Datenbanken - Portfolio

I. Lexikon

1NF: Abk. f. 1. Normalform. Eine ? Relation ist in 1NF, wenn sie keine mengenwertigen ? Attribute enthält, also z.B. Person = ... + Konto1 + Konto2 + ... ist nicht in 1NF.

2NF: Abk. f. 2. Normalform. Eine ? Relation ist in 2NF, wenn sie in 1NF ist und jedes ? Nichtschlüsselattribut ? voll funktional abhängig von allen Schlüsselkandidaten ist.

2PL: Abk. f. Zwei-Phasen-Sperrprotokoll. Das 2PL ist ein Hilfsmittel, um konkurrierende Transaktionen zu ? isolieren. Die Implementation erfolgt mit Hilfe von ? Sperren. Die Transaktion zerfällt unter Anwendung von 2PL in zwei Phasen: 1. Es werden nur Sperren gesetzt; 2. Die Sperren werden wieder freigegeben. Jeder bei Anwendung des 2PL entstehende Schedule ist ? konfliktserialisierbar und damit ? korrekt synchronisiert. Die Synchronisationsprobleme ? lost update, ? inconsistent analysis und ? phantom können nicht mehr auftreten. Zur Vermeidung von Phantomen müssen ganze Tabellen gesperrt werden.

3-Ebenen-Architektur: ANSI / SPARC-Architektur, Study Group on Database Management Systems 1975 beschreibt 1. Externe Ebene des Endbenutzers (der Anwendung), jeder Endbenutzer hat individuelle Datensicht; 2. Konzeptionelle Ebene: Datenmodell des gesamten Informationsbereichs, repräsentiert alle Informationen aus der Datenbank, abstrahiert von der physischen Speicherung, externe Sichten sind Teile der konzeptionellen Sicht; 3. Interne Ebene: tatsächliche physische Speicherung der Daten.

3NF: Abk. f. 3. Normalform. Eine ? Relation ist in 3NF, wenn sie in 2NF ist und kein ? Nichtschlüsselattribut ? funktional abhängig von einem anderen Nichtschlüsselattribut ist.

4NF: Eine Relation ist in 4NF, wenn sie in BCNF ist und alle ? mehrwertig funktional abhängigen (MFD) Attribute auch ? funktional abhängig sind. D.H. alle MFD müssen durch die Schlüsselkandidaten festgelegt sein (und nicht durch Nichtschlüsselattribute).

5NF: Höchste Normalform, bei der alle funktionalen Abhängigkeiten beseitigt sind. Eine Relation ist in 5NF, wenn sie in 4NF ist und jede Produktabhängigkeit bereits durch die Schlüsselkandidaten von R definiert ist (und nicht erst durch Nichtschlüsselattribute).

abhängig: Eine Transaktion Ti in einem Schedule heißt a., wenn eine ihrer Operationen Ai mit einer Operation Ak der T. Tk in ? Konflikt steht, und Ak im Schedule vor Ai steht. Abhängigkeiten können mit einem Abhängigkeits- oder ? Konfliktgraphen bestimmt werden.

Abhängigkeitsgraph: ? Konfliktgraph.

ACID-Prinzip für Transaktionen: Transaktionen sind atomic , consistent, isolated und durable.

Anomalie: Man unterscheidet drei Formen der A. 1. Update-Anomalie: Die Änderung eines Attributs verursacht die Änderung mehrerer Tupel; 2. Lösch-Anomalie: Ein nicht benötigtes ? Tupel verschwindet aus einer ? Relation (z.B. Student durch Löschen seines letzten Vorlesungsbesuchs); 3. Einfüge-Anomalie: Ein Tupel kann ohne die Belegung bestimmter Attribute nicht eingefügt werden (z.B. Student ohne Vorlesungsbesuch).

Assoziativer Entitätstyp: Ein Beziehungstyp, der wie ein normaler Entitätstyp Attribute enthält.

Attribut: Beschreibende Eigenschaft eines Entitätstyps. Jedes Attribut a besitzt einen Wertebereich W_a. Z.B. $a \in$ {Informatik, Architektur, ...}. Ein Attribut kann auch zusammengesetzt sein, d.h. es ist in mehrere Attribute aufteilbar, z.B. $b \in$ {(Robert, Koch), (Albert, Einstein), ...}.

ausstehende Bestätigung: ? dirty read.

BCNF: Abk. f. Boyce-Codd-Normalform. Eine Relation ist in BCNF, wenn sie in 3NF ist und wenn jede ? Determinante ein ? Schlüsselkandidat ist. Gibt es nur einen Schlüsselkandidaten, so entspricht die BCNF der 3NF.

Beziehung: Spezielle Entität; beschreibt eine Beziehung zwischen Entitäten, z.B. ‚Hubert Meier' besucht die Vorlesung ‚Datenbanken'.

Beziehungstyp: Spezieller Entitätstyp, Typ von Beziehungen, z.B. Relationstyp (Vorlesungs-)Besuch.

Checkpoint Record: ? Sicherungspunkt.

Chen, Notation von: Darstellungsweise eines ? ERM in der die Entitätstypen durch Rechtecke und die Beziehungstypen durch Rauten beschrieben werden. Die ? Kardinalität der Beziehung kann entweder in Min-Max-Notation oder in Alternativer Notation (s.u.) erfolgen.

Datenbankobjekt: Bei der Implementation von Datenbanken mit Hilfe von Digitalcomputern bezeichnet man sowohl Tupel von Wertepaaren als auch die gesamte Tupelmenge einer Relation als D. Siehe auch ? Tabelle.

