select deptno, sum(sal) from emp

group by deptno;

DEPTNO SUM(SAL)

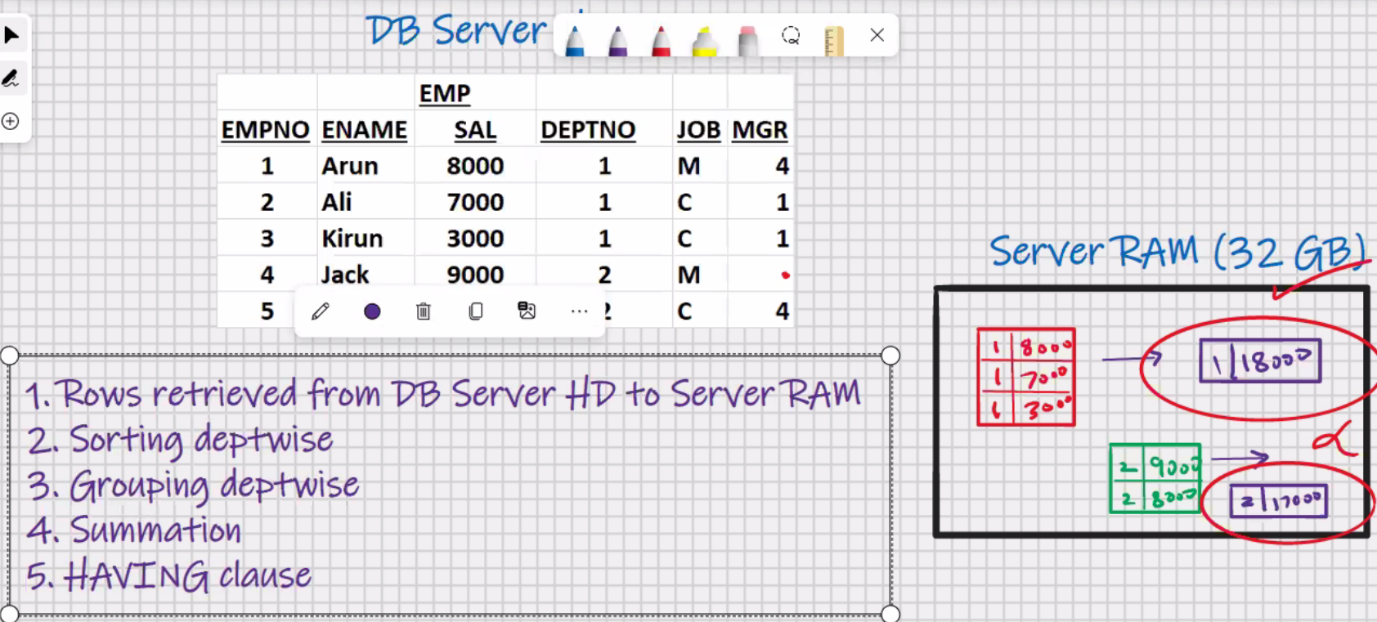
----------- -------------

1. 18000
2. 17000

select deptno, sum(sal) from emp

group by deptno;

**having** sum(sal) > 17000



DEPTNO SUM(SAL)

----------- -------------

1. 18000

* Having clause works after the summation is done
* Having clause is step number 5

select deptno, sum(sal) from emp

group by deptno;

**having** sum > 17000 -> Error

select deptno, sum(sal) from emp

where sal>7000

group by deptno;

* Where clause is used for searching
* Searching takes place in DB server HD
* Where clause is used to restrict the rows
* Where clause is used to retrieve the rows from DB server HD to server Ram
* Having clause works after the summation is done

select deptno, sum(sal) from emp

group by deptno

**having** where deptno = 1; -> this will work but it is inefficient

* Whichever column is present in select clause it can be used in **having** clause
* Its recommended that only **group functions** should be used in **having** clause

select deptno, sum(sal) from emp

group by deptno

having sum(sal)>17000 and sum(sal)<25000;

OR

select deptno, sum(sal) from emp

group by deptno

having sum(sal) between 17001 and 24999;

select deptno, sum(sal) from emp

group by deptno

having count(\*)=3;

select deptno, sum(sal) from emp

group by deptno

order by sum(sal);

DEPTNO SUM(SAL)

----------- -------------

1. 18000
2. 17000

select deptno, sum(sal) from emp

group by deptno

order by 2;

