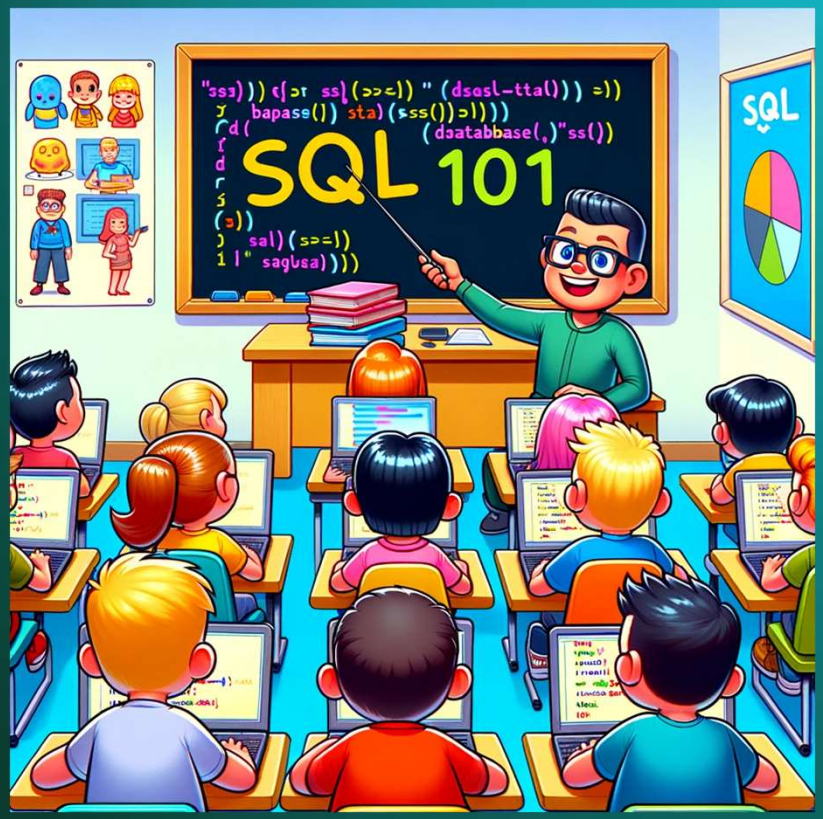


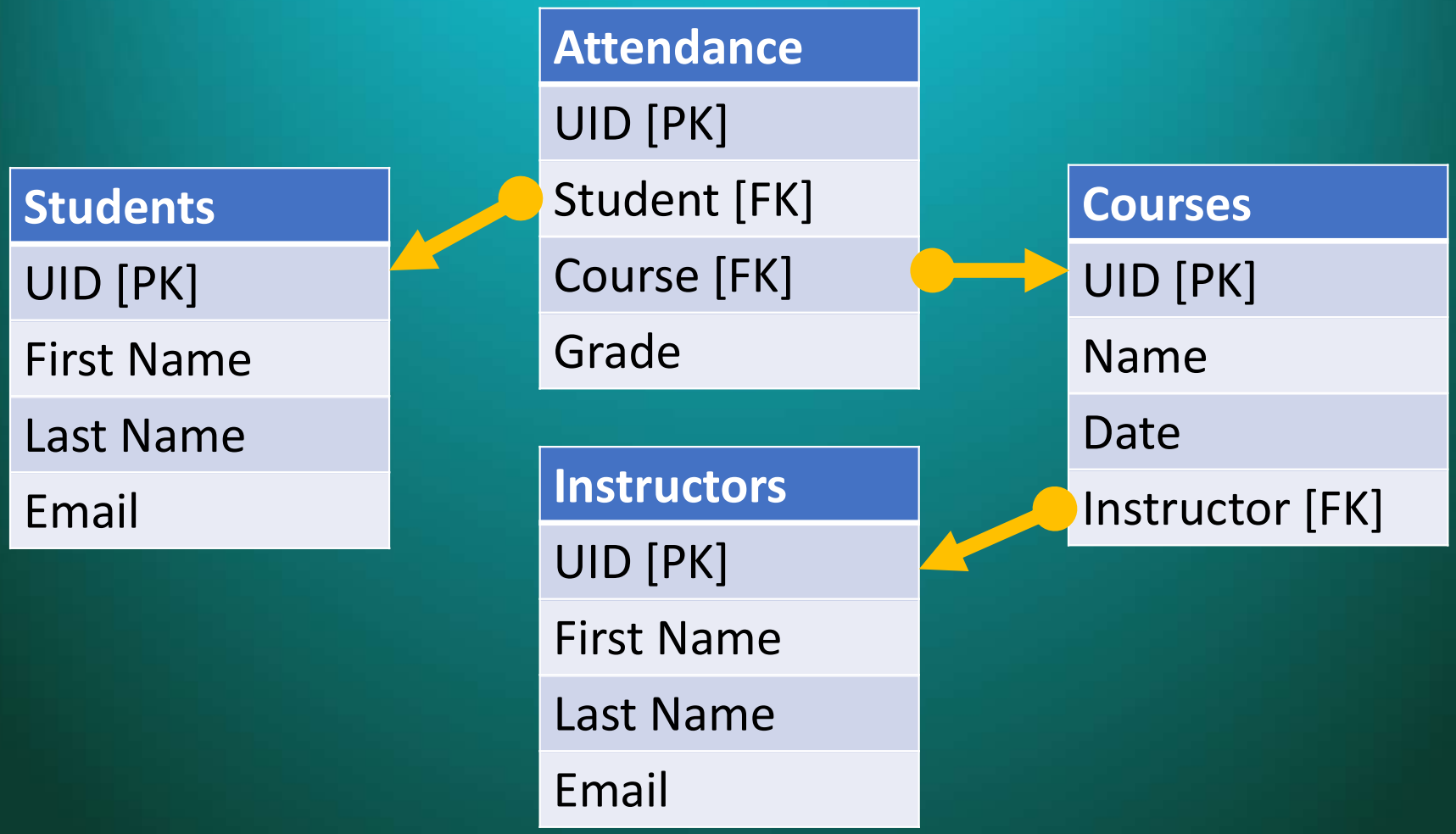
Data Relationships

Monolithic Table

First Name	Last Name	Email	Course	Date	Instructor	Grade
David	Bowie	davie@gmail.com	eng * 1000	23-01-15	E. Myers	78
Ethan	Montgomery	ethanm@uni.edu	eng * 1000	23-01-15	E. Myers	86
Lucas	Bennett	lucas@gmail.com	math * 2010	24-07-07	M. Peabody	
Lucas	Bennett	lucas@gmail.com	eng * 1000	23-01-15	E. Myers	60
Noah	Harrison	noah@google.com				

Monolithic Table

First Name	Last Name	Email	Course	Date	Instructor	Grade
David	Bowie	davie@gmail.com	eng * 1000	23-01-15	E. Myers	78
Ethan	Montgomery	ethanm@uni.edu	eng * 1000	23-01-15	E. Myers	86
Lucas	Bennett	lucas@gmail.com	math * 2010	24-07-07	M. Peabody	
Lucas	Bennett	lucas@gmail.com	eng * 1000	23-01-15	E. Myers	60
Noah	Harrison	noah@google.com				


Attendance
UID [PK]
Student [FK]
Course [FK]
Grade
Students
UID [PK]
First Name
Last Name
Email
Courses
UID [PK]
Name
Date
Instructor [FK]
Instructors
UID [PK]
First Name
Last Name
Email

