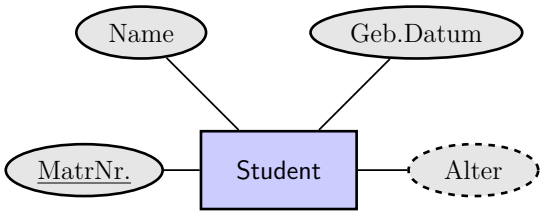
Virtuelle Attribute

Attribut *‚virtuell‘* (*‚abgeleitet‘*, *‚berechnet‘*), wenn aus anderen Attributwerten berechenbar. Angabe der Berechnungsformel notwendig.

Bsp.: Alter kann aus Geburtsdatum berechnet werden:

$$\text{Alter} := \text{Today} - \text{Geburtsdatum}$$

Wird gekennzeichnet durch gestrichelte Umrahmung des Attributs.



Aufgabe: ER-Diagramm Bestellung, Artikel, welches Gesamtpreis enthält.

Relationales Datenbankschema, Tabellen

Tabellen
Relationen
Übersetzung der ER-Diagramme
Vereinfachen der Relationenschemata

Übertragen Entity-Relationship-Diagramme in *‚Tabellen‘*/*‚Relationen‘*. Nächster Schritt zur Datenbank auf dem Rechner.

Formaler Prozess mit mathematischen Hintergrund.

Tabellen

Was ist eine ‚*Tabelle*'? Eigentlich bekannt . . .

Bundesliga (9. Spieltag 2023)

#	Team	Sp.	S	U	N	Tore	+/-	P
1	Bayer 04 Leverkusen	9	8	1	0	27:8	19	25
2	Bayern München	9	7	2	0	34:7	27	23
3	Vfb Stuttgart	9	7	0	2	27:11	18	21
4	Borussia Dortmund	9	6	3	0	20:11	9	21
5	RB Leipzig	9	6	2	1	25:7	18	20
6	TSG Hoffenheim	9	6	0	3	20:16	4	18
7	Eintracht Frankfurt	9	3	5	1	12:9	3	14
8	SC Freiburg	9	4	1	4	10:16	-6	13

Studenten

Name	MatrikelNr.	Studiengang	Geburtsjahr
Donald Duck	123101	Disney Comics	1934
Mickey Maus	123202	Disney Comics	1928
Bruce Wayne	210101	Helden Comics	1939
Batman	210102	Helden Comics	1939
Joker	210202	Helden Comics	1940
Hellboy	310101	Dark Comics	1993
Elisabeth Sherman	320202	Dark Comics	????

Tabelle wird charakterisiert durch

- ▷ Tabellennamen: ‚Studenten',
- ▷ Spaltenüberschriften: ‚Name', ‚MatrikelNr.', ‚Studiengang', ‚Geburtsjahr',
- ▷ Spalteneinträge gleichen Datentyps: Text, Ziffern, Datum.

Man erkennt Elemente aus Entity-Relationship-Diagrammen wieder:

- Tabellenname entspricht Entitätstyp,
- Spaltenüberschriften entsprechen Attributen,
- Spalteneinträge entsprechen Attributwerten.

Werden wir verwenden, um ER-Diagramme in Tabellen/Relationen zu konvertieren.

- Datentyp entspricht Wertebereich eines Attributs.

Hier nun eingeschränkt auf

- Text: Folge von Zeichen,
- Zahl: Folge von Ziffern (mit Dezimalpunkt),
- Datum: Angaben von Tag, Monat, Jahr,
- Boolean: Wahrheitswerte ‚wahr' und ‚falsch'.
- `NULL`: Lückenfüller.

Keine Zelle in einer Tabelle darf leer bleiben: `NULL`

Mehr und konkreter, wenn wir uns SQL anschauen.

Ausblick auf SQL:

```
-- setup table
CREATE TABLE Studenten (
  MatrNr      INTEGER NOT NULL PRIMARY KEY,
  Name        TEXT NOT NULL,
  Studiengang TEXT,
  Geburtsjahr INTEGER
);

-- populate table
INSERT INTO Studenten VALUES
  (123101, 'Donald Duck', 'Disney Comics', 1934);
INSERT INTO Studenten VALUES
  (210102, 'Batman', 'Helden Comics', 1939);
```

Studenten

MatrNr	Name	Studiengang	Geburtsjahr
123101	Donald Duck	Disney Comics	1934
210102	Batman	Helden Comics	1939

Relationen

Und jetzt nochmal abstrakt:

‚Relation‘: Gegeben seien Wertebereiche $D_1, D_2, \ldots, D_n$. Eine *‚Relation‘* R ist eine Teilmenge des kartesischen Produkts der Wertebereiche

$$R \subseteq D_1 \times D_2 \times \ldots \times D_n.$$

n heißt *‚Stelligkeit‘*oder *‚Grad‘* der Relation. Ein Element $r = (d_1, \ldots, d_n) \in R$ heißt *‚Tupel‘* und $d_i \in D_i$ heißt *‚Komponente‘* des Tupels.

Darstellbar als Tabelle

R

D_1	D_2	$\ldots$	D_n
d_1	d_2	$\ldots$	d_n
$\ldots$	$\ldots$	$\ldots$	$\ldots$

Mit $D_i = \text{dom}(A_i)$ als Wertebereich eines Attributs passt es zu den vorigen Tabellen.

Beispiel 6.1: Wertebereiche:

$$D_1 = \{\text{rot}, \text{grün}, \text{blau}\} \quad \text{und} \quad D_2 = \{0, 1\}.$$

Kartesisches Produkt:

$$D_1 \times D_2 = \{(\text{rot}, 0), (\text{rot}, 1), (\text{grün}, 0), (\text{grün}, 1), (\text{blau}, 0), (\text{blau}, 1)\}.$$

Beispiele für Relationen:

$$R_1 = \{(\text{rot}, 0), (\text{blau}, 1)\}$$

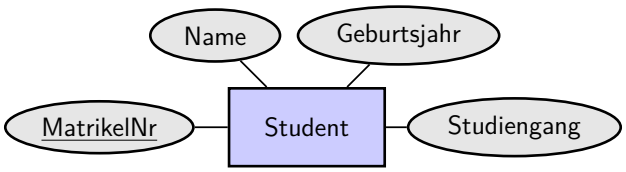
D_1	D_2
rot	0
blau	1

$$R_2 = \{\ \}$$

D_1	D_2

$$R_3 = \{(\text{rot}, 0)\}$$

D_1	D_2
rot	0



Studenten

MatrikelNr.	Name	Studiengang	Geburtsjahr

Fassen Formalismus zusammen in dem *‚relationalen Datenbankschema‘* oder *‚Relationsschema‘*:

$$\text{Student} = \{(\underline{\text{MatrikelNr}} : \text{integer}, \text{Name} : \text{text}, \text{Studiengang} : \text{text}, \text{Geburtsjahr} : \text{date})\}$$

Schlüssel wird wieder unterstrichen.

