**Chapter 3**

**Working with SELECT statement**

**Objectives**

* Use SELECT statement to display columns;
* Apply WHERE clause to filter rows;
* Filter with string pattern;
* Use BETWEEN clause;

We have discussed in the previous chapter data definition language (DDL) commands. Commands that fall in this category are CREATE, DROP, ALTER, DECLARE. Use these command to define, modify, and delete database objects.

Data manipulation language (DDL) commands we have used so far are SELECT, INSERT, UPDATE, and DELETE. These commands are designed to retrieve, insert, modify, and delete rows and columns within a table.

This chapter will explore more of SELECT statement that retrieves data from a table. You may filter result sets with WHERE clause.

During the installation of DB2, we have inserted sample database inside the server. Let us use SAMPLE database and view its tables. Below is the SAMPLE ERD:

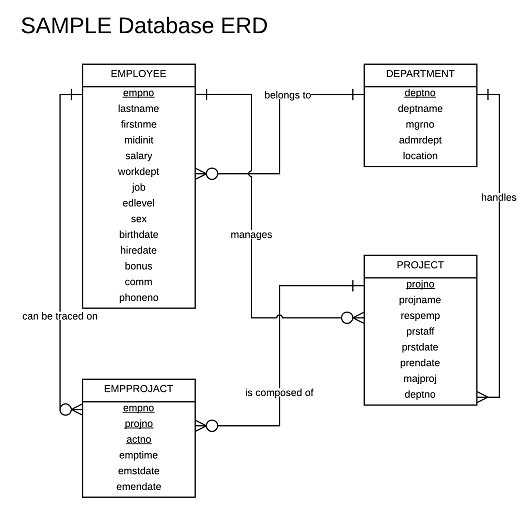


Figure 3-1. SAMPLE Database

This is not the total composition of SAMPLE database. Only EMPLOYEE, DEPARTMENT, EMPPROJACT, and PROJECT tables will be the focused of the discussion of examples.

Launch your DB2 and connect to SAMPLE with this script.

CONNECT TO sample;

List all table for this database:

LIST TABLES

**Display specific columns**

SELECT statement retrieves rows of data from a given table. So far, we have used \* to display all columns from a table. We may, however, include only certain columns that requires certain listing. On this case, we have the syntax:

SELECT <column name 1>, <column name 2>, … < column name n> FROM <table name>

Each column name must be separated by a comma. The arrangement of columns on a table is insignificant which means you may display column name 2 first before column name 1.

Let us see first the structure of EMPLOYEE display all of its columns .

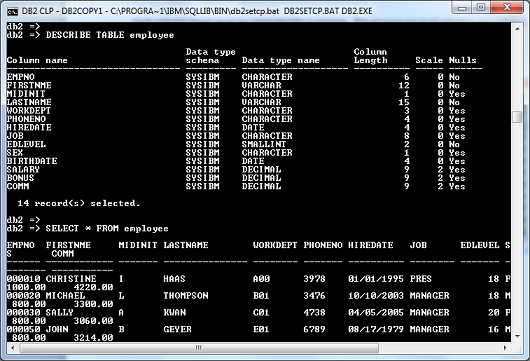


Figure 3-3. Displaying all columns of EMPLOYEE table

All columns can be displayed with the screen size we have that is why it wraps up to the next line. But if we want to display only firstNme, midInit, and lastName, we issue the given statement:

SELECT firstNme, midInit, lastName FROM employee

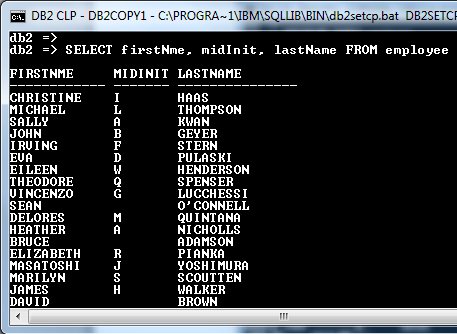


Figure 3-4. List only specific columns in any order.