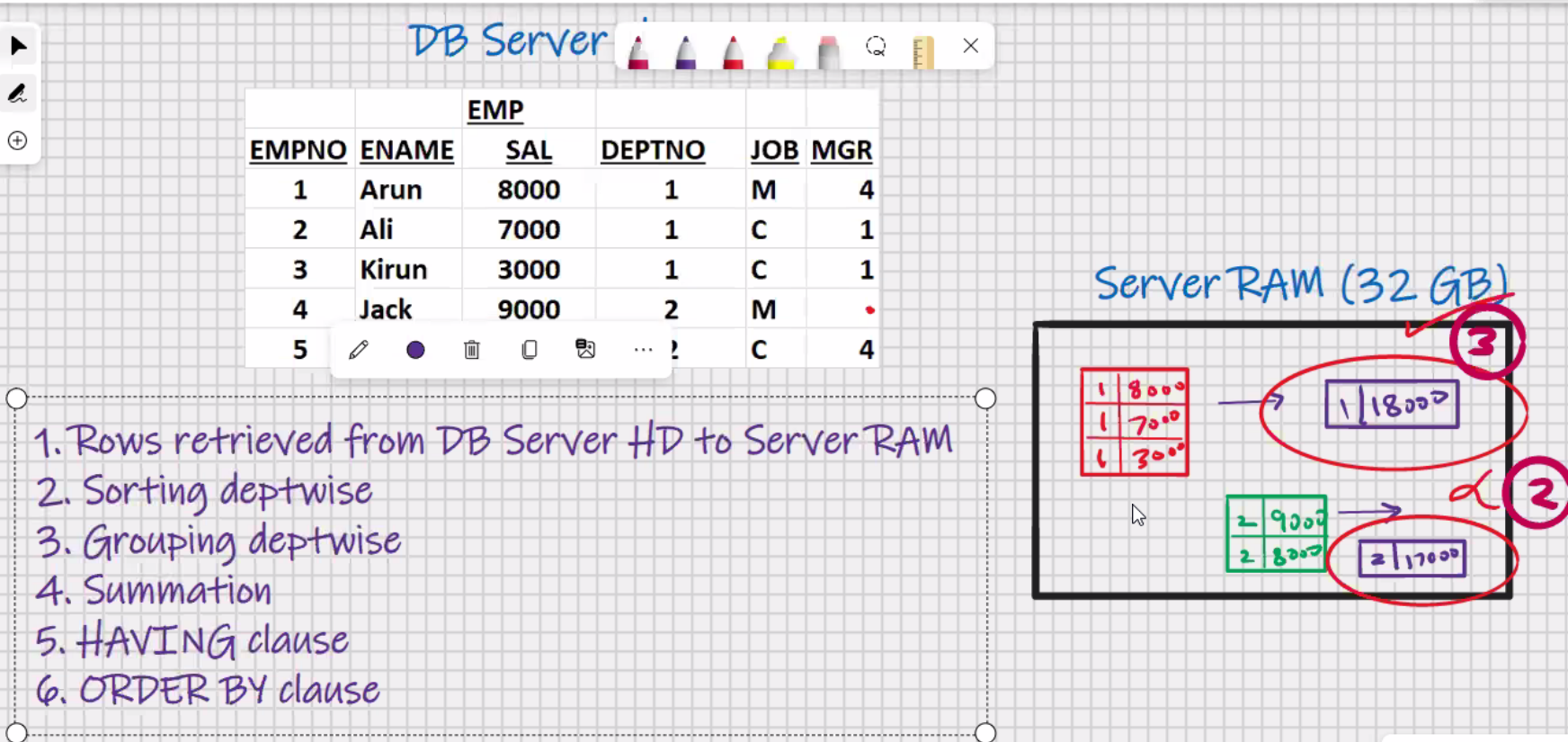
DEPTNO SUM(SAL)

----------- -------------

1. 18000

1 17000

* Order by is the last clause in select statement



Select deptno, sum(sal) from emp

Where sal>7000

Group by deptno

Having sum(sal) >10000

Order by 1;

DEPTNO SUM(SAL)

----------- -------------

2 17000

Select deptno, count(\*), min(sal), max(sal), sum(sal) from emp

Group by deptno

Having count(\*) >=3

Order by 1; -> where 1 is column number

In Oracle:-

Select deptno, sum(sal) from emp

Group by deptno;

