	GROUP BY		AEuP	V 1.0
	Name	Klasse	Datum	

1 Gruppieren von Ergebnissen

Eine denkbare Aufgabenstellung für die nebenstehende Tabelle wäre, die Anzahl der Schüler pro Klasse auszugeben. Solche Statements werden durch das `GROUP BY` ermöglicht. Folgender Merksatz hilft hierbei:

SID	Name	Klasse	D_Note
1	Fischer	11bFI	1
2	Müller	11cFI	2
3	Wagner	11aFI	3
4	Schlegel	11bFI	5
5	Kern	11aFI	6
6	Beyer	11cFI	4
7	Martens	11cFI	2
8	Müller	11bFI	5
9	Wagner	11cFI	2
10	Müller	11aFI	4
11	Schuster	11aFI	6
12	Eckert	11aFI	3
13	Maier	11aFI	4
14	Müller	11bFI	6
15	Berger	11aFI	3
16	Kaminski	11cFI	2
17	Schmidt	11bFI	3
18	König	11aFI	4
19	Mohr	11aFI	4
20	Unger	11bFI	1

Wenn wir **pro** Eigenschaft (oder **je** Eigenschaft) Daten suchen, dann ist meist ein „GROUP BY“ eine mögliche Lösung.

Sehen wir uns hierfür folgendes `SELECT` an:

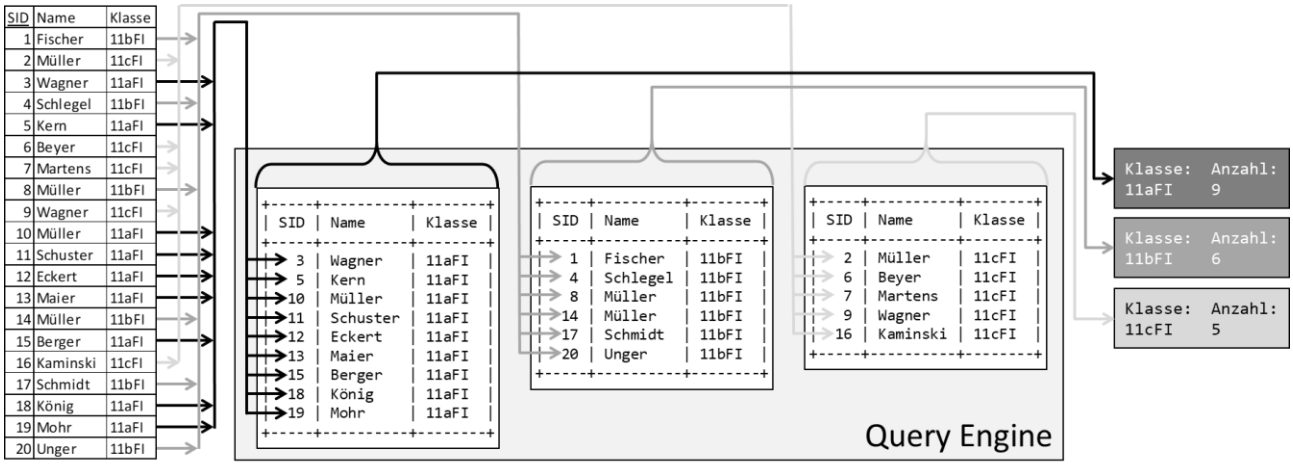
```
SELECT Klasse, COUNT(*) AS Anzahl
FROM Schueler
GROUP BY Klasse;
```

Welches Ergebnis ist hier zu erwarten:

```
+--------+--------+
| Klasse | Anzahl |
+--------+--------+
|
```

Achten Sie hier darauf, dass die alleinige Ausgabe der Anzahl nicht aussagekräftig ist. Wir benötigen immer neben der Anzahl auch die Gruppe, für die die Anzahl gilt!

Es ist wichtig zu verstehen, dass bei einem `GROUP BY` der Zugriff auf einen einzelnen Datensatz nicht mehr möglich ist. Sehen wir uns hierzu folgendes Bild an:



Die Query Engine sammelt also die Datensätze in Gruppen (entsprechend des Feldes / der Felder nach „`GROUP BY`“ und gibt sie dann aggregiert als **ein Datensatz pro Gruppe** aus.

Nach dem „`SELECT`“ sollten demnach nur Eigenschaften stehen, welche für die gesamte Gruppe gelten, also alle Aggregatsfunktionen, bzw. Eigenschaften, welche für jeden Datensatz der Gruppe zutreffen – in unserem Beispiel ist das die Spalte „Klasse“.

Nun können wir die Datensätze, welche durch die Query Engine verarbeitet werden mit `WHERE` filtern. Dies ist also eine Reduktion der verarbeiteten Datensätze am **Eingang der Query Engine** aus dem vorausgegangenen Bild. Wenn wir also nur die Datensätze mit SID < 10 haben wollen, so sieht das `SELECT` wie folgt aus:

```
SELECT Klasse, COUNT(*) AS Anzahl
FROM Schueler
WHERE SID < 10
GROUP BY Klasse;
```

Das Ergebnis ist wie folgt:

```
+--------+--------+
| Klasse | Anzahl |
+--------+--------+
| 11bFI  |      3 |
| 11cFI  |      4 |
| 11aFI  |      2 |
+--------+--------+
```

Es gehen also nur die SIDs kleiner 10 in die Gruppenbildung ein, alle anderen werden ignoriert.

Nun kann man daneben aber auch noch die auszugebenden Gruppen einschränken, was mit der `HAVING` Klausel erfolgt. Dies geht sinnvollerweise jedoch nur bei Aggregatsfunktionsergebnissen. Wollen wir bspw. die vorausgegangene Query so erweitern, dass nur die Gruppen ausgegeben werden sollen, die maximal 3 Datensätze aufweisen, so erfolgt dies durch:

```
SELECT Klasse, COUNT(*) AS Anzahl
FROM Schueler
WHERE SID < 10
GROUP BY Klasse
HAVING Anzahl <= 3;
```

Das Ergebnis ist wie folgt:

```
+--------+--------+
| Klasse | Anzahl |
+--------+--------+
| 11bFI  |      3 |
| 11aFI  |      2 |
+--------+--------+
```

Alternativ wäre auch `HAVING COUNT(*) <= 3` funktioniert.

Formulieren Sie nun für folgende Aufgabenstellung ein SELECT:

Geben Sie die Klassen aus, bei denen die schlechteste Note mindestens die 5 ist: