ist NULL

```
X aterm
mysql> desc buecher;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key  | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| ISBN       | char(10)      | NO   | PRI  |          |       |
| Titel      | varchar(200)  | YES  |      | NULL    |       |
| Verlagsname | varchar(30)   | YES  |      | NULL    |       |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)

mysql> insert into buecher values (NULL, 'Database Systeme', 'AddisonWesley');
ERROR 1048 (23000): Column 'ISBN' cannot be null
mysql>
```

Fremdschlüssel nicht in der DB

```
X aterm
mysql> select * from verlag;
+-----+-----+
| Name          | Ansprechpartner |
+-----+-----+
| AddisonWesley | Herr Schmitt    |
| Fachbuchverlag | Frau Mustermann |
| Pearson       | Herr Schmitt    |
+-----+-----+
3 rows in set (0.00 sec)

mysql> insert into buecher values (1, 'Datenbanksysteme', 'mitp');
ERROR 1452 (23000): Cannot add or update a child row: a foreign key constraint fails (`example/buecher`, CONSTRAINT `buecher_ibfk_1` FOREIGN KEY (`Verlagsname`) REFERENCES `verlag` (`Name`))
mysql>
```



Erweiterungen in SQL-92

Einleitung
create table
Integritäts-
bed.
alter/
drop table
Index
ODL

Neben Primär- und Fremdschlüssel in SQL-92:

- **default**-Klausel: Defaultwerte für Attribute,
- **create domain**-Anweisung
benutzerdefinierte Wertebereiche,
- **check**-Klausel
weitere lokale Integritätsbedingungen
innerhalb der zu definierenden Wertebereiche,
Attribute und Relationenschemata.

Definition eines Wertebereichs

Einleitung
create table
Integritäts-
bed.
alter/
drop table
Index
ODL

```
create domain Gebiete varchar(20)
default 'Informatik'
create table Vorlesungen (
  V_Bezeichnung varchar(80) not null,
  SWS smallint,
  Semester smallint,
  Studiengang Gebiete)
create table Mitarbeiter (
  PANr integer not null,
  AngNr char(10) not null,
  Fachbereich Gebiete,
  Gehalt decimal(10,2),
  Raum integer,
  Einstellung date )
```



Integritätsbedingungen mit check (1)

Einleitung
create table
[Integritäts-
bed.](#)
alter/
drop table
Index
ODL

- **create domain** Gebiete *varchar(20)*
 default 'Informatik',
 check (
 value in ('Informatik',
 'Mathematik',
 'Elektrotechnik',
 'Linguistik')
)
)





Integritätsbedingungen mit check (2)

Einleitung
create table
[Integritäts-
bed.](#)
alter/
drop table
Index
ODL

```
create table Vorlesungen (  
    V_Bezeichnung varchar(80)  
        not null,  
        primary key,  
    SWS smallint,  
        check (SWS  $\geq$  0),  
    Semester smallint,  
        check (Semester between 1 and 9),  
    Studiengang Gebiete )
```

Integritätsbedingungen mit check (3)

Einleitung
create table
[Integritäts-
bed.](#)
alter/
drop table
Index
ODL

```
create table Buch_Versionen (  
ISBN char(10),  
Auflage smallint, check(Auflage > 0),  
Jahr integer,  
    check (Jahr between 1800 and 2020),  
Seiten integer, check(Seiten > 0),  
Preis decimal(8,2), check(Preis ≤ 250),  
    primary key (ISBN, Auflage),  
    foreign key (ISBN)  
        references Bücher (ISBN),  
    check ((select sum(Preis) from  
Buch_Versionen) <  
        (select sum(Budget) from Lehrstühle))
```

→ allgemeines check-Constraint, das sich nicht auf einzelne Attribute beschränkt



alter table in SQL-89

Einleitung

create table

Integritäts-
bed.

alter/
drop table

Index

ODL

- Syntax:
alter table basisrelationenname
 add spaltenname wertebereich
alter table Lehrstühle
 add Budget *decimal(8,2)*
- Wirkung:
 - Änderung des Relationenschemas im Data Dictionary.
Im Beispiel wird dem Relationenschema `Lehrstühle` ein neues Attribut zugeordnet.
 - Erweiterung der existierenden Basisrelation um ein Attribut, das bei jedem existierenden Tupel mit **null** besetzt wird.



alter table in SQL-92

Einleitung
create table
Integritäts-
bed.
[alter/](#)
[drop table](#)
Index
ODL

- Zusätzlich zu
`add spaltenname wertebereich`

auch Angabe von Default-Werten
und **check**-Klauseln erlaubt:

- `add Budget decimal(8,2) default 10000,
check (Budget > No_Planstellen * 1000)`
- `alter table bücher4
add constraint kbkey primary key (ISBN)`



alter- und drop-Klausel für Attribute (1)