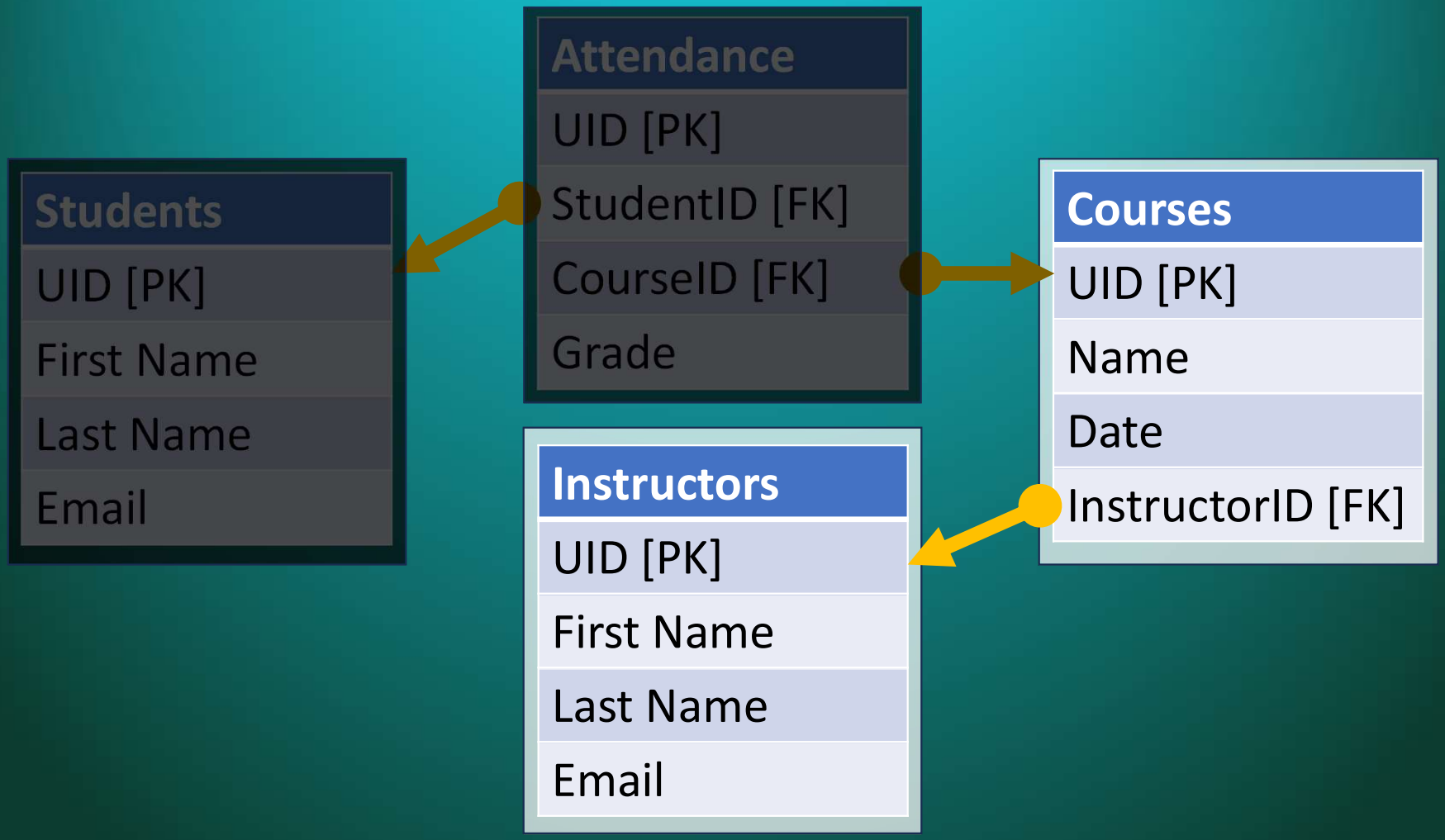
Data Relationships

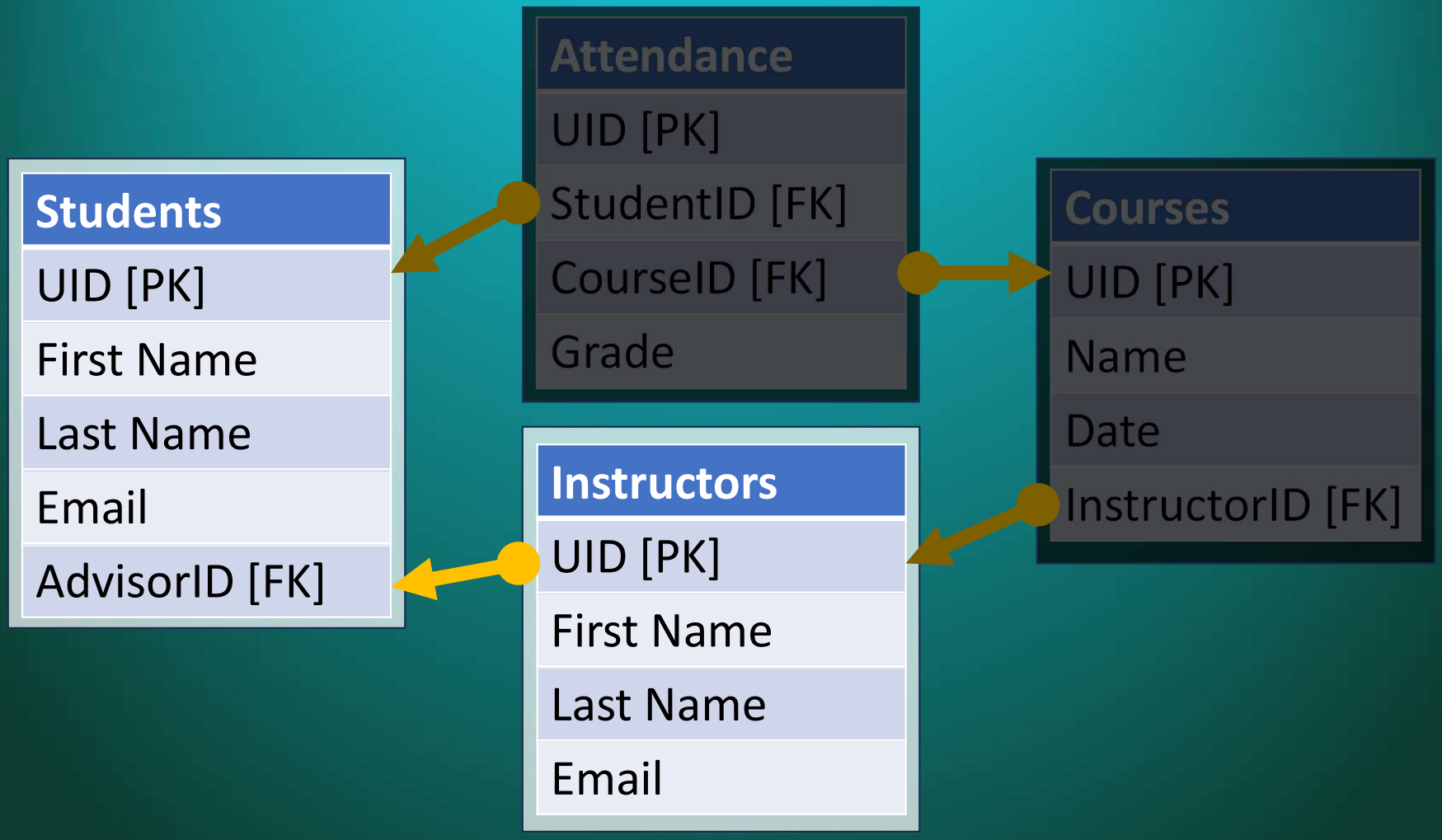
One-to-One

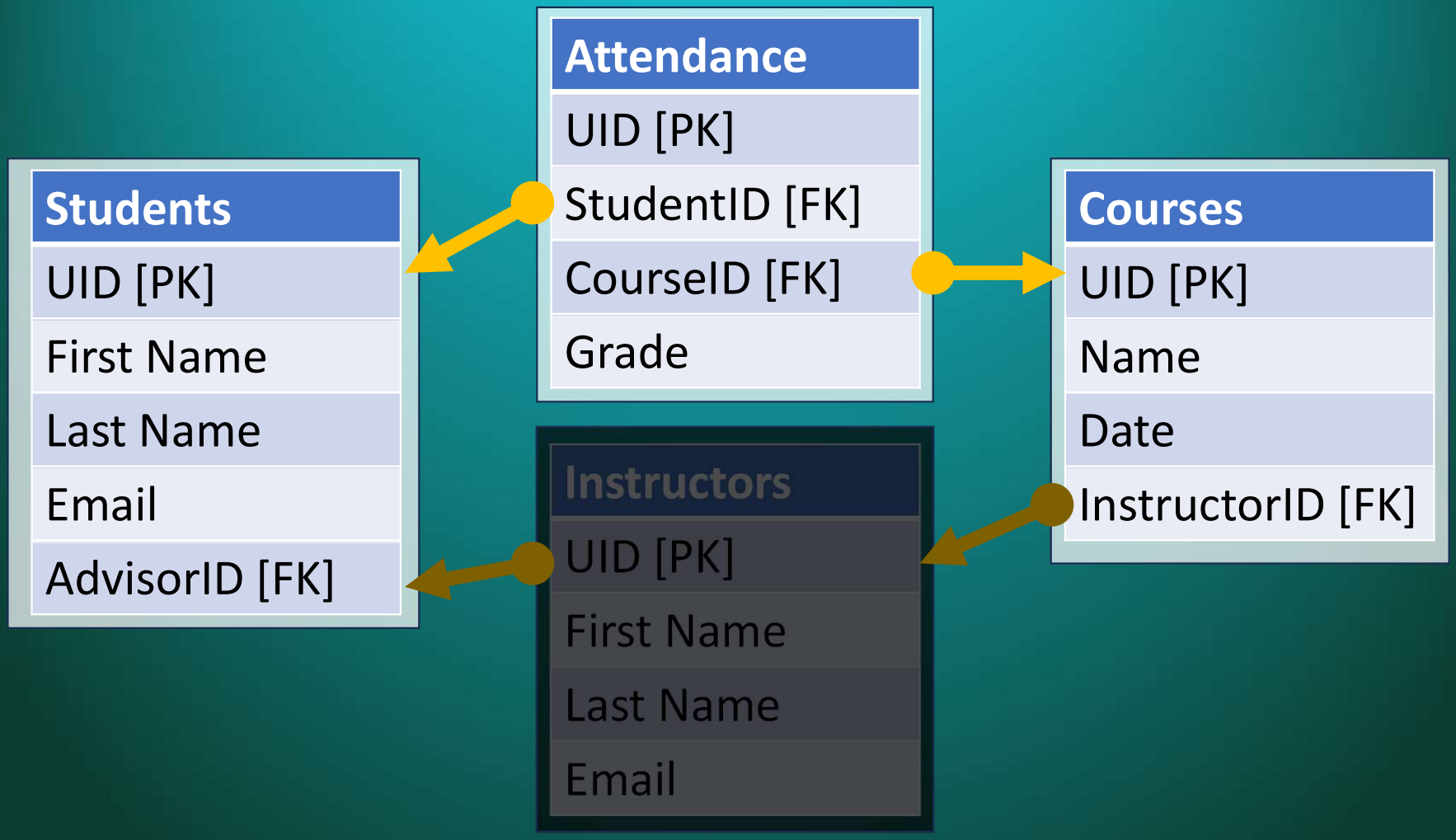
One-to-Many

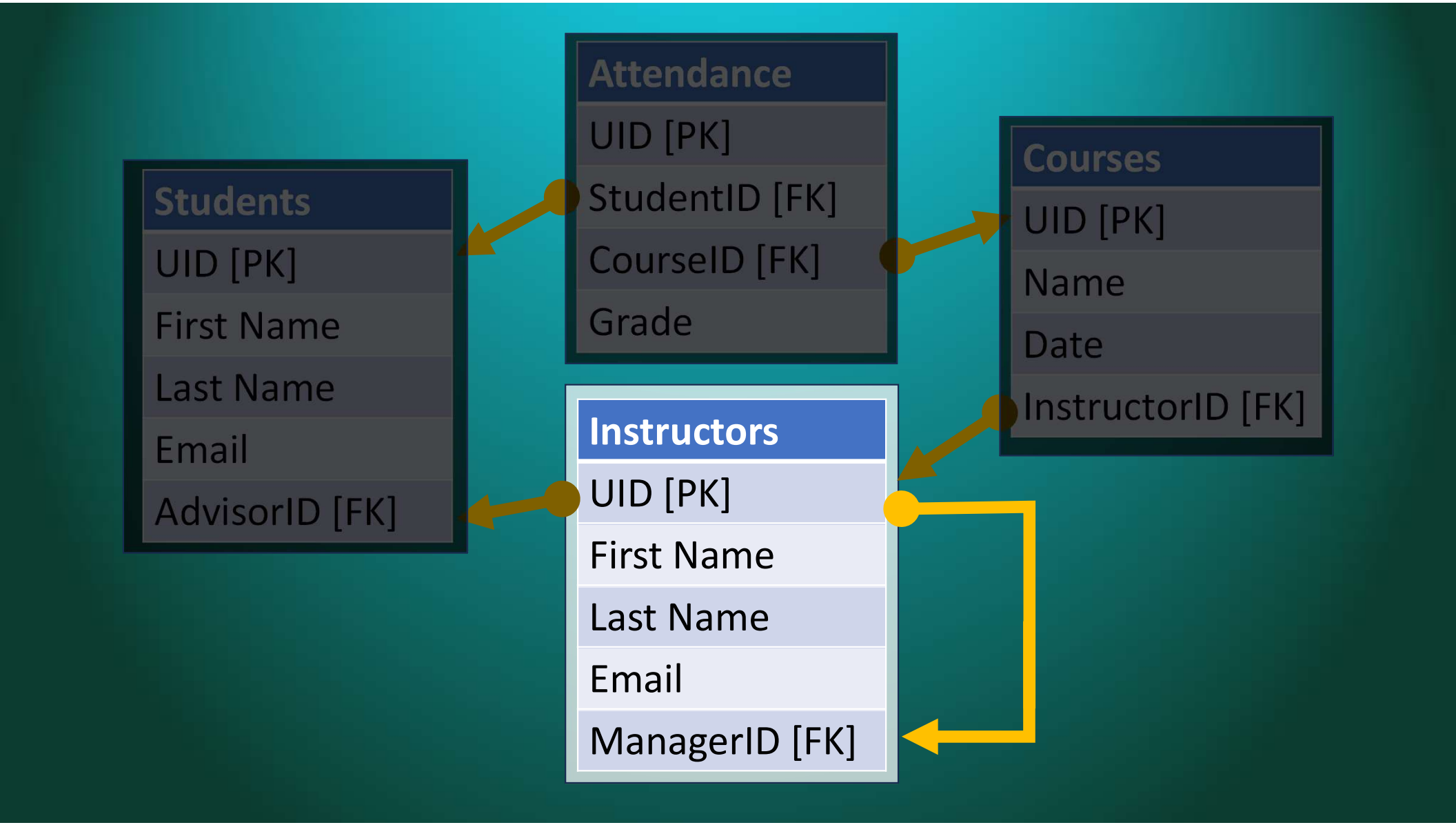
Many-to-Many

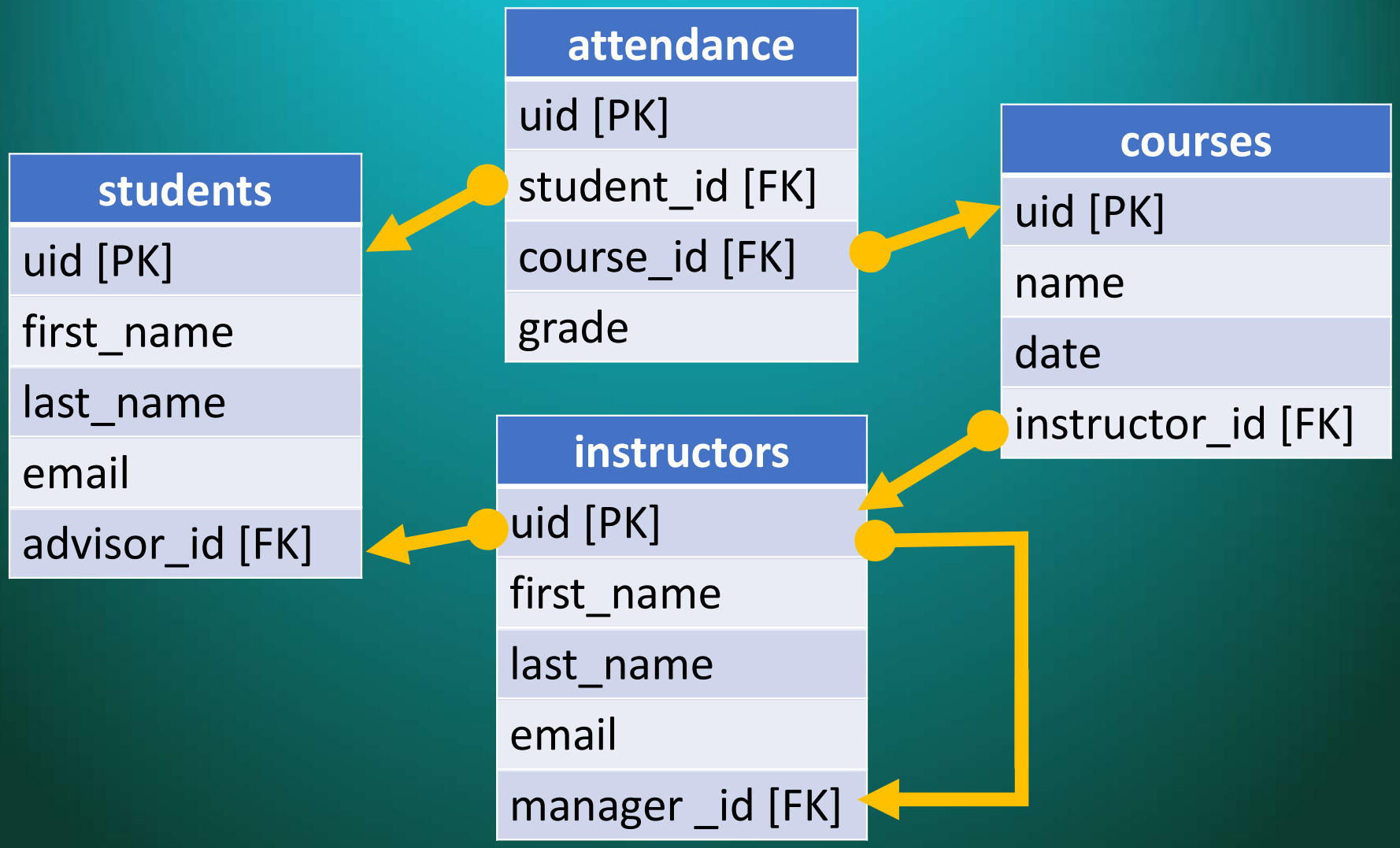
Self-Referencing


Students
UID [PK]
First Name
Last Name
Email
Attendance
UID [PK]
StudentID [FK]
CourseID [FK]
Grade
Courses
UID [PK]
Name
Date
InstructorID [FK]
Instructors
UID [PK]
First Name
Last Name
Email


Students
UID [PK]
First Name
Last Name
Email
AdvisorID [FK]
Attendance
UID [PK]
StudentID [FK]
CourseID [FK]
Grade
Courses
UID [PK]
Name
Date
InstructorID [FK]
Instructors
UID [PK]
First Name
Last Name
Email


Attendance
UID [PK]
StudentID [FK]
