BASIC SQL SYNTAX

Keyword	Explanation
SELECT	Specify columns or expressions for the result.
FROM	Specifies the table from which to select the data.
JOIN	Add another table to the query and connect it to previous tables. Different types of joins exist.
WHERE	Introduces the filter conditions for rows.
GROUP BY	Creates groups of identical values in columns.
HAVING	Specifies that only rows where aggregate values meet the specified conditions should be returned. Used because the WHERE keyword cannot be used with aggregate functions.
ORDER BY	Introduces a list of columns to sort the result by (and their sort order ASC or DESC).

AGGREGATE FUNCTIONS

Function	Explanation
COUNT	Counts occurrences, e.g. number of rows.
SUM	Sums up a column or expression.
AVG	Calculates the average value of a column or expression.
MIN	Retrieve the smallest value in a column or expression.
MAX	Retrieve the largest value in a column or expression.
COUNT(DISTINCT <col>)	Counts only unique values.

COMMON OPERATORS IN CONDITIONS

Operator	Explanation
=	Compares two values if they are equal.
> or >=	Compares two values and returns true if the left is greater than (or equal to) the right.
< or <=	Compares two values and returns true if the left is less than (or equal to) the right.
<>	Compares two values if they are not equal.
LIKE	Compares two strings using the wildcard search operator %.
AND, OR	These operators logically combine two conditions.
NOT	Negates the expression's result.
IS NULL or IS NOT NULL	Special operator to include or exclude NULL values.
DISTINCT	Eliminates duplicates from the results

COMMON MATH FUNCTIONS

Function / Operator	Explanation
+ - / *	The basic arithmetic operators.
POW(a, n)	Calculates a to the power of n.
SQRT	Calculates the square root of a.
ROUND, CEIL, FLOOR	Rounding decimal numbers.

JOIN TYPES

Join	Explanation
INNER JOIN	Only matches in both tables are in the result.
LEFT JOIN	All rows from the left table are in the result. If no match in the right table, NULL values are filled in.
RIGHT JOIN	All rows from the right table are in the result. If no match in the left table, NULL values are filled in.
FULL OUTER JOIN	All values from both tables are in the result. Missing matches are filled with NULL values on both sides.

SUB-QUERIES

You can substitute any single value with a query that returns **one single value**. For example, to calculate a percentage:

```
SELECT COUNT(1) / ( SELECT COUNT(1) FROM all )
FROM all
WHERE gender = 'f'
```

You can also use subqueries instead of tables and select from the result of a subquery:

```
SELECT * FROM
  ( SELECT a, b FROM sometable
    WHERE a > b
    OR x LIKE '%Y%'
  ) sub
WHERE sub.a = 1
```

COMMON STRING FUNCTIONS

Function	Explanation
LENGTH()	Calculates the number of characters in a string.
SUBSTRING(col, p, l)	Get the part of a string starting at **p** with length **l**.
INSTR(col, substring)	Locate the position of the first occurrence of substr column in the given string
UPPER, LOWER	Convert a string to all capital or lower letters.
TRIM	Remove white spaces at both ends of the string.
REGEXP_REPLACE(c, p, r)	Replace all substrings of the specified string value that match the given regexp.

COMMON DATE & TIME FUNCTIONS

Function	Explanation
YEAR, MONTH	Extract the year or month from a date column.
DAYOFYEAR, DAYOFMONTH, DAYOFWEEK	Extracts the day in relation to year, month or week.
WEEKOFYEAR	Extracts the week number 1 - 52 from a date column.
DATEDIFF(end, start)	Returns the number of days from start to end.
DATE_FORMAT	Converts a date/timestamp/string to a value of string in the given format.
DATE_ADD(start, days)	Add days to a given date.
HOUR, MINUTE, SECOND	Extracts the hour/minute/second from a timestamp.

SET OPERATORS

Operator	Explanation
UNION	Creates the union set of two result sets. Contains all rows from both sets (**no** duplicates).
UNION ALL	Creates the union set of two result sets. Contains all rows from both sets (**with** duplicates).
EXCEPT	Subtracts the second set from the first. The result contains only rows that are exclusively in the first set.
INTERSECT	Creates the intersection of two result sets. The result contains only rows that are in both sets.
WHERE EXISTS (...)	Returns true if the given subset is not empty.
WHERE a IN (...)	Compares the values of a if they are in the given subset.

COMMON STATISTICAL FUNCTIONS

Operator	Explanation
MEAN	Same as **AVG**, calculates the arithmetic mean.
PERCENTILE(col, p)	Calculates the percentile p, for example median with p = 0.5
STDDEV()	Calculates the standard deviation.
CORR()	Calculates Pearson's correlation coefficient.

WINDOW FUNCTIONS

A window function performs a calculation across a set of table rows that are somehow related to the current row. This is comparable to the type of calculation that can be done with an aggregate function. But unlike regular aggregate functions, use of a window function does not cause rows to become grouped into a single output row — the rows retain their separate identities. Behind the scenes, the window function is able to access more than just the current row of the query result.

A window function is introduced by the keyword **OVER**:

```
SELECT
  name
 ,score
 ,avg(score) OVER (PARTITION by peergroup) as avgInPeer
FROM all
```

For some window functions, we need to specify an order for the values within the partition:

```
SELECT
  name
 ,score
 ,rank() OVER (PARTITION by peergroup ORDER BY score DESC)
FROM all
```

Function	Explanation
COUNT, SUM, AVG, MIN, MAX	All aggregation functions can be used in the context of a window function.
LAG, LEAD	Returns the value of the previous or next row within the partition with an order.
FIRST_VALUE, LAST_VALUE	Returns the first or last value within a partition with an order.
RANK	Assigns ranks for values within a partition with an order.
ROW_NUMBER	Assigns a row number to every line in the partition with an order.

ARRAY FUNCTIONS

Function	Explanation
`SIZE(col)`	Determines the number of entries in an array.
`EXPLODE(col)`	Creates one row for each entry in an array.
`ARRAY_CONTAINS(col, search)`	Returns TRUE, if the array contains the searched element.
`TRANSFORM(col, func)`	Transforms all entries using the given function.
`ARRAY_DISTINCT`	Remove duplicate entries from an array.