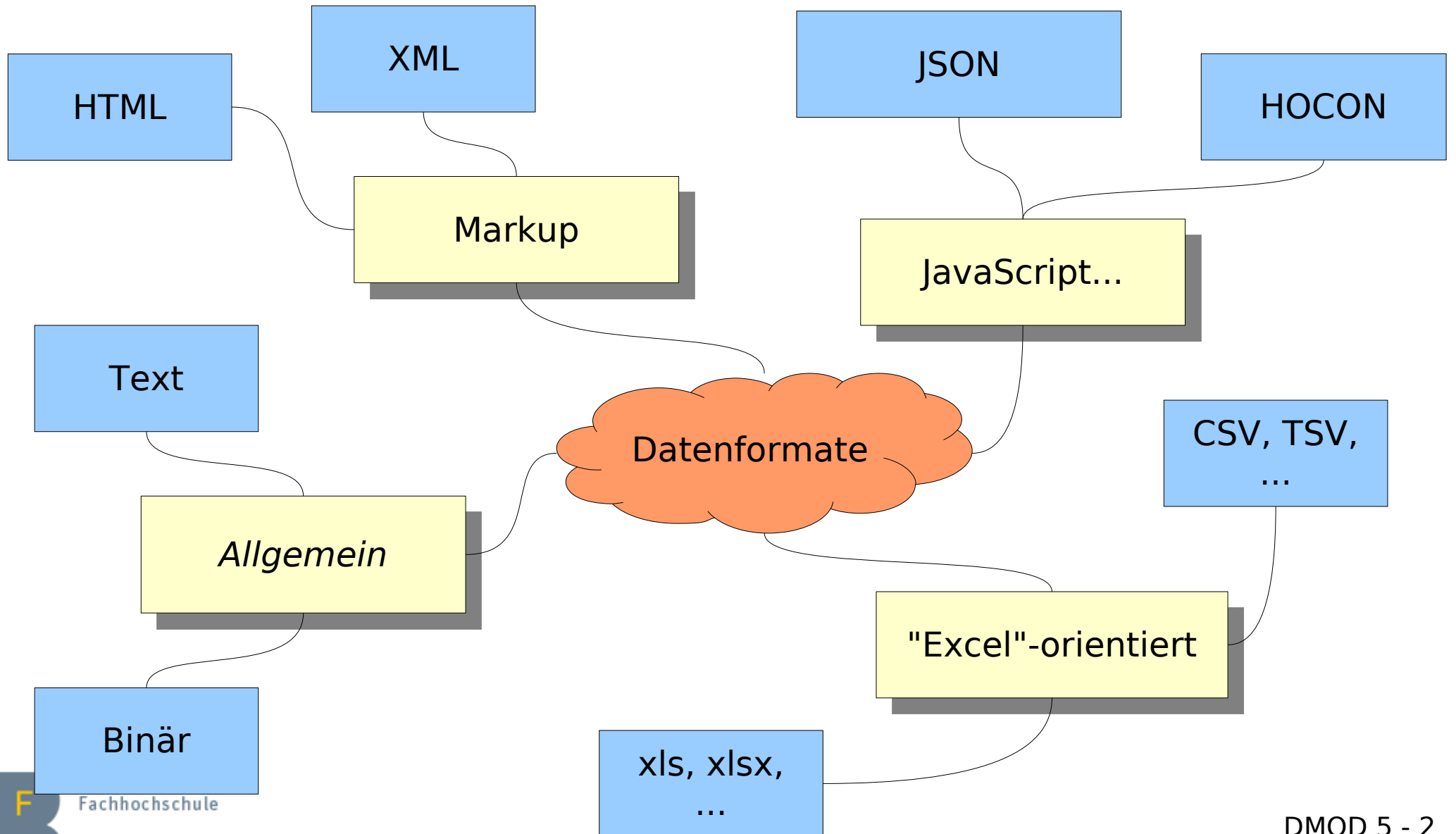


Tag 5

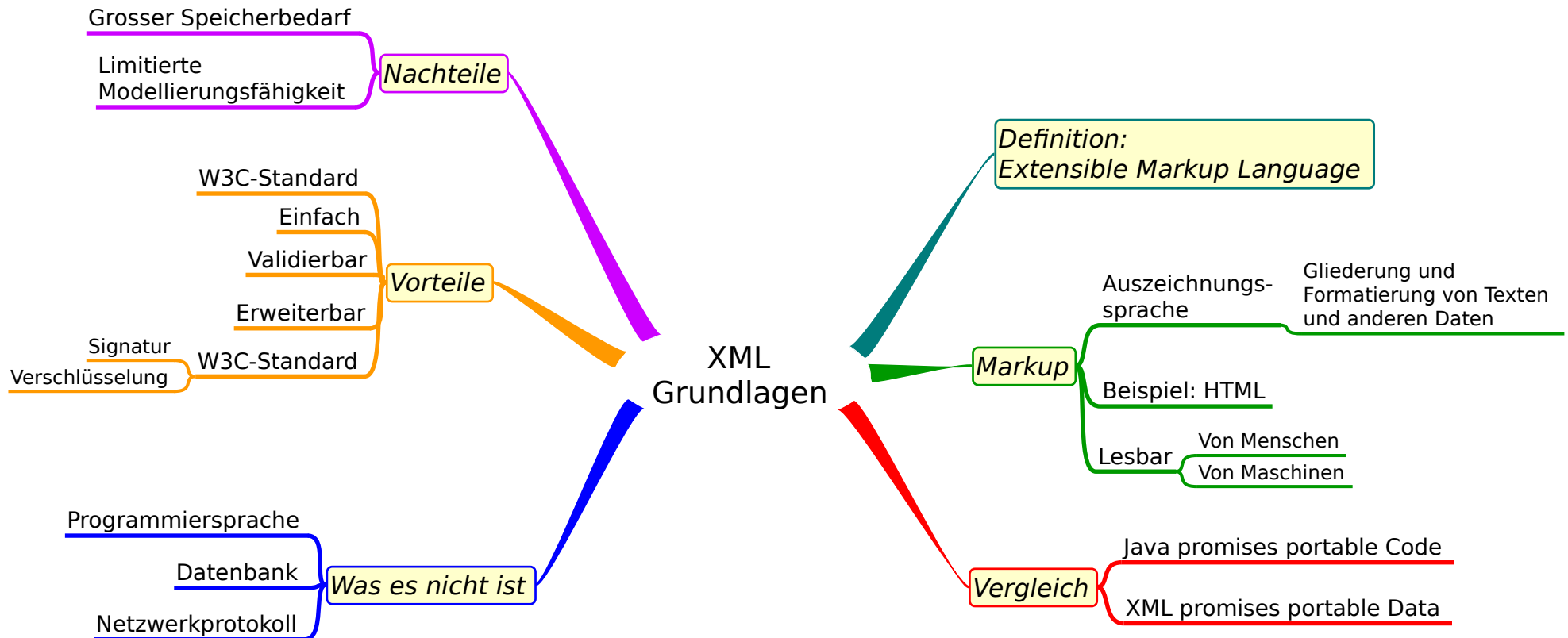
Inhaltsverzeichnis

- Big Picture "Datenformate"
- XML
- JSON
- CSV
- Zeichenkodierung
- Übungen
- BYOQ

Big Picture Datenformate



XML Grundlagen



Quellen:

https://de.wikipedia.org/wiki/Extensible_Markup_Language

https://de.wikipedia.org/wiki/World_Wide_Web_Consortium

Tools:

Windows: Notepad++; Linux: Kate

XML

Inhalt eines Dokuments (1)

```
<Saeugetier>  
  <Raubkatze>  
    <Katze />  
  </Raubkatze>  
</Saeugetier>
```

```
<Katze Farbe="schwarz/weiss">  
Meine Katze heisst Leo  
und hat die Tel-Nummer  
031 123 45 67  
</Katze>
```

- ▶ Eine Menge **Elemente**, delimited by < >
- ▶ Elemente mit </element> beendet oder mit ">" als Abkürzung
- ▶ Aufpassen mit Umläuten: Nur im Text
- ▶ Elemente können **Attribute** enthalten
Syntax: *name = "value"*
- ▶ Elemente können **character data (= Text)** enthalten.
- ▶ XML-Namen sind "case-sensitive"
- ▶ XML ist Leerzeichen und "Ende der Zeile"-Buchstabe **insensitive**

XML

Inhalt eines Dokuments (2)

