Schnittstelle für das Frontend

Team102 | Raum 102

Kevin Kekule, Alex Burcev, Fadi Koch

2016

Inhalt

[Hinzufügen 2](#_Toc457207363)

[Methodenaufruf: 2](#_Toc457207364)

[Typ: 2](#_Toc457207365)

[datenArray: 2](#_Toc457207366)

[Rückgabewert: 2](#_Toc457207367)

[Ändern 3](#_Toc457207368)

[Methodenaufruf 3](#_Toc457207369)

[TYP: 3](#_Toc457207370)

[PrimSchluesselArray: 3](#_Toc457207371)

[AssocDaten: 3](#_Toc457207372)

[Rückgabewert: 3](#_Toc457207373)

[Entfernen 4](#_Toc457207374)

[Methodenaufruf 4](#_Toc457207375)

[TYP 4](#_Toc457207376)

[assocPrimSchluesselArray 4](#_Toc457207377)

# Hinzufügen

Hinzufügen wird für alle Datenbankaufrufe gleich aussehen.

## Methodenaufruf:

db\_insert (const TYP, array datenArray[]);

## Typ:

Der Typ ist als konstante definiert. Es gibt folgende Konstante:

* cLieferant
* cRaum
* cBenutzer
* cKompArt
* cKompArtAttr

## datenArray:

Das Datenarray ist assoziativ, die Eingaben sind Variabel, die **Namen und Datentypen** sind aus dem UML Diagramm der Anlage zu entnehmen.

**Achtung**: Es gibt eine Änderung in einer Tabelle, die von Felix Mühlenstedt vorgeschlagen wurde:  
Tabelle: **komponentenarten**  
Spaltennamen alt:

* ka\_id
* ka\_komponentenart

Spaltennamen **neu**:

* kar\_id
* kar\_bezeichnung

Ein assoziatives Array kann folgendermaßen aussehen:

array datenArray = array();  
datenArray[„r\_id“] = 1;  
datenArray[„r\_nr“] = „130a“;  
datenArray[„r\_bezeichnung“] = „Dritter Stock super geil“;  
datenArray[„r\_notiz“] = „Sehr schoener Raum“;

## Rückgabewert:

Die Methode liefert einen int als Rückgabewert, diese sind wie folgt definiert:

* 0 🡪 Kein Fehler
* 1 🡪 verwendete Konstante als erster Paramater ist nicht zulässig(es gibt diese Konstante nicht)
* 2 🡪 Der zweite Parameter, das Array ist null. Es darf NICHT null sein.
* 3 🡪 Die Daten aus dem Array sind NICHT ausreichend für den Insert. Für weitere Informationen bei Felix Mühlenstedt melden.
* 4 🡪 Die verwendeten Datentypen aus dem Array stimmen nicht mit dem Datentyp der Datenbank überein. Für weitere Informationen bei Felix Mühlenstedt melden.
* 5 🡪 Authorisierung fehlgeschlagen. Der Nutzer ist nicht berechtigt dass zu tun.
* 6 🡪 Unbekannter Fehler. Der Fehler wird weitergeleitet.

# Ändern

Ändern wird für alle Datenbankaufrufe gleich aussehen.

## Methodenaufruf

db\_update(const TYP, array primSchluesselArray, array assocDaten);

## TYP:

Siehe Typ bei Hinzufügen -- [hier](#KonstantenTyp)

## PrimSchluesselArray:

Dieses Array ist assoziativ und braucht alle Primärschlüssel der Tabelle vom angegebenen Typ. Die Primärschlüssel sind dem UML Diagramm der Anlage zu entnehmen.

## AssocDaten:

Hier sind alle zu ändernden Daten in einem assoziativen Array zu übergeben. Die Tabellennamen sind aus dem UML Diagramm der Anlage zu entnehmen

## Rückgabewert:

Die Methode gibt einen integer zurück, der je nach Erfolg/Fehler unterschiedlich ist. Die Werte sind wie folgt:

* 0 🡪 Erfolgreich
* 1 🡪 TYP konstante existiert nicht
* 2 🡪 Primschluesselarray darf nicht null sein
* 3 🡪 AssocDaten darf nicht null sein
* 4 🡪 Primschluesselarray enthält nicht alle nötigen Primärschlüssel um den Datensatz eindeutig zu identifizieren
* 5 🡪 User ist nicht authorisiert für diesen Vorgang
* 6 🡪 Unbekannter Fehler

# Entfernen

Entfernen wird für alle Tabellen gleich aussehen.

## Methodenaufruf

db\_delete(const TYP, array assocPrimSchluesselArray);

TYP:

Für die Beschreibung siehe Hinzufügen – [hier](#KonstantenTyp)

assocPrimSchluesselArray:

Hier wird ein Primschluesselarray übergeben. Siehe: Ändern -- [hier](#PrimSchluesselArray)

Rückgabewert:

Die Methode gibt einen integer zurück, der je nach Erfolg/Fehler unterschiedlich ist. Die Werte sind wie folgt:

* 0 🡪 Erfolg
* 1 🡪 TYP existiert nicht
* 2 🡪 assocPrimSchluesselArray darf nicht null sein
* 3 🡪 Datensatz nicht vorhanden
* 4 🡪 Deleteanomalie
* 5 🡪 User ist nicht authorisiert für diesen Vorgang
* 6 🡪 Unbekannter Fehler