**Filter rows with WHERE clause**

Tables may contain millions of rows. If we want to isolate particular data sets, we use the WHERE predicate:

SELECT <column name 1>… <column name n> FROM <table name>

WHERE <predicate>

Predicate clause here may include comparison(relational) operators, logical operators, IN, BETWEEN, among others in one or more filtering conditions. Comparison or relational operators are:

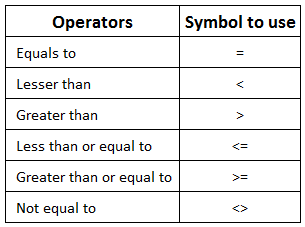


Table 3-1. Comparison operators

Let us have few examples to use the WHERE clause.

You are required to determine the employees who have finished 18 years of schooling from the EMPLOYEE table. List last name, first name, job position, and education level.

To fulfill this requirement we issue the following script:

SELECT lastName, firstNme, job, edLevel FROM employee WHERE edLevel = 18

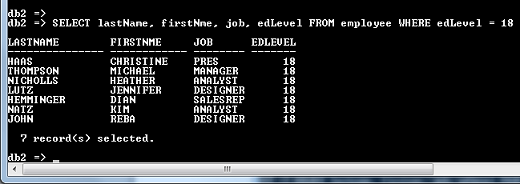


Figure 3-5. Using WHERE clause with comparison operator

Here, we specify column edLevel as the filtering condition. From the 42 rows of EMPLOYEE, the script generated 7 rows that has 18 years of schooling using the WHERE clause.

How about if the requirement wants to filter all CLERK, FIELDREP, and OPERATOR? The best approach here is to use an IN operator within the WHERE predicate:

SELECT lastName, firstNme, job, edLevel FROM employee

WHERE job IN (‘CLERK’, ‘FIELDREP’,’ OPERATOR’)



Figure 3-6. Using IN operator

The text items in the list of IN operator must be enclosed with single quotes separated by comma.

Make sure to enclose the entire list with parentheses.

**Use BETWEEN-AND in filtering range of values**

How about if you need to determine records for set of values in a range? Consider the problem:

List the last name, first name, job, and salary of employees receiving salary ranging from 30,000 to 40,000 dollars annually.

Insert BETWEEN and AND in the script.

SELECT lastName, firstNme, job, salary FROM employee

WHERE salary BETWEEN 30000 AND 40000

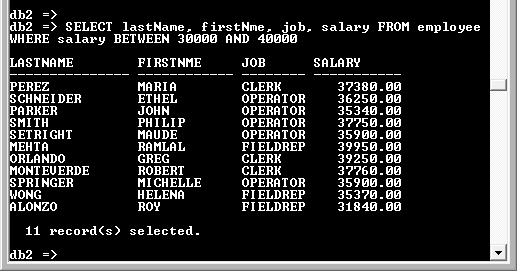


Figure 3-7. BETWEEN and AND clause

The script used salary is to test the range of values that falls BETWEEN 30,000 to 40,000. BETWEEN-AND clause is perfect when filtering values that set range.

**Filter compound conditions**

You may filter rows using WHERE with multiple conditions. Logical operators such as AND, OR or NOT can be used to specific requirement.

Display all employees who are receiving salary 60000 or more. Include last name, first name, job, and salary. Exclude president and manager of the company.

SELECT lastName, firstNme, job, salary FROM employee

WHERE salary >= 60000 AND job NOT IN (‘PRES’, ‘MANAGER’)

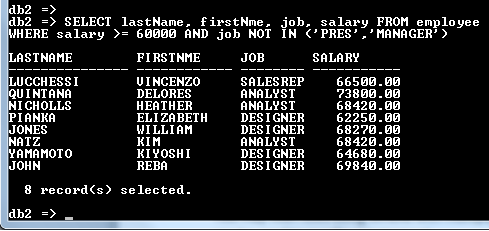


Figure 3-8. WHERE with compound condition

Here we used two conditions: salary >= 6000 and job NOT IN (‘PRES’, ‘MANAGER’). Both must be set to TRUE to filter rows that met the conditions. Comparison operator greater than or equal limits salary that has 60000 or more on the first condition. NOT operator creates a reverse effect excluding president and manager positions in the second condition.

**Limit rows with LIKE for string pattern**

Assume that you need to display last name, first name, and workdept with last name that starts with letter ’ S’. Issue the following script:

SELECT lastName, firstNme, workDept FROM employee

WHERE lastName LIKE ‘S%’



Figure 3-9. LIKE filters last name that starts with ‘S’

LIKE filters string pattern. The percent symbol (%) represents a wildcard --- any letter, symbol or number are included in the pattern regardless of its size.