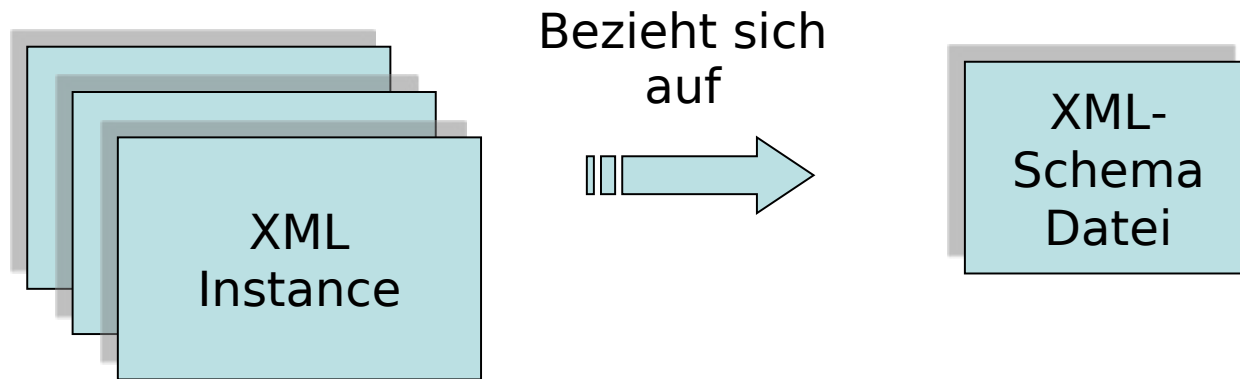
```
<?xml version="1.0"
      encoding="ISO-8859-1"?>
```

```
<meinWurzelElement>
  <einElem
    a=" &quot;1&quot;"/>
  <!-- a contains \"1\" -->
  <![CDATA[ <irgendwas> ]]>
</meinWurzelElement>
```

- ▶ Als Header eine XML-**Deklaration** muss vorhanden sein.
- ▶ Ein XML-Document muss ein sog. **Wurzel-Element** haben
- ▶ **Entity references**: < > & ' " (<, >, &, ', ")
- ▶ **Kommentare**, in Form <!-- ... -->, nicht verschachtelt
- ▶ **CDATA section**, <![CDATA[...]]>, nicht als *XML-Text* interpretiert

XML

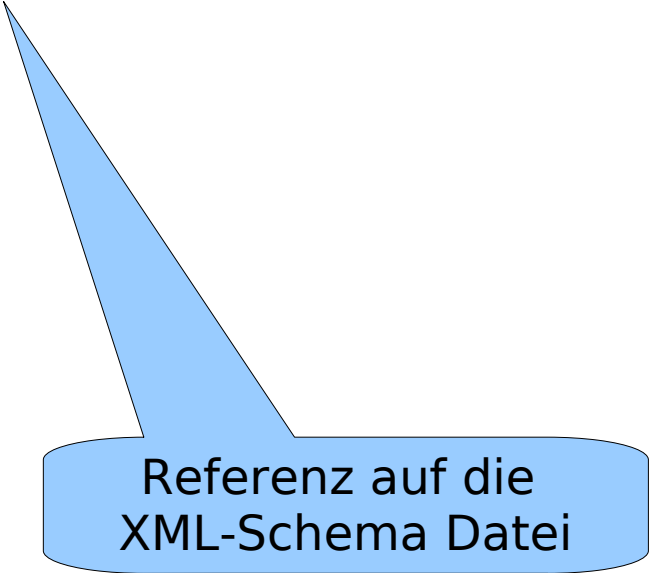
XML-Instanz und XML-Schema



XML

XML-Instanz Beispiel

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<Buch Name="XML Grundlagen"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:noNamespaceSchemaLocation="Buch.xsd">
  <isbn id="031-3384417"/>
  <Verlag nb="1"/>
  <Kapitel Titel="Erstes Kapitel">
    Dieses Kapitel enthält nur eine
    kleine Beschreibung.
  </Kapitel>
  <Kapitel Titel="Letztes Kapitel">
    Ende gut, alles gut ;- )
  </Kapitel>
</Buch>
```



Referenz auf die
XML-Schema Datei

XML

Dazugehöriges XML-Schema (1)

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="Buch">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="isbn" type="isbnType"/>
        <xs:element name="Verlag" type="VerlagType"/>
        <xs:element name="Kapitel" type="KapitelType"
          maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="Name" type="xs:string"
        use="required"/>
    </xs:complexType>
  </xs:element>
  ...

