class starts at : 7.08AM

Agenda:

- outer joins
  - left join
  - right join
  - full join
- using clause
- natural join
- cross join.
- Aggregates.

students

id	name	bid
1	sumit	1
2	karthik	1
3	sonali	null
4	Ayush	2

batches

id	name
1	A1
2	A2
3	A3

→ print the name of every student with the name of their batch

↓ inner join

```
select s.name, b.name
from students s
join batches b
```

On s.bid = b.bid

o/p

s.name	b.name
sumit	A1
Harshit	A1
Ayush	A2
Aashish	null

new student in

If a student does not belong to any batch, print the name of students with batch name as null

Will inner join solve this?

NO

for every row in A (r_A)

for every row in B (r_B)

if r_A and r_B are matching conditions —

r_A and r_B should be part of my final o/p.

n^2

inner join

outer joins

→ In case of a row does not match / satisfy any condition, use outer join concept

Outer joins

- Left joins
- Right joins
- full joins

Left joins

students

id	name	branch
1	Kiran	1
2	Abhay	1
3	Kartheek	null
4	Suresh	2
5	Bharath	5

branches

id	name
1	Dec22
2	feb22
3	Tuesday Morning

left join

→ Returns the values from left table

condition satisfy → right table values

condition not satisfy → null values.

s.name	b.name
Kiran	Dec22

Abhay	Dec22
Karthik	null
Dhanee'	feb22
bharath	null

Inner join → rows are being matched on a given condition.