DBMS: Abk. f. Database Management System, Zugriff auf die Datenbank regelnde Software. Verwendet ? DML und ? DDL.

DDL: Abk. f. Data Definition Language, dient zur Definition und Deklaration der Datenbankobjekte.

Deadlock: Ein D. (Blockade) tritt auf, wenn Transaktionen nach dem Setzen von ? Sperren wechselseitig aufeinander warten. Man erkennt sie an Zyklen im ? Wartegraphen. Sie werden aufgelöst durch das Rücksetzen einer der Transaktionen.

Determinante: Eine Determinante einer Relation ist ein (zusammengesetztes) ? Attribut, von dem ein weiteres Attribut voll funktional abhängig ist.

dirty read: Eine Transaktion liest einen Wert aus der Datenbank, der von einer anderen zwar verändert, jedoch noch nicht ? commited wurde. Durch ein Rollback kann der gelesene Wert ungültig werden.

Division: Als D. bezeichnet man die algebraische Mengenoperation zwischen der Dividendenrelation R und der Divisorrelation S, wobei R nur Attribute besitzen darf, die auch S besitzt. Die D. bildet dabei die Tupel aus S auf eine neue Relation T ab, sodass T das (zusammengesetzte) Attribut t aus den Attributen von R ohne die Attribute aus S besitzt. Es werden dabei nur die Tupel aus

Datenbanken - Portfolio

R projiziert, welche vollständig in S enthalten sind. Die Division lässt sich als den ? primitiven Operationen darstellen: $T = R \div S = p_t(R) - p_t(p_t(R) \times S - R)$.

DML: Abk. f. Data Manipulation Language, unterstützt die Be- und Verarbeitung von Datenbankobjekten.

Entität: Realer oder begrifflicher Gegenstand, z.B. der Student Hubert Meier.

Entitätstyp: Typ von Entitäten, beschrieben durch Attribute. Für jede Instanz des Typs ist den Attributen jeweils ein Attributwert zugeordnet. Z.B. Entitätstyp Student mit den Attributen Matrikelnr., Name, Fach.

Entity Relationship Modell: Abk.: ERM, wörtl. übersetzt Ding-Beziehungs-Modell, beschreibt in einer adäquaten Sprache die Beschaffenheit von ? Entitätstypen und ? Beziehungstypen, also die Art und Beschaffenheit der Entitäten sowie der Beziehung zwischen diesen.

ERM: Abk. f. ? Entity Relationship Modell.

final-state-equivalence: Zwei Schedules gleicher Transaktionen sind f.-s.-e., wenn sie die Datenbank (egal auf welchem Weg) in denselben Zustand überführen, unabhängig vom Ausgangszustand der Datenbank.

final-state-serializable: Ein Schedule heißt final-state-s., wenn er final-state-equivalent zu einem seriellen Schedule ist..

Fremdschlüssel: Sei R eine Relation. Dann bezeichnet man eine Teilmenge der Attribute von R als F., im folgenden FK genannt, wenn es eine zweite Relation mit dem ? Schlüsselkandidaten CK gibt, so dass der Wert von FK gleich dem Wert von CK ist und dieser Wert sich nicht ändert.

funktional abhängig: Das Attribute b ist vom (zusammengesetzten) Attribut a funktional abhängig, wenn der Wert von b in jedem Tupel durch den Wert von a festgelegt ist ($a \rightarrow b$). Die Funktion muss dabei nicht unbedingt trivial sein, deswegen liegt das Finden der f. A. in ? NPV.

Globaler Fehler: Betrifft mehrere Transaktionen. Entweder ein System- oder ein Fehler im Speichermedium, der Teile der Datenbank auf dem Speichermedium beschädigt.

inconsistent analysis: Eine Transaktion T1 verändert mehrere Werte eines ? Datenbankobjekts. Dieselben Werte werden von einer zweiten Transaktion T2 gelesen, teils vor, teils nach der Veränderung. Das Objekt ist für T2 inkonsistent.

Instanz: 1. Ausführende Einheiten, welche auf ? Datenbankobjekte zugreifen wie z.B. Benutzer, Prozesse innerhalb derselben Datenbank oder andere Datenbanken (oder Applikationen); 2. ? Tupel von ? Attributen aus einer gültigen Wertemenge, welche zu einer ? Relation gehören.

Integrität: Sicherstellung von Korrektheit und Vollständigkeit des Datenbestandes. Prüfung, ob Änderungen ausgeführt werden dürfen. Man unterscheidet drei Formen der Integrität: 1. Wertebereichsintegrität: der Wert eines Attributs muss innerhalb des ihm zugehörigen Wertebereichs liegen; 2. Entity-Integrität: Keine Komponente des Primärschlüssels darf den Wert NULL haben; 3. Referentielle Integrität: Der Wert eines Fremdschlüssels muss auf eine existente Entität zeigen, d.h. in der Zielrelation muss ein Tupel mit diesem Wert existieren.

isolation(-levels): Transaktionen heißen isoliert, wenn sie unter Verwendung von ? Sperrprotokollen so synchronisiert sind, dass sie einander nicht beeinflussen. Man unterscheidet zwischen unterschiedlichen i.-levels: ? read uncommitted, ? read committed, ? repeatable read, ? serializable.

Join: Als J. bezeichnet man die Verknüpfung zweier Relationen R und S auf den Attributen r aus R und s aus S über eine logische Formel P, z.B. Gleichheit. Der Join ist keine ? primitive Operation, lässt sich deshalb beschreiben durch eine ? Selektion auf ein ? Produkt: $R \bowtie_P S = s_P(R \times S)$.

