we focus on students

Datenbanken 1

Trigger

Fachhochschule Dortmund
University of Applied Sciences

Inhaltsübersicht

Organisation

- Diese Woche:
  - Dienstag: Vorlesung (in Präsenz mit Livestream) und Praktikum in Präsenz
  - Mittwoch-Freitag: Praktika online (Bahnstreik)
  - Die geplanten Abnahmen werden auf nächste Woche verschoben (Organisation durch Robin)
- Nächste Woche:
  - Dienstag: Prüfungsvorbereitung
  - Keine regulären Praktika, nur noch restliche Abnahmen

- Prüfungsvorbereitung
  - Klausurvorbereitung am 31.01.2024
    - Klausurvorbesprechung von 13 Uhr bis 14 Uhr im Raum A.2.02
    - Klausurvorbereitungstutorium ab 14:15 Uhr in den Räumen C.1.30 und C.1.31
  - Klausur: 08.02.2024, 13 Uhr

Inhaltsübersicht

Definition einer Benutzersicht als Datenbankobjekt

Durch eine Benutzersicht (View) wird eine SQL-Anfrage als Datenbankobjekt gespeichert. Die Auswertung der gespeicherten Anfrage erfolgt bei jedem Aufruf des der Benutzersicht neu. Bei der Viewdefinition entsprechen die Spaltentypen denen der Attribute der Basistabelle. Umbenennung der Attribute über die Attributliste sind möglich.

Syntax

```
CREATE View ‹Viewname› [(Attributliste)]
AS                    ‹Select-Anweisung›
[WITH CHECK OPTION]
```

Beispiel

```
CREATE View KundenAusDortmund
                      (KNr,          KName,    Vorname, Anrede)
AS
          SELECT  Kundennummer, Nachname, Vorname, Anrede
          FROM    Kunde
          WHERE   Ort = 'Dortmund'
```

Attribute des Views

Attribute der Tabelle

Änderungen auf einem einfachen Views

Änderungen auf dem View sind möglich, wenn Tupel eindeutig identifizierbar sind.
Speziell gilt:

- INSERT
  - Bei fehlenden Werte wird NULL bzw. Standardwert ergänzt
  - Nicht möglich bei fehlendem NOT-NULL Werten ohne Default-Wert
- UPDATE
  - Änderbar sind nur die im View sichtbaren Attribute der Basistabelle
- DELETE
  - Entfernung des gesamten Tupels

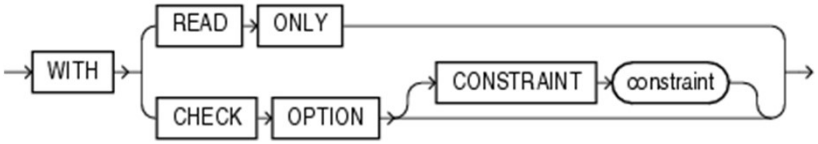
Basistabelle
Kunde

Kunden-nummer	Nachname	Vorname	Anrede	Geburts-datum	Ort
2310	Meitner	Lise	Frau	17.11.1878	Berlin
7562	Einstein	Albert	Herr	14.03.1879	Princeton
8523	Dekanat	Informatik	NULL	NULL	Dortmund

Sicht

Prüfung bei der Änderung (CHECK-Option)

Bei Änderungen auf einem View wird die WHERE-Bedingung nur dann geprüft, wenn die CHECK-Option vorhanden ist. ist.
Bei Oracle erfolgt die Prüfung **kaskardierend**,
d.h. die CHECK-Klausel gilt auch für die eingeschachtelten Views.




```
CREATE View ‹Viewname› [(Attributliste)]
AS                    ‹Select-Anweisung›
[WITH CHECK OPTION]
```

```
CREATE View KundenAusDortmund2
(Kundennummer, Name, Ort)
AS
        SELECT Kundennummer, Nachname, Ort
        FROM KUNDE
        WHERE  Ort='Dortmund'
WITH CHECK OPTION
```

Beispiel – Schachtelung von Views

Views können ineinander verschachtelt werden. Die Änderbarkeit hängt von der Änderbarkeit der verschachtelten Benutzersichten ab. Entscheidend ist, ob die Tupel der resultierenden Benutzersicht eindeutig identifiziert werden können.

Beispiel: Liefere die Anzahl aller Firmenkunden

```
CREATE View AnzahlFirmenKundenAusDortmund (Anzahl)
AS
        SELECT count(*)
        FROM KundenAusDortmund
        WHERE Anrede        IS NULL
```

```
CREATE View KundenAusDortmund
AS
        SELECT Kundennummer, Nachname, Vorname, Anrede
        FROM    Kunde
        WHERE  Ort = 'Dortmund'
```

Änderbarkeitskriterien für Views

Prinzipiell ist eine Änderung über die Benutzersicht möglich, wenn

- Tupel der Benutzersicht eindeutig auf Tupel der Basisrelation abzubilden sind. In der Regel ist das der Fall, wenn der (Primär-) Schlüssel vollständig in der Benutzersicht enthalten ist.

Keine Änderung auf dem View ist möglich, bei:

- fehlenden NOT NULL Spalten
- Verwendung von DISTINCT
- Verwendung von Aggregatfunktionen COUNT, SUM, AVG, MIN und MAX
- Verwendung von Mengenoperationen UNION, INTERSECT und MINUS
- Gruppierung durch GROUP BY und HAVING

Die Änderbarkeit von Views, die Unterabfragen (Subqueries) beinhalten, hängt von dem verwendeten DBMS ab.

Beispiel – Nicht-Änderbare Benutzersicht (Read-Only)

Bei Verbundoperationen müssen alle Schlüssel in der Benutzersicht enthalten und eindeutig sein, damit die betroffenen Tupel eindeutig identifiziert werden können.

Definition:

```
CREATE VIEW Artikelbestand
AS
SELECT Artikelnummer, Artikelname, Lagernummer, Lagerbestand
FROM Artikel a JOIN Lager l ON a.Artikelnummer=l.ANummer
```

Abfrage:

```
SELECT * FROM Artikelbestand
```

ARTIKELNUMMER	ARTIKELNAME	LAGERNUMMER	LAGERBESTAND
4820	Datenbank-Skript	27527	3
4820	Datenbank-Skript	27528	5

Änderung:

```
UPDATE Artikelbestand
SET Artikelname='Datenbanken'
WHERE Artikelnummer=4820
```

SQL-Fehler:
ORA-01779: cannot modify a column which maps to a non key-preserved table
→ (Oracle 11g) Fehler, wenn ein Fremdschlüssel nicht unique (=non key-preserved) ist
(Ein Artikel kann mehrere Lagerplätze besitzen)
ORA-30926: Stabile Zeilengruppe in den Quelltabellen kann nicht eingelesen werden
→ Oracle 12g: Fehler bei Duplikaten *(Ein Artikel hat mehrere Lagerplätze)*

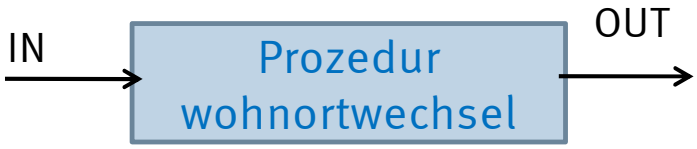
Inhaltsübersicht

Programmierkonzepte (Datenbank)

- Gespeicherte Funktionen (stored function)

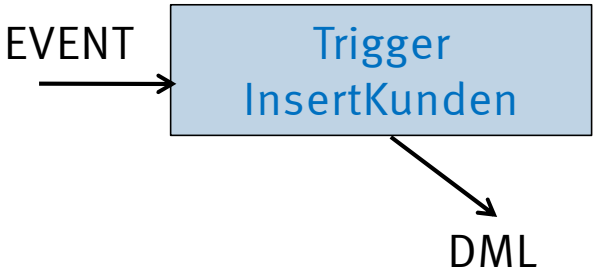

```
SELECT kundenanrede(Kundennummer)
FROM   Kunde
```

- Gespeicherte Prozeduren (stored procedures)




```
CALL kundenliste('Dortmund')
```

- Eventgesteuerte Prozeduren (Trigger)



Anwendungsfälle für Datenbank-Trigger

Trigger werden zur Konsistenzüberwachung und zur Herstellung der Datenintegrität (Wahrung der Korrektheit von Daten) genutzt.

