SQL statistics

COMS10012 Software Tools

Statistics

COUNT

```
SELECT COUNT(1) FROM Student;

SELECT COUNT(grade) FROM Student;

SELECT COUNT(DISTINCT name) FROM Student;
```

Statistics

Any query containing a statistical function is a statistics query.

These queries only return one row (for now).

In a statistics query, *all* columns must be statistics (or constants) – you cannot mix:

```
SELECT name, COUNT(1) FROM Student;
```

Statistics

```
SELECT MIN(grade), MAX(grade),
       AVG(grade), COUNT(grade)
FROM Student;
```

More statistics

https://mariadb.com/kb/en/aggregate-functions/

https://github.com/infusion/udf_infusion

SQLITE: https://www.sqlite.org/contrib (math lib)

Weighted average

Unit	CP	Grade
COMS20001	10	65
COMS20002	20	60

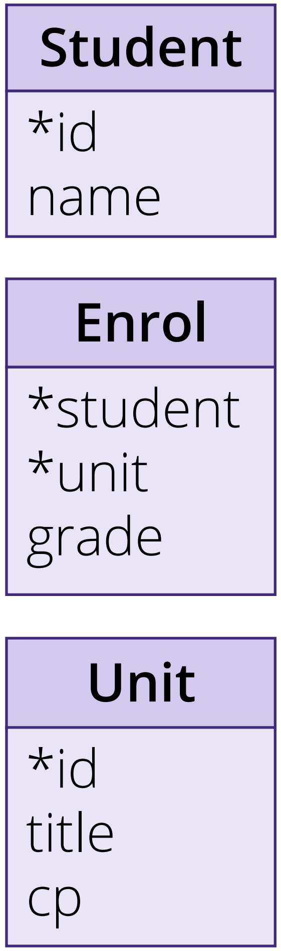
Student
*id
name

Enrol
*student
*unit
grade

Unit
*id
title
cp

SUM

```
SELECT SUM(cp * E.grade) / SUM(cp)
       AS average
FROM Enrol E
INNER JOIN Student
ON E.student = Student.id
INNER JOIN Unit
ON E.unit = Unit.id
WHERE Student.id = 1234567;
```



Rank queries

Bad (syntax error):

```
SELECT name, grade FROM Student
WHERE grade = MAX(grade);
```

Correct:

```
SELECT name, grade FROM Student
WHERE grade =
  (SELECT MAX(grade) FROM Student);
```

GROUP BY

Scenario

Student

id	name
200	David
201	Zoe

Unit

id	title
100	Databases
101	Security

Enrol

unit	student	grade
100	200	60
100	201	75
101	201	85

student	average
200	60
201	80

GROUP BY

```
SELECT Student, AVG(grade) AS average
FROM Enrol
GROUP BY Student;
```

Enrol

unit	student	grade
100	200	60
100	201	75
101	201	85

student	average
200	60
201	80

GROUP BY

GROUP BY queries are always statistical.

1. Take the table defined by the FROM, JOIN and WHERE clauses.

2. Split it into blocks based on the GROUP BY column(s).

3. Compute the statistics once for each block.

GROUP BY

Enrol

unit	student	grade
100	200	60
101	200	50
100	201	75
101	201	85
102	201	80
100	202	65

```
SELECT student,
AVG(grade) AS a
FROM Enrol
GROUP BY student;
```

student	a
200	55.0
201	80.0
202	65.0

Grouping rules

In a GROUP BY query, one row appears for each block. So every column must be one of:

1. A statistic – computed once per block.
2. A column in the GROUP BY clause – these are automatically unique per block.
3. A constant – these are repeated for each block.

Syntax error

```
SELECT student, unit, MAX(grade)
FROM Enrol GROUP BY student;
```

Enrol

unit	student	grade
100	200	60
100	201	75
101	201	85

Syntax error

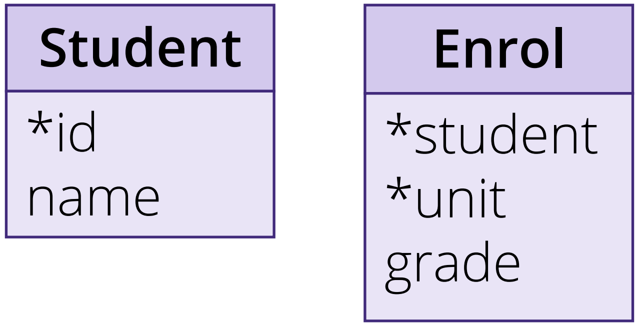
```sql
SELECT student, MAX(grade) FROM Enrol;
```

Enrol

unit	student	grade
100	200	60
100	201	75
101	201	85

Syntax error

```
SELECT Student.id, Student.name, AVG(grade)
FROM Enrol INNER JOIN Student
ON Student.id = Enrol.student
GROUP BY Student.id
```



HAVING

Syntax error

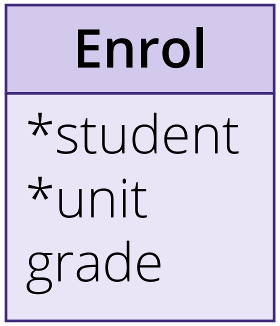
```
SELECT student, AVG(grade) AS average
FROM Enrol GROUP BY student
WHERE average > 50.0;
```

Enrol
*student
*unit
grade

HAVING

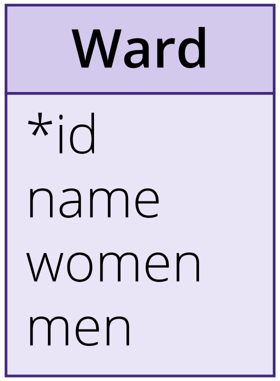
```
SELECT student, AVG(grade) AS average
FROM Enrol GROUP BY student
HAVING average > 50.0;
```

HAVING is a second WHERE that is evaluated after statistics and aliases.



HAVING

```sql
SELECT name,
women/(women + men) AS ratio,
FROM Ward ...
HAVING ratio > 0.5;
```



Query order

```
SELECT <columns>
FROM <tables>
<type> JOIN <joins>
WHERE <conditions>
GROUP BY <groups>
HAVING <conditions>
ORDER BY <orders>;
```

Evaluation order

1. Load FROM tables
2. Process JOINs
3. Filter rows with WHERE
4. GROUP BY and aggregate
5. apply aliases (SELECT)
6. Filter again with HAVING
7. Sort with ORDER BY
8. Filter columns with SELECT