```


XML

Dazugehöriges XML-Schema (2)

...

```
<xs:complexType name="VerlagType">
  <xs:attribute name="nb" type="xs:byte" use="required"/>
</xs:complexType>
<xs:complexType name="isbnType">
  <xs:attribute name="id" type="xs:string" use="required"/>
</xs:complexType>

<xs:complexType name="KapitelType" mixed="true">
  <xs:attribute name="Titel" type="xs:string" use="required"/>
</xs:complexType>
</xs:schema>
```

XML

XML Namespace

Namespace Definition,
als Platzhalter
für eindeutige Definition

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="Buch">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="ISBN" type="isbnType"/>
        <xs:element name="Verlag" type="VerlagType"/>
        <xs:element name="Kapitel" type="KapitelType"
          maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="Name" type="xs:string"
        use="required"/>
    </xs:complexType>
  </xs:element>
  ...

```

Namespace
Anwendung

Ziel: Namenskollisionen **vermeiden**

XML

Unterschied wohl-geformt / validiert

Ein XML-Dokument **muss** wohl-geformt sein.
Es **kann** validiert werden.

Wohl-geformt

- ✓ Ein Wurzel-Element und nur eins
- ✓ Passende Start-/Endelemente
- ✓ Keine überlappende Elemente
- ✓ Eindeutige Attribute innerhalb eines Elements
- ✓ Attribute zwischen "..."
- ✓ etc...

Validiert

Wohl-geformt plus:

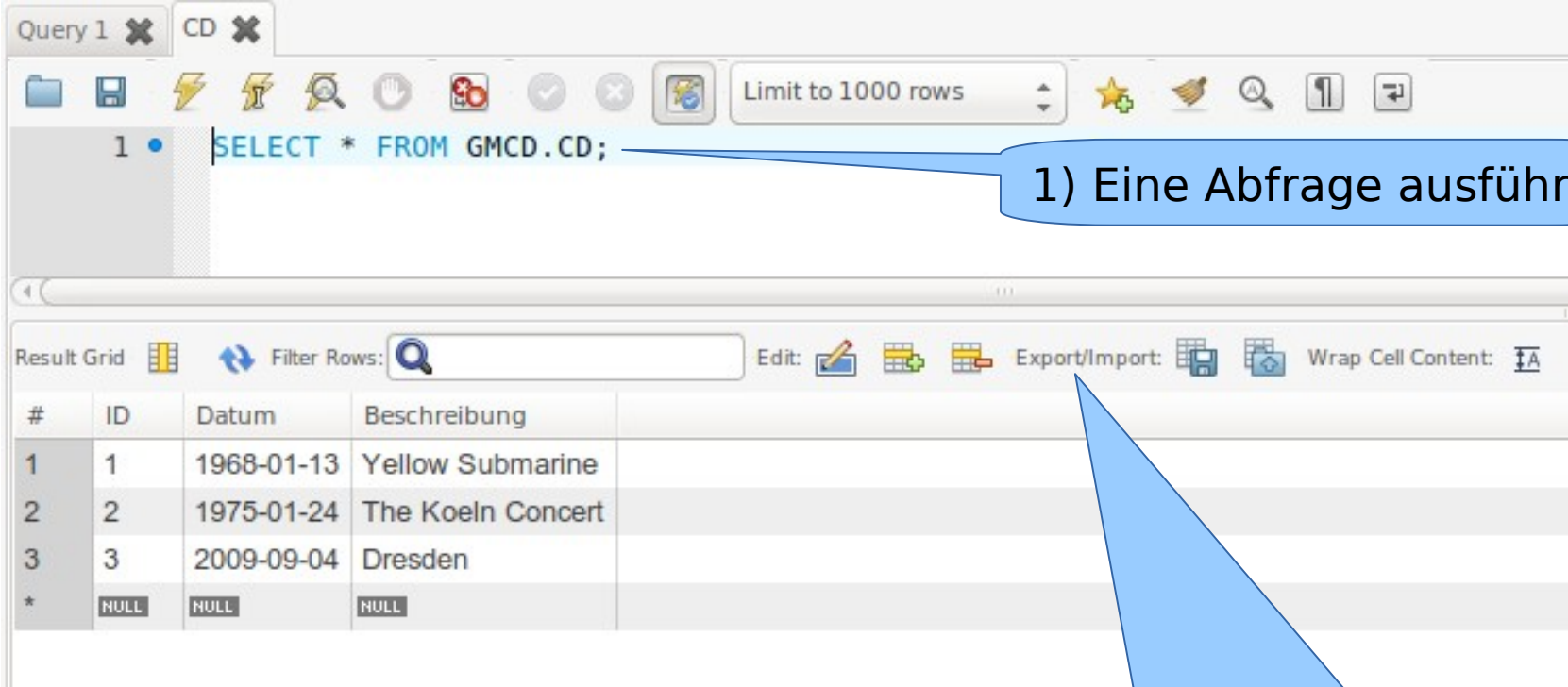
- ✓ Korrekte Dokumentstruktur (Elemente und Attribute)
- ✓ Korrekte Attributtypen
- ✓ XML-Schema basiert

Überprüfung mit einem XML-Parser oder **online**

Reihenfolge
unspezifiziert

XML

Daten-Export mit MySQL-Workbench



The screenshot shows the MySQL Workbench interface. At the top, there are tabs for 'Query 1' and 'CD'. Below the tabs is a toolbar with various icons, including a 'Limit to 1000 rows' dropdown. The main area contains a SQL query: `SELECT * FROM GMCD.CD;`. A blue callout bubble points to the 'Execute' icon (a lightning bolt) in the toolbar, with the text '1) Eine Abfrage ausführen'. Below the query editor is the 'Result Grid' window. It has a toolbar with 'Filter Rows', 'Edit', 'Export/Import', and 'Wrap Cell Content' options. The grid displays the following data:

#	ID	Datum	Beschreibung
1	1	1968-01-13	Yellow Submarine
2	2	1975-01-24	The Koeln Concert
3	3	2009-09-04	Dresden
*	NULL	NULL	NULL

2) In Resultats-Fenster kann ich jetzt die Resultate in XML, JSON, CSV, SQL... exportieren

XML

Beispiel Export aus GMCD / CD

