Anomalies

Anomalies

Anomalies in databases refer to inconsistencies or unexpected issues that can occur during data manipulation or retrieval

Anomalies

There are three main types of anomalies:

Anomalies

There are three main types of anomalies:

- Update Anomalies
- Delete Anomalies
- Insert Anomalies

Anomalies

Update Anomalies

e_id	name	branch	address
1	Arish	dhaka	rampura
2	Nuruddin	comilla	durgapur
3	Payel	dhaka	rampura
4	Hassan	barisal	banglabazar
6	Shifa	dhaka	rampura

Anomalies

Delete Anomalies

e_id	name	branch	address
1	Arish	dhaka	rampura
2	Nuruddin	comilla	durgapur
3	Payel	dhaka	rampura
4	Hassan	barisal	banglabazar
6	Shifa	dhaka	rampura

Anomalies

Delete Anomalies

e_id	name	branch	address
1	Arish	dhaka	rampura
2	Nuruddin	comilla	durgapur
3	Payel	dhaka	rampura
~~4~~	~~Hassan~~	~~barisal~~	~~banglabazar~~
6	Shifa	dhaka	rampura

Anomalies

Delete Anomalies

e_id	name	branch	address
1	Arish	dhaka	rampura
2	Nuruddin	comilla	durgapur
3	Payel	dhaka	rampura
6	Shifa	dhaka	rampura

Anomalies

Insert Anomalies

e_id	name	branch	address
1	Arish	dhaka	rampura
2	Nuruddin	comilla	durgapur
3	Payel	dhaka	rampura
4	Hassan	barisal	banglabazar
6	Shifa	dhaka	rampura

Anomalies

Insert Anomalies

e_id	name	branch	address
1	Arish	dhaka	rampura
2	Nuruddin	comilla	durgapur
3	Payel	dhaka	rampura
4	Hassan	barisal	banglabazar
6	Shifa	dhaka	rampura
7	John	dhaka	mirpur

Anomalies

Insert Anomalies

e_id	name	branch	address
1	Arish	dhaka	rampura
2	Nuruddin	comilla	durgapur
3	Payel	dhaka	rampura
4	Hassan	barisal	banglabazar
6	Shifa	dhaka	rampura
7	John	dhaka	mirpur
8	Kofil		

Anomalies

Insert Anomalies

e_id	name	branch	address
1	Arish	dhaka	rampura
2	Nuruddin	comilla	durgapur
3	Payel	dhaka	rampura
4	Hassan	barisal	banglabazar
6	Shifa	dhaka	rampura
7	John	dhaka	mirpur
8	Kofil		

Anomalies

Insert Anomalies

Employee

e_id	name	branch	address
1	Arish	dhaka	rampura
2	Nuruddin	comilla	durgapur
3	Payel	dhaka	rampura
4	Hassan	barisal	banglabazar
6	Shifa	dhaka	rampura
7	John	dhaka	mirpur
8	Kofil		

branch

Normalization

e_id	name	branch
1	Arish	1
2	Nuruddin	2
3	Payel	1
4	Hassan	3
6	Shifa	1
7	John	1

b_id	branch	address
1	dhaka	rampura
2	comilla	durgapur
3	barisal	banglabazar

Normalization

Normalization

Fubnctional Dependency

Normal Forms

Functional Dependency

Functional dependency in simple terms means that the value of one attribute (or set of attributes) uniquely determines the value of another attribute(s) in a table.

Functional Dependency

x	y
1	0
2	3
3	5
4	7
2	5

Functional Dependency

x	y
1	0
2	3
3	5
4	7
2	5

Functional Dependency

x	y
1	0
2	3
3	5
4	7
2	5

Functional Dependency

x	y
1	0
2	3
3	5
4	7
2	3

Functional Dependency

A functional dependency X→Y exists if, for each value of X, there is only one corresponding value of Y .

x	y
1	0
2	3
3	5
4	7
2	3

Functional Dependency

role	name
manager	mezba
developer	mir
manager	milton
instructor	firoz
developer	nahid
manager	toky

manager → mezba / milton / toky

developer → mir / nahid

Functional Dependency

role	name
manager	mezba
developer	mir
manager	milton
instructor	firoz
developer	nahid
manager	toky

manager → mezba / milton / toky

developer → mir / nahid

Functional Dependency

role	name
manager	mezba
developer	mir
manager	milton
instructor	shafayet
developer	tanmoy
manager	toky

manager → mezba / milton / toky

developer → mir / nahid

role → name

mezba → manager

tanmoy → developer

Functional Dependency

role	name
manager	mezba
developer	mir
manager	milton
instructor	shafayet
developer	tanmoy
manager	toky

manager → mezba / milton / toky

developer → mir / nahid

role → name

mezba → manager

tanmoy → developer

name → role

Functional Dependency

EmployeeID	EmployeeName	Department
101	John	Engineering
102	Emily	Marketing
103	Sarah	Engineering
104	Jack	HR
105	John	Manager

Normalization

Fubnctional Dependency

Normal Forms

Normalization

Normal Forms

A set of rules applied to a database table to reduce redundancy and avoid anomalies in data by organizing it properly.

