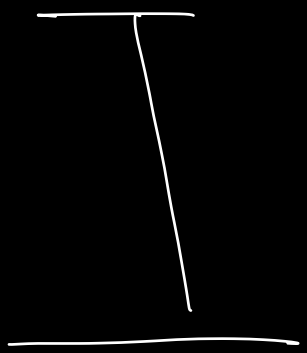
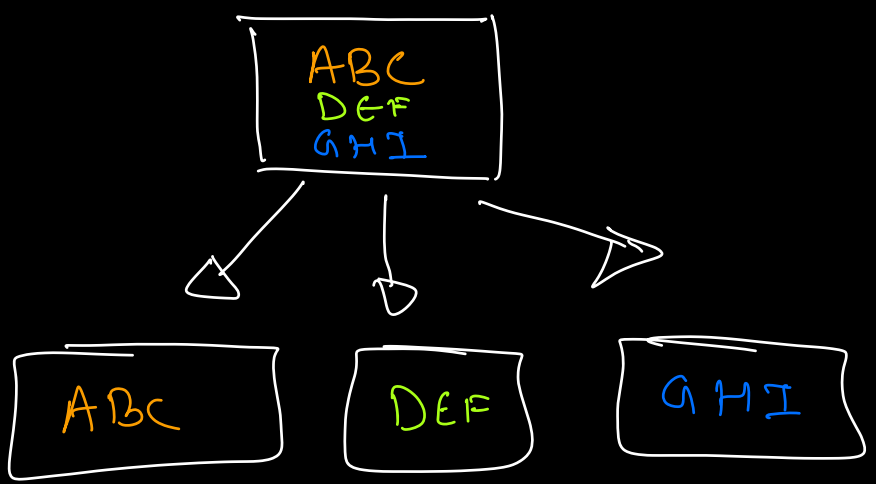
- JOINS
- Solve 2-3 Questions
- Explain



Main Aim of RDBMS = reduce redundancy

- How did we reduce redundancy ??
  - Breaking table



What became complex ??

- Querying.