```
<?xml version="1.0"?>

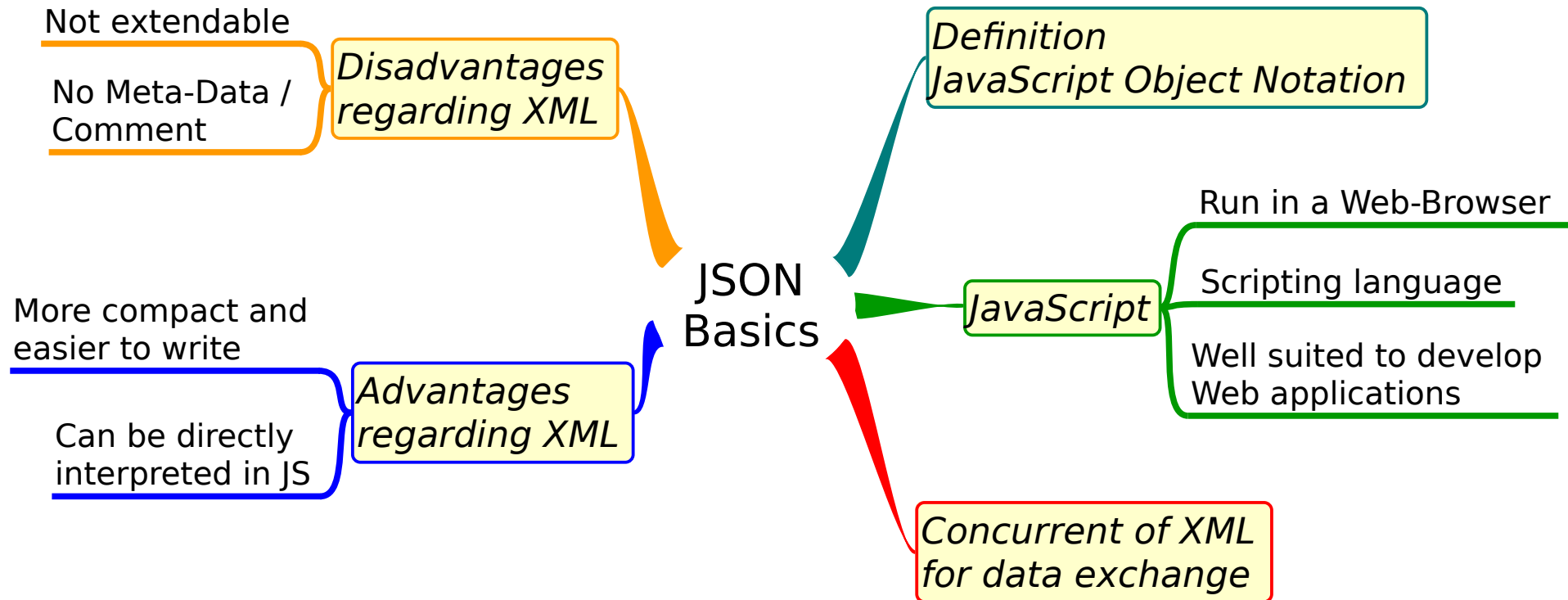
<resultset statement="SELECT * FROM GMCD.CD LIMIT 0, 1000"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