CourseID [FK]
Grade
Students
UID [PK]
First Name
Last Name
Email
AdvisorID [FK]
Courses
UID [PK]
Name
Date
InstructorID [FK]
Instructors
UID [PK]
First Name
Last Name
Email


Students
UID [PK]
First Name
Last Name
Email
AdvisorID [FK]
Attendance
UID [PK]
StudentID [FK]
CourseID [FK]
Grade
Courses
UID [PK]
Name
Date
InstructorID [FK]
Instructors
UID [PK]
First Name
Last Name
Email
ManagerID [FK]


attendance
uid [PK]
student_id [FK]
course_id [FK]
grade
students
uid [PK]
first_name
last_name
email
advisor_id [FK]
courses
uid [PK]
name
date
instructor_id [FK]
instructors
uid [PK]
first_name
last_name
email
manager _id [FK]

Constraints

Constraints

- Primary Key*
- Foreign Key
- Auto Increment

* Not Null and Unique

NOT NULL

```
mysql> DESCRIBE instructors;
+------------+-------------+------+-----+---------+----------------+
| Field      | Type        | Null | Key | Default | Extra          |
+------------+-------------+------+-----+---------+----------------+
| uid        | int         | NO   | PRI | NULL    | auto_increment |
| first_name | varchar(64) | YES  |     | NULL    |                |
| last_name  | varchar(64) | YES  |     | NULL    |                |
| email      | varchar(64) | YES  |     | NULL    |                |
+------------+-------------+------+-----+---------+----------------+
```

NULL means the value is not set.

NULL is different than
zero (0) or
an empty string ("")

```sql
