# Developer Resources

**Tim Kral**

**18.03.2024**

# Inhaltsverzeichnis

**Bemerkung:**

An Lehrpersonen und andere offizielle Stellen:

Die unten aufgeführten Abschnitte „ICT Kompetenzraster“ und „ICT Module“ sollen Übersicht über die Inhalte bieten.

Mit freundlichen Grüssen
Tim Kral

## Abschnitte

KAPITEL 1

# Beiträge

## 1.1 Betriebssysteme

### 1.1.1 Einstieg Betriebssysteme

#### Grundlegendes

#### Was ist ein Betriebssystem?

Der Begriff des Betriebssystems beschreibt eine Platform, welche direkt auf der Hardware läuft und den Umgang mit dieser vereinfacht.

Ein Betriebssystem tut dies, indem sie sich mit dem Download und der Verwaltung von Hardware Treibern rumschlägt, Dateisysteme verwaltet, übergeordnete Software verwaltet und ausführt und dem Nutzer eine Nutzeroberfläche zur Verfügung stellt (grafisch oder textbasiert, etc.).

#### Gängige Betriebssysteme

Die wohl bekanntesten Betriebssysteme sind Linux, MacOS und Windows.

### 1.1.2 Linux Basics

Linux ist ein Betriebssystem, dass aus dem ursprünglichen Unix hervorging. Es ist Open Source (GitHub) und wird bis heute aktiv von der Community weiterentickelt.

#### Grundlegende Beiträge

#### Linux Installieren

#### Einführung

Ein wichtiger Unterschied zwischen Linux und seinen bekanntesten Konkurrenten ist, dass Linux in vielen Formen daher kommt.

Die sogenannten Linux Distributionen erweitern den Linux Kernel mithilfe von Paketen und bereiten diese für bestimmte Anwendungszwecke vor. Siehe Linux Distribution Timeline für eine Übersicht über eine Handvoll der Distributionen.

#### Auswahl Linux Distribution

Die Auswahl der Distribution ist ein sehr individueller Prozess, da diese sehr genau anhand der persönlichen Präferenzen getroffen werden kann, und, da es, wie zuvor erwähnt, für praktisch jeden Zweck eine ordendliche Menge an passenden Systemen gibt.

## 1.2 Netzwerke

### 1.2.1 Netzwerkgeräte

#### Beiträge

#### Access Point

#### Bridge

#### Router

#### Switch

Eine passende Analogie für einen Switch ist wohl ein Kreisel aus dem Strassenverkehr, bei welchem jede Zufahrt auch eine Ausfahrt ist, und jede Ausfahrt eine feste Anschrift hat.

Die Zu-/Ausfahrten sind in diesem Beispiel vergleichbar mit den Anschlüssen an einem Switch, während die Anschriften mit MAC-Adressen vergleichbar sind.

Man schickt also ein Paket, welches eine angegebene Zieladresse (MAC) besitzt, an einen Switch, und es verlässt diesen durch den richtigen Port.

#### L2 Switches

Switches in ihrer Grundausstattung sind Teil der *Sicherungsschicht*. Diese nennen sich L2 Switches (Layer 2 Switches) und dienen dem einfachen Umleiten von Paketen. Sie tun dies mittels der MAC-Adressen in den Paketheadern und sogenannten *Switching Tabellen*.

Diese Tabelle wird, wenn eine Anfrage durch den Switch geleitet wird, populiert, indem der Switch eine Broadcast-Anfrage mit der gesuchten MAC Adresse tätigt. Die Antwort kommt schliesslich aus der Richtung, in welche die Pakete verschickt werden müssen. Letztlich wird diese Route in der Switching Tabelle eingetragen.

#### L3 Switches

Ein L3-Switch baut auf dem Konzept der L2-Switches auf, erweitert diese jedoch um Funktionen, welche Informationen aus den IP-Headern verwenden.

