```
-- Chapter-1(Retrieving Data Using the SQL SELECT Statement)
-- 1.    SELECT
/*
Writing SQL Statements

*   SQL statements are not case-sensitive.
*   SQL statements can be on one or more lines.
*   Keywords cannot be abbreviated or split across lines.
*   Clauses are usually placed on separate lines.
*   Indents are used to enhance readability.
*   In SQL Developer, SQL statements can optionally be terminated by a semicolon (;).
Semicolons are required if you execute multiple SQL statements.
*   In SQL*Plus, you are required to end each SQL statement with a semicolon (;).

Column Heading For SQL

*   Character and Date column headings are left-aligned
*   Number column headings are right-aligned
*   Default heading display: Uppercase

*/

Select Department_Id,  Location_Id
  From   Departments;


--    Arithmetic Expressions (Add +,Subtract -, Multiply *, Divide /)

Select Last_Name, Salary,  Salary + 300, Salary - 300, Salary * 3,Salary / 2 ,
      (Salary + 500)-800/2,((Salary + 500)-800)/2 ,(Salary*3 + 500)-800/2,((Salary + 500)-
800)*2
  From Employees;

--  Operator Precedence
/*

If an arithmetic expression contains more than one operator, multiplication and
division  are  evaluated
first. If operators in an expression are of the same priority, then evaluation is done from
left to right.
You can use parentheses to force the expression that is enclosed by parentheses to be
evaluated first.


--    Null Value
```

```
*   A null is a value that is unavailable, unassigned, unknown,or inapplicable.
*   A null is not the same as a zero or a blank space.

*/
--    Column Alias
--    Concatenation Operator
--    Literal Character Strings
--   Alternative Quote (q) Operator

Select Last_Name,Last_Name  As  Name, Salary,  Salary + 300, Salary - 300, Salary *
3,Salary / 2 ,
      (Salary + 500)-800/2,((Salary + 500)-800)/2 ,(Salary*3 + 500)-800/2,((Salary + 500)-
800)*2,
      Commission_Pct  Comm,Last_Name||Job_Id As "Employees",
      Last_Name ||' is a '||Job_Id As "Employee Details",First_Name ||' it''s assigned
Manager Id: ' ||
      Manager_Id As  "Department  and  Manager"
  From Employees;

--    Duplicate Rows

Select Department_Id  From   Employees;
/
Select Distinct Department_Id
  From Employees;
/
--   DESCRIBE Command

Describe Employees;

   --Chapter-2 (Restricting and Sorting Data)
--    WHERE Clause

Select  Last_Name,Last_Name  As  Name, Salary,  Salary + 300, Salary - 300, Salary *
3,Salary / 2 ,
   (Salary + 500)-800/2,((Salary + 500)-800)/2 ,(Salary*3 + 500)-800/2,((Salary + 500)-
800)*2,
   Commission_Pct  Comm,Last_Name||Job_Id As "Employees",
   Last_Name ||' is a '||Job_Id As "Employee Details",First_Name ||' it''s assigned
Manager Id: ' ||
   Manager_Id As  "Department  and  Manager"
From   Employees
Where  Department_Id = 90 ;
/
```

```
Select  Last_Name,Last_Name  As  Name, Salary,  Salary + 300, Salary - 300, Salary *
3,Salary / 2 ,
   (Salary + 500)-800/2,((Salary + 500)-800)/2 ,(Salary*3 + 500)-800/2,((Salary + 500)-
800)*2,
   Commission_Pct  Comm,Last_Name||Job_Id As "Employees",
   Last_Name ||' is a '||Job_Id As "Employee Details",First_Name ||' it''s assigned
Manager Id: ' ||
   Manager_Id As  "Department  and  Manager"
From   Employees
Where  Last_Name =  'Whalen'  ;
/*
  Comparison Conditions (Equal to                    =
                  Greater than                  >,
                  Greater than or equal to       >=,
                  Less than                    <,
                  Less than or equal to          <=,
                  Not equal to                 <>  OR =!,
                  Between two values (inclusive)    BETWEEN  ...AND...,
                  Match any of a list of values     IN,
                  Match a character pattern         LIKE,
                  Is a null value              NULL
                 )
*/

Select  Last_Name,Last_Name  As  Name, Salary,  Salary + 300, Salary - 300, Salary *
3,Salary / 2 ,
   (Salary + 500)-800/2,((Salary + 500)-800)/2 ,(Salary*3 + 500)-800/2,((Salary + 500)-
800)*2,
   Commission_Pct  Comm,Last_Name||Job_Id As "Employees",
   Last_Name ||' is a '||Job_Id As "Employee Details",First_Name ||' it''s assigned
Manager Id: ' ||
   Manager_Id As  "Department  and  Manager"
From   Employees
Where Salary <= 3000 ;
/
Select  Last_Name,Last_Name  As  Name, Salary,  Salary + 300, Salary - 300, Salary *
3,Salary / 2 ,
   (Salary + 500)-800/2,((Salary + 500)-800)/2 ,(Salary*3 + 500)-800/2,((Salary + 500)-
800)*2,
   Commission_Pct  Comm,Last_Name||Job_Id As "Employees",
   Last_Name ||' is a '||Job_Id As "Employee Details",First_Name ||' it''s assigned
Manager Id: ' ||
   Manager_Id As  "Department  and  Manager"
From   Employees
Where  Salary  Between 2500 And 3500  ;
/
```

```
Select  Last_Name,Last_Name  As  Name, Salary,  Salary + 300, Salary - 300, Salary *
3,Salary / 2 ,
   (Salary + 500)-800/2,((Salary + 500)-800)/2 ,(Salary*3 + 500)-800/2,((Salary + 500)-
800)*2,
   Commission_Pct  Comm,Last_Name||Job_Id As "Employees",
   Last_Name ||' is a '||Job_Id As "Employee Details",First_Name ||' it''s assigned
Manager Id: ' ||
   Manager_Id As  "Department  and  Manager"
From   Employees
Where  Manager_Id  In (100, 101, 201) ;
/
Select Last_Name,Last_Name  As  Name, Salary,  Salary + 300, Salary - 300, Salary *
3,Salary / 2 ,
   (Salary + 500)-800/2,((Salary + 500)-800)/2 ,(Salary*3 + 500)-800/2,((Salary + 500)-
800)*2,
   Commission_Pct  Comm,Last_Name||Job_Id As "Employees",
   Last_Name ||' is a '||Job_Id As "Employee Details",First_Name ||' it''s assigned
Manager Id: ' ||
   Manager_Id As  "Department  and  Manager"
From   Employees
Where  First_Name Like 'S%' ;
/
Select Last_Name,Last_Name  As  Name, Salary,  Salary + 300, Salary - 300, Salary *
3,Salary / 2 ,
   (Salary + 500)-800/2,((Salary + 500)-800)/2 ,(Salary*3 + 500)-800/2,((Salary + 500)-