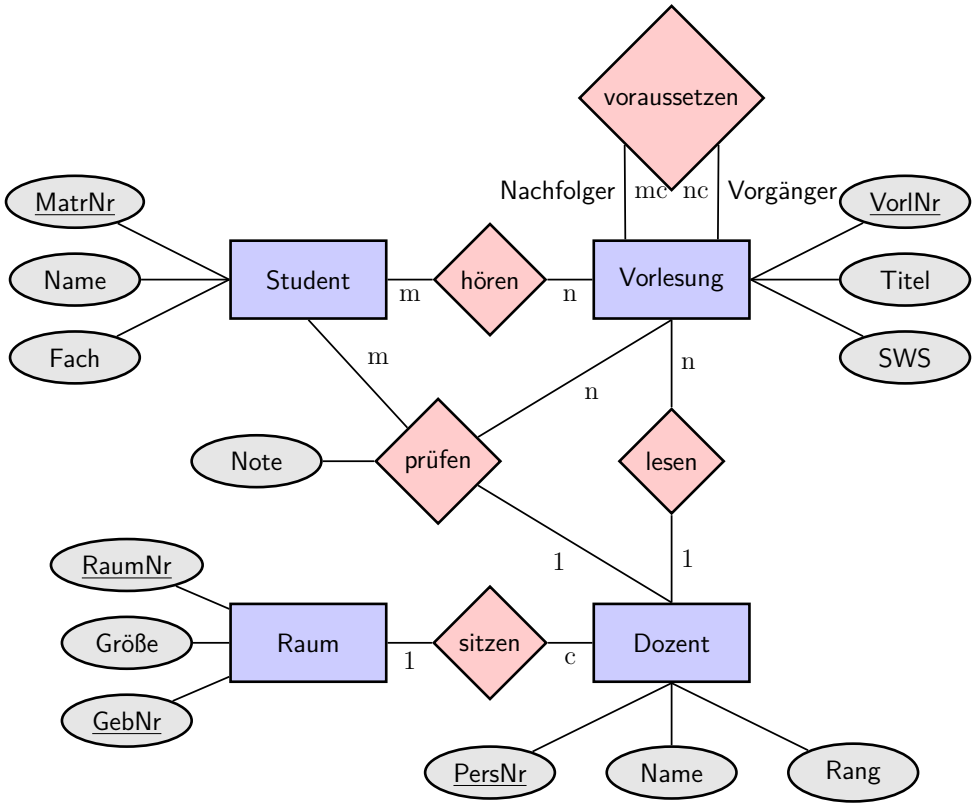
Übersetzung der ER-Diagramme

Übersetzung von ER-Modellen in Relationen/Tabellen mit drei Zielen:

1. Dateninhalt der Tabellen möglichts redundanzfrei,
2. Möglichst Verzicht auf `NULL`–Werte,
3. Möglichst minimale Anzahl von Tabellen.

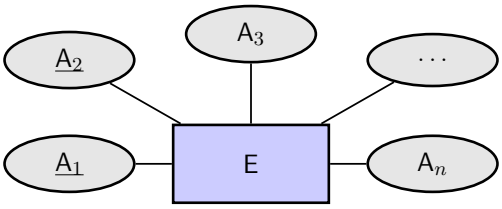
Betrachten nacheinander Übersetzung von Entitätstyp und Beziehungstyp.

Verwenden als mitlaufendes Beispiel die kleine ‚Hochschulwelt‘.



Entitätstyp → Relation

Jeder Entitätstyp entspricht einer eigenen Relation/Tabelle. Attribute werden Spaltenüberschriften. Attribute werden Datentypen zugewiesen. Schlüssel wird unterstrichen.

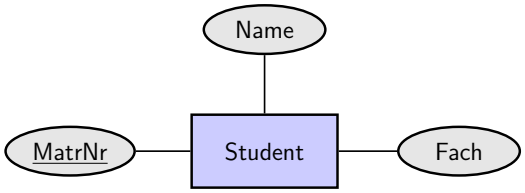


$$\mathsf{E} = \{(\underline{\mathsf{A}_1} : \mathsf{dtype}_1,\ \underline{\mathsf{A}_2} : \mathsf{dtype}_2,\ \mathsf{A}_3 : \mathsf{dtype}_3,\ \ldots : \ldots,\ \mathsf{A}_n : \mathsf{dtype}_n)\}$$

E

$\underline{\mathbf{A}_1}$	$\underline{\mathbf{A}_2}$	$\mathbf{A}_3$	...	$\mathbf{A}_n$

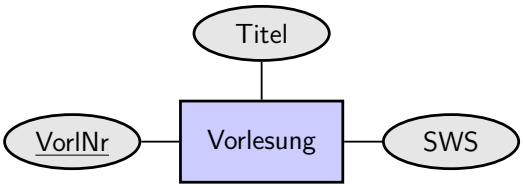
Beispiel 6.2:



Studenten

MatrNr	Name	Fach

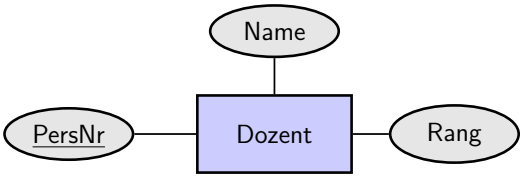
$$\text{Studenten} = \{(\underline{\text{MatrNr}} : \text{integer},\ \text{Name} : \text{text},\ \text{Fach} : \text{text})\}$$



Vorlesungen

VorlNr	Titel	SWS

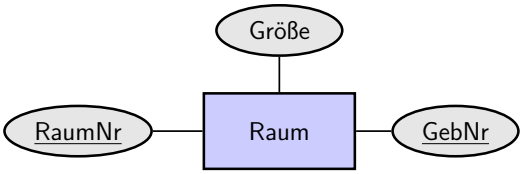
$$\text{Vorlesungen} = \{(\underline{\text{VorlNr}} : \text{integer},\ \text{Titel} : \text{text},\ \text{SWS} : \text{text})\}$$



