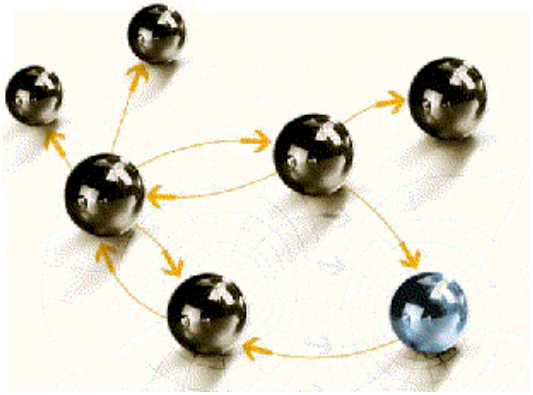


cps4it

DB2 for z/OS

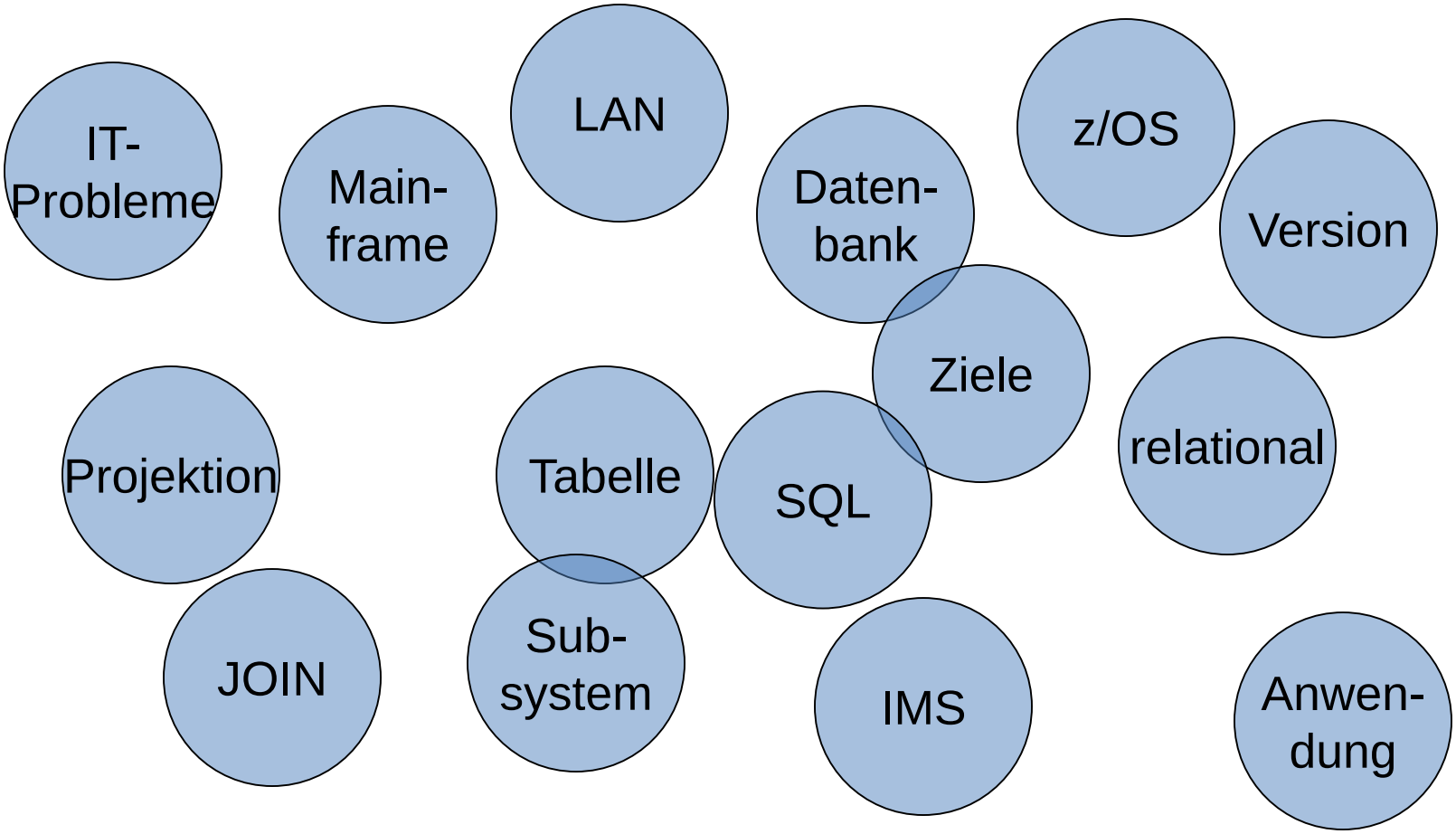
Teil 1 – Grundlagen

cps4it
consulting, projektmanagement und seminare für die informationstechnologie
Ralf Seidler, Stromberger Straße 36A, 55411 Bingen
Fon: +49-6721-992611, Fax: +49-6721-992613, Mail: rseidler@cps4it.de
Internet: http://www.cps4it.de

Inhalt

Einführung

Begriffe



Einführung

Literaturhinweise – 1

- DB2 Universal Database V8
  - George Baklarz, Bill Wong ~62 €
- Understanding DB2 …
  - Raul Chong, u. a. IBM Press ~44 €
- DB2 Developer's Guide
  - Craig S. Mullins Sams ~67 €
- DB2
  - Thomas Wiedmann ~50 €
- DB2 Theorie und Praxis (2 Bände)
  - Norbert Denne (www.dgd-ub.de) ~95 €

Einführung

Literaturhinweise – 2

- Einführung in SQL
  - Alan Beaulieu O'Reilly ~30 €
- SQL. Der Schlüssel …
  - Gregor Kuhlmann, F. Müllmerstadt ~10 €
- SQL. Quick Reference Map
  - Helmut Balzert ~10 €
- The Rational Model …

  - E.F. Codd
  - http://de.wikipedia.org/wiki/Relationale_Datenbank
- http://wiki.cps4it.de

Einführung

Literaturempfehlung

- IBM Bookmanager  0 €
  - DB2 for z/OS
  - Application Programming and SQL Guide
  - Codes
  - Command Reference
  - Diagnosis Guide and Reference
  - Reference Summary DB2
  - SQL Reference
  - Utility Guide & Reference
- DB2 Theorie und Praxis (2 Bände)
  - Norbert Denne (*www.dgd-ub.de / Intranet)* 95 € 

Herausforderungen der heutigen EDV / IT – 1

- Größe
  - Zunahme des Bedarfs an Informationen
  - steigende Nachfrage führt zu Produkten
    - DB-Systeme
    - Anwendungsprogramme
    - Werkzeuge
  - dadurch Trennung der Hardware
    - Mainframe, Midrange, PC
- neue Basistechnologien
  - personal computing
  - relationale Datenbanken
  - lokale Netze

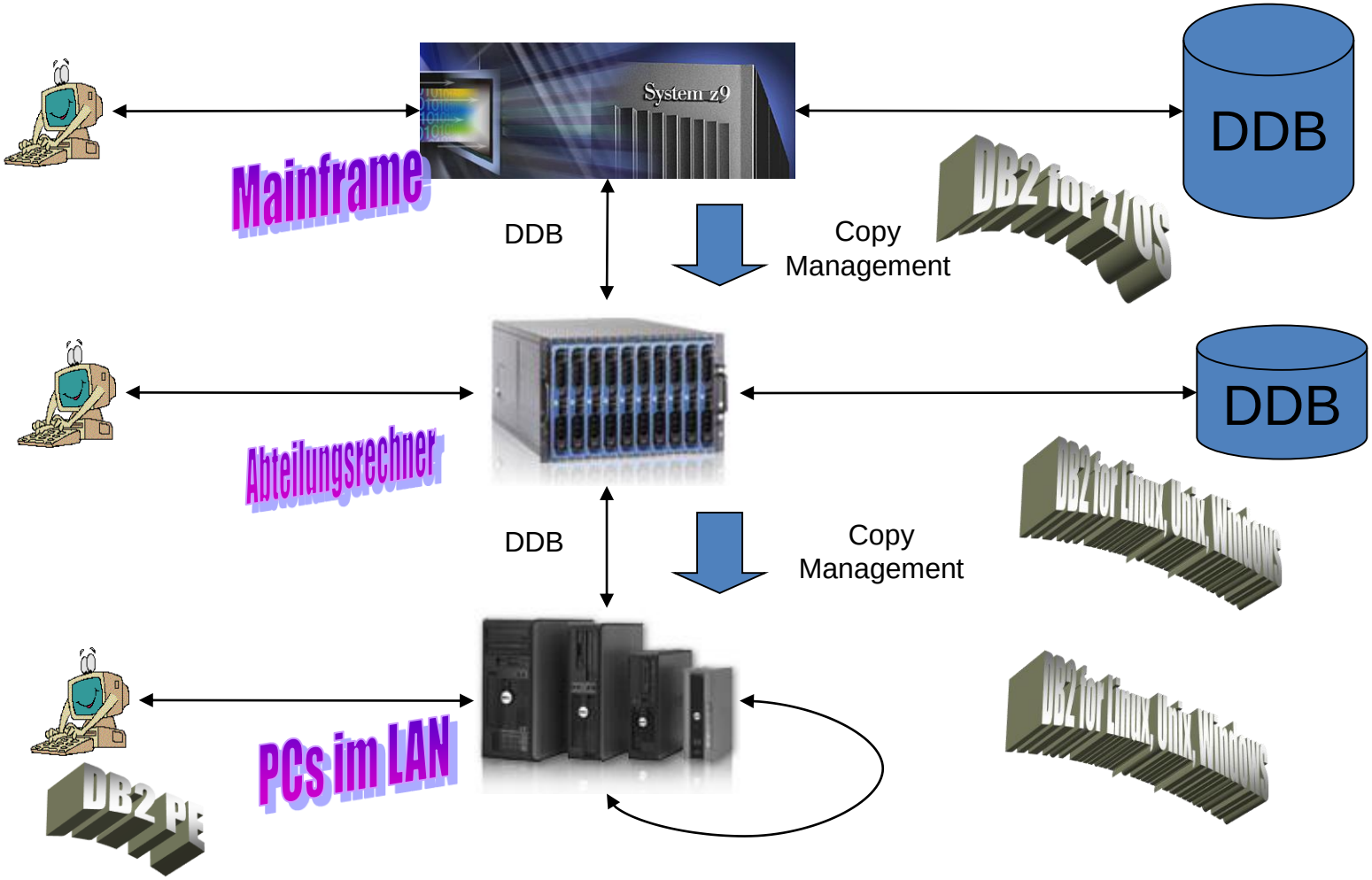
Einführung

Herausforderungen der heutigen EDV / IT – 2

- Integration über DB-Technologien
  - kontrollieren und beherrschen der Daten
  - vertikaler und horizontaler Zugriff
  - Integration systemübergreifender Anwendungen (C/S)
  - Verbesserung der Verbindung der Hardware-Ebenen

Einführung

mögliche Lösung



Einführung

Was ist DB2?

- Datenbanksystem der IBM
- DB2: Database 2 (als Nachfolger von IMS auf den Markt gebracht)
- ist Subsystem des Betriebssystems z/OS mit vielen einzelnen Komponenten

Einführung

Was ist DB2 – Geschichte ;-))

- Codd (s. nächstes Kapitel) war Mitarbeiter der IBM.
- Ausarbeitung war nicht geschützt.
- Mitbewerber erstellt RDB
- IBM geht erst durch Erfolg eines Mitbewerbers aktiv an das Thema RDB.

Einführung

Versionen von DB2 und deren wesentliche Neuerungen – 1

- V2 (1989): referentielle Integrität
- V3 (1993): verteilte Datenbank
- V4 (1995): parallele Verarbeitung
- V5 (1997): OS/390
- V6 (1999): Universal Database
- V7 (2001): 64-bit-Unterstützung
- V8 (2004): Unicode
- V9 (2009): SOA, XML, Gruppenverarbeitung

Einführung

Versionen von DB2 und deren wesentliche Neuerungen – 2

- V10 (2010)
  - CPU-Reduzierung, WLM, Optimizer
- V11 (ESP März 2013, GA offen)
  - CPU-Reduzierung, bessere Hardwarenutzung, noch mehr Wartung online möglich, Einbindung von SPSS (Statistik / Vorhersagen), . . .
  - Details siehe http://www-01.ibm.com/common/ssi/cgi-bin/ssialias?subtype=ca&infotype=an&supplier=897&letternum=ENUS212-364

Einführung

Eigenschaften eines Datenbanksystems

- Datenintegrität
- redundanzfrei
- optimiert für Zugriffe
- Daten und Programme unabhängig
- dynamisch erweiterbar
- Datenschutz
- Datensicherheit

Einführung

Zielsetzung eines Datenbanksystems

- leicht handhabbar
- Orientierung am Endbenutzer
- produktive Anwendungsentwicklung
- relationales Datenmodell
- standardisierte Abfragesprache (SQL)
  - SQL: structured query language

Einführung

Was heißt relational?

- Es gibt Tabellen.
- Der Zugriffspfad ist unabhängig, d.h. es gibt keine Pointer.
- Benutzerbefehle erzeugen wieder Tabellen.
- Es gibt eine Mengen orientierte Verarbeitung (“Set”-Verarbeitung).

