JOINS IN MYSQL

ID	Name
1	ARNOLD
2	BOB
3	CAT
4	DOG

DANCE

ID	Name
3	CAT
4	DOG
5	LION
6	BEAR

MUSIC

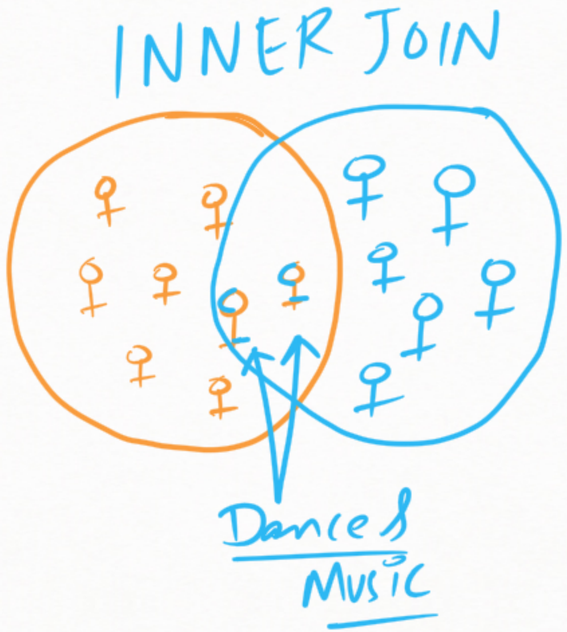
ALIAS => Renaming Column OR Table Name on the fly.

AS

```
SELECT firstname AS name
FROM employee
```

```
mysql> SELECT * FROM DANCE;
+----+--------+
| ID | NAME   |
+----+--------+
|  1 | ARNOLD |
|  2 | BOB    |
|  3 | CAT    |
|  4 | DOG    |
+----+--------+
4 rows in set (0.00 sec)
```

DANCE




```
mysql> SELECT * FROM MUSIC;
+----+-------------+
| ID | NAME        |
+----+-------------+
|  3 | katty perry |
|  4 | snoopdog    |
|  5 | LION        |
|  6 | BEAR        |
+----+-------------+
4 rows in set (0.00 sec)
```

MUSIC

```
Select D.ID, D.Name, M.Name
FROM DANCE AS D
JOIN MUSIC AS M
ON D.ID = M.ID
```

```
mysql> SELECT * FROM DANCE;
+----+--------+
| ID | NAME   |
+----+--------+
|  1 | ARNOLD |
|  2 | BOB    |
|  3 | CAT    |
|  4 | DOG    |
+----+--------+
4 rows in set (0.00 sec)
```

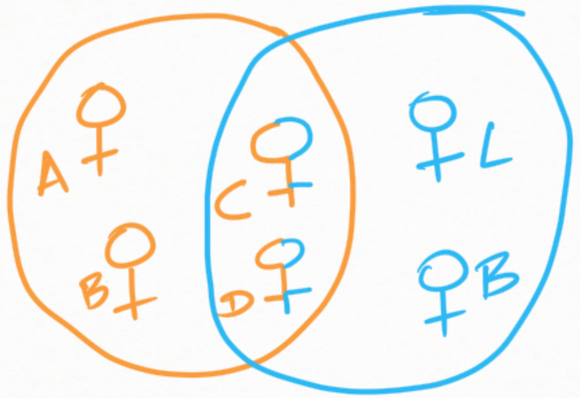
INNER JOIN

```
mysql> SELECT * FROM MUSIC;
+----+-------------+
| ID | NAME        |
+----+-------------+
|  3 | katty perry |
|  4 | snoopdog    |
|  5 | LION        |
|  6 | BEAR        |
+----+-------------+
4 rows in set (0.00 sec)
```

```
mysql> SELECT D.ID,D.NAME AS DANCE_NAME,M.NAME AS MUSIC_NAME
    -> FROM DANCE AS D
    -> JOIN MUSIC AS M
    -> ON D.ID = M.ID;
+----+------------+-------------+
| ID | DANCE_NAME | MUSIC_NAME  |
+----+------------+-------------+
|  3 | CAT        | katty perry |
|  4 | DOG        | snoopdog    |
+----+------------+-------------+
2 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM DANCE;
+----+--------+
| ID | NAME   |
+----+--------+
|  1 | ARNOLD |
|  2 | BOB    |
|  3 | CAT    |
|  4 | DOG    |
+----+--------+
4 rows in set (0.00 sec)
```

DANCE ✓



There is NO ID 1 & 2.

