Datenbankkonzept



Projektarbeit M153

Etienne Roulet

Elia Reutlinger

Modul 153 @GIBM Pratteln

März 2019

[**Management Summary**](#_xsxowbt8vs80) **2**

[**Selbstständigkeitserklärung**](#_ug3vwta43uzg) **2**

[**Namenskonzept**](#_6x5jyfh6jrle) **3**

[**Views**](#_htdpumyh16h6) **3**

[Nach Vorgaben](#_kff9hofgogp) 3

[Abteilungsleiter](#_lxvhd1aesvk1) 3

[IT-Probleme/Tickets](#_f67rxc4at8sj) 4

[Arbeitsplatz](#_lpz7onid4ngi) 5

[Artikel](#_patgs4kviyqi) 6

[Vorträge](#_o2eqjt8qzh7q) 8

[Reservationen](#_iie7n0xbbrj0) 9

[Selbst definierte](#_8ixiikzd8ett) 10

[Fahrzeugstatus](#_r8921c87w3o1) 10

[Beanstandungen](#_x4w8jqooqk7j) 11

[Abteilungsmitarbeiter](#_xbte3xv14fj0) 12

# Management Summary

Dieses Dokument soll bestimmte Punkte zur erstellten Datenbank im Modul 153 genauer erläutern. Dazu wird das Namenskonzept aufgeführt sowie die unterschiedlichen Views, welche laut Auftrag zu erstellen waren. Diese bestehen aus 6 Stück nach Vorlage und 3 weiteren, welche individuell bestimmt werden konnten. Zu jeder View werden Details wie Eigenschaften und Datentypen, sowie 2 Beispiele zur Ausführung festgehalten.

# Selbstständigkeitserklärung

Wir bestätigen hiermit, dass wir die vorstehende Modularbeit (Konzept & ERM) selbstständig angefertigt, keine anderen als die angegebenen Hilfsmittel benutzt und sowohl wörtliche, als auch sinngemäss verwendete Textteile, Grafiken oder Bilder kenntlich gemacht haben. Diese Arbeit ist in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegt worden.

Ort, Datum

Etienne Roulet Elia Reutlinger

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Namenskonzept

Um einen reibungslosen Import und eine gute Übersicht zu garantieren, orientieren sich alle Elemente der Datenbank an einem bestimmten Namenskonzept:

1. **Tabellenname**, **Eigenschaften**, **Beziehungen u.ä.** sind auf Deutsch, und **Umlaute** werden als ue, oe, au, etc. geschrieben.
2. **Tabellennamen** sowie der **Datenbankname** sind kleingeschrieben.
3. **Eigenschaften** haben einen grossen Anfangsbuchstaben.
4. **Eigenschaften**, welche als Fremdschlüssel auf die ID einer anderen Tabelle verweisen, haben einen zweckgebundenen Namen (also nicht unbedingt den Namen der Tabelle) und den Suffix ‘\_ID’ (Ausser ID’s bei Sub-Entitätsmengen).
5. Jede **Tabelle** (ausser bei n:m Beziehungen) hat einen Integer Primärschlüssel ‘ID’, auch wenn sich eine Eigenschaft in der Tabelle als Primärschlüssel eignen würde (Da ihre Werte einzigartig sind).
6. **Eigenschaften**, welche einzigartig sein sollten, tragen einen ‘UNIQUE’-Index, um mehrfacheingaben zu verhindern.
7. **Views** tragen zur Unterscheidung von Tabellen einen Präfix ‘view\_’.
8. **Beziehungen** sind in Gross- und Kleinbuchstaben beschriftet und lassen ihren Zweck in der Beschriftung nachvollziehen.
9. **Beschriftungen** folgen wenn nötig dem ‘Snake\_Case’ Format.
10. **Datentypen** wie VARCHAR haben einen sinnvollen Wertebereich. Was hier sinnvoll ist soll im Internet recherchiert werden (z.B. nach Best-Practice).
11. **Geldbeträge** werden mit dem Datentyp FLOAT gespeichert.
12. Es wird wenn möglich und sinnvoll immer generalisiert/spezialisiert.
13. Wenn bei textuellen Werten die maximale Länge nicht definitiv klar ist, wird MEDIUMTEXT verwendet.
14. ENUM-Eigenschaften nutzen keine Abkürzungen in ihren Optionen.