Normalization

Normal Forms

- 0NF
- 1NF
- 2NF
- 3NF

Normalization

1NF

Normalization

1NF

Rules:

- Atomic Values
- Unique Column Names
- Positional dependency of data
- Column should contain data that are of the same type
- Determine Primary key

Serial_No.	Titles	Courses
11	Xkon	CN, OS
12	Ykon	Java
13	4534	C++, C

Normalization

1NF

Rules:

- Atomic Values
- Unique Column Names
- Positional dependency of data
- Column should contain data that are of the same type
- Determine Primary key

Serial_No.	Titles	Titles	Courses
11	Xkon	mkon	CN, OS
12	Ykon	...	Java
13	4534	...	C++, C

Normalization

1NF

Rules:

- Atomic Values
- Unique Column Names
- Positional dependency of data
- Column should contain data that are of the same type
- Determine Primary key

Serial_No.	Titles	Courses
11	Xkon	CN, OS
12	Ykon	Java
13	4534	C++, C

Normalization

1NF

Rules:

- Atomic Values
- Unique Column Names
- Positional dependency of data
- Column should contain data that are of the same type
- Determine Primary key

Serial_No.	Titles	Courses
11	Xkon	CN, OS
12	Ykon	Java
13	zkon	C++, C

Normalization

1NF

Rules:

- Atomic Values
- Unique Column Names
- Positional dependency of data
- Column should contain data that are of the same type
- Determine Primary key

Serial_No.	Titles	Courses
11	Xkon	CN
11	Xkon	OS
12	Ykon	Java
13	Zkon	C++
13	Zkon	C

Normal Forms

2NF

Normal Forms

2NF

Rules:

- Must be in INF
- No non-key attribute should depend on part of a candidate key

stud_id	c_id	c_name	Instructor
101	1	Math	Prof. Smith
102	2	Science	Prof. Johnson
101	3	History	Prof. Adams
103	1	Math	Prof. Smith

Normal Forms

2NF

Rules:

- Must be in 1NF
- No non-key attribute should depend on part of a candidate key

stud_id	c_id	c_name	Instructor
101	1	Math	Prof. Smith
102	2	Science	Prof. Johnson
101	3	History	Prof. Adams
103	1	Math	Prof. Smith

Normal Forms

2NF

c_id ⟶ c_name

Rules:

- Must be in INF
- No non-key attribute should depend on part of a candidate key

stud_id	c_id	c_name	Instructor
101	1	Math	Prof. Smith
102	2	Science	Prof. Johnson
101	3	History	Prof. Adams
103	1	Math	Prof. Smith

Normal Forms

2NF

Instructor

c_id	Instructor
1	Prof. Smith
2	Prof. Johnson
3	Prof. Adams

Course

c_id	c_name
1	Math
2	Science
3	History

2NF

Instructor

c_id	Instructor
1	Prof. Smith
2	Prof. Johnson
3	Prof. Adams

Course

c_id	c_name
1	Math
2	Science
3	History

Lossy Decomposition

Normal Forms

2NF

Instructor

c_id	Instructor
1	Prof. Smith
2	Prof. Johnson
3	Prof. Adams

Course

c_id	c_name
1	Math
2	Science
3	History

StudentCourse

stud_id	c_id
101	1
102	2
101	3
103	1

2NF

Instructor

c_id	Instructor
1	Prof. Smith
2	Prof. Johnson
3	Prof. Adams

Course

c_id	c_name
1	Math
2	Science
3	History

StudentCourse

stud_id	c_id
101	1
102	2
101	3
103	1

Lossless Decomposition

Normal Forms

3NF

Normal Forms

3NF

- Rules:
  - **Must be in 2NF**
  - Must not contain transitive dependency

stud_Id	stud_name	stud_phone	state	country	stud_age
101	John	12345679	California	USA	20
102	Emily	987654321	Toronto	CANADA	22
103	Alex	555555555	California	USA	21

Normal Forms

$X \rightarrow Y$ & $Y \rightarrow Z$

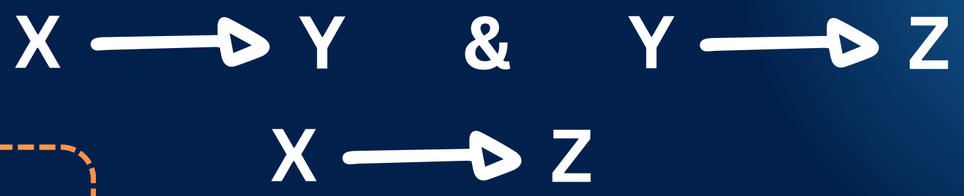
$X \rightarrow Z$

3NF

- Rules:
  - **Must be in 2NF**
  - Must not contain transitive dependency

stud_Id	stud_name	stud_phone	state	country	stud_age
101	John	12345679	California	USA	20
102	Emily	987654321	Toronto	CANADA	22
103	Alex	555555555	California	USA	21

Normal Forms



3NF

- Rules:
  - **Must be in 2NF**
  - Must not contain transitive dependency

stud_Id	stud_name	stud_phone	state	country	stud_age
101	John	12345679	California	USA	20
102	Emily	987654321	Toronto	CANADA	22
103	Alex	555555555	California	USA	21

