Recitation 4:
Modeling Data as Relations

TAs: Federico Cimini & Liang-Yun Cheng
Friday, Sept 22, 2023 @ 1:45 PM | Fagin Auditorium

Penn Engineering

Housekeeping Announcements

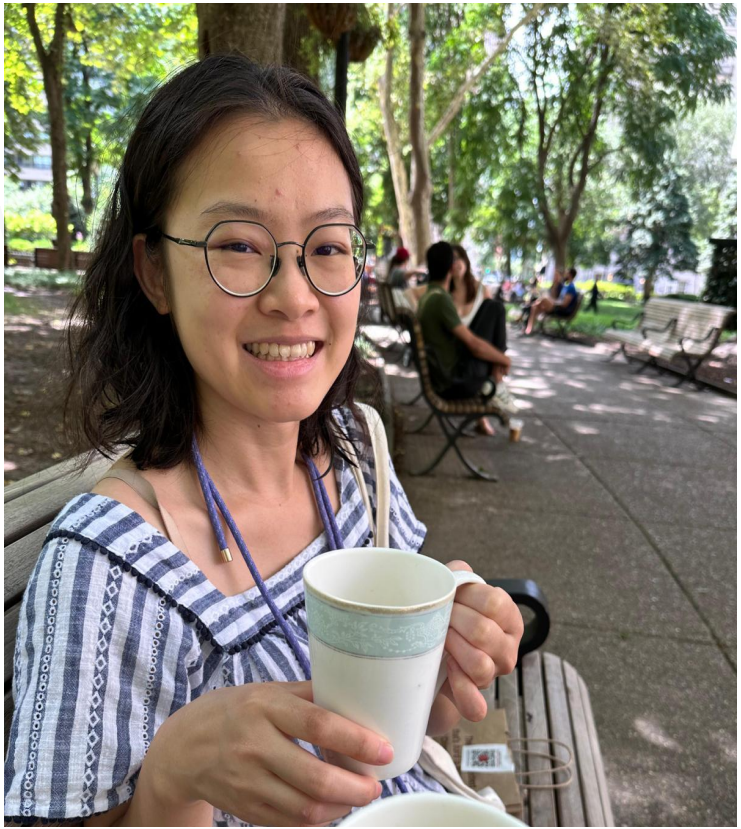
- HW 1 FAQ #65
- HW1: Updates to Autograder #452
- Ed Best Practices for Efficient Response Times #55
- Gradescope Submission Penalty Warnings #251
- HW 1 Large File Hidden Late Submissions #625

Introduction

Liang-Yun (Liang) Cheng

lycheng@seas.upenn.edu

📅 OH: Wednesday 9am - 11am



DATS 2024

Hometown: Tainan, Taiwan

Favorite spot on campus: Williams Cafe

Favorite emoji: 🙈

Hobbies: Cooking/baking, eco-friendly living, procrastinate by watching productivity videos

Fun Fact: Studied in India for 7 years

Penn Engineering

Federico Cimini

fcimini@seas.upenn.edu

📅 OH: Wednesday 9am - 11am

DATS 2024

Hometown: Buenos Aires, Argentina (go Messi!) 🇦🇷

Favorite spot on campus: Campus green terrace behind Alumni House

Favorite emoji: 🙃

Hobbies: Acting, movies, videogames

Fun Fact: I will be directing a play for the first time this fall!

Learning Objective

- Entity-Relationship (ER) Diagram
- Types of Relationships
- Joins
- Translating words into SQL codes
- SQL Order of Execution