Jene Funktionen können VLAN Kompatibilität, NAT Funktionalität, oder die Zusammenarbeit beziehungsweise Verschmelzung mit einem Router sein.

### 1.2.2 Begriffe rund um Netzwerke

#### Adressierung

**MAC Adresse**
: MAC Adressen sind Teil der OSI Layer 2, sind 48 Bit gross und sind bei jedem Gerät fix eingerichtet.

**IP Adresse**
: IP Adressen sind Identifikationsnummern, welche ab der OSI Layer 3 von den dazugehörigen Netzwerkkomponenten verwendet wird.

**IPv4 Adresse**
: Der IPv4 Standart schreibt eine Grösse von 32 Bit vor.

**IPv6 Adresse**
: Der IPv4 Standart schreibt eine Grösse von 128 Bit vor.

#### Netzwerkkomponenten

**Switching Tabelle**
: Eine Tabelle, welche die Ports und alle direkt oder indirekt darüber erreichbaren Geräte anhand deren MAC-Adresse speichert.

### 1.2.3 OSI Modell

#### OSI Modell

#### Netzzugang

#### Bitübertragungsschicht

Die erste Schicht basiert auf physischen oder elektrischen Prinzipien, welche die Übertragung von Binärdatenströmen auf Hardwareebene erlaubt.

So sind also Netzwerkkabel und Repeater ein wesentlicher Teil dieser Schicht.

### Sicherungsschicht

Die Sicherungsschicht definiert erstmalig die grobe Struktur der Datenströme.

Dies ist hier der sogenannte MAC-Frame oder Ethernet-Frame, in welchen dann der tatsächliche Inhalt (bzw. Nutzlast) eingebettet wird.

Für Informationen und Spezifikationen zum MAC-Frame, siehe *Ethernet Frame*.

Die Aufgabe der Sicherungsebene ist es, den erfolgreichen Übertrag eines Streams zu gewährleisten.

Komponenten dieser Schicht sind beispielsweise *Bridges*, *L2 Switches* oder *Access Points*, da diese Verwendung von den Informationen aus dem MAC-Frame machen.

### Internet

### Vermittlungsschicht

Die Vermittlungsschicht ermöglicht die Kommunikation über mehrere Netzwerke und tut dies mittels IP Adressen.

Diese Schicht definiert ebenfalls Informationen die an die Nutzlast angefügt wird (hier: IP-Header).

Für Informationen und Spezifikationen zum IP-Header, siehe *Internet Protocol Header*.

Teil der Vermittlungsebene sind beispielsweise *Router* und *L3 Switches*.

### Transport

### Transportsschicht

Die Transportsschicht behandelt die verschiedenen Protokolle und erlaubt mittels Portnummern mehrere Services unter der selben IP Adresse.

Der Header der Transportsschicht hängt vom verwendeten Protokoll ab. Die wohl bekanntesten Protokolle hier sind TCP und UDP.

Details zum Aufbau sind hier zu finden: *TCP Segmente*, *UDP Segmente*

### Anwendungen

### Sitzungsschicht

*Platzhalter*

### Darstellungsschicht

*Platzhalter*

### Anwendungsschicht

*Platzhalter*

### Ethernet Frame

### Aufbau

### IEEE 802.3 only

1. Preamble (7 octets)
2. Start Frame Delimiter (1 octets)
3. Destination Address (6 octets)
4. Source Address (6 octets)
5. Length (2 octets)
6. Data / Payload (max. 1500 bytes)
7. Frame Checksum (CRC) (4 bytes)

### Details

**Preamble**
: Die Präambel ist ein Überbleibsel aus Zeiten, in welchen die Geräte, die die Pakete empfangen, einen Moment brauchten, um den Lesevorgang zu starten, was zur Folge haben konnte, dass ein Paar Bits verloren gehen, weshalb die Präambel als Lösung einfach als Indikator eines eingehenden Paketes dient.

  Die Präambel enthält also einfach ein alternierendes Muster (0 und 1), beginnend mit Binär 1.