Kardinalität: Grad eines Beziehungstyps .

Konflikt(-paar): Seien Ai und Ak Operationen der Transaktionen Ti bzw. Tk. Ai und Ak stehen in K., wenn sie auf dasselbe Datenobjekt zugreifen und mindestens eine der beiden eine Schreibeoperation ist. Ai und Ak nennt man Konfliktpaar.

Konfliktgraph: Die Knoten des (gerichteten) K. sind die Transaktionen T1, ..., Tn eines Schedules. Von Ti führt genau dann eine Kante nach Tk, wenn Tk von Ti ? abhängig ist. Wenn der K. zyklenfrei ist, ist der Schedule ? view-serializable. Man erhält eine äquivalente ? serielle Ausführung des Schedules durch eine topologische Sortierung des Graphen. Die Zyklenfreiheit ist für die ? View-Equivalence aber nicht zwingend.

konfliktserialisierbar: Ein Schedule ist k. und damit ? view-serializable, wenn sein Konfliktgraph zyklenfrei ist.

korrekt synchronisiert: Ein System von gleichzeitig ablaufenden Transaktionen ist k. s., wenn es eine ? final-state-äquivalente serielle Ausführung der Transaktionen gibt.

Lesen nicht bestätigter Daten: ? dirty read.

Lock: Ein algorithmisches Konstrukt zur Vermeidung von konkurrierenden Zugriffen auf Ressourcen. Man unterscheidet X-Locks für Schreib- und S-Locks für Leseoperationen. Verträglichkeit (s.u.).

Lokaler Fehler: Fehler, der nur die Transaktion betrifft, in der er passierte.

lost update problem: ? non repeatable read.

mehrwertig funktional abhängig: Abk. MFA. Die MFA ist eine Verallgemeinerung der ? funktionalen Abhängigkeit. Gegeben seien die Attribute a, b und c. b ist MFA von a, wenn für alle b ein Tupel (a, c) existiert, sodass b nur voll ? funktional abhängig von a, jedoch funktional unabhängig von c ist.

Nichtschlüsselattribut: Ein Attribut heißt N.-S.-A., wenn es mit allen ? Schlüsselkandidaten eines Entitätstyps leere Schnittmenge besitzt.

nicht wiederholbares Lesen: ? non repeatable read.

non repeatable read: Zwei Transaktionen T1, T2 greifen zeitversetzt auf dasselbe Objekt zu. Nachdem beide den Wert gelesen haben, verändert T1 den Wert

Datenbanken - Portfolio

(d.h. T2 kann den Lesevorgang nicht ? viewäquivalent wiederholen), jedoch arbeitet T2 noch mit dem veralteten Wert weiter und überschreibt die Veränderung von T1 (d.h. diese Änderung ist verloren).

Normalform: N. werden eingesetzt um in einem ERM ? Anomalien zu vermeiden. Es sind ? 1NF, 2NF, 3NF, BCNF, 4NF und 5NF definiert.

Normalisierung: Der Prozess, einen Entitätstyp in Normalform überzuführen. Man erreicht die Normalformen durch den sog. Normalisierungskalkül, zu dem u.a. die Dekomposition von Codd (3NF) und die Synthese von Bernstein und Maier gehört (3NF). Diese Algorithmen laufen in polynomaler Zeit, jedoch ist das Problem, dass diesen die ? funktionalen Abhängigkeiten bereits zur Verfügung stehen müssen.

NPV: Abk. f. nicht polynomal vollständig, eine Klasse aus der Komplexitätstheorie, welche diejenigen Probleme enthält, die nicht in polynomaler Zeit lösbar sind, sondern erst in exponentieller oder höherer Zeit lösbar sind.

Operationale Integrität: Vermeidung von Fehlern durch gleichzeitigen Zugriff mehrerer Benutzer; Kontrolle von (parallelen) Transaktionen.

Optimistische Synchronisation: Man geht davon aus, dass ? Konflikte nur selten auftreten und verzichtet auf ? Sperren. Dies erhöht den Grad von Parallelität und verkürzt Wartezeiten. Man prüft erst vor dem Abschluß einer Transaktion, ob Konflikte aufgetreten sind mittels 3 Phasen: Lesephase, Valisierungs- und Schreibphase. Die Konfliktlösung besteht aus Wiederholung der Transaktion.

pessimistische Synchronisation: Man rechnet mit häufigen Konflikten und sperrt die Objekte so, dass diese immer ? korrekt synchronisiert sind bzw. ? Serialisierbarkeit immer gewährleistet ist. Dies führt zu geringem Datendurchsatz.

phantom: Eine Transaktion T1 liest mehrfach die gleichen Attribute aus einer Tabelle. Zwischen den Lesevorgängen fügt eine Transaktion T2 (P.-)Werte in die Tabelle ein, welche erst bei späteren Lesevorgängen von T1, jedoch nicht anfangs erscheinen.

Primärschlüssel: Wenn es eine Relation gibt, die mehr als einen ? Schlüsselkandidaten enthält, so legt man einen davon als P. fest. Ansonsten ist der einzige vorhandene Schlüsselkandidat P.

primitive Operationen: Eine Algebra heißt relational vollständig, wenn sie sich durch die fünf primitiven Operationen Vereinigung, Subtraktion, Produkt, Selektion und Projektion beschreiben lässt.

