DATABASE SYSTEMS

Dr. Noha Nagy

Lecture 7

Join

SQL

- DDL
  - Create
  - Alter
  - Drop
- DML
  - Insert
  - Update
  - Delete
  - **Select**

Single Table
Multiple Tables

Example


```
SELECT  Count(ProductID) as X
FROM Orders;
```

X
8

Orders

OrderID	ProductID	Quantity
100	1	10
100	2	17
102	2	2
100	5	9
103	3	3
103	4	4
103	5	5
103	6	6

```
SELECT OrderID, Count(ProductID)
as X
FROM Orders
GROUP BY OrderID;
```

Order ID	X
100	3
102	1
103	4

```
SELECT OrderID, Count(ProductID)
as X
FROM Orders
GROUP BY OrderID
HAVING Count(productID) > 3;
```

Order ID	X
103	4

```
SELECT  Count(*) as X
FROM Orders;
```

X
8

DML
Multiple Tables

Schema

Customer

Customer_ID	Customer_Name	City	State	Postal_Code

Product

Product_ID	Product_Name	Product_Description	Product_Finish	Standard_Price

Order

Order_ID	Order_Date	Customer_ID

Order_Line

Order_ID	Product_ID	Ordered_Quantity

SELECT from Multiple Tables

Student			Grade			Course	
ID	First	Last	ID	Code	Mark	Code	Title
S103	John	Smith	S103	DBS	72	DBS	Database Systems
S103	John	Smith	S103	IAI	58	IAI	Intro to AI
S104	Mary	Jones	S104	PR1	68	PR1	Programming 1
S104	Mary	Jones	S104	IAI	65	IAI	Intro to AI
S106	Mark	Jones	S106	PR2	43	PR2	Programming 2
S107	John	Brown	S107	PR1	76	PR1	Programming 1
S107	John	Brown	S107	PR2	60	PR2	Programming 2
S107	John	Brown	S107	IAI	35	IAI	Intro to AI

Student.ID = Grade.ID

Course.Code = Grade.Code

Joins in SQL

- Connect two or more tables:

Product

PName	Price	Category	Manufacturer
Gizmo	$19.99	Gadgets	GizmoWorks
Powergizmo	$29.99	Gadgets	GizmoWorks
SingleTouch	$149.99	Photography	Canon
MultiTouch	$203.99	Household	Hitachi

Company

Cname	StockPrice	Country
GizmoWorks	25	USA
Canon	65	Japan
Hitachi	15	Japan

What is the connection between them ?

Joins

Product (pname, price, category, manufacturer)
Company (cname, stockPrice, country)

Find all products and prices under $200 manufactured in Japan; return their names and prices.

```
SELECT   pname, price
FROM     Product, Company
WHERE    manufacturer=cname AND country='Japan'
         AND price <= 200
```

Joins

Product (pname, price, category, manufacturer)
Company (cname, stockPrice, country)

Find all products under $200 manufactured in Japan; return their names and prices.

```
SELECT   pname, price
FROM     Product, Company
WHERE    manufacturer=cname AND country='Japan'
         AND price <= 200
```

Join between Product and Company

Joins

Product (pname, price, category, manufacturer)
Company (cname, stockPrice, country)

Find all products under $200 manufactured in Japan; return their names and prices.

```
SELECT   pname, price
FROM     Product, Company
WHERE    manufacturer=cname AND country='Japan'
         AND price <= 200
```

Join between Product and Company

Joins in SQL

Product

PName	Price	Category	Manufacturer
Gizmo	$19.99	Gadgets	GizmoWorks
Powergizmo	$29.99	Gadgets	GizmoWorks
SingleTouch	$149.99	Photography	Canon
MultiTouch	$203.99	Household	Hitachi

Company

Cname	StockPrice	Country
GizmoWorks	25	USA
Canon	65	Japan
Hitachi	15	Japan

```
SELECT   pname, price
FROM     Product, Company
WHERE    manufacturer=cname AND country='Japan'
         AND price <= 200
```

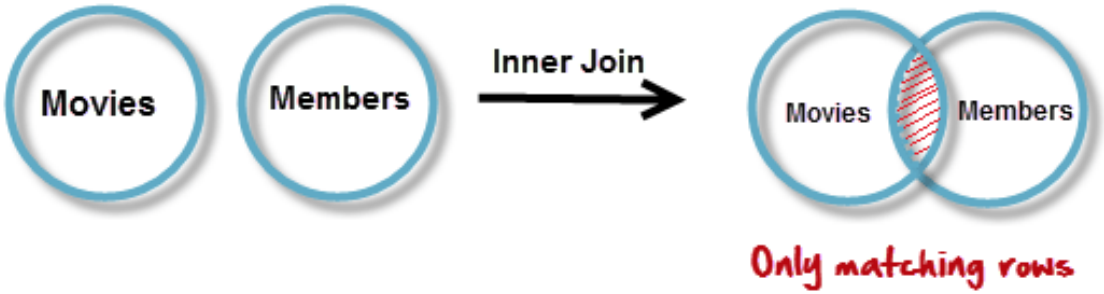
PName	Price
SingleTouch	$149.99

Join Types

- There are Four types of Joins:
  1. Inner Join
  2. Left Outer Join
  3. Right Outer Join
  4. Full Outer Join
  5. Cross Join
- To join tables, you use the cross join, inner join, left join, or right join clause for the corresponding type of join. The join clause is used in the SELECT statement appeared after the FROM clause.