**Start Frame Delimiter (SFD)**
: Der SFD ergänzt das alternierende Muster der Präambel, ersetzt aber die 0 am Schluss mit einer 1 (also 10101011).

**Destination Address**
: Die Zieladresse ist die MAC-Adresse des nächsten Netzwerkknoten (nächster Empfänger).

**Source Address**
: Die Absenderadresse ist die MAC-Adresse des Vorherigen Netzwerkknoten (letzter Absender).

**Length**
: Gibt Auskunft über die Grösse der Nutzlast (in Bytes).

**Data / Payload**
: Nun folgt die eigentliche Nutzlast. Die Grösse ist mittels der Length vorgegeben.

**Frame Checksum (CRC)**
: Letztlich gibt es noch eine Prüfsumme des ganzen Frames.

### Internet Protocol Header

### Aufbau

### IPv4 (gemäss RFC 791)

1. Version (4 bits)
2. Internet Header Length (IHL) (4 bits)
3. Type of Service (ToS) (8 bits)
    1. Precedence (3 bits)
    2. Delay (1 bit)
    3. Throughput (1 bit)
    4. Relibility (1 bit)
    5. Reserved for Future Use (2 bits)
4. Total Length (16 bits)
5. Identification (16 bits)
6. Flags (3 bits)
    1. Reserved (1 bit)
    2. Don't Fragment (DF) (1 bit)
    3. More Fragments (MF) (1 bit)
7. Fragment Offset (13 bits)
8. Time to Live (TTL) (8 bits)
9. Protocol (8 bits)
10. Header Checksum (16 bits)
11. Sender IP-Address (32 bits)
12. Destination IP-Address (32 bits)
13. Options (if $IHL > 5 : (IHL - 5) * 4$bytes )

### Details

**Version**
Die Version gibt die verwendete Spezifikation im übrigen Header an, dies wäre bei IPv4 Binär 4 (0100) oder bei IPv6 Binär 6 (0110).

**Internet Header Length (IHL)**
Die IHL gibt Auskunft über die Grösse des Headers. Die Grösse wird in 4-Byte Schritten angegeben, die Grösse rechnet sich also so:

$s = IHL * 4$bytes

**Type of Service (ToS)**
*Platzhalter*

**Total Length**
Gibt die Länge des Headers in Kombination mit der Nutzlast an.

**Identification**
Der Identifier ist bei fragmentierter Übertragung von Belang, da er angibt, um welches Fragment es sich handelt.

**Flags**
Die nachfolgenden Attribute gibt Auskunft darüber, ob die aktuelle Übertragung fragmentiert wurde, oder ob das aktuelle Paket das letzte fragment ist.

**Don't Fragment (DF)**
Diese Flag gibt an, ob die Daten fragmentiert wurden (0), oder nicht (1).

**More Fragments (MF)**
Diese Flag gibt an, ob dies das letzte Fragment ist (0), oder nicht (1).

**Time to Live (TTL)**
*Platzhalter*

**Protocol**
Das Protokoll beschreibt das Format der nächsten Layer (gemäss RFC 790).

**Header Checksum**
*Platzhalter*

**Source IP-Address**
*Platzhalter*

**Destination IP-Address**
*Platzhalter*

**Options**
*Platzhalter*

### TCP Segmente

#### Aufbau

1. Zielport (16 Bit / 2 Byte)
2. Senderport (16 Bit / 2 Byte)
3. Sequenznummer (32 Bit / 4 Byte)
4. Folgesegmentnummer / Acknowledgment Number (32 Bit / 4 Byte)
5. Datenoffset (4 Bit)
6. Reserviert (4 Bit)
7. Flags (8 Bit / 1 Byte)
    1. Congestion window reduced (CWR)
    2. ECE
    3. URG
    4. ACK
    5. PSH
    6. RST

7. SYN
8. FIN

8. Fenstergrösse (16 Bit / 2 Byte)
9. Prüfsummme (16 Bit / 2 Byte)
10. Offset in der Sequenznummer
11. Optionen (grösse nach Datenoffset: $offs \cdot 32$ )

#### Details

#### UDP Segmente

#### Aufbau

#### Details

## 1.3 Datenschutz und Datensicherheit

### 1.3.1 Schutzwürdige Daten

Viele Informationen über eine Person sind in den Augen des Gesetzes besonders schützenswert und müssen gegen jedwede Form von unbefugtem Zugriff und / oder Manupulation geschützt werden.