If the requirement asks to filter surname with ‘E’ as the second character of the surname, we write:

SELECT lastName, firstNme, workDept FROM employee

WHERE lastName LIKE ‘\_E%’



Figure 3-10.

The underscore represents a character of the data whatever the value is. The script tells us to display all last names that start with any character that has letter E preceding it. You still need to indicate the % symbol to represent any character after E.

**Limit rows with dates**

Let us determine employees born between 1961 and 1971 showing last name, first name, and birth date. We issue the following script:

SELECT lastName, firstNme, birthDate FROM employee

WHERE birthDate >= ‘1961-01-01’ AND birthdate <= ‘1971-12-31’



Figure 3-11. Filtering with date with compound statement

The first condition we have set is to test if birth date is greater than or equal to the first day of 1961; the second condition is to test the last day of 1971. To test dates we use the format yyyy-mm-dd enclosed with single quotes.

We could achieve the same thing with this script:

SELECT lastName, firstNme, birthDate FROM employee

WHERE birthDate BETWEEN ‘1961-01-01’ AND ‘1971-12-31’



Figure 3-12. Using BETWEEN-AND to filter dates

The above script removed the greater than and less than operator and simply replaced with BETWEEN-AND clause.

You may also the YEAR function to isolate the date’s year:

SELECT lastName, firstNme, birthDate FROM employee

WHERE YEAR (birthDate) BETWEEN 1961 AND 1971



Figure 3-13. Using the YEAR () function

As you could see with our previous examples, we could make multiple approaches in addressing a requirement.

**Create column alias name**

We may use another column title for specific column if we do not like to use the column name defined in the structure of the table. Include AS to identify the alternative name for the column enclosed with double quotes. Drop AS if you want and write immediately the alias name still enclosed with double quotation marks.

SELECT lastName AS “Surname “, firstNme “First Name” FROM employee

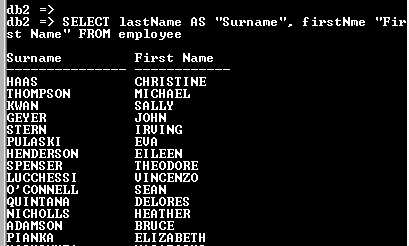


Figure 3- 13. Setting alias name with or without AS

**Derive new column as a result of computation**

Other than specified columns on a table, you may create new column as a result of computation. If we want the current age of employees, we get the birthdate of each employee and deduct it from the current year. The current year on our example is 2014. By simply getting the birth year of the employee using the YEAR () function and deducting it from 2014, we derive age.

SELECT lastName, firstNme, birthDate, 2014-YEAR (birthDate) FROM employee

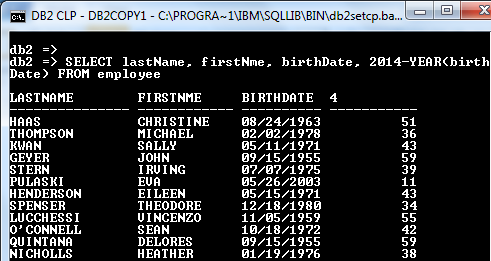


Figure 3-14. Deriving new computed column

The 4 represents the fourth column on the result of the script. We may change this with a new column name by providing an alias name enclosed with double quotation marks.

SELECT lastName, firstNme, birthDate, 2014-YEAR (birthDate) “Age” FROM employee

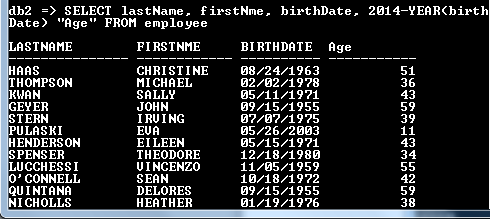


Figure 3-15. Assigning an alias name to a derived column

You may filter those employees who have an age below 20 and see already which records have faulty encoding of birthdates.