DEPTNO SUM(SAL)

----------- -------------

1. 18000
2. 17000

Select sum(sal) from emp

Group by deptno;

SUM(SAL)

-------------

18000

17000

Select max(sum(sal)) from emp

Group by deptno; -> **nesting of group functions is allowed only in oracle RDBMS**

MAX(SUM(SAL))

-------------

18000

**In MySQL:**

Select deptno, sum(sal) from emp

Group by deptno;

DEPTNO SUM(SAL)

----------- -------------

1. 18000
2. 17000

select max(sum\_sal) from

(select sum(sal) sum\_sal from emp

Group by deptno) as tempp;

Max(sum\_sal)

-------------------

18000

**Matrix Report:**

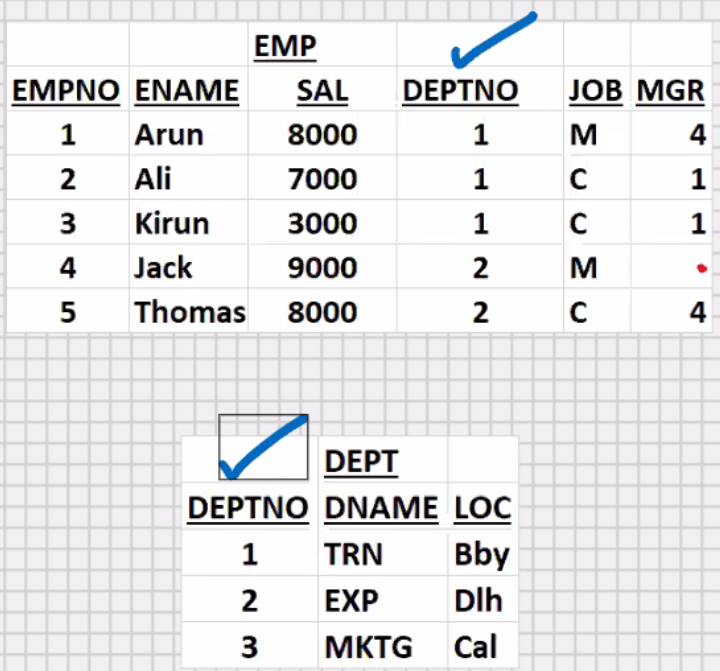
Select deptno, count(\*), min(sal), max(sal),sum(sal) from emp

Group by deptno

Order by 1;

**MySQL – SQL – JOINS**

* Very important topic for interview and job
* Data redundancy-> unnecessary duplication of data(wastage of HD space)
* Normalisation



Select ename, deptno from emp;

+--------+--------+

| ename | deptno |

+--------+--------+

| Arun | 1 |

| Ali | 1 |

| Kiran | 1 |

| Jack | 2 |

| Thomas | 2 |

+--------+--------+

**Dept -> driving table(must contain less no. of rows (work fast))**

**Emp -> driven table**

<<-----

select ename, dname from emp, dept

where dept.deptno = emp.deptno;

(tablename.columnname)

ename dname

--------- -------------

Arun Training

Ali Training

Kiran Training

Jack Exports

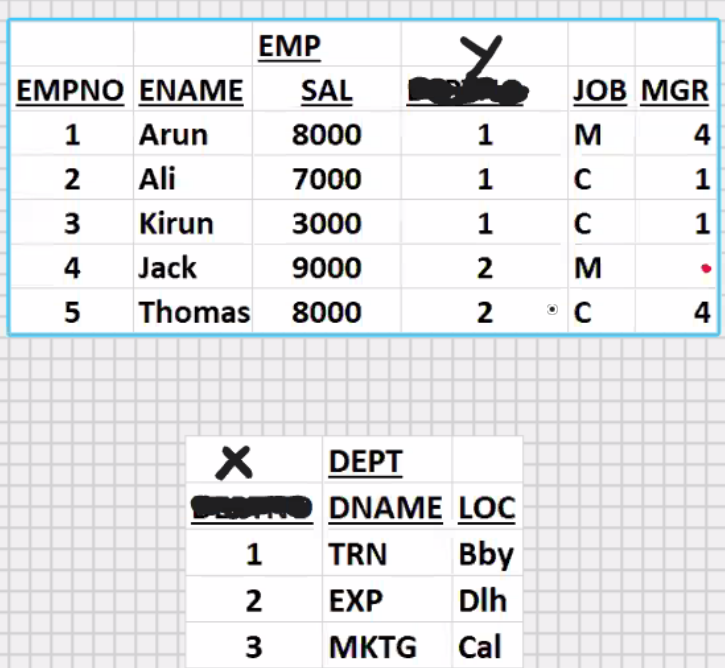
Thomas Exports

* To view the columns of 2 or more tables

select ename, dname from emp, dept

where emp.deptno = dept.deptno ;