Produkt: Als P., auch kartesisches P., „Kreuzprodukt“, zweier Relationen R mit dem (zusammengesetzten) Attribut r und S mit dem Attribut (zusammengesetzten) s bezeichnet man die Relation T aus den Tupel (r, s) mit allen Kombinationen von r und s.

produktabhängig: Die Produktabhängigkeit ist eine Verallgemeinerung der mehrwertigen funktionalen Abhängigkeit. Eine Relation R ist p., wenn sie dem Produkt aus beliebigen (zusammengesetzten) Attributen von R entspricht.

Projektion: Als P. bezeichnet man die Auswahl bestimmter Attribute aus einer Relation R in eine neue Relation S. Während die ausgewählten Tupel in R mehrfach vorkommen können, werden diese in S nur noch einfach abgebildet analog z.B. zur optischen P. eines räumlichen Körpers auf eine flache Leinwand, bei der hintereinanderliegende Punkte ebenfalls nur noch einfach dargestellt werden. Die algebraische Darstellung lautet: $p_a(R)$, wobei a das p. Attribut ist und R die Tupelmenge der Relation.

read committed: ? isolation-level, der das Lesen „unbestätigter“ Daten vermeidet (? dirty read). Beim Lesen werden ? S-Locks benötigt, die sofort nach Beendigung des Lesevorgangs aufgehoben werden. Beim Schreiben sind ? X-Locks zwingend; deren Freigabe erfolgt allerdings erst zum Transaktionsende.

read uncommitted: ? isolation-level, bei dem von der optimistischen Annahme ausgegangen wird, die Transaktionen würden nie konkurrieren. Daher werden keine Lesesperren angelegt. Dem Applikationsprogrammierer wird die volle Verantwortung für die ? Integrität der Daten übertragen. Es kann zu sog. ? dirty read, ? non-repeatable read und ? phantoms kommen;.

redo: Beheben von Änderungen im Datenbestand, die von einer nur teilweise durchgeführten Transaktion stammen. Nur so kann die Konsistenz des Datenbestands sichergestellt werden.

Referentielle Integrität: Es kann kein Objekt gelöscht werden, das noch existiert. In der Datenbanksprache SQL teilt die Anweisung REFERENCES dem System mit, dass eine Referenzelle ? Integrität aufrecht erhalten werden soll.

Relation: Zusammenfassende Bezeichnung für ? Entitätstypen und ? Beziehungstypen. S.a. ? Entity Relationship Modell.

repeatable read: ? isoaltion-level, der garantiert, daß ein gelesener Wert nicht von einer anderen Transaktion überschrieben wird und damit das ? Synchronisationsproblem ? non-repeatable-read vermeidet. Implementation durch ? striktes 2PL und Schreib- bzw. Lesesperren auf einzelne Datensätze oder Seiten.

Schlüsselkandidat: Ein (zusammengesetztes) Attribut k, welches Teilmenge des Entitätstyps E ist, heißt S. wenn k die Instanzen von E eindeutig und minimal bestimmt. Das Finden der Schlüsselkandidaten ist in der Komplexitätsklasse NPV.

Schnittmenge: Die S. ist keine ? primitive Operation: $R \cap S = (R \cup S) - (R - S) - (S - R)$.

Schwacher Entitätstyp: Entitätstyp, der ohne einen sog. Zusammengesetzten Entitätstyp nicht existieren kann.

Selektion: Als S. bezeichnet man die Auswahl einer Teilmenge aus den Tupeln über alle Instanzen eines Entitätstyps anhand des sog. Selektionskriteriums. Die algebraische Schreibweise lautet: $s_a(R)$, wobei a die Auswahlbedingung und R die Tupelmenge der Relation ist.

Semantische Integrität: Regeln, die durch die Interpretation der in der Datenbank enthaltenen Informationen durch den Benutzer festgelegt werden.

seriell: Ein Schedule heißt s., wenn er eine serielle Ausführung von Transaktionen ist.

Datenbanken - Portfolio

serializable: Der einzige ? isolation-level, der vollständige Isolation von Transaktionen und selbst das Nicht-Auftreten von ? Phantomen garantiert. Implementierung durch striktes 2PL und ? Sperren auf gesamte ? Tabellen.

Shadowing: Alle Zugriffe auf die Datenbank erfolgen indirekt über Zeiger, die in einem Katalog gehalten werden. Will eine Transaktion ein Datenobjekt verändern, so wird eine Kopie angelegt. Zum Transaktionsende werden nur die Zeiger der veränderten Objekte auf die Kopie umgelenkt.

Sicherungspunkt: Wird vom DBMS automatisch gesetzt. Dabei werden die Puffer in die physische Datenbank und ein Checkpoint Record (Liste der gerade laufenden Transaktionen) geschrieben.

Singulärer Entitätstyp: ? Schwacher Entitätstyp.

S-Lock: ? Lock.

Sperre: ? Lock.

Sperrgranularität: Definiert die kleinste physische Sperreinheit. Diese ist oft eine Seite, wenn auch einzelne Datensätze gesperrt werden könnten (z.B. Oracle). Auch das Sperren ganzer Tabellen ist möglich.

Sperrprotokoll: Algorithmus zur Implementation von ? Isolation. Siehe ? 2PL und ? striktes 2PL.

striktes 2PL: Wie beim ? 2PL, jedoch werden alle Sperren erst am Ende der ? Transaktion wieder freigegeben. Vermeidet zusätzlich zu 2PL ? uncommited dependency.

Subtraktion: Als S. bezeichnet man die algebraische Mengenoperation zwischen der Subtrahendenrelation R und der Minuendenrelation S, wobei R und S dieselben Attribute besitzen müssen. Die Subtraktion liefert alle in R enthaltenen Tupel ohne die in S enthaltenen. Die algebraische Darstellung ist $R - S$.

