PRACTICE ON MYSQL

**ALTER TABLE AP\_for**

**adding constraint, modifying column sizes, setting NULL/NOT NULL, DEFAULT values**

* Adding CHECK constraints to enforce business rules

alter table ap\_emp add constraint C\_EMP\_EMPNO check (EMPNO between 1000 and 9999);

alter table ap\_project add constraint C\_PROJECT\_PROJID check (PROJID between 100 and 999);

alter table ap\_dept add constraint C\_DEPT\_DEPTNO check (DEPTNO between 10 and 99);

alter table ap\_emp add constraint C\_EMP\_ELNAME check (ELNAME=upper(ELNAME));

* **Test that constraint is in force.**

insert into ap\_emp values (10001, 'Robert','Smith', 'TEMP',7369,sysdate()- interval 40 day,1000,null,10);

* Modify column to add DEFAULT value

alter table ap\_proemp alter HOURS set default 0 ;

* Modify column to allow for NULL

alter table ap\_emp modify job varchar(30) NULL;

* Modify column to change size and set NOT NULL

alter table ap\_emp modify job varchar(20) NOT NULL;

* Adding foreign key constrain

alter table ap\_emp add constraint FK\_EMP\_MGR foreign key (MGR) references ap\_emp(EMPNO);

**ALTER TABLE AP\_to add/remove column**

alter table ap\_emp add BIRTHDATE date;

alter table ap\_emp add ADDRESS varchar (50);

alter table ap\_emp drop column BIRTHDATE, drop column ADDRESS;

**Creating table from other table**

CREATE TABLE ap\_empTEST AS SELECT \* FROM ap\_emp -- CREATE A TABLE FROM EXISITNG TABLE

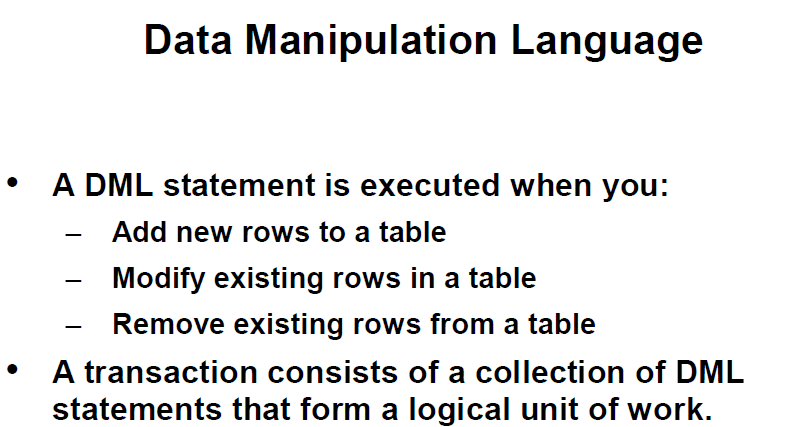
SELECT \* FROM ap\_empTEST -- RETRIVING/SELECTING ALL RECORDS OF A TABLE

TRUNCATE TABLE ap\_empTEST -- REMOVING/DELETING ALL RECORDS OF A TABLE

SELECT \* FROM ap\_empTEST-- RETRIVING/SELECTING ALL RECORDS OF A TABLE

DROP TABLE ap\_empTEST -- DROPPING A TABLE

SELECT \* FROM ap\_empTEST -- RETRIVING/SELECTING ALL RECORDS OF A TABLE



– CREATING A EMPTY TABLE FROM ANOTHER TABLE;

CREATE TABLE ap\_deptTEST AS SELECT \* FROM ap\_dept WHERE 1=2;

CREATE TABLE ap\_empTEST AS SELECT \* FROM ap\_emp WHERE 1=2;

--POPULATING TABLE FROM ANOTHER TABLE;

INSERT INTO ap\_deptTEST SELECT \* FROM ap\_dept;

INSERT INTO ap\_empTEST SELECT \* FROM ap\_emp;

COMMIT;

INSERT INTO ap\_deptTEST (deptno,dname,loc) VALUES ('TRAINING','AUSTIN');

INSERT INTO ap\_deptTEST (deptno,dname,loc) VALUES (NULL,'TRAINING','AUSTIN');

INSERT INTO ap\_deptTEST (deptno,dname,loc) VALUES (400,'TRAINING','AUSTIN');

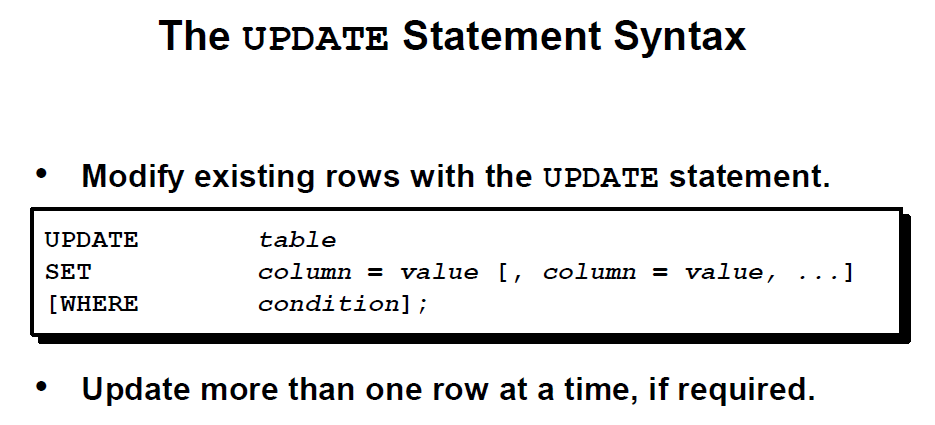
INSERT INTO ap\_deptTEST VALUES (50,'TRAINING','AUSTIN');

SELECT \* FROM ap\_deptTEST;

COMMIT;

INSERT INTO ap\_deptTEST VALUES (60,'LEGAL',NULL);

SELECT \* FROM ap\_deptTEST;



SELECT deptno

FROM ap\_empTEST

WHERE empno=7369;

UPDATE ap\_empTEST

SET deptno=30

WHERE empno=7369;

(IF THIS GENERATED AN ERROR IN SQL WORKBOOK: Error Code: 1175. You are using safe update mode and you tried to update a table without a WHERE that uses a KEY column. To disable safe mode, toggle the option in Preferences -> SQL Editor and reconnect.

