Schema Design 2.

Agenda.

1. Complete schema design - scaler
2. How to decide primary key of a mapping table
3. How to represent FK & indexes.
4. Schema Design of Netflix.

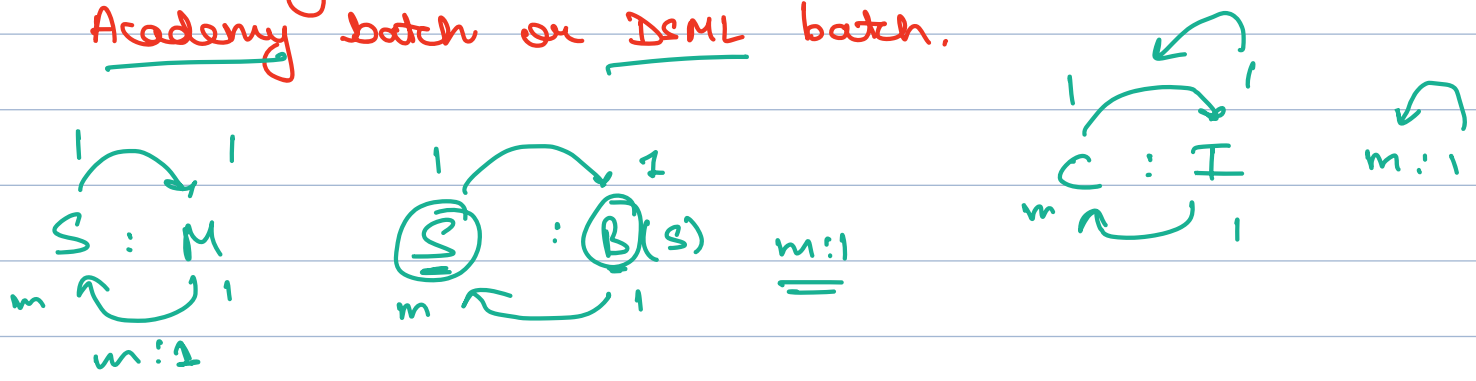
Requirements

1. Scaler will have multiple batches. About each batch we have to store their name, start month, current instructor.
2. Each batch of scaler will have multiple students.
3. Each batch has multiple classes.
4. For each class we have to store the name of the class, date and time of the class, instructor of the class.
5. For every student we store their names, grad year, university name, email, phone number.
6. Every student has a buddy who is also a student.
7. A student may move from one batch to another.
8. For each batch a student goes to we have to store the start date of that batch. [Rel with attributes]

(9) Every student also has a mentor. For every mentor we store their name and current company

(10) We have to store information about all mentor sessions (time, duration, student, mentor, student-rating)

(11) For every batch we have to store if its an Academy batch or DSML batch.



batches

batch_id	name	start_month	instructor_id	batch_type

instructor

instructor_id	name	email	avg_rating

students

student_id	name	grad_year	univ_name	email	phone_no	batch_id
buddy_id	mentor_id					

classes

class_id	name	scheduled_time	inst_id

mentors

mentor_id	name	current_company

mentor_sessions

mentor_session_id	time	duration	stud. start reg	st. id	mentor_id

batch_classes

batch_id	class_id

batch_type

id	name

Student_batches

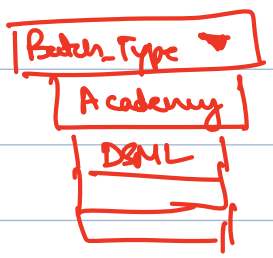
st_id	batch_id	st_date

(Academy | DSML)

Batch_Type → enums → have one of the given set of values

→ used to represent all types of a particular thing

→ one of a fixed set of values.




```
enum Gender {
    MALE,
    FEMALE
}
```

Academy → 0

~~Morning → 1~~

DSML → ~~2~~ 1

```
enum batch_type {
    ACADEMY, 0
    ~~MORNING~~ 1
    DSML 1 ~~2~~
}
```

How to represent enums.