INNER JOIN

- The inner JOIN is used to return rows from both tables that satisfy the given condition.
- Suppose , you want to get list of members who have rented movies together with titles of movies rented by them. You can simply use an INNER JOIN for that, which returns rows from both tables that satisfy the given conditions.



Types of Joins

- Join – a relational operation that causes two or more tables with a common domain to be combined into a single table or view
  - **Cross-join**- a join in which there is no joining condition or join condition is always true
  - **Equi-join** – a join in which the joining condition is based on equality between values in the common columns; common columns appear redundantly in the result table
  - **Natural join** – an equi-join in which one of the duplicate columns is eliminated in the result table
  - **Outer join** – a join in which rows that do not have matching values in common columns are nonetheless included in the result table (as opposed to *inner* join, in which rows must have matching values in order to appear in the result table)

The common columns in joined tables are usually the primary key of the dominant table and the foreign key of the dependent table in 1:M relationships.

INNER JOIN

Student

ID	Name
123	John
124	Mary
125	Mark
126	Jane

Enrolment

ID	Code
123	DBS
124	PRG
124	DBS
126	PRG

```
SELECT *
FROM  Student, Enrolment
Where Student.ID= Enrolment.ID
```

ID	Name	ID	Code
123	John	123	DBS
124	Mary	124	PRG
124	Mary	124	DBS
126	Jane	126	PRG

INNER JOIN

Product

name	Pid
Gizmo	123
Camera	234
OneClick	256

Purchase

Pid	store
123	Wiz
234	Ritz
234	Wiz

```
SELECT Product.name, Purchase.store
FROM   Product
  INNER JOIN Purchase
          ON Product.Pid = Pid
```

name	store
Gizmo	Wiz
Camera	Ritz
Camera	Wiz

Note: another equivalent way to write an INNER JOIN!

Tuple Variables

Get the person names and the address of the company they works fo

Person(pname, address, worksfor)

Company(cname, address)

```
SELECT   DISTINCT pname, address
FROM     Person, Company
WHERE    worksfor = cname
```

Which address ?

```
SELECT   DISTINCT Person.pname, Company.address
FROM     Person, Company
WHERE    Person.worksfor = Company.cname
```

```
SELECT   DISTINCT x.pname, y.address
FROM     Person AS x, Company AS y
WHERE    x.worksfor = y.cname
```

Exercise

Compute for each product, the total number of sales in 'September'.

Get all the products

Product(pid,name, price, categoryid(fk))

Category(Cid,Cname)

Purchase(pid(fk), month, store)

```
SELECT Product.name, count(*) as Total_sales
FROM     Product, Purchase
WHERE    Product.pid = Purchase.pid
         and  Purchase.month = 'September'
GROUP BY Product.name
```

What's wrong ?

(fk) means foreign key

Product

name	Pid
Gizmo	123
Camera	234
OneClick	256

Purchase

Pid	Month	Store
123	September	Wiz
234	September	Ritz
234	September	Wiz

```
SELECT Product.name, count(*) as total_sales
FROM   Product, Purchase
WHERE  Product.Pid = Purchase.Pid
       and  Purchase.month = 'September'
GROUP BY Product.name
```

name	Total_Sales
Gizmo	1
Camera	2

What's wrong ?
We didn't get all the products

Product

name	category
Gizmo	gadget
Camera	Photo
OneClick	Photo

Purchase

prodName	store
Gizmo	Wiz
Camera	Ritz
Camera	Wiz

name	Total sales
Gizmo	1
Camera	2
OneClick	0

Solution

Compute, for each product, the total number of sales in 'September'

Product(name, category)

