Dokumentation EDR DBS

# Titel des Projekts

EDR - Einfach, Direkt, Regional - Datenbankmodell

# Projektteilnehmer

Projektleiter: Florian Weiss - wi18b016

Jakob Neuhauser - wi18b036

Paul Pavlis - wi18b025

Albert Schleidt - wi18b088

# Projektinhalt

Innerhalb dieses Projekts werden wir ein Datenbankmodell sowie eine grafische Benutzeroberfläche für die Anwendung “EDR - Einfach, Direkt, Regional” entwickeln.

Es geht um ein Portal, dass Lebensmittel aus der Region einfach angeboten werden sollen. Jeder User soll, nur mit einem Smartphone, selber in der Lage sein Lebensmittel gratis anzubieten oder zu kaufen. Die Suche nach Lebensmittel soll regional sein und oder auch kategorial. Den Usern des Portals soll eine Social Media Komponente zur Verfügung stehen. Damit sollen User kommunizieren, von Bewertungen bis Direktnachrichten.

Zusätzlich soll es eine Saisonale Komponente geben, die den Usern die reifen Früchte der Region und verfügbare Obstbäume anzeigen. Unter Verfügbaren Obstbäume sind z.B.: von Usern zur Ernte erfasste Bäume in Privaten oder Öffentlichen Raum gemeint.

Zusätzlich soll eine voraus Planung für Landwirte möglich sein. Damit ist gemeint, dass schon vor dem Schlachten das Fleisch angeboten werden kann mit einem entsprechendem Schlachttermin. Damit kann Angebot und Nachfrage optimiert werden und die Preise für Anbieter und Konsument optimal gestalten werden.

Inhalt

[Titel des Projekts 1](#_Toc44755283)

[Projektteilnehmer 1](#_Toc44755284)

[Projektinhalt 1](#_Toc44755285)

[Inhalt 2](#_Toc44755286)

[Tabellen 3](#_Toc44755287)

[Name: User 3](#_Toc44755288)

[Name: Nachrichten 4](#_Toc44755289)

[Name: Rolle 5](#_Toc44755290)

[Name: Adresse 6](#_Toc44755291)

[Name: Ort 6](#_Toc44755292)

[Name: Angebot 7](#_Toc44755293)

[Name: Transaktion 8](#_Toc44755294)

[Name: Angebotskategorie 9](#_Toc44755295)

[Name: Produkt 9](#_Toc44755296)

[Name: Produkt\_Angebot 10](#_Toc44755297)

[Name: Bio\_Faktor 11](#_Toc44755298)

[Name: Saison 11](#_Toc44755299)

[Name: User\_Rolle\_Log 12](#_Toc44755300)

[VIEWs 13](#_Toc44755301)

[PROZEDUREN 14](#_Toc44755302)

[Name: sp\_add\_ort 14](#_Toc44755303)

[Name**:** sp\_add\_adresse 14](#_Toc44755304)

[Name**:**  sp\_add\_rolle 14](#_Toc44755305)

[Name**:**  sp\_add\_user 15](#_Toc44755306)

[Name**:**  sp\_delete\_user 15](#_Toc44755307)

[Name: sp\_addProduct 15](#_Toc44755308)

[Name: sp\_addOffer 16](#_Toc44755309)

[Name: sp\_addTransaction 16](#_Toc44755310)

[Name: sp\_sendMessage 16](#_Toc44755311)

[Name: sp\_updateUserRolle 17](#_Toc44755312)

[Name: sp\_addProductOffer 17](#_Toc44755313)

[FUNKTIONEN 17](#_Toc44755314)

[Name: f\_transaktion\_userx\_cur 17](#_Toc44755315)

[Name: f\_Nachrichten\_UserX\_cur 18](#_Toc44755316)

[Name: f\_angebote\_saison\_cur 18](#_Toc44755317)

[Name: f\_filter\_ang\_prodname\_cur 18](#_Toc44755318)

[TRIGGER 19](#_Toc44755319)

[Name: tr\_br\_i\_welcomeMessage 19](#_Toc44755320)

[Name: tr\_br\_d\_deleteUserCompletely 19](#_Toc44755321)

[Name: tr\_ar\_diu\_logUserRole 19](#_Toc44755322)

[Name: tr\_br\_i\_checkOfferAmmount 19](#_Toc44755323)

