

Datenbanken

Einführung/Repetition



Single/Multiuser

Unterschiede

Single-User Database	Multi-User Database
SQLite	Oracle DBMS
MS Access	MS Sql Server
DBase	Mysql / MariaDB
	MongoDB
	Neo4J



Datenbanksystem Anforderungen

Merkmale eines Datenbanksystems

- **Datenunabhängigkeit**
- **Effizienter Speicherzugriff**
- **Paralleler Datenzugriff**
- **Datenkonsistenz**
- **Gemeinsame Datenbasis**
- **Datenintegrität**
- **Datensicherheit**
- **Wiederherstellungsverfahren**
- **Abfragesprache**
- **Keine / kontrollierte Redundanz**

ACID-Prinzip

Atomicity (Atomarität)

Consistency (Konsistenz)

Isolation (Isolation)

Durability (Dauerhaftigkeit)

Die Theorie hinter dem ACID-Prinzip

Atomacy

Man spricht dann von atomaren (atomacy) Operationen, wenn eine Sequenz von Datei-Operationen entweder ganz oder gar nicht ausgeführt wird.

Die Theorie hinter dem ACID-Prinzip

Atomacy

Man spricht dann von atomaren (atomacy) Operationen, wenn eine Sequenz von Datei-Operationen entweder ganz oder gar nicht ausgeführt wird.

Consistency

Man spricht von einer vorhandenen Datenkonsistenz (consistency), wenn nach einer Sequenz von Datei-Operationen der Datenzustand in einem konsistenten Zustand hinterlassen wird.

Die Theorie hinter dem ACID-Prinzip

Atomacy	Man spricht dann von atomaren (atomacy) Operationen, wenn eine Sequenz von Datei-Operationen entweder ganz oder gar nicht ausgeführt wird.
Consistency	Man spricht von einer vorhandenen Datenkonsistenz (consistency), wenn nach einer Sequenz von Datei-Operationen der Datenzustand in einem konsistenten Zustand hinterlassen wird.
Isolation	Die Isolation (isolation) verhindert, dass sich parallele Ausführungen auf befindliche Datei-Operationen gegenseitig beeinflussen können.

Die Theorie hinter dem ACID-Prinzip

Atomacy	Man spricht dann von atomaren (atomacy) Operationen, wenn eine Sequenz von Datei-Operationen entweder ganz oder gar nicht ausgeführt wird.
Consistency	Man spricht von einer vorhandenen Datenkonsistenz (consistency), wenn nach einer Sequenz von Datei-Operationen der Datenzustand in einem konsistenten Zustand hinterlassen wird.
Isolation	Die Isolation (isolation) verhindert, dass sich parallele Ausführungen auf befindliche Datei-Operationen gegenseitig beeinflussen können.
Durability	Die Dauerhaftigkeit (durability) gewährleistet, dass die Datei-Operationen dauerhaft auf einem Datenträger gesichert sind.



Atomacity

Eine Transaktion ist eine Folge von Datenbank-Operationen, die entweder ganz oder gar nicht ausgeführt wird.

```
START TRANSACTION;  
INSERT INTO konto (id, inhaber_id, betrag)  
VALUES (1, "4711", 50);  
INSERT INTO konto (id, inhaber_id, betrag):  
VALUES (1, "4711", -50);  
COMMIT;
```



Consistency

In Datenbanken werden Regeln festgelegt, wie die Daten strukturiert sein müssen.

```
START TRANSACTION;  
INSERT INTO konto (id, inhaber_id, betrag)  
VALUES (1, "FALSCH", 50);  
COMMIT;
```



Isolation

Oft werden viele Transaktionen gleichzeitig durchgeführt; Datenbanken müssen jedoch gewährleisten, dass die Transaktionen voneinander **isoliert** sind; das heißt, das Endergebnis muss das gleiche sein, als ob die Transaktionen hintereinander durchgeführt würden.

```
START TRANSACTION;  
INSERT INTO konto (id, inhaber_id, betrag)  
VALUES (1, "4711", 50);  
INSERT INTO konto (id, inhaber_id, betrag):  
VALUES (1, "4711", -50);  
COMMIT;
```

```
START TRANSACTION;  
INSERT INTO konto (id, inhaber_id, betrag)  
VALUES (1, "4711", 50);  
INSERT INTO konto (id, inhaber_id, betrag):  
VALUES (1, "4711", -50);  
COMMIT;
```



Durability

Datenbanken sollten vor Stromausfällen sicher sein. Das heißt, sie sind zwar während eines Stromausfalls nicht erreichbar, aber es gehen keine gespeicherten Daten verloren; dazu müssen die Daten auf einer Festplatte gespeichert werden, und diese Daten müssen immer *konsistent* sein.

