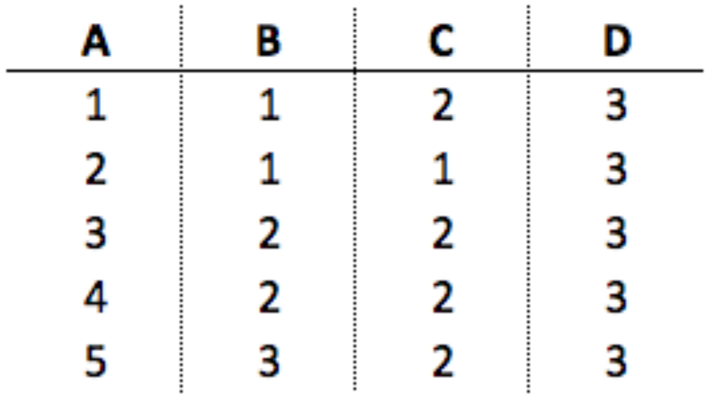
UNIVERSITY *of* WASHINGTON

Week 7: Final ER + SQL Programming

AGENDA

> FD and ERD Review
> ER Translation
> SQL Programming

FD Review



A	B	C	D
1	1	2	3
2	1	1	3
3	2	2	3
4	2	2	3
5	3	2	3

? B -> A
? C->D
? B -> C

? A -> BCD

FD Review

A	B	C	D
1	1	2	3
2	1	1	3
3	2	2	3
4	2	2	3
5	3	2	3

F B -> A
T C->D
F B -> C

T A -> BCD **A is a key because $\{A\}^+ = \{A,B,C,D\}$**

Closure

R (A, B, C, D, E, F, G)

G -> A

B -> CD

A -> B

F -> E

Calculate $\{G,E\}^+$

Closure

R (A, B, C, D, E, F, G)

G -> A

B -> CD

A -> B

F -> E

Calculate **{G,E}**$^+$

X = {G, E}

Closure

R (A, B, C, D, E, F, G)

G -> A

B -> CD

A -> B

F -> E

Calculate $\{G,E\}^+$

X = {G, E}

X = {G, E, A}

Closure

R (A, B, C, D, E, F, G)

*G -> A

B -> CD

A -> B

F -> E

Calculate $\{G,E\}^+$

X = {G, E}

X = {G, E, A}

X = {G, E, A, B}

Closure

R (A, B, C, D, E, F, G)

*G -> A

B -> CD

*A -> B

F -> E

Calculate $\{G,E\}^+$

X = {G, E}

X = {G, E, A}

X = {G, E, A, B}

X = {G, E, A, B, C, D}

Closure

R (A, B, C, D, E, F, G)

*G -> A

*B -> CD

*A -> B

F -> E

X = {G, E}
X = {G, E, A}
X = {G, E, A, B}
X = {G, E, A, B, C, D}

Calculate $\{G,E\}^+$ = {A, B, C, D, G, E}

Closure

R (A, B, C, D)

A -> B

B -> C

C -> A

Find the key?

Closure

R (A, B, C, D)

