

Grundlagen der Datenbanktechnik

Martin Klier

Klug GmbH integrierte Systeme,
Teunz

22.02.2010

- Martin Klier, 30
- Datenbankadministrator für **ORACLE**
- Fachliche Schwerpunkte:
 - Performanceoptimierung / Tuning
 - hochverfügbare Systeme
 - Cluster und Replikation
- Linux seit 1997
- Oracle Database seit 2003
- Kontakt: martin.klier@unix.net
- Weblog: <http://www.usn-it.de>

- Klug GmbH integrierte Systeme
Lindenweg 13
92552 Teunz
- Führendes und erfolgreiches Unternehmen der Software- und Steuerungssysteme für die Intralogistik
- Konzeption, Beratung, Softwareerstellung, Elektronik, Kommissioniertechnik, Hardware, ...
- 210 Mitarbeiter
- 50 Auszubildende



- Vorstellung
- Geschichte
- Aufbau und Architektur
- Relationen
- Interne Abläufe
 - Cache/Absicherung
 - Transaktionsprinzip
 - SELECT
 - UPDATE
- Zusammenfassung

Was ist eine Datenbank?

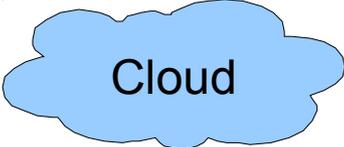


Was ist eine Datenbank?

*„Die wesentliche Aufgabe eines Datenbanksystems ist es, große Datenmengen effizient, widerspruchsfrei und dauerhaft zu speichern und benötigte Teilmengen in unterschiedlichen, bedarfsgerechten Darstellungsformen für Benutzer und Anwendungsprogramme bereitzustellen.“
(Wikipedia)*

- Vorstellung
- Geschichte
- Aufbau und Architektur
- Relationen
- Interne Abläufe
 - Cache/Absicherung
 - Transaktionsprinzip
 - SELECT
 - UPDATE
- Zusammenfassung

- 1970 Theoretische Grundlagen von Edgar F. Codd (IBM Almaden)
- 1974 Ingres RDBMS (QUEL)
- 1977 IBM System R
Sonderanfertigung für Pratt&Whitney (SEQUEL) 
- 1977 SDL (Vorläufer von Oracle) erstellt SEQUEL-RDMBS für C.I.A.
- 1979 RSI (ex SDL) bringt „Oracle RDBMS 2.0“ auf den Markt (SQL)
- 1984 Erstes RDMBS mit Lesekonsistenz

- 1987 RDBM-Systeme für UNIX kommen auf den Markt
- 1988 Prozedurale Programmierung möglich (PL/SQL)
- 1995 64bit-RDBMS erscheinen
- 1997 Datenbanken werden für Internetapplikationen optimiert
- 1999 XML-Unterstützung 
- 2003 Datenbank-Server-Cluster mit voller Cachekohärenz
- 2005 Ableger der großen Enterprise-RDBMS kostenlos verfügbar
- 2008 „Selbstverwaltende Datenbank“ macht Fortschritte
- 2010 Database  Cloud

- Vorstellung
- Geschichte
- Aufbau und Architektur
- Relationen
- Interne Abläufe
 - Cache/Absicherung
 - Transaktionsprinzip
 - SELECT
 - UPDATE
- Zusammenfassung