1 Begin Transaction

Author
author_id : 1
author_name : 'eve'

2 Begin Transaction

3 UPDATE Author
SET author_name = 'peter'
WHERE author_id = 1

Author
author_id : 1
author_name : 'peter'

Dirty Read

Author
author_id : 1
author_name : 'peter'

4 SELECT * FROM author
WHERE author_id = 1

5 Rollback

Author
author_id : 1
author_name : 'eve'

1 Begin Transaction

Author
author_id : 1
author_name : 'eve'

2 Begin Transaction

Author
author_id : 1
author_name : 'eve'

3
SELECT * FROM author
WHERE author_id = 1

4
UPDATE author
SET author_name = 'peter'
WHERE author_id = 1

Author
author_id : 1
author_name : 'peter'

5
SELECT * FROM author
WHERE author_id = 1

Non repeatable Read

6 Commit

Author
author_id : 1
author_name : 'peter'

1 Begin Transaction

Author
author_id : 1
author_name : 'eve'

2 Begin Transaction

3
UPDATE author
SET author_name = 'peter'
WHERE author_id = 1

Author
author_id : 1
author_name : 'peter'

4

UPDATE author
SET author_name = 'alice'
WHERE author_id = 1

Author
author_id : 1
author_name : 'alice'

5 Commit

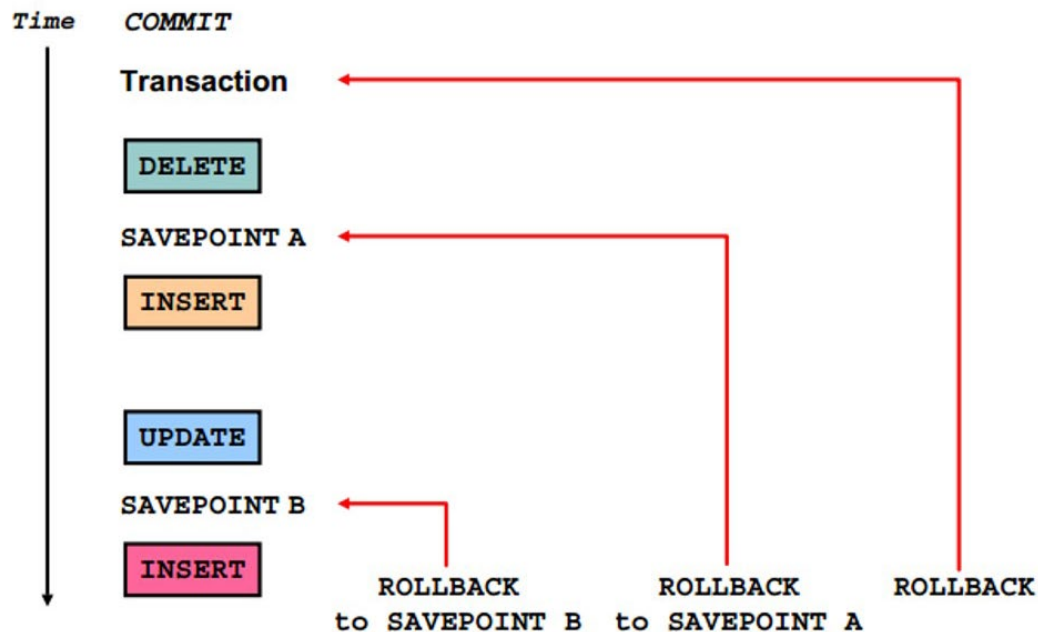
Lost Update

Author
author_id : 1
author_name : 'peter'