Dozenten

PersNr	Name	Rang

$$\text{Dozenten} = \{(\underline{\text{PersNr}} : \text{integer},\ \text{Name} : \text{text},\ \text{Rang} : \text{text})\}$$



Räume

RaumNr	GebNr	Größe

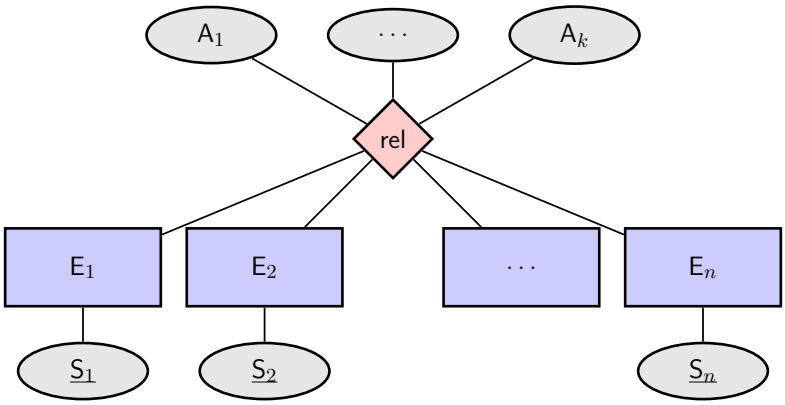
$$\text{Räume} = \{(\underline{\text{RaumNr}} : \text{integer},\ \underline{\text{GebNr}} : \text{integer},\ \text{Größe} : \text{numeric})\}$$

Konkrete Ausprägungen siehe Arbeitsblatt.

Umgang mit mehrwertigen und strukturierten Attributen kommt noch . . .

Beziehungstyp → Relation

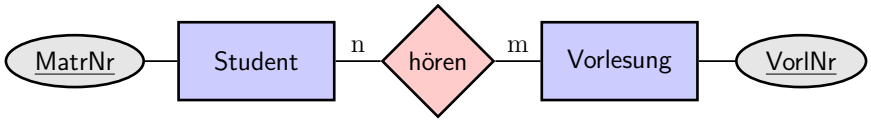
Jeder Beziehungstyp entspricht einer eigenen Relation/Tabelle. Als Spaltenüberschriften dienen die Schlüssel der beteiligten Entitätstypen und die eigenen Attribute.



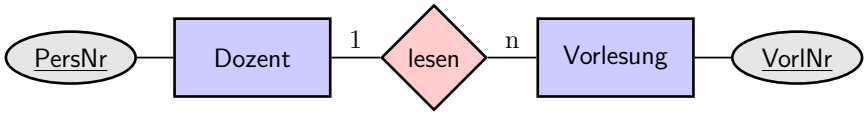
Vereinfachung: Schlüssel atomar, `integer`.

$$R = \{(\mathsf{S}_1 : \text{integer}, \ \ldots : \ldots, \ \mathsf{S}_n : \text{integer}, \ \mathsf{A}_1 : \text{dtype}_1, \ \ldots : \ldots, \ \mathsf{A}_k : \text{dtype}_k)\}$$

Beispiel 6.3:

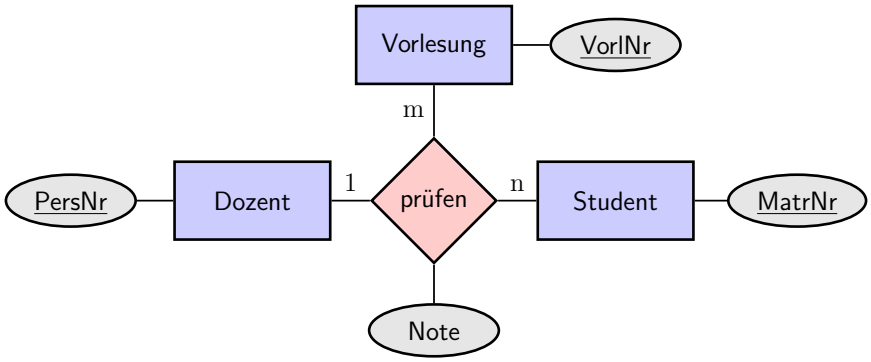


$$\text{Hören} = \{(\text{MatrNr} : \text{integer}, \ \text{VorlNr} : \text{integer})\}$$



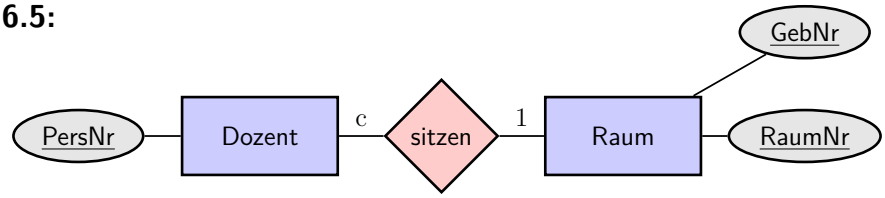
$$\text{Lesen} = \{(\text{PersNr} : \text{integer}, \ \text{VorlNr} : \text{integer})\}$$

Beispiel 6.4:

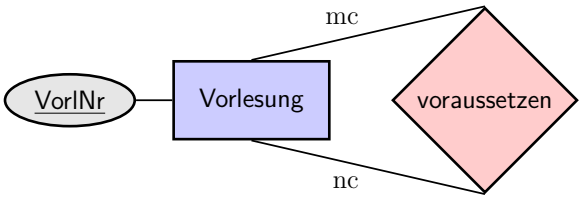


Prüfen $= \{(\text{MatrNr} : \text{integer}, \text{PersNr} : \text{integer}, \text{VorlNr} : \text{integer}, \text{Note} : \text{numeric})\}$

Beispiel 6.5:



Sitzen $= \{(\text{PersNr} : \text{integer}, \text{GebNr} : \text{integer}, \text{RaumNr} : \text{integer})\}$



Vorraussetzen $= \{(\text{VorgängerVorlNr} : \text{integer}, \text{NachfolgerVorlNr} : \text{integer})\}$

Aufgabe: Erstellen Sie zu den obigen Beziehungstypen Beispieltabellen und füllen Sie sie mit jeweils mindestens drei Datensätzen (Tupel).