INSERT INTO instructors(first_name) VALUES ("Mike");
```

```
INSERT INTO instructors VALUES ();
```

```sql
CREATE TABLE instructors(
     uid INT PRIMARY KEY AUTO_INCREMENT,
     first_name VARCHAR(64),
     last_name VARCHAR (64),
     email VARCHAR (64)
);
```

```sql
CREATE TABLE instructors(
    uid INT PRIMARY KEY AUTO_INCREMENT,
    first_name VARCHAR(64) NOT NULL,
    last_name VARCHAR(64) NOT NULL,
    email VARCHAR(64)
);
```

NULL Values

Comparisons: A NULL value comparison is undefined. Use IS NULL or IS NOT NULL.

Arithmetic: All arithmetic operations involving NULL result in NULL.

Logic: The NULL input is ignored.

Aggregate Functions: Sum, Avg, etc. NULL values are not included.

UNIQUE

```sql
CREATE TABLE instructors(
    uid INT PRIMARY KEY AUTO_INCREMENT,
    first_name VARCHAR(64) NOT NULL,
    last_name VARCHAR(64) NOT NULL,
    email VARCHAR(64) UNIQUE
);
```

```sql
ALTER TABLE instructors
MODIFY COLUMN email VARCHAR(64) UNIQUE;
```

DEFAULT

```sql
CREATE TABLE instructors(
    uid INT PRIMARY KEY AUTO_INCREMENT,
    first_name VARCHAR(64),
    last_name VARCHAR(64),
    tenure INT DEFAULT 0,
    email VARCHAR(64) UNIQUE
);
```

```sql
ALTER TABLE instructors
ADD COLUMN tenure DEFAULT 0;
```

CHECK

```sql
CREATE TABLE instructors(
    uid INT PRIMARY KEY AUTO_INCREMENT,
    first_name VARCHAR(64),
    last_name VARCHAR(64),
    tenure INT DEFAULT 0,
    email VARCHAR(64) UNIQUE,
    CHECK (tenure >= 0)
);
```