Einführung

ein relationales Beispiel – die Tabelle Person

PERSON

======

PERSNR	NAME	BRUTTO	ABTNR
1357	Meyer	2000	12
0815	Müller	1750	14
4711	Schneider	3000	12
6231	Schmidt	1250	14
9876	Krause	1400	11
1234	Walther	1550	14

Einführung

ein relationales Beispiel – Mengenoperation – Restriktion

```
SELECT   PERSNR, NAME, BRUTTO, ABTNR
  FROM   PERSON
  WHERE  ABTNR = 14
```

PERSON

PERSNR	NAME	BRUTTO	ABTNR
1357	Meyer	2000	12
0815	Müller	1750	14
4711	Schneider	3000	12
6231	Schmidt	1250	14
9876	Krause	1400	11
1234	Walther	1550	14

Ergebnis:

PERSNR	NAME	BRUTTO	ABTNR
0815	Müller	1750	14
6231	Schmidt	1250	14
1234	Walther	1550	14

Einführung

ein relationales Beispiel – Mengenoperation – Projektion

```
SELECT    PERSNR, NAME
  FROM    PERSON
```

PERSON

PERSNR	NAME	BRUTTO	ABTNR
1357	Meyer	2000	12
0815	Müller	1750	14
4711	Schneider	3000	12
6231	Schmidt	1250	14
9876	Krause	1400	11
1234	Walther	1550	14

Ergebnis:

PERSNR	NAME
1357	Meyer
0815	Müller
4711	Schneider
6231	Schmidt
9876	Krause
1234	Walther

Einführung

ein relationales Beispiel – Mengenoperation – JOIN – 1

PERSON

PERSNR	NAME	BRUTTO	ABTNR
1357	Meyer	2000	12
0815	Müller	1750	14
4711	Schneider	3000	12
6231	Schmidt	1250	14
9876	Krause	1400	11
1234	Walther	1550	14

ABTEILUNG

ABTNR	ABT_NAME
12	Human Ressources
14	Einkauf
11	IT-Zentral

Einführung

ein relationales Beispiel – Mengenoperation – JOIN – 2

```
SELECT   *
  FROM   PERSON, ABTEILUNG
 WHERE   PERSON.ABTNR = ABTEILUNG.ABTNR
```

PERSON

PERSNR	NAME	BRUTTO	ABTNR
1357	Meyer	2000	12
0815	Müller	1750	14
4711	Schneider	3000	12
6231	Schmidt	1250	14
9876	Krause	1400	11
1234	Walther	1550	14

ABTEILUNG

ABTNR	ABT_NAME
12	Human Ressources
14	Einkauf
11	IT-Zentral

Ergebnis:

PERSNR	NAME	BRUTTO	ABTNR	
1357	Meyer	2000	12	Human Ressources
0815	Müller	1750	14	Einkauf
4711	Schneider	3000	12	Human Ressources
6231	Schmidt	1250	14	Einkauf
9876	Krause	1400	11	IT-Zentral
1234	Walther	1550	14	Einkauf

Einführung

ein relationales Beispiel – Mengenoperation

- In relationalen Datenbanken gibt es keine vordefinierten Zugriffspfade.
- Bei einer SET-Verarbeitung (Mengen-verarbeitung) ist das Ergebnis einer Relationsoperation wieder eine Relation d.h. eine Tabelle.
- Ein (Zwischen-)Ergebnis kann für weitere Operationen benutzt werden.

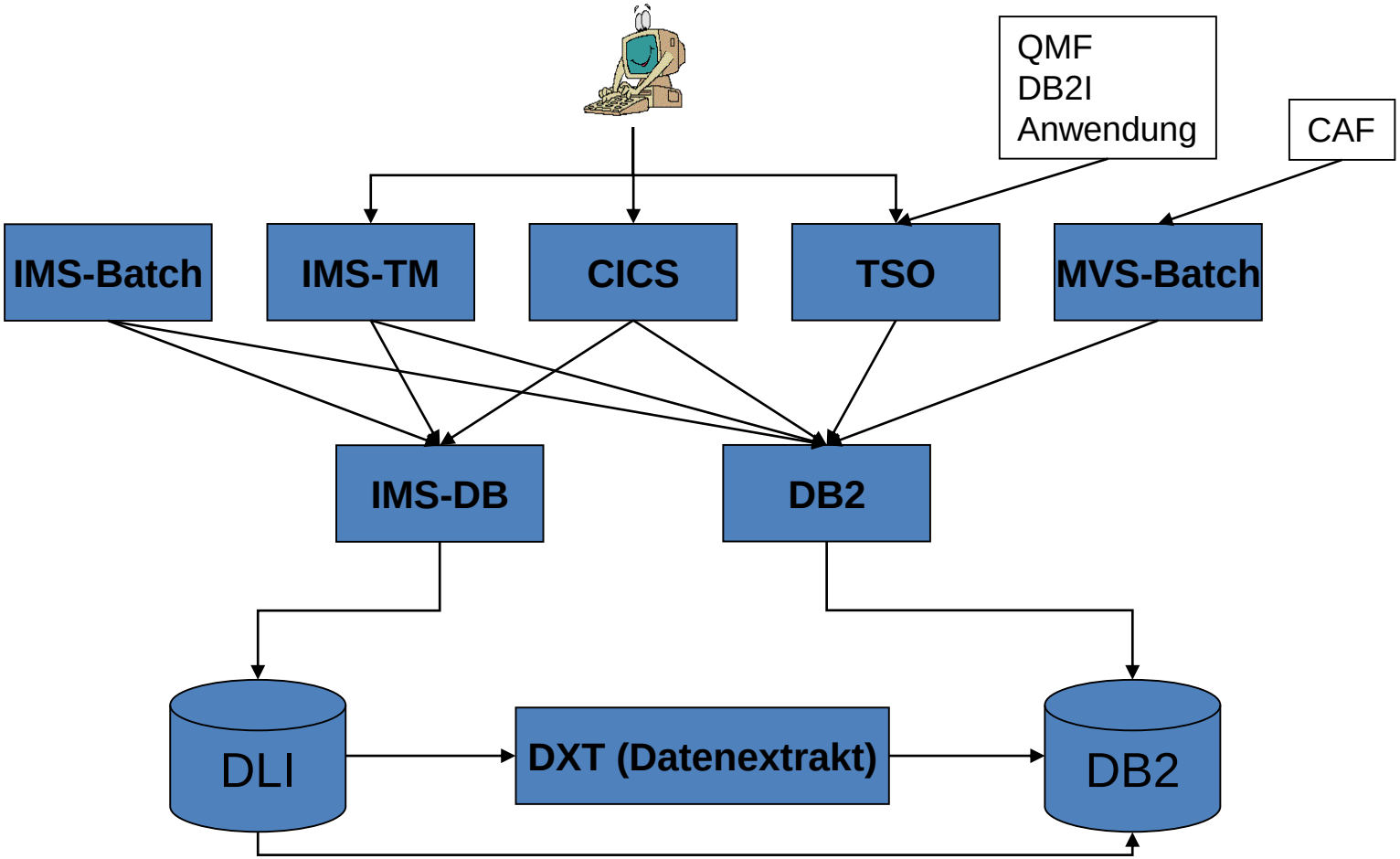
Einführung

zum „Spielen“

- Es gibt von IBM die Freeware DB2-Express für unterschiedliche Plattformen: http://www.ibm.com/developerworks/downloads/im/db2express/index.html
- Forum: http://www.ibm.com/developerworks/forums/forum.jspa?forumID=805
- deutsche Unterlage: http://public.dhe.ibm.com/software/dw/db2/express-c/wiki/Einstieg_in_DB2_Express-C_9.7.pdf

Einführung

Subsysteme des Betriebssystems z/OS



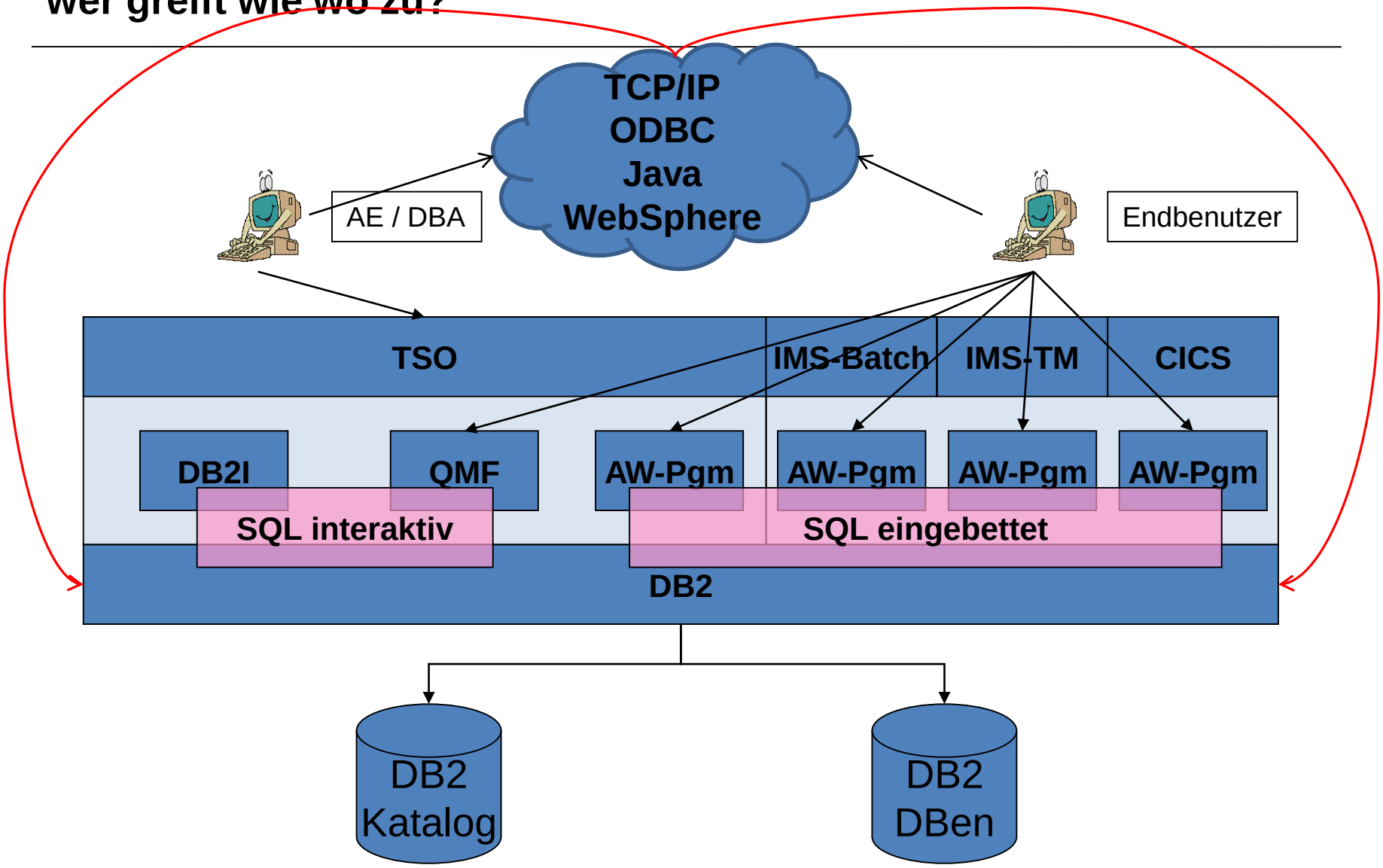
Einführung

Benutzerschnittstelle – SQL

- Interaktiv
  - DB2I
  - QMF
- Anwendungsprogramm
  - SQL im Programm eingebettet
    ASM, C, COBOL, PL1, FORTRAN, APL
  - via Programmgeneratoren wie
    - Visual Age Gen, Advantage Gen / All Fusion Gen