TRY: SET SQL\_SAFE\_UPDATES = 0;)

**select \* from ap\_empTEST;**

**SELECT empno,deptno**

**FROM ap\_emptest;**

**UPDATE ap\_empTEST**

**SET sal=sal+100**

**WHERE job='CLERK';**

**UPDATE ap\_empTEST**

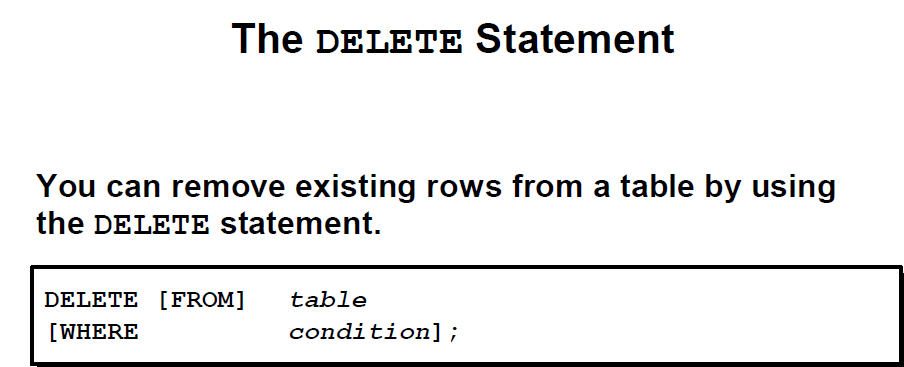
**SET comm=comm +100;**

**select \* from ap\_empTEST;**

**UPDATE ap\_empTEST**

**SET comm=IFNULL(comm,0)+100;**

**SELECT \* FROM ap\_empTEST;**



DELETE FROM ap\_deptTEST;

DELETE FROM ap\_empTEST

WHERE deptno=30;

DELETE FROM ap\_empTEST;

SELECT COUNT(\*) FROM ap\_empTEST;

DROP table ap\_empTEST;

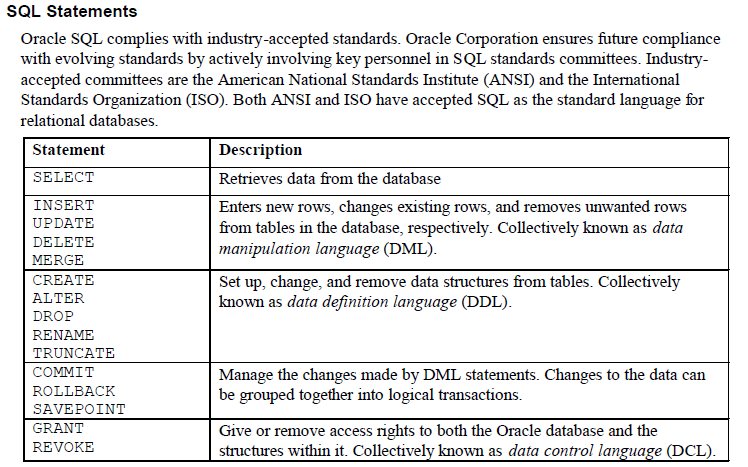
DROP table ap\_deptTEST;

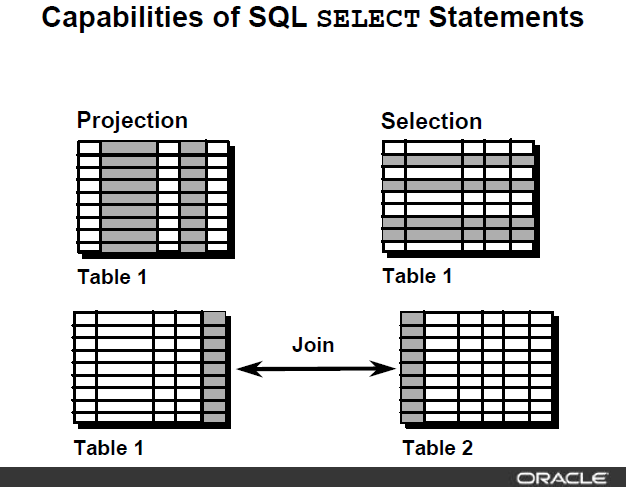
**delete from ap\_emp where ELNAME='Smith';**

**Error Code: 1451. Cannot delete or update a parent row: a foreign key constraint fails**

**SET FOREIGN\_KEY\_CHECKS=0; -- to disable them**

**SET FOREIGN\_KEY\_CHECKS=1; -- to re-enable them**





**Basic SELECT structure**

**SELECT < column list, expressions, literals>**

**FROM <table list>**

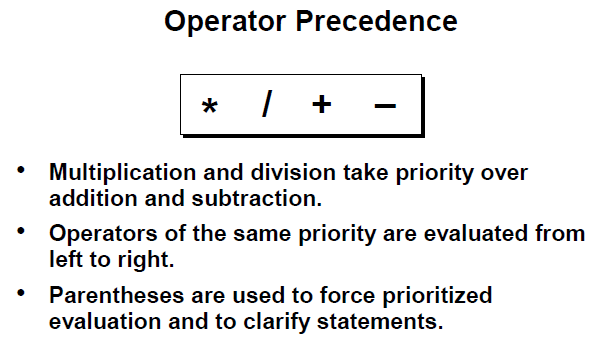
**WHERE <filter conditions with AND/OR/NOT logical operators)**

**GROUP BY <column list for aggregate functions COUNT/SUM/MIN/MAX/AVG etc.>**

**HAVING <filter conditions for grouping results>**

**ORDER BY <column list for sorting result set>**

**(SELECT and FORM clause are mandatory, all other clauses are optional and to be used as required by query result)**



SELECT \*

FROM ap\_emp;

SELECT empno, ELNAME, job, sal

FROM ap\_emp;

SELECT empno "Employee Number", ELNAME as name , job, sal "Monthly Salary USD"

FROM ap\_emp;

SELECT A.ELNAME, A.sal, A.sal+100, A.comm, A.comm+10

from ap\_emp A;

SELECT A.ELNAME, A.sal, A.sal+100, A.comm, IFNULL(A.comm,0)+10

FROM ap\_emp A;

SELECT DISTINCT job FROM ap\_emp;

SELECT DISTINCT job,deptno

FROM ap\_emp;

SELECT DISTINCT job,deptno

FROM ap\_emp

ORDER BY job, deptno;

SELECT DISTINCT job,deptno

FROM ap\_emp

ORDER BY deptno, job;

SELECT DISTINCT job,deptno

FROM ap\_emp

ORDER BY deptno, job desc;

SELECT ELNAME, sal

FROM ap\_emp

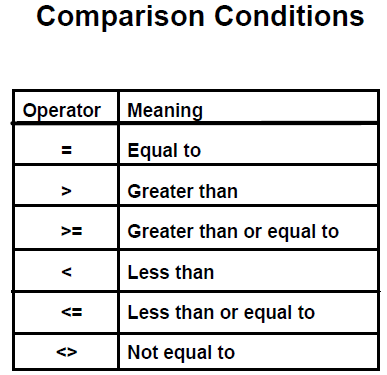
ORDER BY 2;