800)*2,
   Commission_Pct  Comm,Last_Name||Job_Id As "Employees",
   Last_Name ||' is a '||Job_Id As "Employee Details",First_Name ||' it''s assigned
Manager Id: ' ||
   Manager_Id As  "Department  and  Manager"
From   Employees
Where  Last_Name Like '_o%' ;
/
Select  Last_Name,Last_Name  As  Name, Salary,  Salary + 300, Salary - 300, Salary *
3,Salary / 2 ,
   (Salary + 500)-800/2,((Salary + 500)-800)/2 ,(Salary*3 + 500)-800/2,((Salary + 500)-
800)*2,
   Commission_Pct  Comm,Last_Name||Job_Id As "Employees",
   Last_Name ||' is a '||Job_Id As "Employee Details",First_Name ||' it''s assigned
Manager Id: ' ||
   Manager_Id As  "Department  and  Manager"
From   Employees
Where  Manager_Id Is Null ;

/*
```

```
Logical Conditions (AND Returns TRUE if both component conditions are true
                    OR  Returns TRUE if either component condition is true
                    NOT Returns TRUE if the following condition is false  NOT
                   )
*/

Select  Last_Name,Last_Name  As  Name, Salary,  Salary + 300, Salary - 300, Salary *
3,Salary / 2 ,
   (Salary + 500)-800/2,((Salary + 500)-800)/2 ,(Salary*3 + 500)-800/2,((Salary + 500)-
800)*2,
   Commission_Pct  Comm,Last_Name||Job_Id As "Employees",
   Last_Name ||' is a '||Job_Id As "Employee Details",First_Name ||' it''s assigned
Manager Id: ' ||
   Manager_Id As  "Department  and  Manager"
From   Employees
Where  Salary >=10000
And  Job_Id Like '%MAN%' ;
/
Select  Last_Name,Last_Name  As  Name, Salary,  Salary + 300, Salary - 300, Salary *
3,Salary / 2 ,
   (Salary + 500)-800/2,((Salary + 500)-800)/2 ,(Salary*3 + 500)-800/2,((Salary + 500)-
800)*2,
   Commission_Pct  Comm,Last_Name||Job_Id As "Employees",
   Last_Name ||' is a '||Job_Id As "Employee Details",First_Name ||' it''s assigned
Manager Id: ' ||
   Manager_Id As  "Department  and  Manager"
From   Employees
Where  Salary >= 10000
Or  Job_Id Like '%MAN%';
/
Select  Last_Name,Last_Name  As  Name, Salary,  Salary + 300, Salary - 300, Salary *
3,Salary / 2 ,
   (Salary + 500)-800/2,((Salary + 500)-800)/2 ,(Salary*3 + 500)-800/2,((Salary + 500)-
800)*2,
   Commission_Pct  Comm,Last_Name||Job_Id As "Employees",
   Last_Name ||' is a '||Job_Id As "Employee Details",First_Name ||' it''s assigned
Manager Id: ' ||
   Manager_Id As  "Department  and  Manager"
From   Employees
Where  Job_Id Not In ('IT_PROG', 'ST_CLERK', 'SA_REP') ;
/

--   Rules of Precedence

Select  Last_Name,  Job_Id,  Salary
From   Employees
```

```
Where  Job_Id = 'SA_REP'
Or  Job_Id = 'AD_PRES'
And  Salary > 15000;
/
Select Last_Name, Job_Id, Salary
From   Employees
Where  (Job_Id = 'SA_REP'
Or  Job_Id  =  'AD_PRES')
And   Salary > 15000;

--    ORDER BY Clause

Select   Hire_Date,Last_Name,Last_Name  As  Name, Salary,  Salary + 300, Salary -
300, Salary * 3,Salary / 2 ,
   (Salary + 500)-800/2,((Salary + 500)-800)/2 ,(Salary*3 + 500)-800/2,((Salary + 500)-
800)*2,
   Commission_Pct  Comm,Last_Name||Job_Id As "Employees",
   Last_Name ||' is a '||Job_Id As "Employee Details",First_Name ||' it''s assigned
Manager Id: ' ||
   Manager_Id As  "Department  and  Manager"
From  Employees
Order By Hire_Date ;
/
Select   Hire_Date,Last_Name,Last_Name  As  Name, Salary,  Salary + 300, Salary -
300, Salary * 3,Salary / 2 ,
   (Salary + 500)-800/2,((Salary + 500)-800)/2 ,(Salary*3 + 500)-800/2,((Salary + 500)-
800)*2,
   Commission_Pct  Comm,Last_Name||Job_Id As "Employees",
   Last_Name ||' is a '||Job_Id As "Employee Details",First_Name ||' it''s assigned
Manager Id: ' ||
   Manager_Id As  "Department  and  Manager"
From  Employees
Order By Hire_Date Desc
/
Select  Last_Name ||' is a '||Job_Id As "Employee
Details",Hire_Date,Last_Name,Last_Name  As  Name, Salary,  Salary + 300, Salary -
300, Salary * 3,Salary / 2 ,
   (Salary + 500)-800/2,((Salary + 500)-800)/2 ,(Salary*3 + 500)-800/2,((Salary + 500)-
800)*2,
   Commission_Pct  Comm,Last_Name||Job_Id As "Employees",
   First_Name ||' it''s assigned Manager Id: ' ||
   Manager_Id As  "Department  and  Manager"
From   Employees
Order By "Employee Details"  ;
/
```

```
Select  Salary,Last_Name ||' is a '||Job_Id As "Employee
Details",Hire_Date,Last_Name,Last_Name  As  Name, Salary + 300, Salary - 300,
Salary * 3,Salary / 2 ,
   (Salary + 500)-800/2,((Salary + 500)-800)/2 ,(Salary*3 + 500)-800/2,((Salary + 500)-
800)*2,
   Commission_Pct  Comm,Last_Name||Job_Id As "Employees",
   First_Name ||' it"s assigned Manager Id: ' ||
   Manager_Id As  "Department  and  Manager"
From   Employees
Order By Department_Id, Salary Desc;

--    Substitution Variables

Select  Employee_Id,  Last_Name,  Salary,  Department_Id
From   Employees
Where  Employee_Id =  &Employee_Num;
/
Select  Last_Name,  Department_Id,  Salary*12
From   Employees
Where  Job_Id =  '&job_title';
/
Select  Employee_Id,  Last_Name,  Job_Id,&Column_Name
From   Employees
Where  &Condition
Order By &Order_Column ;
/
Select   Employee_Id, Last_Name, Job_Id, &&Column_Name
From  Employees
Order By &Column_Name ;
/
/*
Chapter-3 (Using Single-Row Functions to Customize Output)
  SQL Functions
            1. Single-Row Functions
               a.  Character Functions
                  i.    Case-Manipulation Functions (LOWER, UPPER, INITCAP)
                  ii.   Character-Manipulation Functions (CONCAT, SUBSTR, LENGTH,
INSTR, LPAD, RPAD, TRIM, REPLACE)
*/

      Select  Lower(Last_Name),Upper(Last_Name),Initcap(Last_Name),Concat('Hello',
'World'),
            Substr('HelloWorld',1,5), Length('HelloWorld'), Instr('HelloWorld', 'W'),Salary,
            Lpad(10001,10,'*'), Rpad(10001, 10, '*'), Replace('JACK and JUE','J','BL'),
            Trim('H' From 'HelloWorld')
      From   Employees
```

```
      Where  Department_Id  = 50;
      /

      Select  Employee_Id,  Concat(First_Name,  Last_Name)
            Job_Id, Length (Last_Name), Instr(Last_Name,  'a')  "Contains  'a'?"
