Today's Agenda :-

- Candidate Keys
- Primary Key
- Foreign Key
- Composite Keys

Intro to SQL

Students

Name	email	psp	batch_id	phNo.

{email} ✓

{email, psp}

{phNo} ✓

{phNo, batch_id}

{phNo, batch_id, psp}

In many of the Super Keys, there are columns that didn't really play any role in uniquely identifying a row. Even without them we would have been uniquely identified a row.

Minimal Super key.

Candidate Key :- Superkeys of smallest size.

↳ Super Keys where if you remove any of the columns, the remaining part will not be a super keys

↳ each of the columns are necessary to identify a unique row.

(a, b, c) — x x x

(a, b) x

(a, c) x

(b, c) x

Attendance :-

Composite Primary Key
↳ (St_id, class_id)

Student_id	class_id	attendance_present
400	191	90 %
401	191	40
400	202	30
400	210	40
401	210	40
402	210	80
400	291	2
400	300	30

Super Keys :-

(Student_id) ✗

(class_id) ✗

(attendance) ✗

(Student_id, attendance) ✗

(class_id, attendance) ✗

Composite Super Key

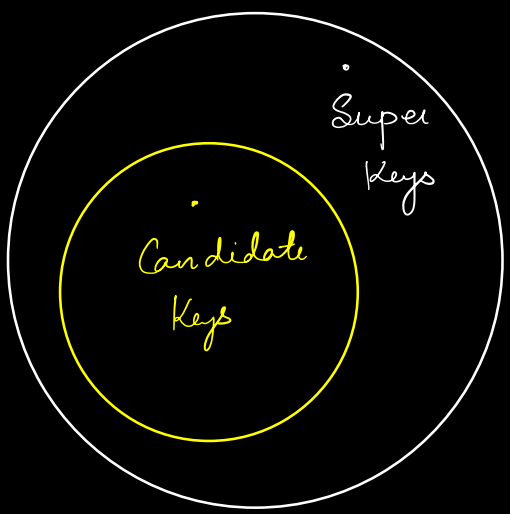
(Student_id, class_id) ✓

(Student_id, class_id, attenda.) ✗

Candidate Keys :- where each col is mandatory to uniquely identify a row.

Not have unnecessary columns
+
Candidate : uniquely identify a row

Super : uniquely identify a row



e_id, dep ✓ ×

email ✓ ✓

f, l ×

l, dep ×

e_id, email ✓ ×

e_id ✓ ✓

email ✓ ✓

both b & c ✓

e_id ✓

email ✓

f, l ×

l, dep ×

C & D ✓

Super Keys : Party Workers

Candidate Keys : Best Super Keys

Primary Key : Best of all CK.

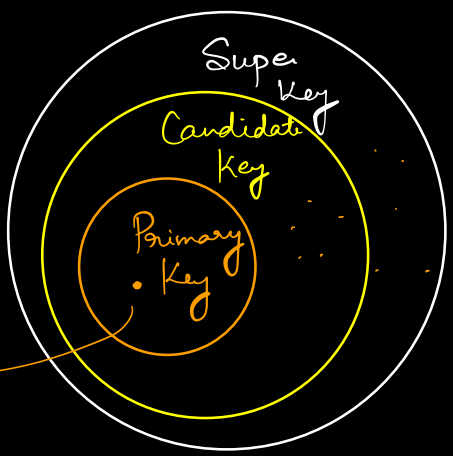
Any database table can have lot of candidate keys

(email) ✓

(ph No) ✓

But when I actually create a table in a SQL database, it forces us to specify a set of colums that will be able to uniquely identify a row.

Primary Key :- A candidate key is selected as primary key.

