Unser kleines DB Cheatsheet

* Daten / (EZ) Datum: Informationen, die etwas beschreiben. Daten einer Datenbank beschreiben also **Attribute** von Entitäten.
* Datenbank / Datenbanksystem: CRUD : Sammlung von Daten, aber die CRUD Operationen müssen vom DB System gewährleistet sein.
* CRUD: Daten können erschaffen werden (Create), Daten müssen gelesen werden können (Read), Daten müssen verändert werden können (Update), Daten müssen gelöscht werden können (Delete)
* Entitätstyp: Eine Klasse gleichartiger Entitäten. Kategorie.
* Entität: Das konkrete Exemplar / Ausprägung/ eines Entitätstyps. Eintrag in die Kategorie.
* Attribute: Eigenschaften von Entitätstypen.
* ID – Attribut: Ist der Identifizierer einer Entität.
* Relationen (Beziehung): Entitätstypen können in Beziehung zu anderen Entitätstypen stehen. (z.B. eine Auto-Entität gehört zur Person-Entität)
* ERM: Ein Entity-Relationship-*Modell* ist eine **gedankliche Zusammenfassung aller Elemente (hauptsächlich Entitätstypen und Relationen)**, die für den Aufbau einer geplanten Datenbank von Bedeutung sein werden.
* ERD: Entity-Relationship-Diagramm: Ein Entity-Relationship-Diagramm ist die graphische Darstellung dieser Elemente. ER-Diagramme können als (grober) „Bauplan“ einer leeren Datenbank betrachtet werden. (Es werden noch keine Daten berücksichtigt)
* *ERD-Symbole*:

Entitätstyp: Rechteck

Attribut: Ellipse

Relation: Raute

* Chen Notation: ERD Syntax mit den zugehörigen Symbolen.
* Kardinalitäten: Angabe über die Anzahl der an einer Beziehung beteiligten Entitäten
* ID-Attribut: Unterstreichung des Attributbezeichners
* Zusammengesetztes Attribut: Ellipse (Im ERD darf es noch zusammengesetzte Attribute geben. Im logischen Datenbankentwurf wird gefordert diese in atomare Attribute aufzuteilen.)

# Relationstypen

1. 1:1 Relation (Person -> Personalausweis
2. 1:n Relation (Person -> Auto / Person (1) fotografiert Foto(n) / Stadt(n) liegt in Land(1) )
3. m:n Relation (Person -> Lied / Person(m) -> Verein(n))

# Kardinalität

* Die zwei Werte aus denen Relationstypen bestehen werden als Kardinalitäten bezeichnet:

1 bedeutet höchstens 1, n oder m bedeutet möglicherweise mehrere (oder beliebig viele).

**Stolperfallen Kardinalitäten:** Exemplar vs Typ (Person – Buch) / Aktuell versus historisch (Person – Land(Präsidentschaft) / Bloß nicht zu voreilig (Gemälde – Person) / Quintessenz: Wir müssen um eine Datenbank zu modellieren einen regen Austausch mit dem Auftraggeber führen.

# Schwache Entität

* Eine „Schwache Entität“ ist eine Entität, deren Existenz von einer anderen (übergeordneten) Entität desselben Modells abhängig ist.
* Entsprechend werden Entitäten, für die eine solche Abhängigkeit nicht gilt, als „Starke Entitäten“ bezeichnet.
* Eine übergeordnete Entität muss nicht notwendigerweise eine Starke Entität sein, da ihre Existenz selbst von einer weiteren Entität abhängig sein könnte.

# Attribute zu Relationen

* Zusätzliche Informationen über Relationen, die wir vom Auftraggeber erhalten, können als Attribute zur Relation eingetragen werden.
* Relationen mit Attributen werden als eigenständige Entitäten eingetragen.

# Reflexive Relation

* Falls Relationen zwischen Entitäten desselben Entitätstyps betrachtet werden, so werden diese als **Reflexive** (= „selbstbezügliche“) **Relationen** bezeichnet.
* Beispiel: Eine Person kann mit einer anderen Person verheiratet sein.

# Mehrstellige Relationen

* Bisher haben wir ausschließlich Relationen zwischen (höchstens) 2 Entitätstypen betrachtet. Diese werden auch als **„Binäre Relationen“** bezeichnet.
* Relationen zwischen mehr als zwei Entitätstypen werden **„mehrstellig“** genannt.
* Im Rahmen des Konzeptionellen Datenbankentwurfs müssen wir in Fällen, in denen eine mehrstellige Relation nicht durch mehrere binäre Relationen aufgelöst werden kann, eine geeignete **weitere Entität** einführen, um darauf aufbauend die Kardinalitäten bestimmen zu können

# Datenbanktypen

* Objektorientierte Datenbank, Objektrelationale Datenbank, Multimedia Datenbank, Hierarchische Datenbank

# Relation (Tabelle) und relationales DB Schema

* Entitätstypen des ERD werden in Relationen (Tabellen) übersetzt.
* Der Name eines Entitätstyps ist der Name der T
* Verbindungen zwischen Tabellen werden nun als Beziehungen bezeichnet.
* Jede Tabellenzeile stellt den Datensatz einer Entität dar. Und die Spaltenbezeichner sind die jeweiligen Attribute.
* Da wir uns in einem Schema nur für die Struktur einer (leeren) Datenbank interessieren

tragen wir noch keine konkreten Daten ein und notieren die Attribute **untereinander**, um Platz zu sparen.

