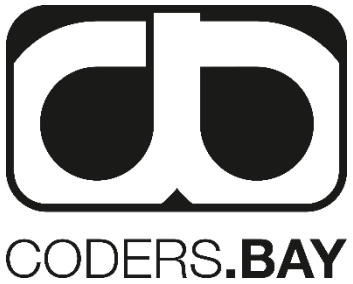
DIE WELT DER DATENBANKEN

DATENANFRAGE MIT SQL

GRUNDLAGEN VON ANFRAGEN



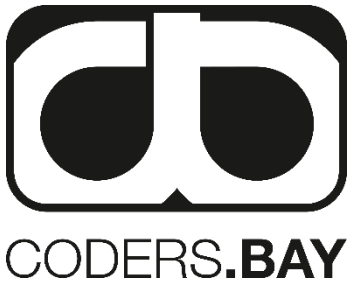
- **Anfrage:** Folge von Operationen
  - Berechnet **Ergebnisrelation** aus **Basisrelation**

- Benutzer formuliert “Was will ich haben?”, und nicht “Wie komme ich an das ran?”

- Ergebnis einer Anfrage ist **wieder eine Relation** und kann wieder als Eingabe für die nächste Anfrage verwendet werden

- **Syntaktisch korrekte Anfragen** können **nicht** zu Endlosschleifen oder unendlichen Ergebnisse führen

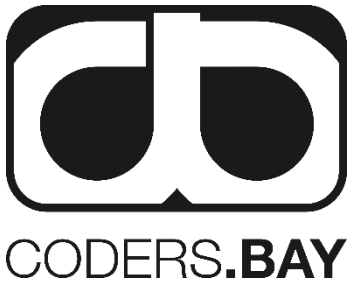
GRUNDLAGEN VON ANFRAGEN



Folgende Anfragen sind möglich

- **Selektion:** Auswahl von Zeilen/Tupel einer Relation
- **Projektion:** Auswahl einer Menge von Spalten einer Relation
- **Kartesisches Produkt:** Verknüpfung jeder Zeile zweier Relationen
- **Umbenennung** von **Attributen** oder **Relationen**
- **Vereinigung:** Liefert die Vereinigung zweier Relationen gleichen Schemas
- **Mengendifferenz:** Liefert Differenz zweier Relationen gleichen Schemas
- **Natürlicher Verbund:** Verknüpfung zweier Relationen über Spalte mit gleichen Attributwerten im gleichen Spaltennamen (doppelt vorkommende Spalten werden weggelassen)
- **Allg. Verbund:** Verknüpfung zweier Relationen, auch wenn sie keine gleichnamige Spalte haben. Verbund aufgrund logischer Bedingung)

GRUNDLAGEN VON ANFRAGEN



Keywords

SELECT: Projektionsliste, Abfrage von Daten
FROM: zu verarbeitende Relation
WHERE: Selektions-, oder Verbundbedingungen
GROUP BY: Gruppierung für Aggregatfunktionen
HAVING: Selektionsbedingungen für Gruppen
ORDER BY: Sortierung der Ergebnisrelation

```
SELECT attribute
FROM tabelle
WHERE bedingungen
```

GRUNDLAGEN VON ANFRAGEN

CODERS.BAY

Beispiel

Geben Sie Personalnummer und Name aller C4-Professoren an:

Professoren

PersNr	Name	Rang	Raum
2125	Sokrates	C4	226
2126	Russel	C4	232
2127	Kopernikus	C3	310
2133	Popper	C3	52
2134	Augustinus	C3	309
2136	Curie	C4	36
2137	Kant	C4	7

```
SELECT PersNr, Name
FROM Professoren
WHERE Rang = 'C4';
```

$$\pi_{\text{PersNr, Name}}(\sigma_{\text{Rang='C4'}}(\text{Professoren}))$$

Ergebnis

PersNr	Name
2125	Sokrates
2126	Russel
2136	Curie
2137	Kant

GRUNDLAGEN VON ANFRAGEN

CODERS.BAY

Beispiel

```
SELECT *
FROM Professoren;
```

Professoren

PersNr	Name	Rang	Raum
2125	Sokrates	C4	226
2126	Russel	C4	232
2127	Kopernikus	C3	310
2133	Popper	C3	52
2134	Augustinus	C3	309
2136	Curie	C4	36
2137	Kant	C4	7

GRUNDLAGEN VON ANFRAGEN

CODERS.BAY

DISTINCT: Ergebnismenge ist frei von Duplikaten

Duplikatelimination

Geben Sie alle Rangbezeichnungen für Professoren ohne Duplikate aus.

```
SELECT DISTINCT Rang
FROM Professoren;
```