Synchronisationsproblem: Der gleichzeitige Zugriff mehrerer Transaktionen auf die Datenbank muß synchronisiert werden, um folgende Probleme zu vermeiden: ? non-repeatable-read, ? dirty-read, ? inconsistent-analysis, ? phantom.

Systemfehler: Absturz des Datenbank-Managers, Stromausfall, ... Betrifft alle laufenden Transaktionen, verursacht aber keine physischen Schäden auf dem Speichermedium der Datenbank.

Tabelle: Die gesamte Tupelmenge einer Relation wird insb. bei Implementationen von Datenbanken (z.B. Microsoft SQL, MySQL, Oracle, Postgres) als Tabelle bezeichnet wegen der naheliegenden Darstellung der Tupelmenge in einem mehrspaltigen Format, wobei jede Spalte einem Attribut entspricht und jede Zeile einem Tupel aus der Tupelmenge.

Transaktion: Man spricht bei einer Übertragung von Informationen vom Benutzer bzw. von einer Anwendung zur Datenbank und umgekehrt von T. Als T. werden auch Informationsübertragungen von verschiedenen Prozessen innerhalb einer Datenbank durch verschiedenen Prozesse und zwischen verschiedenen Datenbanken bezeichnet. Weil nun die Reihenfolge, in der T. Ausgeführt werden, eine erhebliche Rolle spielen, werden ? Sperrprotokolle mit Hilfe von sog. ? Locks implementiert um einen ? Isolationslevel der T. gegeneinander zu erreichen.

Tupel: Mit T. bezeichnet man die Werte der Attribute der Instanz einer ? Relation.

UML: Abk. f. Unified Method Language.

uncommitted dependency: ? dirty read.

verlorene Änderung: ? non repeatable read.

view-equivalence: Zwei Schedules derselben Transaktionen sind v.-e., wenn sie ? final-state-equivalent sind und die Transaktionen T_i jeweils bei beiden Schedules dieselben Werte aus der Datenbank lesen.
wenn sowohl alle Zugriffe die gleichen Informationen aus der Datenbank lesen als auch zum gleichen Ergebnis führen (? final-state-equivalence).

view-serializable: Ein Schedule heißt view-s., wenn er view-equivalent zu einem seriellen Schedule ist..

voll funktional abhängig: Das Attribut b ist v. f. a. vom Attribut a, wenn b ? funktional abhängig ist, aber wenn b nicht funktional abhängig von einer echten Teilmenge von a ist.

Wartegraph: Ein gerichteter Graph, der der Erkennung von ? Deadlocks dient. Knoten sind die Transaktionen T1, ..., Tn eines Schedules. Hat Ti ein ? Lock auf ein Datenbankobjekt angefordert und kann dieses Lock nicht erteilt werden bis Tk ein Lock auf das Objekt freigibt, wird eine Kante von Ti nach Tk in den Graphen eingefügt.

Wertemenge: ? Attribut.

X-Lock: ? Lock.

Zusammengesetzter Entitätstyp: Ein spezieller Entitätstyp, von dem ein Schwacher Entitätstyp abhängig ist.

Zwei-Phasen-Sperrprotokoll: ? 2PL.

II. Hauptteil

1. Semantische Datenmodellierung mit Entity Relationship Modellierung

a. Bestandteile:

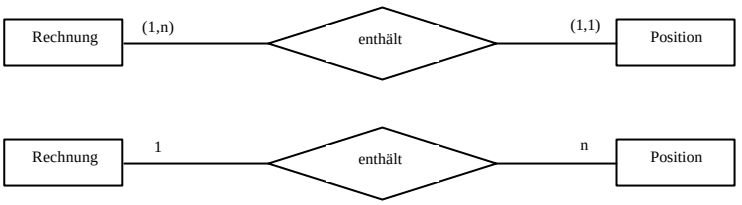
- Definition von Entitäts- und Relationstypen
- Entity-Relationship-Diagramme
- Semantische Beschreibung der Modellelemente in einem Datenkatalog/Data Dictionary

Dient der Modellierung des konzeptionellen Schemas. Begriffe: Entität, Entitätstyp, Attribut, Beziehung, Beziehungstyp.

b. Modellnotation nach Chen

Der Grad (die Kardinalität, Multiplizität) eines Relationstyps wird durch Zahlen dargestellt:

- Min-Max-Notation: Eine Rechnung enthält mehrere, aber mindestens eine Position. Eine Rechnungsposition ist in genau einer Rechnung enthalten.
- Alternative:



c. Schlüsselkandidaten und Primärschlüssel

Gegeben sei ein Entitätstyp E mit den Attributen a_1, a_2, ..., a_n.
Definition: Ein zusammengesetztes Attribut a ist eine Teilmenge der Attribute von E. Die in a enthaltenen Attribute heißen Komponenten.
Definition: Ein zusammengesetztes Attribut k von E heißt Schlüsselkandidat, wenn gilt:

- Eindeutigkeit: Sind E1 und E2 Instanzen von E und hat k für E1 denselben Wert wie für E2, so ist E1=E2.
- Minimalität: Keine Komponente von k kann weggelassen werden, ohne die Eindeutigkeit zu verletzen.