- Anwendungsfälle sind …
  - die Nachverarbeitung von Daten in der Datenbank

    Beispiel:

    Einfügen eines Willkommenspräsents in den Warenkorb eines neuen Kunden
  - die Integritätsprüfung vor der Ausführung der Datenänderung

    Beispiel:

    Das Geburtsdatum darf nicht in der Zukunft liegen
  - die Integritätsherstellung

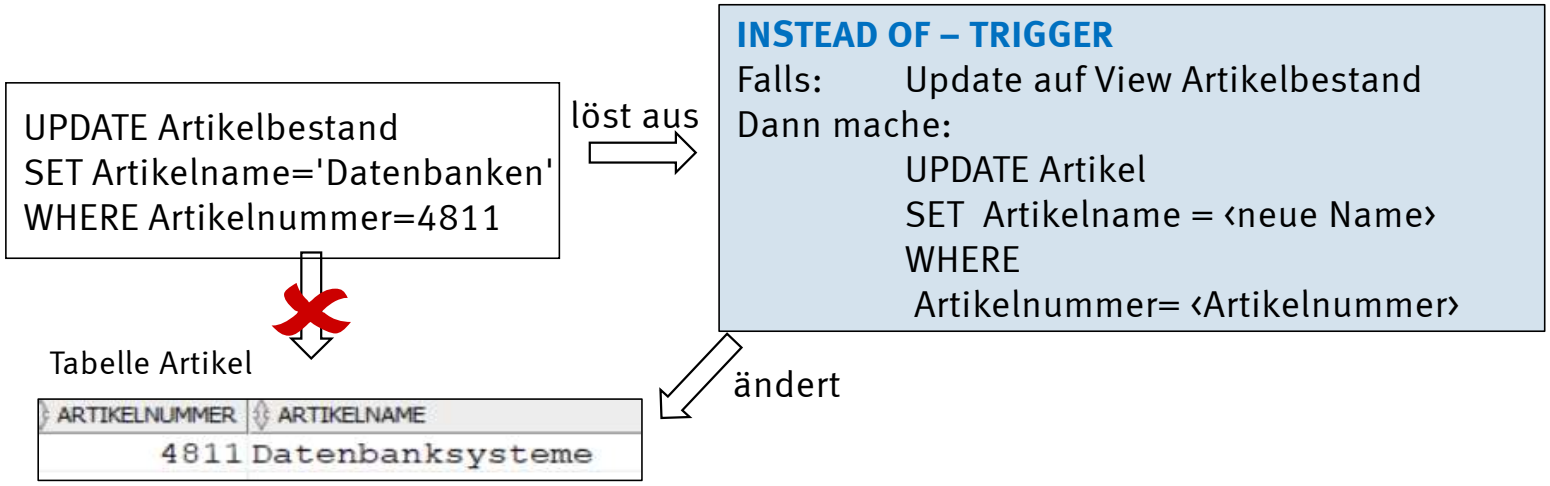
    Beispiel:

    Freigabe des Lagerplatzes beim Löschen eines Artikels

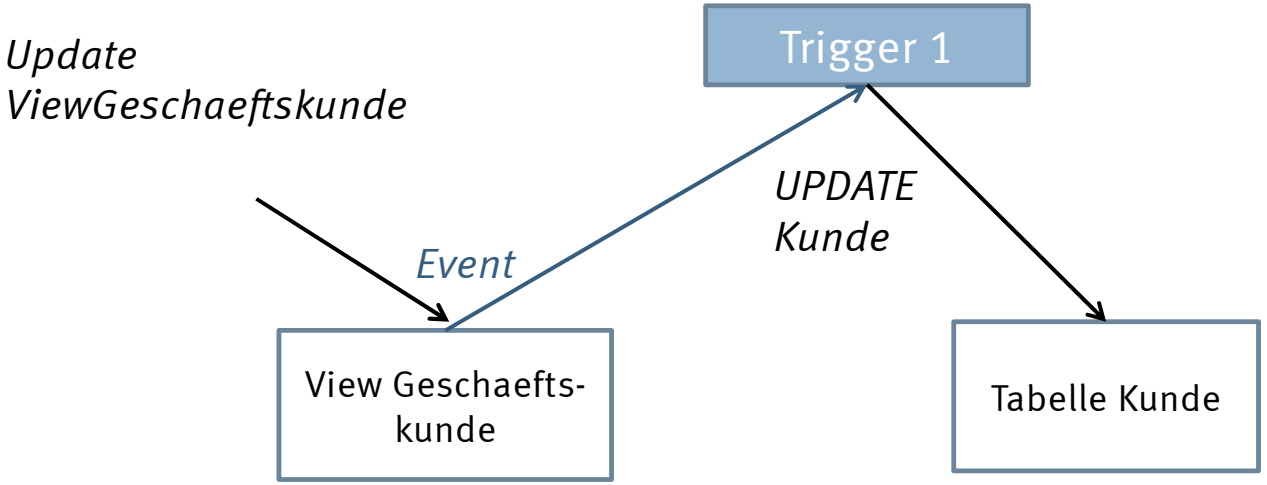
Eventsteuerung in SQL : Trigger

Ein **Trigger** ist ein ausführbares Programm, welches durch ein Ereignis (Event), welches einer Ausführungsbedingung genügt, ausgelöst wird.

Beispiel:

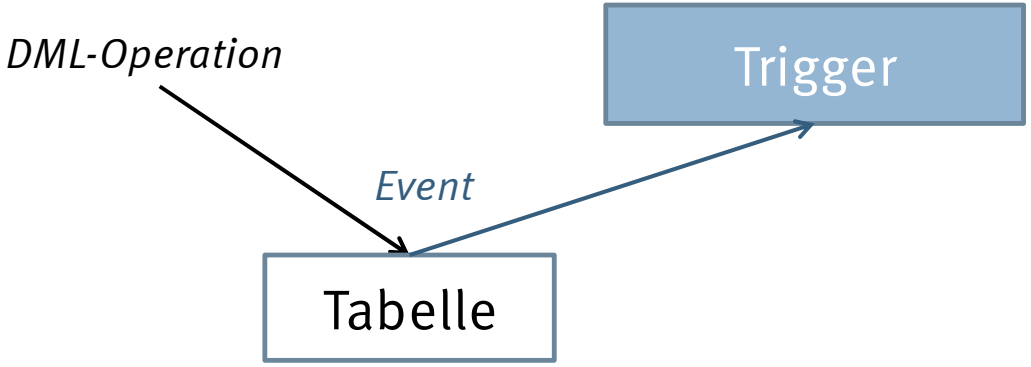


Weitere Triggerarten



Event	MySQL	Oracle
DML-Operation (auf Tabelle)	Ja	DML-Trigger
DML-Operation (auf View)	Nein	Instead-Of-Trigger
DDL-Operation	Nein	DDL-Trigger
DB-Operation (Startup, Shutdown, Logon, Logoff, Servererror)	Nein	Database-Event-Trigger
Abbruch eines Statements (z.B. Überschreitung des Tablespace-Quota)	Nein	After-Suspend Trigger

Syntax eines DML-Triggers (Oracle)