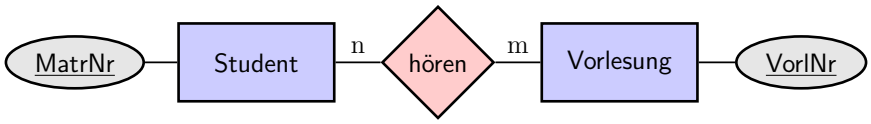
Bisher keine Schlüssel bei Beziehungsrelationen angegeben. Kommt jetzt.

Sclüssel minimale Menge von Attributen, deren Werte ein Tupel (Entität) eindeutig beschreiben. (Exakte Definition kommt noch . . .)

Konsequenz: Datensatz darf nicht doppelt in Relation/Tabelle vorkommen.

Schlüssel für m:n–Beziehungstyp

Alle Schlüssel der beteiligten Entitäten bilden den Schlüssel der Relation.



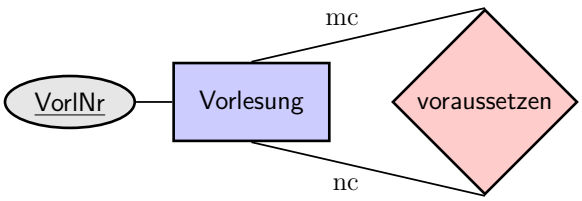
$$\text{Hören} = \{(\underline{\text{MatrNr}} : \text{integer},\ \underline{\text{VorlNr}} : \text{integer})\}$$

Attribut MatrNr verweist auf einen Studenten. VorlNr verweist auf eine Vorlesung.

Zu jeder MatrNr *können* mehrere Einträge in der Tabelle existieren. Dito für jede VorlNr.

Beispieltabelle

Bei rekursiven Beziehungstypen müssen gleichlautende Attribute *umbenannt* werden. Gilt möglicherweise auch bei anderen Beziehungstypen.



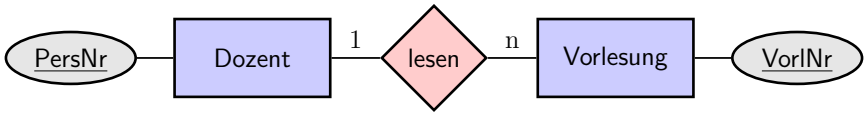
$$\text{Vorraussetzen} = \{(\underline{\text{VorgängerVorlNr}} : \text{integer},\ \underline{\text{NachfolgerVorlNr}} : \text{integer})\}$$

Beispieltabelle

Bemerkung: Hier und auch für folgende Beziehungsrelationen gilt:
‚Fremdschlüssel' VorgängerVorlNr und NachfolgerVorlNr müssen in der Tabelle Vorlesungen existieren. Integritätsbedingung.

Schlüssel für $1{:}\,n$–Beziehungstyp

Nur Schlüssel des n Entitätstyps wird Schlüssel der Relation.



$$\text{Lesen} = \{(\text{PersNr} : \text{integer},\ \underline{\text{VorlNr}} : \text{integer})\}$$

Abbildung

$$\text{lesen} : \text{Vorlesung} \longrightarrow \text{Dozent}$$

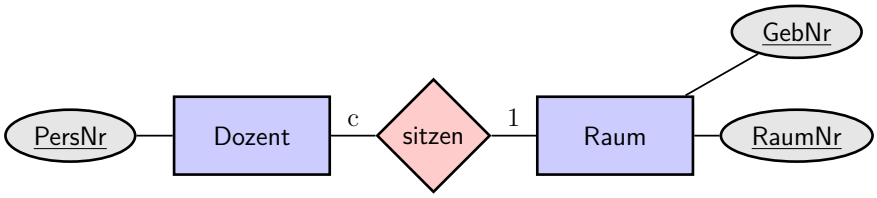
ist eindeutig: Vorlesung wird nur von einem Dozenten gelesen. Umkehrung ist nicht eindeutig: Dozent kann mehrere Vorlesungen lesen.

Kenn ich die Vorlesung, weiß ich *den* Dozenten. Kenn ich den Dozenten, weiß ich nicht (eindeutig) *die* Vorlesung.

Beispieltabelle

Schlüssel für $1{:}\,1$–Beziehungstyp

Nur ein Schlüssel der beiden 1 Entitätstypen wird Schlüssel der Relation. Und dann gibt es zwei Möglichkeiten.



1. Wählen Dozent

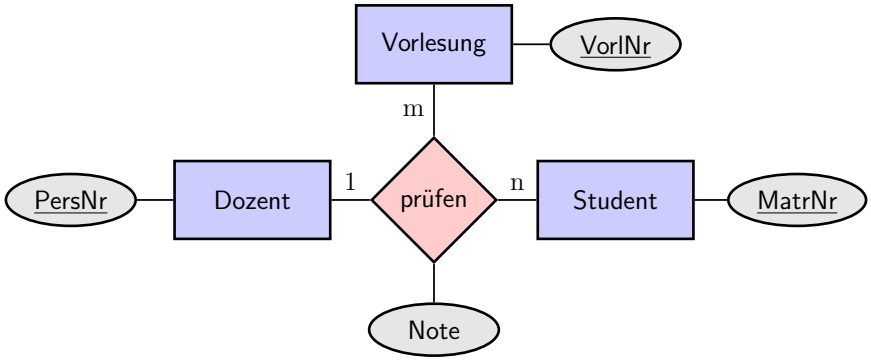
$$\text{Sitzen} = \{(\underline{\text{PersNr}} : \text{integer},\ \text{GebNr} : \text{integer},\ \text{RaumNr} : \text{integer})\}$$

2. Wählen Raum

$$\text{Sitzen} = \{(\text{PersNr} : \text{integer},\ \underline{\text{GebNr}} : \text{integer},\ \underline{\text{RaumNr}} : \text{integer})\}$$

Beispieltabellen

Wie geht man mit ternären Beziehungstypen um? (Oder höheren?)



Ein Dozent prüft zu *verschiedenen* seiner Vorlesungen jeweils *mehrere* Studierende.