* In order for the join to work faster, preferably the driving table should be table with “Lesser” number of rows
* Full table scan – entire table will be searched

select ename, dname from emp, dept

where emp.deptno = dept.deptno

order by dname;

select dname,ename from emp,dept

where dept.x=emp.y;

* The common column in both the

Tables is DEPTNO; the common column

In both the tables, the columnname need

Not be the same in both the tables, because the same column may have a different meaning elsewhere(e.g. import export)

* What matter is that the datatype of the common column in both the tables should be the same

Select dname, loc, ename, job, sal from emp, dept

Where dept.deptno=emp.deptno

Order by 1;

Select \* from emp, dept

Where dept.deptno=emp.deptno

Order by 1;

* Deptno column will display twice

Select dname, loc, ename, job, sal, **deptno** from emp, dept

Where dept.deptno=emp.deptno

Order by 1; 🡪> error for deptno column ambiguity

Select dname, loc, ename, job, sal, dept.**deptno** from emp, dept

Where dept.deptno=emp.deptno

Order by 1;

Select dname, loc, ename, job, sal, emp.**deptno** from emp, dept

Where dept.deptno=emp.deptno

Order by 1;

Select dept.dname, emp.loc, emp.ename, emp.job, emp.sal, emp.**deptno** from emp, dept

Where dept.deptno=emp.deptno

Order by 1;

* Always use tablename.columnname in select statement

Select dept.dname, emp.loc, emp.ename, emp.job, emp.sal, emp.**deptno** from emp, dept

Where x = y // different name then not required to use tablename

Order by 1;

Select deptno, sum(sal) from emp

Group by deptno;

DEPTNO SUM(SAL)

----------- -------------

1. 18000
2. 17000

Select dname, sum(sal) from emp, dept

Where dept.deptno= emp.deptno

Group by dname;

DEPTNO SUM(SAL)

----------- -------------

TRN 18000

EXP 17000

Select upper(dname), sum(sal) from emp, dept

Where dept.deptno= emp.deptno

Group by upper(dname)

Having sum(sal)>10000

Order by 2 desc;

DEPTNO SUM(SAL)

----------- -------------

TRN 18000

EXP 17000

**Types of Joins (5)**

1. **Equijoin**

* Join based on equality condition
* Shows matching rows of both the tables
* Uses:
* Data is stored in multiple tables and you want to view the columns of multiple tables
* View dname, ename
* View cname, sname
* Dname, order\_details
* Etc

Select dname, ename from emp, dept

Where dept.deptno = emp.deptno;

1. **Inequijoin**

* Join based on inequality condition

Select dname, ename from emp, dept

Where dept.deptno **!=** emp.deptno;

| ename | dname |

---------------------------------

| Arun | Exports |

| Ali | Exports |

| Kiran | Exports |

| Arun | Marketing |

| Ali | Marketing |

| Kiran | Marketing |

| Jack | Marketing |

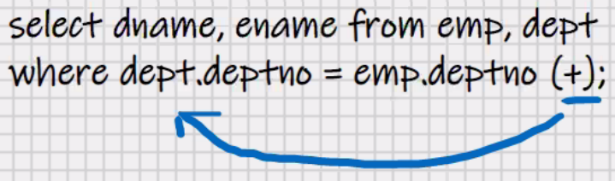
| Thomas | Marketing |

| Jack | Training |

| Thomas | Training |

* Shows non-matching rows of both the tables
* Uses:
* **EXCEPTION REPORTS**

1. **Outerjoin(works only in oracle)**

(dept-> do while emp-> for  )

Select dname, ename from emp, dept

Where dept.deptno = emp.deptno **(+)**;