SELECT ELNAME, sal

FROM ap\_emp

ORDER BY 2 desc, 1;

**WHERE clause (SELECTION of results - Limiting result sets based upon the conditions)**



SELECT \*

FROM ap\_emp

WHERE sal>9000;

SELECT \*

FROM ap\_emp

WHERE sal<9000;

SELECT ELNAME, sal AS salary

FROM ap\_emp

WHERE sal>=9000

ORDER BY salary;

SELECT ELNAME, deptno

FROM ap\_emp

WHERE deptno=10

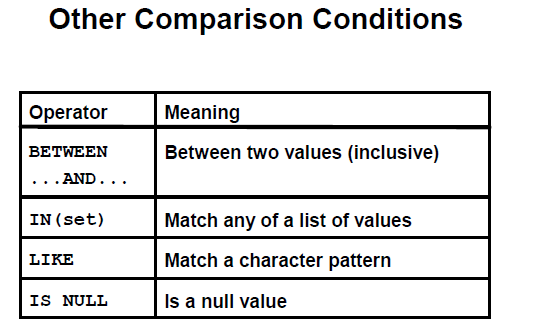
ORDER BY 1;

SELECT ELNAME, deptno

FROM ap\_emp

WHERE deptno<>10

ORDER BY 2, 1;



SELECT ELNAME, sal

FROM ap\_emp

WHERE sal BETWEEN 4500 AND 9000

ORDER BY sal;

SELECT ELNAME, deptno

FROM ap\_emp

WHERE deptno IN (10,30)

ORDER BY 2;

SELECT ELNAME, deptno

FROM ap\_emp

WHERE ELNAME like 'A%'

ORDER BY 1;

SELECT ELNAME, deptno

FROM ap\_emp

WHERE ELNAME like '%N'

ORDER BY 1;

SELECT ELNAME, deptno

FROM ap\_emp

WHERE ELNAME like '%A%'

ORDER BY 1;

SELECT ELNAME, sal, comm

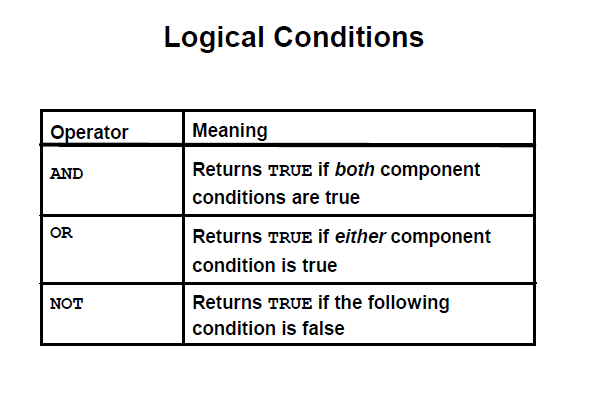
FROM ap\_emp

WHERE comm IS NULL;

SELECT ELNAME, sal, comm

FROM ap\_emp

WHERE comm IS NOT NULL;



SELECT ELNAME, sal, deptno

FROM ap\_emp

WHERE sal>5000 AND deptno=10;

SELECT ELNAME,sal, deptno

FROM ap\_emp

WHERE sal>5000 OR deptno=10;

SELECT ELNAME,sal, deptno, job

FROM ap\_emp

WHERE sal<5000 OR deptno=10 AND job='ANALYST';

SELECT ELNAME,sal, deptno, job

FROM ap\_emp

WHERE (sal<5000 OR deptno=10) AND job='ANALYST';

SELECT ELNAME,sal, deptno, job

FROM ap\_emp

WHERE sal<5000 OR deptno=10 OR job='ANALYST';

SELECT ELNAME, sal

FROM ap\_emp

WHERE sal NOT BETWEEN 4500 AND 9000

ORDER BY sal;

SELECT ELNAME,deptno

FROM ap\_emp

WHERE deptno NOT IN (10,30)

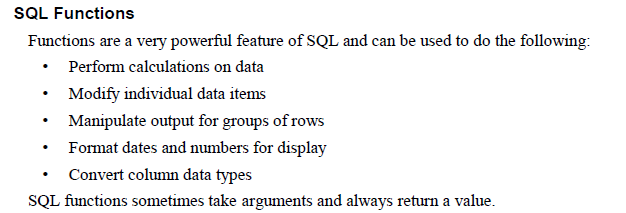
ORDER BY 2;

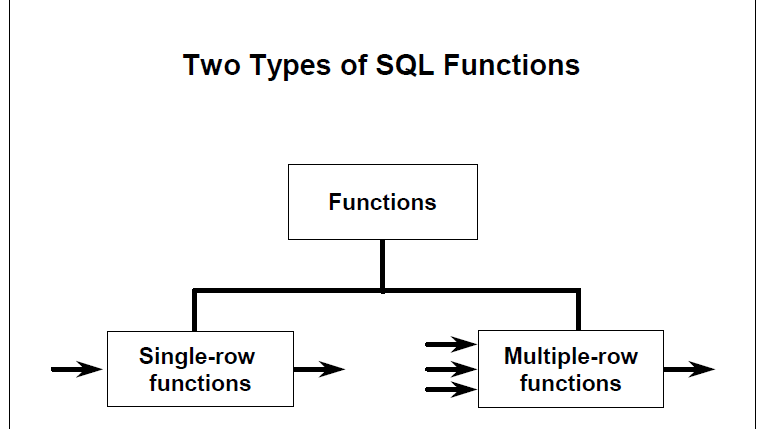
SELECT ELNAME, deptno

FROM ap\_emp

WHERE ELNAME NOT like 'A%'

ORDER BY 1;





**MySQL SQL FUCTIONS**

<https://dev.mysql.com/doc/refman/8.0/en/functions.html>

SELECT ELNAME, lower(ELNAME), substr(ELNAME, 1,2), substr(ELNAME, -3,2), substr(ELNAME, -2), lower(substr( ELNAME, 1,2))

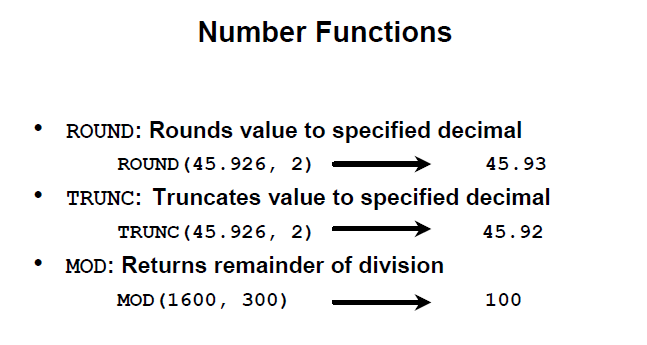
FROM ap\_emp

WHERE UPPER(JOB)='CLERK';