Normal Forms

$X \longrightarrow Y \quad \& \quad Y \longrightarrow Z$

$X \longrightarrow Z$

3NF

- Rules:
  - **Must be in 2NF**
  - Must not contain transitive dependency

stud_Id	stud_name	stud_phone	state	country	stud_age
101	John	12345679	California	USA	20
102	Emily	987654321	Toronto	CANADA	22
103	Alex	555555555	California	USA	21

student_Id ⟶ state

state ⟶ country

student_Id ⟶ country

Normal Forms

3NF

stud_Id	stud_name	stud_phone	stud_age
101	John	12345679	20
102	Emily	987654321	22
103	Alex	555555555	21

state	country
California	USA
Toronto	CANADA
California	USA

Normal Forms

3NF

stud_Id	stud_name	stud_phone	state	stud_age
101	John	12345679	California	20
102	Emily	987654321	Toronto	22
103	Alex	555555555	California	21

state	country
California	USA
Toronto	CANADA
California	USA

Database Design

id	name	email	phone

id	name	email

Determining Entities

Determining Attributes For Each Entities

Relationships Among Entities

Resolving Many to Many Relationship

Resolving Many to Many

Resolving Many to Many

Resolving Many to Many

Resolving Many to Many

Student

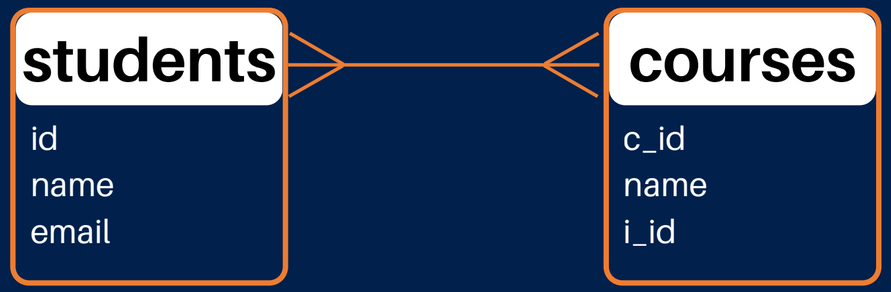
s_id	name	c_ids
101	John	1, 2
102	Emily	2
103	Sarah	1, 3