ECA-Regeln

EVENT →

CONDITION →

ACTION →

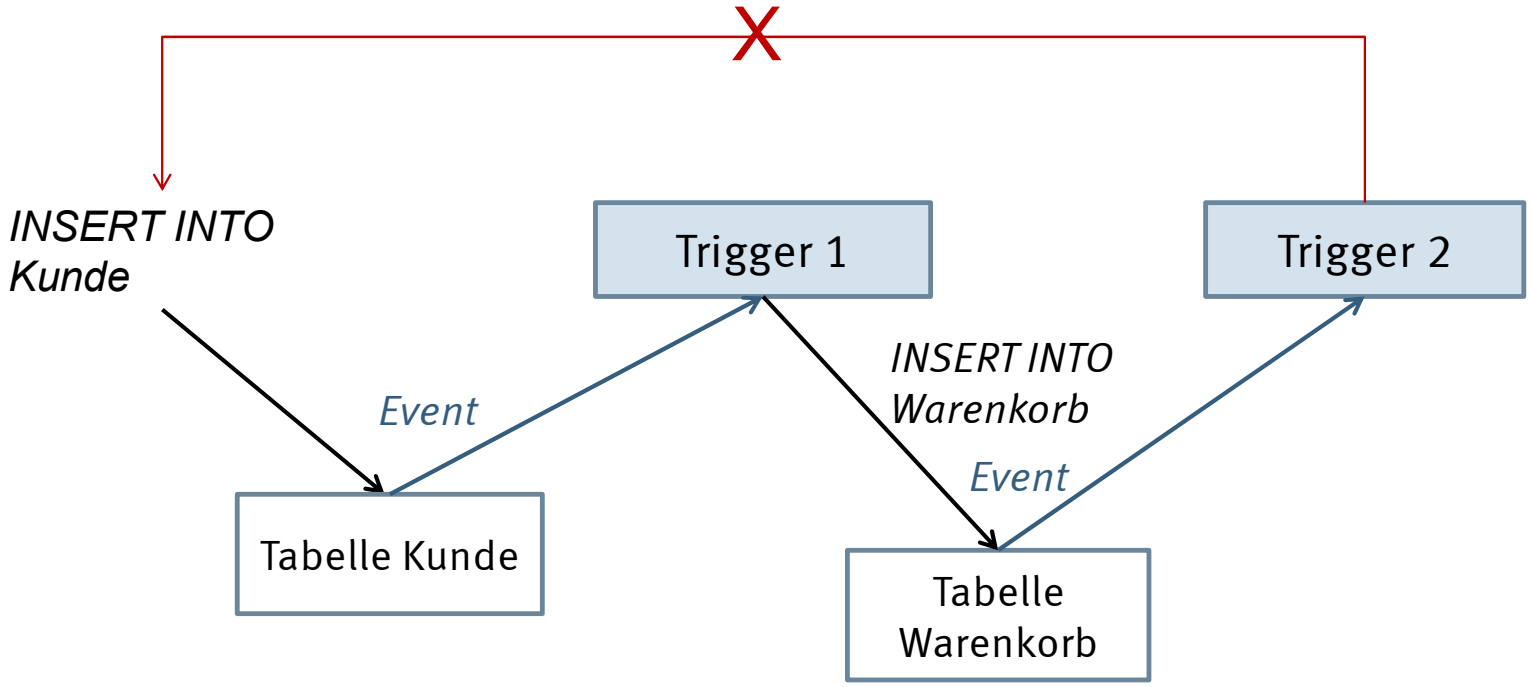
```
CREATE OR REPLACE TRIGGER ‹trigger_name›

{ BEFORE|AFTER}

{INSERT OR UPDATE [of ‹spalte1, spalte2, …›]
        OR DELETE} ON ‹tabellenname›

[FOR EACH ROW]
[WHEN ‹ausführungsbedingung›]

‹anweisungen›
```

Einschränkung des Events, z.B. nur auf Änderung des Attributs Wohnort

Weitere Einschränkung des Events, z.B. auf einen Insert von Kunden aus Dortmund

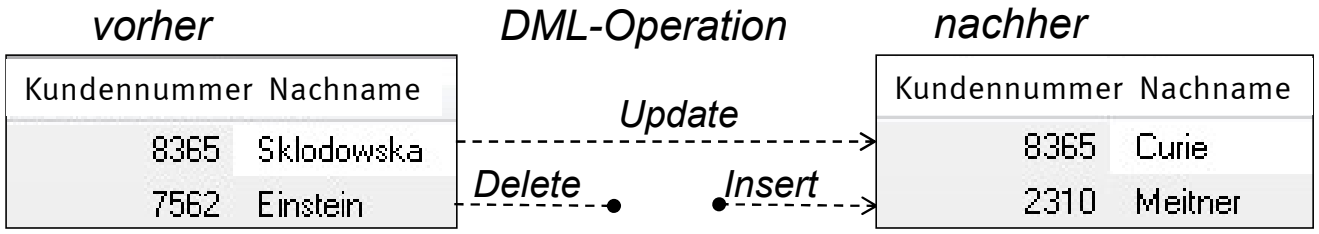
Eventsteuerung in SQL - Trigger

Eine Schleifenbildung durch Trigger ausgelöste DB-Operationen darf nicht erfolgen. Daher kann ein Trigger nur DML-Operation ausführen, die sich **nicht** auf die zugehörende Tabelle beziehen.



Tupelzugriff im zeilenorientierten Trigger

Der Zugriff auf Tupelvariablen erfolgt mittels der old- und new-Präfixe. Old wird für den vorherigen Attributwert und new für den geänderten Wert verwendet.



Im Trigger zur DML-Operation	old. (alter Wert)	new. (neuer Wert)	Beispiel
INSERT	null	neuer Wert	:new.Nachname='Meitner'
UPDATE	vorheriger Wert	neuer Wert	:old.Nachname='Sklodowska' :new.Nachname='Curie'
DELETE	vorheriger Wert	null	:old.Nachname='Einstein'

Beispiel Instead-of-Trigger (Oracle)

Die Anweisungen des Instead-of-Trigger werden anstelle der Änderungsoperation auf dem nicht direkt änderbaren View ausgeführt. Im Beispiel wird so die Änderung des Wohnortes ermöglicht, jedoch nicht des Kundennamens.

Nicht direkt änderbares View:

```
CREATE VIEW view_geschaeftskunde
AS
SELECT Kundennummer, Nachname, Ort, COUNT(Anzahl) Anzahl
FROM    Kunde NATURAL JOIN Warenkorb
WHERE ANREDE IS NULL
GROUP BY Kundennummer, Nachname, Ort
```

Event:

```
UPDATE view_geschaeftskunde
SET Ort = 'Bochum'
WHERE  Kundennummer=8523
```

Trigger:

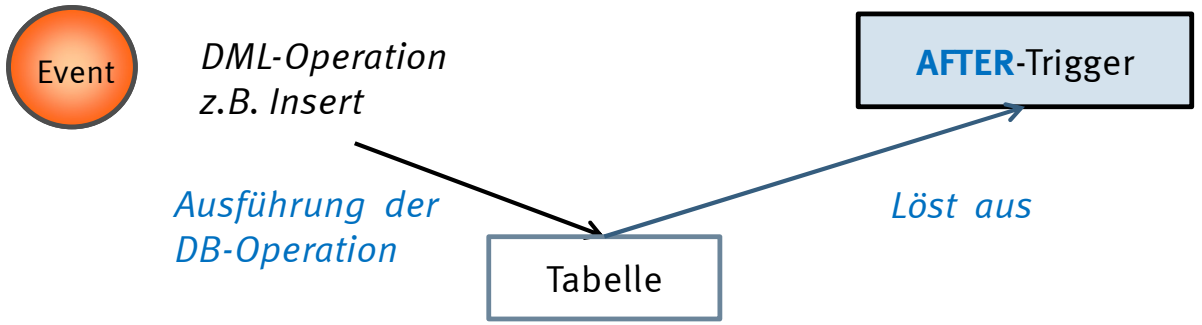
```
CREATE OR REPLACE TRIGGER geschaeftskunde_update
INSTEAD OF UPDATE ON view_geschaeftskunde
FOR EACH ROW
BEGIN
        UPDATE Kunde
        SET Ort = :NEW.ORT
        WHERE Kundennummer= :OLD.Kundennummer;
END;
```

Inhaltsübersicht

Zeitpunkt der Triggeraktivierung

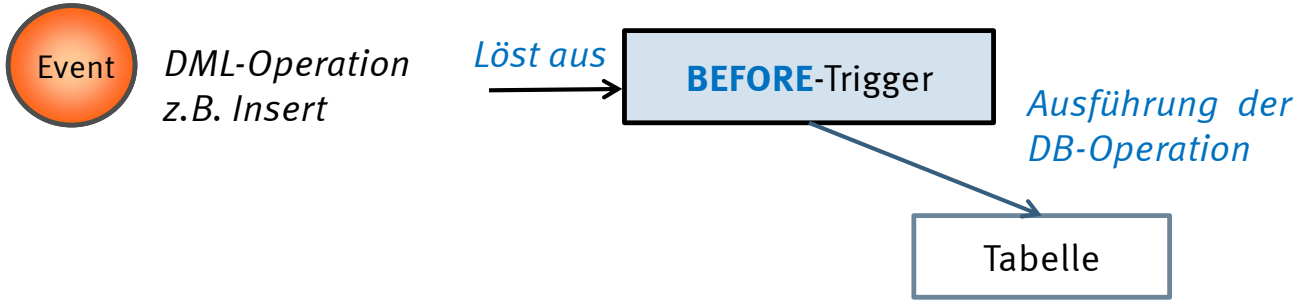
After-Trigger

- Zuerst wird die auslösende Datenbankoperation (hier: Insert) ausgeführt, danach wird der Code des After-Triggers durchgeführt.



