```
-- 1 count(1)

select * from film;

select count(*) from film;

select count(1) as length from film;

select 1;

select 1 from film;

select 'hello' from film;

select count('hello') from film;




-- Group by

-- number of films for each year
select count(*)
from film
group by release_year;

select release_year, count(*) number_of_movies
from film
group by release_year;


-- for every year, for every type of rating, how many movies are there?
select release_year, rating, count(1) number_of_movies
from film
group by release_year, rating;


-- for every year, for every type of rating, how many movies are there with rental rate > 4?
select release_year, rating, count(1) number_of_movies
from film
where rental_rate > 4
group by release_year, rating;
```

```
-- for every year, for every type of rating, how many movies have count > 200?
select release_year, rating, count(1) as number_of_movies
from film
group by release_year, rating
having number_of_movies > 200;


-- HAVING
select release_year, rating, count(1)
from film
group by release_year, rating
having avg(rental_rate) > 3 and avg(release_year)>2006;


select * from film;
select avg(release_year) from film;


select release_year, rating, count(1)
from film
group by release_year, rating
having avg(rental_rate) > 3 and avg(length)>60;

select release_year, rating, count(1), avg(length)
from film
group by release_year, rating
having avg(rental_rate) > 3 and avg(length)>60;


-- code
-- after JOIN and WHERE
-- MySQL will initialise following maps
-- map<(release_year, rating), int> countMap;
-- map<(release_year, rating), double> averageRentalRateMap;
-- map<(release_year, rating), double> averageLengthMap;


-- GROUP BY phase starts
-- for each row of the resultant table.
-- populate the 3 maps.
-- GROUP BY phase finishes.alter

-- for each group in countMap:
```

```
-- if averageRentalRateMap.get(group) > 3 AND averageLengthMap.get(group) > 60
-- print group and countMap.get(group)


drop table students;

create table students(
	name VARCHAR(50)
);

insert into students
VALUES
('naman'),
('saharsh'),
('sahil'),
('deepak');

select * from students;

select max(name) from students;
```