SELECT lastName, firstNme, birthDate, 2014-YEAR (birthDate) “Age”

FROM employee WHERE 2014-YEAR (birthDate) < 20

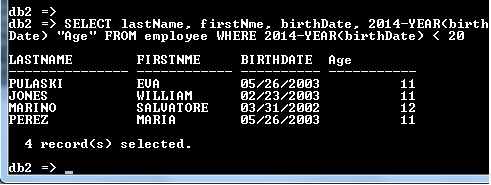


Figure 3-16. Using WHERE on derived columns

Assume that the company requires you to list last name, first name, job, salary, 10% increase on the salary, and the new salary with the added 10%. Your script will look like this:

SELECT lastName, firstNme, job, salary, 0.1\*salary “10% Increase”,

salary + 0.1\*salary “New Salary” FROM employee

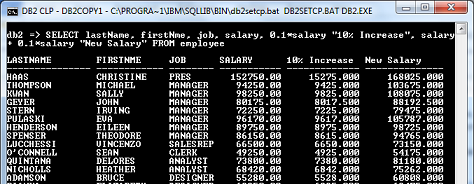


Figure 3-17. Result of computation in derived columns

We multiply salary by 0.1 to get the 10% value of the increase. For the new salary, we add salary with the 10% of the salary to get the new salary. Here we supply “10% Increase” and “New Salary” as aliases of the columns specified.

**CONCAT () to combine columns**

You may combine two or more columns and put it on one derived column. Use CONCAT () function to achieve this:

CONCAT (<column name 1>, <column name 2>)

Notice that in most of our examples last name and first name are in separate column. We may designate column name “Name” as a result of concatenating the two columns.

SELECT CONCAT (CONCAT (lastName, ‘, ‘), firstNme) “Name”, job FROM employee



Figure 3-18. CONCAT () function to combine columns

Since CONCAT is limited in combining two columns only, we nest if with another CONCAT to combine first name with last name added by a comma. Comma here is considered a column that is why it is enclosed by two single quotes and not double quotes. At the end of the CONCAT function we have the alias name “Name” for the operation.

**Use 2 pipe symbols to combine columns**

As alternative to CONCAT () function, you may use double pipe symbols (||) to concatenate columns. If we are to combine lastname and first name with a comma separating it, we have the following script:

SELECT lastName || ‘, ‘ || firstNme “Name”, job FROM employee

**Check NULL value**

When a column contains nothing (even a single space), it is considered NULL. Some columns are restricted to contain a value and some are allowed to be empty or NULL. When you check the content of DEPARTMENT table, you would see dashes on MGRNO (manager’s employee number) and LOCATION columns.

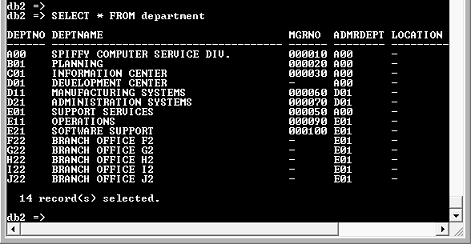


Figure 3-19. Dash represents NULL value

To filter null value:

SELECT deptNo, deptName, mgrNo FROM department

WHERE mgrNo IS NULL

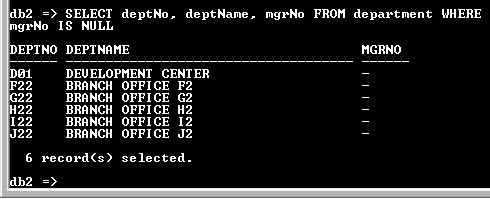


Figure 3-20. Filtering NULL values

**Sort rows using ORDER BY clause**

The way records are in stored on a table is entered in simultaneous manner. Nevertheless, we could present the list arranged according to our specification.

SELECT <column name 1>, <column name 2> … FROM <table name>

ORDER BY <column name 1>, <column name 2> < [ASC | DESC]…

In order to sort rows by last name in alphabetical order at EMPLOYEE table, issue the following script:

SELECT lastName, firstNme, workDept, hireDate FROM employee

ORDER BY lastName

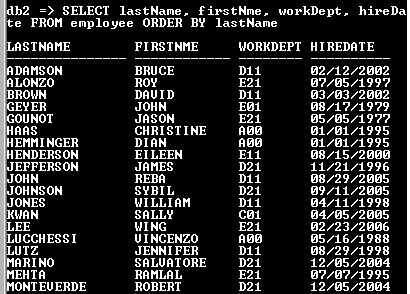


Figure 3-21. List rows by last name in ascending order

Specify the column/s of what will be the basis of the arrangement after ORDER BY. Last name here is used to sort row in ascending order.