# Vorwort:

Um das System testen zu können, muss zuerst die Datenstruktur auf der VM Oracle Instanz die wir in dem DBS Kurs zur Verfügung gestellt bekommen haben, eingespielt werden. Die Scripts müssen in einer möglichen Reihenfolge ausgeführt werden, damit keine Constraint fehler erzeugt werden. Ein möglich weg wäre folgende Reihenfolge:

1. sql\_database\_user.sql
2. sql\_ddl\_creates.sql
3. sql\_dml\_inserts.sql
4. sql\_functions.sql
5. sql\_procedures.sql
6. sql\_triggers.sql
7. sql\_views.sql

Es wurde die Vorlage der Coding Conventions aus dem DBS moodel Kurs verwendet um die Variablen, Blöcke logisch zu benennen und auch die Code-Kommentare wurden in diesem Format geschrieben.

Es wurde ein User erstellt, der über die PHP GUI nur über die Prozeduren und Funktionen auf die Datenbank zugreift.

User: edr\_user5 Password: Oracle

Sollte es Probleme mit dem einrichten geben, dann wenden Sie sich bitte an [wi18b016@technikum-wien.at](mailto:wi18b016@technikum-wien.at)

# GUI:

Die GUI wurde mit PHP realisiert und benötigt folgende Resourcen um gestartet zu werden :

* Apache WEB Server
* Oracle instaclient

Wenn Sie PHP installiert haben, können Sie den code lokal laufen lassen und die Funktionalität überprüfen

# Tabellen

### Name: User

Inhalt: In dieser Tabelle werden daten der User gespeichert

Indizes: -

Spalten:

Name: userid

Datentype: number

Constraint: -

Inhalt: Primary Key

Name: rolleid

Datentype: number

Constraint: Verweis auf die die Tabelle Users

Inhalt: ID der Rolle die dem User zugewiesen ist.

Name: vorname

Datentype: VARCHAR2(30)

Constraint: -

Inhalt: Vorname des Users

Name: nachname

Datentype: VARCHAR2(50)

Constraint: -

Inhalt: Nachname des Users

Name: dob

Datentype: Date

Constraint: -

Inhalt: Geburtstag des Users

Name: adresseid

Datentype: number

Constraint: Verweis auf die Tabelle Adresse

Inhalt: ID der Adresse die dem User zugewiesen ist.

### Name: Nachrichten

Inhalt: In dieser Tabelle werden alle Nachrichten, die mit dem System übermittelt werden gespeichert.

Spalten:

Name: nachrichtenid

Datentype: number

Constraint: -

Inhalt: Primary Key

Name: userid-sender

Datentype: number

Constraint: Verweis auf die Tabelle User

Inhalt: Userid des Senders

Name: userid\_empf

Datentype: number

Constraint: Verweis auf die Tabelle User

Inhalt: Userid des Empfängers

Name: inhalt

Datentype: VARCHAR2(500)

Constraint: -

Inhalt: Textuelle Inhalt einer Nachricht

Name: message\_time

Datentype: Timestamp

Constraint: -

Inhalt: Zeitpunkt an dem die Nachricht versendet wurde

### Name: Rolle

Inhalt: In dieser Tabelle werden sind die Rollen der User gespeichert

Spalten:

Name: rolleid

Datentype: number

Constraint: -

Inhalt: Primary Key

Name: titel

Datentype: VARCHAR2(30)

Constraint: -

Inhalt: Der Titel einer rolle ist der Name dieser Rolle

Name: beschreibung

Datentype: VARCHAR2(100)

Constraint: -

Inhalt: Genauer Beschreibung dieser Rolle

### Name: Adresse

Inhalt: In dieser Tabelle werden alle Adressen des Systems gespeichert

Spalten:

Name: adresseid

Datentype: number

Constraint: -

Inhalt: Primary Key

Name: plz

Datentype: number

Constraint: Verweis auf den auf die Tabelle Ort

Inhalt: Die Postleitzahl der Adresse

Name: strasse

Datentype: VARCHAR2(100)

Constraint: -

Inhalt: Die Bezeichnung der Straße einer Adresse

Name: hausnummer

Datentype: VARCHAR2(50)

Constraint: -

Inhalt: Die genaue Hausnummer einer Adresse

### Name: Ort

Inhalt: Über die Postleitzahl werden in dieser Tabelle die Orte gespeichert

Spalten:

Name: plz

Datentype: number

Constraint: -

Inhalt: Plz und Primarykey

Name: name

Datentype: VARCHAR2(50)

Constraint: -

Inhalt: Der Name des Ortes

### Name: Angebot

Inhalt: Alle Angebote die auf dem System erstellt wurden.