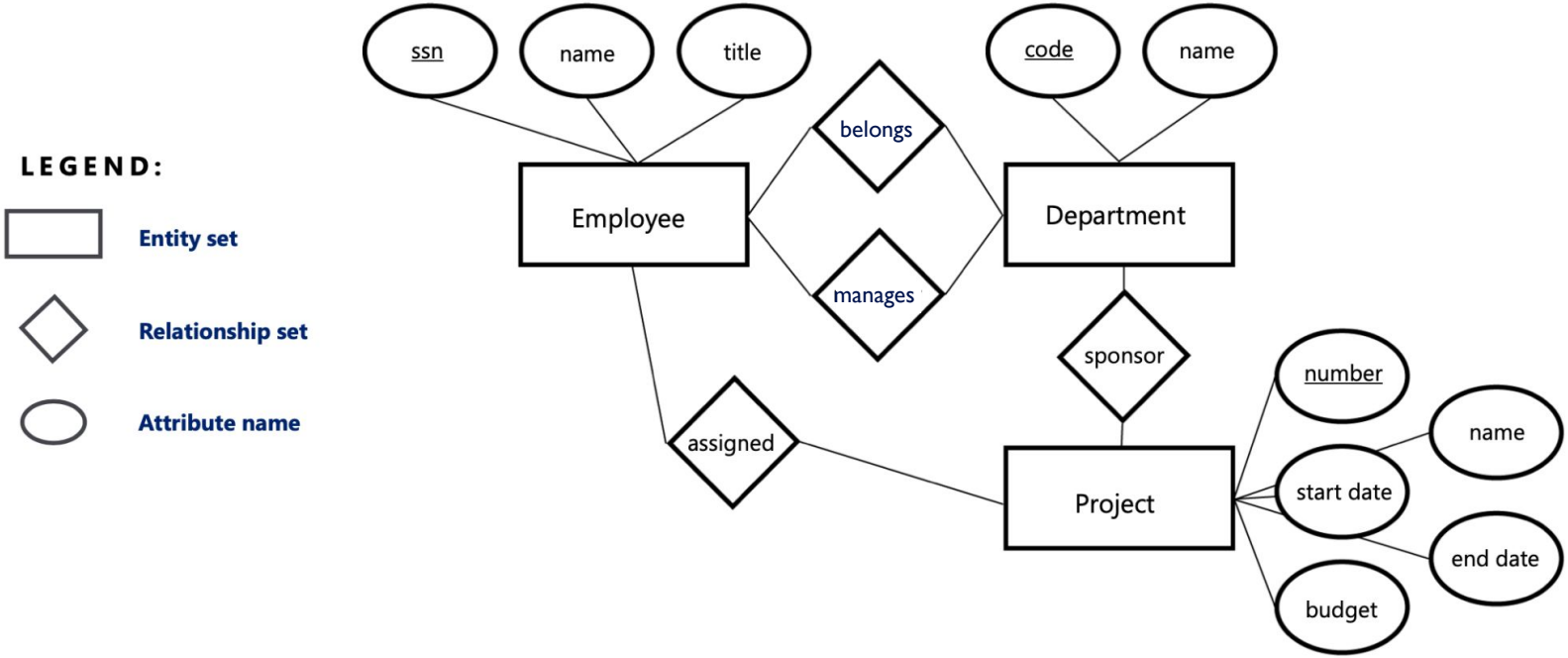
Entity-Relationship (ER) Diagram

Stages of Database Design

1. Requirements Analysis: what data, applications, critical operations
2. **Conceptual DB Design:** high-level description of data and constraints – typically using an Entity Relationship (ER) model
3. **Logical DB Design:** convert ER diagram into a schema
4. Schema Refinement: normalization (eliminating redundancy and other anomalies)

(reference: CIS 450/550)

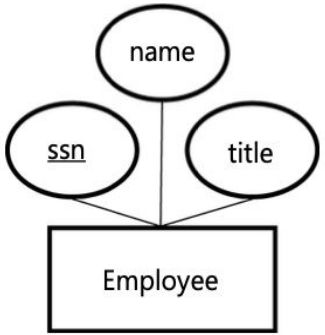
Entity-Relationship Diagrams



Definitions: Entities & Entity Sets

- ***Entity:*** Real-world object distinguishable from other objects.
  - E.g. an employee ('123-33-9875', 'John', 'Associate')
  - Described using a set of ***attributes*.**

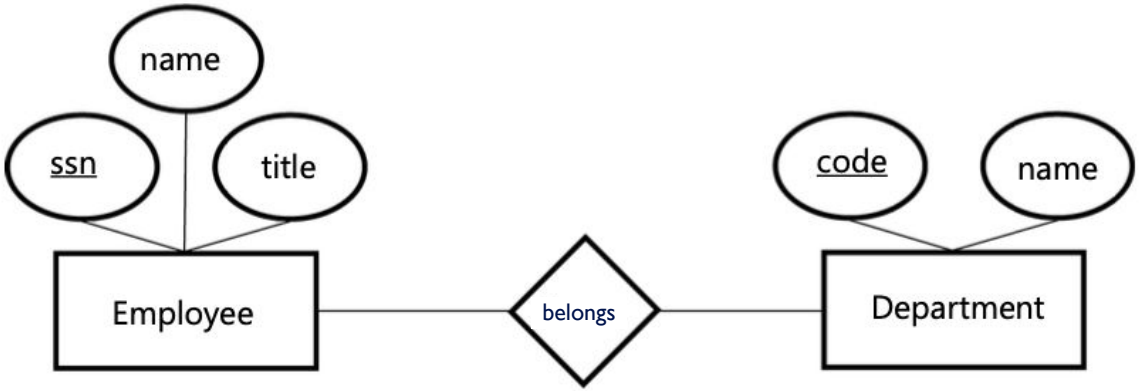
- ***Entity Set*:** Collection of entities that share similar properties
  - eg. all employees



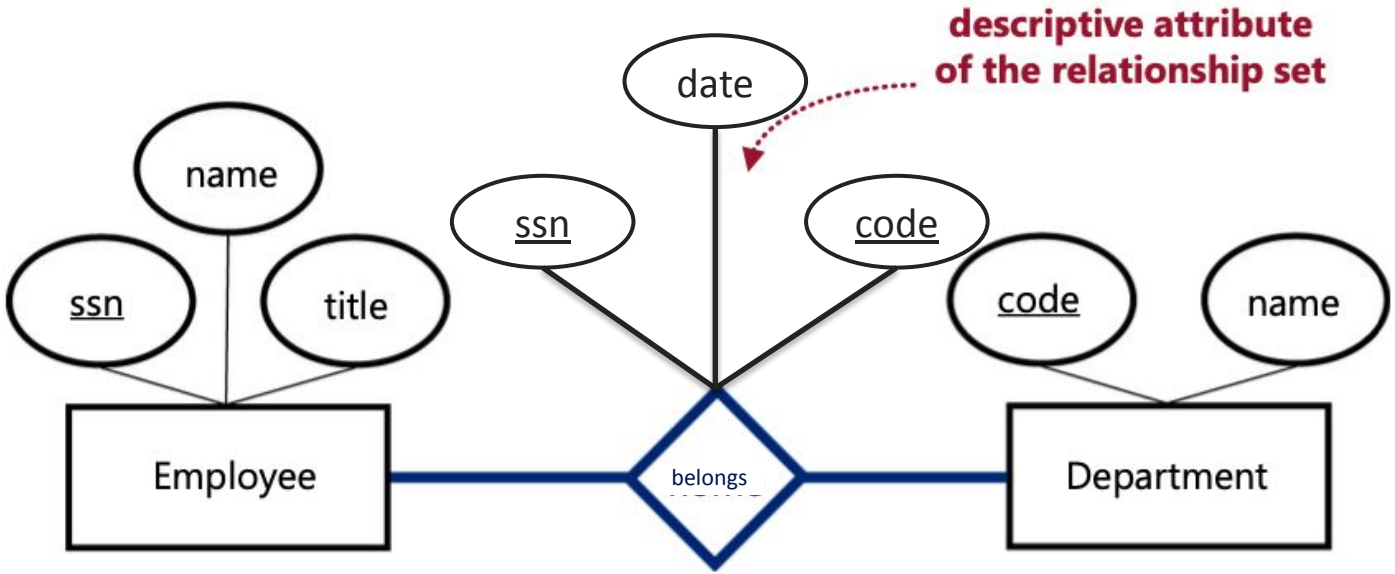
Key attributes are underlined.