SELECT ELNAME, LENGTH(ELNAME), sal, LPAD(sal, 10,0)

FROM ap\_emp

WHERE UPPER(JOB)='CLERK';



SELECT ROUND(45.926,2), ROUND( 45.926) FROM dual;

SELECT TRUNCATE(45.926,2), TRUNCATE( 45.926, 0) FROM dual;

SELECT ELNAME, sal, MOD(sal, 1000)

FROM ap\_emp;

**DATE ARITHMETIC**

**Text

Description automatically generated**

**SELECT** sysdate()

**FROM** DUAL;

**SELECT** date\_add(sysdate(), interval 14 day)

**FROM** DUAL;

**SELECT** date\_add(sysdate(), interval -14 day)

**FROM** DUAL;

**SELECT** date\_sub(sysdate(), interval 14 day)

**FROM** DUAL;

**Text

Description automatically generated**

**A screenshot of a computer

Description automatically generated with medium confidence**

**Graphical user interface, text, application, email

Description automatically generated**

The following table illustrates the specifiers and their meanings that you can use to construct a date format string:

| **Specifier** | **Meaning** |
| --- | --- |
| %a | Three-characters abbreviated weekday name e.g., Mon, Tue, Wed, etc. |
| %b | Three-characters abbreviated month name e.g., Jan, Feb, Mar, etc. |
| %c | Month in numeric e.g., 1, 2, 3…12 |
| %D | Day of the month with English suffix e.g., 0th, 1st, 2nd, etc. |
| %d | Day of the month with leading zero if it is 1 number e.g., 00, 01,02, …31 |
| %e | Day of the month without leading zero e.g., 1,2,…31 |
| %f | Microseconds in the range of 000000..999999 |
| %H | Hour in 24-hour format with leading zero e.g., 00..23 |
| %h | Hour in 12-hour format with leading zero e.g., 01, 02…12 |
| %I | Same as %h |
| %i | Minutes with leading zero e.g., 00, 01,…59 |
| %j | Day of year with leading zero e.g., 001,002,…366 |
| %k | Hour in 24-hour format without leading zero e.g., 0,1,2…23 |
| %l | Hour in 12-hour format without leading zero e.g., 1,2…12 |
| %M | Full month name e.g., January, February,…December |
| %m | Month name with leading zero e.g., 00,01,02,…12 |
| %p | AM or PM, depending on other time specifiers |
| %r | Time in 12-hour format hh:mm:ss AM or PM |
| %S | Seconds with leading zero 00,01,…59 |
| %s | Same as %S |
| %T | Time in 24-hour format hh:mm:ss |
| %U | Week number with leading zero when the first day of week is Sunday e.g., 00,01,02…53 |
| %u | Week number with leading zero when the first day of week is Monday e.g., 00,01,02…53 |
| %V | Same as %U; it is used with %X |
| %v | Same as %u; it is used with %x |
| %W | Full name of weekday e.g., Sunday, Monday,…, Saturday |
| %w | Weekday in number (0=Sunday, 1= Monday,etc.) |
| %X | Year for the week in four digits where the first day of the week is Sunday; often used with %V |
| %x | Year for the week, where the first day of the week is Monday, four digits; used with %v |
| %Y | Four digits year e.g., 2000 and 2001. |
| %y | Two digits year e.g., 10,11,and 12. |
| %% | Add percentage (%) character to the output |

The following are some commonly used date format strings:

| **DATE\_FORMAT string** | **Formatted date** |
| --- | --- |
| %Y-%m-%d | 2013-07-04 |
| %e/%c/%Y | 4/7/2013 |
| %c/%e/%Y | 7/4/2013 |
| %d/%m/%Y | 4/7/2013 |
| %m/%d/%Y | 7/4/2013 |
| %e/%c/%Y %H:%i | 4/7/2013 11:20 |
| %c/%e/%Y %H:%i | 7/4/2013 11:20 |
| %d/%m/%Y %H:%i | 4/7/2013 11:20 |
| %m/%d/%Y %H:%i | 7/4/2013 11:20 |
| %e/%c/%Y %T | 4/7/2013 11:20 |
| %c/%e/%Y %T | 7/4/2013 11:20 |
| %d/%m/%Y %T | 4/7/2013 11:20 |
| %m/%d/%Y %T | 7/4/2013 11:20 |
| %a %D %b %Y | Thu 4th Jul 2013 |
| %a %D %b %Y %H:%i | Thu 4th Jul 2013 11:20 |
| %a %D %b %Y %T | Thu 4th Jul 2013 11:20:05 |
| %a %b %e %Y | Thu Jul 4 2013 |
| %a %b %e %Y %H:%i | Thu Jul 4 2013 11:20 |
| %a %b %e %Y %T | Thu Jul 4 2013 11:20:05 |
| %W %D %M %Y | Thursday 4th July 2013 |
| %W %D %M %Y %H:%i | Thursday 4th July 2013 11:20 |
| %W %D %M %Y %T | Thursday 4th July 2013 11:20:05 |
| %l:%i %p %b %e, %Y | 7/4/2013 11:20 |
| %M %e, %Y | 4-Jul-13 |
| %a, %d %b %Y %T | Thu, 04 Jul 2013 11:20:05 |

SELECT date\_format(sysdate(), '%d-%b-%y %H:%i:%s ')

FROM dual;

SELECT date\_format(sysdate(), '%d-%b-%y %h:%i:%s %p')

FROM dual;

SELECT date\_format(sysdate(), '%d-%b-%Y %h:%i:%s %p')

FROM dual;

SELECT date\_format(sysdate(), '%d-%bth,%Y hh:%i:%s AM')

FROM dual;

SELECT date\_format(sysdate(), '%D-%b “of” %Y %h:%i:%s %p')

FROM dual;

SELECT date\_format(sysdate(), '%d')

FROM dual;

SELECT date\_format(sysdate(), '%a')

FROM dual;

SELECT date\_format(sysdate(), '%b')

FROM dual;

SELECT date\_format(sysdate(), '%c')

FROM dual;

SELECT date\_format(sysdate(), '%D')

FROM dual;

SELECT date\_format(sysdate(), '%e')

FROM dual;

SELECT date\_format(sysdate(), '%f')

FROM dual;

SELECT date\_format(sysdate(), '%j')

FROM dual;

SELECT date\_format(sysdate(), '%k')

FROM dual;

SELECT date\_format(sysdate(), '%l')

FROM dual;

SELECT date\_format(sysdate(), '%M')

FROM dual;

SELECT date\_format(sysdate(), '%m')

FROM dual;