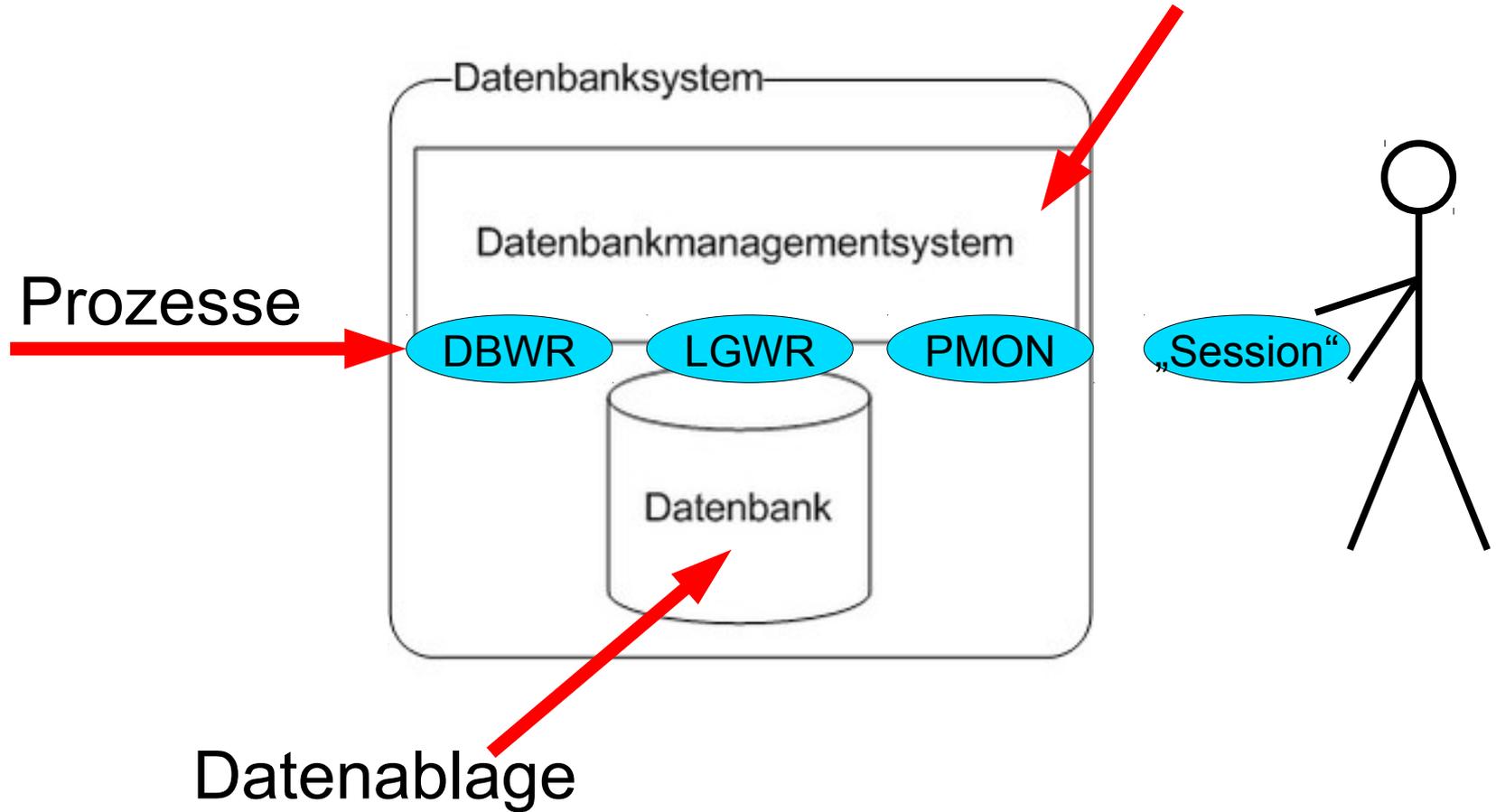
Thanks to Donald K. Burleson



Oracle Internals

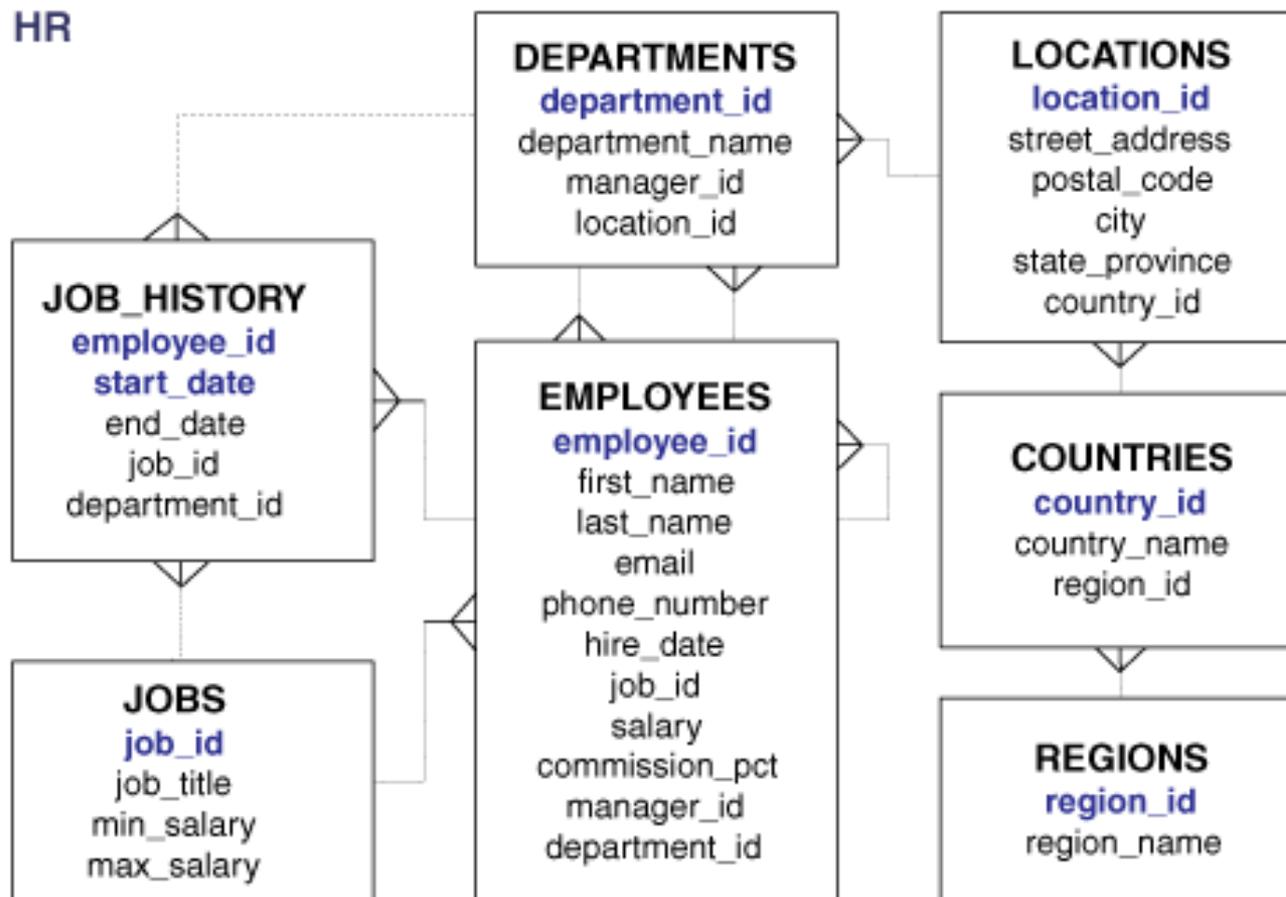
Things look different from the inside

Speicherresidente Teile, SGA



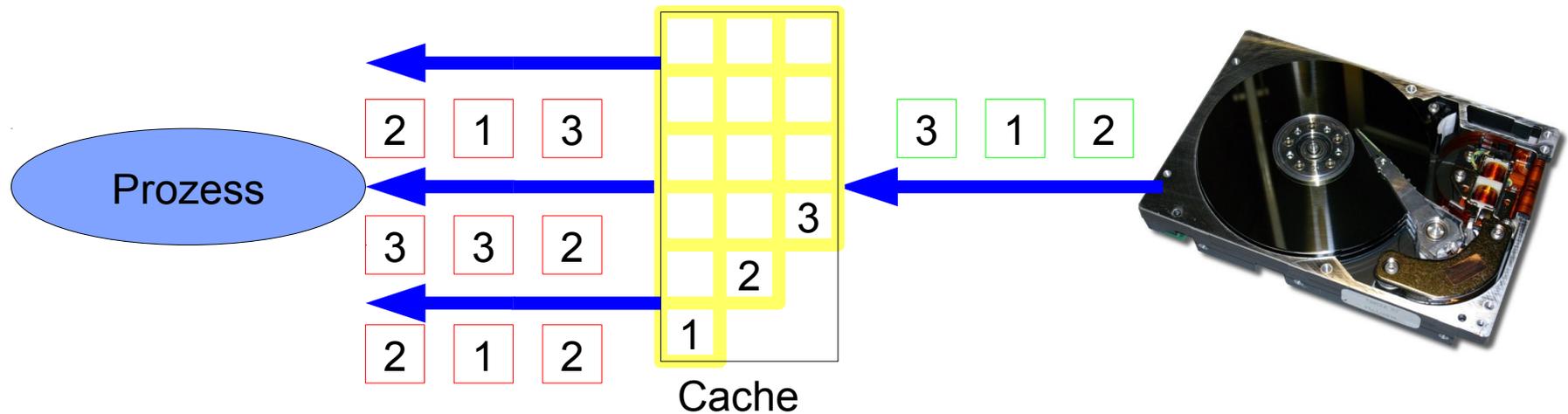
- Vorstellung
- Geschichte
- Aufbau und Architektur
- Relationen
- Interne Abläufe
 - Cache/Absicherung
 - Transaktionsprinzip
 - SELECT
 - UPDATE
- Zusammenfassung