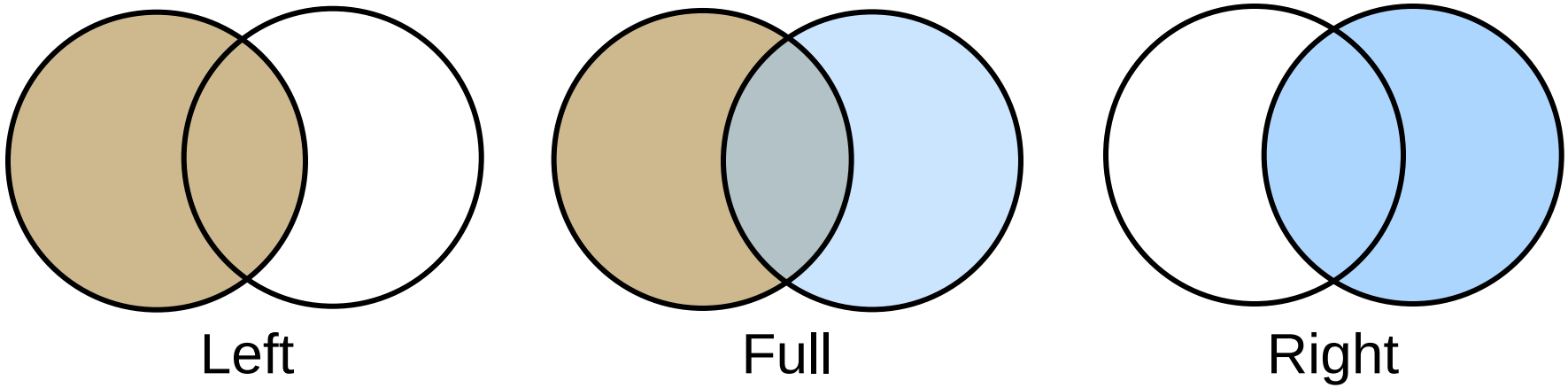
Purchase(prodName, month, store)

```
SELECT Product.name, count(*)
FROM     Product LEFT OUTER JOIN Purchase ON
                 Product. = namePurchase.prodName
               and  Purchase.month = 'September'
GROUP BY Product.name
```

Now we also get the products who sold in 0 quantity

Types of Joins

- **Outer joins**
  - return all matching rows, plus nonmatching rows from one or both tables
  - can be performed on only two tables at a time.



Outer Joins

- Left outer join:
  - Include the left tuple even if there's no match
- Right outer join:
  - Include the right tuple even if there's no match
- Full outer join:
  - Include the both left and right tuples even if there's no match

Table One

X	A
1	a
4	d
2	b

Table Two

X	B
2	x
3	y
5	v

```
select *
   from one left join two
      on one.x = two.x;
```

X	A	X	B
1	a		
2	b	2	x
4	d		

Right Join

Table Two

X	B
2	x
3	y
5	v

Table One

X	A
1	a
4	d
2	b

```
select *
    from two right join one
        on one.x = two.x;
```

X	B	X	A
		1	a
2	x	2	b
		4	d

Full Join

Table One

X	A
1	a
4	d
2	b

Table Two

X	B
2	x
3	y
5	v

```
select *
    from one full join two
        on one.x = two.x;
```

X	A	X	B
1	a		
2	b	2	x
		3	y
4	d		
		5	v

LEFT OUTER JOIN

Product

name	Pid
Gizmo	123
Camera	234
OneClick	256

Purchase

Pid	store
123	Wiz
234	Ritz
234	Wiz

```
SELECT Product.name, Purchase.store
FROM   Product
  LEFT OUTER JOIN Purchase
            ON Product.Pid = Purchase.Pid
```

name	store
Gizmo	Wiz
Camera	Ritz
Camera	Wiz
OneClick	

Right Outer Join

List all the employees and any orders they might have placed

Employee

Name	ID	Salary
Nancy	1	1000
Mark	2	1500
Ali	3	2000

Orders

OID	CID	EID	Odate
10308	1024	1	18/9/2016
10857	1055	2	3/5/2017
10698	1022	1	5/1/2017

SELECT Orders.OID, Employees.Name
FROM Orders **RIGHT JOIN** Employees
ON Orders.EID = Employees.ID;

OID	Name
	Ali
10308	Nancy
10698	Nancy
10857	Mark

Full Outer Join

List all the employees and any orders they might have placed

Employee

Name	ID	Salary
Nancy	1	1000
Mark	2	1500
Ali	3	2000

Orders

OID	CID	EID	Odate
10308	1024	1	18/9/2016
10857	1055	2	3/5/2017
10698	1022		5/1/2017

SELECT Orders.OID, Employees.Name
FROM Orders **Full Outer JOIN** Employees
ON Orders.EID = Employees.ID;

OID	Name
10308	Nancy
10857	Mark
10698	
	Ali

CROSS JOIN

Student

ID	Name
123	John
124	Mary
125	Mark
126	Jane

Enrolment

ID	Code
123	DBS
124	PRG
124	DBS
126	PRG

```
SELECT * FROM
 Student CROSS JOIN
Enrolment
```

ID	Name	ID	Code
123	John	123	DBS
124	Mary	123	DBS
125	Mark	123	DBS
126	Jane	123	DBS
123	John	124	PRG
124	Mary	124	PRG
125	Mark	124	PRG
126	Jane	124	PRG
123	John	124	DBS
124	Mary		DBS

Degree a+b
Cardinality n*m

a: number of attributes in T1
b: number of attributes in T2
n: number of records in T1
m: number of records in T2