Before-Trigger

- Zuerst wird der Code des Before-Triggers ausgeführt,
  danach wird die auslösende Datenbankoperation (hier: Insert) durchgeführt.



Implementierung eines Triggers – Beispiel

Bei einem Trigger handelt es sich um ein Datenbankobjekt. Ein Trigger wird mit dem CREATE Trigger Befehl erstellt und mit dem DROP Trigger Befehl gelöscht. Wird eine Tabelle gelöscht, dann werden die zugehörenden Trigger mitgelöscht.

Event

INSERT INTO Kunde VALUES (3333, 'Herr', ...)

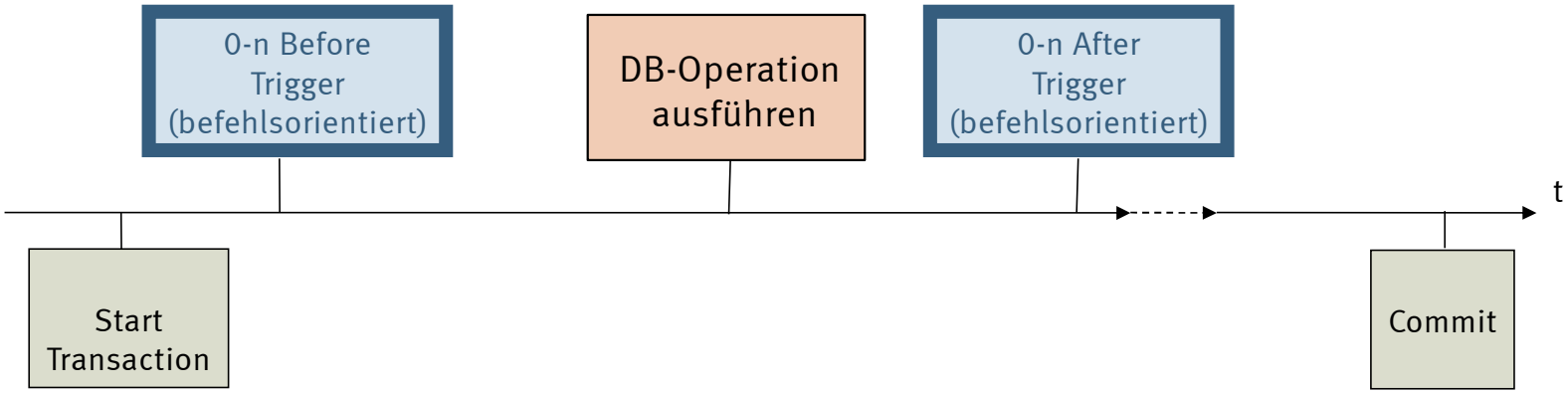
Trigger kompilieren

```
1  CREATE OR REPLACE TRIGGER NEUER_KUNDE
2  AFTER INSERT ON KUNDE
3  FOR EACH ROW
4  BEGIN
5    NULL;
6  END;
```

Dieser Trigger wird durch das Event "INSERT ON Kunde" ausgelöst und nach (= AFTER) dem Event (Einfügen eines Kunden) ausgeführt.

Befehlsorientierte Trigger

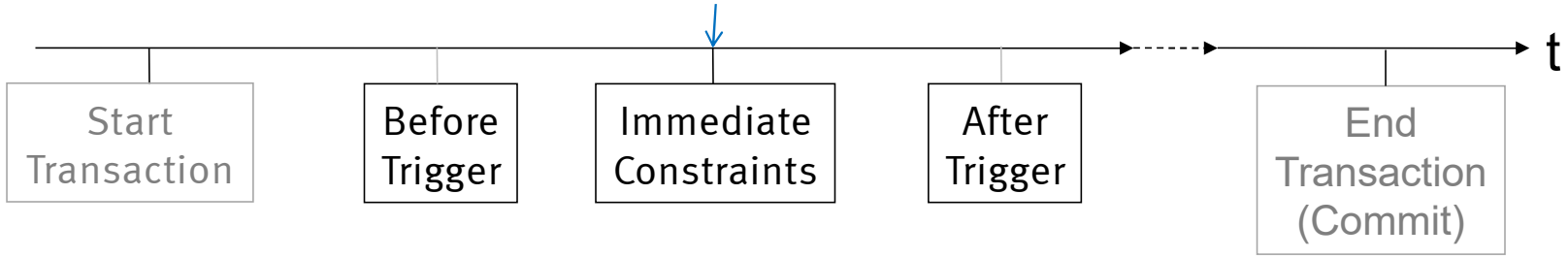
Befehlsorientierte Trigger werden nur 1x pro Event (=Befehl) ausgeführt, zeilenorientierte Trigger für jedes betroffene Tupel. Der Zugriff auf die Tupelattribute über die Schlüsselwörter OLD und NEW ist **nicht** möglich.



Fehler während der Triggerausführung

Der Zeitpunkt der Triggeraktivierung (Before, After) beschreibt, ob der Trigger vor oder nach der auslösenden DML-Operation ausgeführt wird.

Direkte Überprüfung der statischen Integritätsbedingungen der (auslösenden) DML-Operation (z.B. Fremdschlüsselbedingungen)



Fehler während ...	Wirkung eine Fehlers
BEFORE-Trigger	DML-Operation wird nicht ausgeführt
DML-Operation	AFTER-Trigger wird nicht ausgeführt
AFTER-Trigger	Gesamte Anweisung scheitert und Tabellenänderung wird zurückgenommen (Transaktion rollback)

BEFORE oder AFTER-Trigger?

- Ist ein BEFORE- oder ein AFTER-Trigger zu wählen?
  1. Einfügen eines Willkommenspräsents in den Warenkorb eines neuen Kunden
  2. Das Geburtsdatum darf nicht in der Zukunft liegen
  3. Wenn ein Artikel gelöscht wird, dann soll auch der zugehörende Lagerplatz freigegeben werden.

https://vote.fh.do/PXCY

Inhaltsübersicht

Befehlsorientierter Trigger

- Beispiel:
  Es sollen in der Tabelle aenderungen die ausgeführten DML-Operationen mit dem zugehörenden Login auf der Tabelle Kunde dokumentiert werden.

```
CREATE TABLE aenderungen(
   log_datum DATE PRIMARY KEY,
   login     VARCHAR(20),
   aenderung VARCHAR(10));
```

```
CREATE OR REPLACE
TRIGGER KUNDEN_MODIFICATION_TRIGGER
AFTER INSERT OR DELETE OR UPDATE ON KUNDE
DECLARE  action VARCHAR(10);
         userlogin VARCHAR(20);
BEGIN
          select user INTO userlogin from dual;
          IF INSERTING THEN action := 'Insert';
          ELSIF UPDATING THEN action := 'Update';
          ELSIF DELETING THEN action := 'Delete';
          END IF;
          INSERT INTO aenderungen (log_datum, login, aenderung)
          VALUES (SYSDATE, userlogin, action);
END;
```