Definitions: Relationship, Relationship Set

- ***Relationship***: Association between 2 or more entities. E.g., John's home department is Marketing.
- ***Relationship Set***: Named collection of similar relationships. E.g., home



Relationship Sets Can Have Attributes



ssn	name	title
123456789	Jack	Manager
234567890	Sally	Associate

ssn	code	year
123456789	abc123	1998
234567890	abc123	2022

code	name
abc123	HR
xyz456	FA

Types of Relationships

Types of Relationships

1. One-one

E.g. Each person has exactly one passport number

2. One-Many/Many-One

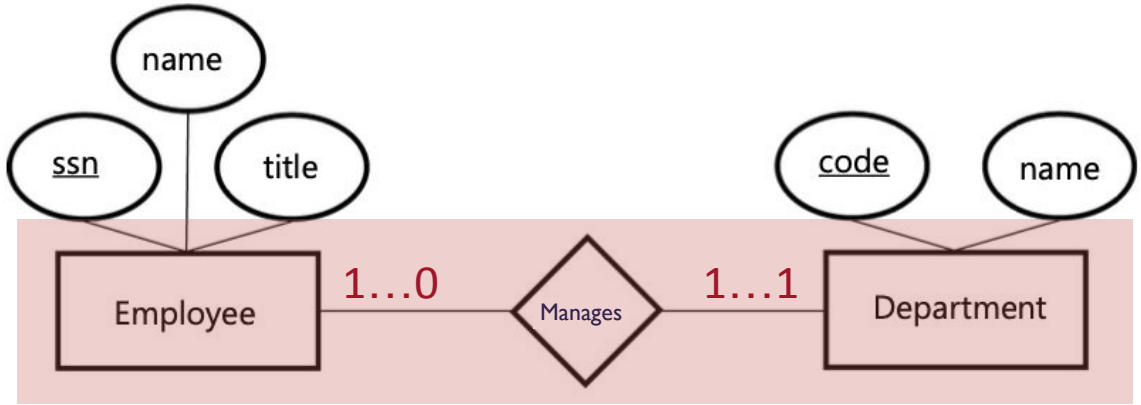
E.g. Supervisor of an Employee

3. Many-many

E.g. Students and Courses (Many students can take a course, A student can take many courses.)

1. One-One Relationships

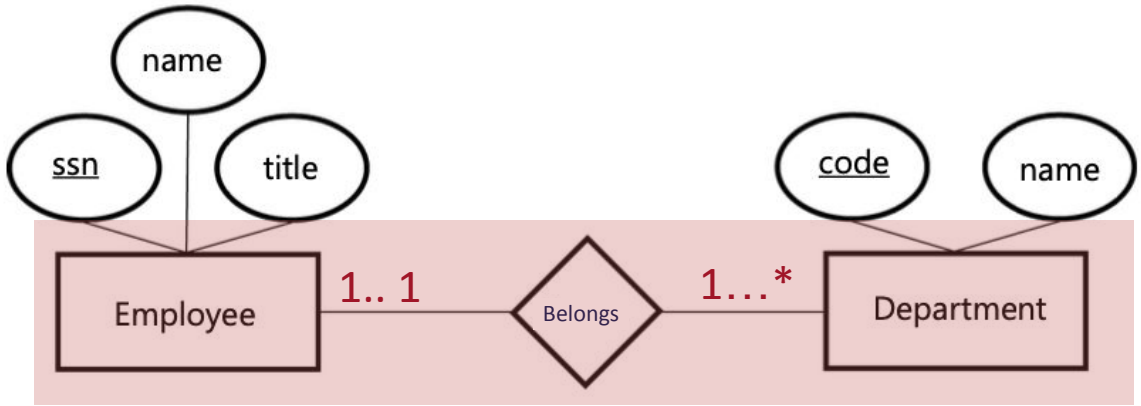
- each item in each table only appears once



Each manager manages one department.
Each department is managed by one person.