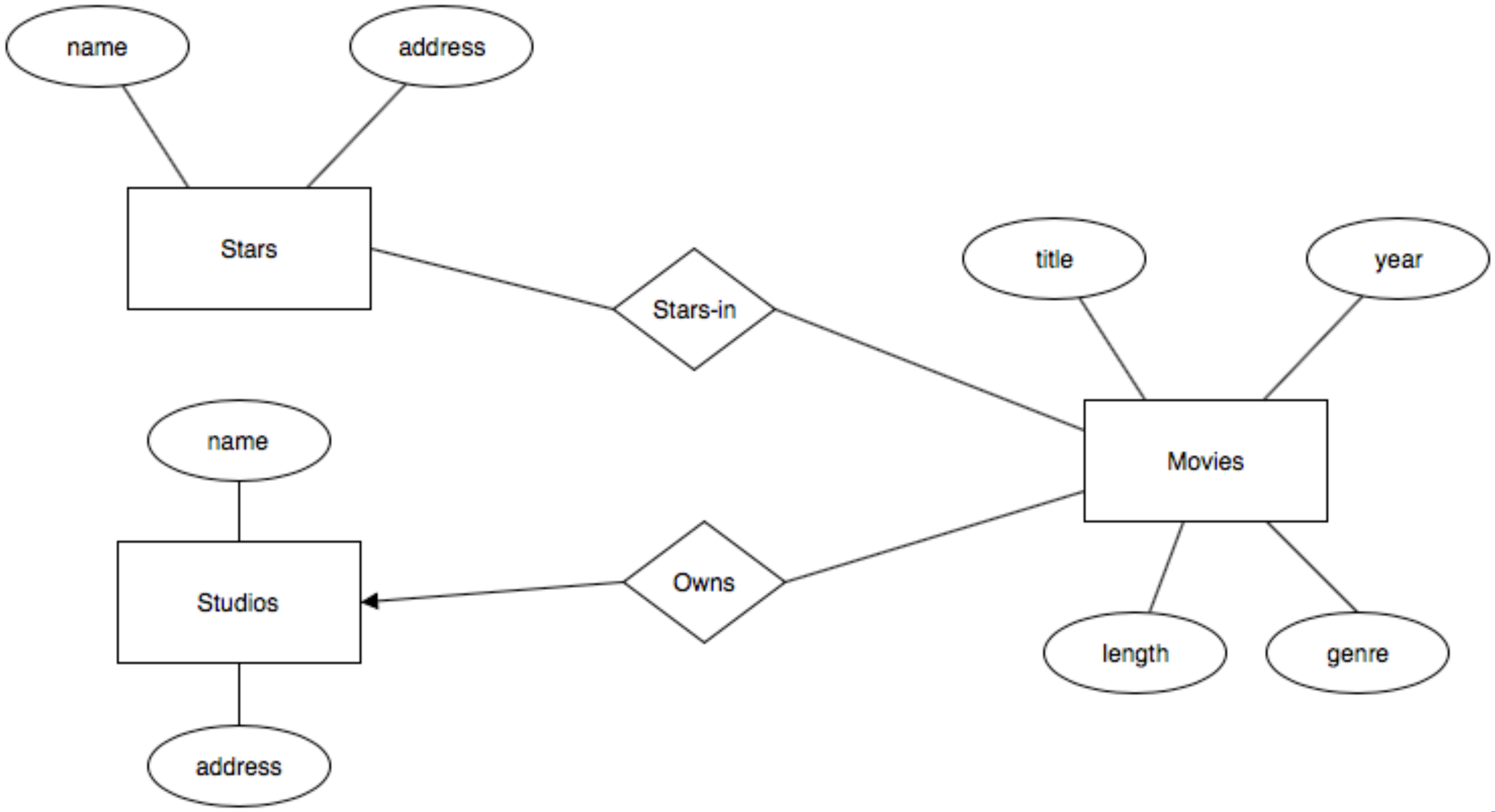
A -> B

B -> C

C -> A

Find the key? $\{A, D\}^+ = \{A, B, C, D\}$

Entity Relationship Diagrams



UNIVERSITY *of* WASHINGTON

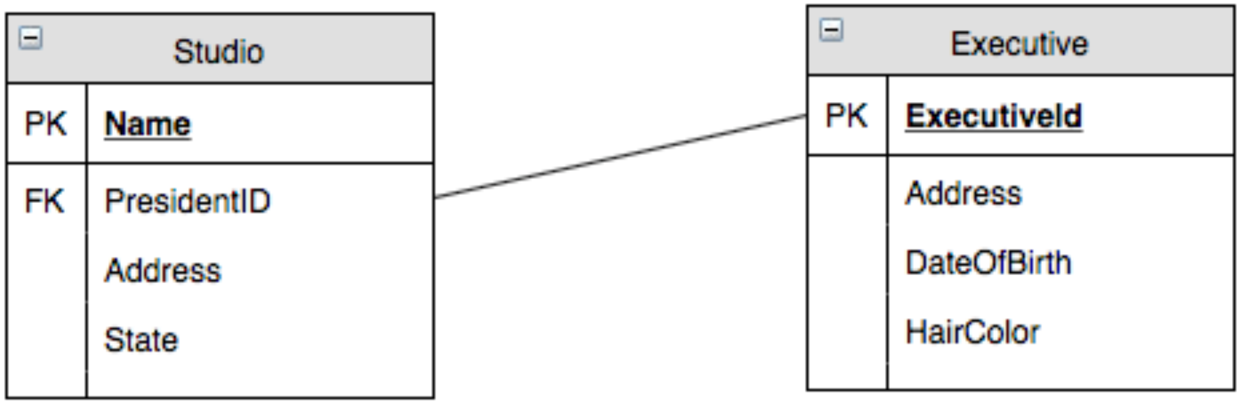
ERD to Table Translation

A quick review on KEYS

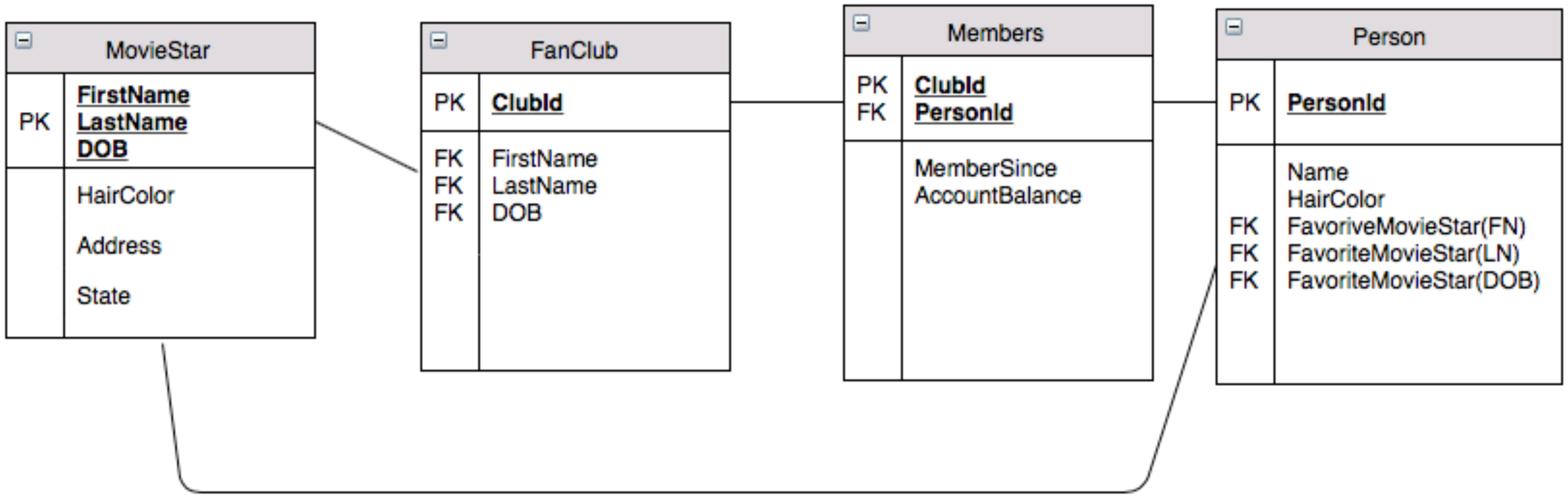
- Primary Keys
  - Unique identifiers of the relation in question
  - Auto generated IDs, Employee Numbers, SSNs (bad), email address.
  - Natural and artificial
- Foreign Keys
  - Identifiers on a dependent relationship (usually on the "many" side of the relationship)

Can be multiple columns (composite) or single (simple)

A SIMPLE KEY EXAMPLE



More keys

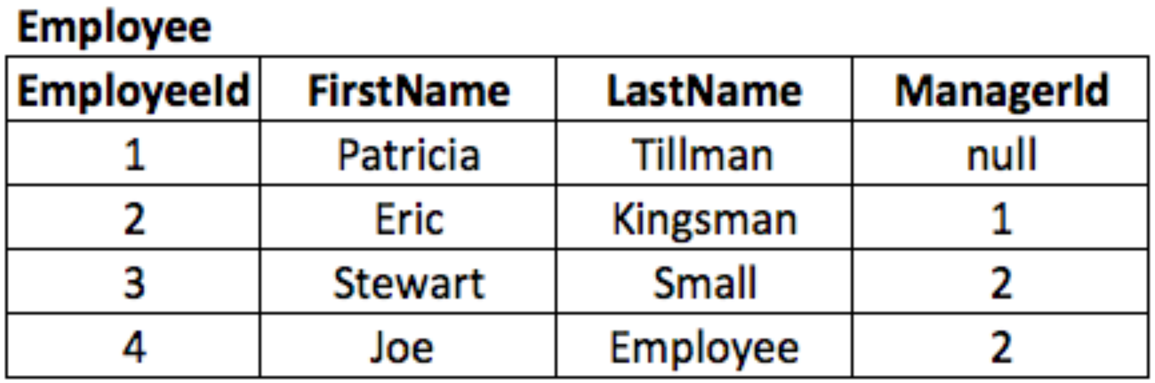


MovieStar
PK FirstName
LastName
DOB
HairColor
Address
State
FanClub
PK ClubId
FK FirstName
FK LastName
FK DOB
Members
PK ClubId
FK PersonId
MemberSince
AccountBalance
Person
PK PersonId
Name
HairColor
FK FavoriveMovieStar(FN)
FK FavoriteMovieStar(LN)
FK FavoriteMovieStar(DOB)