# 

# Views

## Nach Vorgaben

### Abteilungsleiter

**View-Name**

view\_abteilungsleiter

**Zweck/Voraussetzungen**

Zeigen Sie von allen Abteilungsleitern den Namen, den Benutzernamen, die vollständige Adresse, den Namen der Abteilung und den Lohn an.

**Enthaltene Attribute**

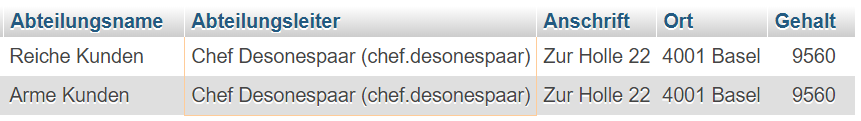
|  |  |  |
| --- | --- | --- |
| **Name (Urspung)** | **Datentyp** | **Beschreibung** |
| Abteilungsname (abteilung) | varchar | Der Name der Abteilung |
| Abteilungsleiter (mitarbeiter) | varchar (mehrere) | Vorname, Nachname und (in Klammern) Benutzername des Abteilungsleiters. |
| Anschrift (kontaktdaten) | varchar (mehrere) | Strasse und Hausnummer des Abteilungsleiters |
| Ort (ortschaft) | int + varchar | Postleitzahl und Ortschaft zur Anschrift |
| Gehalt (gehalt) | float | Gehalt des Abteilungsleiters |

**Beispiele**

Suche nach Details zu einem Benutzernamen:

SELECT \* FROM `view\_abteilungsleiter`

WHERE Abteilungsleiter LIKE '%chef.desonespaar%'



Suche nach Abteilungsleitern mit einem Gehalt grösser 10’000:

SELECT Abteilungsleiter, Gehalt FROM `view\_abteilungsleiter` WHERE Gehalt > 10000



### IT-Probleme/Tickets

**View-Name**

view\_itprobleme

**Zweck/Voraussetzungen**

Zeigen Sie alle IT-Geräte an, für die von Mitarbeitern Fehlermeldungen erfasst wurden, die noch offen sind. Angezeigt werden sollen: interne Bezeichnung, Geräteart, Name des Herstellers, Modellname, Name des Mitarbeiters, der die Meldung erstellt hat, die Fehlermeldung und der Raum, in dem sich das Gerät befindet.

**Enthaltene Attribute**

|  |  |  |
| --- | --- | --- |
| **Name (Urspung)** | **Datentyp** | **Beschreibung** |
| Interne Bezeichnung (komponente\_inventar) | varchar | Die interne Beschreibung des Gerätes. |
| Geraeteart  (komponente\_inventar) | enum (varchar) | Der Geräte-Typ (PC, Drucker, Switch, etc...). |
| Modell (komponente\_katalog) | varchar | Der Modellname des Geräts. |
| Hersteller (komponente\_katalog) | varchar | Der Hersteller des Geräts. |
| Erfasser (mitarbeiter) | varchar (mehrere) | Vor- und Nachname des Erfassers der Meldung. |
| Initiale Nachricht (nachricht) | mediumtext | Die erste Nachricht, welche zur Meldung erfasst wurde (Fehlermeldung). |
| Raum (raum) | int | Die ID des Raums, in welchem sich das Gerät befindet. |

**Beispiele**

Drucker finden, welche gemeldet wurden:

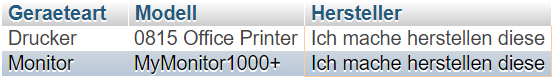
SELECT \* FROM `view\_itprobleme` WHERE Geraeteart = 'Drucker'



Suche gemeldeten Geräten nach Hersteller:

SELECT Geraeteart, Modell, Hersteller FROM `view\_itprobleme`

WHERE Hersteller LIKE '%mache herstellen%'



### Arbeitsplatz

**View-Name**

view\_arbeitsplatz

**Zweck/Voraussetzungen**

Erstellen Sie für einen anzugebenden Arbeitsplatz eine Liste aller IT-Geräte, die ausschliesslich zu diesem Arbeitsplatz gehören. Angezeigt werden sollen: der Arbeitsplatz, der Name des Herstellers, der Modellname, die Geräteart und die interne Bezeichnung.