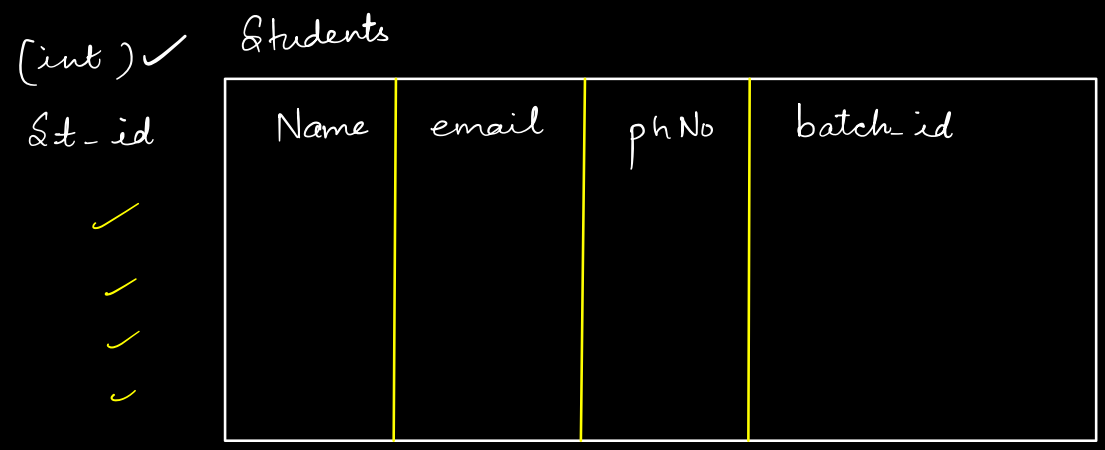


Why ?

- database sorts the data by primary key.
- Outputs the results of every query sorted by primary key.
- creates an index as well on primary key.
  ↳ Indexing

Preferred to have a primary key with a single integer value.

A good primary key :-
- Single column ✓
- integer
- Should never change



	SK	CK	PK
Name	x	x	x
email	✓	✓	
phNo	✓	✓	
batch_id	x	x	x
St_id	✓	✓	✓ good PK

→ better pk out of these 2.

- Often we end up adding a new column called id to the database table
- This id becomes the primary key
- database supports having an id column that auto increment

Introduction to SQL :-

↳ Structured Query Language

↳ used to interact with relational databases.

- Create
- Read
- Update
- delete

(CRUD)

Go through { Data types in SQL }

A simple query to create a table in MySQL looks like this:

```
CREATE TABLE students (
    id INT AUTO_INCREMENT,
    firstName VARCHAR(50) NOT NULL,
    lastName VARCHAR(50) NOT NULL,
    email VARCHAR(100) UNIQUE NOT NULL,
    dateOfBirth DATE NOT NULL,
    enrollmentDate TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    psp DECIMAL(3, 2) CHECK (psp BETWEEN 0.00 AND 100.00),
    batchId INT,
    isActive BOOLEAN DEFAULT TRUE,
    PRIMARY KEY (id),
);
```

& columns

True

(3,2) 0.0

SQL language

Postgre SQL

Composite Keys :- Keys which has more than 1 col.

(phNo) ×

(phNo, email) Composite Sk.

Foreign Keys :- fk has nothing to do with PK SK, CK.

Students

id	Name	email	batch_id fk
1	Naman	- -	1
2	Satish	. - .	2
3	Deep	. . .	2

Batches

id	Name	St_date	end_date	St_count
1	May-24	01/01/24	01/09/24	158
2	June-23	02/03/23	04/01/24	260

Foreign Keys :- A col or set of col^n that helps to uniquely identify a row in another table

Specifying a foreign key in a database helps maintain **data integrity**

Students

id	Name	email	batch_id (fk)
1	Naman	- -	~~1~~ 12
2	Satish	. - -	2
3	Deep	. . .	2
4	Aryan	~	4

Batches

id	Name	St_date	end_date	St_count
12	May_24	01/01/24	01/09/24	158
~~2~~	~~June_23~~	~~02/03/23~~	~~09/01/24~~	~~266~~

X

① insert in Students table

② deleting from batches
- 1) delete Students of that batch
- 2) set batch_id of those Students as NULL
- 3) disallow removing that batch.

