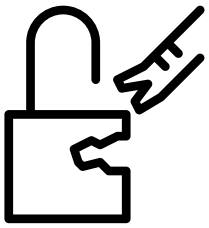
LU10a: Aggregatfunktionen Einführung

0. Datensätze

Wir arbeiten mit **realen Daten** aus unterschiedlichen Bereichen. Das macht die Übungen **praxisnah**: Wir beantworten echte Fragen, erkennen **Trends** und leiten **Kennzahlen** ab. Die drei Datensätze sind aus unterschiedlichen Sparten:

- **Einbrüche** (Kanton Zürich) – Datenbank: `zh_einbrueche`, Tabelle: `einbrueche`
- **Raumfahrt (1957–2022)** – Datenbank: `spacemission`, Tabelle: `missions`
- **Musiktrends (YouTube Top Songs 2025)** – Datenbank: `youtube_top_100_songs_2025`, Tabelle: `youtube_top_100_songs_2025`

Einbrüche (Kanton Zürich)



[1)]

- **Zeitraum:** 2009–2024
- **Quelle:** Kantonspolizei ZH
- **Lizenz:** CC0
- **Granularität:** je **Gemeinde** (Stadt Zürich zusätzlich je **Stadtkreis**)

Was steckt drin? (Auszug Felder)

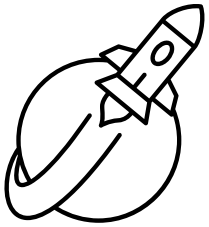
Feld	Bedeutung
`ausgangsjahr`	Berichtsjahr
`gemeindename` / `stadtkreis_name`	örtliche Zuordnung
`straftaten_total`	Vollendet **+** Versucht
`straftaten_vollendet`, `straftaten_versucht`	Aufschlüsselung
`einwohner`	Bevölkerungszahl (Ende Vorjahr)
`haeufigkeitszahl`	Fälle **pro 1’000 Einwohner**

Beispielfragen:

- Wie viele Einbrüche gab es **letztes Jahr** in **meiner Gemeinde**?

- Welche Gemeinden haben im **Durchschnitt** die **höchsten Raten**?

Space Missions



[2)]

- **Zeitraum:** 1957–2022
- **Quelle:** Scrape von nextspaceflight.com
- **Granularität: Start-Ereignis** (Mission)

Was steckt drin? (Auszug Felder)

Feld	Bedeutung
`company`	Betreiber/Organisation (staatlich, militärisch, privat)
`location`	Startplatz (Kosmodrom, Spaceport, Pad)
`launch_date`	Datum/Zeit des Starts
`rocket` / `status_rocket`	Träger & Status (aktiv/retired)
`mission_status`	Erfolg/Fehlschlag u. a.
`price_usd`	Missionskosten (falls vorhanden; Mio. USD)

Beispielfragen:

- Wie viele Starts hat **SpaceX** im Vergleich zu **staatlichen Akteuren** bis **2022**?
- Welche **Jahre** waren **startstark**, welche **pannenanfällig**?
- Welche **aktiven Raketen** haben die **meisten erfolgreichen** Starts?

YouTube Top 100 (2025)

3)

- **Stichtag:** 22.09.2025
- **Umfang:** 100 Songs
- **Granularität: Video/Track** inkl. Kanal-Metadaten

Was steckt drin? (Auszug Felder)

Feld	Bedeutung
`title` / `full_title`	Videotitel (offiziell)
`channel`	Künstler:in/Kanal
`view_count`	Views (bis > 2 Mrd.)
`duration`	Videolänge (ca. 2–6 Minuten)
`tags` / `description`	Metadaten (Genre, Labels, Promo)
`category`	Kategorie (meist **Music**)

Beispielfragen:

- Welche sind die **meistgespielten Songs 2025** und welche **Channels** sind **besonders erfolgreich**?
- Gibt es einen **Zusammenhang** zwischen **Songdauer** und **Anzahl Views**?

Setup (Download & Import)

Laden Sie die vorbereiteten Datensätze herunter und importieren Sie sie:

ZIP mit allen drei SQL-Dumps

Screencast, der zeigt wie die SQL-Files in Webstorm ausgeführt werden. Drei Datenbanken mit Tabellen und Datensätzen werden in MySQL erstellt.