How about if we want to display in descending order with the same set of columns from the previous example? Issue the following script:

SELECT lastName, firstNme, workDept, hireDate FROM employee

ORDER BY lastName DESC



Figure 3-22. Using DESC to display in descending order

We did not specify ASC (for ascending order) anymore in our first example because that is already the default of ORDER BY clause. If you want to display in reverse order, use DESC instead after the column/s subject for order arrangement.

Sorting is not limited with one column only. You can sort multiple columns depending on which column to sort first before proceeding to the next column.

The next requirement is to display employees by working department, last name, and first name from the EMPLOYEE’s table. The sorting script on this case is:

SELECT workDept, lastName, firstNme FROM employee

ORDER BY workDept, lastName

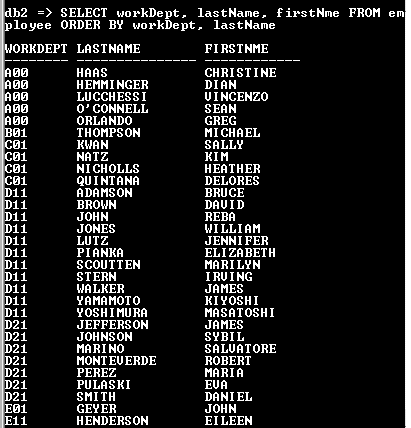


Figure 3-23. Sorting by multiple columns

As you notice from the previous script, the list is sorted according to work department and then by last name in ascending order. The sorting of column in the order of priority is separated by a comma.

Instead of column names, you may replace it with column number. The previous example has 3 columns on its script and sorted according to workDept and last name. workDept is defined as column 1 and last name as column 2. If we sort according by column number we have:

SELECT workDept, lastName, firstNme FROM employee

ORDER BY 1, 2

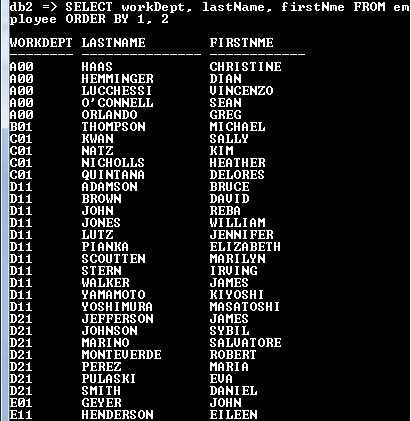


Figure 3-x. Sorting using column number

You may also sort according to a computed value. Let us determine the length of service in years of employees. List employee’s last name, first name, hire date, and computed value length of service. Sort according who stayed most to the least that rendered service.

SELECT lastName, firstNme, hireDate, 2014-YEAR (hireDate) “Length of Service”

FROM employee ORDER BY 2014-YEAR (hireDate) DESC

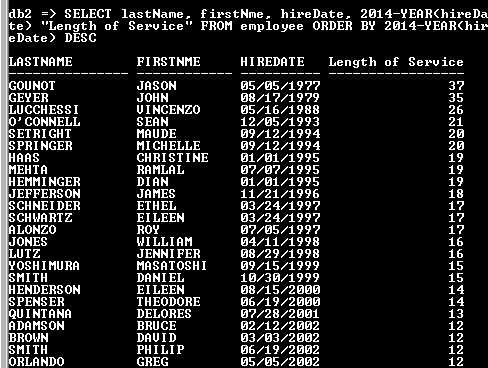


Figure 3-24. Sort according to descending computed value

Here we computed first the length of service and sorted its computed value from highest to lowest.

Be aware of the syntax when working with WHERE with an ORDER BY clause. WHERE should come first before ORDER BY.

Let us determine the highest paid employees of the company. List last name, first name, job, and salary. Sort salary in descending order. Exclude president and manager positions.