③ updating a batch
- 1) update
- 2) NULL

└ 3) disallow

for the same, it's better to define fk constraints :-

1) do what you did :- CASCADE
2) NULL :- SET NULL
3) NOT ALLOW :- NO ACTION

In SQL, when you create a foreign key, you can also specify what to do in case of update/delete etc.

```sql
-- Creating 'batches' table
CREATE TABLE batches (
    batch_id INT PRIMARY KEY,
    batch_name VARCHAR(50) NOT NULL
);

-- Inserting dummy data into 'batches' table
INSERT INTO batches(batch_id, batch_name) VALUES
(1, 'Batch A'),
(2, 'Batch B'),
(3, 'Batch C');

-- Creating 'students' table with ON DELETE and ON UPDATE constraints
CREATE TABLE students (
    student_id INT AUTO_INCREMENT PRIMARY KEY,
    first_name VARCHAR(50) NOT NULL,
    last_name VARCHAR(50) NOT NULL,
    batch_id INT,
    FOREIGN KEY (batch_id) REFERENCES batches(batch_id) ON DELETE CASCADE ON UPDATE CASCADE
);
```

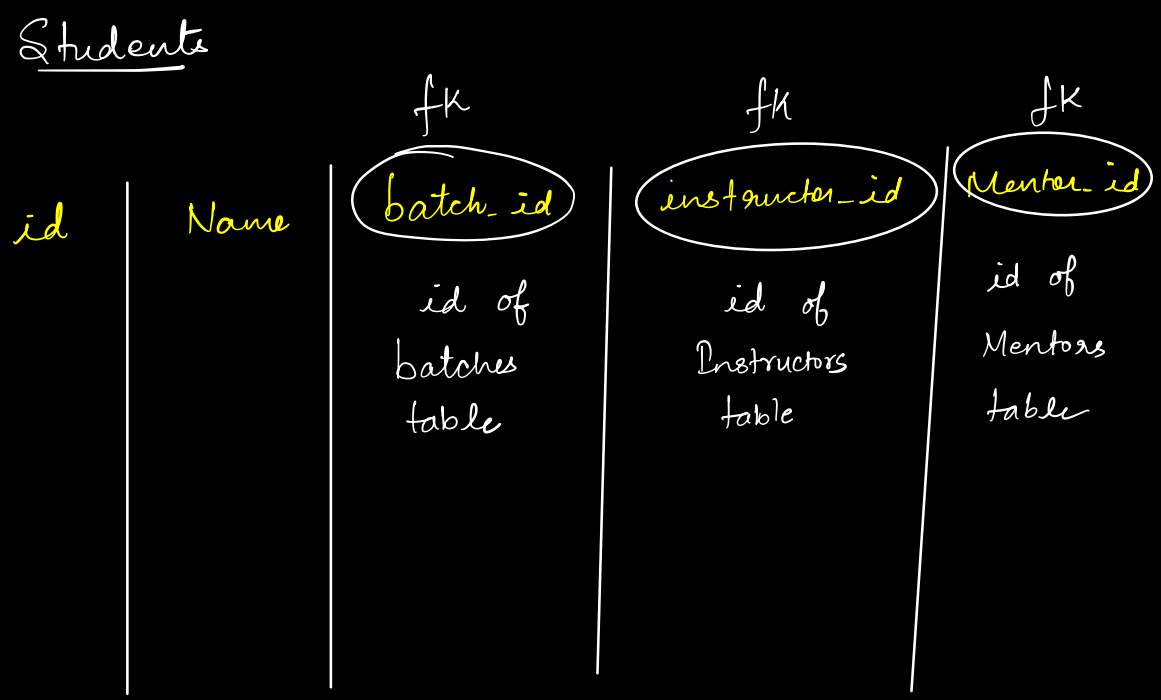
Batches

1	A
2	B
3	C

Students

id f l b_id

Set a fk constraint


Students
fk
fk
fk
id
Name
batch_id
instructor_id
Mentor_id
id of batches table
id of Instructors table
id of Mentors table

Homework :: (before next class)

1) Watch | Read about data types in SQL

2) Install MySQL 8

3) MySQL Workbench

4) Set up Sakilla Database.
   ↳ official MySQL dataset

Thank You