Database Constraints

- Domain Constraints
  - Allowable values for an attribute
- Entity Integrity
  - No Primary Key attribute may be null.
  - When would you force a Foreign Key to not be null?
- Referential Integrity
  - Ensures data quality

Domain Constraints



Employee

EmployeeId	FirstName	LastName	ManagerId
1	Patricia	Tillman	null
2	Eric	Kingsman	1
3	Stewart	Small	2
4	Joe	Employee	2

```
INSERT INTO Employee (EmployeeId, FirstName, LastName, ManagerId) VALUES
  (5, ‘New’, ‘Guy’, 6)
```

-- WITHOUT DOMAIN CONTRAINTS, THIS IS ALLOWED!

Integrity Constraints

Employee

EmployeeId	**FirstName**	**LastName**	**ManagerId**
1	Patricia	Tillman	null
2	Eric	Kingsman	1
3	Stewart	Small	2
4	Joe	Employee	2

Registration

EmployeeId	ClassId
1	2
1	3
3	3

DELETE FROM Employee WHERE EmployeeId = 1

-- What happens to the Registration Table???

Integrity Constraints

Employee

EmployeeId	**FirstName**	**LastName**	**ManagerId**
1	Patricia	Tillman	null
2	Eric	Kingsman	1
3	Stewart	Small	2
4	Joe	Employee	2

Registration

EmployeeId	ClassId
1	2
1	3
3	3

ACTION DEPENTS ON THE CONSTRAINT TYPE:

Restrict – Doesn't allow the deletion of the record. Returns constraint error

Cascade – Delete all of the dependent relations of the parent side

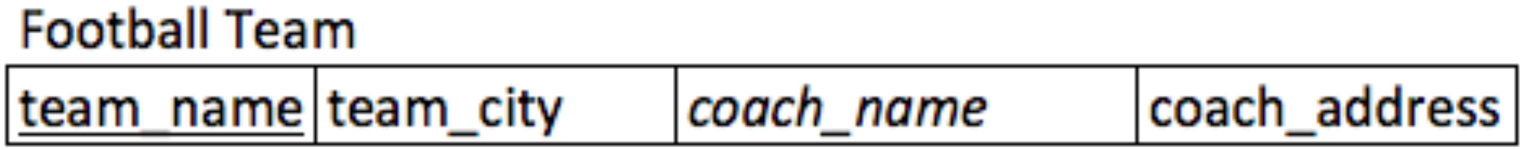
Set to Null or Set to Default – Updates dependent relations to null of default

How to model different multiplicities

- One to One Relationships
  - Most “options” to choose from…
- Imagine a 1:1 relationship between Football Team and Head Coach

Football Team: Head Coach

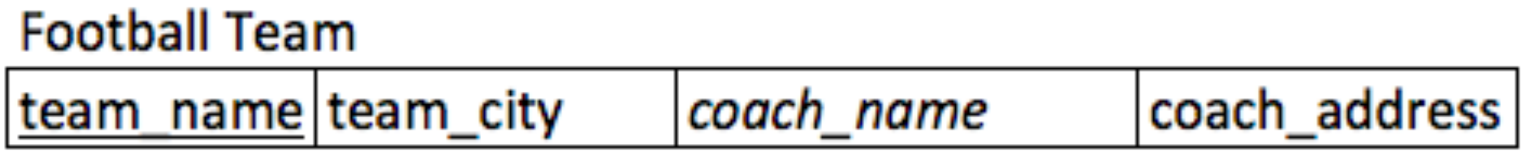
> You _*can*_ put it all into one table


Football Team

team_name	team_city	*coach_name*	coach_address

> What are the advantages?
> What are the disadvantages?

Football Team: Head Coach



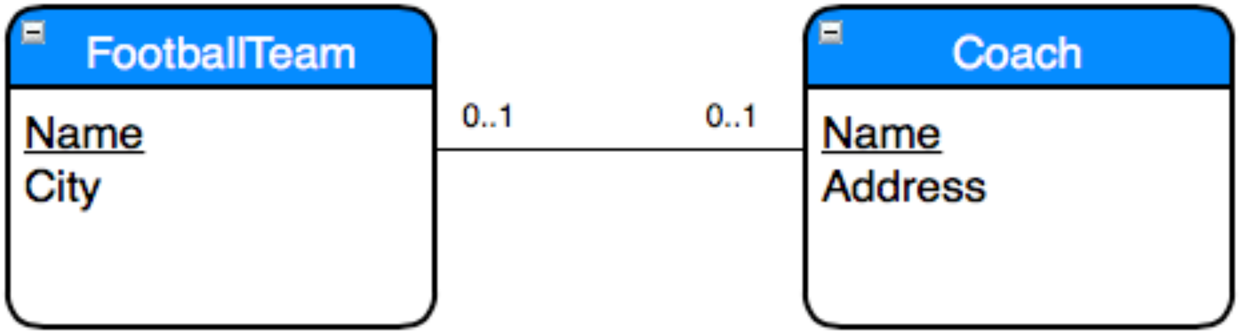
Football Team

team_name	team_city	*coach_name*	coach_address

- Advantage… no join needed. One relationship, guaranteed to be one to one.

- Disadvantage… move a coach from one team to the next?
- Extra wide table

Football Team 1:1 Head Coach



Football Team

<u>team_name</u>	team_city	*head_coach_id*

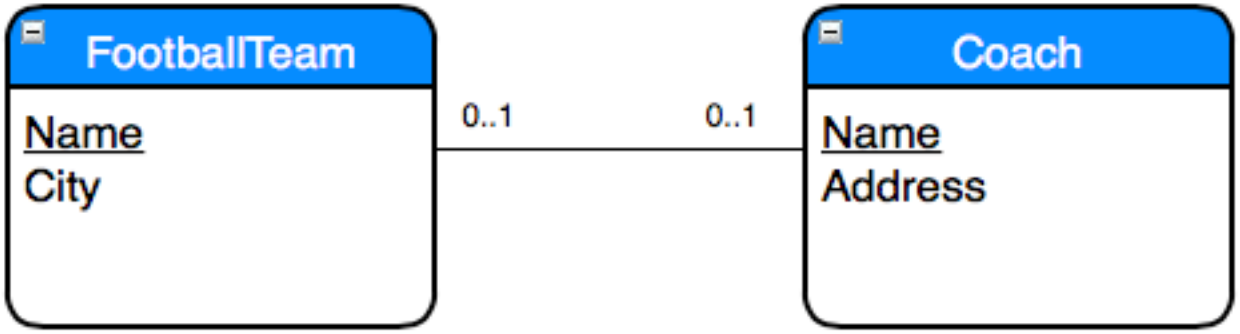
NULL / NOT NULL

UNIQUE?