(a) using strings

batches

id	name	———	batch_type
			"Academy"
			"Academy"
			"DSML"
			"DSML"
			"Academy"

Cons → space → each value takes a lot of space
→ slow string comparisons

Pros → Readability
→ No joins is required

(b) using integers

batches

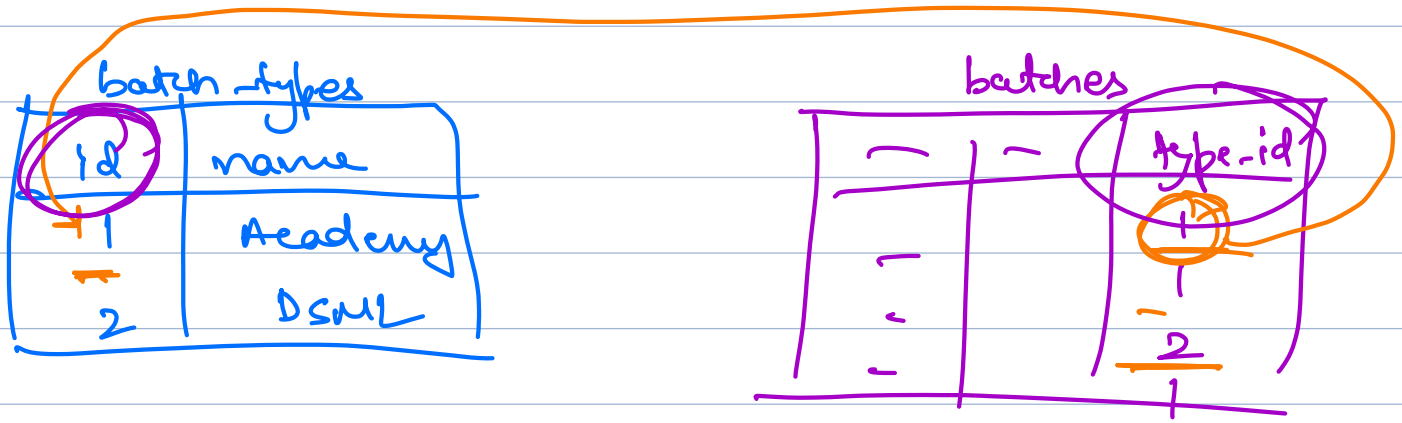
id	name	———	type
			0
			0
			1
			1
			0

Pros → less space
→ faster to search

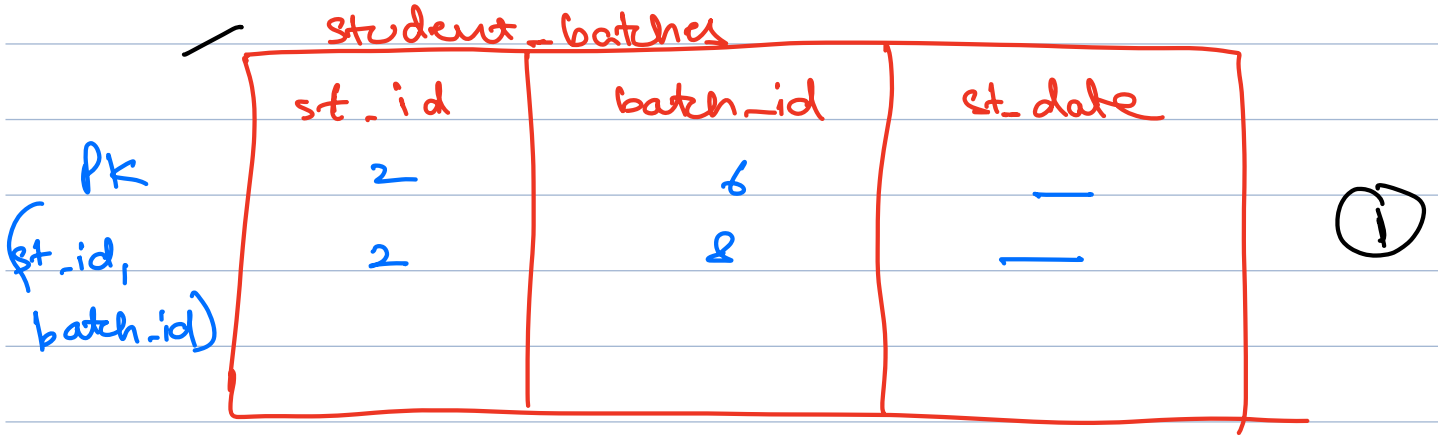
Cons → not readable
→ can't add values in between
→ can't delete values in between
→ what a particular value denotes is not present in DB.