JOINS

⇒ CROSS JOIN ≃ Cartesian Product

n

A	B	C
P_1	P_2	P_3
Q_1	Q_2	Q_3

T1

m

D	E	F
A_1	A_2	A_3
B_1	B_2	B_3

T2

⇒

A	B	C	D	E	F
P_1	P_2	P_3	A_1	A_2	A_3
P_1	P_2	P_3	B_1	B_2	B_3
Q_1	Q_2	Q_3	A_1	A_2	A_3
Q_1	Q_2	Q_3	B_1	B_2	B_3

⇒ $n \times m$

CEO = Kailash }
Emp = Yash }

owner_id	name
3	A
4	B

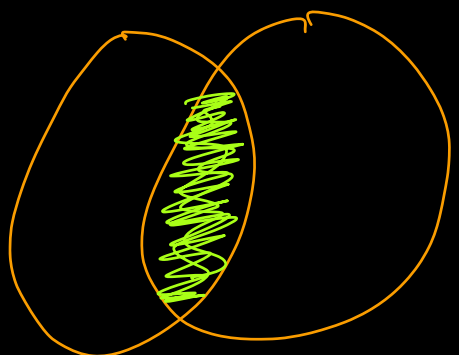
Table 1

C_id	company	owner_id
1	tesla	3
2	audi	4

Table2

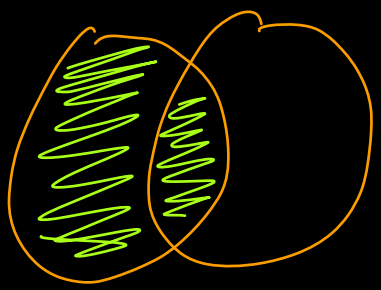
Condition on : owner_id

T1.owner_id	T1.name	T2.C_id	T2.company
3	A	1	tesla
4	B	2	audi



Inner Join

11 Left Join



c-id	name
1	Yash
2	Rahul

c-id	lastname
2	Gupta
3	Sharma

Condition = c-id

c-id	name	lastname
1	Yash	null
2	Rahul	Gupta

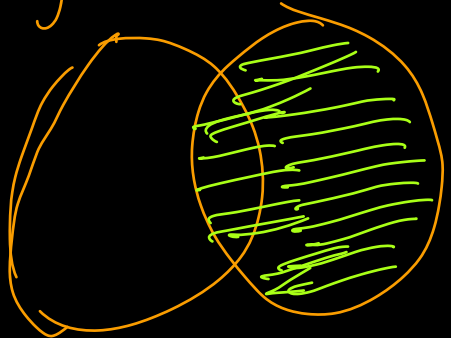
#

user_id	name	age
A	Yash	20
B	Rahul	21
C	Rohan	22
D	Ravi	23
E	Ramsa	24

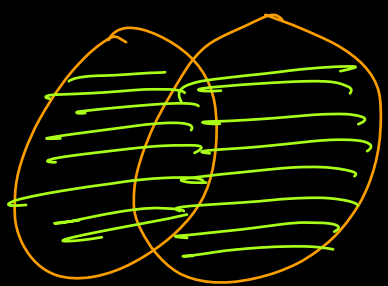
cus_id	user_id
20	A
22	C
24	D
25	E

Left join would work

Right Join



Outer Join



CROSS JOIN VS OUTER JOIN.

