NOTE:
"100\%" if you want to match "100%"

Clauses

SELECT <attrs., func list>
FROM <table list>
WHERE <condition>
GROUP BY <grouping attr(s)>
HAVING <group condition>
ORDER BY <attr list>

union

combines two SELECT queries, removes duplicates

```
select a from ta
UNION
select b from tb
```

$c_1 + c_2$

union all

does not remove duplicates

```
select a from ta
UNION ALL
select b from tb
```

like

% : any substring
_ : single char

```
select name from table1
where name LIKE 'A%'
```

order by

sorts result
ASC (default) / DESC

```
select salary from table1
ORDER BY name desc;
```

distinct

prevent repetition of tuples

```
select DISTINCT fname, lname
from table1;
```

where

corresponds to selection predicate
<> : not equal to

```
select salary from table1
WHERE salary <> 15000 AND
salary < 20000
```

intersect all

if value occurs c_1 times in one query & c_2 times in another,
it occurs here $\min(c_1, c_2)$ times

intersect

only common values between two SELECTS, no duplicates

except all

$\max(c_1 - c_2, 0)$

except

values in first query but not in second query, no duplicates

group by + join

relations first joined based on whatever condition
↓
grouping → aggregation

having

conditions on the entire group of tuples

```
SELECT pno, pname, COUNT(*) from project, works_on
where pno = pnumber GROUP BY pno, pname HAVING COUNT(*) > 2;
```

selects those groups with a count greater than 2

NOTE: select clause arithmetic

SELECT can contain basic operations like +, -, *, /

```
select annual_salary/12 as monthly;
```

NULL values

- (+, -, *, /) → NULL
- Comparison → UNKNOWN [NULL = NULL → UNKNOWN]
- Every NULL value treated as different
- AND, NOT, OR → short circuit evaluation
- WHERE = FALSE/UNKNOWN → tuple not added to result
  - IS NULL / IS NOT NULL
  - IS UNKNOWN / IS NOT UNKNOWN & result of comparison
- NULL values treated as identical for DISTINCT, UNION, INTERSECTION, EXCEPT

Aggregate Functions

SUM, MIN, MAX, COUNT, AVG

- Multiset : set with duplicates
- COUNT (DISTINCT SALARY)
- If all values in collection NULL
  - COUNT = 0
  - everything else = NULL

if one value NULL and you try AVG (example), it will just disregard that value and do it for the remaining

SOME = True if at least one element True
ALL = True if all elements True

```
select count(*) from employees, dept where ssn = essn
    and salary > 45000
    and dno in (select dno from employee group by dno having
                count(*) >2)
    group by dno;
```