MUSIC

```
mysql> SELECT * FROM MUSIC;
+----+-------------+
| ID | NAME        |
+----+-------------+
|  3 | katty perry |
|  4 | snoopdog    |
|  5 | LION        |
|  6 | BEAR        |
+----+-------------+
4 rows in set (0.00 sec)
```

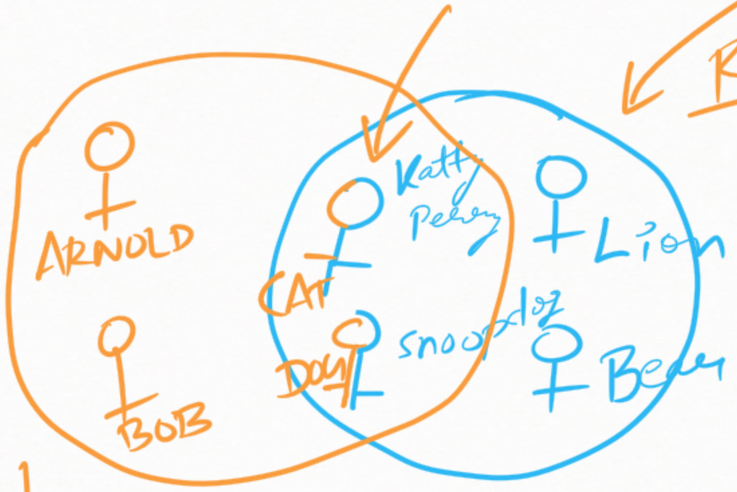
```
mysql> SELECT D.ID,D.NAME AS DANCE_NAME,M.NAME AS MUSIC_NAME
    -> FROM DANCE AS D
    -> LEFT JOIN MUSIC AS M
    -> ON D.ID = M.ID;
+----+-----------+-------------+
| ID | DANCE_NAME | MUSIC_NAME  |
+----+-----------+-------------+
|  1 | ARNOLD     | NULL        |
|  2 | BOB        | NULL        |
|  3 | CAT        | katty perry |
|  4 | DOG        | snoopdog    |
+----+-----------+-------------+
4 rows in set (0.00 sec)
```

We mentioned it before LEFT JOIN

So that's why DANCE is the left Table

```
mysql> SELECT * FROM DANCE;
+----+--------+
| ID | NAME   |
+----+--------+
|  1 | ARNOLD |
|  2 | BOB    |
|  3 | CAT    |
|  4 | DOG    |
+----+--------+
4 rows in set (0.00 sec)
```

There is No ID 5 & 6 here.




```
mysql> SELECT * FROM MUSIC;
+----+--------------+
| ID | NAME         |
+----+--------------+
|  3 | katty perry  |
|  4 | snoopdog     |
|  5 | LION         |
|  6 | BEAR         |
+----+--------------+
4 rows in set (0.00 sec)
```

```
mysql> SELECT M.ID,D.NAME AS DANCE_NAME,M.NAME AS MUSIC_NAME
    -> FROM DANCE AS D
    -> RIGHT JOIN MUSIC AS M
    -> ON D.ID = M.ID;
+----+------------+-------------+
| ID | DANCE_NAME | MUSIC_NAME  |
+----+------------+-------------+
|  3 | CAT        | katty perry |
|  4 | DOG        | snoopdog    |
|  5 | NULL       | LION        |
|  6 | NULL       | BEAR        |
+----+------------+-------------+
4 rows in set (0.00 sec)
```

Left table

Right table

```
mysql> SELECT * FROM MUSIC;
+----+-------------+
| ID | NAME        |
+----+-------------+
|  3 | katty perry |
|  4 | snoopdog    |
|  5 | LION        |
|  6 | BEAR        |
+----+-------------+
4 rows in set (0.00 sec)
```

Can you see 1 & 2?

```
mysql> SELECT * FROM DANCE;
+----+--------+
| ID | NAME   |
+----+--------+
|  1 | ARNOLD |
|  2 | BOB    |
|  3 | CAT    |
|  4 | DOG    |
+----+--------+
4 rows in set (0.00 sec)
```

```
mysql> SELECT D.ID,D.NAME AS DANCE_NAME,M.NAME AS MUSIC_NAME
    -> FROM MUSIC AS M
    -> RIGHT JOIN DANCE AS D
    -> ON D.ID = M.ID;
+----+------------+-------------+
| ID | DANCE_NAME | MUSIC_NAME  |
+----+------------+-------------+
|  1 | ARNOLD     | NULL        |
|  2 | BOB        | NULL        |
|  3 | CAT        | katty perry |
|  4 | DOG        | snoopdog    |
+----+------------+-------------+
4 rows in set (0.00 sec)
```

left

Right

Practical Examples of JOINS

Employee

emp_id	first_name	last_name	birth_date	sex	salary	super_id	branch_id
100	David	Wallace	1967-11-17	M	250,000	NULL	1
101	Jan	Levinson	1961-05-11	F	110,000	100	1
102	Michael	Scott	1964-03-15	M	75,000	100	2
103	Angela	Martin	1971-06-25	F	63,000	102	2
104	Kelly	Kapoor	1980-02-05	F	55,000	102	2
105	Stanley	Hudson	1958-02-19	M	69,000	102	2
106	Josh	Porter	1969-09-05	M	78,000	100	3
107	Andy	Bernard	1973-07-22	M	65,000	106	3
108	Jim	Halpert	1978-10-01	M	71,000	106	3

