

Datenbanken

04 Integritätsbedingungen

Seminaristischer Unterricht

Prof. Dr.-Ing. Hendrik Gärtner

Gliederung

- SQL-Standard
- statische Integritätsbedingungen
 - Bedingungen an den Zustand der Datenbasis
- dynamische Integritätsbedingungen
 - Bedingungen an Zustandsübergänge
- Referentielle Integrität
 - Definition
 - Integritätsverletzungen

SQL

Überblick

- Standardisierte Sprache, besteht aus:
 - Datendefinitionssprache (DDL) – Data Definition Language
 - Datenmanipulationssprache (DML) – Data Manipulation Language
 - Datenkontrollsprache (DCL)- Data Control Language (Zugriffsrechte)
 - (select gehört zur DML)
- PostgreSQL unterstützt vollständig den Standard SQL 2011:
 - 2008: SQL:2008 bzw. ISO/IEC 9075:2008. Als neue Features werden aufgenommen INSTEAD OF-Trigger, TRUNCATE-Statement und FETCH Klausel.
 - ⇒ - 2011: SQL:2011 bzw. ISO/IEC 9075:2011. Als neue Features werden aufgenommen „Zeitbezogene Daten“ (PERIOD FOR). Es gibt Erweiterungen für Window functions und die FETCH Klausel.
 - 2016: SQL:2016 bzw. ISO/IEC 9075:2016. Als neue Features werden aufgenommen JSON und „row pattern matching“.
 - 2019: SQL/MDA:2019. Erweiterungen für einen Datentyp „mehrdimensionales Feld“.¹



Wichtig: Dokumentation des Datenbanksystems - Schauen Sie mal rein!

¹Quelle: Wikipedia, <https://de.wikipedia.org/wiki/SQL>, zugegriffen am 12.10.2020

(Einfache) Datendefinition in SQL

Datentypen

'm' char

text

- **character** (n), **char** (n)
- **character varying** (n), **varchar** (n) - Zeichenketten mit Höchstgrenze ↗
- **text** Zeichenketten ohne Höchstgrenze
- **numeric** (p,s), **integer** ↗
- **date** für Datumsangaben ↗

³⁰
VARCHAR(x) char(x)³⁰

Hans Meier 10 Zeichen

Hans Meier leer 30 Zeichen

Anlegen von Tabellen (DDL):

create table Professoren
 (PersNr **integer**,
 Name **varchar** (30) **not null**,
 Rang **character** (2),
 PRIMARY KEY (PersNr)

primary key

CHAR 10
1 Byte

(PersNr, Rang)

);

Integritätsbedingungen

Begriffsdefinition Integrität

Integrität (lateinisch integritas ‚unversehrt‘, ‚intakt‘, ‚vollständig‘) steht für¹:

- Integrität (Ethik), eine ethische Forderung des philosophischen Humanismus
- Integrität (Persönlichkeitseigenschaft), persönliche Integrität
- **Integritätsbedingung, eine Konsistenzbedingung für Datenbanken**
- **Integrität (Informationssicherheit), ein Schutzziel in der Informationssicherheit**
- Körperintegrität, in der Biologie die körperliche Unversehrtheit
- Territoriale Integrität, im Völkerrecht Begriff der territorialen Unversehrtheit

¹Quelle: Wikipedia, <https://de.wikipedia.org/wiki/Integrit%C3%A4tsbedingung>, Zugriff: 12.10.2020.

Integrität in Datenbanken

- **Konsistenzbedingung für Datenbanken:**

Integritätsbedingungen beschreiben *Annahmen, die über die Daten bzw. den Zustand getroffen werden*, beispielsweise ein bestimmter Datentyp, ein Wertebereich oder eine Abhängigkeitsbeziehung zwischen zwei Objekten. Daten werden als *konsistent* betrachtet, wenn sie die definierten Integritätskriterien erfüllen

