

Class will start at 9:05 PM

Agenda

→ CRUD

- Read

 - Between

 - Like

 - IS NULL

 - ORDER BY

 - LIMIT

- Update

- Delete

 - Delete vs Truncate vs Drop.

Between

select * from film

where release-year BETWEEN 2005 AND 2010;

both are inclusive,

≥ 2005 and ≤ 2010 .

batches.

id	name
1	Acad-July-23
2	DSML-Jan-22
3	Acad-June-22
4	July-Acad-22

acad →

LIKE

```
select * from batches
where name like '%' acad '%';
```

```
select * from films
where title like '% love %';
```

2 wildcards: →

- % (percentage) → Any no of characters (≥ 0)
- (underscore) → Exactly 1 character.

① cat% → Match cat caterpillar catherine.

~~bobcat~~

② %cat → cat bobcat

~~dog~~

③ %cat% → cat, bobcat
[0-] [0-N]

~~boom~~ 3 0

④ -at → cat bat wildcat
g | g |
exactly at at end
1 char

⑤ c_t → cat cot ~~cut~~ cit
~~bobcat~~ cut

⑥ _ _ _ cat → bobcat tomcat popcat

⑦

$\text{cat \%} \rightarrow \text{cat}, \text{bobcat}, \text{beat}$
 bcaterpillar

\downarrow
 son \% X

\%son ✓

son

a) \%.123 \% \rightarrow $\boxed{12123}$

c) --123-- ✓

batches.

id	name	
1	A	← X
2	B	← ✓
3	C	← ✓
4	NULL	← X

$\text{select * from batches}$
 $\text{where name} \neq \text{A};$

ORDER BY

↳ show result in sorted order.

select * from film order by release-year;

↑
by default Ascending order.

select * from film order by release-year desc;

select * from film order by release-year, title;

→ first sort by release-year.

→ use title as tie breaker.

release year	title
2008	A.
2006	B
2008	C

2006	B
2008	A
2008	C

select title from film order by release-year;

Pseudo code with order by :-

ans = []

where [for each row in film :-
if (row matches condition in where)
ans.append(row);

order by [ans.sort(col in order by)

select [filtered-ans = []
for each row in ans :-
filtered-ans.append(row['rating'], row['release_year'])
return filtered-ans;

If the query is having a `DISTINCT` clause then in order by we can only use those col that are there in select.

X `select DISTINCT title from film
order by release-year;`



/ `select Distinct title from film
order by title;`

Break till 10:30 PM

Pseudo code with order by and limit and offset.

ans = []

for each row in film: →

if (row matches condition in where)
ans.append(row);

ans.sort (col in order by)


filtered ans = []

for each row in ans: →

filtered ans.append (row['rating'],

return filtered ans [start of limit, end of limit]

offset 20 limit 10



20

$20 + 10$
 $= 30$

UPDATE

SYNTAX

UPDATE table_name

SET column_name = value

where condition.

eg: →

update film

set release_year = 2006 //

where id = 1;

update film

set release_year = 2006,

rating = 'PG'

where id = 1;

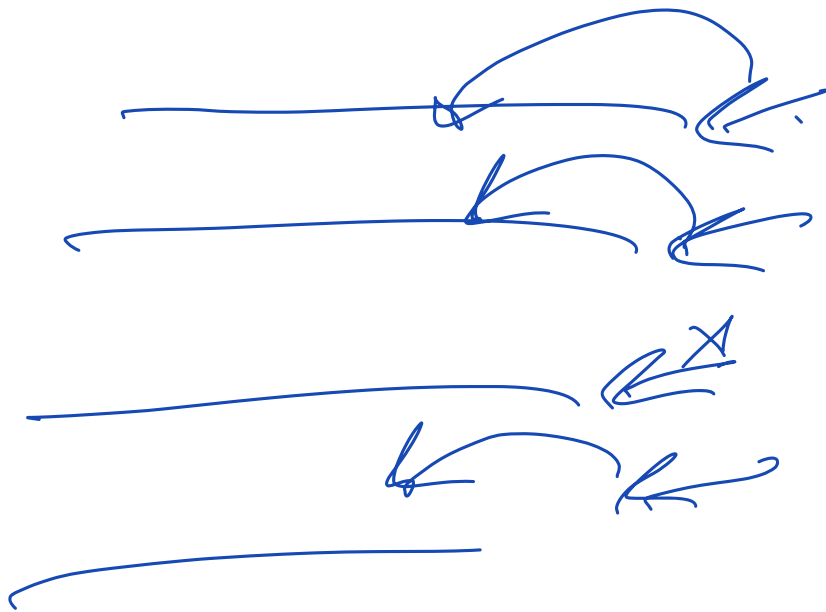
Pseudo code for update

for each row in film

if (row matches cond. in where)

row['release_year'] = 2006;

row['rating'] = 'PG';



DELETE

for each row in table

if (row matches condition in where)
delete row;

TRUNCATE

2 steps

→ DROP schema + data

→ Recreate the schema
(without data)

DROP

remove table as well as data.