② Right join → Outer join = Inner join + where rows are not matching

student

id	name	bid
1	A	1
2	B	2
3	C	null

batches

id	name
1	A1

inner join

id	name	bid	id	name
	A		A1	

only when condition matches

left join

A A1

B null

students

id	name	bid
1	Bhargav	1
2	Ayush	2
3	Dhanya	null
4	Poornima	3
5	Dhanee'	1

batches

id	name
1	A1
2	A2
3	A3
6	null

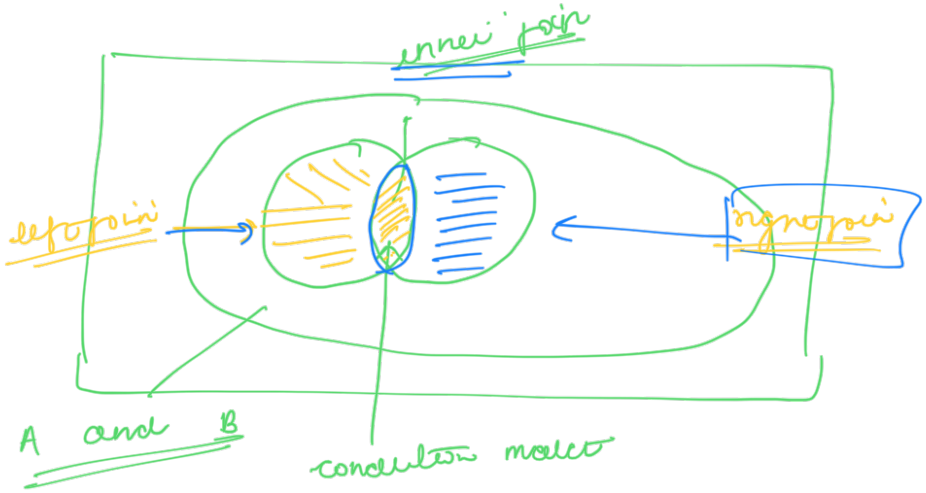
bk

right join

a. name	b.name
Bhargva	A1
Sushil	A1
Ayush	A2
null	A3

null null

Inner join outer join



Inner join C outer join

③ full join {

union of left and right join

students

id	name	bid
1	Ayush	1
2	Dhanya	2
3	Anikant	1
4	Shanu	6

batches

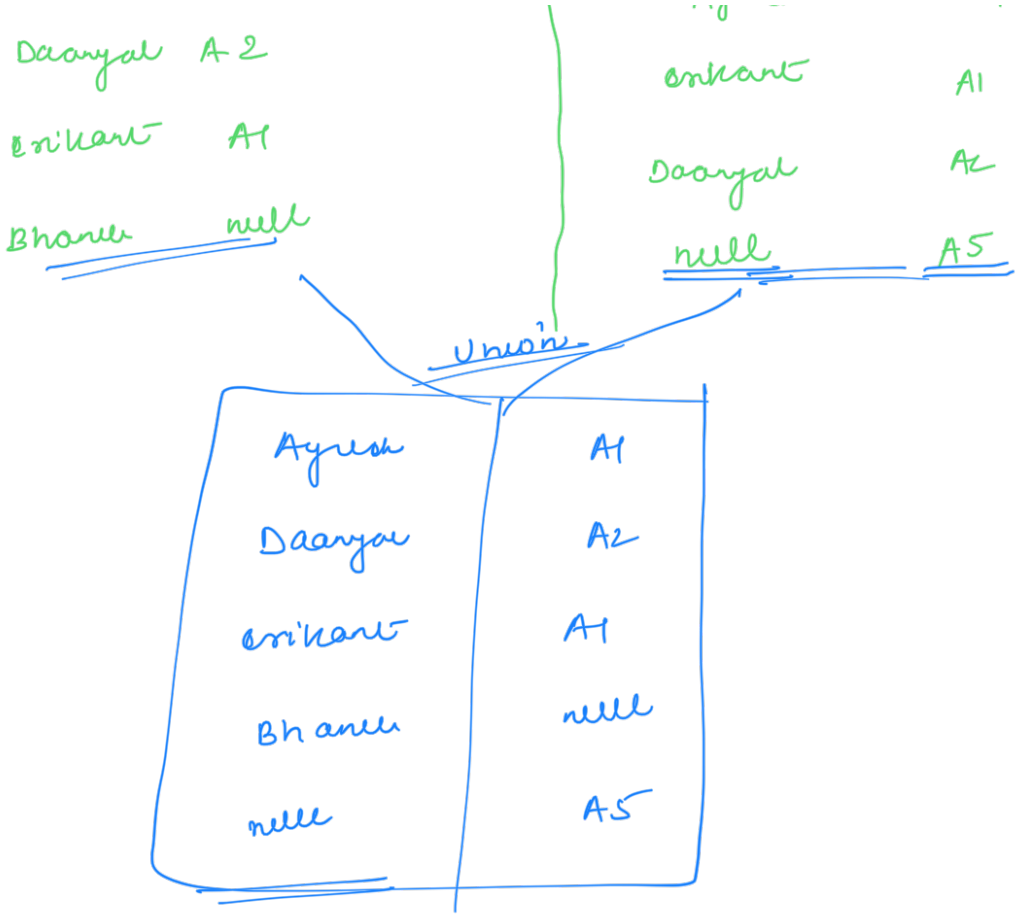
id	name
1	A1
2	A2
5	A5

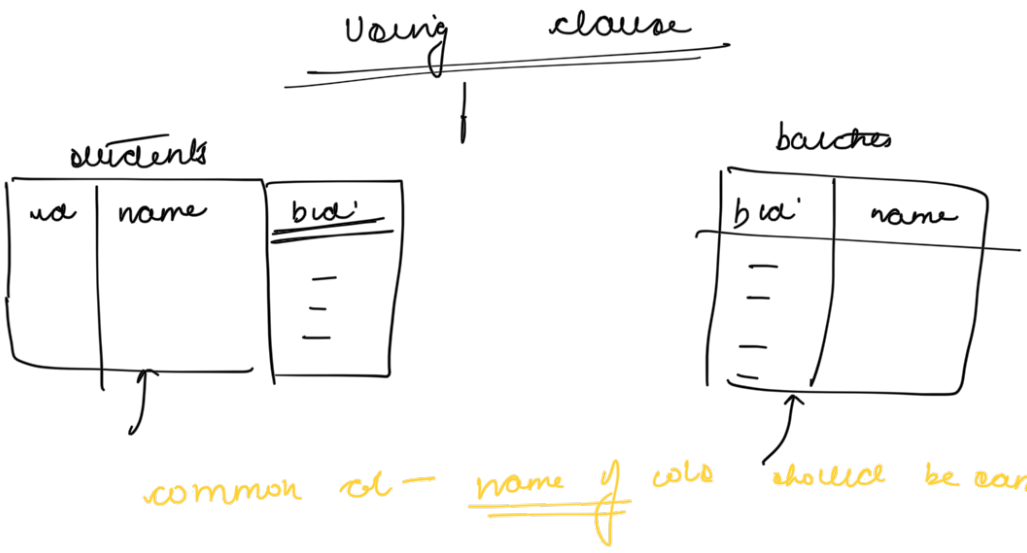
Left join

Ayush A1

Right join

Ayush A1





common col — name of cols should be same

```
select  *
from    students  a
join    batches   b
on      a.bid  =  b.bid
```

students — batch_id

batches — id

```
select *
from student a
join batches b
using (b id);
```

→ using clause simplifies/reduces the efforts of a developer

Natural join

```
On student.b id = batches.id
        ↓ simplify
using (b id)
        ↓ natural join
natural join
```

students

id	name	b id
1	Bharath	1
2	Ankam	1
3	sumit	null
4	ekash	2

batches

b id	name
1	A1
2	A2
3	A3

```
select *
from students a
```

```
select *
from students a
```

join batches b

on s.bid = b.bid

using (bid)

natural join batches b

① what if col name are different?

② what if more than one col with same name?