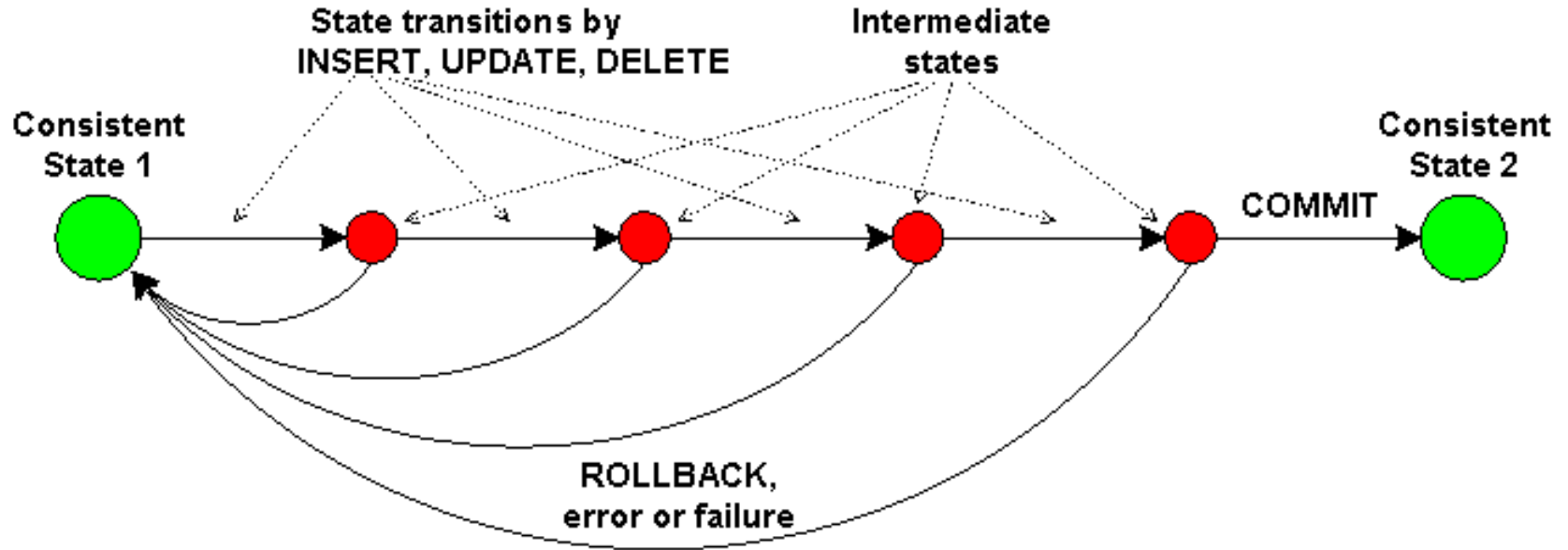
6 Commit

Author
author_id : 1
author_name : 'alice'