      From   Employees
      Where  Substr(Job_Id, 4) = 'REP';
      /
/*
  Number Functions (Rounds value to specified decimal ROUND,
                Truncates value to specified decimal TRUNC,
                Returns remainder of division MOD
               )

      Number Element

   9   Numeric position (number of 9s determine display  999999  1234 width)
   0   Display leading zeros  099999  001234
   $   Floating dollar sign  $999999  $1234
   L   Floating local currency symbol  L999999  FF1234
   D    Returns in the specified position the decimal  99D9999.99 character. The default
is a period (.).
       Decimal point in position specified  999999.99  1234.00
   G  Returns the group separator in the specified  9,999  9G999 position. You can
specify multiple group separators in a number format model.
   ,  Comma in position specified  999,999  1,234
   MI  Minus signs to right (negative values)  999999MI  1234-
   PR  Parenthesize negative numbers  999999PR  <1234>
   EEEE  Scientific notation (format must specify four Es)  99.999EEEE  1.234E+03
   U  Returns in the specified position the "Euro" (or  U9999  €1234 other) dual currency
   V  Multiply by 10 n times  (n = number of 9s after V)  9999V99  123400
   S  Returns the negative or positive value  S9999  -1234 or +1234
   B  Display zero values as blank, not 0  B9999.99  1234.00
*/

   Select Round(45.923,2), Round(45.923,0),Round(45.923,-1),Trunc(45.923,2),
Trunc(45.923),Trunc(45.923,-1),Mod(2000010, 5000)
   From   Dual;

/*
Date Functions

   Element of the date format

      YYYY              Full year in numbers
      YEAR              Year spelled out (in English)
```

```
MM                  Two-digit value for month
MONTH                 Full name of the month
MON                  Three-letter abbreviation of the month
DY                  Three-letter abbreviation of the day of the week
DAY                  Full name of the day of the week
DD                  Numeric day of the month
HH24:MI:SS AM
DD "of" MONTH
AM or PM              Meridian indicator
A.M. or P.M.          Meridian indicator with periods
HH or HH12 or HH24      Hour of day, or hour (1-12), or hour (0-23)
MI                  Minute (0-59)
SS                  Second (0-59)
SSSSS                 Seconds past midnight (0-86399)

  RR Date Format

    The RR date format is similar to the YY element, but you can use it to specify
different centuries. Use the
    RR date format element instead of YY so that the century of the return value varies
according to the
    specified two-digit year and the last two digits of the current year. The table in the
slide
    summarizes the behavior of the RR element.

*/


  Select Last_Name,Hire_Date,(Sysdate-Hire_Date),Sysdate+50,Sysdate-13,(Sysdate-
Hire_Date)/7,
      Months_Between('01-SEP-95','11-JAN-94'),Add_Months (Hire_Date,
6),Next_Day(Hire_Date,  'FRIDAY'),
      Last_Day(Hire_Date),Round(Sysdate,'MONTH'), Round(Sysdate
,'YEAR'),Trunc(Sysdate ,'MONTH'),Trunc(Sysdate ,'YEAR'),
      To_Char(Hire_Date,  'MM/YY'),To_Char(Hire_Date, 'fmDD Month
YYYY'),To_Char(Hire_Date,'fmDdspth  "of"  Month  YYYY  fmHH:MI:SS  AM'),
      To_Char(Salary,
'$99,999.00'),Substr(Phone_Number,1,3),To_Number(Substr(Phone_Number,1,3),
'999999999'),
      To_Date('May  24, 1999', 'fmMonth DD, YYYY'),To_Date('03042016',
'DD/MM/YYYY'),To_Date('01-Jan-90','DD-Mon-YY'),
      To_Date('01-Jan-90','DD-Mon-RR'),Upper(Concat(Substr(Last_Name,  1,  8),
'_US')),
      To_Char(Next_Day(Add_Months(Hire_Date, 6), 'FRIDAY'),'fmDay,  Month  DDth,
YYYY') "Next 6 Month Review" ,
```

```
        Nvl(Commission_Pct,0), Nvl(Hire_Date,'01-JAN-97'), Nvl(Job_Id,'No Job
Yet'),(Salary*12) + (Salary*12*Nvl(Commission_Pct, 0)),
        Nvl2(Commission_Pct,'SAL+COMM', 'SAL'),Length(First_Name)
"expr1",Length(Last_Name)  "expr2",
        Nullif(Length(First_Name), Length(Last_Name))
Result,Coalesce(Manager_Id,Commission_Pct,  -1),
        Case Job_Id When 'IT_PROG'  Then  1.10*Salary
                When 'ST_CLERK' Then  1.15*Salary
                When 'SA_REP'   Then  1.20*Salary
                Else  Salary
        End  "REVISED_SALARY",
        Case When Salary<5000 Then 'Low'
            When Salary<10000 Then 'Medium'
            When Salary<20000 Then 'Good'
            Else 'Excellent'
        End  Qualified_Salary ,
        Decode(Job_Id, 'IT_PROG',   1.10*Salary, 'ST_CLERK', 1.15*Salary,'SA_REP',
1.20*Salary,Salary) Revised_Salary1,
        Decode (Trunc(Salary/2000, 0), 0, 0.00,1,  0.09,2,  0.20,3,  0.30,4,  0.40,5,
0.42,6, 0.44,0.45) Tax_Rate
   From   Employees
   Where  Hire_Date <'01-FEB-10' --MONTHS_BETWEEN (SYSDATE, hire_date) <
120;

/*

   Chapter-4 (Reporting Aggregated Data Using the Group Functions)

          1.    Group Functions(AVG, COUNT, MAX, MIN, STDDEV, SUM, VARIANCE)

*/

        Select Department_Id Dept_Id, Job_Id, Sum(Salary),Count(Distinct
Department_Id),
               Count(*) ,Avg(Salary),Avg(Nvl(Commission_Pct,  0)),
Max(Salary),Min(Salary), Sum(Salary)
        From   Employees
        Where  Job_Id Like '%REP%'
        Group By Department_Id, Job_Id
        Having Sum(Salary) > 5000
        Order By Department_Id,Sum(Salary);


-- Chapter-5 (Displaying Data from Multiple Tables)

--   Natural Joins
```

```
--  Qualifying Ambiguous Column Names

    Select Department_Id, Department_Name,Location_Id, City
    From   Departments
    Natural Join Locations;
    /

    Select L.City, D.Department_Name
    From   Locations L Join Departments D Using (Location_Id)
    Where  Location_Id = 1400;
    /

    Select Department_Id, Department_Name,Locations.Location_Id,City
      From Departments, Locations
     Where Departments.Location_Id = Locations.Location_Id
     And  Locations.Location_Id = 1400;
    /

    Select D.Department_Id, D.Department_Name,L.Location_Id,L.City
      From Departments D, Locations L
     Where D.Location_Id = L.Location_Id--(+)
      --and  l.location_id = 1400;
       /

--  Chapter-6 (Using Sub queries to Solve Queries)
   --  Subquery
   --  Types of Subqueries