- **Integrität im Bereich der Datensicherheit:**

Die Gewährleistung der Integrität besagt, dass *Daten über einen bestimmten Zeitraum vollständig und unverändert* sein sollen. Eine Veränderung kann dabei absichtlich, unabsichtlich oder durch einen technischen Defekt auftreten. Ziel ist es dabei auch, *Veränderungen an den Daten erkennen* zu können

Bisher haben Sie kennen gelernt...

create table Professoren

(PersNr **integer**,
Name **varchar** (30) **not null**,
Rang **character** (2),
PRIMARY KEY (PersNr)
);

Handwritten annotations:
- An arrow points from 'integer' to 'integer'.
- An arrow points from 'varchar' to 'varchar'.
- An arrow points from 'not null' to 'not null'.
- An arrow points from 'character' to 'character'.
- An arrow points from 'PRIMARY KEY (PersNr)' to 'not null'.
- An arrow points from 'PRIMARY KEY (PersNr)' to 'unique'.
- An arrow points from 'FOREIGN Key (...)' to 'PRIMARY KEY (PersNr)'.
- An arrow points from 'FOREIGN Key (...)' to 'unique'.

- Typdefinitionen: INTEGER, VARCHAR, CHARACTER
- Längenangaben für die Zeichenketten
- NOT NULL: Wert des Attributs darf nicht leer/undefiniert sein
- PRIMARY KEY: NOT NULL und UNIQUE

Arten von Integritätsbedingungen

- Statische Integritätsbedingungen
 - Bedingungen an den Zustand der Datenbasis
- Dynamische Integritätsbedingungen
 - Bedingungen an Zustandsübergänge



Referentielle Integrität ist eine statische Integritätsbedingung!

Warum Integritätsbedingungen definieren?

- Konsistenz des Datenbestands wird von der Datenbank und nicht von jeder einzelnen Anwendungsprogramm sichergestellt
- Inkonsistente Zustände der gespeicherten Daten sind somit unmöglich
- Konsistenzcheck der Daten ist einfach abschaltbar (ist z.B. aus Gründen der Performanz sinnvoll bei Datenimports)

Typische Statische Integritätsbedingungen

- Befüllung des Attributs
... **not null** ...
- Wertebereichseinschränkungen ↙
... **check** Semester **between 1 and 13**
- Aufzählungstypen ↘
... **check** Rang **in** ('C2', 'C3', 'C4') ...

Integritätsbedingungen
werden auch **Constraints**
genannt



create table Studenten

(MatrNr **integer primary key,**

Name **varchar(30) not null,**

Semester **integer** ↙ **check** Semester **between 1 and 13);**

↑

Constraints
werden meist hinter
die Spaltendefinitionen
geschrieben.

Dynamische Integritätsbedingungen

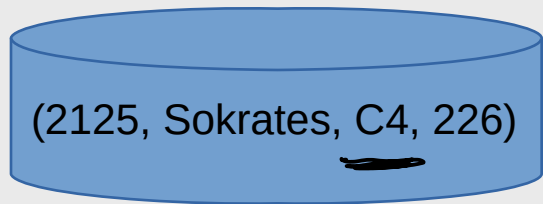
Bedingungen an Zustandsübergänge – Beispiel Professoren
dürfen nicht degradiert werden!

$C2 \rightarrow C3 \rightarrow C4$

$C4 \rightarrow C3$ ~~$C4 \rightarrow C2$~~

↓
Tabelle Professoren : Vor Operation

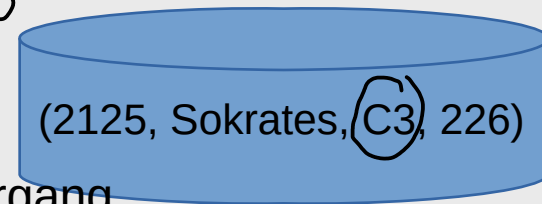
Tabelle Professoren : Nach Operation



update C4 → C3



Zustands-übergang



↓
... **check** Rang in ('C2', 'C3', 'C4') ...

↓
... **check** Rang in ('C2', 'C3', 'C4') ...