ER-Diagramm



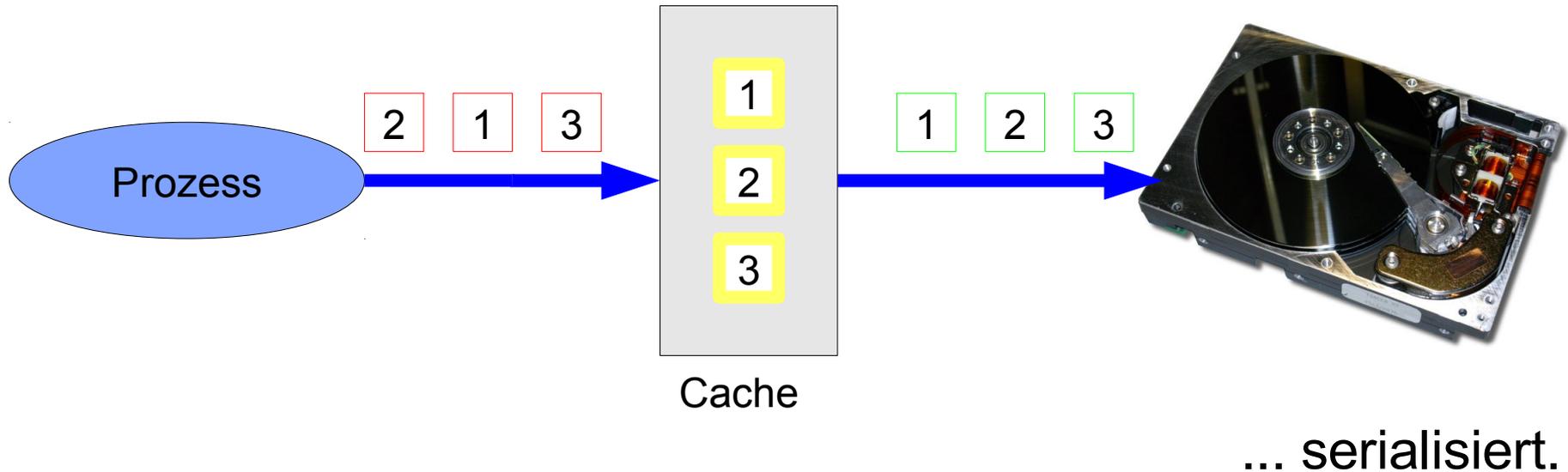
- Vorstellung
- Geschichte
- Aufbau und Architektur
- Relationen
- Interne Abläufe
 - Cache/Absicherung
 - Transaktionsprinzip
 - SELECT
 - UPDATE
- Zusammenfassung

Lese-Cache (RAM nach Disk) ...



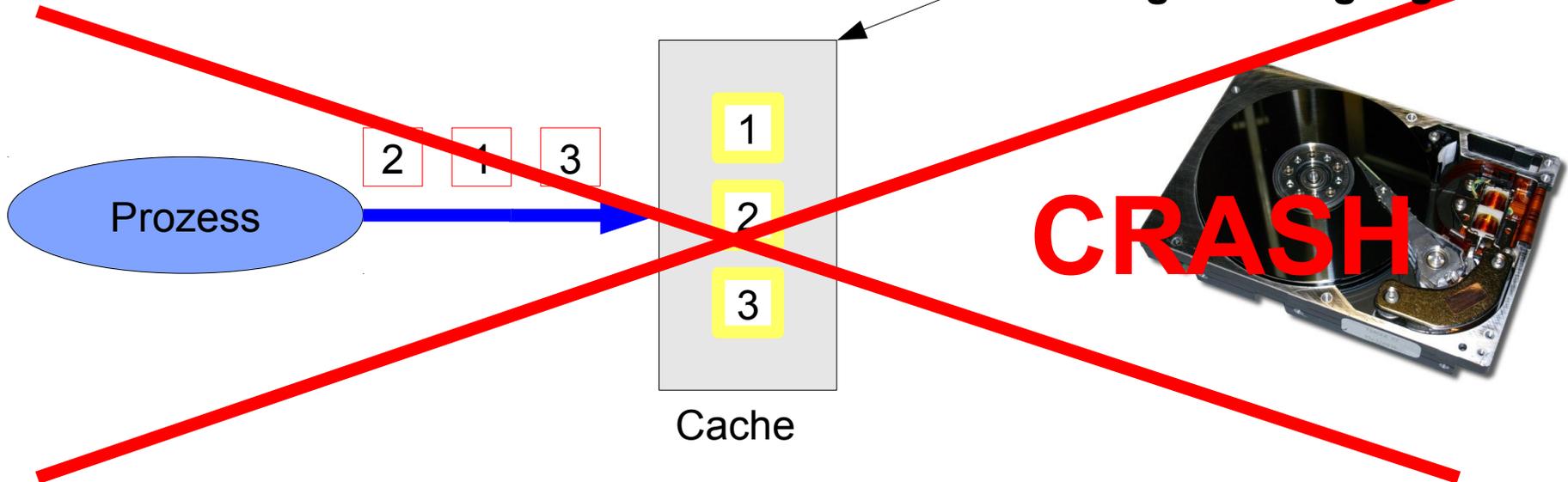
... macht Lesezugriff zu RAM-Zugriff.

Schreib-Cache (RAM vor Disk) ...