Spalten:

Name: angebotid

Datentype: number

Constraint: -

Inhalt: Primary Key

Name: userid\_verkaeufer

Datentype: number

Constraint: Verweis auf die Tabelle User

Inhalt: Id des Verkäufers

Name: kategorieid

Datentype: number

Constraint: Verweis auf die Tabelle Kategorie

Inhalt: Id der entsprechenden Kategorie

Name: anzeigetext

Datentype: VARCHAR2(500)

Constraint: -

Inhalt: Beschreibung des Angebotes

### Name: Transaktion

Inhalt: In dieser Tabelle werden alle Transaktionen gespeichert. Damit sind alle akzeptierten Angebote gemeint.

Spalten:

Name: transaktionid

Datentype: number

Constraint: -

Inhalt: Primary Key

Name: angebotid

Datentype: number

Constraint: Verweis auf die Tabelle Angebot

Inhalt: Id des Angebotes das diese Transaktion betrifft.

Name: userid\_kaeufer

Datentype: number

Constraint: Verweis auf die Tabelle User

Inhalt: Id des Users der das Angebot akzeptiert hat.

Name: bewertung\_verkaeufer\_text

Datentype: VARCHAR2(200)

Constraint: -

Inhalt: Textuelle Bewertung der Transaktion des Verkäufers

Name: bewertung\_verkaeufer\_value

Datentype: number

Constraint: -

Inhalt: Zahlen Wert der Bewertung dieser Transaktion des Verkäufers.

Name: bewertung\_kaeufer\_text

Datentype: VARCHAR2(200)

Constraint: -

Inhalt: Textuelle Bewertung der Transaktion des Käufers

Name: bewertung\_kaeufer\_value

Datentype: number

Constraint: -

Inhalt: Zahlen Wert der Bewertung dieser Transaktion des Käufers

### Name: Angebotskategorie

Inhalt: In dieser Tabelle sind alle Kategorien der Angebote definiert

Spalten:

Name: kategorieid

Datentype: number

Constraint: -

Inhalt: Primary Key

Name: beschreibung

Datentype: VARCHAR2(100)

Constraint: -

Inhalt: Formulierung der Kategorie

### Name: Produkt

Inhalt: In dieser Tabelle werden Produkte mit ihren Merkmalen gespeichert.

Spalten:

Name: produktid

Datentype: number

Constraint: -

Inhalt: Primary key

Name: saisonid

Datentype: number

Constraint: Verweis auf die Tabelle Saison

Inhalt: Id der Saison dieses Produktes

Name: bioid

Datentype: number

Constraint: Verweis auf die Tabelle Bio\_Faktor

Inhalt: ID des BioFaktors dieses Produktes

Name: name

Datentype: VARCHAR2(50)

Constraint: -

Inhalt: Bezeichnung des Produktes

### Name: Produkt\_Angebot

Inhalt: Hilfstabelle um die n:m Beziehung zwischen Angebot und Produkt aufzulösen.

Spalten:

Name: produktid

Datentype: number

Constraint: Verweis auf die Tabelle Produkt

Inhalt: ID des Produktes.

Name: angebotid

Datentype: number

Constraint: Verweis auf die Tabelle Angebot

Inhalt: ID des Entsprechenden Angebots

Name: menge

Datentype: number

Constraint: -

Inhalt: Die Menge des Produktes in einem Angebot.

### Name: Bio\_Faktor

Inhalt: In dieser Tabelle sind die Bio Faktoren gespeichert, die in dem System zur Verfügung stehen

Spalten:

Name: bioid

Datentype: number

Constraint: -

Inhalt: Primary Key

Name: kategorie\_bezeichnung

Datentype: VARCHAR2(30)

Constraint: -

Inhalt: Bezeichnung des BioFaktors.

Name: beschreibung

Datentype: VARCHAR2(300)

Constraint: -

Inhalt: Beschreibung des entsprechenden Bio-Faktors

### Name: Saison

Inhalt: In dieser Tabelle sind alle Saisonen definiert, die in dem System verwendet werden können.