Für jeden Entitätstyp wählt man einen Schlüsselkandidaten als Primärschlüssel aus, der im System zur eindeutigen Identifikation verwendet wird.
Für Relationstypen setzt sich der Primärschlüssel aus den Primärschlüsseln der beteiligten Entitätstypen zusammen. Das ist Standard und wird nicht explizit angegeben.

d. Vererbung

Ein Subtyp besitzt (erbt) alle Attribute und Relationstypen des Supertyps. Er kann zusätzliche Attribute besitzen und in zusätzlichen Relationstypen enthalten sein.

e. Singuläre (schwache) Entitäts- und Relationstypen

Ein Entitätstyp E heißt schwach (singulär), wenn jede Instanz e von E ein privater und abhängiger Teil einer übergeordneten Entität f von F ist.
Privat heißt: e ist Teil von und nur von f und dieses f kann nicht geändert werden.
Abhängig heißt: e kann ohne das zugehörige f nicht existieren.
Der Beziehungstyp zwischen den Entitäten heißt schwacher (singulärer) Beziehungstyp. F heißt dann zusammengesetzter Entitätstyp.
Primärschlüssel schwacher Entitätstypen setzt man aus dem Primärschlüssel der zusammengesetzten Entität und einer Erweiterung zusammen (z.B. @Rechnung, @Position).

f. Eine BNF-Syntax für Katalogeinträge

$=$	ist äquivalent zu	
$+$	und (Sequenz)	$A = B + C$
$[\ldots\|\ldots]$	entweder ... oder (Selektion)	$A = [B \mid C]$
$\{\ldots\}$	Wiederholung (Iteration)	$A = \{B\}$
$(\ldots)$	Option	$A = B + (C)$
...	Kommentar	$A = X$ *großes X* $+ Y$
<...>	kommentierende Ergänzung	Crew=<pilot>name+<copilot>name
"..."	Literal	A=["groß"|"klein"]
@	Primärschlüssel	@FlugNr

g. Zur Dokumentation von Datenmodellen

ER-Diagramm(e)

Zu jedem Entitäts-/Beziehungstyp/Tabelle:
- Kommentar, semantische Beschreibung
- Liste der Attribute
- Primärschlüssel bei Entitätstypen
- Primär- und Fremdschlüssel bei Tabellen

Zu jedem Attribut:
- Kommentar, semantische Beschreibung
- Wertebereich
- Datentyp (einschließlich Feldlänge, z.B. char(20))

2. Das relationale Modell

a. Definition Relation

Gegeben seien Mengen W1, W2, ..., Wn. Eine Relation R über W1, W2, ..., Wn besteht aus einem Relationenkopf und einem Relationenkörper. Der Relationenkopf ist eine feste Menge {A1, ..., An} von Attributen. Der Wertebereich von Ai ist Wi. Der Relationenkörper ist eine endliche Menge von Tupeln. Ein Tupel ist eine Menge {(A1:w1), ..., (An:wn)} von Attribut-Wert-Paaren mit $wi \in Wi\ (1 \leq i \leq n)$.
Der übliche mathematische Begriff der Relation (Definition): Eine Relation R über W1, ..., Wn ist eine Teilmenge von $W1 \times W2 \times \ldots \times Wn$. Die Zahl n wird Grad (Stelligkeit) der Relation genannt (Engl.: arity).
Bemerkungen:
- Relationen im relationalen Datenmodell sind Relationen im mathematischen Sinn.
- Zwei Tupel mit denselben Attributwerten sind gleich.
- Als Wert eines Attributs ist auch NULL erlaubt. NULL heißt: kein Wert.
- Die Anordnung der Spalten ist nicht festgelegt.
- Die Tabellenansicht bringt Probleme mit der Objektidentität mit sich: Identische Zeilen der Tabelle stellen dasselbe Tupel (denselben Datensatz) dar (Vgl. SELECT DISTINCT in SQL).
- Relationen des Entity-Relationship-Modells sind von Relationen des relationalen Datenmodells zu unterscheiden. Für Beziehungen wird deshalb auch „Assoziation“ verwendet.
- Zum relationalen Datenmodell gehören noch Integritätsregeln und Operatoren zur Bearbeitung der Daten (Relationale Algebra, Relationenkalkül).

b. Abbildung von ER-Modellen auf Tabellen

Das normalisierte Datenmodell:
- In allen relationalen Datenbankmodellen sind alle Elemente aus dem Wertebereich W vom selben Typ. Die Werte W sind atomar, d.h. nicht weiter zerlegbar.
- Als Wert eines Attributs sind Relationen (Mengen) nicht erlaubt. M. a. W. die modellierten Relationen müssen in ? 1NF sein.

Als Typen werden von den relationalen DB-Systemen unterstützt: INTEGER, SMALLINT, DECIMAL (Festpunktzahl), FLOAT (Gleitpunktzahl), DOUBLE (Gleitpunktzahl), CHAR(n), VARCHAR(n), LONG VARCHAR(n) (nicht Standard-SQL), DATE (nicht Standard-SQL), TIME (nicht Standard-SQL), BIT(n), LONG RAW (nicht Standard-SQL).

Fremdschlüssel und Integritätsregeln:
Seien R und S Relationen/Tabellen, s ein Schlüsselkandidat von S (z.B. der Primärschlüssel).
Definition: Ein (möglicherweise zusammengesetztes) Attribut r von R heißt Fremdschlüssel in R mit Ziel (S, s), wenn
- immer wenn alle Komponenten von r den Wert NULL oder alle Komponenten einen Wert ungleich NULL haben,
- jeder Nicht-NULL-Wert von r ein Wert von s ist (d.h. eindeutig ein Objekt in S identifiziert).