SELECT date\_format(sysdate(), '%p')

FROM dual;

SELECT date\_format(sysdate(), '%r')

FROM dual;

SELECT date\_format(sysdate(), '%T')

FROM dual;

SELECT date\_format(sysdate(), '%U')

FROM dual;

SELECT date\_format(sysdate(), '%u')

FROM dual;

SELECT date\_format(sysdate(), '%V')

FROM dual;

SELECT date\_format(sysdate(), '%v')

FROM dual;

SELECT date\_format(sysdate(), '%W')

FROM dual;

**The EXTRACT() function extracts a part from a given date.**

**Syntax**

EXTRACT(part FROM date)

## Parameter Values

## 

SELECT EXTRACT(HOUR FROM sysdate());

SELECT EXTRACT(MINUTE FROM sysdate());

SELECT EXTRACT(SECOND FROM sysdate());

SELECT EXTRACT(DAY FROM sysdate());

SELECT EXTRACT(WEEK FROM sysdate());

SELECT EXTRACT(MONTH FROM sysdate());

SELECT EXTRACT(QUARTER FROM sysdate());

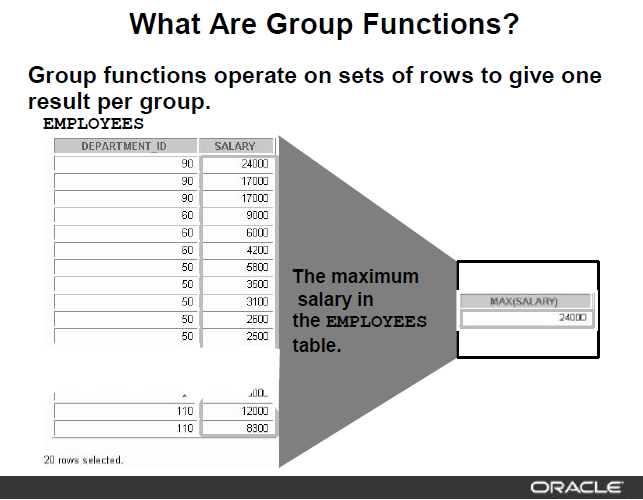
SELECT EXTRACT(YEAR FROM sysdate());

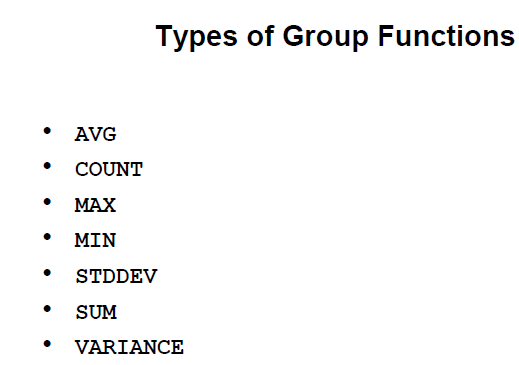
SELECT EXTRACT(YEAR\_MONTH FROM sysdate());

**FORMAT NUMBER TO STRINGS**

SELECT sal, CONCAT('$',FORMAT(sal, 0)) from ap\_emp;

SELECT sal, CONCAT('$',FORMAT(sal, 2)) from ap\_emp;





SELECT COUNT(\*), MIN(sal), MAX(sal), AVG(sal), SUM(sal)

FROM ap\_emp

WHERE deptno=10;

SELECT MIN(hiredate), MAX(hiredate)

FROM ap\_emp

WHERE deptno=20;

SELECT COUNT( DISTINCT job)

FROM ap\_emp

WHERE deptno=20;

SELECT AVG(comm) FROM ap\_emp;

SELECT AVG(IFNULL(comm,0)) from ap\_emp;

SELECT deptno, SUM(sal)

FROM ap\_emp

GROUP BY deptno

ORDER BY 2;

SELECT deptno, job, ROUND(AVG(sal),0)

FROM ap\_emp

GROUP BY deptno, job

ORDER BY 1,2;

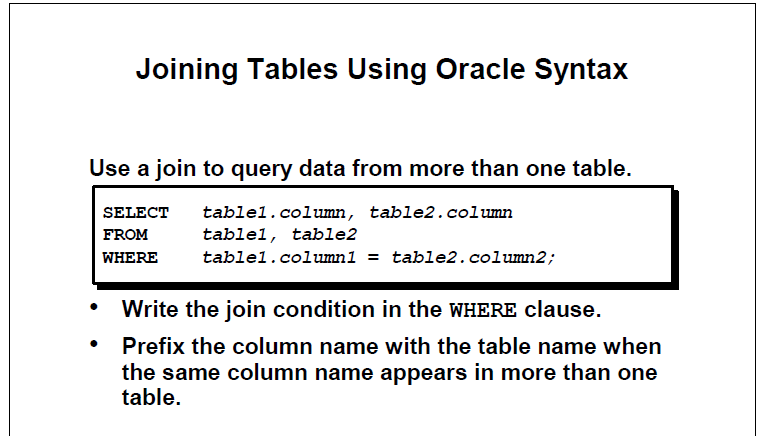
SELECT deptno, job, ROUND(AVG(sal))