- Constraints gelten vor und nach der Operation – Zustandsübergang unerwünscht
 - Solche Bedingungen lassen sich nicht direkt im Schema definieren
 - Lösung: Stored Procedures und Trigger
- Wenn ein Update ausgeführt wurde auf der Tabelle Professoren, dann
- ⋈ pgsql

Referentielle Integrität

Referentielle Integrität

Fremdschlüssel:

- verweisen auf Tupel einer Relation, z.B. *gelesenVon* in *Vorlesungen* verweist auf Tupel in *Professoren*

Referentielle Integrität:

- Fremdschlüssel müssen auf existierende Tupel verweisen oder einen Nullwert enthalten



Vorlesungen			
VorlNr	Titel	SWS	gelesen Von
5001	Grundzüge	4	2137
5041	Ethik	4	2125
5043	Erkenntnistheorie	3	2126
5049	Mäeutik	2	2125
4052	Logik	4	2125

Professoren			
PersNr	Name	Rang	Raum
2125	Sokrates	C4	226
2126	Russel	C4	232
2127	Kopernikus	C3	310
2133	Popper	C3	52
2134	Augustinus	C3	309
...

4053 Quantencomputing & 9999 INSERT

Beispiel aus dem Universitätsschema

CREATE TABLE Professoren

(PersNr INTEGER PRIMARY KEY,
Name VARCHAR(30) NOT NULL,
Rang CHAR(2) CHECK (Rang in ('C2', 'C3', 'C4')),
Raum INTEGER UNIQUE);

CREATE TABLE Vorlesungen

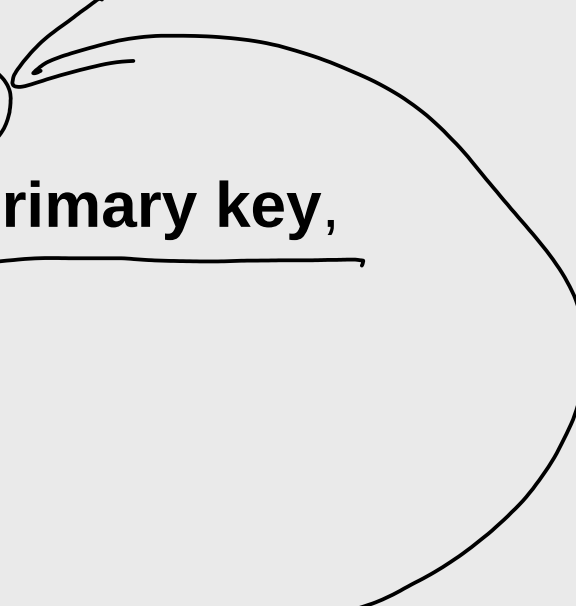
(VorlNr INTEGER PRIMARY KEY,
Titel VARCHAR(30),
SWS INTEGER,
gelesenVon INTEGER REFERENCES Professoren(PersNr));

Referentielle Integrität in SQL

- Kandidatenschlüssel: **unique**
- Primärschlüssel: **primary key** \leftarrow
- Fremdschlüssel: **foreign key** \leftarrow
- Beispiel:

create table R
(α) integer primary key,
...);

create table S
(...,
 κ integer references R);



Beispiel aus dem Universitätsschema

CREATE TABLE Studenten

(MatrNr INTEGER PRIMARY KEY,
Name VARCHAR(30) NOT NULL,
Semester INTEGER);

CREATE TABLE hoeren

(MatrNr INTEGER REFERENCES Studenten(MatNr) ON
DELETE CASCADE,
VorlNr INTEGER REFERENCES Vorlesungen(VorlNr) ON
DELETE CASCADE,
PRIMARY KEY (MatrNr, VorlNr));

Typ gleich

*=
Typ muss gleich sein*

Einhaltung Referentieller Integrität

Integritätsbedingungen müssen überprüft werden bei:

1. Einfügen eines neuen Datensatzes ↵
2. Ändern eines Datensatzes ↵
3. Löschen eines Datensatzes ↵

Strategien bei Verletzung der Integritätsbedingungen:

4. Default: Zurückweisen der Änderungsoperation ↵
5. Propagieren der Änderungen: **cascade** ↵
6. Verweise auf Nullwert setzen: **set null** ↵

Einfügeoperation

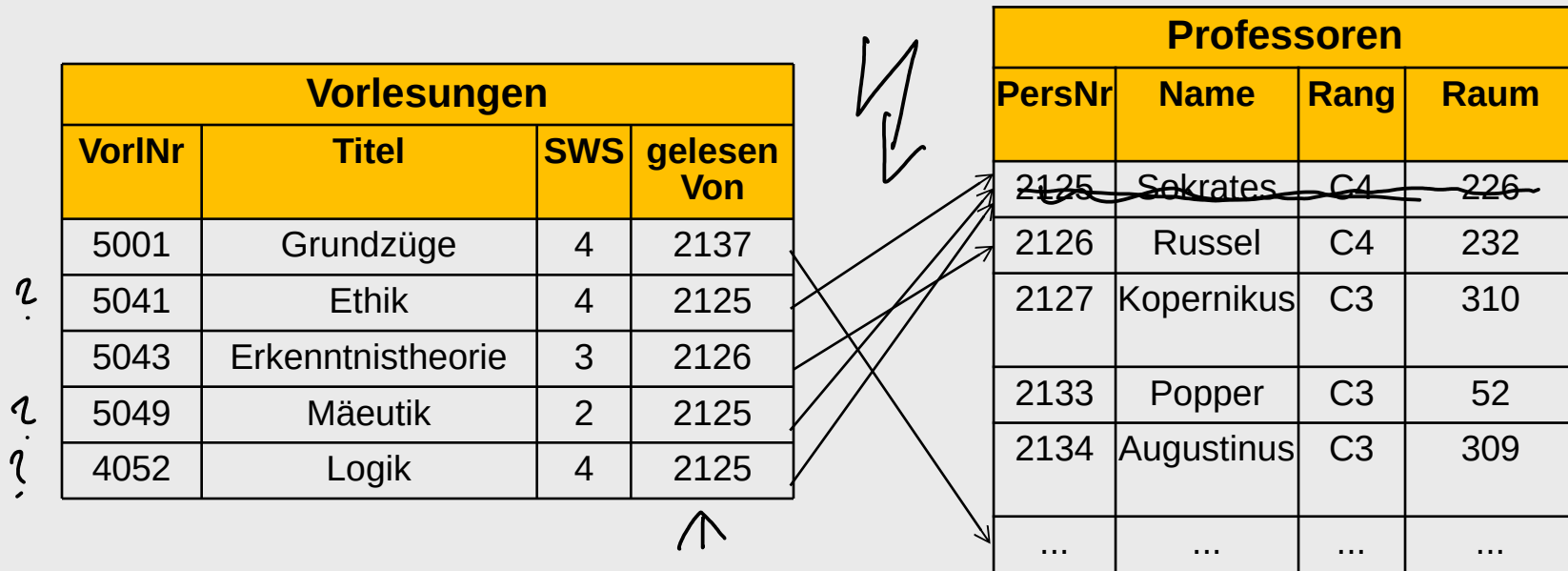
Vorlesungen			
VorlNr	Titel	SWS	gelesen Von
5001	Grundzüge	4	2137
5041	Ethik	4	2125
5043	Erkenntnistheorie	3	2126
5049	Mäeutik	2	2125
4052	Logik	4	2125

Professoren			
PersNr	Name	Rang	Raum
2125	Sokrates	C4	226
2126	Russel	C4	232
2127	Kopernikus	C3	310
2133	Popper	C3	52
2134	Augustinus	C3	309
...