Ergebnis

Rang
C4
C3

Beispiel ohne DISTINCT:

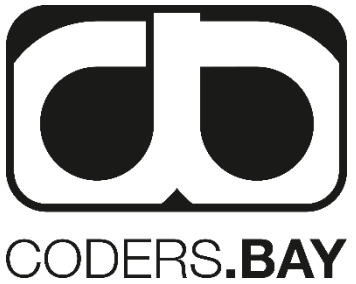
Keine Duplikatelimination

```
SELECT ALL Rang
FROM Professoren;
```

Ergebnis

Rang
C4
C4
C3
C3
C3
C4
C4

GRUNDLAGEN VON ANFRAGEN



ALIASNAME:

- Benennt Spalte in Ergebnisrelation.
- Wird direkt nach dem Spaltennamen angegeben.
- Keyword: **AS**

Spaltenüberschrift

```
SELECT PersNr AS Personalnummer, Name Familienname
FROM Professoren;
```

Ergebnis

PersNr → | **Personalnummer** | **Familienname** | ← Name

Personalnummer	Familienname
2125	Sokrates
2126	Russel
2127	Kopernikus
2133	Popper
2134	Augustinus
2136	Curie
2137	Kant

GRUNDLAGEN VON ANFRAGEN

CODERS.BAY

Sortierung:

- Klausel steht am Ende der Anfrage.
- Keyword: **ORDER-BY**

Beispiel

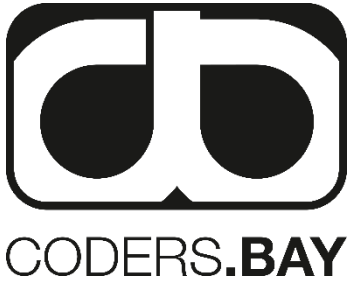
```
SELECT PersNr, Name, Rang
FROM Professoren
ORDER BY Rang DESC, Name ASC;
```

Ergebnis

PersNr	Name	Rang
2136	Curie	C4
2137	Kant	C4
2126	Russel	C4
2125	Sokrates	C4
2134	Augustinus	C3
2127	Kopernikus	C3
2133	Popper	C3

WHERE-CLAUSE

WHERE-CLAUSE



Bei der Auswahl beziehungsweise Filterung mithilfe von

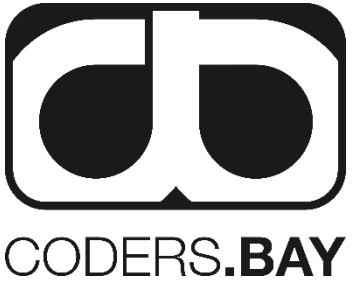
WHERE innerhalb der

SELECT-Anweisung kannst

du Vergleichsoperatoren anwenden

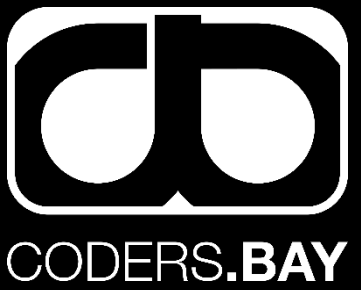
Operator	Bedeutung
=	gleich
<>	ungleich
>	größer als
>=	größer als oder gleich
<	kleiner als
<=	kleiner als oder gleich
NOT	Der Wahrheitswert einer Bedingung wird umgekehrt
AND	Alle Bedingungen müssen zutreffen
OR	Mindestens eine Bedingung muss zutreffen

WHERE-CLAUSE



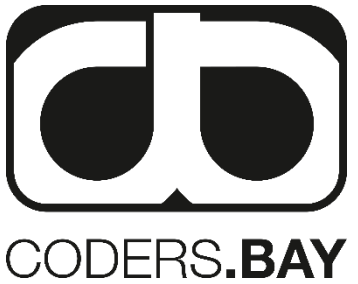
LIKE – für Zeichenketten

- **%** - Beliebige Anzahl unbekannter Zeichen
- _ = genau ein unbekanntes Zeichen



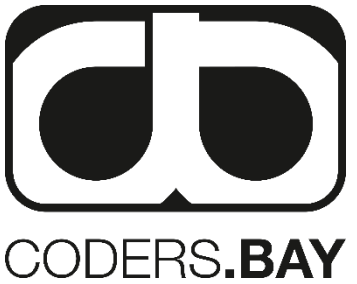
AUFGABEN

SELECT STATEMENTS



- Liste aller Mitarbeiter mit allen Informationen erstellen
- Liste aller Mitarbeiter mit deren Vor- und Nachnamen
- Liste aller Nachnamen alphabetisch geordnet
- Liste aller Managers (manager_id) ohne Duplikate
- Liste aller Mitarbeiter, die den Manager mit der ID 100 haben