2. One-Many / Many-One Relationships

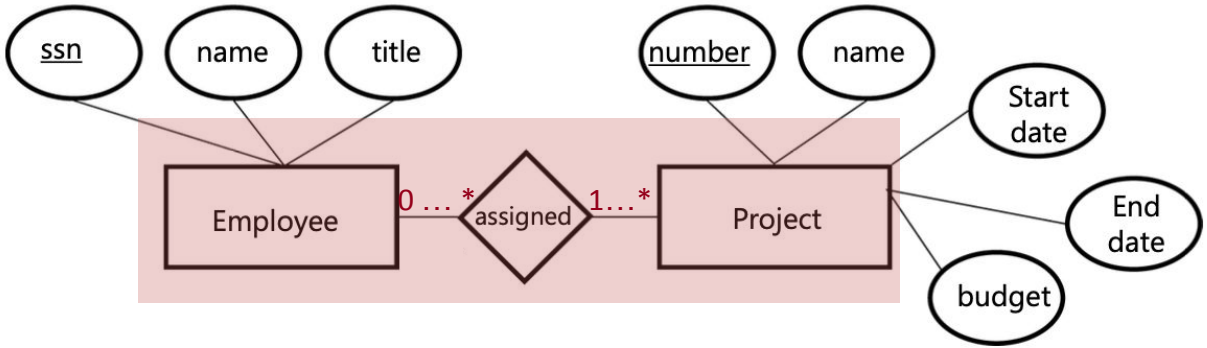
- one item in one table can have a relationship to multiple items in another table.



Each employee belongs to one department.
Each department can have many employees.

3. Many-Many Relationships

- one or more items in one table can have a relationship to one or more items in another table
- usually broken down into two one-many/many-to one relationships



Each employee can be assigned to multiple projects.

Each projects can be worked on by multiple employees.

SQL

Scenario & Tables

Imagine you are a **supervisor** of a subset of teams on a cruise, and your **manager** is asking for some data about your team

crew_df

id	name	rank
1	Jane	9
2	Dan	10
3	Alex	4
4	Jen	4
5	Brandon	1

hours_df

id	crew_id	day	hours
1	1	1	10
2	2	1	5
3	3	1	8
4	4	1	12
5	1	2	5
6	2	2	8
7	3	2	9
8	11	2	8
9	11	2	2

SQL Framework

```
SELECT t1.[name]
        ,t1.[rank]
        ,SUM(t2.[hours]) AS 'total_hour'
        ,COUNT(t2.[day]) AS 'total_day'
FROM crew_df AS t1
    LEFT JOIN hours_df AS t2
    ON t1.[id] = t2.[crew_id]
WHERE t1.[rank] > 5
GROUP BY t1.[name],t1.[rank]
ORDER BY t1.[name]
LIMIT 1
```

Order of Execution behind the scenes:

1. **FROM** (what table?)
2. **WHERE** (Filtering condition)
3. GROUP BY
4. HAVING
5. **SELECT** (What columns to keep)
6. **ORDER BY** (Sort By)
7. **LIMIT** (First x rows)

Basic Query

A list of crew names and their rank

id	name	rank
1	Jane	9
2	Dan	10
3	Alex	4
4	Jen	4
5	Brandon	1

```
FROM crew_df
```

Excel Equivalent
Sheet

A list of crew names and their rank

id	name	rank
1	Jane	9
2	Dan	10
3	Alex	4
4	Jen	4
5	Brandon	1

crew_name	rank
Jane	9
Dan	10
Alex	4
Jen	4
Brandon	1

```
SELECT [name] AS 'crew_name'
       ,[rank]
FROM crew_df
```

Excel Equivalent
Keep Columns

A list of crew names and their rank, just those with rank above 5

id	name	rank
1	Jane	9
2	Dan	10
3	Alex	4
4	Jen	4
5	Brandon	1

crew_name	rank
Jane	9
Dan	10
Alex	4
Jen	4
Brandon	1

crew_name	rank
Jane	9
Dan	10

```
SELECT [name] AS 'crew_name'
       ,[rank]
FROM crew_df
WHERE [rank] > 5
```

Excel Equivalent
Filter

A list of crew names and their rank, just those with rank above 5, most senior at the top of the list

id	name	rank
1	Jane	9
2	Dan	10
3	Alex	4
4	Jen	4
5	Brandon	1

crew_name	rank
Jane	9
Dan	10
Alex	4
Jen	4
Brandon	1

crew_name	rank
Jane	9
Dan	10

crew_name	rank
Dan	10
Jane	9

```
SELECT [name] AS 'crew_name'
       ,[rank]
FROM crew_df
WHERE [rank] > 5
ORDER BY [rank] DESC
```

Excel Equivalent
Sort

A list of crew names and their rank, just those with rank above 5, most senior at the top of the list, just the most senior

id	name	rank
1	Jane	9
2	Dan	10
3	Alex	4
4	Jen	4
5	Brandon	1

crew_name	rank
Jane	9
Dan	10
Alex	4
Jen	4
Brandon	1

crew_name	rank
Jane	9
Dan	10

crew_name	rank
Dan	10
Jane	9

crew_name	rank
Dan	10

```
SELECT [name] AS 'crew_name'
        ,[rank]
FROM crew_df
WHERE [rank] > 5
ORDER BY [rank] DESC
LIMIT 1
```

Excel Equivalent
Top K rows

SQL Framework

```
SELECT t1.[name]
        ,t1.[rank]
        ,SUM(t2.[hours]) AS 'total_hour'
        ,COUNT(t2.[day]) AS 'total_day'
FROM crew_df AS t1
      LEFT JOIN hours_df AS t2
      ON t1.[id] = t2.[crew_id]
WHERE t1.[rank] > 5
GROUP BY t1.[name],t1.[rank]
ORDER BY t1.[name]
LIMIT 1
```

Order of Execution behind the scenes:

1. **FROM** (what table?) + (**JOIN**)
2. **WHERE** (Filtering condition)
3. GROUP BY
4. HAVING
5. **SELECT** (What columns to keep)
6. **ORDER BY** (Sort By)
7. **LIMIT** (First x rows)