Abbildung

$$\text{prüfen} : \text{Student} \times \text{Vorlesung} \longrightarrow \text{Dozent}$$

Schlüssel besteht aus den beiden Schlüsseln zu Student und Vorlesung:

$$\text{Prüfen} = \{(\underline{\text{MatrNr}} : \text{integer},\ \text{PersNr} : \text{integer},\ \underline{\text{VorlNr}} : \text{integer},\ \text{Note} : \text{numeric})\}$$

Vereinfachen der Relationenschemata

Prinzipell damit Entity-Relationship-Diagramme vollständig übersetzt in Tabellen/Relationen, inklusive Schlüssel.

Verfeinern der Schemata bei $1:1$ und $1:n$ Beziehungsrelationen. (*Nicht* $m:n$ Beziehungsrelation.)

Hauptprinzip: Relationen mit gleichem Schlüssel zusammenfassen.

Dabei beachten:

1. Dateninhalt der Tabellen möglichts redundanzfrei,
2. Verzicht auf `NULL`–Werte,
3. Möglichst minimale Anzahl von Tabellen.

Zusammenfassen von $1{:}n$–Beziehungstyp

Spielen dies am Beispiel durch

$$\begin{aligned}\text{Dozenten} &= \{(\underline{\text{PersNr}} : \text{integer},\ \text{Name} : \text{text},\ \text{Rang} : \text{text})\}\\ \text{Vorlesungen} &= \{(\underline{\text{VorlNr}} : \text{integer},\ \text{Titel} : \text{text},\ \text{SWS} : \text{text})\}\\ \text{Lesen} &= \{(\underline{\text{VorlNr}} : \text{integer},\ \text{PersNr} : \text{integer})\}\end{aligned}$$

Vorlesung und Lesen haben Attribut VorlNr gemeinsam. Deshalb zusammenfassen:

$$\begin{aligned}\text{Dozenten} &= \{(\underline{\text{PersNr}} : \text{integer},\ \text{Name} : \text{text},\ \text{Rang} : \text{text})\}\\ \text{Vorlesungen} &= \{(\underline{\text{VorlNr}} : \text{integer},\ \text{Titel} : \text{text},\ \text{SWS} : \text{text},\ \text{LesenderPersNr} : \text{integer})\}\end{aligned}$$

Dozenten

PersNr	Name	Rang
123	Carl Barks	C4
234	Frank Miller	C4
345	Mike Mignola	C3

Vorlesungen

VorlNr	Titel	SWS	LPersNr
11	Tolpatschigkeit	4	123
22	Heldenhaftigkeit	3	234
33	Architektur von Gotham	3	234
66	Geschichten von E.A.Poe	4	345

$$\begin{aligned}\text{Dozenten} &= \{(\underline{\text{PersNr}} : \text{integer},\ \text{Name} : \text{text},\ \text{Rang} : \text{text})\}\\ \text{Vorlesungen} &= \{(\underline{\text{VorlNr}} : \text{integer},\ \text{Titel} : \text{text},\ \text{SWS} : \text{text})\}\\ \text{Lesen} &= \{(\underline{\text{VorlNr}} : \text{integer},\ \text{PersNr} : \text{integer})\}\end{aligned}$$

Nicht Dozent und Lesen zusammenfassen (verschiedene Schlüssel), denn dann

$$\begin{aligned}\text{Dozenten} &= \{(\text{PersNr} : \text{integer},\ \text{Name} : \text{text},\ \text{Rang} : \text{text},\ \underline{\text{LeseVorlNr}} : \text{integer})\}\\ \text{Vorlesungen} &= \{(\underline{\text{VorlNr}} : \text{integer},\ \text{Titel} : \text{text},\ \text{SWS} : \text{text})\}\end{aligned}$$

Dozenten

PersNr	Name	Rang	LVorlNr
123	Carl Barks	C4	11
234	Frank Miller	C4	22
234	Frank Miller	C4	33
345	Mike Mignola	C3	66

Vorlesungen

VorlNr	Titel	SWS
11	Tolpatschigkeit	4
22	Heldenhaftigkeit	3
33	Architektur von Gotham	3
66	Geschichten von E.A.Poe	4

Redundanz in neuer Tabelle. Redundanzen vermeiden!

Zusammenfassen von $1:1$–Beziehungstyp

Spielen dies am Beispiel durch

$$\begin{aligned}\text{Dozenten} &= \{(\underline{\text{PersNr}} : \text{integer},\ \text{Name} : \text{text},\ \text{Rang} : \text{text})\}\\ \text{Räume} &= \{(\underline{\text{RaumNr}} : \text{integer},\ \underline{\text{GebNr}} : \text{integer},\ \text{Größe} : \text{numeric})\}\\ \text{Sitzen} &= \{(\underline{\text{PersNr}} : \text{integer},\ \text{GebNr} : \text{integer},\ \text{RaumNr} : \text{integer})\}\end{aligned}$$

Dozenten und Sitzen haben Attribut PersNr gemeinsam. Deshalb zusammenfassen:

$$\begin{aligned}\text{Dozenten} &= \{(\underline{\text{PersNr}} : \text{integer},\ \text{Name} : \text{text},\ \text{Rang} : \text{text},\ \text{GebNr} : \text{integer},\\ &\qquad \text{RaumNr} : \text{integer})\}\\ \text{Räume} &= \{(\underline{\text{RaumNr}} : \text{integer},\ \underline{\text{GebNr}} : \text{integer},\ \text{Größe} : \text{numeric})\}\end{aligned}$$

Aufgabe: Stellen Sie Beispieltabellen für die obigen Relationen zusammen.

Aufgabe: Wie sähe die Relationen aus, wenn man Sitzen mit Räume zusammenfasst? Stellen Sie Beispieltabellen für diese Relationen zusammen.

Haben nun ‚gute' Tabellen/Relationen erarbeitet. Können direkt in SQL abgebildet und gefüllt werden.

Schauen aber noch auf das, was schief gehen kann . . .

Anomalien und Normalformen

Anomalien
Funktionale Abhängigkeit und Schlüsselbegriff
Normalformen

Prinzipiell mit DB-Entwurf fertig (und zufrieden). Aber . . .

Gehen von einer ‚schlechten' Tabelle aus. Zeigen, was ist ‚schlecht' daran: *‚Anomalien'*.

Definieren exakt *‚Schlüssel'* und *‚Primärschlüssel'*.

Transformieren eine ‚schlechte' Tabelle in ‚gute' Tabellen: *‚Normalformen'*.

Ziel ist stets:

- Redundanz vermeiden,
- NULL vermeiden,
- wenige Tabellen.

Anomalien

Oft entstehen Tabellen ‚quick and dirty' (ungeplant) aus Zusammenstellung von Anwendungsdaten.