# Primärschlüssel

* Jeder Entitätstyp hat ein ID-Attribut bekommen. Dieses Attribut wird als Primärschlüssel einer Tabelle in einem RDB-Schema bezeichnet.
* Zwei zwingende Eigenschaften eines Primärschlüssels:

1. Der Wert des Primärschlüssels muss definiert sein und darf nicht null sein.
2. Der Wert des Primärschlüssel muss eindeutig sein. Es darf nicht zwei Entitäten innerhalb einer Tabelle geben, die einen identischen Primärschlüsselwert haben.

* Die IHK erwartet, dass Primärschlüssel im RDB-Schema durch den Zusatz **„(PK)“** gekennzeichnet werden.

# Fremdschlüssel

* Über den Primärschlüssel können Entitäten in anderen Tabellen verwendet werden. Der Primärschlüssel wird von der fremden Tabelle als Fremdschlüssel bezeichnet. Wir können so eine Beziehung zwischen beiden Tabellen aufbauen.
* Die IHK erwartet, dass Fremdschlüssel im RDB-Schema durch den Zusatz **„(FK)“** gekennzeichnet werden.
* Bei einer 1:1 Beziehung kann die Tabelle in der der Fremdschlüssel eingetragen wird, frei gewählt werden.
* **Bei einer 1:n-Beziehung gibt es keine Alternative:**

Der **Fremdschlüssel** wird stets in der Tabelle mit Kardinalität **n** eingetragen.

Dieser Fremdschlüssel referenziert auf den PK der Tabelle mit Kardinalität **1**.

* Bei einer m:n Beziehung muss daher eine Hilfstabelle eingeführt werden. Bezeichner so einer Hilfstabelle zwischen Person und Lied könnte „Person\_Lied“ heißen. Diese Hilfstabelle hält dann die Fremdschlüssel der beiden verknüpften Entitätstypen.

# Normalisierung

Drei Normalformen um die Qualitätsstandards für Datenbanken zu gewährleisten:

* 1NF: Eine Relationale Datenbank erfüllt die **1. Normalform**, wenn kein Attribut einer Tabelle existiert, dessen Informationsgehalt in eigenständige Teil-Informationen zerlegt werden kann. Man spricht in diesem Fall davon, dass alle Attribute **„atomar“** (also „unteilbar“) sind. Dies wird auch als Atomisierung bezeichnet.
* 2NF: Eine Relationale Datenbank erfüllt die 2. Normalform, wenn sie zum einen die 1. NF erfüllt und darüber hinaus keine Tabelle besitzt, in der ein Nicht-Schlüssel-Attribut eine funktionale Abhängigkeit zu einer echten Teilmenge des Primärschlüssels aufweist.

Verstöße gegen die 2. Normalform sind Hinweise darauf, dass eine Tabelle nicht **„monothematisch“** ist. Wir können die 2. NF einfach erreichen, indem wir stets einen Surrogatschlüssel (das künstliche einspaltige ID-Attribut) verwenden. Für jede Datenbank in der 1. Normalform, in der alle Tabellen ausschließlich **1-spaltige Primärschlüssel** besitzen, gilt also: Diese Datenbanken sind dann stets – und also quasi **„automatisch“** – zugleich auch in der **2. Normalform**!

* 3NF: Eine Relationale Datenbank erfüllt die 3. Normalform, wenn sie zum einen die 2. NF erfüllt und darüber hinaus keine Tabelle besitzt, in der ein Nicht-Schlüssel-Attribut **in transitiver Abhängigkeit** zu einer echten (oder unechten) Teilmenge des Primärschlüssels steht. Eine Beziehung heißt **„transitiv“,** wenn aus der Relation von **„A nach B“** und **„B nach C“** stets folgt, dass dann auch zwingend eine Beziehung von **„A nach C“** gilt. Die Beziehung **„funktional abhängig von“** ist transitiv! Eine solche Transitivität demonstriert die fehlende Monothematik der Tabelle. Lösung wird die **Auslagerung** in eine eigenständige **Tabelle** sein!

Würden wir auf eine Auslagerung in eine eigenständige Tabelle verzichten, so müssten wir viele **Informations-Dubletten** akzeptieren.

# Surrogatschlüssel

* Die fast schon amüsante Lösung um die 2. NF zu erreichen besteht nun schlicht darin, echte Teilmengen eines Primärschlüssels auszuschließen, indem stets ein (einspaltiges) ID-Attribut eingeführt wird. (Dieses „künstliche“ Attribut wird auch als **„Surrogatschlüssel“** bezeichnet). Auf diese Weise entkommen wir der doch recht komplizierten Formulierung der 2. Normalform, indem wir allen Tabellen quasi „die Chance“ nehmen, Abhängigkeiten zu dann nicht mehr existierenden echten Teilmengen des Primärschlüssels zu besitzen.