Figure 3-25. WHERE with ORDER BY clause

**Chapter Exercise**

I. **Multiple choice**: Refer to SAMPLE database. Select the letter of the correct answer.

1. Which of the following SQL scripts displays salary between 40,000 and 50,000?

a. SELECT \* FROM employee WHERE salary >= 40000 OR salary <= 50000

b. SELECT \* FROM employee WHERE salary >=40000 AND salary <= 50000

c. SELECT \* FROM employee WHERE salary BETWEEN 41000 AND 50000

d. SELECT \* FROM employee WHERE salary < 50000

2. What is the correct way to concat last name, first name, and middle initial to column with a name of “Full Name” that will produce like this in the output: ALONZO, ROY R?

a. SELECT CONCAT (lastName, firstNme, midInit) FROM employee

b. SELECT lastName + “, “+ firstNme + “ “ + midInit “Full Name” FROM employee

c. SELECT CONCAT (CONCAT (lastName, “, “), firstNme, “ “ , midInit) “Full Name” FROM employee

d. SELECT CONCAT(CONCAT(CONCAT(CONCAT(lastName, “, “), firstNme), “ “), midInit) “Full Name” FROM employee

3. What would be the appropriate SQL script that will produce the following output?



Figure 3-26. Filter question

1. SELECT lastName, firstNme FROM employee WHERE lastName LIKE “\_CH%” ORDER BY lastName
2. SELECT lastName, firstNme FROM employee WHERE lastName LIKE ‘\_CH%’
3. SELECT lastName, firstNme FROM employee WHERE lastName LIKE ‘\_C%’
4. SELECT lastName, firstNme FROM employee WHERE lastName LIKE ‘\_C%’ ORDER BY lastName

4. How will you display last name, first name, salary, bonus, commission, and the total of salary, bonus, and commission of EMPLOYEE table?

a. SELECT lastName, firstNme, salary, bonus, comm, “Total” FROM employee

b. SELECT lastName, firstNme, salary, bonus, comm, salary+bonus+comm “Total” FROM employee

c. SELECT lastName, firstNme, salary, bonus, comm, salary+bonus+comm ‘Total’ FROM employee

d. SELECT lastName, firstNme, salary, bonus, comm, (salary+bonus+comm) FROM employee

5. What should be the script to produce the Figure 3-27?

a. SELECT workDept, lastName, firstNme, sex FROM employee ORDER BY workdept, lastName

b. SELECT workDept, lastName, firstNme, sex FROM employee ORDER BY workdept, sex

c. SELECT workDept, lastName, firstNme, sex FROM employee ORDER BY workdept, sex, lastName

d. SELECT workDept, lastName, firstNme, sex FROM employee ORDER BY workdept, lastName, sex



Fig 3-27. Sorting problem

6. If we want to display last name, first name, and sex from EMPLOYEE table, and instead of “SEX” as the header title of the column we change it by “GENDER” , what would be the script?

a. SELECT lastName, firstNme, gender FROM employee

b. SELECT lastName, firstNme, “GENDER” FROM employee

c. SELECT lastName, firstNme, sex “GENDER” FROM employee

d. SELECT “LASTNAME”, “FIRSTNME”, “SEX” “GENDER” FROM employee

7. Which of the following will list all females of department A00, D11, E21 sorted by department?

a. SELECT workDept, lastName, firstNme, sex FROM employee WHERE workDept IN (‘A00’, ‘D11’, ‘E21’) ORDER BY workdept

b. SELECT workDept, lastName, firstNme, sex FROM employee ORDER BY workdept WHERE workDept IN (‘A00’, ‘D11’, ‘E21’)

c. SELECT workDept, lastName, firstNme, sex FROM employee WHERE workDept NOT IN (‘A00’, ‘D11’, ‘E21’) ORDER BY workdept

d. SELECT workDept, lastName, firstNme, sex FROM employee WHERE workDept IN (‘A00’, ‘D11’, ‘E21’)

8. How do you print the 5% witholding tax of salary and compute its net? Sort output according to workdept and lastname

a. SELECT workDept, lastName, firstNme, salary, 0.5\*salary, salary – 0.5\*salary FROM employee ORDER BY workDept, lastName

b. SELECT workDept, lastName, firstNme, salary, 0.05\* salary “Tax”, salary – 0.05\*salary “Net FROM employee ORDER BY workDept, lastName

c. SELECT workDept, lastName, firstNme, salary, 0.05\* salary “Tax”, salary – 0.05\*salary “Net FROM employee ORDER BY lastName, workdept

d. SELECT workDept, lastName, firstNme, salary, 0.5\* salary “Tax”, salary – 0.5\*salary “Net FROM employee ORDER BY workDept, lastName