JOINS

A list of crew names and their total working hours, just those above rank 5

id	name	rank
1	Jane	9
2	Dan	10
3	Alex	4
4	Jen	4
5	Brandon	1

id	crew_id	day	hours
1	1	1	10
2	2	1	5
3	3	1	8
4	4	1	12
5	1	2	5
6	2	2	8
7	3	2	9
8	11	2	8
9	11	2	2

```
FROM crew_df AS t1
    LEFT JOIN hours_df AS t2
    ON t1.[id] = t2.[crew_id]
```

Excel Equivalent
VLookUp

A list of crew names and their total working hours, just those above rank 5

id	name	rank
1	Jane	9
2	Dan	10
3	Alex	4
4	Jen	4
5	Brandon	1

id	crew_id	day	hours
1	1	1	10
2	2	1	5
3	3	1	8
4	4	1	12
5	1	2	5
6	2	2	8
7	3	2	9
8	11	2	8
9	11	2	2

crew_df AS t1			hours_df AS t2			
id	name	rank	id	crew_id	day	hours
1	Jane	9	1	1	1	10
2	Dan	10	2	2	1	5
3	Alex	4	3	3	1	8
4	Jen	4	4	4	1	12
1	Jane	9	5	1	2	5
2	Dan	10	6	2	2	8
3	Alex	4	7	3	2	9
5	Brandon	1				

```
FROM crew_df AS t1
    LEFT JOIN hours_df AS t2
    ON t1.[id] = t2.[crew_id]
```

A list of crew names and their total working hours, just those above rank 5

id	name	rank
1	Jane	9
2	Dan	10
3	Alex	4
4	Jen	4
5	Brandon	1

id	crew_id	day	hours
1	1	1	10
2	2	1	5
3	3	1	8
4	4	1	12
5	1	2	5
6	2	2	8
7	3	2	9
8	11	2	8
9	11	2	2

crew_df AS t1			hours_df AS t2			
id	name	rank	id	crew_id	day	hours
1	Jane	9	1	1	1	10
2	Dan	10	2	2	1	5
3	Alex	4	3	3	1	8
4	Jen	4	4	4	1	12
1	Jane	9	5	1	2	5
2	Dan	10	6	2	2	8
3	Alex	4	7	3	2	9
5	Brandon	1				

Observation:

1. **MAIN TABLE**: Only record in the crew_df are preserved

```
FROM crew_df AS t1
    LEFT JOIN hours_df AS t2
    ON t1.[id] = t2.[crew_id]
```

A list of crew names and their total working hours, just those above rank 5

id	name	rank
1	Jane	9
2	Dan	10
3	Alex	4
4	Jen	4
5	Brandon	1

id	crew_id	day	hours
1	1	1	10
2	2	1	5
3	3	1	8
4	4	1	12
5	1	2	5
6	2	2	8
7	3	2	9
8	11	2	8
9	11	2	2

crew_df AS t1			hours_df AS t2			
id	name	rank	id	crew_id	day	hours
1	Jane	9	1	1	1	10
2	Dan	10	2	2	1	5
3	Alex	4	3	3	1	8
4	Jen	4	4	4	1	12
1	Jane	9	5	1	2	5
2	Dan	10	6	2	2	8
3	Alex	4	7	3	2	9
5	Brandon	1				

Observation:

1. **MAIN TABLE**: Only record in the crew_df are preserved
2. **REPEATED ROWS:** crew_df rows are repeated because there are multiple records of it in the hours_df

```
FROM crew_df AS t1
    LEFT JOIN hours_df AS t2
    ON t1.[id] = t2.[crew_id]
```

A list of crew names and their total working hours, just those above rank 5

crew_df AS t1			hours_df AS t2			
id	**name**	**rank**	**id**	**crew_id**	**day**	**hours**
1	Jane	9	1	1	1	10
2	Dan	10	2	2	1	5
3	Alex	4	3	3	1	8
4	Jen	4	4	4	1	12
1	Jane	9	5	1	2	5
2	Dan	10	6	2	2	8
3	Alex	4	7	3	2	9
5	Brandon	1				

crew_df AS t1			hours_df AS t2			
id	**name**	**rank**	**id**	**crew_id**	**day**	**hours**
1	Jane	9	1	1	1	10
2	Dan	10	2	2	1	5
1	Jane	9	5	1	2	5
2	Dan	10	6	2	2	8

```
FROM crew_df AS t1
    LEFT JOIN hours_df AS t2
    ON t1.[id] = t2.[crew_id]
WHERE [rank] > 5
```

Excel Equivalent
Filter

Aggregation

SQL Framework

```
SELECT t1.[name]
        ,t1.[rank]
        ,SUM(t2.[hours]) AS 'total_hour'
        ,COUNT(t2.[day]) AS 'total_day'
FROM crew_df AS t1
      LEFT JOIN hours_df AS t2
      ON t1.[id] = t2.[crew_id]
WHERE t1.[rank] > 5
GROUP BY t1.[name],t1.[rank]
ORDER BY t1.[name]
LIMIT 1
```

Order of Execution behind the scenes:

1. **FROM** (what table?) (**JOIN**)
2. **WHERE** (Filtering condition)
3. **GROUP BY** (Aggregation Dimension)
4. **HAVING** (Filtering condition on for aggregated results)
5. **SELECT** (What columns to keep)
6. **ORDER BY** (Sort By)
7. **LIMIT** (First x rows)