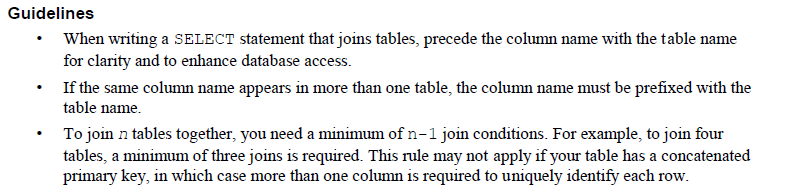
FROM ap\_emp

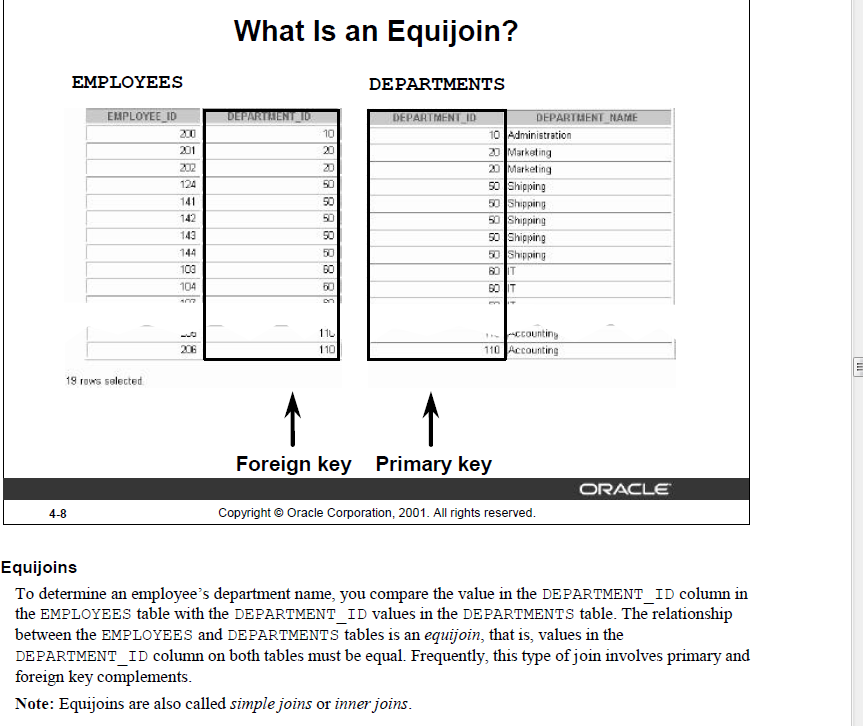
GROUP BY deptno, job

HAVING AVG(sal)>3000

ORDER BY 1,2;







SELECT a.empno, a.ELNAME,a.deptno, b.dname

FROM ap\_emp a INNER JOIN ap\_dept b ON a.deptno=b.deptno

order by 3

;

SELECT a.empno, a.ELNAME,a.deptno, b.dname

FROM ap\_emp a JOIN ap\_dept b ON a.deptno=b.deptno

order by 3

;

SELECT a.empno, a.projid,b.pname,a.hours

FROM ap\_proemp a JOIN ap\_project b ON a.projid=b.projid

WHERE a.hours>50

ORDER BY 1

;

SELECT a.empno, a.ELNAME,a.deptno, b.dname,b.loc,c.hours, d.pname

FROM ap\_emp a JOIN ap\_dept b ON a.deptno=b.deptno JOIN ap\_proemp c ON a.empno=c.empno JOIN ap\_project d ON c.projid=d.projid

WHERE b.loc IN ('NEW YORK', 'DALLAS') AND c.hours>40

ORDER BY 1;

**NATURAL JOIN: when FK and PK column of same name, don’t use join condition and column alias .**

SELECT empno, ELNAME,deptno, dname

FROM ap\_emp NATURAL JOIN ap\_dept;

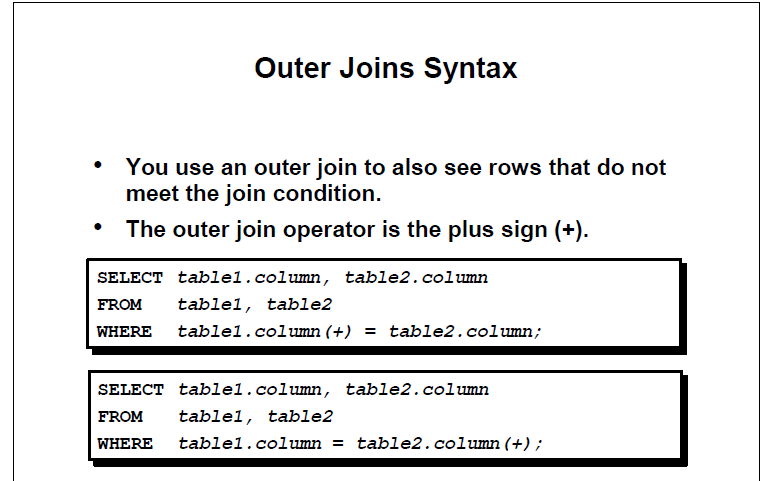
**NON-EQUIJOIN: Join based on other than equality operator**

SELECT a.empno, a.ELNAME,a.sal,b.grade, b.losal, b.hisal

FROM ap\_emp a , ap\_salgrade b

WHERE a.sal between b.losal AND b.hisal

ORDER BY b.grade;



SELECT a.empno, a.ELNAME,b.deptno,b.dname

FROM ap\_emp a RIGHT OUTER JOIN ap\_dept b ON a.deptno=b.deptno

ORDER BY a.deptno desc

;

SELECT a.empno, a.ELNAME,b.deptno,b.dname

FROM ap\_emp a LEFT OUTER JOIN ap\_dept b ON a.deptno=b.deptno

ORDER BY a.deptno desc

;

insert into ap\_emp values (9999, 'JACKIE','CHAN', 'AMBASSADOR',7839,sysdate(),15000,1000,null);

To achieve FULL OUTETR JOIN use UNION of RIGHT OUTER JOIN and LEFT outer join

SELECT a.empno, a.ELNAME,b.deptno,b.dname

FROM ap\_emp a RIGHT OUTER JOIN ap\_dept b ON a.deptno=b.deptno

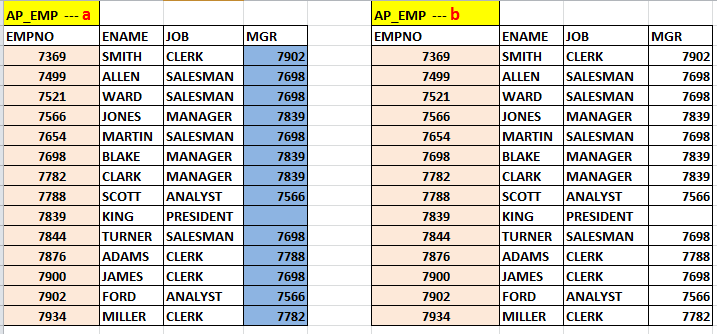
union

SELECT a.empno, a.ELNAME,b.deptno,b.dname

FROM ap\_emp a LEFT OUTER JOIN ap\_dept b ON a.deptno=b.deptno

;

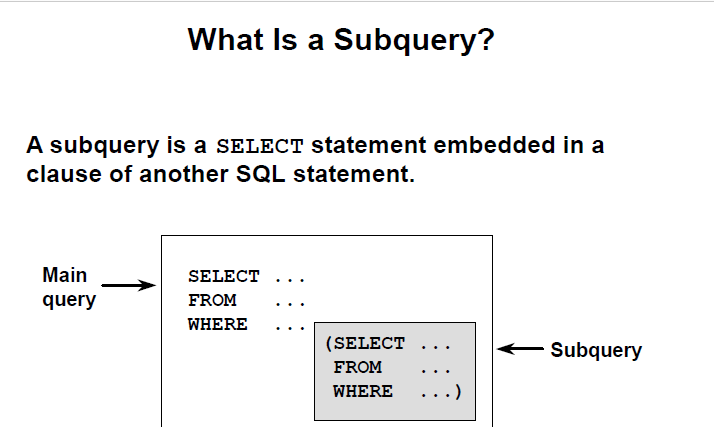
**SELF JOIN**

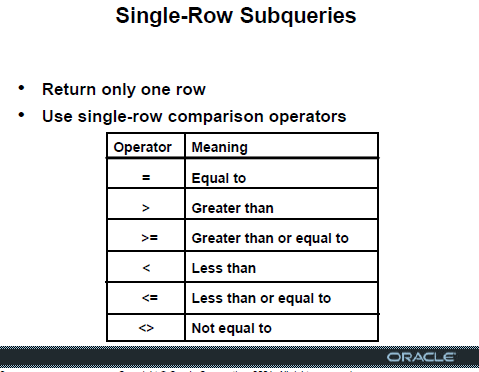


SELECT a.empno,a.ELNAME,a.mgr, b.empno " Manager ID", b.ELNAME "Manager Name"