      --  Single-row subquery (Operator =,>,>=,<,<=,<>)

          Select Last_Name, Salary
          From   Employees
          Where  Salary > (Select Salary
                           From   Employees
                           Where  Last_Name = 'Abel'
                          );
       /

          Select Last_Name, Job_Id
          From   Employees
          Where  Job_Id = (Select Job_Id
                           From   Employees
                           Where  Employee_Id = 141
                          );
          /

          Select  Last_Name,  Job_Id,  Salary
```

```
From   Employees
Where  Job_Id =  (Select Job_Id
                  From   Employees
                  Where  Employee_Id = 141
                 )
  And  Salary >  (Select Salary
                  From Employees
                  Where Employee_Id = 143
                 );
/

Select Last_Name, Job_Id, Salary
From   Employees
Where  Salary = (Select Min(Salary)
                 From   Employees
                );
/

Select Department_Id, Min(Salary)
From  Employees
Group By Department_Id
Having   Min(Salary) > (Select  Min(Salary)
                        From   Employees
                        Where  Department_Id = 50
                       );
/

Select Last_Name, Job_Id
From   Employees
Where  Job_Id = (Select Job_Id
                 From   Employees
                 Where  Last_Name = 'Haas'
                );

--   Multiple-row subquery (Operator IN, ANY, ALL)

Select Employee_Id, Last_Name, Job_Id, Salary
From   Employees
Where  Salary < Any (Select Salary
                     From   Employees
                     Where  Job_Id = 'IT_PROG'  --9000, 6000, 4200
                    )
 And  Job_Id <> 'IT_PROG';
/

Select Employee_Id, Last_Name, Job_Id, Salary
```

```
            From   Employees
            Where  Salary < All (Select Salary
                                 From   Employees
                                 Where  Job_Id = 'IT_PROG' --9000, 6000, 4200
                                )
              And  Job_Id <> 'IT_PROG';
           /

           Select Emp.Last_Name
            From   Employees Emp
            Where  Emp.Employee_Id Not In (Select Mgr.Manager_Id
                                           From   Employees Mgr
                                          );
           /
           Select Last_Name From Employees
            Where  Employee_Id Not In (Select Manager_Id
                                       From   Employees
                                       Where  Manager_Id Is Not Null
                                      );
           /


/*
      Chapter-7 (Using the Set Operators)
           Set Operators ( UNION, UNION ALL, INTERSECT, MINUS)

               The expressions in the SELECT lists must match in number and data type.
               Parentheses can be used to alter the sequence of execution.
               The ORDER BY clause:- Can appear only at the very end of the statement
                Will  accept  the  column  name,  aliases  from  the  first SELECT
statement,
                or the positional notation

*/

          Select  Employee_Id,  Job_Id
          From   Employees
          Union
          Select  Employee_Id,  Job_Id
          From   Job_History;
          /

          Select Employee_Id, Job_Id, Department_Id
          From   Employees
          Union All
          Select  Employee_Id,  Job_Id,  Department_Id
```

```
From   Job_History
Order By  Employee_Id;
/

Select Employee_Id, Job_Id
From   Employees
Intersect
Select  Employee_Id,  Job_Id
From   Job_History;
/

Select Employee_Id
From   Employees
Minus
Select  Employee_Id
From   Job_History;
/

Select Department_Id, To_Number(Null) Location, Hire_Date
From     Employees
Union
Select  Department_Id,  Location_Id, To_Date(Null)
From   Departments;
/

Select  Employee_Id,  Job_Id,Salary
From   Employees
Union
Select  Employee_Id,  Job_Id,0
From   Job_History;
/

Select  'sing'  As  "My  dream",  3  A_Dummy
From Dual
Union
Select  'I"d  like  to  teach',  1  A_Dummy
From Dual
Union
Select  'the  world  to',  2  A_Dummy
From Dual
Order By A_Dummy;
/

/*
    Chapter-8 (Manipulating Data)
       Data Manipulation Language
```

```
            Adding a New Row to a Table
            INSERT Statement Syntax
            Inserting New Rows
            Inserting Rows with Null Values
            UPDATE Statement Syntax
            Updating Rows Based on Another Table
            DELETE Statement
            TRUNCATE
            COMMIT or ROLLBACK
*/


      --  Insert

          Insert Into Departments(Department_Id,Department_Name,  Manager_Id,
Location_Id)
                      Values (70, 'Public Relations', 100, 1700);
          /

          Insert Into Departments (Department_Id,Department_Name  )
                       Values (30, 'Purchasing');
          /

          Insert Into Employees (Employee_Id,First_Name, Last_Name,Email,
Phone_Number,Hire_Date,  Job_Id,  Salary,Commission_Pct,
Manager_Id,Department_Id)
                     Values (113,'Louis', 'Popp','LPOPP', '515.124.4567', Sysdate,
'AC_ACCOUNT', 6900,Null, 205, 100);
          /

          Insert Into Employees
                     Values (114,'Den', 'Raphealy','DRAPHEAL',
'515.127.4561',To_Date('FEB  3,  1999',  'MON  DD,  YYYY'),'AC_ACCOUNT', 11000,
Null, 100, 30);
          /

          Insert Into Departments (Department_Id, Department_Name, Location_Id)
                      Values (&Department_Id, '&department_name',&Location);
          /

          Insert Into Sales_Reps(Id, Name, Salary, Commission_Pct)
                     Select Employee_Id, Last_Name, Salary, Commission_Pct
                      From Employees
                      Where Job_Id Like '%REP%';
          /
```

```
        Insert Into (Select Employee_Id, Last_Name, Email,  Hire_Date,  Job_Id,
Salary,Department_Id
                     From Employees
                     Where Department_Id = 50
                    )
             Values (99999, 'Taylor', 'DTAYLOR',To_Date('07-JUN-99',  'DD-MON-
RR'),'ST_CLERK', 5000, 50);

      --   Update

          Update Employees