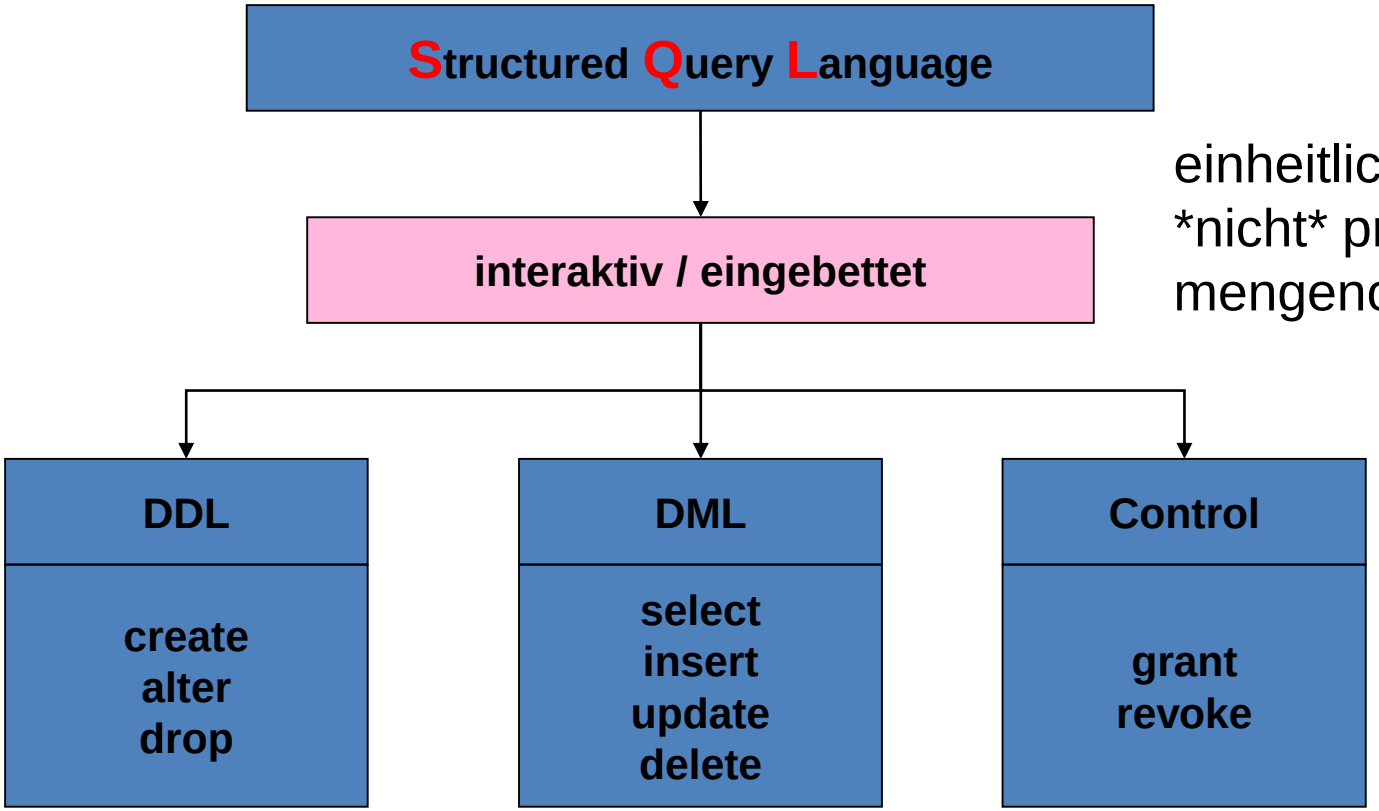
Einführung

wer greift wie wo zu?



Einführung

SQL



einheitlich
nicht prozedural
mengenorientiert

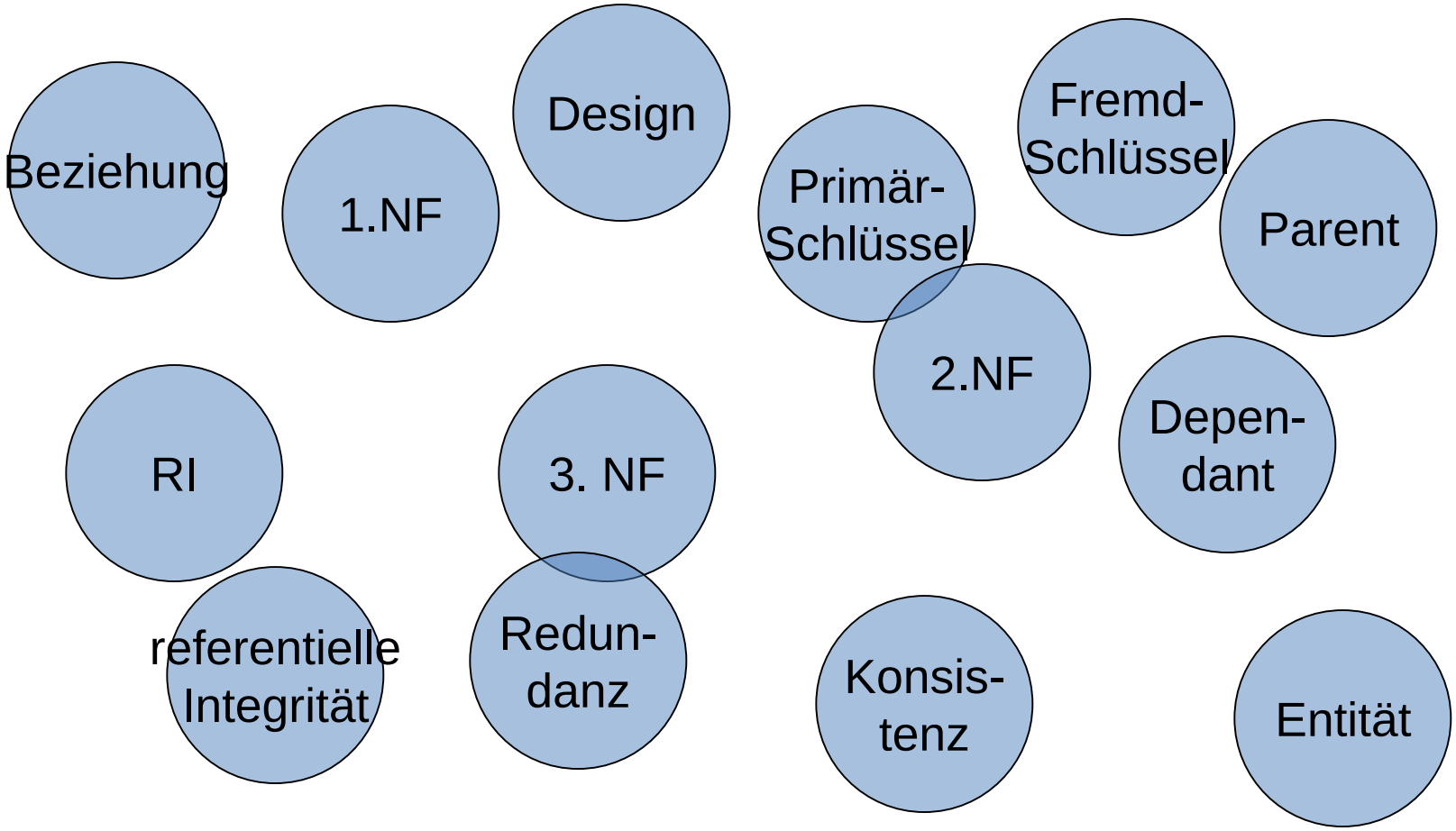
nicht
wie
sondern
was

Inhalt

- Einführung und Überblick
- Datenbank-Design
- Beispiel-Datenbank
- Datendefinition
- Speicherstruktur
- DB2-Menü – DB2I

Datenbank-Design

Begriffe



Datenbank-Design

Definition

- Welche Tabellen brauchen wir?
- Welche Felder brauchen wir?
- Datenmodell
  - konzeptionelles Datenbank-Design
  - objektbezogene Datenbank-Strukturen
  - berücksichtigen der Anwenderforderungen
  - Stabilität der Daten
- vereinfachte Darstellung (s. Literatur)

Datenbank-Design

Beispiel Personaldatenbank – Beschreibung – 1

- Unternehmen hat Abteilungen; neue kommen hinzu, andere fallen weg
- Jede Abteilung hat
  - eine eindeutige Abteilungsnummer,
  - einen Namen
  - einen Abteilungsleiter
  - ein Budget
  - eine Anzahl Mitarbeiter (auch keine)

Datenbank-Design

Beispiel Personaldatenbank – Beschreibung – 2

- Jeder Mitarbeiter hat
  - eine eindeutige Personalnummer,
  - einen Namen
  - eine bestimmte Tätigkeit
  - ein Gehalt
- Kein Mitarbeiter kann zur gleichen Zeit verschiedenen Abteilungen angehören.
- Abteilungsleiter führen nur eine Abteilung.
- Keine Abteilung ist einer anderen über- oder untergeordnet.

Datenbank-Design

Design – 1. Möglichkeit

- 1 Tabelle; Mitarbeiterdaten innerhalb der Tabelle ABTLG

ABTLG	ABTNR	NAME	ABT_LTR	BUDGET	ANZ_MA

PERSON1	PERSNR	NAME	TAET	GEHALT
PERSON2	PERSNR	NAME	TAET	GEHALT
PERSON3	PERSNR	NAME	TAET	GEHALT

Datenbank-Design

Design – 1. Möglichkeit – Bewertung

- Vorteil
  - Die Abteilungsnummer muss nicht für jeden Mitarbeiter erneut genannt werden.
- Nachteile
  - Begrenzung der Anzahl MA pro Abteilung.
  - Abteilungen mit wenig MA haben viele “Null”-Werte.
  - Bei bestimmter Sortierung der Mitarbeiterdaten für jede Abteilung müssen bei Einfügen und Löschen erhebliche Datenbewegungen stattfinden.
  - Einfache Abfragen werden komplex wie
    - In welcher Abteilung ist der MA mit der PERSNR 4711?
    - Wie hoch ist das Durchschnittsgehalt in Abteilung A12?

Datenbank-Design

Design – 2. Möglichkeit

- 1 Tabelle; Abteilungsdaten innerhalb der Tabelle PERSON

PERSON	PERSNR	NAME	TAET	GEHALT	ABTNR
	NAME	ABT_LTR	BUDGET		

Datenbank-Design

Design – 2. Möglichkeit – Bewertung

- Vorteil
  - alle Abteilungsdaten für einen MA sofort im Zugriff
  - schnelles Lesen
- Nachteile
  - redundante Datenhaltung
  - ändern der Abteilungsdaten kritisch wegen möglicher Inkonsistenzen
  - keine Abteilung ohne Mitarbeiter

Datenbank-Design

Design – 3. Möglichkeit

- 2 Tabellen

ABTLG	ABTNR	NAME	ABT_LTR	BUDGET	ANZ_MA

PERSON	PERSNR	NAME	TAET	GEHALT	ABTNR

Datenbank-Design

Design – 3. Möglichkeit – Bewertung

- redundanzfrei
- konsistent

ABTLG	ABTNR	NAME	ABT_LTR	BUDGET	ANZ_MA

PERSON	PERSNR	NAME	TAET	GEHALT	ABTNR

- Daten gehen nicht verloren (beim letzten Mitarbeiter einer Abteilung)
- Abteilungsdaten können eingefügt werden, ohne dass Mitarbeiter vorhanden sein müssen.

Relationenmodell – Modell von CODD

- publiziert 1970
- CODD gilt als “Vater” des relationalen Datenmodells

- Alle relationalen Datenbanksysteme basieren auf dem relationalem Datenmodell.

Relationenmodell – Datenstruktur

- Tabelle ist logische Datenstruktur
- Relation = Tabelle (table)
  - nur diese für Benutzer sichtbar
  - unabhängig von Organisation und Speicher
- Tupel = Zeile (row)
- Attribut = Spalte (column)
  - hierüber erfolgt der Zugriff
  - Menge zulässiger Werte = Wertebereich

Relationenmodell – Entität (Objekt) – 1

- Bestimmen der Entität ist Ausgangspunkt des DB-Designs
- Entität ist individuelles und identifizierbares Exemplar von Dingen, Personen oder Begriffen wie
  - Mitarbeiter, Abteilung, Lehrgang, Buchung, Auftrag, Vertrag, Kunde, Agentur

Datenbank-Design

Relationenmodell – Entität (Objekt) – 2

- Entität hat bestimmte Eigenschaften (Attribute) wie
  - Personalnummer, Namen, Gehalt, Tätigkeit
- Entitäten von gleichem Typ (gleiche Eigenschaften) werden zu Entitätsmengen zusammen gefasst.
- Jede Entitätsmenge (Objektmenge) wird in einer eigenen Relation (Tabelle) dargestellt.