LHS RHS

DNAME ENAME

---------- ----------

TRN ARUN

TRN ALI

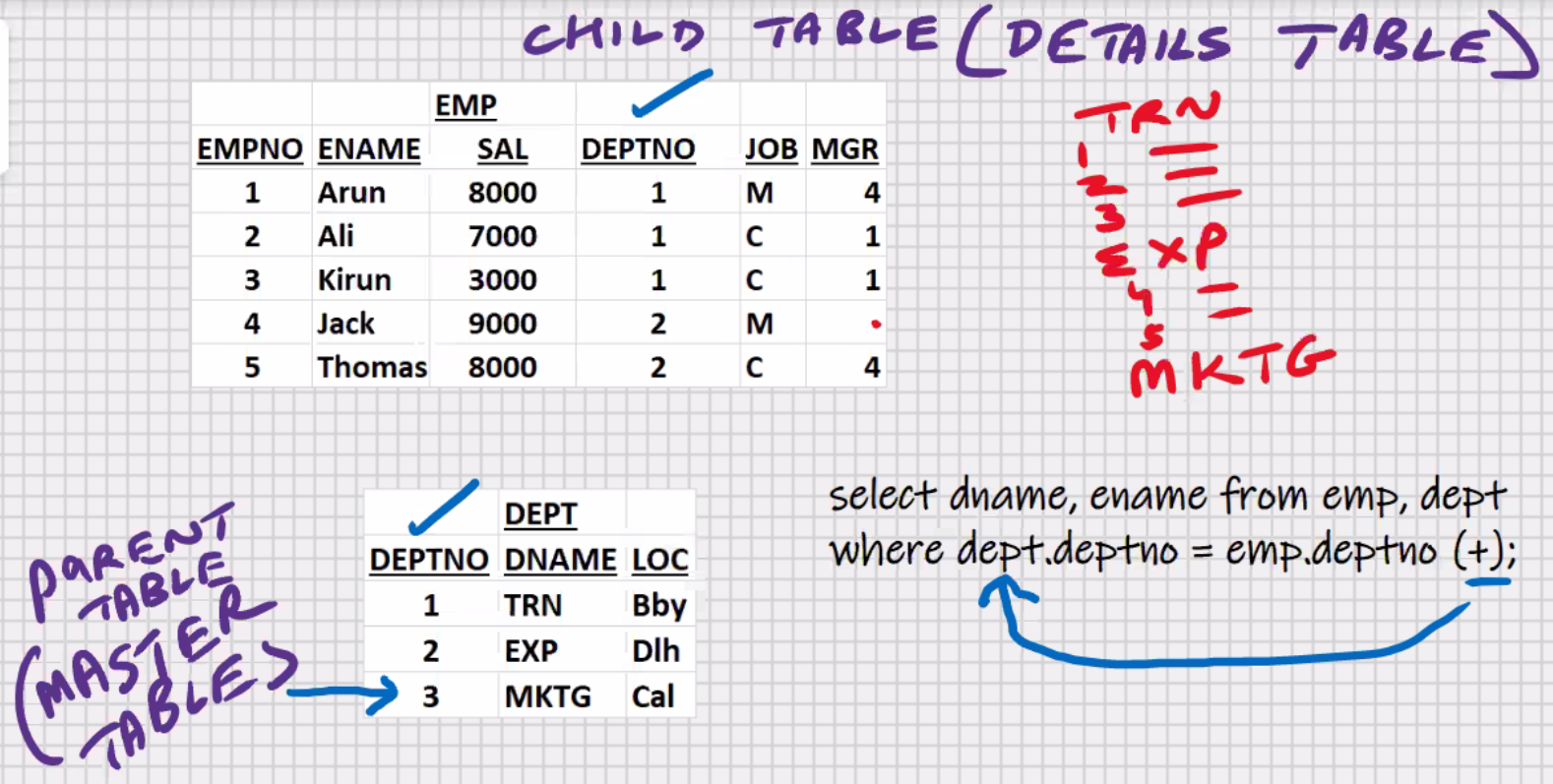
TRN KIRUN

EXP JACK

EXP THOMAS

MKTG .