Coach

<u>coach_id</u>	name	address

Football Team 1:1 Head Coach



Football Team

<u>team_name</u>	team_city	*head_coach_id*

NULL / NOT NULL

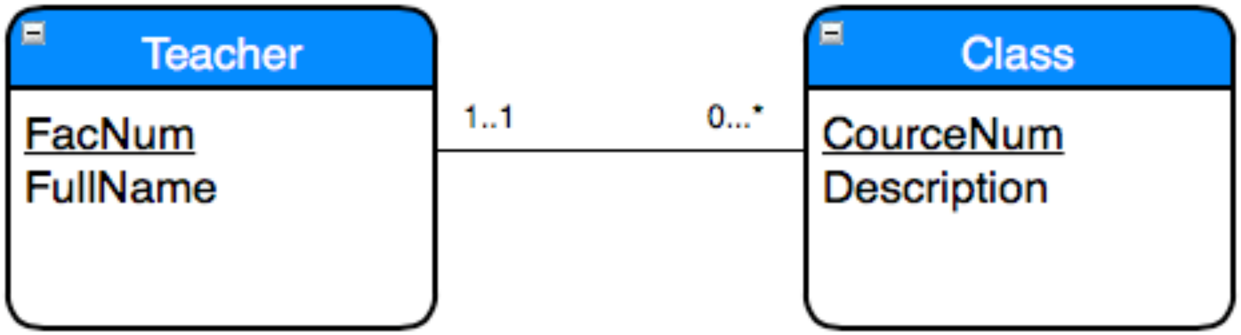
UNIQUE?