Event abfragen

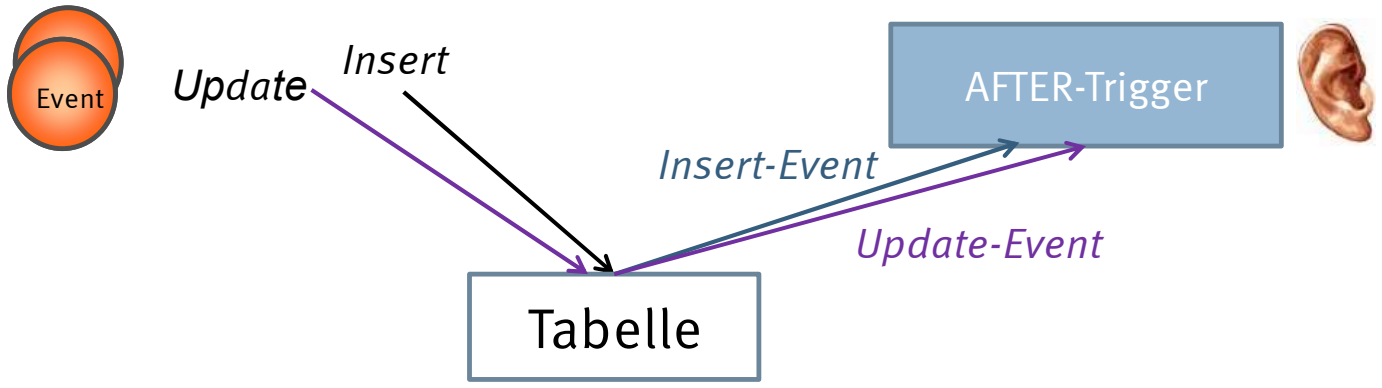
Im Trigger Events erkennen (Oracle)

In Oracle können mehrere Events denselben Trigger auslösen. Zur Unterscheidung, welches Event den Trigger ausgelöst hat, können die booleschen Systemvariablen

- Inserting
- Deleting
- Updating
- Updating(‹Spalte›)

ausgewertet.

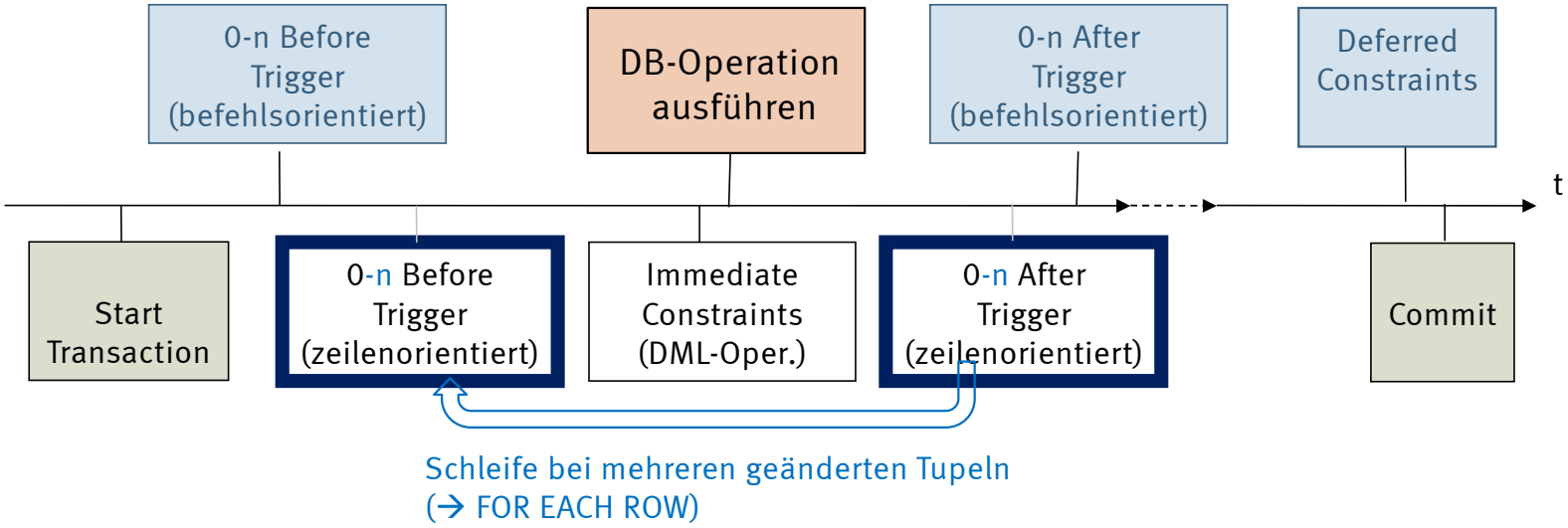
```
IF deleting THEN … END IF;
```



Inhaltsübersicht

Zeilenorientierte Trigger

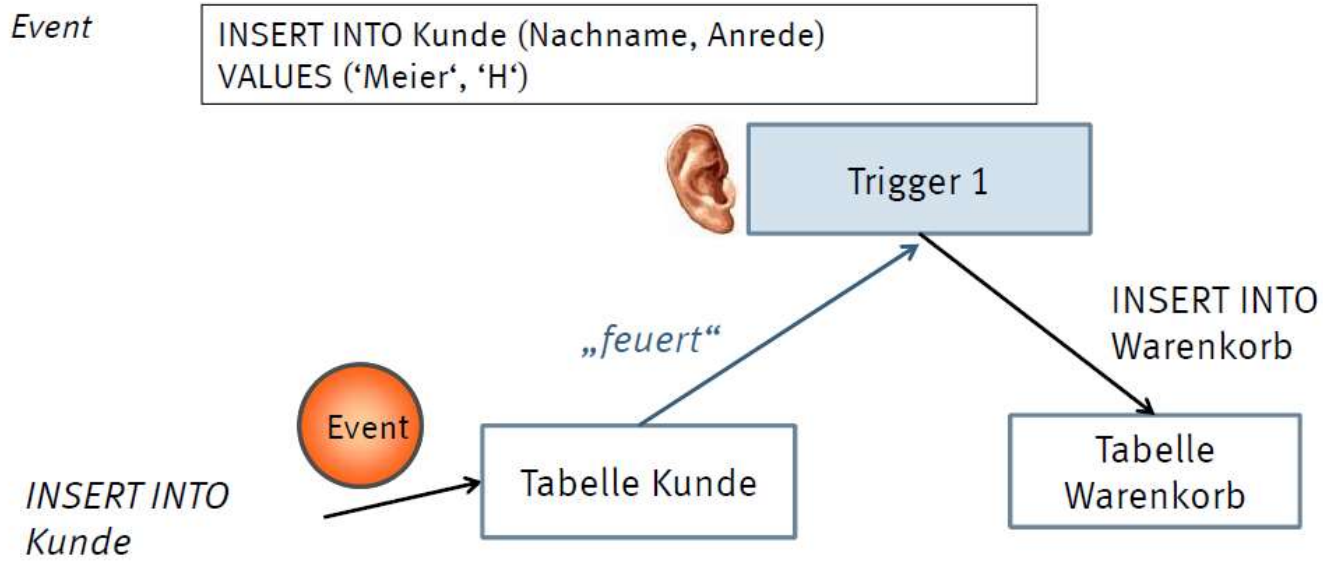
Ein **zeilenorientierter Trigger** enthält die Klausel "FOR EACH ROW". Der Code des Triggers wird für jedes Tupel der auslösenden SQL-Befehls ausgeführt. Der Zugriff auf die Tupelattribute über die Schlüsselwörter OLD und NEW ist möglich.