Spalten:

Name: saisonid

Datentype: number

Constraint: -

Inhalt: Primary Key

Name: name

Datentype: VARCHAR2(30)

Constraint: -

Inhalt: Bezeichnung der Saison

Name: beschreibung

Datentype: VARCHAR2(300)

Constraint: -

Inhalt: Genaue Beschreibung der entsprechenden Saison

### Name: User\_Rolle\_Log

Art: Tabelle

Zweck: Tabelle, um Änderungen in der Rolle eines Users zu loggen.

Spalten:

Name: logid

Datentype: number

Constraint: -

Inhalt: Primary Key

Name: userid

Datentype: number

Constraint: Verweis auf den geänderten User aus der Tabelle User

Inhalt: ID des Users

Name: rolleid\_alt

Datentype: number

Constraint: Verweis auf die alte Rolle

Inhalt: ID der Rolle vor der Änderung

Name: rolleid\_neu

Datentype: number

Constraint: Verweis auf die neue Rolle

Inhalt: ID der neuen Rolle

Name: time

Datentype: Timestamp

Constraint: -

Inhalt: Zeitpunkt der Änderung

Name: aktion

Datentype: VARCHAR(10)

Constraint: -

Inhalt: Textuelle Beschreibung der Änderung

# VIEWs

Name: User\_Details

Inhalt: In dieser View werden die Userdetails aller User angezeigt. Dafür wird die Tabelle User gejoind auf Addresse, auf Ort und auf Rolle.

Name:view\_user\_produkte

Inhalt**:** Diese View zeigt alle Produkte, die von den Usern gerade angeboten werden. Die Verbindung wird über die Relation „Produkt\_Angebot“ hergestellt. Angezeigt werden Angebot ID, Produkt ID, Produktname, User ID des Verkäufers, sowie dessen Vor- und Nachname.

Basisrelationen**:** Produkt, Produkt\_Angebot, Angebot, User

Name: Bio\_Faktor\_User\_Durchschnitt

Inhalt: Eine View welche den durchschnittlichen Bio-Faktor jedes Users ausgibt. Es wird der Durchschnitt aller von einem User angebotenen Produkte verwendet. Auf eine Gewichtung nach Menge wird verzichtet.

# PROZEDUREN

### Name: sp\_add\_ort

Art: Procedure

Zweck: Diese Procedure legt einen neuen Ort an, sollte er noch nicht existieren. Dabei werden alle nötigen Informationen über den Ort in den Parametern übergeben. Sollte der Ort bereits existiern oder erfolgreich erstellt worden sein, so wird der Primärschlüssel des Elements in den Output: gespeichert.

Input: l\_n\_plz\_in – PLZ des Ortes

l\_v\_ortsname\_in – Name des Ortes

Output**:** l\_n\_pk\_out – PK des Elements

l\_v\_error\_out – Eventuelle Fehlermeldung.

### Name**:** sp\_add\_adresse

ArtProcedure

Zweck**:** Diese Procedure legt eine neue Adresse an, sollte sie noch nicht existieren. Dabei werden alle nötigen Informationen der Adresse in den Parametern übergeben. Sollte die Adresse bereits existiern oder erfolgreich erstellt worden sein, so wird der Primärschlüssel des Elements in den Output: gespeichert.

Input**:** l\_v\_strasse\_in – Straßenname der Adresse

l\_v\_hausnummer\_in – Hausnummer der Adresse

l\_n\_plzpk\_in – Hausnummer der Adresse

Output**:** l\_n\_pk\_out – PK des Elements

l\_v\_error\_out – Eventuelle Fehlermeldung.

### Name**:** sp\_add\_rolle

Art**:** Procedure

Zweck**:** Diese Procedure legt eine neue Rolle an, sollte sie noch nicht existieren. Dabei werden alle nötigen Informationen über die Role in den Parametern übergeben. Sollte Rolle bereits existiern oder erfolgreich erstellt worden sein, so wird der Primärschlüssel des Elements in den Output: gespeichert.

Input: l\_v\_titel\_in – Titel der Rolle

l\_v\_beschreibung\_in – Beschreibung der Rolle

Output: l\_n\_pk\_out – PK des Elements

l\_v\_error\_out – Eventuelle Fehlermeldung.

