	SELECT Statement		AEuP	V 1.2
	Name	Klasse	Datum	

1 Grundaufbau von SELECT Statements

Ein `SELECT` Statement wird aus verschiedenen Teilelementen aufgebaut:

Keyword:	Zusatzinformation:	Erklärung:	Pflicht:
`SELECT`	*Spalten*	Schlüsselwort für jede Abfrage.	✓
`FROM`	*Tabellen*	Tabellen, Views oder Subselects.	✓
`WHERE`	*Datensatzeigenschaften*	Bedingung gilt für jeden Datensatz.	
`GROUP BY`	*Gruppierungseigenschaften*	Spalten, nach denen gruppiert wird.[1]	
`HAVING`	*Selektionsbedingung Gruppe*	Meist Aggregatsfunktionen	
`ORDER BY`	*Sortierung der Ausgabe*	Spalten, nach denen sortiert wird.	
`LIMIT`	*Anzahl und Starpunkt*	Limitiert die Ausgabemenge	

Die oberen beiden Teilelemente sind zwingend immer erforderlich. Hier nochmal die einzelnen Elemente:

2 SELECT

Nach dem **`SELECT`** folgen Informationen über die Spaltenausgabe. Dies kann sein:

- Spalten der Quelldaten (also das, was nach dem „`FROM`“ steht)
- Statische Informationen, wie bspw. "x“ – dadurch steht das "x" in jeder Zeile
- Aggregatsfunktionen (wird später näher behandelt)
- Andere SQL-Funktionen wie bspw. Stringmanipulationen, Datumsfunktionen o.Ä.
- Subselects auf Zeilenebene (wird später näher behandelt)
- Kombinationen aus den vorausgegangenen Informationen in Form von Rechnungen, bspw. "Preis * Anzahl"
- * für alle Spalten der Quelldaten (bitte nicht in Softwarepaketen oder Skripten nutzen!)

Die einzelnen Elemente werden über Kommas separiert. Es ist möglich den einzelnen Ausgabespalten einen Aliasnamen zu geben, so dass in der Ausgabe dieser als Spaltenüberschrift entsteht:

```
SELECT Anzahl * Preis AS Gesamtpreis
FROM MyTabelle;
```

```
+-------------+
| Gesamtpreis |
+-------------+
|         100 |
|         150 |
+-------------+
```

Ein Keyword, welches nach dem SELECT ebenfalls erlaubt ist, ist **`DISTINCT`**. `DISTINCT` darf außerhalb von Funktionen immer nur direkt nach dem `SELECT` stehen. Es bewirkt, dass die Ausgabemenge keine Duplikate enthält.

MyTable

ID	X	Y
1	A	1
2	B	1
3	A	2
4	B	2
5	A	1
6	B	1

```
SELECT DISTINCT X, Y
FROM MyTable
```

```
+---+---+
| X | Y |
+---+---+
| A | 1 |
| B | 1 |
| A | 2 |
| B | 2 |
+---+---+
```

3 FROM

Hinter dem „**`FROM`**“ steht immer eine „Datenmenge“. Dies kann entweder:

- eine Tabelle
- ein Subselect (wird später behandelt)
- eine View (wird später behandelt)

sein.

[1] Wird später behandelt

4 WHERE

Hier geben wir die Eigenschaften eines jeden Datensatzes der selektierten Datenmenge an. Pro Datensatz wird entsprechend verglichen, ob die Eigenschaften übereinstimmen, und nur dann wird der Datensatz zur Verarbeitung weitergeleitet.

Bspw. würde `WHERE Preis > 100` nur die Datensätze aus der Datenquelle übernehmen, welche auch wirklich einen Wert **über 100** haben. Sämtliche Vergleichsoperatoren sind gültig (`<`, `<=`, `>`, `>=`, `<>`[2]). Für die Prüfung auf `NULL` ist nur `IS` bzw. `IS NOT` zulässig.

Weiterhin existiert die Möglichkeit mit `LIKE` zu arbeiten, was die Wildcards `%` für kein, ein oder beliebig viele Zeichen steht, bzw. `_` um exakt ein beliebiges Zeichen zu fordern.

Mehrere Bedingungen werden mit `AND` bzw. `OR` verknüpft (wobei `NOT` ebenfalls existiert). Hierbei ist darauf zu achten, dass gilt **AND vor OR**:

Für eine Priorisierung von OR müssen Klammern wie in der Mathematik verwendet werden.

MyTable

ID	X	Y
1	A	1
2	B	1
3	A	2
4	B	2
5	A	1
6	B	1

```
SELECT * FROM MyTable
WHERE Y = 2 OR Y = 1
  AND X = "A";
```

```
+------------+
| ID | X | Y |
+----+---+---+
|  1 | A | 1 |
|  3 | A | 2 |
|  4 | B | 2 |
|  5 | A | 1 |
+----+---+---+
```

Für die Prüfung auf Mengen existiert IN (bzw. NOT IN), bspw. `WHERE FARBE IN ("green", "red")` was einer Verknüpfung der Bedingungen `FARBE = "green" OR FARBE = "red"` entspricht.

Bereiche können auch mit „`BETWEEN`“ angegeben werden. Folgende beiden Bedingungen sind identisch:

```
WHERE Gebdat BETWEEN "2000-01-01" AND "200-01-15"
WHERE Gebdat >= "2000-01-01" AND Gebdat <= "200-01-15"
```

5 ORDER BY

`ORDER BY` sortiert die Ausgabedaten nach den gegebenen Spalten, wobei `ASC` für Aufsteigend und `DESC` für Absteigend steht:

Ohne Angabe von `ASC` bzw. `DESC` wird `ASC` als Standard angenommen.

MyTable

ID	X	Y
1	A	1
2	B	1
3	A	2
4	B	2
5	A	1
6	B	1

```
SELECT * FROM MyTable
ORDER BY
 X ASC, Y DESC;
```

```
+------------+
| ID | X | Y |
+----+---+---+
|  3 | a | 2 |
|  1 | a | 1 |
|  5 | a | 1 |
|  4 | b | 2 |
|  2 | b | 1 |
|  6 | b | 1 |
+----+---+---+
```

6 LIMIT

`LIMIT` schränkt die selektierten Daten basierend auf Anzahl (und optional Position) ein. Der erste Parameter gibt an, wie viele Datensätze bei der Ausgabe übersprungen werden müssen und der zweite die Anzahl der auszugebenden Werte.

Wird nur ein Parameter angegeben, so gibt `LIMIT` die Datensätze ab dem ersten, beschränkt auf die angegebene Zahl aus. Bspw. gibt `LIMIT 3` die erste 3 Datensätze aus

MyTable

ID	X	Y
1	A	1
2	B	1
3	A	2
4	B	2
5	A	1
6	B	1

```
SELECT * FROM MyTable
LIMIT 3, 2;
```

```
+----+---+---+
| ID | X | Y |
+----+---+---+
|  4 | B | 2 |
|  5 | A | 1 |
+----+---+---+
```

[2] Je nach Version auch `!=`