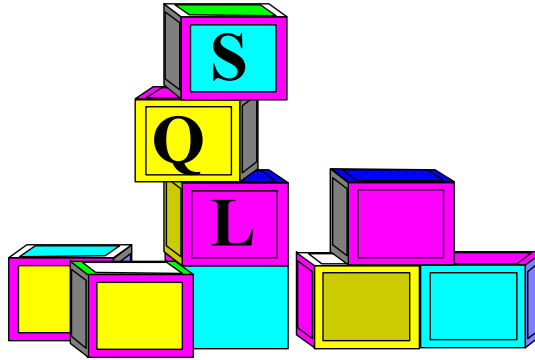


CHAPTER 1

CHAPTER 2

STRUCTURED QUERY LANGUAGE



TOPIC OBJECTIVES

This section introduces the **ANSI** standard SQL statements as used in Oracle.

When you finish this section, you will be able to:

- Understand the standard **ANSI SQL** available in Oracle.
- Comprehend the **SQL*PLUS** environment for executing SQL statements.
- Identify the basic format and structure of the SQL statements.
- Write SQL Select statements to retrieve multiple rows and/or columns.
- Write SQL commands utilizing the **WHERE** clause, **GROUP BY**, **HAVING** and **ORDER BY VERBS**.

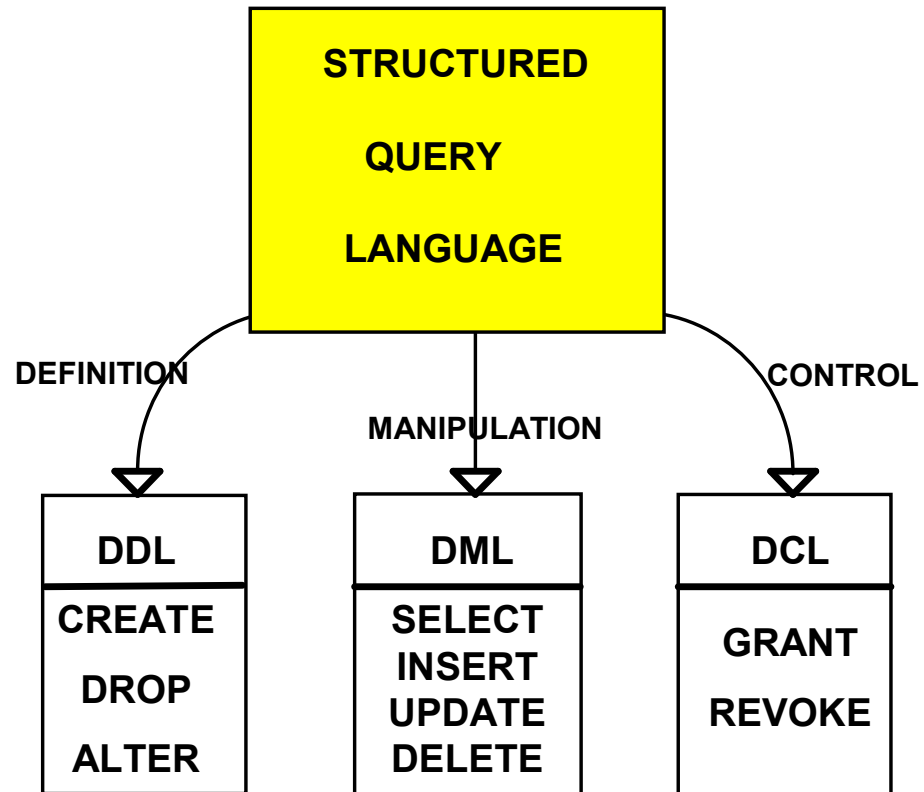
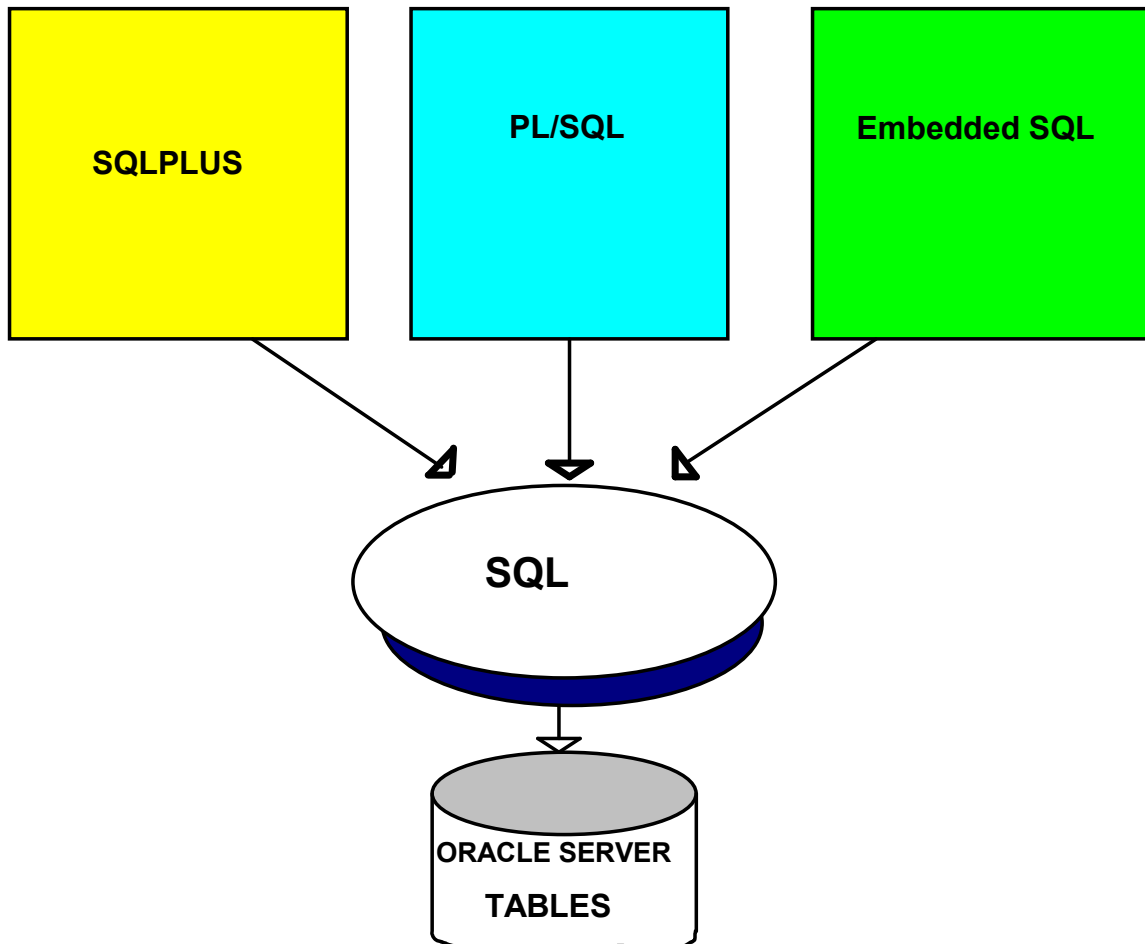
SQL COMPONENTS

TABLE ACCESS METHODS



SQL DATA MANIPULATION LANGUAGE (DML)

RETRIEVAL	BUILT-IN FUNCTIONS	ARITHMETIC OPERATORS
SELECT	MIN MAX SUM AVERAGE COUNT	+ - / *
GROUPING	OTHER OPERATORS	COMPARISON OPERATORS
GROUP BY HAVING	LIKE DISTINCT ANY ALL IN BETWEEN UNION AND OR NOT	= <> < >
MODIFICATIONS	SEQUENCING	
INSERT DELETE UPDATE	ORDER BY	

SAMPLE TABLES**ORG TABLE:**

DEPTNUMB	DEPTNAME	MANAGER	DIVISION	LOCATION
10	HEAD OFFICE	160	CORPORATE	NEW YORK
15	NEW ENGLAND	050	EASTERN	BOSTON
20	MID ATLANTIC	010	EASTERN	WASHINGTON
38	SOUTH ATLANTIC	030	EASTERN	ATLANTA
42	GREAT LAKES	100	MIDWEST	CHICAGO
51	PLAINS	140	MIDWEST	DALLAS
66	PACIFIC	270	WESTERN	SAN FRANCISCO
84	MOUNTAIN	290	WESTERN	DENVER

APPLICANT TABLE:

TEMPID	NAME	ADDRESS	EDLEVEL	COMMENTS
40	FROMMHERZ	SAN JOSE, CA	12	NO SALES EXP.
410	JACOBS	POUGHKEEPSIE, NY	16	GOOD CAND. / DC
420	MONTEZ	DALLAS, TX	13	OFFER SALES POS.
430	RICHOWSKI	TUCSON,AZ	14	CAN'T START WORK UNTIL 12/93
440	REID	ENDICOTT, NY	14	1 YEAR SALES EXP.
450	JEFFYREYS	PHILADELPHIA, PA	12	GOOD CLERICAL BACKGROUND
460	STANLEY	CHICAGO, IL.	11	WANTS P/T JOB
470	CASALS	PALO ALTO. CA	14	EXPERIENCED SALES
480	LEEDS	EAST FISHKILL, NY	12	NEEDS INTERVIEW WITH BROWN
490	GASPARD	PARIS, TX	16	WORKED HERE 1-80

STAFF TABLE:

ID	NAME	DEPT	JOB	YRS	SALARY	COMM
10	SANDERS	20	MGR	7	18357.50	---
20	PERNAL	20	SALES	8	18171.25	612.45
30	MARENGHI	38	MGR	5	17506.75	---
40	O'BRIEN	38	SALES	6	18006.00	846.55
50	HANES	15	MGR	10	20659.80	---
60	QUIGLEY	38	SALES	--	16808.30	650.25
70	ROTHMAN	15	SALES	7	16502.83	1152.00
80	JAMES	--	CLERK	--	13504.60	128.20
90	KOONITZ	42	SALES	6	18001.75	1386.70
100	PLOTZ	42	MGR	7	18352.80	---
110	NGAN	15	CLERK	5	12508.20	206.60
120	NAUGHTON	38	CLERK	--	12954.75	180.00
130	YAMAGUCHI	42	CLERK	6	10505.90	75.60
140	FRAYE	51	MGR	6	21150.00	---
150	WILLIAMS	51	SALES	6	19456.50	637.65
160	MOLINARE	10	MGR	7	22959.20	---
170	KERMISCH	15	CLERK	4	12258.50	110.10
180	ABRAHAMS	38	CLERK	3	12009.75	236.50
190	SNEIDER	20	CLERK	8	14242.75	126.50
200	SCOUTTEN	42	CLERK	--	11508.60	84.20
210	LU	10	MGR	10	20010.00	---
220	SMITH	51	SALES	7	17654.50	992.80
230	LUNDDQUIST	51	CLERK	3	13369.80	189.65
240	DANIELS	10	MGR	5	19260.25	---
250	WHEELER	51	CLERK	6	14460.00	513.30
260	JONES	10	MGR	12	21234.00	---
270	LEA	66	MGR	9	18555.50	---
280	WILSON	66	SALES	9	18674.50	811.50
290	QUILL	84	MGR	10	19818.00	---
300	DAVIS	84	SALES	5	15454.50	806.10
310	GRAHAM	66	SALES	13	21000.00	200.30
320	GONZALES	66	SALES	4	16858.20	844.00
330	BURKE	66	CLERK	1	10988.00	55.50
340	EDWARDS	84	SALES	7	17844.00	1285.00
350	GAFNEY	84	CLERK	5	13030.50	188.00

SQL BASIC RETRIEVAL

Task: Produce a list of all information in **TABLE ORG**.

SQL:

```
SELECT DEPTNUMB,DEPTNAME,MANAGER,DIVISION,
LOCATION from ORG
```

or

```
SELECT * FROM ORG
```

RESULT:

DEPTNUMB	DEPTNAME	MANAGER	DIVISION	LOCATION
10	HEAD OFFICE	160	CORPORATE	NEW YORK
15	NEW ENGLAND	50	EASTERN	BOSTON
20	MID ATLANTIC	10	EASTERN	WASHINGTON
38	S. ATLANTIC	30	EASTERN	ATLANTA
42	GREAT LAKES	100	MIDWEST	CHICAGO
51	PLAINS	140	MIDWEST	DALLAS
66	PACIFIC	260	WESTERN	SAN FRANCISCO
84	MOUNTAIN	290	WESTERN	DENVER

Model Select Statement:

```
SELECT (DISTINCT) ITEMS
FROM TABLE NAMES
{WHERE CONDITIONS}
{GROUP BY COLUMNS {HAVING CONDITIONS} }
{ORDER BY COLUMNS}
```

SQL BASIC RETRIEVAL

Task: Get all information pertaining to departments assigned to the eastern division.

SQL:

```
SELECT * FROM ORG
WHERE DIVISION='EASTERN'
```

RESULT:

DEPTNUMB	DEPTNAME	MANAGER	DIVISION	LOCATION
15	NEW ENGLAND	50	EASTERN	BOSTON
20	MID ATLANTIC	10	EASTERN	WASHINGTON
38	SOUTH ATLANTIC	30	EASTERN	ATLANTA

NOTE:

1. Values to be compared with columns of character data must be enclosed in single quotes (DIVISION='EASTERN').
2. Values to be compared with columns of numeric data must not be in quotes.

USING INEQUALITIES IN THE WHERE CLAUSE

= (equal to)

^= (not equal to)

> (greater than)

>= (greater than or equal to)

< (less than)

<= (less than or equal to)

- Search conditions may contain any of the above.
- Numbers compare algebraically, meaning negative numbers are less than positive numbers regardless of absolute value.
- If comparing two character strings of different lengths, the shorter is padded at the right with blanks to make it the same length.

SQL BASIC RETRIEVAL

Task: List all employees with more than 10 years with the company.

SQL:

```
SELECT * FROM STAFF
WHERE YEARS> 10
```

RESULT:

ID	NAME	DEPT	JOB	YRS	SALARY	COMM
260	JONES	10	MGR	12	21234.00	---
310	GRAHAM	10	SALES	13	21000.00	200.30

Exercises: (Use the ORG, STAFF OR APPLICANT TABLES)

1. Get all the employees who made more than \$1000 commission.
(See ex11.sql)
2. Get all the employees who make no more than \$15,000 salary
(Ignore commission). (See ex12.sql)
3. Get all the employees who are not clerks. (See ex13.sql)

SELECTING SPECIFIC COLUMNS

Task: Get the names and IDs of all employees who are managers.

SQL:

```
SELECT ID, NAME
FROM   STAFF
WHERE  JOB='MGR'
```

RESULT:

ID	NAME
-----	-----
10	SANDERS
30	MARENGHI
40	HANES
100	PLOTZ
140	FRAYE
160	MOLINAIRE
210	LU
240	DANIELS
260	JONES
270	LEA
290	QUILL

RETRIEVAL OF COMPUTED VALUES

Task: Get the monthly salary of all **CLERKS** in Department 38.

SQL:

```
SELECT NAME, SALARY/12
FROM STAFF
WHERE JOB = 'CLERK' AND DEPT=38
```

RESULT:

NAME	SALARY/12
-----	-----
NAUGHTON	1079.56
ABRAHAMS	1000.81

NOTE: Computed values, or expressions,
make use of the following Standard operations:

+	ADD
-	SUBTRACT
*	MULTIPLY
/	DIVIDE

SQL RETRIEVAL EXERCISES (USE STAFF, ORG OR APPLICANT)

1. Give names of all managers making less than \$20,000. (See ex21.sql)

Select name, job, salary

From staff

Where job = 'MGR' and salary < 20000.00

2. For all employees in Department 66, give their name, job function, and total annual earnings (Salary and Commissions). (See ex22.sql)

```
select dept,  
       name,  
       job,  
       salary,  
       comm,  
       salary + comm
```

from staff

Where dept=66

3. Give the name and department number of each employee not in Department 15. (See ex23.sql)

Select name, DEPT

From STAFF

WHERE dept <> 15

4. List all information about employee IDs up to and including 100. (See ex24.sql)

ORDER BY CLAUSE

```
SELECT DEPT, NAME, ID, SALARY
FROM STAFF
ORDER BY DEPT ASC, SALARY DESC
```

- The **ORDER BY** clause describes the order of the query result.
- All columns in the **ORDER BY** clause must also be in the **SELECT** clause.
- Ordering may be ascending or descending on any one field.
- **Cannot** use expressions in the **ORDER BY** clause as an ANSI standard.

NOTE: Oracle will allow you to use expressions.

ORDERING BY DESCENDING

Task: Give the employee ID, name, and department for each manager.
Order the result in descending years of service.

SQL:

```
SELECT ID, NAME, DEPT, JOB, YEARS,
FROM STAFF
WHERE JOB='MGR'
ORDER BY YEARS DESC
```

RESULT:

ID	NAME	DEPT	YEARS
-----	-----	-----	-----
260	JONES	10	12
310	HANES	15	10
210	LU	10	10
290	QUILL	84	10
270	LEA	66	10
10	SANDERS	22	7
100	PLOTZ	42	7
160	MOLINAIRE	10	7
140	FRAYE	51	6
30	MARENGHI	38	5
240	DANIELS	10	5

ORDERING BY COLUMN NUMBERS

Task: Get the name and monthly salary of each employee in Department 38.
Order the output by descending monthly salary.

SQL:

```
SELECT NAME, SALARY/12, DEPT
FROM   STAFF
WHERE  DEPT=38
      ORDER BY SALARY/12 DESC
```

RESULT:

NAME	SALARY/12
O'BRIEN	1500.50
MARENGHI	1458.90
QUIGLEY	1400.69
NAUGHTON	1079.56
ABRAHAMS	1000.81

NOTE: Necessary for expressions, since expressions
cannot appear in the **ORDER BY** clause.

THE DISTINCT OPERAND

Case A: Without the distinct operand

SQL:

```
SELECT EDLEVEL FROM APPLICANT
```

RESULT:

EDLEVEL

12
16
13
14
14
12
11
14
12
16

Case B: With the distinct operand

SQL:

```
SELECT DISTINCT EDLEVEL  
FROM APPLICANT
```

RESULT:

EDLEVEL
11
12
13
14
16

SELECTING ON CONDITIONS

Other conditional clauses:

Multiple conditions	And, Or, and ()
Values in a range	Between X and Y
Values from a list	<u> </u> In (X, Y, Z)
Partial search values	<u> </u> Like '%ABC%' Like '_A_%'
Negative conditions	Not
Checking for nulls	Is Null

MULTIPLE CONDITIONS

Task: Get the ID, Name, and Department for each manager making more than \$21,000.

SQL:

```
SELECT ID, NAME, DEPT
FROM STAFF
WHERE JOB = 'MGR' AND SALARY > 21000
```

RESULT:

ID	NAME	DEPT
140	FRAYE	51
160	MOLINAIRE	10
260	JONES	10

MULTIPLE CONDITION EXERCISES

1. Get the name of each employee who is either a manager or who has over 10 years seniority. (See ex31.sql)
2. Display the ID, name, and total earnings of each clerk who earned between \$12,000 and \$15,000 salary in addition to at least \$150 commission. (See ex32.sql)

```
SELECT id, Name, salary + comm, salary, comm, job
FROM staff
WHERE job = 'CLERK' AND
      SALARY BETWEEN 12000.00 AND 15000.00 AND
      COMM >=150
```

3. Display all department numbers having at least one employee with more than 5 years experience and making less than \$15,000. (See ex33.sql)

THE BETWEEN CLAUSE

Task: Get the name of each employee with 3 to 5 years of service.

SQL:

```
SELECT NAME, YEARS, SALARY + COMM  
FROM STAFF  
WHERE YEARS BETWEEN 3 AND 5  
ORDER BY 1, 3
```

RESULT:

```
NAME  
-----  
MARENGHI  
NGAN  
KERMISCH  
ABRAHAMS  
LUNDQUIST  
DANIELS  
DAVIS  
GONZALEZ  
GAFNEY
```

NOTE: Limits are inclusive.

THE IN CLAUSE

Task: Get the name and address of each applicant whose education level is 11, 12, or 16.