Bsp.: Zettelsammlung von Comics, Freundinnen und verliehenen Comics in einer Tabelle zusammenfassen.

FNr	CNr	Name	EMail	Titel	Serie	Nr	Autor	AusleihD	Zurück
1	3	Clark Kent	s@metro.com	V for Vendetta	-	-	Alan Moore	1.4.2012	Nein
2	5	Lois Lane	l@metro.com	Neverending	All-Star Superman	10	Grant Morrison	8.5.2012	Nein
3	-	Selina Kyle	s@gotham.net	-	-	-	-	-	-
10	7	Bruce Wayne	b@gotham.net	Stadt Ohne Gnade	Sin City	1	Frank Miller	1.1.2012	Ja
-	1	-	-	The Highway	Akira	1	Katsuhiro Otomo	-	-
1	2	Clark Kent	s@metro.com	Pursuit	Akira	2	Katsuhiro Otomo	1.1.2012	Nein

Update, Delete

Erkennen, dass Comics, Freunde und Ausleihe miteinander vermischt sind: Probleme!

Einfüge-Anomalie

oder welche Probleme macht die Tabelle beim Einfügen von Daten (Create).

Einfügen eines neuen Freundes in die Liste. Alle Comic- und Ausleihe-Daten unbesetzt: NULL Werte in der DB. (etwa bei ‚Selina Kyle')

FNr	CNr	Name	EMail	Titel	Serie	Nr	Autor	AusleihD	Zurück
3	-	Selina Kyle	s@gotham.net	-	-	-	-	-	-

Einfügen eines neuen Comics in die Liste. Alle Freund- und Ausleihe-Daten unbesetzt: NULL Werte in der DB. (etwa bei ‚Akira: The Highway')

FNr	CNr	Name	EMail	Titel	Serie	Nr	Autor	AusleihD	Zurück
-	1	-	-	The Highway	Akira	1	Katsuhiro Otomo	-	-

Insbesondere ‚Primärschlüssel' (‚FNr', ‚CNr') durch NULL-Werte nicht mehr eindeutig. Muss vermieden werden.

Änderungs-Anomalie

oder welche Probleme macht die Tabelle beim Ändern von Daten (Update).

‚Clark Kent' erhält neue EMail-Adresse (siehe Tabelle). Dann müssen an zwei Stellen in der Tabelle Änderungen vorgenommen werden.

FNr	CNr	Name	EMail	Titel	Serie	Nr	Autor	AusleihD	Zurück
1	3	Clark Kent	s@metro.com	V for Vendetta	-	-	Alan Moore	1.4.2012	Nein
2	5	Lois Lane	l@metro.com	Neverending	All-Star Superman	10	Grant Morrison	8.5.2012	Nein
3	-	Selina Kyle	s@gotham.net	-	-	-	-	-	-
10	7	Bruce Wayne	b@gotham.net	Stadt Ohne Gnade	Sin City	1	Frank Miller	1.1.2012	Ja
-	1	-	-	The Highway	Akira	1	Katsuhiro Otomo	-	-
1	2	Clark Kent	s@metro.com	Pursuit	Akira	2	Katsuhiro Otomo	1.1.2012	Nein

Fehleranfällig und kann zu Inkonsistenzen führen. Redundanzen vermeiden.

Lösch-Anomalie

oder welche Probleme macht die Tabelle beim Löschen von Daten (Delete).

FNr	CNr	Name	EMail	Titel	Serie	Nr	Autor	AusleihD	Zurück
1	3	Clark Kent	s@metro.com	V for Vendetta	-	-	Alan Moore	1.4.2012	Nein
2	5	Lois Lane	l@metro.com	Neverending	All-Star Superman	10	Grant Morrison	8.5.2012	Nein
3	-	Selina Kyle	s@gotham.net	-	-	-	-	-	-
10	7	Bruce Wayne	b@gotham.net	Stadt Ohne Gnade	Sin City	1	Frank Miller	1.1.2012	Ja
-	1	-	-	The Highway	Akira	1	Katsuhiro Otomo	-	-
1	2	Clark Kent	s@metro.com	Pursuit	Akira	2	Katsuhiro Otomo	1.1.2012	Nein

Wenn wir den Datensatz zu ‚Sin City: Stadt ohne Gnade' löschen (siehe Tabelle), verlieren wir auch unseren Freund ‚Bruce Wayne'.

Wenn wir den Datensatz zu ‚Lois Lane' löschen, verlieren wir auch den Comic ‚All-Star Superman: Neverending'.

Das ist unerwünscht.

Funktionale Abhängigkeit und Schlüsselbegriff

Beispiel der ‚schlechten' Tabelle zeigten Probleme mit Abhängigkeiten in der Struktur einer Tabelle. Betrachten dies nun abstrakt.

Gegeben sei eine Tabelle mit der Attributmenge $\mathcal{T}$.

Bsp.:

FNr	CNr	Name	EMail	Titel	Serie	Nr	Autor	AusleihD	Zurück

$$\mathcal{T} = \{\underline{\text{FNr}},\ \underline{\text{CNr}},\ \text{Name},\ \text{EMail}, \text{Titel},\ \text{Serie},\ \text{Nr},\ \text{Autor}, \text{AusleihD}, \text{Zurück}\}$$

‚Funktionale Abhängigkeit': Eine Menge von Attributen $\mathcal{B} \subseteq \mathcal{T}$ der Tabelle ist *‚funktional abhängig'* von einer Menge von Attributen $\mathcal{A} \subseteq \mathcal{T}$ der Tabelle, wenn zu jeder konkreten Belegung der Attribute aus $\mathcal{A}$ nur maximal eine konkrete Belegung der Attribute aus $\mathcal{B}$ gehört: $\mathcal{A} \to \mathcal{B}$.

Kenne ich für einen Datensatz die Attributwerte von $\mathcal{A}$, dann weiß ich auch die Belegung der Werte von $\mathcal{B}$.

Trivial:

$$\mathcal{T} \to \mathcal{T} \quad , \quad \mathcal{A} \to \mathcal{A} \quad \forall \mathcal{A} \subseteq \mathcal{T}.$$

Bsp.:

$$\{\text{FNr},\ \text{Name}\} \to \{\text{EMail}\}$$

Bsp.: Studierendentabelle

MatrNr	Name	Strasse	PLZ	Ort	EMail	Studiengang

Dann

$$\{\text{MatrNr},\ \text{Name},\ \text{PLZ}\} \to \{\text{Strasse},\ \text{Ort},\ \text{Email},\ \text{Studiengang}\}$$

Offensichtlich lässt sich die Menge links noch verkleinern. Man könnte bspw. ‚Name' herausnehmen. (Und auch noch ‚PLZ'.)