Screencast, der zeigt wie die SQL-Files in Webstorm ausgeführt werden. Drei Datenbanken mit Tabellen und Datensätzen werden in MySQL erstellt.

Schema wählen (wichtig): Entweder im Editor-Dropdown das Schema wählen **oder** mit USE explizit setzen:

```
-- Beispiele:
USE spacemission;
-- USE zh_einbrueche;
-- USE youtube_top_100_songs_2025;
```

1. Was sind Aggregatfunktionen – und warum braucht man

sie?

Aggregatfunktionen fassen **viele Zeilen** zu **Kennzahlen** zusammen (z. B. Anzahl, Summe, Durchschnitt, Minimum/Maximum). Sie sind die Grundlage für **Berichte, Dashboards, BI-Auswertungen**[4] **und Web-Backends**[5].

1.1 Die wichtigsten Aggregatfunktionen

Funktion	Zweck	NULL-Verhalten
`COUNT(*)`	Anzahl Zeilen	zählt **alle** Zeilen
`COUNT(spalte)`	Anzahl **nicht NULL**	**ignoriert** NULL
`SUM(spalte)`	Summe	ignoriert NULL
`AVG(spalte)`	Durchschnitt	ignoriert NULL
`MIN(spalte)`	Kleinster Wert	ignoriert NULL
`MAX(spalte)`	Grösster Wert	ignoriert NULL

Syntax (allgemein)

```
SELECT AGGREGATFUNKTION(ausdruck) AS alias FROM
schema.tabelle;
```

2. Einfache Aggregationen

2.1 COUNT – Wie viele Datensätze sind es?

Frage: Wie viele Missionen befinden sich im Datensatz?

```
SELECT COUNT(*) AS anzahl_missionen
FROM missions;
```

Was passiert? `COUNT(*)` zählt alle Zeilen.

anzahl_missionen
9260

2.2 SUM – Summe bilden

Frage: Wie viele **Views** haben alle 100 YouTube-Songs zusammen?

```
SELECT SUM(view_count) AS views_total
FROM youtube_top_100_songs_2025;
```

Was passiert? `SUM(view_count)` addiert alle Views über die Tabelle.

views_total
10591031907

Ergebnis: 2025 haben die 100 meistgeklickten YouTube-Songs insgesamt über 10 Milliarden Views erzielt.

2.3 AVG – Durchschnitt berechnen

Frage: Wie lang ist ein Song **im Durchschnitt**?

```
SELECT AVG(duration) AS avg_duration
FROM youtube_top_100_songs_2025;
```

Was passiert? `AVG(duration)` berechnet den Mittelwert; NULLs werden ignoriert.

avg_duration
203.9

Ergebnis: 2025 haben die 100 meistgeklickten YouTube-Songs eine durchschnittliche Song-Dauer von 3 min 24s.

2.4 MIN / MAX – Kleinster / Grösster Wert

Fragen: Längster Song? Kürzester Song?

```sql
-- Längster Song
SELECT MAX(duration) AS max_duration
FROM youtube_top_100_songs_2025;

-- Kürzester Song
SELECT MIN(duration) AS min_duration
FROM youtube_top_100_songs_2025;
```

Was passiert? `MAX(…)` gibt den grössten, `MIN(…)` den kleinsten Wert zurück.

max_duration
354

min_duration
120

Ergebnis: Der längste Song in der Tabelle ist „Kendrick Lamar - Not Like Us“ mit einer Dauer von 5 min 54s und der kürzeste Song „Claudia Valentina - Candy (Official Video)“ mit 2 Minuten. </WRAP> </WRAP>

1)

Bildquelle: Robbery von Gofficon, Noun Project – Lizenz: CC BY 3.0

2)

Bildquelle: Space von Zahirulizul, Noun Project – Lizenz: CC BY 3.0
3)

Bildquelle: Song von Puspa Kusuma, Noun Project – Lizenz: CC BY 3.0
4)

Business Intelligence: systematische, kennzahlenbasierte Datenanalyse zur Entscheidungsunterstützung; typische Artefakte sind Reports, Dashboards, Zeitreihen und Rankings.
5)

Serverseitige Logik/Services einer Web-App; liest Datenquellen, validiert, rechnet und stellt Ergebnisse über APIs (Schnittstellen) bereit.