A list of crew names and their total working hours, just those above rank 5

crew_df AS t1			hours_df AS t2			
id	**name**	**rank**	**id**	**crew_id**	**day**	**hours**
1	Jane	9	1	1	1	10
2	Dan	10	2	2	1	5
1	Jane	9	5	1	2	5
2	Dan	10	6	2	2	8

name	total_hour
Jane	15
Dan	13

```
SELECT t1.[name]
        ,SUM(t2.[hours]) AS 'total_hour'
FROM crew_df AS t1
    LEFT JOIN hours_df AS t2
    ON t1.[id] = t2.[crew_id]
WHERE t1.[rank] > 5
GROUP BY t1.[name]
```

Excel Equivalent
Pivot Table

A list of crew names, their rank, and their total working hours, just those above rank 5

crew_df AS t1			hours_df AS t2			
id	**name**	**rank**	**id**	**crew_id**	**day**	**hours**
1	Jane	9	1	1	1	10
2	Dan	10	2	2	1	5
1	Jane	9	5	1	2	5
2	Dan	10	6	2	2	8

name	rank	total_hour
Jane	9	15
Dan	10	13

```
SELECT t1.[name]
        ,t1.[rank]
        ,SUM(t2.[hours]) AS 'total_hour'
FROM crew_df AS t1
    LEFT JOIN hours_df AS t2
    ON t1.[id] = t2.[crew_id]
WHERE t1.[rank] > 5
GROUP BY t1.[name],t1.[rank]
```

Observation:

Every time you add a field, you need to decide if it's a dimension or a measure field

- Dimension: Appear in Group By
- Measure: Wrapped in **aggregation** function in SELECT

Excel Equivalent
Pivot Table

A list of crew names, their rank, and their total working hours and total days worked, just those above rank 5

crew_df AS t1			hours_df AS t2			
id	**name**	**rank**	**id**	**crew_id**	**day**	**hours**
1	Jane	9	1	1	1	10
2	Dan	10	2	2	1	5
1	Jane	9	5	1	2	5
2	Dan	10	6	2	2	8

name	rank	total_hour	total_day
Jane	9	15	2
Dan	10	13	2

```
SELECT t1.[name]
        ,t1.[rank]
        ,SUM(t2.[hours]) AS 'total_hour'
        ,COUNT(t2.[day]) AS 'total_day'
FROM crew_df AS t1
    LEFT JOIN hours_df AS t2
    ON t1.[id] = t2.[crew_id]
WHERE t1.[rank] > 5
GROUP BY t1.[name],t1.[rank]
```

Observation:

Every time you add a field, you need to decide if it's a dimension or a measure field

- Dimension: Appear in Group By
- Measure: Wrapped in **aggregation** function in SELECT

Excel Equivalent
Pivot Table

A list of crew names, their rank, and their total working hours and total days worked, just those above rank 5, just those with total of at least 15 total hours

crew_df AS t1			hours_df AS t2			
id	**name**	**rank**	**id**	**crew_id**	**day**	**hours**
1	Jane	9	1	1	1	10
2	Dan	10	2	2	1	5
1	Jane	9	5	1	2	5
2	Dan	10	6	2	2	8

name	rank	total_hour	total_day
Jane	9	15	2
Dan	10	13	2

name	rank	total_hour	total_day
Jane	9	15	2

```
SELECT t1.[name]
        ,t1.[rank]
        ,SUM(t2.[hours]) AS 'total_hour'
        ,COUNT(t2.[day]) AS 'total_day'
FROM crew_df AS t1
    LEFT JOIN hours_df AS t2
    ON t1.[id] = t2.[crew_id]
WHERE t1.[rank] > 5
GROUP BY t1.[name],t1.[rank]
HAVING SUM(t2.[hours])  >= 15
```

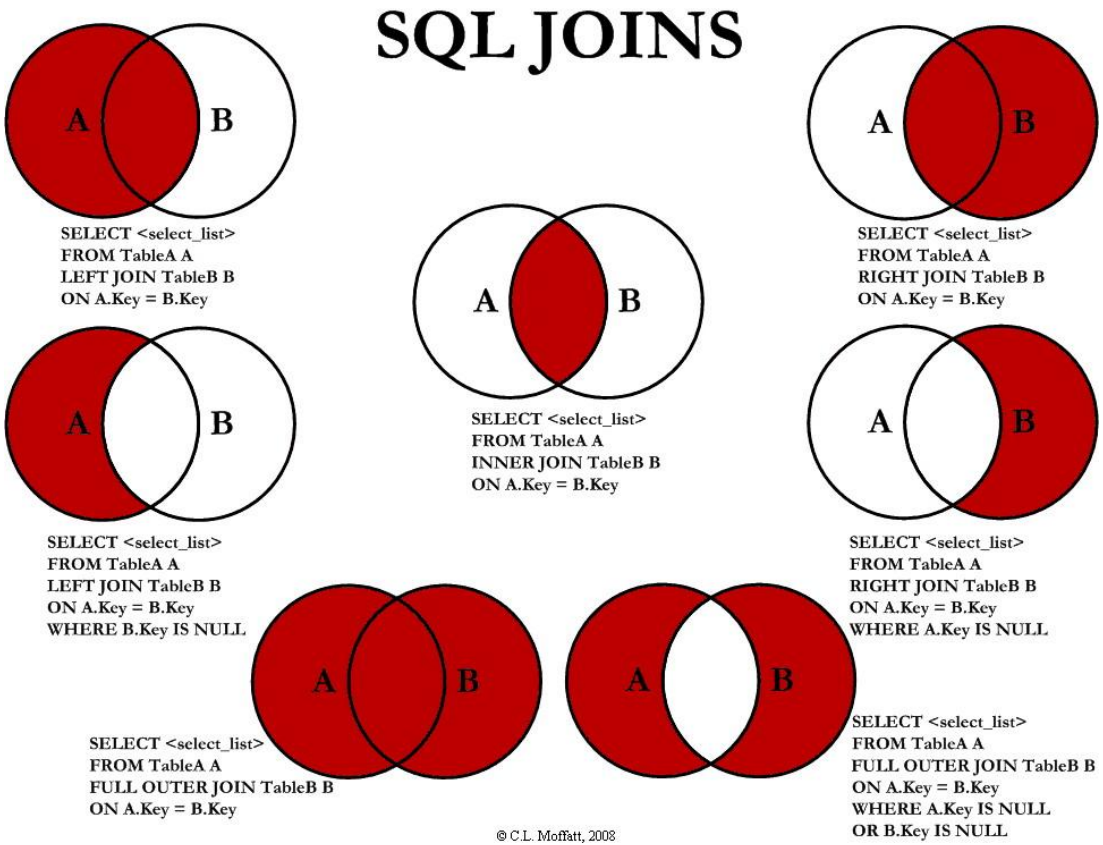
Observation:

1. **HAVING is a filter for aggregated results**
2. **CANNOT use 'total_day' as a reference as it's not in the FROM table**

Excel Equivalent
Pivot Table

Types of Joins

Join Overview



Inner Join

crew_df

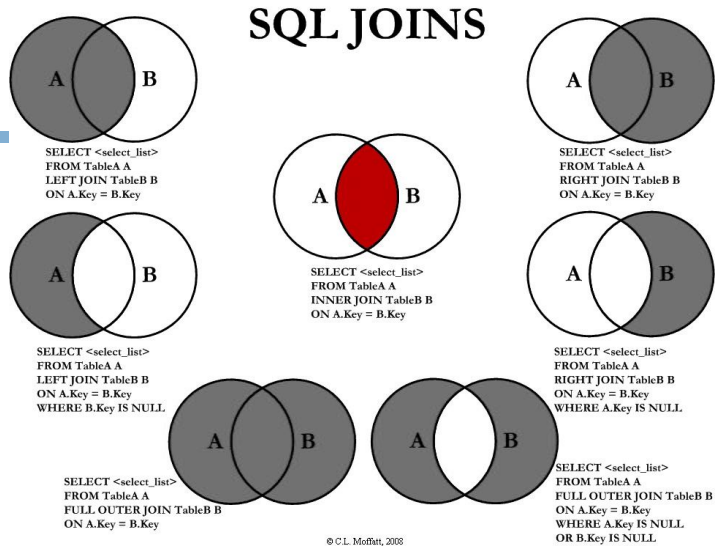
id	name	rank
1	Jane	9
2	Dan	10
3	Alex	4
4	Jen	4

hours_df

id	crew_id	day	hours
1	1	1	10
2	2	1	5
3	3	1	8
5	11	2	8
6	11	2	2

Join_df

id	name	rank	id	crew_id	day	hours
1	Jane	9	1	1	1	10
2	Dan	10	2	2	1	5
3	Alex	4	3	3	1	8




```
SELECT *
FROM crew_df AS t1
    INNER JOIN hours_df AS t2
    ON t1.[id] = t2.[crew_id]
```

Left Join

crew_df

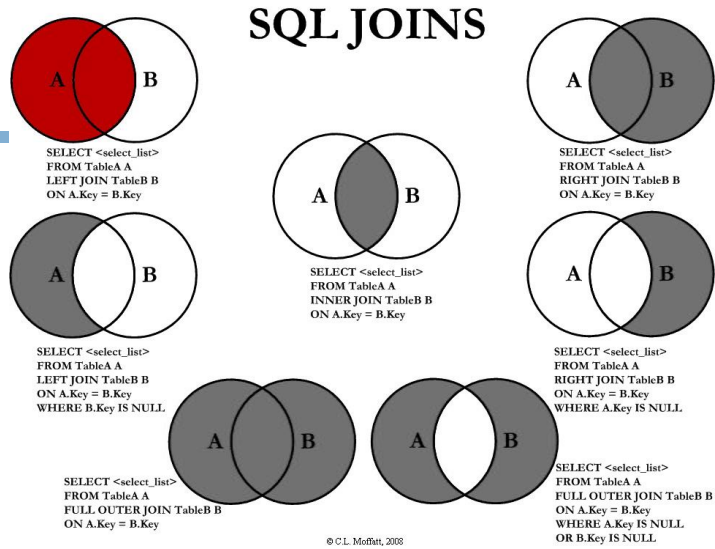
id	name	rank
1	Jane	9
2	Dan	10
3	Alex	4
4	Jen	4

hours_df

id	crew_id	day	hours
1	1	1	10
2	2	1	5
3	3	1	8
5	11	2	8
6	11	2	2

Join_df

id	name	rank	id	crew_id	day	hours
1	Jane	9	1	1	1	10
2	Dan	10	2	2	1	5
3	Alex	4	3	3	1	8
4	Jen	4				




```
SELECT *
FROM crew_df AS t1
    LEFT JOIN hours_df AS t2
    ON t1.[id] = t2.[crew_id]
```

