Joins - II

Good morning! We start at 7:05 AM

Agenda.

- Joining multiple tables
- Compound joins
- Types of joins
  - Inner joins
  - outer joins
    - left
    - right
    - full
- cross joins
- using
- natural joins
- implicit joins
- Joins with where vs joins with ON.

Joining Multiple tables.

→ Till now we have done joins / combined data only from two tables.

students

id	name	instructor_id	batch_id

batches

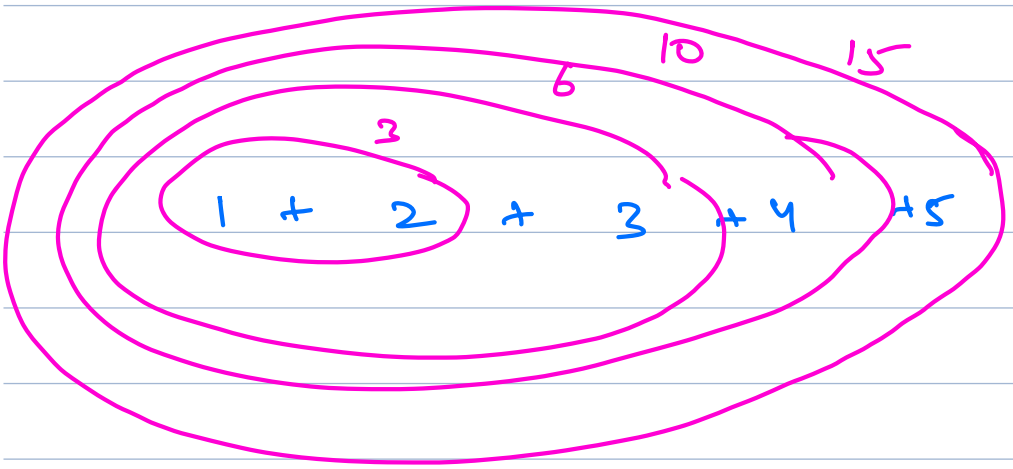
id	name

instructors

id	name

Q for every student, give their name along with name of their instructor who is teaching them and name of their batch.

Shreyas	Aug, 22	Ujjwal



* You can also join multiple tables, very very similar to how you add multiple numbers

↓

Join pair by pair

Can't join instructor + batches as no common/relating column.

(students + instructors) + batches

```
select s.name, i.name, b.name
from students s
join instructors i
 on s.instructor_id = i.id
join batches b
 on (s.)batches_id = (b)id
```

Intermediary table (ans)

students				instructors	
id	name	instructor_id	batch_id	id	name

students				instructors		batches	
id	name	inst_id	batch_id	id	name	id	name

Code

table1 :

table2 :

table3 :

ans1 :

ans :

```
for row1 in table1:
    for row2 in table2:
        if cond" is true :        //on clause
            ans1.add(row1 + row2)
```

```
for row1 in ans1:          → has data stitched for table1 + table2
    for row2 in table3:
        if cond^ is true:
            ans.add(row1 + row2)
```

```
for each row in ans:
    print(row[s.name] + row[i.name] + row[b.rent])
```

COMPOUND JOINS.

Q film→ for every film, name all the films that are released within ± 2 years of the film AND their rental rate > rate of the movie.

films

name	rel. year	rating
Sholay	2000	2
KKHH	1999	3
MMN	1998	2
K3G	1997	4

Sholay	KKHH
KKHH	K3G

MHN	KRHH
MHN	K3G

```
select
from film f1
join film f2
  on (f2.rel_year >= f1.rel_year-2) and
     (f2.rel_year <= f1.rel_year+2) and
      f2.rating > f1.rating
```

on f2.rel between f1.rel_year-2 and
f1.rel_year+2

Learning

① Joins always need not happen on equality of columns

② Joins can also have multiple conditions
- Compound join
- on different columns

Types of joins

Students

id	name	buddy_id
1	A	2
2	B	null
3	C	2
4	D	1

Q for every student print their name & name of their buddy

```
select
from student s
join student b
on s.id = b.buddy_id
```

★ self join can be thought of joins b/w two tables of the same reference.