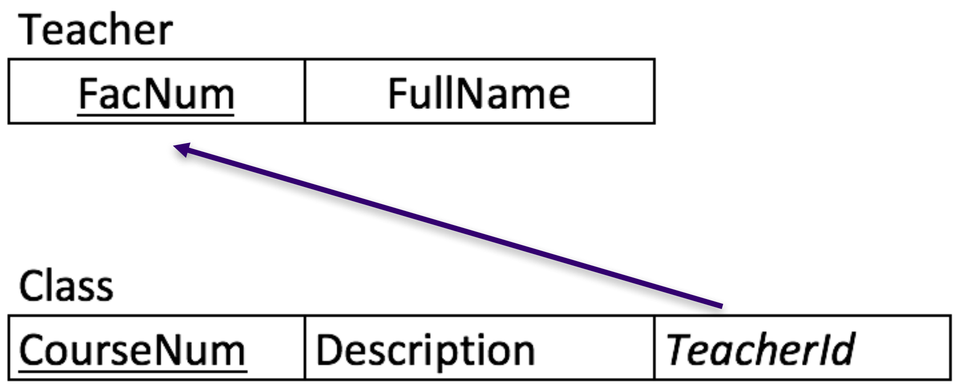
Coach

<u>coach_id</u>	name	address

Making FK unique enforces 1:1
Making FK NOT NULL forces exactly 1

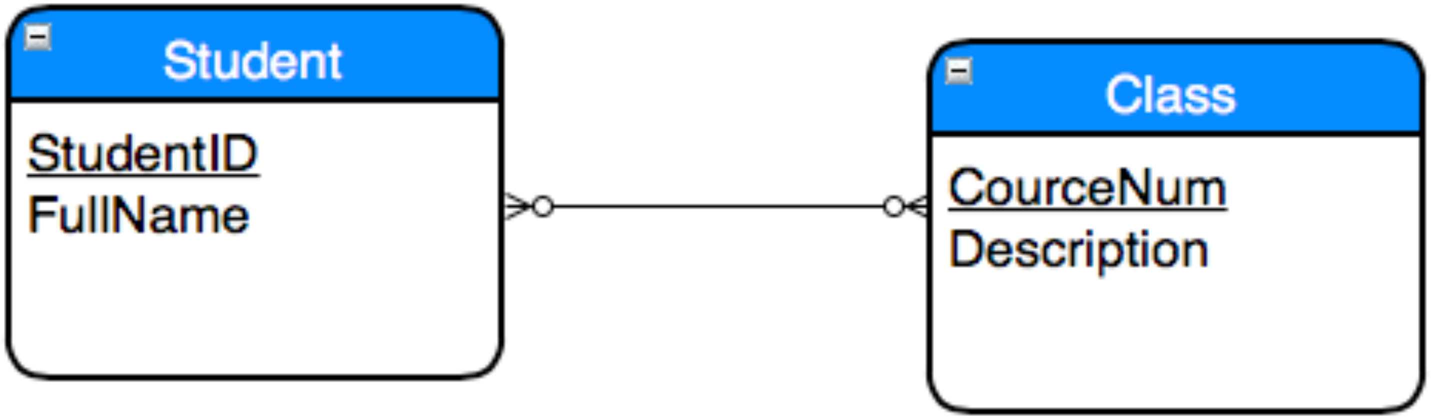
One to Many

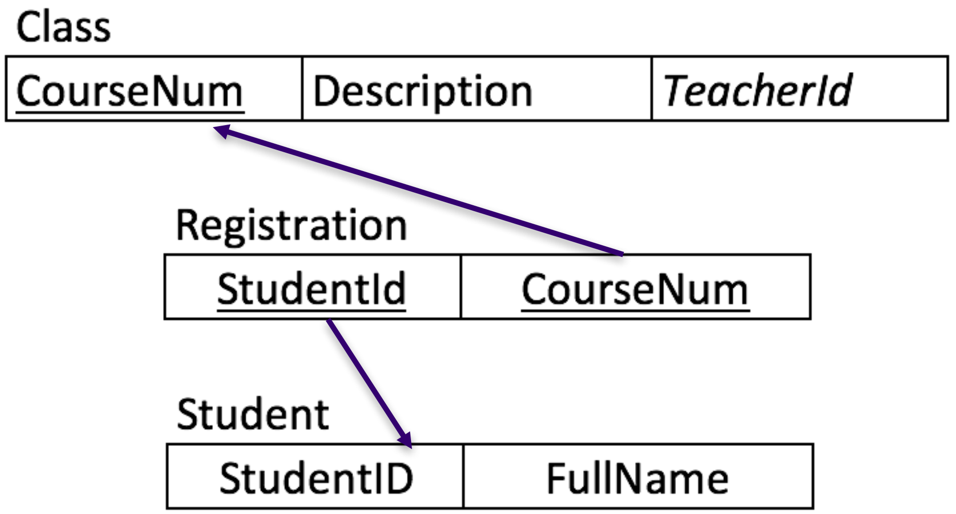




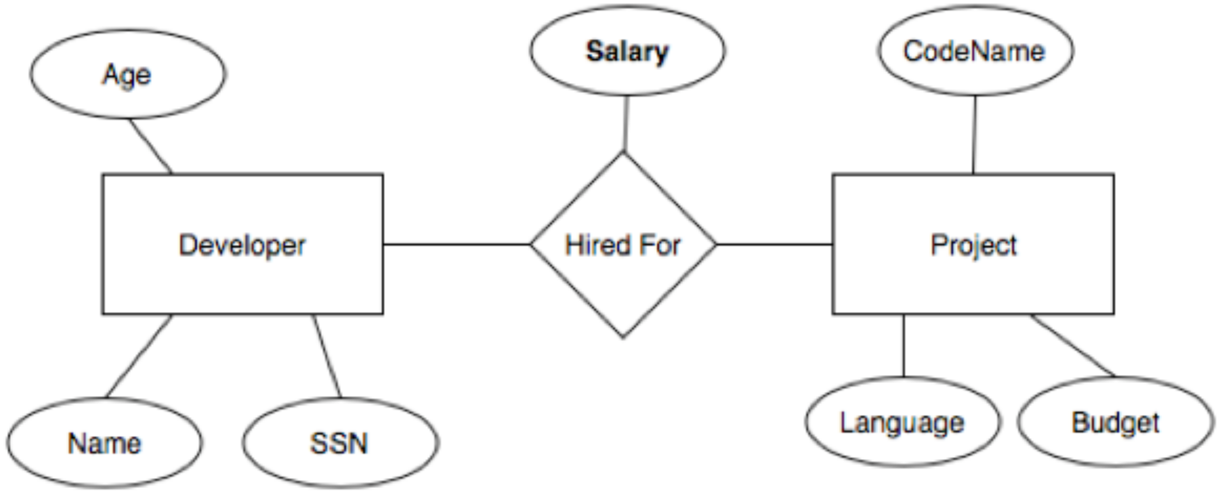
Making FK TeacherId NOT NULL enforces a required 1..1

Many to Many


Student
StudentID
FullName
Class
CourceNum
Description


Class
CourseNum
Description
TeacherId
Registration
StudentId
CourseNum
Student
StudentID
FullName

Attributes on Relationships



Developer

<u>SSN</u>	Name	Age

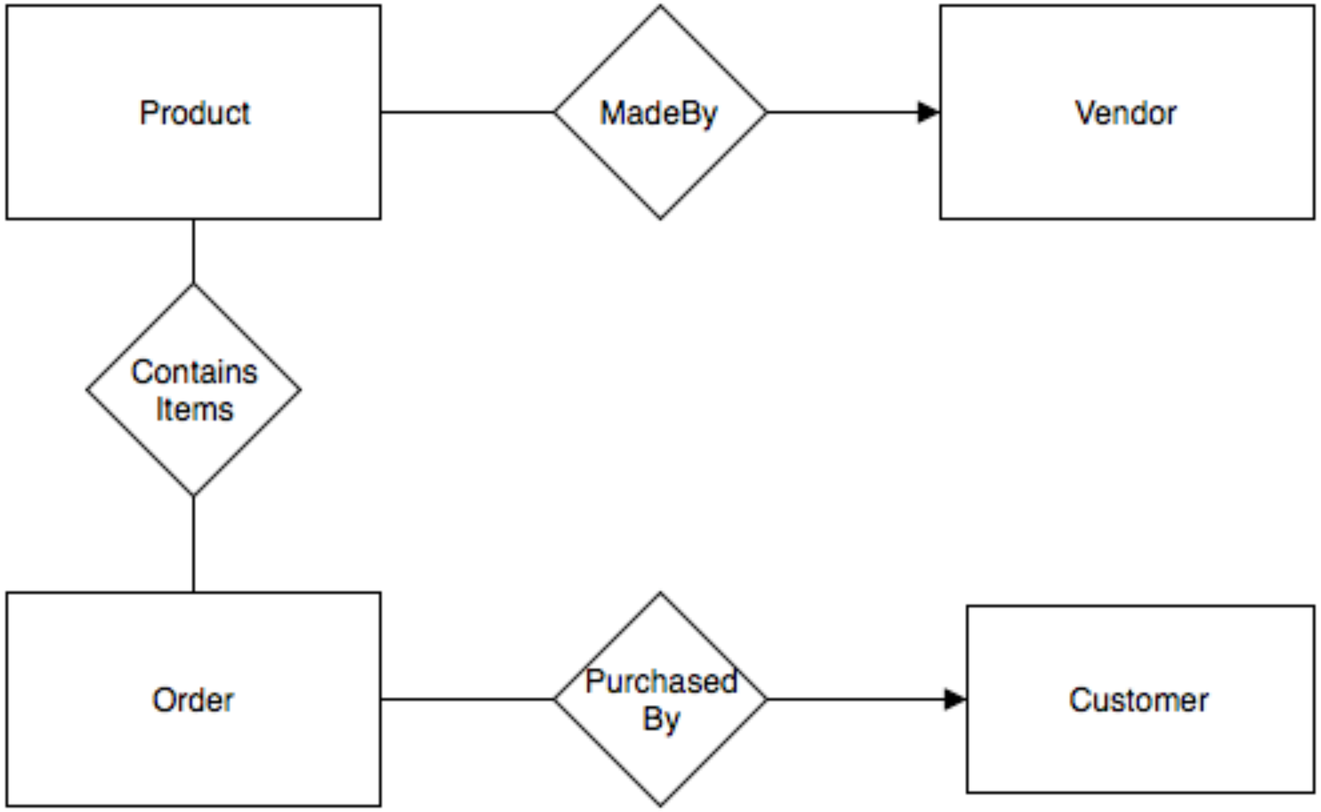
HiredFor

<u>DevId</u>	<u>ProjectId</u>	Salary

Project

<u>CodeName</u>	Language	Budget

Exercise: Convert the following ERD into Tables



Example Solution

Vendor(VendorId, name, city)

Product(ProductId, name, *VendorId*)

Customer(CustomerId, Name)

Order(OrderNumber, DateOrdered, *OrderedById*)