Savepoints



Commit und Rollback



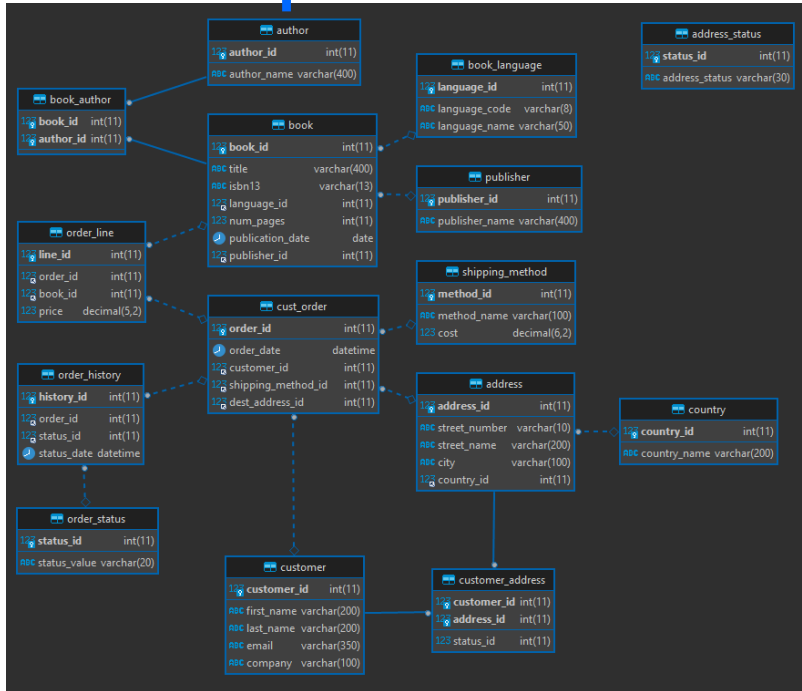
Datenberechtigung / Trennung





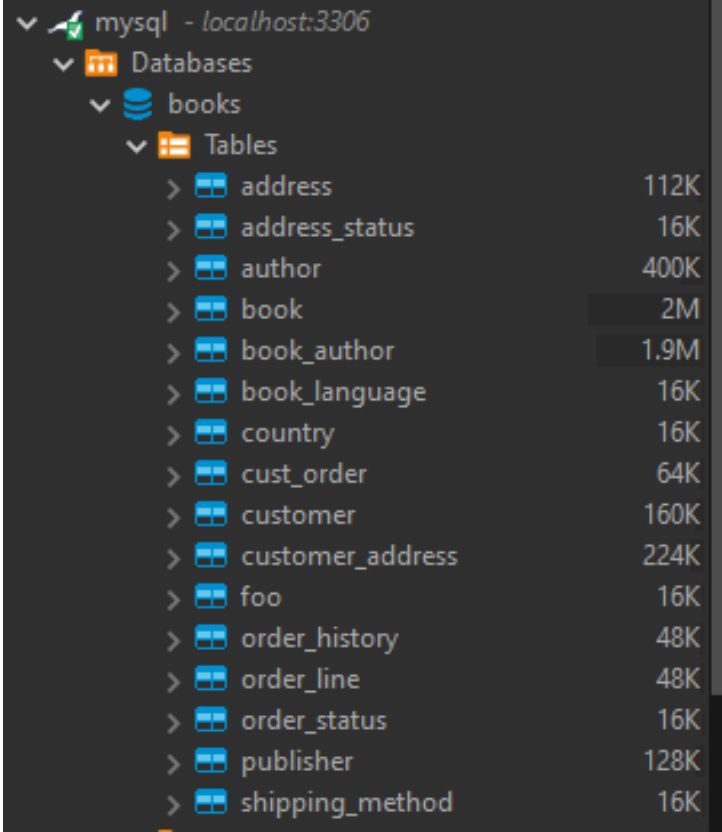
Database Schema

There are two main database schema types that define different parts of the schema: logical and physical.



here you see a logical schema

Physical Schema



mysql - localhost:3306	
Databases	
books	
Tables	
address	112K
address_status	16K
author	400K
book	2M
book_author	1.9M
book_language	16K
country	16K
cust_order	64K
customer	160K
customer_address	224K
foo	16K
order_history	48K
order_line	48K
order_status	16K
publisher	128K
shipping_method	16K

Views

Views can provide advantages over tables:

Views can represent a subset of the data contained in a table. Consequently, a view can limit the degree of exposure of the underlying tables to the outer world: a given user may have permission to query the view, while denied access to the rest of the base table.

Views can join and simplify multiple tables into a single virtual table.

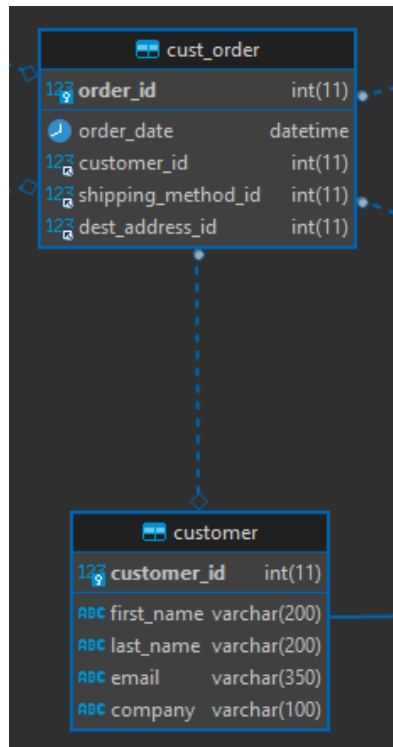
Views can act as aggregated tables, where the database engine aggregates data (sum, average, etc.) and presents the calculated results as part of the data.

Views can hide the complexity of data. For example, a view could appear as Sales2000 or Sales2001, transparently partitioning the actual underlying table.