Students (s)

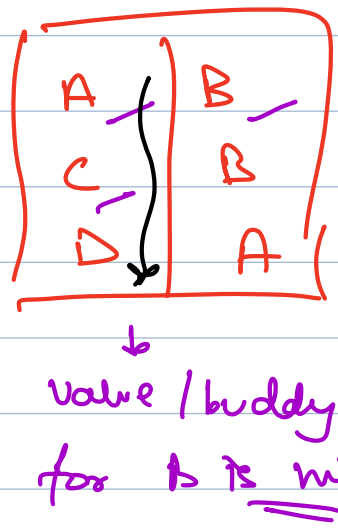
id	name	buddy_id
1	A	2
2	B	null
3	C	2
4	D	1

✓ x x x
x ✓ x ✓
x x x x
x x x x

Students (b)

id	name	buddy_id
1	A	2
2	B	null
3	C	2
4	D	1

1	A
2	B
3	C
2	B
4	D
1	A



when we do a join b/w two tables

from L

join R

if a row of left table doesn't match with any row of right table, it will not be in my answer set - and vice versa

Students

id	name	b_id
1	John	1
2	Jane	2
3	Jim	null
4	Jenny	null
5	Jack	2

batches

id	name
1	A
2	B
3	C

Q for every student print their name along with batch's name

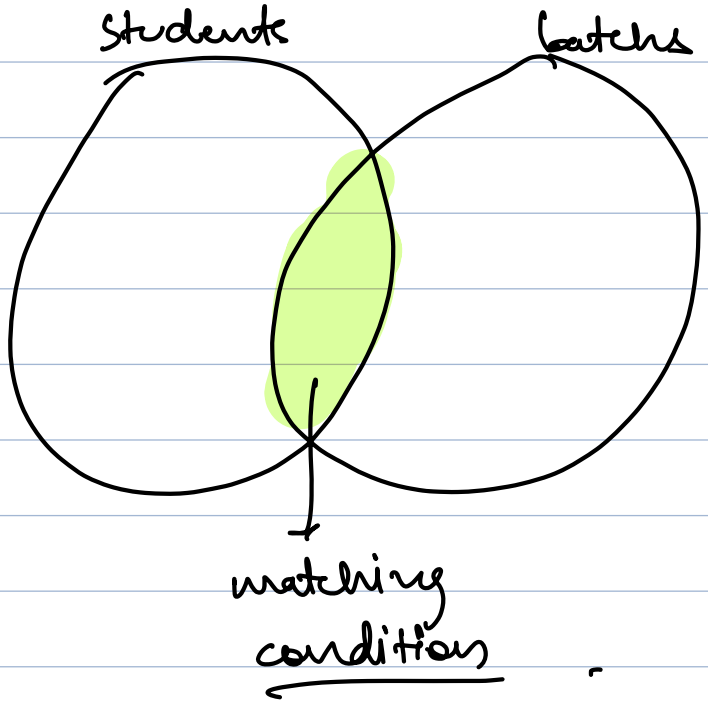
```
select
  from students s
inner join batches b
  on s.b_id = b.id
```

John	A
Jane	A
Jack	B

inner join

↓

Join that includes only rows of both tables that match condition.



Q Get all the students - - - -, even if no batch -

```
select *
from students s
join batches b
on s.b_id = b.id
or s.b_id is NULL
```

students

id	name	b_id
1	John	1
2	Jane	2
3	Jim	null
4	Jenny	null
5	Jack	2

batches

id	name
1	A
2	B
3	C

1	John	1	A
2	Jane	2	B
3	Jim	null	null
3	Jim	null	null
3	Jim	null	null
	Jenny		
	"		
	"		
5	Jack	2	B

08:30 AM.

Outer joins.

→ also allows rows that don't match the condition in the final answer

Types

→ left join
or
left outer join

→ right join
or
right outer join

→ full join
or
full outer join

Left join → will include all the rows that match condition
+

includes all rows of the left table that dont match condition ever.

→ for these rows, we will fill null on other side

students

id	name	b_id
1	John	1
2	Jane	2
3	Jim	null
4	Jenny	null
5	Jack	2

batches

id	name
1	A
2	B
3	C

```
select
from students s
left join batches b
on s.b_id = b.id
```

John	A
Jane	B
Jack	B
Jim	null
Jenny	null

* left join ensures every row of the left side table is atleast once in the answer.

Code

```
for each row1 in table 1 :
    for each row2 in table 2 :
        if match cond :
            ans.add ( row1 + row2 )
```

```
for each row in table1:
    if row not in ans:
        ans.add(row + [null, null...]);

for each row in ans:
    print(  ____   _  )
```

Right join.

→ includes all rows that match the condition

+

→ all rows of right side that didn't match the condition.

students

id	name	b_id
1	John	1
→ 2	Jane	2
3	Jim	null
4	Jenny	null
5	Jack	2

batches

id	name
1	A
2	B
3	C

John	A
Jane	B
Jack	B
null	C

Full join.