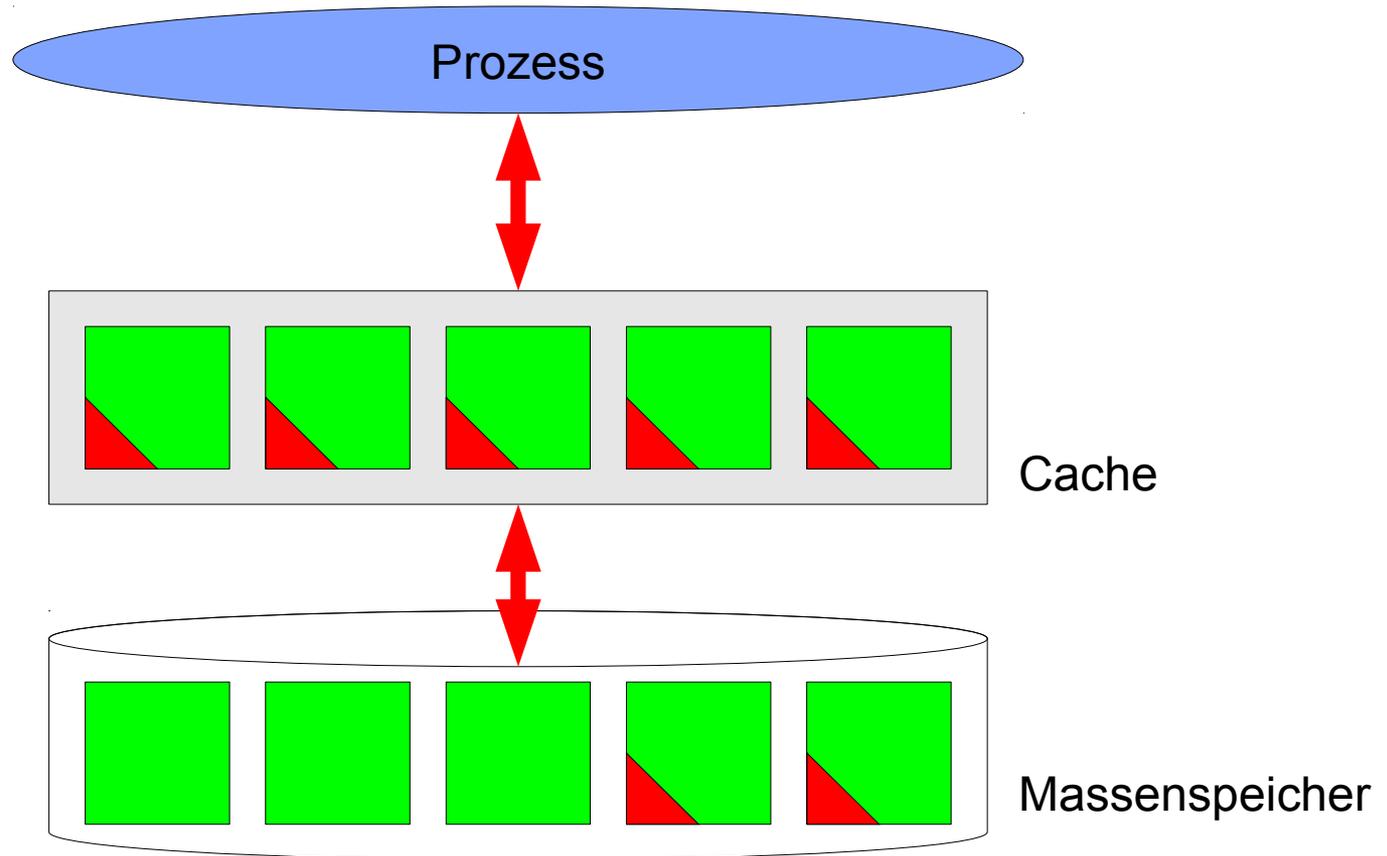


Risiko Schreib-Cache!

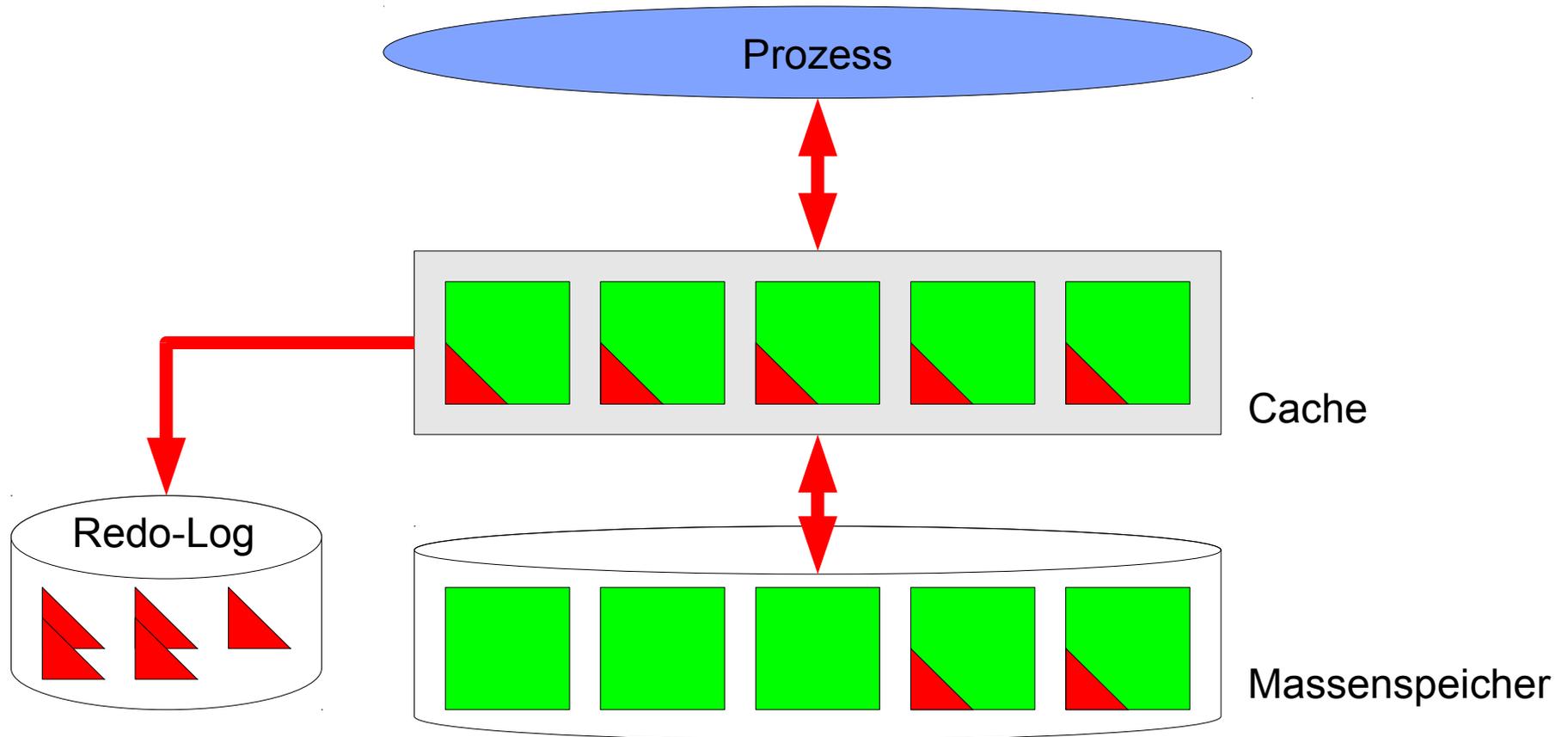
? Energieversorgung ?