INSERT INTO Vorlesungen (vorlNr, titel, sws, gelesenvon) VALUES (1234, 'Quantencomputing', 4, 9999)

→ Operation wird zurückgewiesen, da ein Professor mit der Personalnummer 9999 nicht existiert (mögliche Umgehung: Transaktion)

Deleteoperation



DELETE FROM Vorlesungen WHERE persnr=2125

→ Operation wird zurückgewiesen, da es Vorlesungen gibt, die auf Professor Sokrates verweisen

Deleteoperation

Vorlesungen			
VorlNr	Titel	SWS	gelesen Von
5001	Grundzüge	4	2137
5041	Ethik	4	2125
5043	Erkenntnistheorie	3	2126
5049	Mäeutik	2	2125
4052	Logik	4	2125

Professoren			
PersNr	Name	Rang	Raum
2125	Sokrates	C4	226
2126	Russel	C4	232
2127	Kopernikus	C3	310
2133	Popper	C3	52
2134	Augustinus	C3	309
...

DELETE FROM Vorlesungen WHERE persnr=2125

... gelesenVon INTEGER REFERENCES Professoren(PersNr)
ON DELETE CASCADE ...

→ Ein Löschen des Professors führt zu einem Löschen der zugehörigen Vorlesungen

Deleteoperation

Vorlesungen			
VorlNr	Titel	SWS	gelesen Von
5001	Grundzüge	4	2137
5041	Ethik	4	2125 0
5043	Erkenntnistheorie	3	2126
5049	Mäeutik	2	2125 0
4052	Logik	4	2125 0

2133

Professoren			
PersNr	Name	Rang	Raum
2125	Sokrates	C4	226
2126	Russel	C4	232
2127	Kopernikus	C3	310
2133	Popper	C3	52
2134	Augustinus	C3	309
...

DELETE FROM Vorlesungen WHERE persnr=2125

... gelesenVon INTEGER REFERENCES Professoren(PersNr)
ON DELETE CASCADE ... SET NULL

→ Ein Löschen des Professors führt zu einem NULL setzen der Fremdschlüsseleinträge (nur möglich, wenn Constraint NOT NULL nicht gesetzt).

Updateoperation

Vorlesungen			
VorlNr	Titel	SWS	gelesen Von
5001	Grundzüge	4	2137
5041	Ethik	4	2125
5043	Erkenntnistheorie	3	2126
5049	Mäeutik	2	2125
4052	Logik	4	2125

Professoren			
PersNr	Name	Rang	Raum
2125	Sokrates	C4	226
2126	Russel	C4	232
2127	Kopernikus	C3	310
2133	Popper	C3	52
2134	Augustinus	C3	309
...

↓ ↓
UPDATE Professoren SET persnr=persnr+10000

→ Operation wird zurückgewiesen, da die Schlüssel aus Vorlesungen alle ungültig werden

Updateoperation

Vorlesungen			
VorlNr	Titel	SWS	gelesen Von
5001	Grundzüge	4	1 2137
5041	Ethik	4	1 2125
5043	Erkenntnistheorie	3	1 2126
5049	Mäeutik	2	1 2125
4052	Logik	4	1 2125

Professoren			
PersNr	Name	Rang	Raum
12125	Sokrates	C4	226
12126	Russel	C4	232
12127	Kopernikus	C3	310
12133	Popper	C3	52
12134	Augustinus	C3	309
...

UPDATE Professoren SET persnr=persnr+10000

... gelesenVon INTEGER REFERENCES Professoren(PersNr)
ON UPDATE CASCADE ...



→ Operation sorgt dafür, dass alle Schlüssel ebenfalls umgesetzt werden

Updateoperation

Vorlesungen			
VorlNr	Titel	SWS	gelesen Von
5001	Grundzüge	4	21370
5041	Ethik	4	21250
5043	Erkenntnistheorie	3	21260
5049	Mäeutik	2	21250
4052	Logik	4	21250

Professoren			
PersNr	Name	Rang	Raum
12125	Sokrates	C4	226
12126	Russel	C4	232
12127	Kopernikus	C3	310
12133	Popper	C3	52
12134	Augustinus	C3	309
...

UPDATE Professoren SET persnr=persnr+10000

... gelesenVon INTEGER REFERENCES Professoren(PersNr)
ON UPDATE SET NULL ...

→ Operation löscht die Fremdschlüsseinträge in Vorlesungen (nur möglich, wenn Constraint NOT NULL nicht gesetzt ist).