FROM ap\_emp a JOIN ap\_emp b ON a.mgr=b.empno;

**USING SUBQUERY**





SELECT ELNAME, sal

FROM ap\_emp

WHERE sal>( SELECT sal FROM ap\_emp WHERE ELNAME='ALLEN');

SELECT ELNAME, job

FROM ap\_emp

WHERE deptno=(SELECT deptno FROM ap\_dept WHERE loc='NEW YORK');

SELECT ELNAME, job,sal

FROM ap\_emp

WHERE sal>(SELECT avg(sal) FROM ap\_emp);

SELECT ELNAME, job,sal

FROM ap\_emp

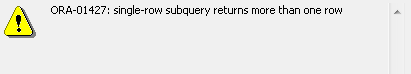
WHERE sal>(SELECT avg(sal) FROM ap\_emp) AND

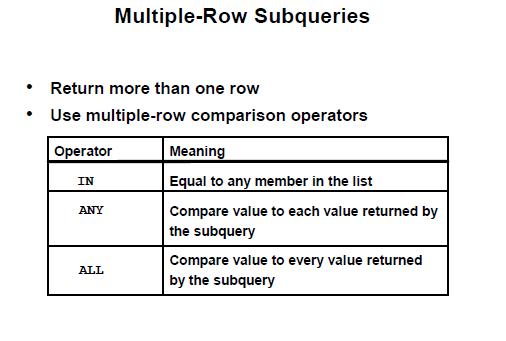
deptno= (SELECT deptno FROM ap\_dept where loc ='NEW YORK');

SELECT ELNAME, sal

FROM ap\_emp

WHERE sal > ( SELECT avg(sal) FROM ap\_emp GROUP BY deptno);





SELECT ELNAME, sal

FROM ap\_emp

WHERE sal >= ALL ( SELECT avg(sal) FROM ap\_emp GROUP BY deptno);

SELECT ELNAME, sal

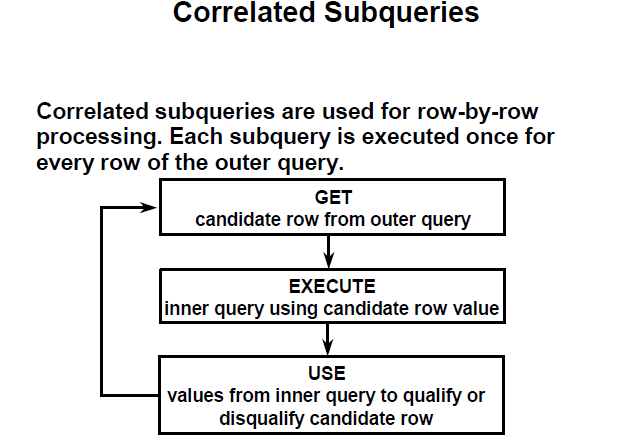
FROM ap\_emp

WHERE sal > ANY ( SELECT avg(sal) FROM ap\_emp GROUP BY deptno);

SELECT ELNAME, sal

FROM ap\_emp

WHERE sal IN ( SELECT avg(sal) FROM ap\_emp GROUP BY deptno);

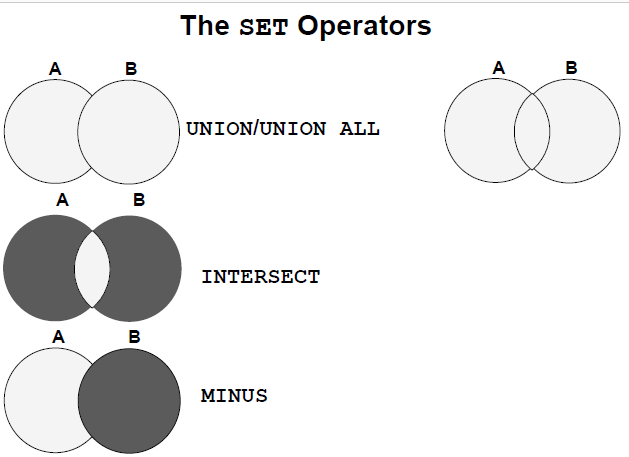


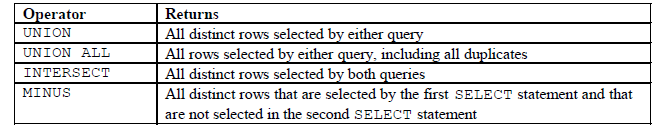
**List employees whose salary is higher than their respective department’s average salary**

SELECT a.empno,a.deptno, a.sal

FROM ap\_emp a

WHERE a.sal > (SELECT AVG(sal) deptavg FROM ap\_emp b WHERE a.deptno=b.deptno);





SELECT empno, ELNAME, deptno,job

FROM ap\_emp

WHERE deptno in (20,30)

**UNION**

SELECT empno, ELNAME,deptno, job

FROM ap\_emp

WHERE job='ANALYST'

ORDER BY deptno,job;

SELECT empno, ELNAME, deptno,job

FROM ap\_emp

WHERE deptno in (20,30)

**UNION ALL**

SELECT empno, ELNAME,deptno, job

FROM ap\_emp

WHERE job='ANALYST'

ORDER BY deptno,job;

* FOLLOWING IS EQUIVALENT TO “INTERSECT”

SELECT empno, ELNAME, deptno,job

FROM ap\_emp

WHERE deptno in (20,30)

**AND empno IN**

(SELECT empno

FROM ap\_emp

WHERE job='ANALYST')

ORDER BY deptno,job;

* FOLLOWINNG IS EQUIPVALENT TO “MINUS”

SELECT empno, ELNAME, deptno,job

FROM ap\_emp

WHERE deptno in (20,30)