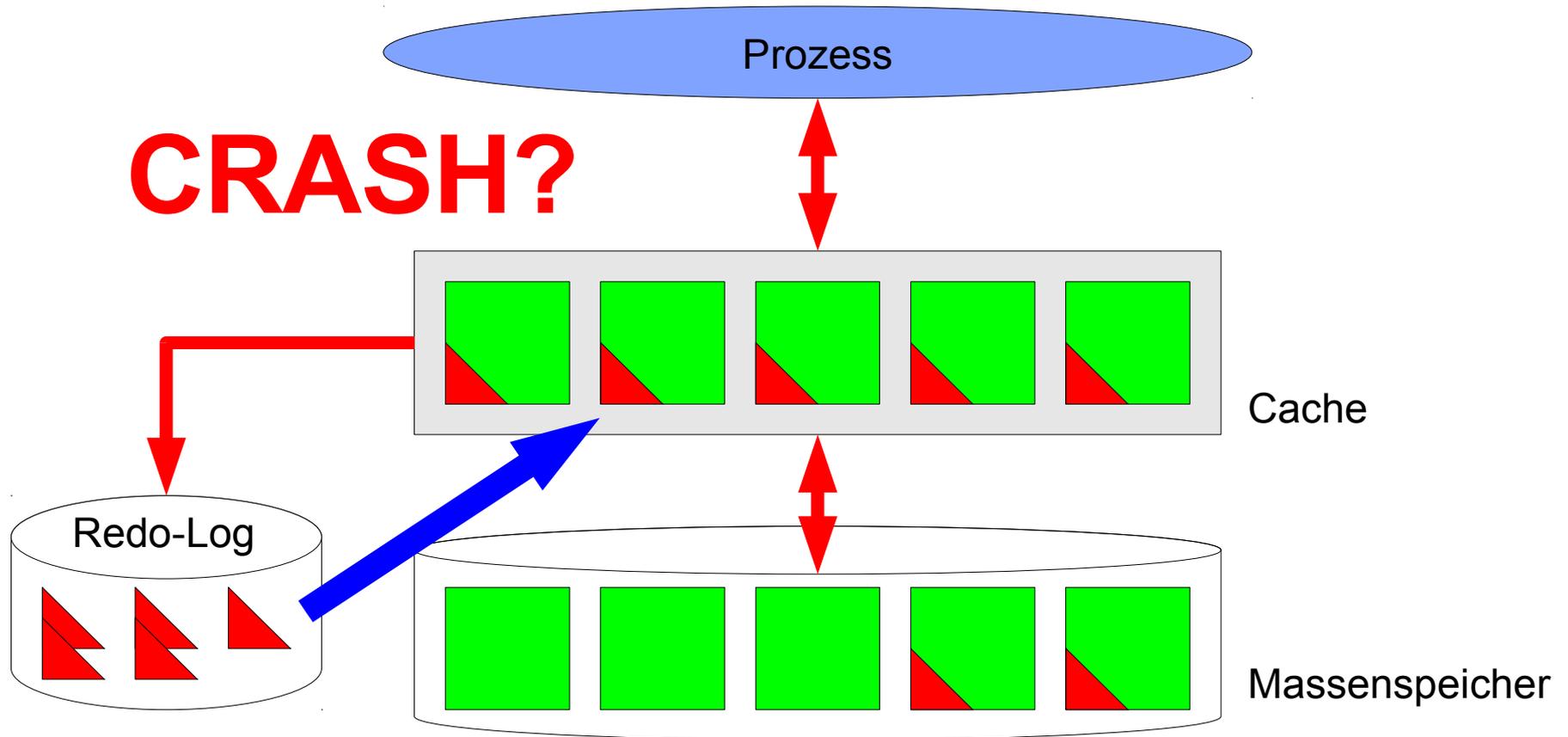


- Cache beschleunigt Zugriffe
- Aber: RAM => Totalverlust bei Spannungsausfall
- Cache erfordert Absicherung!
- Kompromiss notwendig
- Schreiben von „Delta“-Werten auf schnelle HDD



Cache / Absicherung



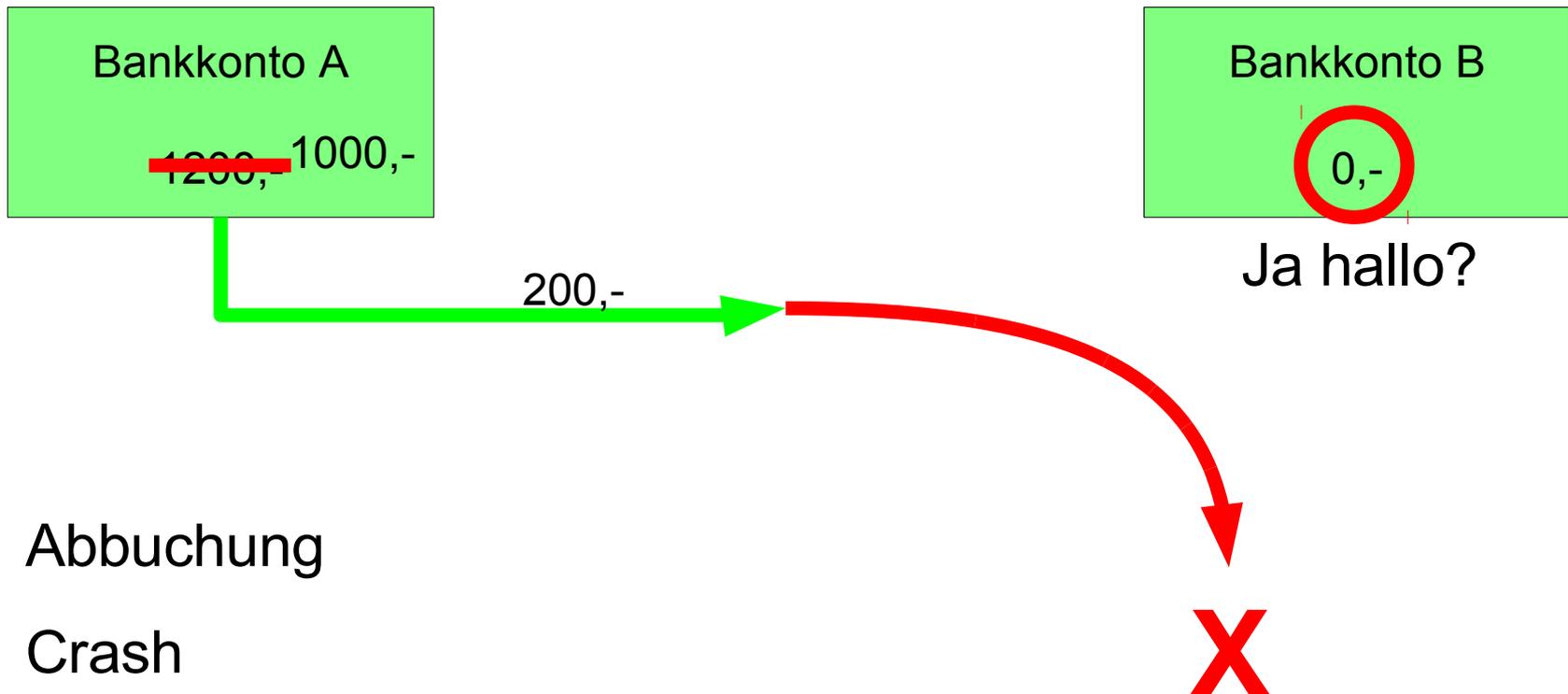


Recovery!