* Shows matching rows of both the tables plus non matching rows of “Outer” table
* OUTER table -> table which is on Outer side of (+) sign
* OUTER table -> table which is on opposite side of (+) sign

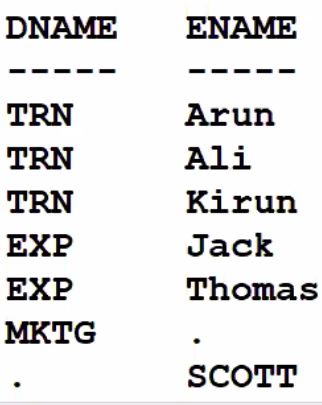


* Uses:
* **Master-detail report (Parent – child Report)**

Select dname, ename from emp, dept dept-> for

Where dept.deptno **(+)** = emp.deptno; emp->do while

LHS RHS

* **Half Outerjoin**:You can + sign on any side(LHS or RHS) but not on both the sides.
  + **Right outerjoin (+sign on RHS of where clause)**
  + **Left outerjoin (+sign on LHS of where clause)**
* **Full Outerjoin(based on Nested do while loop)**

Select dname, ename from emp, dept

Where dept.deptno **(+)** = emp.deptno

union

Select dname, ename from emp, dept

Where dept.deptno = emp.deptno (+);

* Shows matching rows of both the tables plus

Non matching rows of both the tables

**ANSI syntax for full Outerjoin:**

* Supported by all RDBMS including Oracle except for MySQL

Select dname, ename from emp **full** outer join dept

On (dept.deptno = emp.deptno);

**To achieve full outerjoin in MySQL :**

Select dname, ename from emp **right** outer join dept

On (dept.deptno = emp.deptno)

union

Select dname, ename from emp **left** outer join dept

On (dept.deptno = emp.deptno);

**ANSI syntax for Right Outerjoin:**

* Supported by all RDBMS including MySQL and Oracle

Select dname, ename from emp **right** outer join dept

On (dept.deptno = emp.deptno);

**ANSI syntax for Left Outerjoin:**

* Supported by all RDBMS including MySQL and Oracle

Select dname, ename from emp **left** outer join dept

On (dept.deptno = emp.deptno);

* (+) sign for outerjoin only works in Oracle RDBMS
* (+) sign for outerjoin Is not supported by any other RDBMS

--------------------------------------------------------------------------------------------

INNER join -> by default every join is an Inner join

Using the (+) or by using the keyword “Outer”, it becomes an Outer join

* Do not mention inner join in interviews, unless explicitly asked by interviewer

1. **Cartesian join (also known as Cross join)**

* Join without a where clause
* Every row of driving table is combined with each and every row of driven table
* Similar to taking cross product of two tables
* Shows all the combinations
* Uses:
* Printing purposes
* E.g in students table you have all the students names, in subjects table you have all the subjects name; when you’re printing the marksheets, every student name is combined with each and every subject name

Dept -> driving table

Emp -> driven table

Select dname, ename from emp , dept; -> **fast(I/O) between DB server HD and Server RAM is 3(the lesser the I/O faster it will be)**

Select dname, ename from dept, emp ; **-> slow(I/O) between DB server HD and Server RAM is 5(the more the I/O slower it will be)**