→ all rows that match the condition

+

→ all left table rows that never matched the condition

→ all right table rows that never matched condition.

students

id	name	b_id
1	John	1
2	Jane	2
3	Jim	null
4	Jenny	null
5	Jack	2

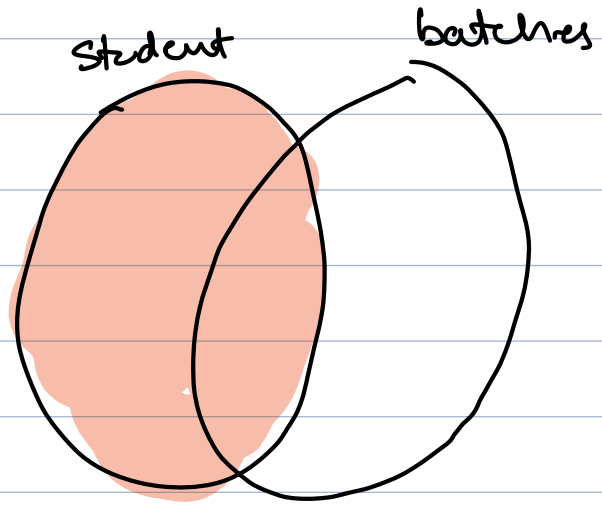
batches

id	name
1	A
2	B
3	C

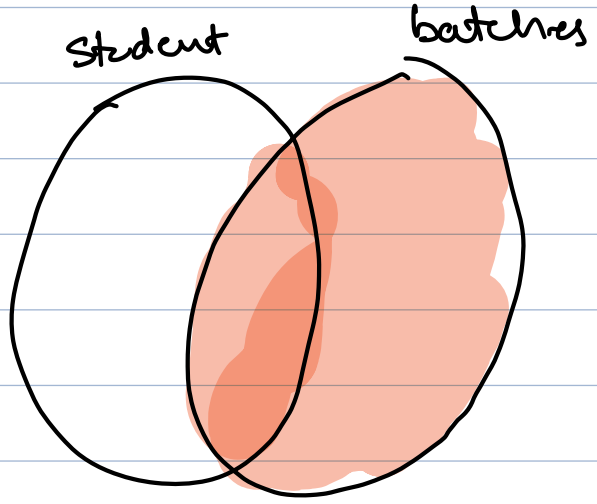
```
select *
from student, batch
where student.b_id
    = batches.bid
```

John	A
Jane	B
Jack	B
Jim	null
Jenny	null
null	C

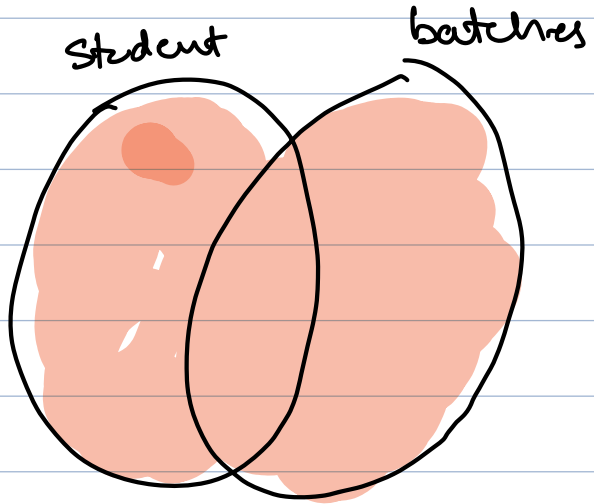
Venn Diagram.



