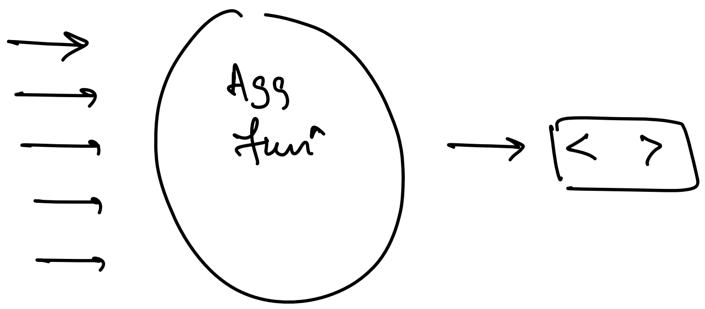
⇒ Aggregation

↳ Combining multiple thing together to achieve some result.

$$\underbrace{\text{SUM}(-) \mid \text{COUNT}(-) \mid \text{AVG} \mid \text{MAX} \mid \text{MIN} \cdots}_{\text{Aggregate functions.}}$$



Students

id	name	- — .	psp
1	A	—	70
2	B	—	80
3	C	—	90
4	D	—	60
5	E	—	50
6	F	—	75

→ Count # of students.

Select count (*) from students; ⇒ 6

Q:

Select count(*), name from students; ✗

↑
Aggregate fun^n.

Invalid Query

Select MAX(psp) from students;
MIN
AVG
SUM

Q: find the total amount spent on rentals by the customers with last name as Smith.

Payments

Customers

GROUP BY

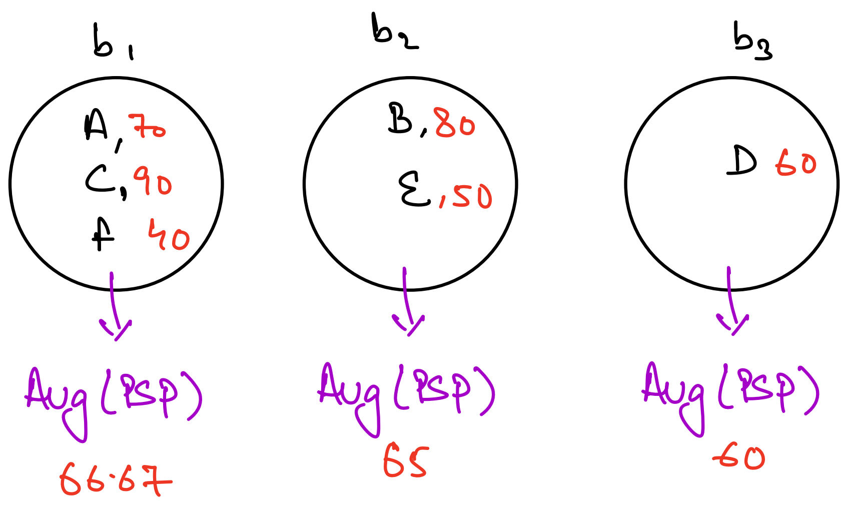
Students

id	name	batchid	psp
1	A	b1	70
2	B	b2	80
3	C	b1	90
4	D	b3	60
5	E	b2	50
6	F	b1	40

Q1: Find the Avg psp of all the students.

Select Avg (psp) from students ; ✓

Q2: Find the Avg psp of every batch



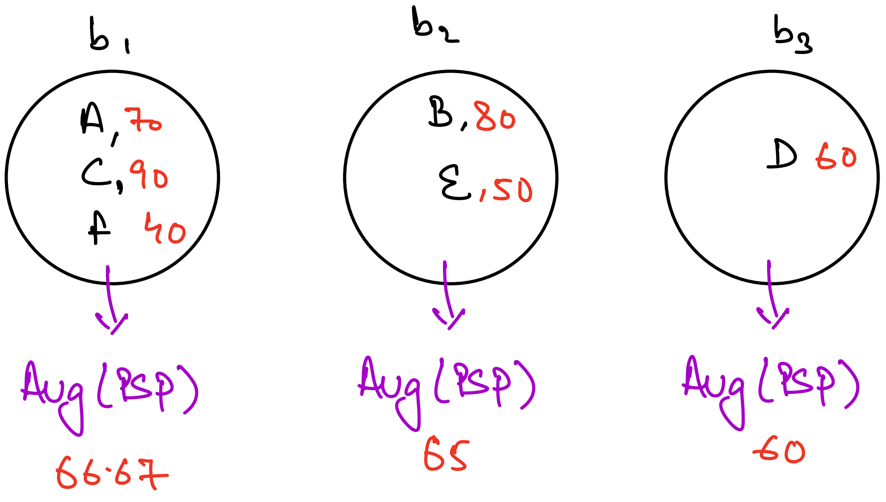
```
Select AVG (psp)          66.7
from students             65
GROUP BY batchid          60
```

```
Select AVG(psp) , batch_id
from students
GROUP BY batch_id
```

66.7	b_1
65	b_2
60	b_3

```
Select AVG(psp) , name
from students
GROUP BY batch_id
```