  >> <row>
  >> >> <field name="ID">1</field>
  >> >> <field name="Datum">1968-01-13</field>
  >> >> <field name="Beschreibung">Yellow Submarine</field>
  >> </row>

  >> <row>
  >> >> <field name="ID">2</field>
  >> >> <field name="Datum">1975-01-24</field>
  >> >> <field name="Beschreibung">The Koeln Concert</field>
  >> </row>

  >> <row>
  >> >> <field name="ID">3</field>
  >> >> <field name="Datum">2009-09-04</field>
  >> >> <field name="Beschreibung">Dresden</field>
  >> </row>
</resultset>
```

JSON Grundlagen



Quellen:

https://de.wikipedia.org/wiki/JavaScript_Object_Notation

<https://de.wikipedia.org/wiki/JavaScript>

Tools:

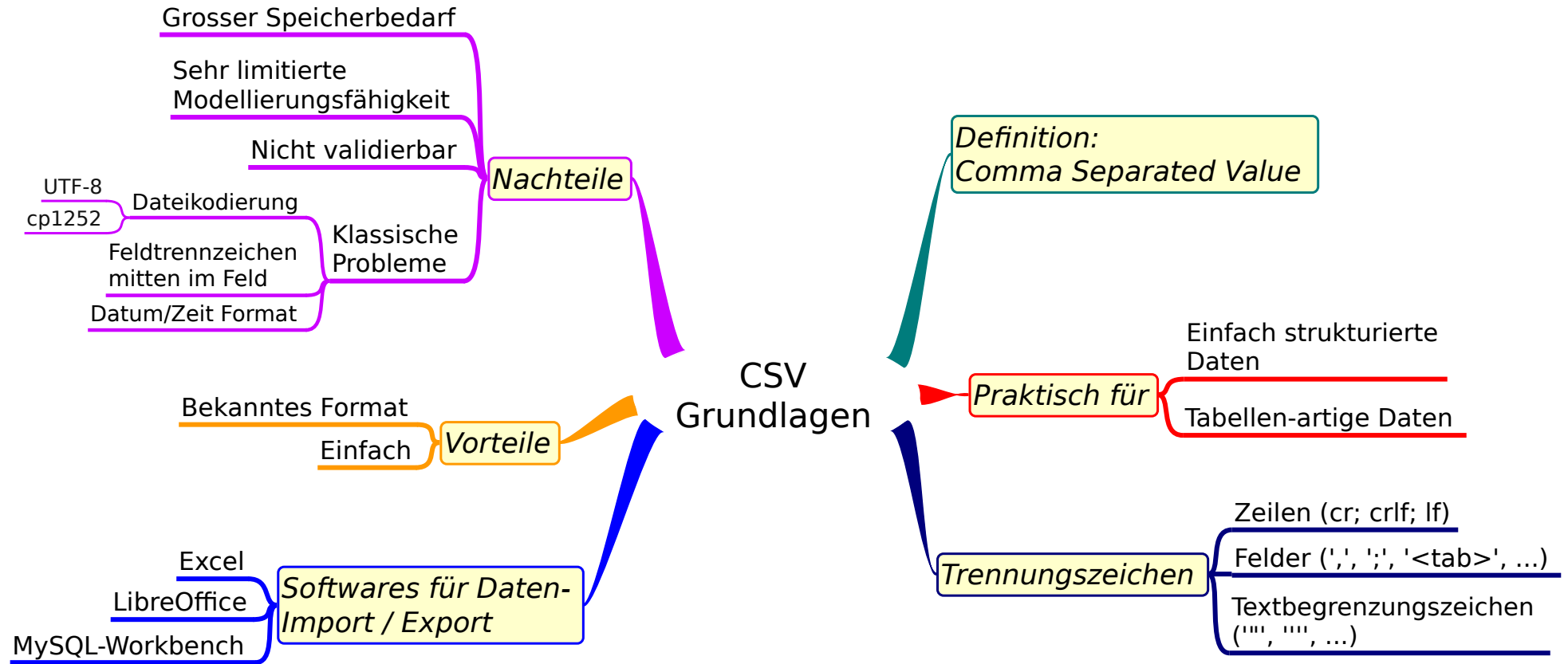
Windows: Notepad++; Linux: Kate

JSON

Beispiel Export aus GMCD / CD