- Vorstellung
- Geschichte
- Aufbau und Architektur
- Relationen
- Interne Abläufe
 - Cache/Absicherung
 - Transaktionsprinzip
 - SELECT
 - UPDATE
- Zusammenfassung

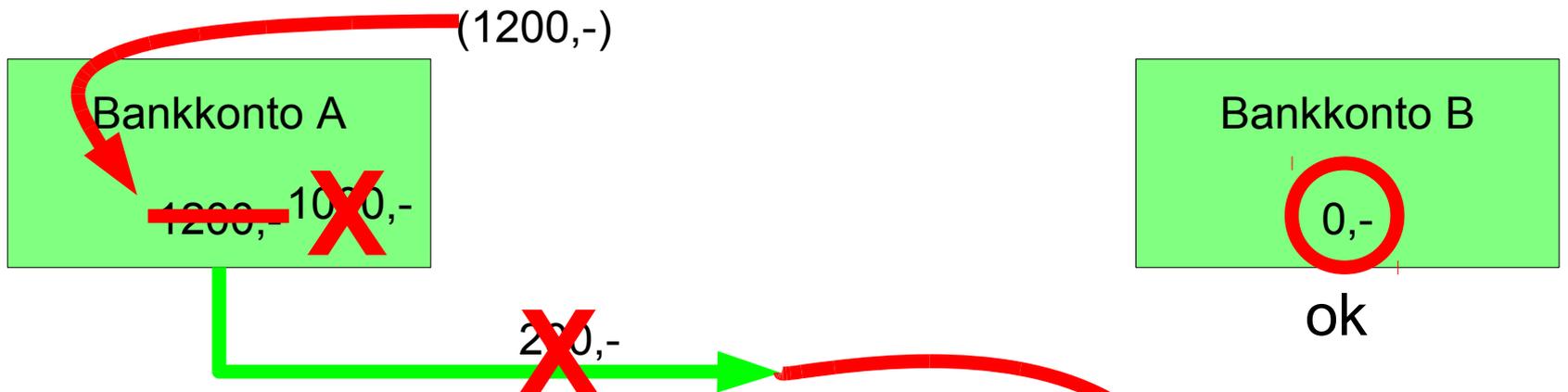
„Eine Transaktion fasst in sich mehrere logisch zusammengehörige, jedoch nicht zwangsweise technisch voneinander abhängige Arbeitsschritte zusammen“

Beispiel „Überweisung“



- Abbuchung
- Crash
- Was nun?

Beispiel „Überweisung“



- Abbuchung
- Crash
- Zurück!

Commit / Rollback

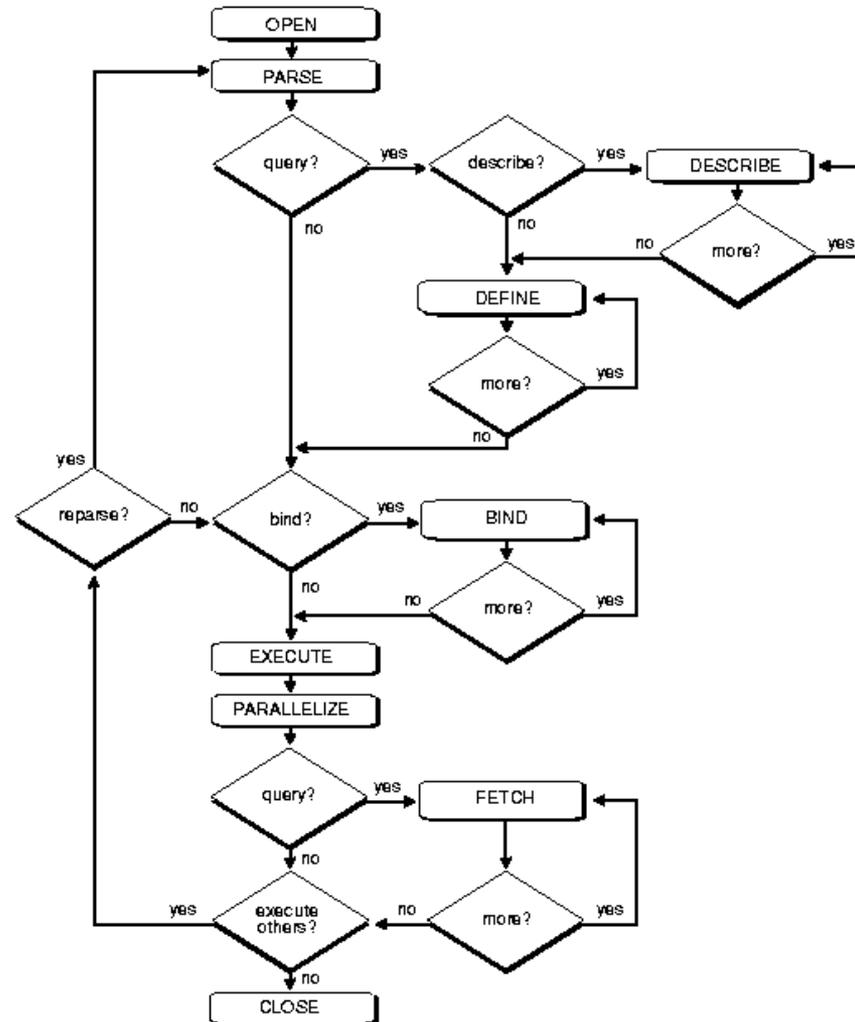
```
update KONTEN where KONTO_NR=A set SALDO=1000.00;  
update KONTEN where KONTO_NR=B set SALDO=200.00;  
commit;
```



```
update KONTEN where KONTO_NR=A set SALDO=1000.00;  
update KONTEN where KONTO_NR=B set SALDO=200.00;  
##### update ERROR 890 - USER NOT SANE #####  
rollback;
```

- Vorstellung
- Geschichte
- Aufbau und Architektur
- Relationen
- Interne Abläufe
 - Cache/Absicherung
 - Transaktionsprinzip
 - SELECT
 - UPDATE
- Zusammenfassung

Behandlung von SQL



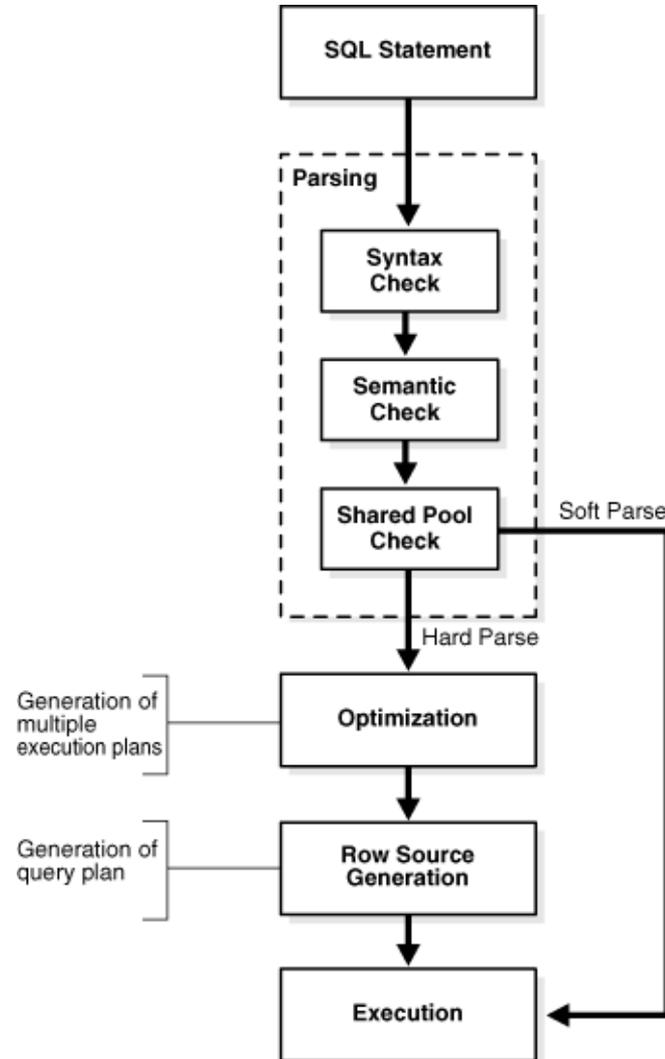
Anforderungen:

- Häufige Ausführung
- Performance / Antwortzeiten

Probleme:

- Endliche, d. h. begrenzte Systemleistung
- Unbekannte Mengenverhältnisse
und Hellsehen ist bestenfalls schwierig
- Unberechenbare „User“

Behandlung von SQL



Beispiel!