Course

c_id	c_name	s_ids
1	Math	101, 103
2	Science	101, 102
3	History	103

Student

id	name	c_ids
101	John	1, 2
102	Emily	2
103	Sarah	1, 3

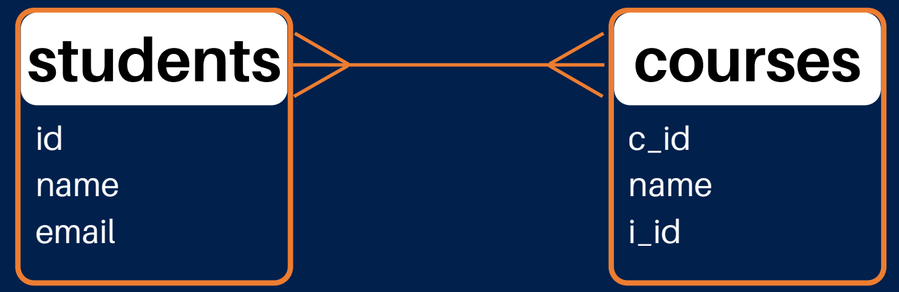


Course

c_id	name	s_ids
1	Math	101, 103
2	Science	101, 102
3	History	103

Student

id	name	c_ids
101	John	1, 2
102	Emily	2
103	Sarah	1, 3

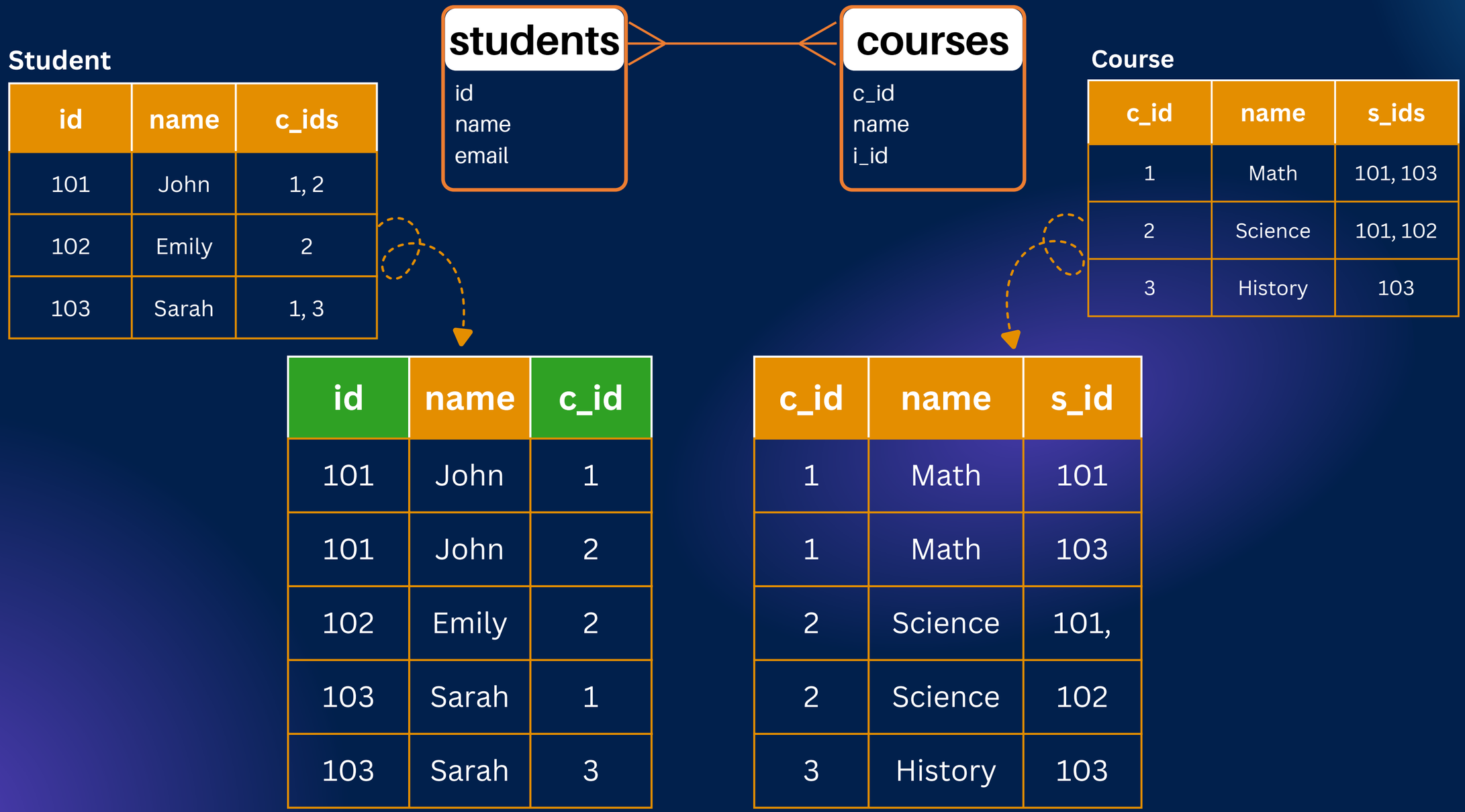


Course

c_id	name	s_ids
1	Math	101, 103
2	Science	101, 102
3	History	103

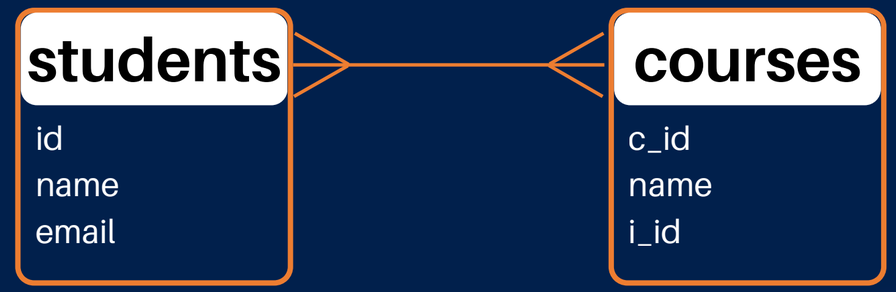
id	name	c_id
101	John	1
101	John	2
102	Emily	2
103	Sarah	1
103	Sarah	3

c_id	name	s_id
1	Math	101
1	Math	103
2	Science	101,
2	Science	102
3	History	103


Student
id
name
c_ids
101
John
1, 2
102
Emily
2
103
Sarah
1, 3
students
id
name
email
courses
c_id
name
i_id
Course
c_id
name
s_ids
1
Math
101, 103
2
Science
101, 102
3
History
103
id
name
c_id
101
John
1
101
John
2
102
Emily
2
103
Sarah
1
103
Sarah
3
c_id
name
s_id
1
Math
101
1
Math
103
2
Science
101,
2
Science
102
3
History
103