Datenbank-Design

Primärschlüssel – Primary Key (PK)

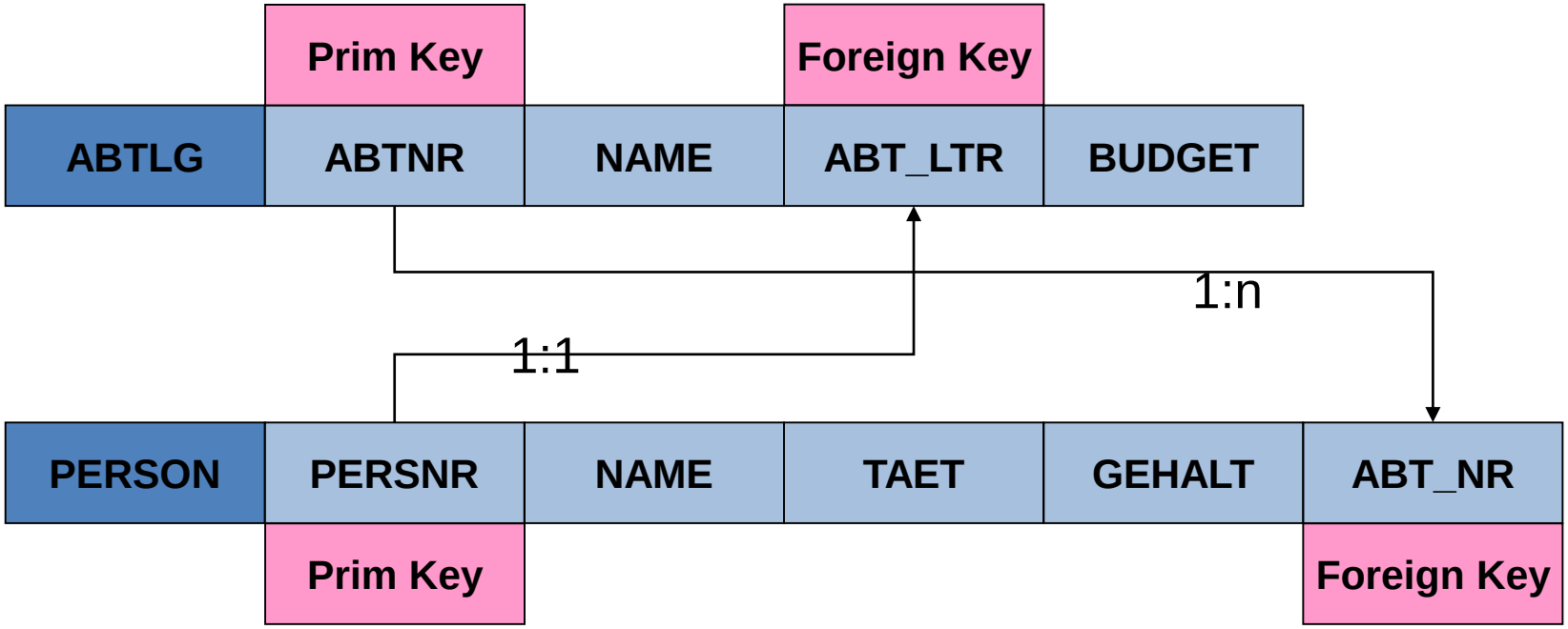
- besteht aus Spaltenwert oder einer Kombi-nation von Spaltenwerten, der/die die Zeile identifiziert
- ist eindeutig und muss bekannt sein (nicht “NULL”)
- Tabelle im Relationenmodell besteht mindestens aus
  - einem Primärschlüssel
  - zusätzlichen Spalten (Attribute), die die Eigenschaften des gleichen Objekts beschreiben (keine anderen Objekte)

Relationenmodell – Schlüsselintegrität

- Die Korrektheit des Primärschlüssels wird als “Entity Integrität” bezeichnet; Eigenschaften:
  - Eindeutigkeit
  - Wert immer bekannt
- Soll DB2 die Primärschlüsselintegrität garantieren, muss der Benutzer dies bei der Datendefinition vorsehen (siehe später)
  - NOT NULL
  - UNIQUE INDEX

Datenbank-Design

Beziehung zwischen Tabellen – Beispiel



Datenbank-Design

Beziehung zwischen Tabellen – Arten

- einfach zu einfach (1:1)
  - 1 MA leitet 1 Abteilung
  - 1 Abteilung wird von 1 MA geleitet
- einfach zu mehrfach (1:n)
  - 1 Abteilung hat viele MA
  - Viele MA gehören zu 1 Abteilung
- mehrfach zu mehrfach (n:m)
  - Verschiedene Teile werden von verschiedenen Lieferanten geliefert. (Wer liefert was?)

Datenbank-Design

Fremdschlüssel – Foreign Key (FK)

- bei 1:n-Beziehung:
  - “Mehrfach”-Tabelle enthält Fremdschlüssel
    “Mehrfach”-Tabelle = DEPENDENT-Tabelle
  - “Einfach”-Tabelle enthält Primärschlüssel
    “Einfach”-Tabelle = PARENT-Tabelle
- Verbindung der Tabellen über FK/PK
- Beziehung durch Aufnahme des PK als FK (Attribut) in andere Tabelle.
- PK und FK mit gleichem Wertebereich
- unterschiedliche Namen für PK und FK ok.

Datenbank-Design

Fremdschlüssel – Referentielle Integrität

- Da die Beziehungen zwischen Tabellen nur durch die Werte der PK und FK abgebildet werden, ist die Korrektheit und Widerspruchs-freiheit dieser Daten sehr wichtig.
- Jeder Fremdschlüsselwert muss mit genau einem Primärschlüsselwert übereinstimmen oder muss den Wert NULL enthalten.
- Referentielle Integrität = RI

Datenbank-Design

n:m-Beziehung – DB Materialbeschaffung – Beschreibung

- Betrieb benötigt verschiedene Teile, die von verschiedenen Lieferanten bezogen werden
- Jeder Lieferant liefert verschiedene Teile.
- Zu jedem Zeitpunkt ist nur eine Lieferung einer Teileart von einem Lieferanten offen.
- Lieferant hat eindeutige Lieferantennummer, Name, Bewertungskennzeichen und Firmenort
- Teileart hat eindeutige Teilenummer, Namen, Farbe, Gewicht und Lagerort

Datenbank-Design

n:m-Beziehung – DB Materialbeschaffung – Design

Lieferant (L)	LNR	LNAME	STATUS	ORT

TEIL (T)	TNR	TNAME	FARBE	GEWICHT	ORT

Auftrag (A)	LNR	TNR	PNR	MENGE

Projekt (P)	PNR	PNAME	ORT

n:m-Beziehung – DB Materialbeschaffung – Beziehungen

- mehrfach zu mehrfach Beziehung zwischen Entitätsmengen “Lieferanten”, “Teil” und „Projekt“
- Jeder Lieferant liefert mehrere Teilearten.
- Für jede Teileart gibt es mehrere Lieferanten.
- Jede Kombination aus Teil und Lieferant kann an verschiedene Projekte gehen.
- n:m-Beziehungen ergeben über die drei Primärschlüssel eine eigene Tabelle (Relation).

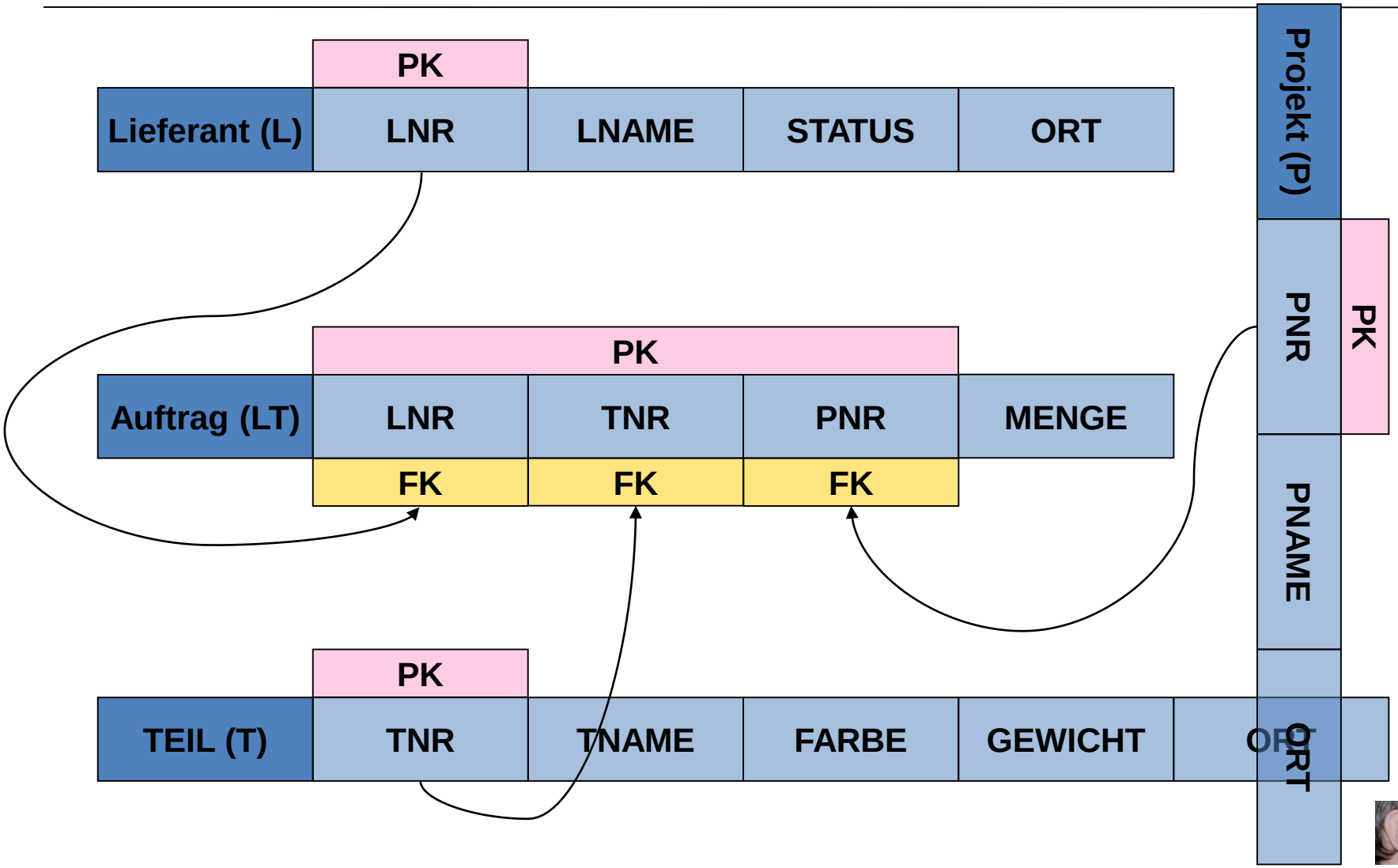
Datenbank-Design

n:m-Beziehung – DB Materialbeschaffung – Tabelle Auftrag

- Jede Zeile stellt einen einzelnen Auftrag dar.
- Die Zeile enthält den Lieferanten, das Teil, den Empfänger und die zu liefernde Menge.
- In der Tabelle “Auftrag” sind “LNR”, “TNR” und „PNR“ Fremdschlüssel.
- Die Menge ist eine Eigenschaft (Attribut) des Auftrags (und nicht von “T” oder “L” oder “P“)
- Primärschlüssel ist die Kombination aus “LNR”, “TNR” und “PNR“. Durch ihn wird Auftrag eindeutig identifiziert.

Datenbank-Design

n:m-Beziehung – DB Materialbeschaffung – Beziehungen



Normalisierung – Design-Qualität

- Um einfache Relationen zu erhalten, wurde formalisierter Zerlegungsprozess für die Daten entwickelt.
- Es werden verschiedene Stufen für die Abhängigkeit der Daten untereinander definiert:
- 1. Normalform
- 2. Normalform
- 3. Normalform

- Eine Relation ist in der 1. NF, wenn alle Attribute direkt (funktional) vom Primärschlüssel abhängig sind.
  oder:
- Jedes Attribut kann nur einen Wert annehmen. Wiederholgruppen sind nicht erlaubt.

- 1970, Codd:
  A relational R is in 1NF if and only if all underlying domains contain atomic values only.

- Eine Relation in der 1. NF ist automatisch in der 2. NF, wenn der Primärschlüssel nicht aus mehreren Attributen zusammen gesetzt ist.
  oder:
- Bei zusammen gesetzten Primärschlüsseln muss jedes Attribut vom gesamten Primärschlüssel direkt abhängig sein.
- 1971, Codd
  A relational R is in 2NF if it is in 1NF and every non-key attribute is fully dependant on the primary key. (Any relation in 1NF and not in 2NF must have a composite key.

- Die 3. NF ist erfüllt, wenn die 2. NF erfüllt ist und alle Attribute, die nicht zum Primärschlüssel gehören, voneinander unabhängig sind.

- 1971, Codd

  A relational R is in 3NF if it is in 2NF and every non-key attribute is non transitively dependant on the primary key.

Normalisierung – 4. Normalform

- Die 4. NF ist erfüllt, wenn die 3. NF erfüllt ist und keine paarweisen mehrwertigen Abhängigkeiten zwischen Attributen bestehen.

- 1977, Fagin
  A normalised relational R is in 4NF if and only if whenever there exists a multivalued dependency in R, say of attribute B on attribute A, all attributes of R are also functionally dependant on A.

Normalisierung – 5. Normalform

- Die 5. NF ist erfüllt, wenn sie notwendig ist, Daten der 4.NF ohne Informationsverlust über einen Join zusammen zu führen.

- 1979, Fagin
  A relational R is in 5NF if and only if every join dependency in R is a consequence of keys of R.

Normalisierung – Fragen

- Ist das denn noch normal?
- Das kann doch keiner mehr verstehen!
- Ist der ganze Quatsch denn notwendig?

Datenbank-Design

Normalisierung – Antworten

- Normalisierungsprozess
  - ist aufwändig
  - liefert die Basis für stabile Datenstrukturen
  - Daten in 1. NF sind nicht sinnvoll verwaltbar
  - Daten in 2. NF sind schwierig verwaltbar
  - mindestens bis 3. NF durchführen
  - bis 5. NF “garantiert” stabile Ergebnisse zur Laufzeit
- Denormalisierung für Physik immer möglich!

Datenbank-Design

Normalisierung – Merksatz

- every entity depends

  on the key,

  the whole key,

  and nothing but the key

cps4it

Konsistenzregeln – referential constraint

- Die referential constraint ist die Regel, nach der verfahren werden muss, um die in Beziehung stehenden Primär- und Fremdschlüsseldaten konsistent zu erhalten. Diese Regel erhält die “referentielle Integrität”:
  - Fremdschlüsselwert = Primärschlüsselwert oder Fremdschlüsselwert = NULL
- Für jede Art der Änderung gibt es bestimmte Regeln. Jeder Primär-Fremdschlüssel-Beziehung sind Regeln zugeordnet.