→ empty

natural join
with duality

condition t1.col
t2.col

student
bid

batches
bid

→ col name is different

on.

s.bid = b.bid

using (bid);

a.bid = b.bid

empty rows

natural join

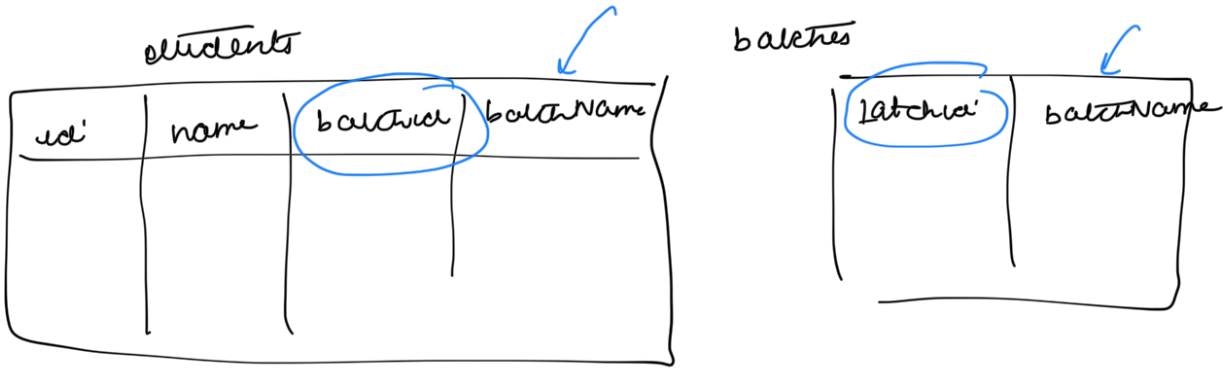
students
uid
bid
batches
bid
name

```
select *
from students
natural join batches;
```

⟹

```
select * from students s
join batches b
on s.id = b.id;
```

≡




```
select * from
students
natural join batches;
```

↓ implicitly.

```
select * from
students s
join batches b
on s.batch_id = b.batchId
&& s.batch_name = b.batch_name;
```

same

↓ using

same

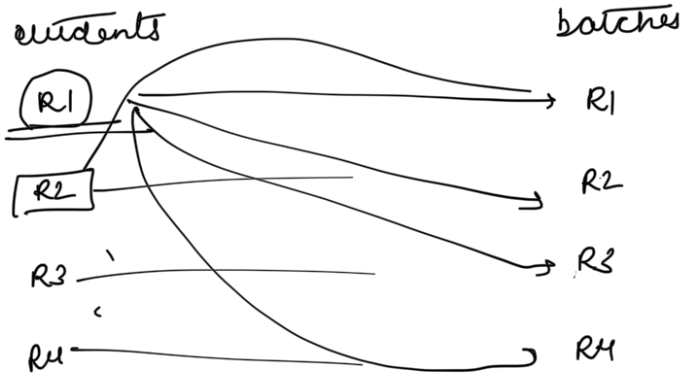
```
select * from
students s
join batches b
using (batch_id, batch_name);
```

Break till 8:48

- cross join
- Aggregates

cross join

↳ inner join with no conditions.



students	batches
A	D
B	e
C	

o/p

A	D
A	e
B	d
B	e
C	d
C	e

Table1 = m rows

Table2 = n rows

final o/p rows for cross join = m*n

Cricket NC (Group stage Matches)

	Group1	Group2
1	India	Australia
2	Bangladesh	England
3	South Africa	NZ
4	Ireland	Afganistan
5	Zimb	Sri Lanka
6		Pakistan

Cross join

India → A, E, NZ, Af

B

SA

Left outer join

- Retain left values
- condition do not match
  - Have null in right values

Right outer join

- Retain right values
- If cond" does not match
  - have null in left values

→

Aggregates

collection / grouping / combining / merging

Till now:

student

id	name	bid
1	Ayush	1
2	Anant	1
3	Krishna	2

```
select * from students
where bid = 1;
```

multiple rows

going through → output → multiple rows

multiple rows

students

id	name	bdp	ecoler_cons
1	Sumit	80	500
2	Suresh	60	600
3	Pooja	70	100
4	Bhavya	50	200

Q1 find max bdp of the students. (single row)

Q2 find average bdp (single row)

Q3 find total no of ecoler cons (single row)

(multi values) → (single value)

fun (set of values (or) multiple rows) → relations → (single value (or) single row)

Various types of sql aggregate

- max()
- min()
- sum()
- avg()
- count()

students

id	name	pcp	bid
1	Poornima	60	1
2	Karan	70	1
3	Prethu	80	2
4	Ayesha	50	3
5	Sumit	90	2

① max()

↳ return max value for any particular col

students → batches → inst

students: id, name, bid
batches: id, inst-id
inst: id, name

Print the instructor for every student?

```
select s.name, i.name
from students s
```

multiple joins

```
join battery b
on b.bid = b.id
join inetmeters i
  on i.id = b.inet_id
```