ename dname

----------- -----------

Arun Training

Arun Exports

Arun Marketing

Ali Training

Ali Exports

Ali Marketing

Kiran Training

Kiran Exports

Kiran Marketing

Jack Training

Jack Exports

Jack Marketing

Thomas Training

Thomas Exports

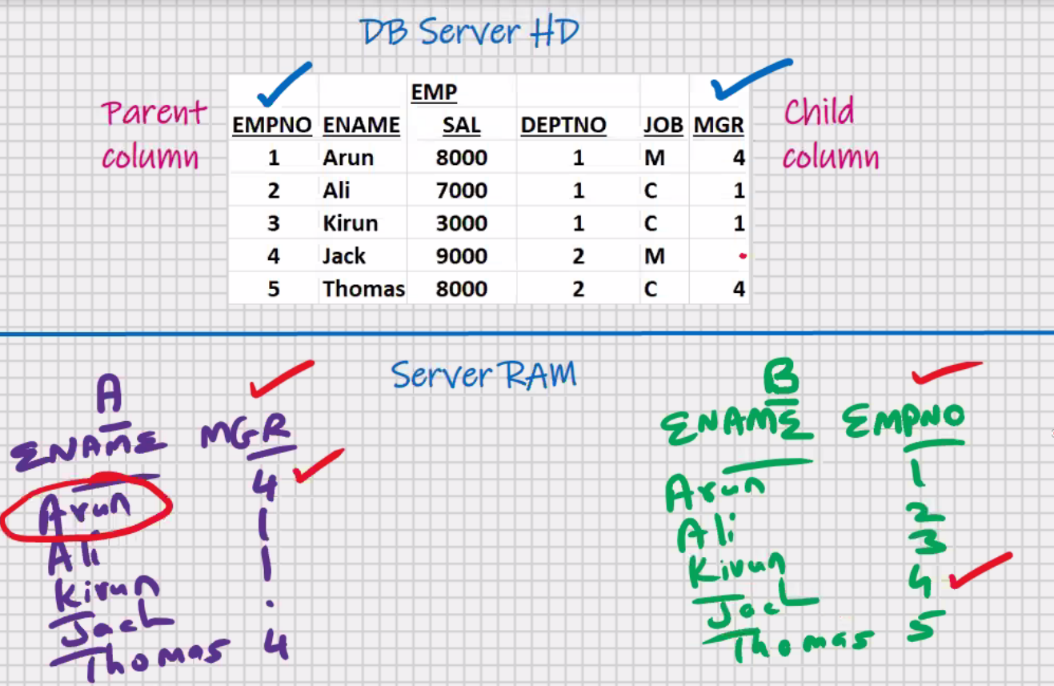
Thomas Marketing

1. **Self join**

* Joining a table to itself
* Used when parent column and child column both are present in the same table

a -> driving table b -> driven table

Select a.ename, b.ename from emp b, emp a

Where a.mgr = b.empno;

Employee Manager

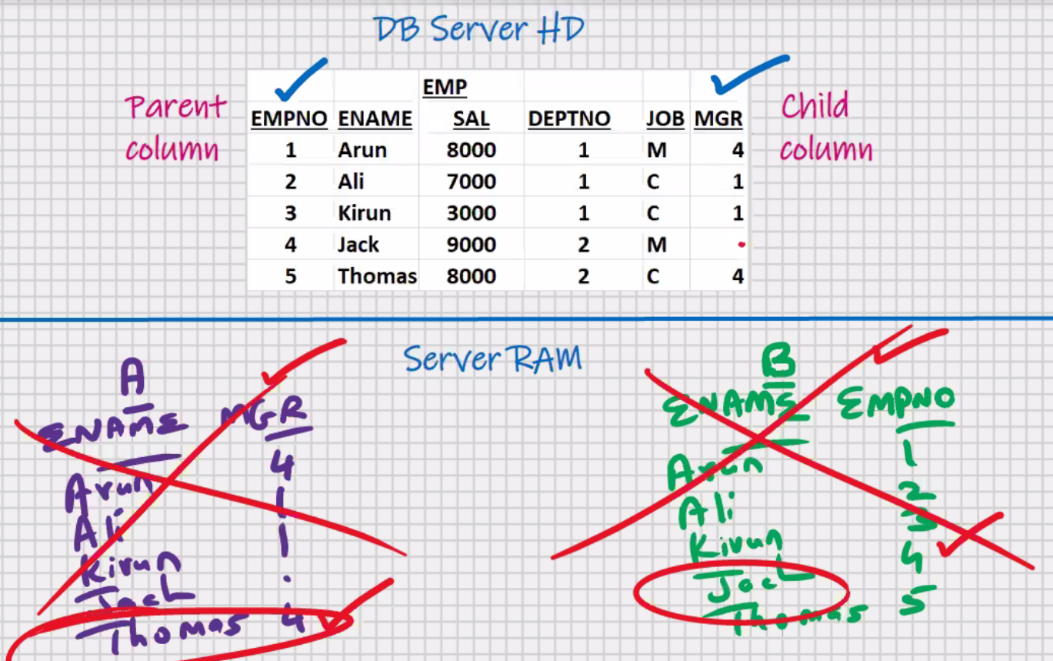
-----------------------------

Arun Jack

Ali Arun

Kiran Arun

Thomas Jack



Temporary tables

Destroy after completion

* Uses:
* Employee name , manager names
* **Slowest join (based on Recursion)**

Select dname, ename from emp e, dept d

Where d.deptno = e.deptno; -> BAD Example

* **Do not give alias to tablename unnecessarily, do so in rare event you write a self-join**

**1.Equijoin**

* Most frequently used (>90%) ; hence it is also known as Natural join

**2.Inequijoin (Non-equijoin)**

**3.Outerjoin**

**4.Cartesian Join (also known as Cross join) (fastest join because there is no where clause and hence there is no searching involved)**

**5.Self join (Based on Recursion) (slowest join)**

* **Internally all joins are based on Nested for loop**

**Except for the Outerjoin where one or more of the loops may be do while loop(rest are all for loop)**