Untersch. Operationen in einer Integritätsbedingung

CREATE TABLE hören

(MatrNr INTEGER REFERENCES Studenten

ON DELETE CASCADE

ON UPDATE SET NULL,

} Unterschiedliche
Bedingungen

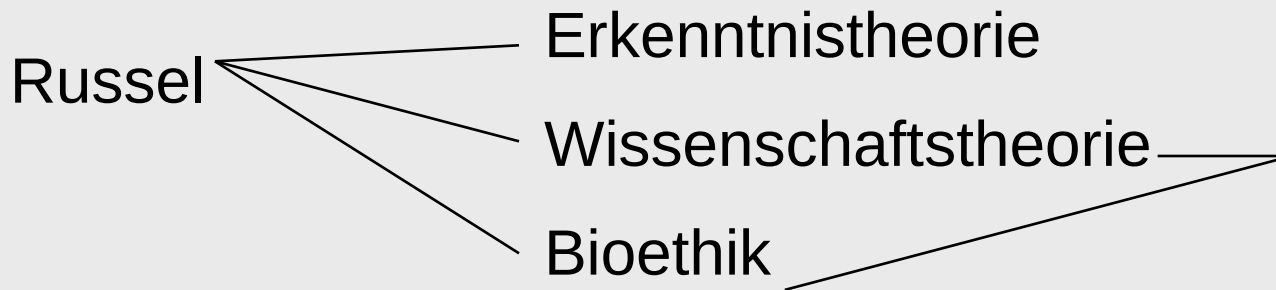
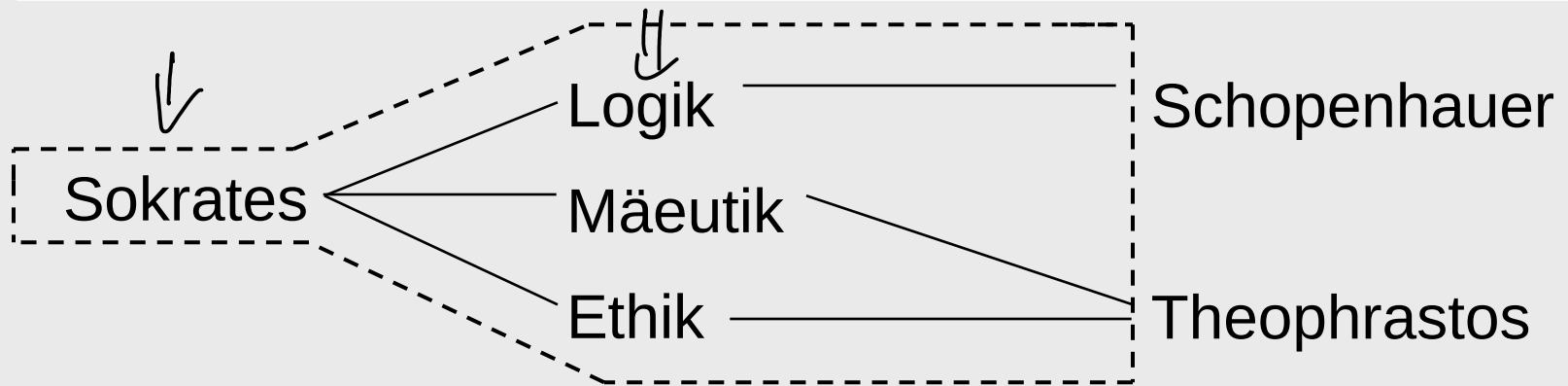
S_{iiu}

VorlNr INTEGER REFERENCES Vorlesungen

ON DELETE CASCADE,

PRIMARY KEY (MatrNr, VorlNr));

Kaskadierendes Löschen



⋮

⋮

⋮

Vorsicht: Unvorsichtiges, kaskadierendes Löschen kann unerwünschte Nebenwirkungen haben!

Vielen Dank für

Ihre Aufmerksamkeit

Prof. Dr.-Ing. Hendrik Gärtner