- Parent-Tabelle =
  Primärschlüsseltabelle =
  Tabelle mit Primärschlüssel

- Dependant-Tabelle =
  Fremdschlüsseltabelle =
  abhängige Tabelle =
  Tabelle, in der sich Fremdschlüssel befinden

Konsistenzregeln – referential constraint – Regeln – 1

- Welche Maßnahmen in den abhängigen Tabellen sind notwendig, wenn Zeilen in der Parent-Tabelle gelöscht, eingefügt oder verändert werden?
  - CASCADE (bei update und delete)
  - RESTRICT (bei update und delete)
  - NO ACTION (bei update und delete)
  - SET NULL (bei update und delete)
  - keine Maßnahme (bei insert)
  - besondere Maßnahme (bei allen Manipulationen)

Konsistenzregeln – referential constraint – Regeln – 2

- Welche Maßnahmen in den übergeordneten Tabellen sind notwendig, wenn Zeilen in der Dependant-Tabelle gelöscht, eingefügt oder verändert werden?
  - CASCADE (nicht erlaubt)
  - RESTRICT (bei update und insert)
  - NO ACTION (nicht erlaubt)
  - SET NULL (nicht erlaubt)
  - keine Maßnahme (bei delete)
  - besondere Maßnahme (bei allen Manipulationen)

Datenbank-Design

Konsistenzregeln – referential constraint im DB2

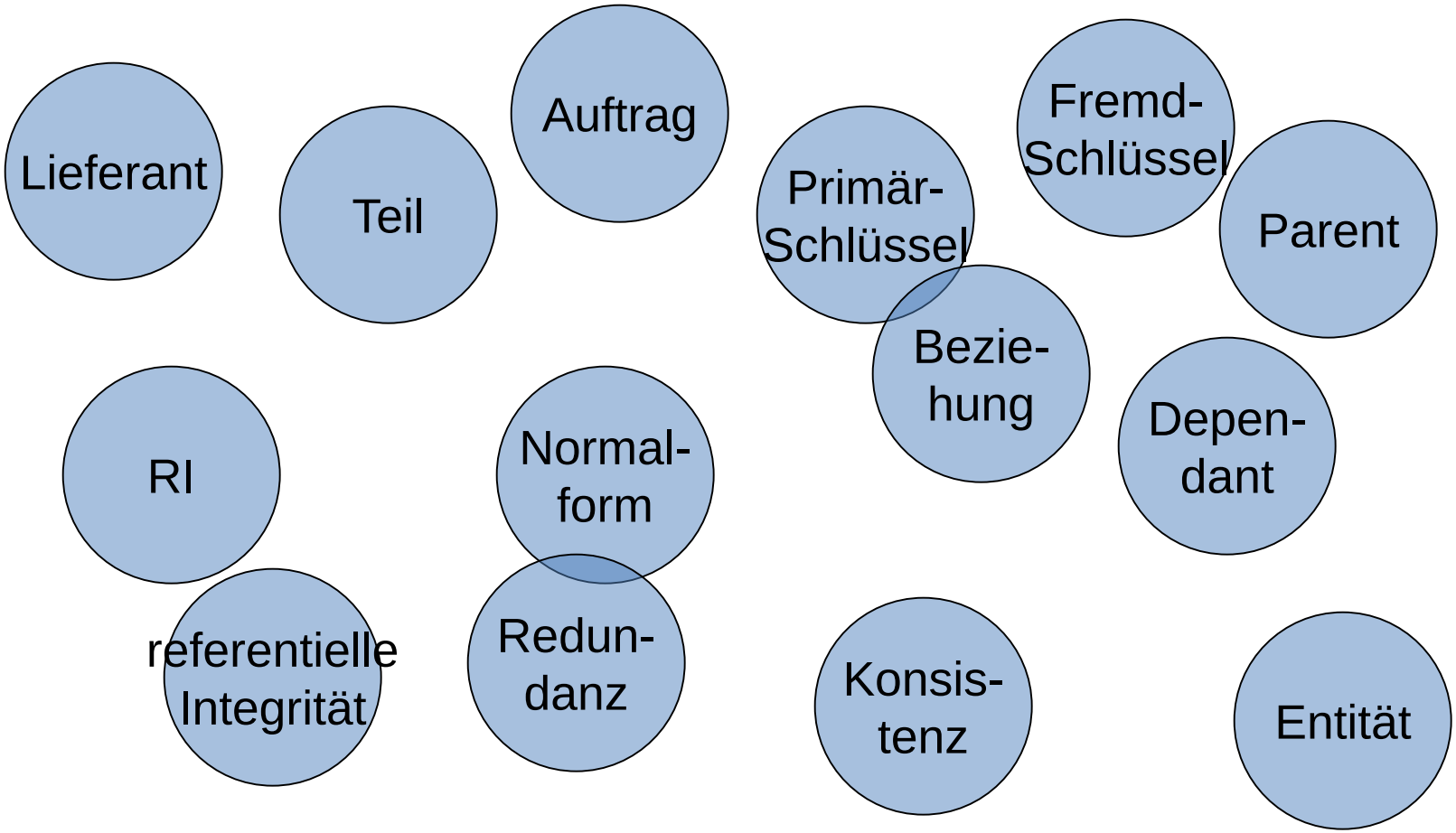
- DB2 unterstützt die Vereinbarungen zur Erhaltung der referentiellen Integrität. Die Durchsetzung der Regeln muss durch den Anwendungsentwickler nicht explizit programmiert werden. Durch entsprechende Fehlerhinweise wird bei der Verarbeitung auch im interaktiven Betrieb auf die Verletzung hingewiesen und die Änderung abgelehnt.

Inhalt

- Einführung und Überblick
- Datenbank-Design
- Beispiel-Datenbank
- Datendefinition
- Speicherstruktur
- DB2-Menü – DB2I

Beispiel-Datenbank

Begriffe



Beispiel-Datenbank

Tabelle Lieferant (Synonym: L)

L	LNR	LNAME	LSTATUS	ORT
	L1	Neumann	30	Berlin
	L2	Schmidt	20	Hamburg
	L3	Krause	30	Hamburg
	L4	Meier	10	Berlin
	L5	Schulz	20	Frankfurt

Beispiel-Datenbank

Tabelle Teil (Synonym: T)

T	TNR	TNAME	FARBE	GEWICHT	ORT
	T1	C	BLAU	19	Berlin
	T2	D	GELB	12	Hamburg
	T3	S	ROT	14	Stuttgart
	T4	S	BLAU	17	Berlin
	T5	B	ROT	17	Hamburg
	T6	N	BLAU	12	Berlin

Beispiel-Datenbank

Tabelle Auftrag (Synonym: A, LTP)

LT	LNR	TNR	MENGE
	L1	T1	400
	L1	T2	300
	L1	T3	500
	L1	T4	300
	L1	T5	200
	L1	T6	200

LNR	TNR	MENGE
L2	T1	400
L2	T2	500
L3	T2	300
L4	T2	300
L4	T4	400
L4	T5	500

Beispiel-Datenbank

zu beachten

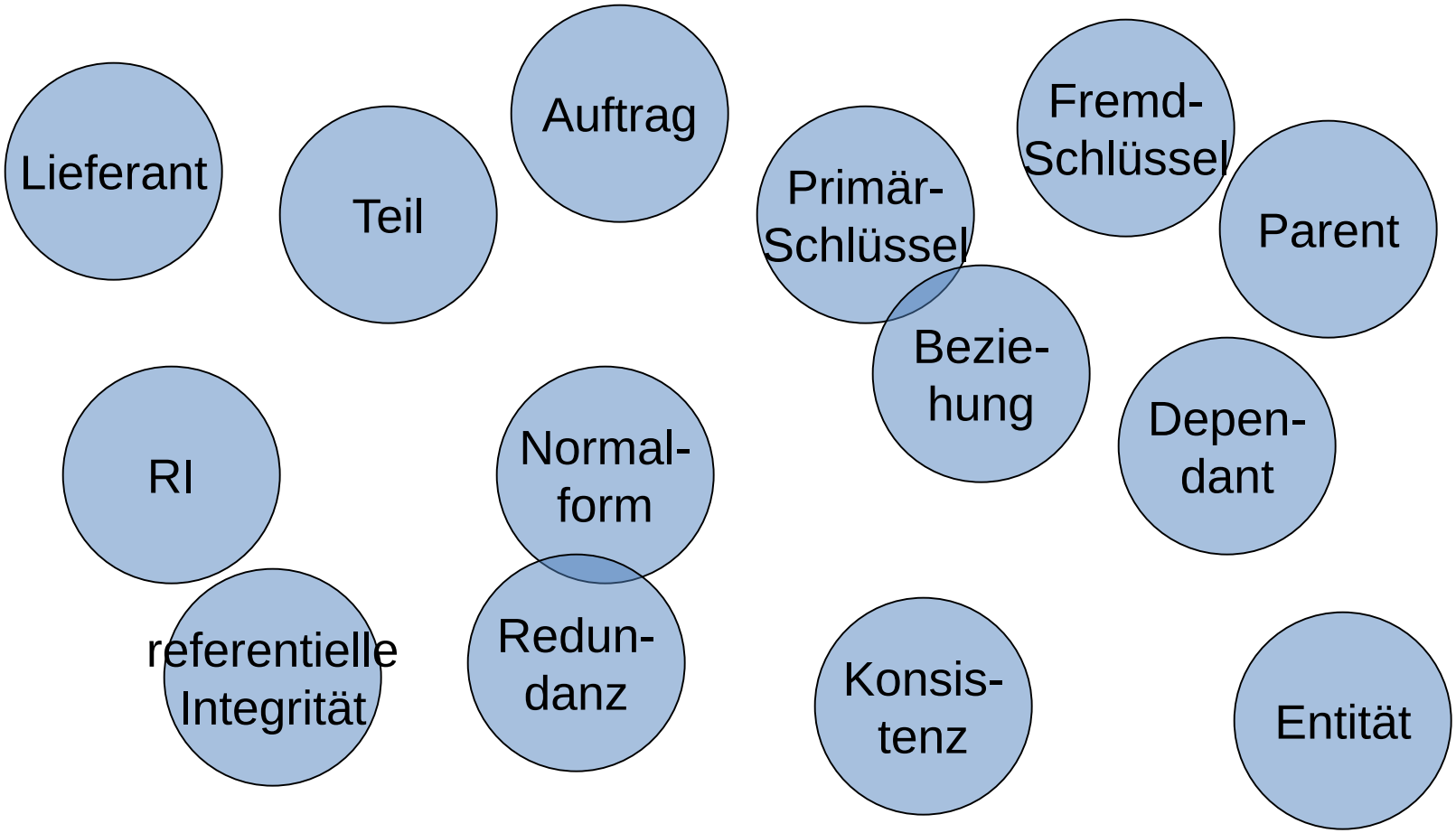
- In der Tabelle LIEFERANT identifiziert der Wert in der Spalte LNR eindeutig eine Zeile dieser Tabelle.
- In der Tabelle TEIL identifiziert der Wert in der Spalte TNR eindeutig eine Zeile dieser Tabelle.
- In der Tabelle Projekt identifiziert der Wert in der Spalte PNR eindeutig eine Zeile dieser Tabelle.
- Die Werte der Spalten LNR, TNR und PNR zusammen identifizieren eine Zeile der Tabelle AUFTRAG.

Inhalt

- Einführung und Überblick
- Datenbank-Design
- Beispiel-Datenbank
- Datendefinition
- Speicherstruktur
- DB2-Menü – DB2I

Datendefinition

Begriffe



Datendefinition

Tabelle – notwendige Parameter

- Spalte
  - Spaltenüberschrift
  - je Spalte ein Datenformat
- Zeile
  - je Spalte ein Wert
  - Zahl der Zeilen variabel
  - zu verschiedenen Zeiten verschiedene Anzahl Zeilen
  - keine geordnete Folge der Zeilen

Datendefinition

Tabelle – SQL-Befehl

```
CREATE TABLE tabellenname
      ( Spaltendefinition [,Spaltendefinition ] ...
       [,PRIMARY KEY Definition]
       [,FOREIGN KEY Definition]    )
      [ andere Parameter]
```

- Spaltendefinition

  `Spaltenname Datenformat [NULL|NOT NULL]`

Datendefinition

Tabelle – Datenformate

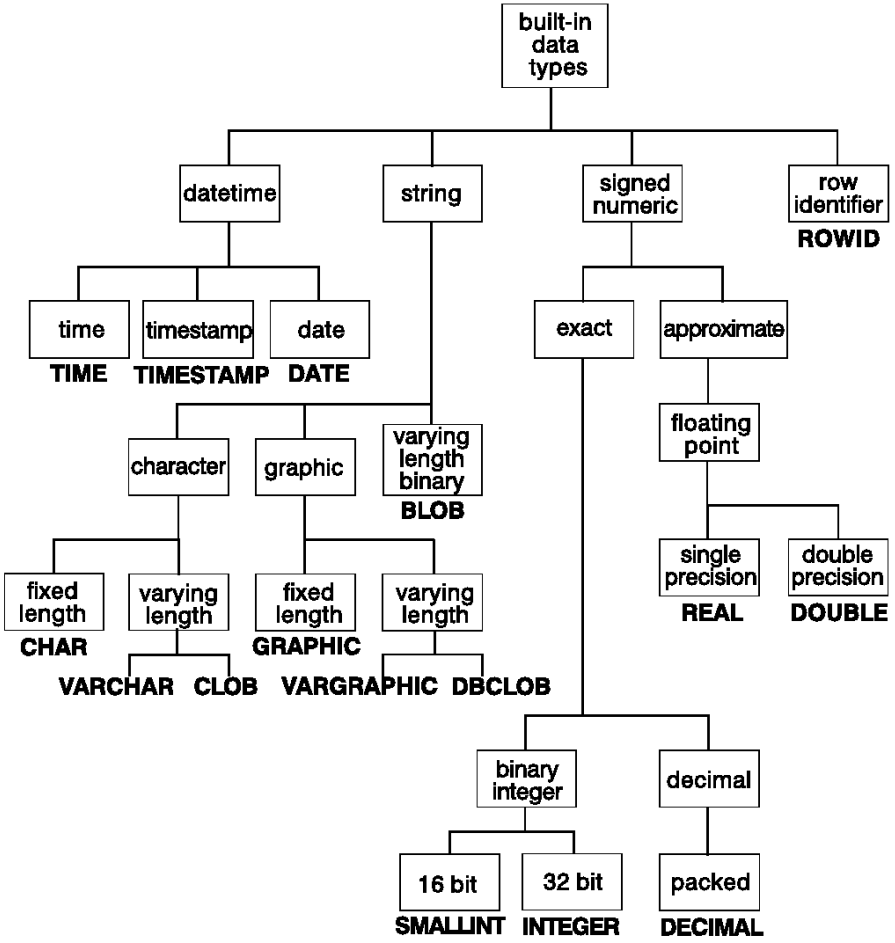


Tabelle – Datenformate – NULL

- NULL
  - Abwesenheit eines Wertes
- NOT NULL
  - Anwesenheit eines Wertes

- Achtung: Besondere Verarbeitungslogik notwendig bei NULL / NOT NULL!