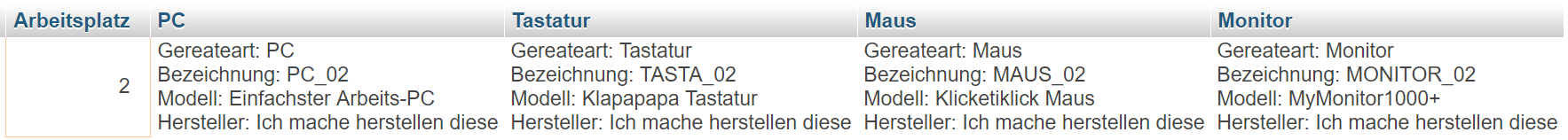
**Enthaltene Attribute**

|  |  |  |
| --- | --- | --- |
| **Name (Urspung)** | **Datentyp** | **Beschreibung** |
| Arbeitsplatz (arbeitsplatz) | int | ID des Arbeitsplatzes. |
| PC (komponente\_inventar)  (komponente\_katalog)  (firma) | varchar (mehrere) + enum | Geräte-Typ, Interne Bezeichnung, Modellname und Hersteller des PCs. |
| Tastatur (komponente\_inventar)  (komponente\_katalog)  (firma) | varchar (mehrere) + enum | Geräte-Typ, Interne Bezeichnung, Modellname und Hersteller der Tastatur. |
| Maus (komponente\_inventar)  (komponente\_katalog)  (firma) | varchar (mehrere) + enum | Geräte-Typ, Interne Bezeichnung, Modellname und Hersteller der Maus. |
| Monitor (komponente\_inventar)  (komponente\_katalog)  (firma) | varchar (mehrere) + enum | Geräte-Typ, Interne Bezeichnung, Modellname und Hersteller des Monitors. |

**Beispiele**

Alle Komponenten an einem Arbeitsplatz finden:

SELECT \* FROM `view\_arbeitsplatz` WHERE Arbeitsplatz = 2



Arbeitsplatz zu Gerät finden (anhand interner Bezeichnung):

SELECT Arbeitsplatz FROM `view\_arbeitsplatz`

WHERE Monitor LIKE '%MONITOR\_02%'



### Artikel

**View-Name**

view\_artikel

**Zweck/Voraussetzungen**

Zeigen Sie alle Artikel. Angezeigt werden sollen der Artikelname, der Name der Kategorie, der Name des jeweiligen Lieferanten, die Anzahl der Zubehörartikel.

**Enthaltene Attribute**

|  |  |  |
| --- | --- | --- |
| **Name (Urspung)** | **Datentyp** | **Beschreibung** |
| Artikelname (artikel) | varchar | Name des Artikels |
| Kategorie (kategorie) | varchar | Kategorie des Artikels |
| Lieferant (firma) | varchar | Lieferant des Artikels |
| Zubehoer (artikel\_hat\_zubehoer) | int | Anzahl Zubehör-Teile zum Artikel |

**Beispiele**

Artikel zu einer bestimmten Kategorie finden:

SELECT \* FROM `view\_artikel`

WHERE Kategorie LIKE '%Entspannung%'



Artikel mit Zubehör finden:

SELECT \* FROM `view\_artikel` WHERE Zubehoer > 0



## 

### Vorträge

**View-Name**

view\_vortraege

**Zweck/Voraussetzungen**

Erstellen Sie eine Liste aller gehaltenen Vorträge. Angezeigt werden sollen das Datum, die Uhrzeit, das Thema, der Name des Haupt-Referenten und die Anzahl angemeldeter Personen. Falls es einen Vorgänger und/oder Nachfolger des Vortrags gibt, sind jeweils deren Themen anzugeben.