OrderDetail(*OrderNum*, *ProductId*, LineNumber)

UNIVERSITY *of* WASHINGTON

SQL PROGRAMMING

SQL 3-Tier Architecture

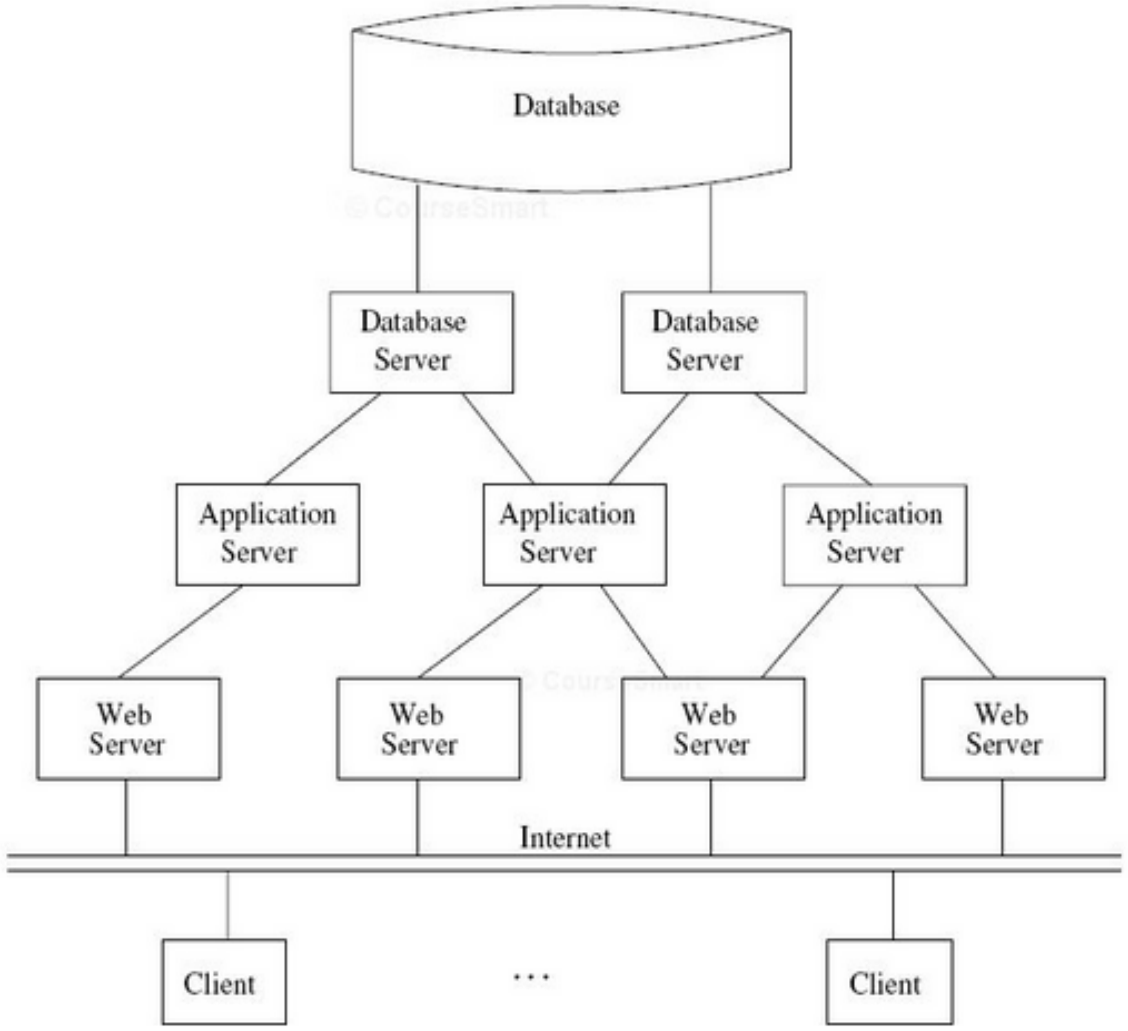
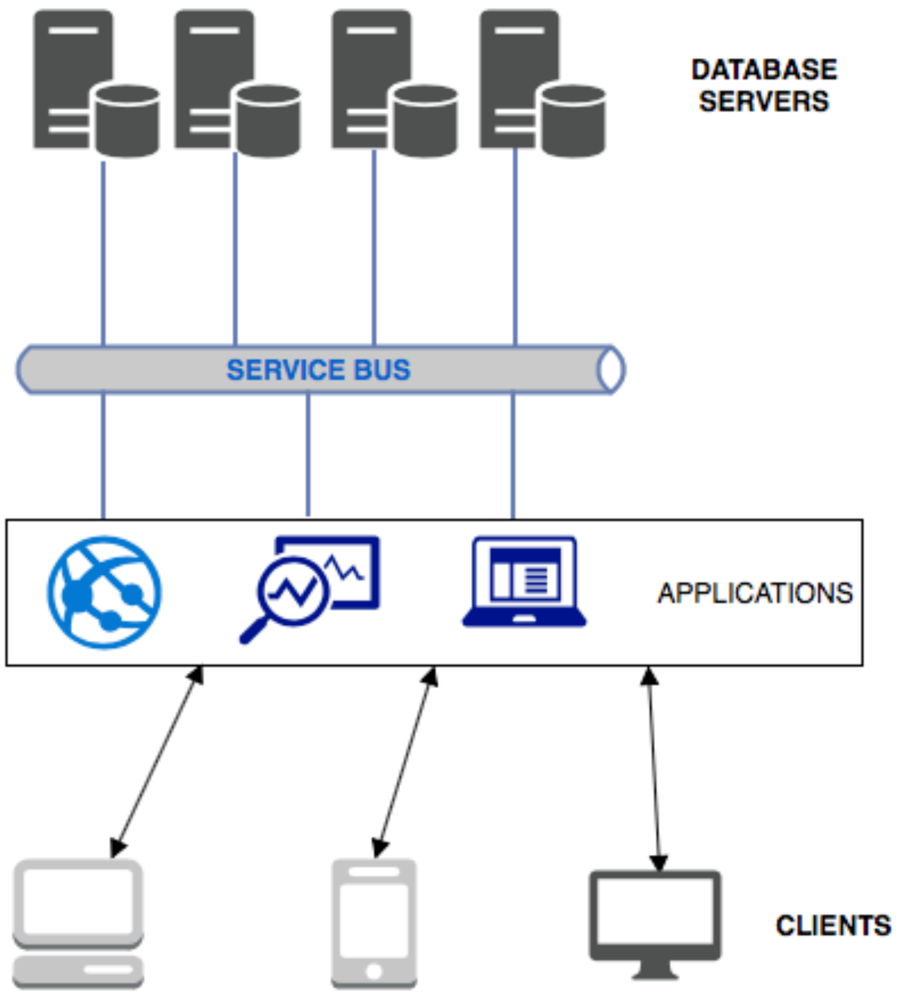


Figure 9.1: The Three-Tier Architecture

Service Oriented Architecture (Service Bus)



Connections

> Before you can do anything, you need to establish a Connection

> When you open your IDE the first thing that happens is a connection is established.