### Name**:** sp\_add\_user

Art**:** Procedure

Zweck**:** Diese Procedure legt einen neuen User an, sollte er noch nicht existieren. Dabei werden alle nötigen Informationen über den User in den Parametern übergeben. Sollte der User bereits existiern oder erfolgreich erstellt worden sein, so wird der Primärschlüssel des Elements in den Output: gespeichert.

Input**:** l\_n\_rollepk\_in – PK der Rolle des Users

l\_v\_vorname\_in – Vorname des Users

l\_v\_nachname\_in – Nachname des Users

l\_v\_dob\_in – Geburtsdatum des Users

l\_n\_adressepk\_in – PK der Adresse des Users

Output**:** l\_n\_pk\_out – PK des Elements

l\_v\_error\_out – Eventuelle Fehlermeldung.

### Name**:** sp\_delete\_user

Art**:** Procedure

Zweck**:** Diese Procedure löscht einen User aus der Datenbank. Um die Datenbestände konsistent und nachvollziehbar zu halten, werden die Artefakte des Users nicht gelöscht sondern einem Delete-User zugewiesen. Somit ist ersichtlich, dass die Einträge einem gelöschten User zugehörig sind, ohne die historische Integrität der Daten zu gefährden.

Davon betroffen sind: Nachrichten, Transaktionen und Angebote

Input**:** l\_n\_userid\_in – userid des zu löschenden Nutzers

Output**:** l\_n\_error\_out – Eventuelle Fehlermeldung.

### Name: sp\_addProduct

Art: Procedure

Zweck**:** Es werden alle, für ein Produkt relevanten Daten an die Prozedur übergeben. Diese überprüft intern, ob die „saisonid“ und die „bioid“ in der Datenbank erfasst sind. Wenn, dem nicht so ist, gibt sie eine Fehlermeldung aus und bricht die Prozedur ab. Bei korrekten Daten wird eine neue „produktid“ erstellt und alle Daten in die Tabelle „Produkt“ eingefügt.

Input**:** l\_n\_saisonid\_in – Die Saisonid des Produktes.

l\_n\_bioid\_in – Die Bioid des Produktes.

l\_v\_name\_in – Der Name des Produktes.

Output: l\_v\_error\_out – Eventuelle Fehlermeldung.

### Name: sp\_addOffer

Art: Prozedure

Zweck**:** Es werden alle, für ein Angebot relevanten Daten an die Prozedur übergeben. Diese überprüft intern, ob die „useridVerkaeufer“ und die „kategorie“ in der Datenbank erfasst sind. Wenn, dem nicht so ist, gibt sie eine Fehlermeldung aus und bricht die Prozedur ab. Bei korrekten Daten wird eine neue „angebotid“ erstellt und alle Daten in die Tabelle „Angebot“ eingefügt.

Input**:** l\_n\_useridVerkaeufer\_in – Die Userid des Verkäufers.

l\_n\_kategorieid\_in – Die Kategorieid des Angebots.

l\_v\_anzeigetext\_in – Der Anzeigetext für das Angebot. Eine kurze Beschreibung oder der Titel.

Output: l\_v\_error\_out – Eventuelle Fehlermeldung.

### Name: sp\_addTransaction

Art: Procedure

Zweck**:** Es werden alle, für eine Transaktion relevanten Daten an die Prozedur übergeben. Diese überprüft intern, ob die „angebotid“ und die „useridKaeufer“ in der Datenbank erfasst sind. Wenn, dem nicht so ist, gibt sie eine Fehlermeldung aus und bricht die Prozedur ab. Bei korrekten Daten wird eine neue „transaktionid“ erstellt und alle Daten in die Tabelle „Angebot“ eingefügt.

Inputl\_n\_angebotid\_in – Die „angebotid“ auf die sich die Transaktion bezieht.

l\_n\_useridKaeufer\_in – Die „userid“ des Käufers.

l\_v\_bewertungVerkaeuferT\_in – Die Bewertung des Verkäufers in Textform.

l\_n\_bewertungVerkaeuferV\_in – Die Bewertung des Verkäufers in Form einer Zahl.

l\_v\_bewertungKaeuferT\_in – Die Bewertung des Käufers in Textform.

l\_n\_bewertungKaeuferV\_in – Die Bewertung des Käufers in Form einer Zahl.

Output**:** l\_v\_error\_out – Eventuelle Fehlermeldung.