Informationen zu möglichen Schutzmassnahmen sind *hier* zu finden.

#### Personenbezogene Daten

Informationen, wie Namen, Geburtsdaten, Adressen, Telefonnummern oder Sozialsversicherungsnummern gelten als schützenswert, da diese sich auf identifizierbare natürliche Personen beziehen.

#### Gesundheitsdaten

Gesunheitsdaten sind ebenfalls heikel, da diese massgeblichen Einfluss auf die Wahrnehmung einer Person haben können, und somit sind Informationen über den Gesundheitszustand einer Person, medizinische Diagnosen, Behandlungen, Medikamentenhistorie und andere gesundheitsbezogene Details, ebenfalls als besonders schützenswert zu erachten.

#### Finanzdaten

Finanzdaten sind genauso schützenswert, da diese sowohl Einfluss auf die öffentliche Wahrnehmung der Person haben können, als auch grosses Missbrauchpotenzial haben, da diese beispielsweise den Zugriff auf die Vermögenswerte einer Person haben könnten.

So gelten also Kreditkartendaten, Bankkontoinformationen, Einkommensdetails und andere finanzielle Informationen, die für Betrügereien und Identitätsdiebstahl anfällig sein können, als besonders schützenswert.

#### Biometrische Daten

Fingerabdrücke, Retina-Scans, Gesichtserkennungsdaten und andere biometrische Merkmale, die zur Identifikation von Personen verwendet werden, sind auch besonders schützenswert.

#### Unternehmensgeheimnisse

Da Unternehmensgeheimnisse wettbewerbentscheidend sein können, fallen diese genauso unter die Kategorie „schützenswert“. Hier einige Beispiele:

Strategische Geschäftspläne, Forschungs- und Entwicklungsdaten, vertrauliche Verträge und andere proprietäre Informationen von Unternehmen.

#### Juristische Informationen

Im Weiteren sind, Rechtsanwaltskommunikation, Gerichtsakten und andere rechtliche Dokumente, die sensible Informationen über Rechtsstreitigkeiten enthalten, ebenfalls besonders schützenswert.

#### Ethnische oder ethnologische Daten

Ethnische oder ethnologische Daten, Informationen über die ethnische Herkunft einer Person, werden ebenfalls als besonders schützenswert angesehen.

#### Kinderdaten

Letztlich, Informationen über Minderjährige, die aufgrund ihrer besonderen Schutzbedürftigkeit als besonders sensibel gelten, sind auch besonder schützenswert.

### 1.3.2 Schutzmassnahmen

#### Verlustsschutz

Ist man im Besitz wichtiger Informationen und möchte sicherstellen, dass diese nicht gelöscht werden, oder auf eine andere Art und Weise korrumpiert werden, dann gibt es eine Hand voll Möglichkeiten, dies zu tun.

#### Backups

Backups sind Zweitsicherungen, also Kopien eines Datensatzes, welche dem Zweck dienen, im Falle des Verlustes der Originaldaten die Möglichkeit der Wiederherstellung des Datensatzes auf den Zeitpunkt des letzten durchgeführten Backups zu ermöglichen.

Ihrem Zweck zur Folge, sind diese meist, auf einem vom ursprüchlichen Gerät unabhängigen System, gesichert.

#### Einfache Kopie

Die wohl einfachste Form eines Backups ist eine simple eins-zu-eins Kopie der Dateien an einen anderen Ort.