Einleitung
create table
Integritäts-
bed.
[alter/](#)
[drop table](#)
Index
ODL

- Die Klausel

alter spaltenname default_änderung

nur Änderung der Defaultwerte,
nicht Änderung von Datentypen.



alter- und drop-Klausel für Attribute (2)

Einleitung
create table
Integritäts-
bed.
[alter/
drop table](#)
Index
ODL

- Die Klausel

`drop spaltenname { restrict | cascade }`
erlaubt Löschen von Attributen, falls

- keine Sichten und Integritätsbedingungen mit Hilfe dieses Attributs definiert wurden (im Fall **restrict**),
 - oder mit gleichzeitiger Löschung dieser Sichten und Integritätsbedingungen (im Fall **cascade**).
- Beispiel:
`alter table buch_versionen
drop column auflage`



Die Anweisung drop table

Einleitung
create table
Integritäts-
bed.
alter/
drop table
Index
ODL

- Syntax:
drop table basisrelationenname
 { **restrict** | **cascade** }
- Wirkung:
 - löschen des Relationenschemas aus dem Data Dictionary
 - **restrict** und **cascade** analog zum **drop** bei Attributen.





Index – Illustration (1)

Einleitung
create table
Integritäts-
bed.
alter/
drop table
Index
ODL

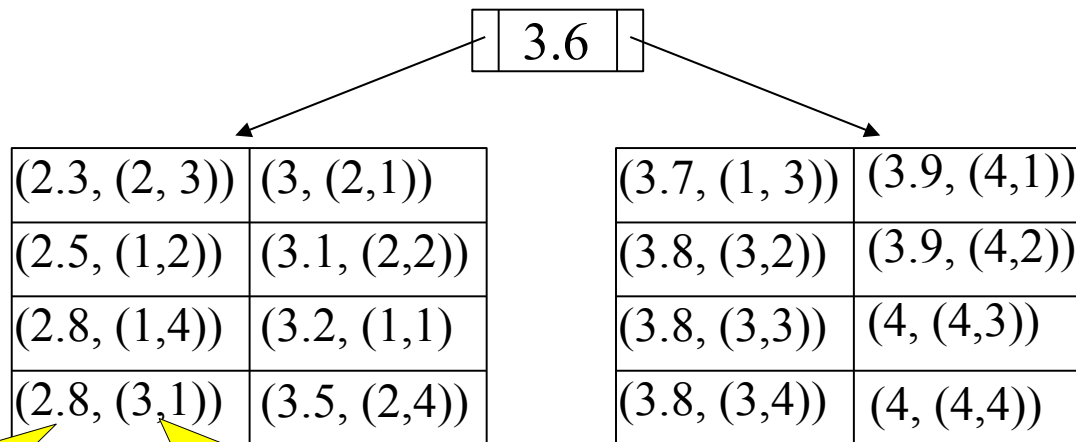
Erläuterung:

- Seitenweise Anordnung der Daten.
- Daten müssen im Hauptspeicher vorliegen, damit Selektion etc. durchgeführt werden kann.
- Seiten – Einheiten des Zugriffs.
- Laden einer Seite in den Hauptspeicher ist teuer, *Zugriffslücke*.



Index – Illustration (2)

- Student(name, age, gpa, major); t(Student) = 16.
- Non-clustered primary B+-tree für Attribut gpa.



Indexwert

Seite, Position

GPA: grade point average, MAJOR: Hauptfach

Tom, 20, 3.2, EE	Mary, 24, 3, ECE	Lam, 22, 2.8, ME	Chris, 22, 3.9, CS
Chang, 18, 2.5, CS	James, 24, 3.1, ME	Kathy, 18, 3.8, LS	Vera, 17, 3.9, EE
Bob, 21, 3.7, CS	Chad, 28, 2.3, LS	Kane, 19, 3.8, ME	Louis, 32, 4, LS
Pat, 19, 2.8, EE	Leila, 20, 3.5, LS	Martha, 29, 3.8, CS	Shideh, 16, 4, CS



Index – Erläuterungen

Einleitung

create table

Integritäts-
bed.

alter/
drop table

Index

ODL

- Index für mehrere Attribute möglich.
- Index für (gpa, name) nicht dasselbe wie für (name, gpa).
- Man kann Index nachträglich anlegen; man kann Index wieder löschen, ohne die Daten selbst zu löschen.
- Index ist Bestandteil der *physischen Ebene*. Index-Definition ist Bestandteil des *internen Schemas*.