9. What script will display employees’ workdept, last name, first name, bonus with bonus having 600, 500 and 300. The output requires to be displayed according to work dept and last name.

a. SELECT workDept, lastName, firstNme, bonus FROM employee WHERE bonus = 600 AND bonus = 500 AND bonus = 300 ORDER BY workDept, lastName

b. SELECT workDept, lastName, firstNme, bonus FROM employee WHERE bonus = 600 AND bonus = 500 AND bonus = 300 ORDER BY workDept, lastName

c. SELECT workDept, lastName, firstNme, bonus FROM employee WHERE bonus = 600 OR bonus = 500 OR bonus = 300 ORDER BY workDept, lastName

d. SELECT workDept, lastName, firstNme, bonus FROM employee WHERE bonus NOT IN (600,500,300) ORDER BY workDept, lastName

10. Which of the following script will determine whose age is between 18 and 25 upon hiring?

a. SELECT workDept, lastName, firstNme, 2014-YEAR(hireDate) “Age hired” FROM employee ORDER workDept, lastName

b. SELECT workDept, lastName, firstNme, YEAR(hireDate)- YEAR(birthDate) “Age hired” FROM employee ORDER workDept, lastName

c. SELECT workDept, lastName, firstNme, YEAR(hireDate)-2014 “Age hired” FROM employee ORDER workDept, lastName

d. SELECT workDept, lastName, firstNme, YEAR(birthDate)- YEAR(hireDate) “Age hired” FROM employee ORDER workDept, lastName

II. **Problems**

Refer to HOPE database. If you have not created the database (or prefer a new one), click this link to download and install all tables under the database. Make sure to SET SCHEMA store after you CONNECT TO hope to access the tables.

1. Determine employees separated from the company. Display employee’s number, last name, first name, hire date, and separation date.
2. Create script that will determine the youngest age employed by the company upon hiring. Display employee’s number, last name, first name, hire date, and age.
3. How many months did the employee stay in the company? List employee’s last name, first name, hire date, separation date, and number of months stayed. HINT: Use MONTH() function.
4. Formulate a script that will determine employees hired on month of May, July, October 2010. Display employee’s number, last name, hire date according to hiredate and last name.
5. Who among the employees has “ch” on their last name? Display employee’s number, last name and first name.
6. Determine individual customers who lives at the California (CA). Individual customer has payterm of ‘COD’. List customer number, name, and address sorted according to customer name.
7. List customer’s number, name, and address who lives outside New York (NY) and California (CA).
8. Show official receipt number, payment date, amount, and transaction number from PAYMENT table for the month of August 2010 amounting to 1,000 and up.
9. Generate report from the biggest to lowest payment made at PAYMENT table. Include all columns of the table.
10. Produce a listing of payment made each month sorted according to biggest to the lowest amount received from the PAYMENT table. HINT: Sort according to year and then month.

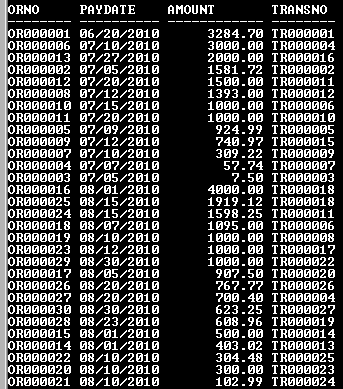


Figure 3-28. Sorted according to biggest to smallest payment made each month

**Answers to selected numbers**

I. Multiple choice

1. B

3. D

5. C

7. A

9. C

II. Problems

1. SELECT empno, lastName, firstName, hireDate, sepDate FROM employee

WHERE sepDate IS NOT NULL

3. SELECT lastName, firstName, hireDate, sepDate, MONTH(sepDate – hiredate)

“Months stayed” FROM employee

WHERE sepDate IS NOT NULL

5. SELECT lastName, firstName FROM employee WHERE lastName LIKE ‘%ch%’

7. SELECT custNo, custName, address FROM customer

WHERE address NOT LIKE ‘% CA %’ AND address NOT LIKE ‘% NY %’

9. SELECT \* FROM paymenter ORDER BY 3 DESC