user.id	name
1	Yash
2	Rahul

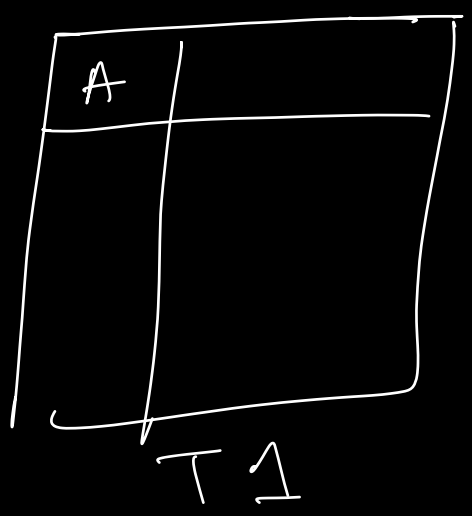
car.id	user.id
3	1
4	5

No of Rows in a cross join ⇒ 4

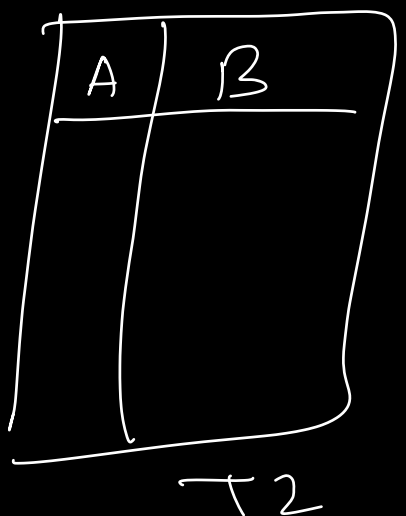
No of Rows in a outer join ⇒ 3

user.id	name	car.id
1	Yash	3
2	Rahul	null
5	null	4

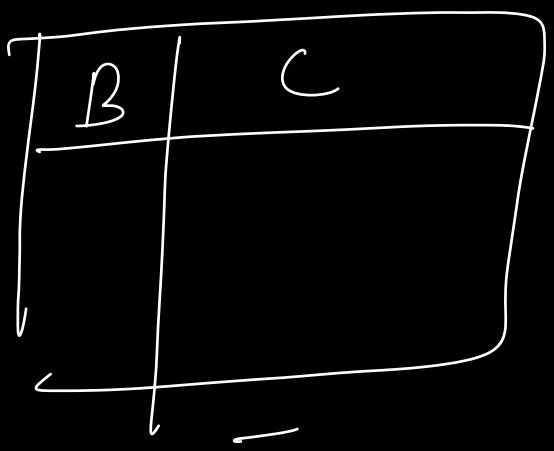
#



T1

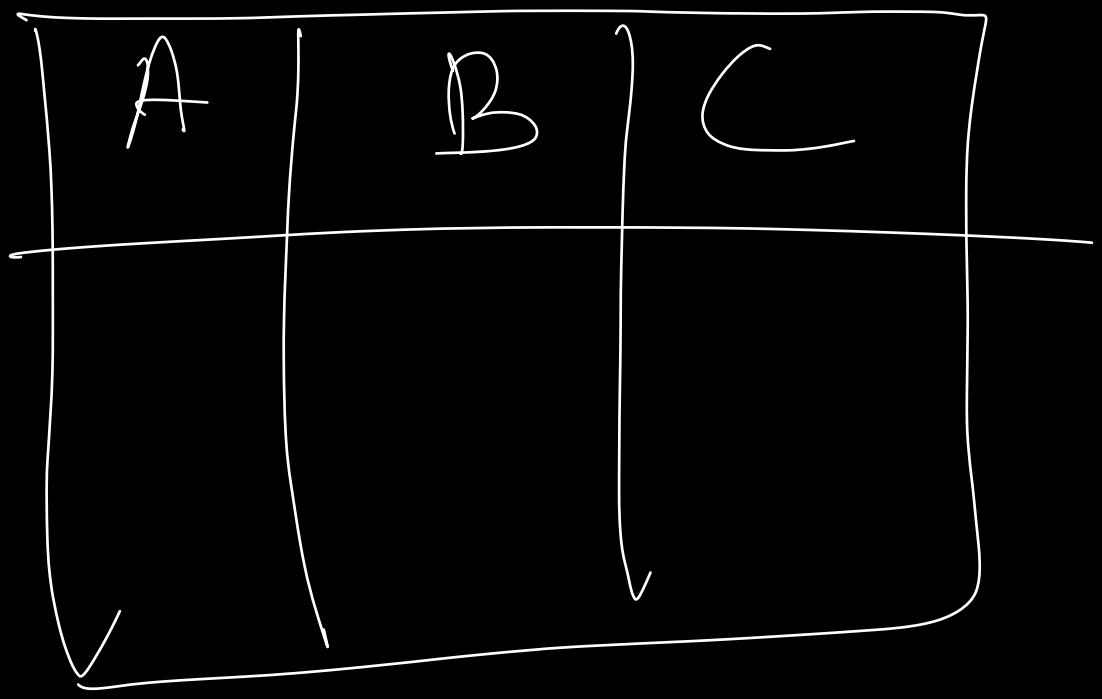


T2

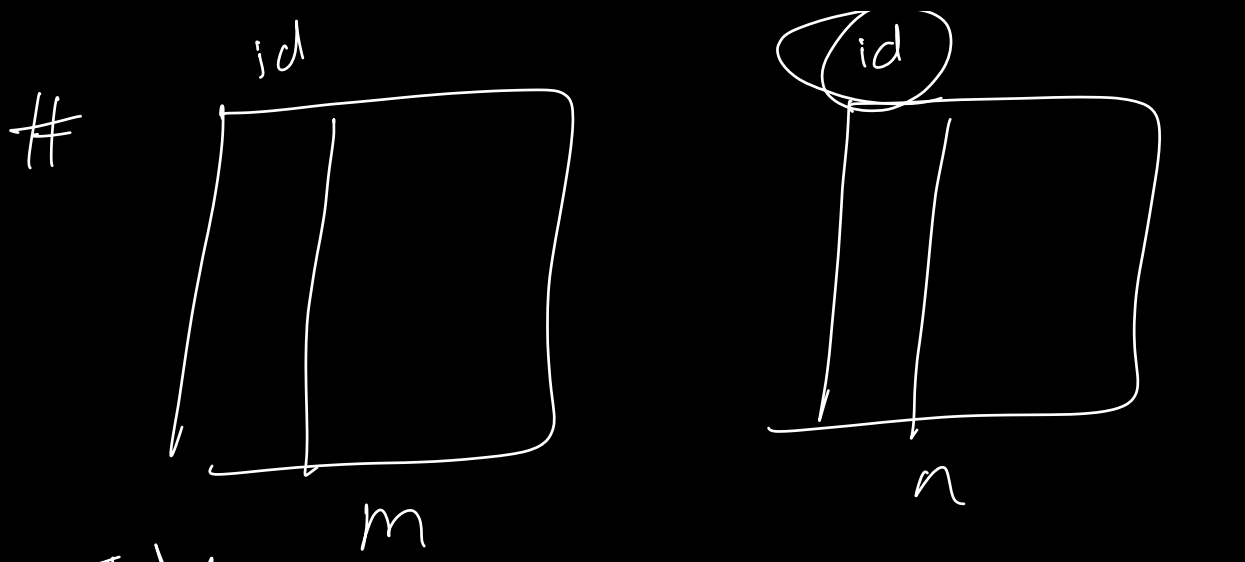


T3

T1 inner join T2
on T1.A = T2.A
inner join T3
on T2.B = T3.B



Explain Keywood ⇒ helps in optimising Query Performance



No Index

Time Complexity ⇒ m x n

With Index Time Complexity ⇒ m.logn