            Set Department_Id = 70
           Where Employee_Id = 113;
          /

          Update Copy_Emp
            Set Department_Id = 110;
          /

          Update Employees
            Set Job_Id  = (Select Job_Id
                           From Employees
                          Where  Employee_Id = 205
                         ),
                Salary  = (Select Salary
                           From Employees
                          Where Employee_Id = 205)
          Where Employee_Id  =  114;
          /

          Update Copy_Emp
            Set Department_Id = (Select Department_Id
                                 From Employees
                                Where Employee_Id = 100
                               )
           Where Job_Id = (Select Job_Id
                           From Employees
                          Where Employee_Id = 200
                         );
          /

          -- Delete

          Delete From Departments
          Where  Department_Name = 'Finance';
          /
```

```
Delete From Employees
     Where Department_Id = (Select Department_Id
                            From Departments
                            Where Department_Name Like '%Public%'
                           );
/

Truncate Table Copy_Emp;
/

Delete From Employees
Where  Employee_Id = 99999;
/
Insert Into Departments
Values (290, 'Corporate Tax', Null, 1700);
/
Commit;
/
Delete From Copy_Emp;
/
Rollback;
/
Delete From Test Where  Id = 100;
/
Select * From   Test Where  Id = 100;
/
Commit;

/*
    Chapter-9(Using DDL Statements to Create and Manage Tables)

      Database Objects
      Table

      Table names and column names:
        Must begin with a letter
        Must be 1-30 characters long
        Must contain only A-Z, a-z, 0-9, _, $, and #
        Must not duplicate the name of another object owned by the same user
        Must not be an Oracle server-reserved word

         - NOT NULL
         - UNIQUE
         - PRIMARY KEY
         - FOREIGN KEY
```

- CHECK

e. Synonym
2. Naming Rules
3. CREATE TABLE Statement
4. Referencing Another User's Tables
5. DEFAULT
6. Data Types
a. VARCHAR2
b. CHAR
c. NUMBER
d. DATE
e. LONG
f. CLOB
g. RAW and LONG RAW
h. BLOB
i. BFILE
j. ROWID
7. Date time Data Types
8. Including Constraints
a. NOT NULL
b. UNIQUE
c. PRIMARY KEY
d. FOREIGN KEY
e. CHECK
9. Constraint Guidelines
10. Defining Constraints
11. Violating Constraints
12. Creating a Table by Using a Subquery
13. ALTER TABLE Statement
14. Dropping a Table

*/

```
Create Table Hire_Dates(Id         Number(8),
                Hire_Date   Date Default Sysdate
               );
/

Create  Table  Dept(Deptno      Number(2),
              Dname       Varchar2(14),
              Loc         Varchar2(13),
              Create_Date Date Default Sysdate
```

```
            );
          /
Create Table Time_Example3(Day_Duration Interval Day (3) To Second
                  );

Insert  Into  Time_Example3(Day_Duration)
              Values (Interval '180' Day(3)
                   );

Select Sysdate + Day_Duration "Half Year"
From   Time_Example3;  --today's date is 11-11-2008
/

Create  Table  Employees(Employee_Id  Number(6) Constraint Emp_Emp_Id_Pk
Primary Key,
                 First_Name      Varchar2(20)
                 );
/

Create  Table Dept80 As Select  Employee_Id, Last_Name,Salary*12 Annsal,Hire_Date
                  From  Employees
                 Where   Department_Id = 80;
/

Create Table Employees(Employee_Id  Number(6),
                 First_Name   Varchar2(20),
                 Job_Id  Varchar2(10) Not Null,
                Constraint Emp_Emp_Id_Pk Primary Key (Employee_Id)
               );
/

Create Table Employees(Employee_Id  Number(6),
                 Last_Name  Varchar2(25) Not Null,
                 Email  Varchar2(25),
                 Salary  Number(8,2),
                 Commission_Pct   Number(2,2),
                 Hire_Date  Date Not Null,
               Constraint Emp_Email_Uk Unique(Email)
                );
/

Create Table Employees(Employee_Id  Number(6),
                 Last_Name  Varchar2(25) Not Null,
                 Email  Varchar2(25),
                 Salary  Number(8,2),
                 Commission_Pct   Number(2,2),
```

```
                Hire_Date  Date Not Null,
                Department_Id  Number(4),
              Constraint Emp_Dept_Fk Foreign Key (Department_Id)
                 References Departments(Department_Id),
              Constraint Emp_Email_Uk Unique(Email)
               );
/

Create Table Employees(Employee_Id      Number(6) Constraint  Emp_Employee_Id
Primary Key,
                First_Name      Varchar2(20),
                Last_Name       Varchar2(25) Constraint  Emp_Last_Name_Nn  Not Null,
                Email           Varchar2(25) Constraint  Emp_Email_Nn  Not Null
                                Constraint  Emp_Email_Uk  Unique ,
                Phone_Number    Varchar2(20),
                Hire_Date       Date Constraint  Emp_Hire_Date_Nn  Not Null,
                Job_Id          Varchar2(10) Constraint  Emp_Job_Nn  Not Null,
                Salary          Number(8,2) Constraint  Emp_Salary_Ck  Check (Salary>0),
                Commission_Pct   Number(2,2),
                Manager_Id      Number(6),
                Department_Id    Number(4) Constraint  Emp_Dept_Fk  References
Departments (Department_Id)
               );
              /

              Update Employees
                Set Department_Id = 55
               Where Department_Id = 110;
              /

              Delete From Departments
              Where  Department_Id = 60;
              /

        Alter Table Departments
              Add (Column12  Varchar2(20));
              /
        Alter Table Departments
            Modify(Column12  Varchar2(200));
        /
        Alter Table Departments  Drop Column Column12;
        /
        Alter Table Departments Add Constraint Fk_Uk_Check Unique(Deptname);
        /
```

```
-- View

Create View Empvu80
        As Select Employee_Id, Last_Name, Salary