Student

id	name	c_ids
101	John	1, 2
102	Emily	2
103	Sarah	1, 3



Course

c_id	name	s_ids
1	Math	101, 103
2	Science	101, 102
3	History	103

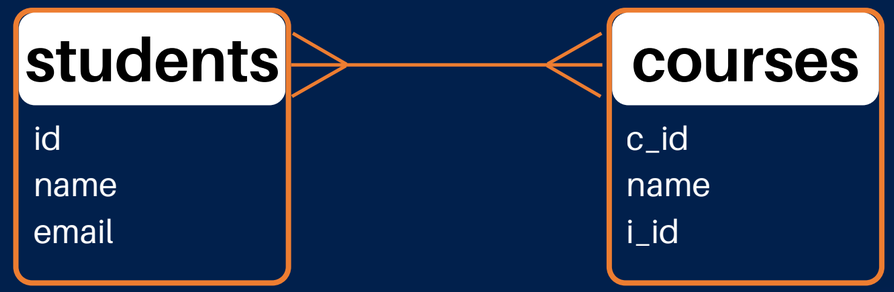
{id, c_id}

id	name	c_id
101	John	1
101	John	2
102	Emily	2
103	Sarah	1
103	Sarah	3

c_id	name	s_id
1	Math	101
1	Math	103
2	Science	101,
2	Science	102
3	History	103

Student

id	name	c_ids
101	John	1, 2
102	Emily	2
103	Sarah	1, 3



Course

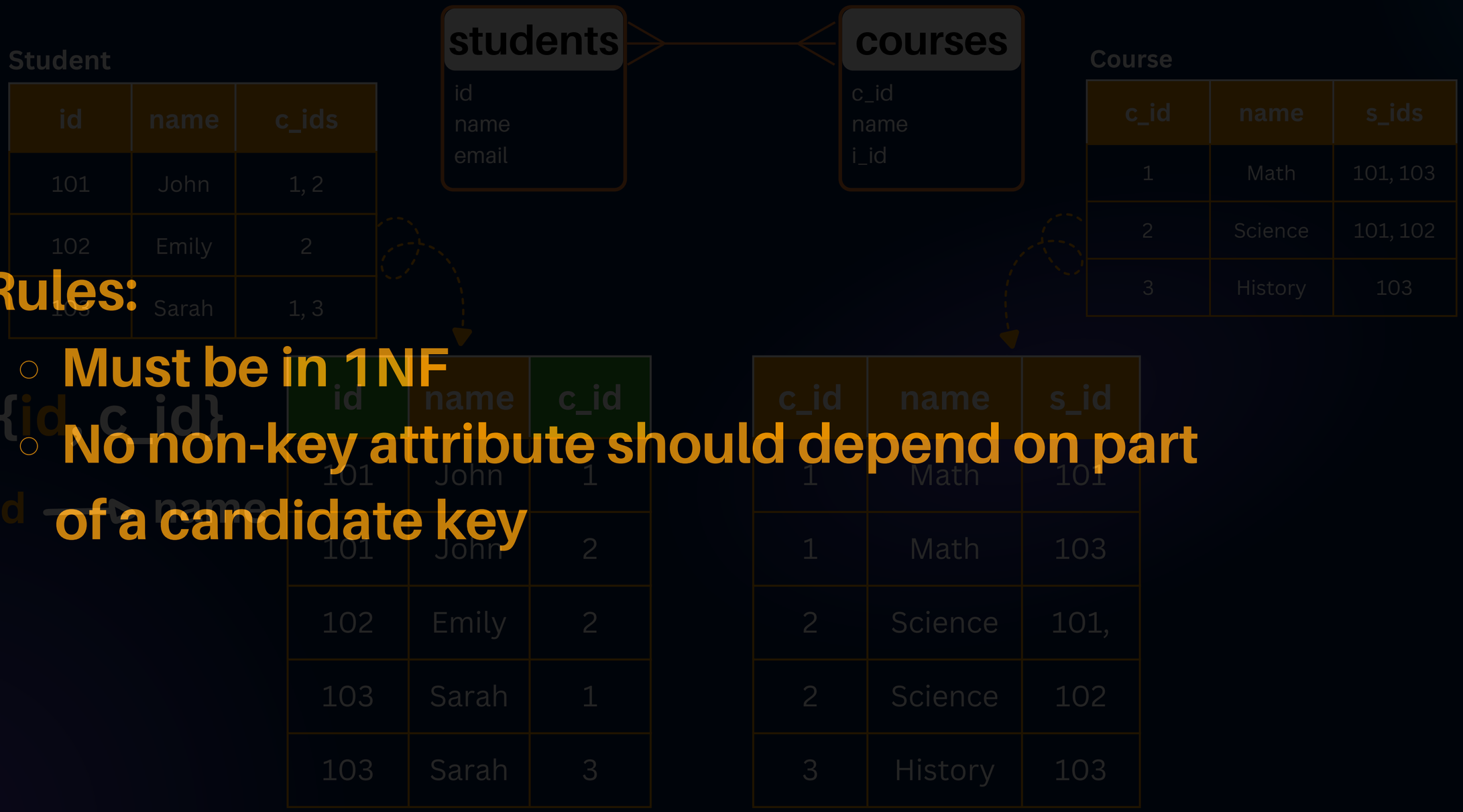
c_id	name	s_ids
1	Math	101, 103
2	Science	101, 102
3	History	103

{id, c_id}

id → name

id	name	c_id
101	John	1
101	John	2
102	Emily	2
103	Sarah	1
103	Sarah	3

c_id	name	s_id
1	Math	101
1	Math	103
2	Science	101,
2	Science	102
3	History	103

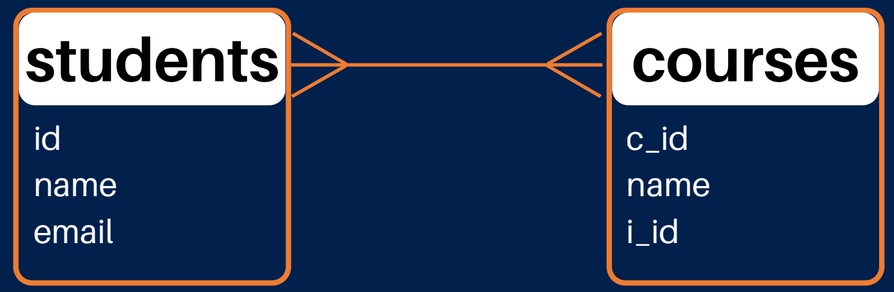


Rules:

- Must be in 1NF
- No non-key attribute should depend on part of a candidate key

Student

id	name	c_ids
101	John	1, 2
102	Emily	2
103	Sarah	1, 3



Course

c_id	name	s_ids
1	Math	101, 103
2	Science	101, 102
3	History	103

{id, c_id}

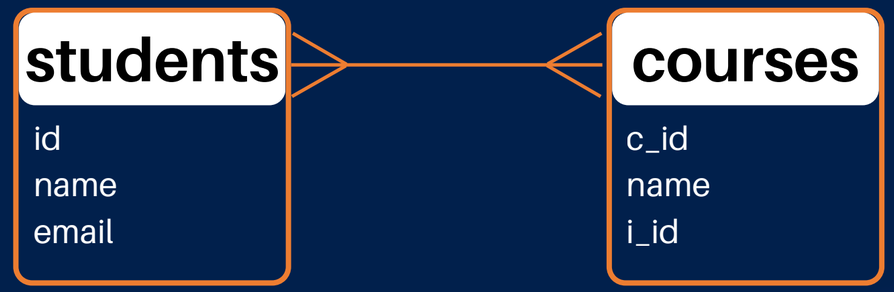
id → name

id	name	c_id
101	John	1
101	John	2
102	Emily	2
103	Sarah	1
103	Sarah	3

c_id	name	s_id
1	Math	101
1	Math	103
2	Science	101,
2	Science	102
3	History	103

Student

id	name	c_ids
101	John	1, 2
102	Emily	2
103	Sarah	1, 3



Course

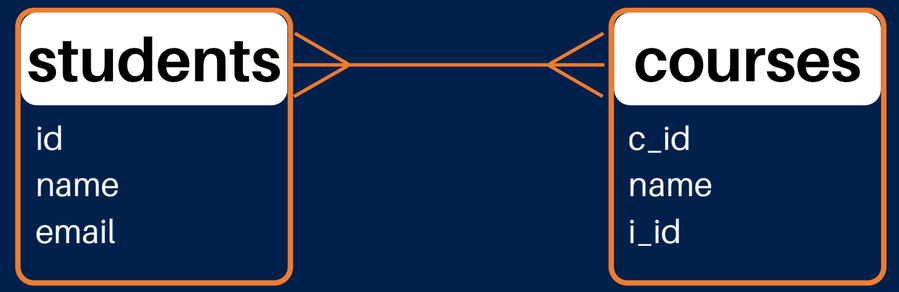
c_id	name	s_ids
1	Math	101, 103
2	Science	101, 102
3	History	103