Branch

branch_id	branch_name	mgr_id	mgr_start_date
1	Corporate	100	2006-02-09
2	Scranton	102	1992-04-06
3	Stamford	106	1998-02-13

Client

client_id	client_name	branch_id
400	Dunmore Highschool	2
401	Lackawana Country	2
402	FedEx	3
403	John Daly Law, LLC	3
404	Scranton Whitepages	2
405	Times Newspaper	3
406	FedEx	2

Works_With

emp_id	client_id	total_sales
105	400	55,000
102	401	267,000
108	402	22,500
107	403	5,000
108	403	12,000
105	404	33,000
107	405	26,000
102	406	15,000
105	406	130,000

Branch Supplier

branch_id	supplier_name	supply_type
2	Hammer Mill	Paper
2	Uni-ball	Writing Utensils
3	Patriot Paper	Paper
2	J.T. Forms & Labels	Custom Forms
3	Uni-ball	Writing Utensils
3	Hammer Mill	Paper
3	Stamford Lables	Custom Forms

Q1. Print Employee Name and their Supervisor Name

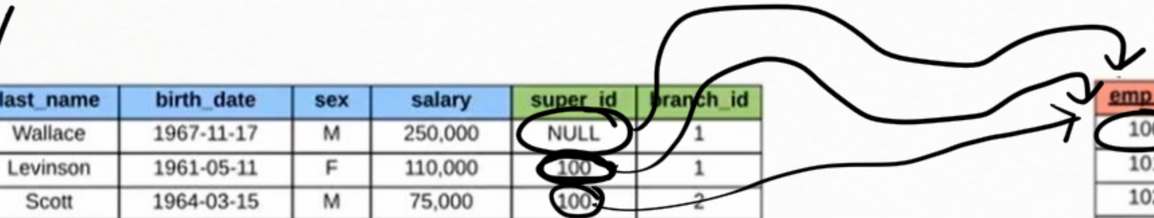
* JOIN can be applied on the same table. Remember ALIASING.

Q2. Print Branch Name and their Manager Name.

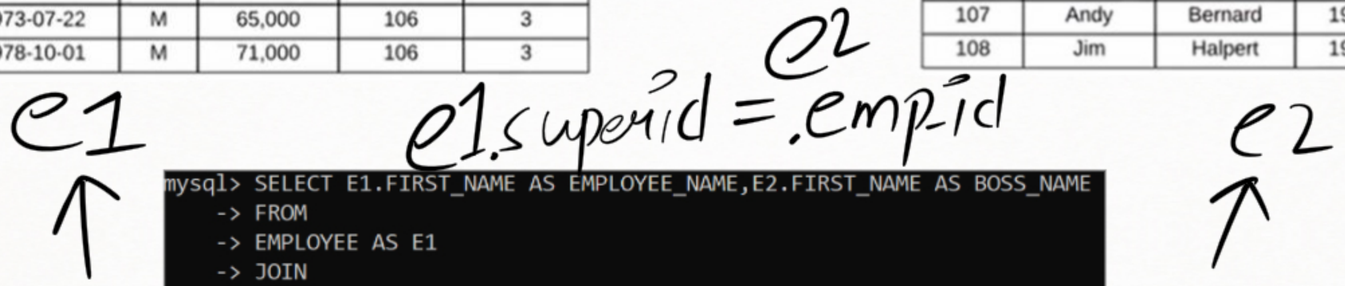
Q3. Print client_name, emp_id and total Sales related.

H.W

emp_id	first_name	last_name	birth_date	sex	salary	super_id	branch_id
100	David	Wallace	1967-11-17	M	250,000	NULL	1
101	Jan	Levinson	1961-05-11	F	110,000	100	1
102	Michael	Scott	1964-03-15	M	75,000	100	2
103	Angela	Martin	1971-06-25	F	63,000	102	2
104	Kelly	Kapoor	1980-02-05	F	55,000	102	2
105	Stanley	Hudson	1958-02-19	M	69,000	102	2
106	Josh	Porter	1969-09-05	M	78,000	100	3
107	Andy	Bernard	1973-07-22	M	65,000	106	3
108	Jim	Halpert	1978-10-01	M	71,000	106	3