            From Employees
           Where Department_Id = 80;
/
Create View Salvu50
        As Select Employee_Id Id_Number, Last_Name Name, Salary*12
Ann_Salary
            From Employees
           Where Department_Id = 50;
/
Select *
From  Salvu50;
/
Create Or Replace View Empvu80(Id_Number, Name, Sal, Department_Id)
                As Select Employee_Id, First_Name || ' '|| Last_Name, Salary,
Department_Id
                    From Employees
                   Where Department_Id = 80;
/
Create Or Replace View Dept_Sum_Vu(Name, Minsal, Maxsal, Avgsal)
                As Select D.Department_Name,
Min(E.Salary),Max(E.Salary),Avg(E.Salary)
                    From Employees E Join Departments D
                     On (E.Department_Id = D.Department_Id)
                Group By D.Department_Name;
/

-- Sequence

Create Sequence Dept_Deptid_Seq
    Increment By 10
    Start  With  120
    Maxvalue  9999
    Nocache
    Nocycle;
/
Insert Into Departments(Department_Id,Department_Name, Location_Id)
            Values (Dept_Deptid_Seq.Nextval,'Support', 2500);
/
Alter Sequence Dept_Deptid_Seq
  Increment By 20
  Maxvalue  999999
  Nocache
```

```
  Nocycle;
/
Drop Sequence Dept_Deptid_Seq;
/

D.    Index

Create Index Emp_Last_Name_Idx
        On Employees(Last_Name);
/
Drop Index Emp_Last_Name_Idx;
/
Create Synonym D_Sum
For  Dept_Sum_Vu;
/

--CHAPTER-10(Creating Other Schema Objects)

--CHAPTER-11(Managing Objects with Data Dictionary Views)



--**************************************************************--
/*
--  Oracle SQL Fundamental 2 V-1

    --  Chapter-1(Controlling User Access)

     *  Differentiate System Privileges From Object Privileges
     *  Grant Privileges On Tables
     *  View Privileges In The Data Dictionary
     *  Grant Roles
     *  Distinguish Between Privileges And Roles

    1.    Privileges

     *  Database Security:
          - System Security
          - Data Security
     *  System Privileges: Gaining Access To The Database
     *  Object Privileges: Manipulating The Content Of The Database Objects
     *  Schemas: Collection Of Objects Such As Tables, Views And Sequences
     *  More Than 100 Privileges Are Available.
     8  The Database Administrator Has High-Level System Privileges For Tasks Such
As:
          - Creating New Users
```

```
        - Removing Users
        - Removing Tables
        - Backing Up Tables


    The Dba Creates Users With The Create User Statement.
*/
    Create User  User1 Identified By   User1;

    /

        -- Create Session
        -- Create Table
        -- Create Sequence
        -- Create View
        -- Create Procedure

    Grant  Create Session, Create Table, Create Sequence, Create View To  Scott;

    /

    Create Role Manager;

    /

    Grant  Create  Table,  Create  View To Manager;

    /

    Grant Manager To Bell, Kochhar;

    /

    Alter User HR Identified By Employ;

    /

    GRANT SELECT ON Employees TO Sue, Rich;

    /

    GRANT  UPDATE (Department_Name, Location_Id)
    ON  Departments
    TO  Scott, Manager;

    /
```

```
GRANT SELECT, INSERT ON Departments TO Scott WITH GRANT OPTION;

/

GRANT SELECT ON Alice.Departments TO PUBLIC;

/

REVOKE  SELECT, INSERT ON  Departments FROM  Scott;

/
/*
   Chapter-2(Managing Schema Objects)
        1.  Alter Table Statement
        2.  Adding A Column
*/

   ALTER TABLE Dept80 ADD(Job_Id VARCHAR2(9));

   /
      -- Modifying A Column

   ALTER TABLE Dept80 MODIFY  (Last_Name VARCHAR2(30));

   /
      -- Dropping A Column

   ALTER  TABLE    Dept80 DROP COLUMN  Job_Id;

   /
      -- Set Unused Option

   ALTER TABLE  Dept80 SET UNUSED(Last_Name);

   /

   ALTER TABLE Dept80 DROP UNUSED COLUMNS;

   /
     --  Adding A Constraint


   ALTER TABLE Emp2 MODIFY Employee_Id PRIMARY KEY;

      /
```

```
ALTER TABLE Emp2 ADD CONSTRAINT Emp_Mgr_Fk
  FOREIGN KEY(Manager_Id) REFERENCES Emp2(Employee_Id);

 /
-- On Delete Cascade

ALTER TABLE Emp2 ADD CONSTRAINT  Emp_Dt_Fk FOREIGN KEY
(Department_Id)
   REFERENCES Departments ON DELETE CASCADE;

  /
--   Deferring Constraints
--.   Dropping A Constraint

ALTER TABLE Emp2 DROP CONSTRAINT Emp_Mgr_Fk;

/

ALTER TABLE Dept2 DROP PRIMARY KEY CASCADE;

    /

--  Disabling Constraints

ALTER TABLE Emp2 DISABLE CONSTRAINT Emp_Dt_Fk;

  /

--   Enabling Constraints


ALTER TABLE  Emp2 ENABLE CONSTRAINT Emp_Dt_Fk;

/

--   Cascading Constraints


ALTER TABLE Emp2 DROP COLUMN Employee_Id CASCADE CONSTRAINTS;

  /

ALTER TABLE Test1 DROP (Pk, Fk, Col1);

/
```

```
-- Overview Of Indexes


CREATE TABLE Emp_Unnamed_Index
(
  Employee_Id   NUMBER (6) PRIMARY KEY,
  First_Name    VARCHAR2 (20),
  Last_Name     VARCHAR2 (25)
);

 /

SELECT Index_Name, Table_Name
  FROM User_Indexes
 WHERE Table_Name = 'EMP_UNNAMED_INDEX';

/

CREATE TABLE New_Emp2(Employee_Id   NUMBER (6)
                      First_Name    Varchar2(20),
                      Last_Name     Varchar2(25)
                     );

/

CREATE INDEX Emp_Id_Idx2
        ON New_Emp2(Employee_Id);

/

ALTER  TABLE New_Emp2 ADD PRIMARY KEY(Employee_Id)  USING  INDEX
Emp_Id_Idx2;

  /

Drop Index Upper_Dept_Name_Idx;

/
-- Function-Based Indexes
-- Removing An Index
-- Drop Table … Purge


DROP TABLE Dept80  PURGE;
```

```
/
--  Flashback Table Statement

DROP TABLE Emp2;

/

SELECT Original_Name, Operation, Droptime FROM Recyclebin;

   /

FLASHBACK TABLE Emp2 TO BEFORE DROP;

/

--  External Tables
--  Creating A Directory For The External Table
--  Querying External Tables


-- Chapter-3(Manipulating Large Data Sets)
   --  Using Subqueries To Manipulate Data


INSERT INTO sales_reps (id, name,salary,commission_pct)
            SELECT employee_id,last_name,salary,commission_pct
              FROM employees
             WHERE job_id LIKE '%REP%';

      /
--  Copying Rows From Another Table
--  Inserting Using A Subquery As A Target


INSERT INTO (SELECT employee_id,last_name, email,
hire_date,job_id,salary,department_id
               FROM empl3
              WHERE department_id = 50
            )
      VALUES (99999,'Taylor','DTAYLOR',TO_DATE ('07-JUN-99', 'DD-MON-