### Name: sp\_sendMessage

Zweck**:** Es werden alle, für eine Nachricht relevanten Daten an die Prozedur übergeben. Diese überprüft intern, ob die „useridSender“ und die „useridEmpf“ in der Datenbank erfasst sind. Wenn, dem nicht so ist, gibt sie eine Fehlermeldung aus und bricht die Prozedur ab. Bei korrekten Daten wird eine neue „nachrichtenid“ erstellt und alle Daten in die Tabelle „Angebot“ eingefügt. Als Zeit wird der Zeitpunkt verwendet, wenn die Prozedur ausgeführt wird. Es ist explizit erwünscht, dass man Nachrichten an sich selbst senden kann. So kann ein User beispielsweise kleine Notizen in der Anwendung speichern.

Input**:** l\_n\_useridSender\_in – Die Userid des Senders.

l\_n\_useridEmpf\_in – Die Userid des Empfängers.

l\_v\_Inhalt\_in – Der Text der Nachricht.

Outputl\_v\_error\_out – Eventuelle Fehlermeldung.

### Name: sp\_updateUserRolle

Art: Prozedure

Zweck: Ändert die Rolle eines Users.

Input: l\_n\_userId – Die id des gewünschten Users.

l\_n\_rolleId – Die neue Rolle des Users.

Output: l\_v\_error\_out – Eventuelle Fehlermeldung.

### Name: sp\_addProductOffer

Art: Prozedure

Zweck: Fügt ein Produkt einem Angebot hinzu.

Input: l\_n\_angebotId\_in – Die id des gewünschten Angebots.

l\_n\_produktId\_in – Die id des gewünschten Produktes.

l\_v\_menge\_in – Die Menge an Produkten.

Output: l\_v\_error\_out – Eventuelle Fehlermeldung.

# FUNKTIONEN

### Name: f\_transaktion\_userx\_cur

Art: Funktion

Zweck: Es werden alle Transaktionen eines Users ausgewertet und zurück geliefert.

Input: i\_n\_userid – userid

Output: SYS\_REFCURSOR -

transaktionsid,

userid\_verkaeufer

anzeigetext

l\_v\_error\_out – Eventuelle Fehlermeldung

### Name: f\_Nachrichten\_UserX\_cur

Art: Function

Zweck: Es werden alle Nachrichten die ein User versendet hat ausgewertet und zurückgeliefert.

Input: i\_n\_userid – userid

Output: SYS\_REFCURSOR-

Vorname des Empfängers,

Nachname des Empfängers,

Inhalt und den Timestamp.

l\_v\_error\_out – Eventuelle Fehlermeldung

### Name: f\_angebote\_saison\_cur

Art: Funktion

Zweck: Es werden alle Angebote einer bestimmten Kategorie ausgewertet und zurück geliefert

Input: i\_in\_sid – saisonid

Output: SYS\_REFCURSOR-

angebotid

Anzeigetext

Produkt.name

l\_v\_error\_out – Eventuelle Fehlermeldung

### Name: f\_filter\_ang\_prodname\_cur

Art**:** Function

ZweckDiese Funktion zeigt alle Angebote, welche ein Produkt inserieren, das den mitgegebenen Suchstring enthält. Sie bezieht sich auf die Basisrelationen „Angebot“, „Produkt\_Angebot“ und „Produkt“.

Input**:** L\_v\_suchstring\_in - Der Suchstring

Output**:** SYS\_REFCURSOR – Cursor aller zutreffenden Einträge

l\_v\_error\_out – Eventuelle Fehlermeldung

# TRIGGER

### Name: tr\_br\_i\_welcomeMessage

Art: Trigger

Type: Before Row Trigger

Type Extension: Insert

Zweck: Schriebt einem neu angelegten User eine Willkommensnachricht .

### Name: tr\_br\_d\_deleteUserCompletely

Art: Trigger

Type: Before Row Trigger

Type Extension: Delete

Zweck: Löscht alles in Relation zu einem User, wenn dieser gelöscht wird.

### Name: tr\_ar\_diu\_logUserRole

Art: Trigger

Type: After Row Trigger

Type Extension: Insert, Update oder Delete

Zweck: Loggt Änderungen der User Rolle.

### Name: tr\_br\_i\_checkOfferAmmount

Art: Trigger

Type: Before Row Trigger

Type Extension: Insert

Zweck: Überprüft ob die Mindestmenge von 5 Stück im Angebot erfüllt ist.