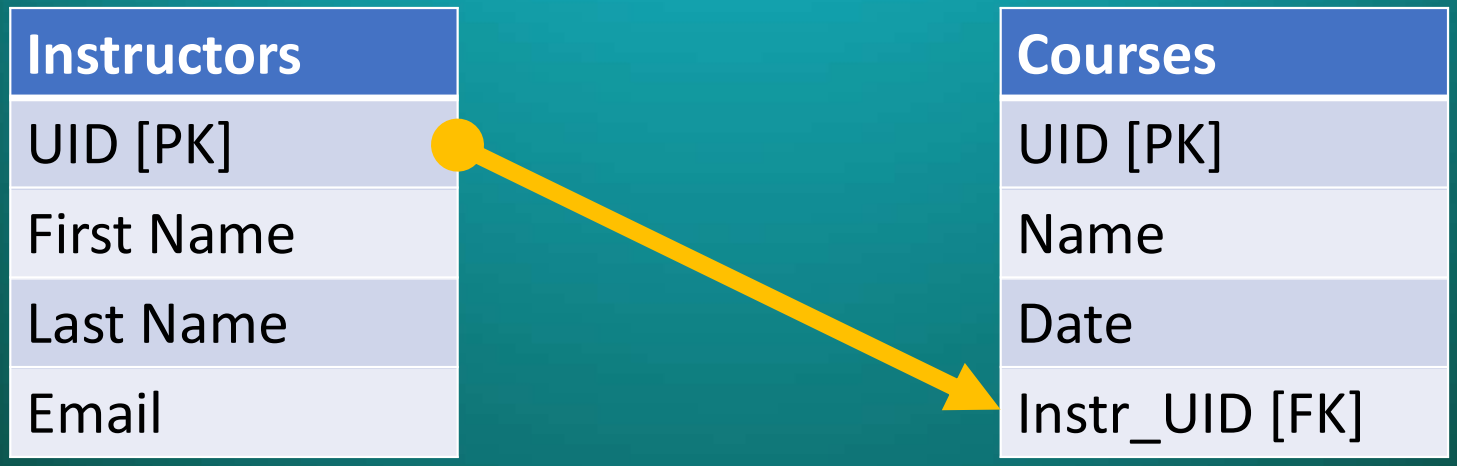
```sql
ALTER TABLE instructors
ADD CHECK (tenure >= 0);
```

CONSTRAINT

```sql
CREATE TABLE instructors(
    uid INT PRIMARY KEY AUTO_INCREMENT,
    first_name VARCHAR(64),
    last_name VARCHAR(64),
    tenure INT DEFAULT 0,
    email VARCHAR(64) UNIQUE,
    CONSTRAINT tenure_positive CHECK (tenure >= 0)
);
```

Joins


INNER JOIN
INNER JOIN
LEFT JOIN
RIGHT JOIN
CROSS JOIN
SQL
Join
FULL
MATTLE
JOIN
RIGHT
SQLE

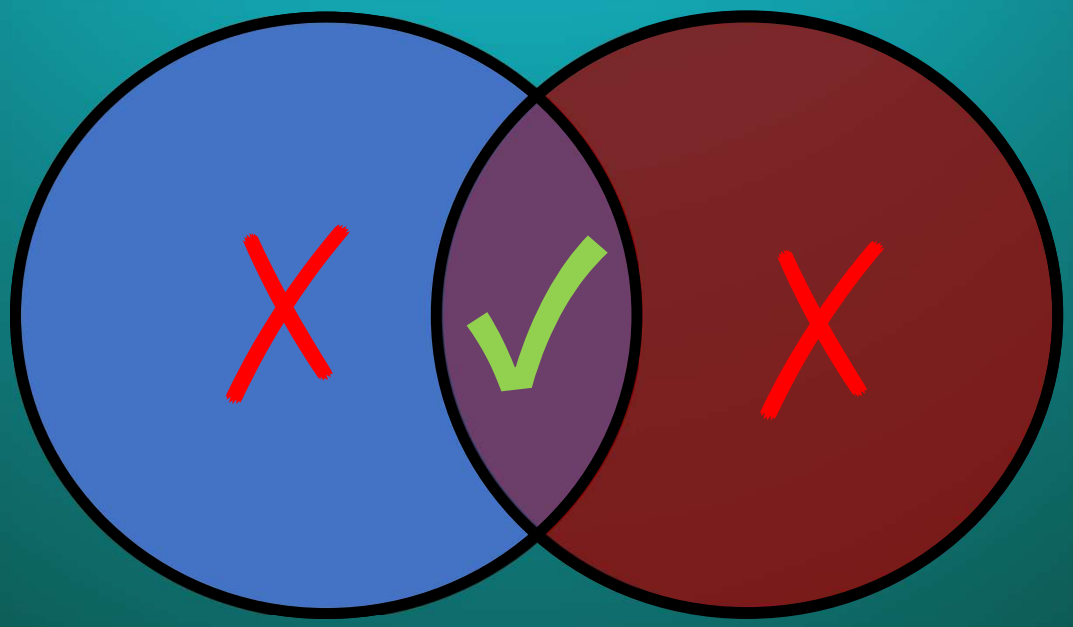

Instructors
UID [PK]
First Name
Last Name
Email
Courses
UID [PK]
Name
Date
Instr_UID [FK]