RR'),'ST_CLERK',5000,50
            );

      /

SELECT employee_id,
```

```
      last_name,
      email,
      hire_date,
      job_id,
      salary,
      department_id
 FROM empl3
WHERE department_id = 50;

/
--   Retrieving Data With A Subquery As Source


SELECT a.last_name,
      a.salary,
      a.department_id,
      b.salavg
 FROM employees a
      JOIN (  SELECT department_id, AVG (salary) salavg
                FROM employees
            GROUP BY department_id) b
        ON a.department_id = b.department_id AND a.salary > b.salavg;

      /
--   Updating Two Columns With A Subquery


UPDATE empl3
  SET job_id =
        (SELECT job_id
           FROM employees
          WHERE employee_id = 205),
      salary =
        (SELECT salary
           FROM employees
          WHERE employee_id = 168)
WHERE employee_id = 114;

/
--   Updating Rows Based On Another Table


UPDATE empl3
  SET department_id =
        (SELECT department_id
           FROM employees
```

```
          WHERE employee_id = 100)
 WHERE job_id = (SELECT job_id
                 FROM employees
                WHERE employee_id = 200);

    /
--    Deleting Rows Based On Another Table


DELETE FROM empl3
     WHERE department_id = (SELECT department_id
                            FROM departments
                           WHERE department_name LIKE '%Public%');

/
--    Using The With Check Option Keyword On Dml
--    Using Explicit Default Values


INSERT INTO deptm3 (department_id, department_name, manager_id)
    VALUES (300, 'Engineering', DEFAULT);

    /

UPDATE deptm3
  SET manager_id = DEFAULT
 WHERE department_id = 10;

    /
   --    Overview Of Multitable Insert Statements
   --    Types Of Multitable Insert Statements
   --    Multitable Insert Statements
   --    Unconditional Insert All
   --    Conditional Insert First


INSERT ALL
WHEN SAL > 10000 THEN
    INTO sal_history
  VALUES (EMPID, HIREDATE, SAL)
WHEN MGR > 200 THEN
    INTO mgr_history
  VALUES (EMPID, MGR, SAL)
  SELECT employee_id EMPID,
        hire_date HIREDATE,
        salary SAL,
```

```
            manager_id MGR
         FROM employees
        WHERE employee_id > 200;

          /

        INSERT FIRST
        WHEN SAL > 25000 THEN
            INTO special_sal
          VALUES (DEPTID, SAL)
        WHEN HIREDATE LIKE ('%00%') THEN
            INTO hiredate_history_00
          VALUES (DEPTID, HIREDATE)
        WHEN HIREDATE LIKE ('%99%') THEN
            INTO hiredate_history_99
          VALUES (DEPTID, HIREDATE)
        ELSE
            INTO hiredate_history
          VALUES (DEPTID, HIREDATE)
            SELECT department_id DEPTID, SUM (salary) SAL, MAX (hire_date)
HIREDATE
              FROM employees
          GROUP BY department_id;

          /

    --  Pivoting Insert
    --  Merge Statement


        MERGE INTO empl3 c
            USING employees e
              ON (c.employee_id = e.employee_id)
        WHEN MATCHED THEN
          UPDATE SET
            c.first_name = e.first_name,
            c.last_name = e.last_name,
            c.department_id = e.department_id
        WHEN NOT MATCHED THEN
          INSERT     VALUES (e.employee_id,
                             e.first_name,
                             e.last_name,
                             e.email,
                             e.phone_number,
                             e.hire_date,
                             e.job_id,
```

```
                e.salary,
                e.commission_pct,
                e.manager_id,
                e.department_id);

    /

MERGE INTO empl3 c
    USING employees e
      ON (c.employee_id = e.employee_id)
WHEN MATCHED THEN
  UPDATE SET c.first_name = e.first_name,
             c.last_name = e.last_name,
             c.email = e.email,
             c.phone_number = e.phone_number,
             c.hire_date = e.hire_date,
             c.job_id = e.job_id,
             c.salary = e.salary,
             c.commission_pct = e.commission_pct,
             c.manager_id = e.manager_id,
             c.department_id = e.department_id
WHEN NOT MATCHED THEN
  INSERT    VALUES (e.employee_id,
                e.first_name,
                e.last_name,
                e.email,
                e.phone_number,
                e.hire_date,
                e.job_id,
                e.salary,
                e.commission_pct,
                e.manager_id,
                e.department_id);

    /

--  Example Of The Flashback Version Query
--  Versions Between Clause



Chapter-4(Generating Reports by Grouping Related Data)
 --  Review of Group Functions


  SELECT AVG (salary),
```

```
        STDDEV (salary),
        COUNT (commission_pct),
        MAX (hire_date)
    FROM employees
   WHERE job_id LIKE 'SA%';

   /
--   Review of the GROUP BY Clause

 SELECT department_id,
       job_id,
       SUM (salary),
       COUNT (employee_id)
  FROM employees
GROUP BY department_id, job_id;

/
--   Review of the HAVING Clause
--   GROUP BY with ROLLUP and CUBE Operators
--   ROLLUP Operator


 SELECT department_id, job_id, SUM (salary)
  FROM employees
 WHERE department_id < 60
GROUP BY ROLLUP (department_id, job_id);

 /
--   CUBE Operator


 SELECT department_id, job_id, SUM (salary)
  FROM employees
 WHERE department_id < 60
GROUP BY CUBE (department_id, job_id);


/
--   GROUPING Function


 SELECT department_id DEPTID,
       job_id JOB,
       SUM (salary),
       GROUPING (department_id) GRP_DEPT,
       GROUPING (job_id) GRP_JOB
```

```
   FROM employees
  WHERE department_id < 50
GROUP BY ROLLUP (department_id, job_id);

 /
--  GROUPING SETS


 SELECT department_id,
       job_id,
       manager_id,
       AVG (salary)
   FROM employees
GROUP BY GROUPING SETS ( (department_id, job_id), (job_id, manager_id));

  /

 SELECT department_id,
       job_id,
       NULL AS manager_id,
       AVG (salary) AS AVGSAL
   FROM employees
GROUP BY department_id, job_id
UNION ALL
 SELECT NULL,
       job_id,
       manager_id,
       AVG (salary) AS AVGSAL
   FROM employees
GROUP BY job_id, manager_id;