```
[
  >> {
  >>   >> "ID" : 1,
  >>   >> "Datum" : "1968-01-13",
  >>   >> "Beschreibung" : "Yellow Submarine"
  >> },
  >> {
  >>   >> "ID" : 2,
  >>   >> "Datum" : "1975-01-24",
  >>   >> "Beschreibung" : "The Koeln Concert"
  >> },
  >> {
  >>   >> "ID" : 3,
  >>   >> "Datum" : "2009-09-04",
  >>   >> "Beschreibung" : "Dresden"
  >> }
]
```

CSV Grundlagen



Quellen:

[https://de.wikipedia.org/wiki/CSV_\(Dateiformat\)](https://de.wikipedia.org/wiki/CSV_(Dateiformat))

<https://tools.ietf.org/html/rfc4180>

Tools:

Windows: Notepad++, Excel, LibreOffice

CSV

Beispiel Export aus GMCD / CD

Comma Separated
Value

```
ID,Datum,Beschreibung
1,1968-01-13,"Yellow Submarine"
2,1975-01-24,"The Koeln Concert"
3,2009-09-04,Dresden
```

Semicolon Separated
Value

```
ID;Datum;Beschreibung
1;1968-01-13;"Yellow Submarine"
2;1975-01-24;"The Koeln Concert"
3;2009-09-04;Dresden
```

TAB Separated
Value

```
ID»      Datum»   Beschreibung
1»      1968-01-13»   "Yellow Submarine"
2»      1975-01-24»   "The Koeln Concert"
3»      2009-09-04»   Dresden
```

Zeichenkodierung

Problematik und Umgang damit

ASCII / ISO-8859-1

- 1 Bytes fix, 0-127 ASCII, Rest abhängig vom Code Page
- Gleiches System, unterschiedliche Namen
 - Windows-1252
 - Latin-1
- Ideal nur für *westeuropäische* Länder
- Nicht ideal für arabische und asiatische Länder

UNICODE / UTF-8

- Variable Länge des Zeichens
- In der Regel UTF-8 im Einsatz
 - Erste 128 Zeichen = ASCII
- Ideal für fast alle / alles
- *Daten leben länger als Programme =>*
 - Konvertierungstools: Notepad++, Kate, LibreOffice, MySQL, iconv

Links:

- 1) [What is the difference between UTF-8 and ISO-8859-1?](#)
- 2) [Minimum Every Developer Must Know About Character Sets](#)

Übungen

XML, JSON und CSV

- 1) Exportieren Sie aus der GMCD Datenbank die Tabelle "Stück" als XML-Datei (Format: XML-MySQL). Und schauen Sie das Resultat an, mit Kate oder Notepad++.
- 2) Exportieren Sie aus der GMCD Datenbank die Tabelle "Stück" als CSV-Datei. Und importieren Sie die Datei in Excel / LibreOffice.
- 3) Exportieren Sie aus der GMCD Datenbank die Tabelle "Stück" als JSON-Datei. Und schauen Sie das Resultat an, mit Kate oder Notepad++.