Join

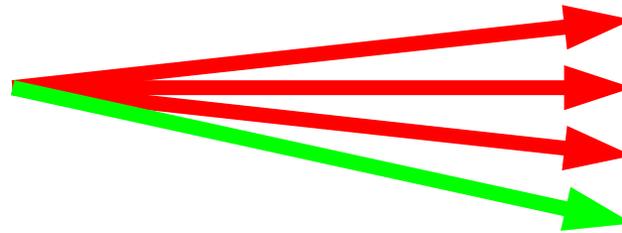
```
select b.employee_id
  from JOBS a, EMPLOYEES b
 where a.JOB_ID=b.JOB_ID
        and a.JOB_ID=20;
```

JOBS

JOB_ID	JOB_TITLE
100	Bäcker
20	Metzger
...	
13	Maler

EMPLOYEES

EM_ID	JOB_ID	NAME
5233	97	Berger
5253	13	Wolf
...		
5243	20	Schulz



- Vorstellung
- Geschichte
- Aufbau und Architektur
- Relationen
- Interne Abläufe
 - Cache/Absicherung
 - Transaktionsprinzip
 - SELECT
 - UPDATE
- Zusammenfassung

Update

```
update EMPLOYEES a  
  set a.salary = 1200  
  where a.JOB_ID = 20;
```



EMPLOYEES

EMP_ID	JOB_ID	SALARY
1	100	2000
3	13	1500
99	22	750
2	12	1200
4	144	1500
5	20	1100
6	233	3000
10	100	2100
22	22	800
19	12	1250
11	144	1330
32	12	1100
8	144	1440
9	20	1150
11	233	2990

1200



1200



commit

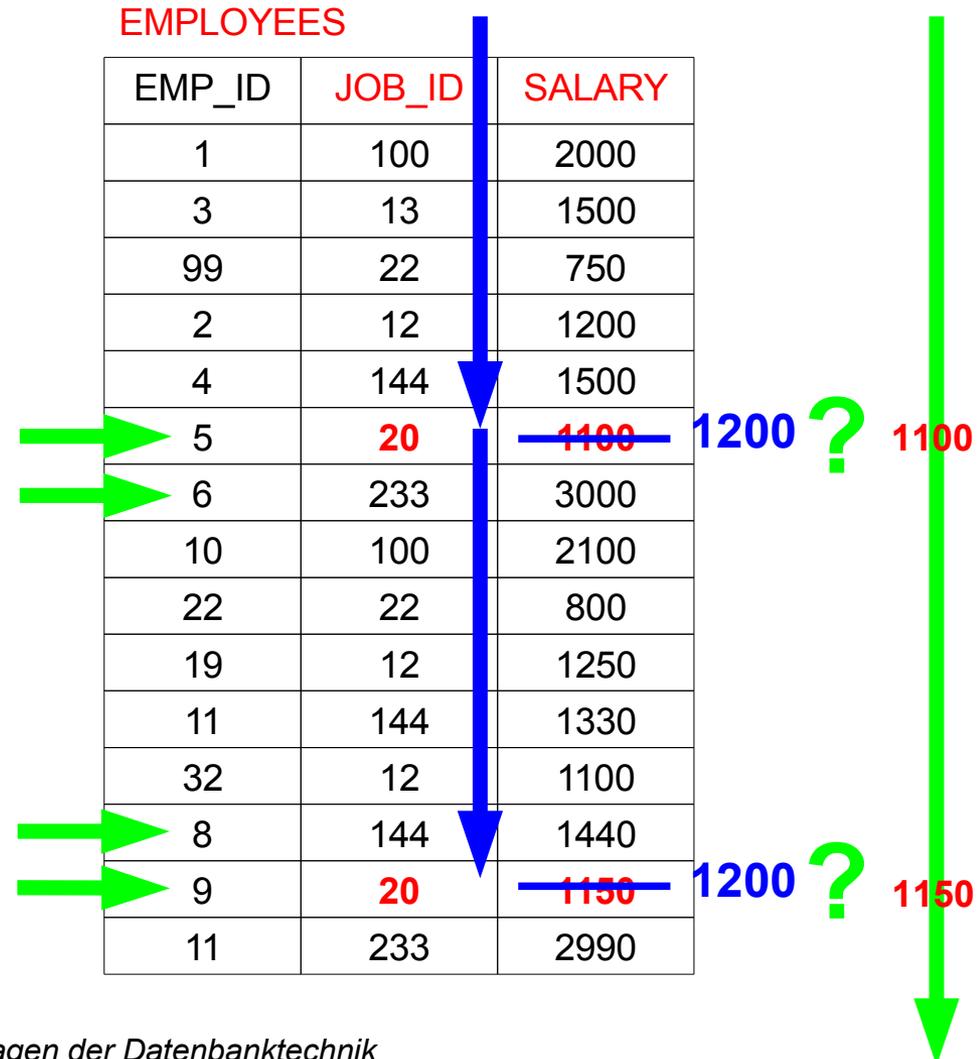
Update: Konsistentes Lesen

```
update EMPLOYEES a
  set a.salary = 1200
  where a.JOB_ID = 20;
```