Right Join

crew_df

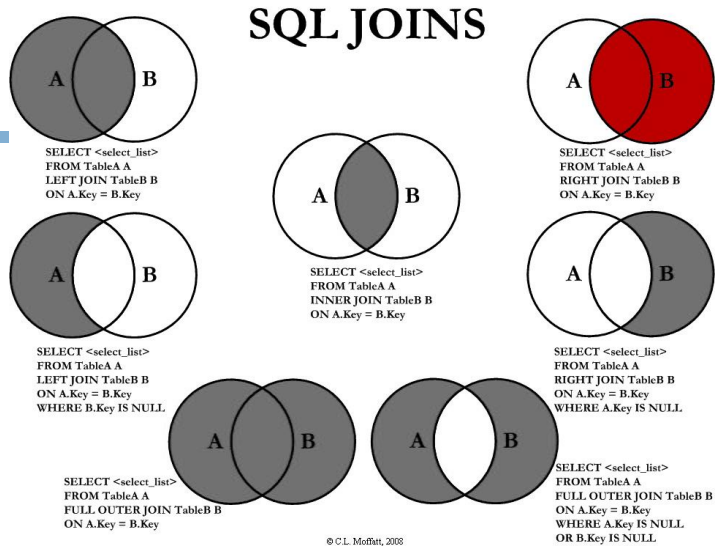
id	name	rank
1	Jane	9
2	Dan	10
3	Alex	4
4	Jen	4

hours_df

id	crew_id	day	hours
1	1	1	10
2	2	1	5
3	3	1	8
5	11	2	8
6	11	2	2

Join_df

id	name	rank	id	crew_id	day	hours
1	Jane	9	1	1	1	10
2	Dan	10	2	2	1	5
3	Alex	4	3	3	1	8
			5	11	2	8
			6	11	2	2




```
SELECT *
FROM crew_df AS t1
    RIGHT JOIN hours_df AS t2
    ON t1.[id] = t2.[crew_id]
```

Full Join

crew_df

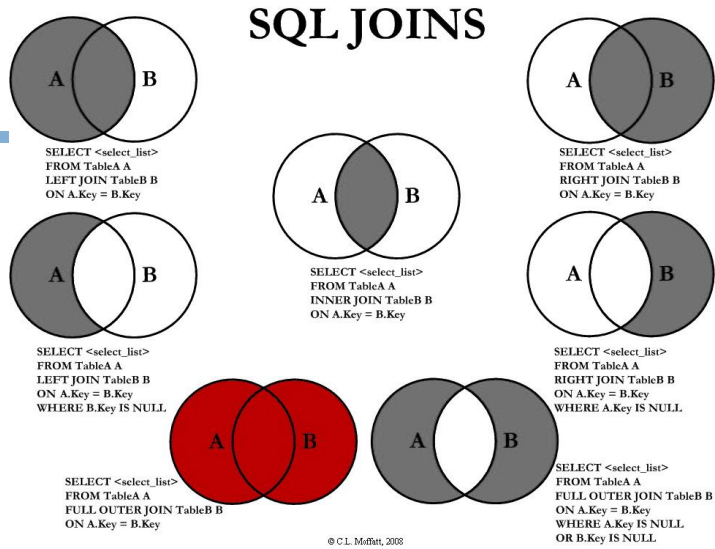
id	name	rank
1	Jane	9
2	Dan	10
3	Alex	4
4	Jen	4

hours_df

id	crew_id	day	hours
1	1	1	10
2	2	1	5
3	3	1	8
5	11	2	8
6	11	2	2

Join_df

id	name	rank	id	crew_id	day	hours
1	Jane	9	1	1	1	10
2	Dan	10	2	2	1	5
3	Alex	4	3	3	1	8
4	Jen	4				
			5	11	2	8
			6	11	2	2




```
SELECT *
FROM crew_df AS t1
    FULL JOIN hours_df AS t2
    ON t1.[id] = t2.[crew_id]
```

Careful!

crew_df

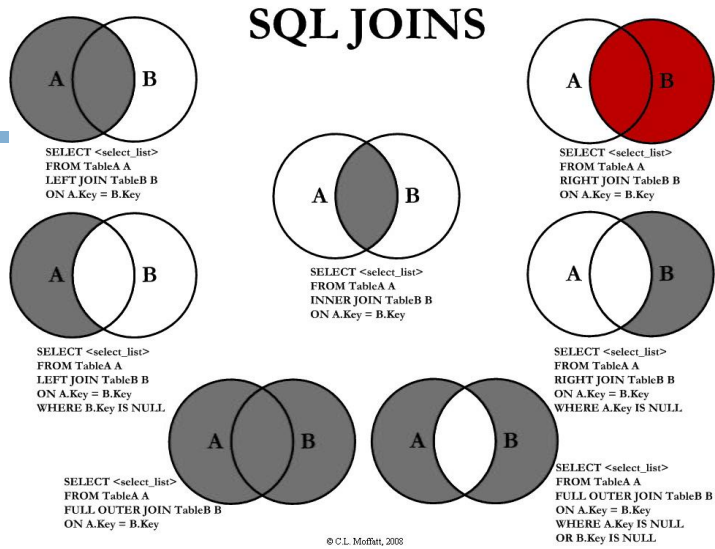
id	name	rank
1	Jane	9
2	Dan	10
3	Alex	4
3	Alex	4

hours_df

id	crew_id	day	hours
1	1	1	10
2	2	1	5
3	3	1	8
5	11	2	8
6	11	2	2

Join_df

id	name	rank	id	crew_id	day	hours
1	Jane	9	1	1	1	10
2	Dan	10	2	2	1	5
3	Alex	4	3	3	1	8
3	Alex	4	3	3	1	8
			5	11	2	8
			6	11	2	2




```
SELECT *
FROM crew_df AS t1
    RIGHT JOIN hours_df AS t2
    ON t1.[id] = t2.[crew_id]
```