emp_id	first_name	last_name	birth_date	sex	salary	super_id	branch_id
100	David	Wallace	1967-11-17	M	250,000	NULL	1
101	Jan	Levinson	1961-05-11	F	110,000	100	1
102	Michael	Scott	1964-03-15	M	75,000	100	2
103	Angela	Martin	1971-06-25	F	63,000	102	2
104	Kelly	Kapoor	1980-02-05	F	55,000	102	2
105	Stanley	Hudson	1958-02-19	M	69,000	102	2
106	Josh	Porter	1969-09-05	M	78,000	100	3
107	Andy	Bernard	1973-07-22	M	65,000	106	3
108	Jim	Halpert	1978-10-01	M	71,000	106	3




```
mysql> SELECT E1.FIRST_NAME AS EMPLOYEE_NAME,E2.FIRST_NAME AS BOSS_NAME
    -> FROM
    -> EMPLOYEE AS E1
    -> JOIN
    -> EMPLOYEE AS E2
    -> ON E1.SUPER_ID = E2.EMP_ID;
+---------------+-----------+
| EMPLOYEE_NAME | BOSS_NAME |
+---------------+-----------+
| Jan           | David     |
| Michael       | David     |
| Angela        | Michael   |
| Kelly         | Michael   |
| Stanley       | Michael   |
| Josh          | David     |
| Andy          | Josh      |
| Jim           | Josh      |
+---------------+-----------+
8 rows in set (0.00 sec)
```

Print Branch Name & their Manager Name

branch_id	branch_name	mgr_id	mgr_start_date
1	Corporate	100	2006-02-09
2	Scranton	102	1992-04-06
3	Stamford	106	1998-02-13

emp_id	first_name	last_name	birth_date	sex	salary	super_id	branch_id
100	David	Wallace	1967-11-17	M	250,000	NULL	1
101	Jan	Levinson	1961-05-11	F	110,000	100	1
102	Michael	Scott	1964-03-15	M	75,000	100	2
103	Angela	Martin	1971-06-25	F	63,000	102	2
104	Kelly	Kapoor	1980-02-05	F	55,000	102	2
105	Stanley	Hudson	1958-02-19	M	69,000	102	2
106	Josh	Porter	1969-09-05	M	78,000	100	3
107	Andy	Bernard	1973-07-22	M	65,000	106	3
108	Jim	Halpert	1978-10-01	M	71,000	106	3

```
mysql> SELECT B1.branch_name,E1.first_name,E1.last_name
    -> FROM
    -> EMPLOYEE AS E1
    -> JOIN
    -> BRANCH AS B1
    -> ON B1.mgr_id = E1.emp_id;
+-------------+------------+-----------+
| branch_name | first_name | last_name |
+-------------+------------+-----------+
| Corporate   | David      | Wallace   |
| Scranton    | Michael    | Scott     |
| Stamford    | Josh       | Porter    |
+-------------+------------+-----------+
3 rows in set (0.00 sec)
```

```
mysql> SELECT branch_name,first_name
    -> FROM
    -> EMPLOYEE
    -> JOIN
    -> BRANCH
    -> ON mgr_id = emp_id;
+-------------+------------+
| branch_name | first_name |
+-------------+------------+
| Corporate   | David      |
| Scranton    | Michael    |
| Stamford    | Josh       |
+-------------+------------+
3 rows in set (0.00 sec)
```

HAVING CLAUSE

=> Used with Group By for conditions on Agg. for results.

=> If Used without group by ≃ WHERE CLAUSE

=> It is always utilised on groups.

=> WHERE condition does not work on groups

=> WHERE & HAVING CAN BE USED IN 1 QUERY.

O.No		Quantity	Price Each
101	DUKE MEN TSHIRT	3	300Rs
101	AMAZONT RIPOD STAND	1	1000Rs

O.N	?itemcount	total
101	4	1900] ✓
102	5	2500

```
mysql> SELECT ORDERNUMBER,SUM(QUANTITYORDERED) AS ITEMCOUNT,
    -> SUM(PRICEEACH*QUANTITYORDERED) AS TOTAL
    -> FROM
    -> ORDERDETAILS
    -> GROUP BY ORDERNUMBER HAVING TOTAL>55000;
+-------------+-----------+----------+
| ORDERNUMBER | ITEMCOUNT | TOTAL    |
+-------------+-----------+----------+
|       10126 |       617 | 57131.92 |
|       10127 |       540 | 58841.35 |
|       10135 |       607 | 55601.84 |
|       10142 |       577 | 56052.56 |
|       10165 |       670 | 67392.85 |
|       10181 |       522 | 55069.55 |
|       10192 |       585 | 55425.77 |
|       10204 |       619 | 58793.53 |
|       10207 |       615 | 59265.14 |
|       10212 |       612 | 59830.55 |
|       10222 |       717 | 56822.65 |
|       10287 |       595 | 61402.00 |
|       10310 |       619 | 61234.67 |
|       10312 |       601 | 55639.66 |
|       10390 |       603 | 55902.50 |
+-------------+-----------+----------+
```