(c) Lookup Table.

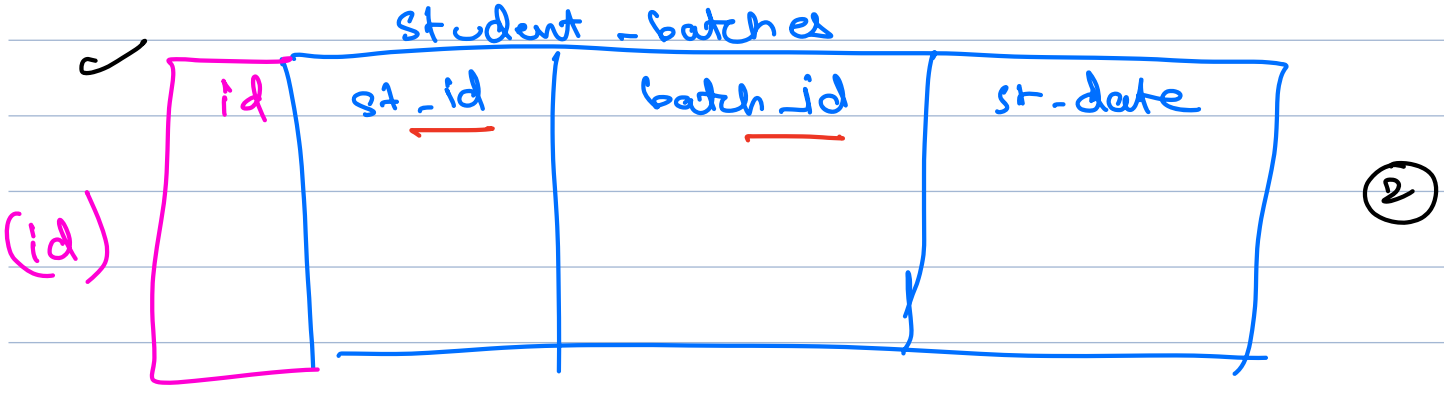
↳ separate table to store all the type.
↳ each type is stored as a separate row



How to decide primary key of a mapping table

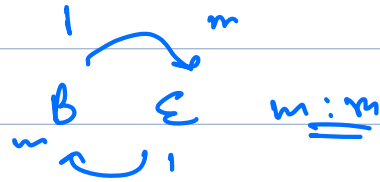


VS



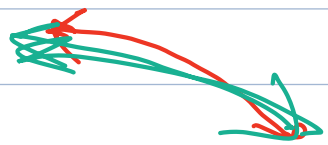
→ size of index will be bigger in (1)

1. One student can belong to multiple batches
2. Every batch has exams
3. Same exam may happen on diff batches on diff date.
4. If a student moves a batch, they may have to give some exams again.



student - batches

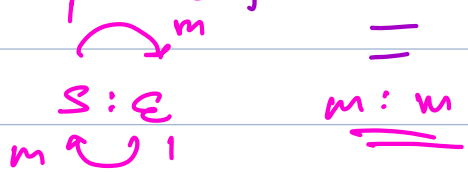
st_id	batch_id	st_date



batch - exams

id	batch_id	exam_id	date

Q which student has given exam for which batch.