Index – Illustration (3)

- Index für (gpa, name):

3.6

((2.3, Chad), (2, 3))	((3, Mary), (2, 1))
((2.5, Chang), (1, 2))	((3.1, James), (2, 2))
((2.8, Lam), (3, 1))	((3.2, Tom), (1, 1))
((2.8, Pat), (1, 4))	((3.5, Leila), (2, 4))

((3.7, Bob), (1, 3))	((3.9, Chris), (4, 1))
((3.8, Kane), (3, 3))	((3.9, Vera), (4, 2))
((3.8, Kathy), (3, 2))	((4, Louis), (4, 3))
((3.8, Martha), (3,4))	((4, Shideh), (4, 4))

Tom, 20, 3.2, EE
Chang, 18, 2.5, CS
Bob, 21, 3.7, CS
Pat, 19, 2.8, EE

Mary, 24, 3, ECE
James, 24, 3.1, ME
Chad, 28, 2.3, LS
Leila, 20, 3.5, LS

Lam, 22, 2.8, ME
Kathy, 18, 3.8, LS
Kane, 19, 3.8, ME
Martha, 29, 3.8, CS

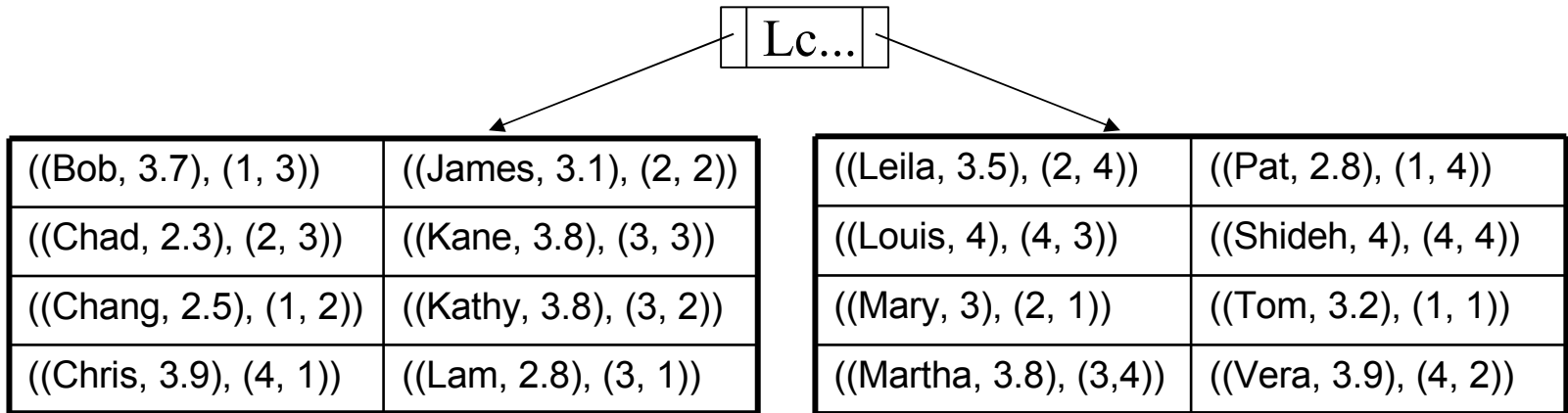
Chris, 22, 3.9, CS
Vera, 17, 3.9, EE
Louis, 32, 4, LS
Shideh, 16, 4, CS

- Suche nach gpa sowie nach gpa und name.
- Suche nach name.
(Ausgefeiltes DBMS würde es implementieren.)

Einleitung
create table
Integritäts-
bed.
alter/
drop table
Index
ODL

Index – Illustration (4)

- Index für (name, gpa):



Tom, 20, 3.2, EE	Mary, 24, 3, ECE	Lam, 22, 2.8, ME	Chris, 22, 3.9, CS
Chang, 18, 2.5, CS	James, 24, 3.1, ME	Kathy, 18, 3.8, LS	Vera, 17, 3.9, EE
Bob, 21, 3.7, CS	Chad, 28, 2.3, LS	Kane, 19, 3.8, ME	Louis, 32, 4, LS
Pat, 19, 2.8, EE	Leila, 20, 3.5, LS	Martha, 29, 3.8, CS	Shideh, 16, 4, CS



Indexe und physische Datenunabhängigkeit – Beispiel (1)

Einleitung

create table

Integritäts-
bed.

alter/
drop table

Index

ODL

- Anfrage
select name **from** Student **where** gpa > 4.0
- Anfrage liefert Ergebnis,
unabhängig davon, ob jener Index existiert
oder nicht – physische Datenunabhängigkeit.
- Wenn Index existiert
 - erkennt DBMS das
und nutzt ihn für die Anfrage-Evaluierung,
 - erhebliche Beschleunigung.