SQL:

```
SELECT NAME, ADDRESS, EDLEVEL
FROM APPLICANT
WHERE EDLEVEL IN (11, 12, 16);
```

RESULT:

NAME	ADDRESS
-----	-----
FROMMHERZ	SAN JOSE, CA
JACOBS	POUGHKEEPSIE, NY.
JEFFYREYS	PHILADELPHIA, PA.
STANLEY	CHICAGO, IL.
LEEDS	EAST FISHKILL, NY.
GASPARD	PARIS, TX.

THE LIKE CLAUSE

Task: Display the department number and department name for each department that has the string **ATLANTIC** embedded somewhere in the department name.

SQL:

```
SELECT DEPTNUMB, DEPTNAME
FROM ORG
WHERE DEPTNAME LIKE '%ATLANTIC%'
```

RESULT:

DEPTNUMB	DEPTNAME
-----	-----
20	MID ATLANTIC
38	SOUTH ATLANTIC

NOTE:

1. The percent sign (%) stands for any number of characters, or none.
2. The % may appear before the string, after the string, or both.

THE LIKE CLAUSE

Task: List the name of each employee whose name contains the characters **ON** in the second and third position.

SQL:

```
SELECT NAME FROM STAFF
WHERE NAME LIKE '_ON'
```

RESULT:

```
NAME
-----
JONES
GONZALEZ
```

NOTE: 1. Each underscore (not hyphen) represents exactly one character.
2. The like clause is only valid with character data.

THE LIKE CLAUSE EXERCISES

1. Display the department name and manager id (from org table) of each department whose location ends with the characters **TON**. (See ex41.sql)

```
Select Deptname, manager, dept  
From staff  
Where location like '%TON'
```

2. Display all department names that contain the characters **AIN** anywhere after the first character. (See ex42.sql)

```
Deptname like '_%AIN%'
```

3. Display all employee names that begin with **S** and contain at least one **D**. (See ex43.sql)

NEGATIVE CONDITIONS

The opposite of any condition can be indicated by placing **NOT** in front of it.

SQL:

```

SELECT DEPTNUMB, DEPTNAME
FROM ORG
WHERE DEPTNUMB NOT BETWEEN 15 AND 66

```

RESULT:

DEPTNUMB	DEPTNAME
-----	-----
10	HEAD OFFICE
84	MOUNTAIN

SQL:

```

SELECT ID, NAME
FROM STAFF
WHERE ID BETWEEN 200 AND 230
AND NAME NOT LIKE 'S%'

```

RESULT:

ID	NAME
---	-----
210	LU
230	LUNDQUIST

NEGATIVE CONDITIONS

Case A:

```

SELECT DEPTNUMB, DEPTNAME
FROM ORG
WHERE NOT DEPTNUMB > 15
AND DIVISION = 'EASTERN'

```

RESULT:

DEPTNUMB	DEPTNAME
15	NEW ENGLAND

Case B:

```

SELECT DEPTNUMB, DEPTNAME
FROM ORG
WHERE NOT (DEPTNUMB > 15
AND DIVISION = 'EASTERN')

```

RESULT:

DEPTNUMB	DEPTNAME
10	HEAD OFFICE
15	NEW ENGLAND
42	GREAT LAKES
51	PLAINS
66	PACIFIC
84	MOUNTAIN

BUILT IN FUNCTIONS

BUILT IN FUNCTION	FINDS FOR A SET OF VALUES
SUM	TOTAL
MIN	THE MINIMUM VALUE OF THE SET
AVG	AVERAGE
MAX	THE MAXIMUM VALUE FOR THE SET
COUNT (DISTINCT...)	THE NUMBER OF DISTINCT VALUES
COUNT (*)	THE NUMBER OF VALUES

If one built-in function is specified in the ***SELECT*** clause, all other items in the select list must also be built-in functions (unless the ***GROUP BY*** clause is used).

NOTE: Null values are not included in any calculation.

FINAL TOTALS USING BUILT-IN FUNCTIONS

Task: Retrieve the maximum salary, maximum commission, total salary average commission, and total number of distinct salaries for the staff table.

SQL:

```
SELECT max(salary) "Max Salary", max(comm) "Max Comm",
sum(salary) "Sum Salary", avg(comm) "Average Comm",
count(distinct salary) "Unique Sal" from staff
```

RESULT:

<u>Max Salary</u>	<u>Max Comm</u>	<u>Sum Salary</u>	<u>Average Comm</u>	<u>Unique Sal</u>
22,959.20	1,386.70	589,192.98	488.07	34

When using built-in functions and the ***SELECT*** clause with only built-in functions declared (see above), then Oracle will return one and only one row.

NOTE: Summary totals bring back only one row.

BUILT-IN FUNCTIONS EXERCISES

1. What is the total number of departments? (See ex51.sql)

RESULTS:

COUNT(*)

8

(1 ROW SELECTED)

2. How many different divisions are there? (See ex52.sql)

RESULTS:

COUNT(DISTINCT DIVISION)

4

(1 ROW SELECTED)

3. List the average salary for managers. (See ex53.sql)

RESULTS:

AVG(SALARY)

19805.8

(1 ROWS SELECTED)

4. What is the maximum salary for employees with less than 5 years of service? (See ex54.sql)

RESULTS:

MAX(SALARY)

16858.2

(1 ROWS SELECTED)

5. What is the total years of service for all employees in departments 15, 20, and 42? (See ex55.sql)

RESULTS:

SUM(YEARS)

68

(1 ROW SELECTED)

GROUP BY CLAUSE



Group salaries by department

- The **GROUP BY** clause is used in conjunction with built-in functions.
- It conceptually rearranges the table into groups, one group per value of the group by field.
- The result consists of one row per value of the **GROUP BY** field.
- Each item in the **SELECT** clause must be single valued per group.

GROUP BY CLAUSE

Task: Display the minimum, average, and maximum salary for each job function.

SQL:

```
SELECT DEPT, JOB, MIN (SALARY) , AVG (SALARY) ,
MAX (SALARY) FROM STAFF
GROUP BY dept, JOB
```

RESULT:

<u>JOB</u>	<u>MIN(SALARY)</u>	<u>AVG(SALARY)</u>	<u>MAX(SALARY)</u>
CLERK	10505.90	12612.61	14460.00
MGR	17506.75	19805.50	22959.20
SALES	15454.50	17869.36	21000.00

NOTE: If used, **GROUP BY** clause must be after the **FROM** and **WHERE** clauses, and before the **ORDER BY** clause

THE HAVING CLAUSE



How do I filter rows from a previous results table?

- Always used with the **GROUP BY** clause.
- Just as the **WHERE** clause eliminates rows from single select query, the **HAVING** clause eliminates whole groups specified in the **GROUP BY** clause.

TASK: For any departments with more than 4 employees, show the total number of employees in those departments.

Example:

```
SELECT DEPT, COUNT (*)
FROM STAFF
WHERE salary * 1.10 > 12000
GROUP BY DEPT
HAVING COUNT(*) > 4
ORDER BY DEPT
```

RESULT:

<u>DEPT</u>	<u>CO UNT(*)</u>
38	5
51	5
66	5

All departments with no more than 4 employees are eliminated from the result.

THE HAVING CLAUSE EXERCISES

1. For each department whose average salary is greater than \$18,000, list the department number, average years of service, and average salary.
(See ex61.sql)

RESULTS:

<u>DEPT</u>	<u>AVG(YEARS)</u>	<u>AVG(SALARY)</u>
10	8.5	20865.9

(1 ROW SELECTED)

2. List each department number where all employees have at least 5 years of service AND Departments must be in 15, 20, 48, 66.
(See ex62.sql)

RESULTS:

<u>DEPT</u>	<u>MIN(YEARS)</u>
10	5
20	7
42	6
84	5

(4 ROWS SELECTED)

```

SELECT DEPT, MIN(YEARS)
FROM STAFF
WHERE DEPT IN (15,20,48,66)
GROUP BY DEPT
HAVING MIN(YEARS) > 5
  
```

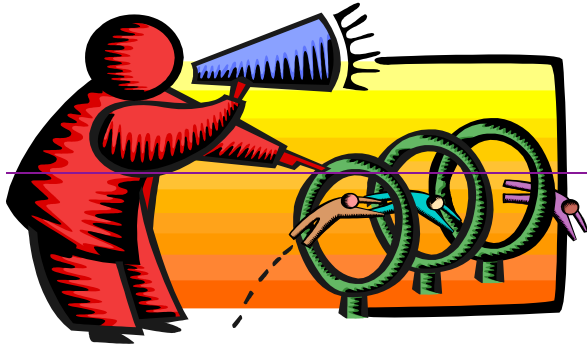
3. List each department number having a total commission of at least \$1200. (See ex63.sql)

RESULTS:

<u>DEPT</u>	<u>SUM(COMM)</u>
15	1468.7
38	1913.3
42	1546.5
51	2333.4
66	1911.3
84	1673.3

(6 ROWS SELECTED)

—
—
—
ROLLUP



ROLLUP ALLOWS YOU TO CALCULATE GRAND TOTALS

ROLLUP DOES GROUP BY ON ALL FIELDS LISTED, PLUS GROUP BY SUBSETS OF THE FIELDS (FROM LEFT), PLUS GRAND TOTALS OF ALL FIELDS.

```
SELECT DEPT,
       SUM (SALARY ) AS SALARY,
       COUNT (*) AS NUMROWS
FROM STAFF
GROUP BY ROLLUP (DEPT)
ORDER BY DEPT
```

RESULT

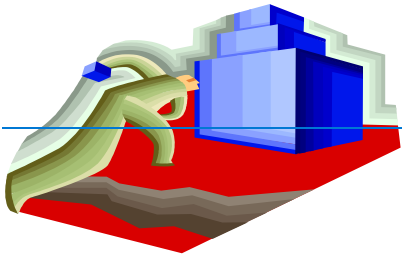
DEPT SALARY NUMROWS

DEPT	SALARY	NUMROWS
A00	128500	3
B01	41250	1
C01	90470	3
D11	222100	9
-	482320	16

THE FOLLOWING ARE EQUIVALENT:

```
ROLLUP(A, B, C) GROUP BY A, B, C
UNION ALL
GROUP BY A,B
UNION ALL
GROUP BY A
UNION ALL
GRAND-TOTAL
```



CUBE**BUILDING SUMMARY TOTALS ON ALL SETS**

CUBE DOES GROUP BY ON ALL FIELDS LISTED, PLUS GROUP BY ON <ALL> SUBSETS OF THE FIELDS PLUS GRAND TOTALS OF ALL FIELDS.