Beispiel

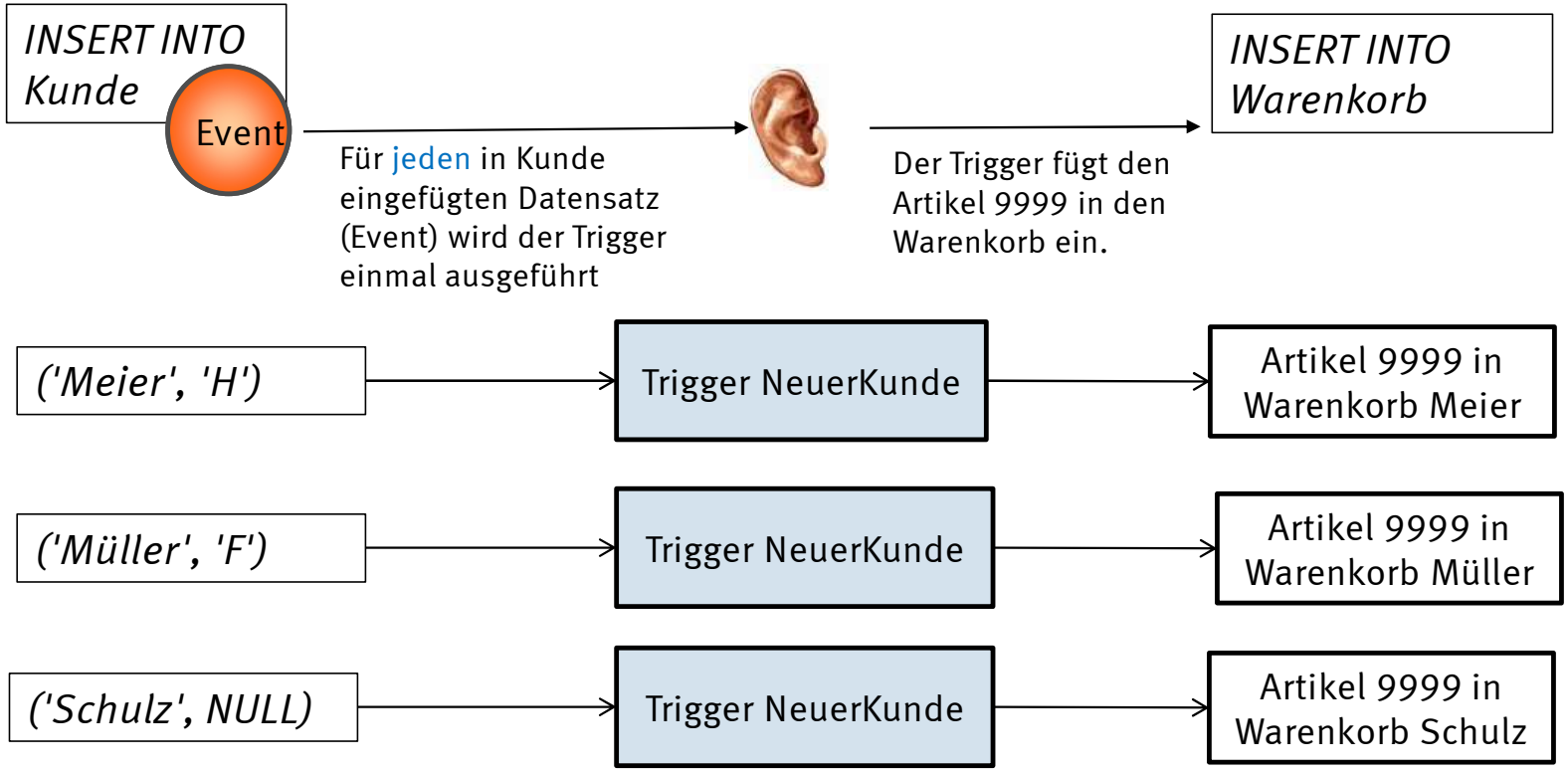
Beispiel:
„Jeder neue Kunde erhält den Artikel 9999 als Willkommenspräsent im Warenkorb."

Lösungsidee:
Das Einfügen eines neuen Kunden (Event) löst die Ausführung (Aktivierung) eines Datenbankprogramms (Trigger) aus.



Zeilenorientierter Trigger

Ein zeilenorientierter Trigger enthält die Klausel "FOR EACH ROW". Der Code des Triggers wird für jedes Tupel der auslösenden SQL-Befehls ausgeführt.



Trigger in Oracle

In einem zeilenorientierten Trigger kann auf die Attribute des bearbeiteten Tupels (Tupelvariable) vor und nach der beabsichtigten Änderung zugegriffen werden. Der Zugriff auf den neuen Attributwert (hier die Kundennummer) des durch den Trigger bearbeiteten Datensatzes erfolgt mit Hilfe des NEW-Präfix.

INSERT INTO Kunde VALUES (3333, 'Herr', ...)

```
1  create or replace Trigger Neuer_kunde
2  AFTER INSERT ON Kunde
3  FOR EACH ROW
4  BEGIN
5    INSERT INTO Warenkorb (kundennummer, artikelnummer, anzahl)
6    VALUES (:new.kundennummer, 9999, 1);
7  END;
8
```

Inhaltsübersicht

Statische und dynamische Integritätsbedingungen

Statische Integritätsbedingungen müssen zu jedem Zeitpunkt eingehalten werden.

Beispiel:

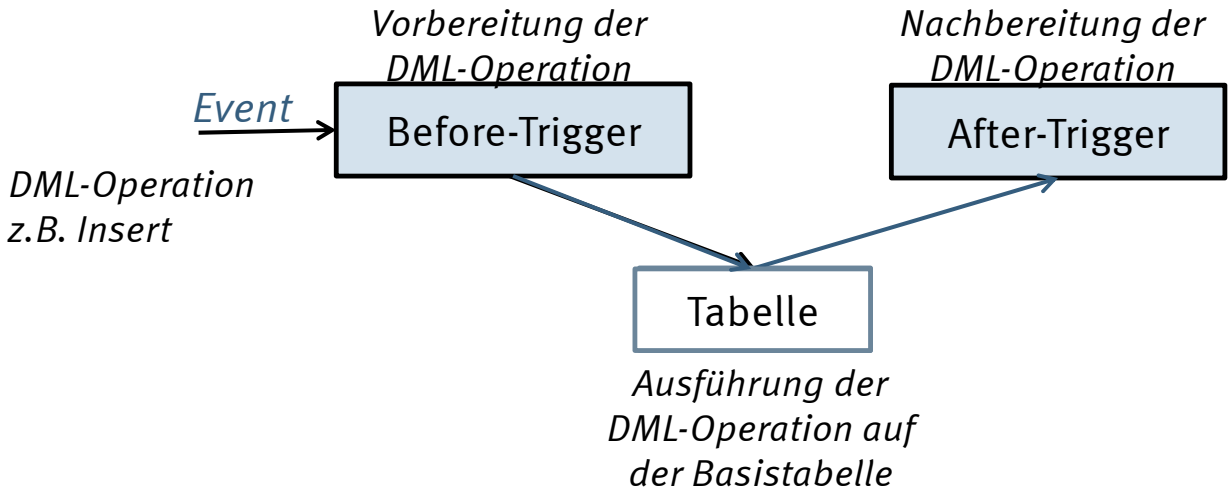
Als Anrede eines Kunden sind nur die Werte „Herr“ und „Frau“ erlaubt.

Dynamische Integritätsbedingungen beziehen sich auf den **aktuellen** Inhalt der Datenbanktabellen und deren Änderungen.

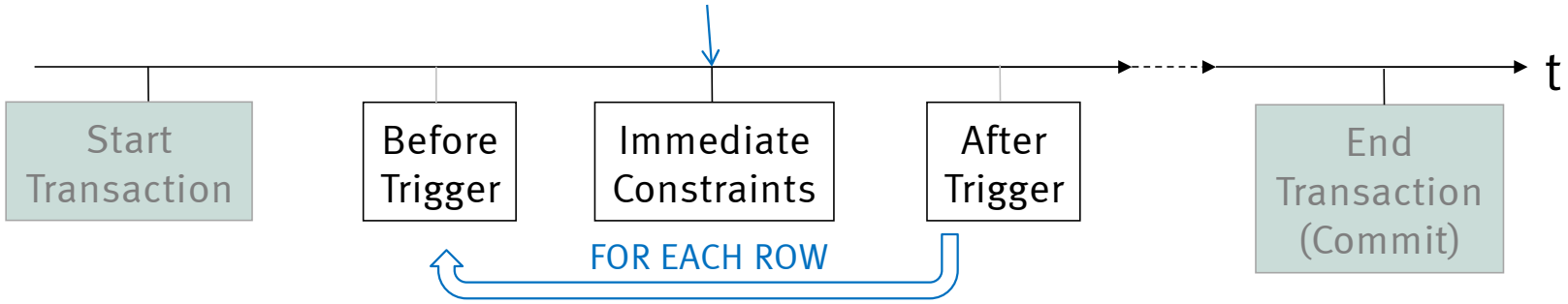
Arten dynamischer Integritätsbedingungen

- Übergangsbedingungen
  - [Zustands-]Änderungen (Create, Einfügen, Ändern, Löschen) dürfen nur unter bestimmten Bedingungen durchgeführt werden.
  - Beispiele:
    - Das Geburtsdatum darf nicht in der Zukunft liegen.
    - Ein Artikel darf nur gelöscht werden, wenn sein Lagerbestand 0 ist.
- Temporale Bedingungen
  - Änderungen sind nur unter bestimmten zeitlichen Bedingungen erlaubt:
  - Beispiel:
    - Ein Kunde erhält im Monat seines Geburtstags einen 10%igen Geburtstagsrabatt.