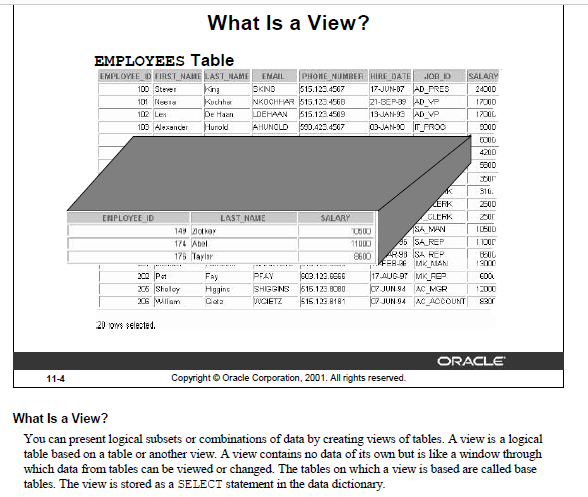
**AND empno NOT IN**

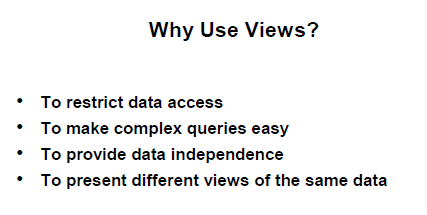
(SELECT empno

FROM ap\_emp

WHERE job='ANALYST')

ORDER BY deptno,job;





**== SIMPLE, SINGLE TABLE VIEW**

create or replace view testview

as

select \* from ap\_emp

where deptno=30

;

SELECT \* FROM testview;

update testview

set sal=sal+100

where empno=7499;

SELECT \* FROM testview;

**== READ ONLY view**

create or replace

DEFINER = CURRENT\_USER

SQL SECURITY DEFINER

view testview

as

select \* from ap\_emp

where deptno=30 ;

**== VIEW WITH CHECK OPTION**

create or replace view testview

as

select \* from ap\_emp

where deptno=30

with CHECK OPTION;

SELECT \* FROM testview;

update testview

set sal=sal+100

where empno=7499;

UPDATE TESTVIEW

SET DEPTNO=20

WHERE EMPNO=7499;

== COMPLEX, MULTI TABLES / AGGREGATE functions

CREATE OR REPLACE VIEW dept30\_V

AS

SELECT a.empno "Employee Number", a.hiredate "Hire Date", a. sal "Monthly Salary", b.dname "Department Name", b. loc "Location"

FROM ap\_emp a, ap\_dept b

WHERE a.deptno=b.deptno AND

a.deptno=30 ;

SELECT \* FROM dept30\_V;

SHOW FULL TABLES

WHERE table\_type = 'VIEW' and Tables\_in\_practice = 'dept30\_v';

CREATE OR REPLACE VIEW emp\_proj\_detail\_V

AS

SELECT a.empno AS Employee, a.ELNAME AS name, a.hiredate AS HireDate, a. sal AS Salary, b.dname AS Department, c.pname AS Project, d.hours AS Hours

FROM ap\_emp a, ap\_dept b, ap\_project c, ap\_proemp d

WHERE a.deptno=b.deptno AND

a.empno=d.empno AND

d.projid=c.projid

ORDER BY 4;

SELECT \* FROM emp\_proj\_detail\_V;

SELECT name,sum(hours)

FROM emp\_proj\_detail\_V

GROUP BY name

ORDER by 2;

CREATE OR REPLACE VIEW dept\_summary\_V

(name,location,minsal,maxsal,avgsal)

AS

SELECT dname , loc, MIN(sal), MAX(sal) , AVG(sal)

FROM ap\_emp a, ap\_dept b

WHERE a.deptno=b.deptno

GROUP BY dname,loc;

SELECT \* FROM dept\_summary\_V;

**==== SUBQUERY in FROM CLAUSE**

* **List employee number, name, department, salary along with their respective department’s total number of employee, total salary, average salary, minimum, and maximum salary**

**SELECT a.empno, a.ELNAME, a.deptno,sal, b.deptempcnt , b.depttotsal , round(b.deptavgsal) deptavgsal ,**

**b.deptminsal, b.deptmaxsal**

**FROM ap\_emp a, ( select deptno, count(\*) deptempcnt, sum(sal) depttotsal, avg(sal) deptavgsal, min(sal)**

**deptminsal, max(sal) deptmaxsal from ap\_emp group by deptno ) b**

**WHERE a.deptno=b.deptno ;**

**==== WITH CLAUSE [just like a temporary table that can be used multiple times in a query as needed**

**WITH DEPTVAL AS**

**( select deptno, count(\*) deptempcnt, sum(sal) depttotsal, avg(sal) deptavgsal, min(sal) deptminsal, max(sal) deptmaxsal from ap\_emp group by deptno)**

**SELECT a.empno, a.ELNAME, a.deptno,sal, b.deptempcnt , b.depttotsal , round(b.deptavgsal) deptavgsal, b.deptminsal, b.deptmaxsal**

**FROM ap\_emp a, DEPTVAL b**

**WHERE a.deptno=b.deptno ;**

**-- Subquery in SELECT clause**

**-- List of department name, and total number of employees in those department**

**select dname name, (select count(\*) from ap\_emp B where A.deptno= B.deptno) as Num\_Of\_Employees**

**from ap\_dept A ;**

**-- Subquery in HAVING clause**

**-- Department wise Total Salary for those departments which has total salary more than that of department 30**

**select deptno,sum(sal) Total\_Salary**

**from ap\_emp**

**group by deptno**

**having sum(sal) >= (select sum(sal) from ap\_emp where deptno=30);**

**== CASE statement [Conditional processing]**

**Increase employee’s salary by $500 if their salary is more than $5000, increase by $300 if their salary is over $3000, otherwise increase by $100.**

**select sal, case**

**when sal>5000 then sal+500**

**when sal>3000 then sal+300**

**else**

**sal+100**

**end “Salary Raise”**

**from ap\_emp;**

**update ap\_emp**

**set sal = case**

**when sal>5000 then sal+500**

**when sal>3000 then sal+300**

**else**

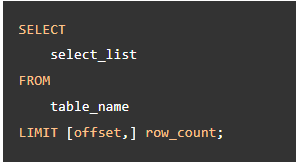
**sal+100**

**end ;**

## MySQL LIMIT clause

The LIMIT clause is used in the [SELECT](https://www.mysqltutorial.org/mysql-select-statement-query-data.aspx) statement to constrain the number of rows to return. The LIMIT clause accepts one or two arguments. The values of both arguments must be zero or positive [integers](https://www.mysqltutorial.org/mysql-int/).

The following illustrates the LIMIT clause syntax with two arguments:



**select \* from ap\_emp**

**order by sal desc**

**LIMIT 4;**

**select \* from ap\_emp**

**order by sal desc**

**LIMIT 3,2 ;**

**select \* from ap\_emp**

**order by sal**

**LIMIT 4;**

**select \* from ap\_emp**

**order by sal**

**LIMIT 2,1;**