EXAMPLE 1

```
SELECT DEPTNO AS DEPARTMENT,
       JOB      AS CATEGORY,
       SUM(SAL) AS TOT_SAL,
       COUNT(*) AS NUMROWS
FROM EMP
GROUP BY CUBE (DEPTNO, JOB)
RESULT
DEPARTMENT CATEGORY TOT_SAL NUMROWS
```

10	CLERK	1300	1
10	MANAGER	2450	1
10	PRESIDENT	5000	1
10		8750	3
20	ANALYST	6000	2
20	CLERK	1900	2
20	MANAGER	2975	1
20		10875	5
30	CLERK	950	1
30	MANAGER	2850	1
30	SALESMAN	5600	4
30		9400	6
	ANALYST	6000	2
	CLERK	4150	4
	MANAGER	8275	3
	PRESIDENT	5000	1
	SALESMAN	5600	4
		29025	14

18 ROWS SELECTED... ..

GRAND TOTALS ALSO PRODUCED FOR EACH COLUMN ABOVE WHEN APPROPRIATE

RETRIEVAL INVOLVING NULL

$$1 + \text{NULL} = \text{NULL}$$

- **RDBMS's** supports the concept of a field having null status
- To test a field for null status, a special conditional form is used:

`COLUMN-NAME IS {NOT} NULL`

Example:

INCORRECT:

```
SELECT ID, NAME
FROM STAFF
WHERE COMM=NULL
```

CORRECT:

```
SELECT ID, NAME, 'COMM IS NULL'
FROM STAFF
WHERE COMM IS NULL
```

TREATMENT OF NULLS

A null value means that there is no value assigned. In other words, the value is unknown.

TEAM	WINS
LIONS	6
BLUE SOX	2
ALLEY CATS	NULL
CARDINALS	2
PANTHERS	0
ORIOLES	NULL
LUMBERJACKS	4

1. What teams have won more games than the Orioles?
2. Are the Orioles tied with the Alley Cats?
3. Are the Panthers tied with the Orioles?
4. What are the total wins for the league?
5. What are the average wins per team?

TREATMENT OF NULLS

Whenever a null is used to derive a result, the result is also null (unknown).

SALARY	COMM	SALARY + COMM
18,000	NULL	NULL
NULL	500	NULL
18,000	500	18,500
NULL	NULL	NULL

NOTE: Nulls are excluded before Oracle performs any calculations.

Null Values are excluded for:

Comparison:

```
WHERE WINS < 4  
WHERE WINS >=4
```

Computations:

```
SALARY + COMM * 1.10
```


FUNCTIONS INVOLVING NULL VALUES

TEAM	WINS
LIONS	6
BLUE SOX	2
ALLEY CATS	
CARDINALS	2
PANTHERS	0
ORIOLES	
LUMBERJACKS	4

COUNT (*) = 7

SUM (WINS) = 14

MAX (WINS) = 6

AVG(WINS) = 2.8

COUNT (DISTINCT WINS) = 4

BUILT-IN FUNCTIONS:

- MIN
- MAX
- AVG
- SUM
- COUNT (DISTINCT)

TREATMENT OF NULLS

Nulls are always listed last when **ORDER BY** clause in ascending sequence is used.

TEAM	WINS
LIONS	6
BLUE SOX	2
ALLEY CATS	
CARDINALS	2
PANTHERS	0
ORIOLES	
LUMBERJACKS	4

(BEFORE)