INNER JOIN

(Old Syntax)

```
SELECT * FROM instructors, courses
WHERE instr_uid = instructors.uid;
```

INNER JOIN

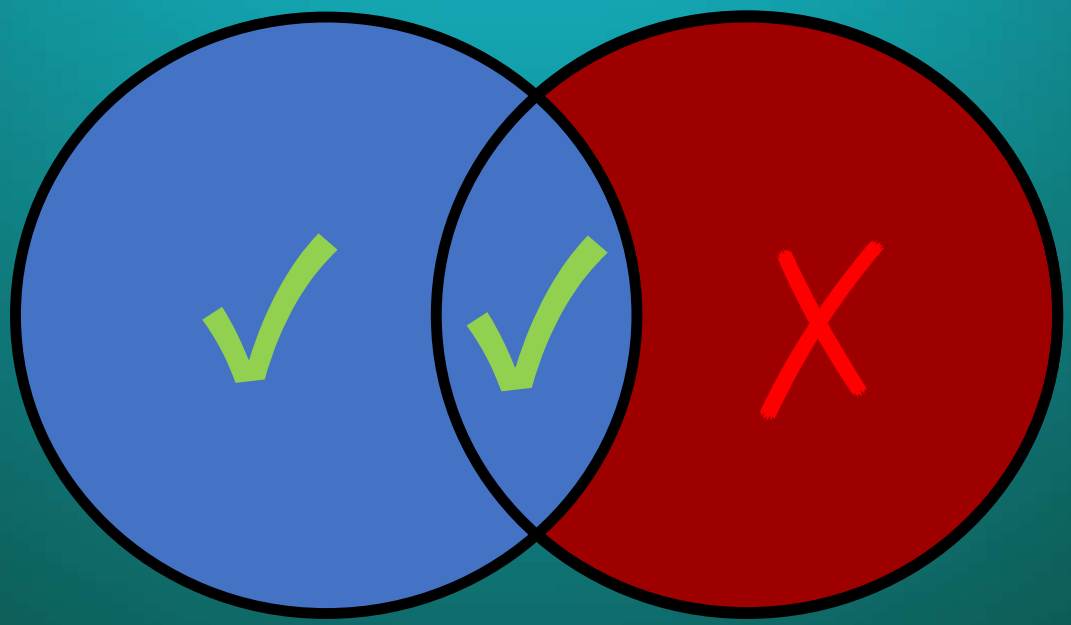


INNER JOIN

(Proper Syntax)

```
SELECT * FROM
instructors JOIN courses
ON instr_uid = instructors.uid;
```

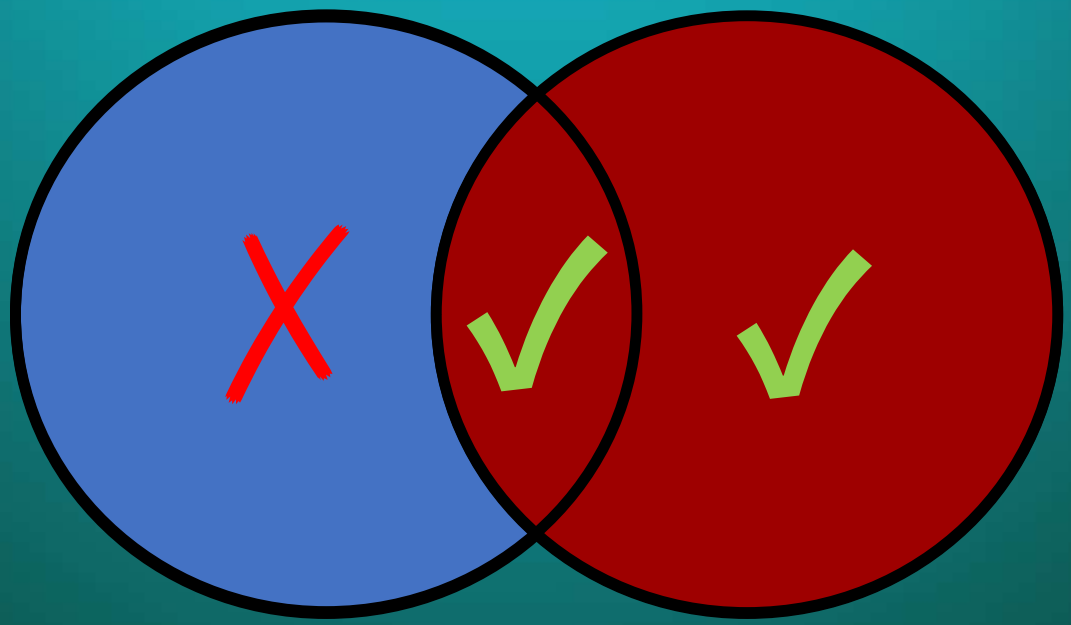
LEFT JOIN



LEFT JOIN

```
SELECT * FROM
instructors LEFT JOIN courses
ON instr_uid = instructors.uid;
```

RIGHT JOIN



RIGHT JOIN

```
SELECT * FROM
instructors RIGHT JOIN courses
ON instr_uid = instructors.uid;
```