#### Git Versionsmanagement

Das Problem bei einfachen Kopien von Dateien ist, dass sowohl die Dateien, welche zwischen verschiedenen Schnappschüssen verändert wurden, wie auch jene, welche nicht bearbeitet wurden, mehrfach gespeichert werden.

Git bietet hier eine Lösung, da hier bei Schnappschüssen lediglich die Änderungen im Vergleich zum jeweils Vorherigen gespeichert werden.

#### Zugriffsschutz

#### Verschlüsselung

Um die Sicherheit der eigenen Daten selbst bei unbefugtem Lesen sicherzustellen, können diese verschlüsselt werden. Dies bedeutet, dass diese mittels eines Schlüssels unleserlich gemacht werden, sodass diese möglichst nur mit dem selben Schlüssel oder oftmals auch nur mit einem passenden Gegenstück entschlüsselt werden können.

#### Symmetrische Verschlüsselung

Die symmetrische Verschlüsselung verwendet lediglich einen Schlüssel, mit welchem die Nachricht ver- und entschlüsselt werden kann.

Der AES (Advanced Encryption Standard) Algorithmus ist beispielsweise symmetrisch. Für Details über die Implementation, siehe [RFC 3826].

#### Asymmetrische Verschlüsselung

Die asymmetrische Verschlüsselung unterscheidet sich in sofern von der symmetrischen Verschlüsselung, als dass diese zwei Schlüssel verwendet, also ein Schlüssel dient der Verschlüsselung und ein Schlüssel dient der Entschlüsselung. Es gilt hier, dass die Schlüssel jeweils nur für einen der Zwecke geeignet sind, und demnach nicht

Der RSA (Rivest, Shamir, Adleman) Algorithmus ist beispielsweise asymmetrisch. Für Details über die Implementation, siehe [RFC 8017].

### 1.3.3 Begriffe rund um Datenschutz und Datensicherheit

#### Grundlegende Begriffe

**Datenschutz**
: Bezieht sich auf die Gesetzeslage rund um kritische Daten.

**Datensicherheit**
: Bezieht sich auf den technischen Hintergrund, also auf Massnahmen, um Daten zu schützen.

## 1.4 Dokumetationen

### 1.4.1 Dokumentation

#### Einstieg

Die Dokumentation ist ein wichtiger, wenn nicht sogar der wichtigste Teil eines Projekts.

Hier möchte ich dem Leser wichtige Konzepte rund um die Dokumentation näherbringen.

#### Dokumentationstaktiken

Die nachfolgenden Dokumentationstaktiken finden im Zusammenhang mit der Niederschrift von Anforderungen seitens der Auftragsgeber Anwendung.

**Lastenhefte**
: Bei einem Lastenheft handelt es sich um ein Dokument, welches detaillierte Spezifikationen beschreibt, sowohl sachlich, wie auch technisch, und lässt auch kategorisierte Anforderungen zu.

**User Stories**
: User Stories behandeln explizit die Perspektive des Endnutzers auf die Applikation. Diese sind in kurzer Form gegeben und behandeln möglichst grundlegende Funktionen. Sie werden oft in der „Als [Benutzerrolle] möchte ich [eine Aktion], um [einen Nutzen zu erhalten]“ Form beschrieben.

**Use Cases**
: Use Cases sind spezifische Szenarien, in welchen welchen das Produkt sich auf eine bestimmte Art und Weise Verhalten soll. Sie legt Fokus auf die Interaktion zwischen dem Produkt und dem Nutzer und liefert leicht verständliche Beispiele zur Funktionsweise des Produktes.

**Prototypen**
: Kleine Prototypen ermöglichen die frühe Ausmärzung möglicher Missverständnisse und sind ebenfalls ein erster Schritt in der Umsetzung, was einen Fortschritt in der Umsetz selbst bedeutet.

**Datenmodelle**
: Datenmodelle (zum Beispiel: Entity-Relationship-Diagramm) ermöglichen eine visuelle Darstellung der Struktur einer Datenbank oder geben auch Auskunft über den Aufbau eines späteren Produktes.