> *‚Volle funktionale Abhängigkeit'*: Eine Menge von Attributen $\mathcal{B} \subseteq \mathcal{T}$ der Tabelle ist *‚voll funktional abhängig'* von einer Menge von Attributen $\mathcal{A} \subseteq \mathcal{T}$, wenn $\mathcal{A} \to \mathcal{B}$ und für jede echte Teilmenge $\mathcal{A}'$ von $\mathcal{A}$ gilt $\mathcal{A}' \not\to \mathcal{B}$.

Suchen nach einer minimalen Menge von Attributen.
Bsp.:

$$\{\text{MatrNr}\} \to \{\text{Ort},\ \text{Name}, \text{Email}\} \quad , \quad \{\text{FNr}\} \to \{\text{Name},\ \text{Email}\}$$

Schauen auf verkürzte Relation

$$\text{Student: } \{\text{MatrNr} : \text{integer},\ \text{Name} : \text{text},\ \text{PLZ} : \text{integer},\ \text{Ort} : \text{text}\}$$

Wählen Attributmengen

$$\mathcal{X} = \{\text{MatrNr}\} \quad , \quad \mathcal{Y} = \{\text{PLZ}\} \quad , \quad \mathcal{Z} = \{\text{Ort}\}.$$

Dann

$$\mathcal{X} \to \mathcal{Y} \to \mathcal{Z} \qquad \text{‚}\textit{transitive Abhängigkeit}\text{'}.$$

Ortsname sicherlich nicht *direkt* vom Studenten abhängig. ‚PLZ' legt aber eindeutig ‚Ort' fest. Liegt aber nicht in der Grundmenge $\mathcal{X}$: *transitiv* abhängig.

Es genügt ‚PLZ' zu kennen, um auf ‚Ort' zu schließen.

Hinweis auf Zerlegung in *zwei* Tabellen/Relationen. (Kommt noch bei Normalformen.)

Mit den neuen Begriffen nun klären, was ein Schlüssel einer Tabelle/Relation ist.

‚Schlüssel‘: Sei $\mathcal{T}$ die Menge *aller* Attribute der Tabelle und sei $\mathcal{S} \subseteq \mathcal{T}$ eine weitere Attributmenge mit $\mathcal{S} \rightarrow \mathcal{T}$. Dann heißt $\mathcal{S}$ *‚Schlüssel‘* der Tabelle.

Eindeutige Zuordnung eines Datensatzes aufgrund der Werte in der Attributmenge $\mathcal{S}$. Keine Forderung der Minimalität. Damit $\mathcal{T}$ stets Schlüssel für Tabelle. Aber kein guter . . .

Praxis interessiert an möglichst *kleinem* Schlüssel.

‚Schlüsselkandidat‘: Sei $\mathcal{T}$ die Menge *aller* Attribute der Tabelle. Sei weiterhin $\mathcal{S} \subseteq \mathcal{T}$ eine Attributmenge, so dass $\mathcal{T}$ voll funktional abhängig ist von $\mathcal{S}$. Dann heißt $\mathcal{S}$ *‚Schlüsselkandidat‘* der Tabelle.

Und letztlich

‚Primärschlüssel‘: ist ein willkürlich ausgewählter Schlüsselkandidat einer Tabelle.

Beliebige minimale Menge von Attributen, die eindeutig auf Datensatz schließen lässt. Häufig künstlicher Primärschlüssel durch Aufzählen.
Aufteilung von $\mathcal{T}$:
‚Schlüsselattribute‘: Menge aller Attribute, die zu einem Schlüsselkandidaten gehören.
‚Nichtschlüsselattribute‘: Menge aller Attribute, die zu *keinem* Schlüsselkandidaten gehören.

Normalformen

Frage: Wie kann man ‚schlecht‘ strukturierte Tabelle in ‚gute‘ Tabellen umformen?

Antwort: Schrittweise über erste, dann zweite und dann dritte *‚Normalformen‘* bringen. Erzeugen neue Tabellen.

Normalformen helfen insbesondere, Redundanzen und NULL-Einträge zu vermeiden.

- 0NF: Nullte Normalform, Ausgangstabelle;
- 1NF, 2NF, 3NF: Erste, zweite, dritte Normalform;
- BCNF: Boyce-Codd-Normalform;
- 4NF, 5NF: Vierte, fünfte Normalform;

Nullte Normalform

Unsere noch nicht normalisierte Ausgangstabelle, wie sie aus der Anwendung herausfällt.

Erste Normalform

Ziel: Auflösen von mengenwertigen und strukturierten Attributen.

> ‚*Erste Normalform*‘: liegt vor, wenn jeder Attributwert eine atomare, nicht weiter zerlegbare Dateneinheit ist.

Vorgehensweise:

- Strukturierte Attribute werden in ihre atomare Unterattribute zerlegt und jedem wird eine Spalte zugeordnet.
- Mengenwertige Attribute werden in einer Spalte zusammengefasst. (Kann zu Redundanzen führen.)
- Jeder Datensatz wird durch einen Primärschlüssel eindeutig identifiziert. (Etwa durch neues Attribut ‚id‘.)

Unser Datenbankentwurf sollte eigentlich zu Tabellen in 1NF geführt haben, aber . . .

Strukturierte Attribute werden in ihre atomare Unterattribute zerlegt und jedem wird eine Spalte zugeordnet.

Name	Adresse	EMail
Clark Kent	123, Metropolis, Daily-Planet-Road 1	s@metropolis.com
Bruce Wayne	321, Gotham, Manor-Place 1	b@gotham.net
Lois Lane	123, Metropolis, Main-Road 42	l@metropolis.com

id	Name	PLZ	Ort	Strasse	Nr	EMail
1	Clark Kent	123	Metropolis	Daily-Planet-Road	1	s@metropolis.com
2	Bruce Wayne	321	Gotham	Manor-Place	1	b@gotham.net
3	Lois Lane	123	Metropolis	Main-Road	42	l@metropolis.com

Hier sicherlich auch Name als Primärschlüssel möglich.