left join

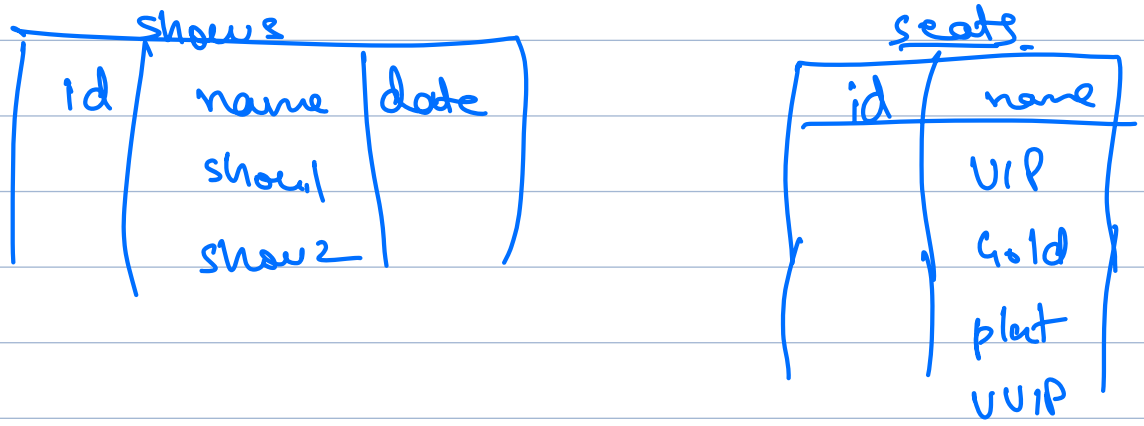


right join



full join

Cross Join



Q Print every show with all the seat types

show1	VIP
show1	Gold
show1	plat
show1	VVIP
show2	VIP
show	—

```
Select
from show
join seats
on true
```

Cross join → returns pairs of all the mapping

select
from show
Cross join seats

9 show
4 seat
36 rows

Q Is cross join same as the full join?

NO

Cross join
A * B

full join
A + B

USING.

{ ① Joining is based on equality
② Name of coln on both sides was same

eg. on students.batch_id = batches.batch_id

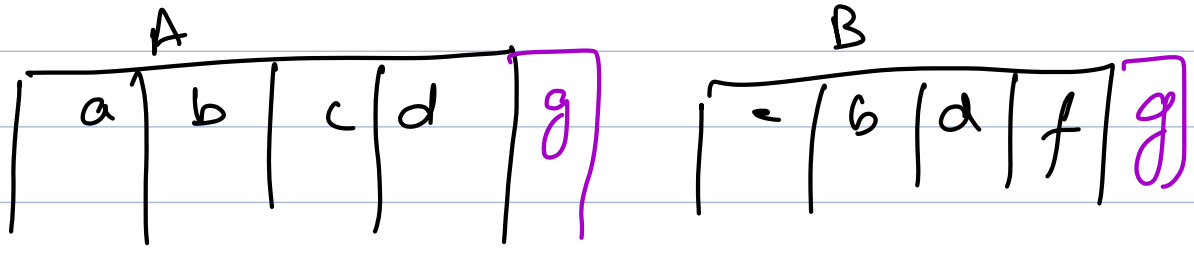
select
from student
join batches
using (batch_id)

using(batch_id, name)

on student.batch_id = batches.batch_id AND
student.name = batches.name

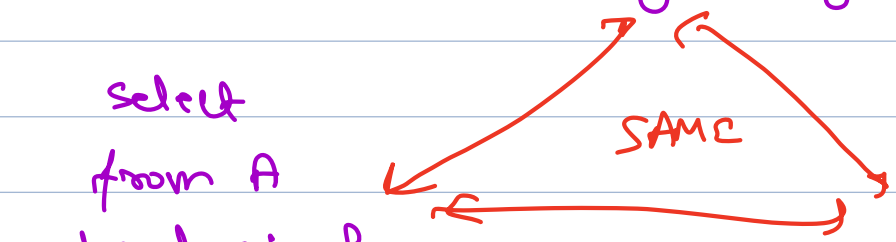
Natural joins.

→ when you want to join two tables based on equality of all columns with same name.




```
select
from A
join B
  on A.b = B.b
  and A.d = B.d
  and A.g = B.g
```

Assuming b, d, g are only columns with same name in two tables.




```
select
from A
natural join B
```

```
select
from A
join B
using( b,d,g)
```

Implicit Join → what if I join without using join keyword

```
select *
from A,B
```

→

```
select *
from students, batches
where students.b_id = batches.id
```

↓

cross join

★ Implicit join is a cross join.

Code — normal join.

```
select *
from A
join B
on A.a = B.a;
```

A = [ ] → 100

B = [ ] → 10

ans = [ ] → filtered data.

```
for every row1 in A :
    for every row2 in B:
        if match cond:
            ans.add(row1 + row2)
```

1000 ✓

```
for every row in ans:
    print(ans[a] + ans[b])
```

```
select *
from A, B
where A.a = B.a;
```

A = [ ]

B = ( )

ans = [ ]

```
select *
from student
where id = 2;
```

```
for row1 in A:
    for row2 in B:
        ans.add(row1 + row2);
```

for every row in ans:
    if row matches cond: // where.
        print(row[a], row[b]);