Indexe und physische Datenunabhängigkeit – Beispiel (2)

Einleitung

create table

Integritäts-
bed.

alter/
drop table

Index

ODL

- Anfrage
select name **from** Student
where gpa > 4.2 **and** age=27
- Ohne Index: Möglicherweise effizienter, erst das age-Prädikat auszuwerten; unklar.
- Mit Index: I. d. R. erst gpa-Prädikat auswerten.
- Datenbank findet in beiden Fällen überlegene Alternative – physische Datenunabhängigkeit.



Die Anweisung create index (1)

Einleitung
create table
Integritäts-
bed.
alter/
drop table
Index
ODL

- SQL-89: Bestandteil der Norm

```
create [unique] index indexname  
      on basisrelationenname (  
      spaltenname_1 ordnung_1, ...,  
      spaltenname_k ordnung_k  
      )
```

- Ordnung: **ascending** oder **descending**
- Beispiel:
CREATE INDEX *typ* **ON** auto (hersteller,
modell, baujahr)
 - Reihenfolge der Attribute → Sortierreihenfolge.
 - Index *typ* hilft uns bei Suche nach Herstellern,
aber nur bedingt bei Suche gemäß Baujahr.



Die Anweisung create index (2)

Einleitung

create table

Integritäts-
bed.

alter/
drop table

Index

ODL

- **CREATE UNIQUE INDEX** `typ ON auto`
(`hersteller, modell, baujahr`)
 - ‚unique‘ – keine identischen Schlüssel im Index, d.h. keine zwei Tupel mit identischem Hersteller, Modell und Baujahr erlaubt
 - zusätzlicher Constraint für Basisrelation



Simulierte Schlüsselbedingung

Einleitung

create table

Integritäts-
bed.

alter/
drop table

Index

ODL

```
create table Bücher (  
    ISBN char(10) not null,  
    Titel varchar(200),  
    Verlagsname varchar(30) )
```

```
create unique index Buchindex  
on Bücher  
    (ISBN asc)
```

- Was passiert, wenn ich Index nachträglich anlege, das entsprechende Attribut **unique**-Eigenschaft aber nicht erfüllt?



ODL (Object Definition Language)

[Einleitung](#)

[create table](#)

[Integritäts-
bed.](#)

[alter/
drop table](#)

[Index](#)

[ODL](#)

- ODL Teil des ODMG-Standards (Object Data Management Group)
- wird hier als Beispiel für Schemadefinition im Objektmodell gebraucht
- ODL findet z.B. in ODBMS wie *MATISSE*, *Objectivity/DB* oder *Orient* Verwendung





Objektorientiertes Modell: ODL

[Einleitung](#)

[create table](#)

[Integritäts-
bed.](#)

[alter/
drop table](#)

[Index](#)

[ODL](#)

```
interface Student : Person (  
    extent Studenten, key matrnr)  
    attribute char matrnr[6];  
    attribute string studienfach;  
    attribute set<struct<float note, string fach>>  
                                                zeugnis;  
  
    relationship Person mutter inverse Person::kind;  
  
    relationship Person vater inverse Person::kind;  
  
    float durchschnittsnote ()  
        raises (keine_note);  
    void exmatrikulation (in string art)  
        raises (buecher_ausgeliehen);  
}
```



Erläuterung ODL-Beispiel (1)

Einleitung
create table
Integritäts-
bed.
alter/
drop table
Index
ODL

Schnittstelle zum Objekttyp `Personen` beschreibt

- *Typhierarchie*: Angabe der Obertypen hinter dem Typnamen (hier: Obertyp `Person`),
- *Extension*, in der die aktuell erzeugten Objekte vom Typ `Student` gesammelt werden sollen (hier: Extension mit dem Namen `Studenten`),
- *Schlüssel* des Objekttyps, eine Auswahl der Attribute, die zur eindeutigen Identifizierung der Objekte unabhängig von der Objektidentität verwendet werden können (hier: nur das Attribut `matrnr`),
- *Attribute* mit Datentypen und Namen.



Erläuterung ODL-Beispiel (2)

Einleitung
create table
Integritäts-
bed.
alter/
drop table
Index
ODL

- *Beziehungen* zu anderen Klassen mit dem Wortsymbol **relationship**
 - auch inverse Beziehungen,
 - ermöglichen Wahl zwischen 1:1-, 1:n, und n:m-Kardinalitäten,
 - hier: zwei 1:n-Beziehungen `Vater` und `Mutter` zwischen `Studenten` und `Personen`, da nur die Rückrichtung einen `Set`-Typ enthält: `set<Person>kind`.
- *Methoden* mit ihrer Schnittstelle und einer spezifizierten Ausnahmebehandlung, die im Fehlerfall ausgelöst wird, etwa bei Verletzung von Integritätsbedingungen.