Datendefinition

Tabelle – Datenformate – Zahlen – 1

- SMALLINT
  - binäre, ganze Zahl
  - 1 Halbwort, 15 Bit mit Vorzeichen
  - Wertebereich: -32.768 bis +32.767
- INTEGER
  - binäre, ganze Zahl
  - 1 Vollwort, 31 Bit mit Vorzeichen
  - Wertebereich: -2.147.483.648 bis +2.147.483.647

Datendefinition

Tabelle – Datenformate – Zahlen – 2

- REAL (früher: FLOAT(21))
  - Gleitkommazahl mit einfacher Genauigkeit
  - 32 Bit mit Vorzeichen
  - Wertebereich: -7.2E+75 bis 7.2E+75
- DOUBLE oder FLOAT (früher: FLOAT(53))
  - Gleitkommazahl mit doppelter Genauigkeit
  - 64 Bit mit Vorzeichen
  - Wertebereich: -7.2E+75 bis 7.2E+75

Datendefinition

Tabelle – Datenformate – Zahlen – 3

- DECIMAL(p, q) oder NUMERIC(p, q)
  - gepackte Dezimalzahl mit maximal 31 Stellen
  - p Stellen insgesamt, q Stellen nach dem Komma
  - Wertebereich: wie definiert
- Konstanten
  - später

Datendefinition

Tabelle – Datenformate – Zeichenkette

- CHAR(n)
  - Zeichenkette mit fester Länge
  - Länge: 1 bis 255 Stellen
- VARCHAR(n)
  - Zeichenkette mit variabler Länge
  - Länge : 1 bis 32.704
- CLOB(n)
  - Zeichenkette für sehr lange Zeichenketten
  - Länge : 1 bis 2.147.483.647

Datendefinition

Tabelle – Datenformate – sonstige Typen

- GRAPHIC – feste Länge 1 bis 127
- VARGRAPHIC – variable Länge 1 bis 1.073.741.823
- DBCLOB – Doublebyte char LOB
- BLOB – binärer String 1 bis 2.147.483.647 Byte
- ROWID – für Default-Generierung durch DB2
- XML – nur als Input für built-in-functions
- DISTINCT – Definition von eigenen Typen

Datendefinition

Tabelle – Datenformate – Datum und Zeit

- DATE
  - Jahr: 0000 bis 9999
  - Monat: 1 bis 12
  - Tag: 1 bis 31 in Abhängigkeit von Monat und Jahr
- TIME
  - Stunde: 0 bis 24
  - Minute und Sekunde: 0 bis 59
- TIMESTAMP
  - Jahr, Monat, Tag, Stunde, Minute, Sekunde, Mikrosekunde

Datendefinition

Tabelle – Datenformate – Datum und Zeit – Formate

- internes Format
  - kein Einfluss
- externes Format
  - ISO yyyy-mm-dd hh.mm.ss
  - USA mm/dd/yyyy hh:mm AM / PM
  - EUR dd.mm.yyyy hh.mm.ss
  - JIS yyyy-mm-dd hh:mm:ss
  - LOCAL beliebig über ASM-Exit
  - yyyy-mm-dd-hh.mm.ss.nnnnnn

Datendefinition

Tabelle erzeugen – Befehl

```
CREATE TABLE     LIEFERANT
       (LNR      CHAR(06)    NOT NULL,
        LNAME    CHAR(15),
        LSTATUS  SMALLINT,
        ORT      CHAR(10),
        PRIMARY KEY (LNR))
     [andere wahlweise Parameter]
```

Datendefinition

Tabelle erzeugen – Tabellenname – 1

- neue, leere Tabelle wird angelegt
- eingetragen im DB2-Katalog
- voller Name: abc.LIEFERANT
- verkürzter Name: LIEFERANT
  - maximal 18 Stellen
  - erste Stelle alphabetisch
  - Rest alphanummerisch
- abc ist Benutzeridentifikation (U-ID) des Inhabers bzw. Erstellers

Datendefinition

Tabelle erzeugen – Tabellenname – 2

- Inhaber darf Tabelle ohne Benutzeridentifikation ansprechen
- Benutzeridentifikation ist eindeutig im System
- verkürzter Tabellenname ist eindeutig innerhalb der Tabellen des gleichen Inhabers

Datendefinition

Tabelle erzeugen – Spaltenname

- voller Name
  - LIEFERANT.LNAME
- verkürzter Name
  - maximal 18 Stellen
  - LNAME
  - nur innerhalb einer Tabelle benutzbar

Datendefinition

Prüfen der Integrity – Definition RI

```
CREATE TABLE      AUFTRAG
       (LNR       CHAR(06)     NOT NULL,
        TNR       CHAR(06)     NOT NULL,
        PNR       CHAR(05)     NOT NULL,
        MENGE     INTEGER,
        PRIMARY KEY (LNR, TNR, PNR)),
        FOREIGN KEY AUF$LIEF (LNR)
        REFERENCES LIEFERANT ON DELETE RESTRICT,
        FOREIGN KEY AUF$TEIL (TNR)
        REFERENCES TEIL      ON DELETE RESTRICT,
        FOREIGN KEY AUF$PROJ (PNR)
        REFERENCES PROJEKT   ON DELETE RESTRICT)
    [andere wahlweise Parameter]
```

```
CREATE TABLE      LIEFERANT
       (LNR       CHAR(06)     NOT NULL,
        LNAME     CHAR(15),
        LSTATUS   SMALLINT,
        ORT       CHAR(10),
        PRIMARY KEY (LNR))
    [andere wahlweise Parameter]
```

Datendefinition

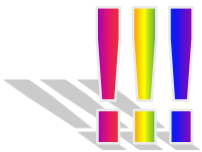
Prüfen der Integrity – Definition Prüfregel

```
CREATE TABLE      LIEFERANT
       (LNR       CHAR(06)     NOT NULL,
        LNAME     CHAR(15),
        LSTATUS   SMALLINT,
        ORT       CHAR(10),
        PRIMARY KEY (LNR),
     CONSTRAINT STATUSGUELT CHECK (LSTATUS > 9))
```

Datendefinition

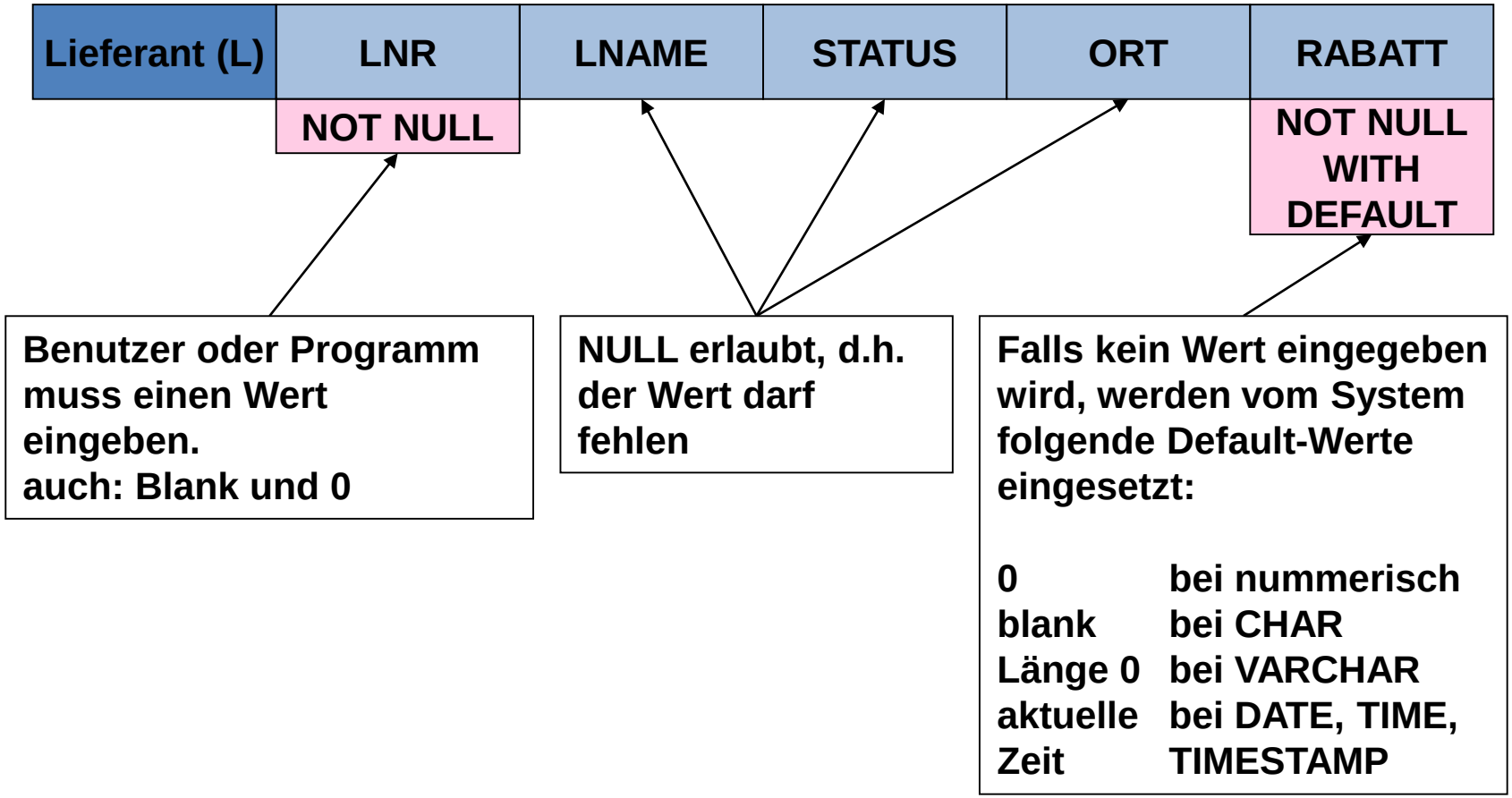
NULL

- NULL ist
  - unbekannt
  - nicht relevant
- NULL ist *nicht*
  - leer im Sinne von blank
  - 0
- zwei Null-Werte sind immer ungleich
  - Ausnahme: bei UNIQUE-INDEX
- spezielle Behandlung in Programm (Indikator)



Datendefinition

NOT NULL



Datendefinition

NULL / NOT NULL

- NOT NULL WITH DEFAULT
  - erleichtert Dateneingabe, da Defaultwerte genommen werden, falls Benutzer oder Programm keinen Wert liefern.
  - einmal gespeichert, unterscheiden sich diese Default-Werte nicht mehr von eingegebenen Werten.
- Primärschlüsselintegrität
  - Die Spalte(n), die eindeutig eine Zeile identifizieren, müssen immer NOT NULL sein.
  - DB2 prüft und setzt eventuell Fehlermeldung

Datendefinition

erweitern einer Tabelle

- Die neue Spalte wird rechts angefügt.
- Alle neuen Spalten enthalten NULL-Werte.
- Eine Definition mit NOT NULL ist nicht möglich.
- Die Beschreibung im DB2-Katalog wird geändert.

```
ALTER TABLE     LIEFERANT
      ADD       LRABATT DECIMAL(4,2)
```

Datendefinition

löschen einer Tabelle

- Die Beschreibung der Tabelle im DB2-Katalog wird entfernt.
- Die Tabelle ist nicht mehr ansprechbar.

```
DROP   TABLE      LIEFERANT
```

Datendefinition

Synonym

- Zweck
  - Der Ersteller (Creator) einer Tabelle kann seine Tabelle verarbeiten, ohne in den SQL-Befehlen seine Benutzeridentifikation davor zu stellen.
  - Ein anderer Benutzer muss diese Tabelle mit der Benutzeridentifikation des Creators qualifizieren oder ein Synonym definieren.

```
CREATE   SYNONYM L FOR ABC.LIEFERANT
```

```
DROP     SYNONYM L
```