Views take very little space to store; the database contains only the definition of a view, not a copy of all the data that it presents.

Depending on the SQL engine used, views can provide extra security.

View example



```
CREATE OR REPLACE
VIEW v_custorders AS
SELECT
    c.first_name,
    c.last_name,
    count(*) AS orders
FROM
    customer c,
    cust_order co
WHERE
    co.customer_id = c.customer_id
GROUP BY
    c.customer_id ;
```

Vertical Slicing

FIRMA A

customer	
123	customer_id int(11)
ABC	first_name varchar(200)
ABC	last_name varchar(200)
ABC	email varchar(350)

customer_address	
123	customer_id int(11)
123	address_id int(11)
123	status_id int(11)

address	
123	address_id int(11)
ABC	street_number varchar(10)
ABC	street_name varchar(200)
ABC	city varchar(100)
123	country_id int(11)

FIRMA B

customer	
123	customer_id int(11)
ABC	first_name varchar(200)
ABC	last_name varchar(200)
ABC	email varchar(350)

customer_address	
123	customer_id int(11)
123	address_id int(11)
123	status_id int(11)


address	
123	address_id int(11)
ABC	street_number varchar(10)
ABC	street_name varchar(200)
ABC	city varchar(100)
123	country_id int(11)

FIRMA C

customer	
123	customer_id int(11)
ABC	first_name varchar(200)
ABC	last_name varchar(200)
ABC	email varchar(350)

customer_address	
123	customer_id int(11)
123	address_id int(11)
123	status_id int(11)

address	
123	address_id int(11)
ABC	street_number varchar(10)
ABC	street_name varchar(200)
ABC	city varchar(100)
123	country_id int(11)



	customer_id	first_name	last_name	email	company
1	1	Ursola	Purdy	upurdy0@cdbaby.com	Firma A
2	2	Ruthanne	Vatini	rvatini1@fema.gov	Firma A
3	3	Reidar	Turbitt	rturbitt2@geocities.jp	Firma A
4	4	Rich	Kirsz	rkirsz3@jalum.net	Firma A
5	5	Carline	Kupis	ckupis4@tamu.edu	Firma A
6	6	Kandy	Adamec	kadamec5@weather.com	Firma A
7	7	Jermain	Giraudau	jgiraudau6@elpais.com	Firma A
8	8	Nolly	Bonicelli	nbonicelli7@examiner.com	Firma A
9	9	Phebe	Curdell	pcurdell8@usa.gov	Firma A
10	10	Euell	Guilder	eguilder9@theforest.net	Firma A
11	11	Teriann	Marritt	tmarritta@va.gov	Firma B
12	12	Filmer	Douse	fdouseb@foxnews.com	Firma B
13	13	Daisey	Lamball	dlamballc@skyrock.com	Firma B
14	14	Gusella	Quogan	gquogand@whitehouse.gov	Firma B
15	15	Lonnie	Cambden	lcambdene@gmpg.org	Firma B
16	16	Debbi	Huyghe	dhuyghef@dot.gov	Firma B
17	17	Ignace	Fursey	ifurseyg@hatena.ne.jp	Firma B
18	18	Andrei	Jefferson	ajeffersonh@live.com	Firma B
19	19	Sanford	Gillbe	sgillbei@telegraph.co.uk	Firma B
20	20	Kali	Sedgbeer	ksedgbeerj@bbc.co.uk	Firma B
21	21	Krishnah	Traite	kttraitek@state.gov	Firma C
22	22	Alley	Selbie	aselbiel@moonfruit.com	Firma C
23	23	Gilligan	Betteson	gbettesonm@paypal.com	Firma C
24	24	Raul	Pentelow	rpentelown@zimbio.com	Firma C
25	25	Garrek	Emnoney	gemnoneyo@nyu.edu	Firma C
26	26	Mathilde	Kleanthous	mkleanthousp@tamu.edu	Firma C
27	27	Dacy	Mabe	dmabeq@cloudflare.com	Firma C
28	28	Rob	Handes	rhandesr@arstechnica.com	Firma C
29	29	Rafaello	Boniface	rbonifaces@marriott.com	Firma C
30	30	Matthiew	Donizeau	mdonizeaut@rakuten.co.jp	Firma C