 /
--  Composite Columns


 SELECT department_id,
       job_id,
       manager_id,
       SUM (salary)
   FROM employees
GROUP BY ROLLUP (department_id, (job_id, manager_id));

/
--    Concatenated Groupings
```

```
 SELECT department_id,
        job_id,
        manager_id,
        SUM (salary)
   FROM employees
GROUP BY department_id, ROLLUP (job_id), CUBE (manager_id);

/


/*

Chapter-5(Managing Data in Different Time Zones)
1.  Time Zones
2.  TIME_ZONE Session Parameter
3.  CURRENT_DATE, CURRENT_TIMESTAMP, and LOCALTIMESTAMP
4.  CURRENT_DATE
5.  CURRENT_TIMESTAMP
6.  LOCALTIMESTAMP
7.  DBTIMEZONE and SESSIONTIMEZONE
8.  TIMESTAMP Data Type
9.  TIMESTAMP Fields
10. Difference Between DATE and TIMESTAMP
11. TIMESTAMP WITH LOCAL TIMEZONE
12. INTERVAL Data Types
13. INTERVAL Fields
14. INTERVAL YEAR TO MONTH Data Type
15. INTERVAL DAY TO SECOND Data Type
16. EXTRACT
17. TIMESTAMP Conversion Using FROM_TZ
18. Time Interval Conversion with TO_YMINTERVAL
19. Daylight Saving Time

Chapter-6(Retrieving Data Using Subqueries)
1.  Multiple-Column Subqueries

*/
SELECT first_name,
       last_name,
       manager_id,
       department_id
  FROM employees
 WHERE manager_id IN (SELECT manager_id
                        FROM employees
                       WHERE first_name = 'Daniel'
                      )
```

```
  AND department_id IN (SELECT department_id
                        FROM employees
                        WHERE first_name = 'Daniel'
                       );

  /
--   Pairwise Comparison Subquery


SELECT employee_id, manager_id, department_id
 FROM employees
 WHERE (manager_id, department_id) IN (SELECT manager_id, department_id
                                       FROM employees
                                       WHERE first_name = 'John'
                                      )
  AND first_name <> 'John';

  /

--   Nonpairwise Comparison Subquery


SELECT employee_id, manager_id, department_id
 FROM employees
 WHERE manager_id IN (SELECT manager_id
                      FROM employees
                      WHERE first_name = 'John'
                     )
  AND department_id IN (SELECT department_id
                        FROM employees
                        WHERE first_name = 'John'
                       )
  AND first_name <> 'John';

 /
--   Scalar Subquery Expressions


SELECT employee_id,
      last_name,
      (CASE
         WHEN department_id = (SELECT department_id
                               FROM departments
                               WHERE location_id = 1800) THEN
           'Canada'
         ELSE
```

```
            'USA'
        END)
         location
  FROM employees;

 /

 SELECT employee_id, last_name
   FROM employees e
ORDER BY (SELECT department_name
          FROM departments d
          WHERE e.department_id = d.department_id);

/

--   Correlated Subqueries


SELECT last_name, salary, department_id
 FROM employees outer
 WHERE salary > (SELECT AVG (salary)
                 FROM employees
                 WHERE department_id = outer.department_id);

 /

SELECT e.employee_id, last_name, e.job_id
 FROM employees e
 WHERE 2 <= (SELECT COUNT (*)
             FROM job_history
             WHERE employee_id = e.employee_id);

  /

--   Using the EXISTS Operator


SELECT employee_id,
      last_name,
      job_id,
      department_id
 FROM employees outer
 WHERE EXISTS
        (SELECT 'X'
         FROM employees
         WHERE manager_id = outer.employee_id);
```

```
  /

SELECT department_id, department_name
  FROM departments d
 WHERE NOT EXISTS
        (SELECT 'X'
          FROM employees
         WHERE department_id = d.department_id);

 /


--    Correlated UPDATE


ALTER TABLE empl6
       ADD(department_name VARCHAR2(25));


/


UPDATE empl6 e
   SET department_name =
        (SELECT department_name
          FROM departments d
         WHERE e.department_id = d.department_id);

/


UPDATE empl6
   SET salary =
        (SELECT empl6.salary + rewards.pay_raise
          FROM rewards
         WHERE     employee_id = empl6.employee_id
             AND payraise_date =
                  (SELECT MAX (payraise_date)
                    FROM rewards
                   WHERE employee_id = empl6.employee_id))
 WHERE empl6.employee_id IN (SELECT employee_id FROM rewards);

/
```

```
--    Correlated DELETE


DELETE FROM emp_history JH
     WHERE employee_id = (SELECT employee_id
                          FROM employees E
                         WHERE JH.employee_id = E.employee_id
                          AND START_DATE = (SELECT MIN (start_date)
                                            FROM job_history JH
                                           WHERE JH.employee_id = E.employee_id
                                          )
                          AND 5 > (SELECT COUNT (*)
                                   FROM job_history JH
                                  WHERE JH.employee_id = E.employee_id
                                GROUP BY EMPLOYEE_ID
                                 HAVING COUNT (*) >= 4
                                )
                    );
     /

   DELETE FROM empl6 E
        WHERE employee_id = (SELECT employee_id
                             FROM emp_history
                            WHERE employee_id = E.employee_id
                           );

   /

--    WITH Clause


--    Chapter-7(Hierarchical Retrieval)


   SELECT employee_id,
          last_name,
          job_id,
          manager_id
     FROM employees
START WITH employee_id = 101
CONNECT BY PRIOR manager_id = employee_id;

/

   SELECT last_name || '  reports  to  ' || PRIOR last_name "Walk Top Down"
     FROM employees
```

```
START WITH last_name = 'King'
CONNECT BY PRIOR employee_id = manager_id;

/

-- Natural Tree Structure
-- Hierarchical Queries
-- Pruning Branches


/*
Chapter-8(Regular Expression Support)
   Regular Expression: Overview
   Meta Characters
   Regular Expression Functions
   Performing Basic Searches
   Checking the Presence of a Pattern
   Example of Extracting Substrings
   Replacing Patterns
*/
```