> In your programs, you need to establish a connection first.

Connection Syntax Varies WIDELY!

> **PHP using PDO**
  – $dsn = ‘mysql:host=cssgate.insttech.washington.edu;dbname=mydb'
  – $db = new PDO($dsn, $userid, $password);

> **PHP using mysqli**
  – $conn = mysqli_connect($host, $user, $pass);

> **Java with MySQL**
  – Conn = DriverManager.getConnection(‘jdbc:mysql://hostname’)

Connection Behavior

- Remains connected until disconnect is sent, or idle timeout is exceeded.
- Connections are costly to spin up, but not secure to leave open.
- Web Apps should close connection at script end

Sessions vs Connections

- A connection consists of sessions.
- One connection can have multiple sessions
  - Each tab of your GUI is usually establishes a session
- Each session has attributes such as default database: “Use dbname” is a session command

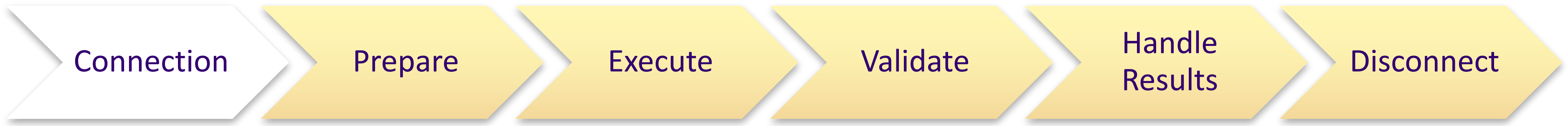
Call Level Interface

> Libraries for inserting SQL into your host programming language (java, php, etc).

> Enables you to embed SQL statements into your program

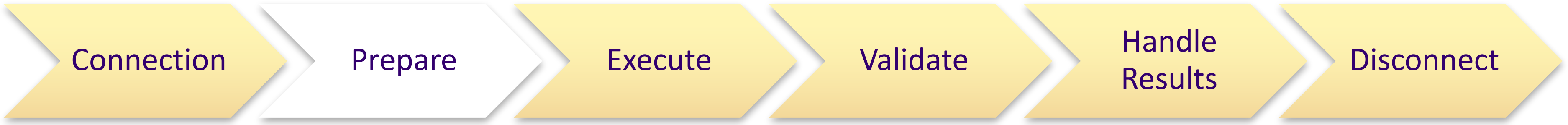
> Syntax and usage also varies widely between interfaces.

Database Query Execution Steps



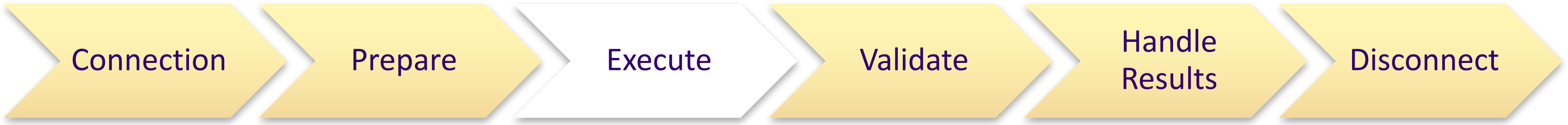
```
$db = new PDO($dsn, $userid, $password); // CATCH EXCEPTIONS
```

Database Query Execution Steps



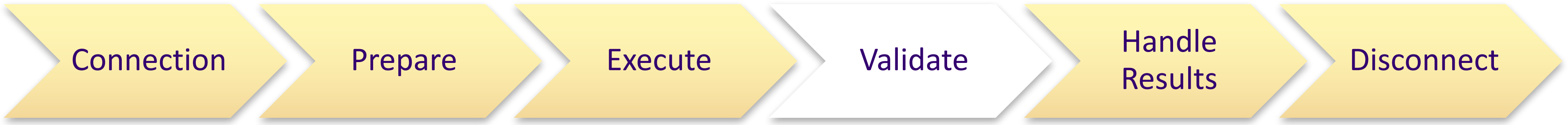
```
$db = new PDO($dsn, $userid, $password);

$q = $db->prepare('SELECT * FROM table
                    WHERE col1 = :var1
                      AND col2 = :var2');
```

Database Query Execution Steps



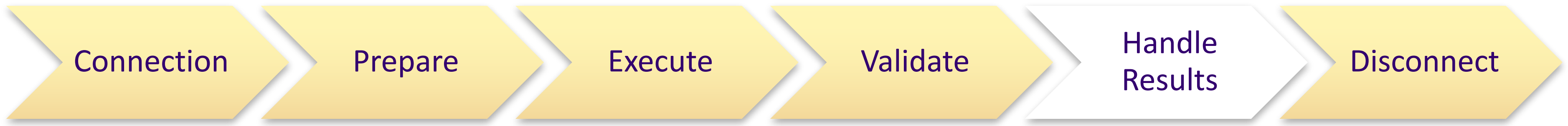
```
$db = new PDO($dsn, $userid, $password);

$q = $db->prepare('SELECT * FROM table
                    WHERE col1 = :var1
                      AND col2 = :var2');

$r = $q->execute(array(':course' => $var1,
                       ':uid’    => $var2));
```

Database Query Execution Steps



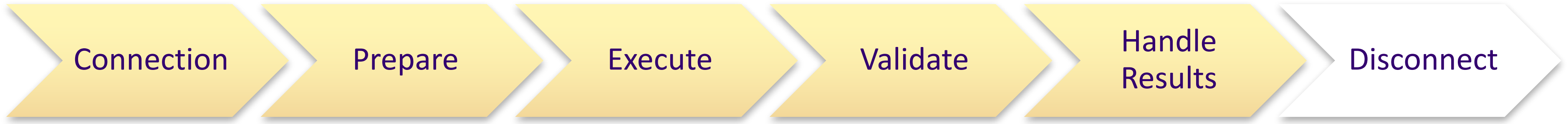
```
$db = new PDO($dsn, $userid, $password);

$q = $db->prepare('SELECT * FROM table
                     WHERE col1 = :var1
                       AND col2 = :var2');

$r = $q->execute(array(':course' => $var1,
                         ':uid’    => $var2));


$message = ($r) ? "Success" : die() ;
```