Datendefinition

Index

- Zweck
  - ermöglicht dem DB2, eine oder mehrere Zeilen einer Tabelle schneller zu finden
  - verhindert, dass in einer Tabelle 2 Zeilen mit dem gleichen Primärschlüsselwert gespeichert werden
- Gebrauch des Index
  - nur in CREATE, ALTER, DROP
  - Index ist für Benutzer nicht sichtbar
  - DB2 entscheidet beim Zugriff über Nutzung des Index

Datendefinition

Index – Befehl

- USING STOGROUP und weitere Parameter beziehen sich auf physische Speicherung
- Sortierfolge ASC oder DESC
- Parameter UNIQUE verhindert Duplikate

```
CREATE [UNIQUE] INDEX indexname
    ON tabellenname (spaltenname [sort]
                    [,spaltenname [sort] …])
   [USING STOGROUP speichergrpname]
   [andere Parameter]
```

Datendefinition

Index – Beispiele

```
CREATE UNIQUE INDEX XLIEFERANT
    ON LIEFERANT (LNR)
```

```
CREATE UNIQUE INDEX X
    ON ATABELLE (AFELD, BFELD DESC, CFELD)
```

Datendefinition

Index – Beschreibung

- Indexname
  - maximal 18 Stellen
  - eingetragen im DB2-Katalog mit Creator
  - Indexname ist pro Benutzeridendifikation eindeutig
  - voller Name: abc.indexname
- UNIQUE
  - optionaler Parameter
  - verhindert Duplikate
  - gewährleistet eindeutige Identifkation der Zeile
  - ermöglicht DB2 Integrität auf PK (mit NOT NULL)

Datendefinition

Index – Cluster

- Ein Cluster-Index beeinflusst und regelt die physische Speicherung der Zeilen.
- DB2 speichert die Zeilen in der Sortierfolge des Indexfeldes bzw. der Indexfelder.
- Je Tabelle kann nur ein CLUSTER-Index definiert werden.
- Ein Cluster-Index sollte auf die Spalte(n) definiert werden, die am meisten sequentiell gelesen wird / werden (Tablespace).

Index – UNIQUE – Hinweise

- Nachträgliches Erstellen eines UNIQUE INDEX ist evtl. nicht mehr sinnvoll oder möglich, wenn Eindeutigkeit schon verletzt ist.
- Zwei NULL-Werte werden bei einem UNIQUE INDEX als gleich angesehen.
- Beim Laden einer Tabelle mit CLUSTER-Index gibt es keine Fehlermeldung, wenn die Daten nicht in Sortierfolge sind.

Datendefinition

Index – anlegen und löschen

- CREATE
  - Indices können jederzeit erstellt werden und gelten ab diesem Zeitpunkt.
- DROP
  - Indices können jederzeit gelöscht werden.

Datendefinition

Zusammenfassung – 1

- SQL-Befehle für Definition von Daten können innerhalb DB2 jederzeit ausgeführt werden.
- Bei nichtrelationalen Datenbank-Systemen ~~ist~~ war das Hinzufügen von neuen Datenobjekten mit höherem Aufwand verbunden wie
  - Stoppen des Systems, entladen, neu definieren, Daten anpassen, laden
- Änderungen an der Datenbank relativ leicht.
- Verlagerung der Logik aus Programm heraus.

Datendefinition

Zusammenfassung – 2

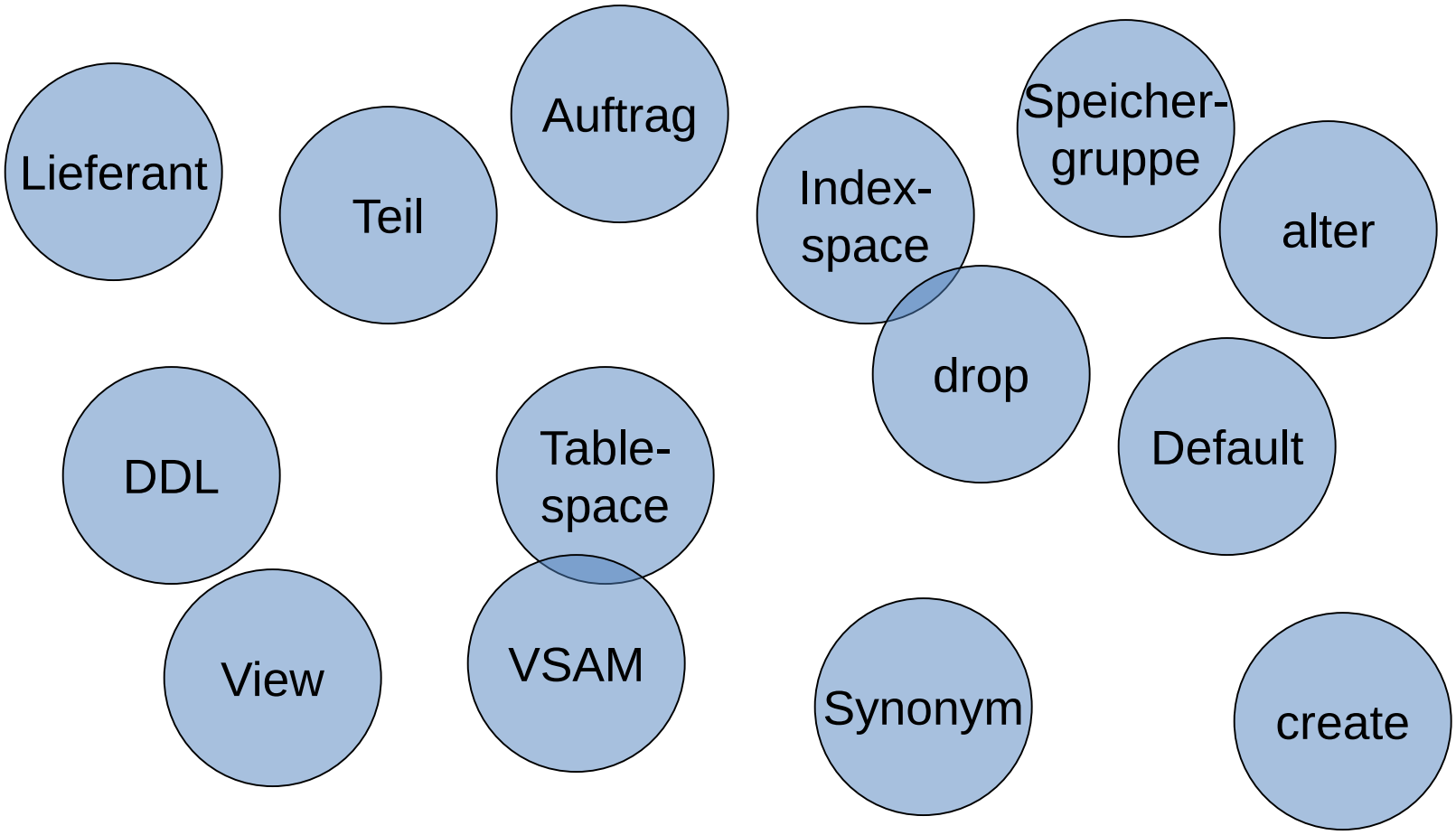
- Es müssen beim Design nicht alle Eventualitäten voraus gesehen werden.
- Das Design muss nicht von Beginn an vollständig und korrekt sein.
- Logisches und physisches Design der Datenbank können getrennt voneinander erfolgen.
- Aber:
  Dies ist kein Freibrief für Schlampereien!!!

Inhalt

- Einführung und Überblick
- Datenbank-Design
- Beispiel-Datenbank
- Datendefinition
- Speicherstruktur
- DB2-Menü – DB2I

Speicherstruktur

Begriffe



Speicherstruktur

DB2-Objekte – Klassifizierung

Table, Index, View, Synonym

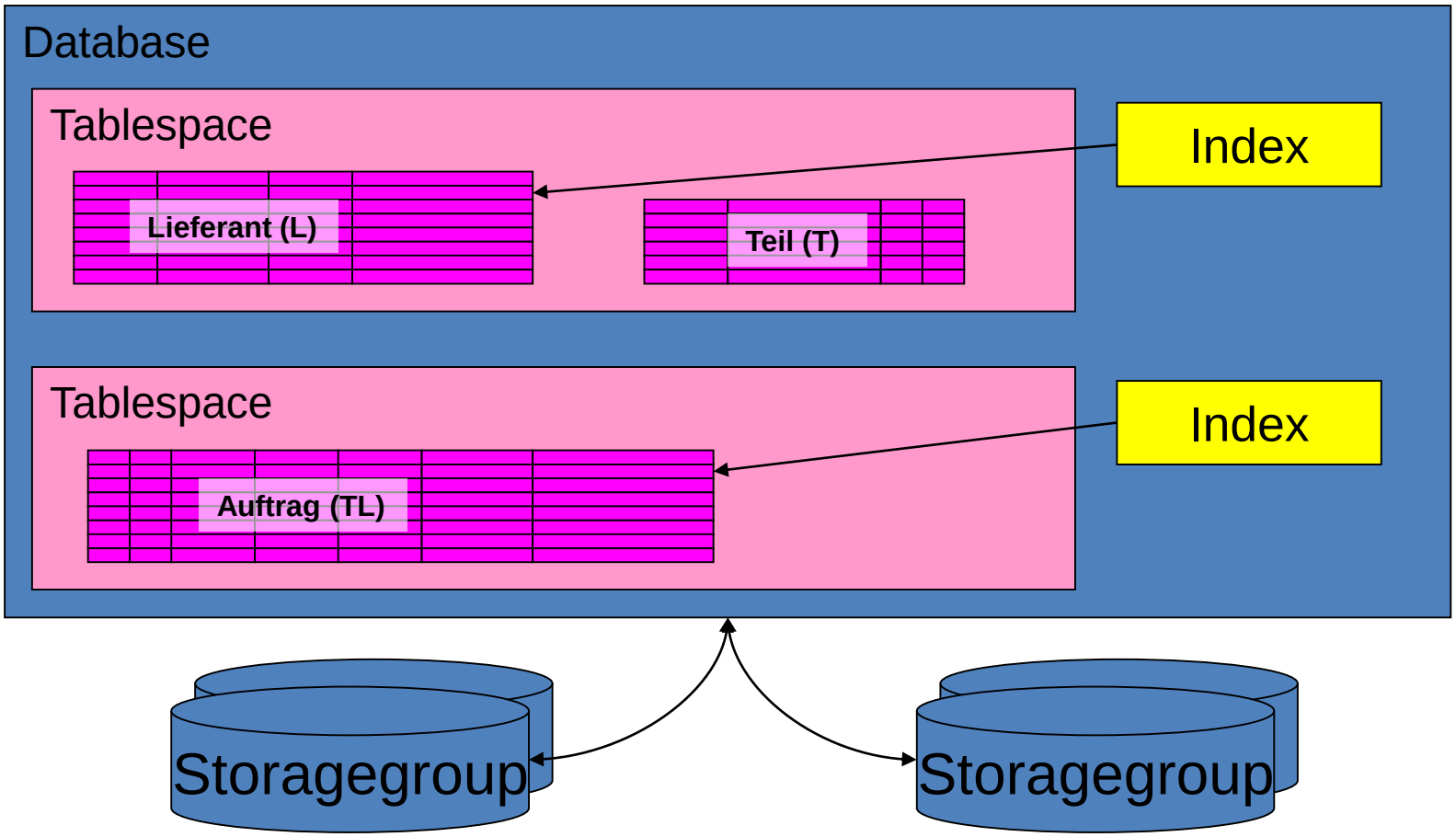
- logische Sicht
  - Definition der DB2-Objekte durch Anwendungs-entwickler, Datenbankadministrator oder Endbenutzer

Tablespace, Database, Storagegroup, Indexspace

- physische Sicht
  - Definition der DB2-Objekte durch Datenbank-administrator oder Systembetreuer

Speicherstruktur

DB2-Objekte – Klassifizierung – Bild



Speicherstruktur

Tablespace – Indexspace

- Tablespace (Tabellenraum)
  - ist ein DB2-interner Name für einen oder mehrere VSAM-Dateien zur Speicherung der Daten
  - enthält die Daten von einer oder mehreren Tabellen
  - ist unterteilt in Pages einer Größe von 4k oder 32k
  - wird auf der Platte immer in 4k-VSAM-CIs gespeichert
- Indexspace
  - ist die Speicherform des Index
  - wird implizit beim CREATE INDEX angelegt

Speicherstruktur

Storagegroup

- ist ein Name für ein oder mehrere Platteneinheiten; eine oder mehrere Platten
- Eine Platte kann mehreren Storagegroups angehören.
- Steuerung bei der Definition des Tablespace (storagegroup-controlled) durch DBA; keine Kenntnisse von VSAM erforderlich; VSAM-Cluster werden automatisch definiert
- Definition einer Storagegroup nur möglich, wenn keine Tablespaces, Indexspaces, Datasets da
- Es gibt auch user-controlled Tablespaces.