#### Wichtige Konzepte

#### Personas

Personas sind Spezifikationen, welche die Zielgruppen des Produktes repräsentieren. Sie ermöglichen das Hineinversetzen in einen Kunden und das Nachvollziehen dessen Bedürfdnisse und so kann das Produkt besser an die Anforderungen des Kunden angepasst werden.

Personas können bereits in der Informationsphase ausgemacht werden und mit dem Kunden ausgearbeitet werden, was so die Personas auch für die Planung und das Fällen von funktionsspezifischen Entscheidungen vereinfacht, da weniger Mutmassungen bezüglich der Ziele des Produktes notwendig sind.

Oftmals werden Nutzermodelle mittels folgender Vorlage formuliert: „Persona A, ein erfahrener Benutzer, benötigt Funktion X, um seine Arbeitsabläufe effizienter zu gestalten“.

### Gut formulierte Anforderungen

### Details in Anforderungen

Es ist überaus wichtig, dass die Anforderungen möglichst tiefschürfend festlegen, wie das Produkt strukturiert sein soll und wie es funktionieren soll. Dies bedeutet, dass die Struktur des Produktes und seiner Komponenten anhand der Anforderungen erahnbar oder gar eindeutig sein sollte.

### Eindeutigkeit der Anforderungen

Bei der Überprüfung einer Anforderung auf deren Eindeutigkeit ist diese mit anderen Anforderungen zu überprüfen und sicherzustellen, dass diese nicht im Widerspruch zu den übrigen Anforderungen steht, also beispielsweise eine Anforderungen bestimmte Technologien voraussetzt, während eine andere Anforderungen andere, konfliktbehaftete Technologien voraussetzen.

### Messbarkeit der Anforderungen

Es gilt sicherzustellen, dass alle Anforderungen eindeutig als „erreicht“ oder „nicht erreicht“ eingestuft werden können. Dies ist wichtig, da damit Meilensteine gesetzt werden können, was die Durchführung von beispielsweise Scrum Stints vereinfacht, und so die Auswertung eines Auftrages anhand einer Anforderung eindeutig und klar ist.

### Anforderungen katalogisieren

### Katorisierung anhand Anforderungstypen

Die Anforderungen können mittels Anforderungstypen kategorisiert werden. Gängige Typen sind zum Beispiel: Funktionale Anforderungen, entwicklungsspezifische Anforderungen und repräsentative Anforderungen.

### Kategorisierung anhand Relevanz

Im Weiteren, ins besondere im Zusammenhang mit Meilensteinen, können Anforderungen anhand deren Relevanz im Generellen oder in Relation zu anderen Anforderungen.

### Aufwandsschätzung Auftrag

### Analyse der Anforderungen

Alle Anforderungen sollten unter die Lupe genommen werden und es soll festgestellt werden, ob, und, gegebenenfalls, welche Anforderungen Komplikationsrisiken mit sich bringen, also den Bedarf an einem Zeitpuffer nahelegen. Ausserdem gilt es, jene Aufträge mit hoher Relevanz zu priorisieren.

#### Zeitrahmen für Umsetzung

Die Implementation einer Anforderung bring manche zeitlichen Faktoren mit sich. Diese währen zum Beispiel die Integration in eine bestehende Infrastruktur oder die Einarbeit in eine neue Technologie im Rahmen des Auftrages. Es gilt, diese Faktoren in die Zeitschätzung einzubeziehen.

#### Zeitrahmen für die Dokumentation

Nach Umsetzung oder währendessen sollte dokumentiert werden, was wann geändert wurde und weshalb es geändert wurde. Es sollte besser Zeit im Überfluss, als spärlich, für die Dokumentation beansprucht werden, da die Dokumentation für andere Entwickler am Projekt enorm relevant sein kann, wenn es darum geht, das Projekt kennenzulernen oder Änderungen und deren Hintergrund nachzuvollziehen.