**Enthaltene Attribute**

|  |  |  |
| --- | --- | --- |
| **Name (Urspung)** | **Datentyp** | **Beschreibung** |
| Thema (vortrag) | varchar | Thema des Vortrags. |
| Datum\_Uhrzeit (vortrag) | datetime | Datum und Zeitpunkt des Vortrags. |
| Hauptreferent (mitarbeiter) | varchar (mehrere) | Vor- und Nachname des Hauptreferenten |
| Teilnehmer (vortrag\_hat\_teilnehmer) | int (Summe) | Gesamtanzahl Teilnehmer |
| Nachfolger (vortrag) | varchar | Thema des Nachfolgers, falls vorhanden. |
| Vorgaenger (vortrag) | varchar | Thema des Vorgängers, falls vorhanden. |

**Beispiele**

Vorträge eines bestimmten Hauptreferenten finden:

SELECT \* FROM `view\_vortraege`

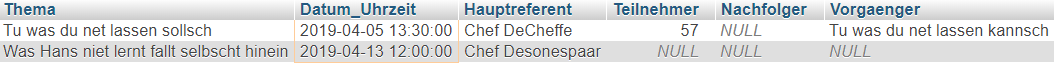
WHERE Hauptreferent LIKE '%DeCheffe%'



Vorträge in einem bestimmten Zeitraum finden:

SELECT \* FROM `view\_vortraege`

WHERE Datum\_Uhrzeit between '2019/04/01' and '2019/04/25'



## 

### Reservationen

**View-Name**

view\_reservationen

**Zweck/Voraussetzungen**

Zeigen Sie alle Reservationen von FCB-Jahreskarten an, die für die Kundenpflege verwendet wurden. Angezeigt werden muss das Datum, der Name des Mitarbeiters, der Name des Kunden, der Rang, der Sektor, die Reihe und die Sitzplatznummer.

**Enthaltene Attribute**

|  |  |  |
| --- | --- | --- |
| **Name (Urspung)** | **Datentyp** | **Beschreibung** |
| Ressource (ressource) | enum (varchar) | Typ der Ressource |
| Mitarbeiter (mitarbeiter) | varchar (mehrere) | Vor- und Nachname des Mitarbeiters |
| Kunde (firma) | varchar | Name des Kundens |
| Rang (jahreskarte) | int | Rang der Karte |
| Sektor (jahreskarte) | varchar | Sektor der Karte |
| Reihe (jahreskarte) | int | Reihe der Karte |
| Sitzplatznummer (jahreskarte) | int | Sitzplatznummer der Karte |

**Beispiele**

Reservationen eines bestimmten Mitarbeiters finden:

SELECT \* FROM `view\_reservationen`

WHERE Mitarbeiter LIKE '%DeCheffe%'



Reservationen in einem bestimmten Zeitraum finden:

SELECT \* FROM `view\_reservationen`

WHERE Datum between '2019/03/01' and '2019/03/25'



## Selbst definierte

### Fahrzeugstatus

**View-Name**

view\_fahrzeugstatus

**Zweck/Voraussetzungen**

Eine Liste, welche alle Firmenfahrzeuge mit Modellname, Kennzeichen und dem Namen der zugehörigen Versicherung auflistet. Damit die Verfügbarkeit des Fahrzeuges geprüft werden kann, soll die Anzahl an ungelösten Problemen beim Fahrzeug angezeigt werden (Ein Fahrzeug mit Problemen sollte nicht genutzt werden). Zusätzlich soll ersichtlich sein, wie viele Reparaturen das Fahrzeug schon hatte. Schlussendlich kann man sehen, welches Fahrzeug gerade genutz werden könnte und welches am meisten Probleme macht.

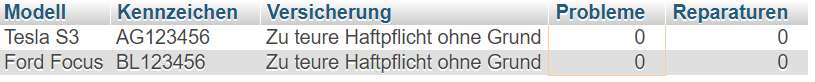
**Enthaltene Attribute**

|  |  |  |
| --- | --- | --- |
| **Name (Urspung)** | **Datentyp** | **Beschreibung** |
| Modell (dienstfahrzeug) | varchar | Modell des Fahrzeugs |
| Kennzeichen (dienstfahrzeug) | varchar | Kennzeichen des Fahrzeugs |
| Versicherung (firma) | varchar | Name der Versicherung |
| Probleme (problem) | int | Anzahl nicht geschlossener/behandelter Probleme |
| Reparaturen (reparatur) | int | Anzahl durchgeführter Reparaturen |