DROP table name;

Delete

→ Removes row by row

→ It is slower than truncate

→ Not reset the key

→ can be rolled back

Truncate

→ Remove complete table (+ data) and then recreate it.

→ fast

→ Reset the key

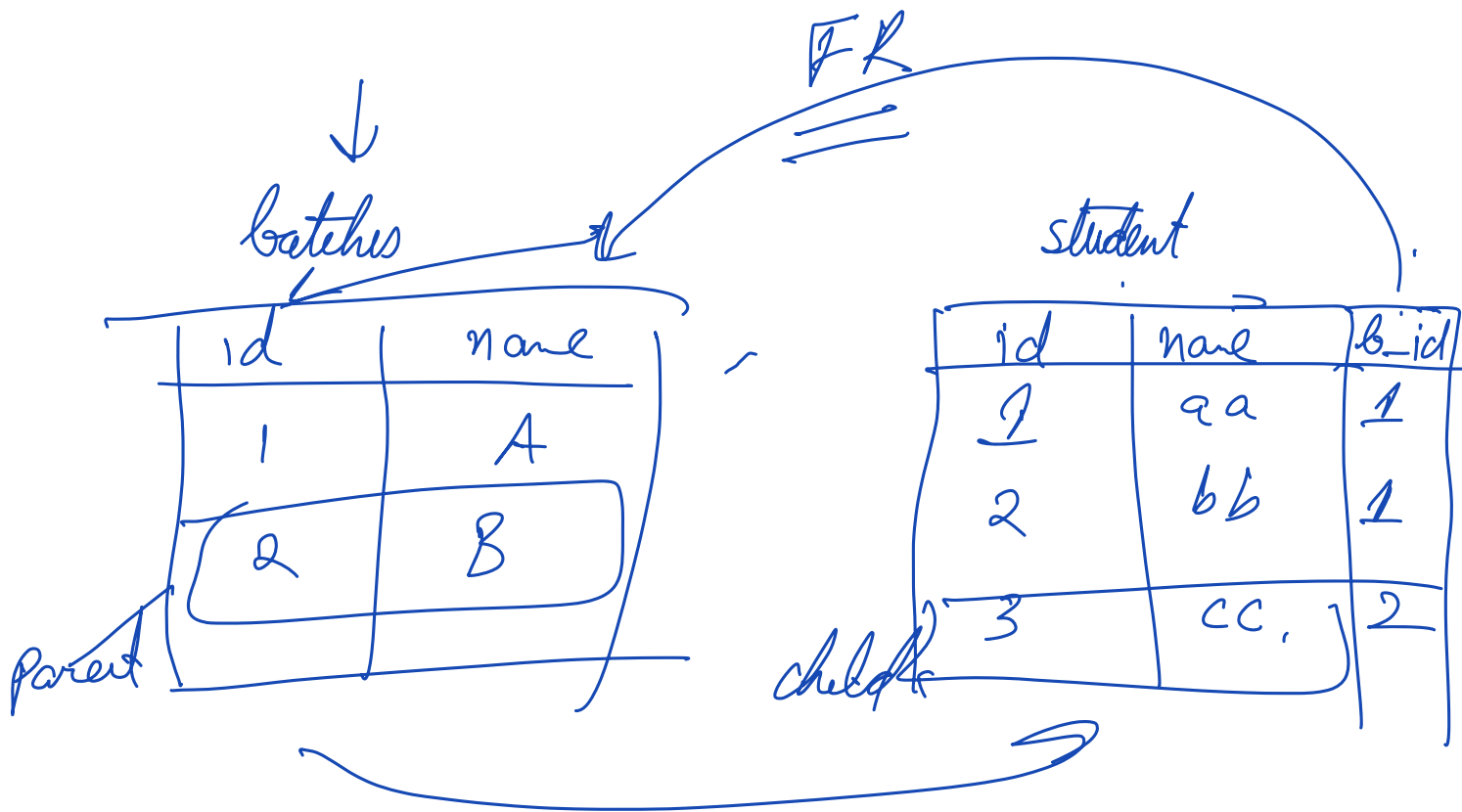
→ Cannot be rolled back

Drop

→ Remove data + structure

→ fastest.

→ Cannot be rolled back;



Cascade → if parent is deleted
delete child

on u
/

id	name
1	A
2	B
3	C

select from <table name>
where <condition>
order by <col>
limit < > offset < >