Speicherstruktur

Definition der physischen Objekte

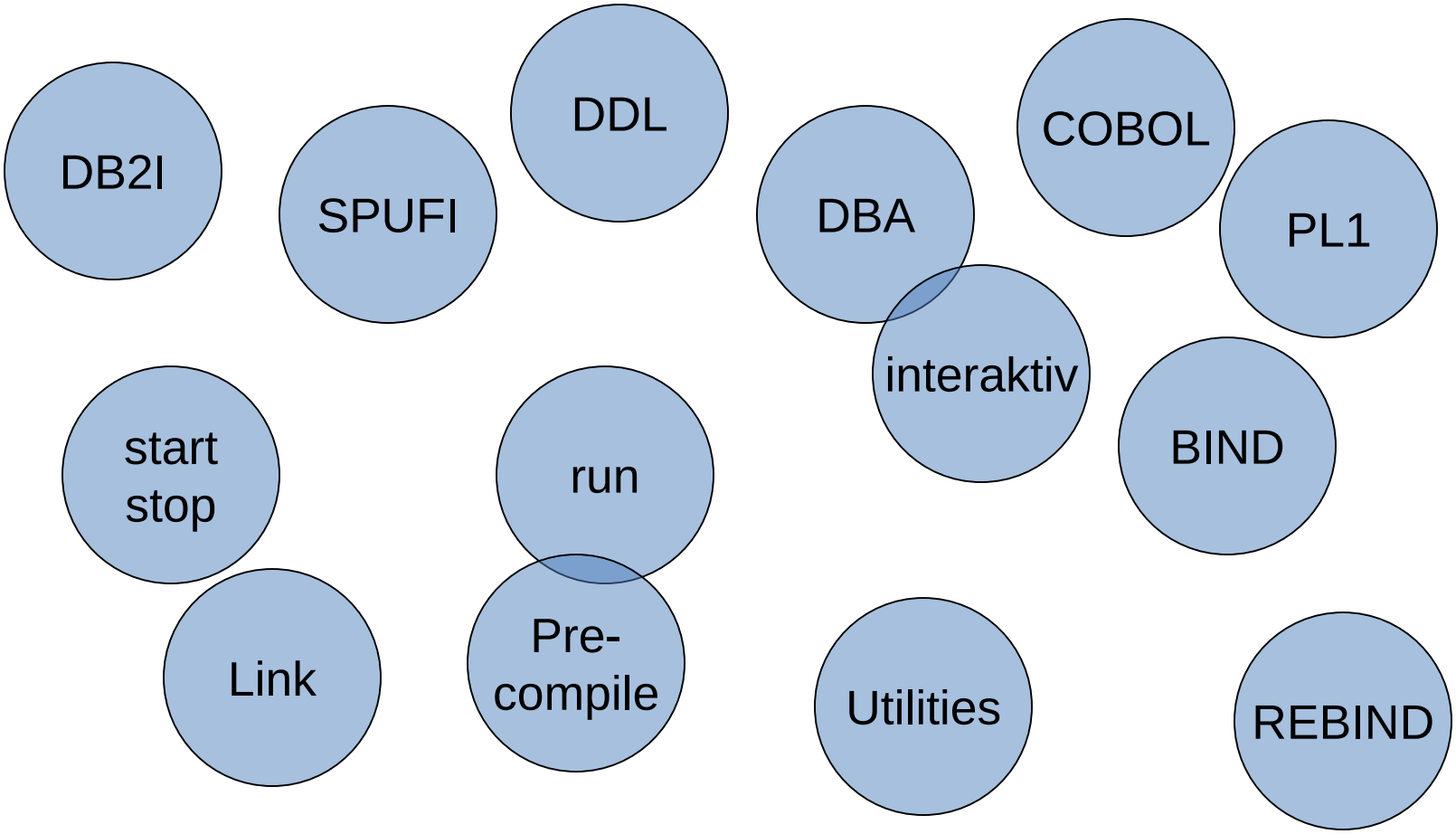
- DDL (Data Definition Language)
- ähnlich der SQL-Sprache für create, alter, drop
- zusätzliche Parameter für detaillierte Angaben zur physischen Speicherung
- DB2 vereinfacht die Definitionen durch sinnvolle (?) Default-Werte

Inhalt

- Einführung und Überblick
- Datenbank-Design
- Beispiel-Datenbank
- Datendefinition
- Speicherstruktur
- DB2-Menü – DB2I

DB2-Menü – DB2I

Begriffe



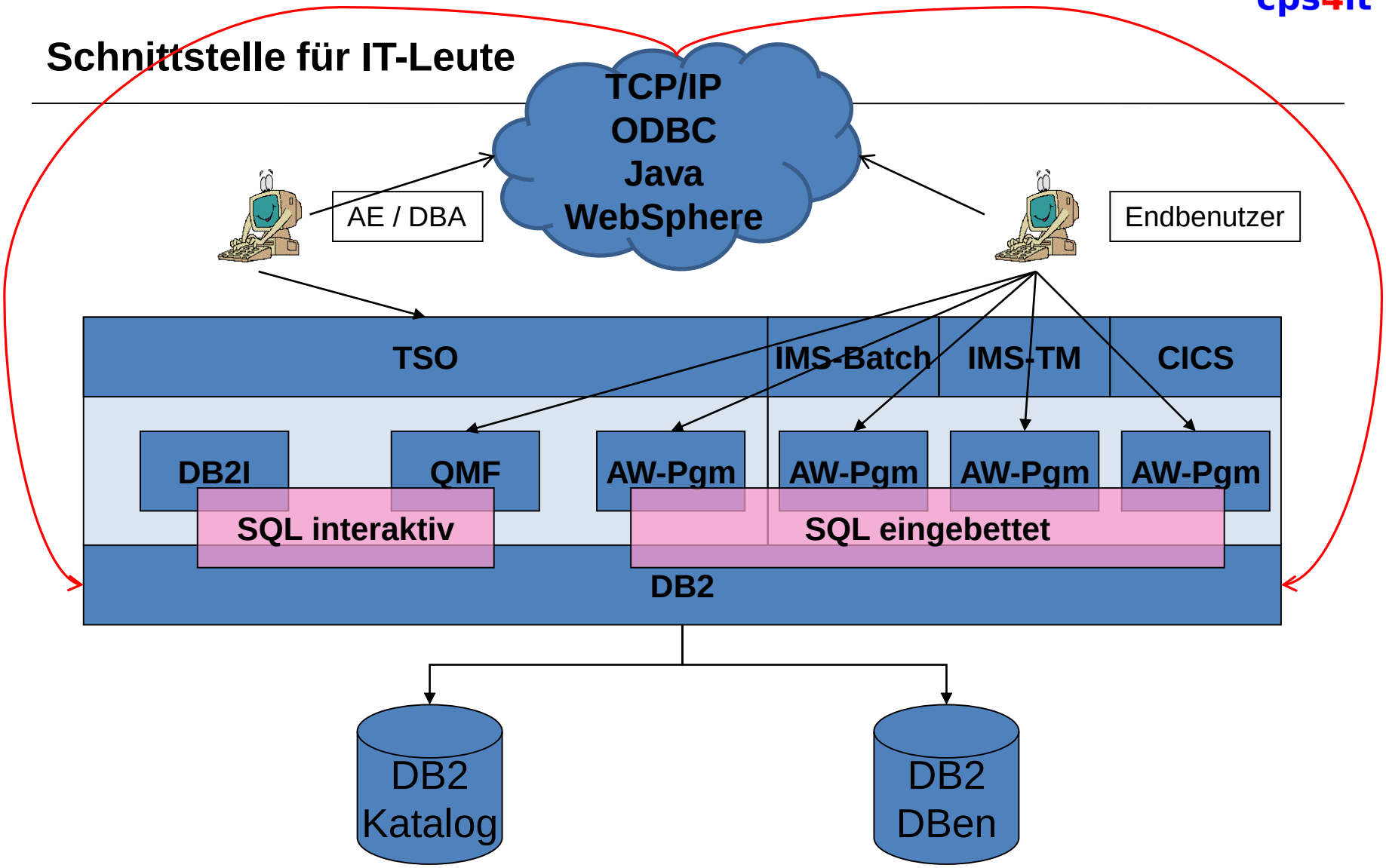
DB2-Menü – DB2I

Schnittstelle für IT-Leute

- interaktives Arbeiten mit DB2
  - DB2I und QMF
- Arbeiten mit Anwendungsprogramm
  - SQL eingebettet im Programm
  - Sprache COBOL, PL1, C, FORTRAN, ASM

DB2-Menü – DB2I

Schnittstelle für IT-Leute



DB2-Menü – DB2I

DB2I – Hauptmenü

```
                          DB2I PRIMARY OPTION MENU          SSID:  DBA1
COMMAND ===>

Select one of the following DB2 functions and press ENTER.

 1  SPUFI                    (Process SQL statements)
 2  DCLGEN                   (Generate SQL and source language declarations)
 3  PROGRAM PREPARATION      (Prepare a DB2 application program to run)
 4  PRECOMPILE               (Invoke DB2 precompiler)
 5  BIND/REBIND/FREE         (BIND, REBIND, or FREE plans or packages)
 6  RUN                      (RUN an SQL program)
 7  DB2 COMMANDS             (Issue DB2 commands)
 8  UTILITIES                (Invoke DB2 utilities)
 D  DB2I DEFAULTS            (Set global parameters)

 P  DB2 OM                   (Performance Monitor)
 C  DC Admin                 (Data Collector Admin)

 X  EXIT                     (Leave DB2I)



 F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
 F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

DB2-Menü – DB2I

DB2I – Hauptmenü

- interaktives Interface zu DB2
  - Fast alle Funktionen des DB2 stehen für den „Online“-Betrieb zur Verfügung.
  - DB2I arbeitet unter der Kontrolle von TSO/ISPF.
  - DB2I erleichtert die Arbeit für System- und Datenbankverwaltung.
  - Unterstützt die Arbeit des Anwendungsentwicklers und erhöht dadurch die Produktivität.
- Aber:
  - Es gibt weitere tolle Tools auf dem Markt.

DB2-Menü – DB2I

DB2I – Hauptmenü – Selektionsmöglichkeiten – 1

- SPUFI (SQL Processor Using File Input)
  - ausführen von SQL-Befehlen (Test)
  - Befehle werden in Datei editiert
  - Datei kann mehrere SQLs enthalten
  - Ausgabedatei enthält Ergebnis und/oder Meldung
  - Ausgabedatei kann weiter benutzt werden
  - SQL-Code für jeden Befehl
  - Anzeige, ob COMMIT / ROLLBACK durchgeführt worden ist

DB2-Menü – DB2I

DB2I – Hauptmenü – Selektionsmöglichkeiten – 2

- DCLGEN
  - aufsetzen und steuern des Deklarations-Generators
- Programmvorbereitung
  - ausführen und steuern von
    - Umwandlung mit Precompiler
    - Umwandlung der Precompilerausgabe
    - Aufruf Linkage Editor
    - Programmausführung
  - Steuerung über Menüs

DB2-Menü – DB2I

DB2I – Hauptmenü – Selektionsmöglichkeiten – 3

- BIND/REBIND/FREE
  - erzeugen Anwendungsplan (BIND)
  - neu binden eines Anwendungsplans, wenn das Umfeld dieser Anwendung verändert wurde
  - Löschen eines Plans
- Programmausführung
  - Ausführung von Programmen (nur TSO, nicht IMS oder CICS o.ä.)

DB2-Menü – DB2I

DB2I – Hauptmenü – Selektionsmöglichkeiten – 4

- Operator-Befehle
  - Menü zur Administration des DB2-Umfeld der Anwendungen
    - Start Datenbank
    - Stopp Datenbank
    - etc.
- Utilities
  - LOAD, CHECK, QUIESCE, REPORT, COPY, RECOVER, REORG, RUNSTATS

DB2-Menü – DB2I

SPUFI – Menü

```
                              SPUFI                                SSID: DBA1
===>

Enter the input data set name:          (Can be sequential or partitioned)
 1  DATA SET NAME ... ===>
 2  VOLUME SERIAL ... ===>              (Enter if not cataloged)
 3  DATA SET PASSWORD ===>              (Enter if password protected)

Enter the output data set name:         (Must be a sequential data set)
 4  DATA SET NAME ... ===>

Specify processing options:
 5  CHANGE DEFAULTS   ===> YES         (Y/N - Display SPUFI defaults panel?)
 6  EDIT INPUT ...... ===> YES         (Y/N - Enter SQL statements?)
 7  EXECUTE ......... ===> YES         (Y/N - Execute SQL statements?)
 8  AUTOCOMMIT ...... ===> YES         (Y/N - Commit after successful run?)
 9  BROWSE OUTPUT ... ===> YES         (Y/N - Browse output data set?)

For remote SQL processing:
10  CONNECT LOCATION  ===>


 F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
 F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

DB2-Tools bei R+V

cps4it

Einstieg und Überblick

- DB2I
- BMC
- FileAid DB2
- Plan Analyzer
- QuickStart

Einführung

Übung(en)

- Kapitel 1.1 Test der User-Iden
- Kapitel 1.2 Bibliothek anlegen
- Kapitel 1.3 Einstieg in DB2I testen

- Kapitel 2.1 Tabellen erstellen
- Kapitel 2.2 Index definieren
- Kapitel 2.3 Synonym definieren

Inhalt

- Einführung und Überblick
- Datenbank-Design
- Beispiel-Datenbank
- Datendefinition
- Speicherstruktur
- DB2-Menü – DB2I