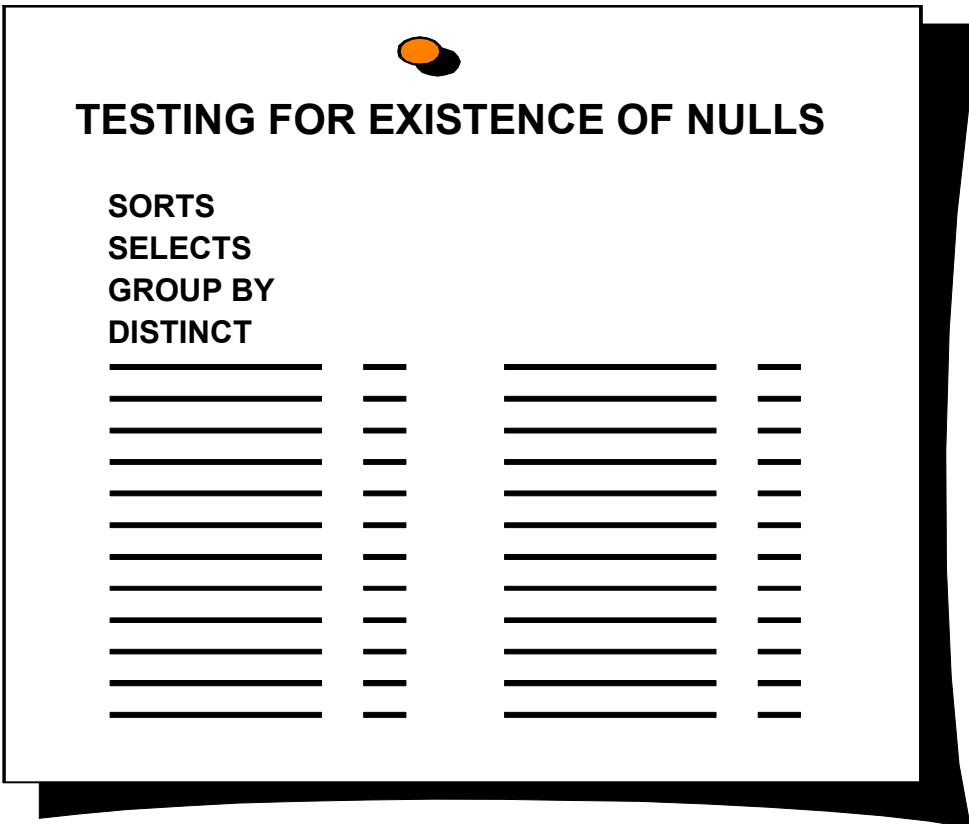
Select * from teams
order by wins;

TEAM	WINS
LIONS	0
LUMBERJACKS	2
BLUE SOX	2
CARDINALS	4
PANTHERS	6
ORIOLES	
ALLEY CATS	

(AFTER)

Select * from teams
order by wins;

NULL VALUES ARE CONSIDERED IN:



TESTING FOR EXISTENCE OF NULLS

SORTS

SELECTS

GROUP BY

DISTINCT

- Tests for existence of nulls where **wins** is (not) null.
- Count (*) where all rows are considered.
- Select distinct clause produces one null value in result.
- Unique indexes which allow one row to have a null key except primary key index.
- **GROUP BY** - Creates a separate group for each null.
- **ORDER BY** - Places all nulls at the **BOTTOM** of the list (ascending order).

TREATMENT OF NULLS

Advantages:



- Allows identification of values not yet specified.
- Allows addition of a new column to an existing table.
- Allows insertion of a row when not all values are known.

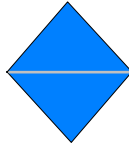
Disadvantages:



- Requires additional processing to handle null decisions.
- Allows insertion of a row containing incomplete data.

CASE EXPRESSION

DECISION CONSTRUCT



-
- Lets one have “if then else” logic in a SQL statement
 - *FIRST CASE THAT MATCHES IS THE ONE USED*
 - If non matches, result is null (default)

Use CASE to expand a value

```
SELECT DEPTNO,ENAME,  
       CASE  
         WHEN DEPTNO = 10 THEN 'CORPORATE'  
         WHEN DEPTNO = 20 THEN 'RESEARCH'  
         WHEN DEPTNO = 30 THEN 'ACCOUNTING'  
         WHEN DEPTNO = 40 THEN 'PUBLISHING'  
         ELSE NULL  
       END AS DEPTNAME  
FROM EMP
```

Note: Any valid comparison operator (>, <, <=, >=, NOT) can be used i.e. (when ename > deptname)

CASE EXPRESSION WITH BUILT-IN FUNCTIONS AND PREDICATES



What else can I do with CASE?

```
SELECT COUNT(*) as TOTAL,
       SUM (CASE WHEN DEPTNO = 10 THEN 1 ELSE 0 END) COUNT_10,
       SUM (CASE WHEN DEPTNO = 20 THEN 1 ELSE 0 END) COUNT_20,
       SUM (CASE WHEN DEPTNO = 30 THEN 1 ELSE 0 END) COUNT_30,
       SUM (CASE WHEN DEPTNO = 40 THEN 1 ELSE 0 END) COUNT_40
FROM EMP
/
```

PREDICATE USAGE:

```
SELECT  ENAME,
        HIREDATE into :emp_name, a_date
FROM EMP
WHERE   ENAME LIKE 'R%'
        AND CASE
            WHEN deptno = 10 THEN :sal_increase * 1.10
            WHEN deptno = 20 THEN :sal_increase * 1.08
            ELSE NULL
        END IS NOT NULL
```

LAB 3 - MORE ABOUT SELECTS



Using SQL*PLUS, create and execute the following queries using the **staff**, **org**, and **applicant** tables. (See Lab3.lst for answers.)

1. Give all information about each clerk.
2. List the name, salary and commission of each employee by department.
3. List the names and job functions of all employees working in departments 20, 42, 51, and 84.
4. Give the average salary for each department.
5. List all sales personnel by descending total earnings (Salary Commission).
6. List the names of all employees whose name contains the characters ON.
7. Display all employees whose weekly salary is greater than \$300.00.
8. Display all employees whose commission is at least 5% of total earnings.
9. List id, name, salary, and commission for all employees except Departments 38 and 41. (Order by Department)
10. List every department with at least one employee with a null commission. (List a department number only one time.)

Page Left intentionally blank