Integritätsregeln im relationalen Datenmodell:
- Wertebereichsintegrität: der Wert eines Attributs muß aus dem definierten Wertebereich sein.
- Entity-Integrität: Keine Komponente des Primärschlüssels darf den Wert NULL haben.
- Referentielle Integrität: der Wert eines Fremdschlüssels muß gebunden sein (Referenzen dürfen nicht ins Leere zeigen).

c. Abbildung von Vererbung auf relationales Datenmodell

- Split Instance Model

Geschäftspartner = @Nr + Name + Adresse Lieferant = @Nr + Termintreue + foreign key references Geschäftspartner Kunde = @Nr + Bonität + foreign key references Geschäftspartner

Vorteil: Geringerer Speicherverbrauch weil keine NULL Attribute eingefügt werden müssen und z.B. ein Geschäftspartner sowohl Kunde als auch Lieferant sein kann. Einfachere Änderung der Datenstruktur, weil z.B. ein neues Attribut nur einfach eingefügt werden muss. Rollenwechsel z.B. zwischen Kunde und Lieferant einfacher möglich.
Nachteil: Verkompliziert die Implementierung durch mehr Tabellen und komplexere Abfragen (mehr Verknüpfungen). Zugriff auf mehrere Tabellen kann zu Effizienzverringerung führen.

- Leaf Overlap Model

Geschäftspartner = @Nr + Name + Adresse + Termintreue Kunde = @Nr + Name + Adresse + Bonität

Vorteil: Vereinfachte Abfragen z.B. auf Kunden, weil diese in einer eigenen Tabelle gespeichert sind.
Nachteil: Redundante Aufführung der Attribute. Redundanz wenn ein Lieferant auch Kunde ist. Rollenwechsel sind aufwendig. Abfragen z.B. auf Geschäftspartner werden verkompliziert.

- Universal Class Model

Geschäftspartner = @Nr + Name + Adresse + Bonität + Termintreue

Vorteil: Abfragen sind relativ einfach, denn es müssen keine Tabellen verknüpft werden. Es entsteht keine Redundanz z.B. durch Kunden die auch Lieferanten sind.
Nachteil: Viele NULL-Werte in der Tabelle z.B. in den Feldern Bonität und Termintreue. Die Vererbungshierarchie ist in der Datenbank nicht vorhanden, das erschwert den Umgang mit den Tabellen. Die Tabellen erhalten eine Vielzahl von Attributen und werden Speicherintensiv. Ineffizienz durch viele Selektierungen, z.B. zur Abfrage der Kunden.

Mischen der Varianten ist möglich und praktisch sinnvoll.

d. Normalformen

Sei R eine Relation.
Definition: Seien a, b zusammengesetzte Attribute von R. b heißt funktional abhängig von a (kurz a? b), falls für jedes Tupel in R der Wert von b durch den Wert von a bestimmt ist.
Definition: b heißt voll funktional abhängig von a, wenn b funktional abhängig von a, aber nicht funktional abhängig von einer echten Teilmenge von a ist.

Beispiel:

Lieferkondition = ArtikelNr + LieferantenNr + Artikelbezeichnung + Rabatt ArtikelNr + LieferantenNr ? Rabatt + Artikelbezeichnung

Definition: Ein Attribut a von R heißt Nicht-Schlüssel-Attribut, wenn es mit allen Schlüsselkandidaten von R leeren Durchschnitt hat.
Definition: Eine Determinante von R ist ein (zusammengesetztes) Attribut einer Relation, von dem wenigstens ein weiteres Attribut voll funktional abhängig ist.

Definition: Eine Relation ist in 1NF, wenn R keine mengenwertigen Attribute besitzt.
Definition: Eine Relation ist in 2NF, wenn R in 1NF ist und jedes Nichtschlüsselattribut voll funktional abhängig von allen Schlüsselkandidaten ist.
Definition: Eine Relation ist in 3NF, wenn R in 2NF ist und kein Nichtschlüsselattribut funktional abhängig von einem anderen Nichtschlüsselattribut ist.
Definition: Eine Relation ist in BCNF, wenn jede Determinante ein Schlüsselkandidat ist.

Bemerkungen:
- Gibt es nur einen Schlüsselkandidaten, so ist BCNF = 3NF.
- Es sind noch 4. und 5. Normalform definiert.
- $1NF \subseteq 2NF \subseteq 3NF \subseteq BCNF \subseteq 4NF \subseteq 5NF$.

3. Transaktionsmanagement

a. Definition Transaktion

- ist eine Folge von logisch zusammengehörigen Datenmanipulationen (Anweisungen in der DML) in einer Datenbank
- fasst alle für eine aus Sicht der Anwendung elementare (atomare) Operation notwendigen Datenmanipulation zusammen.

Die Integrität eines Datenbestands ist nur gewährleistet, wenn Transaktionen vollständig ausgeführt werden. Deswegen müssen Transaktionen, die nur teilweise abgearbeitet wurden (z.B. wegen eines Systemfehlers, ...) rückgängig gemacht werden (? Redo).

Während der Ausführung einer Transaktion kann sich der Datenbestand zeitweise in einen inkonsistenten Zustand befinden, bei Beendigung ist er aber konsistent. Alle durchgeführten Änderungen an den Daten sind dann dauerhaft (? ACID-Prinzip).

COMMIT: Zeigt das erfolgreiche Ende (und den Beginn) einer (neuen) Transaktion an. Alle während der Transaktion gemachten Änderungen werden in der Datenbank gültig. Bei erfolgreichem Abschluss des Anwendungsprogramms erfolgt ein implizites COMMIT.
ROLLBACK: Zeigt das nicht erfolgreiche Ende (und den Beginn) einer (neuen) Transaktion an. Alle während der Transaktion gemachten Änderungen in der Datenbank werden rückgängig gemacht. Bei abnormalem Ende eines Anwendungsprogramms wird implizit eine ROLLBACK-Anweisung gestartet.