X



# Filter out Groups based on some condition.

X

```
Select AVG(psp) as avg_psp , batch_id
from students
GROUP BY batch_id
WHERE avg_psp > = 80
```

HAVING ⇒ Used to filter out groups.

```
Select AVG(psp) as avg-psp , batch-id
from students
GROUP BY batch-id
HAVING avg-psp > 80
```

The HAVING Clause enables you to specify conditions that filter which group results appear in the results. The WHERE clause places conditions on the selected columns, whereas the HAVING clause places conditions on groups created by the GROUP BY clause.

```
Select *
from customer c
Join rental r
```

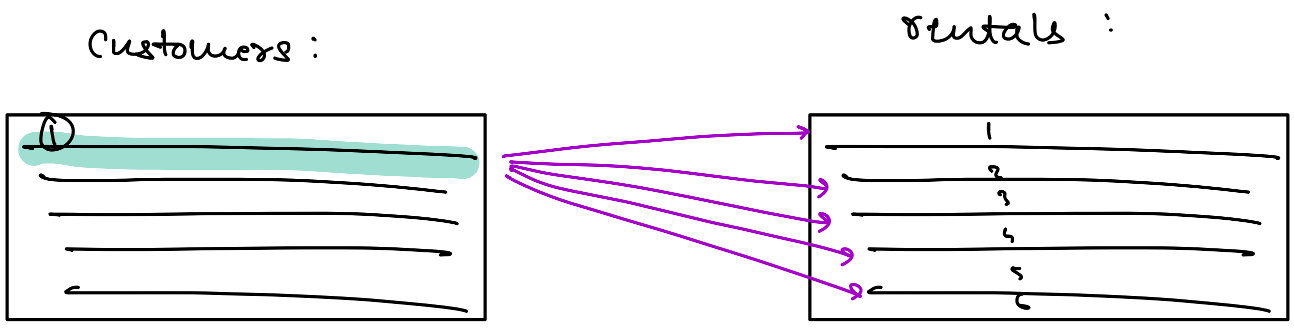
} Cross Join

```
Select *
from customer c
Join rental r
ON c.customer-id = r.customer-id
```

ON → Where ✓

ON vs WHERE.

```
Select *
from customer c
Join rental r
ON c.customer_id = r.customer_id
```



```
ans = [ ]

for row1 in customers :
    for row2 in rentals :
        if (row1.customer_id == row2.customer_id):
            ans.add(row1 + row2)
```

// ans will contain the filtered rows.

```
for row in ans :
    print (row)
```

```
Select  *
from    customer  c
Join    rental    r
```
$\rightarrow$ Cross Join

```
WHERE  c.customer_id  =  r.customer_id
```

```
ans = [ ]

for row1 in customers:
    for row2 in rentals:
        ans.add(row1 + row2)


for row in ans:
    if(c.customer_id == r.customer_id)
        print(row)
```

$\Rightarrow$ Always prefer ON condition rather than WHERE clause.

```
-- Find the total amount spent on rentals by the customers with last name as Smith.
use sakila;
select * from customer;

select * from payment;

select * from rental;

-- Join on customer Id.

select c.last_name as last_name, SUM(p.amount) as total_amount
from customer c
join payment p
on c.customer_id = p.customer_id
where c.last_name = 'SMITH';

-- Q: Find the top 5 customers who have rented the most number of films;

-- Step-1: Find the total no. of films rented by each customer; -- Two Tables (customer & rental)
-- Step-2: Arrange them in descending order.
-- Step-3: Get top-5

select count(r.rental_id) as rental_count, c.customer_id, c.last_name
from customer c
join rental r
on c.customer_id = r.customer_id
GROUP BY c.customer_id
ORDER BY rental_count DESC LIMIT 5;

select * from customer;


create table students(
        student_id INT PRIMARY KEY,
    student_name varchar(30) NOT NULL,
    batch_id INT,
    psp float
);

INSERT INTO students(student_id, student_name, batch_id, psp) VALUES
(101, 'Anjum', 1, 90.0),
(102, 'Tamanna', 4, 89.1),
(105, 'Fatima', 5, 76.5),
(109, 'Manish', 3, 91.4),
(110, 'Deepak', 1, 75.8),
(111, 'Chetan', 2, 55.8),
(112, 'Aditya', 3, 85.8),
(113, 'Pratyush', 4, 65.8),
(114, 'Mukta', 1, 45.8);

select * from students;



select avg(psp) as avg_psp, batch_id, name
from students
group by batch_id;

-- Error Code: 1054. Unknown column 'name' in 'field list'



-- Q: List the customers who have done atleast 30 rentals.
-- display the customer id and the number of rentals they have done.

-- Find the total number of films rented by each customer.

select customer_id, count(rental_id) as rental_count
from rental
group by customer_id
HAVING rental_count >= 30;

-- Q: List the customers who have done atleast 30 rentals.
-- display the customer id, customer last name and the number of rentals they have done.
select c.customer_id, c.last_name, count(r.rental_id) as rental_count
from customer c
join rental r
on c.customer_id = r.customer_id
group by c.customer_id
having rental_count >= 30;
```