#### Einbezug eines Puffers

Für den Fall von unerwarteten Komplikationen Puffer einzuplanen, damit genug Zeit besteht, diese aufzulösen und die Deadline einzuhalten.

#### Qualität der Anforderungen

#### Regelmässigkeit der Absprachen

Die Ansprache im Bezug auf die Anforderungen und Plänen eines Projektes sollte regelmässig gehalten werden und möglichst viele Illustrationen und Demonstrationen beinhalten und sollte, falls notwendig, überarbeitet werden.

#### Anpassungen nach Notwendigkeit

Anforderungen sollten im Falle von Überarbeitungen umgehend angepasst werden und sollten, im Rahmen der Planung, auch an die gegebenen, technischen Aspekte angepasst werden.

## 1.5 Datenbanken

#### MySQL

### 1.5.1 Einführung MySQL

#### Data Definition Language (DDL)

#### CREATE Statement

#### CREATE DATABASE

```
CREATE DATABASE [IF NOT EXISTS] <name>;
```

### CREATE TABLE

```
CREATE [TEMPORARY] TABLE [IF NOT EXISTS] <name> (
   <column1> <type> [<attr1> <attr2> <...>]
);
```

### CREATE PROCEDURE

```
DELIMITER //

CREATE PROCEDURE [IF NOT EXISTS] <name> (
   [IN | OUT | INOUT] <param1> <type>,
   [IN | OUT | INOUT] <param2> <type>,
   ...
) BEGIN
   -- Do Stuff
END

DELIMITER ;
```

### CREATE VIEW

```

```

### DROP Statement

### DROP Database

```
DROP DATABASE [IF EXISTS] <name>;
```

### DROP Table

```
DROP [TEMPORARY] TABLE [IF EXISTS] <name>;
```

### DROP Procedure

```
DROP PROCEDURE [IF EXISTS] <name>;
```

#### DROP VIEW

```
```

#### ALTER Statement

*Platzhalter*

### Data Control Language (DCL)

#### GRANT Statement

```
```

#### REVOKE Statement

```
```

### Data Manipulation Language (DML)

#### INSERT Statement

```
INSERT INTO <table> (<col1> [, <col2>, ...]) VALUES (<val1> [, <val2>, ...]);
```

#### UPDATE Statement

```
UPDATE <table> SET <col1> = <val1> [, <col2> = <val2>, ...] [WHERE <conditions>];
```

#### DELETE Statement

```
DELETE FROM <table> WHERE <conditions>;
```

#### SELECT INTO Statement

```
```

#### Data Query Language (DQL)

#### SELECT FROM Statement

## 1.5.2 MySQL Constraints

**Primary Key**

Implementation:

```
CREATE TABLE sample (
   <pk-col> <type> <attr> PRIMARY KEY <attr>
);

CREATE TABLE sample (
   <pk-col> <type> <attr>,
   PRIMARY KEY (<pk-col>)
);

CREATE TABLE sample (
   <pk-col> <type> <attr>,
   CONSTRAINT <pk-name> PRIMARY KEY (<pk-col>)
);
```

**Foreign Key**

Implementation:

```
CREATE TABLE sample (
   <fk-col> <type> <attr>,
   FOREIGN KEY (<fk-col>) REFERENCES <ref-table> (<ref-pk-col>)
);

CREATE TABLE sample (
   <fk-col> <type> <attr>,
   CONSTRAINT <fk-name> FOREIGN KEY (<fk-col>) REFERENCES <ref-table> (<ref-pk-col>)
);
```

**Not Null**

Implementation:

```
CREATE TABLE sample (
   <col> <type> <attr> NOT NULL <attr>
);
```

**Unique**

Implementation:

```
CREATE TABLE sample (
   <col> <type> <attr> UNIQUE <attr>
);
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
CREATE TABLE sample (
   <col> <type> <attr>
   UNIQUE (col)
);

CREATE TABLE sample (
   <col> <type> <attr>
   CONSTRAINT <uq-name> UNIQUE (col)
);
```

