# Datenbank Raumbelegung

Aufgabe: Planen und erstellen der Datenbank Raumbelegung.

## Anforderungen:

1. Größe der Räume und Anzahl der Arbeitsplätze
   * SELECT \* FROM raeume\_ap;
2. Welcher Computer befindet sich an welchem Arbeitsplatz *(das klingt, als ob jeder einzelne Arbeitsplatz eindeutig identifizierbar sein soll)*
   * SELECT \* FROM computer\_lesbar;  
     bzw.
   * SELECT Raum, AP\_ID, PC\_Marke, PC\_Modell, Netzwerk\_Name, IP\_Addr FROM belegung\_lesbar WHERE NOT(Netzwerk\_Name IS NULL);
3. Informationen zum Computer (Modell, Hersteller, Arbeitsspeicher, IP-Adresse)
   1. SELECT \* FROM computer\_lesbar;
4. Welcher Mitarbeiter belegt welchen Arbeitsplatz (Datensatz)
   1. SELECT \* FROM belegung\_lesbar;
5. Informationen zu den Mitarbeitern (Vorname, Nachname, Geburtsdatum, Adresse, Geschlecht)
   1. SELECT \* FROM mitarbeiter;

*Notiz von Janusz (danke fürs Abtippen!):* Kreuztabelle (Verweis Computer, Raum, Mitarbeiter)

## Gedanken zum Entwurf

### Entitäten / Objekte / Tabellen

* Mitarbeiter
* Räume
* Computer
* Arbeitsplätze
  + Diese könnten z.B. einen Namen bekommen; bei mir haben sie erstmal nur automatisch erzeugte IDs – immerhin sind sie eindeutig.

Wegen Normalisierung (2. und 3. Normalform sollten abgeleitete Tabellen hinzukommen – das habe ich mir weitgehend gespart…)

* Computerhersteller
* Adressen (Mitarbeiter können die selbe Adresse haben...)
  + wiederum transitiv abhängig: PLZ → Ort

### Leitsätze zu den Beziehungen zwischen den Entitäten / Tabellen

*(mir persönlich hilft das, neben dem grafisch-anschaulichen ERM)*

Jeder Mitarbeiter kann 0 oder 1 Arbeitsplatz haben.

Das spricht für eine 1:1-Beziehung (Mitarbeiter ↔ Arbeitsplatz).

Jeder Computer kann an 0 oder 1 Arbeitsplatz sein.

Das spricht für eine 1:1-Beziehung (Computer ↔ Arbeitsplatz).

Jeder Arbeitsplatz kann beliebig viele Mitarbeiter oder Computer haben (auch 0)! *(Ohne diese Annahme wird es natürlich leichter… Und wahrscheinlich sogar realistischer, aber naja...)*

Das spricht gegen eine 1:1-Beziehung!

Konsequenz: Es gibt eine weitere abgeleitete Tabelle: **Belegungen**

Jeder Mitarbeiter kann nur an maximal einer Belegung teilnehmen.

Mitarbeiter ↔ Belegung 1:0/1

Jeder Computer kann nur für maximal eine Belegung verwendet werden.

Computer ↔ Belegung 1:0/1

Jeder Arbeitsplatz kann 0, eine oder mehrere Belegungen aufweisen / jede Belegung findet an genau einem Arbeitsplatz statt (NOT NULL…).

Arbeitsplatz ↔ Belegung 1:n

Jeder Raum kann 0..N Arbeitsplätze haben / jeder Arbeitsplatz ist genau in einem Raum (NOT NULL).

Raum ↔ Arbeitsplatz 1:n

Jeder Computer hat genau einen Hersteller (NOT NULL) / jeder Hersteller kann 0..N Computer bauen.

Computerhersteller ↔ Computer 1:n

Jeder Mitarbeiter hat genau eine Adresse (NOT NULL? - hier könnte man auch zulassen, dass die Adresse unbekannt ist, dann NULL erlauben!) / an jeder Adresse können 0..N Mitarbeiter wohnen.

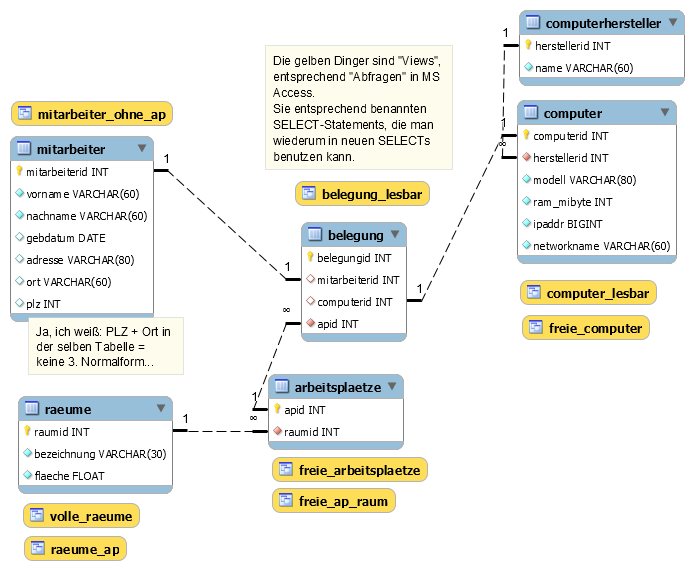
Adresse ↔ Mitarbeiter 1:n

Jeder Ort kann mehrere Postleitzahlen haben / jede Postleitzahl kennzeichnet genau einen Ort (NOT NULL).

Ort ↔ Postleitzahl 1:n

## Entity-Relationship-Modell

(erstellt mit MySQL-Workbench)



## „Test-Driven Development“

Die Entwurfs-Annahmen überprüfen:

* Jeder Mitarbeiter kann nur an einem (oder keinem) Arbeitsplatz „sitzen“.
  + Mitarbeiter können auch ohne Computer arbeiten.
  + *Mglw. überdenken (Ist das überhaupt sinnvoll? Soll das den Arbeitsplatz blockieren?):* Computer können auch ohne Mitarbeiter an einem Arbeitsplatz sein.
  + *Diskutabel:* Es können mehrere Mitarbeiter und/oder Computer an demselben Arbeitsplatz „sitzen“.
    - ABER: In Normallfall sollen Mitarbeiter nur an freie Arbeitsplätze geschickt werden! (→ meint die normalen „INSERTs“)
    - In volle Räume lässt sich kein Mitarbeiter hinzufügen (entspricht: Nur freie Arbeitsplätze dürfen neu besetzt werden).
* Wenn ein Computer gelöscht wird, wird auch eine entsprechende Belegung mit gelöscht.
* Wenn ein Mitarbeiter gelöscht wird, wird auch eine entsprechende Belegung gelöscht.
* Wenn ein Raum gelöscht wird, wird auch eine entsprechende Belegung gelöscht.