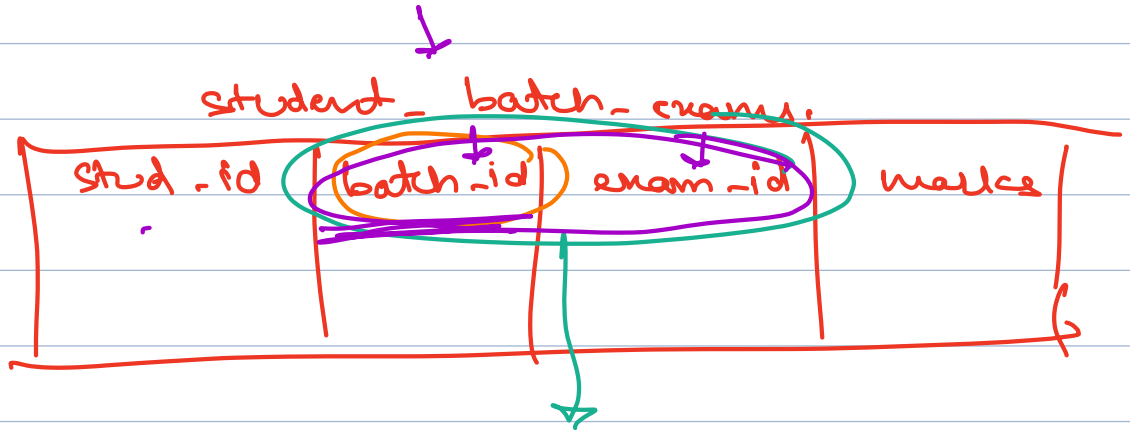


student - exams

stud_id	exam_id	marks
10	22	
10	22	

stud → 10
exam → sql → 22

one student can give same exam twice in diff batches.



batch_exam_id

⇒ Sometimes a mapping table may also have a relation with another entity

↓

in these cases not having a PK can cause problems → space problem

Advantages of sep key

If a rel" is being mapped to another entity → saves cost

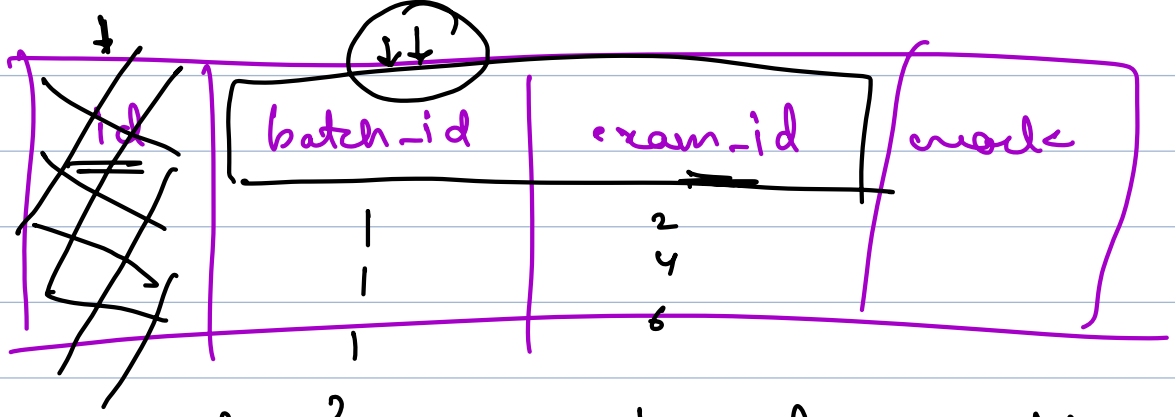
Advantage of no seperate key.

→ (batch_id, exam_id)

→ If no sep PK, queries on (batch_id) will be faster

↓

Bcoz the table will be sorted by batch_id.



get me all exams of batch_id = 2

2	2
2	3

→ Most of the times, not having a separate PK makes more sense, it will make those joins faster.

[Break] → 08:10 AM.

How to represent FK and indexes.

→ Representation doesn't matter
→ explanation matter.

→ Along with schema design, requirement, you also given use cases

↓

[Indexes are governed by usecases only.]

→ FK is mentioned alongside creating schema.

→ table

→ attributes

mention FK ⇒ → ref relations

→ we talk about indexes at the end.

[Depending on use case]