**Check**

Implementation:

```
CREATE TABLE sample (
   <col> <type> <attr>
   CHECK (<condition>)
);

CREATE TABLE sample (
   <col> <type> <attr>
   CONSTRAINT <chk-name> CHECK (<condition>)
);
```

**Default**

Implementation:

```
CREATE TABLE sample (
   <col> <type> <attr> DEFAULT <def-val>
);

CREATE TABLE sample (
   <col> <type> <attr> DEFAULT <func>
);
```

**Indices**

*Platzhalter*

### 1.5.3 Entity Relation Model

### 1.5.4 Normalisierung

**Erste Normalform (1NF)**

Die erste Normalform ist gegeben, wenn alle Spalten einer Tabelle nicht mehr als eine Information beinhalten.

Ein Beispiel:

| ID | Address |
|---|---|
| 1 | Musterstrasse 1, Musterort |

Das Problem hier besteht darin, dass die Adressen kombiniert statt einzeln gespeichert werden, was die Analysierbarkeit einschränkt und die Optimierung der Abspeicherung mittels Auslagerung erschwert.

Eine (keineswegs die einzige) Lösung dazu:

| ID | Street Name | House Number | City |
|---|---|---|---|
| 1 | Musterstrasse | 1 | Musterort |

Hier erhält nun jede Information ihre einzige Spalte, was die zuvor genannten Probleme auflöst.

**Zweite Normalform (2NF)**
Die zweite Normalform sieht vor, dass die erste gegeben ist. Sie setzt jedoch ebenfalls voraus, dass im Verhältnis stehende Informationen mittels eines eindeutigen Primärschlüssels auszumachen sind.

**Dritte Normalform (3NF)**
Letztlich, die dritte Normalform setzt wiederum die vorherigen Begebenheiten voraus und erweitert diese. Sie setzt also voraus, dass keine Redundanz in den Daten besteht, was zur Optimierung der Datenspeicherung verhelfen kann.

KAPITEL 2

# ICT Kompetenzraster

**Achtung:** In Bearbeitung!!

## 2.1 Begleiten von ICT-Projekten (A)

- *Platzhalter*

## 2.2 Unterstützen und Beraten im ICT-Umfeld (B)

### 2.2.1 Den eigenen ICT-Arbeitsplatz einrichten (B1)

#### Computer mit Betriebssystem aufsetzten (B1.1)

- *Einstieg Betriebssysteme*

#### Einrichtung und Test der Netzwerkverbindung (B1.2)

- *Netzwerkgeräte*
- *OSI Modell*
- *Begriffe rund um Netzwerke*

#### Sicherheitsmassnahmen konfigurieren (Firewall, Antivirus, etc.) (B1.3)

- *Platzhalter*

#### Software und Updates (B1.4)

- *Platzhalter*

#### Peripheriegeräte (B1.5)

- *Platzhalter*

#### Bürotisch und Bürostuhl (B1.6)

- *Platzhalter*

### 2.2.2 Beratung in Bezug auf Datenschutz und Datensicherheit (B3)

#### Antwort auf gezielte Fragen zu System, Netzwerk, Software und Daten (B3.1)

- *Schutzwürdige Daten*
- *Schutzmassnahmen*
- *Begriffe rund um Datenschutz und Datensicherheit*

#### Informieren über Gefahren im Netz und Umgang mit schützenswerten Daten (B3.2)

- *Schutzwürdige Daten*
- *Schutzmassnahmen*

#### Empfehlung Schutzmassnahmen (B3.3)

- *Schutzmassnahmen*

## 2.3 Aufbauen und Pflegen von digitalen Daten (C)

- *Platzhalter*

## 2.4 Entwickeln von Applikationen (G)

- *Platzhalter*

## 2.5 Ausliefern und Betreiben von Applikationen (H)

- *Platzhalter*

# Stichwortverzeichnis