Integritätsprüfung



Direkte Überprüfung der statischen Integritätsbedingungen der (auslösenden) DML-Operation (z.B. Fremdschlüsselbedingungen)



Schleife bei mehreren geänderten Tupeln

Trigger zur Integritätsherstellung

Wurde „Fräulein“ als Anrede eingegeben, dann soll dies zu „Frau“ korrigiert werden.

Event

```
INSERT INTO Kunde (Kundennummer, Nachname,  Vorname, Anrede,    Geburtsdatum)
VALUES(4711,              'Müller',  'Mia',    ' Fräulein'    ,'1968-12-24'    );
```

⇩ Trigger wird ausgelöst

Trigger ändert Anrede von 'Fräulein' zu 'Frau', so dass die statische Integritätsbedingung CHECK Anrede IN ('Herr', 'Frau') erfüllt wird.

```
IF :new.Anrede='Fräulein' THEN
            :new.Anrede:='Frau'
END IF;
```

⇩ INSERT-Statement wird ausgeführt

Kunde

Kundennummer	Nachname	Vorname	Anrede	Geburtsdatum
4711	Müller	Mia	Frau ✓	1968-12-24

Inhaltsübersicht

Beispiel – Dynamische Integritätsbedingung

Trigger werden genutzt, um dynamische Integritätsbedingungen umzusetzen oder die Integrität wiederherzustellen.

Beispiel

1. Ein Artikel darf nur gelöscht werden, wenn der zugehörende Lagerbestand gleich 0 ist.
2. Der Lagerplatz zu einem gelöschten Artikel muss freigegeben werden.

DELETE FROM Artikel WHERE Artikelnummer = 4813

Fehler!

Artikel

Artikel-nummer	Artikel-name	Preis	Ausgabe
4812	Datenbanke	29,95 €	gebunden
4813	Märchen vo	12,90 €	broschiert

Lager

Lager-nummer	Standort	ANummer	Lager-bestand
27135	R235	4812	18
27423	R371	4813	0

???

```
CREATE TABLE Lager(
...
FOREIGN KEY(ANummer)
    REFERENCES Artikel (Artikelnummer)
 ON DELETE RESTRICT)
```

Beispiel – Dynamische Integritätsbedingung

Trigger werden genutzt, um dynamische Integritätsbedingungen umzusetzen oder die Integrität wiederherzustellen.

Beispiel

1. Ein Artikel darf nur gelöscht werden, wenn der zugehörende Lagerbestand gleich 0 ist.
2. Der Lagerplatz zu einem gelöschten Artikel muss freigegeben werden.

Event

DELETE FROM Artikel WHERE Artikelnummer = 4713

⇩

Trigger prüft die dynamische Integritätsbedingung:
- SELECT: Ermittlung des Lagerbestandes zu Artikel 4713
- WENN Lagerbestand › 0
- DANN Fehlermeldung
- SONST zugehörender Lagerplatz wird freigegeben

Wird ein BEFORE- oder AFTER-Trigger benötigt?

Implementierung des Triggers

Triggerdeklaration Event: Delete

Variablendeklaration

Ermittlung des zugehörenden Lagerbestandes

Prüfung der Integritätsregel

Herstellung der referentiellen Integrität (Freigabe des Lagerplatzes)

Fehlermeldung signalisieren

```
CREATE OR REPLACE TRIGGER ARTIKELLOESCHEN
BEFORE DELETE ON ARTIKEL
FOR EACH ROW
DECLARE
   lbst         Integer;
   LagerNichtLeer EXCEPTION;
BEGIN
   SELECT SUM(Lagerbestand) INTO lbst
   FROM   Lager
   WHERE  ANummer=:old.Artikelnummer;
   IF lbst>0 THEN
        RAISE LagerNichtLeer;
   ELSE
      UPDATE Lager
      SET ANummer= NULL, Lagerbestand=NULL
      WHERE ANummer=:old.Artikelnummer;
   END IF;
EXCEPTION
   WHEN LagerNichtLeer
   THEN raise_application_error (-20501, 'Lager nicht leer!
Lagerbestand='|| lbst);
END;
```

Vorsicht Falle!


```
Fehler beim Start in Zeile : 2 in Befehl -
DELETE FROM ARtikel where Artikelnummer=4811
Fehlerbericht -
SQL-Fehler: ORA-20501: Lager nicht leer! Lagerbestand=2
ORA-06512: at "SAATZBH.ARTIKELLOESCHEN", line 17
ORA-04088: error during execution of trigger 'SAATZBH.ARTIKELLOESCHEN'
```

```
CREATE TABLE Lager(
FOREIGN KEY(ANummer)
    REFERENCES Artikel (Artikelnummer)
 ON DELETE RESTRICT)
```

```
Fehler beim Start in Zeile : 2 in Befehl -
DELETE FROM ARtikel where Artikelnummer=4811
Fehlerbericht -
SQL-Fehler: ORA-04091: table SAATZBH.LAGER is mutating, trigger/function may not see it
ORA-06512: at "SAATZBH.ARTIKELLOESCHEN", line 5
ORA-04088: error during execution of trigger 'SAATZBH.ARTIKELLOESCHEN'
04091. 00000 -  "table %s.%s is mutating, trigger/function may not see it"
*Cause:    A trigger (or a user defined plsql function that is referenced in
           this statement) attempted to look at (or modify) a table that was
           in the middle of being modified by the statement which fired it.
*Action:   Rewrite the trigger (or function) so it does not read that table.
```

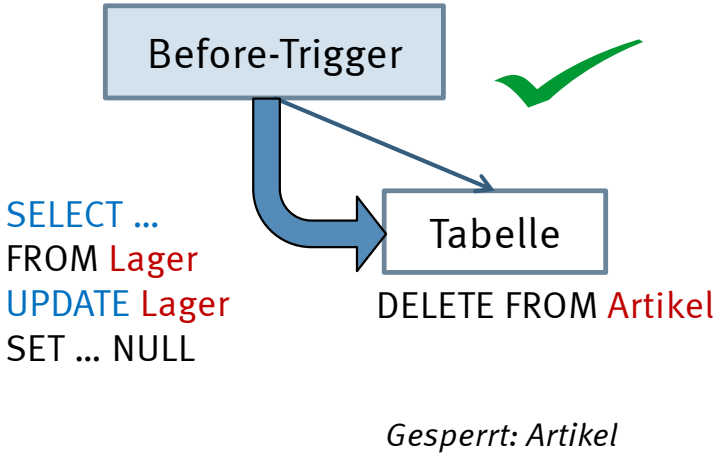
Alternative Lösung zum Entfernen von On Delete Set Null?

```
CREATE TABLE Lager(
FOREIGN KEY(ANummer)
    REFERENCES Artikel (Artikelnummer)
 ON DELETE SET NULL)
```

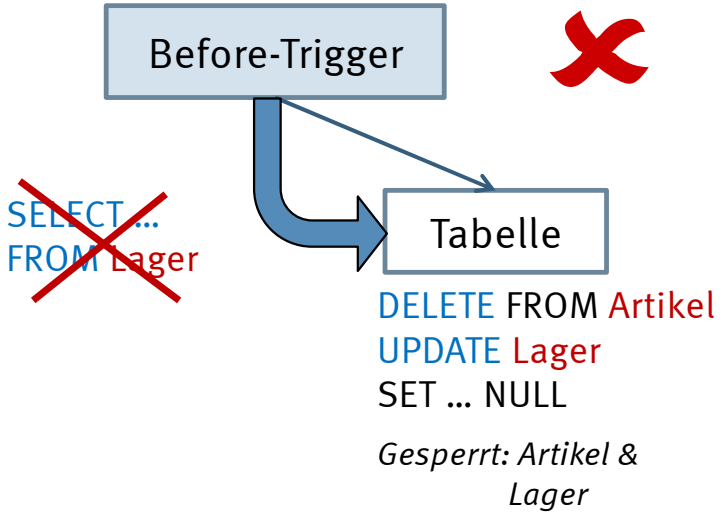
Tabellenzugriff im Trigger

Ein Before-Trigger erfordert Zugriff auf die dem Trigger zugeordnete Tabelle. Diese Tabelle ist aber aufgrund der auslösenden (DML-)Operation bereits für weitere Zugriffe gesperrt (exklusive Schreibsperre). Daher ist im Before-Trigger nur ein Zugriff auf das gerade bearbeitete Tupel erlaubt.

```
CREATE TABLE Lager(
FOREIGN KEY(ANummer)
    REFERENCES Artikel (Artikelnummer)
 ON DELETE RESTRICT)
```