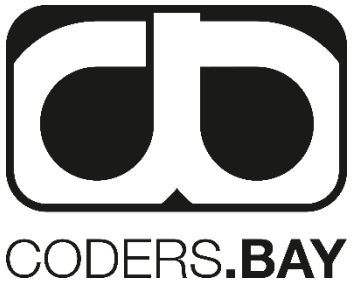
SELECT STATEMENTS



- Gib alle Ländernamen (country_name) der Tabelle „countries“ aus
- Gib alle Städte (city) und den zugehörigen Länder Code (country_id) der Tabelle „locations“ aus
- Gib alle Regionen (region_name) der Tabelle „regions“ aus und gib der Tabellenspalte den Namen „Region“
- Gib alle Jobtitel (job_title) und die zugehörige ID (job_id) der Tabelle „jobs“ aus und ordne sie aufsteigend abhängig vom Jobtitel.
- Gib alle Location IDs (location_id) der Tabelle „departments“ aus und sorge dafür, dass jeder Eintrag nur einmal vorkommt.

MYSQL FUNKTIONEN

MYSQL FUNKTIONEN

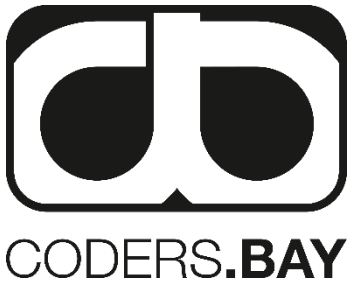


Spaltennamen der Zielrelation definieren mit AS

BSP:

```
SELECT salary AS Gehalt
FROM employees
```

MYSQL FUNKTIONEN

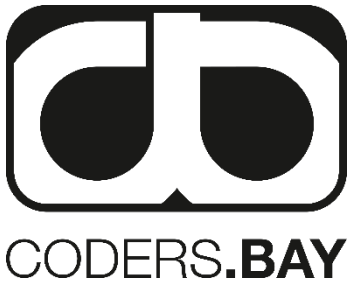


Durchschnitt mit `AVG()`

BSP:

```
SELECT AVG(salary) AS Durchschnittsgehalt
FROM employees
```

MYSQL FUNKTIONEN

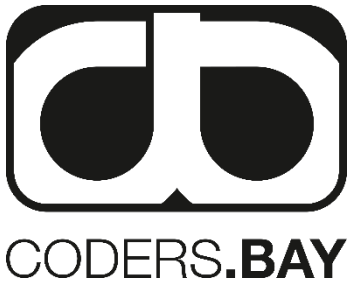


Summenbildung mit `SUM()`

BSP:

```
SELECT SUM(salary), job_id
FROM employees
GROUP BY job_id
ORDER BY SUM(salary) DESC;
```

MYSQL FUNKTIONEN

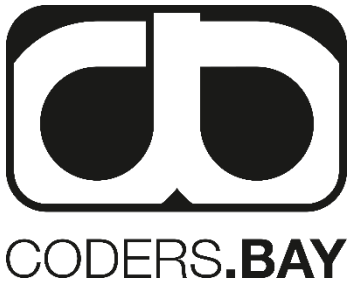


Zeichenketten zusammenführen mit `CONCAT()` / `CONCAT_WS()`

BSP:

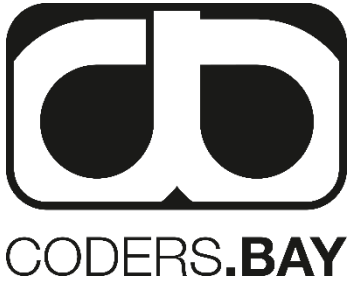
```
SELECT CONCAT(first_name, ' ', last_name) AS Name
FROM employees

SELECT CONCAT_WS(' ', first_name, last_name) AS Name
FROM employees
```

UNION




```
SELECT job_id, department_id
FROM employees
WHERE department_id = 10
UNION
SELECT job_id, department_id
FROM  employees
WHERE department_id = 20
```

AUFGABEN



- Gib eine Liste aller Mitarbeiter aus deren Vorname mit „K“ beginnt
- Gib eine Liste aller Mitarbeiter aus, die nicht Peter oder Eleni heißen
- Gib eine Liste aller Mitarbeiter aus, die mehr als 10 000 verdienen
- Gib eine Liste aller Abteilungen aus, die die location_id 1700 besitzen
- Gib alle Mitarbeiter aus mit einer Telefonnummer welche die Ziffernreihenfolge 121 an beliebiger Stelle beinhaltet