**Beispiele**

Fahrzeug ohne Probleme finden:

SELECT \* FROM `view\_fahrzeugstatus` WHERE Probleme = 0



Fahrzeug mit den meisten Reparaturen finden:

SELECT \* FROM `view\_fahrzeugstatus`

ORDER BY Reparaturen DESC LIMIT 1



### Beanstandungen

**View-Name**

view\_beanstandungen

**Zweck/Voraussetzungen**

Hierbei sollen für jede vorhandene Beanstandung der betroffene Artikel, die Nachricht des Kunden in der Beanstandung und der Hersteller des Artikels sowie dessen Telefonnummer aufgelistet werden. Dadurch kann der Support schnell den Hersteller kontaktieren.

**Enthaltene Attribute**

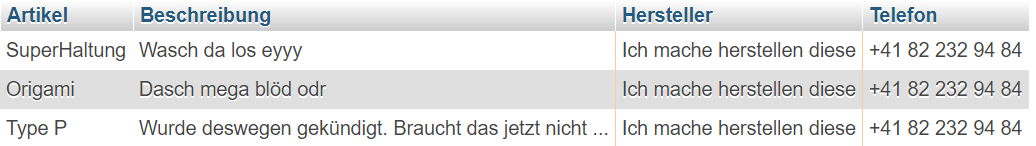
|  |  |  |
| --- | --- | --- |
| **Name (Urspung)** | **Datentyp** | **Beschreibung** |
| Artikel (artikel) | varchar | Name des Artikels |
| Beschreibung (beanstandung) | mediumtext | Beschreibung der Beanstandung |
| Hersteller (firma) | varchar | Name des Herstellers zum Artikel |
| Telefon (kontaktdaten) | varchar | Telefonnummer des Herstellers |

**Beispiele**

Alle Beanstandungen zu Artikel von bestimmtem Hersteller finden:

SELECT \* FROM `view\_beanstandungen`

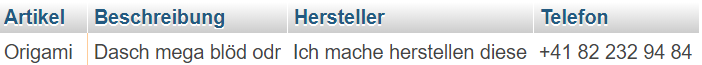
WHERE Hersteller LIKE '%herstellen diese%'



Beanstandungen zu einem bestimmten Artikel finden:

SELECT \* FROM `view\_beanstandungen`

WHERE Artikel LIKE '%Origami%'



## 

### Abteilungsmitarbeiter

**View-Name**

view\_abteilungsmitarbeiter

**Zweck/Voraussetzungen**

Diese Liste soll alle Abteilungen mit ihrem Namen anzeigen. Dazu soll ersichtlich sein, wer der Abteilungsleiter ist (Name) und wie viele Angestellte die Abteilung insgesamt hat. Wenn eine Abteilung eine Übergeordnete Abteilung hat, soll der Name von dieser ebenfalls erscheinen.

**Enthaltene Attribute**

|  |  |  |
| --- | --- | --- |
| **Name (Urspung)** | **Datentyp** | **Beschreibung** |
| Abteilung (abteilung) | varchar | Name der Abteilung |
| Abteilungsleiter (mitarbeiter) | varchar (mehrere) | Vor- und Nachname des Abteilungsleiters |
| Mitarbeiter (mitarbeiter) | int | Anzahl Mitarbeiter in der Abteilung |
| Uebergeordnete\_Abteilung (abteilung) | varchar | Name der Übergeordneten Abteilung, falls vorhanden. |

**Beispiele**

Abteilung nach Abteilungsleiter finden:

SELECT \* FROM `view\_abteilungsmitarbeiter`

WHERE Abteilungsleiter LIKE '%DeCheffe%'



Die Summe Mitarbeitenden in allen untergeordneten Abteilungen einer bestimmten übergeordneten Abteilung herausfinden:

SELECT Uebergeordnete\_Abteilung, SUM(Mitarbeiter) AS Total\_Mitarbeiter

FROM `view\_abteilungsmitarbeiter`

WHERE Uebergeordnete\_Abteilung LIKE '%Abzocken%'