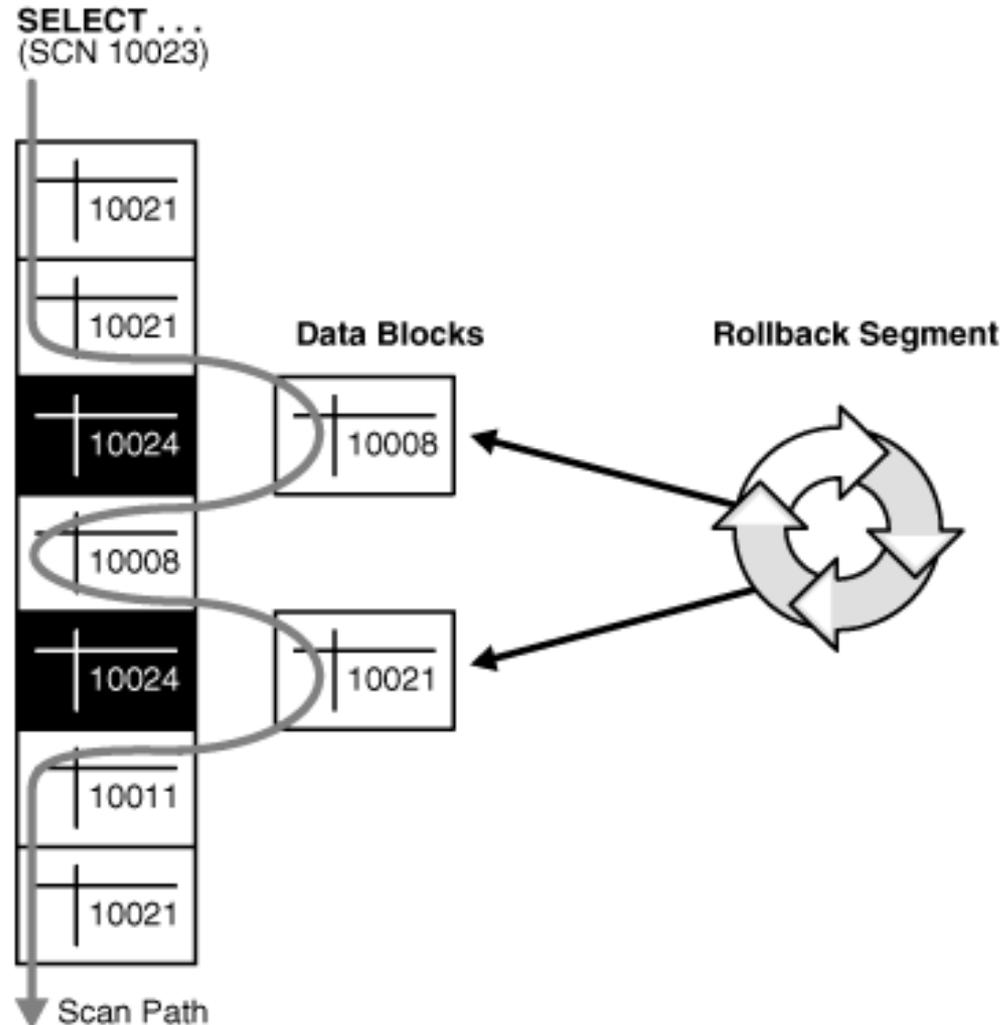
```
select SALARY
from EMPLOYEES
where EMP_ID > 4
  and EMP_ID < 10;
```

EMPLOYEES

EMP_ID	JOB_ID	SALARY
1	100	2000
3	13	1500
99	22	750
2	12	1200
4	144	1500
5	20	1100
6	233	3000
10	100	2100
22	22	800
19	12	1250
11	144	1330
32	12	1100
8	144	1440
9	20	1150
11	233	2990



Update: Konsistentes Lesen



Update: Konkurrenz

```
update EMPLOYEES a
  set a.salary = 1200
  where a.JOB_ID = 20;
```

```
update EMPLOYEES a
  set a.salary = 1500
  where a.EMP_ID = 5;
```

Lock

EMPLOYEES

EMP_ID	JOB_ID	SALARY
1	100	2000
3	13	1500
99	22	750
2	12	1200
4	144	1500
5	20	1100 1200 ?
6	233	3000
10	100	2100
22	22	800
19	12	1250
11	144	1330
32	12	1100
8	144	1440
9	20	1150 1200
11	233	2990

Update

```
update EMPLOYEES a  
  set a.salary = 1200  
  where a.JOB_ID = 20;
```

```
update EMPLOYEES a  
  set a.salary = 1500  
  where a.EMP_ID = 5;
```

Lock

EMPLOYEES

EMP_ID	JOB_ID	SALARY
1	100	2000
3	13	1500
99	22	750
2	12	1200
4	144	1500
5	20	1100 1200 1500
6	233	3000
10	100	2100
22	22	800
19	12	1250
11	144	1330
32	12	1100
8	144	1440
9	20	1150 1200
11	233	2990

commit

Thanks to Donald K. Burleson



Fully Automated SQL Tuning

Sometimes you need a human expert

- Vorstellung
- Geschichte
- Aufbau und Architektur
- Relationen
- Interne Abläufe
 - Cache/Absicherung
 - Transaktionsprinzip
 - SELECT
 - UPDATE
- Zusammenfassung

OLTP – OnLine Transaction Processing

Echtzeitbetrieb

viele kleine Transaktionen

- Kunden-, Adress- und Artikeldatenbank, (Online-)Shop
- Website-backend (dynamische Webseiten)
- Finanzbuchhaltung
- Automatisiertes Lager / Logistik / Warenwirtschaft

Response time, response time, response time!

OLAP – OnLine Analytical Processing

DataWarehouse

wenige große Transaktionen, viel Leseaktivität

- Auswertungen zur Wirtschaftlichkeit, Controlling
- Auswertungen zur Güterverteilung
- Kundenprofil-Erstellung
- Rasterfahndung (BKA/LKA)

„Erkaufen“ von Zeit - durch Aufwand von Speicherplatz

- **Tabellenkalkulation**

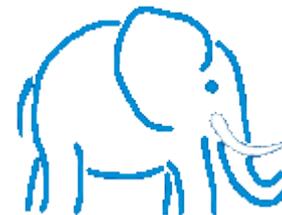
- Excel
- OpenOffice Calc

- **Workstation-Level**

- Access
- BerkeleyDB

- **Mittelklasse**

- MySQL
- PostgreSQL



- **Enterprise-Niveau**

- Oracle
- IBM DB2
- Microsoft SQL-Server
- IBM Informix
- Sybase
- (SAP-DB)

ORACLE®



Microsoft®

Informix®

SYBASE®

„Global Player“

Bewertung der Klassen

Produktklasse	Vorteile	Nachteile
Tabellenkalkulation	schnell erlernbar wenig Aufwand	kein SQL Struktur schwachen
Workstation-Niveau	geringe Größe wenig Administration	keine Mehrbenutzerfähigkeiten fehlende Absicherung (Cache usw.)
Mittelklasse	Kostenlos viele Fähigkeiten	schwache Transaktionssicherheit Sperr- bzw. Konsistenzprobleme
Enterprise-Level	tausende Features exzellenter Support	sehr teuer (~30.000 EUR / CPU) Admin: lange Einarbeitungszeit

Know your needs!

- Vorstellung
- Geschichte
- Aufbau und Architektur
- Relationen
- Interne Abläufe
 - Cache/Absicherung
 - Transaktionsprinzip
 - SELECT
 - UPDATE
 - Index
- Zusammenfassung



Quellen:

- <http://www.oracle.com/technology>
- <http://wikipedia.org>
- <http://www.usn-it.de> (eigenes Blog)

Alle Marken und Logos sind Eigentum der jeweiligen Unternehmen. Diese Präsentation dient zu Schulungszwecken und stellt keine Werbung und keine Leistungszusicherung dar, weder durch den Autor, den Referenten noch durch die Klug GmbH integrierte Systeme. Irrtum und Änderungen vorbehalten.

(c) 2010 by Martin Klier, Klug GmbH integrierte Systeme, Teunz

Dieses Werk steht unter der Creative-Commons-Lizenz „by-sa“

Vielen Dank für Ihre Aufmerksamkeit!