In manchen Systemen kann/muss man mit BEGIN WORK oder BEGIN TRANSACTION den Start einer Transaktion explizit angeben.
Beim Ausführen von COMMIT/ROLLBACK wird ein Synchronisationspunkt gesetzt. Dabei
- werden alle Änderungen seit dem letzten Synchronisationspunkt bestätigt/zurückgesetzt.
- gehen alle Datenbankpositionen verloren (Schließen der Cursor).
- werden alle Sperren (? Locks) von Datensätzen aufgehoben.

In einer Logdatei wird über die Datenmanipulationen Buch geführt. Es enthält:
- Before Images für veränderte Objekte (Objekt-ID, Wert vor Transaktion, Transaktions-ID).
- After Images für veränderte Objekte.
- Einträge für Beginn und Ende einer Tranaktion.

b. Recovery
- bei ? lokalen Fehlern: ROLLBACK der betroffenen Transaktion durch das Anwendungsprogramm, in manchen Fällen (z.B. nicht abgeschlossene Transaktionen bei Sitzungsende) durch das ? DBMS.
- bei ? Systemfehlern: ausgehend von einem ? Sicherungspunkt wird ein konsistenter Datenbankzustand rekonstruiert.

Strategien:
- undo/redo: Transaktionen die während oder nach dem Zeitpunkt der Sicherung aktiv sind werden nach folgenden Kriterien behandelt: Startpunkt vor der Sicherung ? undo, Ende vor Crash ? redo.
- undo/no-redo: Spätestens unmittelbar vor dem Commit einer Transaktion werden die Änderungen dieser Transaktion in die physische Datenbank geschrieben („Transaktionsspezifischer Checkpoint“). Dieses Verfahren ist aber sehr zeitkritisch, da es bei jeder Transaktion die Daten verändert (schreibende) Zugriffe auf den Peripheriespeicher erfordert.
- no-undo/redo: Nur zum Ende einer Transaktion dürfen erzeugte oder veränderte Daten in die physische Datenbank geschrieben werden. Aus dem Datenbank-Cache dürfen nur Daten verdrängt werden, die nicht von einer laufenden Transaktion bearbeitet werden.
- no-undo/no-redo: Veränderte Daten müssen genau zum Transaktionsende in einer atomaren Operation in die physische Datenbank geschrieben werden. Effizient nur mit ? Shadowing realisierbar.

c. Sperren
Will eine Transaktion ein Datenbankobjekt lesen oder verändern, fordert sie eine ? Sperre für das Objekt an.

d. Isolationslevel

	Dirty Read	Non-repeatable read	Phantoms
read uncommitted	X	X	X
read committed	-	X	X
repeatable read	-	-	X
serializeable	-	-	-

4. SQL

SQL	Beschreibung
SELECT [ALL|DISTINCT] * | {ausdruck,}ausdruck FROM tabelle{, tabelle} [WHERE bedingung] [GROUP BY {ausdruck,} ausdruck] [HAVING bedingung] [ORDER BY bedingung]	Selektion
select_1 UNION [ALL] select_2	Vereinigung
select_1 INTERSECT [ALL] select_2	Durchschnitt
select_1 MINUS [ALL] select_2 select_1 EXCEPT [ALL] select_2	Differenz
ALL	Mit ALL werden auch doppelte Zeilen inkludiert.
INSERT INTO tabelle [(spalte{, spalte})]	

VALUES (wert | NULL{, wert | NULL}) | (selektion)	
UPDATE tabelle SET spalte = ausdruck{, spalte = ausdruck} [WHERE bedingung]	
DELETE FROM tabelle [WHERE bedingung]	Löschen einzelner Tupel
DROP TABLE tabelle [CASCADE]	Löschen ganzer Tabellen. CASCADE löscht auch alle Views zu der Tabelle.
CREATE TABLE tabellenname (attributname typ [NOT NULL]{, attributname typ [NOT NULL]}, PRIMARY KEY (attributname{, attributname}) [FOREIGN KEY attributname REFERENCES tabellenname [(attributname)] {, FOREIGN KEY attributname REFERENCES tabellenname [(attributname)]])	Tabelle anlegen.
…WHERE [NOT] EXISTS (selektion)	
…WHERE [NOT] IN (selektion)	
…WHERE attribut [= | <= | >= | <> | < | >] ALL | ANY (selection)	ALL/ANY: Bedingung muss für alle Elemente/irgendein Element aus der Selektion erfüllt sein
… WHERE attribut BETWEEN Ganzzahl AND Ganzzahl	
… WHERE attribut LIKE [upper]`_Eyl%`	% ? Wildcard wie * _ ? Wildcard wie ? upper ? Interpretiert alle Buchstaben als Großbuchstabe Strings in ` ` werden case sensitive interpretiert.
MIN(attribut)	Ermittelt den geringsten Wert einer Spalte.
MAX(attribut)	Ermittelt den größten Wert einer Spalte.
SUM([DISTINCT] attribut)	Addiert alle (unterschiedlichen) Werte einer Spalte. Wenn DISTINCT angegeben wird, darf das Argument nur ein Spaltenname sein; ansonsten sind auch skalare Ausdrücke wie z.B. SUM(unsatz*16) erlaubt.
AVG([DISTINCT] attribut)	Ermittelt den durchschnittlichen Wert aller (unterschiedlichen) Werte einer Spalte.
COUNT([DISTINCT] attribut)	Zählt alle (unterschiedlichen) Elemente der Spalte ohne Nullwerte.
COUNT(*)	Berücksichtigt Nullwerte in der Ergebnismenge