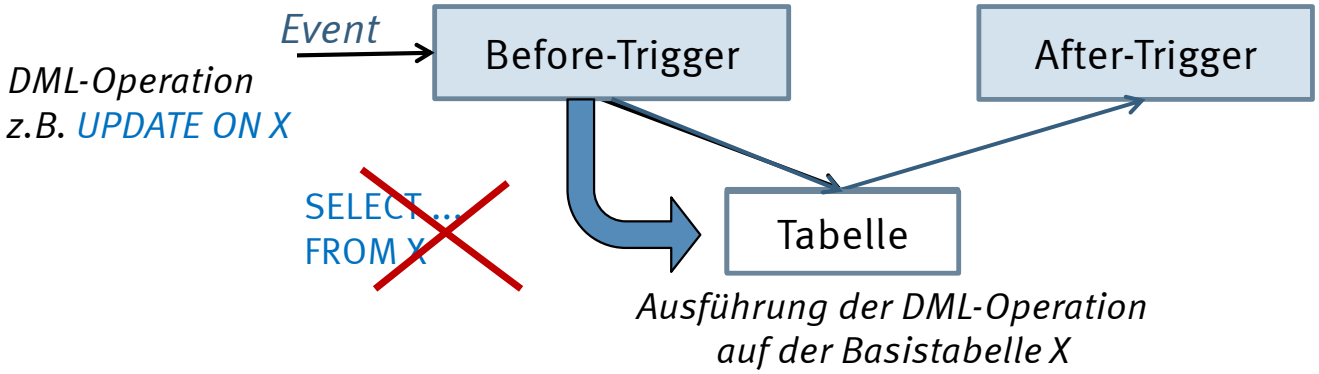



```
CREATE TABLE Lager(
FOREIGN KEY(ANummer)
    REFERENCES Artikel (Artikelnummer)
 ON DELETE SET NULL)
```



Tabellenzugriff im Trigger

Ein Before-Trigger kann nicht auf die Tabelle selbst zugreifen, auf die der Trigger definiert ist. Die Tabelle ist aufgrund der auslösenden (DML-)Operation bereits für weitere Zugriffe gesperrt (exklusive Schreibsperre). Daher ist im Before-Trigger nur ein Zugriff auf das gerade bearbeitete Tupel erlaubt.



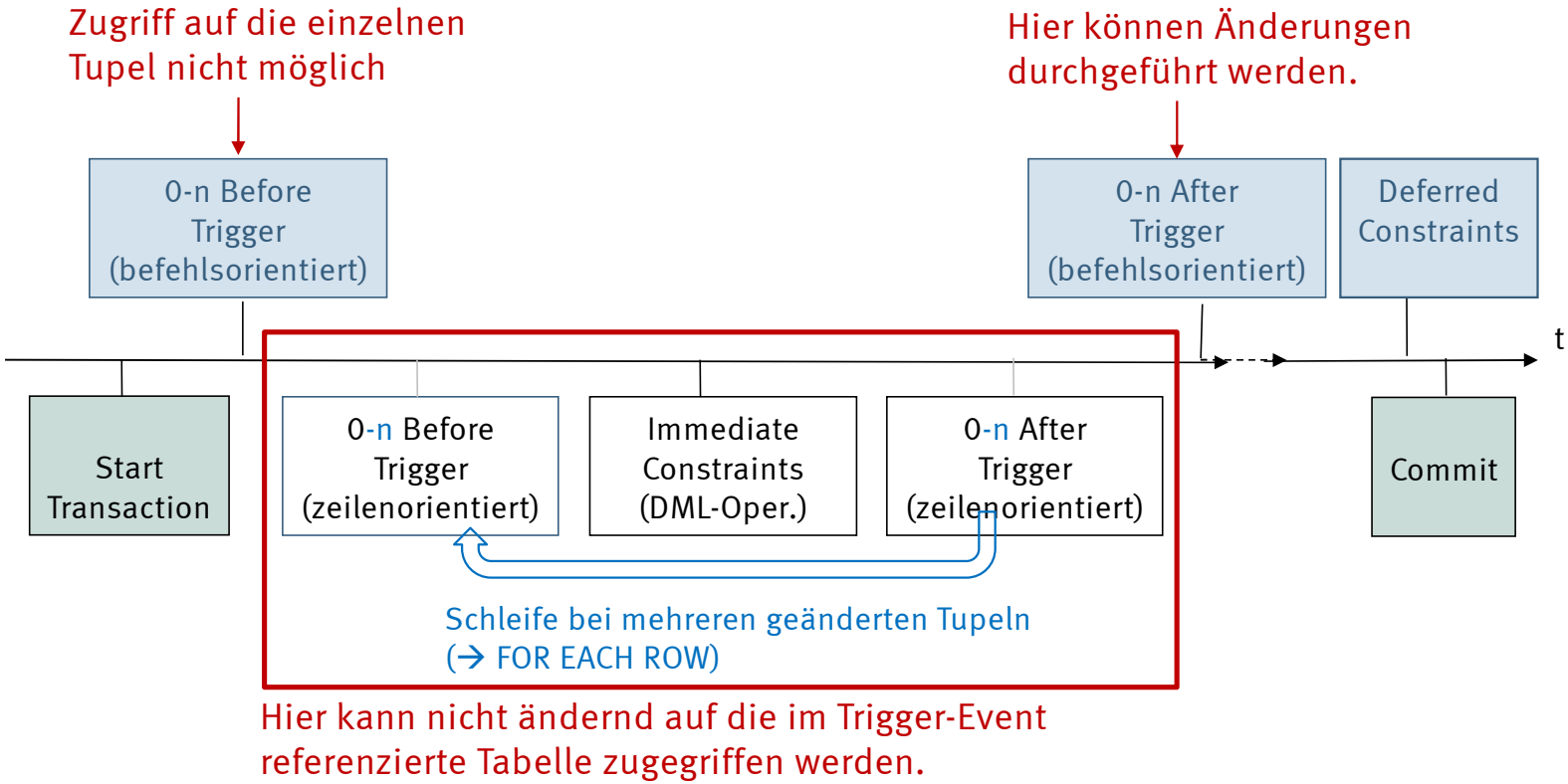
Und was kann ich machen,
wenn ich aber auf diese Tabelle zugreifen muss?

Ausführungsreihenfolge der Integritätsprüfung

Lösungsidee

1. Schritt: In einem BEFORE Trigger die Änderungen in eine Hilfstabelle schreiben
2. Schritt: Die Hilfstabelle mit einem befehlsorientierter After-Trigger auswerten
3. Schritt: Die Inhalte der Hilfstabelle wieder löschen

Alternative (Oracle): Compound Trigger



Inhaltsübersicht

Übersicht

	Gespeicherte Prozedur	Gespeicherte Funktion	Trigger
Aufruf	CALL	SELECT	DML-Operation DDL (Oracle) DCL (Oracle)
Aufruf-parameter	IN INOUT OUT	IN	-
Rückgabe-werte	Mehrere OUT-Parameter	Ein Wert	-
Erlaubte Befehle	DRL DML DDL DCL	DRL	DRL DML DDL (Oracle) DCL (Oracle)
Aufruf von	Funktionen Prozeduren	Funktionen	Funktionen Prozeduren

i.d.R. nicht direkt portierbar zwischen verschiedenen RDBMS

Reflexion

Welche Arten von Triggern gibt es und wofür werden sie verwendet?

Erläutern Sie, wann ein Trigger ausgeführt wird.

Erläutern Sie anhand von Beispielen, wann ein Before und wann ein After-Trigger verwendet wird.

Konzipieren und Implementieren Sie einen Trigger

Wodurch unterscheiden sich befehls- und zeilenorientierte Trigger?

Was ist ein Instead-Of-Trigger und wozu dient dieser?

Vielen Dank
für Ihre Aufmerksamkeit !