id	name	c_id1	c_id2	c_id3
101	John	1	2	
102	Emily		2	
103	Sarah	1		3

c_id	name	s_id
1	Math	101
1	Math	103
2	Science	101,
2	Science	102
3	History	103

Student

id	name	c_ids
101	John	1, 2
102	Emily	2
103	Sarah	1, 3

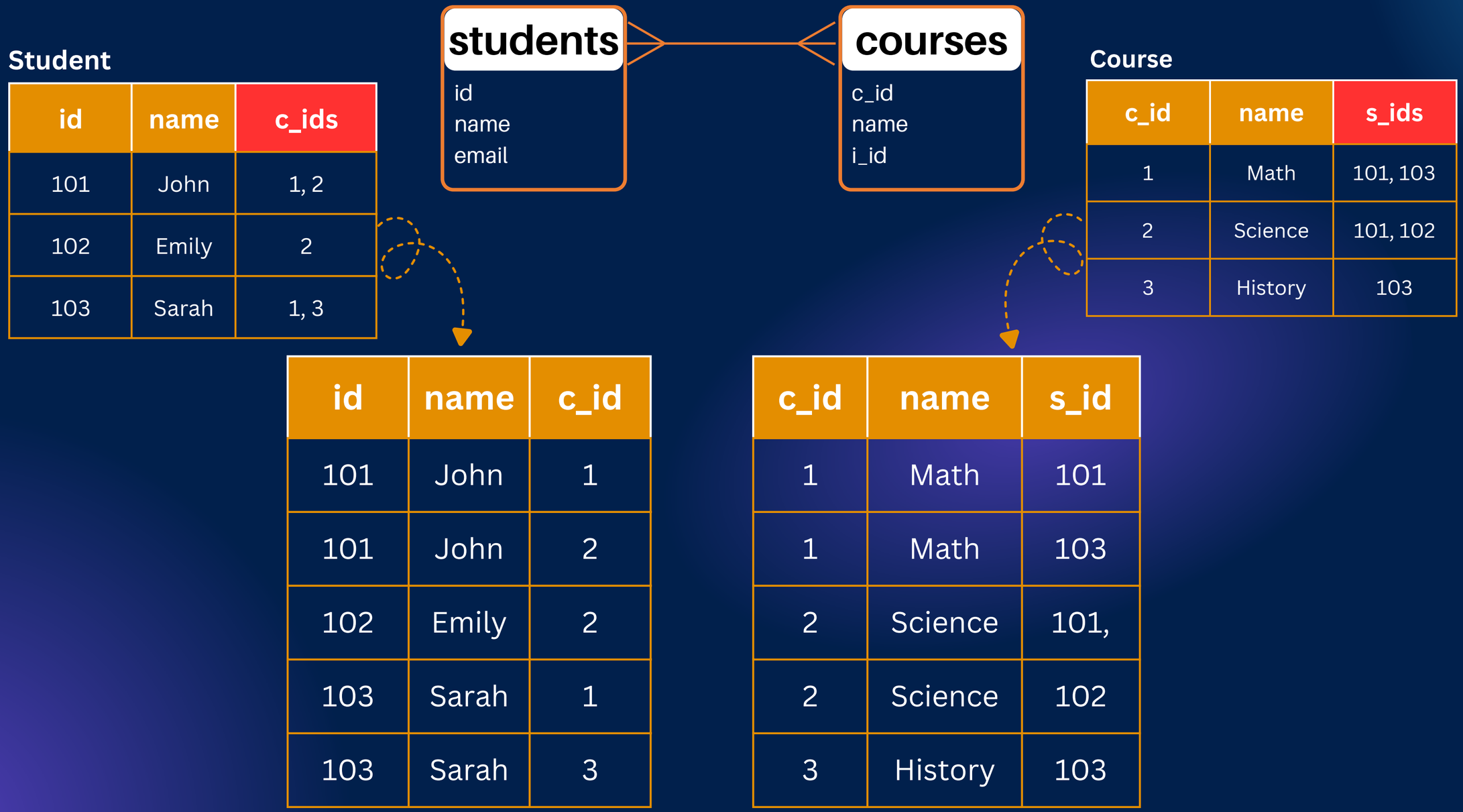


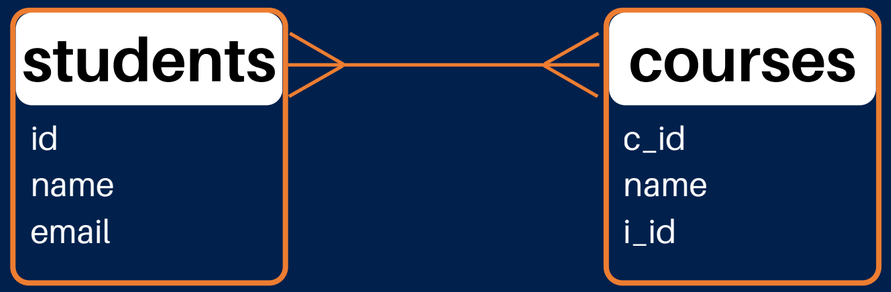
Course

c_id	name	s_ids
1	Math	101, 103
2	Science	101, 102
3	History	103

id	name	c_id
101	John	1
101	John	2
102	Emily	2
103	Sarah	1
103	Sarah	3

c_id	name	s_id
1	Math	101
1	Math	103
2	Science	101,
2	Science	102
3	History	103


Student
id
name
c_ids
101
John
1, 2
102
Emily
2
103
Sarah
1, 3
students
id
name
email
courses
c_id
name
i_id
Course
c_id
name
s_ids
1
Math
101, 103
2
Science
101, 102
3
History
103
id
name
c_id
101
John
1
101
John
2
102
Emily
2
103
Sarah
1
103
Sarah
3
c_id
name
s_id
1
Math
101
1
Math
103
2
Science
101,
2
Science
102
3
History
103