Mengenwertige Attribute werden in einer Spalte zusammengefasst. (Kann zu Redundanzen führen.)

Titel	Serie	Nr	Autor1	Autor2
Schwarze Flamme	B.U.A.P.	4	Mike Mignola	John Arcudi
Old Ghosts	Watchmen	8	Alan Moore	-
Das Heiligtum von Gondwana	Die Abenteuer von Blake und Mortimer	17	Yves Sente	Andre Juillard
In Heaven (Everything is Fine) #1	Spawn	8	Alan Moore	-

Titel	Serie	Nr	Autor
Schwarze Flamme	B.U.A.P.	4	Mike Mignola
Schwarze Flamme	B.U.A.P.	4	John Arcudi
Old Ghosts	Watchmen	8	Alan Moore
Das Heiligtum von Gondwana	Blake und Mortimer	17	Yves Sente
Das Heiligtum von Gondwana	Blake und Mortimer	17	Andre Juillard
In Heaven (Everything is Fine) #1	Spawn	8	Alan Moore

Oder neues Attribut id als Primärschlüssel einführen.

Zweite Normalform

Ziel: Verhindern der (einfachen) Einfüge-, Änderungs- und Lösch-Anomalien.

‚Zweite Normalform': liegt vor, wenn Relation in 1NF ist und jedes Nichtschlüsselattribut voll funktional abhängig vom Primärschlüssel ist.

Vorgehensweise:

- Entfernen der Spalten, die nicht voll funktional vom Primärschlüssel abhängen.
- Zusammenfassen der zu einem Schlüssel gehörenden entfernten Spalten in neuer Tabelle. Zusammen mit dem Schlüssel.

Erzeugt neue Tabellen.

Titel	Serie	Nr	Autor
Schwarze Flamme	B.U.A.P.	4	Mike Mignola
Schwarze Flamme	B.U.A.P.	4	John Arcudi
Old Ghosts	Watchmen	8	Alan Moore
Das Heiligtum von Gondwana	Blake und Mortimer	17	Yves Sente
Das Heiligtum von Gondwana	Blake und Mortimer	17	Andre Juillard
In Heaven (Everything is Fine) #1	Spawn	8	Alan Moore

titelId	Titel	Serie	Nr
1	Schwarze Flamme	B.U.A.P.	4
2	Old Ghosts	Watchmen	8
3	Das Heiligtum von Gondwana	Blake und Mortimer	17
4	In Heaven (Everything is Fine) #1	Spawn	8

titelId	Autor
1	Mike Mignola
1	John Arcudi
2	Alan Moore
3	Yves Sente
3	Andre Juillard
4	Alan Moore

Aber immer noch Redundanz. (Doppelter Eintrag ‚Alan Moore'.) Deshalb . . .

Zerlegen in drei Tabellen. Dritte Tabelle für Relation ‚Autor schreibt Comic'. Jeweils mit eigenem Primärschlüssel.

titelId	Titel	Serie	Nr
1	Schwarze Flamme	B.U.A.P.	4
2	Old Ghosts	Watchmen	8
3	Das Heiligtum von Gondwana	Blake und Mortimer	17
4	In Heaven (Everything is Fine) #1	Spawn	8

autorId	Autor
1	Mike Mignola
2	John Arcudi
3	Alan Moore
4	Yves Sente
5	Andre Juillard

autorId	titelId
1	1
2	1
3	2
4	3
5	3
3	4

Änderungen nur noch an einer Stelle notwendig.
Löschen eines Comics löscht nicht den Autor.
Einfügen von Comics und Autoren unabhängig möglich.

Aufgabe: Wie könnte man noch mit ‚Serien' umgehen? Beachte etwa

Titel	Serie	Nr	Autor1	Autor2
Old Ghosts	Watchmen	8	Alan Moore	-
V for Vendetta	-	-	Alan Moore	-

Dritte Normalform

Ziel: Beseitigen von Abhängigkeiten zwischen Nichtschlüsselattributen. (Verhindern von transitiven Anomalien.)

> *‚Dritte Normalform'*: liegt vor, wenn Relation in 2NF und jedes Nichtschlüsselattribut nicht transitiv abhängig vom Primärschlüssel ist.

Vorgehensweise:

- ▷ Entfernen der Spalten, die transitiv vom Primärschlüssel abhängen.
- ▷ Zusammenfassen der zu einem Schlüssel gehörenden entfernten Spalten in neuer Tabelle. Zusammen mit dem Schlüssel.

Erzeugt neue Tabellen.

MatrNr	Name	PLZ	Ort
123101	Donald Duck	111	Entenhausen
210101	Bruce Wayne	200	Gotham
310101	Hellboy	666	DownUnder

Wissen schon: MatrNr $\rightarrow$ PLZ $\rightarrow$ Ort transitiv abhängig.

MatrNr	Name	PLZ
123101	Donald Duck	111
210101	Bruce Wayne	200
310101	Hellboy	666

PLZ	Ort
111	Entenhausen
200	Gotham
666	DownUnder

Weitere Normalformen

‚Boyce-Codd Normalform‘: liegt vor, wenn Relation in 3NF und alle voll funktionalen Abhängigkeiten vom Primärschlüssel ausgehen.

Machen wir nicht.

Und es gibt noch 4NF, 5NF. Machen wir auch nicht mehr.

Meist genügt 3NF für gute Tabellen im Rechner.

1NF verlangt nur atomare Attribute in der Tabelle. Werte einzeln abrufbar/abfragbar.

2NF verlangt 1NF und das Attribute vom gesamten Primärschlüssel abhängig.

3NF verlangt 2NF und das Nicht-Primärschlüssel-Attribute nicht transitiv abhängig.

Ergebnis:

- Verminderung von Redundanz,
- Vermeidung von Inkonsistenz und Anomalien,
- Übersichtliche kleine Tabellen.

Aber:

- Zu viele kleine Tabellen erschweren Überblick auf Miniwelt.
- Abfragen über mehrere Tabellen sehr aufwendig.

SQL

Allgemeines
CREATE TABLE
DROP TABLE
ALTER TABLE
INSERT INTO
UPADTE
DELETE
SELECT

Gute Tabellen sollen nun in eine DB auf den Rechner. Verwenden dazu DBMS *‚SQLite‘*.

Verwenden Datenbanksprache SQL (structured query language). Grobe Aufteilung in vier Bereiche: DDL, DML, TCL, DCL.