Database Query Execution Steps



```php
foreach($r as $row) { echo $row['colname']; }
```

Database Query Execution Steps



```
foreach($q as $row) { echo $row['colname']; }


$q  = NULL;
$db = NULL;  // MAKE SURE TO KILL ALL OBJECTS
```

READ DETAILS FOR BEST PRACTICES HERE:
http://php.net/manual/en/pdo.connections.php

Persisted Stored Modules (PSM)

- Stored Procedures
  - Stored and complied scripts that can be called as part of the database schema, can return multiple datasets
- Functions
  - Stored code (like stored procedures) but are used within other queries and Stored Procedures

```
CREATE FUNCTION mysqrt(a int)
    RETURNS double
    RETURN POW(a,0.5);

SELECT mysqrt(101);
```

Persisted Stored Modules (PSM)

- Stored Procedures
  - Stored and complied scripts that can be called as part of the database schema, can return multiple datasets
- Functions
  - Stored code (like stored procedures) but are used within other queries and Stored Procedures

```
CREATE FUNCTION mysqrt(a int)
   RETURNS double
   RETURN POW(a,0.5);

SELECT mysqrt(101);
```

A Function Example

```
DELIMITER //
CREATE FUNCTION get_last_completed_class(id int)
  RETURNS VARCHAR(8)
  BEGIN
    DECLARE cname VARCHAR(8);
    SET cname = ( SELECT course_name
                    FROM transcript
                   WHERE student_id = id
                ORDER BY completion_date DESC
                  LIMIT 1);
    RETURN cname;
    END //
DELIMITER ;
```

A Function Example

```
SELECT s.name
     , get_last_completed_class(s.student_id) as "last_class"
 FROM student s;
```

Stored Procedures

- Enables users to put database related logic INTO the database itself.
  - Most DMBS systems implement Turing Complete languages for stored procedures (reclusiveness, etc.)
- Abstracts the database implementation from the Application developer

Stored Procedures

- EXAMPLE:
  - CALL GET_COURSES_FOR_STUDENT(int)
  - (implemented by ‘select course_id FROM courses WHERE is_active = 1’ where not exists (select course_id from registration…)
- What if additional flags are added?

Stored Procedures

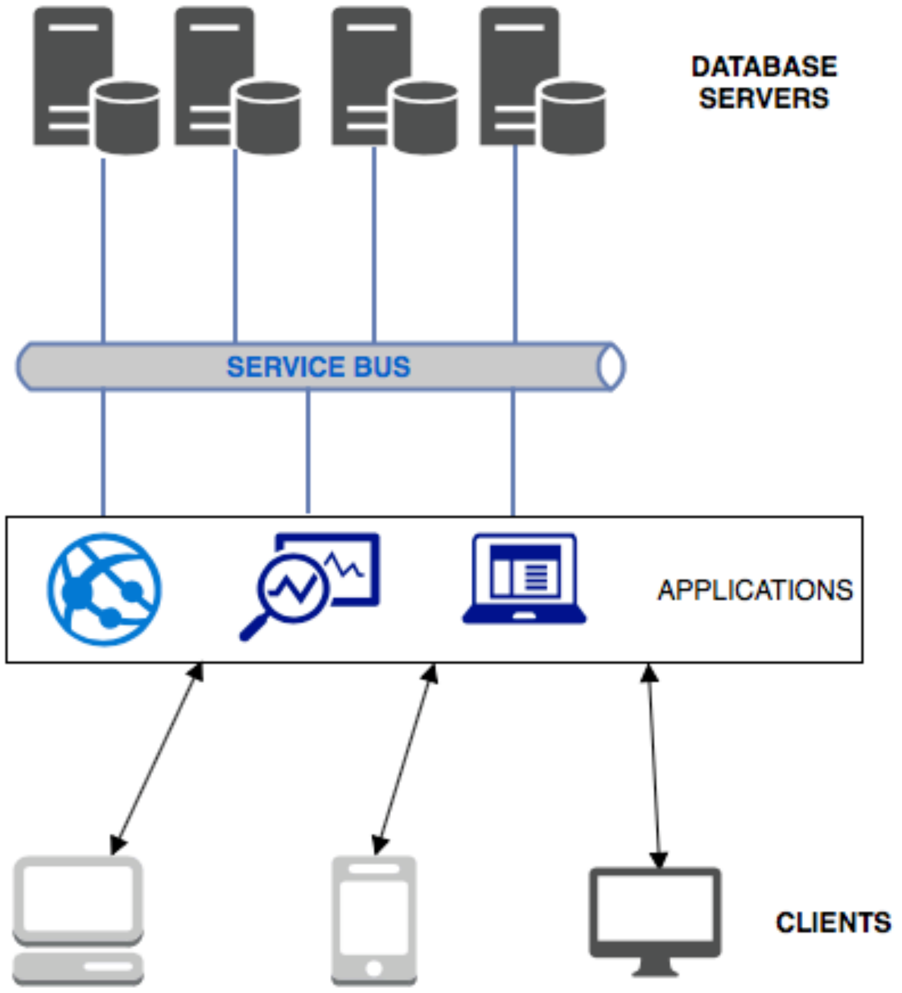
- EXAMPLE 2:
  - UPDATE_HOME_ADDRESS(student, new_address)
  - Implemented as
    - Insert into audit (table, column, old value, new value)…
    - Update student set address = new_address)

Stored Procedures Benefits

> Data logic and business rules can be encapsulated so that users can access data and objects only in ways that developers and database administrators intend.
> Parameterized stored procedures that validate all user input can be used to thwart SQL injection attacks. If you use dynamic SQL, be sure to parameterize your commands, and never include parameter values directly into a query string.
> Ad hoc queries and data modifications can be disallowed. This prevents users from maliciously or inadvertently destroying data or executing queries that impair performance on the server or the network.
> Errors can be handled in procedure code without being passed directly to client applications. This prevents error messages from being returned that could aid in a probing attack. Log errors and handle them on the server.
> Stored procedures can be written once, and accessed by many applications.
> Client applications do not need to know anything about the underlying data structures. Stored procedure code can be changed without requiring changes in client applications as long as the changes do not affect parameter lists or returned data types.
> Stored procedures can reduce network traffic by combining multiple operations into one procedure call.

https://msdn.microsoft.com/en-us/library/bb669058

Service Oriented Architecture (Service Bus)



UNIVERSITY *of* WASHINGTON

PROJECT SAMPLE CODE