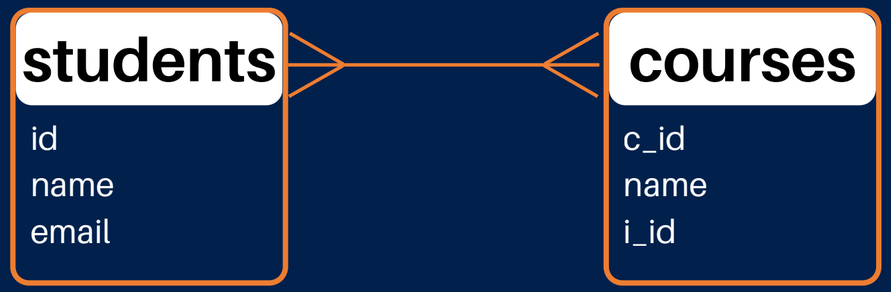


Student

id	name	c_ids
101	John	1, 2
102	Emily	2
103	Sarah	1, 3

Course

c_id	name	s_ids
1	Math	101, 103
2	Science	101, 102
3	History	103

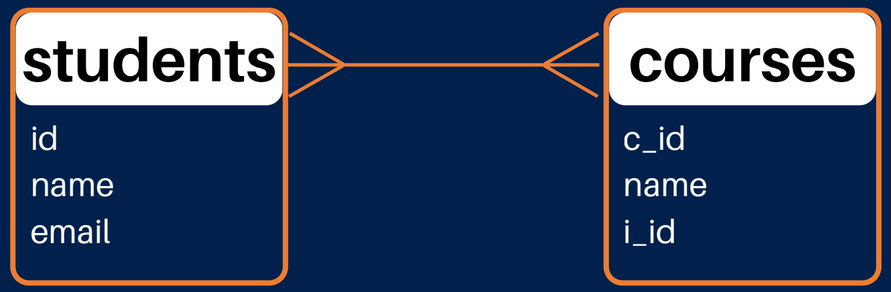


Student

id	name
101	John
102	Emily
103	Sarah

Course

c_id	name
1	Math
2	Science
3	History



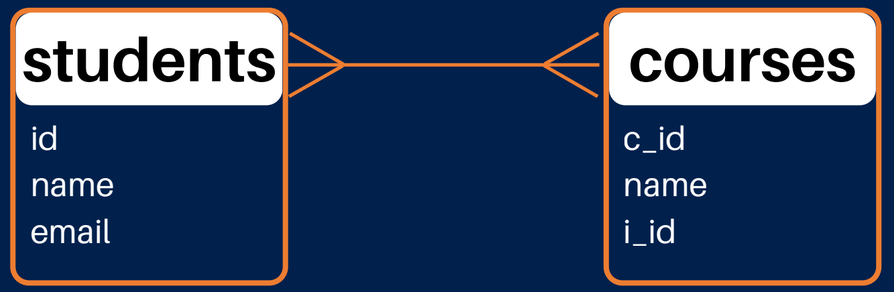
Student

id	name
101	John
102	Emily
103	Sarah

s_id	c_id
101	1
101	2
102	2
103	1
103	3

Course

c_id	name
1	Math
2	Science
3	History



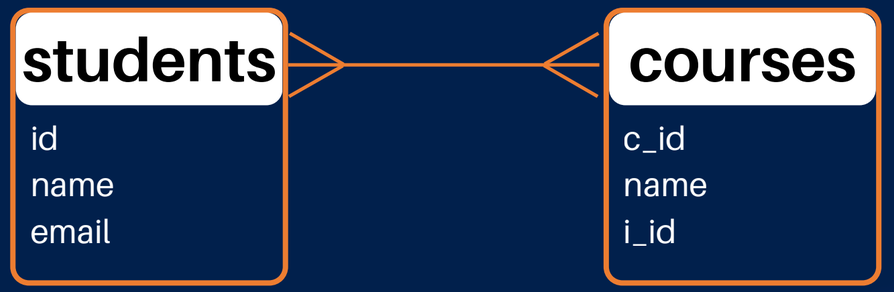
Student

id	name
101	John
102	Emily
103	Sarah

s_id	c_id
101	1
101	2
102	2
103	1
103	3

Course

c_id	name
1	Math
2	Science
3	History



Student

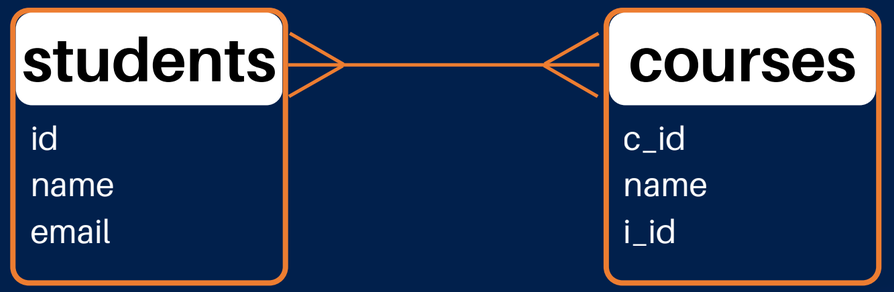
id	name
101	John
102	Emily
103	Sarah

Enrollment

s_id	c_id
101	1
101	2
102	2
103	1
103	3

Course

c_id	name
1	Math
2	Science
3	History



Student

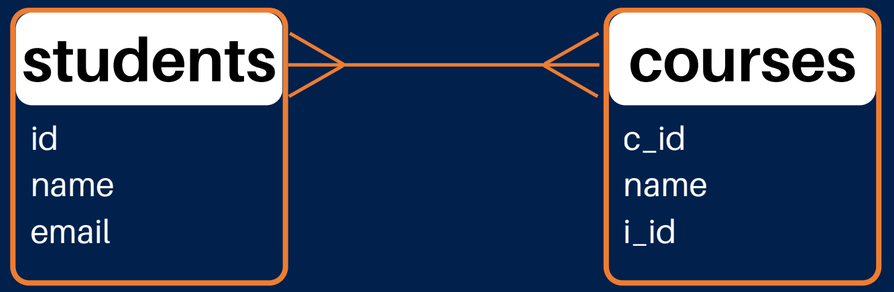
id	name
101	John
102	Emily
103	Sarah

Enrollment

s_id	c_id
101	1
101	2
102	2
103	1
103	3

Course

c_id	name
1	Math
2	Science
3	History



Student

id	name
101	John
102	Emily
103	Sarah

Enrollment

s_id	c_id
101	1
101	2
102	2
103	1
103	3

Course

c_id	name
1	Math
2	Science
3	History

Junction Table

Postgres

Postgres

The World's Most Advanced Open Source Relational Database Management System

Postgres

The World's Most Advanced Open Source Relational Database Management System

DBMS

Postgres

The World's Most Advanced Open Source Relational Database Management System

DBMS

RDBMS

Why Postgres ?

Postgres

- Open Source
- Advanced Data Types
- Scalability
- ACID Compliance
- Modern
- Indexing